Publikationsreihe des John von Neumann-Instituts für Computing (NIC)
NIC-Serie Band 30

John von Neumann-Institut für Computing (NIC)

Achim Kless und Johannes Grotendorst (Hrsg.)

GALA
Grünenthal Applied Life Science Analysis

Bericht

NIC-Serie Band 30 ISBN 3-00-017349-8

Zentralinstitut für Angewandte Mathematik

Die Deutsche Bibliothek - CIP-Einheitsaufnahme
Ein Titeldatensatz für diese Publikation ist bei
Der Deutschen Bibliothek erhältlich.

Herausgeber: NIC-Direktorium
Vertrieb: NIC-Sekretariat
Forschungszentrum Jülich
52425 Jülich
Deutschland
Internet: www.fz-juelich.de/nic
Druck: Graphische Betriebe, Forschungszentrum Jülich

© 2006 John von Neumann-Institut für Computing
Es ist erlaubt, dieses Werk oder Teile davon digital oder auf Papier zum persönlichen Gebrauch oder zu Lehrzwecken zu vervielfältigen, vorausgesetzt die Kopien werden nicht kommerziell genutzt. Kopien müssen diese Copyright-Notiz und das volle Zitat auf ihrer Titelseite enthalten. Andere Vervielfältigung bedarf der vorherigen schriftlichen Genehmigung des oben genannten Herausgebers.

NIC-Serie Band 30 ISBN 3-00-017349-8

Vorwort

Dieser Forschungsbericht enthält eine zusammenfassende Darstellung der wissenschaftlichen Arbeiten und verdeutlicht anhand ausgewählter Beispiele die erzielten Ergebnisse des Data Mining Projektes GALA (Grünenthal Applied Life Science Analysis), einer Industriekooperation zwischen dem Aachener Pharmaunternehmen Grünenthal und dem Forschungszentrum Jülich. Das GALA-Projekt wurde ins Leben gerufen, um neue Wege in der Entwicklung von Medikamenten zu beschreiten. Die Arbeitsschwerpunkte wurden auf einem ersten gemeinsamen Meeting, wenige Tage vor dem Millennium-Jahreswechsel am 21. Dezember 1999 diskutiert und vereinbart. Projektziel war die Erstellung von Computerprogrammen, die die automatische Analyse und Auswertung der sich ständig vergrößernden Datenmengen in der Pharmaforschung bei Grünenthal ermöglichen. Im Prozess der Medikamentenentwicklung müssen aus einer sehr großen Anzahl von chemischen Substanzen diejenigen ausfindig gemacht werden, die im menschlichen Organismus eine bestimmte, gewünschte Wirkung (wie z.B. die Linderung eines Schmerzreizes) erzielen. Der genaue Zusammenhang zwischen der chemischen Struktur einer Substanz und den daraus resultierenden biologischen Eigenschaften ist jedoch meistens weitgehend unbekannt. Eine wichtige Aufgabe besteht nun darin, diese unbekannten Zusammenhänge, d.h. verborgenen Informationen, in vorhandenen Datenbeständen mit Hilfe geeigneter Verfahren des Data Mining aufzudecken.

Die gemeinsamen Arbeiten starteten am 1. Juli 2000 und erstreckten sich zunächst über einen Zeitraum von drei Jahren. Wissenschaftler der Firma Grünenthal und des Forschungszentrums Jülich entwickelten neue Methoden und Verfahren für die Datenanalyse in der pharmazeutischen Forschung. Es wurden Computerprogramme zur Datenvorauswertung und Klassifikation erstellt. Wichtige Aufgaben der Datenvorauswertung sind die Reduktion des Datenraumes durch die Erkennung von signifikanten Abhängigkeiten zwischen den untersuchten Merkmalen (Merkmalsauswahl) sowie die Erkennung von fehlerhaften bzw. auffälligen Datenpunkten (Ausreißereliminierung). Mit Hilfe des Klassifikationsprogramms lassen sich neue chemische Substanzen automatisch in Gruppen (Klassen) mit unterschiedlichen Erfolgsaussichten einteilen.

Im Mittelpunkt der zweiten Kooperationsphase vom 1. Juli 2003 – 30. Juni 2006 stand die Erstellung neuer, leistungsfähigerer Klassifikatoren für große Datensätze in der Pharmaforschung. Es wurden insbesondere die so genannten Support-Vektor-Maschinen (SVMs) untersucht und implementiert. Es handelt sich hierbei um moderne (überwachte) Algorithmen aus dem Forschungsgebiet des Maschinellen Lernens, einem Teilgebiet der künstlichen Intelligenz. SVMs können sowohl zur Erstellung von nichtlinearen Klassifikatoren, als auch zur Vorhersage von exakten Einzelwerten (Regression) verwendet werden. Die fundierten theoretischen Grundlagen sowie die guten Generalisierungseigenschaften führten dazu, dass Support-Vektor-Maschinen mittlerweile zu den beliebtesten modernen Lernmethoden zählen.

Der Bericht ist wie folgt gegliedert: Teil I gibt einen Überblick über die Ziele des Projektes und dokumentiert alle Publikationen und Vorträge, die im Zusammenhang mit dem GALA-Projekt entstanden sind. In Teil II werden die statistischen Verfahren zur Datenvorauswertung, in Teil III die Klassifikationsalgorithmen für Daten aus der Pharmaindustrie beschrieben. In Teil IV sind die Möglichkeiten der Datenanalyse auf der Basis der verwendeten Data Mining Pipeline erläutert. In Teil V schließlich wird am Beispiel 'Cytochrom P450 Profilierung von neuen Wirkstoffen' die Anwendung der neuen Software demonstriert.

An dieser Stelle möchten wir allen Mitarbeitern des Projektes für ihr Engagement und ihre fachlichen Beiträge herzlich danken. Das Thema des GALA-Projektes hatte einen ausgesprochen interdisziplinären Charakter und überdeckte Teilgebiete der Mathematik, Statistik, Informatik, Chemie und Pharmazie. Der Projekterfolg ist nicht zuletzt den jungen Projektmitarbeitern (Diplomanden des Studiengangs Technomathematik und Doktoranden) zu verdanken, die sich mit Enthusiasmus und Freude in das innovative Gebiet des Data Mining eingearbeitet haben. Für die stets gute Zusammenarbeit bei der administrativen Durchführung des GALA-Projektes danken wir der Forschungsleitung der Grünenthal GmbH und dem Technologie-Transfer-Büro des Forschungszentrums Jülich. Unser besonderer Dank geht an die Autoren des Berichtes, die diese ausführliche Projektdokumentation erst möglich gemacht haben.

Aachen und Jülich
November 2006

Achim Kless, Johannes Grotendorst

Inhaltsverzeichnis

Überblick

Achim Kless
Biomedizinische Forschung, Drug Discovery
Grünenthal GmbH
52078 Aachen
E-mail: Achim.Kless@grunenthal.com

1 Projektziele

Ausgangspunkt für das GALA-Projekt ist eine der Kernaufgaben der präklinischen Forschung, die auf den einzelnen Stufen der Forschung anfallenden Daten effizient zu interpretieren, um aus einem Pool von mehreren Millionen Substanzen diejenigen herauszufinden, die für eine humane Anwendung geeignet sind. Dafür müssen chemische Substanzen zahlreiche Hürden überwinden, die durch vier Bereiche repräsentiert werden. Man unterscheidet die Bereiche in silico, in vitro, in vivo und Klinik. Der in silico Bereich umfasst den chemischen Raum, d.h. alle chemisch möglichen bzw. zugänglichen Substanzen. Das können bereits bekannte chemische Wirkstoffe, neu hergestellte Substanzklassen oder einfach so genannte virtuelle Substanzen sein, die zum Zeitpunkt der Betrachtung nur als Computermodelle vorliegen. In der nächsten Stufe wird ein Teil der Substanzen in einem in vitro Modell getestet. Das in vitro Modell stellt ein molekulares Target (Schloss) z.B. in einer Zellmembran dar, für das eine Substanz (Schlüssel) gesucht wird, die eine Interaktion nach dem Schlüssel-Schloss Prinzip zeigt. Im nächsten Schritt der Lead-Optimierung werden die aktiven Hitsubstanzen solange optimiert, bis eine hohe Affinität und Selektivität zum molekularen Target gefunden wird. Dies ist notwendig, um eine möglichst kleine Anzahl von Substanzen in der nachfolgenden in vivo Testphase einzusetzen, in der das pharmakologische Wirkprinzip überprüft wird. Die so gefundenen Substanzen müssen dann noch geeignete ADMET-Eigenschaften (Absorption, Distribution, Metabolization, Excretion, Toxicity) besitzen, was eine Grundvoraussetzung für eine humane Anwendung und die Zulassung durch die Behörden der FDA (Food and Drug Administration) oder BfArm (Bundesinstitut für Arzneimittel und Medizinprodukte) ist. Aus jeder Stufe der

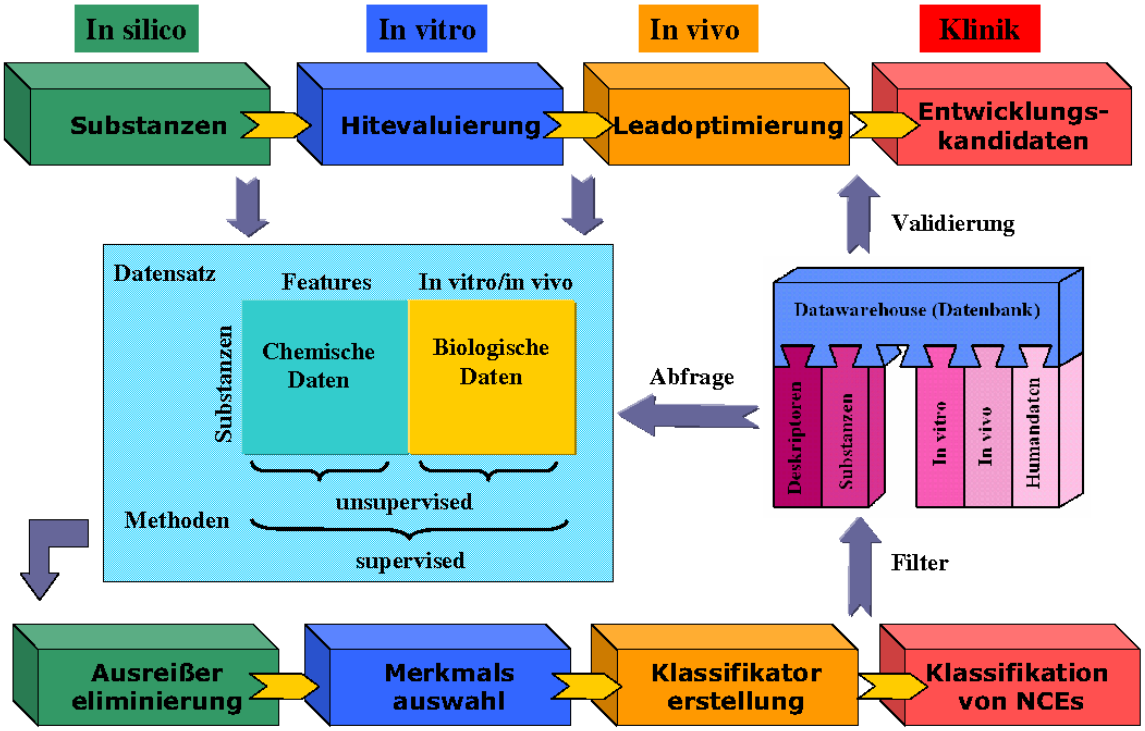


Abbildung 1: Zusammenhänge zwischen F&E-Pipeline und Data Mining.

Entwicklung einer Substanz leiten sich grundsätzliche Fragestellungen ab, die immer wieder beantwortet werden müssen. Das Projektziel ist deshalb die effektive Unterstützung der Prozesse innerhalb der präklinischen Forschung, um die Entwicklungszeiten von 10-15 Jahren für innovative Arzneimittel signifikant zu verkürzen.

2 Fragestellungen und Generierung der Datensätze

Wie in Abbildung 1 dargestellt, müssen auf allen Entwicklungsstufen Entscheidungsprozesse zur Selektion von Substanzen herangezogen werden, da aus Kapazitäts- sowie Zeitgründen nicht alle Substanzen bzw. der gesamte chemische Raum betrachtet werden können. Da die Interpretation von chemischen Strukturen durch den Computer nur indirekt möglich ist, werden diese im ersten Schritt zunächst geeignet transformiert (Abbildung 2). Dazu dienen Verfahren aus der Chemoinformatik. Die Atomkoordinaten der chemischen Strukturen können nicht direkt als Eigenschaften herangezogen werden, weil die Anzahl der Atome von Struktur zu Struktur unterschiedlich ist. Die Anwendung von Lernalgorithmen erzwingt als Input Eigenschaftsvektoren, die sogenannten Feature-Vektoren,

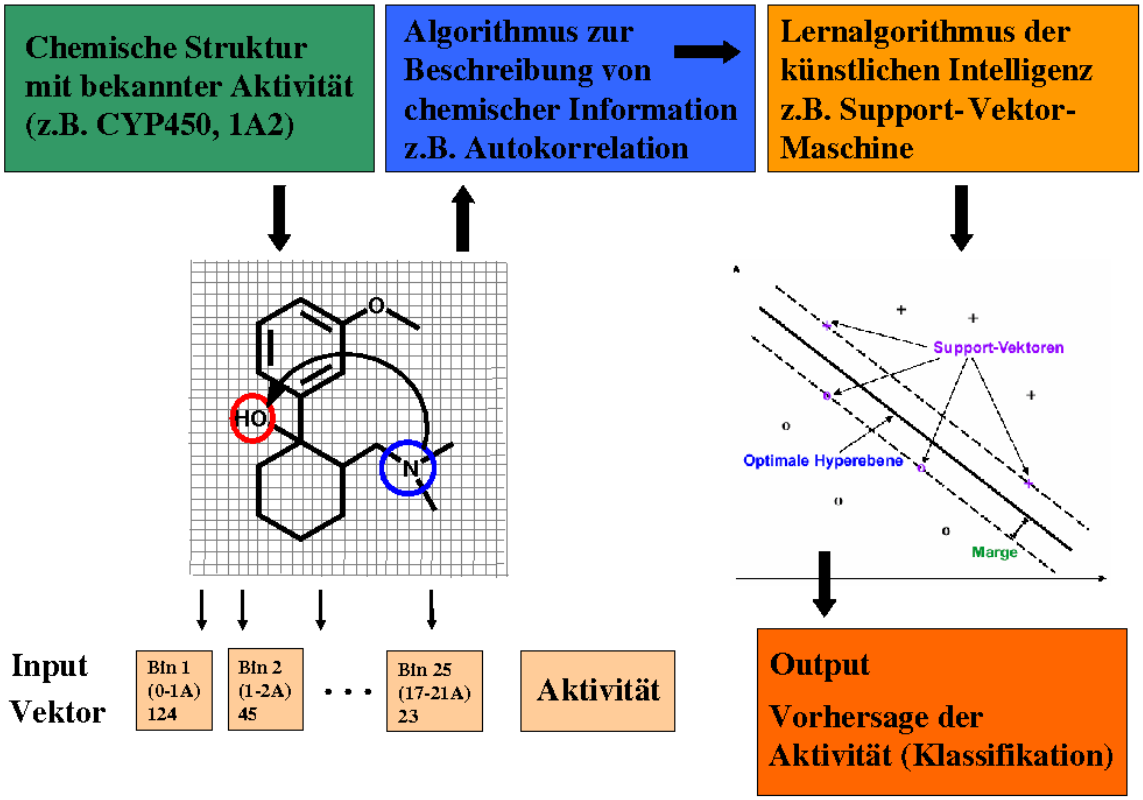


Abbildung 2: Chemische Informationen und Klassifikation.

die immer die gleiche Anzahl an Merkmalen enthalten und eine von der Größe der Moleküle unabhängige, normalisierte Form haben. Die berechneten Eigenschaften müssen auch unabhängig von der Ausrichtung der Moleküle im Raum sein. Eine bekannte Größe ist in diesem Zusammenhang die so genannte Autokorrelation. Für verschiedene Punkte der Moleküloberfläche oder des Raumes, der das Molekül umgibt, wird eine Moleküleigenschaft, wie z.B. die TPSA (Topological Polar Surface Area), berechnet. Aus den Wertepaaren der berechneten Moleküleigenschaft der Punktepaare, deren Abstände in einem vorgegebenen Intervall liegen (in Abbildung 2 werden diese Intervalle mit Bin 1, ..., Bin 25 bezeichnet), wird eine Korrelation berechnet: die Autokorrelation der Moleküleigenschaft für das betrachtete Intervall (Bin) [1, 2]. Durch die Anzahl der Bins werden der Grad der Auflösung und die Länge des Vektors festgelegt. Wählt man als Punkte bestimmte Atome des Moleküls und ordnet den Atomen als ‘Eigenschaft’ den konstanten Wert 1 zu, so ergibt die formale Berechnung der Autokorrelation die Häufigkeitsverteilung der Atomabstände . Der Zusammenhang von zwei Moleküleigenschaften kann entsprechend durch die Kreuzkorrelation der Merkmale beschrieben werden. Wählt man hierbei als Punkte die Atome und als Eigenschaften die Indikatorfunktionen (mögliche Werte: 0,1) zweier Atomtypen

(z.B. Sauerstoff und Stickstoff), dann ergibt die formale Berechnung der Kreuzkorrelation die Häufigkeitsverteilung der Abstände der betrachteten Atomtypen [3]. Eigenschaftsvektoren können auch aus einzelnen, voneinander unabhängigen, molekularen Deskriptoren aufgebaut werden. Dies können beispielsweise physikochemische Eigenschaften wie Molekulargewicht, Löslichkeit oder Dipolmoment sein. Wieder andere fassen die Anzahl von funktionellen Gruppen oder größeren molekularen Untereinheiten (Fragmenten) und Atomkonnektivitäten in binären Strings als sogenannte Fingerprints zusammen. Der Vorgang wird für alle betrachteten Substanzen und idealerweise für einen möglichst großen chemischen Raum durchgeführt. Da die so berechnete Gesamtdatenmenge sehr groß ist und zu jeder chemischen Substanz nur einmal ein Feature-Vektor aufgebaut werden muss, werden die Daten in einer Datenbank mit optimierter Datenstruktur, dem sogenannten Data Warehouse, abgelegt. Damit kann für jede Fragestellung flexibel ein Datensatz erstellt werden.

Im zweiten Schritt werden dann die zu den einzelnen Substanzen erhobenen targetspezifischen (in vitro) bzw. spezifisch für das Indikationsgebiet (in vivo) notwendigen biologischen Daten hinzugefügt. Dies kann ebenfalls durch eine Abfrage im Data Warehouse realisiert werden. Jede einzelne chemische Struktur wird dann durch einen Vektor repräsentiert, der die Struktureigenschaften (Features) mit den in vitro und/oder in vivo Daten verknüpft. Wird dies für jede Struktur wiederholt, erhält man eine Tabelle, die aus einigen hundert Zeilen, eine Zeile pro chemischer Substanz, mehreren hundert Spalten für die zugehörigen Features und wenigen Spalten für die biologischen Eigenschaften besteht. Bei den Datensätzen handelt es sich vereinfacht um eine Matrix, bei der pro Zeile eine chemische Substanz mit allen Merkmalen aus dem Data Warehouse repräsentiert wird. Die so erhaltene Darstellung stellt eine Beziehung zwischen den Substanzeigenschaften und einem gewünschten pharmakologischen Profil für eine Zielindikation her. Durch diesen Integrationsschritt ist es auch möglich, mit Hilfe von Algorithmen eine Automatisierung der Analyse durchzuführen. Diese kann durch zwei fundamental unterschiedliche Data Mining Verfahren erreicht werden. Man unterscheidet so genannte unüberwachte (unsupervised) Algorithmen, bei denen die Daten in kleinere und bezüglich der Deskriptoren homogene Gruppen unterteilt werden, sowie überwachte (supervised) Methoden, welche Klassifikatoren erzeugen, mit deren Hilfe es möglich ist, neue chemische Substanzen mit noch unbekannten Feature-Kombinationen bzgl. der gesuchten Eigenschaft zu profilieren. Die Klassifikatoren können dann nach erfolgreicher Evaluierung als Filter verwendet werden.

3 Datenvorauswertung

Ausgangspunkt sind die aus dem Data Warehouse extrahierten Datensätze, die noch von redundanten Informationen befreit werden müssen. Oberste Prämisse ist die Reduktion der Daten auf der Basis einer rein abstrakten Sichtweise, bei der kein Vorwissen über die

zu analysierenden Datensätze eingebracht wird. Der Prozess der Vorauswertung der Datensätze verfolgt mehrere Ziele. Eine wichtige Aufgabe ist die Reduktion der Dimensionalität des Datenraumes durch die Erkennung von signifikanten Abhängigkeiten zwischen den untersuchten Merkmalen. Da bis zu tausend und mehr Merkmale vorliegen, können hunderttausende von Merkmalspaaren generiert werden, deren Untersuchung manuell zu aufwändig ist. Mit Hilfe einer weiteren Methode, der so genannten Variablenselektion, erfolgt die Eingrenzung auf signifikante Merkmale. Ein weiterer Schritt ist die Erkennung und Priorisierung von auffälligen Datenpunkten, den so genannten Ausreißern. Damit ergibt sich insgesamt eine im Vergleich zum Ausgangsdatensatz reduzierte Matrix, die im Optimalfall dieselbe Information beinhaltet.

3.1 Merkmalsauswahl

Ziel ist die Verringerung der in den Daten enthaltenen Redundanz, die sich zu einem großen Teil in den Korrelationen zwischen den Merkmalen widerspiegelt. Einfache statistische Verfahren zur Aufdeckung linearer Abhängigkeiten sind oft nicht ausreichend, um komplexe Daten zu analysieren. In den konkreten Anwendungsfällen ist es häufig notwendig, nichtlineare Abhängigkeiten zu berücksichtigen. Die üblicherweise verwendete Produkt-Moment-Korrelation ist ein Maß für den linearen Zusammenhang zweier Merkmale und im Allgemeinen nicht geeignet, um ausgeprägt nichtlineare Abhängigkeiten zu charakterisieren. In einigen Fällen sollte die Berechnung der Produkt-Moment-Korrelation daher mit einer Datentransformation, meist eine Logarithmierung, kombiniert werden. Alternativ kann die Rangkorrelation verwendet werden, die invariant gegenüber monotonen Transformationen und somit ein Maß für monotone nichtlineare Zusammenhänge ist. Die Hauptkomponentenanalyse (PCA) ist ein Verfahren zur Dimensionsreduktion, das auf der Eigenwertzerlegung der Korrelationsmatrix basiert. Die Daten werden dabei durch eine kleine Anzahl von unkorrelierten Linearkombinationen, den so genannten Hauptkomponenten, dargestellt, die einen wesentlichen Anteil der in den Daten enthaltenen Streuung repräsentieren. Der Informationsverlust kann dadurch minimal gehalten werden. Man erreicht hierdurch jedoch meist nur, dass die Gesamtheit der Beobachtungspunkte durch den Raum mit reduzierter Dimensionalität gut beschrieben wird. Dass jedoch dieser Satz von Hauptkomponenten dazu geeignet ist, das Problem der Klassenzugehörigkeit zu beschreiben, ist nicht garantiert. Neben dieser Hauptkomponentenanalyse werden Verfahren auf der Grundlage einer Merkmalsauswahl unter Berücksichtigung der durchzuführenden Klassifikation eingesetzt. Es werden diejenigen Merkmale ausgewählt, deren Verteilungen in den Klassen des Trainingsdatensatzes die größten Unterschiede aufweisen. Dazu kann beispielsweise ein Chi-Quadrat-Test verwendet werden, bei dem die Aussagekraft jeder einzelnen Dimension für sich allein bezüglich der Klassenunterscheidung getestet wird. Besteht dann zwischen einigen der selektierten Merkmale eine hohe Korrelation, werden die redundanten Merkmale zusätzlich aus der Datenmatrix entfernt und somit nicht mehr bei der Erstellung der Klassifikatoren berücksichtigt. Die Erkennung der bezüglich der

gestellten Aufgabe entscheidenden Merkmale ist auch für die Diskussion der Klassifikationsergebnisse und die spätere Optimierung der chemischen Substanzen von Bedeutung. Ein häufiges Problem ist die anschließende Interpretation der nicht selten abstrakten physikochemischen Deskriptoren. Durch eine Suche der entsprechenden Deskriptoren in einem großen Pool virtueller Substanzen im Data Warehouse wird versucht, diese anhand der gefundenen Strukturbeispiele plausibel zu erklären.

3.2 Ausreißereliminierung

Eine weitere Reduktion des ursprünglichen Datensatzes ergibt sich mittels Ausreißereliminierung. In diesem Fall werden nicht wie bei der Merkmalsauswahl die Anzahl der Spalten, sondern durch die Eliminierung von Trainingsbeispielen die Anzahl der Zeilen reduziert. Das können Mess- oder Eingabefehler sein, so dass die Ausreißerdetektion auch dazu dient, Inkonsistenzen im Data Warehouse zu erkennen. Die Filterung geeigneter chemischer Strukturen ist generell eng mit dem Problem der Ausreißererkennung verknüpft, da sich häufig interessante Substanzen signifikant in wenigen Merkmalen unterscheiden. Einige der Ausreißer können aus dem Datensatz entfernt werden, was gleichzeitig durch die abnehmende Anzahl an Datenpunkten die Erstellung der Klassifikatoren beschleunigt. Im Mittelpunkt des Interesses stehen dabei nicht die Beobachtungen, die durch extreme Werte einzelner Merkmale auffallen und die leicht aufzudecken sind, sondern diejenigen, die von der multivariaten Struktur der Daten abweichen und hierdurch die Analysen stark beeinflussen können. Die Ausreißeridentifizierung stellt keinen isolierten Arbeitsschritt dar, sondern ist Bestandteil der Datenaufbereitung, dem so genannten Data Cleaning. Die endgültige Analyse wird dann für den reduzierten Datensatz durchgeführt. Zur Entscheidung, ob ein Datenpunkt als Ausreißer zu bewerten ist, ist stets auch eine Aussage zur Signifikanz (Verlässlichkeit) des Ergebnisses notwendig. Die dafür notwendige Hypothesenverteilung kann durch Expertenwissen eingebracht werden und erfordert an dieser Stelle eine interdisziplinäre Diskussion. Alternativ zur Ausreißereliminierung können robuste statistische Verfahren verwendet werden, die den Einfluß einzelner fehlerbehafteter Beobachtungen begrenzen.

4 Lernalgorithmen

Nach erfolgter Reduktion des Datensatzes auf wesentliche Merkmale und chemische Beispiele erfolgt im nächsten Schritt die Erstellung der Klassifikatoren aus diesem Trainingsdatensatz. Ein Klassifikator stellt einen Filter dar, der auf Basis der Feature-Vektoren unbekannte Substanzen in Gruppen (Klassen) einteilt. Dafür werden nichtlineare mathematische Verfahren und wahrscheinlichkeitstheoretische Betrachtungen herangezogen. Der Klassifikator wird dann auf alle Substanzen in der Datenbank angewendet. Die erhaltenen

Gruppen werden anschließend wieder in der Datenbank abgelegt und können so für weitere Analysen herangezogen werden.

4.1 Clusterung (unüberwachte Verfahren)

In den hochdimensionalen Räumen, in denen sich die Daten der pharmazeutischen Forschung befinden, ist man oft an der Bildung geeigneter Cluster interessiert. Typischerweise liegen Beoachtungen mit drei- bis vierstelliger Anzahl an Dimensionen vor. Dabei können einige Raumdimensionen diskret, andere wiederum kontinuierlich sein. Allein die Frage der Normierung der verschiedenen Raumdimensionen zeigt, dass die Metrik des Raumes a priori keineswegs eindeutig festgelegt ist. Teilt man jede der Koordinaten dieses Raumes in nur zwei Abschnitte, so entstehen durch diese Unterteilung schon bei einem hundertdimensionalen Raum 2^{100} Zellen, für die jeweils entschieden werden muss, ob sie Teil eines Clusters sind. Um den Raum durch Beobachtungen abzudecken, braucht man pro Zelle mindestens eine Beobachtung. Da die Zugehörigkeit zu einem Cluster einem Bit Information entspricht, benötigt man einen Datenspeicher mit riesiger Kapazität. Tatsächlich liegen in der Regel in einem Standardtrainingsdatensatz maximal einige tausend Feature-Vektoren vor, die nicht ausreichen, um einen hundertdimensionalen Raum auszufüllen. Dies hat zur Konsequenz, dass die Information in einem hochdimensionalen Datenraum nur sehr dünn besetzt ist.

Im Rahmen des Projektes werden Cluster-Methoden für unterschiedliche Fragestellungen verwendet, z.B. zur Überprüfung einer gegebenen Klassifikation auf einem Trainigsdatensatz oder zur Verkleinerung von einzelnen stark aufgeblähten Klassen, um robuste Klassifikatoren erstellen zu können.

4.2 Erstellung von Klassifikatoren (überwachte Verfahren)

In den letzten Jahren haben sich für komplexe Aufgabenstellungen die nichtlinearen Algorithmen, wie beispielsweise Entscheidungsbäume und neuronale Netze, insbesondere aber auch die sogenannten Support-Vektor-Maschinen (SVMs) als geeignet erwiesen. Es handelt sich dabei um moderne Algorithmen aus dem Forschungsgebiet der künstlichen Intelligenz, die sowohl zur Generierung von nichtlinearen Klassifikatoren für beliebig viele Gruppen, als auch zur Vorhersage von exakten Einzelwerten (Regression) verwendet werden können. Zur Optimierung der Modellparameter muss festgelegt werden, welche Kriterien herangezogen werden, um Fehler zu bewerten, die während der Trainingsphase auftreten. Dieser Aspekt ist besonders kritisch, da intuitiv keine Fehlklassifikationen auftreten sollen, damit aber häufig Klassifikatoren erzeugt werden, die zu stark an die verfügbaren Daten angepasst sind (Overfitting) und im Extremfall zu Fehlerquoten von bis zu 100 Prozent auf unbekannten Daten führen können. Eine besondere Stärke des Verfahrens ist, dass große Feature-Vektoren verwendet werden können und auf redundante

Merkmale weniger sensitiv reagiert wird, als dies bei anderen distanzbasierten Verfahren der Fall ist. Die Möglichkeit, nichtlineare Abhängigkeiten zu modellieren, ist ausschlaggebend für die Güte einer Klassifikation auf komplexen Daten. Das Erlernen ausschließlich linearer Funktionen, auch auf reduzierten Datensätzen, führt im Allgemeinen zu schlechteren Ergebnissen. Im Gesamtergebnis klassifiziert eine nichtlineare Funktion vorhandene Datenpunkte genauer und erfasst die zugrundeliegende Klasseneinteilung besser.

Support-Vektor-Maschinen erlernen eine nichtlineare Klassifikationsfunktion, indem sie alle Datenpunkte geeignet transformieren und dann eine einfache lineare Funktion für die Klassentrennung verwenden. Dabei ist es besonders günstig, dass diese Transformation nicht in einem zusätzlichen Schritt vom Nutzer durchgeführt werden muss, sondern für alle nötigen Berechnungen ein so genannter Kern verwendet wird. Ein Kern ist eine Funktion zur Berechnung von Skalarprodukten im Raum der transformierten Daten, welche jedoch die ursprünglichen Daten verwendet und die Transformation damit in impliziter Form selbst durchführt. Dieses Verfahren führt zu schnellen und sicheren Prognosen auf unbekannten Daten. In der Praxis wählt man meist einen Standardkern, beispielsweise den Gauß-Kern. Damit ähneln Support-Vektor-Maschinen stark den in der Literatur häufig beschriebenen neuronalen Netzen, bei denen die Daten selbst nicht transformiert, jedoch die Gewichte der linearen Klassifikationsfunktion nichtlinear angepasst werden. Im Gegensatz zu rein linearen Klassifikatoren können SVMs mit sehr vielen Merkmalen arbeiten. Wendet man dann den Klassifikator auf einen Testdatensatz mit noch unbekannten, neuen Substanzen – den sogenannten New Chemical Entities (NCEs) – an, können diese aufgrund der erhaltenen Vorhersage bewertet werden.

Ein großer Vorteil der in den folgenden Kapiteln beschriebenen Algorithmen gegenüber klassischen Verfahren wie zum Beispiel der Substruktursuche (Ähnlichkeit) ist die Möglichkeit, innovative Substanzen zu identifizieren, die nicht bereits bekannten Substanzen ähnlich sind (Me-Too-Drugs). Somit stellen die in diesem Band dargestellten Vorgehensweisen und Verfahren ein innovatives Potential zur Entdeckung neuartiger Wirkstoffe dar.

5 Dokumentation

5.1 Publikationen

- Thorsten Dickhaus
  Statistische Verfahren für das Data Mining in einem Industrieprojekt
  Interner Bericht FZJ-ZAM-IB-2003-08, Forschungszentrum Jülich, Mai 2003
- Tatjana Eitrich
  Support-Vektor-Maschinen und ihre Anwendung auf Datensätze aus der Forschung
  Bericht Jül-4096, Forschungszentrum Jülich, Oktober 2003

- Natali Zint
  Robuste Verfahren zur Berechnung von Korrelationsmatrizen sowie zur Hauptkomponentenanalyse
  Beiträge zum Wissenschaftlichen Rechnen – Ergebnisse des Gaststudentenprogramms 2003 des John von Neumann-Instituts für Computing
  Esser, Rüdiger (Hrsg.), Interner Bericht FZJ-ZAM-IB-2003-10, Dezember 2003
- Forschungszentrum Jülich
  Raffinierte Datenanalyse kann Suche nach neuen Medikamenten verkürzen
  Pressemitteilung, Dezember 2003
- Claudia Druska
  Robuste statistische Verfahren für das Data Mining
  Interner Bericht FZJ-ZAM-IB-2004-05, Forschungszentrum Jülich, März 2004
- Achim Kless, Thorsten Dickhaus, Wolfgang Meyer, Johannes Grotendorst
  Data Mining in der pharmazeutischen Forschung und Entwicklung
  Sonderheft LITUS – Laboratory IT User Service, GIT Verlag, April 2004
- Achim Kless, Tatjana Eitrich, Wolfgang Meyer, Johannes Grotendorst
  Data Mining in F&E
  BioWorld, Branchenpublikation für die Biotechnologie, BioTalk GmbH, Schweiz, Mai 2004
- Achim Kless, Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Support Vector Machines Implementing the Nearest Point Algorithm
  Lecture Notes in Artificial Intelligence, Vol. 3303, 191-205, J. A. López, E. Benfenati, W. Dubitzky (Eds.), Springer Verlag, Dezember 2004
- Claudia Albrecht
  Automatisierte Daten-Transformation in einem Data Mining Projekt
  Beiträge zum Wissenschaftlichen Rechnen – Ergebnisse des Gaststudentenprogramms 2005 des John von Neumann-Instituts für Computing
  Esser, Rüdiger (Hrsg.), Interner Bericht FZJ-ZAM-IB-2005-13, Dezember 2005
- Tatjana Eitrich, Bruno Lang
  Analysis of Support Vector Machine Training Costs for Large and Unbalanced Data from Pharmaceutical Industry
  Proceedings of the International Conference on Artificial Intelligence and Machine Learning (AIML), Kairo, Dezember 2005
- Tatjana Eitrich, Achim Kless, Claudia Druska, Wolfgang Meyer, Johannes Grotendorst
  Classification of Highly Unbalanced CYP450 Data of Drugs Using Cost Sensitive Machine Learning Techniques
  Journal of Chemical Information and Modeling, 2007 (first issue)

- Achim Kless, Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Maximum Entropy Methods
  Poster, 16. European Symposium on QSAR and Molecular Modelling, 2006

5.2 Vorträge

- Wolfgang Meyer
  Statistische Modellierung
  GALA Kick-Off-Meeting, Aachen, 21.12.1999
- Johannes Grotendorst
  Computersimulation in Forschung und Anwendung
  GALA Kick-Off-Meeting, Aachen, 21.12.1999
- Heiner Müller-Krumbhaar
  Algorithmen zur Analyse von Grünenthal-Daten
  2. GALA-Meeting, Stolberg, 05.12.2000
- Wolfgang Meyer
  Statistische Analyse von Grünenthal-Daten
  2. GALA-Meeting, Stolberg, 05.12.2000
- Achim Kless
  Data Mining in der pharmazeutischen Industrie
  Kolloquium über Parallelverarbeitung in technisch-naturwissenschaftlichen Anwendungen, Forschungszentrum Jülich, 14.05.2001
- Heiner Müller-Krumbhaar
  Algorithmen zur Analyse von Grünenthal-Daten
  3. GALA-Meeting, Stolberg, 19.07.2001
- Wolfgang Meyer, Thorsten Dickhaus
  Statistische Analyse von Grünenthal-Daten
  3. GALA-Meeting, Stolberg, 19.07.2001
- Wolfgang Meyer, Thorsten Dickhaus
  Explorative Datenanalyse im GALA-Projekt
  ZAM-Jahresabschlusskolloquium, Jülich, 13.12.2001
- Thorsten Dickhaus
  C-Programm für das GALA-Projekt
  4. GALA-Meeting, Stolberg, 21.02.2002
- Wolfgang Meyer
  Analyse von Dosis-Wirkungs-Zeitreihen
  4. GALA-Meeting, Stolberg, 21.02.2002

- Thorsten Dickhaus
  Statistische Datenanalyse für das GALA-Projekt
  5. GALA-Meeting, Aachen, 26.11.2002

- Achim Kless
  Data Mining in der pharmazeutischen Industrie
  Kolloquium "Praxis der Datenverarbeitung"
  Zentrum für Angewandte Informatik, Universität zu Köln, 08.01.2003

- Tatjana Eitrich
  Binäre Support-Vektor-Maschinen
  6. GALA-Meeting, Stolberg, 10.07.2003

- Thorsten Dickhaus
  Neue Entwicklungen im Outlier-Programm
  6. GALA-Meeting, Stolberg, 10.07.2003

- Natali Zint
  Robuste Verfahren zur Kovarianzberechnung und Hauptkomponentenanalyse
  Vortrag im Gaststudenten-Programm, John von Neumann-Institut für Computing, Jülich, 02.10.2003

- Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Support Vector Machines Implementing the Nearest Point Algorithm
  International Symposium Kelsi, Mailand, 26.11.2004

- Claudia Druska
  Statistische Verfahren zur Datenvorauswertung in der pharmazeutischen Forschung
  7. GALA-Meeting, Aachen, 01.03.2005

- Tatjana Eitrich
  Klassifikationsalgorithmen für Daten aus der Pharmaindustrie
  7. GALA-Meeting, Aachen, 01.03.2005

- Johannes Grotendorst
  Kooperation Forschungszentrum Jülich und Grünenthal GmbH - Status und Perspektiven
  7. GALA-Meeting, Aachen, 01.03.2005

- Claudia Albrecht
  Automatisierte Daten-Transformation im GALA-Projekt
  Vortrag im Gaststudenten-Programm, John von Neumann-Institut für Computing, Jülich, 27.09.2005

- Tatjana Eitrich
  Analysis of Support Vector Machine Training Costs for Large and Unbalanced Data from Pharmaceutical Industry
  International Conference on Artificial Intelligence and Machine Learning (AIML), Kairo, 20.12.2005

Literaturverzeichnis

[1] H. Bauknecht, A. Zell, H. Bayer, P. Levi, M. Wagener, J. Sadowski, J. Gasteiger. *Locating Biologically Active Compounds in Medium-Sized Heterogeneous Datasets by Topological Autocorrelation Vectors: Dopamine and Benzodiazepine Agonists.* J. Chem. Inf. Comput. Sci. 1996, 36, 1205-1213.

[2] M. Wagener, J. Sadowski, J. Gasteiger. *Autocorrelation of Molecular Surface Properties for Modeling Corticosteroid Binding Globulin and Cytosolic Ah Receptor Activity by Neural Networks.* J. Am. Chem. Soc. 1995, 117, 7769-7775.

[3] G. Schneider, W. Neidhard, T. Giller, G. Schmid. *"Grundgerüstwechsel" (Scaffold-Hopping) durch topologische Pharmakophorsuche: ein Beitrag zum virtuellen Screening.* Angew. Chem. 1999, 111, 3068-3070.

Statistische Verfahren zur Datenvorauswertung

Claudia Druska, Thorsten Dickhaus, Wolfgang Meyer

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: {c.druska, w.meyer}@fz-juelich.de

1 Einleitung

Die Entwicklung eines neuen Medikamentes ist ein langwieriger Prozess: Aus einer sehr großen Anzahl von chemischen Substanzen müssen diejenigen ausgewählt werden, die im Organismus eine bestimmte, gewünschte Wirkung (wie z.B. die Linderung eines Schmerzreizes) erzielen. Diese Fähigkeit ist eine biologische Eigenschaft der betreffenden Substanz und kann mit geeigneten Messverfahren bzw. Versuchen quantifiziert werden. Solche Messungen werden in der pharmazeutischen Forschung intensiv durchgeführt und ihre Ergebnisse in Datenbanken abgespeichert. Hierbei treten verschiedene Probleme auf:

- Laborexperimente sind oft aufwändig und teuer.
- Die Datenhaltung ist nicht immer einfach zu organisieren, sodass es zu Dateninkonsistenzen, z.B. durch Fehleingaben, kommen kann.
- Die Anzahl an Messwerten wird schnell unüberschaubar groß.

Aus diesen Gründen ist die pharmazeutische Industrie daran interessiert, einerseits nach Möglichkeit nur die aussagekräftigsten Versuche durchzuführen und andererseits die entstehenden Daten von Fehlern bzw. Ausreißern zu befreien. Hier bietet sich ein Ansatzpunkt für statistische Verfahren.
Es ist möglich, Ähnlichkeiten zwischen Messreihen aufzudecken und damit gegebenenfalls

Experimente einzusparen, die keine wesentlich neuen Informationen gegenüber vorangegangenen Versuchen liefern. Diese Ähnlichkeiten werden mathematisch mit der Größe der Korrelation zwischen Variablen beschrieben, eine Messreihe bzw. ein Experiment wird also auf eine Zufallsvariable abgebildet.
Ferner lässt sich auch die Eigenschaft Ausreißer einzelner Messpunkte (chemischer Strukturen, die in einer Messung untersucht worden sind) statistisch präzisieren, und so geartete Daten können erkannt werden. Dies hat zum einen den Effekt, dass der mit der Datenauswertung beschäftigte Wissenschaftler auf eventuelle Messfehler aufmerksam wird (unerwünschte Ausreißer), und zum anderen können gegebenenfalls Strukturen mit besonders günstigen Eigenschaften, die für die Medikamentenentwicklung besonders geeignet sind, schneller aus der großen erfassten Datenmenge herausgefiltert werden (gewünschte Ausreißer).
Eine andere Vorgehensweise besteht darin, die Ursachen der biologischen Eigenschaften einer Substanz in ihren physikochemischen Eigenschaften wie zum Beispiel der Anordnung der Atome und den Winkeln zwischen den Brückenbindungen in den Molekülen zu suchen. Diese physikochemischen Eigenschaften lassen sich in der Regel viel leichter ermitteln und sind für viele relevante Substanzen häufig schon in großer Menge bekannt. Wenn es nun gelingt, aus den physikochemischen Eigenschaften die biologischen Eigenschaften vorherzusagen, lassen sich im experimentellen Bereich zum Teil erhebliche Ressourcen einsparen. Auch hierfür bieten sich Verfahren der mathematischen Statistik an, im Wesentlichen sind dies Klassifikationsalgorithmen.
Die beschriebene Aufgabenstellung, versteckte Informationen in erfassten Daten aufzudecken, wird auch als Data Mining bezeichnet. Im Rahmen des GALA-Projekts werden verschiedene statistische Verfahren des Data Mining eingesetzt, um Daten aus der pharmazeutischen Forschung zu analysieren und auszuwerten. In den folgenden sieben Abschnitten werden die theoretischen Hintergründe dieser Verfahren erläutert, während in Teil IV auf die Implementierung eingegangen wird.

2 Korrelationsberechnung

Wie bereits in den einleitenden Bemerkungen angeführt, ist es oftmals von Interesse, Ähnlichkeiten bzw. Abhängigkeiten von Messreihen zu erkennen und auszunutzen. Eine Möglichkeit dazu besteht darin, die beobachteten Werte als zweidimensionale Streubilder darzustellen und die Formen der sich ergebenden Punktewolken zu analysieren. Dies ist bei der Analyse von einigen wenigen Messreihen ein durchaus gängiges und plausibles Vorgehen. Liegen jedoch wie im GALA-Projekt sehr viele Messreihen vor, so wird die Anzahl der zu betrachtenden Streubilder (englisch: „scatter plots“) schnell unüberschaubar groß. Ist nämlich die Anzahl zu analysierender Messreihen gleich n, so ergeben sich $\frac{n\cdot(n-1)}{2}$ Streubilder, ihre Anzahl wächst also quadratisch mit der Anzahl der Messreihen.
Eine effizientere Form der Analyse großer Datenmengen besteht darin, die Messreihen auf

Zufallsvariablen abzubilden und statistische Kenngrößen der sich ergebenden Größen zu berechnen und systematisch zu analysieren. In der gegebenen Problemstellung des Messens von Ähnlichkeiten bzw. Abhängigkeiten von Zufallsgrößen (und damit der mit ihnen identifizierten Messreihen) ist insbesondere die Berechnung von Kovarianzen und Korrelationen ein geeignetes Hilfsmittel.

2.1 Theoretische Beschreibung

Für zwei gegebene Zufallsvariablen X und Y ist die Kovarianz definiert als

$$\text{Kov}(X,Y) = \mathbb{E}((X - \mathbb{E}(X)) \cdot (Y - \mathbb{E}(Y))) \quad . \tag{1}$$

Sind die Verteilungen der zu Grunde liegenden Zufallsvariablen unbekannt und liegen lediglich Realisierungen $x_1, \ldots, x_n$ von X sowie $y_1, \ldots, y_n$ von Y vor, so verwendet man die empirische Kovarianz

$$\text{Kov}_n(X,Y) = \frac{1}{n-1} \sum_{i=1}^{n} ((x_i - \bar{x}_n) \cdot (y_i - \bar{y}_n))$$

als Schätzung der Kovarianz. Hierbei bezeichnen $\bar{x}_n$ bzw. $\bar{y}_n$ die arithmetischen Mittel der beobachteten Werte der Zufallsvariablen X bzw. Y. Demnach gelten

$$\bar{x}_n = \frac{1}{n} \sum_{i=1}^{n} x_i \quad \text{und} \quad \bar{y}_n = \frac{1}{n} \sum_{i=1}^{n} y_i \quad .$$

Normiert man nun diese Kovarianzen auf das Intervall $[-1;1]$ bzw. berechnet man die Kovarianzen von auf Varianz 1 normierten Zufallsvariablen, so erhält man den Korrelationskoeffizienten nach Pearson (auch Produktmomentkorrelation genannt)

$$\rho(X,Y) = \frac{\text{Kov}(X,Y)}{\sqrt{\text{Var}(X)} \cdot \sqrt{\text{Var}(Y)}} \tag{2}$$

bzw. den empirischen Korrelationskoeffizienten nach Pearson

$$\rho_n(X,Y) = \frac{\text{Kov}_n(X,Y)}{\sqrt{\text{Var}_n(X)} \cdot \sqrt{\text{Var}_n(Y)}}$$

von X und Y. $\text{Var}(X)$ bzw. $\text{Var}_n(X)$ bezeichnen hierbei die Varianz bzw. die empirische Varianz der Zufallsvariable X, d.h.

$$\text{Var}(X) \quad = \quad \mathbb{E}((X - \mathbb{E}(X))^2) \quad \equiv \quad \text{Kov}(X,X) \quad ,$$

$$\text{Var}_n(X) \quad = \quad \tfrac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x}_n)^2 \quad \equiv \quad \text{Kov}_n(X,X) \quad .$$

Der so definierte Pearsonsche Korrelationskoeffizient $\rho(X, Y)$, der im Folgenden auch kurz als die Korrelation zwischen X und Y bezeichnet wird, ist ein Maß für den linearen Zusammenhang zwischen X und Y. $|\rho(X, Y)| = 1$ bedeutet, dass eine eindeutige lineare Beziehung zwischen den beiden Zufallsvariablen X und Y besteht; analog lässt ein betragsmäßig großer Wert von $\rho_n(X, Y)$ auf eine lineare Abhängigkeit der beiden erfassten Stichproben schließen. Betragsmäßig kleine Werte der (empirischen) Korrelation hingegen deuten darauf hin, dass keine lineare Beziehung der betrachteten Zufallsgrößen bzw. deren Realisierungen zueinander besteht. Dies schließt jedoch nicht aus, dass ein nichtlinearer Zusammenhang zwischen X und Y vorliegt.

2.2 Anwendung

Innerhalb des GALA-Projektes werden die Experimente bzw. Laboruntersuchungen und ihre Ergebnisse als Messreihen interpretiert und durch Zufallsvariablen repräsentiert. Die Realisierungen dieser Größen ergeben sich durch die verschiedenen chemischen Substanzen, die in den Experimenten eingesetzt werden. So ergibt sich als Eingabedatensatz eine $(n \times m)$-Datenmatrix. Hierbei ist m die Anzahl der durchgeführten Experimente (Variablen, Merkmale, Attribute, properties), und n bezeichnet die Anzahl der eingesetzten Substanzen (chemische Strukturen). Es wurden nun paarweise die Korrelationen der Spalten dieser Eingabematrix berechnet und betragsmäßig signifikant große Werte in gesonderten Tabellen ausgegeben. Probleme ergaben sich dabei hauptsächlich durch Inkonsistenzen in den Eingabedatensätzen (oft werden nicht alle Experimente für alle Strukturen durchgeführt), die zu fehlenden Werten (Missing Values) führen. Ist die Anzahl dieser Missing Values groß, so ergibt sich unter Umständen eine dünn besetzte Eingabe-Datenmatrix. Praktisch wurde dieses Problem dadurch gelöst, dass die Korrelationsberechnung nur auf der Basis derjenigen Strukturen durchgeführt wurde, für welche beide jeweils betrachteten Experimente Werte lieferten.
Ein weiteres Problem besteht darin, den Terminus „signifikant groß“ mathematisch zu präzisieren. Ein gängiges statistisches Vorgehen dazu besteht darin, Verteilungsannahmen über die Variablen (im vorliegenden Fall durch die Messreihen induziert) zu treffen und, auf ihnen basierend, Quantile der zugehörigen Verteilungsfunktionen als Schwellenwerte für die Größe der berechneten Statistiken zu verwenden. Aufgrund der Datensicherheitsvorgaben liegt jedoch kein ausreichendes Detailwissen über die Daten und ihre Verteilungen vor, und so muss komplett frei von Verteilungsannahmen gearbeitet werden. Aus diesem Grund wurde die erstellte Software so ausgelegt, dass der Benutzer diese Schwellenwerte selbst eingeben und damit die Signifikanz der Korrelationen und in der Konsequenz auch die Größe der ausgegebenen Datensätze eigenhändig steuern kann. In der Anwendung hat sich dies als ein praktikables Vorgehen erwiesen, da bei Tests Ähnlichkeiten innerhalb der Daten erkannt wurden, die bereits im Vorfeld nachgewiesen waren.

3 Ausreißererkennung

3.1 Motivation

Neben der im letzten Abschnitt beschriebenen Aufgabe, Regelmäßigkeiten in den Daten aufzudecken, ist es oftmals ebenso von Interesse, Datenpunkte (Messwerte) zu erkennen, die signifikant aus dieser ermittelten Struktur herausfallen bzw. sich in der Punktewolke stark außerhalb des Gros' der gemessenen Daten befinden. Ein Datenpunkt mit dieser Eigenschaft wird im Folgenden als Ausreißer bezeichnet. Das Vorliegen von Ausreißern kann verschiedene Gründe haben:

1. Eventuell ist eine fehlerhafte Messung durchgeführt oder ein Messwert falsch eingegeben worden. Ein so gearteter Ausreißer ist sicherlich unerwünscht, und der Anwender möchte diese Dateninkonsistenz erkennen und gegebenenfalls beheben.
2. Wie bereits beschrieben, zielen die implementierten Data Mining Methoden darauf ab, chemische Substanzen mit besonders günstigen Eigenschaften systematisch aus großen Datenmengen herauszufiltern. Diese günstigen Eigenschaften lassen sich in experimentell ermittelten Daten als außergewöhnliche Werte und damit als erwünschte Ausreißer ablesen.

Ein weiterer Beweggrund dafür, Ausreißer aufzufinden und unter Umständen zu eliminieren (von der Analyse auszunehmen), soll durch die folgenden Abbildungen von Punktewolken mit den zugehörigen Korrelationswerten motiviert werden.

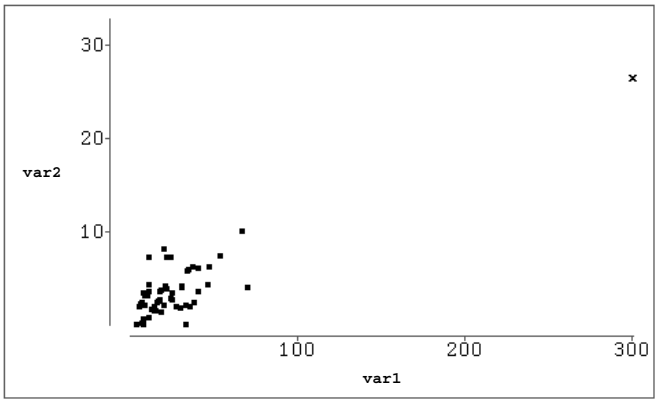


Abbildung 1: Ausreißer induziert signifikante Korrelation.

In dem in Abbildung 1 dargestellten Fall liefert die Korrelationsberechnung einen hohen Wert von $\rho_n(\mathrm{var1}, \mathrm{var2}) = 0,8703$, obwohl der überwiegende Teil der Beobachtungen eine unregelmäßige Struktur aufweist. Hier induziert also der in der Abbildung markierte, weit vom Datenzentrum entfernte Datenpunkt eine lineare Abhängigkeit, die jedoch kein

Charakteristikum der Stichprobe insgesamt darstellt.

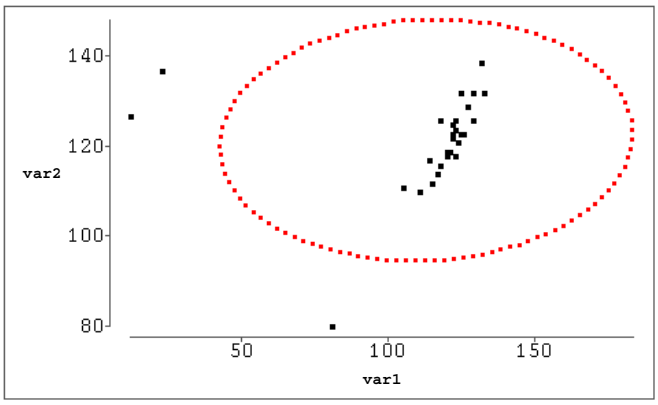


Abbildung 2: Ausreißer verdecken signifikante Korrelation (29 Wertepaare).

Der gegenteilige Fall ist in Abbildung 2 dargestellt. Das Gros der Daten zeigt eine lineare Beziehung zwischen Variable 1 und Variable 2. Die drei Datenpunkte, die außerhalb der eingezeichneten Ellipse liegen, verdecken jedoch diese Abhängigkeit und führen dazu, dass sich kein signifikant großer Wert für $\rho_{29}(\mathrm{var1}, \mathrm{var2})$ errechnet. Die empirische Korrelation ist in diesem Fall $\rho_{29}(\mathrm{var1}, \mathrm{var2}) = 0,0560$.
Nimmt man die drei Ausreißer von der Berechnung aus, so zeigt sich die lineare Beziehung des restlichen Datenmaterials eindeutig in der Kennzahl $\rho_{26}(\mathrm{var1}, \mathrm{var2}) = 0,8609$. Grafisch wird dies in Abbildung 3 deutlich.

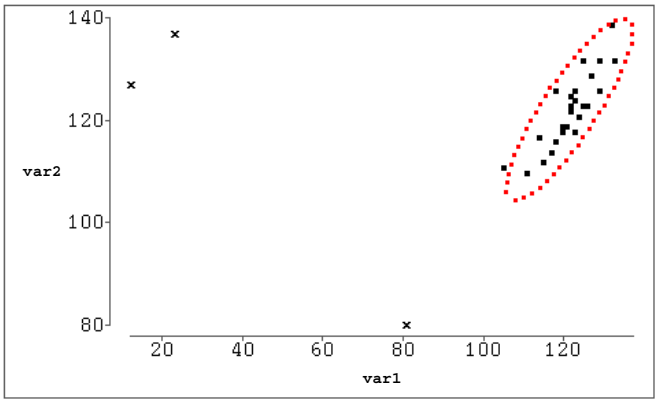


Abbildung 3: Aufgedeckte Korrelation durch Vernachlässigung von Ausreißern (26 Wertepaare).

Wie diese Beispiele zeigen, können einzelne Ausreißer zum Teil erheblichen Einfluss auf die Korrelationsberechnung haben. Dies liegt im Wesentlichen daran, dass in der Formel für $\rho_n(X, Y)$ jedem Datenpunkt das gleiche Gewicht zugeordnet wird und sich daher un-

verhältnismäßig große lokale Abweichungen vom Gesamtverhalten erheblich in der globalen Kenngröße der empirischen Korrelation niederschlagen.
Die Hauptschwierigkeit besteht nun darin, diese lokalen Inhomogenitäten mathematisch zu messen, um beurteilen zu können, wann diese als signifikant zu betrachten sind und nicht lediglich in der üblichen Schwankungsbreite der experimentellen Ungenauigkeit liegen. Außerdem muss dieses Vorgehen automatisiert werden, da nicht jedes Streubild einzeln ausgewertet werden kann.

3.2 Theoretische Beschreibung

Die zunächst naheliegendste Vorgehensweise zum Entdecken von Ausreißern besteht darin, für jeden Datenpunkt $z_i = (x_i, y_i), i = 1, \ldots, n$ der erhobenen zweidimensionalen Stichprobe dessen Euklidischen Abstand vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ zu ermitteln und als Abstandsmaß zu verwenden. Dabei wird der Euklidische Abstand $d(a, b)$ zweier Punkte $a = (a_1, a_2)$ und $b = (b_1, b_2) \in \mathbb{R}^2$ definiert als

$$d(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \quad .$$

Stammt $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n$ und $\bar{y}_n$, so misst

$$d(z_i, \bar{z}_n) = \sqrt{(x_i - \bar{x}_n)^2 + (y_i - \bar{y}_n)^2}$$

den Euklidischen Abstand des Beobachtungspaares $z_i = (x_i, y_i)$ vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$. Dass die Verwendung dieses Abstandsmaßes nicht immer sinnvoll ist, wird durch Abbildung 4 illustriert. Der markierte Datenpunkt hat zwar keinen großen Euklidischen Abstand vom Datenzentrum, fällt aber doch aus der Punktewolke heraus, da er nicht in der generellen Streuungsrichtung liegt.

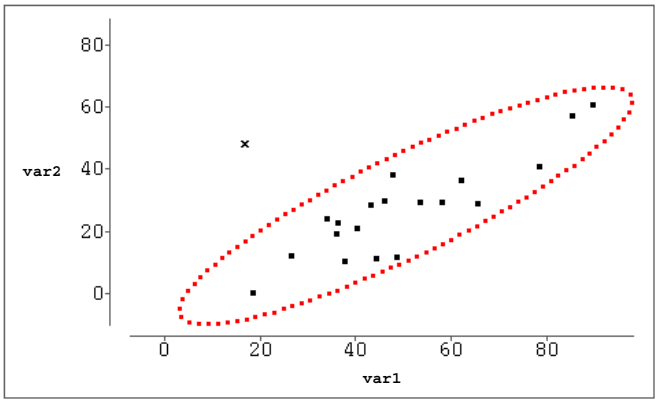


Abbildung 4: Ausreißer trotz geringen Euklidischen Abstandes.

Will man diese Zusatzinformation über die „Ausrichtung“ der Punktewolke mit in die Abstandsberechnung einfließen lassen, so bietet sich die Verwendung der sogenannten Mahalanobis-Distanz $\Delta(z_i, \bar{z}_n)$ an. Hierbei wird in die Formel für den Euklidischen Abstand eine Gewichtung mit der inversen Kovarianzmatrix eingefügt, um der unterschiedlichen Streuung in x- und y-Richtung Rechnung zu tragen. Die quadrierte Mahalanobis-Distanz eines Punktes $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n$ und $\bar{y}_n$ vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ ist definiert als

$$\Delta^2(z_i, \bar{z}_n) = \begin{pmatrix} (x_i - \bar{x}_n) \\ (y_i - \bar{y}_n) \end{pmatrix}^\top \begin{pmatrix} \mathrm{Var}_n(X) & \mathrm{Kov}_n(X,Y) \\ \mathrm{Kov}_n(X,Y) & \mathrm{Var}_n(Y) \end{pmatrix}^{-1} \begin{pmatrix} (x_i - \bar{x}_n) \\ (y_i - \bar{y}_n) \end{pmatrix}. \quad (3)$$

Besonders einfach wird die Berechnung der Mahalanobis-Distanz, falls die zu Grunde liegenden Größen auf Mittelwert 0 und Varianz 1 standardisiert werden. Für die quadrierte Mahalanobis-Distanz eines Punktes $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n = 0$ und $\bar{y}_n = 0$ sowie $\mathrm{Var}_n(X) = \mathrm{Var}_n(Y) = 1$ vom Datenzentrum $\bar{z}_n = (0, 0)$ ergibt sich

$$\begin{aligned} \Delta^2(z_i, \bar{z}_n) &= \begin{pmatrix} x_i \\ y_i \end{pmatrix}^\top \begin{pmatrix} 1 & \rho_n(X,Y) \\ \rho_n(X,Y) & 1 \end{pmatrix}^{-1} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \\ &= \frac{1}{1 - \rho_n(X,Y)^2} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix}^\top \begin{pmatrix} 1 & -\rho_n(X,Y) \\ -\rho_n(X,Y) & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \\ &= \frac{x_i^2 + y_i^2 - 2 \cdot x_i \cdot y_i \cdot \rho_n(X,Y)}{1 - \rho_n(X,Y)^2} \quad . \end{aligned} \quad (4)$$

Aufgrund dessen sollte man in der praktischen Berechnung die Größen X und Y zunächst standardisieren, damit (4) für $\Delta(z_i, \bar{z}_n)$ verwendet werden kann.
Geometrisch lässt sich der Übergang vom Euklidischen Abstand zur Mahalanobis-Distanz so interpretieren, dass das zu Grunde liegende Koordinatensystem im $\mathbb{R}^2$ derart neu gewählt wird, dass die erste Achse in Richtung der stärksten Streuung (repräsentiert über den Richtungsvektor v_1) und die zweite Achse in Richtung $v_2 \perp v_1$ weist. Zudem wird der Koordinatenursprung in den Punkt $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ verschoben. Überführt man nun den Zufallsvektor (X, Y) mittels einer linearen Transformation so in den Zufallsvektor (C_1, C_2), dass dies äquivalent zu einem Basiswechsel von der kanonischen in die oben charakterisierte Basis ist, so bezeichnet man die transformierten Zufallsgrößen C_1 und C_2 als die Hauptkomponenten der zu Grunde liegenden zweidimensionalen Verteilung. Die Mahalanobis-Distanz ist nun, wie im Folgenden gezeigt wird, der Euklidische Abstand in dem veränderten Koordinatensystem, wobei eine Skalierung der neuen Koordinatenachsen in Einheiten der Standardabweichung der Hauptkomponenten stattfindet.

Führt man diese Berechnung für standardisierte Größen durch, so ist dies äquivalent zur Lösung des Eigenwertproblems der Korrelationsmatrix

$$K_n := \begin{pmatrix} 1 & \rho_n(X,Y) \\ \rho_n(X,Y) & 1 \end{pmatrix} \quad ,$$

denn es gilt

$$\begin{aligned} v_1 &= \underset{a \in \mathbb{R}^2}{\operatorname{argmax}} \ \mathrm{Var}_n(a^\top \cdot (X,Y)^\top) \\ &= \underset{a \in \mathbb{R}^2}{\operatorname{argmax}} \ a^\top \cdot \begin{pmatrix} \mathrm{Var}_n(X) & \mathrm{Kov}_n(X,Y) \\ \mathrm{Kov}_n(X,Y) & \mathrm{Var}_n(Y) \end{pmatrix} \cdot a \end{aligned}$$

und (X,Y) wird hier als standardisiert vorausgesetzt, so dass die Varianz- / Kovarianzmatrix in die oben angegebene Korrelationsmatrix K_n übergeht.
Wie Ergebnisse aus der linearen Algebra (Maximierung von quadratischen Formen) zeigen, ergibt sich v_1 als Eigenvektor zum größeren Eigenwert λ_1 und v_2 als Eigenvektor zum kleineren Eigenwert λ_2 von K_n. Ferner ist die Varianz- / Kovarianzmatrix bezüglich des neuen Koordinatensystems gleich der diagonalisierten Korrelationsmatrix

$$K_D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \quad .$$

Anders ausgedrückt ist $\mathrm{Var}_n(C_1) = \lambda_1$ und $\mathrm{Var}_n(C_2) = \lambda_2$. Daraus ergibt sich, dass die oben angeführte Skalierung entlang der Hauptkomponenten dadurch zu geschehen hat, dass die Normierungsfaktoren $\frac{1}{\sqrt{\lambda_i}}, i = 1,2$ für die Richtungen $v_i, i = 1,2$ verwendet werden.
Die konkrete Berechnung des Euklidischen Abstandes in dem veränderten Koordinatensystem ergibt die geschlossene Formel (4). Bezeichne hierzu $\chi_{K_n}(\lambda)$ das charakteristische Polynom der standardisierten Korrelationsmatrix K_n, so ergibt sich

$$\chi_{K_n}(\lambda) = \det(K_n - \lambda E) = \begin{vmatrix} (1-\lambda) & \rho_n(X,Y) \\ \rho_n(X,Y) & (1-\lambda) \end{vmatrix} = (1-\lambda)^2 - \rho_n(X,Y)^2 \quad ,$$

wobei mit E die Einheitsmatrix der Dimension n bezeichnet wird. Bei der Berechnung der Eigenwerte erhält man

$$\chi_{K_n}(\lambda) = 0 \ \Leftrightarrow \ (1-\lambda)^2 = \rho_n(X,Y)^2 \ \Leftrightarrow \ |(1-\lambda)| = |\rho_n(X,Y)| \quad .$$

Daraus folgt

$$\lambda_1 = 1 + |\rho_n(X,Y)| \quad \wedge \quad \lambda_2 = 1 - |\rho_n(X,Y)| \quad \wedge \quad \lambda_1 \geq \lambda_2 \quad .$$

Für den ersten Eigenvektor ergibt sich daher

$$\begin{aligned}
& (K_n - \lambda_1 \cdot E) \cdot v_1 && = && 0 \\
\iff & \begin{pmatrix} -|\rho_n(X,Y)| & \rho_n(X,Y) \\ \rho_n(X,Y) & -|\rho_n(X,Y)| \end{pmatrix} \cdot v_1 && = && 0 \\
\iff & \begin{aligned} -|\rho_n(X,Y)|v_{11} + \rho_n(X,Y)v_{12} &= 0 \\ \rho_n(X,Y)v_{11} - |\rho_n(X,Y)|v_{12} &= 0 \end{aligned}
\end{aligned}$$

und man erhält

$$v_{11} = \text{sgn}(\rho_n(X,Y)) \cdot v_{12} \;\Rightarrow\; v_1 = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ \text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad .$$

Die analoge Berechnung des zweiten Eigenvektors liefert entsprechend

$$\begin{aligned}
& (K_n - \lambda_2 \cdot E) \cdot v_2 && = && 0 \\
\iff & \begin{pmatrix} |\rho_n(X,Y)| & \rho_n(X,Y) \\ \rho_n(X,Y) & |\rho_n(X,Y)| \end{pmatrix} \cdot v_2 && = && 0 \\
\iff & \begin{aligned} |\rho_n(X,Y)|v_{21} + \rho_n(X,Y)v_{22} &= 0 \\ \rho_n(X,Y)v_{21} + |\rho_n(X,Y)|v_{22} &= 0 \end{aligned}
\end{aligned}$$

$$\Rightarrow v_{22} = -\text{sgn}(\rho_n(X,Y)) \cdot v_{21} \;\Rightarrow\; v_2 = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad .$$

Somit hat die Matrix T der entsprechenden Basistransformation die Form

$$T = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 & 1 \\ \text{sgn}(\rho_n(X,Y)) & -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad ,$$

und die transformierten Koordinaten eines Vektors $(x_i, y_i)^\top \in \mathbb{R}^2$, die sich als $(x_i^*, y_i^*) = (x_i, y_i) \cdot T$ berechnen lassen, ergeben sich zu

$$\begin{aligned}
(x_i^*, y_i^*) = (x_i, y_i) \cdot T &= \frac{1}{\sqrt{2}} \cdot (x_i, y_i) \cdot \begin{pmatrix} 1 & 1 \\ \text{sgn}(\rho_n(X,Y)) & -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \\
&= \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} x_i + \text{sgn}(\rho_n(X,Y)) \cdot y_i \\ x_i - \text{sgn}(\rho_n(X,Y)) \cdot y_i \end{pmatrix}^\top \quad .
\end{aligned}$$

Diese Darstellung gestattet nun die Berechnung von $\Delta(z_i, \bar{z}_n)$ eines Punktes (x_i, y_i) aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n = 0$ und $\bar{y}_n = 0$ sowie $\mathrm{Var}_n(X) = \mathrm{Var}_n(Y) = 1$ vom Datenzentrum $(0, 0)$ als Euklidischen Abstand des transformierten Punktes vom Koordinatenursprung unter Beachtung der Normierung entlang der Hauptkomponenten:

$$
\begin{aligned}
\Delta^2(z_i, \bar{z}_n) &= \left(\frac{x_i + \mathrm{sgn}(\rho_n(X,Y)) \cdot y_i}{\sqrt{2}\sqrt{1 + |\rho_n(X,Y)|}} \right)^2 + \left(\frac{x_i - \mathrm{sgn}(\rho_n(X,Y)) \cdot y_i}{\sqrt{2}\sqrt{1 - |\rho_n(X,Y)|}} \right)^2 \\
&= \frac{x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2}{2(1 + |\rho_n(X,Y)|)} + \frac{x_i^2 - 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2}{2(1 - |\rho_n(X,Y)|)} \\
&= \frac{(x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2)(1 - |\rho_n(X,Y)|)}{2(1 - \rho_n(X,Y)^2)} \\
&\quad + \frac{(x_i^2 - 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2)(1 + |\rho_n(X,Y)|)}{2(1 - \rho_n(X,Y)^2)} \\
&= \frac{1}{2(1 - \rho_n(X,Y)^2)} \cdot [2x_i^2 + 2y_i^2 - |\rho_n(X,Y)| \cdot (x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i \\
&\qquad + y_i^2 - x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i - y_i^2)] \\
&= \frac{2x_i^2 + 2y_i^2 - 4 \cdot |\rho_n(X,Y)| \cdot \mathrm{sgn}(\rho_n(X,Y)) \cdot x_i \cdot y_i}{2(1 - \rho_n(X,Y)^2)} \\
&= \frac{x_i^2 + y_i^2 - 2 \cdot x_i \cdot y_i \cdot \rho_n(X,Y)}{1 - \rho_n(X,Y)^2} \quad .
\end{aligned}
$$

Dieses Konzept der Zerlegung der Gesamtstreuung in orthogonale Streukomponenten wird in Abschnitt 6 auf Dimensionen $m \geq 3$ zur so genannten Hauptkomponentenanalyse erweitert.
Als Ausreißer werden Datenpunkte mit einer großen Mahalanobis-Distanz zu $(\bar{x}_n, \bar{y}_n)$ deklariert. In konkreten Anwendungen besteht dann die grundlegende Fragestellung darin, ab wann $\Delta^2(z_i, \bar{z}_n)$ als so groß angesehen werden soll, dass dem zugehörigen Punkt das Attribut Ausreißer zugeordnet wird. Dazu gibt es eine Reihe möglicher Ansätze. Der folgende Abschnitt beschreibt die Methoden, die im Projekt angewendet werden.

3.3 Anwendung

3.3.1 Naiver Ansatz

Der erste, naive Ansatz zur Festlegung einer kritischen Schranke Δ_{krit} für den Wert der Mahalanobis-Distanz bestand darin, besonders markante Streubilder wie etwa in Abbildung 1 oder Abbildung 2 zu betrachten. Hier ist es, wie bereits beschrieben, in jedem Fall wünschenswert, dass die vorhandenen Ausreißer automatisch erkannt werden, da durch ihr Vorhandensein weitergehende Berechnungen wie die der empirischen Korrelation empfindlich gestört werden. Aufgrund dessen wurden mehrere Beispiele, in denen dies der Fall ist, analysiert und festgestellt, wie Δ_{krit} jeweils zu wählen wäre, um diese Ausreißer ausfindig zu machen. Es ergab sich, dass in der Regel bei diesen augenscheinlichen Ausreißern eine Mahalanobis-Distanz von $\Delta \geq 2$, häufig sogar von $\Delta \geq 2,5$ vorlag.
Von diesen empirischen Ergebnissen ausgehend wurde daraufhin der konstante Wert $\Delta_{\text{krit}} = 2$ (für fast sicheres Erkennen) bzw. $\Delta_{\text{krit}} = 2,5$ als kritische Mahalanobis-Distanz vereinbart. Der Vorteil dieser Methode besteht darin, dass sie völlig frei von Verteilungsannahmen ist und daher zu ihrer Anwendbarkeit nichts über die Herkunft des zu Grunde liegenden Datenmaterials bekannt sein muss. Für zuerst ausschließlich betrachtete kleine Stichprobenumfänge wie z.B. $n = 30$ lieferte dieses Vorgehen gute Ergebnisse. Im Falle von Messreihen mit erheblich größerem Stichprobenumfang zeigte sich allerdings der große Nachteil des Verfahrens: Mit wachsendem n wurde der Anteil der so bestimmten Ausreißer immer größer und führte dazu, dass die erzeugte Information für den Analysten nicht mehr überschaubar blieb. Dies lässt sich damit begründen, dass bei kleinen Stichprobenumfängen die Kenngrößen arithmetisches Mittel und empirische Varianz durch einzelne Ausreißer stark beeinflusst werden. Dies führt dazu, dass die Mahalanobis-Distanz dieser Ausreißer zu dem berechneten Datenzentrum nur eine bestimmte Maximalgröße annehmen kann und der kritische Wert Δ_{krit} daher nicht zu groß gewählt werden darf. Liegt hingegen ein großer Stichprobenumfang n vor, so nimmt der Einfluss einzelner Beobachtungen auf die Kenngrößen ab und deswegen wird bei Beibehaltung der ursprünglichen kritischen Schranke ein wesentlich größerer Anteil der Beobachtungen als Ausreißer deklariert. Nachteilig daran ist vor allem, dass viele der so deklarierten Beobachtungen gar keine Ausreißer im ursprünglichen Sinne sind. Es wäre also wünschenswert, dass auch Δ_{krit} mit wachsendem n größer wird, damit nicht etwa bei z.B. $n = 1000$ fast 50 % der Datenpunkte als Ausreißer aufgeführt werden. Dies hätte im Falle der Beibehaltung des naiven Ansatzes jedoch zur Folge, dass für jede Analyse eine neue Konstante gewählt werden müsste und dass diese überdies einen Kausalbezug zu n haben sollte.
Es ist daher naheliegend, Δ_{krit} nicht als konstant, sondern als Funktion des Stichprobenumfangs n anzusetzen. Im Folgenden werden nun zwei Verfahren beschrieben, die einen solchen Bezug zwischen n und Δ_{krit} herstellen.

3.3.2 Verwendung asymptotischer Verteilungsquantile

Wie in Abschnitt 3.2 hergeleitet, misst die Mahalanobis-Distanz den Euklidischen Abstand standardisierter Zufallsgrößen in einem speziellen Koordinatensystem. Bezeichne also $C = (C_1, C_2)$ die Hauptkomponenten und $\mu = (\mu_1, \mu_2)$ den zweidimensionalen Erwartungswert der zu Grunde liegenden Verteilung, so gilt:

$$\Delta^2(C, \mu) = \frac{(C_1 - \mu_1)^2}{\text{Var}(C_1)} + \frac{(C_2 - \mu_2)^2}{\text{Var}(C_2)} \quad . \tag{5}$$

Bei (5) handelt es sich um eine Summe von Quadraten zweier standardisierter Zufallsgrößen. Unterliegen diese Zufallsgrößen nun einer Normalverteilung, so ist die Verteilung von $\Delta^2(C, \mu)$ bekannt als χ^2-Verteilung mit zwei Freiheitsgraden oder kurz χ^2_2-Verteilung. Im Falle der entsprechenden empirischen Größen gilt dies zwar nur asymptotisch, aber es ist dennoch möglich, Quantile der χ^2_2-Verteilung für Δ_{krit} zu verwenden, wenn man eine normalverteilte Grundgesamtheit unterstellt.
Praktisch kann damit z.B. das oben angedeutete Ziel, den mittleren Anteil an Ausreißern zu kontrollieren, verfolgt werden. Die Forderung lautet in mathematischer Notation

$$\mathbb{P}(\Delta > \Delta_{\text{krit}}) = \alpha(n) \stackrel{!}{=} \frac{\mu(n)}{n}$$

mit $\mu(n)$ als gewünschter Anzahl an Ausreißern bei Stichprobengröße n und kann durch die Wahl von $\Delta^2_{\text{krit}}(n) = \chi^2_{2;1-\alpha(n)}$ realisiert werden. $\chi^2_{2;1-\alpha(n)}$ bezeichnet hierbei das $(1 - \alpha(n))$-Quantil der χ^2-Verteilung mit zwei Freiheitsgraden.
In dem konkreten Anwendungsbeispiel der Analyse pharmazeutischer Daten wurde $\mu(n) = \sqrt{n/10}$ festgelegt, d.h. aus 1000 Datenpunkten sollten im Mittel etwa zehn Ausreißer herausgefunden werden. Es sei hier aber noch einmal darauf hingewiesen, dass in die Berechnung die Normalverteilungsannahme eingeht und die für $\Delta^2(z_i, \bar{z}_n)$ verwendete Verteilung nur asymptotisch gilt.

3.3.3 Erkennen von echten Ausreißern

Ein ambitionierteres Vorgehen besteht darin, dass der Forderung nachgegangen wird, nur „echte“ Ausreißer zu erkennen. Hier wird also nicht versucht, die Anzahl der sich ergebenden Ausreißer zu steuern, sondern es wird eine Signifikanzschranke für die Verlässlichkeit der Ausreißerklassifikation gesucht. Präziser formuliert lautet die Forderung, dass mit Wahrscheinlichkeit $(1 - \alpha)$ keine Ausreißer erkannt werden sollen, wenn keine echten Ausreißer vorliegen. Um dies bei gegebenem Signifikanzniveau α zu gewährleisten, wählt man Δ_{krit} derart, dass $\mathbb{P}(\Delta_{\max} > \Delta_{\text{krit}}) \stackrel{!}{=} \alpha$ erfüllt ist. Hierbei bezeichnet $\Delta_{\max}$ die größte in der Stichprobe auftretende Mahalanobis-Distanz. Liegt also eine homogene Stichprobe ohne echte Ausreißer vor, so wird der Messwert mit der größten Mahalanobis-Distanz auch

nur in $\alpha \cdot 100\%$ der Fälle als Ausreißer deklariert.
Schwierigkeiten bereitet hierbei die konkrete Berechnung von $\Delta_{\text{krit}}(n, \alpha)$ bei gegebenem Stichprobenumfang n und vorgegebenem Signifikanzniveau α. In [1] ist diesem Problem nachgegangen worden, und es finden sich auf den Seiten 516f. Tabellen für $\Delta_{\text{krit}}(n, \alpha)$ mit n im Bereich von 3 bis 1000 und $\alpha = 0,01$ sowie $\alpha = 0,05$. Diese Tabellenwerte wurden in der erfolgten Implementierung übernommen, und für $1000 < n \leq 4000$ wurde die Tabelle durch Simulationsrechnungen erweitert. Ab $n > 4000$ wurde dann zur asymptotischen Verteilung, die sich als χ^2_2-Verteilung zur n-ten Potenz ergibt, übergegangen (siehe hierzu auch Abschnitt 6.4.2 mit $p = 2$).

3.3.4 Mehrstufiges Vorgehen

Wie bereits in den einleitenden Bemerkungen zu diesem Abschnitt beschrieben, kann sich das Vorhandensein von Ausreißern verfälschend auf die berechneten Werte von $\rho_n(X, Y)$ auswirken. Des Weiteren kann es dazu kommen, dass Ausreißer mit sehr großen Mahalanobis-Distanzen solche Ausreißer überdecken, für die Δ zwar im Verhältnis zu den erkannten Ausreißern klein ist, die aber dennoch erkannt werden sollten. Als Beispiel dafür sei das in Abbildung 5 dargestellte Streubild von Variable 3 und Variable 4 angeführt.

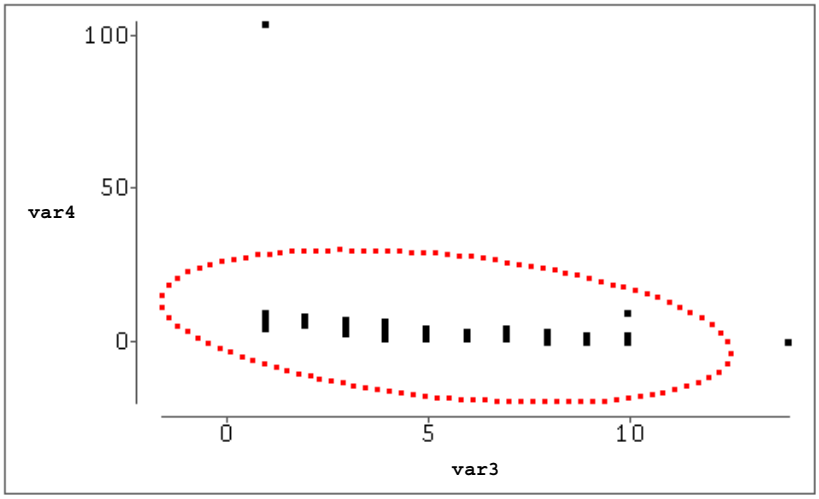


Abbildung 5: Variable 3 gegen Variable 4 (alle 109 Beobachtungen).

Es sind zwei Ausreißer auszumachen, die außerhalb der eingezeichneten Konfidenzellipse zur Mahalanobis-Distanz $\Delta = 2,5$ liegen. Der Rest des Datenmaterials weist optisch keine weiteren Ausreißer auf. Nimmt man die zwei erkannten Datenpunkte nun von der Berechnung aus, d.h. eliminiert man die zwei erkannten Ausreißer, so ergibt sich für die verbleibenden 107 Datenpaare das Streubild in Abbildung 6.

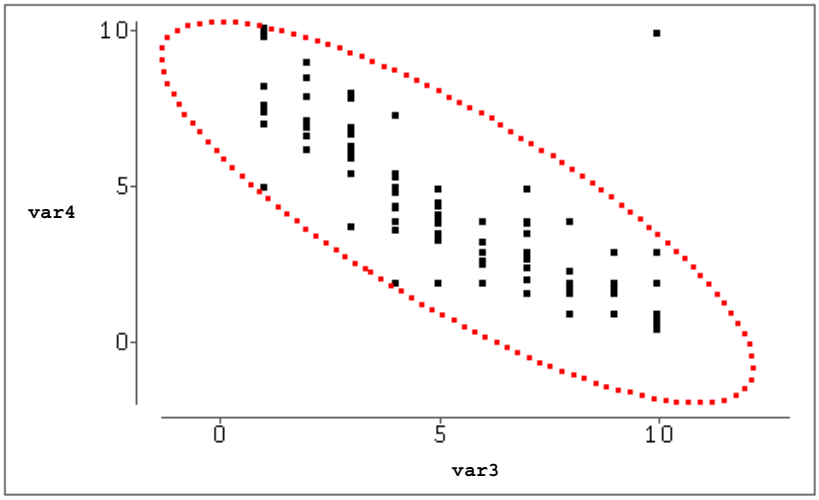


Abbildung 6: Variable 3 gegen Variable 4 (restliche 107 Beobachtungen).

Aufgrund der veränderten Skalierung des Plots klar zu erkennen, zeigt sich nun ein weiterer Ausreißer, für den beide Variablen etwa den Wert 10 annehmen. Dieser Ausreißer wäre unerkannt geblieben, falls die Ausreißererkennung für diese Variablenkombination nur einmalig durchgeführt worden wäre.
Deswegen wurde bei der Umsetzung der Projektziele ein iteratives Vorgehen der Ausreißerdetektion und -elimination umgesetzt. Nachdem auf der Basis aller vorliegender Datenpaare $z_i = (x_i, y_i), i = 1, \ldots, n$ die empirische Korrelation $\rho_n(X, Y)$ berechnet wurde, werden die Mahalanobis-Distanzen $\Delta(z_i, \bar{z}_n)$ dieser Messpunkte vom Datenzentrum bestimmt und diejenigen Datenpunkte $z_i^* = (x_i^*, y_i^*)$, bei welchen $\Delta(z_i^*, \bar{z}_n)$ eine signifikante Größe hat, gekennzeichnet. Danach wird eine Neuberechnung der empirischen Korrelation durchgeführt, wobei die Paare (x_i^*, y_i^*) nicht mehr berücksichtigt werden. Dies kann als eine Robustifizierung der Berechnung von $\rho_n(X, Y)$ angesehen werden. Ferner wird in jeder Iteration mit den neu berechneten Mahalanobis-Distanzen die Ausreißerbestimmung fortgesetzt, um gegebenenfalls noch unerkannte Ausreißer ermitteln zu können.

3.4 Ergebnisse

In Tabelle 1 sollen die Ergebnisse der drei verwendeten Methoden anhand eines Beispieldatensatzes quantitativ gegenüber gestellt werden. Dieser Beispieldatensatz verfügt über 1017 Merkmale mit 11363 Beobachtungen. Die Ergebnisse illustrieren die Vorteile der dritten durchgeführten Methode. Die absolute Anzahl an Ausreißern wird drastisch reduziert, und dabei wird insbesondere der Effekt erzielt, dass nur noch wirklich auffällige Beobachtungen in Erscheinung treten. Dies ist daran erkennbar, dass der Prozentsatz der vier Beobachtungen, die in allen Variablenkombinationen insgesamt am häufigsten als Ausreißer auftreten, also wirklich aus dem Rahmen fallen, im Falle der Anwendung der

	Ausreißer	davon 4 häufigste	Korrelationen $\geq 0,8$
$\Delta_{\text{krit}} = 2,0$	≈ 600.000	5 %	≈ 20.000
$\Delta_{\text{krit}} = 2,5$	≈ 180.000	5 %	≈ 11.000
$\Delta_{\text{krit}}(n) = \chi^2_{2;1-\frac{1}{\sqrt{10n}}}$	≈ 180.000	9 %	≈ 17.500
„Echt“ mit $\alpha = 0,05$	≈ 70.000	11 %	≈ 17.000
„Echt“ mit $\alpha = 0,01$	≈ 40.000	16 %	≈ 16.000

Tabelle 1: Vergleich der Methoden zur Ausreißerbehandlung.

Methoden zur Erkennung „echter“ Ausreißer stark ansteigt; es erfolgt also eine Konzentration auf die auffälligsten Datenpunkte. Ferner wird der Wert von ρ_n und damit die Anzahl der als signifikant betrachteten Korrelationen durch die Wahl der so gearteten Verfahren kaum beeinflusst. Es gibt allerdings einzelne Fälle, in denen durch diese Art der Ausreißerbehandlung signifikante Korrelationen verloren gehen. In Abschnitt 5 wird mit der Rangkorrelation nach Spearman ein Konzept vorgestellt, mit dem dies verhindert werden kann.

4 Transformation von Variablen

4.1 Motivation

Wie bereits in Abschnitt 2 erläutert, ist der (empirische) Korrelationskoeffizient nach Pearson nur ein Maß für lineare Abhängigkeiten zwischen Zufallsvariablen bzw. Messreihen. Allgemein lassen sich nichtlineare Abhängigkeiten in lineare Abhängigkeiten überführen, indem eine oder gegebenenfalls auch beide zu Grunde liegenden Zufallsgrößen geeigneten Transformationen unterzogen wird bzw. werden. In der Regel ist es jedoch keineswegs offensichtlich, wie diese linearisierenden Transformationen zu wählen sind, denn der Typ der vorliegenden Beziehung geht maßgeblich in die Wahl der Transformationen ein.
Es sei an dieser Stelle noch darauf hingewiesen, dass das Konzept der Variablentransformation auch anderen Absichten dienen kann. Wie unter anderem in [8] ausführlich beschrieben, kann einer eindimensionalen Verteilung durch geeignete Transformation die Schiefe genommen oder durch Maßstabsänderung (z.B. Logarithmierung) eine bessere Auflösung bzw. Darstellbarkeit gemessener Daten erreicht werden. Da dies aber nicht zu den Hauptzielen des Projektes zählt, soll hierauf an dieser Stelle nicht näher eingegangen werden.
In den im nachfolgenden Abschnitt beschriebenen Fallbeispielen aus der pharmazeutischen Forschung war eine Identifikation des zu wählenden Transformationstyps aufgrund von Expertenwissen bzw. physikalischen Grundzusammenhängen möglich, und es konnte erreicht werden, dass signifikant große Korrelationen nach getätigten Transformationen berechnet wurden, falls Abhängigkeiten des angenommenen Typs vorlagen.

4.2 Anwendung

Neben der offensichtlichen Anwendung der Transformation von Volumendaten durch Übergang zur dritten Wurzel ist auch im Falle von Konzentrationsdaten das Konzept der Transformation eingesetzt worden, um die Abhängigkeit von Variablen zu linearisieren. Hierzu sind Vorüberlegungen aus dem Bereich der Reaktionskinetik erforderlich. Es werden nur die notwendigen Ergebnisse wiedergegeben; die zu Grunde liegenden biologischen Zusammenhänge werden zum Beispiel genauer in [17] und [4] erläutert.
Betrachtet wird die reversible Bindung von Substanzen an Rezeptormoleküle. Wird einem Organismus eine chemische Substanz verabreicht, so kann es dazu kommen, dass diese an sogenannte Rezeptormoleküle im Organismus bindet und die beiden Moleküle reversibel zu einem neuen Molekül reagieren. Es ist nun möglich, die Konzentrationen der an diesen Reaktionsvorgängen beteiligten Ausgangsmoleküle mit der der resultierenden Moleküle mathematisch in Beziehung zu setzen. Dies geschieht im Wesentlichen durch die Einführung der sogenannten Reaktionskonstanten k_1 und k_{-1}. Das in diesem Fall benutzte Modell entstammt dem Massenwirkungsgesetz. Es besagt, dass die Reaktionsvorgänge proportional von den Ausgangskonzentrationen und die inversen Reaktionsvorgänge (Substanz und Rezeptor lösen sich wieder voneinander) proportional von der Konzentration reagierter (verbundener) Moleküle abhängen, wobei die Proportionalitätskonstanten gerade die oben angeführten Zahlen k_1 bzw. k_{-1} sind.
Bezeichnet S die verabreichte Substanz und R den zugehörigen Rezeptor, so kann der beschriebene Reaktionsvorgang wie folgt symbolisiert werden:

$$a \cdot S + R \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} SR \quad .$$

Mit den Bezeichnungen

$C_R(t)$	Konzentration freier Rezeptoren zum Zeitpunkt t,
$C_S(t)$	Konzentration freier Substanzmoleküle zum Zeitpunkt t,
$C_{SR}(t)$	Konzentration blockierter Rezeptoren (verbundener Moleküle) zum Zeitpunkt t und
$C_{R,\text{ges}} = C_R(t) + C_{SR}(t)$	gesamte Rezeptorkonzentration

ergibt sich unter Verwendung des Massenwirkungsgesetzes für die zeitliche Änderung von $C_{SR}(t)$ die folgende Differentialgleichung:

$$\dot{C}_{SR}(t) = k_1 \cdot C_S(t)^a \cdot C_R(t) - k_{-1} C_{SR}(t) \quad .$$

Geht man nun davon aus, dass sich diese Differentialgleichung für $t \to \infty$ stabilisiert und in einen stationären Endzustand mit $\dot{C}_{SR}(t) \equiv 0$ übergeht, so gilt in diesem eingeschwun-

genen Zustand:

$$\begin{aligned} 0 &= k_1 C_S^a \cdot C_R - k_{-1} \cdot C_{SR} \\ &= k_1 C_S^a \cdot (C_{R,\text{ges}} - C_{SR}) - k_{-1} C_{SR} \quad . \end{aligned}$$

Auf das Argument $t \to \infty$ wurde hier aus Gründen der Notationsvereinfachung bei den Konzentrationswerten verzichtet, da diese im eingeschwungenen Zustand konstant sind. Führt man nun noch die Größe

$$\tilde{C}_{SR} := \frac{C_{SR}}{C_{R,\text{ges}}} \quad \hat{=} \quad \text{Anteil blockierter Rezeptoren}$$

ein, so vereinfacht sich die stationäre Gleichung weiter, indem man beide Seiten durch $C_{R,\text{ges}}$ dividiert:

$$\begin{aligned} & k_1 C_S^a \cdot (1 - \tilde{C}_{SR}) - k_{-1} \cdot \tilde{C}_{SR} = 0 \\ \iff \quad & \tilde{C}_{SR} \cdot (k_{-1} + k_1 C_S^a) = k_1 C_S^a \\ \iff \quad & \tilde{C}_{SR} = \frac{k_1 C_S^a}{k_{-1} + k_1 C_S^a} = \frac{C_S^a}{\tilde{k} + C_S^a} \quad , \end{aligned}$$

wobei $\tilde{k} := \frac{k_{-1}}{k_1}$. Hieraus ergibt sich, dass $\tilde{k}^{\frac{1}{a}}$ genau die Substanzkonzentration ist, bei der die Hälfte der Rezeptoren blockiert wird.
In den Laborexperimenten der pharmazeutischen Forschung werden nun häufig folgende Messwerte ermittelt:

- X Anteil der Rezeptoren, die bei gegebener Konzentration C_0 der Substanz blockiert werden,
- Y Substanzkonzentration, bei der die Hälfte der Rezeptoren blockiert wird (k_i-Wert).

Aus der hergeleiteten Beziehung zwischen $\tilde{C}_{SR}$ und dem zugehörigen k_i-Wert kann nun ein Modellierungsansatz für den Zusammenhang von X und Y abgeleitet werden:

$$\begin{aligned} X &= \frac{C_0^a}{Y^a + C_0^a} \\ \iff \quad X \cdot (Y^a + C_0^a) &= C_0^a \\ \iff \quad X \cdot Y^a &= C_0^a (1 - X) \\ \iff \quad Y^a &= C_0^a \cdot \tfrac{1-X}{X} \\ \iff \quad a \cdot \ln(Y) &= a \cdot \ln(C_0) + \ln\left(\tfrac{1-X}{X}\right) \\ \implies \quad \text{logit}(X) &= a \cdot (\ln(Y) - \ln(C_0)) \quad . \end{aligned}$$

Um eine Linearisierung der Abhängigkeit zwischen X und Y zu erreichen, wurde also in den entsprechenden Fällen die Variable Y einer Transformation mit der natürlichen Logarithmusfunktion und die Variable X der sogenannten Logit-Transformation

$$\text{logit}(X) = \ln\left(\frac{1-X}{X}\right)$$

unterzogen.

5 Rangkorrelation

5.1 Motivation

Im vorherigen Abschnitt wurde die Möglichkeit vorgestellt, mit Hilfe geeigneter Transformationen eine nichtlineare Abhängigkeit zwischen zwei Zufallsgrößen X und Y zu linearisieren, sodass diese mit Hilfe der Produktmomentkorrelation aufgedeckt werden kann. Das Hauptproblem bei dieser Vorgehensweise besteht jedoch darin, dass die Wahl der Transformation(en) von der zu Grunde liegenden Beziehung zwischen X und Y abhängt. Häufig ist jedoch Detailwissen über die Herkunft des Datenmaterials nicht bekannt. Damit kann kein Modell für den Zusammenhang zwischen X und Y hergeleitet werden und mögliche linearisierende Transformationen bleiben unentdeckt.
Es wäre also wünschenswert, eine universelle Transformation zur Verfügung zu haben, die möglichst viele Abhängigkeiten linearisiert. Im Bereich der nichtparametrischen Statistik hat sich das Konzept der Rangkorrelation nach Spearman etabliert, mit welchem sich zumindest alle monotonen Abhängigkeiten identifizieren lassen. Das Vorgehen besteht darin, dass nicht der Pearsonsche Korrelationskoeffizient der Originaldaten, sondern die Produktmomentkorrelation der zugehörigen Positionen der Messwerte in der jeweiligen geordneten Stichprobe (der Ränge der Beobachtungen zu X und Y) berechnet wird. Damit spielt also der absolute Abstand der gemessenen Werte keine Rolle, sondern lediglich die Ordnung der Messwerte geht in die Berechnung ein. Aufgrund dessen liefert die Spearmansche Rangkorrelation für alle Variablenkombinationen (X, Y), in denen X über eine monotone (ggfs. auch nichtlineare) Transformation in Y überführt werden kann, betragsmäßig den Wert 1. Um dies zu illustrieren, wurden drei zweidimensionale Stichproben simuliert. Abbildung 7 liegt das Modell eines quadratischen Zusammenhangs zwischen X und Y mit einem standardnormalverteilten ($\mathcal{N}(0,1)$) Fehlerterm zu Grunde. Hier fällt in dem betrachteten Intervall $[1, 10]$ für X die Nichtlinearität der gegebenen Abhängigkeit bei der Berechnung von $\rho_n(X, Y)$ nicht besonders ins Gewicht, es ergibt sich der hohe Wert $\rho_n(X, Y) = 0,9723$.
Geht man nun zu einem kubischen Zusammenhang $Y = X^3 + \varepsilon, \varepsilon \sim \mathcal{N}(0,1)$ über, so ergibt sich das Streubild in Abbildung 8. Aufgrund der stärkeren Abweichung vom linearen Zusammenhang sinkt der Wert der Produktmomentkorrelation auf $\rho_n(X, Y) = 0,9280$. In Abbildung 9 ist nun schließlich die Beziehung $Y = \exp(X) + \varepsilon, \varepsilon \sim \mathcal{N}(0,1)$ modelliert

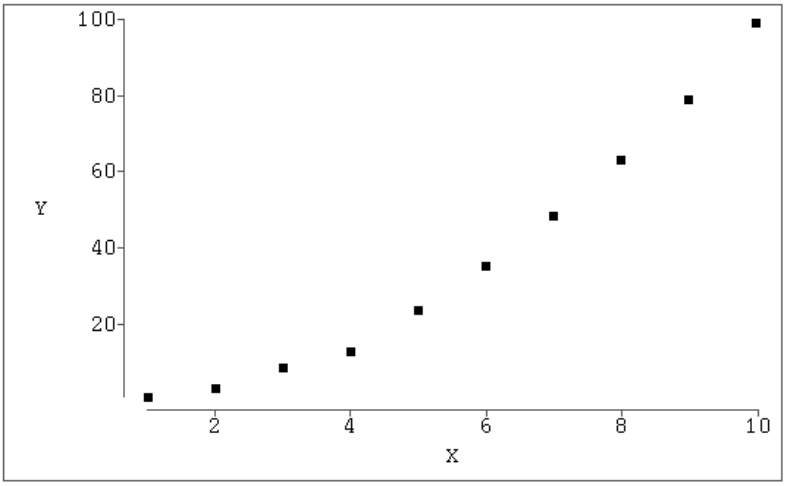


Abbildung 7: Quadratischer Zusammenhang zwischen zwei Variablen.

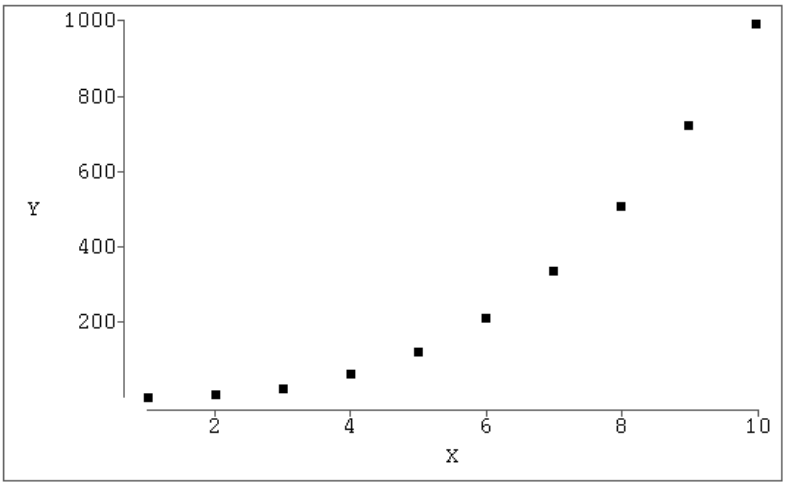


Abbildung 8: Kubischer Zusammenhang zwischen zwei Variablen.

worden. Diese Abhängigkeit ist von $\rho_n(X, Y)$ nur noch schlecht zu erfassen, der berechnete Wert ergibt sich zu $\rho_n(X, Y) = 0,7169$. Sicherlich wäre es jedoch von Interesse, in allen drei Fällen auf die Abhängigkeit zwischen X und Y aufmerksam zu werden. Setzt man nun, wie oben beschrieben, anstelle der tatsächlichen Simulationswerte die zugehörigen Ränge ein, ergibt sich in allen drei Fällen ein identisches Streubild (siehe Abbildung 10): Die Punktewolke entspricht der ersten Winkelhalbierenden, da mit wachsenden Werten für X auch in Y-Richtung immer größere Messwerte beobachtet wurden. Dabei spielt es keine Rolle, in welcher Form das Wachstum in Y-Richtung mit dem in X-Richtung gekoppelt ist. Der Rangkorrelationskoeffizient nach Spearman, hier mit $\rho_n^S(X, Y)$ bezeichnet, errechnet

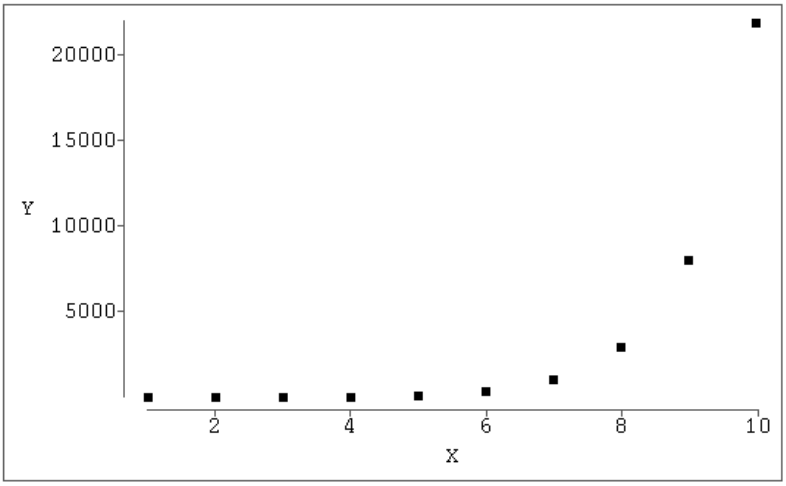


Abbildung 9: Exponentieller Zusammenhang zwischen zwei Variablen.

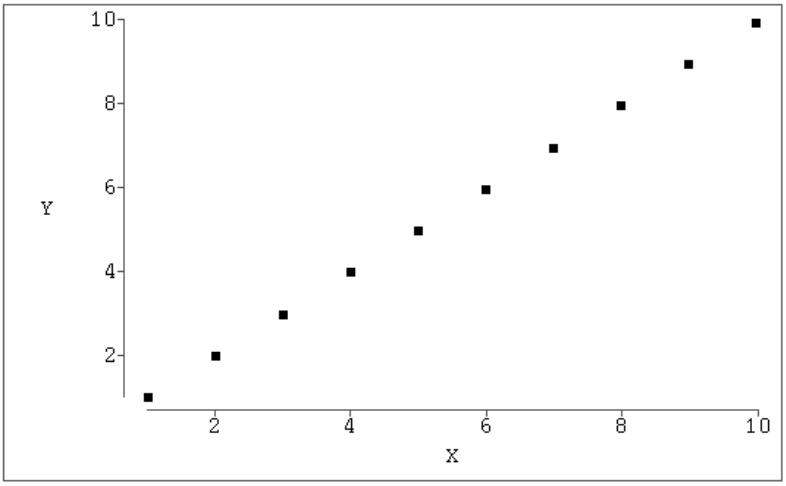


Abbildung 10: Streubild der Ränge zu Abbildungen 7 bis 9.

sich damit in allen drei Modellbeispielen zu $\rho_n^S(X,Y) = 1$.

5.2 Theorie

Um eine Vereinfachung der Notation zu erreichen, wird in diesem Abschnitt die empirische Varianz $\mathrm{Var}_n(X)$ einer Zufallsgröße X auf der Basis von n Realisierungen $(x_1, \ldots, x_n)$ definiert als

$$\mathrm{Var}_n(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}_n)^2 \quad \text{mit} \quad \bar{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i \quad .$$

Es wird also - abweichend von der Festsetzung in Abschnitt 2 - der Normierungsfaktor $\frac{1}{n}$ anstelle von $\frac{1}{n-1}$ verwendet.
Der Rang r_k einer Beobachtung x_k aus einer Stichprobe $(x_1, \ldots, x_n)$ vom Umfang n mit $x_i \neq x_j \ \forall i \neq j$ ist definiert als die Position von x_k in der zugehörigen geordneten Stichprobe, also

$$r_k = \sum_{j=1}^{n} I_{\{x_j \leq x_k\}} \quad .$$

Für das arithmetische Mittel und die empirische Varianz eines Rangvektors $R = (r_1, \ldots, r_n)$ der Beobachtungen $(x_1, \ldots, x_n)$ einer Stichprobe vom Umfang n mit $x_i \neq x_j \ \forall i \neq j$ gelten

$$\bar{r}_n = \frac{n+1}{2} \quad , \quad \mathrm{Var}_n(R) = \frac{n^2-1}{12} \quad ,$$

wegen

$$\bar{r}_n = \ \frac{1}{n}\sum_{i=1}^{n} i \ = \ \frac{1}{n} \cdot \frac{n \cdot (n+1)}{2} \ = \ \frac{n+1}{2}$$

und

$$\begin{aligned}
\mathrm{Var}_n(R) &= \frac{1}{n}\sum_{i=1}^{n}(i-\bar{r}_n)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(i - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\left[\sum_{i=1}^{n} i^2 - \sum_{i=1}^{n}(n+1)\cdot i + \sum_{i=1}^{n}\left(\frac{n+1}{2}\right)^2\right] \\
&= \frac{1}{n}\left[\frac{n\cdot(n+1)\cdot(2n+1)}{6} - \frac{n\cdot(n+1)^2}{2} + \frac{n\cdot(n+1)^2}{4}\right] \\
&= \frac{1}{n}\left[\frac{n\cdot(n+1)\cdot(2n+1)}{6} - \frac{n\cdot(n+1)^2}{4}\right] \\
&= \frac{1}{n}\left[\frac{2n\cdot(n+1)\cdot(2n+1) - 3n\cdot(n+1)^2}{12}\right] \\
&= \frac{1}{n}\left[\frac{4n^3+2n^2+4n^2+2n-3n^3-6n^2-3n}{12}\right] \\
&= \frac{1}{n}\left(\frac{n\cdot(n^2-1)}{12}\right) = \frac{n^2-1}{12} \quad .
\end{aligned}$$

Die Definition des Ranges setzt voraus, dass alle Messwerte unterschiedlich sind. Da dies jedoch nicht immer der Fall ist, sind Überlegungen nötig, die die Behandlung mehrfach auftretender Werte (Ties) möglich machen. Zur Vereinfachung der Darstellung nehmen wir an, dass für die der Größe nach geordneten Beobachtungen $x_{i:n}$ der betrachteten Stichprobe $x_{1:n} = x_{2:n} = \ldots = x_{d_1:n}$ gilt. Dann ist es sicherlich nicht sinnvoll, alle Ränge von 1 bis d_1 zu vergeben, denn damit würde die Information der Gleichheit von $x_{1:n}, \ldots, x_{d_1:n}$ verlorengehen. Ebenfalls sollte der Rang 1 nicht d_1-mal gewählt werden, denn dies würde einen sehr großen Abstand der Werte $x_{1:n}, \ldots, x_{d_1:n}$ zu $x_{d_1+1:n}$ erzeugen, der in Wirklichkeit nicht gegeben sein muss. Ein analoges Argument gilt selbstverständlich für die Festsetzung, dass alle Werte $x_{1:n}, \ldots, x_{d_1:n}$ den Rang d_1 erhalten. Vielmehr wird man dazu übergehen, den Beobachtungen $x_{1:n}, \ldots, x_{d_1:n}$ den mittleren Rang $\frac{d_1+1}{2}$ zuzuweisen. Allgemein geschieht die Zuordnung der Ränge im Falle des Vorhandenseins von Ties wie folgt: Gegeben sei eine Stichprobe $(x_1, \ldots, x_n)$ vom Umfang n mit p unterschiedlichen Werten v_i, also $|\{x_i : 1 \leq i \leq n\}| = p$. Der Wert $v_i, i = 1, \ldots, p$ mit $v_1 < v_2 < \ldots < v_p$ trete dabei d_i-mal auf. Der mittlere Rang r_k^* einer Beobachtung x_k aus einer solchen Stichprobe ist definiert als

$$r_k^* = \sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \quad \forall k : x_k = v_i\,,\ i = 1, \ldots, p \quad .$$

Für das arithmetische Mittel des Vektors der mittleren Ränge $R^* = (r_1^*, \ldots, r_n^*)$ gilt

$$\overline{r^*}_n = \frac{1}{n} \sum_{k=1}^{n} r_k^* = \frac{n+1}{2} \quad .$$

Zum Beweis seien k_1 und k_2 zwei natürliche Zahlen mit $k_2 > k_1$. Dann gilt:

$$\begin{aligned}
(k_2 - k_1) \cdot \left[k_1 + \frac{1}{2} \cdot (k_2 - k_1 + 1) \right] &= \frac{(k_2 - k_1) \cdot (2k_1 + k_2 - k_1 + 1)}{2} \\
&= \frac{(k_2 - k_1) \cdot (k_1 + k_2 + 1)}{2} \\
&= \frac{k_1 k_2 + k_2(k_2+1) - k_1(k_1+1) - k_1 k_2}{2} \\
&= \frac{k_2(k_2+1)}{2} - \frac{k_1(k_1+1)}{2} \\
&= \sum_{i=1}^{k_2} i - \sum_{i=1}^{k_1} i \\
&= \sum_{i=k_1+1}^{k_2} i \quad .
\end{aligned}$$

Wählen wir nun speziell $k_1 = \sum_{j=1}^{i-1} d_j$ und $k_2 = \sum_{j=1}^{i} d_j$, dann gilt offensichtlich $k_2 - k_1 = d_i$ und damit

$$\begin{aligned}
\sum_{k=1}^{n} r_k^* &= \sum_{i=1}^{p} \left[d_i \cdot \left(\sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \right) \right] \\
&= \sum_{i=1}^{p} \sum_{l=\sum_{j=1}^{i-1} d_j + 1}^{\sum_{j=1}^{i} d_j} l \\
&= \sum_{i=1}^{n} i \\
&= \frac{n \cdot (n+1)}{2} \quad .
\end{aligned}$$

Hieraus folgt, dass das arithmetische Mittel der (mittleren) Ränge einer Stichprobe vom Umfang n konstant also insbesondere invariant bezüglich des Vorhandenseins beliebig gearteter Ties ist. Für die empirische Varianz gilt diese Aussage hingegen nicht. Es ist zwar möglich, eine geschlossene Formel anzugeben, diese beinhaltet jedoch einen Korrekturterm, in welchen die Informationen über die vorhandenen verbundenen Werte eingehen. Es gilt

$$\begin{aligned}
\mathrm{Var}_n(R^*) &= \frac{1}{n} \sum_{k=1}^{n} \left(r_k^* - \frac{n+1}{2} \right)^2 \\
&= \frac{n^2 - 1}{12} - \frac{\sum_{i=1}^{p} d_i (d_i^2 - 1)}{12n} \quad . \qquad (6)
\end{aligned}$$

Eine Herleitung von (6) findet sich unter anderem in [13] auf S. 329f. Hier wird sie im Folgenden durch elementares Nachrechnen gezeigt. Dazu treffen wir zur Vereinfachung der Schreibweise folgende Definitionen:

$$\begin{aligned}
s_{R^*}^2 &:= \mathrm{Var}_n(R^*) \\
s_i &:= \sum_{j=1}^{i} d_j \\
a_i &:= s_{i-1} + \frac{1}{2} \cdot (d_i + 1) \quad .
\end{aligned}$$

Mit diesen Vereinbarungen folgt

$$\begin{aligned}
s_{R^*}^2 &= \frac{1}{n}\sum_{k=1}^{n}\left(r_k^* - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p} d_i \cdot \left(a_i - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left(\left(l - \frac{n+1}{2}\right) - (l-a_i)\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[\left(l - \frac{n+1}{2}\right)^2 + (l-a_i)^2 - 2\cdot\left(l - \frac{n+1}{2}\right)\cdot(l-a_i)\right].
\end{aligned}$$

Ein Aufteilen der Summen führt zu

$$\begin{aligned}
s_{R^*}^2 &= \frac{1}{n}\sum_{k=1}^{n}\left(k - \frac{n+1}{2}\right)^2 \\
&\quad + \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[(l-a_i)^2 - 2\cdot\left(l - a_i + a_i - \frac{n+1}{2}\right)\cdot(l-a_i)\right] \\
&= \frac{n^2-1}{12} + \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[-(l-a_i)^2 - 2\cdot\left(a_i - \frac{n+1}{2}\right)\cdot(l-a_i)\right] \\
&= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}(l-a_i)^2 - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\sum_{l=s_{i-1}+1}^{s_i}(l-a_i)\right].
\end{aligned}$$

Sei $l' := l - s_{i-1}$, dann gilt

$$\begin{aligned}
s_{R^*}^2 &= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\sum_{l'=1}^{d_i}\left(l' - \frac{d_i+1}{2}\right)^2 \\
&\quad - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\sum_{l'=1}^{d_i}\left(l' - \frac{d_i+1}{2}\right)\right] \\
&= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\frac{d_i\cdot(d_i^2-1)}{12} \\
&\quad - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\left(\frac{d_i\cdot(d_i+1)}{2} - \frac{d_i\cdot(d_i+1)}{2}\right)\right] \\
&= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{p} d_i\cdot(d_i^2-1)}{12n}.
\end{aligned}$$

Um nun die gegebene Aufgabe, die Rangkorrelation zweier Stichproben $(x_1, \ldots, x_n)$ sowie $(y_1, \ldots, y_n)$ mit zugehörigen mittleren Rängen $(r_1^*, r_2^*, \ldots, r_n^*)$ bzw. $(s_1^*, s_2^*, \ldots, s_n^*)$ zu berechnen, die überdies jeweils Ties an unterschiedlichen Stellen aufweisen können, kann man sich der Formel für die empirische Varianz der Differenz zweier Zufallsgrößen bedienen:

$$\begin{aligned} & \mathrm{Var}_n(X-Y) &=& \mathrm{Var}_n(X) + \mathrm{Var}_n(Y) - 2\mathrm{Kov}_n(X,Y) \\ \iff \quad & \mathrm{Kov}_n(X,Y) &=& \frac{1}{2}(\mathrm{Var}_n(X) + \mathrm{Var}_n(Y) - \mathrm{Var}_n(X-Y)) \end{aligned} \tag{7}$$

Die (empirische) Varianz von $Z := R^* - S^*$ ist aber in dem Fall, dass die Realisierungen von R^* und S^* die Vektoren der (mittleren) Ränge sind, einfach zu berechnen, denn es gilt $\bar{r_n^*} = \bar{s_n^*} \equiv \frac{n+1}{2}$ und damit $\bar{z}_n = \bar{r_n^*} - \bar{s_n^*} = 0$. Es folgt

$$\mathrm{Var}_n(Z) = \frac{1}{n}\sum_{i=1}^{n}(z_i - \bar{z}_n)^2 = \frac{1}{n}\sum_{i=1}^{n} z_i^2 - \bar{z}_n^2 = \frac{1}{n}\sum_{i=1}^{n}(r_i^* - s_i^*)^2 \quad .$$

Setzt man dieses Ergebnis in (7) für die Kovarianz bezüglich der Rangvektoren ein, so ergibt sich mit $D^* := \sum_{i=1}^{n}(r_i^* - s_i^*)^2$

$$\mathrm{Kov}_n(R^*, S^*) = \frac{1}{2}\left(\mathrm{Var}_n(R^*) + \mathrm{Var}_n(S^*) - \frac{2D^*}{n}\right) \quad .$$

Schließlich definiert sich der Rangkorrelationskoeffizient $\rho_n^S(X,Y)$ nach Spearman als Produktmomentkorrelation der zu (X,Y) gehörenden Ranggrößen wie folgt:

Voraussetzungen:

1. Gegeben sind Realisierungen $(x_1, \ldots, x_n)$ und $(y_1, \ldots, y_n)$ zweier Zufallsvariablen X und Y.
2. Die Realisierungen von X weisen p unterschiedliche Werte v_i auf, also $|\{x_i : i \in \{1, \ldots, n\}\}| = p$. Der Wert $v_i, i = 1, \ldots, p$ mit $v_1 < v_2 < \ldots < v_p$ tritt dabei d_i-mal auf.
3. Die Realisierungen von Y weisen q unterschiedliche Werte w_i auf, also $|\{y_i : i \in \{1, \ldots, n\}\}| = q$. Der Wert $w_i, i = 1, \ldots, q$ mit $w_1 < w_2 < \ldots < w_q$ tritt dabei e_i-mal auf.
4. $R^* = (r_1^*, r_2^*, \ldots, r_n^*)$ bezeichnet den Vektor der mittleren Ränge von X.
5. $S^* = (s_1^*, s_2^*, \ldots, s_n^*)$ bezeichnet den Vektor der mittleren Ränge von Y.

Dann heißt

$$\rho_n^S(X,Y) := \rho_n(R^*,S^*) = \frac{\frac{1}{2}(\text{Var}_n(R^*) + \text{Var}_n(S^*) - \frac{2D^*}{n})}{\sqrt{\text{Var}_n(R^*)} \cdot \sqrt{\text{Var}_n(S^*)}}$$

der Rangkorrelationskoeffizient nach Spearman von X und Y. Dabei gelten

$$\begin{aligned}
r_k^* &= \sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \;\; \forall k : x_k = v_i; \;\; i = 1, \ldots, p \quad , \\
s_k^* &= \sum_{j=1}^{i-1} e_j + \frac{1}{2} \cdot (e_i + 1) \;\; \forall k : y_k = w_i; \;\; i = 1, \ldots, q \quad , \\
\text{Var}_n(R^*) &= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{p} d_i(d_i^2-1)}{12n} \quad , \\
\text{Var}_n(S^*) &= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{q} e_i(e_i^2-1)}{12n} \quad , \\
D^* &= \sum_{i=1}^{n} (r_i^* - s_i^*)^2 \quad .
\end{aligned}$$

5.3 Anwendung

Neben der Berechnung der Produktmomentkorrelation bietet die während des Projekts entstandene Software auch die Möglichkeit, den Rangkorrelationskoeffizienten nach Spearman für je ein Variablenpaar zu berechnen. Dies bietet sich insbesondere in dem Fall an, dass eine Ausreißererkennung gemäß der in Abschnitt 3.3.3 vorgestellten Methode des Erkennens „echter“ Ausreißer vorgenommen werden soll. Hierbei werden die kritischen Werte für die Mahalanobis-Distanz, die zu dem Urteil „Datenpunkt ist Ausreißer“ führen, groß, und deswegen reduziert sich die Anzahl der als Ausreißer markierten und damit von der weiteren Analyse ausgenommenen Datenpunkte bei Verwendung des Korrelationskoeffizienten nach Pearson zum Teil erheblich. Dies kann in Einzelfällen dazu führen, dass signifikante Korrelationen unerkannt bleiben bzw. nicht durch Elimination von Ausreißern aufgedeckt werden. Zudem wird im Falle der Verwendung kleiner kritischer Werte für die Mahalanobis-Distanz bei einem nichtlinearen Zusammenhang zwischen zwei betrachteten Variablen ein Großteil der Randbereiche der Daten durch die Ausreißerelimination entfernt, und es erfolgt so eine Konzentration auf die Kernbereiche des Datenmaterials, in denen dann in der Regel eine stark ausgeprägte lineare Beziehung besteht. Werden hingegen große kritische Werte verwendet, so ergibt sich dieser Effekt nicht, und es kann passieren, dass signifikante Zusammenhänge verloren gehen. Betrachtet man in diesen Fällen hingegen die Rangkorrelation nach Spearman, so geht anstelle der absoluten Distanz der Messwerte vom Datenzentrum nur deren Ordnung in die Berechnung ein, und die Gefahr, Ausreißer zu „vorsichtig“ zu entfernen, ist damit deutlich geringer.

6 Hauptkomponentenanalyse

6.1 Motivation

„The central idea of principal component analysis is to reduce the dimensionality of a data set in which there are a large number of interrelated variables, while retaining as much as possible of the variation present in the data set. This reduction is achieved by transforming to a new set of variables, the principal components, which are uncorrelated, and which are ordered so that the first few retain most of the variation present in all of the original variables.“

Dieses Zitat aus der Einleitung zu dem Buch von I. T. Jolliffe [12] beschreibt die Ziele der Hauptkomponentenanalyse (principal component analysis). Der m-dimensionale Datenraum, der durch die gemeinsam (multivariat) zu analysierenden Zufallsvariablen X_1, $X_2, \ldots, X_m$ aufgespannt wird, soll durch einen p-dimensionalen Datenraum mit $p \ll m$ ersetzt werden. Hierbei soll jedoch ein möglichst großer Anteil der Information, die die Originaldaten liefern, erhalten bleiben. Deswegen macht man sich bei dem Konzept der Hauptkomponentenanalyse die Abhängigkeiten (Korrelationen) zwischen den Originalvariablen zu Nutze und bestimmt p geeignete Linearkombinationen der m ursprünglichen Variablen mit möglichst großer Varianz, die dann unkorreliert sind. Geometrisch ausgedrückt bedeutet dies, dass die Streuung (Variation) in den Originaldaten in orthogonale Streurichtungen zerlegt wird, wobei die Streuintensität entlang dieser Richtungen immer weiter abnimmt. Bei diesem Vorgehen wird davon ausgegangen, dass die Gesamtstreuung von $p \ll m$ Richtungen hinreichend gut erfasst wird, also in den verbleibenden $(m-p)$ Richtungen eine nahezu konstante lineare Beziehung zwischen den Originalvariablen besteht. Die durch die oben genannten Linearkombinationen entstehenden neuen p Zufallsvariablen $C_1, C_2, \ldots, C_p$ nennt man die ersten p Hauptkomponenten der zu Grunde liegenden m-dimensionalen Verteilung.

6.2 Theoretischer Hintergrund

Um die in Abschnitt 6.1 postulierten Eigenschaften der Hauptkomponenten mathematisch zu fassen, bezeichnen nun $v_i, i = 1, \ldots, p$ mit $v_i \in \mathbb{R}^m \;\; \forall\, i = 1, \ldots, p$ die Vektoren, die die Koeffizienten der zu den ersten p Hauptkomponenten gehörenden Linearkombinationen enthalten. Diese v_i und damit die Hauptkomponenten $C_1, C_2, \ldots, C_p$ werden durch folgende Optimalitätsforderungen festgelegt:

P1: $v_i^\top \cdot v_j = \delta_{i,j} \;\; \forall i, j = 1, \ldots, p \quad ,$

P2: $(v_1, \ldots, v_k) = \underset{A \in \mathbb{R}^{m \times k}}{\operatorname{argmax}} \operatorname{tr}(A^\top \cdot K \cdot A) \quad \forall k = 1, \ldots, p \quad$ mit $K \in \mathbb{R}^{m \times m}$

Varianz- / Kovarianzmatrix der Originalvariablen $X_1, \ldots, X_m$.

Die erste Forderung entspricht der Orthogonalität der entstehenden Streurichtungen sowie einer Normierung der Vektoren v_i bezüglich der L_2-Norm, und die zweite Forderung formalisiert die sukzessive Bestimmung der Richtungen mit maximaler Streuung, denn $K^* := A^\top \cdot K \cdot A$ ist gerade die Varianz- / Kovarianzmatrix der Linearkombination $A^\top \cdot (X_1, \ldots, X_m)^\top$ (Basiswechsel im k-dimensionalen Teilraum).
Um die v_i zu berechnen, setzen wir zunächst $p = 1$. Damit vereinfachen sich P1 und P2 zu

$$v_1 = \underset{a \in \mathbb{R}^m}{\operatorname{argmax}}\, a^\top \cdot K \cdot a \quad \text{mit } \| a \|_2 = 1 \quad . \tag{8}$$

Wendet man das Verfahren der Lagrange-Multiplikatoren zur Lösung von Maximierungsproblemen unter Nebenbedingungen an (siehe z.B. [3], Kapitel 3.5), so ergibt sich als Lagrange-Funktion der Ausdruck

$$\mathcal{L}(a) = a^\top \cdot K \cdot a - \lambda_1 \cdot (a^\top \cdot a - 1), \quad \lambda_1 \in \mathbb{R} \quad .$$

Partielles Differenzieren nach den Komponenten von a führt auf das Gleichungssystem

$$\begin{aligned} & K \cdot a - \lambda_1 \cdot a &&= 0 \\ \iff \qquad & (K - \lambda_1 \cdot E) \cdot a &&= 0 \quad . \end{aligned}$$

Dieses letzte Gleichungssystem entspricht gerade dem Eigenwertproblem der Varianz- / Kovarianzmatrix K, so dass sich v_1 als ein Eigenvektor von K ergibt. Um nun zu entscheiden, welcher Eigenvektor das Maximierungsproblem löst, beachte man, dass

$$v_1^\top \cdot K \cdot v_1 \;=\; v_1^\top \cdot \lambda_1 \cdot v_1 \;=\; \lambda_1 \cdot v_1^\top \cdot v_1 \;=\; \lambda_1$$

gilt. Also muss λ_1 der größte Eigenwert von K und v_1 der zugehörige Eigenvektor sein. Ferner gilt

$$\text{Var}(C_1) \;=\; \text{Var}(v_1^\top \cdot (X_1, \ldots, X_m)^\top) \;=\; v_1^\top \cdot K \cdot v_1 \;=\; \lambda_1 \quad .$$

Es ergibt sich aufgrund der Orthonormalitätsforderung an die v_i, dass sich die Maximierung der Spur von K^* entkoppelt in p Maximierungsprobleme der Form (8) (siehe dazu auch [12], Seiten 9f.), so dass sich die folgenden Ergebnisse festhalten lassen:

1. v_i sind Eigenvektoren zu den Eigenwerten λ_i von K und es gilt $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$,
2. $\text{Var}(C_i) = \lambda_i, \;\; i = 1, \ldots, m$.

Führt man eine Hauptkomponentenanalyse für standardisierte Zufallsgrößen durch, so geht die Varianz- / Kovarianzmatrix K in die zugehörige Korrelationsmatrix R über. Es gilt

dann $\mathrm{Var}(X_i) = 1 \ \ \forall i = 1, \ldots, m$ und damit $\mathrm{tr}(R) = m$. Will man also bei der Hauptkomponentenanalyse eine implizite Standardisierung bzw. Maßstabsvereinheitlichung der zu analysierenden Variablen erzielen, so kann anstelle der Varianz- / Kovarianzmatrix die Korrelationsmatrix zur Bestimmung der Eigenwerte und Eigenvektoren zur Definition der Hauptkomponenten herangezogen werden. Die so berechneten $(\tilde{\lambda}_i, \tilde{v}_i)$ weichen im Allgemeinen von denen ab, die sich im Falle der Verwendung der Varianz- / Kovarianzmatrix ergeben.
Die im Rahmen dieses Projektes entstandene Implementierung verwendet Korrelationsmatrizen bei der Hauptkomponentenanalyse, wobei in der konkreten Berechnung natürlich wieder zu den entsprechenden empirischen Größen R_n übergegangen wird.

6.3 Anwendung

In praktischen Anwendungsfällen der Hauptkomponentenanalyse besteht die grundlegende Fragestellung offensichtlich darin, wie groß der Parameter p zu wählen ist. Dies wird davon abhängen, welchen Informationsverlust man hinzunehmen bereit ist. Ein häufig gemachter Vorschlag dazu besteht darin, p so zu wählen, dass im Falle der Verwendung von Korrelationsmatrizen $p = \max\{1 \leq i \leq m : \lambda_i \geq 1\}$ gilt. Anschaulich bedeutet diese Wahl von p, dass die einzubeziehenden Hauptkomponenten jeweils den Informationsgehalt mindestens einer Originalvariablen haben. In einigen Situationen kann diese Wahl von p jedoch dazu führen, dass einzelne Variablen komplett unberücksichtigt bleiben und deswegen empfiehlt I.T. Jolliffe in [12] die Wahl von $p = \max\{1 \leq i \leq m : \lambda_i \geq 0,7\}$, wenn möglichst sichergestellt werden soll, dass alle Variablen anteilig in den Hauptkomponenten Berücksichtigung finden.
Ein anderer Ansatz besteht darin, den Anteil der durch die Hauptkomponenten $C_1, \ldots, C_p$ erklärten Streuung, also $\sum_{i=1}^{p} \lambda_i$ mit $\lambda_1 \geq \ldots \geq \lambda_p$ Eigenwerte der Korrelationsmatrix $R \in \mathbb{R}^{m \times m}$ von $(X_1, \ldots, X_m)$ und ihr Verhältnis zu $\mathrm{tr}(R) = m$ (es werden hier standardisierte Variablen X_i betrachtet) zum Entscheidungskriterium für die Größe von p zu machen. In dem Fall, dass sich einzelne Variablen annähernd mit Hauptkomponenten identifizieren lassen, also nur sehr gering mit den restlichen Zufallsgrößen korrelieren, wird es jedoch auch hier dazu kommen, dass diese Variablen durch die ersten Hauptkomponenten fast gar nicht erfasst werden. Da dies eventuell unerwünscht ist, ist zusätzlich eine Information darüber nötig, wie groß im Fall der am schlechtesten erfassten Variable deren Restvarianz ist. Die Restvarianz bezeichnet den Anteil der Varianz der Variable X_i, der nicht durch die Hauptkomponenten $C_1, \ldots C_p$ erklärt wird, also:

$$\mathrm{Var}_{\mathrm{Rest},p}(X_i) = \min_{(a_1,\ldots,a_p)^\top \in \mathbb{R}^p} \mathrm{Var}\left(X_i - \sum_{j=1}^{p} a_j \cdot C_j \right) \quad . \tag{9}$$

Es ergibt sich:

$$\begin{aligned}
&\text{Var}\left(X_i - \sum_{j=1}^{p} a_j \cdot C_j\right) \\
&= \text{Var}(X_i) - 2 \cdot \sum_{j=1}^{p} a_j \cdot \text{Kov}(X_i, C_j) + \sum_{j=1}^{p}\sum_{l=1}^{p} a_j \cdot a_l \cdot \text{Kov}(C_j, C_l) \\
&= \text{Var}(X_i) + \sum_{j=1}^{p} a_j^2 \cdot \text{Var}(C_j) - 2 \cdot \sum_{j=1}^{p} a_j \cdot \sqrt{\text{Var}(C_j)} \cdot \frac{\text{Kov}(X_i, C_j)}{\sqrt{\text{Var}(C_j)}} \\
&\quad + \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} - \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} \qquad (\text{Kov}(C_j, C_l) = 0 \; \forall j \neq l) \\
&= \text{Var}(X_i) + \sum_{j=1}^{p} \left(a_j \cdot \sqrt{\text{Var}(C_j)} - \frac{\text{Kov}(X_i, C_j)}{\sqrt{\text{Var}(C_j)}}\right)^2 - \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} \\
&= \text{Var}(X_i) \cdot \left(1 - \sum_{j=1}^{p} \rho^2(X_i, C_j)\right) + \sum_{j=1}^{p} \text{Var}(C_j) \cdot \left(a_j - \frac{\text{Kov}(X_i, C_j)}{\text{Var}(C_j)}\right)^2 \quad .
\end{aligned}$$

Aus dieser Rechnung folgt, dass die Koeffizienten a_j, welche (9) minimieren, gerade die Regressionskoeffizienten

$$a_j = \frac{\text{Kov}(X_i, C_j)}{\text{Var}(C_j)}, \; \forall j = 1, \ldots, p$$

sind und die gesuchte Restvarianz damit von der Form

$$\text{Var}_{\text{Rest},p}(X_i) = \text{Var}(X_i) \cdot \left(1 - \sum_{j=1}^{p} \rho^2(X_i, C_j)\right)$$

ist. Um schließlich noch $\rho(X_i, C_j)$ zu berechnen, bezeichne $V \in \mathbb{R}^{m \times m}$ die Matrix der im L_2-Sinne normierten Eigenvektoren v_j $(j = 1, \ldots, n)$ von K, die nach der Größe der zugehörigen Eigenwerte absteigend geordnet sind, so dass sich die Hauptkomponenten $C_1, \ldots, C_m$ als $C_j = v_j^\top \cdot (X_1, \ldots, X_m)^\top \; \forall j = 1, \ldots, m$ ergeben. Dann gilt:

$$K \;=\; V \cdot \text{diag}(\lambda_i) \cdot V^\top$$

und man erhält

$$\text{Kov}(X_i, X_k) \;=\; \sum_{j=1}^{m} v_{i,j} \cdot \lambda_j \cdot v_{k,j}$$

bzw.

$$\begin{aligned}
\text{Kov}(X_i, C_j) &= \text{Kov}(X_i, \sum_{k=1}^{m} v_{k,j} \cdot X_k) \\
&= \sum_{k=1}^{m} \text{Kov}(X_i, X_k) \cdot v_{k,j} \\
&= \sum_{k=1}^{m} v_{k,j} \cdot \left(\sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot v_{k,l} \right) \\
&= \sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot \left(\sum_{k=1}^{m} v_{k,j} v_{k,l} \right) \\
&= \sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot \delta_{l,j} \\
&= v_{i,j} \cdot \lambda_j
\end{aligned}$$

und damit (unter Beachtung von $\mathrm{Var}(C_j) = \lambda_j$):

$$\rho(X_i, C_j) = \frac{v_{i,j} \cdot \sqrt{\lambda_j}}{\sqrt{\text{Var}(X_i)}} \quad .$$

Um also dem Benutzer eine möglichst gute Entscheidungshilfe für die Wahl von p an die Hand zu geben, wurde in der erfolgten Implementierung die Tabelle 2 eingebunden.

i	λ_i	$\lambda_{i-1} - \lambda_i$	λ_i / m	$\sum_{j=1}^{i} \lambda_j / m$	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},i}(X_k)$
1	λ_1	-	λ_1 / m	λ_1 / m	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},1}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
p	λ_p	$\lambda_{p-1} - \lambda_p$	λ_p / m	$\sum_{j=1}^{p} \lambda_j / m$	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	λ_m	$\lambda_{m-1} - \lambda_m$	λ_m / m	1	0

Tabelle 2: Schematische Darstellung von Ergebnissen bei der Hauptkomponentenanalyse.

Für die Wahl von p können insbesondere die letzten beiden Spalten dieser Tabelle zu Rate gezogen werden. Überschreitet der Wert in der vorletzten bzw. unterschreitet der Wert in der letzten Spalte in einer Zeile i einen Schwellenwert, so ist dieses i als die Anzahl p zu berücksichtigender Hauptkomponenten zu wählen.

6.4 Ausreißererkennung mit Hauptkomponenten

Wie im Falle der Betrachtung der Originalvariablen kann es auch beim Übergang zu den Hauptkomponenten von Interesse sein, Beobachtungen zu erkennen, die in dem (jetzt durch die Hauptkomponenten aufgespannten) Datenraum auffällig aus dem Gros der Daten herausfallen. Innerhalb des Projekts sind drei Ausprägungen solcher „Auffälligkeiten“ untersucht worden:

- Es kann Datenpunkte geben, die bezüglich der Verteilung einer bestimmten Hauptkomponente signifikant weit vom Datenzentrum entfernt liegen.
- Ein in Koordinaten des durch die Hauptkomponenten $(C_1, \ldots, C_p)$ gegebenen Koordinatensystems ausgedrückter Beobachtungsvektor kann einen signifikant großen Abstand zum Datenzentrum im $\mathbb{R}^p$ haben.
- Im Allgemeinen wird, wie in Abschnitt 6.1 beschrieben, davon ausgegangen, dass aus m Komponenten bestehende Beobachtungen in dem durch die ersten p Hauptkomponenten gegebenen p-dimensionalen Teildatenraum hinreichend gut beschrieben werden. Es kann jedoch Beobachtungen geben, für welche diese Annahme nicht zutrifft.

Datenpunkte, bei welchen eine solche Situation vorliegt, werden hier wieder als Ausreißer bezeichnet. Um diese Ausreißer erkennen zu können, sind Teststatistiken mit zugehörigen Verlässlichkeitsschranken für die obigen Problemstellungen nötig. In den folgenden Abschnitten wird das hierzu gewählte Vorgehen erläutert.

Dazu gelten vorab:

1. Die Ausgangsdaten bestehen aus den Beobachtungsvektoren $x_i = (x_{i,1}, \ldots, x_{i,m})^\top$, $i = 1, \ldots, n$. Ein solcher Beobachtungsvektor enthält die Messwerte aus den Laborexperimenten, die mit einer chemischen Struktur durchgeführt wurden. Zusammengefasst werden diese Beobachtungsvektoren in der $(n \times m)$-Datenmatrix $X = (x_{i,j})$, wobei n die Anzahl der Beobachtungen (Strukturen) und m die Anzahl der Variablen (Experimente) bezeichnet. Die Zeilen von X sind also die transponierten Beobachtungsvektoren x_i. Zudem sind die Spaltenvektoren von X - die jeweils ein Laborexperiment widerspiegeln - auf Mittelwert 0 und Varianz 1 standardisiert.
2. Mit $V \in \mathbb{R}^{m \times m}$ wird die Matrix der im L_2-Sinne normierten Eigenvektoren der Korrelationsmatrix der Originalvariablen bezeichnet. Diese sind nach der Größe der zugehörigen Eigenwerte absteigend geordnet, so dass sich die Hauptkomponenten $C_1, \ldots, C_m$ als $C_j = v_j^\top \cdot (X_1, \ldots, X_m)^\top \ \forall j = 1, \ldots, m$ ergeben.
3. Die Matrix der auf $C_1, \ldots, C_m$ bezogenen Koordinaten der Beobachtungen (im Folgenden auch Matrix der Scores der Beobachtungen genannt) wird mit $Y = X \cdot V$, $Y \in \mathbb{R}^{n \times m}$, bezeichnet. Auch im Falle der Scores liegen die Beobachtungen wieder als Zeilenvektoren vor.

6.4.1 Eindimensionale Ausreißer

Bei der Bestimmung von Ausreißern bezüglich einer speziellen Hauptkomponente C_j wurden zunächst die Laborexperimente bzw. die durch sie induzierten Zufallsvariablen als normalverteilt vorausgesetzt. Diese Annahme wurde in Ermangelung von Informationen über die tatsächlichen Verteilungen getroffen und muss daher im Einzelfall nicht zwingend korrekt sein. Die k-te Spalte von $X, X^{(k)}, k = 1, \ldots, m$, wird also als Vektor von n unabhängigen und identisch $\mathcal{N}(0,1)$-verteilten Zufallsgrößen interpretiert. Damit ergibt sich die mit einer Hauptkomponente C_j assoziierte Spalte $Y^{(j)}, j = 1, \ldots, m$ der Score-Matrix Y ebenfalls als ein Vektor von n normalverteilten Größen, denn sie entsteht durch eine Linearkombination von $\mathcal{N}$(0, 1)-verteilten Variablen. Zudem sind die einzelnen Komponenten von $Y^{(j)}$ wieder unabhängig, denn in die Linearkombination gehen nur Werte der jeweils gleichen Beobachtung ein. Wie in Abschnitt 6.2 hergeleitet, ergibt sich die Varianz der Größen in einer Spalte $Y^{(j)}$ als Wert des zugehörigen Eigenwertes λ_j der zu Grunde liegenden Korrelationsmatrix. Es gelten

$$y_{i,j} \text{ i.i.d.} \sim \mathcal{N}(0, \lambda_j) \quad \forall i = 1, \ldots, n;\ j \in \{1, \ldots, m\}$$

und

$$z_{i,j} := \frac{y_{i,j}}{\sqrt{\lambda_j}} \text{ i.i.d.} \sim \mathcal{N}(0,1) \quad \forall i = 1, \ldots, n;\ j \in \{1, \ldots, m\} \quad .$$

Damit lassen sich die normierten Scores $z_{i,j}$ als Teststatistiken zur Ausreißererkennung verwenden. In Anwendungen ist jedoch zu beachten, dass anstelle der exakten Korrelationsmatrizen die entsprechenden Schätzungen verwendet werden müssen und die hier gemachten Verteilungsaussagen dann nur noch asymptotisch gelten. Wird nun analog zu der Vorgehensweise in Abschnitt 3.3.3 wieder gefordert, dass nur in $\alpha \cdot 100\%$ der Fälle Beobachtungen als Ausreißer deklariert werden, wenn keine tatsächlichen Ausreißer vorliegen, so ergibt sich asymptotisch (Φ bezeichnet die Verteilungsfunktion der $\mathcal{N}(0,1)$-Verteilung):

$$\begin{aligned}
& & \mathbb{P}(\max_{1 \le i \le n} |z_{i,j}| > c) &\stackrel{!}{=} \alpha \\
\iff & & \mathbb{P}(|z_{i,j}| \le c \,\forall i = 1, \ldots, n) &= 1 - \alpha \\
\iff & & \mathbb{P}(-c \le z_{i,j} \le c \,\forall i = 1, \ldots, n) &= 1 - \alpha \\
\underset{\text{Unabhängigkeit}}{\iff} & & \textstyle\prod_{i=1}^{n} \mathbb{P}(-c \le z_{i,j} \le c) &= 1 - \alpha \\
\underset{\text{Verteilungsidentität}}{\iff} & & (\mathbb{P}(-c \le Z \le c))^n &= 1 - \alpha \text{ mit } Z \sim \mathcal{N}(0,1) \\
\iff & & (\Phi(c) - \Phi(-c))^n &= 1 - \alpha \\
\iff & & (\Phi(c) - (1 - \Phi(c)))^n &= 1 - \alpha \\
\iff & & 2 \cdot \Phi(c) - 1 &= \sqrt[n]{(1-\alpha)} \\
\iff & & \Phi(c) &= \tfrac{1}{2} \cdot (1 + \sqrt[n]{(1-\alpha)}) \\
\iff & & c &= \Phi^{-1}\left(\frac{1 + \sqrt[n]{(1-\alpha)}}{2}\right) \quad .
\end{aligned}$$

Überschreitet $|z_{i,j}| = |y_{i,j}/\sqrt{\lambda_j}|$ für eine Beobachtung i in Spalte j der Score-Matrix Y also das Quantil $c = \Phi^{-1}\left(\frac{1+\sqrt[n]{(1-\alpha)}}{2}\right)$ der Standard-Normalverteilung, so wird der zugehörige Datenpunkt als eindimensionaler Ausreißer zur Hauptkomponente C_j angesehen.

6.4.2 *p*-dimensionale Ausreißer

Betrachtet man Ausreißer in dem Raum, der durch die ersten p Hauptkomponenten aufgespannt wird, so bietet es sich aus Gründen der Maßstabsvereinheitlichung wieder an, die normierten Scores $z_{i,j}$ für das Testproblem zu verwenden. Berechnet man den quadrierten p-dimensionalen euklidischen Abstand

$$d_p^2(z_i, 0) = \sum_{k=1}^{p} \left(\frac{y_{i,k}}{\sqrt{\lambda_k}}\right)^2$$

der normierten Scores einer Beobachtung i zu ihrem Mittelwert $0 \in \mathbb{R}^p$, so ist dieser Ausdruck eine Summe von Quadraten asymptotisch standardnormalverteilter Zufallsgrößen, und somit lässt sich die asymptotische Verteilung von $d_p^2(z_i, 0)$ als χ^2-Verteilung mit p Freiheitsgraden (kurz: χ_p^2-Verteilung) angeben. Damit ergibt sich zur Bestimmung des kritischen Wertes für $d_p^2(z_i, 0)$ analog zur obigen Berechnung, wenn wieder mit Konfidenzwahrscheinlichkeit α nur „echte“ Ausreißer erkannt werden sollen:

$$\begin{aligned}
& \mathbb{P}(\max_{1\le i\le n} d_p^2(z_i,0) > c) \overset{!}{=} \alpha \\
\iff \quad & \mathbb{P}(d_p^2(z_i,0) \le c \;\forall i = 1,\dots,n) = 1-\alpha \\
\underset{\text{Unabhängigkeit}}{\iff} \quad & \textstyle\prod_{i=1}^{n} \mathbb{P}(d_p^2(z_i,0) \le c) = 1-\alpha \\
\underset{\text{Verteilungsidentität}}{\iff} \quad & (\mathbb{P}(Z \le c))^n = 1-\alpha \text{ mit } Z \sim \chi_p^2 \\
\iff \quad & (F_{\chi_p^2}(c))^n = 1-\alpha \\
\iff \quad & F_{\chi_p^2}(c) = \sqrt[n]{(1-\alpha)} \\
\iff \quad & c = \chi^2_{p;\,\sqrt[n]{(1-\alpha)}} \quad .
\end{aligned}$$

Als kritischer Wert für $d_p^2(z_i, 0)$ wird also hier das $\sqrt[n]{(1-\alpha)}$-Quantil der χ^2-Verteilung mit p Freiheitsgraden gewählt und Punkten, deren quadrierter p-dimensionaler Euklidischer Abstand vom Nullpunkt diesen Wert c überschreitet, das Attribut „Ausreißer“ zugeordnet.

6.4.3 Schlecht modellierte Beobachtungen

Als letztes werden in diesem Abschnitt die Beobachtungen daraufhin untersucht, inwiefern sie durch p Hauptkomponenten schlecht modelliert werden. Hierzu beachte man zunächst, dass die Matrix V orthogonal und daher invertierbar mit $V^{-1} = V^\top$ ist. Damit ist es möglich, die Originalvariablen, also die Beobachtungsmatrix X, als Funktion der Scores zu schreiben:

$$Y = X \cdot V \underset{\text{V orthogonal}}{\Longleftrightarrow} X = Y \cdot V^\top \quad \text{mit } V \in \mathbb{R}^{m\times m} \text{ und } Y \in \mathbb{R}^{n\times m} \quad .$$

Verwendet man anstelle der vollständigen Matrizen $V \in \mathbb{R}^{m\times m}$ und $Y \in \mathbb{R}^{n\times m}$ nur jeweils deren erste p Spalten, so geht die rechte Seite der obigen Äquivalenz in die Gleichung

$$\hat{X} = \tilde{Y} \cdot \tilde{V}^\top \quad \text{mit } \tilde{V} \in \mathbb{R}^{m\times p} \text{ und } \tilde{Y} \in \mathbb{R}^{n\times p}$$

über, wobei $\hat{X}$ die Matrix der Projektion von X auf den durch die ersten p Hauptkomponenten $(C_1, \ldots, C_p)$ aufgespannten p-dimensionalen Teilraum des $\mathbb{R}^m$ ist.
Will man nun messen, wie gut ein Beobachtungsvektor $x_i, i = 1, \ldots, n$ durch das p-dimensionale Modell beschrieben wird, so bietet sich die Verwendung der Länge des Residuenvektors $x_i - \hat{x}_i$ an. Deswegen wird in [10] in Abschnitt 2.7.2 die dort als Q-Statistik bezeichnete Größe

$$Q_i = (x_i - \hat{x}_i)^\top \cdot (x_i - \hat{x}_i) \quad \text{mit } x_i \in \mathbb{R}^m \text{ Beobachtungsvektor}$$

zur Behandlung der erwähnten Fragestellung eingeführt. Aus Berechnungssicht ist es günstig, dass sich Q_i auch schreiben lässt als

$$Q_i = \sum_{k=p+1}^{m} y_{i,k}^2,$$

wobei y_i den Score-Vektor zum Beobachtungsvektor x_i bezeichnet.
Die Verteilung der so definierten Teststatistik Q_i lässt sich nur mühsam herleiten. In die Berechnung gehen im Wesentlichen die Summen der ersten drei Potenzen der $(m - p)$ kleinsten Eigenwerte der Korrelationsmatrix ein. In [10] wird das Ergebnis dieser Berechnung angegeben und es finden sich weitere Literaturhinweise für dieses Problem. Es ergibt sich, dass eine geschickte Transformation von Q_i zu einer möglichst guten Approximation der Standardnormalverteilung führt, so dass entsprechend transformierte Quantile der $\mathcal{N}(0, 1)$-Verteilungsfunktion als Schranken für Q_i verwendet werden können. Der in der Folge angegebene kritische Wert Q_α für die Teststatistik Q_i wurde der Arbeit von J. Edward Jackson [10] entnommen.

Setzt man

$$\begin{aligned}
\theta_1 &:= \sum_{k=p+1}^{m} \lambda_k \quad , \quad \theta_2 := \sum_{k=p+1}^{m} \lambda_k^2 \quad , \quad \theta_3 := \sum_{k=p+1}^{m} \lambda_k^3 \quad , \\
h_0 &:= 1 - \frac{2 \cdot \theta_1 \cdot \theta_3}{3 \cdot \theta_2^2} \quad , \\
c_\alpha &:= \text{sgn}(h_0) \cdot \Phi^{-1}(\sqrt[n]{1-\alpha}) \quad , \\
Q_\alpha &:= \theta_1 \cdot \left[\frac{c_\alpha \cdot \sqrt{2 \cdot \theta_2 \cdot h_0^2}}{\theta_1} + \frac{\theta_2 \cdot h_0 (h_0 - 1)}{\theta_1^2} + 1 \right]^{\frac{1}{h_0}} \quad ,
\end{aligned}$$

so erhält man

$$\mathbb{P}(Q_i > Q_\alpha) \underset{\text{approx.}}{=} \alpha \quad .$$

Will man die Beobachtungsvektoren $x_i, i = 1, \ldots, n$ daraufhin untersuchen, ob sie signifikant schlecht durch das p-dimensionale lineare Modell der ersten p Hauptkomponenten beschrieben werden, so gilt es zu überprüfen, ob der Wert von

$$Q_i = (x_i - \hat{x}_i)^\top \cdot (x_i - \hat{x}_i) = \sum_{k=p+1}^{m} y_{i,k}^2$$

das zugehörige transformierte $\mathcal{N}(0, 1)$-Quantil überschreitet.

7 Variablenselektion

7.1 Motivation

Im vorangegangenen Abschnitt wurde mit der Hauptkomponentenanalyse eine Methode vorgestellt, die Dimensionalität des zu untersuchenden Datenraumes zu reduzieren und dabei nur einen möglichst geringen Informationsverlust hinnehmen zu müssen. Dies ist zum Beispiel im Hinblick auf die Berechnungs- und Speicherplatzkomplexität von auf das Datenmaterial anzuwendenden Algorithmen günstig, denn damit ist es möglich, diese nur auf dem durch einige Hauptkomponenten gegebenen, niederdimensionalen Datenraum auszuführen und die gelieferten Ergebnisse dann eventuell so zu transformieren, dass wieder Informationen über die Ausgangsdaten gewonnen werden. Verfolgt man jedoch das Ziel, den Datenraum für einen Analysten überschaubar zu machen oder ist man daran interessiert, einige Variablen ganz aus der Betrachtung auszuklammern (also gegebenenfalls einige Laborexperimente gar nicht mehr durchführen zu müssen), so ergeben sich bei der Verwendung von Hauptkomponenten für diese Aufgaben zwei grundlegende Schwierigkeiten:

- Hauptkomponenten sind als Linearkombinationen der Originalvariablen wesentlich schlechter interpretierbar als die Ausgangsdaten, welche unmittelbar die experimentell erhobenen Messwerte wiedergeben.
- In die Hauptkomponenten gehen in der Regel alle Variablen mit einem festgelegten Gewichtsfaktor ein, es kann also kein Experiment fortgelassen werden.

Aus diesen Gründen ist es oftmals günstiger, anstelle von p Variablenkombinationen mit maximalem Informationsgehalt lieber p der m Originalvariablen selbst auszuwählen, die am repräsentativsten für das gesamte Datenmaterial sind, um dadurch die Dimension des zu betrachtenden Datenraums zu verringern. Dieses Vorgehen der Variablenselektion wird in diesem Abschnitt behandelt, und es werden Algorithmen erläutert, die verschiedene Auswahlkriterien für die zu selektierenden Variablen definieren, wobei in der Implementierung zwei generelle Vorgehensweisen umgesetzt worden sind:

1. Verfahren, die sich der Ergebnisse der Hauptkomponentenanalyse bedienen und
2. das Verfahren der Haupt-Variablen (principal variables) nach McCabe.

7.2 Auf Hauptkomponenten basierende Verfahren

Die von I.T. Jolliffe in [11] publizierten und in der GALA-Software implementierten, auf Hauptkomponenten basierenden Verfahren zur Variablenselektion arbeiten derart, dass ausgewählte Spalten der in Abschnitt 6.3 definierten Matrix V, welche die Eigenvektoren der der Hauptkomponentenanalyse zu Grunde gelegten Varianz- / Kovarianzmatrix bzw. Korrelationsmatrix enthält, auf ihre Maximaleinträge hin untersucht werden. Weist ein Spaltenvektor y_j von Y seinen größten Eintrag in Zeile i auf, wobei $1 \leq i, j \leq m$ gilt, so wird

(a) die Variable X_i ausgewählt, falls j einen kleinen Wert hat und X_i demnach mit einem großen Gewicht in die zugehörige Hauptkomponente C_j mit $j \ll m$ eingeht (Selektion).

(b) die Variable X_i als zu vernachlässigen angesehen und aus der Menge der zu untersuchenden Variablen entfernt, falls j nahe bei m liegt (Elimination).

Damit nun genau p Variablen bestimmt werden können, wird das Verfahren der Selektion oder der Elimination iterativ wiederholt, bis genau p Variablen selektiert bzw. genau $m-p$ Variablen eliminiert worden sind. Dabei ist es zum einen möglich, nach jedem Selektions- bzw. Eliminationsschritt eine neue Hauptkomponentenanalyse für die verbleibenden Variablen durchzuführen oder aber die Ergebnisse einer einzigen Hauptkomponentenanalyse für den gesamten Selektions- bzw. Eliminationsvorgang zu nutzen. Entscheidet man sich dazu, die p Variablen über das Verfahren der Selektion zu bestimmen und zudem für jeden Selektionsschritt eine neue Hauptkomponentenanalyse zu tätigen, so gilt es zu beachten, dass die Variablen in der Regel untereinander korreliert sind. Ist jedoch die Variable X_i einmal ausgewählt worden, so soll für die weiteren Auswahlschritte ihr Einfluss auf die

restlichen noch verbliebenen Variablen ausgeschaltet werden. Deswegen gilt es zu untersuchen, wie sich die partielle Varianz- / Kovarianzmatrix $K_{m-1,k}$ berechnet. $K_{m-1,k}$ beinhaltet hierbei die Varianzen/Kovarianzen der $m-1$ verbleibenden Variablen, wenn man die Variable X_k aus der Menge der Variablen $(X_1, \ldots, X_m)$ entfernt und ihren Einfluss auf die restlichen Variablen „herausrechnet".
Nehmen wir zur Lösung dieses Problems an, es soll die partielle Kovarianz zweier Zufallsvariablen X und Y berechnet werden, wobei der Einfluss einer dritten Zufallsvariable Z, die sowohl mit X als auch mit Y korreliert ist, ausgeschaltet werden soll. Dazu definieren wir:

$$X_Z := a_{x,z} + b_{x,z} \cdot Z \quad , \quad Y_Z := a_{y,z} + b_{y,z} \cdot Z$$

mit den Regressionskoeffizienten

$$b_{x,z} = \frac{\text{Kov}(X,Z)}{\text{Var}(Z)} \; , \quad b_{y,z} = \frac{\text{Kov}(Y,Z)}{\text{Var}(Z)} \quad .$$

X_Z bzw. Y_Z bezeichnen den Anteil der Variablen X bzw. Y, der durch ihren linearen Bezug zu Z gegeben ist. Es gilt

$$\begin{aligned}
&\text{Kov}((X - X_Z), (Y - Y_Z)) \\
&= \text{Kov}((X - (a_{x,z} + b_{x,z} \cdot Z)), (Y - (a_{y,z} + b_{y,z} \cdot Z))) \\
&= \text{Kov}(X,Y) - \text{Kov}(X, a_{y,z} + b_{y,z} \cdot Z) - \text{Kov}(Y, a_{x,z} + b_{x,z} \cdot Z) \\
&\quad + \text{Kov}(a_{x,z} + b_{x,z} \cdot Z, a_{y,z} + b_{y,z} \cdot Z) \\
&= \text{Kov}(X,Y) - b_{y,z} \cdot \text{Kov}(X,Z) - b_{x,z} \cdot \text{Kov}(Y,Z) + b_{x,z} \cdot b_{y,z} \cdot \text{Var}(Z) \quad .
\end{aligned}$$

Beachtet man die Definitionen von $b_{x,z}$ und $b_{y,z}$, so vereinfacht sich der Ausdruck weiter:

$$\begin{aligned}
&\text{Kov}((X - X_Z), (Y - Y_Z)) \\
&= \text{Kov}(X,Y) - 2 \cdot \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} + \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} \\
&= \text{Kov}(X,Y) - \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} \quad .
\end{aligned}$$

Es gilt also für die partielle Varianz- / Kovarianzmatrix:

$$K_{m-1,k} = K'_{m,k} - \left(\frac{1}{\text{Var}(X_k)} \cdot K^{(k)} \cdot K^{(k)^\top} \right)'_{m,k}$$

mit

$K_{m-1,k}$	partielle Varianz- / Kovarianzmatrix ohne Einfluss von X_k,
$K^{(k)}$	k-te Spalte der Matrix K,
$M'_{m,k}$	Streichungsmatrix, die aus einer $(m \times m)$-Matrix M durch Streichung von Zeile k und Spalte k hervorgeht.

Aus Sicht des Algorithmus ist hierbei natürlich zu beachten, dass die Indizes der verbleibenden Variablen verschoben werden, wenn $k \neq m$ gilt. In den folgenden Abschnitten wird nun kurz für die vier auf Hauptkomponenten basierenden Verfahren, die im Rahmen dieses Projekts umgesetzt worden sind, das jeweils zugehörige Rekursionsschema angegeben.

7.2.1 Selektion mit *p* Hauptkomponentenanalysen

Zunächst wird eine Hauptkomponentenanalyse der Originalvariablen $(X_1, \ldots, X_m)$ auf Basis der Varianz- / Kovarianzmatrix K bzw. (standardisierte Größen) der Korrelationsmatrix R durchgeführt, um die Matrix V der Eigenvektoren zu erhalten. Zudem wird die Anzahl p auszuwählender Variablen vorgegeben. Die selektierten Variablen $(X_{i_1}, \ldots, X_{i_p})$ werden nun dadurch festgelegt, dass zunächst

$$i_1 = \underset{1 \leq i \leq m}{\operatorname{argmax}}\, v_{i,1}$$

gewählt wird. Die Variable X_{i_1} ist hiermit ausgewählt und, wie oben beschrieben, soll ihr Einfluss auf die restlichen Variablen für den nächsten Selektionsschritt ausgeschaltet werden. Es wird also die partielle Varianz- / Kovarianzmatrix K_{m-1,i_1} der verbleibenden $m-1$ Variablen gebildet und auf ihrer Basis eine erneute Hauptkomponentenanalyse durchgeführt. Nun wird wieder der Maximaleintrag des zum größten Eigenwert gehörenden Eigenvektors von K_{m-1,i_1} gesucht, die entsprechende Zeile mit i_2 bezeichnet und die Variable X_{i_2} dem Satz ausgewählter Variablen hinzugefügt. Dies wiederholt sich solange, bis p selektierte Variablen feststehen.

7.2.2 Selektion mit genau einer Hauptkomponentenanalyse

Will man den nicht unerheblichen Berechnungsaufwand, der sich durch die p zu tätigenden Hauptkomponentenanalysen der in Abschnitt 7.2.1 vorgestellten Methode ergibt, vermeiden, so kann die Selektion von p Variablen auch auf der Basis einer einzigen Hauptkomponentenanalyse erfolgen. Zunächst ergibt sich die Matrix $V = (v_{i,j})$ der Eigenvektoren analog zu dem Vorgehen in Abschnitt 7.2.1. Dann werden jedoch alle Spaltenvektoren $v_j,\ j = 1, \ldots, p$ von V auf ihren Maximaleintrag hin untersucht und die Indizes $i_1, \ldots, i_p$ der selektierten Variablen ergeben sich zu:

$$\begin{aligned}
i_1 &= \underset{1 \leq i \leq m}{\operatorname{argmax}}\, v_{i,1}\ , \\
i_2 &= \underset{1 \leq i \leq m : i \neq i_1}{\operatorname{argmax}}\, v_{i,2}\ , \\
&\vdots \qquad\quad \vdots \\
i_p &= \underset{1 \leq i \leq m : i \notin \{i_1, \ldots, i_{p-1}\}}{\operatorname{argmax}}\, v_{i,p}\quad .
\end{aligned}$$

Hier ist also die Berechnung der partiellen Varianz- / Kovarianzmatrizen vom Algorithmus her nicht gefordert. Für die Darstellung der Ergebnisse (vgl. Abschnitt 7.5) sind jedoch die Restvarianzen erforderlich, so dass auch hier die entsprechenden Größen berechnet wurden.

7.2.3 Elimination mit *(m-p)* Hauptkomponentenanalysen

Diesem Verfahren liegt die umgekehrte Argumentationsrichtung gegenüber dem Verfahren aus Abschnitt 7.2.1 zu Grunde. Es werden nicht die Variablen, die stark in die ersten Hauptkomponenten eingehen, ausgewählt, sondern diejenigen, die sich am besten mit den letzten Hauptkomponenten identifizieren lassen, eliminiert. Hier werden also Indizes $(i_1, \ldots, i_{m-p})$ bestimmt, so dass die Variablen $(X_{i_1}, \ldots, X_{i_{m-p}})$ vernachlässigt und die verbleibenden Variablen mit einem Index i^*, für den $i^* \notin \{i_1, \ldots, i_{m-p}\}$ gilt, als ausgewählt angesehen werden. Es wird also i_1 als

$$i_1 = \underset{1 \leq i \leq m}{\operatorname{argmax}} \, v_{i,m}$$

festgelegt. Danach wird die Varianz- / Kovarianzmatrix K'_{m,i_1} der verbleibenden $m-1$ Variablen, die sich aus der ursprünglichen Varianz- / Kovarianzmatrix K durch Streichung von Zeile i_1 und Spalte i_1 ergibt, für eine neue Hauptkomponentenanalyse benutzt und es werden iterativ die restlichen $m-p-1$ Indizes bestimmt, deren zugehörige Variablen eliminiert werden sollen. Es ist anzumerken, dass in dem Regelfall $p \ll m$ dieses Verfahren den bei weitem größten Rechenaufwand aller hier diskutierten Methoden verursacht.

7.2.4 Elimination mit genau einer Hauptkomponentenanalyse

Das vierte angewendete Verfahren verwirklicht die Grundidee des vorangegangenen, wobei jedoch auf eine erneute Hauptkomponentenanalyse pro Eliminationsschritt verzichtet wird. Die Indizes $(i_1, \ldots, i_{m-p})$ berechnen sich damit in Analogie zu der in Abschnitt 7.2.2 gewählten Vorgehensweise zu:

$$\begin{aligned}
i_1 &= \underset{1 \leq i \leq m}{\operatorname{argmax}} \, v_{i,m} \\
i_2 &= \underset{1 \leq i \leq m : i \neq i_1}{\operatorname{argmax}} \, v_{i,m-1} \\
&\vdots \qquad\qquad \vdots \\
i_{m-p} &= \underset{1 \leq i \leq m : i \notin \{i_1, \ldots, i_{m-p-1}\}}{\operatorname{argmax}} \, v_{i,p+1} \quad .
\end{aligned}$$

7.3 Verfahren der Principal Variables

Das von McCabe in [15] beschriebene Verfahren der Haupt-Variablen (in Anlehnung an den Begriff der Hauptkomponenten) stützt sich nicht auf die Ergebnisse einer Hauptkomponentenanalyse, sondern versucht, die Optimalitätseigenschaft der Hauptkomponenten direkt auf einzelne Variablen zu übertragen. Diese Optimalitätseigenschaft besteht (vgl. hierzu Abschnitt 6.2) darin, dass die erste Hauptkomponente C_1 eine möglichst große Varianz besitzt und damit die Summe der Restvarianzen minimiert. Wie in Abschnitt 6.4 beschrieben, lässt sich der Anteil der durch eine Hauptkomponente C_j erklärten Varianz einer Variable X_i ausdrücken durch $\mathrm{Var}(X_i) \cdot \rho^2(X_i, C_j)$. Setzt man nun anstelle von C_j eine Originalvariable X_j ein und fordert, dass die Summe der Restvarianzen der Variablen $\{X_i : i \neq j\}$ minimiert oder invers formuliert die Summe der durch X_j erklärten Varianz der restlichen Variablen maximiert wird, so führt dies auf die folgende Maximierungsaufgabe:

$$\begin{aligned} j_1 &= \operatorname*{argmax}_{1\leq j\leq m} \sum_{i=1}^{m} \mathrm{Var}(X_i) \cdot \rho^2(X_i, X_j) \\ &= \operatorname*{argmax}_{1\leq j\leq m} \sum_{i=1}^{m} \frac{\mathrm{Kov}^2(X_i, X_j)}{\mathrm{Var}(X_j)} \\ &= \operatorname*{argmax}_{1\leq j\leq m} \frac{1}{\mathrm{Var}(X_j)} \cdot \sum_{i=1}^{m} \mathrm{Kov}^2(X_i, X_j) \quad . \end{aligned} \tag{10}$$

Die zu dem so bestimmten Index j_1 gehörende Variable X_{j_1} wird in diesem Verfahren als erste ausgewählt. Anschließend wird ihr Einfluss auf die restlichen Variablen wieder herausgerechnet, indem die Quadrate der Einträge der partiellen Varianz- / Kovarianzmatrix K_{m-1,j_1} in der Maximierungsaufgabe (10) betrachtet werden, und so bestimmen sich iterativ die übrigen Indizes $(j_2, \ldots, j_p)$ für die Variablenselektion. In der praktischen Verwendung bietet sich dieses Verfahren auf Grund seiner einfachen Implementierbarkeit und geringen Ressourceninanspruchnahme an. Da sich im Falle von Daten aus der pharmazeutischen Forschung gezeigt hat, dass die dabei erzielten Ergebnisse qualitativ nicht hinter denen zurückbleiben, welche sich bei Anwendung der in Abschnitt 7.2 vorgestellten, aufwändigeren Methoden ergeben, erweist sich das Verfahren von McCabe als überlegen im Kosten- / Nutzenvergleich.

7.4 Anwendung

In Analogie zu der Darstellung der Ergebnisse der Hauptkomponentenanalyse werden die Resultate der Variablenselektion, wie in Tabelle 3 gezeigt, wiedergegeben, wobei folgende Vereinbarungen getroffen werden:

j	Iteration
i_j	Nummer der in Iteration j ausgewählten Variable
η_j	durch X_{i_j} erklärte Varianz
$\text{Var}_{\text{Rest},j}(X_k)$	Restvarianz von Variable X_k nach Iteration j.

j	i_j	η_j	$\eta_{j-1} - \eta_j$	η_j/m	$\sum_{k=1}^{j} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},j}(X_k)$
1	i_1	η_1	-	η_1/m	η_1/m	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},1}(X_k)$
2	i_2	η_2	$\eta_1 - \eta_2$	η_2/m	$(\eta_1 + \eta_2)/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},2}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
p	i_p	η_p	$\eta_{p-1} - \eta_p$	η_p/m	$\sum_{k=1}^{p} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$

Tabelle 3: Schematische Darstellung von Ergebnissen bei der Variablenselektion.

7.5 Ergebnisse

Abschließend werden in Tabelle 4 anhand eines Beispieldatensatzes aus der pharmazeutischen Forschung mit $m = 100$ Variablen die Ergebnisse der implementierten Variablenselektionsverfahren gegenübergestellt.

Abschnitt	p	η_p	$\sum_{k=1}^{p} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$
7.2.1	22	0,761	0,903	0,638
7.2.1	45	0,085	0,995	0,046
7.2.2	24	0,727	0,905	0,507
7.2.2	66	0,0084	0,999	0,0497
7.2.3	29	0,881	0,9004	0,328
7.2.3	46	0,3496	0,991	0,0387
7.2.4	25	1,049	0,901	0,6939
7.2.4	53	0,1101	0,9984	0,0169
7.3	21	0,7963	0,9007	0,5983
7.3	44	0,0993	0,9932	0,0469

Tabelle 4: Ergebnisse verschiedener Variablenselektionsverfahren für einen Beispieldatensatz.

In der jeweils ersten Zeile zu jedem Verfahren wurde p so gewählt, dass der kumulierte

Anteil erklärter Streuung über $0,9$ liegt; die zweite Zeile gehört zu dem Wert von p, ab welchem jede Restvarianz den Schwellenwert von 5 % unterschreitet.

8 Robuste statistische Verfahren

8.1 Einleitung

Wie die vorangegangenen Abschnitte zeigen, kann man bei der Betrachtung des Datenmaterials zwei grundlegende Aufgabenbereiche unterscheiden. Zum einen sollte man Regelmäßigkeiten und Abhängigkeiten innerhalb des Datenmaterials erkennen können, zum anderen sollten Ausreißer identifiziert und ihr Einfluss auf die gewonnenen Informationen minimiert werden. Ausreißer können verschiedene Ursachen haben:

- Es wurden fehlerhafte Messungen durchgeführt.
- Einige wenige Beobachtungen besitzen überdurchschnittlich günstige oder ungünstige Eigenschaften.

Die bisher vorgestellten statistischen Verfahren zum Data Mining haben die Eigenschaft, dass ihre Ergebnisse sehr stark von Ausreißern beeinflusst werden und die so gewonnenen Informationen nur eine geringe bis keine Aussagekraft besitzen. An diesem Punkt angelangt, bieten sich zwei Lösungsmöglichkeiten an. Entweder nutzt man Verfahren zur Ausreißererkennung und eliminiert diese, bevor weitere Informationen gewonnen werden, oder es sollten Verfahren genutzt werden, die weitgehend unempfindlich gegenüber Ausreißern sind. Genau an dieser Stelle werden robuste Verfahren wirksam. Sie erlauben, Aussagen über das Datenmaterial zu treffen, die nicht von grenzwertigen Beobachtungen verfälscht werden. Im Verlauf des GALA-Projektes war es ein wichtiges Ziel, robuste Verfahren der Datenanalyse zu implementieren, damit sie wahlweise zu den klassischen Methoden genutzt werden können. Im Folgenden wird ein theoretischer Überblick über robuste Verfahren gegeben. Dabei wird auf univariate und multivariate Problemstellungen eingegangen, und es werden Kriterien vorgestellt, die eine qualitative Beurteilung dieser Verfahren ermöglichen.

8.2 Maßzahlen für Lage und Streuung einer Stichprobe im univariaten Fall

Hat man eine Beobachtungsreihe mit Datenmaterial vorliegen, so möchte man diese Daten meist mit wenigen charakteristischen Größen kennzeichnen. Häufig verwendete Charakteristika sind die Lage und die Streuung der Daten. Eine Lagemaßzahl beschreibt in geeigneter Weise das Zentrum der beobachteten Werte $x_1, ..., x_n$, wohingegen eine Streuungsmaßzahl angibt, wie die Daten von einem Zentrum im Mittel abweichen. Einige dieser Lage- und Streuungsmaßzahlen sollen im Folgenden vorgestellt werden.

8.2.1 Lagemaßzahlen

Aufgrund der leichten Berechenbarkeit erscheint das arithmetische Mittel als ein einfaches und leicht anzuwendendes Hilfsmittel, um das Zentrum einer Beobachtungsreihe zu beschreiben. Untersucht man das arithmetische Mittel hingegen auf seine Empfindlichkeit gegenüber Ausreißern, also gegenüber Werten, die sehr weit entfernt von diesem berechneten Zentrum liegen, so stellt man schnell fest, dass das arithmetische Mittel nicht das optimale Werkzeug zur Beschreibung der Lage ist. Das folgende Beispiel aus [7] soll dies verdeutlichen.
Von einer Holzlatte der Länge 5m werden nacheinander fünf Stücke der Länge 1m abgesägt. Anschließend werden die Abweichungen x_i von der Solllänge 1m gemessen und in mm angegeben, d.h. $x_i = (\xi_i - 1) \cdot 10^3$, wobei ξ_i die Länge des i-ten Stückes in m angibt. Dabei ergeben sich folgende Werte für $x_1, \ldots, x_5$:

$$-0,5;\ 1,5;\ -1,0;\ 0,5;\ -9,5 \quad .$$

Berechnet man das arithmetische Mittel auf der Grundlage aller fünf Beobachtungen, so ergibt sich $\overline{x}_5 = -1,8$. Dieser Durchschnittswert legt die Vermutung nahe, dass die Säge bei jedem abgesägten Stück einen Fehler von ca. $1,8$mm macht. Die ersten vier Lattenstücke wurden jedoch annähernd korrekt mit der gewünschten Länge von 1m gesägt. Da aber bei jedem Sägevorgang 2mm durch Sägespäne verloren gingen, kann das letzte Lattenstück keine Länge von 1m mehr haben und weist die angegebene relativ große Abweichung vom Sollwert auf. Nimmt man hingegen nur die ersten vier Werte zur Berechnung des Mittels, dann ergibt sich ein Wert für das arithmetische Mittel von $\overline{x}_4 = 0,125$. Dieser Wert zeigt offensichtlich eine adäquatere Beschreibung der Sachlage.
Bei diesem einfachen Beispiel war es unproblematisch, den Ausreißerwert zu erkennen und für weitere Berechnungen zu eliminieren. In den allermeisten Fällen ist diese Identifikation jedoch nicht so schnell und sicher zu bewerkstelligen, und man sollte daher zu einer Lagemaßzahl übergehen, die nicht so empfindlich auf extreme Werte in den Rohdaten reagiert. Vorbereitend wird dazu zunächst der Begriff der geordneten Stichprobe eingeführt. Liegen n Beobachtungen $x_1, \ldots, x_n$ einer Stichprobe vom Umfang n vor und werden die Daten der Größe nach sortiert, so bezeichnet $x_{(i)}$ den i-kleinsten Wert, also insbesondere

$$x_{(1)} = \min_{1 \le i \le n} x_i, \qquad x_{(n)} = \max_{1 \le i \le n} x_i \quad .$$

Die der Größe nach sortierte Reihe

$$x_{(1)} \le x_{(2)} \le \ldots \le x_{(n-1)} \le x_{(n)}$$

heißt geordnete Stichprobe von $x_1, \ldots, x_n$, und $x_{(i)}$ wird die i-te Ordnungsgröße genannt. Damit lassen sich nun weitere Lagemaßzahlen angeben.
Das empirische α-Quantil $\tilde{x}_\alpha$ einer Stichprobe $x_1, ..., x_n$ vom Umfang n ist definiert als

$$\tilde{x}_\alpha = \begin{cases} x_{([n \cdot \alpha]+1)} & \text{, falls } n \cdot \alpha \text{ keine ganze Zahl ist} \\ \frac{1}{2}\left(x_{(n \cdot \alpha)} + x_{(n \cdot \alpha + 1)}\right) & \text{, falls } n \cdot \alpha \text{ eine ganze Zahl ist} \end{cases} .$$

Als Spezialfall stellt sich der empirische Median dar. Er ist gegeben durch

$$m_n := \tilde{x}_{0,5} = \begin{cases} x_{\left(\frac{n+1}{2}\right)} & \text{, falls } n \text{ ungerade} \\ \frac{1}{2}\left(x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n+2}{2}\right)}\right) & \text{, falls } n \text{ gerade} \end{cases} ,$$

wenn $x_1, ..., x_n$ eine Stichprobe vom Umfang n ist.
Betrachtet man nun noch einmal das Beispiel der Holzlatte und berechnet den empirischen Median, so erhält man als geordnete Stichprobe zunächst

$$x_{(1)} = -9,5;\ x_{(2)} = -1,0;\ x_{(3)} = -0,5;\ x_{(4)} = 0,5;\ x_{(5)} = 1,5 \quad .$$

Damit ergibt sich der Median zu $m_5 = \tilde{x}_{0,5} = -0,5$, welcher eine „robustere“ Beschreibung der Situation liefert als das arithmetische Mittel.
Bei symmetrisch verteilten Daten ist das midquartile ein geeignetes Lagemaß. Es wird definiert als

$$\frac{1}{2}\left(\tilde{x}_{0,25} + \tilde{x}_{0,75}\right) \quad .$$

Im Beispiel ergibt sich so

$$\frac{1}{2}\left(\tilde{x}_{0,25} + \tilde{x}_{0,75}\right) = \frac{1}{2}(x_{(2)} + x_{(4)}) = -0,25 \quad .$$

Diese kleine Auswahl an möglichen Lagemaßzahlen zeigt schon am simplen Beispiel, wie unterschiedlich auf Ausreißer reagiert wird, und es wird deutlich, dass man bei der Untersuchung experimentell oder auch pseudo-zufällig erzeugter Daten am Rechner auf Verfahren zurückgreifen sollte, die möglichst unempfindlich gegenüber Ausreißern sind.

8.2.2 Streuungsmaßzahlen

Das wohl bekannteste Verfahren zur Bestimmung der Streuung innerhalb einer Stichprobe ist die empirische Standardabweichung. Für eine Beobachtungsreihe $x_1, ..., x_n$ vom Stichprobenumfang n ist sie definiert als

$$s_n = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \overline{x}_n\right)^2} \quad .$$

Die quadrierte empirische Standardabweichung wird empirische Varianz genannt. Angewendet auf die Daten aus dem einleitenden Beispiel, erhält man hier für den arithmetischen Mittelwert $\overline{x}_5 = -1,8$ eine empirische Standardabweichung von $s_5 = 4,41$, bei $\overline{x}_4 = 0,125$ ergibt sich ein Wert von $s_4 = 1,11$. Die Werte des Beispiels zeigen, dass auch die empirische Standardabweichung empfindlich auf Ausreißerwerte reagiert. Dies lässt sich damit begründen, dass große, stark vom Zentrum abweichende Werte durch das

Quadrieren bei der Berechnung der empirischen Standardabweichung einen noch größeren Einfluss gewinnen und so die Streuung der Daten verfälscht wiedergeben. Auch die Verwendung einer ausreißerunempfindlicheren Lagemaßzahl kann dies nicht unterbinden, da das arithmetische Mittel $\bar{x}_n$ gerade der Wert ist, der den Ausdruck $\sum(x_i - a)^2$ für a minimiert.
Eine gegenüber Ausreißern robustere Streuungsmaßzahl ist der Interquartilabstand, mit dem man die Differenz der beiden Größen $\tilde{x}_{0,75}$ und $\tilde{x}_{0,25}$ bezeichnet, also

$$\text{IQR}_n = \tilde{x}_{0,75} - \tilde{x}_{0,25} \quad .$$

Auf den Begriff des Interquartilabstandes wird in Abschnitt 8.4.2 noch näher eingegangen. Für die Stichprobe des Beispiels ergibt sich $\text{IQR}_5 = 1,5$. Dieser Wert zeigt einen robusteren Umgang mit dem Ausreißerwert von $x_{(1)} = -9,5$ und ist daher hier besser geeignet, die Streuung der Daten um ein Lagezentrum zu beschreiben, als der der empirischen Standardabweichung.

8.3 Kriterien für Robustheit

Wie man in dem vorangegangenen Abschnitt an einfachen Beispielen bereits gesehen hat, reagieren unterschiedliche Verfahren zur Bestimmung von Lage und Streuung unterschiedlich auf Ausreißer. Was man in diesem Zusammenhang unter einem robusten Verfahren versteht, ist intuitiv klar, aber wie kann man diese Robustheit mathematisch beschreiben und überprüfen? Dieser Frage soll nun nachgegangen werden. Als Kriterien zur Beurteilung von Robustheit werden im Folgenden die Sensitivitätskurve, die Einflusskurve und der Bruchpunkt definiert.
Bezeichne im Weiteren

$$T_n = T_n(x_1, \ldots, x_n)$$

eine Stichprobenfunktion, auch Statistik genannt, zur Stichprobe $x_1, \ldots, x_n$ vom Umfang n. Beispiele für Stichprobenfunktionen sind das arithmetische Mittel

$$T_n(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} x_i \quad ,$$

der empirische Median

$$T_n(x_1, \ldots, x_n) = \begin{cases} x_{((n+1)/2)} & \text{, falls } n \text{ ungerade} \\ \frac{1}{2}(x_{(n/2)} + x_{((n+2)/2)}) & \text{, falls } n \text{ gerade} \end{cases}$$

und die empirische Standardabweichung

$$T_n(x_1, \ldots, x_n) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)^2} \quad .$$

8.3.1 Die Sensitivitätskurve

Sei $T_n = T_n(x_1, \ldots, x_n)$ eine Stichprobenfunktion basierend auf n Beobachtungen. Dann ist die Sensitivitätskurve definiert als

$$\mathrm{SC}(x; x_1, \ldots, x_{n-1}, T_n) = n \cdot (T_n(x_1, \ldots, x_{n-1}, x) - T_{n-1}(x_1, \ldots, x_{n-1})) \quad .$$

Die Sensitivitätskurve beschreibt die Reaktion einer Stichprobenfunktion, falls zu einer Stichprobe $x_1, \ldots, x_{n-1}$ eine weitere Beobachtung x hinzugefügt wird. Ihr Wert entspricht der Differenz von T_n und T_{n-1} multipliziert mit der Anzahl n der Beobachtungen insgesamt. Das folgende Beispiel veranschaulicht dies.

Die Daten

$$n = 5;\ x_1 = -0,5;\ x_2 = 1,5;\ x_3 = -1,0;\ x_4 = 0,5$$

ergeben:

1. Sensitivitätskurve des arithmetischen Mittels

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_4, T_5) &= 5 \cdot \left(\frac{1}{5}(x_1 + \ldots + x_4 + x) - \frac{1}{4}(x_1 + \ldots + x_4) \right) \\ &= 5 \cdot (0,1 + 0,2x - 0,125) \\ &= x - 0,125 \end{aligned}$$

2. Sensitivitätskurve des Medians

$$\mathrm{SC}(x; x_1, \ldots, x_4, T_5) = \begin{cases} -2,5 & \text{, falls } x < -0,5 \\ 5x & \text{, falls } -0,5 \leq x \leq 0,5 \\ 2,5 & \text{, falls } 0,5 < x \end{cases}$$

In Abbildung 11 zeigt sich am Steigungsverhalten der Kurven deutlich, dass die Hinzunahme eines weiteren beliebigen Wertes x den Wert der Sensitivitätskurve des arithmetischen Mittels und damit auch den Wert für das Zentrum der Datenpunkte so verändern kann, dass der neu berechnete Wert für die Lage der Daten ein stark verfälschtes Bild der Stichprobe liefert. Dies zeigt sich darin, dass die Sensitivitätskurve des arithmetischen Mittels unbeschränkt ist. Im Gegensatz dazu ist die Sensitivitätskurve des Medians beschränkt, so dass auch ein zusätzlicher weit außerhalb des Zentrums liegender Beobachtungswert die geschätzte Lage des Zentrums nicht maßgeblich verändern kann.

Bevor nun weitere Kriterien zur Beurteilung von Robustheitseigenschaften von Maßzahlen eingeführt werden, wird die bisherige Betrachtungsweise aus der Sicht der rein beschreibenden (deskriptiven) Statistik verlassen und das für das weitere Vorgehen zu Grunde liegende mathematische Modell vorgestellt [14]. Dabei bezeichne X eine Zufallsvariable

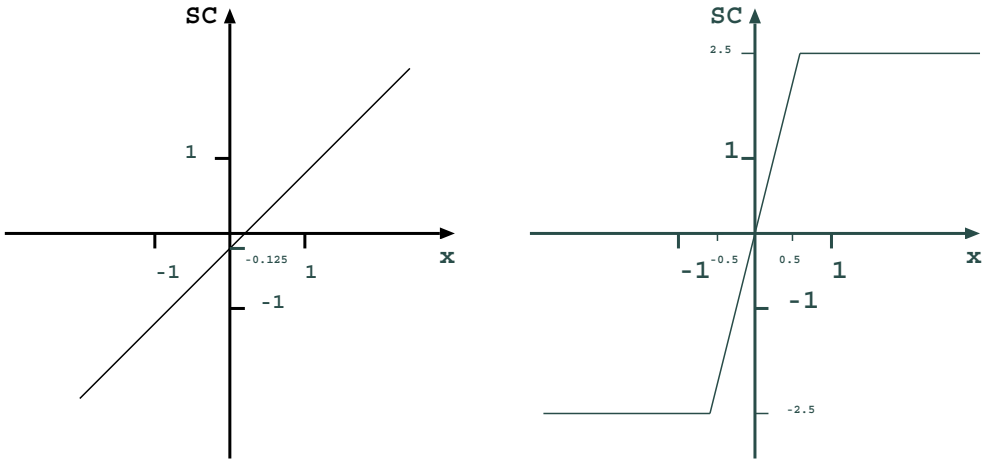


Abbildung 11: Sensitivitätskurven für das arithmetische Mittel (links) und den Median (rechts).

(Zufallsgröße), die auf einem geeigneten Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$ als eine Abbildung von Ω in die Menge der reellen Zahlen definiert ist, und x bezeichne im Folgenden eine Realisation einer Zufallsvariablen X. Des Weiteren sei

$$F : \mathbb{R} \to [0;1] \quad \text{mit} \quad F(x) := \mathbb{P}(X \leq x) \qquad \forall x \in \mathbb{R}$$

die Verteilungsfunktion der Zufallsvariablen X mit den bekannten Eigenschaften. Um sich bei unbekannter Verteilungsfunktion trotzdem ein Bild von der Verteilung der Zufallsgröße X machen zu können, ist die Konstruktion der empirischen Verteilungsfunktion F_n hilfreich. Für eine unabhängige Folge $X_1, \ldots, X_n$ von Zufallsgrößen mit der Verteilungsfunktion F gibt

$$F_n(t, \omega) := \frac{1}{n} \sum_{i=1}^{n} 1_{(-\infty;t]} \left(X_i(\omega) \right)$$

die relative Häufigkeit der X_i mit Werten $\leq t$ an, wobei mit $1_{(-\infty,t]}$ die Indikatorfunktion [1] des Intervalls $(-\infty, t]$ bezeichnet wird. Außerdem heißt eine Stichprobenfunktion, deren Wert einen unbekannten Parameter der den Beobachtungen zu Grunde liegenden Verteilung schätzt, ein Schätzer bzw. eine Schätzfunktion. Alle bisher betrachteten Stichprobenfunktionen sind Schätzer von Lage- bzw. Streuungsparametern der zu Grunde liegenden Verteilung.

8.3.2 Die Einflusskurve

Bei der Berechnung der Sensitivitätskurve stellt man fest, dass diese nicht ausschließlich von der jeweiligen Stichprobenfunktion $T_n(\cdot)$ abhängt, sondern vor allem auch von

[1] $1_A(\omega) = \begin{cases} 1 \text{ , falls } \omega \in A \\ 0 \text{ , falls } \omega \notin A \end{cases}$

den konkret vorliegenden Stichprobendaten und deren Anzahl. Man kann versuchen, diese Abhängigkeiten zu eliminieren, indem man eine Grenzwertbetrachtung für $n \to \infty$ durchführt.
Man betrachte als Beispiel die SC-Kurve des arithmetischen Mittels:

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_{n-1}, \bar{x}_n) &= n \cdot \left[\frac{1}{n} \left(\sum_{i=1}^{n-1} x_i + x \right) - \frac{1}{n-1} \sum_{i=1}^{n-1} x_i \right] \\ &= \sum_{i=1}^{n-1} x_i + x - \frac{n}{n-1} \sum_{i=1}^{n-1} x_i \\ &= -\frac{1}{n-1} \sum_{i=1}^{n-1} x_i + x \quad . \end{aligned}$$

Sind die Beobachtungen unabhängige Realisationen einer Zufallsgröße mit existierendem Erwartungswert μ, dann konvergiert aufgrund des Starken Gesetzes Großer Zahlen (SGGZ) die Sensitivitätskurve SC fast sicher [2] gegen $x - \mu$.

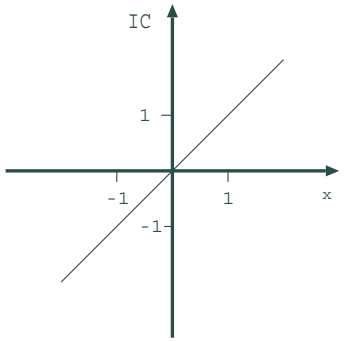


Abbildung 12: Einflusskurve des arithmetischen Mittels für $\mu = 0$.

Betrachtet man jetzt allgemein eine Stichprobenfunktion, die wie das arithmetische Mittel eine Funktion der empirischen Verteilungsfunktion F_n ist, also $T_n = T(F_n)$, dann ergibt sich für die SC-Kurve:

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_{n-1}, T_n) &= n \left[T\left(\frac{n-1}{n} F_{n-1} + \frac{1}{n} \delta_x \right) - T(F_{n-1}) \right] \\ &= \frac{T\left(\frac{n-1}{n} F_{n-1} + \frac{1}{n} \delta_x \right) - T(F_{n-1})}{\frac{1}{n}} \quad , \end{aligned}$$

wobei δ_x die sogenannte Einheitsmasse im Punkt x ist. Für $n \to \infty$ strebt F_{n-1} gegen F, und man erhält für ein hinreichend reguläres T, von dem man voraussetzt, dass es auch in

[2] Der Begriff der fast sicheren Konvergenz wird auf Seite 75 definiert.

F definiert ist, in vielen Fällen [6]:

$$\mathrm{SC}(x; X_1, \ldots, X_{n-1}, T_n) \xrightarrow{f.s.} \lim_{\varepsilon \to 0+} \frac{T((1-\varepsilon)F + \varepsilon\delta_x) - T(F)}{\varepsilon} \quad . \tag{11}$$

In diesem Fall ist $T(F_n)$ ein natürlicher Schätzwert für $T(F)$. Der Grenzwert in (11) wird als die Einflusskurve (influence curve) bezeichnet:

$$\mathrm{IC}(x; F, T) = \lim_{\varepsilon \to 0+} \frac{T((1-\varepsilon)F + \varepsilon\delta_x) - T(F)}{\varepsilon} \quad .$$

Die Einflusskurve gibt also Auskunft über die Reaktion einer Schätzfunktion auf einen hinzugefügten Wert unter der Annahme, dass die Stichprobe aus der zu Grunde liegenden Verteilung von großem Umfang ist. Da bei der Grenzwertbildung wegen des Starken Gesetzes Großer Zahlen die Stichprobe nicht mehr betrachtet wird, hängt die Einflusskurve nur noch von x ab und wird daher auch als lokale Kenngröße bezeichnet. Außerdem kann man den maximalen Einfluss, den eine einzelne Beobachtung auf den zu schätzenden Wert ausüben kann, mit dem maximalen Betrag der Einflusskurve angeben. Diese Größe wird als Ausreißerempfindlichkeit (Gross-Error Sensitivity) bezeichnet. Untersucht man das arithmetische Mittel und den Median hinsichtlich dieser Größe, so stellt man fest, dass das arithmetische Mittel im Gegensatz zum Median einen unendlichen Wert annehmen kann und deshalb bei ausreißerbehafteten Daten nicht geeignet ist. Auch die Steigung der Einflusskurve kann man im Bezug auf die Robustheit interpretieren. Sie liefert einen Anhaltspunkt für das Verhalten der Schätzfunktion bei nur geringen Änderungen, wie sie zum Beispiel bei Rundungen oder Klassenbildung der Rohdaten auftreten können. Solche Änderungen haben einen stärkeren Einfluss auf den zu schätzenden Wert, je steiler die Steigung der Einflusskurve ist. Je größer der maximale Wert des Absolutbetrages der Steigung (Local-Shift Sensitivity) ist, desto stärker können solche Änderungen auf den Schätzwert durchschlagen. Für den Median ergibt sich hier ein unendlicher Wert, während die Größe beim arithmetischen Mittel den endlichen Wert 1 annimmt.
Wie in Abschnitt 8.5 noch genauer erläutert wird, gibt es auch robuste Schätzer, die weit außerhalb liegende Stichprobenwerte vollständig ignorieren, d.h. die Einflusskurve hat ab einem bestimmten Abstand vom zu schätzenden Lageparameter den Wert Null (Beispiele in Abschnitt 8.5.2). Der Grenzwert, der die IC-Kurve definiert, kann als Ableitung des Funktionals T interpretiert werden. Man bezeichnet diese Ableitung als Gâteaux-Differential, das im Folgenden vorgestellt wird.
Es seien F und G Verteilungen aus der Menge $\mathcal{F}$ aller Verteilungsfunktionen und

$$\{(1-\lambda)F + \lambda G,\ 0 \leq \lambda \leq 1\}$$

sei die Menge der Konvexkombinationen aus F und G. Das Funktional T sei definiert auf $F + \lambda(G - F)$ für alle hinreichend kleinen λ. Existiert der Grenzwert

$$d_1 T(F; G - F) = \lim_{\lambda \to 0+} \frac{T(F + \lambda(G - F)) - T(F)}{\lambda} \quad ,$$

so wird dieser als das Gâteaux-Differential des Funktionals T an der Stelle F in Richtung von G bezeichnet. Man erkennt, dass der Ausdruck $d_1T(F; G - F)$ die rechtsseitige Ableitung der reellen Funktion $Q(\lambda) = T(F + \lambda(G - F))$ nach λ an der Stelle $\lambda = 0$ ist. Als nächstes wird die Abweichung der Schätzung $T(F_n)$ von dem wahren Wert $T(F)$ mit Hilfe des Gâteaux-Differentials von T an der Stelle F in Richtung F_n untersucht. Es sei

$$Q(\lambda) = T(F + \lambda(F_n - F)) \quad .$$

Die lineare Taylorapproximation von Q an der Stelle $\lambda = 0$ ergibt

$$\begin{aligned} T(F_n) - T(F) &= Q(1) - Q(0) \\ &\approx Q'(0+) \\ &= d_1T(F; F_n - F) \quad . \end{aligned}$$

Ist das Gâteaux-Differential im zweiten Argument linear, erhält man weiter

$$T(F_n) - T(F) \approx \frac{1}{n}\sum_{i=1}^{n} d_1T(F; \delta_{x_i} - F) \tag{12}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \mathrm{IC}(x_i; F, T) \quad . \tag{13}$$

Dieses Ergebnis liefert eine weitere Begründung für die Bezeichnung Einflusskurve. $\mathrm{IC}(\cdot)$ approximiert den Beitrag der Beobachtung x_i zu dem Schätzfehler $T(F_n) - T(F)$. Darüber hinaus zeigt sich aus der Darstellung des Fehlers in (12), dass dieser näherungsweise als arithmetisches Mittel der IC-Kurve verstanden werden kann. Das arithmetische Mittel ist aufgrund des Zentralen Grenzwertsatzes bei existierender positiver Varianz von $\mathrm{IC}(X; F, T)$ asymptotisch normalverteilt. Damit kann man schließen, dass der Fehler, d.h. die linke Seite der Abschätzung (12), unter bestimmten Voraussetzungen ([18], S. 225f) asymptotisch normalverteilt ist. Sind die Bedingungen

$$\mathrm{Var}(\mathrm{IC}(X; F, T)) > 0 \quad ,$$

$$\sqrt{n}\left((T(F_n) - T(F)) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{IC}(X_i; F, T)\right) \xrightarrow{st.} 0 \tag{14}$$

erfüllt [3,4], so gilt

$$T(F_n) - T(F) \overset{as.}{\sim} \mathcal{N}\left(\mathbb{E}(\mathrm{IC}(X; F, T)),\ \frac{1}{n}\mathrm{Var}(\mathrm{IC}(X; F, T))\right) \quad ,$$

[3] Der Begriff der stochastischen Konvergenz wird auf Seite 75 definiert.
[4] In [18] wird für viele Beispielfälle gezeigt, dass die Bedingungen (14) erfüllt sind.

wobei die Bezeichnung $\overset{as.}{\sim} \mathcal{N}(\ldots)$ bedeutet, dass der Ausdruck

$$\frac{T(F_n) - T(F) - \mathbb{E}(\text{IC}(X; F, T))}{\sqrt{\frac{1}{n}\text{Var}(\text{IC}(X; F, T))}}$$

asymptotisch standardnormalverteilt ist. Im Allgemeinen liegt asymptotische Erwartungstreue vor, d.h. der Erwartungswert der Einflusskurve ist gleich Null und für die Varianz gilt

$$\text{Var}(\text{IC}(X; F, T)) = \mathbb{E}(\text{IC}(X; F, T)^2) \quad .$$

Anhand der Maximum-Likelihood-Schätzung soll im Folgenden das beschriebene Vorgehen veranschaulicht werden.

Seien $X_1, \ldots, X_n$ unabhängig und identisch verteilte Zufallsgrößen mit einer Verteilung F_θ, die zu einer Familie $\{F_\theta, \theta \in \Theta \subset \mathbb{R}\}$ gehört. Außerdem sei vorausgesetzt, dass die Verteilung F_θ eine Dichtefunktion $f(x; \theta)$ besitze. Die Likelihood-Funktion der Stichprobe $x_1, \ldots, x_n$ ist definiert als

$$L(t; x_1, \ldots, x_n) = \prod_{i=1}^{n} f(x_i; t) \quad .$$

Die Maximum-Likelihood-Methode bietet als Schätzer für das unbekannte t jeden Wert $\hat{\theta}$ an, der die Likelihood-Funktion über Θ maximiert. Dabei stellt es eine Erleichterung dar, wenn man zu der negativen, logarithmierten Likelihood-Funktion

$$-\log\left(L(t; x_1, \ldots, x_n)\right) = -\sum_{i=1}^{n} \log\left(f(x_i; t)\right)$$

übergeht und so das Produkt durch eine Summe ersetzen kann. Durch den Vorzeichenwechsel ist nun ein Minimum zu bestimmen. Dazu wird eine Nullstelle der ersten Ableitung gesucht, d.h. die Gleichung

$$\left.\frac{\partial \log L}{\partial t}\right|_{t=\hat{\theta}} = 0 \tag{15}$$

ist zu lösen. (15) entspricht der Integralgleichung

$$\int g(x, t)\, dF_n(x) = 0$$

mit

$$g(x, t) = \frac{d}{dt} \log\left(f(x; t)\right) \quad ,$$

d.h.

$$g(x,t) = \frac{\frac{d}{dt} f(x;t)}{f(x;t)} \quad ,$$

wobei

$$f(x;t) = \frac{d}{dx} F_t(x) \quad .$$

Damit ist der Maximum-Likelihood-Schätzer $\hat{\theta} = T(F_n)$, wobei $T(F)$ das Funktional ist, das durch die Lösung von

$$\int g(x,t)\ dF(x) = 0 \tag{16}$$

definiert wird. Unter gewissen Regularitätsbedingungen ([18], S. 144) an $\{F_\theta, \theta \in \Theta\}$ ergeben sich

$$\int \frac{d}{dt} \log\left(f(x;t)\right)\Big|_{t=\theta}\ dF_\theta(x) = 0 \tag{17}$$

und

$$\int \frac{\frac{d^2}{dt^2} f(x;t)}{f(x;t)}\Big|_{t=\theta}\ dF_\theta(x) = \int \frac{d^2}{dt^2} f(x;t)\Big|_{t=\theta}\ dx = 0 \quad . \tag{18}$$

Aus (17) folgt unmittelbar

$$\int g(x,\theta)\ dF_\theta(x) = 0 \quad ,$$

d.h.

$$T(F_\theta) = \theta \quad .$$

Seien $F_\lambda = F_\theta + \lambda(\delta_{x_0} - F_\theta)$ und $H(t,\lambda) = \int g(x,t)\ dF_\lambda(x)$, dann erhält man durch implizites Differenzieren der Gleichung

$$H(T(F_\lambda), \lambda) = 0$$

nach λ an der Stelle $\lambda = 0$ die Gleichung

$$\frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=\theta,\lambda=0} \cdot \frac{dT(F_\lambda)}{d\lambda}\Big|_{\lambda=0} + \frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=\theta,\lambda=0} = 0$$

und daraus ergibt sich

$$\frac{dT(F_\lambda)}{d\lambda}\Big|_{\lambda=0} = \frac{-\frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=\theta,\lambda=0}}{\frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=\theta,\lambda=0}} \quad . \tag{19}$$

Setzt man nun die bis dahin gewonnenen Ergebnisse ein, so erhält man

$$\begin{aligned}
\left.\frac{\partial H(t,\lambda)}{\partial \lambda}\right|_{t=\theta,\lambda=0} &= \left.\frac{\partial}{\partial \lambda}\left[\int g(x,t)\; dF_\theta(x) + \lambda \int g(x,t)\; d[\delta_{x_0} - F_\theta(x)]\right]\right|_{t=\theta,\lambda=0} \\
&= \int g(x,\theta)\; d\delta_{x_0} - \underbrace{\int g(x,\theta)\; dF_\theta(x)}_{\overset{(17)}{=}0} \\
&= \left.\frac{\frac{d}{dt} f(x_0;t)}{f(x_0;t)}\right|_{t=\theta}
\end{aligned}$$

und

$$\begin{aligned}
\left.\frac{\partial H(t,\lambda)}{\partial t}\right|_{t=\theta,\lambda=0} &= \int \left.\frac{d}{dt} g(x,t)\right|_{t=\theta} dF_\theta(x) \\
&= \int \left.\frac{d}{dt}\frac{\frac{d}{dt} f(x;t)}{f(x;t)}\right|_{t=\theta} dF_\theta(x) \\
&= \int \frac{\left.\frac{d^2}{dt^2} f(x;t)\right|_{t=\theta} \cdot f(x;t)|_{t=\theta} - \left(\left.\frac{d}{dt} f(x;t)\right|_{t=\theta}\right)^2}{f(x;\theta)^2}\; dF_\theta(x) \\
&= \underbrace{\int \frac{\left.\frac{d^2}{dt^2} f(x;t)\right|_{t=\theta}}{f(x;\theta)}\; dF_\theta(x)}_{\overset{(18)}{=}0} - \int \left.\left(\frac{d}{dt}\log\left(f(x;t)\right)\right)^2\right|_{t=\theta} dF_\theta(x) \\
&= -\int \left.\left(\frac{d}{dt}\log\left(f(x;t)\right)\right)^2\right|_{t=\theta} dF_\theta(x) \quad .
\end{aligned}$$

Damit folgt für das Gâteaux-Differential

$$\begin{aligned}
d_1 T(F_\theta; \delta_{x_0} - F_\theta) &= \left.\frac{d}{d\lambda} T(F_\theta + \lambda(\delta_{x_0} - F_\theta))\right|_{\lambda=0} \\
&= \frac{\frac{d}{d\theta} f(x_0;\theta)}{f(x_0;\theta)} \cdot \frac{1}{\int \frac{\left(\frac{d}{d\theta} f(x;\theta)\right)^2}{f(x;\theta)}\; dx}
\end{aligned}$$

und für die Einflusskurve

$$\mathrm{IC}(x; F_\theta, T) = \frac{\frac{d}{d\theta}\log\left(f(x;\theta)\right)}{\int \left(\frac{d}{d\theta}\log\left(f(x;\theta)\right)\right)^2 dF_\theta(x)} \quad .$$

8.3.3 Der Bruchpunkt

Der Bruchpunkt (breakdown point) ε^* einer Schätzung gibt diejenige Grenze an, bis zu welcher der Anteil von Ausreißern in einer Stichprobe ansteigen darf, ohne dass sich dadurch der Schätzwert unbeschränkt verändern kann. Im Gegensatz zur Einflusskurve ist der Bruchpunkt eine globale Kenngröße der Robustheit. Für die Bruchpunkte von arithmetischem Mittel und Median ergeben sich:

(a) Beim arithmetischen Mittel kann bereits ein einzelner Wert innerhalb einer Beobachtungsreihe den Schätzwert für das Zentrum der Stichprobe über alle Grenzen wachsen lassen, daher gilt

$$\varepsilon^*_{\text{arith. Mittel}} = 0 \quad .$$

(b) Liegt der Anteil der Ausreißer unter 50%, so kann keine beliebige Veränderung des Medians vorkommen. Der Median hat daher als Bruchpunkt den Wert

$$\varepsilon^*_{\text{Median}} = 0,5 \quad .$$

8.4 L-Schätzer

Die in Abschnitt 8.2 vorgestellten Schätzer für Lage- und Streuungsmaßzahlen beruhen alle auf Linearkombinationen der Ordnungsstatistiken $x_{(1)}, \ldots, x_{(n)}$ einer Stichprobe vom Umfang n. Daher werden sie L-Schätzer genannt. Seien $a_1, \ldots, a_n$ reelle Zahlen mit $0 \leq a_i \leq 1, i = 1, \ldots, n$ und $\sum_{i=1}^{n} a_i = 1$, dann ist

$$T = \sum_{i=1}^{n} a_i \; x_{(i)}$$

ein L-Schätzer mit Gewichten $a_1, \ldots, a_n$.
Das arithmetische Mittel ist also ebenso ein L-Schätzer wie der Median oder der Interquartilabstand. Die Bezeichnung „L-Schätzer“ sagt also noch nichts über die Eigenschaft der Robustheit aus. Ein Vorteil der L-Schätzer liegt in der leichten Berechenbarkeit, die in der Regel lediglich eine Vorsortierung der Stichprobenelemente erfordert.

8.4.1 L-Schätzer für Lageparameter

8.4.1.1 Das α-gestutzte Mittel

Es sei vorausgesetzt, dass die zu Grunde liegende Verteilung symmetrisch und stetig ist, d.h. sie besitzt eine Dichte. Beim α-gestutzten Mittel als Schätzer für den Median bzw., sofern er existiert, den Erwartungswert der zu Grunde liegenden Verteilung werden aus der geordneten Stichprobe am unteren und oberen Ende eine gleiche Anzahl von Beobachtungen weggelassen und über die verbliebenen Werte das arithmetische Mittel berechnet. Der

Anteil der zu entfernenden Werte wird mit α angegeben. Unter dem α-gestutzten Mittel versteht man daher den Wert

$$\overline{x}_\alpha = \frac{1}{n-2h} \sum_{i=h+1}^{n-h} x_{(i)} \quad ,$$

wobei $0 < \alpha < 0,5$ ist und $h = [n\alpha]$ die größte ganze Zahl kleiner oder gleich $n\alpha$ bezeichnet. Dieser Ansatz stellt damit einen Kompromiss zwischen dem arithmetischen Mittel und dem Median dar, die sich für die Werte $\alpha \to 0$ bzw. $\alpha \to 0,5$ ergeben würden. Eine asymptotisch äquivalente Version des α-getrimmten Mittels ist durch

$$\overline{x}_\alpha = T(F_n)$$

gegeben, wobei für eine beliebige Verteilungsfunktion F das Funktional T durch

$$\begin{aligned} T(F) &= \frac{1}{1-2\alpha} \int_{F^{-1}(\alpha)}^{F^{-1}(1-\alpha)} x \, dF(x) \\ &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} F^{-1}(p) dp \end{aligned}$$

mit $F^{-1}(\alpha) = \inf\{x | F(x) \geq \alpha\}$ definiert ist.
Für die Einflusskurve gilt zunächst nach [18] für eine stetige Verteilung

$$\begin{aligned} \left. \frac{dT[F + \lambda(\delta_{x_0} - F)]}{d\lambda} \right|_{\lambda=0} &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} \frac{p - I(x_0 \leq F^{-1}(p))}{f\left(F^{-1}(p)\right)} \, dp \\ &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} \frac{p - I(F(x_0) \leq p)}{f\left(F^{-1}(p)\right)} \, dp \quad , \end{aligned}$$

wobei $I(\cdot)$ die Indikatorfunktion von Aussagen[5] bezeichnet. Hieraus folgt für eine symmetrische stetige Verteilung

$$\left. \frac{dT[F + \lambda(\delta_{x_0} - F)]}{d\lambda} \right|_{\lambda=0}$$

$$= \begin{cases} \frac{1}{1-2\alpha}[F^{-1}(\alpha) - F^{-1}(1/2)] & \text{, falls} \quad x < F^{-1}(\alpha) \\ \frac{1}{1-2\alpha}(x - F^{-1}(1/2)) & \text{, falls} \quad F^{-1}(\alpha) \leq x \leq F^{-1}(1-\alpha) \\ \frac{1}{1-2\alpha}[F^{-1}(1-\alpha) - F^{-1}(1/2)] & \text{, falls} \quad x > F^{-1}(1-\alpha) \end{cases} .$$

[5] $I(A) = \begin{cases} 1 \text{ , falls die Aussage } A \text{ wahr ist.} \\ 0 \text{ , falls die Aussage } A \text{ falsch ist.} \end{cases}$

8.4.1.2 Das α-winsorisierte Mittel

Beim α-winsorisierten Mittelwert wird im Gegensatz zum gestutzten Mittelwert die Anzahl der Beobachtungen beibehalten. Jedoch werden die ersten h Beobachtungen durch den $(h+1)$-ten Wert ersetzt und die h letzten Werte durch den $(n-h)$-ten Wert, dann wird das arithmetische Mittel gebildet, d.h.

$$\bar{x}_{\alpha,win} = \frac{1}{n}\left(\sum_{i=h+1}^{n-h} x_{(i)} + h(x_{(h+1)} + x_{(n-h)})\right) \quad .$$

8.4.2 L-Schätzer für die Streuung

8.4.2.1 Der Interquartilabstand

Der Begriff des Interquartilabstandes wurde bereits in Abschnitt 8.2.2 definiert. Er dient an dieser Stelle der Motivation des robusten Skalenschätzers MAD, der im nächsten Unterabschnitt eingeführt wird. Der Bruchpunkt des Interquartilabstandes beträgt $\varepsilon^*_{\text{IQR}} = 0,25$. Mit den bekannten Werten aus Beispiel 8.2.1 erhält man wie in Abschnitt 8.2.2 bereits angegeben

$$\tilde{x}_{0,25} = -1,0\ ,\ \tilde{x}_{0,75} = 0,5 \quad \Rightarrow \quad \text{IQR} = 1,5 \quad .$$

Ergänzt man $x_6 = 10,5$, so ergibt sich der Wert IQR $= 2,5$, jedoch erhält man mit $\hat{x}_6 = -10,5$ einen IQR von 10,0. Ursache für diesen auffälligen Unterschied ist die Lage der beiden Ausreißer am gleichen Ende der Stichprobe. Ihr Anteil an der Stichprobe beträgt $\frac{2}{6} = 0,33$ und übersteigt damit den Bruchpunkt.

8.4.2.2 Der Median der absoluten Abweichung vom Median

Sei F eine beliebige Verteilungsfunktion, dann ist

$$T(F) = \frac{1}{2}\left[F^{-1}\left(\frac{3}{4}\right) - F^{-1}\left(\frac{1}{4}\right)\right]$$

das Funktional, das die Hälfte des Interquartilabstandes angibt. Zu diesem Funktional kann man nun eine symmetrische Version $\tilde{T}$ konstruieren, indem man

$$\bar{F}(x) = 1 - F(2 \cdot F^{-1}(1/2) - x)$$

und

$$\tilde{F}(x) = \frac{1}{2}[F(x) + \bar{F}(x)]$$

setzt. $\tilde{F}(x)$ entsteht dabei aus $F(x)$ durch eine Symmetrisierung am Median m_n. Definiert man anschließend

$$\tilde{T}(F) := T(\tilde{F}) \quad ,$$

so ist $\tilde{T}$ der Median der absoluten Abweichung vom Median (MAD). Als Median der absoluten Abweichung vom Median einer Stichprobe vom Umfang n bezeichnet man daher

$$\text{MAD}_n = \text{med}\{|x_i - m_n|\} \quad ,$$

wobei

$$\text{med}\{x_i\} = m_n \quad .$$

Der Vorteil der symmetrisierten Version des halben Interquartilabstandes liegt darin, dass sich der Wert für den Bruchpunkt verdoppelt, d.h.

$$\varepsilon^*_{\text{MAD}} = 0,5 = 2 \cdot 0,25 = 2 \cdot \varepsilon^*_{\text{IQR}} \quad .$$

Darüber hinaus ist die Einflusskurve beschränkt, so dass einzelne Beobachtungswerte nur begrenzten Einfluss auf den Schätzwert haben.

Das folgende Beispiel der Schätzung der Standardabweichung einer Normalverteilung zeigt, wie man gegebenenfalls mittels einer Multiplikation mit einer Konstanten einen Schätzwert für einen gesuchten Parameter erhält.
Für eine symmetrische Verteilung ist der MAD eine Schätzung des halben Interquartilabstandes. Sind $x_1, \ldots, x_n$ Realisationen aus einer $\mathcal{N}(\mu, \sigma^2)$-Verteilung, so ist der MAD ein Schätzer für

$$\Phi^{-1}(0.75) \cdot \sigma \quad ,$$

wobei $\Phi^{-1}(0.75) = 0,6745$ das $0,75$-Quantil der Standardnormalverteilung ist [7]. Somit ist $\frac{1}{0,6745} \cdot \text{MAD}$ ein Schätzer für σ. Verwendet man den Interquartilabstand, so ergibt sich entsprechend, dass $\frac{1}{2} \cdot \frac{1}{0,6745} \cdot \text{IQR} \approx 0,7413 \cdot \text{IQR}$ ein Schätzer für σ ist.

8.4.2.3 Die α-gestutzte Varianz

Die α-gestutzte Varianz ist definiert als die skalierte Varianz einer α-gestutzten Stichprobe. Die Varianz einer α-gestutzten Stichprobe

$$s^2_{gest} = \frac{1}{n - 2[n\alpha]} \sum_{i=[n\alpha]+1}^{n-[n\alpha]} (x_{(i)} - \bar{x}_\alpha)^2 \tag{20}$$

unterschätzt die Varianz der den Beobachtungen zugrundeliegenden Verteilung und ist keine konsistente Schätzung. Daher benutzt man einen Korrekturterm

$$\frac{1 - 2\alpha}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha})} \quad ,$$

wobei $\tilde{x}_{1-\alpha}$ das $(1-\alpha)$-Quantil und ϕ die Dichte der Standardnormalverteilung ist. Damit ist

$$\hat{\sigma}^2_{gest} = \frac{1 - 2\alpha}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha})} \cdot s^2_{gest}$$

eine konsistente Schätzung für σ^2 unter der Normalverteilung.

8.4.2.4 Die α-winsorisierte Varianz

Die α-winsorisierte Varianz ist definiert als

$$s^2_{win} = \frac{1}{n-1}\left(\sum_{i=[n\alpha]+1}^{n-[n\alpha]} (x_{(i)} - \bar{x}_{\alpha,win})^2 \right.$$
$$\left. + [n\alpha](x_{([n\alpha]+1)} - \bar{x}_{\alpha,win})^2 + [n\alpha](x_{(n-[n\alpha])} - \bar{x}_{\alpha,win})^2 \right) \quad ,$$

wobei auch dies zunächst kein konsistenter Schätzer für die Varianz ist. Analog zur α-gestutzten Varianz kann man auch für diesen Fall einen Korrekturterm angeben, so dass

$$\hat{\sigma}^2_{win} = \frac{1}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha}) + 2\alpha\tilde{x}^2_{1-\alpha}} \cdot s^2_{win}$$

insgesamt ein konsistenter Schätzer für σ^2 unter der Normalverteilung ist.

8.5 M-Schätzer

8.5.1 Theoretischer Hintergrund

Die Theorie der M-Schätzer basiert auf einer Verallgemeinerung der Maximum-Likelihood-Methode (s. Abschnitt 8.3.2). Wie dort sei $F_\theta(x) := F(x;\theta)$ eine Verteilung aus der Verteilungsfamilie $\{F_\theta |\ \theta \in \Theta \subset \mathbb{R}\}$ mit einer Dichte $f_\theta(x) := f(x;\theta)$. Die Maximum-Likelihood-Methode liefert einen Schätzer für den unbekannten Parameter θ, indem der Ausdruck $\sum_{i=1}^{n}(-\log(f(x_i;\theta)))$ minimiert wird. Ist f differenzierbar, so führt dies zur Lösung des Gleichungssystems

$$\sum_{i=1}^{n} \frac{\partial}{\partial\theta}(-\log(f(x_i;\theta))) = 0 \quad .$$

Dieser Ansatz führt unter der Voraussetzung, dass eine Normalverteilung mit Erwartungswert μ und Varianz σ^2 vorliegt, zu folgender Problemstellung bei der Suche nach einem Schätzer für den Erwartungswert:

$$\sum_{i=1}^{n} \frac{(x_i - \mu)^2}{2} \to \min_{\mu} \quad ,$$

d.h.

$$\sum_{i=1}^{n}(x_i - \mu) = 0 \quad .$$

Das Ergebnis ist das arithmetische Mittel $\bar{x}_n$. Probleme bei diesem Vorgehen tauchen dann auf, wenn Ausreißer in den Daten vorhanden sind. Das heißt, die Ausreißerempfindlichkeit der ML-Methode ist nicht befriedigend. Um diese zu verbessern, ersetzt man $-\log(f(x;\theta))$ durch eine Funktion $\rho(x,\theta)$, die weniger sensitiv auf Ausreißer reagiert. Damit lautet das verallgemeinerte Vorgehen:

$$\text{minimiere den Ausdruck } \sum_{i=1}^{n} \rho(x_i,\theta) \tag{21}$$

bzw.

$$\text{löse das Gleichungssystem } \sum_{i=1}^{n} \psi(x_i,\theta) = 0 \quad , \tag{22}$$

wobei $\psi(x,\theta)$ die Ableitung von $\rho(x,\theta)$ nach θ ist. Jede Lösung von (21) bzw. (22) wird M-Schätzer genannt.

8.5.1.1 Grundlagen

Zu einer beliebigen Funktion $\psi(x,t)$ sei ein zugehöriges Funktional T auf einer Menge von Verteilungsfunktionen F wie folgt definiert. Es ist $T(F)$ die Lösung t_0 der Gleichung

$$\int \psi(x,t_0)\; dF(x) = 0 \quad . \tag{23}$$

Ein solches T wird das zu ψ gehörende M-Funktional genannt. Für eine Stichprobe $x_1,\ldots,x_n$ aus einer Verteilung F ist der zu ψ gehörende M-Schätzer T_n die Lösung der Gleichung

$$\sum_{i=1}^{n} \psi(x_i,T_n) = 0 \quad ,$$

die sich aus (23) mit $F = F_n$ ergibt, d.h. $T_n = T(F_n)$. Dabei ist zu beachten, dass mehrere Lösungen existieren können. Gleichung (23) ergibt sich in vielen Fällen aus der Minimierung des Ausdrucks

$$\int \rho(x,t_0)\; dF(x) \quad ,$$

wobei dann die Funktion ψ bestimmt ist durch

$$\psi(x,t) = c \cdot \frac{\partial}{\partial t}\rho(x,t)$$

mit einer Konstanten c, falls $\rho(x,\cdot)$ hinreichend glatt ist. Gilt für ein Funktional T auf der Verteilungsfamilie $\{F_\theta |\ \theta \in \Theta \subset \mathbb{R}\}$

$$T(F_\theta) = \theta \quad ,$$

dann ist der zugehörige Schätzer $T_n = T(F_n)$ ein Schätzer für θ. Ein Kriterium zur Beurteilung eines Schätzers stellt die Einflusskurve (s. Abschnitt 8.3.2) dar, die somit zur Bestimmung einer geeigneten ψ-Funktion und eines geeigneten M-Schätzers herangezogen werden kann. Ziel ist es nun, das Gâteaux-Differential eines M-Funktionals zu berechnen, d. h. es ist $d_1T(F, G-F)$ für ein Funktional T gesucht, das Lösung t_0 der Gleichung

$$\int \psi(x, t_0) dF(x) = 0$$

ist. $\psi(x,t)$ sei dabei eine beliebige Funktion. Es sei

$$H(t, \lambda) = \int \psi(x,t) dF_\lambda(x) \quad .$$

Mit

$$F_\lambda(x) = F(x) + \lambda(G(x) - F(x))$$

folgt wie in Abschnitt 8.3.2 (Maximum-Likelihood-Schätzung) durch implizite Differentiation der Gleichung $H(T(F_\lambda), \lambda) = 0$ nach λ an der Stelle $\lambda = 0$ [6]

$$\begin{aligned}
d_1T(F, G-F) &= \frac{d}{d\lambda} T(F + \lambda(G-F))\Big|_{\lambda=0} \\
&= -\frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=T(F),\lambda=0} \Big/ \frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=T(F),\lambda=0} \\
&= \frac{-\frac{\partial}{\partial \lambda}\left[\int \psi(x,t)\, dF(x) + \lambda \int \psi(x,t)\, d[G(x)-F(x)]\right]\big|_{t=T(F),\lambda=0}}{\frac{d}{dt}\int \psi(x,t)\, dF(x)\big|_{t=T(F)}} \\
&= -\frac{\int \psi(x,t)\, dG(x)\big|_{t=T(F)} - \overbrace{\int \psi(x,t)\, dF(x)\Big|_{t=T(F)}}^{\stackrel{(23)}{=}0}}{\frac{d}{dt}\int \psi(x,t)\, dF(x)\big|_{t=T(F)}} \\
&= -\frac{\int \psi(x, T(F)) dG(x)}{\lambda_F'(T(F))} \quad ,
\end{aligned}$$

vorausgesetzt, dass $\lambda_F'(T(F)) \neq 0$ gilt mit

$$\lambda_F(t) = \int \psi(x,t)\, dF(x), \quad -\infty < t < \infty \quad .$$

Damit besitzt die Einflusskurve von ψ die Form:

$$IC(x; F, T) = -\frac{\psi(x, T(F))}{\lambda_F'(T(F))}, \quad -\infty < x < \infty \quad .$$

[6] g in Abschnitt 8.3.2 entspricht hier ψ.

Aufgrund der Tatsache, dass die Einflusskurve proportional zu ψ ist, besitzen M-Schätzer die Eigenschaft, dass Bedingungen an die Einflusskurve über die Wahl von ψ gesteuert werden können. Eine geeignete Wahl von ψ führt zu vorteilhaften Eigenschaften der Einflusskurve und damit des M-Schätzers.

8.5.1.2 Asymptotische Eigenschaften von M-Schätzern

In diesem Abschnitt werden M-Schätzer hinsichtlich Konsistenz und asymptotischer Normalität untersucht. Zunächst sei dazu auf die verschiedenen Konvergenzbegriffe in der Wahrscheinlichkeitstheorie hingewiesen:

- Seien $X_1, X_2, \ldots$ und X Zufallsgrößen über einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$. Dann konvergiert $(X_n)_{n\geq 1}$ stochastisch gegen X, falls für alle $\varepsilon > 0$

$$\lim_{n\to\infty} \mathbb{P}(|X_n - X| < \varepsilon) = 1 \quad (\text{Bez.} : X_n \xrightarrow{st.} X \quad (n \to \infty)) \quad .$$

- Seien $X_1, X_2, \ldots$ und X Zufallsgrößen über einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$. $(X_n)_{n\geq 1}$ konvergiert fast sicher gegen X, falls

$$\mathbb{P}(\lim_{n\to\infty} X_n = X) = 1 \quad (\text{Bez.} : X_n \xrightarrow{f.s.} X \quad (n \to \infty)) \quad .$$

- Seien $F_1(\cdot), F_2(\cdot), \ldots$ und $F(\cdot)$ Verteilungsfunktionen und $X_1, X_2, \ldots$ sowie X Zufallsgrößen (nicht notwendig über einem gemeinsamen Wahrscheinlichkeitsraum) mit den angegebenen Verteilungen. Dann konvergiert $(X_n)_{n\geq 1}$ der Verteilung nach gegen X, falls für alle Stetigkeitsstellen t von F

$$\lim_{n\to\infty} F_n(t) = F(t) \quad (\text{Bez.} : X_n \xrightarrow{n.V.} X \quad (n \to \infty)) \quad .$$

Konsistenz eines M-Schätzers

Eine Folge $\{T_n\}$ von Schätzern für eine parametrische Funktion $g(\theta)$ heißt konsistent, falls T_n gegen $g(\theta)$ in geeigneter Weise konvergiert. Man spricht von schwacher Konsistenz, falls

$$T_n \xrightarrow{st.} g(\theta) \qquad (n \to \infty) \quad ,$$

und von starker Konsistenz im Falle von

$$T_n \xrightarrow{f.s.} g(\theta) \qquad (n \to \infty) \quad .$$

Sei $\psi(x,t)$ eine Funktion und $\lambda_F(t) = \int \psi(x,t)\, dF(x)$. Außerdem besitze die Gleichung $\lambda_F(t) = 0$ eine Lösung t_0 und die „empirische“ Gleichung $\lambda_{F_n}(t) = 0$ ebenfalls eine Lösung T_n. Im Folgenden werden Bedingungen genannt, unter denen $T_n \xrightarrow{f.s.} t_0$ gilt. Dazu sei $x_1, \ldots, x_n$ eine Stichprobe aus einer nach F verteilten Grundgesamtheit mit empirischer Verteilungsfunktion F_n.
Falls t_0 eine isolierte Lösung der Gleichung $\lambda_F(t) = 0$ ist und $\psi(x,t)$ monoton in t, dann

ist t_0 eindeutig, und jede Folge $\{T_n\}$ von Lösungen der empirischen Gleichung $\lambda_{F_n}(t) = 0$ konvergiert fast sicher gegen t_0. Ist außerdem $\psi(x,t)$ stetig in t in einer Umgebung von t_0, so existiert eine solche Folge von Lösungen. Für den Beweis dieser Behauptung sei an dieser Stelle auf [18], Abschnitt 7.2.1, verwiesen. t_0 muss nicht notwendigerweise eine Nullstelle von $\lambda_F(t)$ sein. Es reicht aus, dass die Funktion in jeder hinreichend kleinen Umgebung genau einmal das Vorzeichen wechselt.
Das arithmetische Mittel, der Median und der Huber-Schätzer auf den Seiten 77 bis 79 sind Beispiele für konsistente Schätzer der jeweiligen Lageparameter. Für den Hampel-Schätzer (s. Beispiel auf Seite 80) reicht dieses Ergebnis jedoch nicht aus, und man benötigt die folgende Aussage.
Sei t_0 eine isolierte Nullstelle der Gleichung $\lambda_F(t) = 0$, und sei $\psi(x,t)$ stetig in t und beschränkt. Dann besitzt die empirische Gleichung $\lambda_{F_n}(t) = 0$ eine Folge $\{T_n\}$ von Lösungen, die mit Wahrscheinlichkeit 1 gegen t_0 konvergiert. Der Beweis hierzu ist ebenfalls in [18], Abschnitt 7.2.1, zu finden.

Asymptotische Normalverteilung

Sei $\psi(x,t)$ gegeben und $\lambda_F(t) = \int \psi(x,t)\, dF(x)$. Außerdem sei $t_0 = T(F)$ eine Lösung der Gleichung $\lambda_F(t) = 0$. $\{X_i\}$ ist eine Folge von unabhängig und identisch verteilten Zufallsgrößen mit einer Verteilung F und $T_n = T(F_n)$ eine zu t_0 konsistente Folge von Lösungen von $\lambda_{F_n}(t) = 0$. In dem vorliegenden Abschnitt sollen nun Bedingungen angegeben werden, unter denen

$$\sqrt{n}(T_n - t_0) \overset{as.}{\sim} \mathcal{N}(0, \sigma^2(T,F)) \tag{24}$$

gilt. Dabei ist $\sigma^2(T,F)$ abhängig von den Annahmen an $\psi(x,t)$ entweder gegeben durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{(\lambda_F'(t_0))^2}$$

oder durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{\left(\int \left(\frac{\partial\psi(x,t)}{\partial t}\right)_{t=t_0} dF(x)\right)^2} \quad .$$

Es sei die Funktion $\psi(x,t)$ monoton in t und außerdem t_0 eine isolierte Nullstelle von $\lambda_F(t) = 0$. Des Weiteren sei $\lambda_F(t)$ differenzierbar in $t = t_0$ mit $\lambda_F'(t_0) \neq 0$ sowie $\int \psi^2(x,t)\, dF(x)$ endlich für t in einer Umgebung von t_0 und stetig in $t = t_0$. Dann gilt nach [18], Abschnitt 7.2.2, dass jede Folge T_n von Lösungen der empirischen Gleichung $\lambda_{F_n}(t) = 0$ die Bedingung (24) erfüllt, wobei $\sigma^2(T,F)$ gegeben ist durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{[\lambda_F'(t_0)]^2} \quad .$$

Darüber hinaus gilt $T_n \xrightarrow{f.s.} t_0$ nach dem Ergebnis von Seite 75.

8.5.2 M-Schätzer für Lageparameter

Sind die das M-Funktional bestimmenden Funktionen ψ und ρ von der Form $\psi(x,t) = \tilde{\psi}(x-t)$ bzw. $\rho(x,t) = \tilde{\rho}(x-t)$, so nennt man $T(F)$ einen Lageparameter. Ist zudem F symmetrisch zu θ, ρ gerade und ψ ungerade, so gilt $T(F) = \theta$, d.h. alle M-Schätzer mit ungeradem ψ schätzen das Symmetriezentrum θ einer symmetrischen Verteilung. Außerdem sollten M-Schätzer translations- und skalierungsäquivariant sein, d.h. es sollte gelten:

$$T_n(x_1,\ldots,x_n) = BT_n\left(\frac{x_1-A}{B},\ldots,\frac{x_n-A}{B}\right)+A \quad .$$

Die meisten M-Schätzer für einen Lageparameter müssen daher eine Skalierung der Daten berücksichtigen, so dass sie diese Bedingung erfüllen. Dabei stellen das arithmetische Mittel und der Median zwei Ausnahmen dar, denn für den Mittelwert gilt

$$\psi(x,t) = x - t$$

und damit folgt

$$\begin{aligned}\psi(bx+a,bt+a) &= bx+a-bt-a \\ &= bx-bt \\ &= b(x-t) \\ &= b\psi(x,t) \quad .\end{aligned}$$

Analoges gilt für den Median mit $\psi(x,t) = \text{sgn}(x-t)$. Alle anderen M-Schätzer benötigen die Skalierung der Argumente von ρ und ψ. Daher wird ein Skalierungsschätzer s_n als Funktion von $x_1,\ldots,x_n$ benutzt, der zusammen mit einer Tuning-Konstante C den Ausdruck $x_i - t$ neu skaliert. Damit erhält man zentrierte und skalierte Daten über den Ausdruck

$$u_i = \frac{x_i - t}{C \cdot s_n} \quad .$$

8.5.2.1 Das arithmetische Mittel

Sucht man einen kleinste-Quadrate-Schätzer, so will man den Ausdruck

$$\sum_{i=1}^{n}(x_i-\theta)^2$$

minimieren. Als ρ und ψ ergeben sich in diesem Falle

$$\rho(u) = \frac{1}{2}u^2 \quad , \quad \psi(u) = u \quad , \quad -\infty < u < \infty \quad .$$

Damit ist das M-Funktional T der Erwartungswert und der M-Schätzer das arithmetische Mittel $\overline{x}_n$. Die ψ-Funktion ist eine Gerade und nach beiden Seiten unbeschränkt, d.h. Ausreißer haben einen starken Einfluss.

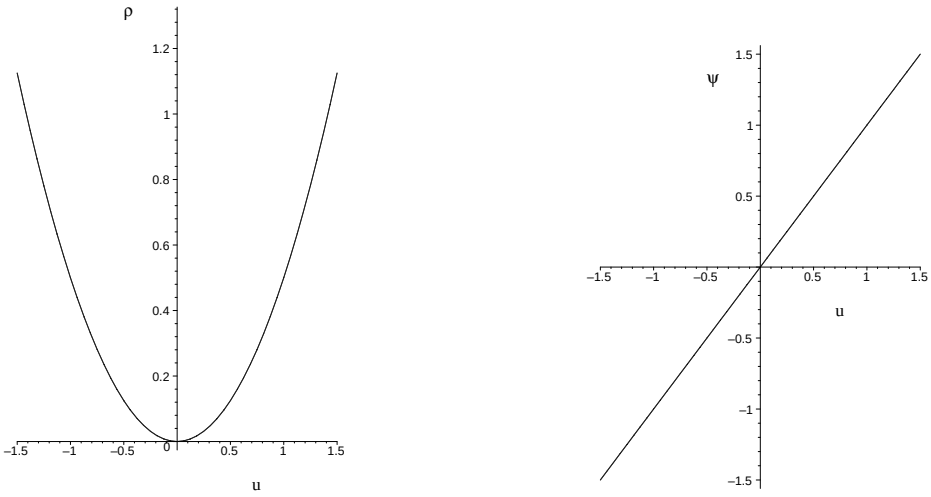


Abbildung 13: Arithmetisches Mittel $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.2 Der Median

Möchte man die absoluten Abweichungen minimieren, also den Ausdruck

$$\sum_{i=1}^{n} |x_i - \theta| \quad ,$$

so lauten die zugehörigen Funktionen für ρ und ψ:

$$\rho(u) = |u| \quad , \quad \psi(u) = \text{sgn}(u) \quad , \quad -\infty < u < \infty \quad .$$

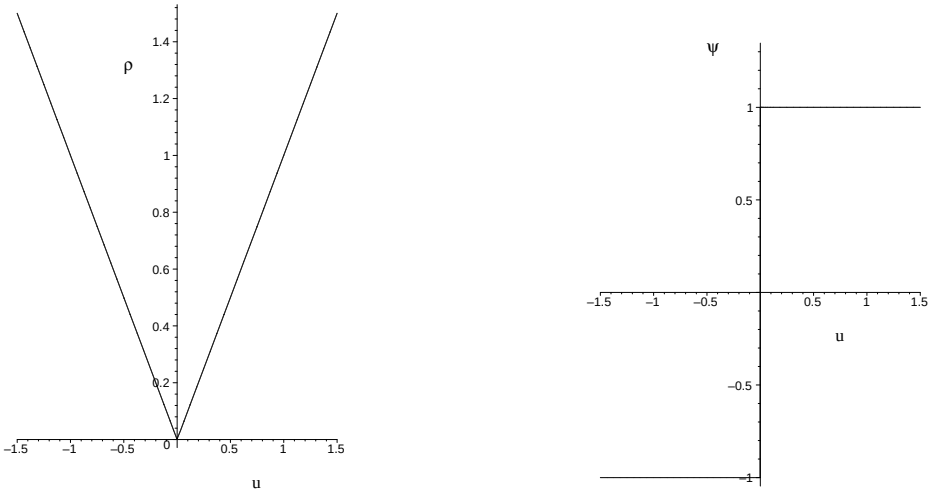


Abbildung 14: Median $\rho(u)$ (links) und $\psi(u)$ (rechts).

Hier ist der M-Schätzer der Stichprobenmedian. Er ist unempfindlich gegenüber Ausreißern, da die ψ-Funktion beschränkt ist. Die Sprungstelle von ψ bei $u = 0$ zeigt jedoch an, dass der Median sehr stark von den „mittleren“ Werten abhängt.

8.5.2.3 Der Huber-Schätzer

Als Familie der Huber-Schätzer bezeichnet man die M-Schätzer, die

$$\sum_{i=1}^{n} \rho(u_i)$$

minimieren. Dabei ist ρ von der Form

$$\rho(u) = \begin{cases} \frac{1}{2}u^2 & \text{, falls} \quad |u| \leq k \\ k|u| - \frac{1}{2}k^2 & \text{, falls} \quad |u| > k \end{cases} .$$

Das entsprechende ψ lautet

$$\psi(u) = \begin{cases} u & \text{, falls} \quad |u| \leq k \\ k \cdot \text{sgn}(u) & \text{, falls} \quad |u| > k \end{cases} .$$

Dabei hat sich in Anwendungen gezeigt, dass sich als Skalierungsschätzer s_n der normalisierte MAD empfiehlt. C wird gleich Eins gesetzt, und für k sollte man einen Wert im Intervall $[1; 2]$ wählen. Der zugehörige M-Schätzer ist von der Art eines winsorisierten Mittelwertes, d.h. es ist der Stichprobenmittelwert über die x_i's, wobei diejenigen x_i durch $T_n \pm ks_n$ ersetzt werden, für die $|u_i| > k$ gilt. Hierbei wird der Einfluss von Ausreißern gedämpft, aber nicht vollständig ausgeräumt.

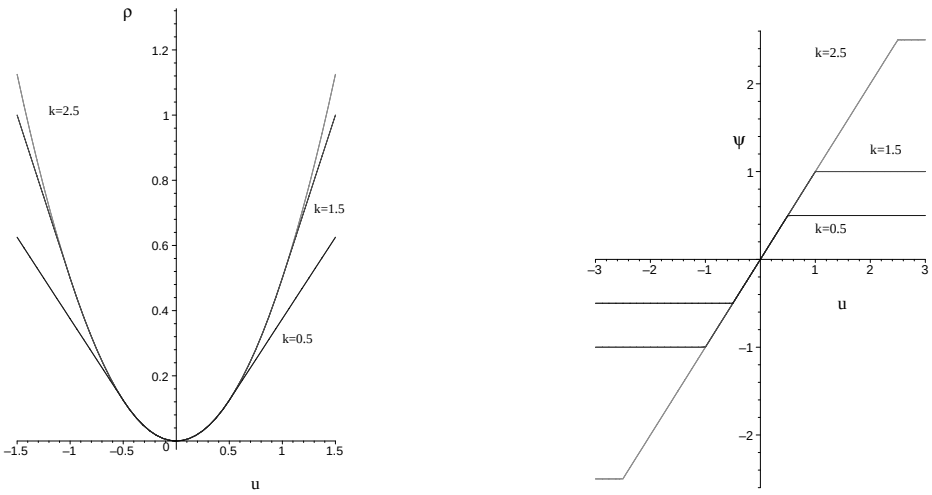


Abbildung 15: Huber-Schätzer $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.4 Der Hampel-Schätzer

Hampel [5] hat den Huber-Schätzer im Bezug auf die Ausreißerempfindlichkeit (Gross-Error-Sensitivity) modifiziert. Er verlangt, dass $\psi(u)$ für Ausreißer den Wert Null annehmen soll. Analog zu den bisherigen Fällen ist auch hier der Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

zu minimieren. Dabei sind

$$\rho(u) = \begin{cases} \frac{1}{2}u^2 & \text{, falls} \quad |u| \le a \\ a|u| - \frac{1}{2}a^2 & \text{, falls} \quad a < |u| \le b \\ ab - \frac{1}{2}a^2 + (c-b)\frac{a}{2}\left[1 - \left(\frac{c-|u|}{c-b}\right)^2\right] & \text{, falls} \quad b < |u| \le c \\ ab - \frac{1}{2}a^2 + (c-b)\frac{a}{2} & \text{, falls} \quad |u| > c \end{cases}$$

und

$$\psi(u) = \begin{cases} u & \text{, falls} \quad |u| \le a \\ a \cdot \text{sgn}(u) & \text{, falls} \quad a < |u| \le b \\ a \cdot \left(\frac{c-|u|}{c-b}\right) \cdot \text{sgn}(u) & \text{, falls} \quad b < |u| \le c \\ 0 & \text{, falls} \quad |u| > c \end{cases} .$$

Hierbei sind a, b und c positive Konstanten mit $a < b < c$ aus dem Intervall $[2; 9]$ (zum Beispiel $a = 2, b = 4$ und $c = 8$). Als Skalierungsschätzer s_n wird wie im vorangegangenen Beispiel beim Huber-Schätzer der normalisierte MAD benutzt und C gleich Eins gesetzt. Beim Hampel-Schätzer bleiben Ausreißer vollständig unberücksichtigt.

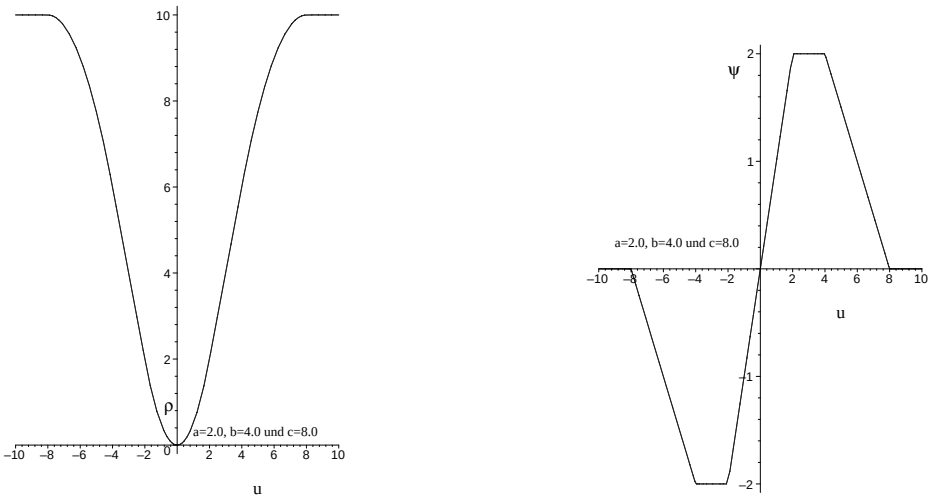


Abbildung 16: Hampel-Schätzer $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.5 Andrew's wave

Man geht hier ebenfalls von dem zu minimierenden Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

aus mit

$$\rho(u) = \begin{cases} \frac{1}{\pi^2} \cdot (1 - \cos(\pi u)) & \text{, falls} \quad |u| \leq 1 \\ \frac{2}{\pi^2} & \text{, falls} \quad |u| > 1 \end{cases} .$$

Die zugehörige ψ-Funktion ist gegeben durch

$$\psi(u) = \begin{cases} \frac{1}{\pi} \sin(\pi u) & \text{, falls} \quad |u| \leq 1 \\ 0 & \text{, falls} \quad |u| > 1 \end{cases}$$

und eliminiert ebenfalls Ausreißer vollständig. Hier wird als Skalierungsschätzer ebenfalls $s_n = \text{MAD}$ empfohlen und C sollte im Bereich $[1,5\pi;\ 2,4\pi]$ liegen.

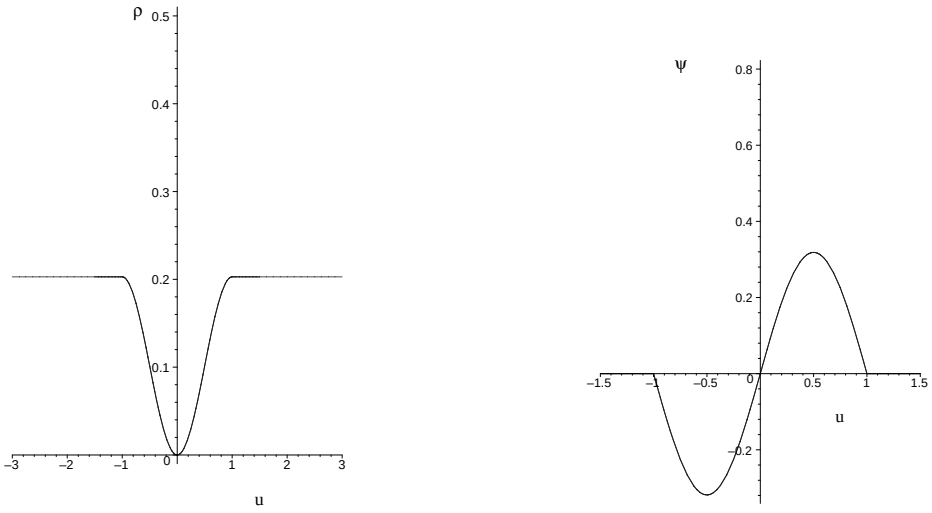


Abbildung 17: Andrew's wave $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.6 Tukey's biweight

Zu dem zu minimierenden Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

mit

$$\rho(u) = \begin{cases} \frac{1}{6} \cdot (1 - (1 - u^2)^3) & \text{, falls} \quad |u| \leq 1 \\ \frac{1}{6} & \text{, falls} \quad |u| > 1 \end{cases}$$

lautet das zugehörige ψ:

$$\psi(u) = \begin{cases} u(1-u^2)^2 & \text{, falls} \quad |u| \leq 1 \\ 0 & \text{, falls} \quad |u| > 1 \end{cases} .$$

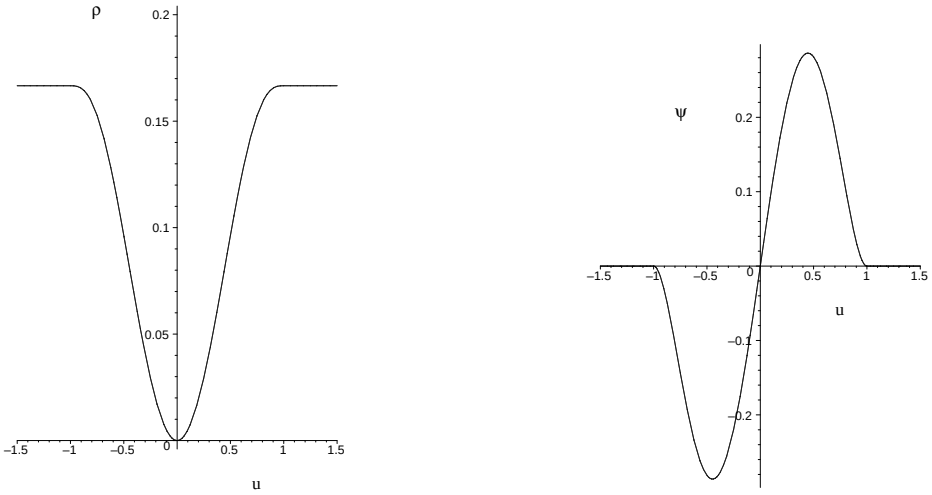


Abbildung 18: Tukey's biweight $\rho(u)$ (links) und $\psi(u)$ (rechts).

Der MAD wird als Skalierungsschätzer s_n verwendet, und für C empfehlen sich Werte aus dem Intervall $[6; 12]$. Auch hier werden Ausreißer mit einem zu großen Abstand zum Zentrum der Datenpunkte eliminiert. Für kleine u gilt $\psi(u) \approx u$. Daher verhält sich dieser Schätzer in der Nähe des Datenzentrums ähnlich dem arithmetischen Mittel.

Von diesen sechs Schätzern wurden der Huber-Schätzer, Andrew's wave und Tukey's biweight in der GALA-Software implementiert und dienen dort der Berechnung von robusten Lagemaßzahlen. Auf der Grundlage dieser robusten Schätzungen können dann verlässliche Aussagen über das Zentrum der Daten gemacht werden.

8.5.3 Diskussion der Eigenschaften verschiedener M-Schätzer

Die Robustheitseigenschaften der im vorangegangenen Abschnitt vorgestellten M-Schätzer kann man anhand der ψ-Funktion diskutieren. Folgende Eigenschaften sollte die ψ-Funktion eines robusten Lage-M-Schätzers einer symmetrischen Verteilung besitzen:

(a) hoher Bruchpunkt,

(b) ψ ist beschränkt,

(c) ψ besitzt „annähernd" stetiges Verhalten,

(d) ψ besitzt endlichen Ablehnungspunkt,

(e) $\psi(u) \approx k \cdot u, k \neq 0$ für kleine Werte von u und

(f) $\psi(-u) = -\psi(u)$.

Für die wichtigsten sechs M-Schätzer fasst Tabelle 5 die geforderten Eigenschaften zusammen:

Schätzer	a	b	c	d	e	f
Mittelwert	nein	nein	ja	nein	ja	ja
Median	ja	ja	nein	nein	nein	ja
Huber	ja	ja	ja	nein	ja	ja
Hampel	ja	ja	ja	ja	ja	ja
Andrew‘s wave	ja	ja	ja	ja	ja	ja
Tukey‘s biweight	ja	ja	ja	ja	ja	ja

Tabelle 5: Eigenschaften der wichtigsten Schätzer.

Die Spaltenbezeichnungen beziehen sich auf die Aufzählung zu Beginn dieses Abschnittes.
Die Tuning-Konstante k beim Huber-Schätzer kann bei bekanntem Ausreißeranteil ε folgendermaßen bestimmt werden:

$$\frac{2\phi(k)}{k} - 2\Phi(-k) = \frac{\varepsilon}{1-\varepsilon} \quad ,$$

wobei Φ die Standard-Normalverteilungsfunktion bezeichnet und ϕ die zugehörige Dichtefunktion. Die Tuning-Konstanten bei der Familie der Hampel-Schätzer können auf analoge Weise bestimmt werden [8]. Möchte man die im vorangegangenen Abschnitt vorgestellten M-Schätzer mit den L-Schätzern aus Abschnitt 8.4 vergleichen, so fällt auf, dass die M-Schätzer in der Regel einen hohen Bruchpunkt nahe bei $\varepsilon^* = 0,5$ haben und eine hohe Effizienz in einer Umgebung der Modellverteilung aufweisen [9]. L-Schätzer hingegen besitzen in diesem Falle eine geringere Effizienz oder einen kleineren Bruchpunkt.

8.5.4 Berechnung von M-Schätzern

8.5.4.1 Berechnung mittels Newton-Raphson-Verfahren

Zur Berechnung eines Lage-M-Schätzers ist T_n als Lösung der Gleichung

$$\sum_{i=1}^{n} \psi\left(\frac{x_i - T_n}{C \cdot S_n}\right) = 0$$

zu bestimmen. Im Allgemeinen ist diese Lösung jedoch nicht explizit angebbar. Einzige Ausnahme stellt der Fall des arithmetischen Mittels dar mit $\psi(x_i, T_n) = x_i - T_n$. In allen anderen Fällen muss daher mit Hilfe eines geeigneten Iterationsverfahrens eine Lösung gesucht werden. Als geeignet hat sich das Newton-Raphson-Verfahren erwiesen. Dabei wird

eine Nullstelle einer Funktion $h(t)$ gesucht. Diese Nullstelle wird durch Auswertungen von h und h' an einer Näherungsstelle $t^{(k)}$ bestimmt. $t^{(k+1)}$ ist dann die Stelle, an der die Tangente an h im Punkt $t^{(k)}$ die t-Achse schneidet. Bei der Berechnung eines M-Schätzers lautet die Funktion h:

$$h(T_n) = \sum_{i=1}^{n} \psi\left(\frac{x_i - T_n}{C \cdot S_n}\right) \quad .$$

Für die Iterationsvorschrift ergibt sich daraus

$$\begin{aligned} T_n^{(k+1)} &= T_n^{(k)} - \frac{h(T_n^{(k)})}{h'(T_n^{(k)})} \\ &= T_n^{(k)} + C \cdot S_n \frac{\sum_{i=1}^{n} \psi(u_i^{(k)})}{\sum_{i=1}^{n} \psi'(u_i^{(k)})} \quad . \end{aligned}$$

Als Startwert sollte für $T_n^{(0)}$ der Median als erste robuste Näherung gewählt werden Der gesuchte M-Schätzer ergibt sich dann als Grenzwert der Folge $(T_n^{(k)})_{k\geq 1}$. Das Ergebnis ist in den meisten Fällen ein robuster Schätzer.

8.5.4.2 1-Schritt-Verfahren

Führt man beim Newton-Raphson-Verfahren nur einen einzigen Iterationsschritt aus, so wird das daraus resultierende Ergebnis für den gesuchten Schätzer 1-Schritt-M-Schätzer genannt. Bei diesem Vorgehen ist offensichtlich die Wahl des Startwertes für $T_n^{(0)}$ sehr wichtig. Falls als Startwert der arithmetische Mittelwert benutzt wird, ist das Ergebnis nach einem Iterationsschritt unbefriedigend. Es sollte daher der Median als Anfangswert genutzt werden, damit das Ergebnis einen robusten M-Schätzer liefert.

8.5.5 M-Schätzer für Streuungsparameter

Grundlage sei eine Verteilungsfamilie mit Dichten $\frac{1}{\sigma} f_0\left(\frac{x-\mu}{\sigma}\right) = f(x;\mu,\sigma)$ mit Lageparameter μ und Skalierungsparameter σ. Wendet man das Prinzip der Maximum-Likelihood-Schätzung auf diesen Fall an, so ergibt sich folgendes zu lösende Problem:

$$\sum_{i=1}^{n} -\log\left(\frac{1}{\sigma} f_0\left(\frac{x_i - \mu}{\sigma}\right)\right) \to \min_{\mu,\sigma} \quad .$$

Aufgrund der Rechenregeln für die Logarithmusfunktion ergibt sich daraus

$$\sum_{i=1}^{n} \Bigg(\log(\sigma) + \underbrace{\left(-\log\left(f_0\left(\frac{x_i - \mu}{\sigma}\right)\right)\right)}_{=:\rho_{f_0}\left(\frac{x_i-\mu}{\sigma}\right)} \Bigg) \to \min_{\mu,\sigma} \quad .$$

Zur Bestimmung des Minimums werden nun die partiellen Ableitungen nach μ und σ gebildet, diese gleich Null gesetzt und das so entstandene System der Normalengleichungen gelöst, wobei $\psi_f = \rho'_f$ ist. Man erhält

$$\sum_{i=1}^{n} \frac{1}{\sigma} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \left(\frac{1}{\sigma} - \frac{x_i - \mu}{\sigma^2} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) \right) = 0$$

und daraus ergibt sich

$$\sum_{i=1}^{n} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \left(\frac{x_i - \mu}{\sigma} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) - 1 \right) = 0 \quad .$$

Um nun simultane M-Schätzer für Lage und Streuung zu erhalten, wird dieses zu lösende System verallgemeinert zu

$$\sum_{i=1}^{n} \psi \left(\frac{x_i - T}{S} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \chi \left(\frac{x_i - T}{S} \right) = 0 \quad .$$

Im Allgemeinen ist dabei ψ eine ungerade und χ eine gerade Funktion. In Funktionalschreibweise ergeben sich damit folgende Integralgleichungen:

$$\int \psi \left(\frac{x - T}{S} \right) \, dF(x) = 0 \quad \wedge \quad \int \chi \left(\frac{x - T}{S} \right) \, dF(x) = 0 \quad .$$

Die Schätzer für den Lageparameter und den Streuungsparameter werden durch simultanes Lösen dieser beiden Gleichungen bestimmt.

8.5.5.1 Huber-Schätzer für Lage und Streuung

Mit

$$\psi(x) = \max(-k, \min(k, x))$$

und

$$\chi(x) = \min(c^2, x^2) - \beta \quad ,$$

wobei $\beta = \beta(c) = \int \min(c^2, x^2) \; \phi(x) \; dx$ und $0 < \beta < c^2$, ergibt sich ein unter Normalverteilung konsistenter Schätzer für die Streuung mit $\phi(\cdot)$ als Dichtefunktion der Standardnormalverteilung.

8.5.5.2 Median und MAD

Für

$$\psi(x) = \text{sgn}(x)$$

und

$$\chi(x) = \text{sgn}(|x| - 1)$$

ergeben sich der Median als Schätzer für den Lageparameter und der MAD als Schätzer für die Streuung.

8.5.5.3 A-Schätzer

Eine weitere Möglichkeit, einen robusten Schätzer für die Varianz bzw. Streuung einer Stichprobe zu erhalten, ist die Verwendung von sogenannten A-Schätzern.
Die A-Schätzer

$$s_T = \frac{C \cdot \mathrm{MAD}\sqrt{n} \cdot \sqrt{\sum_{i=1}^{n} \psi(u_i)^2}}{|\sum_{i=1}^{n} \psi'(u_i)|}$$

erhält man aus der asymptotischen Varianz der M-Lageparameterschätzer (s. Abschnitt 8.5.2) mit $u_i = \frac{x_i - m_n}{C \cdot \mathrm{MAD}}$.
Setzt man

$$\psi(u) = \begin{cases} \sin(\pi u) & \text{, falls} \quad |u| < 1 \\ 0 & \text{, falls} \quad |u| \geq 1 \end{cases} ,$$

so führt dies zu dem Skalenschätzer

$$s_{wa} = \frac{(C \cdot \mathrm{MAD})\sqrt{n}[\sum_{|u_i|<1} \sin^2(\pi u_i)]^{1/2}}{\pi|\sum_{|u_i|<1} \cos(\pi u_i)|} \quad , \tag{25}$$

der auf Andrew‘s wave Schätzer für einen Lageparameter (s. Seite 81) basiert.
Ein weiteres Beispiel zeigt einen A-Schätzer für die Streuung, der auf dem Biweight-Lageschätzer (s. Seite 81) beruht. Sei

$$\psi(u) = \begin{cases} u(1-u^2)^2 & \text{, falls} \quad |u| < 1 \\ 0 & \text{, falls} \quad |u| \geq 1 \end{cases} ,$$

dann erhält man

$$s_{bi} = \frac{\sqrt{n}\left[\sum_{|u_i|<1}(x_i - m_n)^2(1-u_i^2)^4\right]^{1/2}}{|\sum_{|u_i|<1}(1-u_i^2)(1-5u_i^2)|} \tag{26}$$

als Skalenschätzer. Die Skalenschätzer (25) und (26) sind in der GALA-Software implementiert worden.

8.6 Robuste Schätzer für multivariate Lage und Streuung

8.6.1 Einleitung und Motivation

Bisher lagen n Beobachtungen aus einer Beobachtungsreihe vor, für die in der Regel *ein* Lageparameter und *ein* Parameter für die Streuung zu schätzen waren. Im vorliegenden

Abschnitt seien nun mehrere Beobachtungsreihen gegeben, d.h. es seien $x^{(1)}, \ldots, x^{(n)}$ p-variate Beobachtungsdaten. Die robuste Schätzung eines Lageparameters für diesen multivariaten Fall führt daher zur Bestimmung eines robusten Lagevektors. Ein multivariater Lagevektor kann auf zwei unterschiedliche Arten geschätzt werden. Zum einen kann man komponentenweise für jede Beobachtungsreihe einen einzelnen robusten Lageschätzer bestimmen. Dazu eignen sich die in Abschnitt 8.4 und Abschnitt 8.5 vorgestellten L- bzw. M-Schätzer. Eine weitere Möglichkeit besteht in der multivariaten, simultanen Schätzung des Lagevektors, bei der alle Beobachtungen gemeinsam eingehen. Dieses Vorgehen stellt eine Erweiterung der M-Schätzer für den multivariaten Fall dar. Im Abschnitt 8.6.3 werden dazu M-Schätzer für multivariate Lage und Streuung betrachtet. Die Schätzung der Streuung läuft auf die Berechnung von Kovarianz- bzw. Korrelationsmatrizen hinaus. Auf der Grundlage der Matrizen, die sich aus diesen Kovarianzen bzw. Korrelationen aufbauen, wird zum Beispiel die Hauptkomponentenanalyse (s. Abschnitt 6) oder die Faktoranalyse durchgeführt. Darüber hinaus dienen Kovarianz- und Korrelationsmatrizen dem Test auf Unabhängigkeit. Die entsprechenden empirischen Matrizen sind jedoch sehr ausreißerempfindlich, so dass robusten Methoden eine große Bedeutung zukommt.
Für zwei gegebene Zufallsgrößen X und Y sind die Kovarianz $\text{Kov}(X, Y)$ und die Korrelation $\text{Korr}(X, Y) = \rho(X, Y)$ gemäß (1) und (2) gegeben. Seien nun $X_1, \ldots, X_n$ Zufallsvariablen. Dann ist die zugehörige Kovarianzmatrix definiert als

$$\Sigma = \begin{pmatrix} \text{Var}X_1 & \text{Kov}(X_1, X_2) & \ldots & \text{Kov}(X_1, X_n) \\ \text{Kov}(X_2, X_1) & \text{Var}X_2 & \ldots & \text{Kov}(X_2, X_n) \\ \vdots & \ddots & \ddots & \vdots \\ \text{Kov}(X_n, X_1) & \ldots & \text{Kov}(X_n, X_{n-1}) & \text{Var}X_n \end{pmatrix},$$

und als Korrelationsmatrix wird

$$R = \begin{pmatrix} 1 & \text{Korr}(X_1, X_2) & \ldots & \text{Korr}(X_1, X_n) \\ \text{Korr}(X_1, X_2) & 1 & \ldots & \text{Korr}(X_2, X_n) \\ \vdots & \ddots & \ddots & \vdots \\ \text{Korr}(X_1, X_n) & \ldots & \text{Korr}(X_{n-1}, X_n) & 1 \end{pmatrix}$$

bezeichnet.
Bei der Berechnung robuster Kovarianz- bzw. Korrelationsmatrizen unterscheidet man zwei Vorgehensweisen. Zum einen kann man jeden Eintrag in der Kovarianz- bzw. Korrelationsmatrix separat robust schätzen. Dieses Vorgehen führt zu univariaten Analysen für die Varianzen und bivariaten Analysen für die Kovarianzen bzw. Korrelationen. Dieses Verfahren hat aber den Nachteil, dass die sich ergebenden Matrizen im Allgemeinen nicht positiv semidefinit sind. Die zweite Vorgehensweise besteht in der simultanen Schätzung aller Elemente der Kovarianzmatrix und führt zu einer multivariaten Analyse. Die sich aus

dieser Methode ergebenden Matrizen sind positiv semidefinit. Die Methode der separaten Schätzung bietet jedoch Vorteile, wenn ein größerer Anteil der Beobachtungen fehlt.

8.6.2 Separate robuste Schätzer für die Elemente der Kovarianz- bzw. Korrelationsmatrix

Ein einfacher Ansatz zur Schätzung der Kovarianz zwischen zwei Zufallsvariablen X und Y beruht auf der Gleichung

$$\text{Kov}(X,Y) = \frac{1}{4}[\text{Var}(X+Y) - \text{Var}(X-Y)] \quad .$$

Die Schätzung der Kovarianz kann somit zurückgeführt werden auf die Schätzung von Varianzen. Auf der Grundlage der robusten univariaten Streuungsschätzer aus den Abschnitten 8.4.2 und 8.5.5 ist es also möglich, robuste Schätzer für Kovarianz- und Korrelationsmatrix anzugeben.
Einen robusten Schätzer s^*_{xy} für die Kovarianz zwischen X und Y erhält man mittels

$$s^*_{xy} = \frac{1}{4}[\hat{\sigma}^{*2}_1 - \hat{\sigma}^{*2}_2] \quad ,$$

wobei $\hat{\sigma}^*_1$ und $\hat{\sigma}^*_2$ robuste Skalenschätzer für $X+Y$ und $X-Y$ sind. Damit erhält man auch eine robuste Schätzung der Korrelation zwischen X und Y:

$$r^*_{xy} = \frac{s^*_{xy}}{\sqrt{s^*_{xx} \cdot s^*_{yy}}} \quad ,$$

wobei s^*_{xx} und s^*_{yy} robuste Schätzungen der Varianzen von X und Y sind.
Berechnet man für zwei Zufallsgrößen X und Y die herkömmliche Korrelation

$$r_{xy} = \frac{s_{xy}}{s_x s_y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x}_n)^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \bar{y}_n)^2}} \quad ,$$

so liegt der Wert aufgrund der Gültigkeit der Cauchy-Schwarz-Ungleichung immer im Intervall $[-1;1]$. Da aber r^*_{xy} nicht auf den normalen Standardabweichungen, sondern auf robusten Schätzungen für die Varianzen basiert, muss r^*_{xy} nicht zwangsläufig im Intervall $[-1;1]$ liegen. Um diese Eigenschaft der Korrelation sicher zu stellen, wendet man eine Modifikation an. Seien

$$Z_x = \frac{X}{\sqrt{s^*_{xx}}} \quad \text{bzw.} \quad Z_y = \frac{Y}{\sqrt{s^*_{yy}}}$$

die standardisierten Formen von X und Y mit robusten Schätzungen der Varianz s^*_{xx} bzw. s^*_{yy}. Definiert man jetzt

$$\tilde{r}^*_{xy} = \frac{\hat{\sigma}^{*2}_3 - \hat{\sigma}^{*2}_4}{\hat{\sigma}^{*2}_3 + \hat{\sigma}^{*2}_4} \quad , \tag{27}$$

wobei $\hat{\sigma}_3^{*2}$ und $\hat{\sigma}_4^{*2}$ robuste Schätzungen der Varianzen von $Z_1 + Z_2$ bzw. $Z_1 - Z_2$ sind, so ergibt sich, dass $\tilde{r}_{xy}^*$ im Intervall $[-1; 1]$ liegt. Die robusten Varianzschätzungen sind im Allgemeinen nicht konsistent. Eine konsistente Schätzung der Varianz bei Normalverteilung erhält man durch geeignete Normierungskonstanten. Diese müssen bei der Berechnung der Korrelationen jedoch nicht berücksichtigt werden, da sie sich herauskürzen. Aus Gleichung (27) lässt sich leicht auch der entsprechende robuste Kovarianzschätzer herleiten:

$$\tilde{s}_{xy}^* = \tilde{r}_{xy}^* \cdot \sqrt{s_{xx}^* s_{yy}^*} \quad .$$

Dieses Verfahren wurde unter Verwendung verschiedener robuster Varianzschätzer implementiert.

8.6.3 M-Schätzer für multivariate Lage- und Skalierungsparameter

Wie im univariaten Fall ergeben sich die M-Schätzer durch Verallgemeinerung der Maximum-Likelihood-Schätzung. Zunächst wird die Klasse der Normalverteilungen, für die die zugehörigen Maximum-Likelihood-Schätzer nicht robust sind, zu einer Verteilungsklasse erweitert, die auch Verteilungsfamilien mit stärker besetzten Tails enthält, deren Maximum-Likelihood-Schätzer robust sind.
Ein p-dimensionaler Zufallsvektor X mit der Verteilungsdichte

$$f(x) = (\det V)^{-\frac{1}{2}} h\left((x-t)^\top V^{-1}(x-t)\right) \quad ,$$

wobei h eine nichtnegative reellwertige Funktion, $t \in \mathbb{R}^p$ und V eine positiv definite Matrix ist, heißt elliptisch symmetrisch verteilt.
Mit

$$h(u) = \frac{1}{(2\pi)^{p/2}} \exp(-\frac{u}{2})$$

ergibt sich die Normalverteilung und mit

$$h(u) = \frac{\Gamma(\frac{\nu+p}{2})}{(\pi\nu)^{p/2}\Gamma(\nu/2)} \cdot \frac{1}{(1+\frac{1}{\nu}u)^{(\nu+p)/2}} \tag{28}$$

die Familie der p-dimensionalen t-Verteilung mit ν Freiheitsgraden. Wie im univariaten Fall sind die Tails der multivariaten t-Verteilung stärker besetzt als die der Normalverteilung. Die auf der t-Verteilung basierende Maximum-Likelihood-Schätzung gewichtet stark abweichende Beobachtungen schwächer als die auf der Normalverteilung basierende Maximum-Likelihood-Schätzung.
Die Maximum-Likelihood-Schätzung für eine elliptisch symmetrische Verteilung ergibt sich durch Maximierung der Likelihood-Funktion

$$\prod_{i=1}^{n} f(x^{(i)}) = \prod_{i=1}^{n} \frac{1}{\sqrt{\det(V)}} h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)$$

bezüglich t und V, wobei mit $x^{(i)}$ der Vektor der i-ten Beobachtungsreihe bezeichnet wird, d.h.

$$x^{(i)} = \left(x_1^{(i)}, \ldots, x_p^{(i)}\right)^\top \quad , \quad 1 \leq i \leq n \quad .$$

Logarithmiert man die Likelihood-Funktion und maximiert bzgl. V^{-1} statt V, so ergibt sich die äquivalente Aufgabe

$$\log \prod_{i=1}^{n} f(x^{(i)}) = \frac{n}{2} \log(\det V^{-1}) + \sum_{i=1}^{n} \log\left(h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)\right) \to \max_{t,V^{-1}} \quad .$$

Zur Bestimmung des Maximums werden nun die beiden partiellen Ableitungen nach t und V^{-1} gebildet:

$$\frac{\partial}{\partial t}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \sum_{i=1}^{n} \frac{-2h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x^{(i)}-t)^\top \cdot V^{-1}$$

$$\frac{\partial}{\partial v^{kl}}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \frac{n}{2} \underbrace{\frac{\frac{\partial}{\partial v^{kl}} \det(V^{-1})}{\det(V^{-1})}}_{=\frac{\det(V^{kl})}{\det(V^{-1})} = v_{lk}} + \sum_{i=1}^{n} \frac{h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x_k^{(i)}-t_k)(x_l^{(i)}-t_l) \quad ,$$

wobei v^{kl} das (k,l)-te Element der inversen Matrix V^{-1} bezeichnet und V^{kl} die Matrix ist, die aus der Matrix V entsteht, indem das (k,l)-te Element gleich Eins gesetzt wird und die restlichen Elemente der k-ten Zeile und l-ten Spalte gleich Null gesetzt werden. Damit ergibt sich insgesamt für die partielle Ableitung nach der inversen Matrix V^{-1}:

$$\frac{\partial}{\partial V^{-1}}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \frac{n}{2}V + \sum_{i=1}^{n} \frac{h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x^{(i)}-t)(x^{(i)}-t)^\top \quad .$$

Beide partiellen Ableitungen werden anschließend gleich Null gesetzt, wobei als abkürzende Schreibweisen

$$d_i^2 = (x^{(i)}-t)^\top V^{-1}(x^{(i)}-t) \quad ,$$

$$w(d_i^2) = \frac{-2h'(d_i^2)}{h(d_i^2)} \tag{29}$$

verwendet werden.

Damit ergibt sich folgendes Gleichungssystem:

$$\begin{aligned} \sum_{i=1}^{n} w(d_i^2)(x^{(i)} - t) &= 0 \\ \sum_{i=1}^{n} \left(w(d_i^2)(x^{(i)} - t)(x^{(i)} - t)^\top - V \right) &= 0 \quad . \end{aligned}$$

Durch Verallgemeinerung dieses Gleichungssystems ergeben sich die M-Schätzer für multivariate Lage und Streuung. Die Lösungen des Gleichungssystems

$$\sum_{i=1}^{n} w_1(d_i^2)(x^{(i)} - t) = 0 \tag{30}$$

$$\sum_{i=1}^{n} \left(w_2(d_i^2)(x^{(i)} - t)(x^{(i)} - t)^\top - v(d_i^2)V \right) = 0 \tag{31}$$

heißen M-Schätzer für den multivariaten Lageparameter und die Kovarianzmatrix.

Verfahren	**Korrelation**	**Ausreißer**	**Tuning-Parameter**
einfache Korrelation	-0,35	keiner	
	-0,81	(1; 105)	
	-0,88	(1; 105), (10; 10)	
	-0,89	(1; 105), (10; 10), (14; 1)	
0,01-getrimmt	-0,81	(1; 105), (10; 10)	
0,05-getrimmt	-0,90	(1; 105), (10; 10), (14; 1)	
0,01-winsorisiert	-0,75	(1; 105), (10; 10)	
0,05-winsorisiert	-0,90	(1; 105), (10; 10), (14; 1)	
Tukey‘s biweight	-0,90	(1; 105), (10; 10), (14; 1)	9
(A-Schätzer)	-0,91	(1; 105), (10; 10), (14; 1)	6
Andrew‘s wave	-0,90	(1; 105), (10; 10), (14; 1)	6,6
(A-Schätzer)			
Huber	-0,91	(1; 105), (10; 10), (14; 1)	1,5
(A-Schätzer)			
Huber	-0.89	(1; 105), (10; 10), (14; 1)	
(Simultan)			

Tabelle 6: Daten zur Korrelation bei verschiedenen M-Schätzern.

Hieraus ergeben sich die Maximum-Likelihood-Schätzer der p-dimensionalen t-Verteilung direkt aus (28) und (29):

$$w_1(d) = \frac{m+\nu}{\nu+d^2} = w_2(d^2) \quad , \quad v(d_i^2) = 1 \quad ,$$

während bei den auf Hubers ψ-Funktionen basierenden Schätzern

$$w_1(d) = \begin{cases} 1 & \text{, falls} \quad d \leq a \\ \frac{a}{d} & \text{, falls} \quad d > a \end{cases} \quad ,$$
$$w_2(d^2) = w_1(d)^2 \quad \text{und} \quad v(d_i^2) = c$$

gelten, wobei c ein Korrekturterm für die asymptotische Erwartungstreue unter der Normalverteilung ist und gemäß [16], S. 225 berechnet werden kann.
Das folgende Beispiel liefert anhand eines Testdatensatzes einen Überblick über robuste und "konventionelle" Korrelationsberechnungen. Die für dieses Beispiel genutzten Daten werden in Tabelle 7 im Anhang angegeben. Aus den zunächst 109 Datenpaaren werden drei Beobachtungswerte als Ausreißer identifiziert und nacheinander eliminiert (Abbildung 19). Die Angaben zur Korrelation in Tabelle 6 zeigen deutlich, dass bei der herkömmlichen Korrelationsberechnung mit den drei Ausreißern der Wert keine Aussage über eine Beziehung zulässt. Die Werte, die sich bei Benutzung der robusten Verfahren ergeben, ähneln sich und zeigen eine Korrelation von ca. $0,9$.

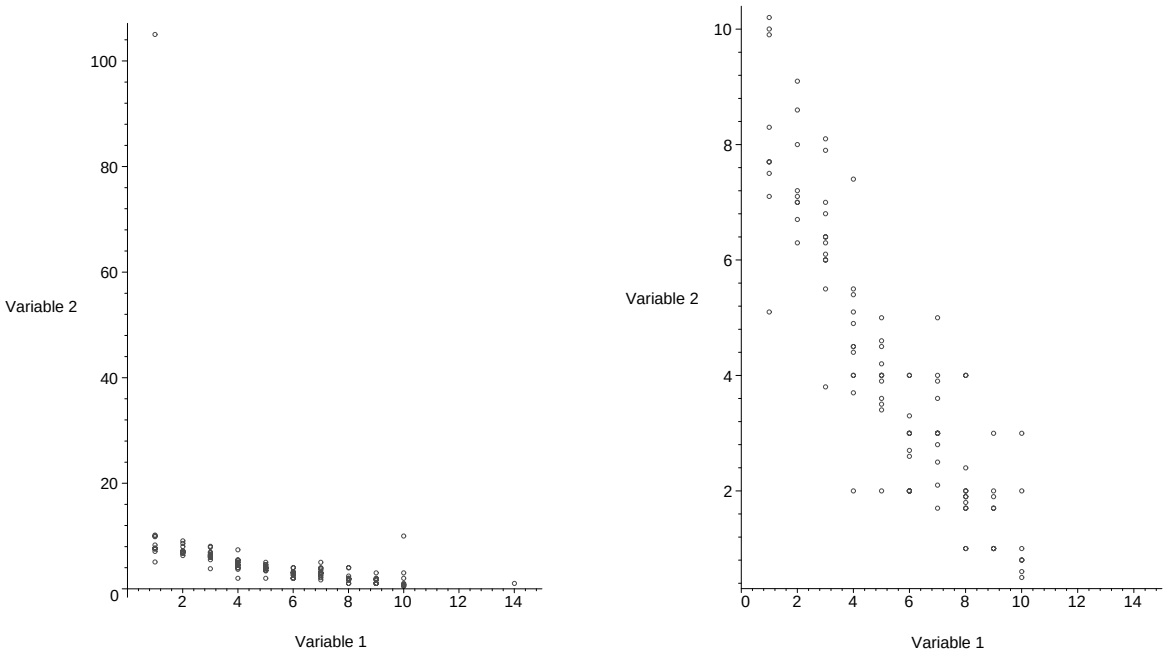


Abbildung 19: Streubild zweier Variablen: 109 Datenpaare (links), 106 Datenpaare (rechts).

8.7 Robuste Hauptkomponentenanalyse

Ausgehend von den allgemeinen Bemerkungen zur Hauptkomponentenanalyse aus Abschnitt 6 werden in der Arbeit von Jackson [10] drei unterschiedliche Methoden kurz vorgestellt, die zur Durchführung der robusten Hauptkomponentenanalyse verwendet werden können.

- Eliminiere zunächst Ausreißer aus den Rohdaten und führe anschließend eine konventionelle Hauptkomponentenanalyse auf den verbliebenen Daten durch.
- Benutze eine robuste Schätzung der Kovarianz- bzw. Korrelationsmatrix und führe eine konventionelle Hauptkomponentenanalyse durch.
- Verwende robuste Schätzungen der Eigenwerte und Eigenvektoren von Kovarianzmatrizen.

Ein Verfahren zur Bestimmung robuster Hauptkomponenten auf der Grundlage robuster Schätzungen der Eigenwerte und Eigenvektoren wird in [2] angegeben und soll im Folgenden vorgestellt werden. Die Berechnung erfolgt ebenfalls iterativ nach folgender Iterationsvorschrift:

(1) Berechne Startschätzung V für die Kovarianzmatrix und berechne den ersten Eigenvektor u_1.

(2) Initialisiere $w_m \equiv 1$.

(3) Bestimme die Hauptkomponenten-Scores $y_m = u_1^\top x_m$.

(4) Berechne M-Schätzungen für den Mittelwert und die Varianz zu y_m (wie in Abschnitt 8.6) iterativ:

$$\bar{y} = \frac{\sum w_m^{(\text{neu})} y_m}{\sum w_m^{(\text{neu})}}$$

$$s^2 = \frac{\sum \left(w_m^{(\text{neu})}\right)^2 (y_m - \bar{y})^2}{\sum \left(w_m^{(\text{neu})}\right)^2 - 1}$$

mit

$$w_m^{(\text{neu})} = w_m^{(\text{neu})}(d_m) = \begin{cases} 1 & \text{, falls} \quad d_m \le d_0 \\ \frac{d_0}{d_m} \exp\left(\frac{-(d_m - d_0)^2}{2 \cdot 1{,}25^2}\right) & \text{, falls} \quad d_m > d_0 \end{cases}$$

und

$$d_m^2 = \frac{(y_m - \bar{y})^2}{s^2}, \qquad d_0 = \sqrt{v} + \sqrt{2} \quad .$$

Als Startwerte werden der Median und $(0,74 \cdot \text{IQR})^2$ empfohlen.

(5) Setze $w_m = \min\{w_m, w_m^{(\text{neu})}\}$ und berechne $\bar{x}$ und V mit dem gerade bestimmten w_m

$$\bar{x} = \frac{\sum w_m x_m}{\sum w_m} \quad ,$$

$$V = \frac{\sum w_m^2 (x - \bar{x})(x - \bar{x})^\top}{\sum w_m^2 - 1} \quad .$$

(6) Berechne den ersten Eigenwert e_1 und den zugehörigen Eigenvektor u_1 von V.

(7) Wiederhole die Schritte (3) - (6) so lange, bis Konvergenz vorliegt.

Zur Bestimmung der weiteren Hauptkomponenten $u_i, 2 \leq i \leq p$, muss man die Daten auf den Raum orthogonal zu den bisherigen Eigenvektoren $u_1, \ldots, u_{i-1}$ projizieren und anschließend die Schritte (2) bis (7) wiederholen. Als Erweiterung ergibt sich hierfür:

(8) Bilde $x_{i,m} = (I - U_{i-1}U_{i-1}^\top)x_m$, wobei $U_{i-1} = (u_1, \ldots, u_{i-1})$.
Wähle als Startwert für den ersten Eigenvektor den zweiten Eigenvektor der letzten Iteration für den vorherigen Eigenvektor.

(9) Wiederhole die Schritte (2) bis (7) mit $x_{i,m}$ anstelle von x_m und bestimme den ersten Eigenvektor u, der dann u_i ist.

Die Schritte (7) bis (9) werden so lange wiederholt, bis alle p Eigenwerte e_i und Eigenvektoren u_i mit entsprechenden Gewichten bestimmt sind.
Schließlich kann eine robuste Schätzung der Kovarianz- bzw. Korrelationsmatrix durch $UEU^\top$ alternativ zu V gefunden werden, die positiv semidefinit ist. Dabei ist U die Matrix, die sich spaltenweise aus den Eigenvektoren u_i zusammensetzt und E eine Diagonalmatrix mit den Eigenwerten e_i als Einträge auf der Hauptdiagonalen.

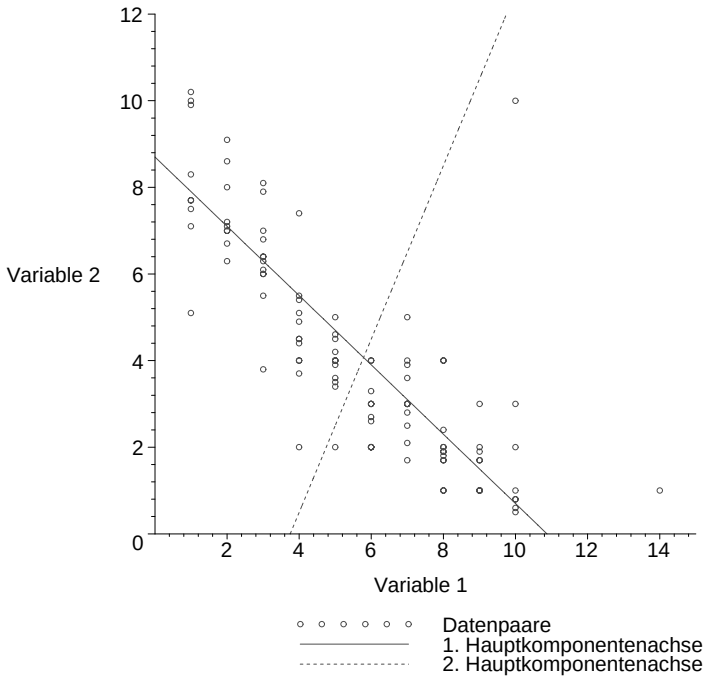


Abbildung 20: 108 von 109 Beobachtungen zweier Variablen und beide Hauptkomponentenachsen.

Für den Datensatz aus Tabelle 7 im Anhang mit 109 Beobachtungen (das Datenpaar $(1,0; 105,0)$ wird in der Grafik nicht berücksichtigt) ergibt sich nach Berechnung der robusten Hauptkomponentenanalyse die Situation wie in Abbildung 20 dargestellt. Das Verfahren zur robusten Hauptkomponentenanalyse nach Campbell wurde ebenfalls in der GALA-Software implementiert und kann als Alternative zur gewöhnlichen Hauptkomponentenanalyse mittels eines Optionsschalters beim Programmaufruf gewählt werden.

Anhang

Daten zur Korrelationsberechnung

Nr.	Var 1	Var 2	Nr.	Var 1	Var 2	Nr.	Var 1	Var 2	Nr.	Var 1	Var 2
1	6,0	2,0	29	4,0	7,4	57	10,0	0,8	85	14,0	1,0
2	7,0	3,0	30	9,0	1,0	58	4,0	4,5	86	10,0	2,0
3	6,0	3,0	31	6,0	2,0	59	4,0	4,5	87	10,0	3,0
4	3,0	6,0	32	6,0	2,0	60	9,0	1,7	88	9,0	2,0
5	4,0	4,9	33	6,0	4,0	61	2,0	7,0	89	4,0	5,5
6	6,0	4,0	34	10,0	1,0	62	1,0	7,7	90	8,0	1,8
7	5,0	5,0	35	1,0	105,0	63	3,0	6,0	91	2,0	8,0
8	8,0	1,9	36	7,0	1,7	64	4,0	2,0	92	8,0	1,7
9	8,0	1,9	37	6,0	3,0	65	4,0	5,4	93	7,0	5,0
10	7,0	2,1	38	4,0	5,1	66	7,0	2,8	94	3,0	7,9
11	5,0	4,2	39	7,0	3,0	67	4,0	3,7	95	2,0	9,1
12	4,0	4,0	40	2,0	8,6	68	5,0	3,6	96	2,0	7,1
13	2,0	7,2	41	3,0	6,3	69	6,0	2,0	97	6,0	3,3
14	1,0	8,3	42	3,0	6,4	70	10,0	10,0	98	5,0	4,5
15	1,0	10,0	43	3,0	5,5	71	5,0	4,0	99	2,0	6,7
16	2,0	7,0	44	1,0	7,1	72	7,0	3,0	100	5,0	4,6
17	2,0	6,3	45	9,0	1,7	73	10,0	0,8	101	10,0	0,6
18	3,0	7,0	46	5,0	2,0	74	4,0	4,4	102	9,0	1,9
19	5,0	4,0	47	3,0	6,8	75	9,0	3,0	103	3,0	8,1
20	1,0	9,9	48	9,0	1,0	76	8,0	2,0	104	5,0	3,4
21	8,0	1,0	49	8,0	4,0	77	7,0	2,5	105	1,0	10,2
22	8,0	2,0	50	9,0	1,0	78	3,0	3,8	106	1,0	7,7
23	8,0	1,0	51	6,0	2,6	79	8,0	1,7	107	1,0	5,1
24	4,0	4,0	52	5,0	3,5	80	9,0	1,0	108	1,0	7,5
25	7,0	3,0	53	6,0	2,7	81	6,0	3,0	109	3,0	6,4
26	3,0	6,1	54	7,0	3,6	82	5,0	4,0			
27	8,0	2,4	55	5,0	3,9	83	8,0	4,0			
28	10,0	0,5	56	7,0	3,9	84	7,0	4,0			

Tabelle 7: Datensatz (109 Beobachtungen)

Literaturverzeichnis

[1] V. Barnett and T. Lewis. *Outliers in statistical data.* John Wiley & Sons, New York, 1979.

[2] N. A. Campbell. *Robust procedures in multivariate analysis I: robust covariance estimation.* Journal of Applied Statistics 1980, 29:231–237.

[3] G. Dikta. *Operations research.* Unpublished script, FH Aachen, 2002.

[4] P. Érdi and J. Tóth. *Mathematical models of chemical reactions: theory and applications of deterministic and stochastic models.* Manchester University Press, 1989.

[5] F. R. Hampel.(1974) *The influence curve and its role in robust estimation.* Journal of the American Statistical Association 1974, 346:383–393.

[6] F. R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, W.A. Stahel. *Robust statistics: The approach based on influence functions.* John Wiley & Sons, New York, 1986.

[7] J. Hartung, B. Elpelt. *Statistik - Lehr- und Handbuch der angewandten Statistik.* R. Oldenbourg Verlag, München, 1995.

[8] D. C. Hoaglin, F. Mosteller, and J. W. Tukey (eds.). *Understanding robust and exploratory data analysis.* John Wiley & Sons, New York, 1983.

[9] P. J. Huber. *Robust statistics.* John Wiley & Sons, New York, 1981.

[10] J. E. Jackson. *A user's guide to principal components.* Wiley-Interscience, 1991.

[11] I. T. Jolliffe. *Discarding variables in a principal component analysis. I: Artificial data.* Applied Statistics 1972, 21:160–173.

[12] I. T. Jolliffe. *Principal component analysis.* Springer-Verlag, New York, 1986.

[13] E. L. Lehmann. *Nonparametrics: statistical methods based on ranks.* Holden-Day, Inc., San Francisco, 1975.

[14] J. Lehn, H. Wegmann. *Einführung in die Statistik.* Teubner Verlag, Stuttgart, 1992.

[15] G. P. McCabe. *Principal variables.* Technometrics 1984, 26:137–144.

[16] A. Marazzi. *Algorithms, routines, and S functions for robust statistics: the FORTRAN library ROBETH with an interface to S-Plus.* Wadsworth & Brooks/Cole Publishing Company, 1993.

[17] L. A. Segel. *Mathematical models in molecular and cellular biology.* Cambridge University Press, New York, 1980.

[18] R. J. Serfling. *Approximation theorems of mathematical statistics.* John Wiley & Sons, New York, 1980.

Klassifikationsalgorithmen für Daten aus der Pharmaindustrie

Tatjana Eitrich

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: t.eitrich@fz-juelich.de

1 Einleitung

Support-Vektor-Maschinen sind moderne Verfahren, welche in das Gebiet der Künstlichen Intelligenz, der Erforschung und Simulation von natürlichem intelligenten Verhalten, eingeordnet werden. Sie realisieren Vorgänge des Maschinellen Lernens, einem zur Zeit sehr bedeutenden Teilgebiet der Künstlichen Intelligenz. Maschinelles Lernen versucht Systeme zu entwickeln, die in der Lage sind, durch Benutzung von Eingabeinformationen neues Wissen zu konstruieren und vorhandenes Wissen zu verbessern. Dieses Forschungsgebiet findet immer stärkeren Zuspruch, da dadurch beispielsweise sehr aufwändige Experimente verkürzt werden können. Noch vor wenigen Jahrzehnten wurde diese Vorstellung kritisch betrachtet, weil nicht abzusehen war, dass sich die Rechenleistung von Computern so rasant entwickeln würde. Beispiele für Realisierungen von Maschinellem Lernen sind Algorithmen zur Handhabung binärer und multipler Klassifikationsaufgaben sowie zur Umsetzung von Regressionsverfahren.

Support-Vektor-Maschinen sind gekennzeichnet durch die Kombination von linearen Lernalgorithmen und hochdimensionalen kerninduzierten Abbildungen. Diese Kombination führt auf mächtige nichtlineare Entscheidungsverfahren. Geeignete Algorithmen zur Implementierung ergeben sich aus der statistischen Lerntheorie und der konvexen Optimierungstheorie. Support-Vektor-Maschinen zeichnen sich dadurch aus, dass sie globale Lösun-

gen erzeugen, was auf dem Gebiet des Maschinellen Lernens nicht selbstverständlich ist. Außerdem ist ihre Generalisierungsfähigkeit sehr gut, weil sie eine besonders günstige Form der Risikominimierung umsetzen. Ihre Anwendungsmöglichkeiten in der Praxis sind sehr vielfältig. Dazu zählen beispielsweise Bild- und Schrifterkennung, Umsatzprognosen für Unternehmen und automatische Textklassifikationen.

Ein wichtiges Ziel des GALA-Projektes besteht in der Untersuchung und Implementierung von Support-Vektor-Maschinen zur binären Klassifikation. Die folgenden Abschnitte setzen sich nach einer allgemeinen Einführung mit dem Aufbau von Support-Vektor-Maschinen für binäre Klassifikationsaufgaben sowie mit den dafür notwendigen mathematischen Theorien auseinander. Ausgehend von diesen Betrachtungen werden Algorithmen zur Umsetzung von Support-Vektor-Maschinen vorgestellt. Diese werden innerhalb des Projektes implementiert, für die speziellen Fragestellungen optimiert und zur Untersuchung aktueller Datensätze verwendet.

2 Überwachtes Lernen

Die leistungsfähigen Computer der heutigen Zeit sollen sehr komplexe Aufgaben lösen. Dabei sorgen Algorithmen dafür, dass genau festgelegt wird, wie ein bestimmtes Ergebnis durch die einfließenden Informationen entstehen soll. Doch es gibt immer mehr Aufgaben, bei denen dieser Zusammenhang zwar besteht, jedoch nicht sofort durch einen Algorithmus formuliert werden kann. Er muss zuerst erkannt oder erlernt werden. Mit Aufgaben dieser Art beschäftigen sich Verfahren des überwachten Lernens, die man in das Gebiet des Maschinellen Lernens einordnet. Maschinelles Lernen umfasst drei unterschiedliche Methoden:

(1) Überwachtes Lernen (supervised learning) gibt eine bestimmte Menge von Beispielen vor, die einen funktionalen Zusammenhang widerspiegeln. Auf deren Grundlage soll eine Eingabe-Ausgabe-Beziehung erlernt werden.

(2) Unüberwachtes Lernen (unsupervised learing) versucht, eine bestimmte Struktur in gegebenen Daten zu erkennen. Im Unterschied zu überwachtem Lernen gibt es keine vorgegebenen Ausgabewerte, nur der Aufbau der Eingabedaten soll nachvollzogen werden.

(3) Verstärkungslernen (reinforcement learning) findet Anwendung in der Spieltheorie. Ein Agent soll in Experimenten gute Handlungen erlernen, indem er für Erfolge belohnt und für Misserfolge bestraft wird.

Die in diesem Abschnitt betrachtete Methode soll die des überwachten Lernens sein.

2.1 Einordnung und Definitionen

Zwischen 1955 und 1957 wurde von Allen Newell und Herbert Alexander Simon der Logic Theorist entwickelt, ein Programm, das von vielen als erste Form Künstlicher Intelligenz überhaupt betrachtet wird [20]. Der Logic Theorist stellte Entscheidungsprobleme in einem dafür generierten Baum dar und verfolgte dann gemäß der erlernten Vorschrift immer denjenigen Ast, der zur Lösung des Problems führte. Im Jahr 1956 entwarf Frank Rosenblatt ein Modell, welches ebenfalls einen überwachten Lernprozess durchlaufen konnte [5]. Sein Perceptron löste binäre Klassifikationsprobleme, indem es Parameter erlernte, mit denen die Eingabewerte gewichtet wurden, bevor man sie summierte. Dieses Modell wurde mehr als zwanzig Jahre später mit dem Begriff des neuronalen Netzes identifiziert. Die Theorie der neuronalen Netze begann sich rasant ab 1986 zu entwickeln. Das Jahr der Entstehung von Support-Vektor-Maschinen wird auf etwa 1992 datiert. Dabei wurden zwei Ideen aus den sechziger Jahren kombiniert – die implizite Berechnung von Skalarprodukten zwischen transformierten Daten über Funktionen im Eingaberaum sowie die Theorie der optimal trennenden Hyperebene [33]. Diese ließen ein Verfahren völlig neuer Form entstehen.

Lernalgorithmen für überwachtes Lernen haben die Aufgabe, sogenannte Hypothesenfunktionen zu entwickeln und können auf eine Vielzahl von Problemen angewandt werden. Je nach Struktur der Ausgabewerte, die eine solche Funktion liefert, kann man die Algorithmen in drei Gruppen einteilen:

(1) Ausgabewerte aus $\{-1, 1\} \rightarrow$ binäre Klassifikationsprobleme,

(2) Ausgabewerte aus endlich vielen Zahlen $\rightarrow$ multiple Klassifikationsprobleme,

(3) Ausgabewerte aus den reellen Zahlen $\rightarrow$ Regressionsprobleme.

Die Wahl eines bestimmten Modells ist also von den zur Verfügung stehenden Daten abhängig. Innerhalb des Projektes werden im Allgemeinen binäre Klassifikationsprobleme definiert, untersucht und gelöst. Die Erweiterung der Lösungskonzepte bis hin zur Regression ist möglich und wird in der Literatur ausführlich erläutert. Zu multiplen Klassifikationsaufgaben siehe zum Beispiel [13], Regression wird unter anderem in [32], Kapitel 9 vorgestellt. Die drei schon erwähnten Modelle - Entscheidungsbäume, neuronale Netze und Support-Vektor-Maschinen - sind für binäre Klassifikationen anwendbar. Sie alle implementieren überwachtes Lernen. Das folgende kleine Beispiel zur Textklassifikation soll kurz verdeutlichen, warum man Algorithmen braucht, welche diese Form der Künstlichen Intelligenz umsetzen.

Jeder für eine Sammlung wissenschaftlicher Arbeiten eintreffende Text soll auf Eignung für die Aufnahme in die Datenbank überprüft werden. Das kann unter Umständen sehr zeitaufwändig sein. Kann man aber einen Algorithmus entwickeln, der anhand bestimmter Wortkombinationen ungeeignete Texte sofort ablehnt oder bei Auftreten bestimmter Schlüsselworte eine weitere Prüfung empfiehlt, spart man sich viel Arbeit. Dazu benötigt

man eine repräsentative Sammlung aufgenommener sowie abgelehnter Texte. Diese stellt man einem Lernalgorithmus zur Verfügung, der dann eine Hypothese entwickelt, mit der alle Texte, die in Zukunft eintreffen, bewertet werden.

Mit $X \subseteq \mathbb{R}^n$ $(n \in \mathbb{N})$ wird im Folgenden immer der Raum der Eingabevektoren bezeichnet. Wir suchen ein Verfahren, welches Vektoren aus X genau einer der beiden Klassen 1 und -1 zuordnet. Wir sprechen dabei auch von positiver und negativer Klasse. Das trägt der Tatsache Rechnung, dass man oft gewisse gute und schlechte Daten identifizieren und trennen will. Was genau mit „gut“ und „schlecht“ gemeint ist, hängt immer vom vorgegebenen Problem ab. Ein für die Medizin typisches Klassifikationsproblem beschäftigt sich beispielsweise mit der Frage, ob bestimmte Medikamente oder Substanzen einer erkrankten Person helfen können oder nicht.

Für einen Eingabevektor $\boldsymbol{x}$ aus X sei $y \in \{-1, 1\}$ der vorgegebene Ausgabewert. Dann heißt $(\boldsymbol{x}, y)$ Eingabe-Ausgabe-Paar für binäre Klassifikationsaufgaben. Solche Eingabe-Ausgabe-Paare bilden die praktische Grundlage eines lernenden Systems und beeinflussen neben dem gewählten Modell die Güte einer erlernten Funktionalität. Sie sollten möglichst zahlreich vorhanden, repräsentativ und fehlerfrei sein.

Sei $l \in \mathbb{N}$. Ein l-Tupel von Eingabe-Ausgabe-Paaren $\left\{(\boldsymbol{x}^1, y_1), ..., (\boldsymbol{x}^l, y_l)\right\}$ wird im Folgenden als Datensatz S_l bezeichnet. Wird S_l zum Erlernen einer Funktion benutzt, spricht man vom sogenannten Trainingsdatensatz; die l Datenpaare bezeichnet man dann auch als Trainings- oder Lernpaare. Wird S_l jedoch dazu benutzt, für eine feste Funktion zu zeigen, dass diese sinnvolle Ergebnisse produziert, spricht man vom Testdatensatz. Ein Datensatz wird nur mit S bezeichnet, falls dessen Größe variiert.

Überwachte Lernvorgänge setzen die Existenz von Trainings- und Testdaten voraus, die voneinander verschieden sein sollten. Der Grund dafür ist, dass jeder Test von Lernalgorithmen unabhängig vom Lernvorgang sein muss. In der Praxis wird der Trainingsdatensatz zusätzlich gesplittet, damit Teile der Trainingsdaten der Anpassung wichtiger Modellparameter dienen können. Die Details dazu werden später beschrieben. Die Menge der Funktionen, welche einem Lernalgorithmus als Kandidaten zur Auswahl stehen, wird Raum der Zielfunktionen $\mathcal{H}$ genannt.

Die Wahl eines geeigneten Raumes $\mathcal{H}$ für Lernalgorithmen ist nicht trivial. Er darf nicht zu klein sein, denn sonst gibt es keine Funktion in $\mathcal{H}$, die in der Lage ist, komplizierte Zusammenhänge darzustellen. Wenn er aber zu reichhaltig ist, dauert die Suche zu lange oder es entsteht Überanpassung. Als überangepasstes Modell bezeichnen wir eine Trennfunktion, die auf den Trainingsdaten eine hervorragende Performance zeigt, jedoch auf neuen Testdaten versagt. Überanpassung ist ein großes Problem innerhalb des überwachten Lernens. Besteht $\mathcal{H}$ aus affin-linearen Funktionen, wie beispielsweise bei der linearen Regression, ist es unmöglich, nichtlineare Abhängigkeiten zwischen Eingaben und Ausgaben darzustellen. Oft hat man es mit Daten zu tun, die experimentell ermittelt wurden. Sie können fehlerhaft und unvollständig sein. Es kann auch vorkommen, dass der funktionale Zusammenhang nicht eindeutig ist oder nicht existiert, beispielsweise wenn Variablen

untersucht worden sind, die mit der Klasseneinteilung nichts zu tun haben. Dann wird es umso schwerer, gute Modelle zu produzieren. Es gibt aber Ansätze, die genau solche Probleme auffangen und unter abgeschwächten Bedingungen lernen. Derartige Methoden werden wir im Zusammenhang mit Support-Vektor-Maschinen im Abschnitt 3 vorstellen.

In den nächsten vier Abschnitten werden Modelle für die Implementierung überwachter Lernvorgänge vorgestellt. Mit diesen wurden in den letzten Jahren und Jahrzehnten dauerhaft gute Ergebnisse für binäre Klassifikationsaufgaben erzielt und sie sind in der Praxis weit verbreitet. Für Klassifikationsaufgaben innerhalb des Projektes ist unter anderem das mächtige Verfahren der Support-Vektor-Maschinen gewählt worden. Die Gründe dafür werden im Anschluss motiviert, eine detaillierte Beschreibung folgt im Abschnitt 3.

2.2 Lineare Klassifikationsalgorithmen

Lineare Verfahren für binäre Klassifikationsaufgaben definieren einen sehr einfachen Raum der Zielfunktionen $\mathcal{H}$. Dieser enthält nur Funktionen der Form

$$f_{\text{lin}}(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle_2 + b = \sum_{k=1}^{n} w_k x_k + b \quad (\boldsymbol{x} \in X) \,, \tag{1}$$

wobei $\boldsymbol{w} \in \mathbb{R}^n$ und $b \in \mathbb{R}$ zunächst unbekannt sind und vom Lernalgorithmus mit Hilfe der Lernpaare bestimmt werden sollen. Einem beliebigen Eingabevektor $\boldsymbol{x}$ aus dem Raum X kann dann mittels (1) ein Funktionswert zugeordnet werden. Die Zielfunktion f_{lin} ist reell, also $f_{\text{lin}} : X \to \mathbb{R}$. Wie entsteht daraus eine binäre Klassifikation? Ein oft angewendetes Verfahren wird jetzt definiert, zunächst folgt noch ein technisches Detail.

In diesem Bericht wird ausschließlich die abgewandelte Signumfunktion

$$\text{sgn}(t) := \begin{cases} 1 & \text{falls } t \geq 0 \\ -1 & \text{sonst} \end{cases} \qquad (t \in \mathbb{R})$$

verwendet.

Im Folgenden wird unter einer linearen Entscheidungsvorschrift $h_{\text{lin}}(\cdot)$ die Hintereinanderausführung einer linearen Zielfunktion f_{lin} und der Signumfunktion verstanden, formal:

$$h_{\text{lin}}(\cdot) := \text{sgn}(f_{\text{lin}}(\cdot)) \,.$$

Eine lineare Entscheidungsvorschrift ordnet $\boldsymbol{x} = (x_1, ..., x_n) \in X$ der Klasse 1 zu, falls $f_{\text{lin}}(\boldsymbol{x}) \geq 0$ gilt, anderenfalls wird angenommen, $\boldsymbol{x}$ gehörte in die Klasse -1. Die Hyperebene $f_{\text{lin}} \equiv 0$ teilt den Raum X in zwei Teile. Man hofft, dass dadurch auch die zwei Klassen gut getrennt sind. Überprüfen kann man das allerdings nur für vorhandene Eingabe-Ausgabe-Paare.

Die Datenpaare eines Trainingsdatensatzes heißen linear separabel (linear trennbar), falls eine lineare Entscheidungsvorschrift existiert, welche alle diese Paare der richtigen Klasse zuordnet, also falls $h_{\text{lin}}(\boldsymbol{x}^i) = y_i$ für alle $i = 1, ..., l$ gilt. Man sagt auch, es existiert eine trennende Hyperebene für diese Datenpunkte.

Sei $(\boldsymbol{x}^i, y_i)$ ein Trainingspaar. Als funktionalen Abstand dieses Paares bezüglich einer trennenden Hyperebene $f_{\text{lin}} \equiv 0$ mit Parametern $\boldsymbol{w}$ und b bezeichnet man den Wert

$$\gamma_i := y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i \rangle_2 + b) = y_i \cdot f_{\text{lin}}(\boldsymbol{x}^i) \; . \tag{2}$$

Für alle Lernpaare eines Trainingsdatensatzes gilt $\gamma_i \in \mathbb{R}$ $(i = 1, ..., l)$. Falls $\gamma_i > 0$, so gilt $\text{sgn}(f_{\text{lin}}(\boldsymbol{x}^i)) = y_i$ und die Funktion f_{lin} hat die Eingabe $\boldsymbol{x}^i$ richtig klassifiziert. Im Falle $\gamma_i = 0$ ist y_i zu prüfen.

Abbildung 1 zeigt einen Punkt, der durch eine erlernte Funktion f nicht korrekt klassifiziert wurde und deshalb einen negativen funktionalen Abstand hat.

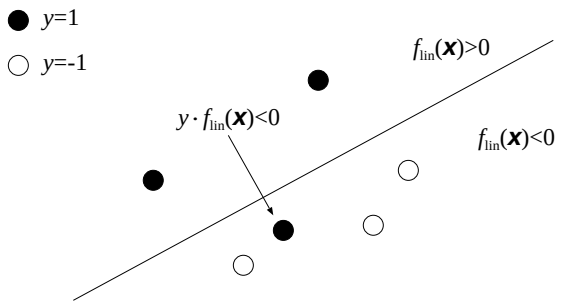


Abbildung 1: Lage eines Punktes mit negativem funktionalen Abstand γ.

Man betrachte nun die normalisierte Funktion f_{lin} mit Parametern $\left(\frac{1}{\|\boldsymbol{w}\|_2}\boldsymbol{w}, \frac{1}{\|\boldsymbol{w}\|_2}b\right)$. Dann kann man auch hier für alle Lernbeispiele die funktionalen Abstände berechnen. In diesem Fall werden sie als geometrische Abstände γ_i^g $(i = 1, ..., l)$ bezeichnet.

Als (funktionale) Marge oder auch Trenngüte γ einer Hyperebene bezüglich eines Trainingsdatensatzes S_l bezeichnet man das Minimum aller funktionalen Abstände γ_i, $1 \leq i \leq l$. Analog dazu ist die geometrische Marge oder auch geometrische Trenngüte γ^g einer Hyperebene bezüglich S_l definiert als das Minimum aller geometrischen Abstände γ_i^g, $1 \leq i \leq l$.

Wie man sieht, ist die Marge als Begriff nur dann sinnvoll, wenn alle Punkte richtig klassifiziert wurden. Dann ist sie positiv, und alle geometrischen Abstände können als Euklidische Distanzen zwischen Punkten und Hyperebene interpretiert werden.

Es ist möglich, dass man es mit Trainingsdaten zu tun hat, die nicht linear separabel sind. Man sucht dann intuitiv die bestmögliche Lösung, bei der so wenige Trainingspunkte wie möglich falsch zugeordnet werden. Auskunft über die Güte der Zuordnungen geben sogenannte Schlupfvariablen (slack variables). Man gibt sich eine gewünschte Marge $\gamma > 0$

vor und berechnet, wie stark diese von den funktionalen Abständen der Trainingspunkte verfehlt wird.

Sei $i \in \{1, ..., l\}$. Die Variable ξ_i, die definiert ist als

$$\xi_i := \max \{0, \gamma - y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i \rangle_2 + b)\}$$

mit vorgegebenem $\gamma > 0$, nennt man Schlupfvariable des Lernpaares $(\boldsymbol{x}^i, y_i)$.

Falls $\xi_i = 0$ gilt, liegt der i-te Punkt des Trainingsdatensatzes weit genug von der Hyperebene entfernt, andernfalls erfüllt er die Bedingung nicht. Falls dann zusätzlich $\xi_i > \gamma$ gilt, ist der i-te Eingabevektor falsch klassifiziert worden. Im Bereich $\xi_i \in (0, \gamma)$ liegen Punkte, die zwar richtig zugeordnet wurden, sich aber zu nah an der Hyperebene befinden.

Man beachte, dass während der Trainingsphase meist nur wenige Vertreter einer Gesamtheit von möglichen Punkten zur Verfügung stehen. Die trennende Hyperebene, die auch zur Theorie der Support-Vektor-Maschinen genutzt wird, stellt nur eine Annahme der Klassenaufteilung dar. Falsche Klassifikationen innerhalb eines Testdatensatzes durch ein lineares Modell können verschiedene Ursachen haben:

- Die Funktionalität ist nichtlinear, das Modell ist ungeeignet.
- Es gibt zu wenig Trainingsdaten, um die Klassen gut zu trennen.
- Die Klassen lassen sich auch mit vielen Daten nicht gut voneinander trennen, beispielsweise wenn die Daten stark verrauscht sind.
- Die eingeführten Variablen sind nicht für die beobachtete Klasseneinteilung der Trainingsdaten verantwortlich.
- Die Zuordnung der Trainingsdaten zu Klassen entspricht nicht der Realität. Beispiel: Wird an Testpersonen untersucht, ob ein bestimmter Wirkstoff Schmerzen lindert, können diese bei der Beurteilung dem Placebo- oder dem Nocebo-Effekt unterliegen und das Ergebnis stark verfälschen. Der Nocebo-Effekt ist das Gegenstück zum bekannten Placebo-Effekt. Es kommt oft vor, dass in Erwartung eines wirkungslosen oder schädigenden Medikaments auch nachweisbar wirksame, verträgliche Medikamente ihre Wirkung ganz oder teilweise einbüßen und andere Beschwerden auslösen.

Support-Vektor-Maschinen sowie die ersten einfachen Modelle neuronaler Netze basieren auf linearen Klassifikationsalgorithmen. Informationen zu den Erweiterungen, die dann zu diesen Verfahren führten, folgen später.

2.3 Entscheidungsbäume

Lernalgorithmen auf Basis von Entscheidungsbäumen sind einfach, erfolgreich und leicht zu implementieren. Sie waren die ersten Verfahren für die Umsetzung von Klassifikationsaufgaben mittels Computern. Abbildung 2 zeigt einen einfachen Entscheidungsbaum und soll den prinzipiellen Aufbau verdeutlichen.

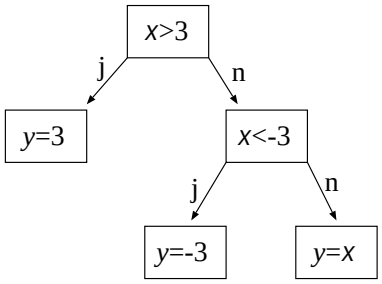


Abbildung 2: Entscheidungsbaum.

Dort soll

$$y = \max\ \{-3, \min\ \{3, x\}\} \quad (x \in \mathbb{R})$$

berechnet werden. Alle Knoten in einem Baum, hier sind es zwei, entsprechen einem Test. Je nach Ergebnis des Tests muss einer der möglichen Wege eingeschlagen werden. In diesem Beispiel gibt es immer nur zwei Wege, es können aber auch mehr sein. Jedes Blatt des Baumes stellt eine Entscheidung des Problems dar, das heißt, sobald ein Blatt erreicht wird, ist das Ausgangsproblem gelöst.

Hat ein Entscheidungsbaum seine endgültige Form schon angenommen, ist es nicht schwer, neue Daten zu klassifizieren. Man startet in der Wurzel und sucht den passenden Pfad. Dabei kann man immer einen solchen finden, da bei jedem Test alle möglichen Ergebnisse abgedeckt sein müssen. Handelt es sich zum Beispiel um den Test, welche Werte eine Variable annimmt, darf man nicht auf Alternativen *kleiner eins* oder *größer eins* testen, wenn es vorkommen kann, dass diese auch gelegentlich *genau eins* ist. Das muss bei Generierung jedes Baumes beachtet werden. Aber auch als Anwender eines festen Baumes kann man beispielsweise keine reellen Zahlen verwenden, wenn einer der Tests nur ganzzahlige Werte verarbeiten kann.

Bisher wurde nur von Bäumen mit fester Struktur gesprochen, die für ein bestimmtes Problem angepasst wurden. Aber wie generiert man einen solchen Baum? Wie schon beschrieben, stehen dafür die Eingabe-Ausgabe-Paare zur Verfügung. Aufgrund der Verzweigungen eines Baumes kann die Anzahl der Variablen in den Daten jedoch nichts darüber aussagen, wieviele Tests in einem Baum auftreten. Man weiß nur, dass an jedem Blatt einer der möglichen Ausgabewerte stehen muss, hier sind es zwei mögliche. Zunächst versucht man, einen Baum zu finden, sodass dieser für alle Paare des Trainingsdatensatzes bei erneuter Klassifikation die vorgegebenen Ausgabewerte liefert. Diese Aufgabe ist trivial. Man konstruiert einen Baum, der für jedes Paar einen Pfad zu einem Blatt mit dem gewünschten Ausgabewert besitzt. Dieser Entscheidungsbaum kann sehr aufgebläht sein. Außerdem ist intuitiv klar, dass er zu schlechten Ergebnissen führen kann. Dazu sei auf sogenannte übertrainierte Algorithmen verwiesen. Wie hier, können sich solche im Extremfall nur die Trainingsdaten „merken“. Von Künstlicher Intelligenz kann dann nicht mehr gesprochen werden. Daher sollte man zusätzlich versuchen, einen Baum aufzubauen, der eine Vielzahl von möglichen Fällen prägnant beschreiben kann, also bestimmte Varia-

blen auswählt, für die die oben erwähnten Tests durchgeführt werden sollen. Dabei muss man unter Umständen Fehler beim Training in Kauf nehmen.

Der auf Seite 105 angegebene Algorithmus verdeutlicht, wie man einen ersten sinnvollen Entscheidungsbaum generiert.

1.	S seien die aktuell betrachteten Lernpaare, $\mathcal{V}_{\text{aktuell}}$ die aktuelle Menge der Variablen und d_{aktuell} der zu verwendende Vorgabewert.
2.	S enthält keine Daten $\rightarrow$ Gib d_{aktuell} zurück, STOP. Alle y_i für Paare in S haben den gleichen Wert $y \rightarrow$ Gib y zurück, STOP. $\mathcal{V}_{\text{aktuell}} = \emptyset \rightarrow$ Gib den Ausgabewert zurück, der in S am häufigsten auftritt (Mehrheitsprinzip), STOP.
3.	Bestimme $\text{var}_{\text{aktuell}}$, die wichtigste Variable aus $\mathcal{V}_{aktuell}$. Benutze dafür S. Lege eine neue Wurzel mit Test auf $\text{var}_{\text{aktuell}}$ an.
4.	Bestimme alle κ unterschiedlichen Werte v_k von $\text{var}_{\text{aktuell}}$ in S.
5.	Führe 6. für alle $k = 1, ..., \kappa$ parallel und unabhängig aus.
6.	S_k bestehe nur aus den Lernpaaren $(\boldsymbol{x}^i, y_i)$ von S mit $\text{var}_{\text{aktuell}}(\boldsymbol{x}^i) = v_k$, $\mathcal{V}_{\text{aktuell}} = \mathcal{V}_{\text{aktuell}} \setminus \{\text{var}_{\text{aktuell}}\}$. Hänge einen Zweig mit Ausgabewert v_k an den Baum. Bestimme den aktuellen Teilbaum k an Wurzel $\text{var}_{\text{aktuell}}$ durch Start bei 1. mit S_k, $\mathcal{V}_{\text{aktuell}}$ und dem Vorgabewert d^*_{aktuell} als der Ausgabe, die in S_k die Mehrheit hat.

Tabelle 1: Basisalgorithmus zur Generierung eines Entscheidungsbaumes.

Der Algorithmus startet mit dem l-Tupel

$$S_l = \left\{(\boldsymbol{x}^1, y_1), ..., (\boldsymbol{x}^l, y_l)\right\}$$

von Lernpaaren, der Menge $\mathcal{V} := \{\text{var}_i,\ i = 1, ..., n\}$ aller Variablen sowie einem Vorgabewert d aus $\{-1, 1\}$ für die Entscheidung. Man beginne in Schritt 1. und arbeite al-

le weiteren Schritte ab. Die im dritten Schritt angegebene Bestimmung der sogenannten wichtigsten Variablen wurde noch nicht erklärt, es sollte immer die „beste“ der noch freien Variablen sein. Die Auswahl wichtiger Variablen als Wurzeln aller Teilbäume sowie die Suche einer geeigneten Variablen als Wurzel des gesamten Baumes werden vorgenommen, um die Tiefe des Baumes zu minimieren. Man versucht Variablen zu bestimmen, welche die Klassentrennung besonders gut erklären. Eine perfekte Variable würde also alle Trainingsdaten schon richtig trennen. Eine sehr schlechte Variable dagegen trägt zu keiner Verkleinerung des Problems bei. Aber wie kann man diese Wichtigkeit messen und ordnen? Dazu sollte man sich ein sinnvolles Maß definieren. Die beste Variable sollte dieses Maß maximieren, die schlechteste minimieren. Ein denkbares Maß kann zum Beispiel der erwartete Informationsgewinn sein, den man von einer Variablen und ihren Realisierungen erwarten kann. Als Vorschlag für die Suche nach guten Variablen wird im Folgenden [31] zitiert. Alle bisher benutzten Symbole gelten auch weiterhin.

Die wichtigste Variable in Schritt drei des Basisalgorithmus kann man bestimmen durch

$$\mathrm{var}_{\mathrm{aktuell}} = \underset{\mathrm{var}_i \in \mathcal{V}_{\mathrm{aktuell}}}{\arg\max}\, G(\mathrm{var}_i)$$

mit

$$G(\mathrm{var}_i) = I\left(\frac{S^+}{S^+ + S^-}, \frac{S^-}{S^+ + S^-}\right) - R(\mathrm{var}_i)\ .$$

Die Funktionen I und R seien definiert als

$$I(a,b) := -a\log_2(a) - b\log_2(b)$$

mit $a,\ b \in \mathbb{R}$ sowie

$$R(\mathrm{var}_i) := \sum_{k=1}^{\kappa} \frac{S_k^+ + S_k^-}{S^+ + S^-} \cdot I\left(\frac{S_k^+}{S_k^+ + S_k^-}, \frac{S_k^-}{S_k^+ + S_k^-}\right)\ .$$

Dabei haben die Symbole folgende Bedeutungen:

- S^+ (S^-) ist die Anzahl der Paare in S mit $y = 1$ $(y = -1)$ und
- S_k^+ (S_k^-) ist die Anzahl der Paare in S_k mit $y = 1$ $(y = -1)$.

Man beachte, dass ein Tupel S_k immer von der gewählten Variablen und deren Realisierungen auf den an dieser Stelle betrachteten Lernpaaren abhängig ist, siehe dazu die Schritte $4.$ und $5.$ im Algorithmus.

Welchen Hintergrund haben I, R und G? Man stelle sich einen Punkt vor, den man klassifizieren will. Welche Informationen benötigt man für eine korrekte Antwort? Vor den Tests kann man nur Schätzer für die Wahrscheinlichkeiten, in eine bestimmte Klasse zu fallen, angeben. Diese bauen auf den Trainingsdaten auf und sind einfach die relativen Häufigkeiten positiver und negativer Punkte, denn weitere Informationen sind zu diesem

Zeitpunkt nicht verfügbar. Betrachtet man dann eine Variable $\mathrm{var}_{\mathrm{aktuell}}$, kann diese nur einen Teil der benötigten Informationen liefern, denn es wird angenommen, dass perfekte Variablen nicht vorkommen. Die genaue Höhe der durch $\mathrm{var}_{\mathrm{aktuell}}$ zur Verfügung gestellten Informationen bestimmt man als Differenz der Informationen, welche vor und nach dem Test für eine korrekte Klassifikation benötigt werden und nennt diese G wie Gewinn. Dazu berechnet die Funktion R, wieviel Information im Durchschnitt nach einem Test auf eine bestimmte Variable für eine korrekte Klassifikation vorhanden sein muss. R steht für Rest und stellt die Summe der Informationen eines jeden Zweiges gewichtet mit dem Anteil der Lernpaare, die vom Testknoten aus diesen Zweig besuchen, dar. Eine besonders gute Variable zeichnet sich durch einen hohen Gewinn aus.

Ist ein Entscheidungsbaum generiert, kann man diesen Baum auch noch sinnvoll verkleinern. Um wie schon erwähnt nicht alle Variablen in die Entscheidung einzubauen, entfernt man diejenigen, die als irrelevant erkannt werden können. So kann man auch vermeiden, einen zu stark an die Trainingsdaten angepassten Baum zu erhalten. Dieses sogenannte Beschneiden (pruning) ist beispielsweise in der Software *C4.5* umgesetzt. Zu den Details lese man in [31] oder in [30] nach. Dabei werden nur Teilbäume abgeschnitten, die Grundstruktur des Baumes wird nicht verändert.

Wie man im vierten Schritt des Basisalgorithmus erkennen kann, wird zunächst davon ausgegangen, dass die Realisierungen aller Variablen nur eine bestimmte überschaubare Anzahl von Werten annehmen. In der Realität gibt es aber viele Experimente, in denen Variablen für jede Messung eine andere reelle Zahl annehmen. Um derartige Daten mit Entscheidungsbäumen bearbeiten zu können, muss der Basisalgorithmus geändert werden. Realisierungen stetiger Variablen kann man beispielsweise in sinnvolle Gruppen zusammenfassen [31] und somit diskretisieren.

2.4 Neuronale Netze

Neuronen sind Zellen, die elektrische Signale im menschlichen Gehirn verarbeiten. Diese Strukturen der Biologie wurden in die Informatik übernommen. Die Neuroinformatik rechtfertigt dieses Vorgehen damit, dass das Gehirn des Menschen das einzige existierende lernende intelligente System ist und deshalb intelligente Systeme des maschinellen Lernens nur analog zum Gehirn funktionieren können. Neuronen, mit denen man es bei Lernalgorithmen zu tun hat, sind informationsverarbeitende Einheiten.

Nach der allgemeinsten Definition sind neuronale Netze gerichtete Graphen von Neuronen. Die Kanten solcher Graphen stellen gewichtete Verbindungen zwischen Ausgaben eines Neurons und Eingaben eines anderen dar. Lernalgorithmen auf dieser Grundlage versuchen, diese Gewichte sinnvoll anzupassen. Neuronen eines Netzes sind immer in mindestens zwei Schichten angeordnet. Man unterscheidet Eingabeneuronen zur Signalaufnahme, Ausgabeneuronen zur Signalabgabe und Zwischenneuronen zur Signalübertragung. Letztere liegen in sogenannten verborgenen Schichten und sind nicht an den Ein-

und Ausgaben des Modells beteiligt. Je nach Richtung der Signalübertragungen werden zwei Gruppen von Netzen unterschieden:

- Rückgekoppelte neuronale Netze (feedback neural networks) und
- Vorwärtsgerichtete neuronale Netze (feedforward neural networks).

Die meisten Vertreter der zweiten Gruppe sind sogenannte mehrschichtige Netze (multi layer networks), bei denen es keine Verbindungen innerhalb der Schichten gibt. Neuronale Netze für binäre Klassifikation können mehrschichtig sein, müssen aber genau zwei Ausgabewerte zur Verfügung stellen. Die Begriffe neuronales Netz und Perceptron werden teilweise im gleichen Kontext verwendet, siehe dazu die Ausführungen in [10]. Im Allgemeinen gelten Perceptrons jedoch als diejenigen Netze, die keine Zwischenschichten besitzen. So wird es in [31] erklärt und auch hier verwendet.

Was geschieht mit den Eingabewerten in den Zwischenschichten? In jedem Neuron werden die eintreffenden gewichteten Eingabedaten summiert. Danach wird diese Summe zur Auswertung einer nichtlinearen Komponente, der sogenannten Aktivierungsfunktion a, benutzt. Der so berechnete Wert verlässt das Neuron als Ausgabe. Ein Netz mit fester Struktur kann im Gegensatz zu einem Baum durch Variierung der Funktion a unterschiedliche Ergebnisse liefern, innerhalb eines Netzes ist diese dann aber für alle Neuronen bindend. Oft verwendete Aktivierungsfunktionen sind

$$a(t) = \begin{cases} 1 & \text{falls } t \geq c \ (c \text{ konstant}) \\ -1 & \text{sonst} \end{cases} \qquad \text{und} \quad a(t) = \frac{1}{1 - e^{-t}} \ .$$

Abbildung 3 stellt ein Perceptron für binäre Klassifikation dar.

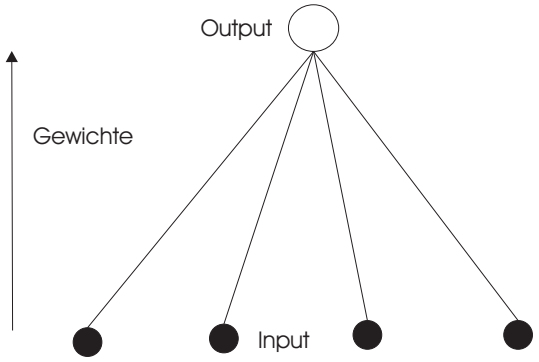


Abbildung 3: Graphische Darstellung eines Perceptrons.

Die vier Knoten im unteren Teil der Abbildung bedeuten, dass immer genau vier Variablen in den Algorithmus eingehen sollen. Im oberen Teil wird die berechnete Klasse ausgegeben. Jedes Element einer Eingabe $\boldsymbol{x} \in X$ soll in genau ein Eingabeneuron eingehen. Im einfachen Fall des Perceptrons wird der Trainingsdatensatz dazu verwendet, Gewichte

$w_j \in \mathbb{R}$ $(j = 1, ..., n)$ und eine Konstante $b \in \mathbb{R}$ zu bestimmen, die zu einer Hypothesenfunktion

$$h(\boldsymbol{x}) = \mathrm{sgn}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle_2 + b)$$

führen.

Anfang der sechziger Jahre des letzten Jahrhunderts wurde untersucht, wie man alle

$$w_j \quad (1 \leq j \leq n)$$

simultan bestimmen könnte. Frank Rosenblatt schlug den in Tabelle 2 angegebenen Algorithmus vor. Gegeben sei dafür S_l sowie ein $\eta \in \mathbb{R}_+$. η ist ein Modellparameter und wird Lernrate genannt. Das Verfahren bestimmt iterativ den Vektor $\boldsymbol{w}$ sowie die Konstante b.

1.	$\boldsymbol{w}^0 = 0,\ b_0 = 0,\ k = 0,\ R = \max\limits_{1 \leq i \leq l} \|\boldsymbol{x}^i\|$
2.	$i = 1$, $\mathrm{err} = 0$
3.	Falls $y_i \cdot (\langle \boldsymbol{w}^k, \boldsymbol{x}^i \rangle_2 + b_k) \leq 0$, dann gehe zu 4., sonst gehe zu 6.
4.	$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \eta y_i \boldsymbol{x}^i,\ b_{k+1} = b_k + \eta y_i R^2$
5.	$\mathrm{err} = 1,\ k = k + 1$
6.	$i = i + 1$, falls $i \leq l$ gehe zu 3., sonst gehe zu 7.
7.	Falls $\mathrm{err} = 1$ gehe zu 2., sonst STOP.

Tabelle 2: Lernen im Perceptron.

Die Struktur des so generierten Perceptrons reicht nicht aus, um komplizierte Zusammenhänge zu erlernen, zum Beispiel um Punkte zu trennen, die nicht linear trennbar sind. Deshalb wendet man sich den schon erwähnten mehrstufigen Netzen zu. In Abbildung 4 ist ein zweischichtiges neuronales Netz (two layer feedforward network) dargestellt.

Eingabeneuronen werden bei Angabe der Schichten im Allgemeinen nicht gezählt, denn sie verarbeiten keine Daten. Sowohl die Struktur als auch die Aktivierungsfunktion sollten zu Beginn des Trainings fest gewählt sein. Damit besteht die Aufgabe eines Lernalgorithmus auf Grundlage eines neuronalen Netzes nur in der Anpassung der Gewichte. Aber wer bestimmt diese Struktur? Die Bestimmung der Zahl der Neuronen und deren Anordnung in Schichten ist nicht einfach. Dazu gibt es zwei praktische Ansätze:

- Man startet mit einem sehr aufgeblähten Netz und verkleinert es iterativ.
- Man startet mit einem sehr einfachen Netz und vergrößert es iterativ.

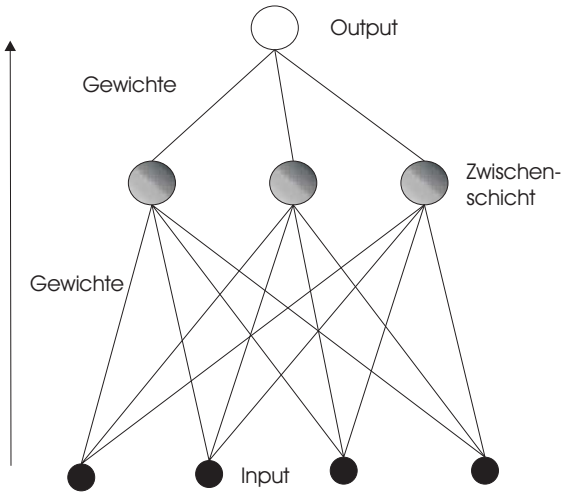


Abbildung 4: Zweischichtiges neuronales Netz.

Beim zweiten Ansatz kann man im Extremfall mit nur einem Neuron starten und von dort aus ein Netz aufbauen. Diese Vorgehensweise ähnelt dem vorgestellten Algorithmus zur Generierung eines Entscheidungsbaumes. Nach der Festlegung der Struktur agiert der eigentliche Lernalgorithmus. Der bekannteste Algorithmus zum Trainieren mehrschichtiger Netze wird zum Abschluss kurz vorgestellt.

Die Methode des back propagation (Rückwärts-Weitergabe) [31] wurde schon im Jahre 1969 entwickelt. Aufgrund zu schwacher Rechenleistung der damaligen Computer wurde sie jedoch abgelehnt und nicht genutzt. Ab dem Jahr 1986 erhielt der Algorithmus verstärkt Zuspruch. Zu dieser Zeit waren die Rechenkapazitäten soweit gestiegen, dass diese rechenintensive Methode gute Ergebnisse liefern konnte. Der Lernalgorithmus sucht das globale Minimum einer Fehlerfunktion auf den Trainingsdaten mittels Gradientenverfahren [9]. Begonnen wird dabei mit Startgewichten. Auch bei dieser Methode wird der Modellparameter $\eta \in \mathbb{R}_+$ für die Aktualisierungen benötigt und muss je nach Problem vorgegeben werden. Es wird sich zeigen, dass dieser Nachteil der Parameterwahl für neuronale Netze auch bei den Support-Vektor-Maschinen erhalten bleibt. Oft wird die Tatsache bemängelt, dass Lösungen des Gradientenverfahrens nur lokale Minima liefern und man das Training mehrmals mit unterschiedlichen Startparametern durchführen muss. Dieses Problem können Support-Vektor-Maschinen gezielt umgehen, mehr dazu folgt in Abschnitt 3.

Im Allgemeinen gelten neuronale Netze als black boxes, das bedeutet, im Gegensatz zu Entscheidungsbäumen erhält man keine Informationen darüber, wie die Eingaben auszusehen haben, um einen gewünschten Ausgabewert zu generieren.

2.5 Klassifikation mit Support-Vektor-Maschinen

An dieser Stelle soll kurz der Abschnitt 3 motiviert werden. Support-Vektor-Maschinen sind Lernalgorithmen für große Datenmengen, mit denen sich komplizierte Zusammenhän-

ge trainieren lassen. Sie wählen eine Zielfunktion aus linearen Abbildungen, da diese sehr gut handhabbar sind. Um Nichtlinearität zu ermöglichen, wird die Technik der Kerne benutzt. Die Wahl einer Zielfunktion erfolgt mit Hilfe der nichtlinearen, konvexen Optimierung. Die Qualität dieser Funktion sichert die sogenannte Generalisierungstheorie. Sie untersucht, wann eine Funktion in der Lage ist, gute Ergebnisse zu erzielen, wenn keine Beispiele, sondern nur noch unbekannte Daten in den Algorithmus eingehen. Support-Vektoren, die dem Verfahren den Namen geben, sind einige wenige der Beispieldaten, die während des Lernvorgangs ausgesucht werden und deren Eigenschaften bei allen späteren Klassifikationen neuer Punkte genutzt werden. Die bisher kurz erwähnten Komponenten werden in diesem Bericht detailliert vorgestellt, um das Verfahren der Support-Vektor-Maschinen zu verdeutlichen.

Auch wenn man auf den ersten Blick viele Gemeinsamkeiten finden kann, unterscheiden sich Support-Vektor-Algorithmen in vielen Details von neuronalen Netzen. Der erste Unterschied betrifft die Struktur. Wie im letzten Abschnitt zu sehen war, können Netze verschiedene Anzahlen von Schichten aufweisen. Für alle Verbindungen, die sich dann aus der Architektur ergeben, müssen Gewichte erlernt werden. Die Struktur von Support-Vektor-Verfahren ist fest und nicht durch die Biologie geprägt. Man wird im Abschnitt 3 sehen können, dass Support-Vektor-Maschinen keine Probleme mit lokalen Minima haben. Das ist ein klarer Vorteil für den Anwender.

Support-Vektor-Algorithmen zur Klassifikation finden verstärkt Zuspruch. Sie können für sehr viele Probleme und Forschungsgebiete angewendet werden. Dazu gehören beispielsweise:

(1) Klassifikation von Textdokumenten [15],

(2) Bild- und Schrifterkennung [28],

(3) Bioinformatik [22].

Dabei ist jedoch noch unklar, wie die Nutzung von Support-Vektor-Maschinen für binäre Modelle, wie sie beispielsweise im nächsten Abschnitt beschrieben werden, effizient auf multiple Klassifikationsaufgaben ausgeweitet werden kann. Sowohl die theoretische Formulierung dieses Problems als auch die daraus resultierenden Optimierungsprobleme sind an vielen Stellen besprochen worden, siehe zum Beispiel [13]. Allerdings konnte noch nicht geklärt werden, welcher Ansatz zur Umsetzung dieses Problems der günstigste ist. Aus diesem Grund werden oft die sehr erfolgreichen neuronalen Netze weiterbenutzt, da man den Nutzen von Support-Vektor-Maschinen für mehrklassige Fragestellungen nicht immer abschätzen kann.

3 Support-Vektor-Maschinen

Als Support-Vektor-Maschinen bezeichnet man Algorithmen, welche in kerninduzierten, hochdimensionalen Merkmalsräumen Parameter affin-linearer Funktionen mit Hilfe der

nichtlinearen Optimierung erlernen. Zunächst werden die nötigen Werkzeuge in den beiden folgenden Abschnitten bereitgestellt, bevor im Anschluss die zu lösenden Aufgaben diskutiert werden. Die Generalisierungstheorie wird in Abschnitt 3.4 Erkenntnisse über die Güte erlernbarer Funktionen liefern. Alle vier Abschnitte sind stark miteinander verknüpft und bilden die Grundlage für weitere Betrachtungen, insbesondere für die Beschreibung der implementierten Algorithmen.

In vielen Arbeiten wird das Lernverfahren der Support-Vektor-Maschinen losgelöst von den Theorien des Merkmalsraumes und der Kerne vorgestellt ([5], Kapitel 6). In diesem Bericht werden der Merkmalsraum und die Kerne als Werkzeuge interpretiert und rechtzeitig in die Lerntheorie eingebettet. Dieses Vorgehen kann zunächst zu Verwirrungen führen, ermöglicht es jedoch zu erkennen, an welchen Stellen Kerne benutzt werden können. Es sollte noch erwähnt werden, dass die Theorie der Kerne nicht einfach ein Modell ist, welches die Support-Vektor-Maschinen verfeinert. Sie ist eine ihrer Grundlagen und unterscheidet sie von linearen Lernverfahren.

3.1 Nichtlineare Optimierung

Wie noch zu sehen sein wird, führt das Lernverfahren einer Support-Vektor-Maschine auf das Problem einer konvexen Optimierungsaufgabe. Deshalb soll schon an dieser Stelle geklärt werden, wie man solche Aufgaben lösen kann. Dazu wird in diesem Abschnitt untersucht, wie eine bestimmte Klasse von Funktionen unter linearen Nebenbedingungen zu minimieren ist. Die Optimierungstheorie bietet gute Verfahren, um solche Probleme zu lösen. Die Minimierung einer beliebigen Funktion kann schwierig sein und auf viele lokale Minima führen. Schränkt man die Optimierung aber wie hier auf konvexe Funktionen ein, erhält man nur globale Lösungen. In diesem Abschnitt wird auf die Theorie von Lagrange eingegangen und gezeigt, in welchem günstigen Verhältnis primale und duale Optimierungsprobleme im konvexen Fall stehen.

Gegeben seien stetig differenzierbare, konvexe Funktionen

$$f : \mathbb{R}^n \to \mathbb{R}, \quad g_i : \mathbb{R}^n \to \mathbb{R} \ \ (i = 1, .., m_1)$$

sowie affin-lineare Funktionen

$$h_j : \mathbb{R}^n \to \mathbb{R} \ \ (j = 1, ..., m_2) \ .$$

Dann bezeichnet man die Aufgabe

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) \tag{3}$$

unter den Nebenbedingungen

$$g_i(\boldsymbol{x}) \leq 0 \ \ (i = 1, ..., m_1) \ ,$$
$$h_j(\boldsymbol{x}) = 0 \ \ (j = 1, ..., m_2)$$

als primale konvexe Optimierungsaufgabe.

Der sogenannte zulässige Bereich Z einer solchen Aufgabe ist definiert als

$$Z := \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}\} \; . \tag{4}$$

Globale Lösung einer Aufgabe der Form (3) ist ein Punkt $\boldsymbol{x}^*$ aus dem zulässigen Bereich Z, sodass es keinen anderen Punkt $\boldsymbol{x} \in Z$ gibt mit $f(\boldsymbol{x}) < f(\boldsymbol{x}^*)$.

Es gilt, dass jedes lokale Minimum von Aufgaben der Form (3) auch stets ein globales Minimum ist. Dazu sei angenommen, $\boldsymbol{x}^* \in Z$ sei ein lokales, aber kein globales Minimum von (3). Dann existiert ein $\tilde{\boldsymbol{x}} \in Z$ mit $f(\tilde{\boldsymbol{x}}) < f(\boldsymbol{x}^*)$. Deshalb und wegen der Konvexität von f folgt für alle Punkte $\boldsymbol{x}^\lambda = \lambda \boldsymbol{x}^* + (1-\lambda)\tilde{\boldsymbol{x}}$ mit $\lambda \in (0,1)$ die strikte Ungleichung

$$\begin{aligned} f(\boldsymbol{x}^\lambda) &\leq \lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\tilde{\boldsymbol{x}}) \\ &< \lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\boldsymbol{x}^*) \\ &= f(\boldsymbol{x}^*) \; . \end{aligned}$$

Für $\lambda \to 1$ wäre die Eigenschaft von $\boldsymbol{x}^*$, lokales Minimum zu sein, verletzt.

Man nennt die Nebenbedingung g_i aktiv im Punkt $\boldsymbol{x}$, falls $g_i(\boldsymbol{x}) = 0$ gilt. Andernfalls nennt man sie inaktiv. Die Menge der aktiven Restriktionen eines Punktes $\boldsymbol{x}$ ist eine Teilmenge von $\{1, ..., m_1\}$ und sei mit A bezeichnet.

Der Tangentialkegel $\mathcal{T}_Z(\boldsymbol{x})$ von Z in $\boldsymbol{x}$ hat die Form

$$\mathcal{T}_Z(\boldsymbol{x}) := \{\boldsymbol{d} \in \mathbb{R}^n \mid \exists \{\boldsymbol{x}^k\} \subseteq Z, \{t_k\} \subseteq \mathbb{R} : t_k \downarrow 0, \boldsymbol{x}^k \to \boldsymbol{x}, (\boldsymbol{x}^k - \boldsymbol{x})/t_k \to \boldsymbol{d}\} \; \forall \boldsymbol{x} \in Z \; .$$

Der linearisierte Tangentialkegel $\mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x})$ von Z in $\boldsymbol{x}$ hat die Form

$$\mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) := \{\boldsymbol{d} \in \mathbb{R}^n \mid \nabla g_i(\boldsymbol{x})^T \boldsymbol{d} \leq 0 \; \forall i \in A \; , \nabla h_j(\boldsymbol{x})^T \boldsymbol{d} = 0 \; \forall j = 1, ..., m_2\} \; \forall \boldsymbol{x} \in Z \; .$$

Mit ∇ sei hier und auch im Folgenden immer der Gradient einer differenzierbaren Funktion bezeichnet.

Für differenzierbare Funktionen $g : \mathbb{R}^n \to \mathbb{R}^{m_1}$ und $h : \mathbb{R}^n \to \mathbb{R}^{m_2}$ gilt

$$\mathcal{T}_Z(\boldsymbol{x}) \subseteq \mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) \; \forall \boldsymbol{x} \in Z \; . \tag{5}$$

Zum Nachweis siehe [8].

Für einen zulässigen Punkt $\boldsymbol{x} \in Z$ eines Optimierungsproblems der Form (3) gelte

$$\mathcal{T}_Z(\boldsymbol{x}) = \mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) \; . \tag{6}$$

Dann sagt man, $\boldsymbol{x}$ erfüllt die Constraint-Qualification.

Diese Bedingung ist nicht leicht nachprüfbar. Deswegen werden häufig sogenannte Regularitätsbedingungen angegeben, unter denen die Constraint-Qualification für alle zulässigen Punkte erfüllt ist. Von den vielen vorhandenen Ansätzen seien zwei aus [8] zitiert.

Betrachte das Optimierungsproblem (3). Dann ist jede der folgenden Bedingungen hinreichend dafür, dass (6) für alle $\boldsymbol{x} \in Z$ gilt:

(1) Es gilt die Slater-Regularitätsbedingung, die definiert ist als

$$Z^0 := \{\boldsymbol{x} \in Z \mid g_i(\boldsymbol{x}) < 0,\ i = 1, ..., m_1\} \neq \emptyset\ . \tag{7}$$

(2) Neben den h_j $(j = 1, ..., m_2)$ sind auch alle Funktionen g_i $(i = 1, ..., m_1)$ affinlinear.

$\boldsymbol{x}^*$ sei Minimalstelle von (3) und erfülle die Constraint-Qualification. Dann existieren $\boldsymbol{\alpha}^* \in \mathbb{R}^{m_1}$ und $\boldsymbol{\beta}^* \in \mathbb{R}^{m_2}$, sodass die sogenannten Karush-Kuhn-Tucker-Bedingungen (KKT)

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m_1} \alpha_i^* \nabla g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j^* \nabla h_j(\boldsymbol{x}^*) = 0\ , \tag{8}$$

$$h_j(\boldsymbol{x}^*) = 0\ \ (j = 1, ..., m_2)\ , \tag{9}$$

$$g_i(\boldsymbol{x}^*) \leq 0\ \ (i = 1, ..., m_1)\ , \tag{10}$$

$$\alpha_i^* g_i(\boldsymbol{x}^*) = 0\ \ (i = 1, ..., m_1)\ , \text{ und} \tag{11}$$

$$\alpha_i^* \geq 0\ \ (i = 1, ..., m_1) \tag{12}$$

gelten. Die Forderungen (9) und (10) ergeben sich sofort aus der Zulässigkeit von $\boldsymbol{x}^*$. Die übrigen Bedingungen (8), (11) und (12) folgen aus der Anwendung des bekannten Farkas-Lemmas, siehe dazu [8].

Nun soll noch gezeigt werden, dass die KKT-Bedingungen sogar ohne weitere Bedingungen hinreichend für die Existenz einer Lösung sind.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Tripel, welches die KKT-Bedingungen (8) - (12) erfüllt (KKT-Punkt). Dann muss gelten

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)\ \forall \boldsymbol{x} \in Z.$$

Wegen (9) und (10) gilt $\boldsymbol{x}^* \in Z$.
Für den Beweis werden neben den KKT-Bedingungen noch die Voraussetzungen

a) f ist konvex ,

b) alle g_i $(i = 1, ..., m_1)$ sind konvex , und

c) alle h_j $(j = 1, ..., m_2)$ sind affin-linear
verwendet. Diese führen auf

$$\begin{aligned} f(\boldsymbol{x}) &\overset{a)}{\geq} f(\boldsymbol{x}^*) + \langle \nabla f(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \\ &\overset{(8)}{=} f(\boldsymbol{x}^*) - \sum_{i=1}^{m_1} \alpha_i^* \langle \nabla g_i(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 - \sum_{j=1}^{m_2} \beta_j^* \langle \nabla h_j(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \\ &\overset{b),c),(4)}{\geq} f(\boldsymbol{x}^*) - \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}) + \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) \\ &\overset{(10),(11),(12)}{\geq} f(\boldsymbol{x}^*) \, . \end{aligned} \tag{13}$$

Damit ist $\boldsymbol{x}^*$ eine Minimalstelle der Optimierungsaufgabe (3).

Wir werden später zeigen, dass Optimierungsaufgaben, wie sie beim Verfahren Support-Vektor-Maschinen entstehen, nur affin-lineare Nebenbedingungen haben. Für diese Form konvexer Optimierungsaufgaben kann man die bisherigen Aussagen zusammenfassen, denn die Voraussetzungen implizieren die Constraint-Qualification. Es folgt: Für Aufgaben der Form (3) mit ausschließlich affin-linearen Nebenbedingungen ist die Tatsache, dass $\boldsymbol{x}^*$ globale Lösung ist, äquivalent dazu, dass es Multiplikatoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ gibt, die zusammen mit $\boldsymbol{x}^*$ einen KKT-Punkt bilden.

3.1.1 Theorie von Lagrange

Die Abbildung $L : \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \to \mathbb{R}$, definiert durch

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &:= f(\boldsymbol{x}) + \sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}) \\ &= f(\boldsymbol{x}) + \langle \boldsymbol{\alpha}, \boldsymbol{g}(\boldsymbol{x}) \rangle_2 + \langle \boldsymbol{\beta}, \boldsymbol{h}(\boldsymbol{x}) \rangle_2 \, , \end{aligned} \tag{14}$$

heißt Lagrange-Funktion, wobei $\boldsymbol{\alpha}$ und $\boldsymbol{\beta}$ Vektoren von Lagrange-Multiplikatoren genannt werden.

Ein Tripel $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ mit $\boldsymbol{\alpha}^* \geq 0$ wird Sattelpunkt von L genannt, falls für alle $(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ mit $\boldsymbol{\alpha} \geq 0$ die Ungleichungen

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \leq L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \leq L(\boldsymbol{x}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \tag{15}$$

erfüllt sind.

Wir betrachten erneut die primale Aufgabe (3) und die zugehörige Lagrange-Funktion. Dann gilt, dass das Tripel $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ die KKT-Bedingungen erfüllt, genau dann, wenn $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt von L ist. Diese Aussage ist als Sattelpunkttheorem bekannt. Wir beweisen die Aussage kurz.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein KKT-Punkt. Dann ist $\boldsymbol{x}^*$ ein stationärer Punkt von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$, denn wegen (8) gilt

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \mathbf{0} \,. \tag{16}$$

Lineare Funktionen sind konvex. Somit ist die Funktion $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ als Linearkombination konvexer und linearer Funktionen ebenfalls konvex. In $\boldsymbol{x}^*$ liegt ein globales Minimum von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ vor, denn für alle $\boldsymbol{x} \in \mathbb{R}^n$ gilt:

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(16)}{=} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) + \langle \nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \overset{(13)}{\leq} L(\boldsymbol{x}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,.$$

Der erste Teil der Sattelpunktungleichung wird direkt über die KKT-Eigenschaften gezeigt:

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(9),(11)}{=} f(\boldsymbol{x}^*) \overset{(9),(10),(12)}{\geq} f(\boldsymbol{x}^*) + \sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}^*) = L(\boldsymbol{x}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \,.$$

Damit ist $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt der Lagrange-Funktion.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt von L, dann ist $\boldsymbol{x}^*$ globales Minimum von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ und somit gelten (8) sowie die Ungleichung

$$\sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}^*) \leq \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j^* h_j(\boldsymbol{x}^*) \,.$$

Wegen $\boldsymbol{\alpha} \geq \mathbf{0}$ und $\boldsymbol{\beta} \in \mathbb{R}^{m_2}$ könnte der linke Teil der Ungleichung im Falle $\boldsymbol{g}(\boldsymbol{x}^*) > \mathbf{0}$ oder $\boldsymbol{h}(\boldsymbol{x}^*) \neq \mathbf{0}$ beliebig wachsen. Das impliziert $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$ sowie $\boldsymbol{h}(\boldsymbol{x}^*) = \mathbf{0}$ und sichert die Bedingungen (9) und (10) an einen KKT-Punkt.

Für die zulässigen Werte $\boldsymbol{\alpha} = \mathbf{0}$ und $\boldsymbol{\beta} = \mathbf{0}$ folgt schließlich aus obiger Ungleichung $\sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) \geq 0$ und wegen $\boldsymbol{\alpha}^* \geq \mathbf{0}$ sowie $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$ sind alle Komponenten der Summe genau Null.

Alle bisherigen Ergebnisse des Abschnittes 3.1 werden an dieser Stelle kurz zusammengefasst, bevor bestimmte Dualitätsaussagen beleuchtet werden.

Konvexe Optimierungsaufgaben der Form (3) haben die folgenden Eigenschaften:

(1) Ist $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt der Lagrange-Funktion, dann ist $\boldsymbol{x}^*$ globale Minimalstelle der Optimierungsaufgabe.

(2) Ist $\boldsymbol{x}^*$ globale Minimalstelle und erfüllt es die Slater-Regularitätsbedingung, dann existieren Vektoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ von Lagrange-Multiplikatoren, sodass $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ Sattelpunkt der Lagrange-Funktion ist.

(3) Für Aufgaben mit ausschließlich affin-linearen Nebenbedingungen ist $\boldsymbol{x}^*$ genau dann globale Minimalstelle, wenn Vektoren von Lagrange-Multiplikatoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ existieren, sodass $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ Sattelpunkt von L ist.

3.1.2 Dualität

Die Dualitätstheorie transformiert die bisher betrachteten primalen Optimierungsaufgaben und gibt Auskunft darüber, in welchem Verhältnis Lösungen dieser transformierten Aufgaben zu den ursprünglichen Lösungen stehen.

Die Funktion θ sei definiert als

$$\theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) := \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \; . \tag{17}$$

Dann hat die zum Ausgangsproblem duale Aufgabe in Lagrange-Form die Darstellung

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^{m_1},\, \boldsymbol{\beta} \in \mathbb{R}^{m_2}} \theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{18}$$

unter den Nebenbedingungen

$$\boldsymbol{\alpha} \geq \mathbf{0} \; .$$

Im Folgenden soll der Zusammenhang zwischen primalen und dualen Aufgaben gezeigt werden. Dieser führt in der konvexen Optimierung dazu, dass ohne Einschränkung die duale anstelle der primalen Aufgabe gelöst werden kann. Für die Anwendung von SV-Maschinen wird diese Tatsache von besonderer Wichtigkeit sein. Zunächst gilt der sogenannte schwache Dualitätssatz: Sei $\boldsymbol{x}^0$ ein zulässiger Punkt der primalen Aufgabe (3) und $(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0)$ ein zulässiger Punkt der dualen Aufgabe (18), dann ist

$$\theta(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \leq f(\boldsymbol{x}^0) \; ,$$

denn nach Definition gilt

$$\begin{aligned} \theta(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) &= \inf_{\boldsymbol{u} \in \mathbb{R}^n} L(\boldsymbol{u}, \boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \\ &\leq L(\boldsymbol{x}^0, \boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \\ &= f(\boldsymbol{x}^0) + \langle \boldsymbol{\alpha}^0, \boldsymbol{g}(\boldsymbol{x}^0) \rangle_2 + \langle \boldsymbol{\beta}^0, \boldsymbol{h}(\boldsymbol{x}^0) \rangle_2 \\ &\leq f(\boldsymbol{x}^0) \; . \end{aligned}$$

Letztere Ungleichung gilt, da für alle zulässigen Punkte $\boldsymbol{g}(\boldsymbol{x}^0) \leq \mathbf{0}$, $\boldsymbol{h}(\boldsymbol{x}^0) = \mathbf{0}$ und $\boldsymbol{\alpha}^0 \geq \mathbf{0}$ erfüllt sein müssen.

Das Maximum der dualen Aufgabe liegt also nie über dem Minimum der primalen Aufgabe. Dadurch ist aber noch nicht gesichert, dass die Lösungen nah beieinander liegen. Die sogenannte Dualitätslücke, der Abstand der Zielfunktionswerte, kann unter Umständen groß sein. Die Übereinstimmung der optimalen Funktionswerte sichert erst der starke Dualitätssatz. Für konvexe Aufgaben, wie sie hier betrachtet werden, gilt folgende Aussage: Gegeben sei eine konvexe Optimierungsaufgabe, für die ein Karush-Kuhn-Tucker-Punkt $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ existiert, dann ist

$$f(\boldsymbol{x}^*) = \theta(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,, \tag{19}$$

das heißt, der Wert der Dualitätslücke beträgt Null. Diese Aussage läßt sich schnell zeigen. $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ist offensichtlich ein Sattelpunkt der Lagrange-Funktion und somit ist $\boldsymbol{x}^*$ Minimalstelle der primalen Zielfunktion f. Daraus erhält man sofort:

$$f(\boldsymbol{x}^*) \overset{(9),(11)}{=} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(15)}{=} \inf L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \theta(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,.$$

Die Dualitätstheorie garantiert, dass es für die Lösung konvexer Optimierungsaufgaben hinreichend ist, die zugehörigen dualen Aufgaben zu betrachten. Diese weisen sehr einfache Nebenbedingungen auf, was sich positiv auf die Handhabbarkeit auswirkt. Zusätzlich ermöglicht die Dualitätstheorie den Einsatz sogenannter Kernfunktionen, denn erst die Formulierung der dualen Aufgabe einer Support-Vektor-Maschine lässt erkennen, welchen einfachen Weg man einschlagen kann. Weitere Informationen dazu folgen in den nächsten beiden Abschnitten. Wie sich noch zeigen wird, reicht es für die Anwendung von Support-Vektor-Maschinen sogar aus, quadratische Optimierungsprobleme zu betrachten. Das sind Aufgaben mit quadratischer Zielfunktion und linearen Nebenbedingungen. Selbstverständlich gilt die allgemeinere Theorie der konvexen Aufgaben, wie sie in diesem Abschnitt untersucht wurde, denn quadratische Aufgaben gehören in die Klasse konvexer Optimierungsprobleme.

3.2 Merkmalsräume und Kerne

Affin-lineare Funktionen zur Erklärung von Eingabe-Ausgabe-Zusammenhängen sind oft nicht geeignet, um komplizierte Abhängigkeiten zu erfassen. Support-Vektor-Maschinen nutzen lineare Verfahren zur Klassifikation. Da sie aber auch nichtlineare Zusammenhänge erlernen sollen, wenden sie sich zusätzlich einer anderen Methode zu.

Die Idee der Merkmalsräume ist, die Eingabedaten in einen hochdimensionalen Raum abzubilden und dort einfache lineare Funktionen zu trainieren. Man hofft, durch eine nichtlineare Transformation der Daten eine lineare Trennbarkeit zu erzeugen. Support-Vektor-Maschinen ersparen dem Anwender die mühsame Transformation der Daten, sie verwenden Kerne (kernels), um diese Arbeit zu umgehen. Diese Technik der impliziten Abbildung

in einen anderen Raum und die Nutzung von Kernen ist eine der wichtigen Eigenschaften von Support-Vektor-Maschinen, aber auch eine ihrer größten Herausforderungen, denn das Auffinden geeigneter Kerne bleibt oft Sache des Benutzers. Auf diese Aufgabe wird später noch eingegangen.

Zunächst soll in diesem Abschnitt gezeigt werden, welche Eigenschaften Kerne haben und warum man sie verwenden kann. Die wichtigste Grundlage dafür ist die Dualitätstheorie des letzten Abschnittes.

3.2.1 Grundlagen

Zu Beginn geht man davon aus, dass die Daten des Raumes X transformiert werden sollen, um lineare Trennbarkeit zu ermöglichen. Dazu wird eine Abbildung

$$\boldsymbol{\phi} : X \to F \tag{20}$$

definiert. $\boldsymbol{\phi}$ ist die sogenannte Merkmalsabbildung, wobei F ein Hilbertraum ist. Oft wählt man den Hilbertraum $\mathbb{R}^N,\ N \in \mathbb{N}$. Diese Transformation liefert einen Merkmalsvektor, also

$$\boldsymbol{x} = (x_1, .., x_n) \mapsto \boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), ..., \phi_N(\boldsymbol{x}))\ .$$

F kann auch unendlichdimensional sein. In diesem Fall wählt man den Hilbertraum l_2, den Raum aller Folgen $\boldsymbol{\psi} = \{\psi_1, ..., \psi_j, ...\}$, für die gilt

$$\|\boldsymbol{\psi}\|_{l_2}^2 = \langle \boldsymbol{\psi}, \boldsymbol{\psi} \rangle_{l_2} := \sum_{j=1}^{\infty} \psi_j^2 < \infty\ .$$

Der Raum

$$F \supseteq \{\boldsymbol{\phi}(\boldsymbol{x}) \mid \boldsymbol{x} \in X\}$$

wird Merkmalsraum genannt.

Für $N \in \mathbb{N} \cup \{\infty\}$ werden die $\phi_j(\boldsymbol{x}),\ j = 1, ..., N$, als Merkmale bezeichnet. Oft spricht man auch von Features und dem Featureraum, wobei diese Bezeichnung kritisch ist, da auch Variablenselektion oft als *feature selection* bezeichnet wird. Damit sind dann wiederum die Originalvariablen gemeint.

Das Skalarprodukt zwischen Merkmalsvektoren in F sei definiert als

$$\langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F := \sum_{j=1}^{N} \phi_j(\boldsymbol{x}) \phi_j(\boldsymbol{z})\ . \tag{21}$$

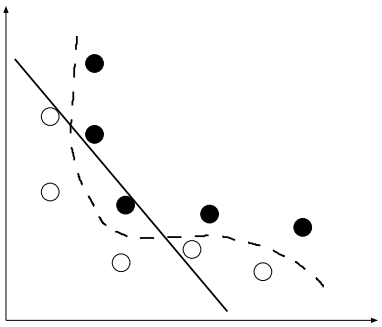


Abbildung 5: Nichtlineare Abhängigkeiten im Eingaberaum.

Abbildung 5 zeigt beispielhaft Daten, die im Eingaberaum nicht durch eine lineare Funktion separiert werden können.

Eine Transformation der Daten kann dazu führen, dass die Merkmale im Raum F linear trennbar sind, siehe dazu Abbildung 6. Man beachte, dass die Dimension der Räume X und F hier nur aus Darstellungsgründen immer zwei ist.

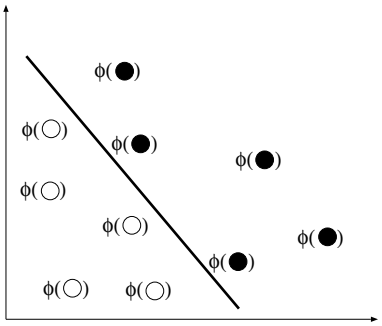


Abbildung 6: Lineare Trennbarkeit im Merkmalsraum.

Die wichtige Frage ist: Wie soll die Funktion ϕ gewählt werden und wie sieht der Wert für N aus? Als erste Idee kann man eine Dimensionsverkleinerung durchführen, also ein $N < n$ wählen. Dabei sollte die Anzahl der Merkmale N so klein wie möglich sein, wobei die wichtigsten Informationen, welche die Daten enthalten, nicht verloren gehen dürfen. Dieser Ansatz erscheint zu Beginn sinnvoll, denn:

(1) Die Performance von Algorithmen sinkt mit steigender Merkmalsanzahl, weil mehr Daten bearbeitet werden müssen.

(2) Bei wachsender Dimension des Raumes leidet möglicherweise die Generalisierungsfähigkeit. Alle Informationen dazu folgen im Abschnitt 3.4.

Problematisch bei diesem Ansatz ist die Tatsache, dass mit wachsendem N, also steigender Anzahl von Merkmalen, eine lineare Schätzfunktion auch immer bessere Ergebnisse erzielen wird. Diese Aussage geht zurück auf ein Theorem in [4]. Hier stehen sich also zwei Zielsetzungen konkurrierend gegenüber. Support-Vektor-Maschinen umgehen dieses

Problem, indem sie die Daten zwar in einen hochdimensionalen Raum abbilden, dies jedoch nur implizit tun, sodass alle Berechnungen im ursprünglichen Raum stattfinden und unabhängig von der Dimension des Merkmalsraumes sind. Als Mittel zur Realisierung dieses Ansatzes werden Kernfunktionen benutzt.

Jetzt soll untersucht werden, wie sich das Lernproblem verändert, falls man die Daten wie oben beschrieben transformiert. Zunächst kann man festhalten, dass es sich nicht mehr um ein lineares Problem handelt, da die Merkmalsabbildung $\boldsymbol{\phi}$ nicht linear sein muss. Dennoch stellt die gesuchte Funktion f auch hier eine Linearkombination dar - eine Kombination der Komponenten des Merkmalsvektors $\boldsymbol{\phi}$. Die zu lernende Funktion in X hat dann nach Abschnitt 2.2 folgende Gestalt:

$$f(\boldsymbol{x}) = \sum_{j=1}^{N} w_j \phi_j(\boldsymbol{x}) + b = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b \tag{22}$$

für zunächst unbekannte $\boldsymbol{w} \in F$ und $b \in \mathbb{R}$. Eine für einen gegebenen Datensatz optimale Funktion f, charakterisiert durch $(\boldsymbol{w}, b)$, kann, wie im Abschnitt 3.3 aufbauend auf Abschnitt 3.1 noch gezeigt wird (siehe Gleichung (42)), durch Substitution von

$$\boldsymbol{w} = \sum_{i=1}^{l} y_i \alpha_i \boldsymbol{\phi}(\boldsymbol{x}^i)$$

dual dargestellt werden als Auswertungsfunktion

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b \, . \tag{23}$$

Dabei müssen im Falle dieser Form der Zielfunktion gewisse α_i anstelle der w_j erlernt werden. Man sieht, dass für beliebige $\boldsymbol{x} \in X$ alle Trainingspaare an der Bestimmung von $f(\boldsymbol{x})$ beteiligt sind. Dabei ist wichtig, dass die Dimension des Raumes F nun für die Anzahl der Summenglieder keine Rolle mehr spielt. Jedoch sei festgehalten, dass F noch über das Skalarprodukt in die Berechnungen eingeht.

Kann man $\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F$ in (23) direkt über eine Funktion im Eingaberaum X berechnen, spart man sich die Abbildung in den Merkmalsraum und trainiert die Zielfunktion f direkt im Eingaberaum.

Mit der Wahl einer solchen Funktion umgeht man daher folgende Schritte:

(1) Suche einer geeigneten Abbildung $\boldsymbol{\phi}$,

(2) explizite Berechnung von Skalarprodukten in einem hochdimensionalen Merkmalsraum.

Man muss aber klären, welche Funktionen überhaupt dafür in Frage kommen und welche davon besonders geeignet sind. Gesucht sind also Funktionen, die Skalarprodukte in Merkmalsräumen implizit berechnen können.

Eine Funktion $K : X \times X \to \mathbb{R}$ wird Kern genannt, falls eine Abbildung $\boldsymbol{\phi}$ der Form (20) existiert mit

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \ \ \forall \boldsymbol{x}, \boldsymbol{z} \in X. \tag{24}$$

Gegeben seien ein Kern K und l Vektoren $\boldsymbol{x}^1, ..., \boldsymbol{x}^l \in X$. Dann nennt man die $l \times l$-Matrix $\boldsymbol{K}$ mit den Elementen $k_{ij} = K(\boldsymbol{x}^i, \boldsymbol{x}^j),\ 1 \leq i, j \leq l$, Grammatrix oder auch Kernmatrix.

Kerne sind geeignete Abbildungen, um Berechnungen im Merkmalsraum zu umgehen. Kombiniert man (23) und (24), erhält die Zielfunktion f die Form

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i K(\boldsymbol{x}^i, \boldsymbol{x}) + b \ . \tag{25}$$

f wird in diesem Fall Kernfunktion genannt um auszudrücken, dass nicht mit Merkmalen aus F gearbeitet wird.

Notwendige Eigenschaften, die jeder Kern erfüllen muss, ergeben sich aus den allgemeinen Eigenschaften von Skalarprodukten [2]:

(1)

$$\begin{aligned} K(\boldsymbol{x}, \boldsymbol{z}) &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \\ &\overset{\text{Symmetrie}}{=} \langle \boldsymbol{\phi}(\boldsymbol{z}), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F \\ &= K(\boldsymbol{z}, \boldsymbol{x}) \ , \end{aligned}$$

(2)

$$\begin{aligned} (K(\boldsymbol{x}, \boldsymbol{z}))^2 &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F^2 \\ &\overset{\text{Cauchy-Schwarz}}{\leq} \|\boldsymbol{\phi}(\boldsymbol{x})\|_F^2 \|\boldsymbol{\phi}(\boldsymbol{z})\|_F^2 \\ &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F \langle \boldsymbol{\phi}(\boldsymbol{z}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \\ &= K(\boldsymbol{x}, \boldsymbol{x}) K(\boldsymbol{z}, \boldsymbol{z}) \ . \end{aligned}$$

Diese Eigenschaften sind aber nicht hinreichend. Es bleibt noch die Frage, für welche Funktionen K eine Abbildung $\boldsymbol{\phi}$ in einen Merkmalsraum F existiert, sodass K ein Kern ist. Folgende bekannte Aussagen sollen kurz in Erinnerung gerufen werden:

- Symmetrische Matrizen haben nur reelle Eigenwerte [2].
- Eine symmetrische Matrix ist genau dann positiv semidefinit, wenn alle ihre Eigenwerte nichtnegativ sind [2].

3.2.2 Theorem von Mercer

Man betrachte wieder den Fall $N \in \mathbb{N} \cup \{\infty\}$ und einen Merkmalsvektor $\boldsymbol{\phi}(\boldsymbol{x})$. Das Skalarprodukt in F sei wie bisher von der Form (21).

Das Theorem von Mercer beschäftigt sich mit der Frage, wann eine Funktion K die Darstellung

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{z}) \tag{26}$$

haben kann, also ein Kern ist.

Sei X eine kompakte Teilmenge des $\mathbb{R}^n$ und $K : X \times X \to \mathbb{R}$ eine stetige, symmetrische Funktion, für die der Integraloperator $T_K : L_2(X) \to L_2(X)$,

$$(T_K f)(\cdot) = \int_X K(\cdot, \boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x},$$

nichtnegativ ist, also

$$\int_{X \times X} K(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{x}) f(\boldsymbol{z}) d\boldsymbol{x} d\boldsymbol{z} \geq 0 \tag{27}$$

für alle $f \in L_2(X)$ gilt. Für $j = 1, ..., N$ seien $\phi_j^M \in L_2(X)$ die normierten orthogonalen Eigenfunktionen von T_K mit zugehörigen positiven Eigenwerten $\lambda_j^M > 0$. Dann gilt für alle $\boldsymbol{x},\ \boldsymbol{z} \in X$, dass K die Darstellung

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{N} \lambda_j^M \phi_j^M(\boldsymbol{x}) \phi_j^M(\boldsymbol{z}) \tag{28}$$

besitzt. Für $N = \infty$ konvergiert die Reihe gleichmäßig [23].

Die gleichmäßige Konvergenz kann mit Hilfe des Satzes von Dini gezeigt werden. Dieser sowie ein ausführlicher Beweis des Theorems von Mercer sind in [16], § 8 zu finden.

Die speziellen Abbildungen K in (28) nennt man Mercer-Kerne. Die zugehörigen

$$\phi_j^M(\boldsymbol{x})\ (j = 1, ..., N)$$

werden als Mercer-Merkmale bezeichnet. Man kann zeigen, dass die Bedingung (27) äquivalent dazu ist, dass für jede beliebige endliche Teilmenge $\boldsymbol{x}^1, ..., \boldsymbol{x}^l$ aus X die zugehörige Grammatrix positiv semidefinit ist. Diese Aussage wird in [5] hergeleitet.

Fazit: Falls die Abbildung $\boldsymbol{\phi}$ die Form

$$\phi_j(\boldsymbol{x}) = \sqrt{\lambda_j^M}\phi_j^M(\boldsymbol{x}) \;\; (j = 1, ..., N) \tag{29}$$

hat, folgt aus der Darstellung (28) unmittelbar, dass Mercer-Kerne Kerne nach unserer Definition sind.

3.2.3 Kernreproduzierende Räume

F sei ein durch eine abzählbare Menge linear unabhängiger Merkmalsabbildungen ϕ_j $(j = 1, ..., N)$ gegebener Merkmalsraum. Die Eigenschaften der Abbildung $\boldsymbol{\phi}^M$ aus (29) müssen nicht gelten.

Es sei

$$K(\boldsymbol{x}, \boldsymbol{z}) := \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F = \sum_{j=1}^{N} \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{z}) \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \; . \tag{30}$$

Man betrachte eine Abbildung T mit $T(F) = \mathcal{H}$. Der Raum $\mathcal{H}$ über der Menge X entsteht durch die Vorschrift

$$T : \boldsymbol{\psi} \mapsto \sum_{j=1}^{N} \psi_j\phi_j(\cdot) \quad (\boldsymbol{\psi} \in F, \; N \in \mathbb{N} \cup \{\infty\}) \; .$$

$\mathcal{H}$ entspricht damit bis auf die additive Konstante b dem gewünschten Raum der Zielfunktionen.

Bei Verknüpfung der dualen Aufgabe mit der Theorie der Kerne - siehe Gleichung (25) - erhält man die zulässige Darstellung der realisierbaren Auswertungsfunktionen als

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i\alpha_i K(\boldsymbol{x}^i, \boldsymbol{x}) + b \quad (l \in \mathbb{N}, \; \alpha_i \in \mathbb{R}) \; ,$$

die zunächst völlig unabhängig vom Merkmalsraum ist. Es stellt sich die folgende Frage, ob die Menge

$$\mathcal{G} = \left\{ \sum_{i=1}^{l} y_i\alpha_i K(\boldsymbol{x}^i, \cdot) \mid l \in \mathbb{N}, \; \alpha_i \in \mathbb{R}, \; \boldsymbol{x}^i \in X, \; y_i \in \{-1, 1\} \right\}$$

mit dem Raum $\mathcal{H}$ über X übereinstimmt.

Für den Fall $N < \infty$ liefern die Darstellung von $\boldsymbol{w}$ auf Seite 121 sowie die vorausgesetzte Unabhängigkeit der Merkmale ϕ_j, dass alle Funktionen aus $\mathcal{H}$ realisierbar sind, das heißt: $\mathcal{G} = \mathcal{H}$. Für $N = \infty$ ist diese Gleichheit nicht gesichert. Die Menge der Abbildungen in

$\mathcal{H}$ könnte nicht alle denkbaren Auswertungsfunktionen enthalten, beispielsweise wenn sie Bilder von Punkten ohne endliche Norm in F sind. $\mathcal{H}$ könnte auch zu viele Funktionen enthalten. Diese Sorge ist unbegründet, da $\mathcal{G} \subseteq \mathcal{H}$ gilt und jede Zielfunktion aus $\mathcal{H}$ beliebig genau durch ein Element aus $\mathcal{G}$ approximiert werden kann. Zum Nachweis benötigt man noch ein Skalarprodukt in $\mathcal{H}$, welches nun definiert wird.

Für Funktionen $f(\boldsymbol{x}) = \sum_{j=1}^{N} \psi_j \phi_j(\boldsymbol{x})$ und $g(\boldsymbol{x}) = \sum_{j=1}^{N} \psi'_j \phi_j(\boldsymbol{x})$ aus $\mathcal{H}$ sei das Skalarprodukt in $\mathcal{H}$ definiert als

$$\langle f(\cdot), g(\cdot) \rangle_{\mathcal{H}} := \sum_{j=1}^{N} \psi_j \psi'_j \; . \tag{31}$$

Für $\overline{\mathcal{G}}$, den Abschluss von $\mathcal{G}$, gilt $\overline{\mathcal{G}} = \mathcal{H}$. Wir beweisen das im Folgenden mittels Mengeninklusion.

Es gilt $K(\cdot, \boldsymbol{z}) \in \mathcal{H}$, denn

$$K(\cdot, \boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\cdot)\phi_j(\boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\boldsymbol{z})\phi_j(\cdot) = T(\boldsymbol{\phi}(\boldsymbol{z})) \; .$$

Daraus kann man direkt folgern, dass alle Funktionen der Form

$$f(\cdot) = \sum_{i=1}^{l} y_i \alpha_i K(\cdot, \boldsymbol{x}^i) \quad (l \in \mathbb{N})$$

ebenfalls in $\mathcal{H}$ liegen. Da $\mathcal{H}$ abgeschlossen ist, folgt $\overline{\mathcal{G}} \subseteq \mathcal{H}$.

Für $f(\cdot) = \sum_{j=1}^{N} \psi_j \phi_j(\cdot)$ und $K(\cdot, \boldsymbol{z})$ aus $\mathcal{H}$ liefert die Anwendung des in $\mathcal{H}$ eingeführten Skalarproduktes die sogenannte Reproduktionseigenschaft

$$\langle f(\cdot), K(\cdot, \boldsymbol{z}) \rangle_{\mathcal{H}} = \sum_{j=1}^{N} \psi_j \phi_j(\boldsymbol{z}) = f(\boldsymbol{z}) \; . \tag{32}$$

Sei nun $f \in \mathcal{H}$. Da $\mathcal{H}$ ein Hilbertraum ist, kann man die Funktion f darstellen als $f = f_1 + f_2$ mit $f_1 \in \overline{\mathcal{G}}$ und $f_2 \perp \overline{\mathcal{G}}$. Die Reproduktionseigenschaft (32) sichert, dass für alle $\boldsymbol{z} \in X$ gilt:

$$f_2(\boldsymbol{z}) = \langle f_2(\cdot), K(\cdot, \boldsymbol{z}) \rangle_{\mathcal{H}} = 0 \; .$$

Daraus folgt, dass f_2 die Nullfunktion ist und $f = f_1$ gilt. Dies liefert sofort $f \in \overline{\mathcal{G}}$. Daraus folgt $\mathcal{H} \subseteq \overline{\mathcal{G}}$.

3.2.4 Bildung neuer Kerne

Bevor wir zeigen, wie man Kerne aus anderen Kernen, Funktionen und Konstanten erzeugen kann, erinnern wir an einige bekannte Definitionen.

- Gegeben seien zwei reelle Matrizen $\mathbf{A} = (a_{ij})_{1 \leq i,j \leq m}$ und $\mathbf{B} = (b_{ij})_{1 \leq i,j \leq m}$. Dann wird die $m^2 \times m^2$-Matrix

$$\mathbf{C} := \mathbf{A} \otimes \mathbf{B} = (a_{ij} \cdot \mathbf{B})_{1 \leq i,j \leq m}$$

  als Kronecker-Produkt von $\mathbf{A}$ und $\mathbf{B}$ bezeichnet.

- Gegeben seien zwei reelle Matrizen $\mathbf{A} = (a_{ij})_{1 \leq i,j \leq m}$ und $\mathbf{B} = (b_{ij})_{1 \leq i,j \leq m}$. Dann wird die $m \times m$-Matrix

$$\mathbf{C} := (a_{ij} \cdot b_{ij})_{1 \leq i,j \leq m}$$

  als Hadamard-Produkt von $\mathbf{A}$ und $\mathbf{B}$ bezeichnet.

- Eine Hauptuntermatrix einer quadratischen Matrix entsteht durch Streichen von Zeilen und Spalten mit demselben Index.

Seien K_1 und K_2 Mercer-Kerne, κ eine reelle Zahl, $f : X \to \mathbb{R}$ eine reelle Funktion und p ein Polynom mit positiven Koeffizienten. Dann sind

(1) $K(\boldsymbol{x}, \boldsymbol{z}) := K_1(\boldsymbol{x}, \boldsymbol{z}) + K_2(\boldsymbol{x}, \boldsymbol{z})$,

(2) $K(\boldsymbol{x}, \boldsymbol{z}) := \kappa \cdot K_1(\boldsymbol{x}, \boldsymbol{z})$,

(3) $K(\boldsymbol{x}, \boldsymbol{z}) := K_1(\boldsymbol{x}, \boldsymbol{z}) \cdot K_2(\boldsymbol{x}, \boldsymbol{z})$,

(4) $K(\boldsymbol{x}, \boldsymbol{z}) := f(\boldsymbol{x}) \cdot f(\boldsymbol{z})$,

(5) $K(\boldsymbol{x}, \boldsymbol{z}) := p(K_1(\boldsymbol{x}, \boldsymbol{z}))$, und

(6) $K(\boldsymbol{x}, \boldsymbol{z}) := e^{K(\boldsymbol{x}, \boldsymbol{z})}$

ebenfalls Mercer-Kerne, denn offensichtlich sind alle Funktionen $(1). - (6).$ symmetrisch. Für alle Beispiele bleibt zu zeigen, dass die Grammatrix $\boldsymbol{K}$ einer endlichen Anzahl beliebiger Punkte $\boldsymbol{x}^1, ..., \boldsymbol{x}^l$ $(\boldsymbol{x}^i \in X)$ positiv semidefinit ist, also

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} \geq 0$$

für alle $\boldsymbol{a} \in \mathbb{R}^l$ gilt.

(1) $\boldsymbol{a}^T(\boldsymbol{K_1} + \boldsymbol{K_2})\boldsymbol{a} = \boldsymbol{a}^T \boldsymbol{K_1} \boldsymbol{a} + \boldsymbol{a}^T \boldsymbol{K_2} \boldsymbol{a} \geq 0$.

(2) $\boldsymbol{a}^T(\boldsymbol{\kappa K_1})\boldsymbol{a} = \kappa \boldsymbol{a}^T(\boldsymbol{K_1})\boldsymbol{a} \geq 0$.

(3) Man betrachte das Kronecker-Produkt von $\boldsymbol{K_1}$ und $\boldsymbol{K_2}$, es sei mit $\boldsymbol{A}$ bezeichnet. Dann ist $\boldsymbol{A}$ positiv semidefinit. Die zur Funktion $K_1 \cdot K_2$ gehörende Matrix $\boldsymbol{K}$ ist das Hadamard-Produkt von $\boldsymbol{K_1}$ und $\boldsymbol{K_2}$. Offensichtlich ist $\boldsymbol{K} \in \mathbb{R}^{l \times l}$ eine Hauptuntermatrix von $\boldsymbol{A} \in \mathbb{R}^{l^2 \times l^2}$ und somit existiert zu jedem $\boldsymbol{a} \in \mathbb{R}^l$ ein $\boldsymbol{\beta} \in \mathbb{R}^{l^2}$, so dass

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} = \boldsymbol{\beta}^T \boldsymbol{A} \boldsymbol{\beta} \geq 0$$

gilt. Damit ist $\boldsymbol{K}$ positiv semidefinit.

(4)

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} = \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j f(\boldsymbol{x}^i) f(\boldsymbol{x}^j)$$

$$= \sum_{i=1}^{l} a_i f(\boldsymbol{x}^i) \sum_{j=1}^{l} a_j f(\boldsymbol{x}^j) = \left(\sum_{i=1}^{l} a_i f(\boldsymbol{x}^i) \right)^2 \geq 0 \, .$$

(5) $\boldsymbol{K}$ ist positiv semidefinit nach $1., 2.$ und $4.$

(6) Exponentialfunktionen können beliebig genau durch Polynome mit positiven Koeffizienten approximiert werden [5]. Anwendung von $5.$ liefert den Beweis.

Diese Beispiele zeigen, dass man in einem SVM-Modell auch mehrere Kerne über eine Linearkombination verwenden kann. Die optimalen Gewichte müssen dann allerdings günstig geschätzt werden.

Ein wichtiger Mercer-Kern, der auch in der Software zur Verfügung steht, ist der Gauß-Kern, der definiert ist als

$$K(\boldsymbol{x}, \boldsymbol{z}) := e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|^2}{\sigma^2}} \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \, . \tag{33}$$

Dabei ist $\sigma > 0$ ein zu wählender Parameter. Dass der Gauß-Kern tatsächlich ein Mercer-Kern und damit ein Kern nach unserer Definition ist, kann man leicht zeigen. Dazu formt man den Ausdruck (33) um:

$$K(\boldsymbol{x}, \boldsymbol{z}) = e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|^2}{\sigma^2}} = e^{\frac{-\|\boldsymbol{x}\|^2}{\sigma^2}} \cdot e^{\frac{-\|\boldsymbol{z}\|^2}{\sigma^2}} \cdot e^{\frac{2\langle \boldsymbol{x}, \boldsymbol{z} \rangle}{\sigma^2}} = f_1(\boldsymbol{x}) \cdot f_1(\boldsymbol{z}) \cdot K_1(\boldsymbol{x}, \boldsymbol{z}) \, .$$

$f_1(\boldsymbol{x}) \cdot f_1(\boldsymbol{z})$ ist ein Kern (K_2). $K_2 \cdot K_1$ ist ein Kern und damit ist K ein Kern.

Der Gauß-Kern gehört zur Gruppe der Kerne, die sich für sehr viele Probleme eignen und wird neben dem Polynomial-Kern

$$K(\boldsymbol{x}, \boldsymbol{z}) := (\langle \boldsymbol{x}, \boldsymbol{z} \rangle_2 + 1)^d, \; d = 1, 2, ... \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \tag{34}$$

oft als Standardkern bezeichnet. Der Polynomial-Kern besitzt ebenfalls einen Parameter.

Als Ergänzung zum Gauß-Kern wird ein sogenannter Slater-Kern verwendet. Er basiert auf der bekannten Slater-Funktion und unterscheidet sich vom Gauß-Kern lediglich in der Berechnung der Norm. Der Slater-Kern hat die Form

$$K(\boldsymbol{x}, \boldsymbol{z}) := e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|}{\sigma^2}} \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \,. \tag{35}$$

Zusätzlich ist ein Tanimoto-Kern implementiert worden. Er basiert auf dem Tanimoto-Koeffizienten und ist für binäre Datensätze geeignet. Er hat die Form

$$K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2)}{2I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) + 2I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2)} \,. \tag{36}$$

Die Funktion $I_{i,j}(\cdot, \cdot)$ berechnet, wie oft der Wert i in der ersten Komponente auftritt während **gleichzeitig** der Wert j in der zweiten Komponente zu verzeichnen ist. Die Reihenfolge ist von Bedeutung. Dieser Kern kann nur für $0/1$-wertige Daten sinnvoll ausgewertet werden. Für reelle Daten liefert er zwar auch Ergebnisse, da der Abgleich intern über Skalarprodukte implementiert ist, jedoch sind die Ergebnisse dabei nicht interpretierbar.

Anhand der Definition der Funktion $I_{i,j}(\cdot, \cdot)$ ist klar, dass folgende Gleichungen gelten:

- $I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_2$,
- $I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_1 - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2)$,
- $I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_2^T \boldsymbol{x}_2 - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2)$, und
- $I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2) = n - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) - I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) - I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Damit kann man die Darstellung (36) umformen:

$$\begin{aligned} K(\boldsymbol{x}_1, \boldsymbol{x}_2) &= \frac{\boldsymbol{x}_1^T \boldsymbol{x}_2 + n + \boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2}{n + \boldsymbol{x}_2^T \boldsymbol{x}_2 + \boldsymbol{x}_1^T \boldsymbol{x}_1 - 2\boldsymbol{x}_1^T \boldsymbol{x}_2} \\ &= \frac{n + 2\boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2}{n - 2\boldsymbol{x}_1^T \boldsymbol{x}_2 + \boldsymbol{x}_1^T \boldsymbol{x}_1 + \boldsymbol{x}_2^T \boldsymbol{x}_2} \\ &= \frac{n + t}{n - t} \,, \end{aligned}$$

wobei wir t definieren als

$$t = 2\boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2 \,.$$

Der Tanimoto-Kern besitzt keinen Parameter.

3.3 Klassifikation

In der Praxis werden je nach Zielsetzung und Möglichkeiten zwei Arten binärer Support-Vektor-Maschinen unterschieden. Diese werden als nächstes vorgestellt. Dabei wird sich auch zeigen, dass die darin entstehenden Optimierungsaufgaben immer quadratisch und somit konvex sind. Im Abschnitt 3.1 wurden Optimierungsaufgaben im $\mathbb{R}^n$ $(n \in \mathbb{N})$ betrachtet. Oft hat man es jedoch implizit mit einem unendlichdimensionalen Merkmalsraum zu tun, siehe dazu Abschnitt 3.2. Kann man die Theorie von Lagrange und alle Dualitätsaussagen dennoch nutzen?

Alle Konzepte der konvexen Optimierung sind auch im Raum F für $N = \infty$ anwendbar. Die Grundlagen dazu findet man in [21], Kapitel 8 - „Global theory of constrained optimization“. Im Einzelnen können dort die Theorie von Lagrange, Eigenschaften und Existenz von Sattelpunkten sowie Dualitätsaussagen nachgelesen werden. Die Problematik von $N = \infty$ innerhalb konvexer Optimierungsaufgaben wurde in der vorliegenden Literatur zu SV-Maschinen allerdings nicht erwähnt. Prinzipiell ist die Brücke zwischen Optimierung und maschinellem Lernen noch sehr klein und begrenzt sich im Allgemeinen auf theoretische Betrachtungen und die Nutzung von klassischen Optimierungsalgorithmen.

3.3.1 Klassifikation maximaler Trenngüte

Dieser Lernalgorithmus kann nur für linear trennbare Trainingsdaten verwendet werden. Er erzeugt eine Hyperebene mit größtmöglicher geometrischer Marge (maximal margin classifier). Oft wird dieses Verfahren als hard margin classifier bezeichnet [5, 33]. [34] reserviert diesen Namen jedoch für ein Modell, das für nichtseparierbare Daten angewendet wird. Der Vorteil solcher Hyperebenen liegt in der Tatsache, dass der Generalisierungsfehler bei wachsender geometrischer Trenngüte sinkt. Damit dieser Abschnitt nicht verwirrt, folgen die Gründe dafür erst im Abschnitt 3.4 nach einer Einführung in die Generalisierungstheorie. Es sei aber ab hier schon angenommen, dass eine große Marge für gute Qualität eines Klassifikators spricht.

Gegeben sei ein linear trennbarer Trainingsdatensatz S_l. Eine Lösung $(\boldsymbol{w}^*, b^*)$ des Optimierungsproblems

$$\min_{\boldsymbol{w} \in F, b \in \mathbb{R}} \quad \langle \boldsymbol{w}, \boldsymbol{w} \rangle_F \tag{37}$$

unter den Nebenbedingungen

$$y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b) \geq 1 \;\; (i = 1, ..., l)$$

realisiert die Hyperebene maximaler geometrischer Trenngüte. Die Marge hat dann den Wert $\gamma^g = 1/\|\boldsymbol{w}^*\|_F$. Beweis: Die Skalierung der Hyperebene $(\boldsymbol{w}^*, b^*)$ mit einem beliebigen $\lambda \in \mathbb{R}$ verändert deren geometrische Lage nicht, variiert aber die funktionale Marge.

Aus Abschnitt 2 ist bekannt, dass $\gamma^g = \gamma/\|\boldsymbol{w}^*\|_F$ gilt. Man kann daher, anstatt die geometrische Trenngüte zu maximieren, auch die funktionale Trenngüte konstant halten und die Norm des Parametervektors minimieren. Das obige Minimierungsproblem entsteht bei der Wahl einer funktionalen Trenngüte mit Wert 1.

An dieser Stelle geht man zur dualen Form der Aufgabe (37) über, denn im Abschnitt 3.2 hat sich gezeigt, dass diese den Einsatz von Kernen möglich macht. Außerdem vereinfachen sich die Nebenbedingungen, wie noch zu sehen sein wird.

Die Lagrange-Funktion für das Ausgangsproblem (37) hat die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) := \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F - \sum_{i=1}^{l} \alpha_i \cdot \Big(y_i\big(\langle \boldsymbol{w}\ , \boldsymbol{\phi}(\boldsymbol{x}^i)\rangle_F + b\big) - 1\Big)\ . \tag{38}$$

Der Faktor $1/2$ wurde zur Vereinfachung der weiteren Gleichungen gewählt. Er verändert als monotone Transformation das Ergebnis nicht.

Die notwendigen Bedingungen für einen stationären Punkt der Lagrange-Funktion in $(\boldsymbol{w}, b)$ bei festem $\boldsymbol{\alpha}$

$$\frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{l} y_i\alpha_i\boldsymbol{\phi}(\boldsymbol{x}^i) \stackrel{!}{=} 0, \qquad \frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial b} = -\sum_{i=1}^{l} y_i\alpha_i \stackrel{!}{=} 0 \tag{39}$$

werden in (38) eingesetzt und es ergibt sich die Funktion W, die der Funktion θ in (17) entspricht:

$$\begin{aligned} W(\boldsymbol{\alpha}) &:= \inf_{\boldsymbol{w}\in F, b\in\mathbb{R}} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) \\ &= \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j\langle\boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j)\rangle_F \\ &\stackrel{(24)}{=} \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j)\ . \end{aligned} \tag{40}$$

Die entstandene Funktion W ist nur noch von $\boldsymbol{\alpha}$ abhängig, denn alle anderen Informationen sind durch S_l und die Wahl einer impliziten Datentransformation über den Kern K gegeben.

Die bisher gewonnenen Erkenntnisse werden an dieser Stelle zusammengefasst. Sei S_l ein linear trennbarer Trainingsdatensatz und $\boldsymbol{\alpha}^*$ löse das duale Optimierungsproblem

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) \tag{41}$$

unter den Nebenbedingungen

$$\begin{aligned} \sum_{i=1}^{l} y_i\alpha_i &= 0 \ \ (i=1,..,l)\ , \\ \alpha_i &\geq 0 \ \ (i=1,...,l)\ . \end{aligned}$$

Dann realisiert der Vektor

$$\boldsymbol{w}^* = \sum_{i=1}^{l} y_i\alpha_i^*\boldsymbol{\phi}(\boldsymbol{x}^i) \in F \tag{42}$$

die Hyperebene maximaler geometrischer Trenngüte.

Um zu zeigen, wie Support-Vektor-Maschinen zu ihrem Namen kamen, werden erneut die wichtigen Karush-Kuhn-Tucker-Bedingungen von Seite 114 betrachtet. Diese führen zu den folgenden Forderungen:

$$\alpha_i^* \cdot \Big(y_i\big(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i)\rangle_F + b^*\big) - 1\Big) = 0 \quad (i=1,...,l)\ .$$

Sie sagen aus, dass nur diejenigen α_i^* streng positiv sein können, für welche die zugehörigen Merkmale $\boldsymbol{\phi}(\boldsymbol{x}^i)$ einen funktionalen Abstand zur Hyperebene von genau 1 haben. An dieser Stelle kann man den Begriff der Support-Vektoren erklären.

Für alle i mit $\alpha_i^* \neq 0$ bezeichnet man die Eingabevektoren $\boldsymbol{x}^i$ des Trainingsdatensatzes als Support-Vektoren.

Nach (42) tragen nicht alle Lernpaare, sondern nur die Support-Vektoren zur Bestimmung von $\boldsymbol{w}^*$ bei. Abbildung 7 stellt die Situation vereinfacht dar.

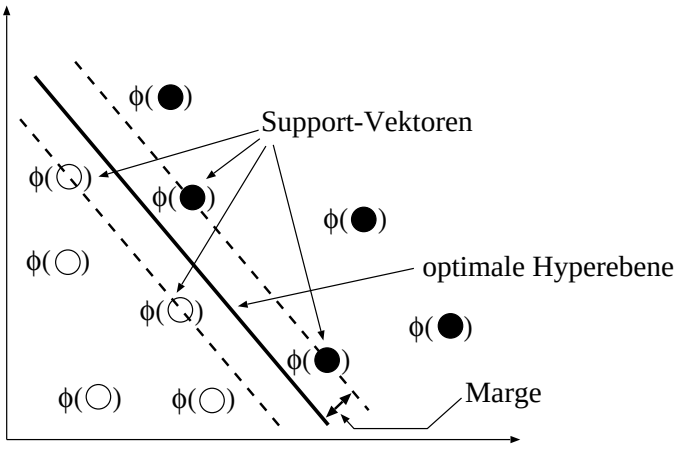


Abbildung 7: Lage von Support-Vektoren im Merkmalsraum.

Das Verfahren maximaler Trenngüte kann nur aus Hypothesen auswählen, welche alle Trainingsdaten korrekt zuordnen und dabei eine funktionale Marge von eins einhalten. Beispiele, die falsch klassifiziert werden oder sich zu nah an der Hyperebene befinden, haben funktionale Abstände, die kleiner als 1 sind. Im Fall falscher Klassifikationen sind diese sogar negativ, siehe Abschnitt 2.2. Für Algorithmen, die mit solchen Situationen arbeiten, sei auf den nächsten Abschnitt verwiesen.

Wenn die Menge aller i mit $\alpha_i^* \neq 0$ als $\mathcal{SV}$ bezeichnet wird, dann kann der Wert der Zielfunktion f an einer Stelle $\boldsymbol{x}$ berechnet werden als

$$\begin{aligned} f(\boldsymbol{x}) &\overset{(22)}{=} \sum_{j=1}^{N} w_j^* \phi_j(\boldsymbol{x}) + b^* \overset{(42)}{=} \sum_{j=1}^{N} \left(\sum_{i=1}^{l} y_i \alpha_i^* \phi_j(\boldsymbol{x}^i) \right) \phi_j(\boldsymbol{x}) + b^* \\ &= \sum_{i=1}^{l} \sum_{j=1}^{N} y_i \alpha_i^* \phi_j(\boldsymbol{x}^i) \phi_j(\boldsymbol{x}) + b^* = \sum_{i=1}^{l} y_i \alpha_i^* \sum_{j=1}^{N} \phi_j(\boldsymbol{x}^i) \phi_j(\boldsymbol{x}) + b^* \\ &\overset{(21)}{=} \sum_{i=1}^{l} y_i \alpha_i^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b^* = \sum_{i \in \mathcal{SV}} y_i \alpha_i^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b^* \\ &\overset{(24)}{=} \sum_{i \in \mathcal{SV}} y_i \alpha_i^* K(\boldsymbol{x}^i, \boldsymbol{x}) + b^* \,. \end{aligned} \tag{43}$$

Als einzige Unbekannte bleibt noch der Wert von b^* übrig. Dieser kann theoretisch leicht bestimmt werden.

Es gilt

$$b^* = \frac{1}{y_i} - \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F \quad (i \in \mathcal{SV}) \,, \tag{44}$$

denn für $i \in \mathcal{SV}$ gilt $\alpha_i^* \cdot [y_i \left(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b^* \right) - 1] = 0$ (KKT-Bedingung). Umgestellt nach b^* folgt (44).

Praktisch hat (44) wenig Nutzen, da die Merkmalsabbildung $\boldsymbol{\phi}$ im Allgemeinen nicht explizit bekannt ist. Dualitätsaussagen und die Theorie der Kerne können aber auch hier helfen.

Die KKT-Bedingungen

$$\alpha_i^* \cdot \left[y_i \left(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b^* \right) - 1 \right] = 0 \;\; (i = 1, .., l)$$

kann man umformulieren zu

$$\alpha_i^* \cdot \left[y_i \cdot f(\boldsymbol{x}^i) - 1 \right] \overset{(43)}{=} \alpha_i^* \cdot \left[y_i \left(\sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) + b^* \right) - 1 \right] = 0 \,.$$

Für $\alpha_i^* > 0$ folgt

$$y_i \cdot \left(\sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) + b^* \right) = 1 \, .$$

Wegen $y_i^2 = 1 \;\; \forall i = 1, .., l$ gilt

$$b^* = y_i - \sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) \, , \tag{45}$$

falls $\boldsymbol{x}^i$ ein Support-Vektor ist. (45) kann direkt ausgewertet werden, ohne dass Kenntnis von $\mathcal{F}$ oder $\boldsymbol{\phi}$ nötig ist.

Als weitere Konsequenz aus den KKT-Bedingungen erhält man, dass die geometrische Marge der optimalen Hyperebene in Abhängigkeit von den positiven Lagrange-Multiplikatoren berechnet werden kann als:

$$\begin{aligned}
\gamma^g &= \frac{1}{\|\boldsymbol{w}^*\|_F} = \frac{1}{\sqrt{\langle \boldsymbol{w}^*, \boldsymbol{w}^* \rangle_F}} \overset{(42)}{=} \frac{1}{\sqrt{\sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i^* \alpha_j^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F}} \\
&= \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} y_i \alpha_i^* \sum_{j \in \mathcal{SV}} y_j \alpha_j^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F}} \overset{(44)}{\underset{y_i \cdot f(\boldsymbol{x}^i)=1}{=}} \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} y_i \alpha_i^* \left(\frac{1}{y_i} - b^* \right)}} \\
&= \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} \alpha_i^* - b^* \sum_{i \in \mathcal{SV}} y_i \alpha_i^*}} \overset{\sum y_i \alpha_i^* = 0}{=} \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} \alpha_i^*}} \, .
\end{aligned}$$

Für ein fertig trainiertes Modell kann man also - ohne Kenntnis der Merkmale - aus den Lagrange-Multiplikatoren die Marge im Merkmalsraum bestimmen. $\sum \alpha_i^*$ ist immer streng positiv, da Support-Vektoren streng positive Lagrange-Multiplikatoren haben müssen.

3.3.2 Klassifikation nichtseparierbarer Daten

Die Methode der maximalen Trenngüte hat einen entscheidenden Nachteil für Anwendungen aus der Praxis. Sie produziert nur Ergebnisse für linear trennbare Lerndaten. Diese Voraussetzung ist leider oft nicht erfüllt, beispielsweise wenn man verrauschte Daten nutzen muss. Deswegen werden in diesem Abschnitt ausgehend von der Klassifikation maximaler Trenngüte abgeschwächte Verfahren vorgestellt. Diese werden als soft margin classifier bezeichnet. Vladimir Vapnik führte zunächst einen sogenannten hard margin classifier ein, welcher aus Gründen der einfacheren Umsetzung in einen soft margin classifier umformuliert wurde und hier nicht vorgestellt werden soll. Details lese man in [34]

nach. Der Begriff des hard margin classifiers wird heutzutage fälschlicherweise gleichgesetzt mit dem sogenannten maximal margin classifier des letzten Abschnittes.

Unter nichtseparierbaren Daten werden auch diejenigen Daten verstanden, welche nur eine Klassifikationsfunktion maximaler Trenngüte erzeugen können, die durch Überanpassung gekennzeichnet ist, also für eine gute Klassifikation unbrauchbar sein wird.

Die im Folgenden beschriebene Methode der weichen Trennung setzt an den funktionalen Abständen der Trainingsdaten an und formuliert schwächere Bedingungen zur Datentrennung.

Bisher wurden für alle Trainingspaare $i = 1, ..., l$ funktionale Abstände von $\gamma_i \geq 1$ vorausgesetzt. Jetzt benutzt man die schon bekannten Schlupfvariablen aus Abschnitt 2 und formuliert neue Bedingungen

$$\gamma_i = y_i \cdot f(\boldsymbol{x}^i) \geq 1 - \xi_i.$$

Dabei wird gefordert, dass $\xi_i \geq 0$ für alle $i = 1, ..., l$ gelten. Die Schlupfvariablen drücken aus, dass nun auch kleinere funktionale Abstände toleriert werden. Damit können verrauschte Daten beziehungsweise Datensätze, die im gewählten Merkmalsraum nicht linear trennbar sind, klassifiziert werden. Als problematisch erweist sich jedoch die Größe der ξ_i, denn für jeden Lösungsvektor $\boldsymbol{\xi}$ gibt es einen Lösungsvektor $\boldsymbol{\xi'}$ mit $\xi_i \leq \xi_i'$. Es ist ersichtlich, dass $\boldsymbol{\xi}$ gewählt werden sollte. Alle i mit $\xi_i > 0$ werden als Fehler gewertet, deshalb sollten sie nicht unnötig vergrößert werden, sondern den Tatsachen entsprechen. Das muss noch in die Formulierung der Aufgabe aufgenommen werden, indem wachsende ξ_i auch die Zielfunktion wachsen lassen. Dafür wird eine Norm des Vektors $\boldsymbol{\xi}$, gewichtet mit einem noch unbekannten Faktor $C > 0$, zum Wert der Zielfunktion addiert.

In der Literatur zu Support-Vektor-Maschinen werden die beiden folgenden Verfahren vorgestellt, siehe beispielsweise [5]. Die Ansätze sind sehr ähnlich.

Betragssumme

$$\min_{\boldsymbol{w} \in F, b \in \mathbb{R}, \boldsymbol{\xi} \in \mathbb{R}^l} \left(\langle \boldsymbol{w}, \boldsymbol{w} \rangle_F + C \sum_{i=1}^{l} \xi_i \right) \tag{46}$$

unter den Nebenbedingungen

$$\begin{aligned} y_i \cdot f(\boldsymbol{x}^i) &\geq 1 - \xi_i \ \ (i = 1, .., l) \ , \\ \xi_i &\geq 0 \ \ (i = 1, ..., l) \ . \end{aligned}$$

Die Schlupfvariablen gehen hier mit der gewichteten Manhatten-Norm (1-Norm) in die Zielfunktion ein.

Die Lagrange-Funktion hat dann die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\alpha}') = \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i\Big(y_i\big(\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{w}\rangle_F + b\big) - 1 + \xi_i\Big) - \sum_{i=1}^{l}\alpha_i'\xi_i$$

mit $\alpha_i \geq 0$ und $\alpha_i' \geq 0$.

Bei Herleitung der dualen Aufgabe kommen zu den Gleichungen (39) aus dem letzten Abschnitt noch die folgenden durch Ableiten nach ξ_i dazu:

$$C - \alpha_i - \alpha_i' = 0 \;\; (i = 1, ..., l) \; .$$

Die duale Aufgabe hat nach Substitution der Variablen die Form von (41), jedoch unter den zusätzlichen Bedingungen $\alpha_i \leq C \;\; (i = 1, ..., l)$.

Euklidische Distanz

$$\min_{\boldsymbol{w}\in F, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^l} \left(\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + C\sum_{i=1}^{l}\xi_i^2 \right) \tag{47}$$

unter den Nebenbedingungen

$$y_i \cdot f(\boldsymbol{x}^i) \geq 1 - \xi_i \;\; (i = 1, ..., l) \; .$$

Die Schlupfvariablen gehen hier mit dem gewichteten Quadrat der Euklidischen Norm (2-Norm) in die Zielfunktion ein.

Die Lagrange-Funktion hat dann die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 - \sum_{i=1}^{l}\alpha_i\Big(y_i\big(\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{w}\rangle_F + b\big) - 1 + \xi_i\Big)$$

mit $\alpha_i \geq 0$.

Die zusätzlichen Ableitungen haben jetzt die Form

$$C\xi_i - \alpha_i = 0 \;\; (i = 1, ..., l) \; .$$

Die Zielfunktion der dualen Aufgabe ergibt sich daraus als

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j)\rangle_F - \frac{1}{4C}\langle \boldsymbol{\alpha}, \boldsymbol{\alpha}\rangle_2 \; . \tag{48}$$

Sie unterscheidet sich von (41) durch einen zusätzlichen Term, der von C und dem Vektor der Lagrange-Multiplikatoren abhängig ist.

Durch Zusammenfassung der beiden hinteren Summanden von (48) ergibt sich eine etwas elegantere Form der Zielfunktion:

$$\begin{aligned} W(\boldsymbol{\alpha}) &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(\langle \boldsymbol{\phi}(\boldsymbol{x}^i) \, , \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F + \frac{1}{2C} \delta_{ij} \right) \\ &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C} \delta_{ij} \right) \, . \end{aligned}$$

Man hat analog zu (41) die Aufgabe

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^l} \quad W(\boldsymbol{\alpha}) \tag{49}$$

unter den Nebenbedingungen

$$\begin{aligned} \sum_{i=1}^{l} y_i \alpha_i &= 0 \, , \\ \alpha_i &\geq 0 \ \ (i = 1, ..., l) \end{aligned}$$

zu lösen.

Die Minimierungsaufgaben (46) und (47) unterscheiden sich nur dadurch, dass im ersten Fall alle ξ_i einfach aufsummiert werden, sie im zweiten Modell aber vorher quadriert werden. Welches Ziel wird mit der Quadrierung verfolgt? Eine positive Schlupfvariable ξ_i $(i \in \{1, ..., l\})$ verdeutlicht, dass der i-te Punkt den gewünschten funktionalen Abstand von mindestens eins verfehlt hat. Dabei kann $\boldsymbol{x}^i$ dennoch richtig klassifiziert sein, sodass ξ_i im Intervall $(0, 1]$ liegt, andernfalls gilt $\xi_i > 1$, der Punkt $\boldsymbol{x}^i$ ist also falsch klassifiziert worden. Quadrierung der Schlupfvariablen führt demnach dazu, dass Fehlklassifikationen $(\gamma_i < 0)$ stärker gewichtet werden als Klassifikationen mit positiven funktionalen Abständen γ_i $(0 \leq \gamma_i < 1)$. Bei beiden Ansätzen bleibt die Wahl des Parameters C offen. Klar ist, dass dieser eine Möglichkeit bietet, zu bestimmen, wie stark Klassifikationsfehler berücksichtigt werden sollen. Die Festlegung von C kann mit einer der Methoden, die unter dem Begriff Kreuzvalidierung zusammengefasst sind, erfolgen. Diese werden in Abschnitt 5.2 vorgestellt. Es ist auch möglich, die Schlupfvariablen mit einem Wert $k > 2$ zu potenzieren. Derartige Anwendungen sind aber bisher nicht bekannt.

3.4 Generalisierungstheorie

Ziel von Support-Vektor-Maschinen und anderen Methoden des überwachten Lernens ist immer, eine Funktion zu schätzen, die möglichst viele Daten richtig klassifiziert. Dabei handelt es sich entweder um Testdatensätze oder um bisher unbekannte Datensätze. Ein unbekannter Datensatz besteht nur aus Eingabewerten $\boldsymbol{x} \in X$, die einer Klasse zugeordnet werden sollen, ohne dass dabei weitere Informationen zur Verfügung stehen. Da man für solche Punkte nicht überprüfen kann, ob die Klassifikationen sinnvoll sind, ist es wichtig, bestimmte qualitative Merkmale der Entscheidungsfunktion zu untersuchen. Wie kann man die Qualität einer Schätzung bewerten?

Als Generalisierungs- oder Verallgemeinerungsfähigkeit bezeichnet man die Eigenschaft einer Hypothese, vom Training unabhängige Eingabedaten möglichst gut zu klassifizieren.

Diese Eigenschaft ist wünschenswert und kann daher als Maß für die Qualität von Lernalgorithmen betrachtet werden. Die Generalisierungstheorie untersucht also, ob und wann bestimmte Lernverfahren in der Lage sind, sinnvolle Ausgabewerte zu produzieren. Die schon vorgestellten Kernfunktionen erhöhen die Leistung von Lernalgorithmen, können aber auch dazu beitragen, dass Überanpassung entsteht. Ein Lernalgorithmus zeigt - wie schon auf Seite 100 erklärt wurde - überangepasstes Verhalten, wenn er die Trainingsdaten mit sehr wenigen Fehlern klassifiziert, jedoch bei Testdaten versagt.

Auch mit diesem Phänomen beschäftigt sich die Generalisierungstheorie. Zunächst wird die grundlegende Annahme dieser Theorie getroffen:

Der Trainingsdatensatz S_l ist eine zufällige Stichprobe aus dem Eingabe-Ausgabe-Raum $X \times Y$ gemäß einer festen aber unbekannten Verteilung P. Alle Paare $(\boldsymbol{x}^i, y_i),\ i = 1, ..., l$, sind somit Realisierungen unabhängiger und identisch verteilter Zufallsgrößen mit der Verteilung P. Diese Annahme gilt auch für alle Testdatensätze. Die Verteilung von S_l ist das l-fache Produktmaß von P. Es sei mit P_l bezeichnet.

Support-Vektor-Maschinen erlernen eine Zielfunktion der Form

$$f : X \to \mathbb{R}\ ,$$

welche dann im Fall binärer Klassifikation in die Entscheidungsfunktion

$$h : X \to Y,\ \ h := \operatorname{sgn}(f)$$

umgewandelt wird, siehe dazu Abschnitt 2.

Die Qualität einer Zielfunktion kann durch Einführung von Verlustfunktionen bewertet werden.

Eine Verlustfunktion ist eine Abbildung

$$c : X \times Y \times \mathbb{R} \to [0, \infty)\ . \tag{50}$$

Eine Verlustfunktion $c(\boldsymbol{x}, y, t)$ quantifiziert den „Verlust“ der entsteht, wenn die Zielfunktion f für ein Eingabe-Ausgabe-Paar $(\boldsymbol{x}, y)$ den Wert $t \in \mathbb{R}$ annimmt.

Die bekannteste Verlustfunktion ist die Funktion des $(0, 1)$-Verlustes

$$c(\boldsymbol{x}, y, t) = \begin{cases} 0 & \text{falls } \operatorname{sgn}(t) = y \\ 1 & \text{sonst} \end{cases} . \tag{51}$$

Hier werden alle Klassifikationsfehler gleich stark gewichtet. Der Verlust hängt nur davon ab, ob $\boldsymbol{x}$ durch die Zielfunktion in die richtige Klasse eingeordnet wurde oder nicht.

Etwas flexibler ist die Verlustfunktion

$$c(\boldsymbol{x}, y, t) = \begin{cases} 0 & \text{falls } \operatorname{sgn}(t) = y \\ \tilde{c}(\boldsymbol{x}, y) & \text{sonst} \end{cases} .$$

Durch eine Funktion $\tilde{c} : X \times Y \to [0, \infty)$ können bestimmte Fehler stärker gewichtet werden, zum Beispiel, wenn es von Interesse ist, in welcher Klasse der falsch eingeordnete Punkt liegt.

Die bisher vorgestellten Verlustfunktionen ordnen allen Entscheidungen, welche einen Eingabepunkt richtig klassifizieren, keinen Verlust zu. Das Konzept der Marge besagte aber, dass Klassifikationen innerhalb eines gewissen Grenzbereiches der trennenden Hyperebene unerwünscht sind. Deshalb gibt es auch Verlustfunktionen, die alle funktionalen Abstände, die kleiner als eins sind, als Verluste deklarieren:

$$c(\boldsymbol{x}, y, t) = \max\{0,\ 1 - y \cdot t\} = \begin{cases} 0 & \text{falls } y \cdot t \geq 1 \\ 1 - y \cdot t & \text{sonst .} \end{cases}$$

beziehungsweise

$$c(\boldsymbol{x}, y, t) = (\max\{0,\ (1 - y \cdot t)\})^2 .$$

Hat man sich entschieden, durch welche Verlustfunktion Fehler festgehalten werden sollen, kann man den erwarteten Verlust einer Zielfunktion betrachten.

Sei f eine Zielfunktion und c eine Verlustfunktion. Dann ist der erwartete Verlust, auch Risiko genannt, definiert als

$$R[f] := \mathbb{E}[c(\boldsymbol{x}, y, f(\boldsymbol{x}))] = \int_{X \times Y} c(\boldsymbol{x}, y, f(\boldsymbol{x}))\, dP(\boldsymbol{x}, y) . \tag{52}$$

Es ist klar, dass genau diese Größe Aussagen darüber macht, wie gut die Zielfunktion ist, aber man kann sie weder berechnen noch minimieren, da die Verteilung P unbekannt ist. Ein Ausweg aus diesem Dilemma wurde mit dem Prinzip der empirischen Risikominimierung zur Bewertung von Hypothesen (empirical risk minimization inductive principle) gefunden.

3.4.1 Empirische Risikominimierung

Das ERM-Verfahren setzt sich damit auseinander, zu überprüfen, wieviele Daten eines Trainingsdatensatzes durch eine feste Hypothese richtig klassifiziert werden. Die zu Grunde liegende Idee ist, dass aus einer Menge von Funktionen immer die gewählt werden sollte, die den Trainingsdatensatz am besten klassifiziert. Fehler, die bei Klassifikation des Trainingsdatensatzes mit der erlernten Funktion auftreten, liefern dieses empirische Maß. Es ist einfach zu bestimmen. Es wird davon ausgegangen, dass damit das Risiko von zukünftigen Fehlklassifikationen abgeschätzt werden kann. Es hat sich gezeigt, dass diese Methode nicht immer zu einer ausreichenden Generalisierungsfähigkeit führt und mit einer weiteren Idee kombiniert werden sollte [33]. Zunächst aber soll das ERM-Prinzip erklärt werden.

Für einen Trainingsdatensatz S_l, eine Zielfunktion f und eine Verlustfunktion c ist das empirische Risiko definiert als

$$R_{\text{emp}}[f] := \frac{1}{l} \sum_{i=1}^{l} c(\boldsymbol{x}^i, y_i, f(\boldsymbol{x}^i)) \; . \tag{53}$$

Das empirische Risiko erhält man also durch Einsetzen der empirischen Verteilung in (52).

Man betrachte die Verlustfunktion (51) und die Entscheidungsfunktion $h(\boldsymbol{x}) = \text{sgn}(f(\boldsymbol{x}))$. Dann ergibt sich für jedes Trainingspaar $(\boldsymbol{x}^i, y_i),\ i = 1, ..., l$, ein Verlust von

$$c_i = \frac{|h(\boldsymbol{x}^i) - y_i|}{2} \in \{0, 1\} \; .$$

Im Folgenden wird nur noch diese Verlustfunktion betrachtet. Sie ist typisch für das Problem der binären Klassifikation. Deshalb werden ab jetzt auch die Symbole $R[h]$ und $R_{\text{emp}}[h]$ anstelle von $R[f]$ und $R_{\text{emp}}[f]$ verwendet. Mit $\tilde{\mathcal{H}}$ sei der Raum aller Entscheidungsfunktionen $h = \text{sgn}(f),\ f \in \mathcal{H}$, bezeichnet. Damit entspricht $\tilde{\mathcal{H}}$ dem gewünschten Hypothesenraum.

Unter allen bisherigen Annahmen gilt für alle $\varepsilon > 0$, $h \in \tilde{\mathcal{H}}$ und $l \in \mathbb{N}$ die Ungleichung

$$P_l\{|R_{\text{emp}}[h] - R[h]| \geq \varepsilon\} \leq 2e^{-2l\varepsilon^2} \; . \tag{54}$$

Sie zeigt, wie nah das unbekannte Risiko R und das empirische Fehlermaß R_{emp} beieinander liegen. Zum Beweis wird ein Spezialfall der Hoeffding-Ungleichung [11], die Ungleichung von Chernoff [32], genutzt:

$$P_l\{|R_{\text{emp}}[h] - R[h]| \geq \varepsilon\} = P_l\left\{\left|\frac{1}{l}\sum_{i=1}^{l} c_i - \mathbb{E}[c]\right| \geq \varepsilon\right\} \overset{\text{Chernoff}}{\leq} 2e^{-2l\varepsilon^2} \; .$$

Die Minimierung des empirischen Risikos besteht darin, eine Funktion h zu suchen, die R_{emp} auf einem festen Trainingsdatensatz S_l minimiert. Das ERM-Prinzip schlägt diese Minimierung vor und geht davon aus, dass diese Methode ausreichend ist, um generalisierungsfähige Funktionen zu erlernen. Minimierung des empirischen Risikos kann unter Umständen zu Hypothesenfunktionen mit überangepasstem Verhalten führen, die im Extremfall alle Trainingspunkte korrekt klassifizieren, aber keine weiteren guten Klassifikationen hervorbringen. Das Prinzip ist daher nur dann sinnvoll, wenn einerseits das Risiko der gewählten Klassifikationsfunktion dem optimalen Risiko nahe kommt und wenn andererseits das empirische Risiko, welches diese Funktion verursacht, ein guter Schätzer für das wahre Risiko ist. Wie auch in [33] werden diese beiden Forderungen im Folgenden als Konsistenz der empirischen Risikominimierung bezeichnet, eine formale Definition dazu folgt nach einer kurzen Bemerkung zu zwei in diesem Abschnitt verwendeten Notationen.

(1) Eine Folge von Zufallsvariablen $\{X_n\}_{n\in\mathbb{N}}$ konvergiert stochastisch gegen eine Zufallsvariable X, wenn $\lim_{n\to\infty} P[\{|X_n - X| > \varepsilon\}] = 0 \ \forall\varepsilon \in (0,\infty)$; formal $X_n \xrightarrow[n\to\infty]{P} 0$.

(2) Für $a \in \mathbb{R}$ sei mit a_+ der Positivteil von a bezeichnet. Für reelle Zahlen a ist der Positivteil definiert als $\max\{0,\ a\}$.

$h^{\text{opt}} \in \tilde{\mathcal{H}}$ sei diejenige Funktion, die $R[h]$ minimiert. Dagegen sei $h^{S_l} \in \tilde{\mathcal{H}}$ die Funktion mit dem kleinsten empirischen Risiko für einen bestimmten Trainingsdatensatz S_l. Empirische Risikominimierung wird als konsistent bezeichnet, falls

$$R[h^{S_l}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0 \tag{55}$$

und

$$R_{\text{emp}}[h^{S_l}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0 \tag{56}$$

gelten.

Im Folgenden sollen hinreichende Bedingungen für die Konsistenz des ERM-Prinzips hergeleitet werden.

Für einen Hypothesenraum $\tilde{\mathcal{H}}$ sowie eine Verteilung P ist die Bedingung

$$\sup_{h\in\tilde{\mathcal{H}}}(R[h] - R_{\text{emp}}[h])_+ \xrightarrow[l\to\infty]{P} 0 \tag{57}$$

hinreichend für die Konsistenz der empirischen Risikominimierung.

Wir zeigen zunächst (55):

Für alle $h \in \tilde{\mathcal{H}}$ gelten die Ungleichungen

$$\begin{aligned} R[h] - R[h^{\text{opt}}] &\geq 0 \text{ und} \\ R_{\text{emp}}[h] - R_{\text{emp}}[h^{S_l}] &\geq 0 \,. \end{aligned}$$

Damit gelten für $h = h^{S_l}$ beziehungsweise $h = h^{\text{opt}}$

$$\underbrace{R[h^{S_l}] - R[h^{\text{opt}}]}_{(*)} \geq 0 \text{ und} \tag{58}$$

$$\underbrace{R_{\text{emp}}[h^{\text{opt}}] - R_{\text{emp}}[h^{S_l}]}_{(**)} \geq 0 \,. \tag{59}$$

Addition von (58) und (59) liefert

$$\begin{aligned} 0 &\leq R[h^{S_l}] - R[h^{\text{opt}}] + R_{\text{emp}}[h^{\text{opt}}] - R_{\text{emp}}[h^{S_l}] \\ &= R[h^{S_l}] - R_{\text{emp}}[h^{S_l}] + R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \\ &\leq \sup_{h \in \tilde{\mathcal{H}}}(R[h] - R_{\text{emp}}[h]) + R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \,. \end{aligned}$$

Aus (54) folgt: $R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0$.

Durch zusätzliche Verwendung von (57) erhält man die Aussage, dass $(*)$ und $(**)$ jeweils stochastisch gegen 0 konvergieren, also ist (55) gesichert.

(56) folgt direkt aus:

$$R_{\text{emp}}[h^{S_l}] - R[h^{\text{opt}}] = \underbrace{R_{\text{emp}}[h^{S_l}] - R_{\text{emp}}[h^{\text{opt}}]}_{\xrightarrow[l\to\infty]{P} 0} + \underbrace{R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}]}_{\xrightarrow[l\to\infty]{P} 0 \text{ nach } (54)} \,.$$

3.4.2 Theorie von Vapnik und Chervonenkis

Diese Theorie beschäftigt sich mit der Frage, wie Lernalgorithmen aufgebaut sein müssen, damit sie zu Hypothesen mit guten Generalisierungsfähigkeiten führen. Vapnik und Chervonenkis haben gezeigt, dass die Bedingung (57) für die sogenannte nicht-triviale Konsistenz der empirischen Risikominimierung nicht nur hinreichend, wie für die Konsistenz

nach (55) und (56), sondern auch notwendig ist. Zur Definition der nicht-trivialen Konsistenz siehe Abschnitt 2.1 in [33], das Theorem zur Konsistenz folgt dann im Abschnitt 2.2. An dieser Stelle wird weiterhin versucht, hinreichende Bedingungen für die Konsistenz des ERM-Prinzips anzugeben. Deshalb muss eine sinnvolle obere Schranke für den Ausdruck

$$P_l \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R[h] - R_{\text{emp}}[h]) > \varepsilon \right\} \tag{60}$$

hergeleitet werden. Vapnik und Chervonenkis treffen für $l\varepsilon^2 \geq 2$ die folgende Abschätzung [34]:

$$P_l \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R[h] - R_{\text{emp}}[h]) > \varepsilon \right\} \leq 2P_{2l} \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R^1_{\text{emp}}[h] - R^2_{\text{emp}}[h]) > \varepsilon/2 \right\} . \tag{61}$$

Dabei ist P_{2l} das $2l$-fache Produktmaß von P und bezieht sich auf Datensätze der Länge $2l$, die mit S_{2l} bezeichnet seien. R^1_{emp} bezieht sich dabei auf die ersten l Datenpaare, R^2_{emp} auf die andere Hälfte. Die Abschätzung ist nachvollziehbar: Liegen die empirischen Fehler zweier unabhängiger Datensätze nah beieinander, so muss das empirische Risiko nah am erwarteten Risiko liegen. Der rechte Teil von (61) muss weiter untersucht werden, um (60) nach oben abzuschätzen. Zunächst sollen einige Begriffe der VC-Theorie eingeführt werden.

Für eine Anzahl $l \in \mathbb{N}$ von Vektoren aus X gibt es maximal 2^l verschiedene Möglichkeiten, diese nach $\{-1, 1\}$ abzubilden. Es gilt also, dass die Funktionen $h \in \tilde{\mathcal{H}}$ eingeschränkt auf l Eingaben entweder genau 2^l oder weniger unterscheidbare Klassifikationen bereitstellen werden. Sei S_l ein Datensatz. Dann bezeichnet $\mathcal{N}^{S_l}(\tilde{\mathcal{H}})$ die maximale Anzahl der Funktionen aus $\tilde{\mathcal{H}}$, die man allein durch die Werte $h(\boldsymbol{x}^1), ..., h(\boldsymbol{x}^l)$ unterscheiden kann. Sei $l \in \mathbb{N}$ fest. Als Shattering-Koeffizient (shatter: zerrütten, zerschlagen) $\mathcal{N}^l(\tilde{\mathcal{H}})$ wird das Maximum von $\mathcal{N}^{S_l}(\tilde{\mathcal{H}})$ über alle möglichen S_l bezeichnet. Der Shattering-Koeffizient gibt an, wieviele verschiedene Ausgabefunktionen maximal auf einem Datensatz der Länge l durch $\tilde{\mathcal{H}}$ produziert werden können. Anders interpretiert gibt er die maximale Anzahl der Möglichkeiten an, einen Datensatz der Länge l durch die zur Verfügung stehenden Hypothesen in zwei Klassen einzuteilen. Existiert ein Datensatz S_l, sodass die Funktionen aus $\tilde{\mathcal{H}}$ in der Lage sind, die Eingabevektoren aller Datenpaare auf jede mögliche Art zu klassifizieren, dann gilt offensichtlich

$$\mathcal{N}^l(\tilde{\mathcal{H}}) = 2^l . \tag{62}$$

Unter allen bisherigen Annahmen gilt für das Produktmaß der rechten Seite von (61)

$$P_{2l}\left\{\sup_{h\in\tilde{\mathcal{H}}}(R^1_{\mathrm{emp}}[h]-R^2_{\mathrm{emp}}[h])>\varepsilon/2\right\}\leq 2\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\,e^{-\frac{l\varepsilon^2}{8}}\ .$$

Zum Beweis siehe [32]. Daraus und aus (61) folgt für $l\varepsilon^2\geq 2$ die Ungleichung

$$P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])>\varepsilon\right\}\leq 4\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\cdot e^{-\frac{l\varepsilon^2}{8}}\ . \tag{63}$$

Die folgende Abschätzung fasst alle bisherigen Ergebnisse zusammen und gibt eine praktische Aussage über das wahre Risiko bei Anwendung einer Klassifikationsfunktion.

Sei $\delta\in(0,1)$. Für Trainingsdatensätze der Länge l gelten mit einer Wahrscheinlichkeit von mindestens $(1-\delta)$ die Ungleichungen

$$R[h]\leq R_{\mathrm{emp}}[h]+\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)+\ln(4/\delta)\right)}\quad(h\in\tilde{\mathcal{H}})\ , \tag{64}$$

sofern l und $\varepsilon=\sqrt{(8/l)(\ln(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})])+\ln(4/\delta))}$ die Bedingung $l\varepsilon^2\geq 2$ erfüllen. Sei dazu $\delta\in(0,1)$ vorgegeben. Setze $\delta=4\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\cdot e^{-\frac{l\varepsilon^2}{8}}$, dann ergibt sich

$$\varepsilon=\sqrt{(8/l)(\ln(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})])+\ln(4/\delta))}\ .$$

Aus der Ungleichung (63) folgt für $l\varepsilon^2\geq 2$

$$\begin{aligned}
&P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])>\varepsilon\right\}\leq\delta\\
\Rightarrow\quad&P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])\leq\varepsilon\right\}\geq 1-\delta\\
\Rightarrow\quad&P_l\left\{(R[h]-R_{\mathrm{emp}}[h])\leq\varepsilon\ \ \forall h\in\tilde{\mathcal{H}}\right\}\geq 1-\delta\\
\Rightarrow\quad&P_l\left\{R[h]\leq R_{\mathrm{emp}}[h]+\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)+\ln(4/\delta)\right)}\ \ \forall h\in\tilde{\mathcal{H}}\right\}\geq 1-\delta\ .
\end{aligned}$$

Aus diesem Satz folgt, dass für $\delta\in(0,1)$ und $l\varepsilon^2\geq 2$ mit einer Wahrscheinlichkeit von mindestens $(1-\delta)$ die Ungleichung

$$R[h^{S_l}] \leq R_{\text{emp}}[h^{S_l}] + \underbrace{\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right) + \ln(4/\delta)\right)}}_{(KT)} \tag{65}$$

gilt.

Welche Vorteile oder Nachteile liefert die Abschätzung (65)?

- Vorteil: Die Abschätzung gilt für alle Funktionen des Hypothesenraumes in gleichem Maße. Die Ungleichung (65) ist daher nicht nur für Lernalgorithmen, die das ERM-Prinzip umsetzen, anwendbar, sondern bietet eine Abschätzung des Risikos für eine von einem beliebigen Lernalgorithmus gewählte Klassifikationsfunktion h^{S_l}.
- Nachteil: Kennt man von der gesuchten Hypothese h mehr als nur den Raum $\tilde{\mathcal{H}}$, aus dem sie stammt, wird man genauere Abschätzungen als (65) finden können.

Man betrachte nun die Struktur von (65). Der unbekannte Fehler wird durch zwei Terme additiv bestimmt. Wie verhalten sich die einzelnen Komponenten? Je reichhaltiger der Hypothesenraum, umso eher kann man eine Funktion mit kleinem empirischen Fehler wählen. Der zweite Ausdruck (KT) kann aber stark wachsen, wenn sich der Hypothesenraum vergrößert. Dazu sei an die Definition des Shattering-Koeffizienten erinnert. Im Folgenden sei dieser Teil als Kapazitätsterm bezeichnet.

Aus der Abschätzung (65) beziehungsweise (63) folgt die Bedingung (57), wenn ein geeigneter Hypothesenraum gewählt wird. Man kann also festhalten, dass man Konsistenz der empirischen Risikominimierung durch Festlegung geeigneter Räume $\tilde{\mathcal{H}}$ sichern kann. Was man unter einem geeigneten Hypothesenraum versteht und mit welchen weiteren Mitteln man das Risiko von Klassifikationsfunktionen abschätzen kann, wird am Ende dieses Abschnittes erklärt. Zunächst werden einige wichtige Werkzeuge der VC-Theorie vorgestellt, die dazu benutzt werden sollen, den Kapazitätsterm in (65) weiter nach oben abzuschätzen.

Die Höhe von $\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)$ im Kapazitätsterm beeinflusst das Risiko. Für $l \in \mathbb{N}$ und einen Hypothesenraum $\tilde{\mathcal{H}}$ ist die Wachstumsfunktion definiert als

$$G_{\tilde{\mathcal{H}}}(l) := \max_{S_l \in (X \times \{-1,1\})^l} \ln \mathcal{N}^{S_l}(\tilde{\mathcal{H}}) \, .$$

Offensichtlich gilt $G_{\tilde{\mathcal{H}}}(l) = \ln \mathcal{N}^l(\tilde{\mathcal{H}})$, denn die natürliche Logarithmusfunktion ist im gesamten Definitionsbereich streng monoton wachsend. Daraus erhält man die wichtige Beziehung

$$\ln\left(\mathbb{E}[\mathcal{N}^{S_l}(\tilde{\mathcal{H}})]\right) \leq G_{\tilde{\mathcal{H}}}(l) \, . \tag{66}$$

Das heißt, ist eine Aussage über den Wert der Wachstumsfunktion möglich, so lässt sich die Ungleichung (64) auswerten. In [33] wird eine wichtige Eigenschaft der Wachstumsfunktion angegeben, die hier zitiert werden soll. Entweder, die Wachstumsfunktion erfüllt für alle $l > 0$ die Gleichung

$$G_{\tilde{\mathcal{H}}}(l) = l \cdot \ln(2) \tag{67}$$

oder es existiert ein $\mathcal{D} \in \mathbb{N}$, sodass für $l \leq \mathcal{D}$ die Gleichung (67) gilt und für $l > \mathcal{D}$

$$G_{\tilde{\mathcal{H}}}(l) \leq \mathcal{D}\left(\ln \frac{l}{\mathcal{D}} + 1\right) \tag{68}$$

erfüllt ist. Diese Funktion wächst also linear mit der Größe des Trainingsdatensatzes, bis $l = \mathcal{D}$ erreicht ist. Danach steigt ihr Wert nur noch sehr langsam. Den Wert $\mathcal{D}$ nennt man nach Vapnik und Chervonenkis VC-Dimension. Sie gibt die maximale Anzahl l der Punkte an, welche die Bedingung (62) erfüllen. Falls für alle $l > 0$ Bedingung (67) erfüllt ist, setzt man $\mathcal{D} = \infty$.

Sei $X \subseteq \mathbb{R}^2$. Die VC-Dimension des Raumes der trennenden Hyperebenen in X, in diesem Fall aller Geraden, ist die größte Zahl $l \in \mathbb{N}$, sodass alle l-Tupel von Punkten aus X durch Hyperebenen auf jede erdenkliche Weise getrennt werden können. Man kann sich schnell klarmachen, dass dies für drei Punkte immer möglich ist. Bei vier Punkten kann es Situationen geben, in denen die gewünschten Zuordnungen zu 1 und -1 nicht ausschließlich durch Geraden erreicht werden können. Ein Beispiel ist in Abbildung 8 dargestellt. Daraus folgt, dass die VC-Dimension trennender Hyperebenen im $\mathbb{R}^2$ genau drei beträgt. Im Allgemeinen ist die VC-Dimension (n-1)-dimensionaler Hyperebenen immer n+1.

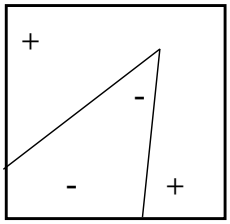


Abbildung 8: Nichtlineare Trennung von vier Punkten im $\mathbb{R}^2$.

Die Kenntnis der VC-Dimension ist ausreichend für eine Aussage über das Verhalten der Wachstumsfunktion und damit auch des Kapazitätsterms. Für eine bessere Übersicht soll die Ungleichung (64) mit diesen Erkenntnissen erweitert werden:

$$
\begin{aligned}
R[h] &\overset{(64)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right) + \ln(4/\delta)\right)} \\
&\overset{(66)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(G_{\tilde{\mathcal{H}}}(2l) + \ln(4/\delta)\right)} \\
&\overset{(68)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(\mathcal{D}\left(\ln\frac{2l}{\mathcal{D}} + 1\right) + \ln(4/\delta)\right)}. \qquad (69)
\end{aligned}
$$

Für $\mathcal{D} = \infty$ gilt (67), sodass der Kapazitätsterm wegen

$$
\frac{8}{l}\left(G_{\tilde{\mathcal{H}}}(2l)\right) \overset{(67)}{=} \frac{8}{l}\left(2l \cdot \ln(2)\right) = 16 \cdot \ln(2)
$$

eine Konstante enthalten würde. Bedingung (57) ist dann nicht notwendigerweise erfüllt. Zusätzlich existieren Abschätzungen nach unten [5], aus denen folgt, dass die lineare Lerntheorie in Merkmalsräumen sehr großer VC-Dimension versagt. Eben solche Räume sind laut Abschnitt 3.2 gewünscht. Der auf Seite 127 vorgestellte Kern führt sogar zu $\mathcal{D} = \infty$ [32], falls die Grammatrix vollen Rang besitzt, d.h. falls $\sigma \neq 0$ gilt und alle Trainingspunkte verschieden sind. Mit diesem Problem und dessen Lösung setzt sich der nächste Abschnitt auseinander. Im Folgenden werden Ansätze vorgestellt, mit denen das Risiko von Hypothesen genauer als bisher abgeschätzt werden kann.

3.4.3 Strukturelle Risikominimierung

Es wurde gezeigt, dass der Kapazitätsterm in (65) nicht von den Eigenschaften einer bestimmten Funktion in $\tilde{\mathcal{H}}$ abhängt, sondern durch Merkmale dieses Raumes bestimmt wird. Aus diesem Grund ist es nicht ausreichend, eine geeignete Funktion h zu wählen. Die Generalisierungsfähigkeit wird sich erst verbessern, wenn günstige obere Schranken für den Kapazitätsterm gefunden werden.

Ein erster Ansatz dafür ist die Einführung einer Struktur in $\tilde{\mathcal{H}}$, sodass die Summe aus empirischem Risiko und Kapazitätsterm über die Wahl der Struktur minimiert werden kann. Diese Methode wird als strukturelle Risikominimierung (structural risk minimization inductive principle) bezeichnet und erweitert das ERM-Prinzip. Strukturelle Risikominimierung innerhalb der VC-Theorie ist ein wichtiger Punkt, in dem sich Support-Vektor-Maschinen ganz klar von der Theorie der neuronalen Netze unterscheiden. Diese minimieren bestimmte Fehlermaße auf der Trainingsmenge, betrachten aber nicht die Eigenschaften der Hypothesenräume. Die Generalisierungsfähigkeit wird aber genau dadurch besser.

Idee: Man wähle einen Hypothesenraum $\tilde{\mathcal{H}}_1$, der sehr klein ist. Mit jedem Schritt vergrößere man diesen, also $\tilde{\mathcal{H}}_1 \subset \tilde{\mathcal{H}}_2 \subset ....$ Der Raum $\tilde{\mathcal{H}} := \tilde{\mathcal{H}}_k$ wird dann so gewählt, dass es eine Funktion $h \in \tilde{\mathcal{H}}$ mit einem kleinen empirischen Fehler gibt, wobei k möglichst klein sein soll. Gesucht sind also Hypothesenräume $\tilde{\mathcal{H}}_k \subset \tilde{\mathcal{H}}$ mit $\mathcal{D}_k \ll \infty$, in denen das ERM-Prinzip angewendet werden kann. Problematisch bei diesem Ansatz ist die Tatasche, dass im Allgemeinen für $\tilde{\mathcal{H}}$ keine Struktur bekannt ist, die die obige Idee realisieren könnte. Er ist also von geringem praktischen Nutzen. Sinnvoller ist es, wenn man Informationen nutzt, die durch die Trainingsdaten verfügbar sind. Damit beschäftigt sich der zweite Ansatz.

Wie man an (69) erkennen kann, liefert die VC-Theorie Schranken für das Risiko von Hypothesen, die von der Verteilung P unabhängig sind. Diese Schranken sind im Allgemeinen sehr groß, weil sie auch für ungünstige Verteilungen gelten müssen. Die Abschätzung des tatsächlichen Risikos wird damit schwierig. Kann man voraussetzen, dass die möglichen Verteilungen günstig sind, gibt es signifikant kleinere Schranken. Support-Vektor-Maschinen greifen diese Idee auf und suchen datenabhängige Fehlerschranken. Dabei werden Maße für die „Gutartigkeit“ von Verteilungen definiert, die dann angewendet auf den Trainingsdatensatz die zugehörigen empirischen Größen liefern. Diese Methode wird von SV-Maschinen umgesetzt und ist unter der Bezeichnung der datenabhängigen strukturellen Risikominimierung bekannt.

In Abschnitt 2.2 wurden die funktionalen Abstände der Punkte eines Trainingsdatensatzes von einer Hyperebene eingeführt. Die Güte einer Klassifikation kann deshalb über die Verteilung dieser Abstände bestimmt werden. Die Verfahren des Abschnittes 3.3 basieren auf Merkmalen dieser Verteilung. Die im Folgenden angegebenen datenabhängigen Risikoabschätzungen werden diese Verfahren rechtfertigen.

Die Klassifikation maximaler Trenngüte verwendet das Minimum der Verteilung der funktionalen Abstände. Es gilt [5]:

Sei $\mathcal{H}_1$ der Raum aller linearen Zielfunktionen f, die einen normierten Richtungsvektor $\boldsymbol{w}$ besitzen, $\gamma > 0$ sei eine reelle Zahl. Für jede Verteilung P auf $X \times \{-1, 1\}$, deren Träger in X in einer Kugel mit Radius R um den Ursprung liegt, gelten mit einer Wahrscheinlichkeit von $(1 - \delta)$ für Trainingsdatensätze S_l der Länge l die Ungleichungen

$$R[f] \leq \varepsilon(l, \mathcal{H}_1, \delta, \gamma) = \frac{2}{l}\left(\frac{64R^2}{\gamma^2}\log_2\frac{el\gamma}{8R^2}\log_2\frac{32l}{\gamma^2} + \log_2\frac{4}{\delta}\right) \quad \forall f \in \mathcal{H}_1 \,, \tag{70}$$

falls $l\varepsilon^2 \geq 2$ sowie $64R^2/\gamma^2 < l$ und falls eine Marge von mindestens γ eingehalten wird.

Man beachte, dass die Abschätzung (70) nicht mehr von der VC-Dimension abhängt. Kleine Fehlerschranken können realisiert werden, falls γ und l hinreichend groß sind. Damit ist die Marge direkt für die Fähigkeit der Generalisierung verantwortlich.

Verfahren für die Klassifikation nichtseparierbarer Trainingsdaten versuchen, eine vorgegebene funktionale Marge γ zu erreichen und berechnen dann das Minimum der Verteilung

einer Norm des Vektors der Schlupfvariablen $\boldsymbol{\xi}$. Es gelten die folgenden Abschätzungen, die ebenfalls aus [5] stammen:

Es sei $\mathcal{H}_1$ der Raum aller linearen Zielfunktionen f, die einen normierten Richtungsvektor $\boldsymbol{w}$ besitzen, $\gamma > 0$ sei eine reelle Zahl. Dann existiert eine Konstante c, sodass für jede Verteilung P auf $X \times \{-1, 1\}$, deren Träger in X in einer Kugel mit Radius R um den Ursprung liegt, mit einer Wahrscheinlichkeit von $(1 - \delta)$ für Trainingsdatensätze S_l der Länge l die folgenden Ungleichungen gelten:

$$R[f] \leq \frac{c}{l}\left(\frac{R^2 + \|\boldsymbol{\xi}\|_2^2}{\gamma^2}\log_2^2 l + \log_2\frac{1}{\delta}\right) \quad \forall f \in \mathcal{H}_1\ , \tag{71}$$

falls die 2-Norm verwendet wird, beziehungsweise

$$R[f] \leq \frac{c}{l}\left(\frac{R^2 + \|\boldsymbol{\xi}\|_1^2 \log_2(1/\gamma)}{\gamma^2}\log_2^2 l + \log_2\frac{1}{\delta}\right) \quad \forall f \in \mathcal{H}_1\ , \tag{72}$$

falls die 1-Norm gewählt wird.

In diesen Fällen ist der Generalisierungsfehler zusätzlich davon abhängig, wie stark jeder einzelne Trainingspunkt eine vorgegebene Marge γ verfehlt. An dieser Stelle soll noch gezeigt werden, wie man von diesen Abschätzungen zu den Verfahren des Abschnittes 3.3.2 gelangt. Im Fall der 2-Norm in (71) kann man erkennen, dass der Ausdruck

$$\frac{R^2 + \|\boldsymbol{\xi}\|_2^2}{\gamma^2} \tag{73}$$

direkten Einfluss auf die Fehlerschranke hat. Deshalb wäre es sinnvoll, diesen Ausdruck zu minimieren, wobei zu beachten ist, dass nach Voraussetzung $\|\boldsymbol{w}\| = 1$ gilt. Hebt man diese Einschränkung auf, erhält man den zu minimierenden Term

$$\frac{1}{\left(\gamma/\|\boldsymbol{w}\|\right)^2}\left(R^2 + \left(\frac{\|\boldsymbol{\xi}\|_2}{\|\boldsymbol{w}\|}\right)^2\right)\ .$$

Für $\gamma = 1$ und $C := 1/R^2$ ergibt sich die schon bekannte Darstellung

$$\min\left(\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_2^2\right)\ .$$

Die Abschätzung (72) kann in gleicher Weise untersucht werden und führt auf die Minimierung von

$$\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_1^2 \log_2(\|\boldsymbol{w}\|)\ .$$

Im Rahmen der Support-Vektor-Maschinen (siehe beispielsweise [5]) wird die leicht abgewandelte Aufgabe

$$\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_1$$

betrachtet.

Zusammenfassend zu diesem Abschnitt über Generalisierungstheorie passt folgendes Zitat aus [15]:

„ ... we can generalize well in high dimensional spaces, if our hypothesis has a small weight vector ... “.

3.5 Zusammenfassung

In diesem Abschnitt geben wir die für die Implementierung zu Grunde liegenden Modelle in kompakter Darstellung an. Ausgewählt wurden die beiden in Abschnitt 3.3.2 vorgestellten Modelle, die Trainingsfehler zugunsten einer generalisierungsfähigen Funktion tolerieren.

Die primale Aufgabe des 1-Norm Ansatzes hat die Form

$$\min_{\boldsymbol{w}\in\mathcal{F}, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^l} \quad \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i\right) \tag{74}$$

mit den Nebenbedingungen

$$\begin{aligned} y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i\rangle + b) &\geq 1 - \xi_i \ \ (i = 1, ..., l)\ , \\ \xi_i &\geq 0 \ \ (i = 1, ..., l)\ . \end{aligned}$$

Die zugehörige duale Aufgabe lautet

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) - \sum_{i=1}^{l}\alpha_i \tag{75}$$

mit den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T\boldsymbol{y} &= 0\ , \\ \mathbf{0} &\leq \boldsymbol{\alpha}\ , \\ \boldsymbol{\alpha} &\leq \boldsymbol{C}\ . \end{aligned}$$

Die primale Aufgabe des 2-Norm Ansatzes hat die Form

$$\min_{\boldsymbol{w}\in\mathcal{F},b\in\mathbb{R},\boldsymbol{\xi}\in\mathbb{R}^l} \quad \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i^2\right) \tag{76}$$

mit den Nebenbedingungen

$$y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i\rangle + b) \geq 1 - \xi_i \;\; (i = 1, ..., l) \; .$$

Die duale Aufgabe hat die Darstellung

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) &= \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C}\delta_{i,j}\right) - \sum_{i=1}^{l}\alpha_i \\ &= \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \tilde{K}(\boldsymbol{x}^i, \boldsymbol{x}^j) - \sum_{i=1}^{l}\alpha_i \end{aligned} \tag{77}$$

mit den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T\boldsymbol{y} &= 0 \, , \\ \boldsymbol{\alpha} &\geq \mathbf{0} \, . \end{aligned}$$

Wir haben hier die dualen Aufgaben als Minimierungsprobleme dargestellt, wobei sich keine Veränderung der Lösungen ergibt. Wie innerhalb dieses Abschnitts herausgearbeitet wurde, sind die dualen Darstellungen besonders wichtig für die Implementierung. Dazu seien die folgenden Punkte genannt.

- Die dualen Aufgaben sind konvex und besitzen deshalb keine lokalen Extrema.
- Jede duale Aufgabe besitzt genau einen globalen Extrempunkt.
- Die primalen Aufgaben arbeiten direkt mit Punkten des Merkmalsraumes. Innerhalb der Zielfunktionen der dualen Aufgaben treten ausschließlich Skalarprodukte zwischen Punkten im Merkmalsraum auf. Diese können durch Kernfunktionen die auf den Originaldaten arbeiten, ersetzt werden. Das ermöglicht die Nutzung hochdimensionaler Merkmalsräume.

Aus (40), (41) und (77) folgt direkt: Das $L2$-Norm-Modell entspricht dem *hard margin classifier* bis auf den modifizierten Kern

$$\tilde{K}(\boldsymbol{x}^i, \boldsymbol{x}^j) = K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C}\delta_{i,j} \; . \tag{78}$$

4 Algorithmen

In diesem Abschnitt werden die implementierten SVM-Algorithmen vorgestellt. Der hard margin classifier (37) ist für die Praxis nicht relevant, sodass die beiden soft margin Modelle umgesetzt worden sind. Sollte man im einen oder anderen Fall dennoch auf einer perfekten Trennung bestehen, kann das über die Wahl $C \approx \infty$, d.h. über einen sehr großen Wert für C erzwungen werden.

Abschnitt 4.1 wird sich mit der Implementierung des $L2$-Norm-Modells auseinandersetzen, wohingegen im Abschnitt 4.2 die Umsetzung einer $L1$-Norm Support-Vektor-Maschine vorgestellt wird. Wir werden im Anschluss daran erklären, welche Unterschiede zwischen diesen Modellen für die Praxis bestehen.

4.1 Nearest-Point-Algorithmen

Der iterative Nearest-Point-Algorithmus (NPA) wurde erstmals im Jahr 1999 vorgestellt [18]. Er löst SVM-Klassifikationsprobleme über folgende Aufgabe: Bestimme den minimalen Abstand zwischen zwei konvexen Polytopen, wobei die Polytope über die jeweiligen Trainingspunkte im Merkmalsraum definiert sind. Im Folgenden wird dieser Algorithmus vorgestellt. Im Anschluss daran wird gezeigt, wie der Algorithmus an die Projektaufgaben angepasst wurde.

Der Nearest-Point-Algorithmus löst das Trainingsproblem einer $L2$-Norm Support-Vektor-Maschine. Die zugehörige Optimierungsaufgabe ist in primaler (76) und dualer Form (77) bereits angegeben worden. NPA arbeitet unter zwei Bedingungen, die wir als erstes vorstellen, bevor im Anschluss auf die Details des Verfahrens eingegangen wird.

Es sei S_l ein Trainingsdatensatz. Seien $I := \{i \in \{1, ..., l\} \mid y_i = 1\}$ und $J := \{j \in \{1, ..., l\} \mid y_j = -1\}$.

Annahme 1: $I \neq \emptyset$ und $J \neq \emptyset$.

Diese Annahme stellt keine Einschränkung dar, denn ein Lernproblem mit ausschließlich Vertretern einer Klasse ist nicht sinnvoll. Dieser Fall könnte jedoch beim Aufspalten von Trainingsdaten für Validierungszwecke auftreten, siehe dazu Abschnitt 5.2. Insbesondere geschieht das oft beim Einsatz von Zufallsroutinen. Auch in den verkleinerten Trainingsdaten sollten immer Punkte beider Klassen vorhanden sein.

Aus den Eigenschaften von S_l folgen $I \cup J = \{1, ..., l\}$ und $I \cap J = \emptyset$.

Annahme 2: Es existieren $\boldsymbol{w}$ und b im zulässigen Bereich von (37).

Um auch nichtseparierbare Daten zu klassifizieren, wurden die abgeschwächten Klassifikationsverfahren eingeführt. Annahme 2 kann somit immer erfüllt werden. Es sei nochmals erwähnt, dass die Aufgabe (77) gelöst werden kann, indem man in der Ausgangsaufgabe (41) die Kernmatrix des Trainingsdatensatzes mit dem Parameter C anpasst. Der neue Kern $\tilde{K}$ wurde auf Seite 150 bereits definiert.

Im Folgenden wird deshalb nur (37) betrachtet und gezeigt, wie diese Aufgabe als Nearest-Point-Problem formuliert und gelöst werden kann.

Das in [18] vorgestellte Verfahren betrachtet Mengen und Punkte im Merkmalsraum F, dennoch wird die Merkmalsabbildung ϕ nicht als bekannt vorausgesetzt. Alle Berechnungen werden auch hier über einen Kern K realisiert. Das wird durch eine auf Skalarprodukten basierende Form ermöglicht. Details dazu folgen später.

4.1.1 Nearest-Point-Problem (NPP)

Sei $Z = \{\boldsymbol{z}^1, ..., \boldsymbol{z}^l\}$ eine endliche Menge von Punkten in F. Dann wird die konvexe Hülle $co(Z)$ von Z als

$$co(Z) := \left\{ \sum_{k=1}^{l} \beta_k \boldsymbol{z}^k \mid \boldsymbol{z}^k \in Z,\ \beta_k \geq 0,\ \sum_{k=1}^{l} \beta_k = 1 \right\}$$

definiert. Man betrachte jetzt die Mengen

$$\begin{aligned} U &:= co(\{\boldsymbol{\phi}(\boldsymbol{x}^i) \mid i \in I\}) \quad \text{und} \\ V &:= co(\{\boldsymbol{\phi}(\boldsymbol{x}^j) \mid j \in J\}) , \end{aligned}$$

die über die Merkmale in F definiert sind. Da I und J wegen $l \in \mathbb{N}$ endlich sind, stellen U und V konvexe Polytope in F dar.

Im Folgenden soll die Aufgabe

$$\min_{\boldsymbol{u} \in U,\ \boldsymbol{v} \in V} \quad \|\boldsymbol{u} - \boldsymbol{v}\|_F \tag{79}$$

betrachtet werden. Sie wird Nearest-Point-Problem genannt. Abbildung 9 veranschaulicht das Problem.

Das Paar $(\boldsymbol{w}^*, b^*)$ löst die Aufgabe (37) genau dann wenn $\boldsymbol{u}^* \in U$ und $\boldsymbol{v}^* \in V$ existieren, sodass $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ die Aufgabe (79) löst und

$$\boldsymbol{w}^* = \frac{2(\boldsymbol{u}^* - \boldsymbol{v}^*)}{\|\boldsymbol{u}^* - \boldsymbol{v}^*\|_F^2} \qquad b^* = \frac{\|\boldsymbol{v}^*\|_F^2 - \|\boldsymbol{u}^*\|_F^2}{\|\boldsymbol{u}^* - \boldsymbol{v}^*\|_F^2}$$

gelten ([18], Kapitel 2). Damit ist gesichert, dass ein Support-Vektor-Problem auch über die Minimierung eines Abstandes zwischen konvexen Polytopen gelöst werden kann.

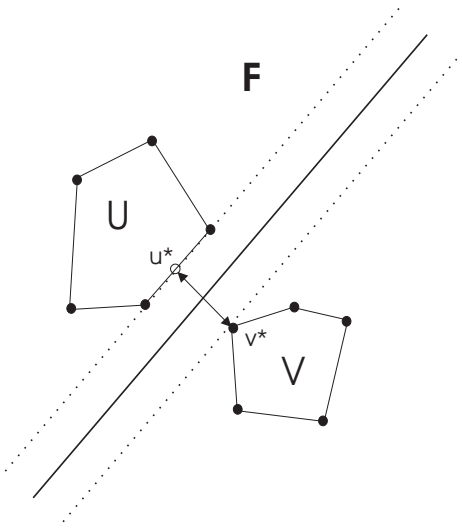


Abbildung 9: Nearest-Point-Problem im zweidimensionalen Merkmalsraum.

4.1.2 Optimalitätsbedingungen für NPP

Sei P eine kompakte Menge in F. Die sogenannte Support-Funktion $h_P : F \to \mathbb{R}$ wird definiert als

$$h_P(\boldsymbol{\vartheta}) := \max_{\boldsymbol{p} \in P} \quad \langle \boldsymbol{\vartheta}, \boldsymbol{p} \rangle_F \quad (\boldsymbol{\vartheta} \in F) \ .$$

Mit $s_P(\boldsymbol{\vartheta}) \in P$ wird eine zugehörige Lösung bezeichnet, d.h. $h_P(\boldsymbol{\vartheta}) = \langle \boldsymbol{\vartheta}, s_P(\boldsymbol{\vartheta}) \rangle_F$.

Sei nun P ein konvexes Polytop in F, also $P = co(Z)$ für eine Menge Z von Punkten in P. Es ist bekannt, dass das Maximum einer affin-linearen Funktion auf P in einem Punkt aus Z angenommen wird, genau genommen wird es in einem Punkt der minimalen Stützmenge angenommen, diese Menge ist aber in Z enthalten. Formal bedeutet das:

$$h_P(\boldsymbol{\vartheta}) = h_Z(\boldsymbol{\vartheta}) = \max_{\boldsymbol{z} \in Z} \quad \langle \boldsymbol{\vartheta}, \boldsymbol{z} \rangle_F \ ,$$

$$\exists\, \boldsymbol{z}^* \in Z : \ s_P(\boldsymbol{\vartheta}) = s_Z(\boldsymbol{\vartheta}) = \boldsymbol{z}^*, \quad \langle \boldsymbol{\vartheta}, \boldsymbol{z}^* \rangle_F = h_Z(\boldsymbol{\vartheta}) \ .$$

Die Funktion $g_P : F \times P \to \mathbb{R}$ wird definiert als

$$g_P(\boldsymbol{\vartheta}, \boldsymbol{p}) := h_P(\boldsymbol{\vartheta}) - \langle \boldsymbol{\vartheta} \ , \boldsymbol{p} \rangle_F .$$

Für alle $\boldsymbol{u} \in U$ und $\boldsymbol{v} \in V$ gelten

(1) $g(\boldsymbol{u}, \boldsymbol{v}) := g_U(\boldsymbol{v} - \boldsymbol{u}, \boldsymbol{u}) + g_V(\boldsymbol{u} - \boldsymbol{v}, \boldsymbol{v}) \geq 0$ **,**
denn $g_P(\boldsymbol{\vartheta}, \boldsymbol{p}) \geq 0 \ \ \forall\, \boldsymbol{\vartheta} \in F, \ \boldsymbol{p} \in P.$

(2) $\exists\, \overline{\boldsymbol{u}} \in U$ mit $\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{u}}\rangle_F < \langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{u}\rangle_F$
$\Rightarrow\ \exists\, \widetilde{\boldsymbol{u}} \in co(\{\boldsymbol{u}, \overline{\boldsymbol{u}}\})$ mit $\|\widetilde{\boldsymbol{u}}-\boldsymbol{v}\|_F < \|\boldsymbol{u}-\boldsymbol{v}\|_F$,
denn für eine Funktion ζ mit $\zeta(t) := \|\boldsymbol{u}+t(\overline{\boldsymbol{u}}-\boldsymbol{u})-\boldsymbol{v}\|_F^2\ (t\in\mathbb{R})$
gilt $\zeta'(t) = \sum 2(u_i + t(\overline{u}_i - u_i) - v_i)(\overline{u}_i - u_i)$.
Für $t=0$ folgt $\zeta'(0) = 2\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{u}}-\boldsymbol{u}\rangle_F < 0$.

(3) $\exists\, \overline{\boldsymbol{v}} \in V$ mit $\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{v}}\rangle_F > \langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}\rangle_F$
$\Rightarrow\ \exists\, \widetilde{\boldsymbol{v}} \in co(\{\boldsymbol{v}, \overline{\boldsymbol{v}}\})$ mit $\|\boldsymbol{u}-\widetilde{\boldsymbol{v}}\|_F < \|\boldsymbol{u}-\boldsymbol{v}\|_F$
Diese Aussage kann analog zu 2. gezeigt werden.

(4) $(\boldsymbol{u},\boldsymbol{v})$ löst $(79) \Leftrightarrow g(\boldsymbol{u},\boldsymbol{v}) = 0$
Gelte $g(\boldsymbol{u},\boldsymbol{v}) = 0$ und seien $\hat{\boldsymbol{u}} \in U,\ \hat{\boldsymbol{v}} \in V$ beliebig gewählt.
Wegen $g_U(\boldsymbol{v}-\boldsymbol{u}, \boldsymbol{u}) = 0$ und $g_V(\boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}) = 0$ folgen
$\langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{u}\rangle_F \le \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}\rangle_F$ und $\langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}\rangle_F \ge \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{v}}\rangle_F$.
Somit gilt $\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 \le \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\rangle_F$.
Zusammenfassend erhält man:

$$\begin{aligned}\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 &\le \|\boldsymbol{u}-\boldsymbol{v}\|_F^2 + \|(\boldsymbol{u}-\boldsymbol{v}) - (\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}})\|_F^2 \\ &= \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|_F^2 + 2\big(\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 - \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\rangle_F\big) \\ &\le \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|_F^2\ .\end{aligned}$$

Daraus folgt, dass $(\boldsymbol{u},\boldsymbol{v})$ Lösung von (79) ist.
Sei nur $(\boldsymbol{u},\boldsymbol{v})$ optimal für (79). Weiterhin seien $\overline{\boldsymbol{u}} = s_U(\boldsymbol{v}-\boldsymbol{u})$ und $\overline{\boldsymbol{v}} = s_V(\boldsymbol{u}-\boldsymbol{v})$.
Unter Beachtung von 1. und 2. folgt $g(\boldsymbol{u},\boldsymbol{v}) = 0$.

Algorithmen, die nach endlich vielen Schritten eine Lösung $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ finden, müssen viele Matrixoperationen durchführen. Sie sind für große Punktmengen nicht geeignet. Deshalb werden iterative Verfahren genutzt, die pro Schritt wenig Speicher benötigen, sich der Lösung aber nur asymptotisch nähern. Hat man geeignete Abbruchkriterien, führen diese Algorithmen dennoch zu befriedigenden Näherungen für $(\boldsymbol{u}^*, \boldsymbol{v}^*)$.

Der NP-Algorithmus aus [18] basiert auf zwei traditionellen Algorithmen, die sich sinnvoll ergänzen. Sowohl deren Grundideen als auch die Art und Weise ihrer Verknüpfung werden kurz vorgestellt.

4.1.3 Algorithmus von Gilbert

Eine Lösung für (79) kann auch über die Lösung eines äquivalenten Problems gefunden werden. Auf dieser Überlegung basiert die Methode nach Gilbert aus dem Jahr 1966.

Sei $Z := U \ominus V = \{\boldsymbol{u} - \boldsymbol{v} \mid \boldsymbol{u} \in U,\ \boldsymbol{v} \in V\}$ die sogenannte Minkowski-Differenz der Polytope U und V.

Offensichtlich gilt, dass die Aufgaben

$$\min_{\boldsymbol{z} \in Z} \quad \|\boldsymbol{z}\|_F \tag{80}$$

und (79) äquivalent sind.

Seien $\boldsymbol{u} \in U$ und $\boldsymbol{v} \in V$ beliebig. Dann haben sie die Darstellungen

$$\boldsymbol{u} = \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i), \quad \sum_{i \in I} \beta_i = 1, \quad \beta_i \geq 0$$

und

$$\boldsymbol{v} = \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j), \quad \sum_{j \in J} \beta_j' = 1, \quad \beta_j' \geq 0\ .$$

Der Punkt $\boldsymbol{z} = \boldsymbol{u} - \boldsymbol{v}$ ist also von der Form

$$\begin{aligned} \boldsymbol{z} &= \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i) - \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j) \\ &= \left(\sum_{j \in J} \beta_j'\right) \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i) - \left(\sum_{i \in I} \beta_i\right) \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j) \\ &= \sum_{i \in I} \sum_{j \in J} \beta_i \beta_j' \left(\boldsymbol{\phi}(\boldsymbol{x}^i) - \boldsymbol{\phi}(\boldsymbol{x}^j)\right)\ . \end{aligned}$$

Es gilt $\sum_{i \in I} \sum_{j \in J} \beta_i \beta_j' = \left(\sum_{i \in I} \beta_i\right) \left(\sum_{j \in J} \beta_j'\right) = 1$. Daraus folgt für die Menge Z als Differenz von Punkten aus U und V: $Z = co(Z_0)$, wobei

$$Z_0 := \{\boldsymbol{\phi}(\boldsymbol{x}^i) - \boldsymbol{\phi}(\boldsymbol{x}^j) \mid i \in I, j \in J\}\ .$$

Es gelten ([18], Theorem 2):

(1) Sei $\boldsymbol{z} \in Z$ beliebig. $\exists\ \overline{\boldsymbol{z}} \in Z$ mit $\|\boldsymbol{z}\|_F^2 - \langle \boldsymbol{z}, \overline{\boldsymbol{z}} \rangle_F > 0$, dann $\exists\ \widetilde{\boldsymbol{z}} \in co(\{\boldsymbol{z}, \overline{\boldsymbol{z}}\})$, sodass $\|\widetilde{\boldsymbol{z}}\|_F < \|\boldsymbol{z}\|_F$ gilt.

(2) Die Bedingung $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$ ist notwendig und hinreichend dafür, dass $\boldsymbol{z} \in Z$ die Aufgabe (80) löst.

Der Algorithmus von Gilbert nähert sich der Lösung $\boldsymbol{z}^*$ von (80) iterativ unter Zuhilfenahme dieser Erkenntnisse nach dem Schema in Tabelle 3.

Abbildung 10 stellt den Start und zwei Iterationen des Verfahrens im $\mathbb{R}^2$ dar.

1.	Wähle $\boldsymbol{z} \in Z$ beliebig.
2.	Berechne $h_Z(-\boldsymbol{z})$ und $g_Z(-\boldsymbol{z}, \boldsymbol{z})$.
3.	Falls $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$, dann $\boldsymbol{z}^* = \boldsymbol{z}$, STOP. Sonst setze $\overline{\boldsymbol{z}} = s_Z(-\boldsymbol{z})$.
4.	Berechne $\widetilde{\boldsymbol{z}}$ als den Punkt auf $\overline{\boldsymbol{z}\overline{\boldsymbol{z}}}$, der die kleinste Norm besitzt.
5.	Setze $\boldsymbol{z} = \widetilde{\boldsymbol{z}}$ und gehe zu 2.

Tabelle 3: Iterativer Algorithmus von Gilbert.

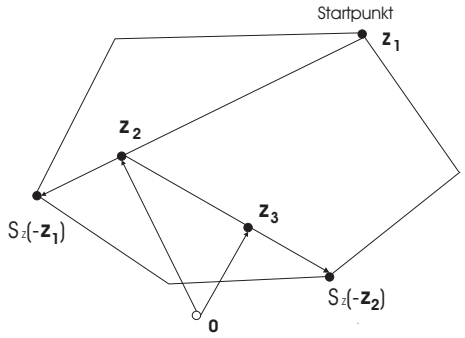


Abbildung 10: Zwei Iterationen nach Gilbert.

Der Algorithmus kann nur dann im Original verwendet werden, wenn die Merkmalsabbildung bekannt ist. Support-Vektor-Maschinen arbeiten jedoch mit einem Kern, der ohne Kenntnis des Merkmalsraumes gewählt wird. Deshalb ist es sinnvoll, alle $\boldsymbol{z} \in Z$ als konvexe Linearkombinationen der Elemente aus Z_0 darzustellen, das heißt:

$$\boldsymbol{z} := \sum_t \gamma_t \boldsymbol{z}_0^t, \quad \sum_t \gamma_t = 1, \quad \gamma_t \geq 0, \quad \boldsymbol{z}_0^t \in Z_0 \,, \tag{81}$$

wobei die Anzahl der Komponenten davon abhängt, wieviele Vertreter Z_0 hat, also wie sich die Trainingsdaten zusammensetzen. Die wesentliche Rolle im Algorithmus haben dann die Koeffizienten γ_t. Für weitere Details sei auf [18] verwiesen.

Der Algorithmus wird bei Annäherung an die Lösung sehr langsam. Den Grund dafür kann man sich an Abbildung 10 verdeutlichen. In jeder Iteration wird ein im Sinne der Aufgabe „guter“ Punkt $\boldsymbol{z}_0^{t'} = s_Z(-\boldsymbol{z})$ aus Z_0 bestimmt. Der zugehörige Koeffizient γ_t' in der Darstellung des aktuellen Punktes $\boldsymbol{z}$ wird vergrößert, betrachte zum Beispiel den Übergang von $\boldsymbol{z}_1$ nach $\boldsymbol{z}_2$ in Abbildung 10. Gleichzeitig werden alle anderen Koeffizienten in der Darstellung von $\boldsymbol{z}$ gleichmäßig verkleinert, unabhängig davon, ob sie zu relativ „guten“ oder „schlechten“ Eckpunkten gehören. Es wäre jedoch sinnvoller, wenn die Gewichte „schlechter“ Punkte schneller gegen Null gehen würden, als es hier der Fall ist. Die Methode nach Gilbert allein ist daher keine optimale Wahl für die Behandlung des Nearest-Point-Problems.

4.1.4 Algorithmus von Mitchell-Dem'yanov-Malozemov (MDM)

Der MDM-Algorithmus wurde erstmals im Jahr 1974 vorgestellt, siehe dazu [24]. Er löst ebenfalls die Aufgabe (80), wobei er die Darstellung (81) schon in seiner Ursprungsform voraussetzt.

Es kann Situationen geben, in denen der Algorithmus von Gilbert zu schlechter Performance führt. Man stelle sich vor, der aktuelle Punkt $\boldsymbol{z} \in Z$ liegt in der Nähe von $\boldsymbol{z}^* \in Z$. Trotzdem kann es $\gamma_t > 0$ geben, für die gilt

$$\langle \boldsymbol{z}, \boldsymbol{z}_0^t \rangle_F \gg \langle \boldsymbol{z}, \boldsymbol{z} \rangle_F \ . \tag{82}$$

Es ist einleuchtend, dass solche Koeffizienten γ_t den Punkt $\boldsymbol{z} \in Z$ nicht mitbestimmen, sondern eliminiert werden sollten. Dazu wird zunächst eine weitere Funktion eingeführt.

Die Funktion Δ über Z wird definiert als

$$\Delta(\boldsymbol{z}) := h_Z(-\boldsymbol{z}) - \underbrace{\min_{\gamma_t > 0} \langle -\boldsymbol{z}, \boldsymbol{z}_0^t \rangle_F}_{(*)} \ .$$

Der Punkt $\boldsymbol{z}_0^t$, der das Minimum $(*)$ realisiert, wird als $\boldsymbol{z}_0^{t_{min}}$ bezeichnet.

Es gelten $\Delta(\boldsymbol{z}) \geq 0$ auf ganz Z und $\Delta(\boldsymbol{z}) = 0$ genau dann wenn $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$. Aus diesem Grund ist es zulässig, $\Delta(\boldsymbol{z}) = 0$ anstelle von $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$ als Abbruchkriterium zu verwenden, siehe dazu Schritt drei im Algorithmus von Gilbert. Der MDM-Algorithmus bewegt sich ausgehend von einem $\boldsymbol{z}$ immer entlang der Richtung $s_Z(-\boldsymbol{z}) - \boldsymbol{z}_0^{t_{min}}$. Daraus leitet sich folgendes Prinzip ab: Für einen besonders „guten" Punkt $\boldsymbol{z}_0^{t'}$ aus Z_0 wird der Koeffizient in der Darstellung von $\boldsymbol{z}$ vergrößert, im Unterschied zu Gilbert aber auf Kosten eines einzigen „schlechten" Punktes $\boldsymbol{z}_0^{t_{min}}$. Die anderen Koeffizienten werden nicht angetastet. Somit ist es möglich, besonders unwichtige Punkte frühzeitig aus der Darstellung (81) zu eliminieren. Falls es also zu Phänomenen der Form (82) kommt, reagiert dieses Verfahren besser und führt schneller zu einem Abbruch, wie Tests im Zusammenhang mit der Arbeit [18] zeigten.

4.1.5 Nearest-Point-Algorithmus (NPA)

Auch wenn der MDM-Algorithmus in den meisten Fällen gute Ergebnisse liefert, ist es möglich, ihn zu verbessern. Das kann durch sinnvolle Kombinationen der beiden bisher vorgestellten Algorithmen erreicht werden.

Der Algorithmus vor Gilbert berechnet in seinen Iterationen hauptsächlich $h_Z(-\boldsymbol{z})$ und $s_Z(-\boldsymbol{z})$, siehe dazu nochmals Tabelle 3. Diese werden jedoch auch vom MDM-Algorithmus

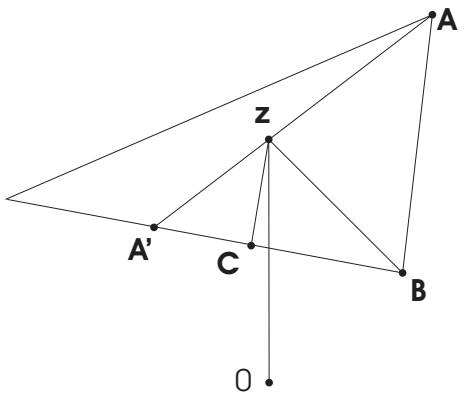


Abbildung 11: Mischalgorithmus für NPP.

zur Verfügung gestellt. Deshalb stellt es keinen Mehraufwand dar, die Ideen des Algorithmus von Gilbert im MDM-Algorithmus auszunutzen. Eine sehr günstige Variante soll noch kurz erklärt werden, siehe dazu Abbildung 11.

Die Bezeichnungen in Abbildung 11 haben folgende Bedeutungen:

- $\boldsymbol{A} := \boldsymbol{z}_0^{t_{min}}$ ist ein ungünstiger Punkt.
- $\boldsymbol{A'}$ realisiert die Verbindung von $\overline{\boldsymbol{Az}}$ mit dem Rand von Z.
- $\boldsymbol{B} := s_Z(-\boldsymbol{z})$ ist ein günstiger Punkt.
- $\boldsymbol{C}$ ist so gewählt, dass $\overline{\boldsymbol{zC}}$ parallel zu $\overline{\boldsymbol{AB}}$ ist und $\boldsymbol{C}$ auf der Strecke $\overline{\boldsymbol{A'B}}$ liegt.

Die Strecke $\overline{\boldsymbol{zB}}$ definiert den Bereich, auf dem Gilbert arbeitet, $\overline{\boldsymbol{zC}}$ hingegen jenen, auf dem MDM agiert. Eine Mischform wäre dann beispielsweise die Suche eines Punktes mit minimaler Norm im Dreieck $\boldsymbol{zCB}$. Falls der gefundene Punkt im Inneren dieses Dreiecks liegt, besagt diese Vorgehensweise:

- Der Koeffizient von $s_Z(-\boldsymbol{z})$ wird wie bei den ursprünglichen Verfahren vergrößert.
- Der Koeffizient von $\boldsymbol{z}_0^{t_{min}}$ wird deutlich stärker reduziert als bei Gilbert, aber er wird nicht so schnell eliminiert wie es beim MDM-Verfahren der Fall ist.
- Die restlichen Koeffizienten werden gleichmäßig reduziert und verzeichnen dadurch eine schwächere Abnahme als es beim Algorithmus von Gilbert der Fall ist. Sie werden jedoch im Gegensatz zum MDM-Algorithmus nicht völlig außer Acht gelassen.

Der in [18] vorgestellte Algorithmus geht sogar noch einen Schritt weiter. Er arbeitet auf dem Dreieck $\boldsymbol{zA'B}$. Ein durch eine solche Mischung entstandener Algorithmus ist schneller als alle in diesem Abschnitt vorgestellten Methoden. Details sollten in [18] nachgelesen werden.

Der Pseudocode in [18] sowie viele hilfreiche Anregungen von S. Sathiya Keerthi und Ricardo Henao trugen dazu bei, den NP-Algorithmus erfolgreich zu implementieren und einige Fehler der Originalarbeit zu korrigieren.

4.1.6 Gewichteter Nearest-Point-Algorithmus (GNPA)

Das allgemeine SVM-Modell unterscheidet grundsätzlich keine Fehlerarten. Klassifikationsfehler in den Trainingsdaten werden einfach oder quadratisch aufsummiert und mit dem Parameter C gewichtet. Damit gehen die Klasseninformationen komplett verloren. Ein Ausweg aus dieser Situation wird beispielweise in [26] vorgestellt.

Die Idee ist, Fehler erster Art härter zu bestrafen als Fehler zweiter Art. Das kann sehr einfach realisiert werden, indem der Parameter C auf zwei Parameter C^+ und C^- gesplittet wird. Bei einem Fehler erster Art wird mit C^+ gewichtet und bei einem Fehler zweiter Art mit C^-. Die Feststellung der Fehlerart ist trivial, da der Vektor $\boldsymbol{y}$ die Klassen jederzeit bereit hält.

Die Nutzung von C^+ und C^- führt im erfolgreichen Fall dazu, dass die optimale Hyperebene etwas näher an die Klasse -1 rückt (Abbildung 12). Anders ausgedrückt kann man sagen, dass sich die Gesamtmarge nicht mehr als Summe der beiden gleichen Einzelmargen m, sondern als $m_+ + m_-$ berechnet, wobei $m_+ > m_-$ gelten wird, falls $C^+ > C^-$. Der mit $\boldsymbol{x}$ gekennzeichnete Punkt in Abbildung 12 ist besonders interessant. Man sollte sich bewusst sein, dass bei sehr wenigen Punkten einer Klasse die Beeinflußung der Hypothese durch Ausreißer, beispielsweise durch falsche Angabe des Labels, viel höher ist als bei balancierten Daten, insbesondere wenn man mit einem großen Wert für C^+ arbeitet. Dieser führt dazu, dass um den Punkt $\boldsymbol{x}$ herum die Klasse 1 definiert wird. Ist dieser jedoch ein unerwünschter Ausreißer und gehört eigentlich in Klasse -1 oder als positiver Punkt in die Region der anderen Punkte der Klasse 1, ist die ursprüngliche Hyperebene die bessere.

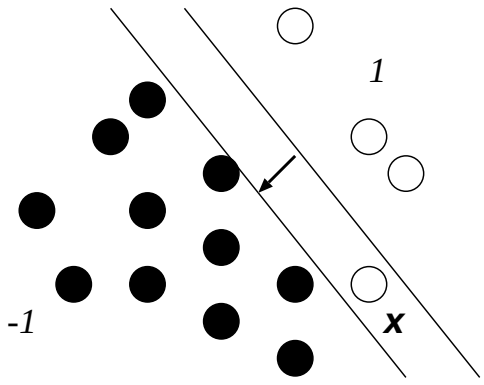


Abbildung 12: Verschiebung der Hyperebene durch C^-.

Die Konsequenzen daraus sind:

- Elemente des Klasse 1 werden verstärkt korrekt klassifiziert (true positive), auch wenn sie im Bereich der ursprünglichen Trennlinie liegen. Die Fehler erster Art nehmen ab.
- Elemente des Klasse -1 werden zunehmend falsch klassifiziert (false positive), wenn sie im Bereich der ursprünglichen Trennlinie liegen. Die Fehler zweiter Art nehmen zu.

Diesen Effekt kann man sich anhand der Abbildung 13 verdeutlichen, wobei sich die Trennfunktion auf der rechten Seite der Abbildung in Richtung der negativen Klasse verschoben hat.

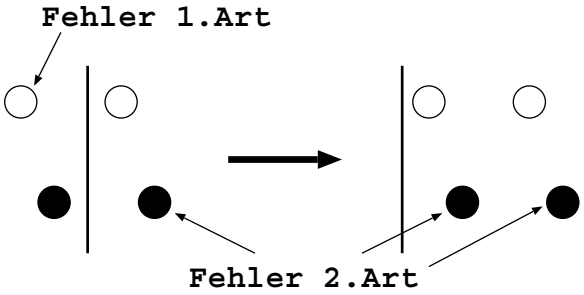


Abbildung 13: Fehler erster und zweiter Art in Konkurrenz.

Die Zielfunktion der ursprünglichen primalen Optimierungsaufgabe einer $L2$-Norm Support-Vektor-Maschine ändert sich wie folgt:

$$\min_{\boldsymbol{w}\in\mathcal{F},b\in\mathbb{R},\boldsymbol{\xi}\in\mathbb{R}^l} \quad \frac{1}{2}\|\boldsymbol{w}\|_{\mathcal{F}}^2 + C^+ \sum_{i:y_i=1} \xi_i^2 + C^- \sum_{j:y_j=-1} \xi_j^2 \,. \tag{83}$$

Die Nebenbedingungen bleiben identisch. Bei der Summierung der Fehler wird unterschieden, ob ein falsch negativer (erste Summe) oder falsch positiver Punkt (zweite Summe) betroffen ist.

Die Integration eines zweiten Fehlergewichtes war in diesem Fall sehr einfach. In der ursprünglichen dualen Optimierungsaufgabe (77) wird der Parameter C ausschließlich für den modifizierten Kern $\tilde{K}$, siehe (78), verwendet. Wir änderten diesen Kern wie in Abbildung 14 dargestellt. Weitere Änderungen waren nicht notwendig.

Die neue Zielfunktion der dualen Aufgabe hat dann wegen (41), (78) und den hier betrachteten Gewichten die Form

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left[K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \delta_{i,j} \left(\frac{\delta_{y_i,1}}{2C^+} + \frac{\delta_{y_i,-1}}{2C^-} \right) \right] \tag{84}$$

```
function kernel(i, j)
    kernel = compute-normal-kernel(i, j)
    if (i == j) then
        if (y_i == 1) kernel = kernel + 1/2C^+
        if (y_i == −1) kernel = kernel + 1/2C^−
    end if
end function kernel
```

Abbildung 14: Anpassung eines Kerns für das $L2$-Norm-Modell mit Fehlergewichtung.

unter den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T \boldsymbol{y} &= 0\,, \\ \boldsymbol{\alpha} &\geq \mathbf{0}\,. \end{aligned}$$

Die Nutzung dieser Gewichte ist für alle Kerne möglich, denn die ursprüngliche Kernfunktion wird nicht modifiziert, es werden lediglich die Ausgaben für Diagonalelemente der Kernmatrix geändert.

4.2 Sequential-Minimal-Optimization

Der Algorithmus Sequential-Minimal-Optimization (SMO) wurde 1998 von J. Platt entwickelt und vorgestellt [29]. Der SMO-Algorithmus gehört zu den sogenannten working set-Verfahren, welche die $L1$-Norm-Aufgabe (75) lösen. Diese Methoden verfahren nach dem Prinzip der iterativen Optimierung auf Teilmengen. Dazu wird in jeder Iteration ein reduziertes Optimierungsproblem gelöst. Dabei wird der Vektor $\boldsymbol{\alpha}$ in einen aktiven und einen nichtaktiven Teil zerlegt. Die Optimierungsarbeit findet dann nur auf dem aktiven Teil des Vektors (Arbeitsmenge, working set, chunk) statt. Die aktiven und inaktiven Indizes müssen nach jedem Schritt aktualisiert werden.

Das besondere am SMO-Algorithmus ist die Einschränkung der Größe des working set auf zwei Elemente. Im Folgenden werden wir zeigen, welche Aufgabe in jedem Teilschritt des Verfahrens gelöst werden muss und nach welcher Methode die globale Lösung generiert wird.

Die Matrix $\boldsymbol{Q}$ sei definiert als

$$Q_{ij} := y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) y_j \quad (1 \leq i, j \leq l)\,,$$

dann kann die Zielfunktion (75) kurz als

$$W(\boldsymbol{\alpha}) := \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Q}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{1} \tag{85}$$

angegeben werden.

4.2.1 Teiloptimierung

Wie schon erwähnt, wird die Aufgabe (85) iterativ auf einem Teil von $\boldsymbol{\alpha}$ gelöst. Wir definieren dazu eine Zerlegung von $\boldsymbol{\alpha}$, $\boldsymbol{y}$ und $\boldsymbol{Q}$:

$$\boldsymbol{\alpha} := \begin{pmatrix} \boldsymbol{\alpha}_{\mathrm{d}} \\ \boldsymbol{\alpha}_{\mathrm{f}} \end{pmatrix},$$

$$\boldsymbol{y} := \begin{pmatrix} \boldsymbol{y}_{\mathrm{d}} \\ \boldsymbol{y}_{\mathrm{f}} \end{pmatrix},$$

$$\boldsymbol{Q} := \begin{pmatrix} \boldsymbol{Q}_{\mathrm{dd}} & \boldsymbol{Q}_{\mathrm{df}} \\ \boldsymbol{Q}_{\mathrm{fd}} & \boldsymbol{Q}_{\mathrm{ff}} \end{pmatrix}.$$

Mit d bezeichnen wir jeweils den in der aktuellen Iteration dynamischen Teil von $\boldsymbol{\alpha}$. O.B.d.A. liegen die Daten jeweils sortiert nach dynamischen und festen Teilen f vor. Der Iterationsindex ist aus Gründen der Übersichtlichkeit nicht angegeben.

Diese Zerlegung kann in die Zielfunktion übernommen werden

$$W(\boldsymbol{\alpha}) = \frac{1}{2}\left(\boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{\mathrm{dd}}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{fd}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{df}\boldsymbol{\alpha}_{\mathrm{f}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{\mathrm{ff}}\boldsymbol{\alpha}_{\mathrm{f}}\right) - \boldsymbol{\alpha}_{\mathrm{d}}^T \mathbf{1} - \boldsymbol{\alpha}_{\mathrm{f}}^T \mathbf{1}. \tag{86}$$

Da der Vektor $\boldsymbol{\alpha}_{\mathrm{f}}$ für die spezielle Iteration konstant ist, können die Terme $\frac{1}{2}\boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{\mathrm{ff}}\boldsymbol{\alpha}_{\mathrm{f}}$ und $\boldsymbol{\alpha}_{\mathrm{f}}^T \mathbf{1}$ entfallen und wir erhalten eine neue Funktion von $\boldsymbol{\alpha}_{\mathrm{d}}$. Diese hat die Form

$$W_{\mathrm{d}}(\boldsymbol{\alpha}_{\mathrm{d}}) = \frac{1}{2}\boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{\mathrm{dd}}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{fd}\boldsymbol{\alpha}_{\mathrm{d}} - \boldsymbol{\alpha}_{\mathrm{d}}^T \mathbf{1}\,. \tag{87}$$

Die angepassten Nebenbedingungen lauten

$$\begin{aligned} \boldsymbol{y}_{\mathrm{d}}^T \boldsymbol{\alpha}_{\mathrm{d}} &= -\boldsymbol{y}_{\mathrm{f}}^T \boldsymbol{\alpha}_{\mathrm{f}}\,, \\ \boldsymbol{\alpha}_{\mathrm{d}} &\geq \mathbf{0}\,, \\ \boldsymbol{\alpha}_{\mathrm{d}} &\leq \boldsymbol{C}\,. \end{aligned}$$

Sie sichern, dass

$$\boldsymbol{\alpha}^{k+1} := \begin{pmatrix} \boldsymbol{\alpha}_{\mathrm{d}}^{k+1} \\ \boldsymbol{\alpha}_{\mathrm{f}}^{k} \end{pmatrix}$$

in jeder Iteration zulässig ist.

Für Sequential-Minimal-Optimization gilt: $|\boldsymbol{\alpha}_\mathrm{d}| = 2$. Das ist die minimale Größe, die ein working set haben darf. Das kann sehr einfach erklärt werden: Im k-ten Optimierungsschritt muss zur Einhaltung der Nebenbedingungen gelten

$$\boldsymbol{y}_\mathrm{d}^T \boldsymbol{\alpha}_\mathrm{d}^k = \boldsymbol{y}_\mathrm{d}^T \boldsymbol{\alpha}_\mathrm{d}^{k+1} = c \quad (c \in \mathbb{R} \text{ konstant}) \,. \tag{88}$$

Sollte $\boldsymbol{\alpha}_\mathrm{d}^k$ nur ein Element haben, führt jede Änderung dieses Multiplikators zu einer Verletzung dieser Bedingung. Eine Optimierung auf zwei Elementen dagegen ist möglich. Der Vorteil von SMO gegenüber anderen working set-Methoden mit großen Arbeitsmengen liegt in der Tatsache, dass das Teilproblem mit der Zielfunktion (87) bei zwei Punkten analytisch gelöst werden kann. Das soll im Folgenden gezeigt werden, wobei auch die entsprechenden Lösungen mit angegeben werden.

Wir bezeichnen mit i und j die in der k-ten Iteration betrachteten Indizes von $\boldsymbol{\alpha}$. Aus der Bedingung (88) ergibt sich

$$\begin{aligned} y_i\alpha_i^k + y_j\alpha_j^k = y_i\alpha_i^{k+1} + y_j\alpha_j^{k+1} &= c \quad | \cdot y_i \\ \alpha_i^k + s(i,j)\alpha_j^k = \alpha_i^{k+1} + s(i,j)\alpha_j^{k+1} &= \tilde{c}\,, \end{aligned} \tag{89}$$

wobei $s(i,j) = y_i \cdot y_j$ und $\tilde{c} \in \mathbb{R}$ nicht von α_i und α_j abhängen. Außerdem muss

$$0 \leq \alpha_i^k, \alpha_j^k, \alpha_i^{k+1}, \alpha_j^{k+1} \leq C \,. \tag{90}$$

auch weiterhin gelten.

Die Optimierung verläuft sehr einfach. O.B.d.A. soll α_j zuerst berechnet werden. Aus (89) und (90) folgen

1. für $y_i = y_j$:

$$\alpha_i^k + \alpha_j^k = \alpha_i^{k+1} + \alpha_j^{k+1} \;\Rightarrow\; \alpha_j^{k+1} = \alpha_i^k + \alpha_j^k - \alpha_i^{k+1}$$

$$\Rightarrow \alpha_j^{k+1} \in \Big[\max\big\{0, \alpha_i^k + \alpha_j^k - C\big\},\ \min\big\{C, \alpha_i^k + \alpha_j^k\big\} \Big]$$

und

2. für $y_i \neq y_j$:

$$\alpha_i^k - \alpha_j^k = \alpha_i^{k+1} - \alpha_j^{k+1} \;\Rightarrow\; \alpha_j^{k+1} = \alpha_i^{k+1} + \alpha_i^k - \alpha_j^k$$

$$\Rightarrow \alpha_j^{k+1} \in \Big[\max\big\{0, \alpha_j^k - \alpha_i^k\big\},\ \min\big\{C, C - \alpha_i^k + \alpha_j^k\big\} \Big] \,.$$

Bezeichnet man jeweils mit UG die untere und mit OG die obere Grenze, kann man das Ergebnis symbolisch darstellen als

$$\alpha_j \in [\mathrm{UG}, \mathrm{OG}] \tag{91}$$

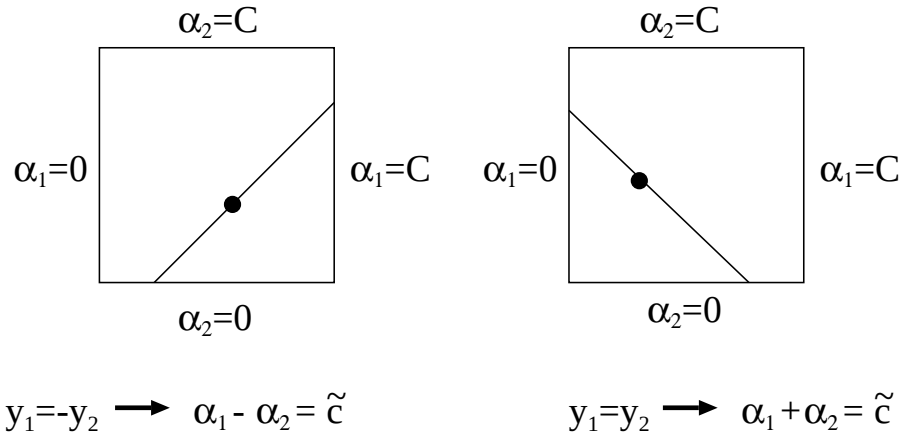


Abbildung 15: Boxen für zwei Lagrange-Multiplikatoren nach SMO.

und zeigt damit, dass der zweite Lagrange-Multiplikator immer zwischen bestimmten Grenzen liegt. Abbildung 15 stellt alle Bedingungen graphisch dar.

Mit $f(\boldsymbol{x})$ sei wie üblich der Wert der Klassifikationsfunktion bezeichnet, der sich bei der Auswertung an der Stelle $\boldsymbol{x}$ mit den aktuellen Parametern $\boldsymbol{\alpha}$ und b sowie einem fest gewählten Kern ergibt, also

$$f(\boldsymbol{x}) = \sum_{t=1}^{l} \alpha_t y_t K(\boldsymbol{x}^t, \boldsymbol{x}) + b. \tag{92}$$

Das bedeutet, der Wert für b muss ebenfalls immer wieder verändert werden, die Formeln dafür folgen später.

E_i symbolisiert die Differenz zwischen dem aktuellen Wert der Klassifikationsfunktion und der vorgegebenen Ausgabe im Punkt $\boldsymbol{x}^i$, also

$$E_i = f(\boldsymbol{x}^i) - y_i = \sum_{t=1}^{l} \alpha_t y_t K(\boldsymbol{x}^t, \boldsymbol{x}^i) + b - y_i \; .$$

Dieser Wert kann selbst bei richtiger Zuordnung von $\boldsymbol{x}^i$ groß werden.

Die α_t für $t \notin \{i, j\}$ sind in jeder Iteration konstant. Die Zielfunktion der dualen Aufgabe (85) kann man daher umformulieren zu

$$\begin{aligned} W(\alpha_i, \alpha_j) \;\; = \;\; & 0.5\alpha_i^2 K(\boldsymbol{x}^i, \boldsymbol{x}^i) + 0.5\alpha_j^2 K(\boldsymbol{x}^j, \boldsymbol{x}^j) + \alpha_i \alpha_j s(i,j) K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\ & + \sum_{t \notin \{i,j\}} \alpha_i \alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_j \alpha_t y_j y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) - \alpha_i - \alpha_j + c \; , \end{aligned}$$

wobei $c \in \mathbb{R}$ nicht von α_i und α_j abhängt, also konstant ist.

Ersetzt man α_i in der Zielfunktion gemäß (89) durch $\tilde{c} - s(i,j)\alpha_j$, ergibt sich eine Funktion von α_j:

$$\begin{aligned} W(\alpha_j) &= 0.5(\tilde{c} - s(i,j)\alpha_j)^2 K(\boldsymbol{x}^i, \boldsymbol{x}^i) + 0.5\alpha_j^2 K(\boldsymbol{x}^j, \boldsymbol{x}^j) \\ &\quad + s(i,j)(\tilde{c} - s(i,j)\alpha_j)\alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \sum_{t \notin \{i,j\}} (\tilde{c} - s(i,j)\alpha_j)\alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) \\ &\quad + \sum_{t \notin \{i,j\}} \alpha_j \alpha_t y_j y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) - \tilde{c} + s(i,j)\alpha_j - \alpha_j + c\,. \end{aligned}$$

Die notwendige Bedingung an einen stationären Punkt von W lautet

$$\begin{aligned} \frac{\partial W(\alpha_j)}{\partial \alpha_j} &= -s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^i)(\tilde{c} - s(i,j)\alpha_j) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)\alpha_j \\ &\quad + s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^j)\tilde{c} - 2K(\boldsymbol{x}^i, \boldsymbol{x}^j)\alpha_j - s(i,j) \sum_{t \notin \{i,j\}} \alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) \\ &\quad + y_j \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) + s(i,j) - 1 = 0\,. \end{aligned}$$

Deshalb und wegen $s(i,j)y_i = y_j$ muss für einen stationären Wert α_j^{st} von α_j die Gleichung

$$\begin{aligned} 0 &= s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^j)\tilde{c} - y_j\Bigg(\sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t)\Bigg) \\ &\quad - s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^i)(\tilde{c} - s(i,j)\alpha_j^{st}) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)\alpha_j^{st} - 2K(\boldsymbol{x}^i, \boldsymbol{x}^j)\alpha_j^{st} \qquad (93) \\ &\quad - 1 + s(i,j) \end{aligned}$$

erfüllt sein.

Durch nochmaliges Differenzieren erkennt man, dass α_j^{st} Maximalstelle ist, falls

$$\frac{\partial^2 W(\alpha_j^{st})}{\partial^2 \alpha_j^{st}} = 2K(\boldsymbol{x}^i, \boldsymbol{x}^j) - K(\boldsymbol{x}^i, \boldsymbol{x}^i) - K(\boldsymbol{x}^j, \boldsymbol{x}^j) < 0$$

gilt.

Sei $\eta = -2K(\boldsymbol{x}^i, \boldsymbol{x}^j) + K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)$. Einsetzen in (93) und Nutzung von $s(i,j) \in \{-1, 1\}$ liefert

$$\begin{aligned} -\alpha_2^{st} \cdot \eta &= -1 + s(i,j) - s(i,j) \cdot \tilde{c}(K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^i, \boldsymbol{x}^j)) \\ &\quad - y_j\Bigg(\sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) - \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t)\Bigg)\,. \end{aligned}$$

Multiplikation mit y_j ergibt

$$\begin{aligned}
-\alpha_j^{st} \cdot \eta \cdot y_j &= -y_j + y_i - y_i \cdot \tilde{c}(K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^i, \boldsymbol{x}^j)) \\
&\quad - \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) \\
&\overset{(92)}{=} -y_j + y_i + \tilde{c} \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^i) + \tilde{c} \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad + y_i \alpha_i^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) + f(\boldsymbol{x}^i) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad - f(\boldsymbol{x}^j) + y_i \alpha_i^k K(\boldsymbol{x}^j, \boldsymbol{x}^i) + y_j \alpha_j^k K(\boldsymbol{x}^j, \boldsymbol{x}^j) \; .
\end{aligned}$$

Elimination von $\tilde{c}$ durch Nutzung von (89) führt zur Darstellung

$$\begin{aligned}
-\alpha_j^{st} \cdot \eta \cdot y_j &= y_j - y_i + (\alpha_i^k + s(i,j) \cdot \alpha_j^k) \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad - (\alpha_i^a + s(i,j) \cdot \alpha_j^k) \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) + f(\boldsymbol{x}^i) \\
&\quad - y_i \alpha_i^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) - f(\boldsymbol{x}^j) \\
&\quad + y_i \alpha_i^k K(\boldsymbol{x}^j, \boldsymbol{x}^i) + y_j \alpha_j^k K(\boldsymbol{x}^k, \boldsymbol{x}^k) \\
&= y_j - y_i + f(\boldsymbol{x}^i) - f(\boldsymbol{x}^j) + y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) \\
&\quad - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) + y_j \alpha_j^k K(\boldsymbol{x}^j, \boldsymbol{x}^j) \\
&= -\alpha_j^k \eta y_j + \big(f(\boldsymbol{x}^i) - y_i\big) - \big(f(\boldsymbol{x}^j) - y_j\big)
\end{aligned}$$

und damit

$$\alpha_j^{st} = \alpha_j^k - \frac{y_j(E_i - E_j)}{\eta} \; .$$

Der Wert für α_j^{k+1} ergibt sich dann aus α_j^{st} oder aus einer der Grenzen aus (91), falls diese von α_j^{st} unter- oder überschritten werden. Die anschließende Berechnung von α_i^{k+1} ist trivial.

Der maximale Wert der Zielfunktion (85) unter der Bedingung, dass nur α_i und α_j geändert werden dürfen, wird also für die Werte

$$\begin{aligned}
\alpha_j^{k+1} &= \max\Big\{\mathrm{UG},\ \min\big\{\mathrm{OG}, \alpha_j^k - y_j(E_i - E_j)/\eta\big\}\Big\} \qquad (94) \\
&= \max\Big\{\max\big\{0, \alpha_j^k - \alpha_i^k\big\},\ \min\big\{C, C - \alpha_i^k + \alpha_j^k, \alpha_j^k - y_j(E_i - E_j)/\eta\big\}\Big\}
\end{aligned}$$

und

$$\alpha_i^{k+1} = \alpha_i^k + y_i y_j (\alpha_j^k - \alpha_j^{k+1})$$

angenommen, falls

$$\eta := -2K(\boldsymbol{x}^1, \boldsymbol{x}^2) + K(\boldsymbol{x}^1, \boldsymbol{x}^1) + K(\boldsymbol{x}^2, \boldsymbol{x}^2)$$

strikt positiv ist.

Wie auch in [29] nachzulesen ist, entsteht durch die Forderung $\eta > 0$ keine besondere Problematik. Folgende Fälle sollte man aber ausschließen:

(1) Erfüllt der Kern K nicht die Bedingungen des Theorems von Mercer, kann $\eta < 0$ auftreten.

(2) Sind zwei Eingabevektoren identisch, wird $\eta = 0$ auftreten, sobald diese beiden ausgewählt werden.

4.2.2 Heuristiken

Bisher wurde hergeleitet, wie zwei ausgewählte Lagrange-Multiplikatoren zu optimieren sind. Es bleibt zu klären, wie man in jedem Iterationsschritt des Verfahrens zwei passende Indizes auswählt und welche Größen zusätzlich angepasst werden müssen.

Die Konvergenzgeschwindigkeit des Verfahrens erhöht sich, wenn diejenigen Indizes i und j gewählt werden, welche die KKT-Bedingungen am stärksten verletzen. Diese Bedingungen für alle Punkte auszuwerten erfordert einen zu großen Rechenaufwand. Deshalb werden Heuristiken herangezogen, die die KKT-Bedingungen nur für einen Teil der Daten überprüfen und so den Gesamtaufwand reduzieren. Die Auswahl dieser aktiven Punkte erfolgt derart, dass die Zielfunktion einen möglichst großen Zuwachs erfährt.

- Heuristik für i: Dieser Index wird aus der Menge der Punkte ermittelt, welche die KKT-Bedingungen verletzen. Dafür wird eine Fehlertoleranz ε benutzt. Der Wert für ε sollte nach [29] zwischen 10^{-2} und 10^{-3} liegen. Dabei wird zunächst nicht darauf geachtet, in welchem Intervall die zugehörigen Lagrange-Multiplikatoren liegen, sondern der erste gefundene Punkt wird ausgewählt. Im Anschluss bestimmt die zweite Heuristik - siehe nächster Punkt - den anderen an der Optimierung beteiligten Index und ein Iterationsschritt kann durchgeführt werden. In den weiteren Iterationen sucht diese Heuristik nur noch nach Punkten, welche die KKT-Bedingungen verletzen **und** deren Lagrange-Multiplikatoren im offenen Intervall $(0, C)$ liegen, bis der erste gefunden ist. Erst wenn keine solchen Punkte mehr vorhanden sind, wird die Suche wieder auf alle Werte für α_i ausgedehnt.
- Heuristik für j: Der Index muss so gewählt werden, dass der während der Optimierung durchgeführte Schritt möglichst groß ist. Man betrachte nochmals die Darstellung (94). Offensichtlich ist die Änderung des zweiten Lagrange-Multiplikators von

η sowie von $|E_i - E_j|$ abhängig. Die Berechnung von η ist wegen der drei Kernauswertungen sehr zeitintensiv. Deshalb sollte man j so wählen, dass es den Ausdruck $|E_i - E_j|$ maximiert. Der Vorteil dieser Wahl ist, dass der Vektor $\boldsymbol{E}$ im Gegensatz zur Kernmatrix während der Laufzeit mitgeführt wird. Damit kann viel Rechenzeit eingespart werden.

Unter ungünstigen Voraussetzungen kann es dazu kommen, dass diese Heuristik keine Verbesserung liefert [29]. In diesen Fällen benutzt SMO bis zu zwei weitere Heuristiken. Zunächst werden alle Indizes $j \neq i$, deren Lagrange-Multiplikatoren α_j im Intervall $(0, C)$ liegen, der Reihe nach untersucht, wobei bei einem zufällig bestimmten Punkt angefangen wird. Liefert auch diese Suche keine Verbesserung, werden im letzten Schritt alle Punkte überprüft, also auch die Grenzfälle $\alpha_j = 0$ und $\alpha_j = C$, wobei auch hier bei einem zufälligen Punkt gestartet wird. Liefern diese drei Möglichkeiten zu einem bestimmten i kein passendes j, wird keine Optimierung und Aktualisierung durchgeführt, sondern SMO wählt ein neues i. Laut [29] tritt dieser Fall aber sehr selten auf.

4.2.3 Aktualisierungen

Der Wert von b muss nach jedem Iterationsschritt aktualisiert werden, denn die beiden betrachteten Punkte müssen nach dem Optimierungsschritt die KKT-Bedingungen erfüllen. Zu Beginn des Verfahrens gilt $b = 0$.

Zunächst betrachte man α_i:

$$\alpha_i^{k+1} \in (0, C) \qquad \Rightarrow f(\boldsymbol{x}^i) \stackrel{!}{=} y_i \; . \tag{95}$$

In diesem Fall ergibt sich ein b_1 aus

$$-b_1 = E_i^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^i) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^i, \boldsymbol{x}^j) - b^k \; .$$

Analog wird α_j ausgewertet:

$$\alpha_j^{k+1} \in (0, C) \qquad \Rightarrow f(\boldsymbol{x}^j) \stackrel{!}{=} y_j \; . \tag{96}$$

In diesem Fall ergibt sich b_2 aus

$$-b_2 = E_j^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^j) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^j, \boldsymbol{x}^j) - b^k \; .$$

Falls nur eine der Bedingungen erfüllt ist, wird $b^{k+1} = b_1$ (95), beziehungsweise $b^{k+1} = b_2$ (96) gesetzt. Falls beide erfüllt sind, gilt $b^{k+1} = b_1 = b_2$. Für den Fall, dass beide Lagrange-Multiplikatoren einen Grenzfall annehmen, also

$$\alpha_i^{k+1} \in \{0, C\} \text{ und } \alpha_j^{k+1} \in \{0, C\} \; ,$$

stellen alle Werte zwischen b_1 und b_2 zulässige Werte für b^{k+1} dar. Das Verfahren SMO wählt in diesem Fall die Mitte des Intervalls.

Der Fehlervektor $\boldsymbol{E}$ wird während der gesamten Laufzeit mitgeführt und muss für alle Indizes t $(t \neq i, j)$ mit $0 < \alpha_t < C$ angepasst werden durch

$$E_t^{k+1} = E_t^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^t) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^j, \boldsymbol{x}^t) - b^k + b^{k+1} \ .$$

Der SMO-Algorithmus nach J. Platt wurde im Rahmen des GALA Projektes implementiert. Der Pseudocode in [29] war dazu äußerst hilfreich. John Platt weist auf seiner Homepage auf einige Fehler im Paper hin. Seine Hinweise wurden berücksichtigt.

4.2.4 Gewichteter SMO-Algorithmus (GSMO)

Analog zum Algorithmus GNPA, der auf Seite 159 vorgestellt wurde, kann auch das SMO-Verfahren in einer gewichteten Variante definiert werden. Dabei ändert sich bei der Aufgabe (75) lediglich die dritte Nebenbedingung. Sie lautet dann

$$\forall i = 1, \ldots, l: \quad \alpha_i \leq \begin{cases} C^+ & \text{falls } y_i = 1 \ , \\ C^- & \text{sonst} \ . \end{cases} \tag{97}$$

Bei der Implementierung der gewichteten Variante von SMO muss an den jeweiligen Stellen im Quellcode, an denen der Parameter C verwendet wird, in Abhängigkeit der Klassenzugehörigkeit des betrachteten Trainingspunktes entweder C^+ oder C^- gewählt werden. Eine einfache Änderung der Kernfunktion analog zu der Vorgehensweise bei GNPA, siehe Abbildung 14, ist leider nicht möglich, da der Parameter C nicht in die Kernfunktion integriert werden kann. Details zum Quellcode von SMO können in [5] nachgelesen werden.

4.3 Zusammenfassung

In diesem Abschnitt sind die implementierten Verfahren zum Trainieren von Support-Vektor-Maschinen vorgestellt worden. Grob gesagt, stehen für die Algorithmen zwei unterschiedliche Modelle zur Verfügung – $L1$-Norm und $L2$-Norm. Das hard margin Modell kann über $C = \infty$ sowohl mit NPA als auch mit SMO trainiert werden. Fraglich für die Anwendung ist, welcher Algorithmus erfolgreicher arbeitet. Diese Frage läßt sich allgemein nicht beantworten. $L1$-Norm- und $L2$-Norm-Modell unterscheiden sich in der Formulierung der primalen Optimierungsaufgabe ausschließlich in der Art der Summierung der Schlupfvariablen ξ_i $(i = 1, ..., l)$. Für das $L2$-Norm-Modell werden diese Werte quadriert. Bei zahlreichen Tests hat sich gezeigt, dass die mittels NPA und SMO generierten Ergebnisse sehr ähnlich sind, wobei der Nearest-Point-Algorithmus weniger Rechenzeit in Anspruch nimmt. Im Gegenzug wirkt sich das $L1$-Norm-Modell positiver auf die Anzahl

der Support-Vektoren aus, ihre Anzahl ist geringer als mit NPA. Zusätzlich zu den Standardalgorithmen haben wir die Möglichkeit mittels C^+ und C^- Fehler zu gewichten und somit falsch positive oder falsch negative Klassifikationen zu verringern.

5 Optimierung

Die Algorithmen aus den Abschnitten 4.1 und 4.2 allein reichen nicht aus, um Support-Vektor-Maschinen erfolgreich zu trainieren. Für die Untersuchung von großen empirischen Datensätzen ist es außerdem erforderlich, hinreichend gutartige Trainingsdaten zu verwenden sowie ein geeignetes Modell mit optimalen Parameterwerten festzulegen. In den folgenden beiden Abschnitten werden wir kurz auf diese Aufgaben eingehen.

5.1 Datenvorverarbeitung

Datensäuberung und -reduktion als Bestandteile des Data Mining sind sehr wichtig als vorverarbeitende Schritte bei der Klassifikation von Daten. Diese Abhängigkeit wird oft unterschätzt. Auch das beste Klassifikationsverfahren erzielt schlechte Ergebnisse, wenn sich in den Daten zu viele Ausreißer und Fehler unerkannt verbergen.

Zur Datensäuberung zählen beispielsweise

- Entfernen von offensichtlichen Fehlern, z.B. nichtreelle Einträge,
- Ersetzen von Missing Values durch „sinnvolle“ Werte,
- Standardisierung oder Transformation von Variablen,
- Erkennen und Entfernen von Variablen ohne Varianz.

Diese Aufgaben werden durch die in Teil II dieses Bandes vorgestellten Methoden abgedeckt. Datensäuberung führt dazu, dass das SVM-Training einfacher und zuverlässiger wird; das konnte in verschiedenen Tests gezeigt werden.

Zusätzlich zur Säuberung fehlerhafter Daten beschäftigen wir uns mit Methoden zur Datenreduktion. Durch eine Verkleinerung des Datenraumes können beispielsweise die Rechenzeiten von Klassifikationsverfahren reduziert werden. Im Allgemeinen bezweckt man damit jedoch bessere Ergebnisse und Interpretationsmöglichkeiten der Klassifikationen.

Wir nutzen Methoden der Datenreduktion für die beiden Aufgaben

- Erkennen und Entfernen von unerwünschten Ausreißern und
- Entfernen unwichtiger Variablen.

Im Teil II dieses Berichtes sind diese Methoden detailliert beschrieben.

Beim Zusammenspiel von Datenreduktion und überwachtem Lernen mit Support-Vektor-Maschinen oder anderen Klassifikationsverfahren gibt es ein wichtiges Detail, welches bisher nicht erwähnt wurde. Das Identifizieren von unerwünschten Messungen und überflüssigen Variablen beruht auf statistischen Annahmen, die völlig frei von Informationen zur Verteilung der Klassen sind. Diese Methoden zählen dann zu den sogenannten unüberwachten Verfahren. Eine starke Datenreduktion ohne Betrachtung der Klassen kann dabei durchaus zum Verlust signifikanter Abhängigkeiten und nichtlinearer Funktionalitäten führen. Einen Ausweg aus diesem Dilemma stellt eine getrennte Untersuchung der beiden Klassen dar. Eine weitere Aufgabe innerhalb des GALA-Projektes bestand in der Untersuchung und Umsetzung von überwachten Methoden zur Variablenselektion. Diese werden in diesem Band jedoch nicht näher beschrieben. Einen kleinen Einblick zu dieser Arbeit gibt [19].

5.2 Modelloptimierung

Für das Training einer Support-Vektor-Maschine müssen neben der Wahl eines geeigneten Kerns verschiedene Parameterwerte an die jeweiligen Daten angepasst werden. Über diese Parameter und deren Funktion ist schon ausführlich in den Abschnitten 3.2.3 und 3.3 diskutiert worden. Die Generalisierungsfähigkeit einer SVM, d.h. die Qualität der Klassifikation, hängt von den gewählten Parameterwerten ab [25]. Ungünstige Parameter führen zum Scheitern einer Support-Vektor-Maschine, und, je besser man die Parameterwerte festlegt, umso verlässlicher ist auch die erlernte Klassifikationsfunktion. Das Problem der Parameteroptimierung für Support-Vektor-Maschinen ist ein in den letzten Jahren stark diskutiertes Thema [17, 25, 3, 14], wird aber dennoch zu oft unterschätzt. Oft werden Ergebnisse von neu entwickelten Methoden mit Ergebnissen von SVM-Software ohne Parametertuning verglichen [1].

Wir führen Parameteroptimierung auf der Grundlage der während einer Kreuzvalidierung entstandenen Fehler durch. Der Trainingsdatensatz S_l wird dafür in $p,\ 1 < p \leq l$, Teile möglichst gleicher Größe aufgeteilt. Der Testdatensatz wird vorerst nicht betrachtet. Abbildung 16 stellt die von uns implementierte Form der zehnfachen Kreuzvalidierung dar. Zusätzlich haben wir für moderate Datensätze die sogenannte leave one out Kreuzvalidierung $(p = l)$ implementiert.

Für eine Parameteroptimierung werden iterativ Kreuzvalidierungen mit verschiedenen Parameterwerten durchgeführt, siehe dazu Tabelle 4.

An dieser Stelle bleibt noch zu klären, welches Maß darüber entscheidet, ob ein Ergebnis zufriedenstellend ist, und wie die neu zu testenden Parameterwerte definiert werden (Schritt 5).

Wir stellen nun das implementierte Gütemaß vor. Dazu müssen zunächst einige grundlegende Kennzahlen definiert werden.

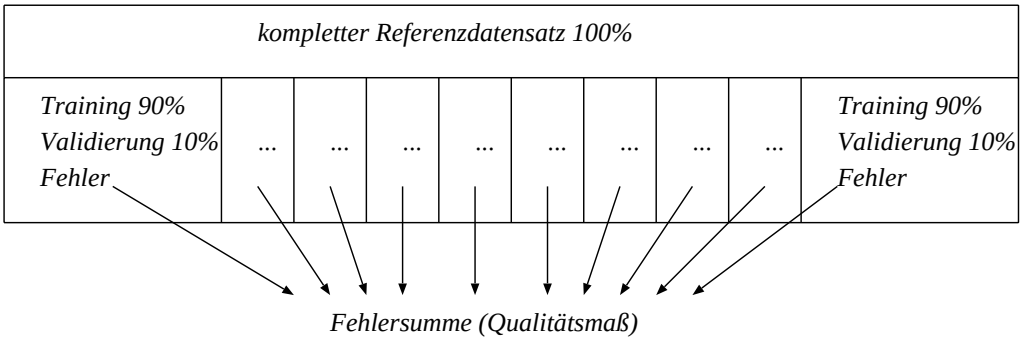


Abbildung 16: Zehnfache Kreuzvalidierung.

1.	Wähle Startwerte für Parameter, z.B. $\sigma = 1$ und $C = 1$; $i = 1$
2.	Führe das Lernverfahren mit aktuellen Parametern ohne Teil i der Daten durch.
3.	Klassifiziere Teil i von S_l.
4.	Falls $i < p$, dann $i = i + 1$ und gehe zu Schritt 2.
5.	Sind die Ergebnisse nicht zufriedenstellend, verändere die Parameterwerte, z.B. für σ und/oder C; i=1; gehe zu Schritt 2.
6.	Speichere die optimalen Parameterwerte, z.B: σ^* und C^*, STOP.

Tabelle 4: Algorithmus zur Parameterbestimmung mittels Kreuzvalidierung.

- Anzahl positiver und negativer Punkte:

$$\text{pos} := |\{i \in S_l : \ y_i = 1\}| \quad , \qquad \text{neg} := |\{i \in S_l : \ y_i = -1\}|$$

- Anzahl falscher Klassifikationen bei der Validierung:

$$\text{fk} := |\left\{i \in S_l : \ y_i \cdot h(\boldsymbol{x}^i) < 0\right\}|$$

- Anzahl richtig positiver und falsch positiver Klassifikationen:

$$\begin{aligned} \text{rp} &:= |\left\{i \in S_l : \ y_i = 1, \ h(\boldsymbol{x}^i) = 1\right\}| \, , \\ \text{fp} &:= |\left\{i \in S_l : \ y_i = -1, \ h(\boldsymbol{x}^i) = 1\right\}| \end{aligned}$$

- Anzahl richtig negativer und falsch negativer Klassifikationen:

$$\begin{aligned} \text{rn} &:= |\left\{i \in S_l : \ y_i = -1, \ h(\boldsymbol{x}^i) = -1\right\}| \, , \\ \text{fn} &:= |\left\{i \in S_l : \ y_i = 1, \ h(\boldsymbol{x}^i) = -1\right\}| \end{aligned}$$

- Genauigkeit (accuracy):
$$\mathrm{ac} = \frac{\mathrm{rp} + \mathrm{rn}}{\mathrm{pos} + \mathrm{neg}} \tag{98}$$
- Sensitivität (sensitivity):
$$\mathrm{se} = \frac{\mathrm{rp}}{\mathrm{pos}} \tag{99}$$
- Spezifität (specificity):
$$\mathrm{sp} = \frac{\mathrm{rn}}{\mathrm{neg}} \tag{100}$$
- Präzision (precision):
$$\mathrm{pr} = \frac{\mathrm{rp}}{\mathrm{rp} + \mathrm{fp}} \tag{101}$$

Eine der Kennzahlen (98) oder (101) könnte zur Parameteroptimierung herangezogen werden, was jedoch sehr einseitig wäre. Es gibt Einzelmaße, die aufgrund ihrer Struktur besser geeignet sind, um Ergebnisse zu bewerten. Dazu gehören die gewichtete Fehlersummierung mittels Kostenmatrix, der Anreicherungsfaktor und die sogenannten F-Maße. Ersteres gehört in die Kategorie der zu minimierenden Fehlermaße, alle anderen Funktionen zählen zu den Gütemaßen. Wir haben uns gegen die Arbeit mit Kostenmatrizen entschieden, denn diese stehen in den seltensten Fällen zur Verfügung.

Der Anreicherungsfaktor ef berechnet das Verhältnis zwischen dem Anteil tatsächlich positiver in den als positiv klassifizierten Testpunkten und dem Anteil positiver Punkte in den gesamtem Testdaten [19]. Er besagt demnach, um welchen Faktor die Wahrscheinlichkeit einen positiven Punkt zu ziehen ansteigt, wenn man anstelle des Gesamtdatenbestandes nur aus den Punkten mit Hypothese 1 auswählt. Formal ist dieser Faktor demnach definiert als

$$\mathrm{ef} := \frac{\mathrm{rp}}{\mathrm{rp} + \mathrm{fp}} \Big/ \frac{\mathrm{pos}}{l} = \frac{\mathrm{pr} \cdot l}{\mathrm{pos}} \ .$$

Es gilt $\mathrm{ef} \geq 0$ sowie

$$\mathrm{ef} \begin{cases} > 1 & \text{Anreicherung,} \\ = 1 & \text{kein Effekt der Anreicherung,} \\ < 1 & \text{Konzentrationsabnahme von Klasse } 1. \end{cases}$$

Der Anreicherungsfaktor ist nach oben beschränkt durch die Anzahl der Daten. Es gilt

$$\mathrm{ef} \leq \frac{l}{\mathrm{pos}},$$

wobei Gleichheit auftreten kann, wenn es keine falsch positiven Punkte gibt.

Der Anreicherungsfaktor kann Hinweise dazu geben, ob eine Klassifikationsfunktion geeignet ist, um positive, also interessante, Daten zu lokalisieren. Dennoch ist der Anreicherungsfaktor als Optimierungskennzahl zu einseitig, da er stark auf die Präzision abzielt. Zudem ist er für einen Performancevergleich auf verschiedenen Datensätzen nicht geeignet, da er nach oben beschränkt ist durch die Anzahl der Daten.

Eine andere Gruppe interessanter Performancekennzahlen stellen die sogenannten F-Maße (*f-measures*) dar. Ein F-Maß FM ist eine Abbildung von Präzision und Sensitivität auf einen einzelnen Wert. Es entspricht einer Gleichgewichtung von Präzision und Sensitivität (*harmonic mean*) zur besseren Bewertung von Klassifikationen auf möglicherweise unbalancierten Daten und wird im Allgemeinen definiert als [27]

$$\mathrm{FM} \;=\; 2\,\frac{\mathrm{pr}\cdot\mathrm{se}}{\mathrm{pr}+\mathrm{se}}\,. \tag{102}$$

Wegen $\mathrm{pr}\in[0,1]$ und $\mathrm{se}\in[0,1]$ gilt $\mathrm{FM}\in[0,1]$. Detaillierte Beschreibungen zum F-Maß und seinen Weiterentwicklungen innerhalb unserer Arbeit sind in [6, 7] zu finden. Das F-Maß wird im Anschluss an eine vollständige Kreuzvalidierung berechnet und ausgegeben.

In Tabelle 4 wird erwähnt, dass in jedem Optimierungsschritt die Parameterwerte neu zu wählen sind. Das bedeutet, eine iterative Suche wird durchgeführt. Unter einer iterativen Suche versteht man das einfache Durchprobieren einiger Parameterkombinationen, wobei nach jedem Durchlauf abhängig von der Performance entweder gestoppt oder wieder mit neu bestimmten Parameterwerten validiert wird. Diese Methode ist von den Ideen und der Flexibilität des Nutzers abhängig.

Wir haben zusätzlich zur iterativen Suche, die konkret nur aus mehrmaligem Aufruf der Kreuzvalidierung besteht, Standardmodelle und -parameter definiert sowie eine Rastersuche implementiert.

Die SVM-Software beinhaltet default-Werte für alle Modellparameter, um sicherzustellen, dass jeder Nutzer ein Ergebnis generieren kann. Wir verwenden als Standard das $L2$-Norm-Modell mittels NP-Algorithmus sowie den Gauß-Kern. Als Standardwerte sind

$$C=1 \qquad \text{und} \qquad \sigma=1$$

definiert. Sollten Slater- oder Polynomial-Kern gewählt werden, stehen ebenfalls Standardwerte zur Verfügung. Teilweise kann man mit diesen Voreinstellungen schon gute Ergebnisse erzielen.

Die Rastersuche, auch grid search oder Gittersuche genannt, stellt eine sehr einfache und weit verbreitete Methode der Parameteroptimierung dar. Dabei wird a priori für jeden Parameter eine endliche Anzahl von Werten angegeben, die getestet werden sollen. Es werden alle möglichen Parameterkombinationen verwendet. Diese Art der Parametersuche wird beispielsweise in [12] vorgeschlagen. Die Rastersuche wird von unserer Software zur Verfügung gestellt. Zu den Details sei auf Teil IV zur Softwaredokumentation hingewiesen.

Nach erfolgreich durchgeführter Parameteroptimierung verfügt man über alle benötigten Parameterwerte. Diese werden dann zusammen mit dem vollständigen Trainingsdatensatz dazu verwendet, ein einfaches Lernverfahren durchzuführen, um dann mit der erlernten Funktion einen Testdatensatz oder neue Daten zu klassifizieren. Stehen ausreichend viele Daten beider Klassen zur Verfügung, können, anstelle der Durchführung einer Kreuzvalidierung, auch einfach drei Datensätze - Trainings-, Validierungs-, und Testdatensatz - gebildet werden. Diese Variante steht ebenfalls zur Verfügung. Der Umweg über Validierungsdaten kann nur dann vollständig eingespart werden, wenn die Modellparameter bekannt oder fest vorgegeben sind.

Bei der Wahl der Parameter sei abschließend auf eine typische Fehlerquelle hingewiesen. Das in [32] auf Seite 128 beschriebene data snooping, bei dem die Anpassung der Modellparameter anhand des Testdatensatzes vollzogen wird, sollte unbedingt vermieden werden.

Literaturverzeichnis

[1] A. V. Anghelescu and I. B. Muchnik. *Optimization of SVM in a space of two parameters: weak margin and intercept - application in text classifier design.* Rutgers University, webpage, 2003.

[2] I. N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühlig. *Taschenbuch der Mathematik.* Verlag Harry Deutsch, 1999.

[3] O. Chapelle, V. N. Vapnik, O. Bousquet, and S. Mukherjee. *Choosing multiple parameters for support vector machines.* Machine Learning 2002, 46(1):131–159.

[4] T. M. Cover. *Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition.* IEEC 1965, 14:326–334.

[5] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods.* Cambridge University Press, Cambridge, UK, 2000.

[6] T. Eitrich and B. Lang. *Efficient optimization of support vector machine learning parameters for unbalanced datasets.* Journal of Computational and Applied Mathematics 2006, 196:425-436.

[7] T. Eitrich and B. Lang. *Parallel tuning of support vector machine learning parameters for large and unbalanced data sets.* In Proceedings of the First International Symposium on Computational Life, Science, Springer LNCS, Konstanz, Germany, September 2005
M. R. Berthold and R. Glen and K. Diederichs and O. Kohlbacher and I. Fischer (eds.), Springer Lecture Notes in Bioinformatics Vol. 3695, p. 253-264, 2005.

[8] A. Fischer. *Kontinuierliche Optimierung.* Unpublished script, TU Dresden, 2003.

[9] C. Großmann and J. Terno.4 *Numerik der Optimierung.* B. G. Teubner, Stuttgart, 1997.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2001.

[11] R. Herbrich. *Learning kernel classifiers: theory and algorithms.* MIT Press, Cambridge, MA, USA, 2001.

[12] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. *A practical guide to support vector classification.*
`http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`.

[13] C. Hsu and C. Lin. *A comparison of methods for multi-class support vector machines.* IEEE Transactions on Neural Networks 2002, 13:415–425.

[14] A. Jakulin, M. Mozina, J. Demsar, I. Bratko, and B. Zupan. *Nomograms for visualizing linear support vector machines.* Publications database, 2004.

[15] T. Joachims. *Text categorization with support vector machines: learning with many relevant features.* In Proceedings of ECML-98, 10th European Conference on Machine Learning, pages 137–142, Chemnitz, Germany, 1998. Springer Verlag.

[16] K. Jörgens. *Lineare Integraloperatoren.* B. G. Teubner, Stuttgart, 1970.

[17] S. S. Keerthi. *Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms.* IEEE Transactions on Neural Networks 2002, 13:1225–1229.

[18] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. *A fast iterative nearest point algorithm for support vector machine classifier design.* IEEE Transactions on Neural Networks 2000, 11(1):124–136.

[19] A. Kless and T. Eitrich. *Cytochrome p450 classification of drugs with support vector machines implementing the nearest point algorithm.* In J. A. López, E. Benfenati, and W. Dubitzky, editors, Knowledge Exploration in Life Science Informatics, International Symposium,KELSI 2004, Milan, Italy, November 25-26, 2004, Proceedings, volume 3303 of Lecture Notes in Computer Science, pages 191–205. Springer, 2004.

[20] J. E. Laird and P. Rosenbloom. *The evolution of the soar cognitive architecture.* In D. M. Steier and T. M. Mitchell, editors, Mind Matters: A Tribute to Allen Newell, pages 1–50. Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, 1996.

[21] D. G. Luenberger. *Optimization by vector space methods.* John Wiley & Sons, New York, 1969.

[22] F. Markowetz. *Support vector machines in bioinformatics.* Master's thesis, University of Heidelberg, 2001.

[23] J. Mercer. *Functions of positive and negative type and their connection with the theory of integral equations.* Philosophical Transactions of the Royal Society 1909, London A, 209:415–446.

[24] B. Mitchell, V. Demýanov, and V. Malozemov. *Finding the point of a polyhedron closest to the origin.* SIAM Journal on Control and Optimization 1974, 12(1):19–26.

[25] M. Momma and K. P. Bennett. *A pattern search method for model selection of support vector regression.* In R. L. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, editors, Proceedings of the Second SIAM International Conference on Data Mining, Arlington, VA, USA, April 11-13, 2002. SIAM, 2002.

[26] K. Morik, P. Brockhausen, and T. Joachims. *Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring.* In Proc. 16th International Conf. on Machine Learning, pages 268–277. Morgan Kaufmann, San Francisco, CA, 1999.

[27] D. R. Musicant, V. Kumar, and A. Ozgur. *Optimizing F-measure with support vector machines.* In I. Russell and S. M. Haller, editors, Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference, May 12-14, 2003, St. Augustine, Florida, USA, pages 356–360. AAAI Press, 2003.

[28] E. Osuna, R. Freund, and F. Girosi. *Support vector machines: training and applications.* AI Memo 1602, Massachusetts Institute of Technology, 1997.

[29] J. Platt. *Fast training of support vector machines using sequential minimal optimization.* In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning, pages 185–208, Cambridge, MA, 1999. MIT Press.

[30] J. R. Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann Publishers Inc., 1993.

[31] S. J. Russell and P. Norvig. *Artificial intelligence: a modern approach.* Prentice-Hall International, 1995.

[32] B. Schölkopf and A. J. Smola. *Learning with kernels.* MIT Press, Cambridge, MA, 2002.

[33] V. N. Vapnik. *The nature of statistical learning theory.* Springer, New York, 1995.

[34] V. N. Vapnik. *Statistical learning theory.* John Wiley & Sons, New York, 1998.

Datenanalyse

Claudia Druska, Tatjana Eitrich, Wolfgang Meyer

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: {*c.druska, t.eitrich, w.meyer*}*@fz-juelich.de*

In diesem Teil stellen wir die konkreten Möglichkeiten der Datenanalyse vor, die sich aus den Implementierungen der in den bisherigen Teilen beschriebenen Methoden und weiteren Ergänzungen ergeben. Das innerhalb des Projektes entstandene Softwarepaket ermöglicht eine umfassende Säuberung und Analyse von pharmazeutischen Daten.

In Abbildung 1 ist der grobe Ablauf der Data Mining Pipeline dargestellt. Im Folgenden werden die einzelnen Schritte genau beschrieben. Sie enthalten eine Vielzahl von wichtigen Data Mining Methoden, die in ihrer Gesamtheit zu verlässlichen Informationen über Auffälligkeiten und Strukturen in Daten führen.

1 Datensatz

Grundlage der Data Mining Pipeline ist der zur Verfügung stehende Datensatz. Als einen Datensatz bezeichnen wir eine Menge von Rohdaten mit der folgenden Struktur:

- eine Matrix, deren Zeilen aus den jeweiligen Substanzen bestehen und deren Spalten die zu untersuchenden Attribute sind;
- eine Liste, welche die Namen der Substanzen enthält; sowie
- eine Liste, welche die Namen der Attribute beinhaltet.

Optional kann für die Erstellung von Klassifikatoren zusätzlich ein Klassenvektor bereitgestellt werden. Weitere Details dazu folgen im Abschnitt 4.2.

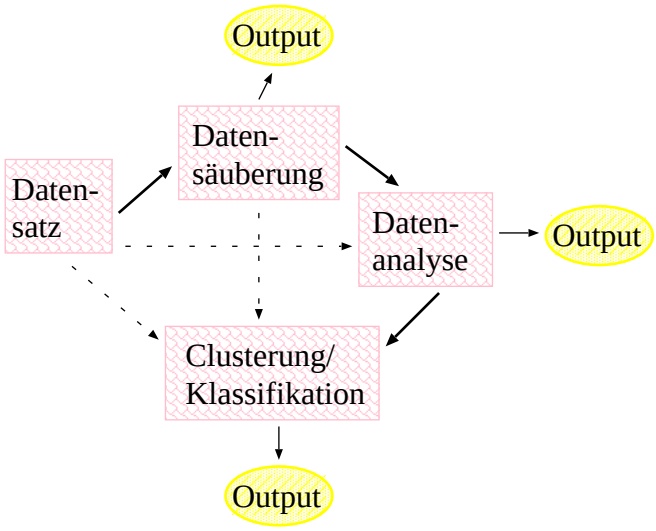


Abbildung 1: Data Mining Pipeline im GALA-Projekt.

2 Datensäuberung

In einem ersten Schritt nach dem Einlesen der Rohdaten und der zugehörigen Namenslisten für Substanzen und Attribute erfolgt eine Überprüfung der Datenmatrix auf Nullspalten. Da in der Regel große Datenmengen (bis zu 100000 Substanzen und 1000 Attribute) aus dem Data Warehouse zur weiteren Analyse gefiltert werden, kann es vorkommen, dass sich Nullspalten in die Rohdatenmatrix “einschleichen”. Diese werden vor Beginn der eigentlichen Datenaufbereitung entfernt. Die entsprechende Programmroutine erzeugt gegebenenfalls neue Eingabedaten, d.h. es wird eine um die Nullspalten bereinigte Rohdatenmatrix und die ebenfalls angepasste Liste der Attributnamen zur Verfügung gestellt. Im Anschluss daran können nun erste inhaltliche Datensäuberungen angestoßen werden.

2.1 Missing Value Behandlung

Neben der Entfernung von Nullspalten ist zu Beginn der Untersuchungen auch zu entscheiden wie Missing Values, d.h. fehlende oder offensichtlich falsche Einträge in der Rohdatenmatrix, zu behandeln sind. Es wurden zwei mögliche Behandlungsarten implementiert. Zum einen können unvollständige Beobachtungen, das heißt Beobachtungen, die mindestens einen fehlenden Wert aufweisen, unberücksichtigt bleiben. Sie werden aus dem Datensatz gestrichen. Der aus den vollständigen Beobachtungen bestehende verkleinerte Datensatz ist dann Basis der weiteren Betrachtungen. Eine andere Methode besteht darin, fehlende Beobachtungswerte durch die arithmetischen Mittelwerte der betroffenen Variablen zu ersetzen. Dadurch vermeidet man, auf wichtige Beobachtungen frühzeitig wegen des Fehlens einzelner Werte zu verzichten.

2.2 Ausreißerbestimmung

Die in den Rohdaten enthaltenen Informationen können durch extreme Werte beeinflusst sein. Dabei muss man zwischen fehlerbehafteten Daten und Daten mit überdurchschnittlichen Eigenschaften unterscheiden. Es wird empfohlen, erstere vor weiteren Analysen zu eliminieren, um spätere Ergebnisse nicht zu verfälschen. Ausreißer der zweiten Kategorie sollten identifiziert und kenntlich gemacht werden. Da diese Unterscheidung in der Regel pharmakologisches Fachwissen voraussetzt, identifiziert das Programm zunächst die Daten mit stark abweichenden Werten und der Anwender entscheidet über die weitere Behandlung.
Für eine Variable sind diejenigen Beobachtungen Ausreißer, deren standardisierte Werte betragsmäßig größer als ein kritischer Wert sind, für ein Variablenpaar diejenigen Beobachtungen, deren Mahalanobisdistanzen (3) (Teil II, S. 20) diese Bedingung erfüllen. Der kritische Wert wird entweder fest vorgegeben (Abschnitt 3.3.1, Teil II) oder durch Vorgabe des Niveaus des Diskordanztests für einen Ausreißer bei eindimensionaler (Abschnitt 6.4.1, Teil II) bzw. zweidimensionaler (Abschnitt 3.3.3, Teil II) Normalverteilung festgelegt. Für zweidimensionale Ausreißer besteht außerdem die Möglichkeit, den kritischen Wert so zu wählen, dass im Falle normalverteilter Daten der Anteil der als Ausreißer deklarierten Beobachtungen etwa $\sqrt{1/10n}$ beträgt, wobwi n die Anzahl der vollständigen Wertepaare ist (Abschnitt 3.3.2, Teil II).

2.3 Transformation

Ein Ziel von Transformationen ist es, nichtlineare Abhängigkeiten auf lineare Abhängigkeiten abzubilden. So können beispielsweise Volumendaten durch die Transformation mit dritter Wurzel linearisiert werden, um anschließend mit Hilfe der Korrelationsrechnung Abhängigkeiten aufdecken zu können.
Diese Programmoption transformiert die Eingabevariablen gemäß der Vorgaben durch eine benutzerdefinierte Eingabedatei. Enthält diese Datei das Paar $\{n \in \mathbb{N}, k \in \mathbb{N}_{<8}\}$, wird die n-te Variable wie folgt transformiert:

Wert des Parameters k	Transformationsregel
0	keine Transformation
1	Quadratwurzel, Missing Value für negative Werte
2	Quadrat
3	3. Wurzel
4	3. Potenz
5	Reziproker Wert, Missing Value für 0.0
6	Natürlicher Logarithmus, Missing Value für nichtpositive Werte
7	Exponentialfunktion

3 Datenanalyse

Mit den bisher beschriebenen Methoden ist es möglich, die Rohdaten aus dem Data Warehouse zu säubern und erste Informationen zur Datenstruktur zu sammeln. In diesem Abschnitt wird nun auf die eigentliche Datenanalyse eingegangen.

3.1 Korrelation

Die Programmoption zur Korrelationsberechnung ermöglicht die Untersuchung der Rohdaten oder transformierten Daten auf (lineare) Abhängigkeiten zwischen den Messreihen. Dabei stehen drei Arten der Korrelationsberechnung zur Verfügung. Man kann zum einen zwischen den klassischen Verfahren nach Pearson (Abschnitt 2, Teil II) oder Spearman (Abschnitt 5, Teil II) wählen oder man wendet eine robuste Methode (Abschnitt 8.6, Teil II) an.
Die Pearsonsche Produktmomentkorrelation aller Variablenpaare wird unter Verwendung aller vollständigen Wertepaare berechnet. Dabei werden die Mittelwerte und Standardabweichungen der Variablen berechnet. Wird das Verfahren nach Spearman ausgewählt, erfolgt eine Rangtransformation für alle Variablen, d.h. die Beobachtungswerte einer Variable werden durch die Ränge ersetzt, und anschließend wird die Pearsonsche Produktmomentkorrelation aller Variablenpaare unter Verwendung der vollständigen Wertepaare berechnet. Diese Vorgehensweise ermöglicht das Aufdecken monotoner Abhängigkeiten, die durch lineare Verfahren auf Originaldaten nicht entdeckt werden. Außerdem können auf Grundlage der Ergebnisse der Korrelationsberechnung Ausreißer identifiziert und eliminiert werden. Die Identifikation wird wie in Abschnitt 2.2 beschrieben durchgeführt. Über einen Parameter kann man die Anzahl der Korrelationsberechnungen steuern. Führt man mehr als eine Korrelationsberechnung durch, werden bei den aufeinanderfolgenden Rechnungen die zuvor entdeckten Ausreißer nicht mehr berücksichtigt. Auf diese Art und Weise können die Rohdaten iterativ von Ausreißern gesäubert werden. Dabei sollte die Anzahl der Läufe jedoch nicht zu groß sein, da sonst ein signifikanter Teil der Rohdaten entfernt werden müsste. Die Anzahl der Iterationen zur Ausreißereliminierung ist daher mit 2 zunächst klein voreingestellt, kann jedoch über eine Option gesteuert werden.

3.2 PCA

Diese Programmoption ermöglicht die Durchführung einer Hauptkomponentenanalyse. Dabei kann der Anwender zwischen folgenden drei Verfahren wählen:

(1) Hauptkomponentenanalyse einer Pearsonschen Produktmomentkorrelationsmatrix,

(2) Hauptkomponentenanalyse einer Rangkorrelationsmatrix und

(3) robuste Hauptkomponentenanalyse nach Campbell (iterative Bestimmung robuster Eigenwerte und Eigenvektoren, s. Abschnitt 8.7, Teil II).

Zusätzlich kann bestimmt werden, welche Scores der Hauptkomponenten berechnet und ausgegeben werden. Entweder werden die Scores der Hauptkomponenten zu den k ($k = 1, 2, \ldots$) größten Eigenwerten oder die Scores der Hauptkomponenten, deren zugehörige Eigenwerte größer als $1 + \frac{k}{10}$ ($k = 0, -1, \ldots, -10$) sind, berechnet. Da das Programm die Hauptkomponenten der Korrelationsmatrix bestimmt, werden die Scores mittels standardisierter Daten berechnet. Zur Berechnung der Scores der Hauptkomponenten einer Rangkorrelationsmatrix werden nicht die Rohdaten, sondern stets rangtransformierte Daten verwendet. Außerdem werden die Korrelationen zwischen den Variablen und den Hauptkomponenten berechnet und ausgegeben. Darüber hinaus wird für jede Variable der Anteil ihrer Varianz berechnet, der durch die ersten k ($k = 1, ...,$ Anzahl der der PCA zugrundeliegenden Variablen) Hauptkomponenten nicht erklärt wird.

3.2.1 Ausreißerbestimmung mittels Hauptkomponenten

Die Ergebnisse der Hauptkomponentenanalyse können ebenfalls der Ausreißersuche dienen. Wir unterscheiden dabei drei Szenarien.

Univariate Ausreißer
Bezüglich einer Hauptkomponente sind diejenigen Beobachtungen Ausreißer, deren standardisierte Scores für die betrachtete Hauptkomponente dem Betrag nach größer als ein kritischer Wert sind. Bestimmt werden die Ausreißer für die k Hauptkomponenten mit den größten zugehörigen Eigenwerten. Für $k = 0$ wird keine Ausreißerbehandlung durchgeführt.

Multivariate Ausreißer bzgl. der k Hauptkomponenten mit den größten Eigenwerten
Ausreißer sind die Beobachtungen, deren quadrierte Mahalanobis-Distanzen bzgl. der k Hauptkomponenten mit den größten Eigenwerten größer als ein kritischer Wert sind.

Multivariate Ausreißer bezüglich der (Anzahl der Variablen - k) Hauptkomponenten mit den kleinsten Eigenwerten
Ausreißer sind die Beobachtungen, für die die Quadratsumme der Scores der letzten (Anzahl der Variablen - k) Hauptkomponenten größer als ein kritischer Wert ist.

Der kritische Wert wird entweder fest vorgegeben oder durch Vorgabe des Niveaus des Diskordanztests für einen Ausreißer bei ein- bzw. mehrdimensionaler Normalverteilung festgelegt (Abschnitt 6.4, Teil II).

3.3 Variablenselektion

Neben der Hauptkomponentenanalyse bietet die Variablenselektion eine weitere Möglichkeit zur Dimensionsreduzierung an (Teil II, Abschnitt 7). Bei der Variablenselektion wird

eine fest vorgegebene oder eine durch Optimierungsmethoden berechnete Anzahl von Variablen für die weiteren Analyseverfahren ausgesucht. Die restlichen Variablen verlassen die Data Mining Pipeline. Die Variablenselektion stellt eine wichtige Schnittstelle zwischen den Datensäuberungs- bzw. -analysemethoden auf der einen und den Cluster- bzw. Klassifikationsverfahren auf der anderen Seite dar. Aus diesem Grund sind zwei grundsätzlich unterschiedliche Vorgehensweisen implementiert worden:

- unüberwachte Variablenselektionen ohne Verwendung eines Klassenvektors, sowie
- überwachte Variablenselektionen mit expliziter Auswertung des Klassenvektors.

Keines der Verfahren kann favorisiert werden, da die erzielten Ergebnisse stark daten- und parameterabhängig sind. Über Vor- und Nachteile muss individuell entschieden werden [5].

3.3.1 Unüberwachte Variablenselektion

Die Durchführung der Variablenselektion basiert auf einer Korrelationsmatrix, die man wie in Abschnitt 3.1 beschrieben näher spezifizieren kann. Innerhalb der Variablenselektion kann man zwischen fünf Verfahren wählen. Zur Auswahl steht zum einen die Methode der Principal Variables nach McCabe [7], bei der iterativ jeweils die Variable ausgewählt wird, welche die Summe der Residualvarianzen minimiert. Darüber hinaus wurden zwei auf einer PCA basierenden Selektionsverfahren implementiert. Bei der Selektionsvariante werden iterativ Variablen ausgewählt, die zusammen einen möglichst großen Anteil der Streuung aller Variablen erklären, während bei der Eliminationsvariante iterativ Variablen entfernt werden, deren Streuung von den verbleibenden Variablen möglichst gut erklärt wird. In beiden Verfahren wird iterativ jeder Hauptkomponente die Variable mit dem größten Eintrag in dem zugehörigen Eigenvektor (Faktorladung) unter den noch freien Variablen zugeordnet. Bei der Selektionsvariante werden die Hauptkomponenten in der Reihenfolge fallender zugehöriger Eigenwerte, in der Eliminationsvariante in der Reihenfolge wachsender zugehöriger Eigenwerte abgearbeitet. Das vierte und fünfte Verfahren unterscheidet sich in der Anzahl der durchzuführenden Hauptkomponentenanalysen. Hier wird in jedem Iterationsschritt eine neue PCA durchgeführt. Bei der Selektionsvariante erfolgt eine PCA der partiellen Korrelationsmatrix der verbliebenen Variablen bezüglich der ausgewählten Variablen. Bei der Eliminationsvariante handelt es sich um ein Verfahren, bei dem die PCA der Korrelationsmatrix der verbliebenen Variablen verwendet wird.

3.3.2 Überwachte Variablenselektion

Das implementierte Verfahren zur überwachten Variablenselektion ordnet alle Attribute in eine Rangliste, wobei diejenigen Attribute einen hohen Rang zugeordnet bekommen, die am besten dazu geeignet sind, die zugrundeliegende Klassenstruktur der Trainingsdaten zu

erklären. Das zugehörige Verfahren arbeitet auf den eindimensionalen Verteilungen der einzelnen Variablen. Für jede Variable k kann man die Menge der Realisierungen in zwei Gruppen einteilen: positive und negative Realisierungen. Die beiden daraus entstehenden eindimensionalen Verteilungen werden untersucht. Weichen diese in Lage und Struktur stark voneinander ab, wird ein funktionaler Zusammenhang zwischen der Variable k und der Klassenstruktur angenommen. Als Grundlage der Tests dient der Chi-Quadrat ($\mathcal{X}^2$) Homogenitätstest [6]. Der $\mathcal{X}^2$-Test ist geeignet für diskrete Daten, d.h. es wird eine Größe X mit c Kategorien und g Gruppen betrachtet ($c,\ g \in \mathbb{N}$). Die Implementierung arbeitet mit $g = 2$, da ausschließlich binäre Probleme vorliegen. Um diesen diskreten Test anwenden zu können, werden alle auf Varianz 1 skalierten Werte in Kategorien eingeordnet. Dazu wird mit einer vordefinierten Menge an Intervallen gestartet. Im Anschluss wird die Anzahl von Vertretern in jeder Kategorie geprüft und falls weniger als 5 Vertreter vorhanden sind, wird die jeweilige Kategorie mit einem Nachbarn zu einer gemeinsamen neuen Kategorie verbunden. Für jede Variable wird gesondert entschieden, ob diese zur Klasseneinteilung beitragen kann oder nicht.

Das beschriebene Verfahren zur überwachten Variablenselektion allein garantiert noch nicht eine optimale Auswahl von Attributen. Zwischen den Variablen mit hohen Rängen können große Korrelationen auftreten. Diese sind unerwünscht und sollten eliminiert werden. Aus diesem Grund schließt sich an das Ordnen der Variablen ein hierarchisches Clusterverfahren an [3]. Dazu wird die gut bekannte complete-link Methode eingesetzt. Diese definiert den Abstand zwischen zwei Clustern $\mathcal{C}_0$ and $\mathcal{C}_1$ als Maximum aller paarweisen Abstände zwischen Punkten in $\mathcal{C}_0$ und $\mathcal{C}_1$ [3]. Die Rangkorrelationen zwischen Variablenpaaren werden verwendet, um einen Baum aufzubauen, der es dann ermöglicht, hochkorrelierte Teilbäume zu erkennen und durch eine Variable mit hohem Rang zu ersetzen.

Resultat dieser zweistufigen Vorgehensweise ist eine Menge von Variablen, welche die Klassenzugehörigkeiten in den Daten erklärt, jedoch frei von redundanten Informationen ist.

Die beiden Methoden des überwachten und unüberwachten Selektierens von Variablen können auch hintereinandergeschaltet werden.

4 Clusteranalyse und Klassifikationsverfahren

Im Anschluss an die Schritte der Datensäuberung und -analyse können Methoden des überwachten und des unüberwachten Lernens eingesetzt werden. Diese Methoden ordnen Datenpunkte in Gruppen ein, wobei diese Gruppen entweder schon vorgegeben sind oder noch definiert werden müssen.

4.1 Clusteranalyse

Für das unüberwachte Lernen von Clustern stehen zwei übernommene Verfahren zur Verfügung:

- AutoClass [1] ist eine frei verfügbare Software zur unüberwachten Bayes-Klassifikation, der sogenannten Bayes-Clusterung. Ein sehr wichtiger Aspekt dieser Software ist, dass die Anzahl der Cluster automatisch bestimmt wird. Das ermöglicht dem Benutzer, weitere Trends in den im Allgemeinen nur in zwei Gruppen gegliederten Daten zu finden. Zusätzlich zur klassischen Clusteranalyse kann auch eine echte Klassifikation stattfinden. Dabei werden noch unbekannte Testfälle in die vorher generierten Cluster eingeordnet, ohne dass diese für die Modellierung eingesetzt worden sind. AutoClass beschreibt die Daten als Mischung unabhängiger Klassen, wobei jede Klasse über eine Wahrscheinlichkeitsverteilung über dem Raum, den die Variablen aufspannen, definiert ist. Gauß'sche Verteilungen für die stetigen Variablen und Bernoulli-Verteilungen für die diskreten Variablen werden angenommen. Das Verfahren bestimmt eine Menge von Klassen, die am wahrscheinlichsten zu den Daten und den weiteren Annahmen passen.
- Subset ist ein Clustering-Programm, welches sowohl zur Datenreduktion als auch zur Clusteranalyse verwendet werden kann. Subset arbeitet in der Originalversion ausschließlich mit binären Variablen. Aus diesem Grund steht nur der Tanimoto-Koeffizient als Abstandsmaß zur Verfügung. Um den Algorithmus auch für die Analyse stetiger Variablen verwenden zu können, wurde das Programm um ein auf dem Euklidischen Abstand basierendes Abstandsmaß erweitert und steht somit zur Reduktion und Clusterung von Daten mit stetigen sowie diskreten Variablen zur Verfügung. Die Grundlage der Clusteranalyse bildet eine in [9] beschriebene stochastische Clustermethode. Die Datenreduktionsfunktion innerhalb von Subset ist insbesondere für große Datensätze von Interesse, da das Programm eine repräsentative Teilmenge der Daten so bestimmen kann, dass aus jedem vorhandenen Cluster eine gleich große Anzahl von Substanzen erhalten bleibt. Vor allem für sehr unausgeglichende Daten bietet das die Möglichkeit, eine sehr stark aufgeblähte Gruppe so zu verkleinern, dass keine wichtigen Informationen verloren gehen. In diesem Sinne ist diese Methode insbesondere als Vorstufe zu echten Klassifikationsverfahren, die sehr empfindlich auf unausgeglichene Klassen reagieren, zu sehen.

Die vorhandenen Cluster-Methoden werden innerhalb des Projektes für folgende Fragenstellungen verwendet:

- für ungelabelte Daten
  - Clusteranalyse (klassisch) und
  - Ziehen einer repräsentativen Stichprobe aus einem großen Datensatz,
- für gelabelte Daten

– Clusteranalyse für zwei Gruppen auf einem Trainingsdatensatz zur Überprüfung der gegebenen Klassifikationen,

– Clusteranalyse auf einer Klasse des Trainingsdatensatzes, um zu prüfen, ob in dieser Klasse große strukturelle Unterschiede zwischen den Punkten vorhanden sind, und

– Verkleinerung von ganzen Datensätzen beziehungsweise von einzelnen stark aufgeblähten Klassen, um robuste Klassifikatoren erstellen zu können.

4.2 Klassifikationsverfahren

Klassifikation ist eine der wichtigsten Fragestellungen innerhalb des GALA-Projektes. Basierend auf gesäuberten und in beiden Dimensionen reduzierten Daten ist es die Aufgabe eines Klassifikationsverfahrens, eine Funktionalität zwischen Eingaben und gewünschten Ausgaben in einem Datensatz zu erkennen und mittels der zur Verfügung stehenden Funktionen zu formulieren. Diese Funktionalität kann dann für weitere Daten zur Klassenvorhersage verwendet werden. Im Teil III dieses Bandes ist dieses überwachte Lernen ausführlich beschrieben worden.

Bei dieser Form des Data Minings gibt es immer zwei Arten von Daten:

- die Trainingspunkte als Grundlage für die Modellierung, sowie
- die Punkte, die klassifiziert werden sollen.

Zusätzlich zu einer Datenmatrix ist es notwendig, jedem Trainingspunkt, der in das Modell einfließen soll, ein Klassenlabel zuzuordnen. Im Falle binärer Klassifikation, die am häufigsten als Problemstellung auftritt, bedeutet das, für jede Substanz i des Trainingsdatensatzes einen Klassenwert $y_i \in \{-1, 1\}$ anzugeben.

Die Erstellung von Klassifikatoren erfolgt innerhalb des Projektes hauptsächlich mit dem modernen Verfahren der Support-Vektor-Maschinen, welches im Teil III dieses Bandes detailliert vorgestellt wurde und zu den mächtigsten nichtlinearen Klassifikationsverfahren der heutigen Zeit zählt.

Die Anwendung einer Support-Vektor-Maschine gliedert sich in zwei hintereinanderfolgende Phasen:

- Zu Beginn wird basierend auf den Informationen des Trainingsdatensatzes eine nichtlineare Klassifikationsfunktion erlernt und gespeichert.
- Im Anschluss kann diese Funktion verwendet werden, um neue Daten zu labeln beziehungsweise um schon bekannte Labels zu überprüfen.

Mehrklassige Modelle können einfach über ein mehrmaliges binäres Lernen mit einem der Ansätze aus [2] generiert werden.

4.2.1 Trainingsphase

Das Erlernen einer Klassifikationsfunktion, das sogenannte Training, bedeutet im Wesentlichen, dass ein großes quadratisches Optimierungsproblem zu lösen ist. Dazu sei auf Abschnitt 3.5 in Teil III verwiesen. Es stehen dabei zwei verschiedene Modelle zur Verfügung:

- Das Modell der 1-Norm summiert schwache Klassifikationen und Fehler während des Trainings einfach auf. Siehe dazu Aufgabe (74) auf Seite 149.
- Das Modell der 2-Norm summiert schwache Klassifikationen und Fehler in quadrierter Form auf. Das bedeutet, Punkte, die in der Nähe der gewünschten Klassifikationsgüte liegen (Fehler im Intervall $[0, 1]$), werden etwas mehr toleriert, wohingegen echte Fehler im Intervall $(1, \infty)$ hart bestraft werden. Siehe dazu Aufgabe (76) auf Seite 150.

Die Algorithmen zur Umsetzung dieser Modelle sind in den Abschnitten 4.1.5 und 4.2 in Teil III vorgestellt worden. Der SMO-Algorithmus von J. Platt [8] implementiert das 1-Norm Verfahren, während der NP-Algorithmus von S. Keerthi et. al [4] die 2-Norm Methode umsetzt.

Für die Nutzung einer nichtlinearen Support-Vektor-Maschine ist es notwendig eine Kernfunktion zu definieren. Für den Lernvorgang stehen vier vordefinierte Kerne zur Verfügung:

- der Gauß-Kern (siehe (33) auf Seite 127),
- der Slater-Kern (siehe (35) auf Seite 128),
- der Polynomial-Kern (siehe (34) auf Seite 127), sowie
- der Tanimoto-Kern (siehe (36) auf Seite 128).

Zusätzlich kann auch ein selbst definierter Kern verwendet werden. Dafür muss zwar der Quellcode geändert werden, diese Änderung betrifft jedoch nur eine einzige Routine, die Funktion zur Berechnung des Kerns. Diese ist losgelöst von den Algorithmen implementiert worden. Eine Änderung wird damit sowohl für den SMO- als auch für den NP-Algorithmus wirksam.

4.2.2 Optimierungsstufen

Ein einzelnes SVM-Training generiert eine Hypothese, die für die verwendeten Parameter optimal ist. Die Schwierigkeit bei der Anwendung von Support-Vektor-Maschinen für komplizierte Datensätze ist die konkrete Wahl der Parameterwerte. Diese beeinflussen die Güte der Klassifikationsfunktion, sind aber nicht direkt berechenbar. Vielmehr müssen sie während einer Optimierungsphase erlernt werden.

Die hier betrachteten Parameter sind

- Kern-Parameter, zum Beispiel $\sigma > 0$ für den Gauß-Kern,

- Strafparameter C^+ für positive Trainingspunkte, und
- Strafparameter C^- für negative Trainingspunkte.

Zur Verfügung steht eine automatisierte Parameteroptimierung, die nach folgendem Schema abläuft:

(1) Für jeden zu optimierenden Parameter wird eine Datei mit gewünschten Werten zur Verfügung gestellt. Feste Parameterwerte können ebenfalls vorgegeben werden, falls sie schon bekannt sind. Das verkürzt die Optimierungszeit.

(2) Für alle möglichen Parameterkombinationen werden die Schritte (3) bis (5) durchgeführt.

(3) Zehn- oder l-fache Kreuzvalidierung mit den aktuellen Parameterwerten.

(4) Basierend auf den internen Testergebnissen, Berechnung des Gütemaßes.

(5) Aktualisierung des bisher besten Gütemaßes und Speicherung des zugehörigen Parametertupels.

(6) Im Anschluss wird das optimale Parametertupel für das reguläre SVM-Training auf dem kompletten Datensatz verwendet.

Zu den Details der Kreuzvalidierung verweisen wir auf Abschnitt 5.2 im Teil III. Das verwendete Gütemaß ist insbesondere auf Sensitivität ausgelegt. Es wurde bereits vorgestellt, siehe (102) auf Seite 174.

4.2.3 Recall

Zusätzlich zu den Kreuzvalidierungstests ist die Funktionalität des Recalls in die Software eingebettet worden. Unter Recall wird ein unabhängiger Test auf Teilen der Trainingsdaten verstanden. Dieser Test findet im Anschluss an das reguläre Training statt. Damit dieser Test unabhängig von den zur Parameteroptimierung und zum finalen Training verwendeten Daten ist, wird bei der Wahl der Recalloption die gewünschte Anzahl von Substanzen zurückgehalten und nicht zum Training freigegeben. In Anschluss an das Training werden diese Substanzen klassifiziert und der Benutzer erhält eine Information zu den dabei entstandenen Fehlklassifikationen in beiden Klassen.

4.2.4 Test und Klassifikation

Zusätzlich zu den Trainingsdaten, auf denen Parameteroptimierung, Validierung, Training und Recall durchgeführt werden können, kann eine weitere Klasse von Punkten bearbeitet werden. Je nach Bedarf werden die Punkte entweder als unbekannte Substanzen klassifiziert oder es findet eine Klassifikation gekoppelt an eine Auswertung statt. Letzterer Fall

setzt voraus, dass die Labels der Punkte bekannt sind. Dieser Test erscheint zunächst redundant, da die gleiche Support-Vektor-Maschine eingesetzt wird, welche auch schon auf den Recall-Punkten gearbeitet hat. Die Test-Option ist jedoch immer dann wichtig, wenn ganz bestimmte Substanzen getestet werden sollen. Bei der Recall-Option wird diese Möglichkeit nicht gegeben, da dort immer auf einer zufällig gewählten Datenmenge getestet wird, um die Verteilung der Klassen in etwa widerzuspiegeln. Durch die Test-Option gewinnt der Benutzer deutlich an Flexibilität. Die Auswertung erfolgt auch hier getrennt nach Klassen.

Literaturverzeichnis

[1] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. *AutoClass: a Bayesian classification system.* Proceedings of the Fifth International Conference on Machine Learning, Morgan Kaufmann Publishers, San Francisco, 1988, pages 54–64.

[2] C. Hsu and C. Lin. *A comparison of methods for multi-class support vector machines.* IEEE Transactions on Neural Networks 2002, 13:415–425.

[3] A. Jain, M. Murty, P. Flynn, P. *Data clustering: a review.* ACM Computing Surveys 1999, 31(3) 264–323.

[4] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. *A fast iterative nearest point algorithm for support vector machine classifier design.* IEEE Transactions on Neural Networks 2000, 11(1):124–136.

[5] A. Kless and T. Eitrich. *Cytochrome P450 classification of drugs with support vector machines implementing the nearest point algorithm.* In J. A. López, E. Benfenati, and W. Dubitzky, editors, Knowledge Exploration in Life Science Informatics, International Symposium, KELSI 2004, Milan, Italy, November 25-26, 2004, Proceedings, volume 3303 of Lecture Notes in Computer Science, pages 191–205, Springer, 2004.

[6] S. Kotz, N. L. Johnson. *Encyclopedia of Statistical Sciences (volume 3).* John Wiley & Sons, 1983.

[7] G. P. McCabe. *Principal variables.* Technometrics 1984, 26:137–144.

[8] J. Platt. *Fast training of support vector machines using sequential minimal optimization.* In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning, pages 185–208, Cambridge, MA, MIT Press, 1999.

[9] C. H. Reynolds, R. Druker, L. B. Pfahler. *Lead discovery using stochastic cluster analysis (SCA): a new method for clustering structurally similar compounds.* J. Chem. Inf. Comput. Sci. 1998, 38:305–312.

Anwendungsbeispiel: Cytochrom P450 Profilierung von neuen Wirkstoffen

Achim Kless

Biomedizinische Forschung, Drug Discovery
Grünenthal GmbH
52078 Aachen
E-mail: Achim.Kless@grunenthal.com

1 Einleitung

Der Prozess der Identifikation von neuen potentiellen Wirkstoffkandidaten hängt nicht nur von der Auswahl der potentesten Substanz für ein spezifisches Target ab. Durch unsere Erfahrungen in der Vergangenheit wissen wir, dass der Metabolismus und die Pharmakokinetik für eine Substanzauswahl gleichberechtigt berücksichtigt werden müssen. Cytochrom P450 Enzyme sind metabolisierende Enzyme, die in verschiedenen Geweben und Blut vorkommen und hauptsächlich in der Leber eine aktive Rolle im oxidativen Abbau der Substanzen spielen. Insbesondere sind die Isoformen 1A2, 2C19, 2C9, 2D6 und 3A4 wichtig, da diese am Metabolismus von 90% der Substanzen beteiligt sind [10, 11]. Wie von Flockhart [12] näher beschrieben, blocken Inhibitoren die Funktion dieser Enzyme, so dass die verbleibende Aktivität reduziert ist. In unserem Beispiel haben wir uns auf die 2D6-Isoform konzentriert, da bei dieser Isoform einige inaktive polymorphe Formen in der Bevölkerung existieren, was analog zu einer 2D6-Inhibition ist. Dies kann dann bei Wirkstoffen, die ein 2D6 Substrat sind, zu toxisch hohen Plasmaspiegeln führen [5]. CYP2D6 ist ein Enzym, das primär durch Substanzen mit einem Stickstoffatom und einem hydrophoben Part blockiert wird, was durch Pharmakophoranalysen gezeigt werden konnte. Wenn dieses Enzym blockiert wird, kann es auch mit einer zweiten Substanz zu unerwünschten Wirkstoff-Wechselwirkungen kommen, die für eine Anwendung am Menschen unerwünscht und bei der Selektion von neuen Wirkstoffkandidaten zu vermeiden

sind. Es ist deshalb wünschenswert, für eine neue chemische Struktur die Aktivität an diesem Enzym vorhersagen zu können, da hochqualitative Daten unter in vivo Bedingungen nur zu einem sehr späten Zeitpunkt in der präklinischen Forschung erhoben werden. Aus diesem Grund ist die in silico Vorhersage zu einer Schlüsseltechnologie in den frühen Phasen der Forschung geworden [4, 6]. Unsere Arbeiten sollen zeigen, dass unsere Methoden zur Erstellung von Klassifikatoren dieser Art geeignet sind und bereits etablierte Verfahren wie Pharmakophoranalyse, Homologiemodelle von Proteinen und quantenmechanische Berechnungen sinnvoll ergänzen. Der Bedarf an genauen Modellen, die eine hochoptimierte Analyse von virtuellen Substanzen erlauben, hat in der Vergangenheit dazu geführt, dass sich viele Arbeitsgruppen mit diesem Problem beschäftigt haben und ein ganzes Arsenal von Computermethoden zur Klassifikation von CYP450 Enzymen existiert [1, 2, 3]. Dazu gehören in der neueren Literatur Verfahren, die verschiedene Klassifikatoren miteinander verknüpfen. Man spricht dann auch von Konsensmodellen, die das Gesamtergebnis verbessern. Die zugrundeliegenden Algorithmen sind Entscheidungsbaumverfahren wie z.B. rekursives Partitionieren und Ensemble-Methoden wie AdaBoost oder auch die von uns in Teil III beschriebenen Support-Vektor-Maschinen. Hauptproblem in diesem Zusammenhang ist der Zugang zu genügend validierten, experimentellen Daten. Normalerweise ist die Generalisierungsfähigkeit einer Klassifikation mit der Anzahl der eingehenden Datenpunkte korreliert. Daraus kann aber nicht notwendigerweise gefolgert werden, dass ein Datensatz mit Daten aus verschiedenen Quellen auch zu einer besseren Klassifikation führt. Mit dem Fokus auf Datenqualität wurde erkannt, dass man mit unbalancierten Datensätzen trainieren muss, da die Anzahl an bekannten, validierten Datenpunkten von 2D6-Inhibitoren begrenzt ist. Umgekehrt führt die Reduktion von Nicht-Inhibitoren, um einen balancierten Datensatz zu erhalten, zu einem ungewollten Informationsverlust, da normalerweise mehr Informationen zu den Nicht-Inhibitoren vorliegen. Analog zur Anzahl der Trainingsinstanzen verhält sich die Reduktion von Eigenschaften, bei der irrelevante Deskriptoren zu einer verschlechterten Klassifikation führen. Wir haben deshalb unsere in den vorangegangenen Teilen beschriebenen Verfahren angewendet, um die metabolischen Eigenschaften von virtuellen Molekülen schnell vorherzusagen bzw. neue Moleküle mit gewünschtem metabolischem Verhalten zu synthetisieren.

2 Beschreibung der verwendeten Daten und die Vorgehensweise

Um unsere Algorithmen anwenden zu können, ist eine Darstellung in Feature-Vektoren notwendig, bei der jede chemische Struktur durch mehrere hundert physikochemische Deskriptoren oder Substruktureigenschaften, die in so genannten Fingerprints dargestellt werden, beschrieben wird. Ein Feature-Vektor ist eine normalisierte Darstellung der chemischen Struktur durch eine Anzahl von Eigenschaften, die unabhängig von der Anzahl der Atome ist. Im zweiten Schritt werden die Informationen zur 2D6-Inhibition hinzugefügt.

Wir wiederholen diesen Schritt für jedes Molekül im Datensatz, so dass wir letztendlich eine zweidimensionale Beziehung zwischen den Eigenschaften und der 2D6-Inhibition herstellen können.

Im Gegensatz zur visuellen Analyse der Klassifikationsergebnisse ist es für das Erstellen der Klassifikatoren notwendig, die zugrundeliegenden chemischen Strukturen, wie in Teil I näher beschrieben, durch physikochemische Eigenschaften zu repräsentieren. Zu diesem Zweck haben wir in unserem Fall das MOE (Molecular operating environment; chemical computing group) Programmpaket eingesetzt [13]. Zusammenfassend haben wir so 557 Deskriptoren zu unseren Molekülen berechnet, wie z.B. die Anzahl von bestimmten Atomtypen, Indizes von molekularen Graphen, topologische Deskriptoren, Autokorrelationen, Deskriptoren zur Aromatizität sowie 3D-Deskriptoren, die die Van der Waals Oberfläche der Moleküle in Kombination mit physikochemischen Eigenschaften enthalten.Für die Berechnung der 3D-Koordinaten der chemischen Strukturen haben wir CORINA (Arbeitsgruppe Prof. Gasteiger, Molecular Networks) verwendet. Die Berechnung der binären Vektoren, die eine Länge von 458 Bits haben, wurde mit dem Programm CACTVS (Dr. Ihlenfeldt, Xemistry) durchgeführt. Insgesamt wurden auf diese Weise vier Datensätze erzeugt. Die Ensemble-Datensätze (jeweils ein Training- und Testdatensatz) enthalten pro Molekül eine Zeile mit den entsprechenden Eigenschaften des gesamten Moleküls. Die Binärdatensätze (jeweils ein Training- und Testdatensatz) enthalten alle auf Substrukturen basierende Fragmente in binärer Form.

Auf der Basis eines Datensatzes mit 263 chemisch unterschiedlichen Wirkstoffen, die wir aus publizierten Daten extrahiert haben, wurde ein Trainingsdatensatz mit 185 Substanzen und ein Testdatensatz mit 78 Substanzen für die Validierung gewonnen. Weil Daten aus verschiedenen Quellen abhängig von der experimentellen Methode sind, mit denen sie erzeugt wurden, haben wir alle Substanzen mit mikromolarer Aktivität als Inhibitoren definiert. Für unseren Datensatz haben wir nur Substanzen ausgewählt, von denen entweder die Inhibition publiziert ist oder diese so weit charakterisiert sind, dass eine 2D6-Inhibition ausgeschlossen werden kann. Aus diesem Grund haben wir diejenigen Substanzen als Nicht-Inhibitoren festgelegt, bei denen die CYP2D6-Inhibition als unwahrscheinlich einzustufen ist. Der Trainingsdatensatz mit 185 Substanzen enthält 35 Substanzen, die die CYP2D6-Isoform inhibieren und 150 Nicht-Inhibitoren. Wir beziehen uns im ersten Fall auf die Klasse 1 und bei den negativen Trainingspunkten auf die Klasse 0. Der Testdatensatz enthält die gleichen Anteile von positiven und negativen Punkten wie der Trainingsdatensatz. Für die Oversampling-Methode [9] haben wir die Anzahl an Positiven im Trainingsdatensatz durch Wiederholungen der Feature-Vektoren verdreifacht. Der Testdatensatz wurde über alle angewendeten Methoden konstant gehalten. Der Split von Test- und Trainingsdatensatz wurde auf Basis von Tanimoto-Koeffizienten durchgeführt, die ein Maß für die Ähnlichkeit darstellen [7].

Die so erzeugten Datensätze wurden dann von redundanten Informationen, wie z.B. Nullspalten befreit. Zusätzlich wurde bei einem Teil der Datensätze die Anzahl der Merkmale mit dem im Teil II beschriebenen Variablenselektionsverfahren nach McCabe reduziert.

Für jeden Datensatz wurde anschließend die SVM-Methode mit unseren eigenen Kernen angewendet. Im Falle der Ensemble-Datensätze liegt eine Kombination von numerischen und kategorischen Eigenschaften vor. Die Erstellung der Modelle und die Vorhersagen wurden auf Basis von ASCII-Dateien durchgeführt. Die Optimierung der SVM-Parameter wurde durch eine Gridsuche realisiert, die im Algorithmus implementiert ist. Die Erzeugung der CYP2D6-Modelle beanspruchte auf einem PC wenige Sekunden pro Lauf während der Optimierung. Nachdem der Algorithmus trainiert war, konnten die Substanzen aus dem Testdatensatz klassifiziert werden. Die Klassifikation wurde dann anhand von Genauigkeitsmaßen bewertet [15].

3 Methoden für unbalancierte Trainingsdatensätze

Im Fall von unbalancierten Datensätzen überwiegt die Anzahl der negativen Trainingsbeispiele im Vergleich zu der Anzahl der positiven Beispiele. Zusätzlich zu der unbalancierten Klassenverteilung muss der Klassifikationsfehler berücksichtigt werden, der bei einer falsch negativen Vorhersage ungünstiger ist als bei einer falsch positiven Vorhersage [8]. Normalerweise sind die Ergebnisse von Verfahren des Maschinellen Lernens für unbalancierte Datensätze nicht überzeugend, da eine Tendenz zur Klassifikation neuer Substanzen in die überrepräsentierte Klasse besteht. Aus diesem Grund ist es notwendig, den Datensatz oder den Algorithmus so zu verändern, dass die unterrepräsentierte Klasse gleichwertig berücksichtigt wird. Innerhalb des SVM-Algorithmus kann das z.B. durch die Einführung eines Gewichtungsschemas erreicht werden. Dadurch wird allerdings die Datenverteilung nicht verändert und führt bei verrauschten, unbalancierten Datensätzen unter Umständen zu übertrainierten Modellen. Unbalancierte Datensätze können entweder durch Reduktion der überrepräsentierten Klasse bzw. durch ein wiederholtes Training von Datenpunkten der unterrepräsentierten Klasse modifiziert werden. Die Methode der Datenreduktion führt allerdings auch dazu, dass validierte negative Trainingsinformation nicht genutzt wird und damit die Generalisierungsfähigkeit des Modells reduziert wird. Aus diesem Grund haben wir die sogenannte Oversampling-Methode verwendet und die unterrepräsentierte Klasse dreifach trainiert. Die Klassifikationsergebnisse wurden anschließend noch durch Schwellwertverschiebung angepasst. Dabei wird der Schwellwert für die überrepräsentierte Klasse angehoben, damit eine Zuordnung von Testinstanzen zu dieser Klasse erschwert wird.

4 Diskussion der Ergebnisse

Nachfolgend werden die Ergebnisse mit unseren verwendeten Algorithmen diskutiert. Im Fall der Ensemble-Datensätze haben wir eine Kombination von numerischen und kategorischen Eigenschaften vorliegen. Im Fall der Binärdatensätze sind die vorhandenen strukturellen Merkmale durch sogenannte Fingerprints beschrieben, in denen die Anwesenheit ei-

ner strukturellen Einheit durch “1” bzw. bei Abwesenheit durch “0” repräsentiert wird. Um den Einfluss einer Merkmalsauswahl zu untersuchen, haben wir mit jeweils 5, 10 oder 20 bzw. allen berechneten Eigenschaften Modelle erstellt. Die Parameteroptimierung haben wir durch eine Gridsuche über einen jeweils methodenspezifischen Parameterraum durchgeführt. Die Schwellwertverschiebung wurde nach der Klassifikation mit einem Wert von 0.5 für die SVM durchgeführt. Die Erstellung der Modelle umfasst zunächst die Bestimmung der optimalen Parameter für den Algorithmus durch wiederholte 10-fache Kreuzvalidierung. Die daraus erhaltenen optimalen Parameterkombinationen wurden für die finale Erstellung des Klassifikators eingesetzt. Im nachfolgenden Schritt wurde dann ein Testdatensatz verwendet, der zu keiner Zeit im Optimierungszyklus eingesetzt wurde. Zur Diskussion der Ergebnisse wurden die Sensitivität (Hits) und die Anzahl der falsch positiven Klassifikationsergebnisse (falscher Alarm) herangezogen. Im Fall der SVM mit Gauß-Kern konnte eine dramatisch erhöhte Sensitivität beobachtet werden, die jedoch durch zusätzliche falsch positive Punkte beeinflusst wurde. Hingegen ergab die Oversampling-Methode keine deutliche Verbesserung der Sensitivität. Wurde jedoch zusätzlich der Schwellwert verschoben, wurden für alle Datensätze bessere Ergebnisse erzielt. Die besten Resultate ergab dann eine zusätzliche Merkmalsauswahl von fünf Eigenschaften. Im Vergleich dazu konnte mit dem Slater-Kern im Standardfall keine Verbesserung der Sensitivität erreicht werden. Allerdings war dort die Spezifität erhöht und zusammen mit einer Schwellwertverschiebung konnten bis zu 92% der 2D6-Inhibitoren im Testdatensatz erkannt werden. Die weitere Verwendung von Oversampling war ohne Einfluss auf die Klassifikationsergebnisse. Mit steigender Anzahl der Eigenschaften konnte ein Anstieg der Sensitivität beobachtet werden, der jedoch ebenfalls auf Kosten der Spezifität ging. Zusammenfassend kann gesagt werden, dass höhere Spezifität zu einer dramatischen Verringerung der Sensitivität geführt hat. Der Einfluss der Oversampling-Methode auf die Klassifikationsergebnisse führte zu geringeren Raten falschen Alarms, aber im Gegensatz zu unseren Annahmen war damit auch immer eine Abnahme der Sensitivität verbunden. Für die Klassifikation der Binärdatensätze haben wir den Tanimoto-Kern getestet. Besonders auffällig war die damit erzielte geringe Sensitivität im Fall der Datensätze mit 5 und 10 Eigenschaften. Da sich dieser Befund mit 20 beziehungsweise allen Eigenschaften verbesserte, kann abgeleitet werden, dass die Anzahl der verwendeten Merkmale entscheidend ist, um gute Klassifikationsergebnisse zu erhalten. Eine Merkmalsauswahl sollte deshalb bei Fingerprint-basierten Datensätzen nur vorsichtig eingesetzt werden. Die Oversampling-Methode erhöhte die Sensitivität zusammen mit geringer Abnahme der Spezifität. Nachfolgend werden die erhaltenen Ergebnisse auf Basis der chemischen Strukturen und selektierten Eigenschaften diskutiert. Basierend auf der unüberwachten Methode der Merkmalsauswahl nach McCabe selektierten wir die 20 besten Deskriptoren aus dem Ensembledatensatz. Die wichtigsten sind in Tabelle 1 mit der Beschreibung ihrer Bedeutung zusammengefasst. Die entsprechend selektierten Deskriptoren passen zu bereits in der Literatur beschriebenen Pharmakophor-, und Homologiemodellen. Diese lassen sich in Eigenschaften zusammenfassen, die die räumliche Gestalt und die Verknüpfung innerhalb der Moleküle beschreiben, wie z.B. die

chi0, chi0_C	Konnektivitätsindizes, die die Verknüpfung im Kohlenstoffgerüst beschreiben
TPSA, PEOE_VSA_HYD PEOE_PC+, Q_VSA_POS	Beschreibung der polaren und hydrophoben Anteile der Molekülоberfläche auf Basis der Gasteiger-Marsili-Ladungen
a_nN, b_single, MACCS Keys	Anzahl der N, S Atome, Einfachbindungen, SO Gruppen, Ringatome, Nitrogruppen, Heteroatome in 5-Ringen, Hydroxygruppen, Dreifachbindungen
SMR, SMR_VSA0 SMR_VSA7 bin 7	Molekularer Brechungsindex und deren Anteile
kS_sI	Elektrotopologische Eigenschaft von funktionellen Gruppen

Tabelle 1: Wichtige durch das McCabe-Verfahren identifizierte Eigenschaften.

Konnektivitätsindizes chi0 und chi0_C. Zusätzlich wurden einige Bins aus dem 166-Bin MACCS-Vektor, die Anzahl von bestimmten funktionellen Gruppen sowie elektrotopologische Indizes, die die Ladungsverteilung innerhalb der Moleküle beschreiben, selektiert. Zusammenfassend kann gesagt werden, dass die Merkmalsauswahl nur 2D-Eigenschaften selektiert hat, die somit die Basis unserer Modelle mit Merkmalsauswahl darstellen. Im Vergleich über alle verwendeten Methoden und Kerne stellte dies allerdings keine Einschränkung für die letztendlich erreichte Qualität und Genauigkeit der Modelle dar.

Abbildung 4 zeigt eine repräsentative Auswahl von diversen Verbindungen aus dem Trainingsdatensatz zusammen mit der CYP2D6-Inhibition. Wie aus dem direkten Strukturvergleich von Amitriptyline und Imipramine zu sehen ist, können kleine Abweichungen in der chemischen Struktur zu einer unterschiedlichen Klassifikation führen. Ein weiteres Beispiel dieser Art ist die gleiche Anzahl der Bindungen zwischen dem aromatischen Zentrum und dem Stickstoffatom in Venlafaxine und Dexfenfluramine aber unterschiedlicher CYP2D6-Inhibition. In den Abbildungen 1, 2 und 3 haben wir Verbindungen aus unserem Testdatensatz ausgewählt, die einige der gefundenen Unterschiede in der Klassifikation der verwendeten Algorithmen aufzeigen.

Verbindungen wie Yohimbine, Desmethylsertraline und Prevastatin wurden überwiegend richtig klassifiziert. Im Gegensatz dazu wurden Clemastine, Fentanyl und Perazine hauptsächlich falsch klassifiziert. Dieser Befund kann zum Teil durch einen Vergleich von Imi-

pramine aus dem Trainingsdatensatz und dem falsch negativ klassifizierten Perazine aus dem Testdatensatz erklärt werden. Beide Strukturen sind chemisch sehr ähnlich, aber unterscheiden sich in einem zusätzlichen Schwefelatom. Ein ähnliches Phänomen tritt bei den falsch positiven Clemastine und Fentanyl auf, bei denen Eigenschaften der 2D6-Inhibitoren aus dem Trainingsdatensatz wie O-N Abstände in Venlafaxine und Clemastine, zu einer falschen Klassifikation aufgrund einer funktionellen Untereinheit führen. Für eine Anzahl von Substanzen ist die Art der im Algorithmus verwendeten Eigenschaften von Bedeutung. Insbesondere die korrekte Klassifizierung von Nefazodone gelang mit Hilfe der numerischen Ensemble-Eigenschaften im Gegensatz zum Binärdatensatz, der zu einer falschen Klassifikation führte. Umgekehrt kann man aber auch Beispiele wie das Metoprolol finden, in denen der Binärdatensatz besser zur Klassifikation geeignet war. Die meisten der restlichen richtig klassifizierten Substanzen können auf Basis ihrer vergleichbaren Ähnlichkeit erklärt werden. Im Verlauf der Untersuchungen an diesen Datensätzen konnte generell festgestellt werden, dass nur diejenige Information im Testset abrufbar ist, die auch im Trainingsdatensatz enthalten ist. Die Möglichkeiten der Klassifikation sind durch den aufgespannten chemischen Raum der Trainingsdaten limitiert. Diese Aussage gilt unabhängig vom Algorithmus, der Art wie der Datensatz erstellt wurde und durch welche physikochemischen Eigenschaften die chemischen Strukturen beschrieben werden.

5 Zusammenfassung und Ausblick

Die humane Cytochrom P450 2D6-Isoform spielt eine Schlüsselrolle im Metabolismus von vielen neuen Wirkstoffen in der präklinischen Forschung. Wir haben einen Datensatz aus publizierten Quellen extrahiert und physikochemische Eigenschaften der Substanzen mit Methoden aus der Chemoinformatik berechnet. Auf der Basis dieser Daten haben wir mit unseren beschriebenen Methoden aus den vorangegangenen Teilen II und III Klassifikatoren erstellt. Zur Anwendung des Maschinellen Lernens haben wir unsere eigene Implementierung einer Support-Vektor-Maschine mit neuen Kernen eingesetzt. Die im Teil II beschriebene unüberwachte McCabe-Methode wurde dabei zur Merkmalsauswahl als vorgeschalteter Schritt für das überwachte Lernen eingesetzt. Nachfolgend wurde eine Analyse der klassifizierten Strukturen gezeigt und mit Beispielen aus dem Trainingsdatensatz verglichen. Es konnte gezeigt werden, dass die von uns umgesetzten Algorithmen eine effiziente Methode zur in silico Klassifikation von potentiellen neuen Wirkstoffen darstellen. Insbesondere die Verwendung von Mehrfachtraining und Schwellwertverschiebung in unseren CYP2D6-Datensätzen mit ungünstigem aktiv/nicht-aktiv Verhältnis führten so zu hochgenauen und sensiblen Klassifikatoren. Die eingesetzte Variablenselektion ergab Eigenschaften, die auf dem molekularen Level interpretiert werden können. Zusätzlich zu den bereits beschriebenen SVM-Methoden haben wir begonnen, die Maximum-Entropy-Methode (ME) [14], insbesondere den GIS-Algorithmus in Kombination mit Oversampling, zur Klassifikation einzusetzen. Diese wird normalerweise im Bereich der Spracher-

kennung genutzt. Dazu verwenden wir zur Zeit die Implementierung der frei verfügbaren openNLP-Software (sf.net/opennlp). Die ME-Methode ist ein elegantes Klassifikationsverfahren, welches auf der Tatsache beruht, dass Informationen über die Wahrscheinlichkeitsverteilung im Trainingsdatensatz bekannt sind und deshalb eine Verteilung mit der größten Entropie (nach Shannon) bestimmt werden kann. Erste Ergebnisse am Beispiel des CYP2D6-Datensatzes sind ebenfalls vielversprechend, so dass in Zukunft vergleichende Analysen vorgenommen werden.

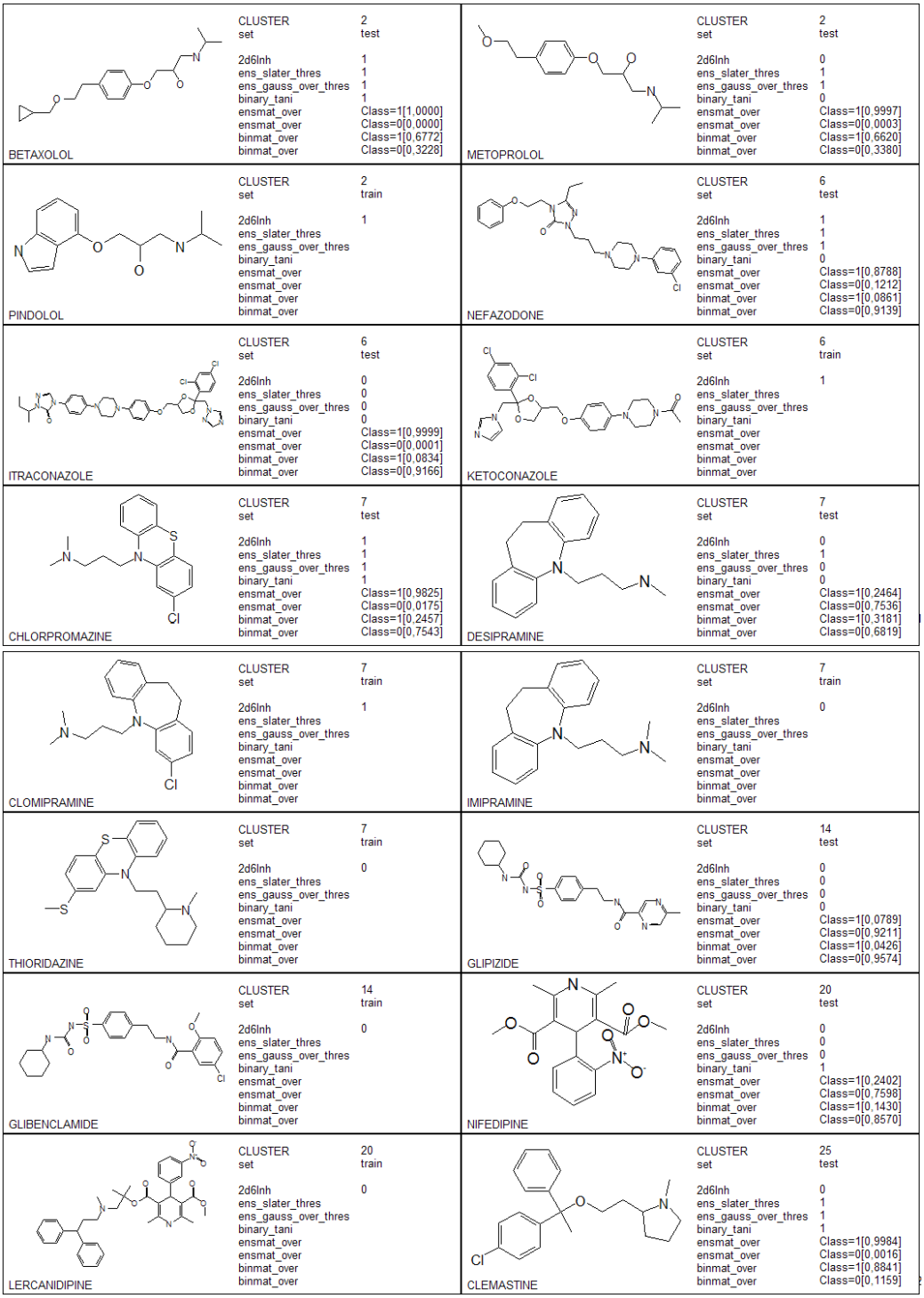


Abbildung 1: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (I).

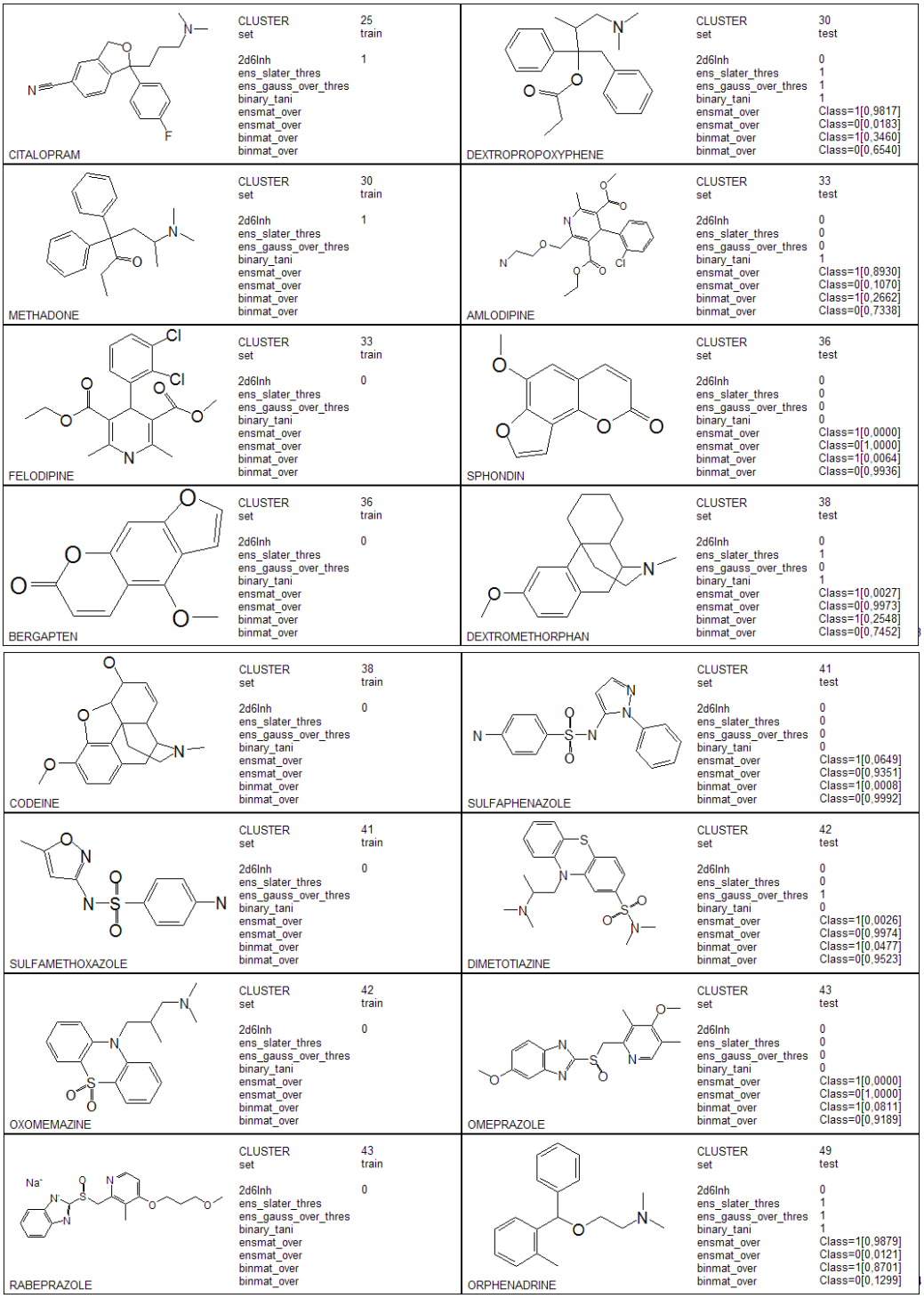


Abbildung 2: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (II).

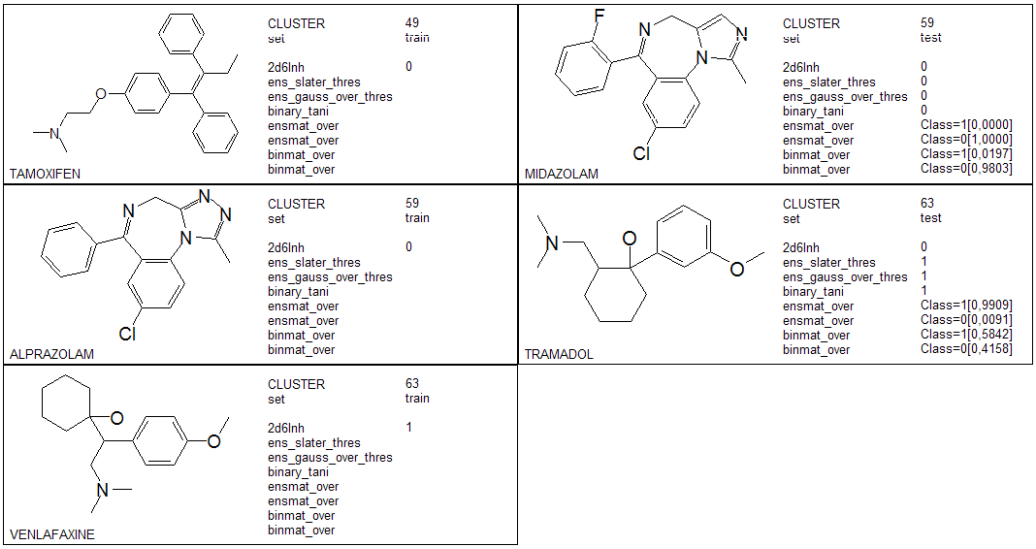


Abbildung 3: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (III).

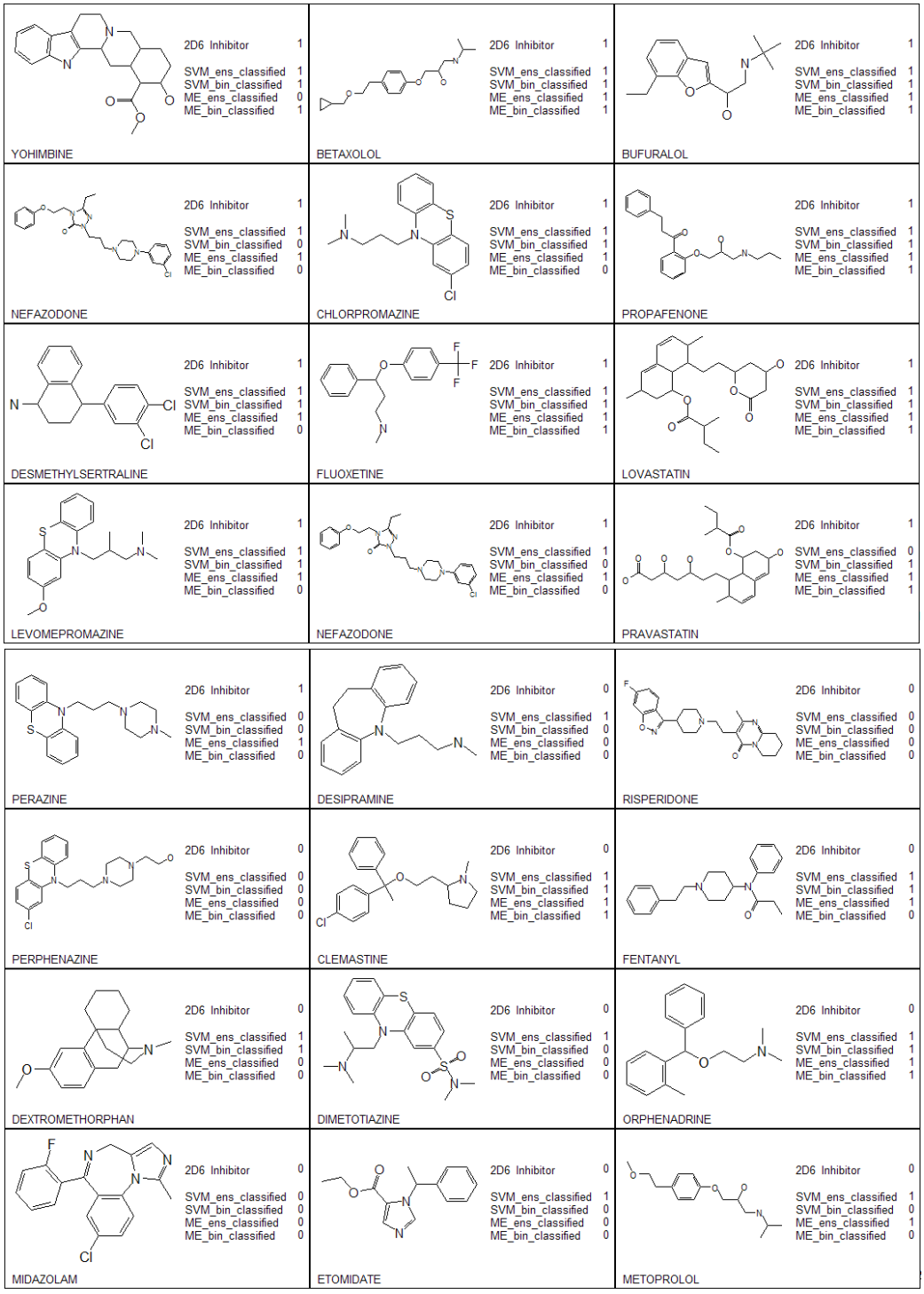


Abbildung 4: Ergebnisse aus dem Testdatensatz im Vergleich mit den verwendeten Methoden.

Literaturverzeichnis

[1] D. F. Lewis, S. Modi, M. Dickins. *Structure-activity relationship for human cytochrome P450 substrates and inhibitors.* Drug Metab. Rev. 2002, 34(1-2):69–82.

[2] C. de Graaf, N. P. Vermeulen, K. A. Feenstra. *Cytochrome P450 in silico: an integrative modeling approach.* J. Med. Chem. April 21, 2005, 48(8):2725–55.

[3] M. J. de Groot, S. Ekins. *Pharmacophore modeling of cytochromes P450.* AdV. Drug DeliVery ReV. 2002, 54(3):367–383.

[4] C. A. Kemp, J. U. Flanagan, A. J. van Eldik, J.-D. Marechal, C. R. Wolf, G. C. K. Roberts, M. J. I. Paine, M. J. Sutcliffe. *Validation of model of cytochrome P450 2D6: an in silico tool for predicting metabolism and inhibition.* Journal of Medicinal Chemistry, 2004, Vol. 47, No. 22.

[5] S. Ekins, J. Berbaum, R. K. Harrison. *Generation and validation of rapid computational filters for CYP2D6 and CYP3A4.* Drug Metab. Dispos. 2003, 31(9):1077–1080.

[6] J. M. Kriegl, T. Arnhold, B. Beck, T. Fox. *Prediction of human cytochrome P450 inhibition using support vector machines.* QSAR & Combinatorial Science 24, 2005, 491–502.

[7] P. Willett, J. M. Barnard, G. M. Downs. *Chemical similarity searching.* J. Chem. Inf. Comput. Sci. 1998, 38(6):983–996.

[8] S. Lessmann. *Solving unbalanced classification problems with support vector machines.* Proceedings of the International Conference on Artificial Intelligence, IC-AI'04 2004, 214–220.

[9] G. Rajarshi, P. C. Jurs. *Determining the validity of a QSAR model - a classification approach.* J. Chem. Inf. Model. 2005, Vol. 45, No. 1, 73.

[10] S. Rendic, F. J. Di Carlo. *Human cytochrome P450 enzymes: a status report summarizing their reactions, substrates, inducers and inhibitors.* Drug Metabolism Reviews 1997, 29, 413–580.

[11] S. Rendic. *Summary of information on human CYP enzymes: human P450 metabolism data.* Drug Metab. Rev. 2002, 34(1-2):83–448.

[12] D. Flockhart. *Cytochrome P450 drug interaction table.* `http://medicine.iupui.edu/flockhart`

[13] MOE (The Molecular Operating Environment) Version 2005.06. Software available from Chemical Computing Group Inc., 1010 Sherbrooke Street West, Suite 910, Montreal, Canada H3A 2R7. `http://www.chemcomp.com`

[14] A. Ratnaparkhi. *A simple introduction to maximum entropy models for natural language processing.* NSF Science and Technology Center for Research in Cognitive Science, Technical Report, 1997, 97–108.

[15] T. Eitrich, A. Kless, C. Druska, W. Meyer, J. Grotendorst. *Classification of highly unbalanced CYP450 data of drugs using cost sensitive machine learning techniques.* Journal of Chemical Information and Modeling, 2007 (first issue).

Bisher sind erschienen:

Modern Methods and Algorithms of Quantum Chemistry - Proceedings
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 1
ISBN 3-00-005618-1, Februar 2000, 562 Seiten
nicht mehr lieferbar

Modern Methods and Algorithms of Quantum Chemistry - Poster Presentations
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 2
ISBN 3-00-005746-3, Februar 2000, 77 Seiten
nicht mehr lieferbar

Modern Methods and Algorithms of Quantum Chemistry - Proceedings, Second Edition
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 3
ISBN 3-00-005834-6, Dezember 2000, 638 Seiten
nicht mehr lieferbar

Nichtlineare Analyse raum-zeitlicher Aspekte der hirnelektrischen Aktivität von Epilepsiepatienten
Jochen Arnold
NIC-Serie Band 4
ISBN 3-00-006221-1, September 2000, 120 Seiten

Elektron-Elektron-Wechselwirkung in Halbleitern: Von hochkorrelierten kohärenten Anfangszuständen zu inkohärentem Transport
Reinhold Lövenich
NIC-Serie Band 5
ISBN 3-00-006329-3, August 2000, 146 Seiten

Erkennung von Nichtlinearitäten und wechselseitigen Abhängigkeiten in Zeitreihen
Andreas Schmitz
NIC-Serie Band 6
ISBN 3-00-007871-1, Mai 2001, 142 Seiten

Multiparadigm Programming with Object-Oriented Languages - Proceedings
Kei Davis, Yannis Smaragdakis, Jörg Striegnitz (Hrsg.)
Workshop MPOOL, 18. Mai 2001, Budapest
NIC-Serie Band 7
ISBN 3-00-007968-8, Juni 2001, 160 Seiten

Europhysics Conference on Computational Physics - Book of Abstracts
Friedel Hossfeld, Kurt Binder (Hrsg.)
Konferenz, 5. - 8. September 2001, Aachen
NIC-Serie Band 8
ISBN 3-00-008236-0, September 2001, 500 Seiten

NIC Symposium 2001 - Proceedings
Horst Rollnik, Dietrich Wolf (Hrsg.)
Symposium, 5. - 6. Dezember 2001, Forschungszentrum Jülich
NIC-Serie Band 9
ISBN 3-00-009055-X, Mai 2002, 514 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Lecture Notes
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 10
ISBN 3-00-009057-6, Februar 2002, 548 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Poster Presentations
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 11
ISBN 3-00-009058-4, Februar 2002, 194 Seiten

Strongly Disordered Quantum Spin Systems in Low Dimensions: Numerical Study of Spin Chains, Spin Ladders and Two-Dimensional Systems
Yu-cheng Lin
NIC-Serie Band 12
ISBN 3-00-009056-8, Mai 2002, 146 Seiten

Multiparadigm Programming with Object-Oriented Languages - Proceedings
Jörg Striegnitz, Kei Davis, Yannis Smaragdakis (Hrsg.)
Workshop MPOOL 2002, 11. Juni 2002, Malaga
NIC-Serie Band 13
ISBN 3-00-009099-1, Juni 2002, 132 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Audio-Visual Lecture Notes
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 14
ISBN 3-00-010000-8, November 2002, DVD

Numerical Methods for Limit and Shakedown Analysis
Manfred Staat, Michael Heitzer (Hrsg.)
NIC-Serie Band 15
ISBN 3-00-010001-6, Februar 2003, 306 Seiten

Design and Evaluation of a Bandwidth Broker that Provides Network Quality of Service for Grid Applications
Volker Sander
NIC-Serie Band 16
ISBN 3-00-010002-4, Februar 2003, 208 Seiten

Automatic Performance Analysis on Parallel Computers with SMP Nodes
Felix Wolf
NIC-Serie Band 17
ISBN 3-00-010003-2, Februar 2003, 168 Seiten

Haptisches Rendern zum Einpassen von hochaufgelösten Molekülstrukturdaten in niedrigaufgelöste Elektronenmikroskopie-Dichteverteilungen
Stefan Birmanns
NIC-Serie Band 18
ISBN 3-00-010004-0, September 2003, 178 Seiten

Auswirkungen der Virtualisierung auf den IT-Betrieb
Wolfgang Gürich (Hrsg.)
GI Conference, 4. - 5. November 2003, Forschungszentrum Jülich
NIC-Serie Band 19
ISBN 3-00-009100-9, Oktober 2003, 126 Seiten

NIC Symposium 2004
Dietrich Wolf, Gernot Münster, Manfred Kremer (Hrsg.)
Symposium, 17. - 18. Februar 2004, Forschungszentrum Jülich
NIC-Serie Band 20
ISBN 3-00-012372-5, Februar 2004, 482 Seiten

Measuring Synchronization in Model Systems and Electroencephalographic Time Series from Epilepsy Patients
Thomas Kreutz
NIC-Serie Band 21
ISBN 3-00-012373-3, Februar 2004, 138 Seiten

Computational Soft Matter: From Synthetic Polymers to Proteins - Poster Abstracts
Norbert Attig, Kurt Binder, Helmut Grubmüller, Kurt Kremer (Hrsg.)
Winterschule, 29. Februar - 6. März 2004, Gustav-Stresemann-Institut Bonn
NIC-Serie Band 22
ISBN 3-00-012374-1, Februar 2004, 120 Seiten

Computational Soft Matter: From Synthetic Polymers to Proteins - Lecture Notes
Norbert Attig, Kurt Binder, Helmut Grubmüller, Kurt Kremer (Hrsg.)
Winterschule, 29. Februar - 6. März 2004, Gustav-Stresemann-Institut Bonn
NIC-Serie Band 23
ISBN 3-00-012641-4, Februar 2004, 440 Seiten

Synchronization and Interdependence Measures and their Applications to the Electroencephalogram of Epilepsy Patients and Clustering of Data
Alexander Kraskov
NIC-Serie Band 24
ISBN 3-00-013619-3, Mai 2004, 106 Seiten

High Performance Computing in Chemistry
Johannes Grotendorst (Hrsg.)
Bericht des Verbundprojekts:
High Performance Computing in Chemistry - HPC-Chem
NIC-Serie Band 25
ISBN 3-00-013618-5, Dezember 2004, 160 Seiten

Zerlegung von Signalen in unabhängige Komponenten: Ein informationstheoretischer Zugang
Harald Stögbauer
NIC-Serie Band 26
ISBN 3-00-013620-7, April 2005, 110 Seiten

Multiparadigm Programming 2003
Joint Proceedings of the
3rd International Workshop on Multiparadigm Programming with Object-Oriented Languages (MPOOL'03)
and the
1st International Workshop on Declarative Programming in the Context of Object-Oriented Languages (PD-COOL'03)
Jörg Striegnitz, Kei Davis (Editors)
NIC-Serie Band 27
ISBN 3-00-016005-1, Juli 2005, 300 Seiten

Integration von Programmiersprachen durch strukturelle Typanalyse und partielle Auswertung
Jörg Striegnitz
NIC-Serie Band 28
ISBN 3-00-016006-X, Mai 2005, 306 Seiten

OpenMolGRID - Open Computing Grid for Molecular Science and Engineering
Final Report
Mathilde Romberg (Editor)
NIC-Serie Band 29
ISBN 3-00-016007-8, Juli 2005, 86 Seiten

Computational Nanoscience: Do It Yourself!
Lecture Notes
Johannes Grotendorst, Stefan Blügel, Dominik Marx (Hrsg.)
Winterschule, 14. - 22. Februar 2006, Forschungszentrum Jülich
NIC-Serie Band 31
ISBN 3-00-017350-1, Februar 2006, 528 Seiten

NIC Symposium 2006 - Proceedings
G. Münster, D. Wolf, M. Kremer (Hrsg.)
Symposium, 1. - 2. März 2006, Forschungszentrum Jülich
NIC-Serie Band 32
ISBN 3-00-017351-X, Februar 2006, 384 Seiten

Alle Bände stehen online zur Verfügung unter

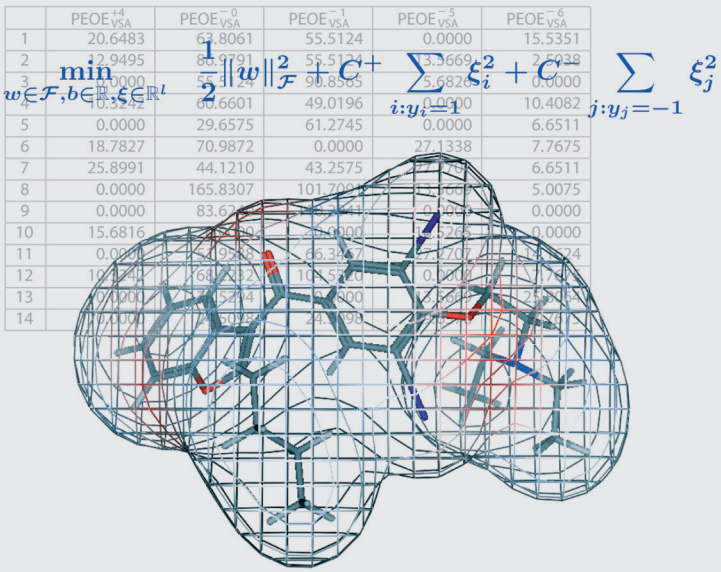
www.fz-juelich.de/nic-series/.

John von Neumann-Institut für Computing

Achim Kless, Johannes Grotendorst (Hrsg.)

GALA

Grünenthal Applied Life Science Analysis



Bericht

Zentralinstitut für Angewandte Mathematik

Publikationsreihe des John von Neumann-Instituts für Computing (NIC)
NIC-Serie Band 30

John von Neumann-Institut für Computing (NIC)

Achim Kless und Johannes Grotendorst (Hrsg.)

GALA
Grünenthal Applied Life Science Analysis

Bericht

NIC-Serie Band 30 ISBN 3-00-017349-8

Zentralinstitut für Angewandte Mathematik

Die Deutsche Bibliothek - CIP-Einheitsaufnahme
Ein Titeldatensatz für diese Publikation ist bei
Der Deutschen Bibliothek erhältlich.

Herausgeber: NIC-Direktorium
Vertrieb: NIC-Sekretariat
Forschungszentrum Jülich
52425 Jülich
Deutschland
Internet: www.fz-juelich.de/nic
Druck: Graphische Betriebe, Forschungszentrum Jülich

© 2006 John von Neumann-Institut für Computing
Es ist erlaubt, dieses Werk oder Teile davon digital oder auf Papier zum persönlichen Gebrauch oder zu Lehrzwecken zu vervielfältigen, vorausgesetzt die Kopien werden nicht kommerziell genutzt. Kopien müssen diese Copyright-Notiz und das volle Zitat auf ihrer Titelseite enthalten. Andere Vervielfältigung bedarf der vorherigen schriftlichen Genehmigung des oben genannten Herausgebers.

NIC-Serie Band 30 ISBN 3-00-017349-8

Vorwort

Dieser Forschungsbericht enthält eine zusammenfassende Darstellung der wissenschaftlichen Arbeiten und verdeutlicht anhand ausgewählter Beispiele die erzielten Ergebnisse des Data Mining Projektes GALA (Grünenthal Applied Life Science Analysis), einer Industriekooperation zwischen dem Aachener Pharmaunternehmen Grünenthal und dem Forschungszentrum Jülich. Das GALA-Projekt wurde ins Leben gerufen, um neue Wege in der Entwicklung von Medikamenten zu beschreiten. Die Arbeitsschwerpunkte wurden auf einem ersten gemeinsamen Meeting, wenige Tage vor dem Millennium-Jahreswechsel am 21. Dezember 1999 diskutiert und vereinbart. Projektziel war die Erstellung von Computerprogrammen, die die automatische Analyse und Auswertung der sich ständig vergrößernden Datenmengen in der Pharmaforschung bei Grünenthal ermöglichen. Im Prozess der Medikamentenentwicklung müssen aus einer sehr großen Anzahl von chemischen Substanzen diejenigen ausfindig gemacht werden, die im menschlichen Organismus eine bestimmte, gewünschte Wirkung (wie z.B. die Linderung eines Schmerzreizes) erzielen. Der genaue Zusammenhang zwischen der chemischen Struktur einer Substanz und den daraus resultierenden biologischen Eigenschaften ist jedoch meistens weitgehend unbekannt. Eine wichtige Aufgabe besteht nun darin, diese unbekannten Zusammenhänge, d.h. verborgenen Informationen, in vorhandenen Datenbeständen mit Hilfe geeigneter Verfahren des Data Mining aufzudecken.

Die gemeinsamen Arbeiten starteten am 1. Juli 2000 und erstreckten sich zunächst über einen Zeitraum von drei Jahren. Wissenschaftler der Firma Grünenthal und des Forschungszentrums Jülich entwickelten neue Methoden und Verfahren für die Datenanalyse in der pharmazeutischen Forschung. Es wurden Computerprogramme zur Datenvorauswertung und Klassifikation erstellt. Wichtige Aufgaben der Datenvorauswertung sind die Reduktion des Datenraumes durch die Erkennung von signifikanten Abhängigkeiten zwischen den untersuchten Merkmalen (Merkmalsauswahl) sowie die Erkennung von fehlerhaften bzw. auffälligen Datenpunkten (Ausreißereliminierung). Mit Hilfe des Klassifikationsprogramms lassen sich neue chemische Substanzen automatisch in Gruppen (Klassen) mit unterschiedlichen Erfolgsaussichten einteilen.

Im Mittelpunkt der zweiten Kooperationsphase vom 1. Juli 2003 – 30. Juni 2006 stand die Erstellung neuer, leistungsfähigerer Klassifikatoren für große Datensätze in der Pharmaforschung. Es wurden insbesondere die so genannten Support-Vektor-Maschinen (SVMs) untersucht und implementiert. Es handelt sich hierbei um moderne (überwachte) Algorithmen aus dem Forschungsgebiet des Maschinellen Lernens, einem Teilgebiet der künstlichen Intelligenz. SVMs können sowohl zur Erstellung von nichtlinearen Klassifikatoren, als auch zur Vorhersage von exakten Einzelwerten (Regression) verwendet werden. Die fundierten theoretischen Grundlagen sowie die guten Generalisierungseigenschaften führten dazu, dass Support-Vektor-Maschinen mittlerweile zu den beliebtesten modernen Lernmethoden zählen.

Der Bericht ist wie folgt gegliedert: Teil I gibt einen Überblick über die Ziele des Projektes und dokumentiert alle Publikationen und Vorträge, die im Zusammenhang mit dem GALA-Projekt entstanden sind. In Teil II werden die statistischen Verfahren zur Datenvorauswertung, in Teil III die Klassifikationsalgorithmen für Daten aus der Pharmaindustrie beschrieben. In Teil IV sind die Möglichkeiten der Datenanalyse auf der Basis der verwendeten Data Mining Pipeline erläutert. In Teil V schließlich wird am Beispiel 'Cytochrom P450 Profilierung von neuen Wirkstoffen' die Anwendung der neuen Software demonstriert.

An dieser Stelle möchten wir allen Mitarbeitern des Projektes für ihr Engagement und ihre fachlichen Beiträge herzlich danken. Das Thema des GALA-Projektes hatte einen ausgesprochen interdisziplinären Charakter und überdeckte Teilgebiete der Mathematik, Statistik, Informatik, Chemie und Pharmazie. Der Projekterfolg ist nicht zuletzt den jungen Projektmitarbeitern (Diplomanden des Studiengangs Technomathematik und Doktoranden) zu verdanken, die sich mit Enthusiasmus und Freude in das innovative Gebiet des Data Mining eingearbeitet haben. Für die stets gute Zusammenarbeit bei der administrativen Durchführung des GALA-Projektes danken wir der Forschungsleitung der Grünenthal GmbH und dem Technologie-Transfer-Büro des Forschungszentrums Jülich. Unser besonderer Dank geht an die Autoren des Berichtes, die diese ausführliche Projektdokumentation erst möglich gemacht haben.

Aachen und Jülich
November 2006

Achim Kless, Johannes Grotendorst

Inhaltsverzeichnis

Überblick

Achim Kless
Biomedizinische Forschung, Drug Discovery
Grünenthal GmbH
52078 Aachen
E-mail: Achim.Kless@grunenthal.com

1 Projektziele

Ausgangspunkt für das GALA-Projekt ist eine der Kernaufgaben der präklinischen Forschung, die auf den einzelnen Stufen der Forschung anfallenden Daten effizient zu interpretieren, um aus einem Pool von mehreren Millionen Substanzen diejenigen herauszufinden, die für eine humane Anwendung geeignet sind. Dafür müssen chemische Substanzen zahlreiche Hürden überwinden, die durch vier Bereiche repräsentiert werden. Man unterscheidet die Bereiche in silico, in vitro, in vivo und Klinik. Der in silico Bereich umfasst den chemischen Raum, d.h. alle chemisch möglichen bzw. zugänglichen Substanzen. Das können bereits bekannte chemische Wirkstoffe, neu hergestellte Substanzklassen oder einfach so genannte virtuelle Substanzen sein, die zum Zeitpunkt der Betrachtung nur als Computermodelle vorliegen. In der nächsten Stufe wird ein Teil der Substanzen in einem in vitro Modell getestet. Das in vitro Modell stellt ein molekulares Target (Schloss) z.B. in einer Zellmembran dar, für das eine Substanz (Schlüssel) gesucht wird, die eine Interaktion nach dem Schlüssel-Schloss Prinzip zeigt. Im nächsten Schritt der Lead-Optimierung werden die aktiven Hitsubstanzen solange optimiert, bis eine hohe Affinität und Selektivität zum molekularen Target gefunden wird. Dies ist notwendig, um eine möglichst kleine Anzahl von Substanzen in der nachfolgenden in vivo Testphase einzusetzen, in der das pharmakologische Wirkprinzip überprüft wird. Die so gefundenen Substanzen müssen dann noch geeignete ADMET-Eigenschaften (Absorption, Distribution, Metabolization, Excretion, Toxicity) besitzen, was eine Grundvoraussetzung für eine humane Anwendung und die Zulassung durch die Behörden der FDA (Food and Drug Administration) oder BfArm (Bundesinstitut für Arzneimittel und Medizinprodukte) ist. Aus jeder Stufe der

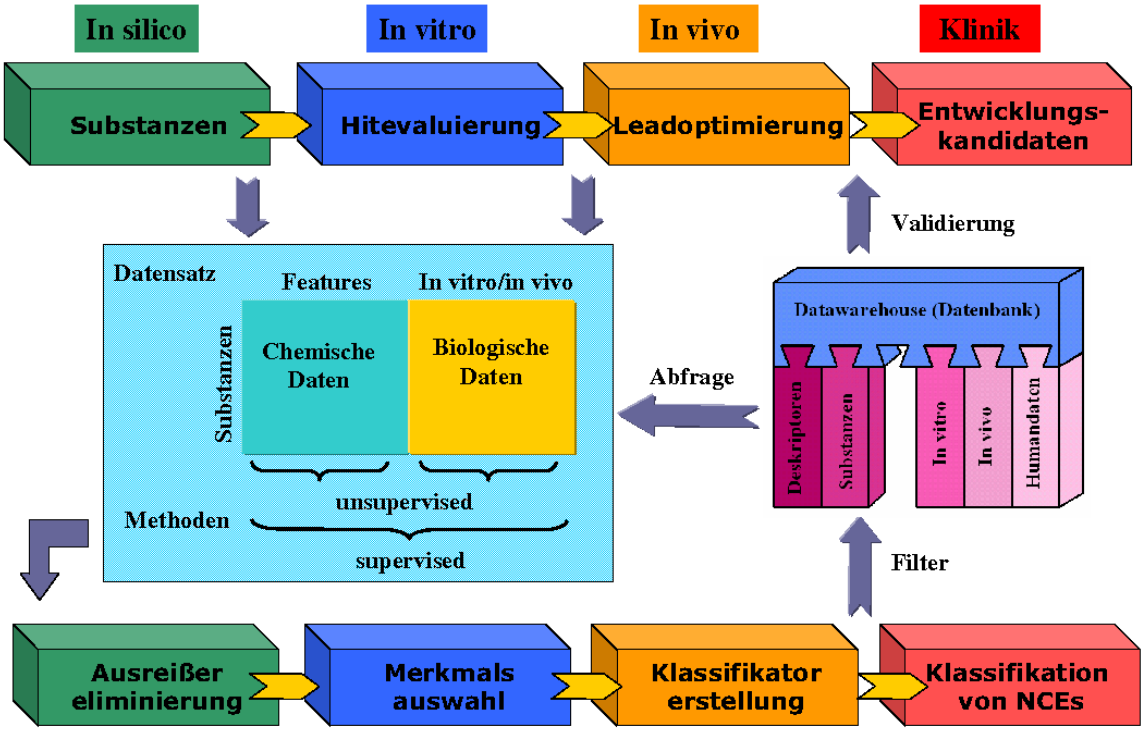


Abbildung 1: Zusammenhänge zwischen F&E-Pipeline und Data Mining.

Entwicklung einer Substanz leiten sich grundsätzliche Fragestellungen ab, die immer wieder beantwortet werden müssen. Das Projektziel ist deshalb die effektive Unterstützung der Prozesse innerhalb der präklinischen Forschung, um die Entwicklungszeiten von 10-15 Jahren für innovative Arzneimittel signifikant zu verkürzen.

2 Fragestellungen und Generierung der Datensätze

Wie in Abbildung 1 dargestellt, müssen auf allen Entwicklungsstufen Entscheidungsprozesse zur Selektion von Substanzen herangezogen werden, da aus Kapazitäts- sowie Zeitgründen nicht alle Substanzen bzw. der gesamte chemische Raum betrachtet werden können. Da die Interpretation von chemischen Strukturen durch den Computer nur indirekt möglich ist, werden diese im ersten Schritt zunächst geeignet transformiert (Abbildung 2). Dazu dienen Verfahren aus der Chemoinformatik. Die Atomkoordinaten der chemischen Strukturen können nicht direkt als Eigenschaften herangezogen werden, weil die Anzahl der Atome von Struktur zu Struktur unterschiedlich ist. Die Anwendung von Lernalgorithmen erzwingt als Input Eigenschaftsvektoren, die sogenannten Feature-Vektoren,

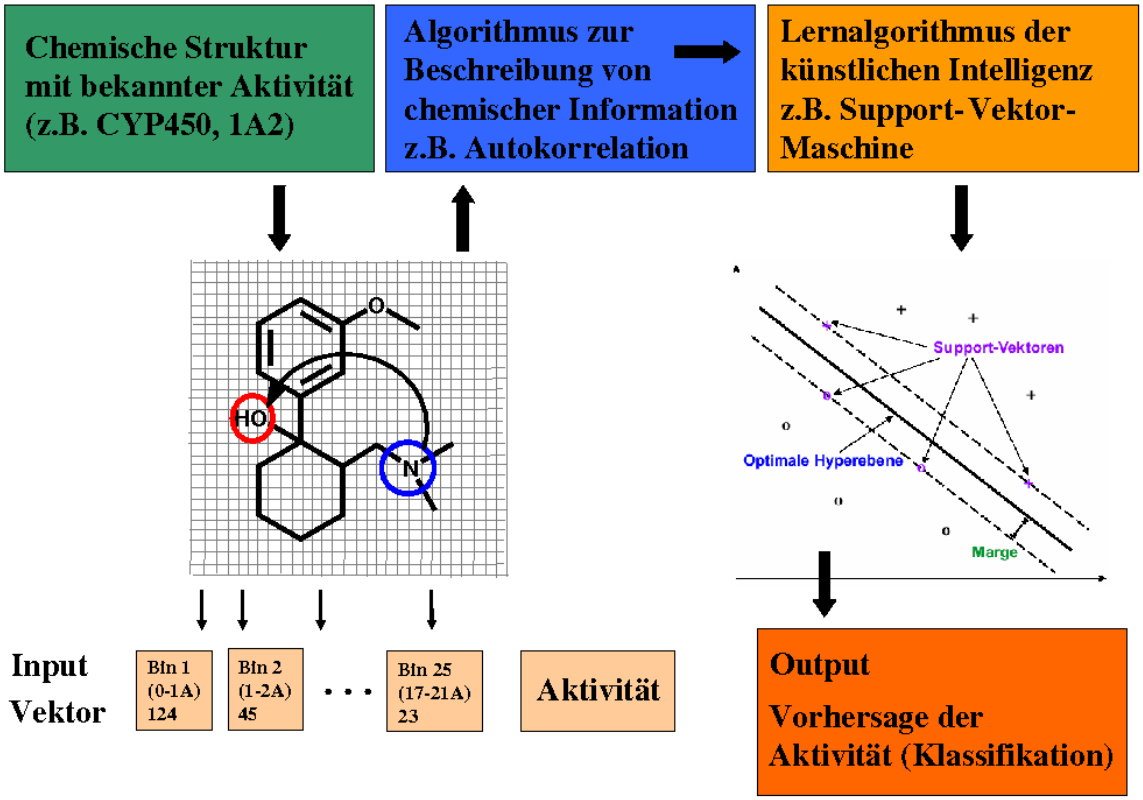


Abbildung 2: Chemische Informationen und Klassifikation.

die immer die gleiche Anzahl an Merkmalen enthalten und eine von der Größe der Moleküle unabhängige, normalisierte Form haben. Die berechneten Eigenschaften müssen auch unabhängig von der Ausrichtung der Moleküle im Raum sein. Eine bekannte Größe ist in diesem Zusammenhang die so genannte Autokorrelation. Für verschiedene Punkte der Moleküloberfläche oder des Raumes, der das Molekül umgibt, wird eine Moleküleigenschaft, wie z.B. die TPSA (Topological Polar Surface Area), berechnet. Aus den Wertepaaren der berechneten Moleküleigenschaft der Punktepaare, deren Abstände in einem vorgegebenen Intervall liegen (in Abbildung 2 werden diese Intervalle mit Bin 1, ..., Bin 25 bezeichnet), wird eine Korrelation berechnet: die Autokorrelation der Moleküleigenschaft für das betrachtete Intervall (Bin) [1, 2]. Durch die Anzahl der Bins werden der Grad der Auflösung und die Länge des Vektors festgelegt. Wählt man als Punkte bestimmte Atome des Moleküls und ordnet den Atomen als ‘Eigenschaft’ den konstanten Wert 1 zu, so ergibt die formale Berechnung der Autokorrelation die Häufigkeitsverteilung der Atomabstände . Der Zusammenhang von zwei Moleküleigenschaften kann entsprechend durch die Kreuzkorrelation der Merkmale beschrieben werden. Wählt man hierbei als Punkte die Atome und als Eigenschaften die Indikatorfunktionen (mögliche Werte: 0,1) zweier Atomtypen

(z.B. Sauerstoff und Stickstoff), dann ergibt die formale Berechnung der Kreuzkorrelation die Häufigkeitsverteilung der Abstände der betrachteten Atomtypen [3]. Eigenschaftsvektoren können auch aus einzelnen, voneinander unabhängigen, molekularen Deskriptoren aufgebaut werden. Dies können beispielsweise physikochemische Eigenschaften wie Molekulargewicht, Löslichkeit oder Dipolmoment sein. Wieder andere fassen die Anzahl von funktionellen Gruppen oder größeren molekularen Untereinheiten (Fragmenten) und Atomkonnektivitäten in binären Strings als sogenannte Fingerprints zusammen. Der Vorgang wird für alle betrachteten Substanzen und idealerweise für einen möglichst großen chemischen Raum durchgeführt. Da die so berechnete Gesamtdatenmenge sehr groß ist und zu jeder chemischen Substanz nur einmal ein Feature-Vektor aufgebaut werden muss, werden die Daten in einer Datenbank mit optimierter Datenstruktur, dem sogenannten Data Warehouse, abgelegt. Damit kann für jede Fragestellung flexibel ein Datensatz erstellt werden.

Im zweiten Schritt werden dann die zu den einzelnen Substanzen erhobenen targetspezifischen (in vitro) bzw. spezifisch für das Indikationsgebiet (in vivo) notwendigen biologischen Daten hinzugefügt. Dies kann ebenfalls durch eine Abfrage im Data Warehouse realisiert werden. Jede einzelne chemische Struktur wird dann durch einen Vektor repräsentiert, der die Struktureigenschaften (Features) mit den in vitro und/oder in vivo Daten verknüpft. Wird dies für jede Struktur wiederholt, erhält man eine Tabelle, die aus einigen hundert Zeilen, eine Zeile pro chemischer Substanz, mehreren hundert Spalten für die zugehörigen Features und wenigen Spalten für die biologischen Eigenschaften besteht. Bei den Datensätzen handelt es sich vereinfacht um eine Matrix, bei der pro Zeile eine chemische Substanz mit allen Merkmalen aus dem Data Warehouse repräsentiert wird. Die so erhaltene Darstellung stellt eine Beziehung zwischen den Substanzeigenschaften und einem gewünschten pharmakologischen Profil für eine Zielindikation her. Durch diesen Integrationsschritt ist es auch möglich, mit Hilfe von Algorithmen eine Automatisierung der Analyse durchzuführen. Diese kann durch zwei fundamental unterschiedliche Data Mining Verfahren erreicht werden. Man unterscheidet so genannte unüberwachte (unsupervised) Algorithmen, bei denen die Daten in kleinere und bezüglich der Deskriptoren homogene Gruppen unterteilt werden, sowie überwachte (supervised) Methoden, welche Klassifikatoren erzeugen, mit deren Hilfe es möglich ist, neue chemische Substanzen mit noch unbekannten Feature-Kombinationen bzgl. der gesuchten Eigenschaft zu profilieren. Die Klassifikatoren können dann nach erfolgreicher Evaluierung als Filter verwendet werden.

3 Datenvorauswertung

Ausgangspunkt sind die aus dem Data Warehouse extrahierten Datensätze, die noch von redundanten Informationen befreit werden müssen. Oberste Prämisse ist die Reduktion der Daten auf der Basis einer rein abstrakten Sichtweise, bei der kein Vorwissen über die

zu analysierenden Datensätze eingebracht wird. Der Prozess der Vorauswertung der Datensätze verfolgt mehrere Ziele. Eine wichtige Aufgabe ist die Reduktion der Dimensionalität des Datenraumes durch die Erkennung von signifikanten Abhängigkeiten zwischen den untersuchten Merkmalen. Da bis zu tausend und mehr Merkmale vorliegen, können hunderttausende von Merkmalspaaren generiert werden, deren Untersuchung manuell zu aufwändig ist. Mit Hilfe einer weiteren Methode, der so genannten Variablenselektion, erfolgt die Eingrenzung auf signifikante Merkmale. Ein weiterer Schritt ist die Erkennung und Priorisierung von auffälligen Datenpunkten, den so genannten Ausreißern. Damit ergibt sich insgesamt eine im Vergleich zum Ausgangsdatensatz reduzierte Matrix, die im Optimalfall dieselbe Information beinhaltet.

3.1 Merkmalsauswahl

Ziel ist die Verringerung der in den Daten enthaltenen Redundanz, die sich zu einem großen Teil in den Korrelationen zwischen den Merkmalen widerspiegelt. Einfache statistische Verfahren zur Aufdeckung linearer Abhängigkeiten sind oft nicht ausreichend, um komplexe Daten zu analysieren. In den konkreten Anwendungsfällen ist es häufig notwendig, nichtlineare Abhängigkeiten zu berücksichtigen. Die üblicherweise verwendete Produkt-Moment-Korrelation ist ein Maß für den linearen Zusammenhang zweier Merkmale und im Allgemeinen nicht geeignet, um ausgeprägt nichtlineare Abhängigkeiten zu charakterisieren. In einigen Fällen sollte die Berechnung der Produkt-Moment-Korrelation daher mit einer Datentransformation, meist eine Logarithmierung, kombiniert werden. Alternativ kann die Rangkorrelation verwendet werden, die invariant gegenüber monotonen Transformationen und somit ein Maß für monotone nichtlineare Zusammenhänge ist. Die Hauptkomponentenanalyse (PCA) ist ein Verfahren zur Dimensionsreduktion, das auf der Eigenwertzerlegung der Korrelationsmatrix basiert. Die Daten werden dabei durch eine kleine Anzahl von unkorrelierten Linearkombinationen, den so genannten Hauptkomponenten, dargestellt, die einen wesentlichen Anteil der in den Daten enthaltenen Streuung repräsentieren. Der Informationsverlust kann dadurch minimal gehalten werden. Man erreicht hierdurch jedoch meist nur, dass die Gesamtheit der Beobachtungspunkte durch den Raum mit reduzierter Dimensionalität gut beschrieben wird. Dass jedoch dieser Satz von Hauptkomponenten dazu geeignet ist, das Problem der Klassenzugehörigkeit zu beschreiben, ist nicht garantiert. Neben dieser Hauptkomponentenanalyse werden Verfahren auf der Grundlage einer Merkmalsauswahl unter Berücksichtigung der durchzuführenden Klassifikation eingesetzt. Es werden diejenigen Merkmale ausgewählt, deren Verteilungen in den Klassen des Trainingsdatensatzes die größten Unterschiede aufweisen. Dazu kann beispielsweise ein Chi-Quadrat-Test verwendet werden, bei dem die Aussagekraft jeder einzelnen Dimension für sich allein bezüglich der Klassenunterscheidung getestet wird. Besteht dann zwischen einigen der selektierten Merkmale eine hohe Korrelation, werden die redundanten Merkmale zusätzlich aus der Datenmatrix entfernt und somit nicht mehr bei der Erstellung der Klassifikatoren berücksichtigt. Die Erkennung der bezüglich der

gestellten Aufgabe entscheidenden Merkmale ist auch für die Diskussion der Klassifikationsergebnisse und die spätere Optimierung der chemischen Substanzen von Bedeutung. Ein häufiges Problem ist die anschließende Interpretation der nicht selten abstrakten physikochemischen Deskriptoren. Durch eine Suche der entsprechenden Deskriptoren in einem großen Pool virtueller Substanzen im Data Warehouse wird versucht, diese anhand der gefundenen Strukturbeispiele plausibel zu erklären.

3.2 Ausreißereliminierung

Eine weitere Reduktion des ursprünglichen Datensatzes ergibt sich mittels Ausreißereliminierung. In diesem Fall werden nicht wie bei der Merkmalsauswahl die Anzahl der Spalten, sondern durch die Eliminierung von Trainingsbeispielen die Anzahl der Zeilen reduziert. Das können Mess- oder Eingabefehler sein, so dass die Ausreißerdetektion auch dazu dient, Inkonsistenzen im Data Warehouse zu erkennen. Die Filterung geeigneter chemischer Strukturen ist generell eng mit dem Problem der Ausreißererkennung verknüpft, da sich häufig interessante Substanzen signifikant in wenigen Merkmalen unterscheiden. Einige der Ausreißer können aus dem Datensatz entfernt werden, was gleichzeitig durch die abnehmende Anzahl an Datenpunkten die Erstellung der Klassifikatoren beschleunigt. Im Mittelpunkt des Interesses stehen dabei nicht die Beobachtungen, die durch extreme Werte einzelner Merkmale auffallen und die leicht aufzudecken sind, sondern diejenigen, die von der multivariaten Struktur der Daten abweichen und hierdurch die Analysen stark beeinflussen können. Die Ausreißeridentifizierung stellt keinen isolierten Arbeitsschritt dar, sondern ist Bestandteil der Datenaufbereitung, dem so genannten Data Cleaning. Die endgültige Analyse wird dann für den reduzierten Datensatz durchgeführt. Zur Entscheidung, ob ein Datenpunkt als Ausreißer zu bewerten ist, ist stets auch eine Aussage zur Signifikanz (Verlässlichkeit) des Ergebnisses notwendig. Die dafür notwendige Hypothesenverteilung kann durch Expertenwissen eingebracht werden und erfordert an dieser Stelle eine interdisziplinäre Diskussion. Alternativ zur Ausreißereliminierung können robuste statistische Verfahren verwendet werden, die den Einfluß einzelner fehlerbehafteter Beobachtungen begrenzen.

4 Lernalgorithmen

Nach erfolgter Reduktion des Datensatzes auf wesentliche Merkmale und chemische Beispiele erfolgt im nächsten Schritt die Erstellung der Klassifikatoren aus diesem Trainingsdatensatz. Ein Klassifikator stellt einen Filter dar, der auf Basis der Feature-Vektoren unbekannte Substanzen in Gruppen (Klassen) einteilt. Dafür werden nichtlineare mathematische Verfahren und wahrscheinlichkeitstheoretische Betrachtungen herangezogen. Der Klassifikator wird dann auf alle Substanzen in der Datenbank angewendet. Die erhaltenen

Gruppen werden anschließend wieder in der Datenbank abgelegt und können so für weitere Analysen herangezogen werden.

4.1 Clusterung (unüberwachte Verfahren)

In den hochdimensionalen Räumen, in denen sich die Daten der pharmazeutischen Forschung befinden, ist man oft an der Bildung geeigneter Cluster interessiert. Typischerweise liegen Beoachtungen mit drei- bis vierstelliger Anzahl an Dimensionen vor. Dabei können einige Raumdimensionen diskret, andere wiederum kontinuierlich sein. Allein die Frage der Normierung der verschiedenen Raumdimensionen zeigt, dass die Metrik des Raumes a priori keineswegs eindeutig festgelegt ist. Teilt man jede der Koordinaten dieses Raumes in nur zwei Abschnitte, so entstehen durch diese Unterteilung schon bei einem hundertdimensionalen Raum 2^{100} Zellen, für die jeweils entschieden werden muss, ob sie Teil eines Clusters sind. Um den Raum durch Beobachtungen abzudecken, braucht man pro Zelle mindestens eine Beobachtung. Da die Zugehörigkeit zu einem Cluster einem Bit Information entspricht, benötigt man einen Datenspeicher mit riesiger Kapazität. Tatsächlich liegen in der Regel in einem Standardtrainingsdatensatz maximal einige tausend Feature-Vektoren vor, die nicht ausreichen, um einen hundertdimensionalen Raum auszufüllen. Dies hat zur Konsequenz, dass die Information in einem hochdimensionalen Datenraum nur sehr dünn besetzt ist.

Im Rahmen des Projektes werden Cluster-Methoden für unterschiedliche Fragestellungen verwendet, z.B. zur Überprüfung einer gegebenen Klassifikation auf einem Trainigsdatensatz oder zur Verkleinerung von einzelnen stark aufgeblähten Klassen, um robuste Klassifikatoren erstellen zu können.

4.2 Erstellung von Klassifikatoren (überwachte Verfahren)

In den letzten Jahren haben sich für komplexe Aufgabenstellungen die nichtlinearen Algorithmen, wie beispielsweise Entscheidungsbäume und neuronale Netze, insbesondere aber auch die sogenannten Support-Vektor-Maschinen (SVMs) als geeignet erwiesen. Es handelt sich dabei um moderne Algorithmen aus dem Forschungsgebiet der künstlichen Intelligenz, die sowohl zur Generierung von nichtlinearen Klassifikatoren für beliebig viele Gruppen, als auch zur Vorhersage von exakten Einzelwerten (Regression) verwendet werden können. Zur Optimierung der Modellparameter muss festgelegt werden, welche Kriterien herangezogen werden, um Fehler zu bewerten, die während der Trainingsphase auftreten. Dieser Aspekt ist besonders kritisch, da intuitiv keine Fehlklassifikationen auftreten sollen, damit aber häufig Klassifikatoren erzeugt werden, die zu stark an die verfügbaren Daten angepasst sind (Overfitting) und im Extremfall zu Fehlerquoten von bis zu 100 Prozent auf unbekannten Daten führen können. Eine besondere Stärke des Verfahrens ist, dass große Feature-Vektoren verwendet werden können und auf redundante

Merkmale weniger sensitiv reagiert wird, als dies bei anderen distanzbasierten Verfahren der Fall ist. Die Möglichkeit, nichtlineare Abhängigkeiten zu modellieren, ist ausschlaggebend für die Güte einer Klassifikation auf komplexen Daten. Das Erlernen ausschließlich linearer Funktionen, auch auf reduzierten Datensätzen, führt im Allgemeinen zu schlechteren Ergebnissen. Im Gesamtergebnis klassifiziert eine nichtlineare Funktion vorhandene Datenpunkte genauer und erfasst die zugrundeliegende Klasseneinteilung besser.

Support-Vektor-Maschinen erlernen eine nichtlineare Klassifikationsfunktion, indem sie alle Datenpunkte geeignet transformieren und dann eine einfache lineare Funktion für die Klassentrennung verwenden. Dabei ist es besonders günstig, dass diese Transformation nicht in einem zusätzlichen Schritt vom Nutzer durchgeführt werden muss, sondern für alle nötigen Berechnungen ein so genannter Kern verwendet wird. Ein Kern ist eine Funktion zur Berechnung von Skalarprodukten im Raum der transformierten Daten, welche jedoch die ursprünglichen Daten verwendet und die Transformation damit in impliziter Form selbst durchführt. Dieses Verfahren führt zu schnellen und sicheren Prognosen auf unbekannten Daten. In der Praxis wählt man meist einen Standardkern, beispielsweise den Gauß-Kern. Damit ähneln Support-Vektor-Maschinen stark den in der Literatur häufig beschriebenen neuronalen Netzen, bei denen die Daten selbst nicht transformiert, jedoch die Gewichte der linearen Klassifikationsfunktion nichtlinear angepasst werden. Im Gegensatz zu rein linearen Klassifikatoren können SVMs mit sehr vielen Merkmalen arbeiten. Wendet man dann den Klassifikator auf einen Testdatensatz mit noch unbekannten, neuen Substanzen – den sogenannten New Chemical Entities (NCEs) – an, können diese aufgrund der erhaltenen Vorhersage bewertet werden.

Ein großer Vorteil der in den folgenden Kapiteln beschriebenen Algorithmen gegenüber klassischen Verfahren wie zum Beispiel der Substruktursuche (Ähnlichkeit) ist die Möglichkeit, innovative Substanzen zu identifizieren, die nicht bereits bekannten Substanzen ähnlich sind (Me-Too-Drugs). Somit stellen die in diesem Band dargestellten Vorgehensweisen und Verfahren ein innovatives Potential zur Entdeckung neuartiger Wirkstoffe dar.

5 Dokumentation

5.1 Publikationen

- Thorsten Dickhaus
  Statistische Verfahren für das Data Mining in einem Industrieprojekt
  Interner Bericht FZJ-ZAM-IB-2003-08, Forschungszentrum Jülich, Mai 2003
- Tatjana Eitrich
  Support-Vektor-Maschinen und ihre Anwendung auf Datensätze aus der Forschung
  Bericht Jül-4096, Forschungszentrum Jülich, Oktober 2003

- Natali Zint
  Robuste Verfahren zur Berechnung von Korrelationsmatrizen sowie zur Hauptkomponentenanalyse
  Beiträge zum Wissenschaftlichen Rechnen – Ergebnisse des Gaststudentenprogramms 2003 des John von Neumann-Instituts für Computing
  Esser, Rüdiger (Hrsg.), Interner Bericht FZJ-ZAM-IB-2003-10, Dezember 2003
- Forschungszentrum Jülich
  Raffinierte Datenanalyse kann Suche nach neuen Medikamenten verkürzen
  Pressemitteilung, Dezember 2003
- Claudia Druska
  Robuste statistische Verfahren für das Data Mining
  Interner Bericht FZJ-ZAM-IB-2004-05, Forschungszentrum Jülich, März 2004
- Achim Kless, Thorsten Dickhaus, Wolfgang Meyer, Johannes Grotendorst
  Data Mining in der pharmazeutischen Forschung und Entwicklung
  Sonderheft LITUS – Laboratory IT User Service, GIT Verlag, April 2004
- Achim Kless, Tatjana Eitrich, Wolfgang Meyer, Johannes Grotendorst
  Data Mining in F&E
  BioWorld, Branchenpublikation für die Biotechnologie, BioTalk GmbH, Schweiz, Mai 2004
- Achim Kless, Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Support Vector Machines Implementing the Nearest Point Algorithm
  Lecture Notes in Artificial Intelligence, Vol. 3303, 191-205, J. A. López, E. Benfenati, W. Dubitzky (Eds.), Springer Verlag, Dezember 2004
- Claudia Albrecht
  Automatisierte Daten-Transformation in einem Data Mining Projekt
  Beiträge zum Wissenschaftlichen Rechnen – Ergebnisse des Gaststudentenprogramms 2005 des John von Neumann-Instituts für Computing
  Esser, Rüdiger (Hrsg.), Interner Bericht FZJ-ZAM-IB-2005-13, Dezember 2005
- Tatjana Eitrich, Bruno Lang
  Analysis of Support Vector Machine Training Costs for Large and Unbalanced Data from Pharmaceutical Industry
  Proceedings of the International Conference on Artificial Intelligence and Machine Learning (AIML), Kairo, Dezember 2005
- Tatjana Eitrich, Achim Kless, Claudia Druska, Wolfgang Meyer, Johannes Grotendorst
  Classification of Highly Unbalanced CYP450 Data of Drugs Using Cost Sensitive Machine Learning Techniques
  Journal of Chemical Information and Modeling, 2007 (first issue)

- Achim Kless, Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Maximum Entropy Methods
  Poster, 16. European Symposium on QSAR and Molecular Modelling, 2006

5.2 Vorträge

- Wolfgang Meyer
  Statistische Modellierung
  GALA Kick-Off-Meeting, Aachen, 21.12.1999
- Johannes Grotendorst
  Computersimulation in Forschung und Anwendung
  GALA Kick-Off-Meeting, Aachen, 21.12.1999
- Heiner Müller-Krumbhaar
  Algorithmen zur Analyse von Grünenthal-Daten
  2. GALA-Meeting, Stolberg, 05.12.2000
- Wolfgang Meyer
  Statistische Analyse von Grünenthal-Daten
  2. GALA-Meeting, Stolberg, 05.12.2000
- Achim Kless
  Data Mining in der pharmazeutischen Industrie
  Kolloquium über Parallelverarbeitung in technisch-naturwissenschaftlichen Anwendungen, Forschungszentrum Jülich, 14.05.2001
- Heiner Müller-Krumbhaar
  Algorithmen zur Analyse von Grünenthal-Daten
  3. GALA-Meeting, Stolberg, 19.07.2001
- Wolfgang Meyer, Thorsten Dickhaus
  Statistische Analyse von Grünenthal-Daten
  3. GALA-Meeting, Stolberg, 19.07.2001
- Wolfgang Meyer, Thorsten Dickhaus
  Explorative Datenanalyse im GALA-Projekt
  ZAM-Jahresabschlusskolloquium, Jülich, 13.12.2001
- Thorsten Dickhaus
  C-Programm für das GALA-Projekt
  4. GALA-Meeting, Stolberg, 21.02.2002
- Wolfgang Meyer
  Analyse von Dosis-Wirkungs-Zeitreihen
  4. GALA-Meeting, Stolberg, 21.02.2002

- Thorsten Dickhaus
  Statistische Datenanalyse für das GALA-Projekt
  5. GALA-Meeting, Aachen, 26.11.2002

- Achim Kless
  Data Mining in der pharmazeutischen Industrie
  Kolloquium "Praxis der Datenverarbeitung"
  Zentrum für Angewandte Informatik, Universität zu Köln, 08.01.2003

- Tatjana Eitrich
  Binäre Support-Vektor-Maschinen
  6. GALA-Meeting, Stolberg, 10.07.2003

- Thorsten Dickhaus
  Neue Entwicklungen im Outlier-Programm
  6. GALA-Meeting, Stolberg, 10.07.2003

- Natali Zint
  Robuste Verfahren zur Kovarianzberechnung und Hauptkomponentenanalyse
  Vortrag im Gaststudenten-Programm, John von Neumann-Institut für Computing, Jülich, 02.10.2003

- Tatjana Eitrich
  Cytochrome P450 Classification of Drugs with Support Vector Machines Implementing the Nearest Point Algorithm
  International Symposium Kelsi, Mailand, 26.11.2004

- Claudia Druska
  Statistische Verfahren zur Datenvorauswertung in der pharmazeutischen Forschung
  7. GALA-Meeting, Aachen, 01.03.2005

- Tatjana Eitrich
  Klassifikationsalgorithmen für Daten aus der Pharmaindustrie
  7. GALA-Meeting, Aachen, 01.03.2005

- Johannes Grotendorst
  Kooperation Forschungszentrum Jülich und Grünenthal GmbH - Status und Perspektiven
  7. GALA-Meeting, Aachen, 01.03.2005

- Claudia Albrecht
  Automatisierte Daten-Transformation im GALA-Projekt
  Vortrag im Gaststudenten-Programm, John von Neumann-Institut für Computing, Jülich, 27.09.2005

- Tatjana Eitrich
  Analysis of Support Vector Machine Training Costs for Large and Unbalanced Data from Pharmaceutical Industry
  International Conference on Artificial Intelligence and Machine Learning (AIML), Kairo, 20.12.2005

Literaturverzeichnis

[1] H. Bauknecht, A. Zell, H. Bayer, P. Levi, M. Wagener, J. Sadowski, J. Gasteiger. *Locating Biologically Active Compounds in Medium-Sized Heterogeneous Datasets by Topological Autocorrelation Vectors: Dopamine and Benzodiazepine Agonists.* J. Chem. Inf. Comput. Sci. 1996, 36, 1205-1213.

[2] M. Wagener, J. Sadowski, J. Gasteiger. *Autocorrelation of Molecular Surface Properties for Modeling Corticosteroid Binding Globulin and Cytosolic Ah Receptor Activity by Neural Networks.* J. Am. Chem. Soc. 1995, 117, 7769-7775.

[3] G. Schneider, W. Neidhard, T. Giller, G. Schmid. *"Grundgerüstwechsel" (Scaffold-Hopping) durch topologische Pharmakophorsuche: ein Beitrag zum virtuellen Screening.* Angew. Chem. 1999, 111, 3068-3070.

Statistische Verfahren zur Datenvorauswertung

Claudia Druska, Thorsten Dickhaus, Wolfgang Meyer

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: {c.druska, w.meyer}@fz-juelich.de

1 Einleitung

Die Entwicklung eines neuen Medikamentes ist ein langwieriger Prozess: Aus einer sehr großen Anzahl von chemischen Substanzen müssen diejenigen ausgewählt werden, die im Organismus eine bestimmte, gewünschte Wirkung (wie z.B. die Linderung eines Schmerzreizes) erzielen. Diese Fähigkeit ist eine biologische Eigenschaft der betreffenden Substanz und kann mit geeigneten Messverfahren bzw. Versuchen quantifiziert werden. Solche Messungen werden in der pharmazeutischen Forschung intensiv durchgeführt und ihre Ergebnisse in Datenbanken abgespeichert. Hierbei treten verschiedene Probleme auf:

- Laborexperimente sind oft aufwändig und teuer.
- Die Datenhaltung ist nicht immer einfach zu organisieren, sodass es zu Dateninkonsistenzen, z.B. durch Fehleingaben, kommen kann.
- Die Anzahl an Messwerten wird schnell unüberschaubar groß.

Aus diesen Gründen ist die pharmazeutische Industrie daran interessiert, einerseits nach Möglichkeit nur die aussagekräftigsten Versuche durchzuführen und andererseits die entstehenden Daten von Fehlern bzw. Ausreißern zu befreien. Hier bietet sich ein Ansatzpunkt für statistische Verfahren.
Es ist möglich, Ähnlichkeiten zwischen Messreihen aufzudecken und damit gegebenenfalls

Experimente einzusparen, die keine wesentlich neuen Informationen gegenüber vorangegangenen Versuchen liefern. Diese Ähnlichkeiten werden mathematisch mit der Größe der Korrelation zwischen Variablen beschrieben, eine Messreihe bzw. ein Experiment wird also auf eine Zufallsvariable abgebildet.
Ferner lässt sich auch die Eigenschaft Ausreißer einzelner Messpunkte (chemischer Strukturen, die in einer Messung untersucht worden sind) statistisch präzisieren, und so geartete Daten können erkannt werden. Dies hat zum einen den Effekt, dass der mit der Datenauswertung beschäftigte Wissenschaftler auf eventuelle Messfehler aufmerksam wird (unerwünschte Ausreißer), und zum anderen können gegebenenfalls Strukturen mit besonders günstigen Eigenschaften, die für die Medikamentenentwicklung besonders geeignet sind, schneller aus der großen erfassten Datenmenge herausgefiltert werden (gewünschte Ausreißer).
Eine andere Vorgehensweise besteht darin, die Ursachen der biologischen Eigenschaften einer Substanz in ihren physikochemischen Eigenschaften wie zum Beispiel der Anordnung der Atome und den Winkeln zwischen den Brückenbindungen in den Molekülen zu suchen. Diese physikochemischen Eigenschaften lassen sich in der Regel viel leichter ermitteln und sind für viele relevante Substanzen häufig schon in großer Menge bekannt. Wenn es nun gelingt, aus den physikochemischen Eigenschaften die biologischen Eigenschaften vorherzusagen, lassen sich im experimentellen Bereich zum Teil erhebliche Ressourcen einsparen. Auch hierfür bieten sich Verfahren der mathematischen Statistik an, im Wesentlichen sind dies Klassifikationsalgorithmen.
Die beschriebene Aufgabenstellung, versteckte Informationen in erfassten Daten aufzudecken, wird auch als Data Mining bezeichnet. Im Rahmen des GALA-Projekts werden verschiedene statistische Verfahren des Data Mining eingesetzt, um Daten aus der pharmazeutischen Forschung zu analysieren und auszuwerten. In den folgenden sieben Abschnitten werden die theoretischen Hintergründe dieser Verfahren erläutert, während in Teil IV auf die Implementierung eingegangen wird.

2 Korrelationsberechnung

Wie bereits in den einleitenden Bemerkungen angeführt, ist es oftmals von Interesse, Ähnlichkeiten bzw. Abhängigkeiten von Messreihen zu erkennen und auszunutzen. Eine Möglichkeit dazu besteht darin, die beobachteten Werte als zweidimensionale Streubilder darzustellen und die Formen der sich ergebenden Punktewolken zu analysieren. Dies ist bei der Analyse von einigen wenigen Messreihen ein durchaus gängiges und plausibles Vorgehen. Liegen jedoch wie im GALA-Projekt sehr viele Messreihen vor, so wird die Anzahl der zu betrachtenden Streubilder (englisch: „scatter plots“) schnell unüberschaubar groß. Ist nämlich die Anzahl zu analysierender Messreihen gleich n, so ergeben sich $\frac{n\cdot(n-1)}{2}$ Streubilder, ihre Anzahl wächst also quadratisch mit der Anzahl der Messreihen.
Eine effizientere Form der Analyse großer Datenmengen besteht darin, die Messreihen auf

Zufallsvariablen abzubilden und statistische Kenngrößen der sich ergebenden Größen zu berechnen und systematisch zu analysieren. In der gegebenen Problemstellung des Messens von Ähnlichkeiten bzw. Abhängigkeiten von Zufallsgrößen (und damit der mit ihnen identifizierten Messreihen) ist insbesondere die Berechnung von Kovarianzen und Korrelationen ein geeignetes Hilfsmittel.

2.1 Theoretische Beschreibung

Für zwei gegebene Zufallsvariablen X und Y ist die Kovarianz definiert als

$$\text{Kov}(X,Y) = \mathbb{E}((X - \mathbb{E}(X)) \cdot (Y - \mathbb{E}(Y))) \quad . \tag{1}$$

Sind die Verteilungen der zu Grunde liegenden Zufallsvariablen unbekannt und liegen lediglich Realisierungen $x_1, \ldots, x_n$ von X sowie $y_1, \ldots, y_n$ von Y vor, so verwendet man die empirische Kovarianz

$$\text{Kov}_n(X,Y) = \frac{1}{n-1} \sum_{i=1}^{n} ((x_i - \bar{x}_n) \cdot (y_i - \bar{y}_n))$$

als Schätzung der Kovarianz. Hierbei bezeichnen $\bar{x}_n$ bzw. $\bar{y}_n$ die arithmetischen Mittel der beobachteten Werte der Zufallsvariablen X bzw. Y. Demnach gelten

$$\bar{x}_n = \frac{1}{n} \sum_{i=1}^{n} x_i \quad \text{und} \quad \bar{y}_n = \frac{1}{n} \sum_{i=1}^{n} y_i \quad .$$

Normiert man nun diese Kovarianzen auf das Intervall $[-1;1]$ bzw. berechnet man die Kovarianzen von auf Varianz 1 normierten Zufallsvariablen, so erhält man den Korrelationskoeffizienten nach Pearson (auch Produktmomentkorrelation genannt)

$$\rho(X,Y) = \frac{\text{Kov}(X,Y)}{\sqrt{\text{Var}(X)} \cdot \sqrt{\text{Var}(Y)}} \tag{2}$$

bzw. den empirischen Korrelationskoeffizienten nach Pearson

$$\rho_n(X,Y) = \frac{\text{Kov}_n(X,Y)}{\sqrt{\text{Var}_n(X)} \cdot \sqrt{\text{Var}_n(Y)}}$$

von X und Y. $\text{Var}(X)$ bzw. $\text{Var}_n(X)$ bezeichnen hierbei die Varianz bzw. die empirische Varianz der Zufallsvariable X, d.h.

$$\text{Var}(X) \quad = \quad \mathbb{E}((X - \mathbb{E}(X))^2) \quad \equiv \quad \text{Kov}(X,X) \quad ,$$

$$\text{Var}_n(X) \quad = \quad \tfrac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x}_n)^2 \quad \equiv \quad \text{Kov}_n(X,X) \quad .$$

Der so definierte Pearsonsche Korrelationskoeffizient $\rho(X, Y)$, der im Folgenden auch kurz als die Korrelation zwischen X und Y bezeichnet wird, ist ein Maß für den linearen Zusammenhang zwischen X und Y. $|\rho(X, Y)| = 1$ bedeutet, dass eine eindeutige lineare Beziehung zwischen den beiden Zufallsvariablen X und Y besteht; analog lässt ein betragsmäßig großer Wert von $\rho_n(X, Y)$ auf eine lineare Abhängigkeit der beiden erfassten Stichproben schließen. Betragsmäßig kleine Werte der (empirischen) Korrelation hingegen deuten darauf hin, dass keine lineare Beziehung der betrachteten Zufallsgrößen bzw. deren Realisierungen zueinander besteht. Dies schließt jedoch nicht aus, dass ein nichtlinearer Zusammenhang zwischen X und Y vorliegt.

2.2 Anwendung

Innerhalb des GALA-Projektes werden die Experimente bzw. Laboruntersuchungen und ihre Ergebnisse als Messreihen interpretiert und durch Zufallsvariablen repräsentiert. Die Realisierungen dieser Größen ergeben sich durch die verschiedenen chemischen Substanzen, die in den Experimenten eingesetzt werden. So ergibt sich als Eingabedatensatz eine $(n \times m)$-Datenmatrix. Hierbei ist m die Anzahl der durchgeführten Experimente (Variablen, Merkmale, Attribute, properties), und n bezeichnet die Anzahl der eingesetzten Substanzen (chemische Strukturen). Es wurden nun paarweise die Korrelationen der Spalten dieser Eingabematrix berechnet und betragsmäßig signifikant große Werte in gesonderten Tabellen ausgegeben. Probleme ergaben sich dabei hauptsächlich durch Inkonsistenzen in den Eingabedatensätzen (oft werden nicht alle Experimente für alle Strukturen durchgeführt), die zu fehlenden Werten (Missing Values) führen. Ist die Anzahl dieser Missing Values groß, so ergibt sich unter Umständen eine dünn besetzte Eingabe-Datenmatrix. Praktisch wurde dieses Problem dadurch gelöst, dass die Korrelationsberechnung nur auf der Basis derjenigen Strukturen durchgeführt wurde, für welche beide jeweils betrachteten Experimente Werte lieferten.
Ein weiteres Problem besteht darin, den Terminus „signifikant groß“ mathematisch zu präzisieren. Ein gängiges statistisches Vorgehen dazu besteht darin, Verteilungsannahmen über die Variablen (im vorliegenden Fall durch die Messreihen induziert) zu treffen und, auf ihnen basierend, Quantile der zugehörigen Verteilungsfunktionen als Schwellenwerte für die Größe der berechneten Statistiken zu verwenden. Aufgrund der Datensicherheitsvorgaben liegt jedoch kein ausreichendes Detailwissen über die Daten und ihre Verteilungen vor, und so muss komplett frei von Verteilungsannahmen gearbeitet werden. Aus diesem Grund wurde die erstellte Software so ausgelegt, dass der Benutzer diese Schwellenwerte selbst eingeben und damit die Signifikanz der Korrelationen und in der Konsequenz auch die Größe der ausgegebenen Datensätze eigenhändig steuern kann. In der Anwendung hat sich dies als ein praktikables Vorgehen erwiesen, da bei Tests Ähnlichkeiten innerhalb der Daten erkannt wurden, die bereits im Vorfeld nachgewiesen waren.

3 Ausreißererkennung

3.1 Motivation

Neben der im letzten Abschnitt beschriebenen Aufgabe, Regelmäßigkeiten in den Daten aufzudecken, ist es oftmals ebenso von Interesse, Datenpunkte (Messwerte) zu erkennen, die signifikant aus dieser ermittelten Struktur herausfallen bzw. sich in der Punktewolke stark außerhalb des Gros' der gemessenen Daten befinden. Ein Datenpunkt mit dieser Eigenschaft wird im Folgenden als Ausreißer bezeichnet. Das Vorliegen von Ausreißern kann verschiedene Gründe haben:

1. Eventuell ist eine fehlerhafte Messung durchgeführt oder ein Messwert falsch eingegeben worden. Ein so gearteter Ausreißer ist sicherlich unerwünscht, und der Anwender möchte diese Dateninkonsistenz erkennen und gegebenenfalls beheben.
2. Wie bereits beschrieben, zielen die implementierten Data Mining Methoden darauf ab, chemische Substanzen mit besonders günstigen Eigenschaften systematisch aus großen Datenmengen herauszufiltern. Diese günstigen Eigenschaften lassen sich in experimentell ermittelten Daten als außergewöhnliche Werte und damit als erwünschte Ausreißer ablesen.

Ein weiterer Beweggrund dafür, Ausreißer aufzufinden und unter Umständen zu eliminieren (von der Analyse auszunehmen), soll durch die folgenden Abbildungen von Punktewolken mit den zugehörigen Korrelationswerten motiviert werden.

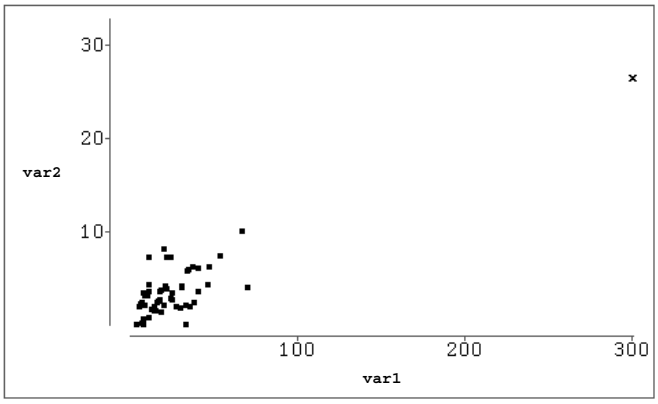


Abbildung 1: Ausreißer induziert signifikante Korrelation.

In dem in Abbildung 1 dargestellten Fall liefert die Korrelationsberechnung einen hohen Wert von $\rho_n(\mathrm{var1}, \mathrm{var2}) = 0,8703$, obwohl der überwiegende Teil der Beobachtungen eine unregelmäßige Struktur aufweist. Hier induziert also der in der Abbildung markierte, weit vom Datenzentrum entfernte Datenpunkt eine lineare Abhängigkeit, die jedoch kein

Charakteristikum der Stichprobe insgesamt darstellt.

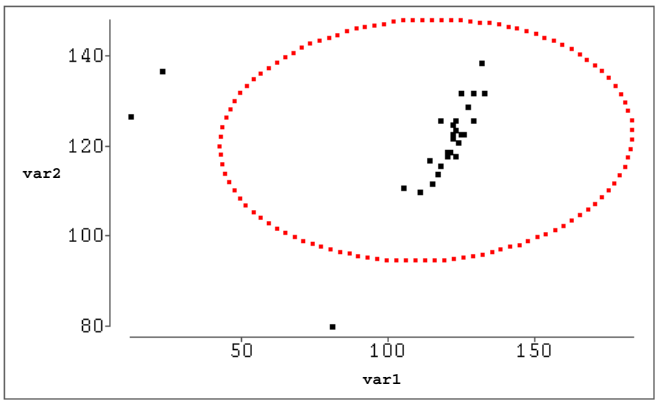


Abbildung 2: Ausreißer verdecken signifikante Korrelation (29 Wertepaare).

Der gegenteilige Fall ist in Abbildung 2 dargestellt. Das Gros der Daten zeigt eine lineare Beziehung zwischen Variable 1 und Variable 2. Die drei Datenpunkte, die außerhalb der eingezeichneten Ellipse liegen, verdecken jedoch diese Abhängigkeit und führen dazu, dass sich kein signifikant großer Wert für $\rho_{29}(\mathrm{var1}, \mathrm{var2})$ errechnet. Die empirische Korrelation ist in diesem Fall $\rho_{29}(\mathrm{var1}, \mathrm{var2}) = 0,0560$.
Nimmt man die drei Ausreißer von der Berechnung aus, so zeigt sich die lineare Beziehung des restlichen Datenmaterials eindeutig in der Kennzahl $\rho_{26}(\mathrm{var1}, \mathrm{var2}) = 0,8609$. Grafisch wird dies in Abbildung 3 deutlich.

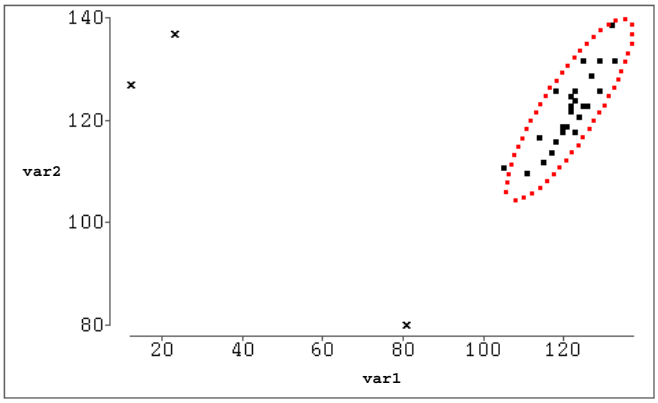


Abbildung 3: Aufgedeckte Korrelation durch Vernachlässigung von Ausreißern (26 Wertepaare).

Wie diese Beispiele zeigen, können einzelne Ausreißer zum Teil erheblichen Einfluss auf die Korrelationsberechnung haben. Dies liegt im Wesentlichen daran, dass in der Formel für $\rho_n(X, Y)$ jedem Datenpunkt das gleiche Gewicht zugeordnet wird und sich daher un-

verhältnismäßig große lokale Abweichungen vom Gesamtverhalten erheblich in der globalen Kenngröße der empirischen Korrelation niederschlagen.
Die Hauptschwierigkeit besteht nun darin, diese lokalen Inhomogenitäten mathematisch zu messen, um beurteilen zu können, wann diese als signifikant zu betrachten sind und nicht lediglich in der üblichen Schwankungsbreite der experimentellen Ungenauigkeit liegen. Außerdem muss dieses Vorgehen automatisiert werden, da nicht jedes Streubild einzeln ausgewertet werden kann.

3.2 Theoretische Beschreibung

Die zunächst naheliegendste Vorgehensweise zum Entdecken von Ausreißern besteht darin, für jeden Datenpunkt $z_i = (x_i, y_i), i = 1, \ldots, n$ der erhobenen zweidimensionalen Stichprobe dessen Euklidischen Abstand vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ zu ermitteln und als Abstandsmaß zu verwenden. Dabei wird der Euklidische Abstand $d(a, b)$ zweier Punkte $a = (a_1, a_2)$ und $b = (b_1, b_2) \in \mathbb{R}^2$ definiert als

$$d(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \quad .$$

Stammt $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n$ und $\bar{y}_n$, so misst

$$d(z_i, \bar{z}_n) = \sqrt{(x_i - \bar{x}_n)^2 + (y_i - \bar{y}_n)^2}$$

den Euklidischen Abstand des Beobachtungspaares $z_i = (x_i, y_i)$ vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$. Dass die Verwendung dieses Abstandsmaßes nicht immer sinnvoll ist, wird durch Abbildung 4 illustriert. Der markierte Datenpunkt hat zwar keinen großen Euklidischen Abstand vom Datenzentrum, fällt aber doch aus der Punktewolke heraus, da er nicht in der generellen Streuungsrichtung liegt.

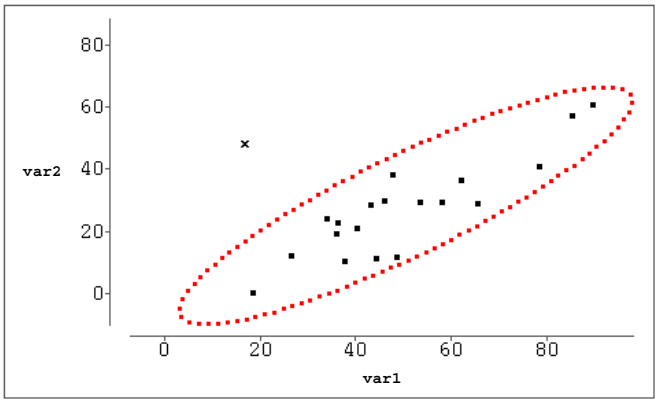


Abbildung 4: Ausreißer trotz geringen Euklidischen Abstandes.

Will man diese Zusatzinformation über die „Ausrichtung“ der Punktewolke mit in die Abstandsberechnung einfließen lassen, so bietet sich die Verwendung der sogenannten Mahalanobis-Distanz $\Delta(z_i, \bar{z}_n)$ an. Hierbei wird in die Formel für den Euklidischen Abstand eine Gewichtung mit der inversen Kovarianzmatrix eingefügt, um der unterschiedlichen Streuung in x- und y-Richtung Rechnung zu tragen. Die quadrierte Mahalanobis-Distanz eines Punktes $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n$ und $\bar{y}_n$ vom Datenzentrum $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ ist definiert als

$$\Delta^2(z_i, \bar{z}_n) = \begin{pmatrix} (x_i - \bar{x}_n) \\ (y_i - \bar{y}_n) \end{pmatrix}^\top \begin{pmatrix} \mathrm{Var}_n(X) & \mathrm{Kov}_n(X,Y) \\ \mathrm{Kov}_n(X,Y) & \mathrm{Var}_n(Y) \end{pmatrix}^{-1} \begin{pmatrix} (x_i - \bar{x}_n) \\ (y_i - \bar{y}_n) \end{pmatrix}. \quad (3)$$

Besonders einfach wird die Berechnung der Mahalanobis-Distanz, falls die zu Grunde liegenden Größen auf Mittelwert 0 und Varianz 1 standardisiert werden. Für die quadrierte Mahalanobis-Distanz eines Punktes $z_i = (x_i, y_i)$ aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n = 0$ und $\bar{y}_n = 0$ sowie $\mathrm{Var}_n(X) = \mathrm{Var}_n(Y) = 1$ vom Datenzentrum $\bar{z}_n = (0, 0)$ ergibt sich

$$\begin{aligned} \Delta^2(z_i, \bar{z}_n) &= \begin{pmatrix} x_i \\ y_i \end{pmatrix}^\top \begin{pmatrix} 1 & \rho_n(X,Y) \\ \rho_n(X,Y) & 1 \end{pmatrix}^{-1} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \\ &= \frac{1}{1 - \rho_n(X,Y)^2} \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix}^\top \begin{pmatrix} 1 & -\rho_n(X,Y) \\ -\rho_n(X,Y) & 1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \\ &= \frac{x_i^2 + y_i^2 - 2 \cdot x_i \cdot y_i \cdot \rho_n(X,Y)}{1 - \rho_n(X,Y)^2} \quad . \end{aligned} \quad (4)$$

Aufgrund dessen sollte man in der praktischen Berechnung die Größen X und Y zunächst standardisieren, damit (4) für $\Delta(z_i, \bar{z}_n)$ verwendet werden kann.
Geometrisch lässt sich der Übergang vom Euklidischen Abstand zur Mahalanobis-Distanz so interpretieren, dass das zu Grunde liegende Koordinatensystem im $\mathbb{R}^2$ derart neu gewählt wird, dass die erste Achse in Richtung der stärksten Streuung (repräsentiert über den Richtungsvektor v_1) und die zweite Achse in Richtung $v_2 \perp v_1$ weist. Zudem wird der Koordinatenursprung in den Punkt $\bar{z}_n = (\bar{x}_n, \bar{y}_n)$ verschoben. Überführt man nun den Zufallsvektor (X, Y) mittels einer linearen Transformation so in den Zufallsvektor (C_1, C_2), dass dies äquivalent zu einem Basiswechsel von der kanonischen in die oben charakterisierte Basis ist, so bezeichnet man die transformierten Zufallsgrößen C_1 und C_2 als die Hauptkomponenten der zu Grunde liegenden zweidimensionalen Verteilung. Die Mahalanobis-Distanz ist nun, wie im Folgenden gezeigt wird, der Euklidische Abstand in dem veränderten Koordinatensystem, wobei eine Skalierung der neuen Koordinatenachsen in Einheiten der Standardabweichung der Hauptkomponenten stattfindet.

Führt man diese Berechnung für standardisierte Größen durch, so ist dies äquivalent zur Lösung des Eigenwertproblems der Korrelationsmatrix

$$K_n := \begin{pmatrix} 1 & \rho_n(X,Y) \\ \rho_n(X,Y) & 1 \end{pmatrix} \quad ,$$

denn es gilt

$$\begin{aligned} v_1 &= \underset{a \in \mathbb{R}^2}{\operatorname{argmax}} \ \mathrm{Var}_n(a^\top \cdot (X,Y)^\top) \\ &= \underset{a \in \mathbb{R}^2}{\operatorname{argmax}} \ a^\top \cdot \begin{pmatrix} \mathrm{Var}_n(X) & \mathrm{Kov}_n(X,Y) \\ \mathrm{Kov}_n(X,Y) & \mathrm{Var}_n(Y) \end{pmatrix} \cdot a \end{aligned}$$

und (X,Y) wird hier als standardisiert vorausgesetzt, so dass die Varianz- / Kovarianzmatrix in die oben angegebene Korrelationsmatrix K_n übergeht.
Wie Ergebnisse aus der linearen Algebra (Maximierung von quadratischen Formen) zeigen, ergibt sich v_1 als Eigenvektor zum größeren Eigenwert λ_1 und v_2 als Eigenvektor zum kleineren Eigenwert λ_2 von K_n. Ferner ist die Varianz- / Kovarianzmatrix bezüglich des neuen Koordinatensystems gleich der diagonalisierten Korrelationsmatrix

$$K_D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \quad .$$

Anders ausgedrückt ist $\mathrm{Var}_n(C_1) = \lambda_1$ und $\mathrm{Var}_n(C_2) = \lambda_2$. Daraus ergibt sich, dass die oben angeführte Skalierung entlang der Hauptkomponenten dadurch zu geschehen hat, dass die Normierungsfaktoren $\frac{1}{\sqrt{\lambda_i}}, i = 1,2$ für die Richtungen $v_i, i = 1,2$ verwendet werden.
Die konkrete Berechnung des Euklidischen Abstandes in dem veränderten Koordinatensystem ergibt die geschlossene Formel (4). Bezeichne hierzu $\chi_{K_n}(\lambda)$ das charakteristische Polynom der standardisierten Korrelationsmatrix K_n, so ergibt sich

$$\chi_{K_n}(\lambda) = \det(K_n - \lambda E) = \begin{vmatrix} (1-\lambda) & \rho_n(X,Y) \\ \rho_n(X,Y) & (1-\lambda) \end{vmatrix} = (1-\lambda)^2 - \rho_n(X,Y)^2 \quad ,$$

wobei mit E die Einheitsmatrix der Dimension n bezeichnet wird. Bei der Berechnung der Eigenwerte erhält man

$$\chi_{K_n}(\lambda) = 0 \ \Leftrightarrow \ (1-\lambda)^2 = \rho_n(X,Y)^2 \ \Leftrightarrow \ |(1-\lambda)| = |\rho_n(X,Y)| \quad .$$

Daraus folgt

$$\lambda_1 = 1 + |\rho_n(X,Y)| \quad \wedge \quad \lambda_2 = 1 - |\rho_n(X,Y)| \quad \wedge \quad \lambda_1 \geq \lambda_2 \quad .$$

Für den ersten Eigenvektor ergibt sich daher

$$\begin{aligned}
& (K_n - \lambda_1 \cdot E) \cdot v_1 && = && 0 \\
\iff & \begin{pmatrix} -|\rho_n(X,Y)| & \rho_n(X,Y) \\ \rho_n(X,Y) & -|\rho_n(X,Y)| \end{pmatrix} \cdot v_1 && = && 0 \\
\iff & \begin{aligned} -|\rho_n(X,Y)|v_{11} + \rho_n(X,Y)v_{12} &= 0 \\ \rho_n(X,Y)v_{11} - |\rho_n(X,Y)|v_{12} &= 0 \end{aligned}
\end{aligned}$$

und man erhält

$$v_{11} = \text{sgn}(\rho_n(X,Y)) \cdot v_{12} \;\Rightarrow\; v_1 = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ \text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad .$$

Die analoge Berechnung des zweiten Eigenvektors liefert entsprechend

$$\begin{aligned}
& (K_n - \lambda_2 \cdot E) \cdot v_2 && = && 0 \\
\iff & \begin{pmatrix} |\rho_n(X,Y)| & \rho_n(X,Y) \\ \rho_n(X,Y) & |\rho_n(X,Y)| \end{pmatrix} \cdot v_2 && = && 0 \\
\iff & \begin{aligned} |\rho_n(X,Y)|v_{21} + \rho_n(X,Y)v_{22} &= 0 \\ \rho_n(X,Y)v_{21} + |\rho_n(X,Y)|v_{22} &= 0 \end{aligned}
\end{aligned}$$

$$\Rightarrow v_{22} = -\text{sgn}(\rho_n(X,Y)) \cdot v_{21} \;\Rightarrow\; v_2 = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad .$$

Somit hat die Matrix T der entsprechenden Basistransformation die Form

$$T = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 & 1 \\ \text{sgn}(\rho_n(X,Y)) & -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \quad ,$$

und die transformierten Koordinaten eines Vektors $(x_i, y_i)^\top \in \mathbb{R}^2$, die sich als $(x_i^*, y_i^*) = (x_i, y_i) \cdot T$ berechnen lassen, ergeben sich zu

$$\begin{aligned}
(x_i^*, y_i^*) = (x_i, y_i) \cdot T &= \frac{1}{\sqrt{2}} \cdot (x_i, y_i) \cdot \begin{pmatrix} 1 & 1 \\ \text{sgn}(\rho_n(X,Y)) & -\text{sgn}(\rho_n(X,Y)) \end{pmatrix} \\
&= \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} x_i + \text{sgn}(\rho_n(X,Y)) \cdot y_i \\ x_i - \text{sgn}(\rho_n(X,Y)) \cdot y_i \end{pmatrix}^\top \quad .
\end{aligned}$$

Diese Darstellung gestattet nun die Berechnung von $\Delta(z_i, \bar{z}_n)$ eines Punktes (x_i, y_i) aus einer Stichprobe zweier Merkmale X und Y vom Umfang n mit arithmetischen Mitteln $\bar{x}_n = 0$ und $\bar{y}_n = 0$ sowie $\mathrm{Var}_n(X) = \mathrm{Var}_n(Y) = 1$ vom Datenzentrum $(0, 0)$ als Euklidischen Abstand des transformierten Punktes vom Koordinatenursprung unter Beachtung der Normierung entlang der Hauptkomponenten:

$$
\begin{aligned}
\Delta^2(z_i, \bar{z}_n) &= \left(\frac{x_i + \mathrm{sgn}(\rho_n(X,Y)) \cdot y_i}{\sqrt{2}\sqrt{1 + |\rho_n(X,Y)|}} \right)^2 + \left(\frac{x_i - \mathrm{sgn}(\rho_n(X,Y)) \cdot y_i}{\sqrt{2}\sqrt{1 - |\rho_n(X,Y)|}} \right)^2 \\
&= \frac{x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2}{2(1 + |\rho_n(X,Y)|)} + \frac{x_i^2 - 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2}{2(1 - |\rho_n(X,Y)|)} \\
&= \frac{(x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2)(1 - |\rho_n(X,Y)|)}{2(1 - \rho_n(X,Y)^2)} \\
&\quad + \frac{(x_i^2 - 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i + y_i^2)(1 + |\rho_n(X,Y)|)}{2(1 - \rho_n(X,Y)^2)} \\
&= \frac{1}{2(1 - \rho_n(X,Y)^2)} \cdot [2x_i^2 + 2y_i^2 - |\rho_n(X,Y)| \cdot (x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i \\
&\qquad + y_i^2 - x_i^2 + 2\mathrm{sgn}(\rho_n(X,Y)) x_i y_i - y_i^2)] \\
&= \frac{2x_i^2 + 2y_i^2 - 4 \cdot |\rho_n(X,Y)| \cdot \mathrm{sgn}(\rho_n(X,Y)) \cdot x_i \cdot y_i}{2(1 - \rho_n(X,Y)^2)} \\
&= \frac{x_i^2 + y_i^2 - 2 \cdot x_i \cdot y_i \cdot \rho_n(X,Y)}{1 - \rho_n(X,Y)^2} \quad .
\end{aligned}
$$

Dieses Konzept der Zerlegung der Gesamtstreuung in orthogonale Streukomponenten wird in Abschnitt 6 auf Dimensionen $m \geq 3$ zur so genannten Hauptkomponentenanalyse erweitert.
Als Ausreißer werden Datenpunkte mit einer großen Mahalanobis-Distanz zu $(\bar{x}_n, \bar{y}_n)$ deklariert. In konkreten Anwendungen besteht dann die grundlegende Fragestellung darin, ab wann $\Delta^2(z_i, \bar{z}_n)$ als so groß angesehen werden soll, dass dem zugehörigen Punkt das Attribut Ausreißer zugeordnet wird. Dazu gibt es eine Reihe möglicher Ansätze. Der folgende Abschnitt beschreibt die Methoden, die im Projekt angewendet werden.

3.3 Anwendung

3.3.1 Naiver Ansatz

Der erste, naive Ansatz zur Festlegung einer kritischen Schranke Δ_{krit} für den Wert der Mahalanobis-Distanz bestand darin, besonders markante Streubilder wie etwa in Abbildung 1 oder Abbildung 2 zu betrachten. Hier ist es, wie bereits beschrieben, in jedem Fall wünschenswert, dass die vorhandenen Ausreißer automatisch erkannt werden, da durch ihr Vorhandensein weitergehende Berechnungen wie die der empirischen Korrelation empfindlich gestört werden. Aufgrund dessen wurden mehrere Beispiele, in denen dies der Fall ist, analysiert und festgestellt, wie Δ_{krit} jeweils zu wählen wäre, um diese Ausreißer ausfindig zu machen. Es ergab sich, dass in der Regel bei diesen augenscheinlichen Ausreißern eine Mahalanobis-Distanz von $\Delta \geq 2$, häufig sogar von $\Delta \geq 2,5$ vorlag.
Von diesen empirischen Ergebnissen ausgehend wurde daraufhin der konstante Wert $\Delta_{\text{krit}} = 2$ (für fast sicheres Erkennen) bzw. $\Delta_{\text{krit}} = 2,5$ als kritische Mahalanobis-Distanz vereinbart. Der Vorteil dieser Methode besteht darin, dass sie völlig frei von Verteilungsannahmen ist und daher zu ihrer Anwendbarkeit nichts über die Herkunft des zu Grunde liegenden Datenmaterials bekannt sein muss. Für zuerst ausschließlich betrachtete kleine Stichprobenumfänge wie z.B. $n = 30$ lieferte dieses Vorgehen gute Ergebnisse. Im Falle von Messreihen mit erheblich größerem Stichprobenumfang zeigte sich allerdings der große Nachteil des Verfahrens: Mit wachsendem n wurde der Anteil der so bestimmten Ausreißer immer größer und führte dazu, dass die erzeugte Information für den Analysten nicht mehr überschaubar blieb. Dies lässt sich damit begründen, dass bei kleinen Stichprobenumfängen die Kenngrößen arithmetisches Mittel und empirische Varianz durch einzelne Ausreißer stark beeinflusst werden. Dies führt dazu, dass die Mahalanobis-Distanz dieser Ausreißer zu dem berechneten Datenzentrum nur eine bestimmte Maximalgröße annehmen kann und der kritische Wert Δ_{krit} daher nicht zu groß gewählt werden darf. Liegt hingegen ein großer Stichprobenumfang n vor, so nimmt der Einfluss einzelner Beobachtungen auf die Kenngrößen ab und deswegen wird bei Beibehaltung der ursprünglichen kritischen Schranke ein wesentlich größerer Anteil der Beobachtungen als Ausreißer deklariert. Nachteilig daran ist vor allem, dass viele der so deklarierten Beobachtungen gar keine Ausreißer im ursprünglichen Sinne sind. Es wäre also wünschenswert, dass auch Δ_{krit} mit wachsendem n größer wird, damit nicht etwa bei z.B. $n = 1000$ fast 50 % der Datenpunkte als Ausreißer aufgeführt werden. Dies hätte im Falle der Beibehaltung des naiven Ansatzes jedoch zur Folge, dass für jede Analyse eine neue Konstante gewählt werden müsste und dass diese überdies einen Kausalbezug zu n haben sollte.
Es ist daher naheliegend, Δ_{krit} nicht als konstant, sondern als Funktion des Stichprobenumfangs n anzusetzen. Im Folgenden werden nun zwei Verfahren beschrieben, die einen solchen Bezug zwischen n und Δ_{krit} herstellen.

3.3.2 Verwendung asymptotischer Verteilungsquantile

Wie in Abschnitt 3.2 hergeleitet, misst die Mahalanobis-Distanz den Euklidischen Abstand standardisierter Zufallsgrößen in einem speziellen Koordinatensystem. Bezeichne also $C = (C_1, C_2)$ die Hauptkomponenten und $\mu = (\mu_1, \mu_2)$ den zweidimensionalen Erwartungswert der zu Grunde liegenden Verteilung, so gilt:

$$\Delta^2(C, \mu) = \frac{(C_1 - \mu_1)^2}{\text{Var}(C_1)} + \frac{(C_2 - \mu_2)^2}{\text{Var}(C_2)} \quad . \tag{5}$$

Bei (5) handelt es sich um eine Summe von Quadraten zweier standardisierter Zufallsgrößen. Unterliegen diese Zufallsgrößen nun einer Normalverteilung, so ist die Verteilung von $\Delta^2(C, \mu)$ bekannt als χ^2-Verteilung mit zwei Freiheitsgraden oder kurz χ^2_2-Verteilung. Im Falle der entsprechenden empirischen Größen gilt dies zwar nur asymptotisch, aber es ist dennoch möglich, Quantile der χ^2_2-Verteilung für Δ_{krit} zu verwenden, wenn man eine normalverteilte Grundgesamtheit unterstellt.
Praktisch kann damit z.B. das oben angedeutete Ziel, den mittleren Anteil an Ausreißern zu kontrollieren, verfolgt werden. Die Forderung lautet in mathematischer Notation

$$\mathbb{P}(\Delta > \Delta_{\text{krit}}) = \alpha(n) \stackrel{!}{=} \frac{\mu(n)}{n}$$

mit $\mu(n)$ als gewünschter Anzahl an Ausreißern bei Stichprobengröße n und kann durch die Wahl von $\Delta^2_{\text{krit}}(n) = \chi^2_{2;1-\alpha(n)}$ realisiert werden. $\chi^2_{2;1-\alpha(n)}$ bezeichnet hierbei das $(1 - \alpha(n))$-Quantil der χ^2-Verteilung mit zwei Freiheitsgraden.
In dem konkreten Anwendungsbeispiel der Analyse pharmazeutischer Daten wurde $\mu(n) = \sqrt{n/10}$ festgelegt, d.h. aus 1000 Datenpunkten sollten im Mittel etwa zehn Ausreißer herausgefunden werden. Es sei hier aber noch einmal darauf hingewiesen, dass in die Berechnung die Normalverteilungsannahme eingeht und die für $\Delta^2(z_i, \bar{z}_n)$ verwendete Verteilung nur asymptotisch gilt.

3.3.3 Erkennen von echten Ausreißern

Ein ambitionierteres Vorgehen besteht darin, dass der Forderung nachgegangen wird, nur „echte“ Ausreißer zu erkennen. Hier wird also nicht versucht, die Anzahl der sich ergebenden Ausreißer zu steuern, sondern es wird eine Signifikanzschranke für die Verlässlichkeit der Ausreißerklassifikation gesucht. Präziser formuliert lautet die Forderung, dass mit Wahrscheinlichkeit $(1 - \alpha)$ keine Ausreißer erkannt werden sollen, wenn keine echten Ausreißer vorliegen. Um dies bei gegebenem Signifikanzniveau α zu gewährleisten, wählt man Δ_{krit} derart, dass $\mathbb{P}(\Delta_{\max} > \Delta_{\text{krit}}) \stackrel{!}{=} \alpha$ erfüllt ist. Hierbei bezeichnet $\Delta_{\max}$ die größte in der Stichprobe auftretende Mahalanobis-Distanz. Liegt also eine homogene Stichprobe ohne echte Ausreißer vor, so wird der Messwert mit der größten Mahalanobis-Distanz auch

nur in $\alpha \cdot 100\%$ der Fälle als Ausreißer deklariert.
Schwierigkeiten bereitet hierbei die konkrete Berechnung von $\Delta_{\text{krit}}(n, \alpha)$ bei gegebenem Stichprobenumfang n und vorgegebenem Signifikanzniveau α. In [1] ist diesem Problem nachgegangen worden, und es finden sich auf den Seiten 516f. Tabellen für $\Delta_{\text{krit}}(n, \alpha)$ mit n im Bereich von 3 bis 1000 und $\alpha = 0,01$ sowie $\alpha = 0,05$. Diese Tabellenwerte wurden in der erfolgten Implementierung übernommen, und für $1000 < n \leq 4000$ wurde die Tabelle durch Simulationsrechnungen erweitert. Ab $n > 4000$ wurde dann zur asymptotischen Verteilung, die sich als χ^2_2-Verteilung zur n-ten Potenz ergibt, übergegangen (siehe hierzu auch Abschnitt 6.4.2 mit $p = 2$).

3.3.4 Mehrstufiges Vorgehen

Wie bereits in den einleitenden Bemerkungen zu diesem Abschnitt beschrieben, kann sich das Vorhandensein von Ausreißern verfälschend auf die berechneten Werte von $\rho_n(X, Y)$ auswirken. Des Weiteren kann es dazu kommen, dass Ausreißer mit sehr großen Mahalanobis-Distanzen solche Ausreißer überdecken, für die Δ zwar im Verhältnis zu den erkannten Ausreißern klein ist, die aber dennoch erkannt werden sollten. Als Beispiel dafür sei das in Abbildung 5 dargestellte Streubild von Variable 3 und Variable 4 angeführt.

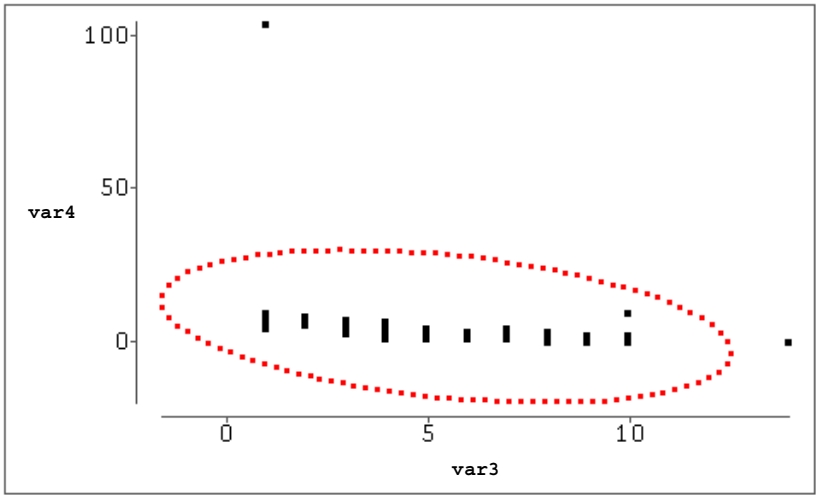


Abbildung 5: Variable 3 gegen Variable 4 (alle 109 Beobachtungen).

Es sind zwei Ausreißer auszumachen, die außerhalb der eingezeichneten Konfidenzellipse zur Mahalanobis-Distanz $\Delta = 2,5$ liegen. Der Rest des Datenmaterials weist optisch keine weiteren Ausreißer auf. Nimmt man die zwei erkannten Datenpunkte nun von der Berechnung aus, d.h. eliminiert man die zwei erkannten Ausreißer, so ergibt sich für die verbleibenden 107 Datenpaare das Streubild in Abbildung 6.

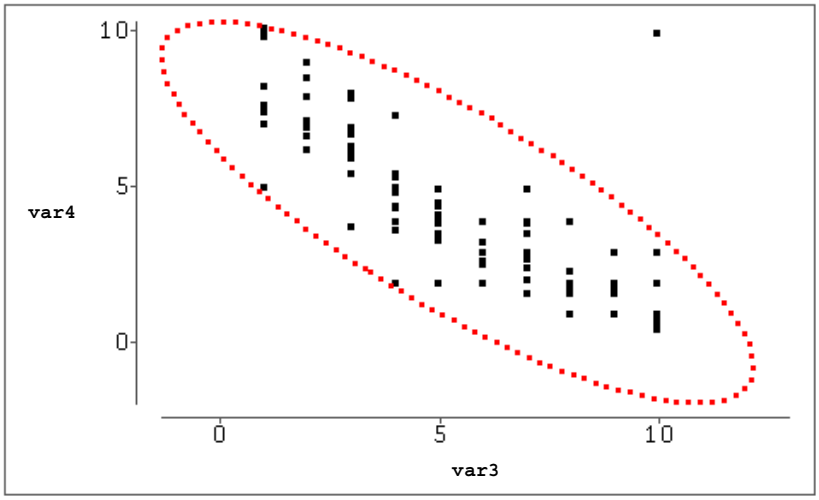


Abbildung 6: Variable 3 gegen Variable 4 (restliche 107 Beobachtungen).

Aufgrund der veränderten Skalierung des Plots klar zu erkennen, zeigt sich nun ein weiterer Ausreißer, für den beide Variablen etwa den Wert 10 annehmen. Dieser Ausreißer wäre unerkannt geblieben, falls die Ausreißererkennung für diese Variablenkombination nur einmalig durchgeführt worden wäre.
Deswegen wurde bei der Umsetzung der Projektziele ein iteratives Vorgehen der Ausreißerdetektion und -elimination umgesetzt. Nachdem auf der Basis aller vorliegender Datenpaare $z_i = (x_i, y_i), i = 1, \ldots, n$ die empirische Korrelation $\rho_n(X, Y)$ berechnet wurde, werden die Mahalanobis-Distanzen $\Delta(z_i, \bar{z}_n)$ dieser Messpunkte vom Datenzentrum bestimmt und diejenigen Datenpunkte $z_i^* = (x_i^*, y_i^*)$, bei welchen $\Delta(z_i^*, \bar{z}_n)$ eine signifikante Größe hat, gekennzeichnet. Danach wird eine Neuberechnung der empirischen Korrelation durchgeführt, wobei die Paare (x_i^*, y_i^*) nicht mehr berücksichtigt werden. Dies kann als eine Robustifizierung der Berechnung von $\rho_n(X, Y)$ angesehen werden. Ferner wird in jeder Iteration mit den neu berechneten Mahalanobis-Distanzen die Ausreißerbestimmung fortgesetzt, um gegebenenfalls noch unerkannte Ausreißer ermitteln zu können.

3.4 Ergebnisse

In Tabelle 1 sollen die Ergebnisse der drei verwendeten Methoden anhand eines Beispieldatensatzes quantitativ gegenüber gestellt werden. Dieser Beispieldatensatz verfügt über 1017 Merkmale mit 11363 Beobachtungen. Die Ergebnisse illustrieren die Vorteile der dritten durchgeführten Methode. Die absolute Anzahl an Ausreißern wird drastisch reduziert, und dabei wird insbesondere der Effekt erzielt, dass nur noch wirklich auffällige Beobachtungen in Erscheinung treten. Dies ist daran erkennbar, dass der Prozentsatz der vier Beobachtungen, die in allen Variablenkombinationen insgesamt am häufigsten als Ausreißer auftreten, also wirklich aus dem Rahmen fallen, im Falle der Anwendung der

	Ausreißer	davon 4 häufigste	Korrelationen $\geq 0,8$
$\Delta_{\text{krit}} = 2,0$	≈ 600.000	5 %	≈ 20.000
$\Delta_{\text{krit}} = 2,5$	≈ 180.000	5 %	≈ 11.000
$\Delta_{\text{krit}}(n) = \chi^2_{2;1-\frac{1}{\sqrt{10n}}}$	≈ 180.000	9 %	≈ 17.500
„Echt“ mit $\alpha = 0,05$	≈ 70.000	11 %	≈ 17.000
„Echt“ mit $\alpha = 0,01$	≈ 40.000	16 %	≈ 16.000

Tabelle 1: Vergleich der Methoden zur Ausreißerbehandlung.

Methoden zur Erkennung „echter“ Ausreißer stark ansteigt; es erfolgt also eine Konzentration auf die auffälligsten Datenpunkte. Ferner wird der Wert von ρ_n und damit die Anzahl der als signifikant betrachteten Korrelationen durch die Wahl der so gearteten Verfahren kaum beeinflusst. Es gibt allerdings einzelne Fälle, in denen durch diese Art der Ausreißerbehandlung signifikante Korrelationen verloren gehen. In Abschnitt 5 wird mit der Rangkorrelation nach Spearman ein Konzept vorgestellt, mit dem dies verhindert werden kann.

4 Transformation von Variablen

4.1 Motivation

Wie bereits in Abschnitt 2 erläutert, ist der (empirische) Korrelationskoeffizient nach Pearson nur ein Maß für lineare Abhängigkeiten zwischen Zufallsvariablen bzw. Messreihen. Allgemein lassen sich nichtlineare Abhängigkeiten in lineare Abhängigkeiten überführen, indem eine oder gegebenenfalls auch beide zu Grunde liegenden Zufallsgrößen geeigneten Transformationen unterzogen wird bzw. werden. In der Regel ist es jedoch keineswegs offensichtlich, wie diese linearisierenden Transformationen zu wählen sind, denn der Typ der vorliegenden Beziehung geht maßgeblich in die Wahl der Transformationen ein.
Es sei an dieser Stelle noch darauf hingewiesen, dass das Konzept der Variablentransformation auch anderen Absichten dienen kann. Wie unter anderem in [8] ausführlich beschrieben, kann einer eindimensionalen Verteilung durch geeignete Transformation die Schiefe genommen oder durch Maßstabsänderung (z.B. Logarithmierung) eine bessere Auflösung bzw. Darstellbarkeit gemessener Daten erreicht werden. Da dies aber nicht zu den Hauptzielen des Projektes zählt, soll hierauf an dieser Stelle nicht näher eingegangen werden.
In den im nachfolgenden Abschnitt beschriebenen Fallbeispielen aus der pharmazeutischen Forschung war eine Identifikation des zu wählenden Transformationstyps aufgrund von Expertenwissen bzw. physikalischen Grundzusammenhängen möglich, und es konnte erreicht werden, dass signifikant große Korrelationen nach getätigten Transformationen berechnet wurden, falls Abhängigkeiten des angenommenen Typs vorlagen.

4.2 Anwendung

Neben der offensichtlichen Anwendung der Transformation von Volumendaten durch Übergang zur dritten Wurzel ist auch im Falle von Konzentrationsdaten das Konzept der Transformation eingesetzt worden, um die Abhängigkeit von Variablen zu linearisieren. Hierzu sind Vorüberlegungen aus dem Bereich der Reaktionskinetik erforderlich. Es werden nur die notwendigen Ergebnisse wiedergegeben; die zu Grunde liegenden biologischen Zusammenhänge werden zum Beispiel genauer in [17] und [4] erläutert.
Betrachtet wird die reversible Bindung von Substanzen an Rezeptormoleküle. Wird einem Organismus eine chemische Substanz verabreicht, so kann es dazu kommen, dass diese an sogenannte Rezeptormoleküle im Organismus bindet und die beiden Moleküle reversibel zu einem neuen Molekül reagieren. Es ist nun möglich, die Konzentrationen der an diesen Reaktionsvorgängen beteiligten Ausgangsmoleküle mit der der resultierenden Moleküle mathematisch in Beziehung zu setzen. Dies geschieht im Wesentlichen durch die Einführung der sogenannten Reaktionskonstanten k_1 und k_{-1}. Das in diesem Fall benutzte Modell entstammt dem Massenwirkungsgesetz. Es besagt, dass die Reaktionsvorgänge proportional von den Ausgangskonzentrationen und die inversen Reaktionsvorgänge (Substanz und Rezeptor lösen sich wieder voneinander) proportional von der Konzentration reagierter (verbundener) Moleküle abhängen, wobei die Proportionalitätskonstanten gerade die oben angeführten Zahlen k_1 bzw. k_{-1} sind.
Bezeichnet S die verabreichte Substanz und R den zugehörigen Rezeptor, so kann der beschriebene Reaktionsvorgang wie folgt symbolisiert werden:

$$a \cdot S + R \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} SR \quad .$$

Mit den Bezeichnungen

$C_R(t)$	Konzentration freier Rezeptoren zum Zeitpunkt t,
$C_S(t)$	Konzentration freier Substanzmoleküle zum Zeitpunkt t,
$C_{SR}(t)$	Konzentration blockierter Rezeptoren (verbundener Moleküle) zum Zeitpunkt t und
$C_{R,\text{ges}} = C_R(t) + C_{SR}(t)$	gesamte Rezeptorkonzentration

ergibt sich unter Verwendung des Massenwirkungsgesetzes für die zeitliche Änderung von $C_{SR}(t)$ die folgende Differentialgleichung:

$$\dot{C}_{SR}(t) = k_1 \cdot C_S(t)^a \cdot C_R(t) - k_{-1} C_{SR}(t) \quad .$$

Geht man nun davon aus, dass sich diese Differentialgleichung für $t \to \infty$ stabilisiert und in einen stationären Endzustand mit $\dot{C}_{SR}(t) \equiv 0$ übergeht, so gilt in diesem eingeschwun-

genen Zustand:

$$\begin{aligned} 0 &= k_1 C_S^a \cdot C_R - k_{-1} \cdot C_{SR} \\ &= k_1 C_S^a \cdot (C_{R,\text{ges}} - C_{SR}) - k_{-1} C_{SR} \quad . \end{aligned}$$

Auf das Argument $t \to \infty$ wurde hier aus Gründen der Notationsvereinfachung bei den Konzentrationswerten verzichtet, da diese im eingeschwungenen Zustand konstant sind. Führt man nun noch die Größe

$$\tilde{C}_{SR} := \frac{C_{SR}}{C_{R,\text{ges}}} \quad \hat{=} \quad \text{Anteil blockierter Rezeptoren}$$

ein, so vereinfacht sich die stationäre Gleichung weiter, indem man beide Seiten durch $C_{R,\text{ges}}$ dividiert:

$$\begin{aligned} & k_1 C_S^a \cdot (1 - \tilde{C}_{SR}) - k_{-1} \cdot \tilde{C}_{SR} = 0 \\ \iff \quad & \tilde{C}_{SR} \cdot (k_{-1} + k_1 C_S^a) = k_1 C_S^a \\ \iff \quad & \tilde{C}_{SR} = \frac{k_1 C_S^a}{k_{-1} + k_1 C_S^a} = \frac{C_S^a}{\tilde{k} + C_S^a} \quad , \end{aligned}$$

wobei $\tilde{k} := \frac{k_{-1}}{k_1}$. Hieraus ergibt sich, dass $\tilde{k}^{\frac{1}{a}}$ genau die Substanzkonzentration ist, bei der die Hälfte der Rezeptoren blockiert wird.
In den Laborexperimenten der pharmazeutischen Forschung werden nun häufig folgende Messwerte ermittelt:

- X Anteil der Rezeptoren, die bei gegebener Konzentration C_0 der Substanz blockiert werden,
- Y Substanzkonzentration, bei der die Hälfte der Rezeptoren blockiert wird (k_i-Wert).

Aus der hergeleiteten Beziehung zwischen $\tilde{C}_{SR}$ und dem zugehörigen k_i-Wert kann nun ein Modellierungsansatz für den Zusammenhang von X und Y abgeleitet werden:

$$\begin{aligned} X &= \frac{C_0^a}{Y^a + C_0^a} \\ \iff \quad X \cdot (Y^a + C_0^a) &= C_0^a \\ \iff \quad X \cdot Y^a &= C_0^a (1 - X) \\ \iff \quad Y^a &= C_0^a \cdot \tfrac{1-X}{X} \\ \iff \quad a \cdot \ln(Y) &= a \cdot \ln(C_0) + \ln\left(\tfrac{1-X}{X}\right) \\ \implies \quad \text{logit}(X) &= a \cdot (\ln(Y) - \ln(C_0)) \quad . \end{aligned}$$

Um eine Linearisierung der Abhängigkeit zwischen X und Y zu erreichen, wurde also in den entsprechenden Fällen die Variable Y einer Transformation mit der natürlichen Logarithmusfunktion und die Variable X der sogenannten Logit-Transformation

$$\text{logit}(X) = \ln\left(\frac{1-X}{X}\right)$$

unterzogen.

5 Rangkorrelation

5.1 Motivation

Im vorherigen Abschnitt wurde die Möglichkeit vorgestellt, mit Hilfe geeigneter Transformationen eine nichtlineare Abhängigkeit zwischen zwei Zufallsgrößen X und Y zu linearisieren, sodass diese mit Hilfe der Produktmomentkorrelation aufgedeckt werden kann. Das Hauptproblem bei dieser Vorgehensweise besteht jedoch darin, dass die Wahl der Transformation(en) von der zu Grunde liegenden Beziehung zwischen X und Y abhängt. Häufig ist jedoch Detailwissen über die Herkunft des Datenmaterials nicht bekannt. Damit kann kein Modell für den Zusammenhang zwischen X und Y hergeleitet werden und mögliche linearisierende Transformationen bleiben unentdeckt.
Es wäre also wünschenswert, eine universelle Transformation zur Verfügung zu haben, die möglichst viele Abhängigkeiten linearisiert. Im Bereich der nichtparametrischen Statistik hat sich das Konzept der Rangkorrelation nach Spearman etabliert, mit welchem sich zumindest alle monotonen Abhängigkeiten identifizieren lassen. Das Vorgehen besteht darin, dass nicht der Pearsonsche Korrelationskoeffizient der Originaldaten, sondern die Produktmomentkorrelation der zugehörigen Positionen der Messwerte in der jeweiligen geordneten Stichprobe (der Ränge der Beobachtungen zu X und Y) berechnet wird. Damit spielt also der absolute Abstand der gemessenen Werte keine Rolle, sondern lediglich die Ordnung der Messwerte geht in die Berechnung ein. Aufgrund dessen liefert die Spearmansche Rangkorrelation für alle Variablenkombinationen (X, Y), in denen X über eine monotone (ggfs. auch nichtlineare) Transformation in Y überführt werden kann, betragsmäßig den Wert 1. Um dies zu illustrieren, wurden drei zweidimensionale Stichproben simuliert. Abbildung 7 liegt das Modell eines quadratischen Zusammenhangs zwischen X und Y mit einem standardnormalverteilten ($\mathcal{N}(0,1)$) Fehlerterm zu Grunde. Hier fällt in dem betrachteten Intervall $[1, 10]$ für X die Nichtlinearität der gegebenen Abhängigkeit bei der Berechnung von $\rho_n(X, Y)$ nicht besonders ins Gewicht, es ergibt sich der hohe Wert $\rho_n(X, Y) = 0,9723$.
Geht man nun zu einem kubischen Zusammenhang $Y = X^3 + \varepsilon, \varepsilon \sim \mathcal{N}(0,1)$ über, so ergibt sich das Streubild in Abbildung 8. Aufgrund der stärkeren Abweichung vom linearen Zusammenhang sinkt der Wert der Produktmomentkorrelation auf $\rho_n(X, Y) = 0,9280$. In Abbildung 9 ist nun schließlich die Beziehung $Y = \exp(X) + \varepsilon, \varepsilon \sim \mathcal{N}(0,1)$ modelliert

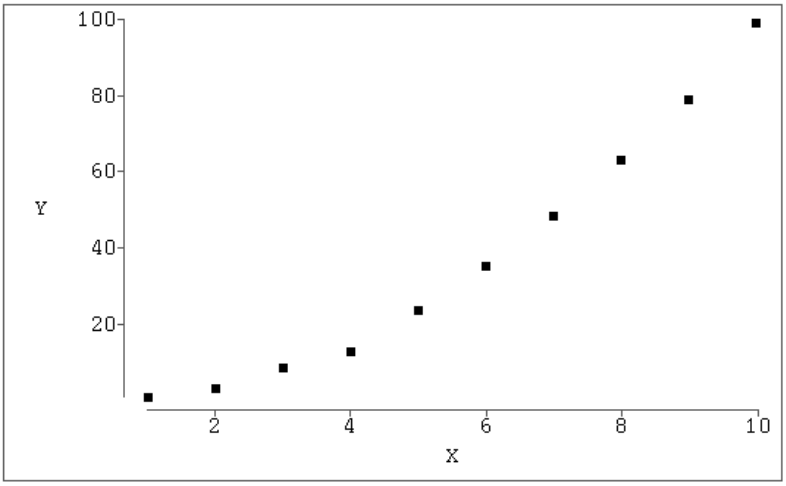


Abbildung 7: Quadratischer Zusammenhang zwischen zwei Variablen.

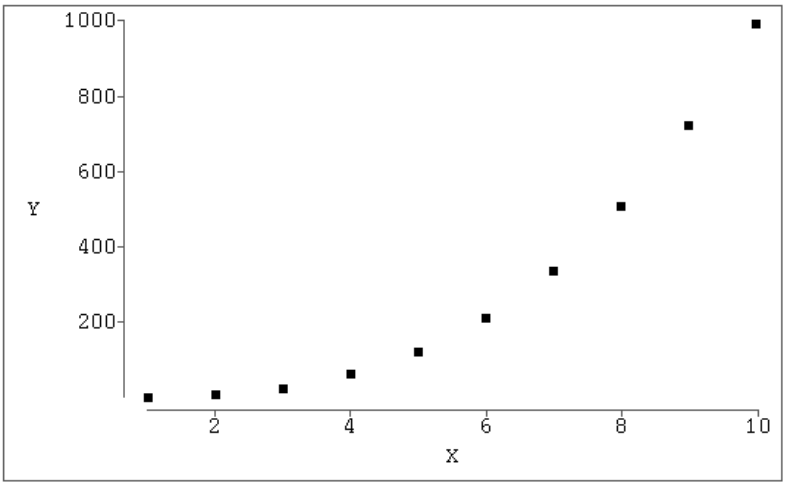


Abbildung 8: Kubischer Zusammenhang zwischen zwei Variablen.

worden. Diese Abhängigkeit ist von $\rho_n(X, Y)$ nur noch schlecht zu erfassen, der berechnete Wert ergibt sich zu $\rho_n(X, Y) = 0,7169$. Sicherlich wäre es jedoch von Interesse, in allen drei Fällen auf die Abhängigkeit zwischen X und Y aufmerksam zu werden. Setzt man nun, wie oben beschrieben, anstelle der tatsächlichen Simulationswerte die zugehörigen Ränge ein, ergibt sich in allen drei Fällen ein identisches Streubild (siehe Abbildung 10): Die Punktewolke entspricht der ersten Winkelhalbierenden, da mit wachsenden Werten für X auch in Y-Richtung immer größere Messwerte beobachtet wurden. Dabei spielt es keine Rolle, in welcher Form das Wachstum in Y-Richtung mit dem in X-Richtung gekoppelt ist. Der Rangkorrelationskoeffizient nach Spearman, hier mit $\rho_n^S(X, Y)$ bezeichnet, errechnet

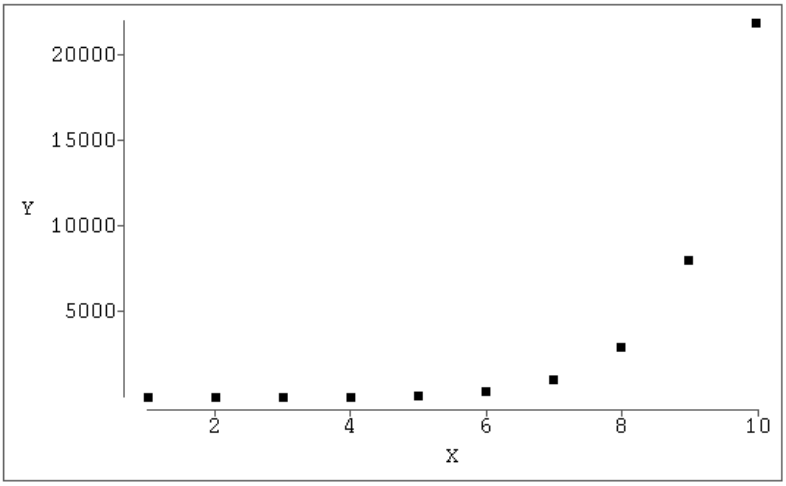


Abbildung 9: Exponentieller Zusammenhang zwischen zwei Variablen.

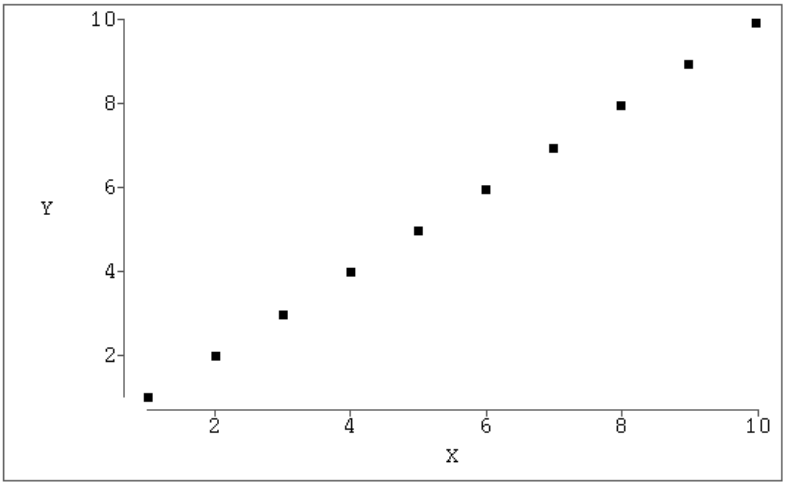


Abbildung 10: Streubild der Ränge zu Abbildungen 7 bis 9.

sich damit in allen drei Modellbeispielen zu $\rho_n^S(X,Y) = 1$.

5.2 Theorie

Um eine Vereinfachung der Notation zu erreichen, wird in diesem Abschnitt die empirische Varianz $\mathrm{Var}_n(X)$ einer Zufallsgröße X auf der Basis von n Realisierungen $(x_1, \ldots, x_n)$ definiert als

$$\mathrm{Var}_n(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}_n)^2 \quad \text{mit} \quad \bar{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i \quad .$$

Es wird also - abweichend von der Festsetzung in Abschnitt 2 - der Normierungsfaktor $\frac{1}{n}$ anstelle von $\frac{1}{n-1}$ verwendet.
Der Rang r_k einer Beobachtung x_k aus einer Stichprobe $(x_1, \ldots, x_n)$ vom Umfang n mit $x_i \neq x_j \ \forall i \neq j$ ist definiert als die Position von x_k in der zugehörigen geordneten Stichprobe, also

$$r_k = \sum_{j=1}^{n} I_{\{x_j \leq x_k\}} \quad .$$

Für das arithmetische Mittel und die empirische Varianz eines Rangvektors $R = (r_1, \ldots, r_n)$ der Beobachtungen $(x_1, \ldots, x_n)$ einer Stichprobe vom Umfang n mit $x_i \neq x_j \ \forall i \neq j$ gelten

$$\bar{r}_n = \frac{n+1}{2} \quad , \quad \mathrm{Var}_n(R) = \frac{n^2-1}{12} \quad ,$$

wegen

$$\bar{r}_n = \ \frac{1}{n}\sum_{i=1}^{n} i \ = \ \frac{1}{n} \cdot \frac{n \cdot (n+1)}{2} \ = \ \frac{n+1}{2}$$

und

$$\begin{aligned}
\mathrm{Var}_n(R) &= \frac{1}{n}\sum_{i=1}^{n}(i-\bar{r}_n)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(i - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\left[\sum_{i=1}^{n} i^2 - \sum_{i=1}^{n}(n+1)\cdot i + \sum_{i=1}^{n}\left(\frac{n+1}{2}\right)^2\right] \\
&= \frac{1}{n}\left[\frac{n\cdot(n+1)\cdot(2n+1)}{6} - \frac{n\cdot(n+1)^2}{2} + \frac{n\cdot(n+1)^2}{4}\right] \\
&= \frac{1}{n}\left[\frac{n\cdot(n+1)\cdot(2n+1)}{6} - \frac{n\cdot(n+1)^2}{4}\right] \\
&= \frac{1}{n}\left[\frac{2n\cdot(n+1)\cdot(2n+1) - 3n\cdot(n+1)^2}{12}\right] \\
&= \frac{1}{n}\left[\frac{4n^3+2n^2+4n^2+2n-3n^3-6n^2-3n}{12}\right] \\
&= \frac{1}{n}\left(\frac{n\cdot(n^2-1)}{12}\right) = \frac{n^2-1}{12} \quad .
\end{aligned}$$

Die Definition des Ranges setzt voraus, dass alle Messwerte unterschiedlich sind. Da dies jedoch nicht immer der Fall ist, sind Überlegungen nötig, die die Behandlung mehrfach auftretender Werte (Ties) möglich machen. Zur Vereinfachung der Darstellung nehmen wir an, dass für die der Größe nach geordneten Beobachtungen $x_{i:n}$ der betrachteten Stichprobe $x_{1:n} = x_{2:n} = \ldots = x_{d_1:n}$ gilt. Dann ist es sicherlich nicht sinnvoll, alle Ränge von 1 bis d_1 zu vergeben, denn damit würde die Information der Gleichheit von $x_{1:n}, \ldots, x_{d_1:n}$ verlorengehen. Ebenfalls sollte der Rang 1 nicht d_1-mal gewählt werden, denn dies würde einen sehr großen Abstand der Werte $x_{1:n}, \ldots, x_{d_1:n}$ zu $x_{d_1+1:n}$ erzeugen, der in Wirklichkeit nicht gegeben sein muss. Ein analoges Argument gilt selbstverständlich für die Festsetzung, dass alle Werte $x_{1:n}, \ldots, x_{d_1:n}$ den Rang d_1 erhalten. Vielmehr wird man dazu übergehen, den Beobachtungen $x_{1:n}, \ldots, x_{d_1:n}$ den mittleren Rang $\frac{d_1+1}{2}$ zuzuweisen. Allgemein geschieht die Zuordnung der Ränge im Falle des Vorhandenseins von Ties wie folgt: Gegeben sei eine Stichprobe $(x_1, \ldots, x_n)$ vom Umfang n mit p unterschiedlichen Werten v_i, also $|\{x_i : 1 \leq i \leq n\}| = p$. Der Wert $v_i, i = 1, \ldots, p$ mit $v_1 < v_2 < \ldots < v_p$ trete dabei d_i-mal auf. Der mittlere Rang r_k^* einer Beobachtung x_k aus einer solchen Stichprobe ist definiert als

$$r_k^* = \sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \quad \forall k : x_k = v_i\,,\ i = 1, \ldots, p \quad .$$

Für das arithmetische Mittel des Vektors der mittleren Ränge $R^* = (r_1^*, \ldots, r_n^*)$ gilt

$$\overline{r^*}_n = \frac{1}{n} \sum_{k=1}^{n} r_k^* = \frac{n+1}{2} \quad .$$

Zum Beweis seien k_1 und k_2 zwei natürliche Zahlen mit $k_2 > k_1$. Dann gilt:

$$\begin{aligned}
(k_2 - k_1) \cdot \left[k_1 + \frac{1}{2} \cdot (k_2 - k_1 + 1) \right] &= \frac{(k_2 - k_1) \cdot (2k_1 + k_2 - k_1 + 1)}{2} \\
&= \frac{(k_2 - k_1) \cdot (k_1 + k_2 + 1)}{2} \\
&= \frac{k_1 k_2 + k_2(k_2+1) - k_1(k_1+1) - k_1 k_2}{2} \\
&= \frac{k_2(k_2+1)}{2} - \frac{k_1(k_1+1)}{2} \\
&= \sum_{i=1}^{k_2} i - \sum_{i=1}^{k_1} i \\
&= \sum_{i=k_1+1}^{k_2} i \quad .
\end{aligned}$$

Wählen wir nun speziell $k_1 = \sum_{j=1}^{i-1} d_j$ und $k_2 = \sum_{j=1}^{i} d_j$, dann gilt offensichtlich $k_2 - k_1 = d_i$ und damit

$$\begin{aligned}
\sum_{k=1}^{n} r_k^* &= \sum_{i=1}^{p} \left[d_i \cdot \left(\sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \right) \right] \\
&= \sum_{i=1}^{p} \sum_{l=\sum_{j=1}^{i-1} d_j + 1}^{\sum_{j=1}^{i} d_j} l \\
&= \sum_{i=1}^{n} i \\
&= \frac{n \cdot (n+1)}{2} \quad .
\end{aligned}$$

Hieraus folgt, dass das arithmetische Mittel der (mittleren) Ränge einer Stichprobe vom Umfang n konstant also insbesondere invariant bezüglich des Vorhandenseins beliebig gearteter Ties ist. Für die empirische Varianz gilt diese Aussage hingegen nicht. Es ist zwar möglich, eine geschlossene Formel anzugeben, diese beinhaltet jedoch einen Korrekturterm, in welchen die Informationen über die vorhandenen verbundenen Werte eingehen. Es gilt

$$\begin{aligned}
\mathrm{Var}_n(R^*) &= \frac{1}{n} \sum_{k=1}^{n} \left(r_k^* - \frac{n+1}{2} \right)^2 \\
&= \frac{n^2 - 1}{12} - \frac{\sum_{i=1}^{p} d_i (d_i^2 - 1)}{12n} \quad . \qquad (6)
\end{aligned}$$

Eine Herleitung von (6) findet sich unter anderem in [13] auf S. 329f. Hier wird sie im Folgenden durch elementares Nachrechnen gezeigt. Dazu treffen wir zur Vereinfachung der Schreibweise folgende Definitionen:

$$\begin{aligned}
s_{R^*}^2 &:= \mathrm{Var}_n(R^*) \\
s_i &:= \sum_{j=1}^{i} d_j \\
a_i &:= s_{i-1} + \frac{1}{2} \cdot (d_i + 1) \quad .
\end{aligned}$$

Mit diesen Vereinbarungen folgt

$$\begin{aligned}
s_{R^*}^2 &= \frac{1}{n}\sum_{k=1}^{n}\left(r_k^* - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p} d_i \cdot \left(a_i - \frac{n+1}{2}\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left(\left(l - \frac{n+1}{2}\right) - (l-a_i)\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[\left(l - \frac{n+1}{2}\right)^2 + (l-a_i)^2 - 2\cdot\left(l - \frac{n+1}{2}\right)\cdot(l-a_i)\right].
\end{aligned}$$

Ein Aufteilen der Summen führt zu

$$\begin{aligned}
s_{R^*}^2 &= \frac{1}{n}\sum_{k=1}^{n}\left(k - \frac{n+1}{2}\right)^2 \\
&\quad + \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[(l-a_i)^2 - 2\cdot\left(l - a_i + a_i - \frac{n+1}{2}\right)\cdot(l-a_i)\right] \\
&= \frac{n^2-1}{12} + \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}\left[-(l-a_i)^2 - 2\cdot\left(a_i - \frac{n+1}{2}\right)\cdot(l-a_i)\right] \\
&= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\sum_{l=s_{i-1}+1}^{s_i}(l-a_i)^2 - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\sum_{l=s_{i-1}+1}^{s_i}(l-a_i)\right].
\end{aligned}$$

Sei $l' := l - s_{i-1}$, dann gilt

$$\begin{aligned}
s_{R^*}^2 &= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\sum_{l'=1}^{d_i}\left(l' - \frac{d_i+1}{2}\right)^2 \\
&\quad - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\sum_{l'=1}^{d_i}\left(l' - \frac{d_i+1}{2}\right)\right] \\
&= \frac{n^2-1}{12} - \frac{1}{n}\sum_{i=1}^{p}\frac{d_i\cdot(d_i^2-1)}{12} \\
&\quad - \frac{2}{n}\sum_{i=1}^{p}\left[\left(a_i - \frac{n+1}{2}\right)\cdot\left(\frac{d_i\cdot(d_i+1)}{2} - \frac{d_i\cdot(d_i+1)}{2}\right)\right] \\
&= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{p} d_i\cdot(d_i^2-1)}{12n}.
\end{aligned}$$

Um nun die gegebene Aufgabe, die Rangkorrelation zweier Stichproben $(x_1, \ldots, x_n)$ sowie $(y_1, \ldots, y_n)$ mit zugehörigen mittleren Rängen $(r_1^*, r_2^*, \ldots, r_n^*)$ bzw. $(s_1^*, s_2^*, \ldots, s_n^*)$ zu berechnen, die überdies jeweils Ties an unterschiedlichen Stellen aufweisen können, kann man sich der Formel für die empirische Varianz der Differenz zweier Zufallsgrößen bedienen:

$$\begin{aligned} & \mathrm{Var}_n(X-Y) &=& \mathrm{Var}_n(X) + \mathrm{Var}_n(Y) - 2\mathrm{Kov}_n(X,Y) \\ \iff \quad & \mathrm{Kov}_n(X,Y) &=& \frac{1}{2}(\mathrm{Var}_n(X) + \mathrm{Var}_n(Y) - \mathrm{Var}_n(X-Y)) \end{aligned} \tag{7}$$

Die (empirische) Varianz von $Z := R^* - S^*$ ist aber in dem Fall, dass die Realisierungen von R^* und S^* die Vektoren der (mittleren) Ränge sind, einfach zu berechnen, denn es gilt $\bar{r_n^*} = \bar{s_n^*} \equiv \frac{n+1}{2}$ und damit $\bar{z}_n = \bar{r_n^*} - \bar{s_n^*} = 0$. Es folgt

$$\mathrm{Var}_n(Z) = \frac{1}{n}\sum_{i=1}^{n}(z_i - \bar{z}_n)^2 = \frac{1}{n}\sum_{i=1}^{n} z_i^2 - \bar{z}_n^2 = \frac{1}{n}\sum_{i=1}^{n}(r_i^* - s_i^*)^2 \quad .$$

Setzt man dieses Ergebnis in (7) für die Kovarianz bezüglich der Rangvektoren ein, so ergibt sich mit $D^* := \sum_{i=1}^{n}(r_i^* - s_i^*)^2$

$$\mathrm{Kov}_n(R^*, S^*) = \frac{1}{2}\left(\mathrm{Var}_n(R^*) + \mathrm{Var}_n(S^*) - \frac{2D^*}{n}\right) \quad .$$

Schließlich definiert sich der Rangkorrelationskoeffizient $\rho_n^S(X,Y)$ nach Spearman als Produktmomentkorrelation der zu (X,Y) gehörenden Ranggrößen wie folgt:

Voraussetzungen:

1. Gegeben sind Realisierungen $(x_1, \ldots, x_n)$ und $(y_1, \ldots, y_n)$ zweier Zufallsvariablen X und Y.
2. Die Realisierungen von X weisen p unterschiedliche Werte v_i auf, also $|\{x_i : i \in \{1, \ldots, n\}\}| = p$. Der Wert $v_i, i = 1, \ldots, p$ mit $v_1 < v_2 < \ldots < v_p$ tritt dabei d_i-mal auf.
3. Die Realisierungen von Y weisen q unterschiedliche Werte w_i auf, also $|\{y_i : i \in \{1, \ldots, n\}\}| = q$. Der Wert $w_i, i = 1, \ldots, q$ mit $w_1 < w_2 < \ldots < w_q$ tritt dabei e_i-mal auf.
4. $R^* = (r_1^*, r_2^*, \ldots, r_n^*)$ bezeichnet den Vektor der mittleren Ränge von X.
5. $S^* = (s_1^*, s_2^*, \ldots, s_n^*)$ bezeichnet den Vektor der mittleren Ränge von Y.

Dann heißt

$$\rho_n^S(X,Y) := \rho_n(R^*,S^*) = \frac{\frac{1}{2}(\text{Var}_n(R^*) + \text{Var}_n(S^*) - \frac{2D^*}{n})}{\sqrt{\text{Var}_n(R^*)} \cdot \sqrt{\text{Var}_n(S^*)}}$$

der Rangkorrelationskoeffizient nach Spearman von X und Y. Dabei gelten

$$\begin{aligned}
r_k^* &= \sum_{j=1}^{i-1} d_j + \frac{1}{2} \cdot (d_i + 1) \;\; \forall k : x_k = v_i; \;\; i = 1, \ldots, p \quad , \\
s_k^* &= \sum_{j=1}^{i-1} e_j + \frac{1}{2} \cdot (e_i + 1) \;\; \forall k : y_k = w_i; \;\; i = 1, \ldots, q \quad , \\
\text{Var}_n(R^*) &= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{p} d_i(d_i^2-1)}{12n} \quad , \\
\text{Var}_n(S^*) &= \frac{n^2-1}{12} - \frac{\sum_{i=1}^{q} e_i(e_i^2-1)}{12n} \quad , \\
D^* &= \sum_{i=1}^{n} (r_i^* - s_i^*)^2 \quad .
\end{aligned}$$

5.3 Anwendung

Neben der Berechnung der Produktmomentkorrelation bietet die während des Projekts entstandene Software auch die Möglichkeit, den Rangkorrelationskoeffizienten nach Spearman für je ein Variablenpaar zu berechnen. Dies bietet sich insbesondere in dem Fall an, dass eine Ausreißererkennung gemäß der in Abschnitt 3.3.3 vorgestellten Methode des Erkennens „echter“ Ausreißer vorgenommen werden soll. Hierbei werden die kritischen Werte für die Mahalanobis-Distanz, die zu dem Urteil „Datenpunkt ist Ausreißer“ führen, groß, und deswegen reduziert sich die Anzahl der als Ausreißer markierten und damit von der weiteren Analyse ausgenommenen Datenpunkte bei Verwendung des Korrelationskoeffizienten nach Pearson zum Teil erheblich. Dies kann in Einzelfällen dazu führen, dass signifikante Korrelationen unerkannt bleiben bzw. nicht durch Elimination von Ausreißern aufgedeckt werden. Zudem wird im Falle der Verwendung kleiner kritischer Werte für die Mahalanobis-Distanz bei einem nichtlinearen Zusammenhang zwischen zwei betrachteten Variablen ein Großteil der Randbereiche der Daten durch die Ausreißerelimination entfernt, und es erfolgt so eine Konzentration auf die Kernbereiche des Datenmaterials, in denen dann in der Regel eine stark ausgeprägte lineare Beziehung besteht. Werden hingegen große kritische Werte verwendet, so ergibt sich dieser Effekt nicht, und es kann passieren, dass signifikante Zusammenhänge verloren gehen. Betrachtet man in diesen Fällen hingegen die Rangkorrelation nach Spearman, so geht anstelle der absoluten Distanz der Messwerte vom Datenzentrum nur deren Ordnung in die Berechnung ein, und die Gefahr, Ausreißer zu „vorsichtig“ zu entfernen, ist damit deutlich geringer.

6 Hauptkomponentenanalyse

6.1 Motivation

„The central idea of principal component analysis is to reduce the dimensionality of a data set in which there are a large number of interrelated variables, while retaining as much as possible of the variation present in the data set. This reduction is achieved by transforming to a new set of variables, the principal components, which are uncorrelated, and which are ordered so that the first few retain most of the variation present in all of the original variables.“

Dieses Zitat aus der Einleitung zu dem Buch von I. T. Jolliffe [12] beschreibt die Ziele der Hauptkomponentenanalyse (principal component analysis). Der m-dimensionale Datenraum, der durch die gemeinsam (multivariat) zu analysierenden Zufallsvariablen X_1, $X_2, \ldots, X_m$ aufgespannt wird, soll durch einen p-dimensionalen Datenraum mit $p \ll m$ ersetzt werden. Hierbei soll jedoch ein möglichst großer Anteil der Information, die die Originaldaten liefern, erhalten bleiben. Deswegen macht man sich bei dem Konzept der Hauptkomponentenanalyse die Abhängigkeiten (Korrelationen) zwischen den Originalvariablen zu Nutze und bestimmt p geeignete Linearkombinationen der m ursprünglichen Variablen mit möglichst großer Varianz, die dann unkorreliert sind. Geometrisch ausgedrückt bedeutet dies, dass die Streuung (Variation) in den Originaldaten in orthogonale Streurichtungen zerlegt wird, wobei die Streuintensität entlang dieser Richtungen immer weiter abnimmt. Bei diesem Vorgehen wird davon ausgegangen, dass die Gesamtstreuung von $p \ll m$ Richtungen hinreichend gut erfasst wird, also in den verbleibenden $(m-p)$ Richtungen eine nahezu konstante lineare Beziehung zwischen den Originalvariablen besteht. Die durch die oben genannten Linearkombinationen entstehenden neuen p Zufallsvariablen $C_1, C_2, \ldots, C_p$ nennt man die ersten p Hauptkomponenten der zu Grunde liegenden m-dimensionalen Verteilung.

6.2 Theoretischer Hintergrund

Um die in Abschnitt 6.1 postulierten Eigenschaften der Hauptkomponenten mathematisch zu fassen, bezeichnen nun $v_i, i = 1, \ldots, p$ mit $v_i \in \mathbb{R}^m \;\; \forall\, i = 1, \ldots, p$ die Vektoren, die die Koeffizienten der zu den ersten p Hauptkomponenten gehörenden Linearkombinationen enthalten. Diese v_i und damit die Hauptkomponenten $C_1, C_2, \ldots, C_p$ werden durch folgende Optimalitätsforderungen festgelegt:

P1: $v_i^\top \cdot v_j = \delta_{i,j} \;\; \forall i, j = 1, \ldots, p \quad ,$

P2: $(v_1, \ldots, v_k) = \underset{A \in \mathbb{R}^{m \times k}}{\operatorname{argmax}} \operatorname{tr}(A^\top \cdot K \cdot A) \quad \forall k = 1, \ldots, p \quad$ mit $K \in \mathbb{R}^{m \times m}$

Varianz- / Kovarianzmatrix der Originalvariablen $X_1, \ldots, X_m$.

Die erste Forderung entspricht der Orthogonalität der entstehenden Streurichtungen sowie einer Normierung der Vektoren v_i bezüglich der L_2-Norm, und die zweite Forderung formalisiert die sukzessive Bestimmung der Richtungen mit maximaler Streuung, denn $K^* := A^\top \cdot K \cdot A$ ist gerade die Varianz- / Kovarianzmatrix der Linearkombination $A^\top \cdot (X_1, \ldots, X_m)^\top$ (Basiswechsel im k-dimensionalen Teilraum).
Um die v_i zu berechnen, setzen wir zunächst $p = 1$. Damit vereinfachen sich P1 und P2 zu

$$v_1 = \underset{a \in \mathbb{R}^m}{\operatorname{argmax}}\, a^\top \cdot K \cdot a \quad \text{mit } \| a \|_2 = 1 \quad . \tag{8}$$

Wendet man das Verfahren der Lagrange-Multiplikatoren zur Lösung von Maximierungsproblemen unter Nebenbedingungen an (siehe z.B. [3], Kapitel 3.5), so ergibt sich als Lagrange-Funktion der Ausdruck

$$\mathcal{L}(a) = a^\top \cdot K \cdot a - \lambda_1 \cdot (a^\top \cdot a - 1), \quad \lambda_1 \in \mathbb{R} \quad .$$

Partielles Differenzieren nach den Komponenten von a führt auf das Gleichungssystem

$$\begin{aligned} & K \cdot a - \lambda_1 \cdot a &&= 0 \\ \iff \qquad & (K - \lambda_1 \cdot E) \cdot a &&= 0 \quad . \end{aligned}$$

Dieses letzte Gleichungssystem entspricht gerade dem Eigenwertproblem der Varianz- / Kovarianzmatrix K, so dass sich v_1 als ein Eigenvektor von K ergibt. Um nun zu entscheiden, welcher Eigenvektor das Maximierungsproblem löst, beachte man, dass

$$v_1^\top \cdot K \cdot v_1 \;=\; v_1^\top \cdot \lambda_1 \cdot v_1 \;=\; \lambda_1 \cdot v_1^\top \cdot v_1 \;=\; \lambda_1$$

gilt. Also muss λ_1 der größte Eigenwert von K und v_1 der zugehörige Eigenvektor sein. Ferner gilt

$$\text{Var}(C_1) \;=\; \text{Var}(v_1^\top \cdot (X_1, \ldots, X_m)^\top) \;=\; v_1^\top \cdot K \cdot v_1 \;=\; \lambda_1 \quad .$$

Es ergibt sich aufgrund der Orthonormalitätsforderung an die v_i, dass sich die Maximierung der Spur von K^* entkoppelt in p Maximierungsprobleme der Form (8) (siehe dazu auch [12], Seiten 9f.), so dass sich die folgenden Ergebnisse festhalten lassen:

1. v_i sind Eigenvektoren zu den Eigenwerten λ_i von K und es gilt $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$,
2. $\text{Var}(C_i) = \lambda_i, \;\; i = 1, \ldots, m$.

Führt man eine Hauptkomponentenanalyse für standardisierte Zufallsgrößen durch, so geht die Varianz- / Kovarianzmatrix K in die zugehörige Korrelationsmatrix R über. Es gilt

dann $\mathrm{Var}(X_i) = 1 \ \ \forall i = 1, \ldots, m$ und damit $\mathrm{tr}(R) = m$. Will man also bei der Hauptkomponentenanalyse eine implizite Standardisierung bzw. Maßstabsvereinheitlichung der zu analysierenden Variablen erzielen, so kann anstelle der Varianz- / Kovarianzmatrix die Korrelationsmatrix zur Bestimmung der Eigenwerte und Eigenvektoren zur Definition der Hauptkomponenten herangezogen werden. Die so berechneten $(\tilde{\lambda}_i, \tilde{v}_i)$ weichen im Allgemeinen von denen ab, die sich im Falle der Verwendung der Varianz- / Kovarianzmatrix ergeben.
Die im Rahmen dieses Projektes entstandene Implementierung verwendet Korrelationsmatrizen bei der Hauptkomponentenanalyse, wobei in der konkreten Berechnung natürlich wieder zu den entsprechenden empirischen Größen R_n übergegangen wird.

6.3 Anwendung

In praktischen Anwendungsfällen der Hauptkomponentenanalyse besteht die grundlegende Fragestellung offensichtlich darin, wie groß der Parameter p zu wählen ist. Dies wird davon abhängen, welchen Informationsverlust man hinzunehmen bereit ist. Ein häufig gemachter Vorschlag dazu besteht darin, p so zu wählen, dass im Falle der Verwendung von Korrelationsmatrizen $p = \max\{1 \leq i \leq m : \lambda_i \geq 1\}$ gilt. Anschaulich bedeutet diese Wahl von p, dass die einzubeziehenden Hauptkomponenten jeweils den Informationsgehalt mindestens einer Originalvariablen haben. In einigen Situationen kann diese Wahl von p jedoch dazu führen, dass einzelne Variablen komplett unberücksichtigt bleiben und deswegen empfiehlt I.T. Jolliffe in [12] die Wahl von $p = \max\{1 \leq i \leq m : \lambda_i \geq 0,7\}$, wenn möglichst sichergestellt werden soll, dass alle Variablen anteilig in den Hauptkomponenten Berücksichtigung finden.
Ein anderer Ansatz besteht darin, den Anteil der durch die Hauptkomponenten $C_1, \ldots, C_p$ erklärten Streuung, also $\sum_{i=1}^{p} \lambda_i$ mit $\lambda_1 \geq \ldots \geq \lambda_p$ Eigenwerte der Korrelationsmatrix $R \in \mathbb{R}^{m \times m}$ von $(X_1, \ldots, X_m)$ und ihr Verhältnis zu $\mathrm{tr}(R) = m$ (es werden hier standardisierte Variablen X_i betrachtet) zum Entscheidungskriterium für die Größe von p zu machen. In dem Fall, dass sich einzelne Variablen annähernd mit Hauptkomponenten identifizieren lassen, also nur sehr gering mit den restlichen Zufallsgrößen korrelieren, wird es jedoch auch hier dazu kommen, dass diese Variablen durch die ersten Hauptkomponenten fast gar nicht erfasst werden. Da dies eventuell unerwünscht ist, ist zusätzlich eine Information darüber nötig, wie groß im Fall der am schlechtesten erfassten Variable deren Restvarianz ist. Die Restvarianz bezeichnet den Anteil der Varianz der Variable X_i, der nicht durch die Hauptkomponenten $C_1, \ldots C_p$ erklärt wird, also:

$$\mathrm{Var}_{\mathrm{Rest},p}(X_i) = \min_{(a_1,\ldots,a_p)^\top \in \mathbb{R}^p} \mathrm{Var}\left(X_i - \sum_{j=1}^{p} a_j \cdot C_j \right) \quad . \tag{9}$$

Es ergibt sich:

$$\begin{aligned}
&\text{Var}\left(X_i - \sum_{j=1}^{p} a_j \cdot C_j\right) \\
&= \text{Var}(X_i) - 2 \cdot \sum_{j=1}^{p} a_j \cdot \text{Kov}(X_i, C_j) + \sum_{j=1}^{p}\sum_{l=1}^{p} a_j \cdot a_l \cdot \text{Kov}(C_j, C_l) \\
&= \text{Var}(X_i) + \sum_{j=1}^{p} a_j^2 \cdot \text{Var}(C_j) - 2 \cdot \sum_{j=1}^{p} a_j \cdot \sqrt{\text{Var}(C_j)} \cdot \frac{\text{Kov}(X_i, C_j)}{\sqrt{\text{Var}(C_j)}} \\
&\quad + \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} - \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} \qquad (\text{Kov}(C_j, C_l) = 0 \; \forall j \neq l) \\
&= \text{Var}(X_i) + \sum_{j=1}^{p} \left(a_j \cdot \sqrt{\text{Var}(C_j)} - \frac{\text{Kov}(X_i, C_j)}{\sqrt{\text{Var}(C_j)}}\right)^2 - \sum_{j=1}^{p} \frac{\text{Kov}(X_i, C_j)^2}{\text{Var}(C_j)} \\
&= \text{Var}(X_i) \cdot \left(1 - \sum_{j=1}^{p} \rho^2(X_i, C_j)\right) + \sum_{j=1}^{p} \text{Var}(C_j) \cdot \left(a_j - \frac{\text{Kov}(X_i, C_j)}{\text{Var}(C_j)}\right)^2 \quad .
\end{aligned}$$

Aus dieser Rechnung folgt, dass die Koeffizienten a_j, welche (9) minimieren, gerade die Regressionskoeffizienten

$$a_j = \frac{\text{Kov}(X_i, C_j)}{\text{Var}(C_j)}, \; \forall j = 1, \ldots, p$$

sind und die gesuchte Restvarianz damit von der Form

$$\text{Var}_{\text{Rest},p}(X_i) = \text{Var}(X_i) \cdot \left(1 - \sum_{j=1}^{p} \rho^2(X_i, C_j)\right)$$

ist. Um schließlich noch $\rho(X_i, C_j)$ zu berechnen, bezeichne $V \in \mathbb{R}^{m \times m}$ die Matrix der im L_2-Sinne normierten Eigenvektoren v_j $(j = 1, \ldots, n)$ von K, die nach der Größe der zugehörigen Eigenwerte absteigend geordnet sind, so dass sich die Hauptkomponenten $C_1, \ldots, C_m$ als $C_j = v_j^\top \cdot (X_1, \ldots, X_m)^\top \; \forall j = 1, \ldots, m$ ergeben. Dann gilt:

$$K \;=\; V \cdot \text{diag}(\lambda_i) \cdot V^\top$$

und man erhält

$$\text{Kov}(X_i, X_k) \;=\; \sum_{j=1}^{m} v_{i,j} \cdot \lambda_j \cdot v_{k,j}$$

bzw.

$$\begin{aligned}
\text{Kov}(X_i, C_j) &= \text{Kov}(X_i, \sum_{k=1}^{m} v_{k,j} \cdot X_k) \\
&= \sum_{k=1}^{m} \text{Kov}(X_i, X_k) \cdot v_{k,j} \\
&= \sum_{k=1}^{m} v_{k,j} \cdot \left(\sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot v_{k,l} \right) \\
&= \sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot \left(\sum_{k=1}^{m} v_{k,j} v_{k,l} \right) \\
&= \sum_{l=1}^{m} v_{i,l} \cdot \lambda_l \cdot \delta_{l,j} \\
&= v_{i,j} \cdot \lambda_j
\end{aligned}$$

und damit (unter Beachtung von $\mathrm{Var}(C_j) = \lambda_j$):

$$\rho(X_i, C_j) = \frac{v_{i,j} \cdot \sqrt{\lambda_j}}{\sqrt{\text{Var}(X_i)}} \quad .$$

Um also dem Benutzer eine möglichst gute Entscheidungshilfe für die Wahl von p an die Hand zu geben, wurde in der erfolgten Implementierung die Tabelle 2 eingebunden.

i	λ_i	$\lambda_{i-1} - \lambda_i$	λ_i / m	$\sum_{j=1}^{i} \lambda_j / m$	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},i}(X_k)$
1	λ_1	-	λ_1 / m	λ_1 / m	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},1}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
p	λ_p	$\lambda_{p-1} - \lambda_p$	λ_p / m	$\sum_{j=1}^{p} \lambda_j / m$	$\max_{k \in \{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	λ_m	$\lambda_{m-1} - \lambda_m$	λ_m / m	1	0

Tabelle 2: Schematische Darstellung von Ergebnissen bei der Hauptkomponentenanalyse.

Für die Wahl von p können insbesondere die letzten beiden Spalten dieser Tabelle zu Rate gezogen werden. Überschreitet der Wert in der vorletzten bzw. unterschreitet der Wert in der letzten Spalte in einer Zeile i einen Schwellenwert, so ist dieses i als die Anzahl p zu berücksichtigender Hauptkomponenten zu wählen.

6.4 Ausreißererkennung mit Hauptkomponenten

Wie im Falle der Betrachtung der Originalvariablen kann es auch beim Übergang zu den Hauptkomponenten von Interesse sein, Beobachtungen zu erkennen, die in dem (jetzt durch die Hauptkomponenten aufgespannten) Datenraum auffällig aus dem Gros der Daten herausfallen. Innerhalb des Projekts sind drei Ausprägungen solcher „Auffälligkeiten“ untersucht worden:

- Es kann Datenpunkte geben, die bezüglich der Verteilung einer bestimmten Hauptkomponente signifikant weit vom Datenzentrum entfernt liegen.
- Ein in Koordinaten des durch die Hauptkomponenten $(C_1, \ldots, C_p)$ gegebenen Koordinatensystems ausgedrückter Beobachtungsvektor kann einen signifikant großen Abstand zum Datenzentrum im $\mathbb{R}^p$ haben.
- Im Allgemeinen wird, wie in Abschnitt 6.1 beschrieben, davon ausgegangen, dass aus m Komponenten bestehende Beobachtungen in dem durch die ersten p Hauptkomponenten gegebenen p-dimensionalen Teildatenraum hinreichend gut beschrieben werden. Es kann jedoch Beobachtungen geben, für welche diese Annahme nicht zutrifft.

Datenpunkte, bei welchen eine solche Situation vorliegt, werden hier wieder als Ausreißer bezeichnet. Um diese Ausreißer erkennen zu können, sind Teststatistiken mit zugehörigen Verlässlichkeitsschranken für die obigen Problemstellungen nötig. In den folgenden Abschnitten wird das hierzu gewählte Vorgehen erläutert.

Dazu gelten vorab:

1. Die Ausgangsdaten bestehen aus den Beobachtungsvektoren $x_i = (x_{i,1}, \ldots, x_{i,m})^\top$, $i = 1, \ldots, n$. Ein solcher Beobachtungsvektor enthält die Messwerte aus den Laborexperimenten, die mit einer chemischen Struktur durchgeführt wurden. Zusammengefasst werden diese Beobachtungsvektoren in der $(n \times m)$-Datenmatrix $X = (x_{i,j})$, wobei n die Anzahl der Beobachtungen (Strukturen) und m die Anzahl der Variablen (Experimente) bezeichnet. Die Zeilen von X sind also die transponierten Beobachtungsvektoren x_i. Zudem sind die Spaltenvektoren von X - die jeweils ein Laborexperiment widerspiegeln - auf Mittelwert 0 und Varianz 1 standardisiert.
2. Mit $V \in \mathbb{R}^{m \times m}$ wird die Matrix der im L_2-Sinne normierten Eigenvektoren der Korrelationsmatrix der Originalvariablen bezeichnet. Diese sind nach der Größe der zugehörigen Eigenwerte absteigend geordnet, so dass sich die Hauptkomponenten $C_1, \ldots, C_m$ als $C_j = v_j^\top \cdot (X_1, \ldots, X_m)^\top \ \forall j = 1, \ldots, m$ ergeben.
3. Die Matrix der auf $C_1, \ldots, C_m$ bezogenen Koordinaten der Beobachtungen (im Folgenden auch Matrix der Scores der Beobachtungen genannt) wird mit $Y = X \cdot V$, $Y \in \mathbb{R}^{n \times m}$, bezeichnet. Auch im Falle der Scores liegen die Beobachtungen wieder als Zeilenvektoren vor.

6.4.1 Eindimensionale Ausreißer

Bei der Bestimmung von Ausreißern bezüglich einer speziellen Hauptkomponente C_j wurden zunächst die Laborexperimente bzw. die durch sie induzierten Zufallsvariablen als normalverteilt vorausgesetzt. Diese Annahme wurde in Ermangelung von Informationen über die tatsächlichen Verteilungen getroffen und muss daher im Einzelfall nicht zwingend korrekt sein. Die k-te Spalte von $X, X^{(k)}, k = 1, \ldots, m$, wird also als Vektor von n unabhängigen und identisch $\mathcal{N}(0,1)$-verteilten Zufallsgrößen interpretiert. Damit ergibt sich die mit einer Hauptkomponente C_j assoziierte Spalte $Y^{(j)}, j = 1, \ldots, m$ der Score-Matrix Y ebenfalls als ein Vektor von n normalverteilten Größen, denn sie entsteht durch eine Linearkombination von $\mathcal{N}$(0, 1)-verteilten Variablen. Zudem sind die einzelnen Komponenten von $Y^{(j)}$ wieder unabhängig, denn in die Linearkombination gehen nur Werte der jeweils gleichen Beobachtung ein. Wie in Abschnitt 6.2 hergeleitet, ergibt sich die Varianz der Größen in einer Spalte $Y^{(j)}$ als Wert des zugehörigen Eigenwertes λ_j der zu Grunde liegenden Korrelationsmatrix. Es gelten

$$y_{i,j} \text{ i.i.d.} \sim \mathcal{N}(0, \lambda_j) \quad \forall i = 1, \ldots, n;\ j \in \{1, \ldots, m\}$$

und

$$z_{i,j} := \frac{y_{i,j}}{\sqrt{\lambda_j}} \text{ i.i.d.} \sim \mathcal{N}(0,1) \quad \forall i = 1, \ldots, n;\ j \in \{1, \ldots, m\} \quad .$$

Damit lassen sich die normierten Scores $z_{i,j}$ als Teststatistiken zur Ausreißererkennung verwenden. In Anwendungen ist jedoch zu beachten, dass anstelle der exakten Korrelationsmatrizen die entsprechenden Schätzungen verwendet werden müssen und die hier gemachten Verteilungsaussagen dann nur noch asymptotisch gelten. Wird nun analog zu der Vorgehensweise in Abschnitt 3.3.3 wieder gefordert, dass nur in $\alpha \cdot 100\%$ der Fälle Beobachtungen als Ausreißer deklariert werden, wenn keine tatsächlichen Ausreißer vorliegen, so ergibt sich asymptotisch (Φ bezeichnet die Verteilungsfunktion der $\mathcal{N}(0,1)$-Verteilung):

$$\begin{aligned}
& & \mathbb{P}(\max_{1 \le i \le n} |z_{i,j}| > c) &\stackrel{!}{=} \alpha \\
\iff & & \mathbb{P}(|z_{i,j}| \le c \,\forall i = 1, \ldots, n) &= 1 - \alpha \\
\iff & & \mathbb{P}(-c \le z_{i,j} \le c \,\forall i = 1, \ldots, n) &= 1 - \alpha \\
\underset{\text{Unabhängigkeit}}{\iff} & & \textstyle\prod_{i=1}^{n} \mathbb{P}(-c \le z_{i,j} \le c) &= 1 - \alpha \\
\underset{\text{Verteilungsidentität}}{\iff} & & (\mathbb{P}(-c \le Z \le c))^n &= 1 - \alpha \text{ mit } Z \sim \mathcal{N}(0,1) \\
\iff & & (\Phi(c) - \Phi(-c))^n &= 1 - \alpha \\
\iff & & (\Phi(c) - (1 - \Phi(c)))^n &= 1 - \alpha \\
\iff & & 2 \cdot \Phi(c) - 1 &= \sqrt[n]{(1-\alpha)} \\
\iff & & \Phi(c) &= \tfrac{1}{2} \cdot (1 + \sqrt[n]{(1-\alpha)}) \\
\iff & & c &= \Phi^{-1}\left(\frac{1 + \sqrt[n]{(1-\alpha)}}{2}\right) \quad .
\end{aligned}$$

Überschreitet $|z_{i,j}| = |y_{i,j}/\sqrt{\lambda_j}|$ für eine Beobachtung i in Spalte j der Score-Matrix Y also das Quantil $c = \Phi^{-1}\left(\frac{1+\sqrt[n]{(1-\alpha)}}{2}\right)$ der Standard-Normalverteilung, so wird der zugehörige Datenpunkt als eindimensionaler Ausreißer zur Hauptkomponente C_j angesehen.

6.4.2 *p*-dimensionale Ausreißer

Betrachtet man Ausreißer in dem Raum, der durch die ersten p Hauptkomponenten aufgespannt wird, so bietet es sich aus Gründen der Maßstabsvereinheitlichung wieder an, die normierten Scores $z_{i,j}$ für das Testproblem zu verwenden. Berechnet man den quadrierten p-dimensionalen euklidischen Abstand

$$d_p^2(z_i, 0) = \sum_{k=1}^{p} \left(\frac{y_{i,k}}{\sqrt{\lambda_k}}\right)^2$$

der normierten Scores einer Beobachtung i zu ihrem Mittelwert $0 \in \mathbb{R}^p$, so ist dieser Ausdruck eine Summe von Quadraten asymptotisch standardnormalverteilter Zufallsgrößen, und somit lässt sich die asymptotische Verteilung von $d_p^2(z_i, 0)$ als χ^2-Verteilung mit p Freiheitsgraden (kurz: χ_p^2-Verteilung) angeben. Damit ergibt sich zur Bestimmung des kritischen Wertes für $d_p^2(z_i, 0)$ analog zur obigen Berechnung, wenn wieder mit Konfidenzwahrscheinlichkeit α nur „echte“ Ausreißer erkannt werden sollen:

$$\begin{aligned}
& \mathbb{P}(\max_{1\le i\le n} d_p^2(z_i,0) > c) \overset{!}{=} \alpha \\
\iff \quad & \mathbb{P}(d_p^2(z_i,0) \le c \;\forall i = 1,\dots,n) = 1-\alpha \\
\underset{\text{Unabhängigkeit}}{\iff} \quad & \textstyle\prod_{i=1}^{n} \mathbb{P}(d_p^2(z_i,0) \le c) = 1-\alpha \\
\underset{\text{Verteilungsidentität}}{\iff} \quad & (\mathbb{P}(Z \le c))^n = 1-\alpha \text{ mit } Z \sim \chi_p^2 \\
\iff \quad & (F_{\chi_p^2}(c))^n = 1-\alpha \\
\iff \quad & F_{\chi_p^2}(c) = \sqrt[n]{(1-\alpha)} \\
\iff \quad & c = \chi^2_{p;\,\sqrt[n]{(1-\alpha)}} \quad .
\end{aligned}$$

Als kritischer Wert für $d_p^2(z_i, 0)$ wird also hier das $\sqrt[n]{(1-\alpha)}$-Quantil der χ^2-Verteilung mit p Freiheitsgraden gewählt und Punkten, deren quadrierter p-dimensionaler Euklidischer Abstand vom Nullpunkt diesen Wert c überschreitet, das Attribut „Ausreißer“ zugeordnet.

6.4.3 Schlecht modellierte Beobachtungen

Als letztes werden in diesem Abschnitt die Beobachtungen daraufhin untersucht, inwiefern sie durch p Hauptkomponenten schlecht modelliert werden. Hierzu beachte man zunächst, dass die Matrix V orthogonal und daher invertierbar mit $V^{-1} = V^\top$ ist. Damit ist es möglich, die Originalvariablen, also die Beobachtungsmatrix X, als Funktion der Scores zu schreiben:

$$Y = X \cdot V \underset{\text{V orthogonal}}{\Longleftrightarrow} X = Y \cdot V^\top \quad \text{mit } V \in \mathbb{R}^{m\times m} \text{ und } Y \in \mathbb{R}^{n\times m} \quad .$$

Verwendet man anstelle der vollständigen Matrizen $V \in \mathbb{R}^{m\times m}$ und $Y \in \mathbb{R}^{n\times m}$ nur jeweils deren erste p Spalten, so geht die rechte Seite der obigen Äquivalenz in die Gleichung

$$\hat{X} = \tilde{Y} \cdot \tilde{V}^\top \quad \text{mit } \tilde{V} \in \mathbb{R}^{m\times p} \text{ und } \tilde{Y} \in \mathbb{R}^{n\times p}$$

über, wobei $\hat{X}$ die Matrix der Projektion von X auf den durch die ersten p Hauptkomponenten $(C_1, \ldots, C_p)$ aufgespannten p-dimensionalen Teilraum des $\mathbb{R}^m$ ist.
Will man nun messen, wie gut ein Beobachtungsvektor $x_i, i = 1, \ldots, n$ durch das p-dimensionale Modell beschrieben wird, so bietet sich die Verwendung der Länge des Residuenvektors $x_i - \hat{x}_i$ an. Deswegen wird in [10] in Abschnitt 2.7.2 die dort als Q-Statistik bezeichnete Größe

$$Q_i = (x_i - \hat{x}_i)^\top \cdot (x_i - \hat{x}_i) \quad \text{mit } x_i \in \mathbb{R}^m \text{ Beobachtungsvektor}$$

zur Behandlung der erwähnten Fragestellung eingeführt. Aus Berechnungssicht ist es günstig, dass sich Q_i auch schreiben lässt als

$$Q_i = \sum_{k=p+1}^{m} y_{i,k}^2,$$

wobei y_i den Score-Vektor zum Beobachtungsvektor x_i bezeichnet.
Die Verteilung der so definierten Teststatistik Q_i lässt sich nur mühsam herleiten. In die Berechnung gehen im Wesentlichen die Summen der ersten drei Potenzen der $(m - p)$ kleinsten Eigenwerte der Korrelationsmatrix ein. In [10] wird das Ergebnis dieser Berechnung angegeben und es finden sich weitere Literaturhinweise für dieses Problem. Es ergibt sich, dass eine geschickte Transformation von Q_i zu einer möglichst guten Approximation der Standardnormalverteilung führt, so dass entsprechend transformierte Quantile der $\mathcal{N}(0, 1)$-Verteilungsfunktion als Schranken für Q_i verwendet werden können. Der in der Folge angegebene kritische Wert Q_α für die Teststatistik Q_i wurde der Arbeit von J. Edward Jackson [10] entnommen.

Setzt man

$$\begin{aligned}
\theta_1 &:= \sum_{k=p+1}^{m} \lambda_k \quad , \quad \theta_2 := \sum_{k=p+1}^{m} \lambda_k^2 \quad , \quad \theta_3 := \sum_{k=p+1}^{m} \lambda_k^3 \quad , \\
h_0 &:= 1 - \frac{2 \cdot \theta_1 \cdot \theta_3}{3 \cdot \theta_2^2} \quad , \\
c_\alpha &:= \text{sgn}(h_0) \cdot \Phi^{-1}(\sqrt[n]{1-\alpha}) \quad , \\
Q_\alpha &:= \theta_1 \cdot \left[\frac{c_\alpha \cdot \sqrt{2 \cdot \theta_2 \cdot h_0^2}}{\theta_1} + \frac{\theta_2 \cdot h_0 (h_0 - 1)}{\theta_1^2} + 1 \right]^{\frac{1}{h_0}} \quad ,
\end{aligned}$$

so erhält man

$$\mathbb{P}(Q_i > Q_\alpha) \underset{\text{approx.}}{=} \alpha \quad .$$

Will man die Beobachtungsvektoren $x_i, i = 1, \ldots, n$ daraufhin untersuchen, ob sie signifikant schlecht durch das p-dimensionale lineare Modell der ersten p Hauptkomponenten beschrieben werden, so gilt es zu überprüfen, ob der Wert von

$$Q_i = (x_i - \hat{x}_i)^\top \cdot (x_i - \hat{x}_i) = \sum_{k=p+1}^{m} y_{i,k}^2$$

das zugehörige transformierte $\mathcal{N}(0, 1)$-Quantil überschreitet.

7 Variablenselektion

7.1 Motivation

Im vorangegangenen Abschnitt wurde mit der Hauptkomponentenanalyse eine Methode vorgestellt, die Dimensionalität des zu untersuchenden Datenraumes zu reduzieren und dabei nur einen möglichst geringen Informationsverlust hinnehmen zu müssen. Dies ist zum Beispiel im Hinblick auf die Berechnungs- und Speicherplatzkomplexität von auf das Datenmaterial anzuwendenden Algorithmen günstig, denn damit ist es möglich, diese nur auf dem durch einige Hauptkomponenten gegebenen, niederdimensionalen Datenraum auszuführen und die gelieferten Ergebnisse dann eventuell so zu transformieren, dass wieder Informationen über die Ausgangsdaten gewonnen werden. Verfolgt man jedoch das Ziel, den Datenraum für einen Analysten überschaubar zu machen oder ist man daran interessiert, einige Variablen ganz aus der Betrachtung auszuklammern (also gegebenenfalls einige Laborexperimente gar nicht mehr durchführen zu müssen), so ergeben sich bei der Verwendung von Hauptkomponenten für diese Aufgaben zwei grundlegende Schwierigkeiten:

- Hauptkomponenten sind als Linearkombinationen der Originalvariablen wesentlich schlechter interpretierbar als die Ausgangsdaten, welche unmittelbar die experimentell erhobenen Messwerte wiedergeben.
- In die Hauptkomponenten gehen in der Regel alle Variablen mit einem festgelegten Gewichtsfaktor ein, es kann also kein Experiment fortgelassen werden.

Aus diesen Gründen ist es oftmals günstiger, anstelle von p Variablenkombinationen mit maximalem Informationsgehalt lieber p der m Originalvariablen selbst auszuwählen, die am repräsentativsten für das gesamte Datenmaterial sind, um dadurch die Dimension des zu betrachtenden Datenraums zu verringern. Dieses Vorgehen der Variablenselektion wird in diesem Abschnitt behandelt, und es werden Algorithmen erläutert, die verschiedene Auswahlkriterien für die zu selektierenden Variablen definieren, wobei in der Implementierung zwei generelle Vorgehensweisen umgesetzt worden sind:

1. Verfahren, die sich der Ergebnisse der Hauptkomponentenanalyse bedienen und
2. das Verfahren der Haupt-Variablen (principal variables) nach McCabe.

7.2 Auf Hauptkomponenten basierende Verfahren

Die von I.T. Jolliffe in [11] publizierten und in der GALA-Software implementierten, auf Hauptkomponenten basierenden Verfahren zur Variablenselektion arbeiten derart, dass ausgewählte Spalten der in Abschnitt 6.3 definierten Matrix V, welche die Eigenvektoren der der Hauptkomponentenanalyse zu Grunde gelegten Varianz- / Kovarianzmatrix bzw. Korrelationsmatrix enthält, auf ihre Maximaleinträge hin untersucht werden. Weist ein Spaltenvektor y_j von Y seinen größten Eintrag in Zeile i auf, wobei $1 \leq i, j \leq m$ gilt, so wird

(a) die Variable X_i ausgewählt, falls j einen kleinen Wert hat und X_i demnach mit einem großen Gewicht in die zugehörige Hauptkomponente C_j mit $j \ll m$ eingeht (Selektion).

(b) die Variable X_i als zu vernachlässigen angesehen und aus der Menge der zu untersuchenden Variablen entfernt, falls j nahe bei m liegt (Elimination).

Damit nun genau p Variablen bestimmt werden können, wird das Verfahren der Selektion oder der Elimination iterativ wiederholt, bis genau p Variablen selektiert bzw. genau $m-p$ Variablen eliminiert worden sind. Dabei ist es zum einen möglich, nach jedem Selektions- bzw. Eliminationsschritt eine neue Hauptkomponentenanalyse für die verbleibenden Variablen durchzuführen oder aber die Ergebnisse einer einzigen Hauptkomponentenanalyse für den gesamten Selektions- bzw. Eliminationsvorgang zu nutzen. Entscheidet man sich dazu, die p Variablen über das Verfahren der Selektion zu bestimmen und zudem für jeden Selektionsschritt eine neue Hauptkomponentenanalyse zu tätigen, so gilt es zu beachten, dass die Variablen in der Regel untereinander korreliert sind. Ist jedoch die Variable X_i einmal ausgewählt worden, so soll für die weiteren Auswahlschritte ihr Einfluss auf die

restlichen noch verbliebenen Variablen ausgeschaltet werden. Deswegen gilt es zu untersuchen, wie sich die partielle Varianz- / Kovarianzmatrix $K_{m-1,k}$ berechnet. $K_{m-1,k}$ beinhaltet hierbei die Varianzen/Kovarianzen der $m-1$ verbleibenden Variablen, wenn man die Variable X_k aus der Menge der Variablen $(X_1, \ldots, X_m)$ entfernt und ihren Einfluss auf die restlichen Variablen „herausrechnet".
Nehmen wir zur Lösung dieses Problems an, es soll die partielle Kovarianz zweier Zufallsvariablen X und Y berechnet werden, wobei der Einfluss einer dritten Zufallsvariable Z, die sowohl mit X als auch mit Y korreliert ist, ausgeschaltet werden soll. Dazu definieren wir:

$$X_Z := a_{x,z} + b_{x,z} \cdot Z \quad , \quad Y_Z := a_{y,z} + b_{y,z} \cdot Z$$

mit den Regressionskoeffizienten

$$b_{x,z} = \frac{\text{Kov}(X,Z)}{\text{Var}(Z)} \; , \quad b_{y,z} = \frac{\text{Kov}(Y,Z)}{\text{Var}(Z)} \quad .$$

X_Z bzw. Y_Z bezeichnen den Anteil der Variablen X bzw. Y, der durch ihren linearen Bezug zu Z gegeben ist. Es gilt

$$\begin{aligned}
&\text{Kov}((X - X_Z), (Y - Y_Z)) \\
&= \text{Kov}((X - (a_{x,z} + b_{x,z} \cdot Z)), (Y - (a_{y,z} + b_{y,z} \cdot Z))) \\
&= \text{Kov}(X,Y) - \text{Kov}(X, a_{y,z} + b_{y,z} \cdot Z) - \text{Kov}(Y, a_{x,z} + b_{x,z} \cdot Z) \\
&\quad + \text{Kov}(a_{x,z} + b_{x,z} \cdot Z, a_{y,z} + b_{y,z} \cdot Z) \\
&= \text{Kov}(X,Y) - b_{y,z} \cdot \text{Kov}(X,Z) - b_{x,z} \cdot \text{Kov}(Y,Z) + b_{x,z} \cdot b_{y,z} \cdot \text{Var}(Z) \quad .
\end{aligned}$$

Beachtet man die Definitionen von $b_{x,z}$ und $b_{y,z}$, so vereinfacht sich der Ausdruck weiter:

$$\begin{aligned}
&\text{Kov}((X - X_Z), (Y - Y_Z)) \\
&= \text{Kov}(X,Y) - 2 \cdot \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} + \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} \\
&= \text{Kov}(X,Y) - \frac{\text{Kov}(X,Z) \cdot \text{Kov}(Y,Z)}{\text{Var}(Z)} \quad .
\end{aligned}$$

Es gilt also für die partielle Varianz- / Kovarianzmatrix:

$$K_{m-1,k} = K'_{m,k} - \left(\frac{1}{\text{Var}(X_k)} \cdot K^{(k)} \cdot K^{(k)^\top} \right)'_{m,k}$$

mit

$K_{m-1,k}$	partielle Varianz- / Kovarianzmatrix ohne Einfluss von X_k,
$K^{(k)}$	k-te Spalte der Matrix K,
$M'_{m,k}$	Streichungsmatrix, die aus einer $(m \times m)$-Matrix M durch Streichung von Zeile k und Spalte k hervorgeht.

Aus Sicht des Algorithmus ist hierbei natürlich zu beachten, dass die Indizes der verbleibenden Variablen verschoben werden, wenn $k \neq m$ gilt. In den folgenden Abschnitten wird nun kurz für die vier auf Hauptkomponenten basierenden Verfahren, die im Rahmen dieses Projekts umgesetzt worden sind, das jeweils zugehörige Rekursionsschema angegeben.

7.2.1 Selektion mit *p* Hauptkomponentenanalysen

Zunächst wird eine Hauptkomponentenanalyse der Originalvariablen $(X_1, \ldots, X_m)$ auf Basis der Varianz- / Kovarianzmatrix K bzw. (standardisierte Größen) der Korrelationsmatrix R durchgeführt, um die Matrix V der Eigenvektoren zu erhalten. Zudem wird die Anzahl p auszuwählender Variablen vorgegeben. Die selektierten Variablen $(X_{i_1}, \ldots, X_{i_p})$ werden nun dadurch festgelegt, dass zunächst

$$i_1 = \underset{1 \leq i \leq m}{\operatorname{argmax}}\, v_{i,1}$$

gewählt wird. Die Variable X_{i_1} ist hiermit ausgewählt und, wie oben beschrieben, soll ihr Einfluss auf die restlichen Variablen für den nächsten Selektionsschritt ausgeschaltet werden. Es wird also die partielle Varianz- / Kovarianzmatrix K_{m-1,i_1} der verbleibenden $m-1$ Variablen gebildet und auf ihrer Basis eine erneute Hauptkomponentenanalyse durchgeführt. Nun wird wieder der Maximaleintrag des zum größten Eigenwert gehörenden Eigenvektors von K_{m-1,i_1} gesucht, die entsprechende Zeile mit i_2 bezeichnet und die Variable X_{i_2} dem Satz ausgewählter Variablen hinzugefügt. Dies wiederholt sich solange, bis p selektierte Variablen feststehen.

7.2.2 Selektion mit genau einer Hauptkomponentenanalyse

Will man den nicht unerheblichen Berechnungsaufwand, der sich durch die p zu tätigenden Hauptkomponentenanalysen der in Abschnitt 7.2.1 vorgestellten Methode ergibt, vermeiden, so kann die Selektion von p Variablen auch auf der Basis einer einzigen Hauptkomponentenanalyse erfolgen. Zunächst ergibt sich die Matrix $V = (v_{i,j})$ der Eigenvektoren analog zu dem Vorgehen in Abschnitt 7.2.1. Dann werden jedoch alle Spaltenvektoren $v_j,\ j = 1, \ldots, p$ von V auf ihren Maximaleintrag hin untersucht und die Indizes $i_1, \ldots, i_p$ der selektierten Variablen ergeben sich zu:

$$\begin{aligned}
i_1 &= \underset{1 \leq i \leq m}{\operatorname{argmax}}\, v_{i,1}\ , \\
i_2 &= \underset{1 \leq i \leq m : i \neq i_1}{\operatorname{argmax}}\, v_{i,2}\ , \\
&\vdots \qquad\quad \vdots \\
i_p &= \underset{1 \leq i \leq m : i \notin \{i_1, \ldots, i_{p-1}\}}{\operatorname{argmax}}\, v_{i,p}\quad .
\end{aligned}$$

Hier ist also die Berechnung der partiellen Varianz- / Kovarianzmatrizen vom Algorithmus her nicht gefordert. Für die Darstellung der Ergebnisse (vgl. Abschnitt 7.5) sind jedoch die Restvarianzen erforderlich, so dass auch hier die entsprechenden Größen berechnet wurden.

7.2.3 Elimination mit *(m-p)* Hauptkomponentenanalysen

Diesem Verfahren liegt die umgekehrte Argumentationsrichtung gegenüber dem Verfahren aus Abschnitt 7.2.1 zu Grunde. Es werden nicht die Variablen, die stark in die ersten Hauptkomponenten eingehen, ausgewählt, sondern diejenigen, die sich am besten mit den letzten Hauptkomponenten identifizieren lassen, eliminiert. Hier werden also Indizes $(i_1, \ldots, i_{m-p})$ bestimmt, so dass die Variablen $(X_{i_1}, \ldots, X_{i_{m-p}})$ vernachlässigt und die verbleibenden Variablen mit einem Index i^*, für den $i^* \notin \{i_1, \ldots, i_{m-p}\}$ gilt, als ausgewählt angesehen werden. Es wird also i_1 als

$$i_1 = \underset{1 \leq i \leq m}{\operatorname{argmax}} \, v_{i,m}$$

festgelegt. Danach wird die Varianz- / Kovarianzmatrix K'_{m,i_1} der verbleibenden $m-1$ Variablen, die sich aus der ursprünglichen Varianz- / Kovarianzmatrix K durch Streichung von Zeile i_1 und Spalte i_1 ergibt, für eine neue Hauptkomponentenanalyse benutzt und es werden iterativ die restlichen $m-p-1$ Indizes bestimmt, deren zugehörige Variablen eliminiert werden sollen. Es ist anzumerken, dass in dem Regelfall $p \ll m$ dieses Verfahren den bei weitem größten Rechenaufwand aller hier diskutierten Methoden verursacht.

7.2.4 Elimination mit genau einer Hauptkomponentenanalyse

Das vierte angewendete Verfahren verwirklicht die Grundidee des vorangegangenen, wobei jedoch auf eine erneute Hauptkomponentenanalyse pro Eliminationsschritt verzichtet wird. Die Indizes $(i_1, \ldots, i_{m-p})$ berechnen sich damit in Analogie zu der in Abschnitt 7.2.2 gewählten Vorgehensweise zu:

$$\begin{aligned}
i_1 &= \underset{1 \leq i \leq m}{\operatorname{argmax}} \, v_{i,m} \\
i_2 &= \underset{1 \leq i \leq m : i \neq i_1}{\operatorname{argmax}} \, v_{i,m-1} \\
&\vdots \qquad\qquad \vdots \\
i_{m-p} &= \underset{1 \leq i \leq m : i \notin \{i_1, \ldots, i_{m-p-1}\}}{\operatorname{argmax}} \, v_{i,p+1} \quad .
\end{aligned}$$

7.3 Verfahren der Principal Variables

Das von McCabe in [15] beschriebene Verfahren der Haupt-Variablen (in Anlehnung an den Begriff der Hauptkomponenten) stützt sich nicht auf die Ergebnisse einer Hauptkomponentenanalyse, sondern versucht, die Optimalitätseigenschaft der Hauptkomponenten direkt auf einzelne Variablen zu übertragen. Diese Optimalitätseigenschaft besteht (vgl. hierzu Abschnitt 6.2) darin, dass die erste Hauptkomponente C_1 eine möglichst große Varianz besitzt und damit die Summe der Restvarianzen minimiert. Wie in Abschnitt 6.4 beschrieben, lässt sich der Anteil der durch eine Hauptkomponente C_j erklärten Varianz einer Variable X_i ausdrücken durch $\mathrm{Var}(X_i) \cdot \rho^2(X_i, C_j)$. Setzt man nun anstelle von C_j eine Originalvariable X_j ein und fordert, dass die Summe der Restvarianzen der Variablen $\{X_i : i \neq j\}$ minimiert oder invers formuliert die Summe der durch X_j erklärten Varianz der restlichen Variablen maximiert wird, so führt dies auf die folgende Maximierungsaufgabe:

$$\begin{aligned} j_1 &= \operatorname*{argmax}_{1\leq j\leq m} \sum_{i=1}^{m} \mathrm{Var}(X_i) \cdot \rho^2(X_i, X_j) \\ &= \operatorname*{argmax}_{1\leq j\leq m} \sum_{i=1}^{m} \frac{\mathrm{Kov}^2(X_i, X_j)}{\mathrm{Var}(X_j)} \\ &= \operatorname*{argmax}_{1\leq j\leq m} \frac{1}{\mathrm{Var}(X_j)} \cdot \sum_{i=1}^{m} \mathrm{Kov}^2(X_i, X_j) \quad . \end{aligned} \tag{10}$$

Die zu dem so bestimmten Index j_1 gehörende Variable X_{j_1} wird in diesem Verfahren als erste ausgewählt. Anschließend wird ihr Einfluss auf die restlichen Variablen wieder herausgerechnet, indem die Quadrate der Einträge der partiellen Varianz- / Kovarianzmatrix K_{m-1,j_1} in der Maximierungsaufgabe (10) betrachtet werden, und so bestimmen sich iterativ die übrigen Indizes $(j_2, \ldots, j_p)$ für die Variablenselektion. In der praktischen Verwendung bietet sich dieses Verfahren auf Grund seiner einfachen Implementierbarkeit und geringen Ressourceninanspruchnahme an. Da sich im Falle von Daten aus der pharmazeutischen Forschung gezeigt hat, dass die dabei erzielten Ergebnisse qualitativ nicht hinter denen zurückbleiben, welche sich bei Anwendung der in Abschnitt 7.2 vorgestellten, aufwändigeren Methoden ergeben, erweist sich das Verfahren von McCabe als überlegen im Kosten- / Nutzenvergleich.

7.4 Anwendung

In Analogie zu der Darstellung der Ergebnisse der Hauptkomponentenanalyse werden die Resultate der Variablenselektion, wie in Tabelle 3 gezeigt, wiedergegeben, wobei folgende Vereinbarungen getroffen werden:

j	Iteration
i_j	Nummer der in Iteration j ausgewählten Variable
η_j	durch X_{i_j} erklärte Varianz
$\text{Var}_{\text{Rest},j}(X_k)$	Restvarianz von Variable X_k nach Iteration j.

j	i_j	η_j	$\eta_{j-1} - \eta_j$	η_j/m	$\sum_{k=1}^{j} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},j}(X_k)$
1	i_1	η_1	-	η_1/m	η_1/m	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},1}(X_k)$
2	i_2	η_2	$\eta_1 - \eta_2$	η_2/m	$(\eta_1 + \eta_2)/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},2}(X_k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
p	i_p	η_p	$\eta_{p-1} - \eta_p$	η_p/m	$\sum_{k=1}^{p} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$

Tabelle 3: Schematische Darstellung von Ergebnissen bei der Variablenselektion.

7.5 Ergebnisse

Abschließend werden in Tabelle 4 anhand eines Beispieldatensatzes aus der pharmazeutischen Forschung mit $m = 100$ Variablen die Ergebnisse der implementierten Variablenselektionsverfahren gegenübergestellt.

Abschnitt	p	η_p	$\sum_{k=1}^{p} \eta_k/m$	$\max_{k\in\{1,\ldots,m\}} \text{Var}_{\text{Rest},p}(X_k)$
7.2.1	22	0,761	0,903	0,638
7.2.1	45	0,085	0,995	0,046
7.2.2	24	0,727	0,905	0,507
7.2.2	66	0,0084	0,999	0,0497
7.2.3	29	0,881	0,9004	0,328
7.2.3	46	0,3496	0,991	0,0387
7.2.4	25	1,049	0,901	0,6939
7.2.4	53	0,1101	0,9984	0,0169
7.3	21	0,7963	0,9007	0,5983
7.3	44	0,0993	0,9932	0,0469

Tabelle 4: Ergebnisse verschiedener Variablenselektionsverfahren für einen Beispieldatensatz.

In der jeweils ersten Zeile zu jedem Verfahren wurde p so gewählt, dass der kumulierte

Anteil erklärter Streuung über $0,9$ liegt; die zweite Zeile gehört zu dem Wert von p, ab welchem jede Restvarianz den Schwellenwert von 5 % unterschreitet.

8 Robuste statistische Verfahren

8.1 Einleitung

Wie die vorangegangenen Abschnitte zeigen, kann man bei der Betrachtung des Datenmaterials zwei grundlegende Aufgabenbereiche unterscheiden. Zum einen sollte man Regelmäßigkeiten und Abhängigkeiten innerhalb des Datenmaterials erkennen können, zum anderen sollten Ausreißer identifiziert und ihr Einfluss auf die gewonnenen Informationen minimiert werden. Ausreißer können verschiedene Ursachen haben:

- Es wurden fehlerhafte Messungen durchgeführt.
- Einige wenige Beobachtungen besitzen überdurchschnittlich günstige oder ungünstige Eigenschaften.

Die bisher vorgestellten statistischen Verfahren zum Data Mining haben die Eigenschaft, dass ihre Ergebnisse sehr stark von Ausreißern beeinflusst werden und die so gewonnenen Informationen nur eine geringe bis keine Aussagekraft besitzen. An diesem Punkt angelangt, bieten sich zwei Lösungsmöglichkeiten an. Entweder nutzt man Verfahren zur Ausreißererkennung und eliminiert diese, bevor weitere Informationen gewonnen werden, oder es sollten Verfahren genutzt werden, die weitgehend unempfindlich gegenüber Ausreißern sind. Genau an dieser Stelle werden robuste Verfahren wirksam. Sie erlauben, Aussagen über das Datenmaterial zu treffen, die nicht von grenzwertigen Beobachtungen verfälscht werden. Im Verlauf des GALA-Projektes war es ein wichtiges Ziel, robuste Verfahren der Datenanalyse zu implementieren, damit sie wahlweise zu den klassischen Methoden genutzt werden können. Im Folgenden wird ein theoretischer Überblick über robuste Verfahren gegeben. Dabei wird auf univariate und multivariate Problemstellungen eingegangen, und es werden Kriterien vorgestellt, die eine qualitative Beurteilung dieser Verfahren ermöglichen.

8.2 Maßzahlen für Lage und Streuung einer Stichprobe im univariaten Fall

Hat man eine Beobachtungsreihe mit Datenmaterial vorliegen, so möchte man diese Daten meist mit wenigen charakteristischen Größen kennzeichnen. Häufig verwendete Charakteristika sind die Lage und die Streuung der Daten. Eine Lagemaßzahl beschreibt in geeigneter Weise das Zentrum der beobachteten Werte $x_1, ..., x_n$, wohingegen eine Streuungsmaßzahl angibt, wie die Daten von einem Zentrum im Mittel abweichen. Einige dieser Lage- und Streuungsmaßzahlen sollen im Folgenden vorgestellt werden.

8.2.1 Lagemaßzahlen

Aufgrund der leichten Berechenbarkeit erscheint das arithmetische Mittel als ein einfaches und leicht anzuwendendes Hilfsmittel, um das Zentrum einer Beobachtungsreihe zu beschreiben. Untersucht man das arithmetische Mittel hingegen auf seine Empfindlichkeit gegenüber Ausreißern, also gegenüber Werten, die sehr weit entfernt von diesem berechneten Zentrum liegen, so stellt man schnell fest, dass das arithmetische Mittel nicht das optimale Werkzeug zur Beschreibung der Lage ist. Das folgende Beispiel aus [7] soll dies verdeutlichen.
Von einer Holzlatte der Länge 5m werden nacheinander fünf Stücke der Länge 1m abgesägt. Anschließend werden die Abweichungen x_i von der Solllänge 1m gemessen und in mm angegeben, d.h. $x_i = (\xi_i - 1) \cdot 10^3$, wobei ξ_i die Länge des i-ten Stückes in m angibt. Dabei ergeben sich folgende Werte für $x_1, \ldots, x_5$:

$$-0,5;\ 1,5;\ -1,0;\ 0,5;\ -9,5 \quad .$$

Berechnet man das arithmetische Mittel auf der Grundlage aller fünf Beobachtungen, so ergibt sich $\overline{x}_5 = -1,8$. Dieser Durchschnittswert legt die Vermutung nahe, dass die Säge bei jedem abgesägten Stück einen Fehler von ca. $1,8$mm macht. Die ersten vier Lattenstücke wurden jedoch annähernd korrekt mit der gewünschten Länge von 1m gesägt. Da aber bei jedem Sägevorgang 2mm durch Sägespäne verloren gingen, kann das letzte Lattenstück keine Länge von 1m mehr haben und weist die angegebene relativ große Abweichung vom Sollwert auf. Nimmt man hingegen nur die ersten vier Werte zur Berechnung des Mittels, dann ergibt sich ein Wert für das arithmetische Mittel von $\overline{x}_4 = 0,125$. Dieser Wert zeigt offensichtlich eine adäquatere Beschreibung der Sachlage.
Bei diesem einfachen Beispiel war es unproblematisch, den Ausreißerwert zu erkennen und für weitere Berechnungen zu eliminieren. In den allermeisten Fällen ist diese Identifikation jedoch nicht so schnell und sicher zu bewerkstelligen, und man sollte daher zu einer Lagemaßzahl übergehen, die nicht so empfindlich auf extreme Werte in den Rohdaten reagiert. Vorbereitend wird dazu zunächst der Begriff der geordneten Stichprobe eingeführt. Liegen n Beobachtungen $x_1, \ldots, x_n$ einer Stichprobe vom Umfang n vor und werden die Daten der Größe nach sortiert, so bezeichnet $x_{(i)}$ den i-kleinsten Wert, also insbesondere

$$x_{(1)} = \min_{1 \le i \le n} x_i, \qquad x_{(n)} = \max_{1 \le i \le n} x_i \quad .$$

Die der Größe nach sortierte Reihe

$$x_{(1)} \le x_{(2)} \le \ldots \le x_{(n-1)} \le x_{(n)}$$

heißt geordnete Stichprobe von $x_1, \ldots, x_n$, und $x_{(i)}$ wird die i-te Ordnungsgröße genannt. Damit lassen sich nun weitere Lagemaßzahlen angeben.
Das empirische α-Quantil $\tilde{x}_\alpha$ einer Stichprobe $x_1, ..., x_n$ vom Umfang n ist definiert als

$$\tilde{x}_\alpha = \begin{cases} x_{([n \cdot \alpha]+1)} & \text{, falls } n \cdot \alpha \text{ keine ganze Zahl ist} \\ \frac{1}{2}\left(x_{(n \cdot \alpha)} + x_{(n \cdot \alpha + 1)}\right) & \text{, falls } n \cdot \alpha \text{ eine ganze Zahl ist} \end{cases} .$$

Als Spezialfall stellt sich der empirische Median dar. Er ist gegeben durch

$$m_n := \tilde{x}_{0,5} = \begin{cases} x_{\left(\frac{n+1}{2}\right)} & \text{, falls } n \text{ ungerade} \\ \frac{1}{2}\left(x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n+2}{2}\right)}\right) & \text{, falls } n \text{ gerade} \end{cases} ,$$

wenn $x_1, ..., x_n$ eine Stichprobe vom Umfang n ist.
Betrachtet man nun noch einmal das Beispiel der Holzlatte und berechnet den empirischen Median, so erhält man als geordnete Stichprobe zunächst

$$x_{(1)} = -9,5;\ x_{(2)} = -1,0;\ x_{(3)} = -0,5;\ x_{(4)} = 0,5;\ x_{(5)} = 1,5 \quad .$$

Damit ergibt sich der Median zu $m_5 = \tilde{x}_{0,5} = -0,5$, welcher eine „robustere“ Beschreibung der Situation liefert als das arithmetische Mittel.
Bei symmetrisch verteilten Daten ist das midquartile ein geeignetes Lagemaß. Es wird definiert als

$$\frac{1}{2}\left(\tilde{x}_{0,25} + \tilde{x}_{0,75}\right) \quad .$$

Im Beispiel ergibt sich so

$$\frac{1}{2}\left(\tilde{x}_{0,25} + \tilde{x}_{0,75}\right) = \frac{1}{2}(x_{(2)} + x_{(4)}) = -0,25 \quad .$$

Diese kleine Auswahl an möglichen Lagemaßzahlen zeigt schon am simplen Beispiel, wie unterschiedlich auf Ausreißer reagiert wird, und es wird deutlich, dass man bei der Untersuchung experimentell oder auch pseudo-zufällig erzeugter Daten am Rechner auf Verfahren zurückgreifen sollte, die möglichst unempfindlich gegenüber Ausreißern sind.

8.2.2 Streuungsmaßzahlen

Das wohl bekannteste Verfahren zur Bestimmung der Streuung innerhalb einer Stichprobe ist die empirische Standardabweichung. Für eine Beobachtungsreihe $x_1, ..., x_n$ vom Stichprobenumfang n ist sie definiert als

$$s_n = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \overline{x}_n\right)^2} \quad .$$

Die quadrierte empirische Standardabweichung wird empirische Varianz genannt. Angewendet auf die Daten aus dem einleitenden Beispiel, erhält man hier für den arithmetischen Mittelwert $\overline{x}_5 = -1,8$ eine empirische Standardabweichung von $s_5 = 4,41$, bei $\overline{x}_4 = 0,125$ ergibt sich ein Wert von $s_4 = 1,11$. Die Werte des Beispiels zeigen, dass auch die empirische Standardabweichung empfindlich auf Ausreißerwerte reagiert. Dies lässt sich damit begründen, dass große, stark vom Zentrum abweichende Werte durch das

Quadrieren bei der Berechnung der empirischen Standardabweichung einen noch größeren Einfluss gewinnen und so die Streuung der Daten verfälscht wiedergeben. Auch die Verwendung einer ausreißerunempfindlicheren Lagemaßzahl kann dies nicht unterbinden, da das arithmetische Mittel $\bar{x}_n$ gerade der Wert ist, der den Ausdruck $\sum(x_i - a)^2$ für a minimiert.
Eine gegenüber Ausreißern robustere Streuungsmaßzahl ist der Interquartilabstand, mit dem man die Differenz der beiden Größen $\tilde{x}_{0,75}$ und $\tilde{x}_{0,25}$ bezeichnet, also

$$\text{IQR}_n = \tilde{x}_{0,75} - \tilde{x}_{0,25} \quad .$$

Auf den Begriff des Interquartilabstandes wird in Abschnitt 8.4.2 noch näher eingegangen. Für die Stichprobe des Beispiels ergibt sich $\text{IQR}_5 = 1,5$. Dieser Wert zeigt einen robusteren Umgang mit dem Ausreißerwert von $x_{(1)} = -9,5$ und ist daher hier besser geeignet, die Streuung der Daten um ein Lagezentrum zu beschreiben, als der der empirischen Standardabweichung.

8.3 Kriterien für Robustheit

Wie man in dem vorangegangenen Abschnitt an einfachen Beispielen bereits gesehen hat, reagieren unterschiedliche Verfahren zur Bestimmung von Lage und Streuung unterschiedlich auf Ausreißer. Was man in diesem Zusammenhang unter einem robusten Verfahren versteht, ist intuitiv klar, aber wie kann man diese Robustheit mathematisch beschreiben und überprüfen? Dieser Frage soll nun nachgegangen werden. Als Kriterien zur Beurteilung von Robustheit werden im Folgenden die Sensitivitätskurve, die Einflusskurve und der Bruchpunkt definiert.
Bezeichne im Weiteren

$$T_n = T_n(x_1, \ldots, x_n)$$

eine Stichprobenfunktion, auch Statistik genannt, zur Stichprobe $x_1, \ldots, x_n$ vom Umfang n. Beispiele für Stichprobenfunktionen sind das arithmetische Mittel

$$T_n(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} x_i \quad ,$$

der empirische Median

$$T_n(x_1, \ldots, x_n) = \begin{cases} x_{((n+1)/2)} & \text{, falls } n \text{ ungerade} \\ \frac{1}{2}(x_{(n/2)} + x_{((n+2)/2)}) & \text{, falls } n \text{ gerade} \end{cases}$$

und die empirische Standardabweichung

$$T_n(x_1, \ldots, x_n) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)^2} \quad .$$

8.3.1 Die Sensitivitätskurve

Sei $T_n = T_n(x_1, \ldots, x_n)$ eine Stichprobenfunktion basierend auf n Beobachtungen. Dann ist die Sensitivitätskurve definiert als

$$\mathrm{SC}(x; x_1, \ldots, x_{n-1}, T_n) = n \cdot (T_n(x_1, \ldots, x_{n-1}, x) - T_{n-1}(x_1, \ldots, x_{n-1})) \quad .$$

Die Sensitivitätskurve beschreibt die Reaktion einer Stichprobenfunktion, falls zu einer Stichprobe $x_1, \ldots, x_{n-1}$ eine weitere Beobachtung x hinzugefügt wird. Ihr Wert entspricht der Differenz von T_n und T_{n-1} multipliziert mit der Anzahl n der Beobachtungen insgesamt. Das folgende Beispiel veranschaulicht dies.

Die Daten

$$n = 5;\ x_1 = -0,5;\ x_2 = 1,5;\ x_3 = -1,0;\ x_4 = 0,5$$

ergeben:

1. Sensitivitätskurve des arithmetischen Mittels

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_4, T_5) &= 5 \cdot \left(\frac{1}{5}(x_1 + \ldots + x_4 + x) - \frac{1}{4}(x_1 + \ldots + x_4) \right) \\ &= 5 \cdot (0,1 + 0,2x - 0,125) \\ &= x - 0,125 \end{aligned}$$

2. Sensitivitätskurve des Medians

$$\mathrm{SC}(x; x_1, \ldots, x_4, T_5) = \begin{cases} -2,5 & \text{, falls } x < -0,5 \\ 5x & \text{, falls } -0,5 \leq x \leq 0,5 \\ 2,5 & \text{, falls } 0,5 < x \end{cases}$$

In Abbildung 11 zeigt sich am Steigungsverhalten der Kurven deutlich, dass die Hinzunahme eines weiteren beliebigen Wertes x den Wert der Sensitivitätskurve des arithmetischen Mittels und damit auch den Wert für das Zentrum der Datenpunkte so verändern kann, dass der neu berechnete Wert für die Lage der Daten ein stark verfälschtes Bild der Stichprobe liefert. Dies zeigt sich darin, dass die Sensitivitätskurve des arithmetischen Mittels unbeschränkt ist. Im Gegensatz dazu ist die Sensitivitätskurve des Medians beschränkt, so dass auch ein zusätzlicher weit außerhalb des Zentrums liegender Beobachtungswert die geschätzte Lage des Zentrums nicht maßgeblich verändern kann.

Bevor nun weitere Kriterien zur Beurteilung von Robustheitseigenschaften von Maßzahlen eingeführt werden, wird die bisherige Betrachtungsweise aus der Sicht der rein beschreibenden (deskriptiven) Statistik verlassen und das für das weitere Vorgehen zu Grunde liegende mathematische Modell vorgestellt [14]. Dabei bezeichne X eine Zufallsvariable

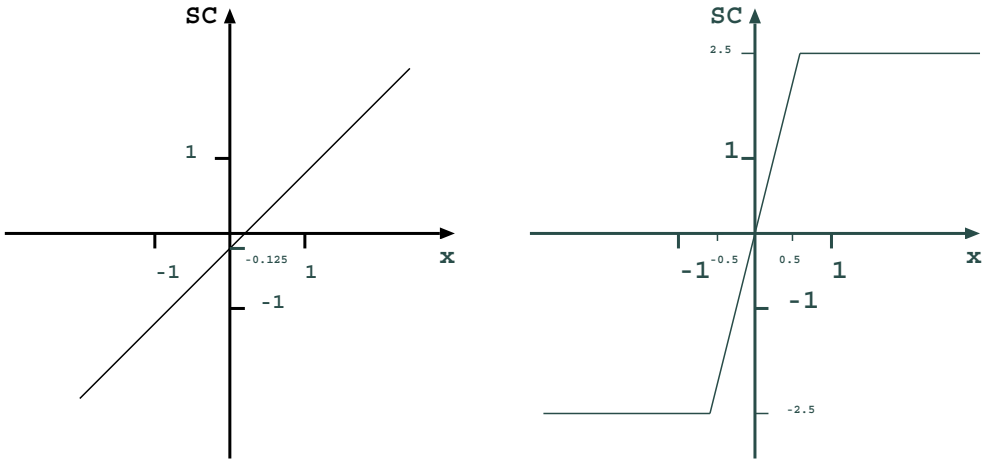


Abbildung 11: Sensitivitätskurven für das arithmetische Mittel (links) und den Median (rechts).

(Zufallsgröße), die auf einem geeigneten Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$ als eine Abbildung von Ω in die Menge der reellen Zahlen definiert ist, und x bezeichne im Folgenden eine Realisation einer Zufallsvariablen X. Des Weiteren sei

$$F : \mathbb{R} \to [0;1] \quad \text{mit} \quad F(x) := \mathbb{P}(X \leq x) \qquad \forall x \in \mathbb{R}$$

die Verteilungsfunktion der Zufallsvariablen X mit den bekannten Eigenschaften. Um sich bei unbekannter Verteilungsfunktion trotzdem ein Bild von der Verteilung der Zufallsgröße X machen zu können, ist die Konstruktion der empirischen Verteilungsfunktion F_n hilfreich. Für eine unabhängige Folge $X_1, \ldots, X_n$ von Zufallsgrößen mit der Verteilungsfunktion F gibt

$$F_n(t, \omega) := \frac{1}{n} \sum_{i=1}^{n} 1_{(-\infty;t]} \left(X_i(\omega) \right)$$

die relative Häufigkeit der X_i mit Werten $\leq t$ an, wobei mit $1_{(-\infty,t]}$ die Indikatorfunktion [1] des Intervalls $(-\infty, t]$ bezeichnet wird. Außerdem heißt eine Stichprobenfunktion, deren Wert einen unbekannten Parameter der den Beobachtungen zu Grunde liegenden Verteilung schätzt, ein Schätzer bzw. eine Schätzfunktion. Alle bisher betrachteten Stichprobenfunktionen sind Schätzer von Lage- bzw. Streuungsparametern der zu Grunde liegenden Verteilung.

8.3.2 Die Einflusskurve

Bei der Berechnung der Sensitivitätskurve stellt man fest, dass diese nicht ausschließlich von der jeweiligen Stichprobenfunktion $T_n(\cdot)$ abhängt, sondern vor allem auch von

[1] $1_A(\omega) = \begin{cases} 1 \text{ , falls } \omega \in A \\ 0 \text{ , falls } \omega \notin A \end{cases}$

den konkret vorliegenden Stichprobendaten und deren Anzahl. Man kann versuchen, diese Abhängigkeiten zu eliminieren, indem man eine Grenzwertbetrachtung für $n \to \infty$ durchführt.
Man betrachte als Beispiel die SC-Kurve des arithmetischen Mittels:

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_{n-1}, \bar{x}_n) &= n \cdot \left[\frac{1}{n} \left(\sum_{i=1}^{n-1} x_i + x \right) - \frac{1}{n-1} \sum_{i=1}^{n-1} x_i \right] \\ &= \sum_{i=1}^{n-1} x_i + x - \frac{n}{n-1} \sum_{i=1}^{n-1} x_i \\ &= -\frac{1}{n-1} \sum_{i=1}^{n-1} x_i + x \quad . \end{aligned}$$

Sind die Beobachtungen unabhängige Realisationen einer Zufallsgröße mit existierendem Erwartungswert μ, dann konvergiert aufgrund des Starken Gesetzes Großer Zahlen (SGGZ) die Sensitivitätskurve SC fast sicher [2] gegen $x - \mu$.

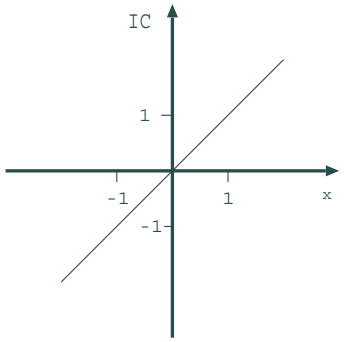


Abbildung 12: Einflusskurve des arithmetischen Mittels für $\mu = 0$.

Betrachtet man jetzt allgemein eine Stichprobenfunktion, die wie das arithmetische Mittel eine Funktion der empirischen Verteilungsfunktion F_n ist, also $T_n = T(F_n)$, dann ergibt sich für die SC-Kurve:

$$\begin{aligned} \mathrm{SC}(x; x_1, \ldots, x_{n-1}, T_n) &= n \left[T\left(\frac{n-1}{n} F_{n-1} + \frac{1}{n} \delta_x \right) - T(F_{n-1}) \right] \\ &= \frac{T\left(\frac{n-1}{n} F_{n-1} + \frac{1}{n} \delta_x \right) - T(F_{n-1})}{\frac{1}{n}} \quad , \end{aligned}$$

wobei δ_x die sogenannte Einheitsmasse im Punkt x ist. Für $n \to \infty$ strebt F_{n-1} gegen F, und man erhält für ein hinreichend reguläres T, von dem man voraussetzt, dass es auch in

[2] Der Begriff der fast sicheren Konvergenz wird auf Seite 75 definiert.

F definiert ist, in vielen Fällen [6]:

$$\mathrm{SC}(x; X_1, \ldots, X_{n-1}, T_n) \xrightarrow{f.s.} \lim_{\varepsilon \to 0+} \frac{T((1-\varepsilon)F + \varepsilon\delta_x) - T(F)}{\varepsilon} \quad . \tag{11}$$

In diesem Fall ist $T(F_n)$ ein natürlicher Schätzwert für $T(F)$. Der Grenzwert in (11) wird als die Einflusskurve (influence curve) bezeichnet:

$$\mathrm{IC}(x; F, T) = \lim_{\varepsilon \to 0+} \frac{T((1-\varepsilon)F + \varepsilon\delta_x) - T(F)}{\varepsilon} \quad .$$

Die Einflusskurve gibt also Auskunft über die Reaktion einer Schätzfunktion auf einen hinzugefügten Wert unter der Annahme, dass die Stichprobe aus der zu Grunde liegenden Verteilung von großem Umfang ist. Da bei der Grenzwertbildung wegen des Starken Gesetzes Großer Zahlen die Stichprobe nicht mehr betrachtet wird, hängt die Einflusskurve nur noch von x ab und wird daher auch als lokale Kenngröße bezeichnet. Außerdem kann man den maximalen Einfluss, den eine einzelne Beobachtung auf den zu schätzenden Wert ausüben kann, mit dem maximalen Betrag der Einflusskurve angeben. Diese Größe wird als Ausreißerempfindlichkeit (Gross-Error Sensitivity) bezeichnet. Untersucht man das arithmetische Mittel und den Median hinsichtlich dieser Größe, so stellt man fest, dass das arithmetische Mittel im Gegensatz zum Median einen unendlichen Wert annehmen kann und deshalb bei ausreißerbehafteten Daten nicht geeignet ist. Auch die Steigung der Einflusskurve kann man im Bezug auf die Robustheit interpretieren. Sie liefert einen Anhaltspunkt für das Verhalten der Schätzfunktion bei nur geringen Änderungen, wie sie zum Beispiel bei Rundungen oder Klassenbildung der Rohdaten auftreten können. Solche Änderungen haben einen stärkeren Einfluss auf den zu schätzenden Wert, je steiler die Steigung der Einflusskurve ist. Je größer der maximale Wert des Absolutbetrages der Steigung (Local-Shift Sensitivity) ist, desto stärker können solche Änderungen auf den Schätzwert durchschlagen. Für den Median ergibt sich hier ein unendlicher Wert, während die Größe beim arithmetischen Mittel den endlichen Wert 1 annimmt.
Wie in Abschnitt 8.5 noch genauer erläutert wird, gibt es auch robuste Schätzer, die weit außerhalb liegende Stichprobenwerte vollständig ignorieren, d.h. die Einflusskurve hat ab einem bestimmten Abstand vom zu schätzenden Lageparameter den Wert Null (Beispiele in Abschnitt 8.5.2). Der Grenzwert, der die IC-Kurve definiert, kann als Ableitung des Funktionals T interpretiert werden. Man bezeichnet diese Ableitung als Gâteaux-Differential, das im Folgenden vorgestellt wird.
Es seien F und G Verteilungen aus der Menge $\mathcal{F}$ aller Verteilungsfunktionen und

$$\{(1-\lambda)F + \lambda G,\ 0 \leq \lambda \leq 1\}$$

sei die Menge der Konvexkombinationen aus F und G. Das Funktional T sei definiert auf $F + \lambda(G - F)$ für alle hinreichend kleinen λ. Existiert der Grenzwert

$$d_1 T(F; G - F) = \lim_{\lambda \to 0+} \frac{T(F + \lambda(G - F)) - T(F)}{\lambda} \quad ,$$

so wird dieser als das Gâteaux-Differential des Funktionals T an der Stelle F in Richtung von G bezeichnet. Man erkennt, dass der Ausdruck $d_1T(F; G - F)$ die rechtsseitige Ableitung der reellen Funktion $Q(\lambda) = T(F + \lambda(G - F))$ nach λ an der Stelle $\lambda = 0$ ist. Als nächstes wird die Abweichung der Schätzung $T(F_n)$ von dem wahren Wert $T(F)$ mit Hilfe des Gâteaux-Differentials von T an der Stelle F in Richtung F_n untersucht. Es sei

$$Q(\lambda) = T(F + \lambda(F_n - F)) \quad .$$

Die lineare Taylorapproximation von Q an der Stelle $\lambda = 0$ ergibt

$$\begin{aligned} T(F_n) - T(F) &= Q(1) - Q(0) \\ &\approx Q'(0+) \\ &= d_1T(F; F_n - F) \quad . \end{aligned}$$

Ist das Gâteaux-Differential im zweiten Argument linear, erhält man weiter

$$T(F_n) - T(F) \approx \frac{1}{n}\sum_{i=1}^{n} d_1T(F; \delta_{x_i} - F) \tag{12}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \mathrm{IC}(x_i; F, T) \quad . \tag{13}$$

Dieses Ergebnis liefert eine weitere Begründung für die Bezeichnung Einflusskurve. $\mathrm{IC}(\cdot)$ approximiert den Beitrag der Beobachtung x_i zu dem Schätzfehler $T(F_n) - T(F)$. Darüber hinaus zeigt sich aus der Darstellung des Fehlers in (12), dass dieser näherungsweise als arithmetisches Mittel der IC-Kurve verstanden werden kann. Das arithmetische Mittel ist aufgrund des Zentralen Grenzwertsatzes bei existierender positiver Varianz von $\mathrm{IC}(X; F, T)$ asymptotisch normalverteilt. Damit kann man schließen, dass der Fehler, d.h. die linke Seite der Abschätzung (12), unter bestimmten Voraussetzungen ([18], S. 225f) asymptotisch normalverteilt ist. Sind die Bedingungen

$$\mathrm{Var}(\mathrm{IC}(X; F, T)) > 0 \quad ,$$

$$\sqrt{n}\left((T(F_n) - T(F)) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{IC}(X_i; F, T)\right) \xrightarrow{st.} 0 \tag{14}$$

erfüllt [3,4], so gilt

$$T(F_n) - T(F) \overset{as.}{\sim} \mathcal{N}\left(\mathbb{E}(\mathrm{IC}(X; F, T)),\ \frac{1}{n}\mathrm{Var}(\mathrm{IC}(X; F, T))\right) \quad ,$$

[3] Der Begriff der stochastischen Konvergenz wird auf Seite 75 definiert.
[4] In [18] wird für viele Beispielfälle gezeigt, dass die Bedingungen (14) erfüllt sind.

wobei die Bezeichnung $\overset{as.}{\sim} \mathcal{N}(\ldots)$ bedeutet, dass der Ausdruck

$$\frac{T(F_n) - T(F) - \mathbb{E}(\text{IC}(X; F, T))}{\sqrt{\frac{1}{n}\text{Var}(\text{IC}(X; F, T))}}$$

asymptotisch standardnormalverteilt ist. Im Allgemeinen liegt asymptotische Erwartungstreue vor, d.h. der Erwartungswert der Einflusskurve ist gleich Null und für die Varianz gilt

$$\text{Var}(\text{IC}(X; F, T)) = \mathbb{E}(\text{IC}(X; F, T)^2) \quad .$$

Anhand der Maximum-Likelihood-Schätzung soll im Folgenden das beschriebene Vorgehen veranschaulicht werden.

Seien $X_1, \ldots, X_n$ unabhängig und identisch verteilte Zufallsgrößen mit einer Verteilung F_θ, die zu einer Familie $\{F_\theta, \theta \in \Theta \subset \mathbb{R}\}$ gehört. Außerdem sei vorausgesetzt, dass die Verteilung F_θ eine Dichtefunktion $f(x; \theta)$ besitze. Die Likelihood-Funktion der Stichprobe $x_1, \ldots, x_n$ ist definiert als

$$L(t; x_1, \ldots, x_n) = \prod_{i=1}^{n} f(x_i; t) \quad .$$

Die Maximum-Likelihood-Methode bietet als Schätzer für das unbekannte t jeden Wert $\hat{\theta}$ an, der die Likelihood-Funktion über Θ maximiert. Dabei stellt es eine Erleichterung dar, wenn man zu der negativen, logarithmierten Likelihood-Funktion

$$-\log\left(L(t; x_1, \ldots, x_n)\right) = -\sum_{i=1}^{n} \log\left(f(x_i; t)\right)$$

übergeht und so das Produkt durch eine Summe ersetzen kann. Durch den Vorzeichenwechsel ist nun ein Minimum zu bestimmen. Dazu wird eine Nullstelle der ersten Ableitung gesucht, d.h. die Gleichung

$$\left.\frac{\partial \log L}{\partial t}\right|_{t=\hat{\theta}} = 0 \tag{15}$$

ist zu lösen. (15) entspricht der Integralgleichung

$$\int g(x, t)\, dF_n(x) = 0$$

mit

$$g(x, t) = \frac{d}{dt} \log\left(f(x; t)\right) \quad ,$$

d.h.

$$g(x,t) = \frac{\frac{d}{dt} f(x;t)}{f(x;t)} \quad ,$$

wobei

$$f(x;t) = \frac{d}{dx} F_t(x) \quad .$$

Damit ist der Maximum-Likelihood-Schätzer $\hat{\theta} = T(F_n)$, wobei $T(F)$ das Funktional ist, das durch die Lösung von

$$\int g(x,t)\ dF(x) = 0 \tag{16}$$

definiert wird. Unter gewissen Regularitätsbedingungen ([18], S. 144) an $\{F_\theta, \theta \in \Theta\}$ ergeben sich

$$\int \frac{d}{dt} \log\left(f(x;t)\right)\Big|_{t=\theta}\ dF_\theta(x) = 0 \tag{17}$$

und

$$\int \frac{\frac{d^2}{dt^2} f(x;t)}{f(x;t)}\Big|_{t=\theta}\ dF_\theta(x) = \int \frac{d^2}{dt^2} f(x;t)\Big|_{t=\theta}\ dx = 0 \quad . \tag{18}$$

Aus (17) folgt unmittelbar

$$\int g(x,\theta)\ dF_\theta(x) = 0 \quad ,$$

d.h.

$$T(F_\theta) = \theta \quad .$$

Seien $F_\lambda = F_\theta + \lambda(\delta_{x_0} - F_\theta)$ und $H(t,\lambda) = \int g(x,t)\ dF_\lambda(x)$, dann erhält man durch implizites Differenzieren der Gleichung

$$H(T(F_\lambda), \lambda) = 0$$

nach λ an der Stelle $\lambda = 0$ die Gleichung

$$\frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=\theta,\lambda=0} \cdot \frac{dT(F_\lambda)}{d\lambda}\Big|_{\lambda=0} + \frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=\theta,\lambda=0} = 0$$

und daraus ergibt sich

$$\frac{dT(F_\lambda)}{d\lambda}\Big|_{\lambda=0} = \frac{-\frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=\theta,\lambda=0}}{\frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=\theta,\lambda=0}} \quad . \tag{19}$$

Setzt man nun die bis dahin gewonnenen Ergebnisse ein, so erhält man

$$\begin{aligned}
\left.\frac{\partial H(t,\lambda)}{\partial \lambda}\right|_{t=\theta,\lambda=0} &= \left.\frac{\partial}{\partial \lambda}\left[\int g(x,t)\; dF_\theta(x) + \lambda \int g(x,t)\; d[\delta_{x_0} - F_\theta(x)]\right]\right|_{t=\theta,\lambda=0} \\
&= \int g(x,\theta)\; d\delta_{x_0} - \underbrace{\int g(x,\theta)\; dF_\theta(x)}_{\overset{(17)}{=}0} \\
&= \left.\frac{\frac{d}{dt} f(x_0;t)}{f(x_0;t)}\right|_{t=\theta}
\end{aligned}$$

und

$$\begin{aligned}
\left.\frac{\partial H(t,\lambda)}{\partial t}\right|_{t=\theta,\lambda=0} &= \int \left.\frac{d}{dt} g(x,t)\right|_{t=\theta} dF_\theta(x) \\
&= \int \left.\frac{d}{dt}\frac{\frac{d}{dt} f(x;t)}{f(x;t)}\right|_{t=\theta} dF_\theta(x) \\
&= \int \frac{\left.\frac{d^2}{dt^2} f(x;t)\right|_{t=\theta} \cdot f(x;t)|_{t=\theta} - \left(\left.\frac{d}{dt} f(x;t)\right|_{t=\theta}\right)^2}{f(x;\theta)^2}\; dF_\theta(x) \\
&= \underbrace{\int \frac{\left.\frac{d^2}{dt^2} f(x;t)\right|_{t=\theta}}{f(x;\theta)}\; dF_\theta(x)}_{\overset{(18)}{=}0} - \int \left.\left(\frac{d}{dt}\log\left(f(x;t)\right)\right)^2\right|_{t=\theta} dF_\theta(x) \\
&= -\int \left.\left(\frac{d}{dt}\log\left(f(x;t)\right)\right)^2\right|_{t=\theta} dF_\theta(x) \quad .
\end{aligned}$$

Damit folgt für das Gâteaux-Differential

$$\begin{aligned}
d_1 T(F_\theta; \delta_{x_0} - F_\theta) &= \left.\frac{d}{d\lambda} T(F_\theta + \lambda(\delta_{x_0} - F_\theta))\right|_{\lambda=0} \\
&= \frac{\frac{d}{d\theta} f(x_0;\theta)}{f(x_0;\theta)} \cdot \frac{1}{\int \frac{\left(\frac{d}{d\theta} f(x;\theta)\right)^2}{f(x;\theta)}\; dx}
\end{aligned}$$

und für die Einflusskurve

$$\mathrm{IC}(x; F_\theta, T) = \frac{\frac{d}{d\theta}\log\left(f(x;\theta)\right)}{\int \left(\frac{d}{d\theta}\log\left(f(x;\theta)\right)\right)^2 dF_\theta(x)} \quad .$$

8.3.3 Der Bruchpunkt

Der Bruchpunkt (breakdown point) ε^* einer Schätzung gibt diejenige Grenze an, bis zu welcher der Anteil von Ausreißern in einer Stichprobe ansteigen darf, ohne dass sich dadurch der Schätzwert unbeschränkt verändern kann. Im Gegensatz zur Einflusskurve ist der Bruchpunkt eine globale Kenngröße der Robustheit. Für die Bruchpunkte von arithmetischem Mittel und Median ergeben sich:

(a) Beim arithmetischen Mittel kann bereits ein einzelner Wert innerhalb einer Beobachtungsreihe den Schätzwert für das Zentrum der Stichprobe über alle Grenzen wachsen lassen, daher gilt

$$\varepsilon^*_{\text{arith. Mittel}} = 0 \quad .$$

(b) Liegt der Anteil der Ausreißer unter 50%, so kann keine beliebige Veränderung des Medians vorkommen. Der Median hat daher als Bruchpunkt den Wert

$$\varepsilon^*_{\text{Median}} = 0,5 \quad .$$

8.4 L-Schätzer

Die in Abschnitt 8.2 vorgestellten Schätzer für Lage- und Streuungsmaßzahlen beruhen alle auf Linearkombinationen der Ordnungsstatistiken $x_{(1)}, \ldots, x_{(n)}$ einer Stichprobe vom Umfang n. Daher werden sie L-Schätzer genannt. Seien $a_1, \ldots, a_n$ reelle Zahlen mit $0 \leq a_i \leq 1, i = 1, \ldots, n$ und $\sum_{i=1}^{n} a_i = 1$, dann ist

$$T = \sum_{i=1}^{n} a_i \; x_{(i)}$$

ein L-Schätzer mit Gewichten $a_1, \ldots, a_n$.
Das arithmetische Mittel ist also ebenso ein L-Schätzer wie der Median oder der Interquartilabstand. Die Bezeichnung „L-Schätzer“ sagt also noch nichts über die Eigenschaft der Robustheit aus. Ein Vorteil der L-Schätzer liegt in der leichten Berechenbarkeit, die in der Regel lediglich eine Vorsortierung der Stichprobenelemente erfordert.

8.4.1 L-Schätzer für Lageparameter

8.4.1.1 Das α-gestutzte Mittel

Es sei vorausgesetzt, dass die zu Grunde liegende Verteilung symmetrisch und stetig ist, d.h. sie besitzt eine Dichte. Beim α-gestutzten Mittel als Schätzer für den Median bzw., sofern er existiert, den Erwartungswert der zu Grunde liegenden Verteilung werden aus der geordneten Stichprobe am unteren und oberen Ende eine gleiche Anzahl von Beobachtungen weggelassen und über die verbliebenen Werte das arithmetische Mittel berechnet. Der

Anteil der zu entfernenden Werte wird mit α angegeben. Unter dem α-gestutzten Mittel versteht man daher den Wert

$$\overline{x}_\alpha = \frac{1}{n-2h} \sum_{i=h+1}^{n-h} x_{(i)} \quad ,$$

wobei $0 < \alpha < 0,5$ ist und $h = [n\alpha]$ die größte ganze Zahl kleiner oder gleich $n\alpha$ bezeichnet. Dieser Ansatz stellt damit einen Kompromiss zwischen dem arithmetischen Mittel und dem Median dar, die sich für die Werte $\alpha \to 0$ bzw. $\alpha \to 0,5$ ergeben würden. Eine asymptotisch äquivalente Version des α-getrimmten Mittels ist durch

$$\overline{x}_\alpha = T(F_n)$$

gegeben, wobei für eine beliebige Verteilungsfunktion F das Funktional T durch

$$\begin{aligned} T(F) &= \frac{1}{1-2\alpha} \int_{F^{-1}(\alpha)}^{F^{-1}(1-\alpha)} x \, dF(x) \\ &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} F^{-1}(p) dp \end{aligned}$$

mit $F^{-1}(\alpha) = \inf\{x | F(x) \geq \alpha\}$ definiert ist.
Für die Einflusskurve gilt zunächst nach [18] für eine stetige Verteilung

$$\begin{aligned} \left. \frac{dT[F + \lambda(\delta_{x_0} - F)]}{d\lambda} \right|_{\lambda=0} &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} \frac{p - I(x_0 \leq F^{-1}(p))}{f\left(F^{-1}(p)\right)} \, dp \\ &= \frac{1}{1-2\alpha} \int_{\alpha}^{1-\alpha} \frac{p - I(F(x_0) \leq p)}{f\left(F^{-1}(p)\right)} \, dp \quad , \end{aligned}$$

wobei $I(\cdot)$ die Indikatorfunktion von Aussagen[5] bezeichnet. Hieraus folgt für eine symmetrische stetige Verteilung

$$\left. \frac{dT[F + \lambda(\delta_{x_0} - F)]}{d\lambda} \right|_{\lambda=0}$$

$$= \begin{cases} \frac{1}{1-2\alpha}[F^{-1}(\alpha) - F^{-1}(1/2)] & \text{, falls} \quad x < F^{-1}(\alpha) \\ \frac{1}{1-2\alpha}(x - F^{-1}(1/2)) & \text{, falls} \quad F^{-1}(\alpha) \leq x \leq F^{-1}(1-\alpha) \\ \frac{1}{1-2\alpha}[F^{-1}(1-\alpha) - F^{-1}(1/2)] & \text{, falls} \quad x > F^{-1}(1-\alpha) \end{cases} .$$

[5] $I(A) = \begin{cases} 1 \text{ , falls die Aussage } A \text{ wahr ist.} \\ 0 \text{ , falls die Aussage } A \text{ falsch ist.} \end{cases}$

8.4.1.2 Das α-winsorisierte Mittel

Beim α-winsorisierten Mittelwert wird im Gegensatz zum gestutzten Mittelwert die Anzahl der Beobachtungen beibehalten. Jedoch werden die ersten h Beobachtungen durch den $(h+1)$-ten Wert ersetzt und die h letzten Werte durch den $(n-h)$-ten Wert, dann wird das arithmetische Mittel gebildet, d.h.

$$\bar{x}_{\alpha,win} = \frac{1}{n}\left(\sum_{i=h+1}^{n-h} x_{(i)} + h(x_{(h+1)} + x_{(n-h)})\right) \quad .$$

8.4.2 L-Schätzer für die Streuung

8.4.2.1 Der Interquartilabstand

Der Begriff des Interquartilabstandes wurde bereits in Abschnitt 8.2.2 definiert. Er dient an dieser Stelle der Motivation des robusten Skalenschätzers MAD, der im nächsten Unterabschnitt eingeführt wird. Der Bruchpunkt des Interquartilabstandes beträgt $\varepsilon^*_{\text{IQR}} = 0,25$. Mit den bekannten Werten aus Beispiel 8.2.1 erhält man wie in Abschnitt 8.2.2 bereits angegeben

$$\tilde{x}_{0,25} = -1,0\ ,\ \tilde{x}_{0,75} = 0,5 \quad \Rightarrow \quad \text{IQR} = 1,5 \quad .$$

Ergänzt man $x_6 = 10,5$, so ergibt sich der Wert IQR $= 2,5$, jedoch erhält man mit $\hat{x}_6 = -10,5$ einen IQR von 10,0. Ursache für diesen auffälligen Unterschied ist die Lage der beiden Ausreißer am gleichen Ende der Stichprobe. Ihr Anteil an der Stichprobe beträgt $\frac{2}{6} = 0,33$ und übersteigt damit den Bruchpunkt.

8.4.2.2 Der Median der absoluten Abweichung vom Median

Sei F eine beliebige Verteilungsfunktion, dann ist

$$T(F) = \frac{1}{2}\left[F^{-1}\left(\frac{3}{4}\right) - F^{-1}\left(\frac{1}{4}\right)\right]$$

das Funktional, das die Hälfte des Interquartilabstandes angibt. Zu diesem Funktional kann man nun eine symmetrische Version $\tilde{T}$ konstruieren, indem man

$$\bar{F}(x) = 1 - F(2 \cdot F^{-1}(1/2) - x)$$

und

$$\tilde{F}(x) = \frac{1}{2}[F(x) + \bar{F}(x)]$$

setzt. $\tilde{F}(x)$ entsteht dabei aus $F(x)$ durch eine Symmetrisierung am Median m_n. Definiert man anschließend

$$\tilde{T}(F) := T(\tilde{F}) \quad ,$$

so ist $\tilde{T}$ der Median der absoluten Abweichung vom Median (MAD). Als Median der absoluten Abweichung vom Median einer Stichprobe vom Umfang n bezeichnet man daher

$$\text{MAD}_n = \text{med}\{|x_i - m_n|\} \quad ,$$

wobei

$$\text{med}\{x_i\} = m_n \quad .$$

Der Vorteil der symmetrisierten Version des halben Interquartilabstandes liegt darin, dass sich der Wert für den Bruchpunkt verdoppelt, d.h.

$$\varepsilon^*_{\text{MAD}} = 0,5 = 2 \cdot 0,25 = 2 \cdot \varepsilon^*_{\text{IQR}} \quad .$$

Darüber hinaus ist die Einflusskurve beschränkt, so dass einzelne Beobachtungswerte nur begrenzten Einfluss auf den Schätzwert haben.

Das folgende Beispiel der Schätzung der Standardabweichung einer Normalverteilung zeigt, wie man gegebenenfalls mittels einer Multiplikation mit einer Konstanten einen Schätzwert für einen gesuchten Parameter erhält.
Für eine symmetrische Verteilung ist der MAD eine Schätzung des halben Interquartilabstandes. Sind $x_1, \ldots, x_n$ Realisationen aus einer $\mathcal{N}(\mu, \sigma^2)$-Verteilung, so ist der MAD ein Schätzer für

$$\Phi^{-1}(0.75) \cdot \sigma \quad ,$$

wobei $\Phi^{-1}(0.75) = 0,6745$ das $0,75$-Quantil der Standardnormalverteilung ist [7]. Somit ist $\frac{1}{0,6745} \cdot \text{MAD}$ ein Schätzer für σ. Verwendet man den Interquartilabstand, so ergibt sich entsprechend, dass $\frac{1}{2} \cdot \frac{1}{0,6745} \cdot \text{IQR} \approx 0,7413 \cdot \text{IQR}$ ein Schätzer für σ ist.

8.4.2.3 Die α-gestutzte Varianz

Die α-gestutzte Varianz ist definiert als die skalierte Varianz einer α-gestutzten Stichprobe. Die Varianz einer α-gestutzten Stichprobe

$$s^2_{gest} = \frac{1}{n - 2[n\alpha]} \sum_{i=[n\alpha]+1}^{n-[n\alpha]} (x_{(i)} - \bar{x}_\alpha)^2 \tag{20}$$

unterschätzt die Varianz der den Beobachtungen zugrundeliegenden Verteilung und ist keine konsistente Schätzung. Daher benutzt man einen Korrekturterm

$$\frac{1 - 2\alpha}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha})} \quad ,$$

wobei $\tilde{x}_{1-\alpha}$ das $(1-\alpha)$-Quantil und ϕ die Dichte der Standardnormalverteilung ist. Damit ist

$$\hat{\sigma}^2_{gest} = \frac{1 - 2\alpha}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha})} \cdot s^2_{gest}$$

eine konsistente Schätzung für σ^2 unter der Normalverteilung.

8.4.2.4 Die α-winsorisierte Varianz

Die α-winsorisierte Varianz ist definiert als

$$s^2_{win} = \frac{1}{n-1}\left(\sum_{i=[n\alpha]+1}^{n-[n\alpha]} (x_{(i)} - \bar{x}_{\alpha,win})^2 \right.$$
$$\left. + [n\alpha](x_{([n\alpha]+1)} - \bar{x}_{\alpha,win})^2 + [n\alpha](x_{(n-[n\alpha])} - \bar{x}_{\alpha,win})^2 \right) \quad ,$$

wobei auch dies zunächst kein konsistenter Schätzer für die Varianz ist. Analog zur α-gestutzten Varianz kann man auch für diesen Fall einen Korrekturterm angeben, so dass

$$\hat{\sigma}^2_{win} = \frac{1}{1 - 2\alpha - 2\tilde{x}_{1-\alpha}\phi(\tilde{x}_{1-\alpha}) + 2\alpha\tilde{x}^2_{1-\alpha}} \cdot s^2_{win}$$

insgesamt ein konsistenter Schätzer für σ^2 unter der Normalverteilung ist.

8.5 M-Schätzer

8.5.1 Theoretischer Hintergrund

Die Theorie der M-Schätzer basiert auf einer Verallgemeinerung der Maximum-Likelihood-Methode (s. Abschnitt 8.3.2). Wie dort sei $F_\theta(x) := F(x;\theta)$ eine Verteilung aus der Verteilungsfamilie $\{F_\theta |\ \theta \in \Theta \subset \mathbb{R}\}$ mit einer Dichte $f_\theta(x) := f(x;\theta)$. Die Maximum-Likelihood-Methode liefert einen Schätzer für den unbekannten Parameter θ, indem der Ausdruck $\sum_{i=1}^{n}(-\log(f(x_i;\theta)))$ minimiert wird. Ist f differenzierbar, so führt dies zur Lösung des Gleichungssystems

$$\sum_{i=1}^{n} \frac{\partial}{\partial\theta}(-\log(f(x_i;\theta))) = 0 \quad .$$

Dieser Ansatz führt unter der Voraussetzung, dass eine Normalverteilung mit Erwartungswert μ und Varianz σ^2 vorliegt, zu folgender Problemstellung bei der Suche nach einem Schätzer für den Erwartungswert:

$$\sum_{i=1}^{n} \frac{(x_i - \mu)^2}{2} \to \min_{\mu} \quad ,$$

d.h.

$$\sum_{i=1}^{n}(x_i - \mu) = 0 \quad .$$

Das Ergebnis ist das arithmetische Mittel $\bar{x}_n$. Probleme bei diesem Vorgehen tauchen dann auf, wenn Ausreißer in den Daten vorhanden sind. Das heißt, die Ausreißerempfindlichkeit der ML-Methode ist nicht befriedigend. Um diese zu verbessern, ersetzt man $-\log(f(x;\theta))$ durch eine Funktion $\rho(x,\theta)$, die weniger sensitiv auf Ausreißer reagiert. Damit lautet das verallgemeinerte Vorgehen:

$$\text{minimiere den Ausdruck } \sum_{i=1}^{n} \rho(x_i,\theta) \tag{21}$$

bzw.

$$\text{löse das Gleichungssystem } \sum_{i=1}^{n} \psi(x_i,\theta) = 0 \quad , \tag{22}$$

wobei $\psi(x,\theta)$ die Ableitung von $\rho(x,\theta)$ nach θ ist. Jede Lösung von (21) bzw. (22) wird M-Schätzer genannt.

8.5.1.1 Grundlagen

Zu einer beliebigen Funktion $\psi(x,t)$ sei ein zugehöriges Funktional T auf einer Menge von Verteilungsfunktionen F wie folgt definiert. Es ist $T(F)$ die Lösung t_0 der Gleichung

$$\int \psi(x,t_0)\; dF(x) = 0 \quad . \tag{23}$$

Ein solches T wird das zu ψ gehörende M-Funktional genannt. Für eine Stichprobe $x_1,\ldots,x_n$ aus einer Verteilung F ist der zu ψ gehörende M-Schätzer T_n die Lösung der Gleichung

$$\sum_{i=1}^{n} \psi(x_i,T_n) = 0 \quad ,$$

die sich aus (23) mit $F = F_n$ ergibt, d.h. $T_n = T(F_n)$. Dabei ist zu beachten, dass mehrere Lösungen existieren können. Gleichung (23) ergibt sich in vielen Fällen aus der Minimierung des Ausdrucks

$$\int \rho(x,t_0)\; dF(x) \quad ,$$

wobei dann die Funktion ψ bestimmt ist durch

$$\psi(x,t) = c \cdot \frac{\partial}{\partial t}\rho(x,t)$$

mit einer Konstanten c, falls $\rho(x,\cdot)$ hinreichend glatt ist. Gilt für ein Funktional T auf der Verteilungsfamilie $\{F_\theta |\ \theta \in \Theta \subset \mathbb{R}\}$

$$T(F_\theta) = \theta \quad ,$$

dann ist der zugehörige Schätzer $T_n = T(F_n)$ ein Schätzer für θ. Ein Kriterium zur Beurteilung eines Schätzers stellt die Einflusskurve (s. Abschnitt 8.3.2) dar, die somit zur Bestimmung einer geeigneten ψ-Funktion und eines geeigneten M-Schätzers herangezogen werden kann. Ziel ist es nun, das Gâteaux-Differential eines M-Funktionals zu berechnen, d. h. es ist $d_1T(F, G-F)$ für ein Funktional T gesucht, das Lösung t_0 der Gleichung

$$\int \psi(x, t_0) dF(x) = 0$$

ist. $\psi(x,t)$ sei dabei eine beliebige Funktion. Es sei

$$H(t, \lambda) = \int \psi(x,t) dF_\lambda(x) \quad .$$

Mit

$$F_\lambda(x) = F(x) + \lambda(G(x) - F(x))$$

folgt wie in Abschnitt 8.3.2 (Maximum-Likelihood-Schätzung) durch implizite Differentiation der Gleichung $H(T(F_\lambda), \lambda) = 0$ nach λ an der Stelle $\lambda = 0$ [6]

$$\begin{aligned}
d_1T(F, G-F) &= \frac{d}{d\lambda} T(F + \lambda(G-F))\Big|_{\lambda=0} \\
&= -\frac{\partial H(t,\lambda)}{\partial \lambda}\Big|_{t=T(F),\lambda=0} \Big/ \frac{\partial H(t,\lambda)}{\partial t}\Big|_{t=T(F),\lambda=0} \\
&= \frac{-\frac{\partial}{\partial \lambda}\left[\int \psi(x,t)\, dF(x) + \lambda \int \psi(x,t)\, d[G(x)-F(x)]\right]\big|_{t=T(F),\lambda=0}}{\frac{d}{dt}\int \psi(x,t)\, dF(x)\big|_{t=T(F)}} \\
&= -\frac{\int \psi(x,t)\, dG(x)\big|_{t=T(F)} - \overbrace{\int \psi(x,t)\, dF(x)\Big|_{t=T(F)}}^{\stackrel{(23)}{=}0}}{\frac{d}{dt}\int \psi(x,t)\, dF(x)\big|_{t=T(F)}} \\
&= -\frac{\int \psi(x, T(F)) dG(x)}{\lambda_F'(T(F))} \quad ,
\end{aligned}$$

vorausgesetzt, dass $\lambda_F'(T(F)) \neq 0$ gilt mit

$$\lambda_F(t) = \int \psi(x,t)\, dF(x), \quad -\infty < t < \infty \quad .$$

Damit besitzt die Einflusskurve von ψ die Form:

$$IC(x; F, T) = -\frac{\psi(x, T(F))}{\lambda_F'(T(F))}, \quad -\infty < x < \infty \quad .$$

[6] g in Abschnitt 8.3.2 entspricht hier ψ.

Aufgrund der Tatsache, dass die Einflusskurve proportional zu ψ ist, besitzen M-Schätzer die Eigenschaft, dass Bedingungen an die Einflusskurve über die Wahl von ψ gesteuert werden können. Eine geeignete Wahl von ψ führt zu vorteilhaften Eigenschaften der Einflusskurve und damit des M-Schätzers.

8.5.1.2 Asymptotische Eigenschaften von M-Schätzern

In diesem Abschnitt werden M-Schätzer hinsichtlich Konsistenz und asymptotischer Normalität untersucht. Zunächst sei dazu auf die verschiedenen Konvergenzbegriffe in der Wahrscheinlichkeitstheorie hingewiesen:

- Seien $X_1, X_2, \ldots$ und X Zufallsgrößen über einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$. Dann konvergiert $(X_n)_{n\geq 1}$ stochastisch gegen X, falls für alle $\varepsilon > 0$

$$\lim_{n\to\infty} \mathbb{P}(|X_n - X| < \varepsilon) = 1 \quad (\text{Bez.} : X_n \xrightarrow{st.} X \quad (n \to \infty)) \quad .$$

- Seien $X_1, X_2, \ldots$ und X Zufallsgrößen über einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \mathbb{P})$. $(X_n)_{n\geq 1}$ konvergiert fast sicher gegen X, falls

$$\mathbb{P}(\lim_{n\to\infty} X_n = X) = 1 \quad (\text{Bez.} : X_n \xrightarrow{f.s.} X \quad (n \to \infty)) \quad .$$

- Seien $F_1(\cdot), F_2(\cdot), \ldots$ und $F(\cdot)$ Verteilungsfunktionen und $X_1, X_2, \ldots$ sowie X Zufallsgrößen (nicht notwendig über einem gemeinsamen Wahrscheinlichkeitsraum) mit den angegebenen Verteilungen. Dann konvergiert $(X_n)_{n\geq 1}$ der Verteilung nach gegen X, falls für alle Stetigkeitsstellen t von F

$$\lim_{n\to\infty} F_n(t) = F(t) \quad (\text{Bez.} : X_n \xrightarrow{n.V.} X \quad (n \to \infty)) \quad .$$

Konsistenz eines M-Schätzers

Eine Folge $\{T_n\}$ von Schätzern für eine parametrische Funktion $g(\theta)$ heißt konsistent, falls T_n gegen $g(\theta)$ in geeigneter Weise konvergiert. Man spricht von schwacher Konsistenz, falls

$$T_n \xrightarrow{st.} g(\theta) \qquad (n \to \infty) \quad ,$$

und von starker Konsistenz im Falle von

$$T_n \xrightarrow{f.s.} g(\theta) \qquad (n \to \infty) \quad .$$

Sei $\psi(x,t)$ eine Funktion und $\lambda_F(t) = \int \psi(x,t)\, dF(x)$. Außerdem besitze die Gleichung $\lambda_F(t) = 0$ eine Lösung t_0 und die „empirische“ Gleichung $\lambda_{F_n}(t) = 0$ ebenfalls eine Lösung T_n. Im Folgenden werden Bedingungen genannt, unter denen $T_n \xrightarrow{f.s.} t_0$ gilt. Dazu sei $x_1, \ldots, x_n$ eine Stichprobe aus einer nach F verteilten Grundgesamtheit mit empirischer Verteilungsfunktion F_n.
Falls t_0 eine isolierte Lösung der Gleichung $\lambda_F(t) = 0$ ist und $\psi(x,t)$ monoton in t, dann

ist t_0 eindeutig, und jede Folge $\{T_n\}$ von Lösungen der empirischen Gleichung $\lambda_{F_n}(t) = 0$ konvergiert fast sicher gegen t_0. Ist außerdem $\psi(x,t)$ stetig in t in einer Umgebung von t_0, so existiert eine solche Folge von Lösungen. Für den Beweis dieser Behauptung sei an dieser Stelle auf [18], Abschnitt 7.2.1, verwiesen. t_0 muss nicht notwendigerweise eine Nullstelle von $\lambda_F(t)$ sein. Es reicht aus, dass die Funktion in jeder hinreichend kleinen Umgebung genau einmal das Vorzeichen wechselt.
Das arithmetische Mittel, der Median und der Huber-Schätzer auf den Seiten 77 bis 79 sind Beispiele für konsistente Schätzer der jeweiligen Lageparameter. Für den Hampel-Schätzer (s. Beispiel auf Seite 80) reicht dieses Ergebnis jedoch nicht aus, und man benötigt die folgende Aussage.
Sei t_0 eine isolierte Nullstelle der Gleichung $\lambda_F(t) = 0$, und sei $\psi(x,t)$ stetig in t und beschränkt. Dann besitzt die empirische Gleichung $\lambda_{F_n}(t) = 0$ eine Folge $\{T_n\}$ von Lösungen, die mit Wahrscheinlichkeit 1 gegen t_0 konvergiert. Der Beweis hierzu ist ebenfalls in [18], Abschnitt 7.2.1, zu finden.

Asymptotische Normalverteilung

Sei $\psi(x,t)$ gegeben und $\lambda_F(t) = \int \psi(x,t)\, dF(x)$. Außerdem sei $t_0 = T(F)$ eine Lösung der Gleichung $\lambda_F(t) = 0$. $\{X_i\}$ ist eine Folge von unabhängig und identisch verteilten Zufallsgrößen mit einer Verteilung F und $T_n = T(F_n)$ eine zu t_0 konsistente Folge von Lösungen von $\lambda_{F_n}(t) = 0$. In dem vorliegenden Abschnitt sollen nun Bedingungen angegeben werden, unter denen

$$\sqrt{n}(T_n - t_0) \overset{as.}{\sim} \mathcal{N}(0, \sigma^2(T,F)) \tag{24}$$

gilt. Dabei ist $\sigma^2(T,F)$ abhängig von den Annahmen an $\psi(x,t)$ entweder gegeben durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{(\lambda_F'(t_0))^2}$$

oder durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{\left(\int \left(\frac{\partial\psi(x,t)}{\partial t}\right)_{t=t_0} dF(x)\right)^2} \quad .$$

Es sei die Funktion $\psi(x,t)$ monoton in t und außerdem t_0 eine isolierte Nullstelle von $\lambda_F(t) = 0$. Des Weiteren sei $\lambda_F(t)$ differenzierbar in $t = t_0$ mit $\lambda_F'(t_0) \neq 0$ sowie $\int \psi^2(x,t)\, dF(x)$ endlich für t in einer Umgebung von t_0 und stetig in $t = t_0$. Dann gilt nach [18], Abschnitt 7.2.2, dass jede Folge T_n von Lösungen der empirischen Gleichung $\lambda_{F_n}(t) = 0$ die Bedingung (24) erfüllt, wobei $\sigma^2(T,F)$ gegeben ist durch

$$\frac{\int \psi^2(x,t_0)\, dF(x)}{[\lambda_F'(t_0)]^2} \quad .$$

Darüber hinaus gilt $T_n \xrightarrow{f.s.} t_0$ nach dem Ergebnis von Seite 75.

8.5.2 M-Schätzer für Lageparameter

Sind die das M-Funktional bestimmenden Funktionen ψ und ρ von der Form $\psi(x,t) = \tilde{\psi}(x-t)$ bzw. $\rho(x,t) = \tilde{\rho}(x-t)$, so nennt man $T(F)$ einen Lageparameter. Ist zudem F symmetrisch zu θ, ρ gerade und ψ ungerade, so gilt $T(F) = \theta$, d.h. alle M-Schätzer mit ungeradem ψ schätzen das Symmetriezentrum θ einer symmetrischen Verteilung. Außerdem sollten M-Schätzer translations- und skalierungsäquivariant sein, d.h. es sollte gelten:

$$T_n(x_1,\ldots,x_n) = BT_n\left(\frac{x_1-A}{B},\ldots,\frac{x_n-A}{B}\right)+A \quad .$$

Die meisten M-Schätzer für einen Lageparameter müssen daher eine Skalierung der Daten berücksichtigen, so dass sie diese Bedingung erfüllen. Dabei stellen das arithmetische Mittel und der Median zwei Ausnahmen dar, denn für den Mittelwert gilt

$$\psi(x,t) = x - t$$

und damit folgt

$$\begin{aligned}\psi(bx+a,bt+a) &= bx+a-bt-a \\ &= bx-bt \\ &= b(x-t) \\ &= b\psi(x,t) \quad .\end{aligned}$$

Analoges gilt für den Median mit $\psi(x,t) = \text{sgn}(x-t)$. Alle anderen M-Schätzer benötigen die Skalierung der Argumente von ρ und ψ. Daher wird ein Skalierungsschätzer s_n als Funktion von $x_1,\ldots,x_n$ benutzt, der zusammen mit einer Tuning-Konstante C den Ausdruck $x_i - t$ neu skaliert. Damit erhält man zentrierte und skalierte Daten über den Ausdruck

$$u_i = \frac{x_i - t}{C \cdot s_n} \quad .$$

8.5.2.1 Das arithmetische Mittel

Sucht man einen kleinste-Quadrate-Schätzer, so will man den Ausdruck

$$\sum_{i=1}^{n}(x_i-\theta)^2$$

minimieren. Als ρ und ψ ergeben sich in diesem Falle

$$\rho(u) = \frac{1}{2}u^2 \quad , \quad \psi(u) = u \quad , \quad -\infty < u < \infty \quad .$$

Damit ist das M-Funktional T der Erwartungswert und der M-Schätzer das arithmetische Mittel $\overline{x}_n$. Die ψ-Funktion ist eine Gerade und nach beiden Seiten unbeschränkt, d.h. Ausreißer haben einen starken Einfluss.

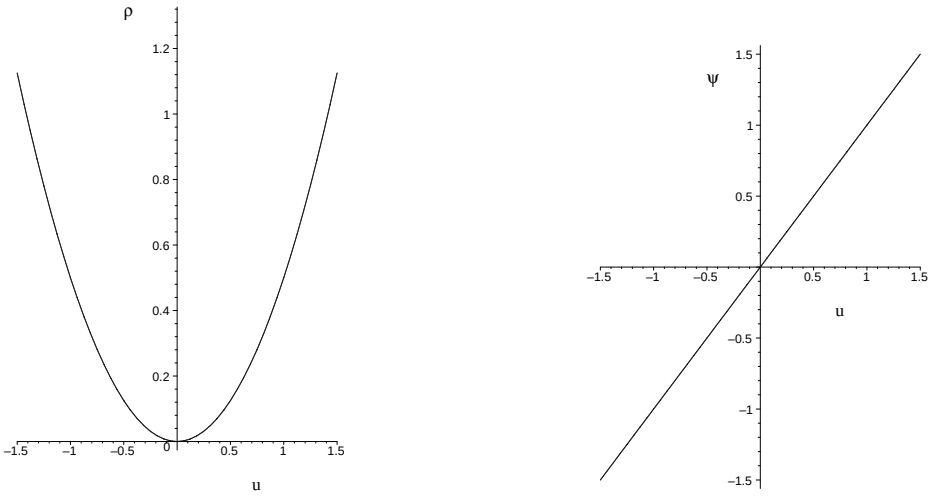


Abbildung 13: Arithmetisches Mittel $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.2 Der Median

Möchte man die absoluten Abweichungen minimieren, also den Ausdruck

$$\sum_{i=1}^{n} |x_i - \theta| \quad ,$$

so lauten die zugehörigen Funktionen für ρ und ψ:

$$\rho(u) = |u| \quad , \quad \psi(u) = \text{sgn}(u) \quad , \quad -\infty < u < \infty \quad .$$

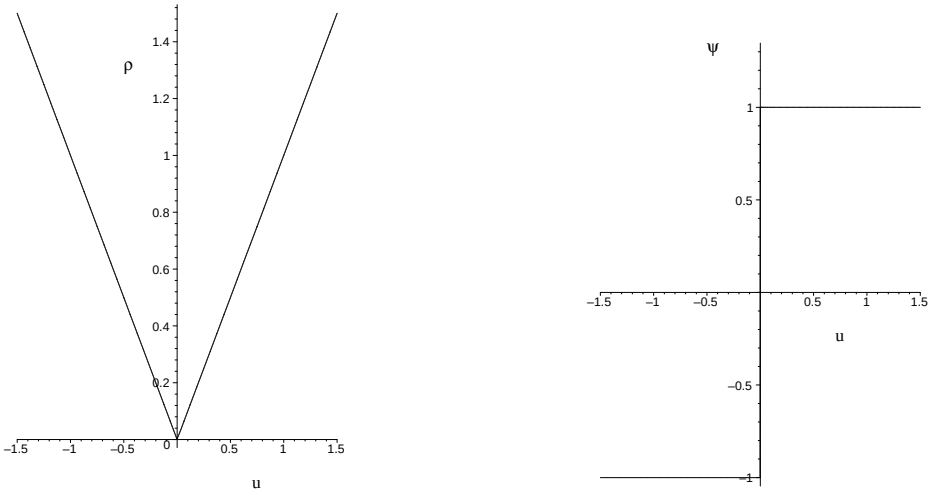


Abbildung 14: Median $\rho(u)$ (links) und $\psi(u)$ (rechts).

Hier ist der M-Schätzer der Stichprobenmedian. Er ist unempfindlich gegenüber Ausreißern, da die ψ-Funktion beschränkt ist. Die Sprungstelle von ψ bei $u = 0$ zeigt jedoch an, dass der Median sehr stark von den „mittleren“ Werten abhängt.

8.5.2.3 Der Huber-Schätzer

Als Familie der Huber-Schätzer bezeichnet man die M-Schätzer, die

$$\sum_{i=1}^{n} \rho(u_i)$$

minimieren. Dabei ist ρ von der Form

$$\rho(u) = \begin{cases} \frac{1}{2}u^2 & \text{, falls} \quad |u| \leq k \\ k|u| - \frac{1}{2}k^2 & \text{, falls} \quad |u| > k \end{cases} .$$

Das entsprechende ψ lautet

$$\psi(u) = \begin{cases} u & \text{, falls} \quad |u| \leq k \\ k \cdot \text{sgn}(u) & \text{, falls} \quad |u| > k \end{cases} .$$

Dabei hat sich in Anwendungen gezeigt, dass sich als Skalierungsschätzer s_n der normalisierte MAD empfiehlt. C wird gleich Eins gesetzt, und für k sollte man einen Wert im Intervall $[1; 2]$ wählen. Der zugehörige M-Schätzer ist von der Art eines winsorisierten Mittelwertes, d.h. es ist der Stichprobenmittelwert über die x_i's, wobei diejenigen x_i durch $T_n \pm ks_n$ ersetzt werden, für die $|u_i| > k$ gilt. Hierbei wird der Einfluss von Ausreißern gedämpft, aber nicht vollständig ausgeräumt.

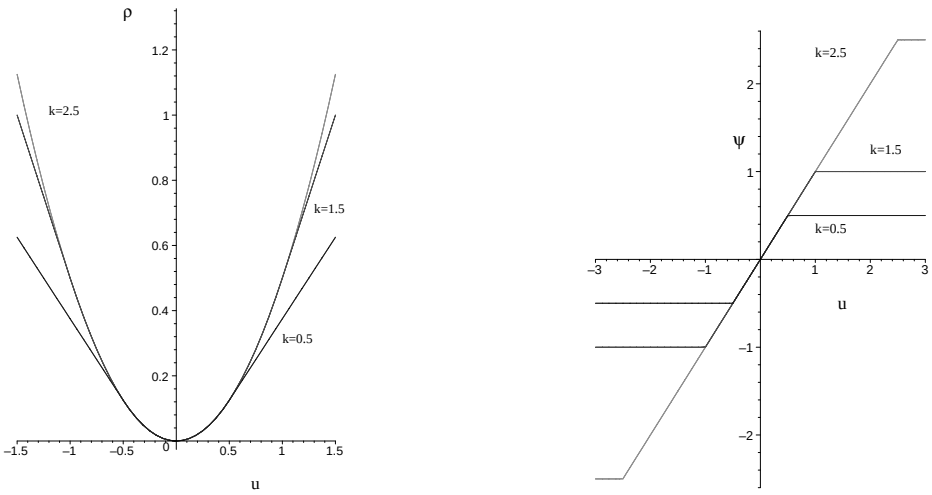


Abbildung 15: Huber-Schätzer $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.4 Der Hampel-Schätzer

Hampel [5] hat den Huber-Schätzer im Bezug auf die Ausreißerempfindlichkeit (Gross-Error-Sensitivity) modifiziert. Er verlangt, dass $\psi(u)$ für Ausreißer den Wert Null annehmen soll. Analog zu den bisherigen Fällen ist auch hier der Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

zu minimieren. Dabei sind

$$\rho(u) = \begin{cases} \frac{1}{2}u^2 & \text{, falls} \quad |u| \le a \\ a|u| - \frac{1}{2}a^2 & \text{, falls} \quad a < |u| \le b \\ ab - \frac{1}{2}a^2 + (c-b)\frac{a}{2}\left[1 - \left(\frac{c-|u|}{c-b}\right)^2\right] & \text{, falls} \quad b < |u| \le c \\ ab - \frac{1}{2}a^2 + (c-b)\frac{a}{2} & \text{, falls} \quad |u| > c \end{cases}$$

und

$$\psi(u) = \begin{cases} u & \text{, falls} \quad |u| \le a \\ a \cdot \text{sgn}(u) & \text{, falls} \quad a < |u| \le b \\ a \cdot \left(\frac{c-|u|}{c-b}\right) \cdot \text{sgn}(u) & \text{, falls} \quad b < |u| \le c \\ 0 & \text{, falls} \quad |u| > c \end{cases} .$$

Hierbei sind a, b und c positive Konstanten mit $a < b < c$ aus dem Intervall $[2; 9]$ (zum Beispiel $a = 2, b = 4$ und $c = 8$). Als Skalierungsschätzer s_n wird wie im vorangegangenen Beispiel beim Huber-Schätzer der normalisierte MAD benutzt und C gleich Eins gesetzt. Beim Hampel-Schätzer bleiben Ausreißer vollständig unberücksichtigt.

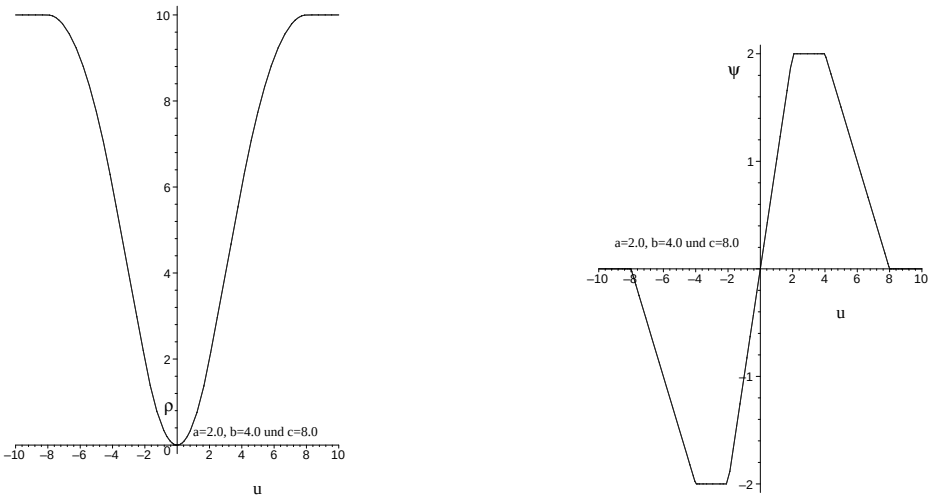


Abbildung 16: Hampel-Schätzer $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.5 Andrew's wave

Man geht hier ebenfalls von dem zu minimierenden Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

aus mit

$$\rho(u) = \begin{cases} \frac{1}{\pi^2} \cdot (1 - \cos(\pi u)) & \text{, falls} \quad |u| \leq 1 \\ \frac{2}{\pi^2} & \text{, falls} \quad |u| > 1 \end{cases} .$$

Die zugehörige ψ-Funktion ist gegeben durch

$$\psi(u) = \begin{cases} \frac{1}{\pi} \sin(\pi u) & \text{, falls} \quad |u| \leq 1 \\ 0 & \text{, falls} \quad |u| > 1 \end{cases}$$

und eliminiert ebenfalls Ausreißer vollständig. Hier wird als Skalierungsschätzer ebenfalls $s_n = \text{MAD}$ empfohlen und C sollte im Bereich $[1,5\pi;\ 2,4\pi]$ liegen.

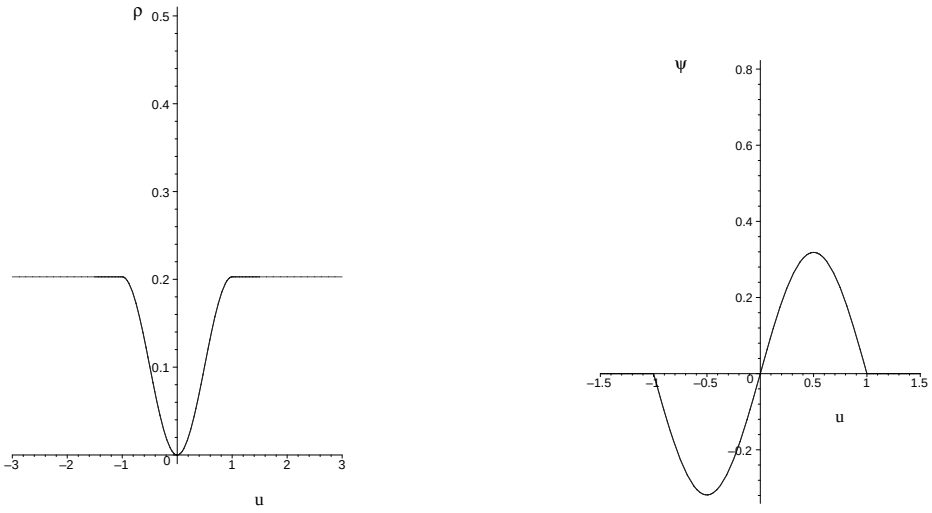


Abbildung 17: Andrew's wave $\rho(u)$ (links) und $\psi(u)$ (rechts).

8.5.2.6 Tukey's biweight

Zu dem zu minimierenden Ausdruck

$$\sum_{i=1}^{n} \rho(u_i)$$

mit

$$\rho(u) = \begin{cases} \frac{1}{6} \cdot (1 - (1 - u^2)^3) & \text{, falls} \quad |u| \leq 1 \\ \frac{1}{6} & \text{, falls} \quad |u| > 1 \end{cases}$$

lautet das zugehörige ψ:

$$\psi(u) = \begin{cases} u(1-u^2)^2 & \text{, falls} \quad |u| \leq 1 \\ 0 & \text{, falls} \quad |u| > 1 \end{cases} .$$

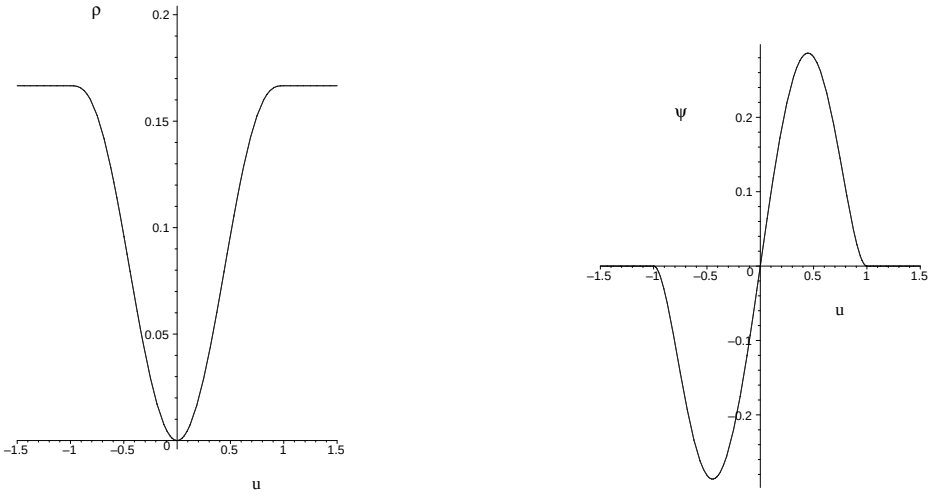


Abbildung 18: Tukey's biweight $\rho(u)$ (links) und $\psi(u)$ (rechts).

Der MAD wird als Skalierungsschätzer s_n verwendet, und für C empfehlen sich Werte aus dem Intervall $[6; 12]$. Auch hier werden Ausreißer mit einem zu großen Abstand zum Zentrum der Datenpunkte eliminiert. Für kleine u gilt $\psi(u) \approx u$. Daher verhält sich dieser Schätzer in der Nähe des Datenzentrums ähnlich dem arithmetischen Mittel.

Von diesen sechs Schätzern wurden der Huber-Schätzer, Andrew's wave und Tukey's biweight in der GALA-Software implementiert und dienen dort der Berechnung von robusten Lagemaßzahlen. Auf der Grundlage dieser robusten Schätzungen können dann verlässliche Aussagen über das Zentrum der Daten gemacht werden.

8.5.3 Diskussion der Eigenschaften verschiedener M-Schätzer

Die Robustheitseigenschaften der im vorangegangenen Abschnitt vorgestellten M-Schätzer kann man anhand der ψ-Funktion diskutieren. Folgende Eigenschaften sollte die ψ-Funktion eines robusten Lage-M-Schätzers einer symmetrischen Verteilung besitzen:

(a) hoher Bruchpunkt,

(b) ψ ist beschränkt,

(c) ψ besitzt „annähernd" stetiges Verhalten,

(d) ψ besitzt endlichen Ablehnungspunkt,

(e) $\psi(u) \approx k \cdot u, k \neq 0$ für kleine Werte von u und

(f) $\psi(-u) = -\psi(u)$.

Für die wichtigsten sechs M-Schätzer fasst Tabelle 5 die geforderten Eigenschaften zusammen:

Schätzer	a	b	c	d	e	f
Mittelwert	nein	nein	ja	nein	ja	ja
Median	ja	ja	nein	nein	nein	ja
Huber	ja	ja	ja	nein	ja	ja
Hampel	ja	ja	ja	ja	ja	ja
Andrew‘s wave	ja	ja	ja	ja	ja	ja
Tukey‘s biweight	ja	ja	ja	ja	ja	ja

Tabelle 5: Eigenschaften der wichtigsten Schätzer.

Die Spaltenbezeichnungen beziehen sich auf die Aufzählung zu Beginn dieses Abschnittes.
Die Tuning-Konstante k beim Huber-Schätzer kann bei bekanntem Ausreißeranteil ε folgendermaßen bestimmt werden:

$$\frac{2\phi(k)}{k} - 2\Phi(-k) = \frac{\varepsilon}{1-\varepsilon} \quad ,$$

wobei Φ die Standard-Normalverteilungsfunktion bezeichnet und ϕ die zugehörige Dichtefunktion. Die Tuning-Konstanten bei der Familie der Hampel-Schätzer können auf analoge Weise bestimmt werden [8]. Möchte man die im vorangegangenen Abschnitt vorgestellten M-Schätzer mit den L-Schätzern aus Abschnitt 8.4 vergleichen, so fällt auf, dass die M-Schätzer in der Regel einen hohen Bruchpunkt nahe bei $\varepsilon^* = 0,5$ haben und eine hohe Effizienz in einer Umgebung der Modellverteilung aufweisen [9]. L-Schätzer hingegen besitzen in diesem Falle eine geringere Effizienz oder einen kleineren Bruchpunkt.

8.5.4 Berechnung von M-Schätzern

8.5.4.1 Berechnung mittels Newton-Raphson-Verfahren

Zur Berechnung eines Lage-M-Schätzers ist T_n als Lösung der Gleichung

$$\sum_{i=1}^{n} \psi\left(\frac{x_i - T_n}{C \cdot S_n}\right) = 0$$

zu bestimmen. Im Allgemeinen ist diese Lösung jedoch nicht explizit angebbar. Einzige Ausnahme stellt der Fall des arithmetischen Mittels dar mit $\psi(x_i, T_n) = x_i - T_n$. In allen anderen Fällen muss daher mit Hilfe eines geeigneten Iterationsverfahrens eine Lösung gesucht werden. Als geeignet hat sich das Newton-Raphson-Verfahren erwiesen. Dabei wird

eine Nullstelle einer Funktion $h(t)$ gesucht. Diese Nullstelle wird durch Auswertungen von h und h' an einer Näherungsstelle $t^{(k)}$ bestimmt. $t^{(k+1)}$ ist dann die Stelle, an der die Tangente an h im Punkt $t^{(k)}$ die t-Achse schneidet. Bei der Berechnung eines M-Schätzers lautet die Funktion h:

$$h(T_n) = \sum_{i=1}^{n} \psi\left(\frac{x_i - T_n}{C \cdot S_n}\right) \quad .$$

Für die Iterationsvorschrift ergibt sich daraus

$$\begin{aligned} T_n^{(k+1)} &= T_n^{(k)} - \frac{h(T_n^{(k)})}{h'(T_n^{(k)})} \\ &= T_n^{(k)} + C \cdot S_n \frac{\sum_{i=1}^{n} \psi(u_i^{(k)})}{\sum_{i=1}^{n} \psi'(u_i^{(k)})} \quad . \end{aligned}$$

Als Startwert sollte für $T_n^{(0)}$ der Median als erste robuste Näherung gewählt werden Der gesuchte M-Schätzer ergibt sich dann als Grenzwert der Folge $(T_n^{(k)})_{k\geq 1}$. Das Ergebnis ist in den meisten Fällen ein robuster Schätzer.

8.5.4.2 1-Schritt-Verfahren

Führt man beim Newton-Raphson-Verfahren nur einen einzigen Iterationsschritt aus, so wird das daraus resultierende Ergebnis für den gesuchten Schätzer 1-Schritt-M-Schätzer genannt. Bei diesem Vorgehen ist offensichtlich die Wahl des Startwertes für $T_n^{(0)}$ sehr wichtig. Falls als Startwert der arithmetische Mittelwert benutzt wird, ist das Ergebnis nach einem Iterationsschritt unbefriedigend. Es sollte daher der Median als Anfangswert genutzt werden, damit das Ergebnis einen robusten M-Schätzer liefert.

8.5.5 M-Schätzer für Streuungsparameter

Grundlage sei eine Verteilungsfamilie mit Dichten $\frac{1}{\sigma} f_0\left(\frac{x-\mu}{\sigma}\right) = f(x;\mu,\sigma)$ mit Lageparameter μ und Skalierungsparameter σ. Wendet man das Prinzip der Maximum-Likelihood-Schätzung auf diesen Fall an, so ergibt sich folgendes zu lösende Problem:

$$\sum_{i=1}^{n} -\log\left(\frac{1}{\sigma} f_0\left(\frac{x_i - \mu}{\sigma}\right)\right) \to \min_{\mu,\sigma} \quad .$$

Aufgrund der Rechenregeln für die Logarithmusfunktion ergibt sich daraus

$$\sum_{i=1}^{n} \Bigg(\log(\sigma) + \underbrace{\left(-\log\left(f_0\left(\frac{x_i - \mu}{\sigma}\right)\right)\right)}_{=:\rho_{f_0}\left(\frac{x_i-\mu}{\sigma}\right)} \Bigg) \to \min_{\mu,\sigma} \quad .$$

Zur Bestimmung des Minimums werden nun die partiellen Ableitungen nach μ und σ gebildet, diese gleich Null gesetzt und das so entstandene System der Normalengleichungen gelöst, wobei $\psi_f = \rho'_f$ ist. Man erhält

$$\sum_{i=1}^{n} \frac{1}{\sigma} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \left(\frac{1}{\sigma} - \frac{x_i - \mu}{\sigma^2} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) \right) = 0$$

und daraus ergibt sich

$$\sum_{i=1}^{n} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \left(\frac{x_i - \mu}{\sigma} \psi_{f_0} \left(\frac{x_i - \mu}{\sigma} \right) - 1 \right) = 0 \quad .$$

Um nun simultane M-Schätzer für Lage und Streuung zu erhalten, wird dieses zu lösende System verallgemeinert zu

$$\sum_{i=1}^{n} \psi \left(\frac{x_i - T}{S} \right) = 0 \quad \wedge \quad \sum_{i=1}^{n} \chi \left(\frac{x_i - T}{S} \right) = 0 \quad .$$

Im Allgemeinen ist dabei ψ eine ungerade und χ eine gerade Funktion. In Funktionalschreibweise ergeben sich damit folgende Integralgleichungen:

$$\int \psi \left(\frac{x - T}{S} \right) \, dF(x) = 0 \quad \wedge \quad \int \chi \left(\frac{x - T}{S} \right) \, dF(x) = 0 \quad .$$

Die Schätzer für den Lageparameter und den Streuungsparameter werden durch simultanes Lösen dieser beiden Gleichungen bestimmt.

8.5.5.1 Huber-Schätzer für Lage und Streuung

Mit

$$\psi(x) = \max(-k, \min(k, x))$$

und

$$\chi(x) = \min(c^2, x^2) - \beta \quad ,$$

wobei $\beta = \beta(c) = \int \min(c^2, x^2) \; \phi(x) \; dx$ und $0 < \beta < c^2$, ergibt sich ein unter Normalverteilung konsistenter Schätzer für die Streuung mit $\phi(\cdot)$ als Dichtefunktion der Standardnormalverteilung.

8.5.5.2 Median und MAD

Für

$$\psi(x) = \text{sgn}(x)$$

und

$$\chi(x) = \text{sgn}(|x| - 1)$$

ergeben sich der Median als Schätzer für den Lageparameter und der MAD als Schätzer für die Streuung.

8.5.5.3 A-Schätzer

Eine weitere Möglichkeit, einen robusten Schätzer für die Varianz bzw. Streuung einer Stichprobe zu erhalten, ist die Verwendung von sogenannten A-Schätzern.
Die A-Schätzer

$$s_T = \frac{C \cdot \mathrm{MAD}\sqrt{n} \cdot \sqrt{\sum_{i=1}^{n} \psi(u_i)^2}}{|\sum_{i=1}^{n} \psi'(u_i)|}$$

erhält man aus der asymptotischen Varianz der M-Lageparameterschätzer (s. Abschnitt 8.5.2) mit $u_i = \frac{x_i - m_n}{C \cdot \mathrm{MAD}}$.
Setzt man

$$\psi(u) = \begin{cases} \sin(\pi u) & \text{, falls} \quad |u| < 1 \\ 0 & \text{, falls} \quad |u| \geq 1 \end{cases} ,$$

so führt dies zu dem Skalenschätzer

$$s_{wa} = \frac{(C \cdot \mathrm{MAD})\sqrt{n}[\sum_{|u_i|<1} \sin^2(\pi u_i)]^{1/2}}{\pi|\sum_{|u_i|<1} \cos(\pi u_i)|} \quad , \tag{25}$$

der auf Andrew‘s wave Schätzer für einen Lageparameter (s. Seite 81) basiert.
Ein weiteres Beispiel zeigt einen A-Schätzer für die Streuung, der auf dem Biweight-Lageschätzer (s. Seite 81) beruht. Sei

$$\psi(u) = \begin{cases} u(1-u^2)^2 & \text{, falls} \quad |u| < 1 \\ 0 & \text{, falls} \quad |u| \geq 1 \end{cases} ,$$

dann erhält man

$$s_{bi} = \frac{\sqrt{n}\left[\sum_{|u_i|<1}(x_i - m_n)^2(1-u_i^2)^4\right]^{1/2}}{|\sum_{|u_i|<1}(1-u_i^2)(1-5u_i^2)|} \tag{26}$$

als Skalenschätzer. Die Skalenschätzer (25) und (26) sind in der GALA-Software implementiert worden.

8.6 Robuste Schätzer für multivariate Lage und Streuung

8.6.1 Einleitung und Motivation

Bisher lagen n Beobachtungen aus einer Beobachtungsreihe vor, für die in der Regel *ein* Lageparameter und *ein* Parameter für die Streuung zu schätzen waren. Im vorliegenden

Abschnitt seien nun mehrere Beobachtungsreihen gegeben, d.h. es seien $x^{(1)}, \ldots, x^{(n)}$ p-variate Beobachtungsdaten. Die robuste Schätzung eines Lageparameters für diesen multivariaten Fall führt daher zur Bestimmung eines robusten Lagevektors. Ein multivariater Lagevektor kann auf zwei unterschiedliche Arten geschätzt werden. Zum einen kann man komponentenweise für jede Beobachtungsreihe einen einzelnen robusten Lageschätzer bestimmen. Dazu eignen sich die in Abschnitt 8.4 und Abschnitt 8.5 vorgestellten L- bzw. M-Schätzer. Eine weitere Möglichkeit besteht in der multivariaten, simultanen Schätzung des Lagevektors, bei der alle Beobachtungen gemeinsam eingehen. Dieses Vorgehen stellt eine Erweiterung der M-Schätzer für den multivariaten Fall dar. Im Abschnitt 8.6.3 werden dazu M-Schätzer für multivariate Lage und Streuung betrachtet. Die Schätzung der Streuung läuft auf die Berechnung von Kovarianz- bzw. Korrelationsmatrizen hinaus. Auf der Grundlage der Matrizen, die sich aus diesen Kovarianzen bzw. Korrelationen aufbauen, wird zum Beispiel die Hauptkomponentenanalyse (s. Abschnitt 6) oder die Faktoranalyse durchgeführt. Darüber hinaus dienen Kovarianz- und Korrelationsmatrizen dem Test auf Unabhängigkeit. Die entsprechenden empirischen Matrizen sind jedoch sehr ausreißerempfindlich, so dass robusten Methoden eine große Bedeutung zukommt.
Für zwei gegebene Zufallsgrößen X und Y sind die Kovarianz $\text{Kov}(X, Y)$ und die Korrelation $\text{Korr}(X, Y) = \rho(X, Y)$ gemäß (1) und (2) gegeben. Seien nun $X_1, \ldots, X_n$ Zufallsvariablen. Dann ist die zugehörige Kovarianzmatrix definiert als

$$\Sigma = \begin{pmatrix} \text{Var}X_1 & \text{Kov}(X_1, X_2) & \ldots & \text{Kov}(X_1, X_n) \\ \text{Kov}(X_2, X_1) & \text{Var}X_2 & \ldots & \text{Kov}(X_2, X_n) \\ \vdots & \ddots & \ddots & \vdots \\ \text{Kov}(X_n, X_1) & \ldots & \text{Kov}(X_n, X_{n-1}) & \text{Var}X_n \end{pmatrix},$$

und als Korrelationsmatrix wird

$$R = \begin{pmatrix} 1 & \text{Korr}(X_1, X_2) & \ldots & \text{Korr}(X_1, X_n) \\ \text{Korr}(X_1, X_2) & 1 & \ldots & \text{Korr}(X_2, X_n) \\ \vdots & \ddots & \ddots & \vdots \\ \text{Korr}(X_1, X_n) & \ldots & \text{Korr}(X_{n-1}, X_n) & 1 \end{pmatrix}$$

bezeichnet.
Bei der Berechnung robuster Kovarianz- bzw. Korrelationsmatrizen unterscheidet man zwei Vorgehensweisen. Zum einen kann man jeden Eintrag in der Kovarianz- bzw. Korrelationsmatrix separat robust schätzen. Dieses Vorgehen führt zu univariaten Analysen für die Varianzen und bivariaten Analysen für die Kovarianzen bzw. Korrelationen. Dieses Verfahren hat aber den Nachteil, dass die sich ergebenden Matrizen im Allgemeinen nicht positiv semidefinit sind. Die zweite Vorgehensweise besteht in der simultanen Schätzung aller Elemente der Kovarianzmatrix und führt zu einer multivariaten Analyse. Die sich aus

dieser Methode ergebenden Matrizen sind positiv semidefinit. Die Methode der separaten Schätzung bietet jedoch Vorteile, wenn ein größerer Anteil der Beobachtungen fehlt.

8.6.2 Separate robuste Schätzer für die Elemente der Kovarianz- bzw. Korrelationsmatrix

Ein einfacher Ansatz zur Schätzung der Kovarianz zwischen zwei Zufallsvariablen X und Y beruht auf der Gleichung

$$\text{Kov}(X,Y) = \frac{1}{4}[\text{Var}(X+Y) - \text{Var}(X-Y)] \quad .$$

Die Schätzung der Kovarianz kann somit zurückgeführt werden auf die Schätzung von Varianzen. Auf der Grundlage der robusten univariaten Streuungsschätzer aus den Abschnitten 8.4.2 und 8.5.5 ist es also möglich, robuste Schätzer für Kovarianz- und Korrelationsmatrix anzugeben.
Einen robusten Schätzer s^*_{xy} für die Kovarianz zwischen X und Y erhält man mittels

$$s^*_{xy} = \frac{1}{4}[\hat{\sigma}^{*2}_1 - \hat{\sigma}^{*2}_2] \quad ,$$

wobei $\hat{\sigma}^*_1$ und $\hat{\sigma}^*_2$ robuste Skalenschätzer für $X+Y$ und $X-Y$ sind. Damit erhält man auch eine robuste Schätzung der Korrelation zwischen X und Y:

$$r^*_{xy} = \frac{s^*_{xy}}{\sqrt{s^*_{xx} \cdot s^*_{yy}}} \quad ,$$

wobei s^*_{xx} und s^*_{yy} robuste Schätzungen der Varianzen von X und Y sind.
Berechnet man für zwei Zufallsgrößen X und Y die herkömmliche Korrelation

$$r_{xy} = \frac{s_{xy}}{s_x s_y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x}_n)^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \bar{y}_n)^2}} \quad ,$$

so liegt der Wert aufgrund der Gültigkeit der Cauchy-Schwarz-Ungleichung immer im Intervall $[-1;1]$. Da aber r^*_{xy} nicht auf den normalen Standardabweichungen, sondern auf robusten Schätzungen für die Varianzen basiert, muss r^*_{xy} nicht zwangsläufig im Intervall $[-1;1]$ liegen. Um diese Eigenschaft der Korrelation sicher zu stellen, wendet man eine Modifikation an. Seien

$$Z_x = \frac{X}{\sqrt{s^*_{xx}}} \quad \text{bzw.} \quad Z_y = \frac{Y}{\sqrt{s^*_{yy}}}$$

die standardisierten Formen von X und Y mit robusten Schätzungen der Varianz s^*_{xx} bzw. s^*_{yy}. Definiert man jetzt

$$\tilde{r}^*_{xy} = \frac{\hat{\sigma}^{*2}_3 - \hat{\sigma}^{*2}_4}{\hat{\sigma}^{*2}_3 + \hat{\sigma}^{*2}_4} \quad , \tag{27}$$

wobei $\hat{\sigma}_3^{*2}$ und $\hat{\sigma}_4^{*2}$ robuste Schätzungen der Varianzen von $Z_1 + Z_2$ bzw. $Z_1 - Z_2$ sind, so ergibt sich, dass $\tilde{r}_{xy}^*$ im Intervall $[-1; 1]$ liegt. Die robusten Varianzschätzungen sind im Allgemeinen nicht konsistent. Eine konsistente Schätzung der Varianz bei Normalverteilung erhält man durch geeignete Normierungskonstanten. Diese müssen bei der Berechnung der Korrelationen jedoch nicht berücksichtigt werden, da sie sich herauskürzen. Aus Gleichung (27) lässt sich leicht auch der entsprechende robuste Kovarianzschätzer herleiten:

$$\tilde{s}_{xy}^* = \tilde{r}_{xy}^* \cdot \sqrt{s_{xx}^* s_{yy}^*} \quad .$$

Dieses Verfahren wurde unter Verwendung verschiedener robuster Varianzschätzer implementiert.

8.6.3 M-Schätzer für multivariate Lage- und Skalierungsparameter

Wie im univariaten Fall ergeben sich die M-Schätzer durch Verallgemeinerung der Maximum-Likelihood-Schätzung. Zunächst wird die Klasse der Normalverteilungen, für die die zugehörigen Maximum-Likelihood-Schätzer nicht robust sind, zu einer Verteilungsklasse erweitert, die auch Verteilungsfamilien mit stärker besetzten Tails enthält, deren Maximum-Likelihood-Schätzer robust sind.
Ein p-dimensionaler Zufallsvektor X mit der Verteilungsdichte

$$f(x) = (\det V)^{-\frac{1}{2}} h\left((x-t)^\top V^{-1}(x-t)\right) \quad ,$$

wobei h eine nichtnegative reellwertige Funktion, $t \in \mathbb{R}^p$ und V eine positiv definite Matrix ist, heißt elliptisch symmetrisch verteilt.
Mit

$$h(u) = \frac{1}{(2\pi)^{p/2}} \exp(-\frac{u}{2})$$

ergibt sich die Normalverteilung und mit

$$h(u) = \frac{\Gamma(\frac{\nu+p}{2})}{(\pi\nu)^{p/2}\Gamma(\nu/2)} \cdot \frac{1}{(1+\frac{1}{\nu}u)^{(\nu+p)/2}} \tag{28}$$

die Familie der p-dimensionalen t-Verteilung mit ν Freiheitsgraden. Wie im univariaten Fall sind die Tails der multivariaten t-Verteilung stärker besetzt als die der Normalverteilung. Die auf der t-Verteilung basierende Maximum-Likelihood-Schätzung gewichtet stark abweichende Beobachtungen schwächer als die auf der Normalverteilung basierende Maximum-Likelihood-Schätzung.
Die Maximum-Likelihood-Schätzung für eine elliptisch symmetrische Verteilung ergibt sich durch Maximierung der Likelihood-Funktion

$$\prod_{i=1}^{n} f(x^{(i)}) = \prod_{i=1}^{n} \frac{1}{\sqrt{\det(V)}} h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)$$

bezüglich t und V, wobei mit $x^{(i)}$ der Vektor der i-ten Beobachtungsreihe bezeichnet wird, d.h.

$$x^{(i)} = \left(x_1^{(i)}, \ldots, x_p^{(i)}\right)^\top \quad , \quad 1 \leq i \leq n \quad .$$

Logarithmiert man die Likelihood-Funktion und maximiert bzgl. V^{-1} statt V, so ergibt sich die äquivalente Aufgabe

$$\log \prod_{i=1}^{n} f(x^{(i)}) = \frac{n}{2} \log(\det V^{-1}) + \sum_{i=1}^{n} \log\left(h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)\right) \to \max_{t,V^{-1}} \quad .$$

Zur Bestimmung des Maximums werden nun die beiden partiellen Ableitungen nach t und V^{-1} gebildet:

$$\frac{\partial}{\partial t}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \sum_{i=1}^{n} \frac{-2h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x^{(i)}-t)^\top \cdot V^{-1}$$

$$\frac{\partial}{\partial v^{kl}}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \frac{n}{2} \underbrace{\frac{\frac{\partial}{\partial v^{kl}} \det(V^{-1})}{\det(V^{-1})}}_{=\frac{\det(V^{kl})}{\det(V^{-1})} = v_{lk}} + \sum_{i=1}^{n} \frac{h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x_k^{(i)}-t_k)(x_l^{(i)}-t_l) \quad ,$$

wobei v^{kl} das (k,l)-te Element der inversen Matrix V^{-1} bezeichnet und V^{kl} die Matrix ist, die aus der Matrix V entsteht, indem das (k,l)-te Element gleich Eins gesetzt wird und die restlichen Elemente der k-ten Zeile und l-ten Spalte gleich Null gesetzt werden. Damit ergibt sich insgesamt für die partielle Ableitung nach der inversen Matrix V^{-1}:

$$\frac{\partial}{\partial V^{-1}}\left[\log \prod_{i=1}^{n} f(x^{(i)})\right] = \frac{n}{2}V + \sum_{i=1}^{n} \frac{h'\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)}{h\left((x^{(i)}-t)^\top V^{-1}(x^{(i)}-t)\right)} \cdot (x^{(i)}-t)(x^{(i)}-t)^\top \quad .$$

Beide partiellen Ableitungen werden anschließend gleich Null gesetzt, wobei als abkürzende Schreibweisen

$$d_i^2 = (x^{(i)}-t)^\top V^{-1}(x^{(i)}-t) \quad ,$$

$$w(d_i^2) = \frac{-2h'(d_i^2)}{h(d_i^2)} \tag{29}$$

verwendet werden.

Damit ergibt sich folgendes Gleichungssystem:

$$\begin{aligned} \sum_{i=1}^{n} w(d_i^2)(x^{(i)} - t) &= 0 \\ \sum_{i=1}^{n} \left(w(d_i^2)(x^{(i)} - t)(x^{(i)} - t)^\top - V \right) &= 0 \quad . \end{aligned}$$

Durch Verallgemeinerung dieses Gleichungssystems ergeben sich die M-Schätzer für multivariate Lage und Streuung. Die Lösungen des Gleichungssystems

$$\sum_{i=1}^{n} w_1(d_i^2)(x^{(i)} - t) = 0 \tag{30}$$

$$\sum_{i=1}^{n} \left(w_2(d_i^2)(x^{(i)} - t)(x^{(i)} - t)^\top - v(d_i^2)V \right) = 0 \tag{31}$$

heißen M-Schätzer für den multivariaten Lageparameter und die Kovarianzmatrix.

Verfahren	**Korrelation**	**Ausreißer**	**Tuning-Parameter**
einfache Korrelation	-0,35	keiner	
	-0,81	(1; 105)	
	-0,88	(1; 105), (10; 10)	
	-0,89	(1; 105), (10; 10), (14; 1)	
0,01-getrimmt	-0,81	(1; 105), (10; 10)	
0,05-getrimmt	-0,90	(1; 105), (10; 10), (14; 1)	
0,01-winsorisiert	-0,75	(1; 105), (10; 10)	
0,05-winsorisiert	-0,90	(1; 105), (10; 10), (14; 1)	
Tukey‘s biweight	-0,90	(1; 105), (10; 10), (14; 1)	9
(A-Schätzer)	-0,91	(1; 105), (10; 10), (14; 1)	6
Andrew‘s wave	-0,90	(1; 105), (10; 10), (14; 1)	6,6
(A-Schätzer)			
Huber	-0,91	(1; 105), (10; 10), (14; 1)	1,5
(A-Schätzer)			
Huber	-0.89	(1; 105), (10; 10), (14; 1)	
(Simultan)			

Tabelle 6: Daten zur Korrelation bei verschiedenen M-Schätzern.

Hieraus ergeben sich die Maximum-Likelihood-Schätzer der p-dimensionalen t-Verteilung direkt aus (28) und (29):

$$w_1(d) = \frac{m+\nu}{\nu+d^2} = w_2(d^2) \quad , \quad v(d_i^2) = 1 \quad ,$$

während bei den auf Hubers ψ-Funktionen basierenden Schätzern

$$w_1(d) = \begin{cases} 1 & \text{, falls} \quad d \leq a \\ \frac{a}{d} & \text{, falls} \quad d > a \end{cases} \quad ,$$
$$w_2(d^2) = w_1(d)^2 \quad \text{und} \quad v(d_i^2) = c$$

gelten, wobei c ein Korrekturterm für die asymptotische Erwartungstreue unter der Normalverteilung ist und gemäß [16], S. 225 berechnet werden kann.
Das folgende Beispiel liefert anhand eines Testdatensatzes einen Überblick über robuste und "konventionelle" Korrelationsberechnungen. Die für dieses Beispiel genutzten Daten werden in Tabelle 7 im Anhang angegeben. Aus den zunächst 109 Datenpaaren werden drei Beobachtungswerte als Ausreißer identifiziert und nacheinander eliminiert (Abbildung 19). Die Angaben zur Korrelation in Tabelle 6 zeigen deutlich, dass bei der herkömmlichen Korrelationsberechnung mit den drei Ausreißern der Wert keine Aussage über eine Beziehung zulässt. Die Werte, die sich bei Benutzung der robusten Verfahren ergeben, ähneln sich und zeigen eine Korrelation von ca. $0,9$.

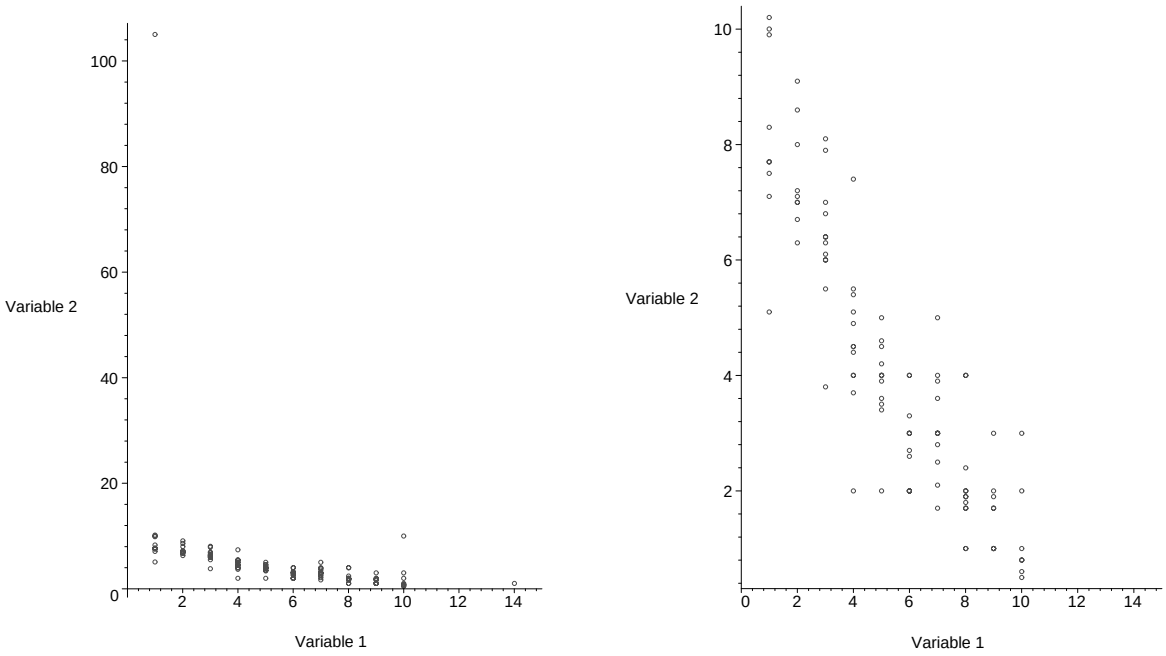


Abbildung 19: Streubild zweier Variablen: 109 Datenpaare (links), 106 Datenpaare (rechts).

8.7 Robuste Hauptkomponentenanalyse

Ausgehend von den allgemeinen Bemerkungen zur Hauptkomponentenanalyse aus Abschnitt 6 werden in der Arbeit von Jackson [10] drei unterschiedliche Methoden kurz vorgestellt, die zur Durchführung der robusten Hauptkomponentenanalyse verwendet werden können.

- Eliminiere zunächst Ausreißer aus den Rohdaten und führe anschließend eine konventionelle Hauptkomponentenanalyse auf den verbliebenen Daten durch.
- Benutze eine robuste Schätzung der Kovarianz- bzw. Korrelationsmatrix und führe eine konventionelle Hauptkomponentenanalyse durch.
- Verwende robuste Schätzungen der Eigenwerte und Eigenvektoren von Kovarianzmatrizen.

Ein Verfahren zur Bestimmung robuster Hauptkomponenten auf der Grundlage robuster Schätzungen der Eigenwerte und Eigenvektoren wird in [2] angegeben und soll im Folgenden vorgestellt werden. Die Berechnung erfolgt ebenfalls iterativ nach folgender Iterationsvorschrift:

(1) Berechne Startschätzung V für die Kovarianzmatrix und berechne den ersten Eigenvektor u_1.

(2) Initialisiere $w_m \equiv 1$.

(3) Bestimme die Hauptkomponenten-Scores $y_m = u_1^\top x_m$.

(4) Berechne M-Schätzungen für den Mittelwert und die Varianz zu y_m (wie in Abschnitt 8.6) iterativ:

$$\bar{y} = \frac{\sum w_m^{(\text{neu})} y_m}{\sum w_m^{(\text{neu})}}$$

$$s^2 = \frac{\sum \left(w_m^{(\text{neu})}\right)^2 (y_m - \bar{y})^2}{\sum \left(w_m^{(\text{neu})}\right)^2 - 1}$$

mit

$$w_m^{(\text{neu})} = w_m^{(\text{neu})}(d_m) = \begin{cases} 1 & \text{, falls} \quad d_m \le d_0 \\ \frac{d_0}{d_m} \exp\left(\frac{-(d_m - d_0)^2}{2 \cdot 1{,}25^2}\right) & \text{, falls} \quad d_m > d_0 \end{cases}$$

und

$$d_m^2 = \frac{(y_m - \bar{y})^2}{s^2}, \qquad d_0 = \sqrt{v} + \sqrt{2} \quad .$$

Als Startwerte werden der Median und $(0,74 \cdot \text{IQR})^2$ empfohlen.

(5) Setze $w_m = \min\{w_m, w_m^{(\text{neu})}\}$ und berechne $\bar{x}$ und V mit dem gerade bestimmten w_m

$$\bar{x} = \frac{\sum w_m x_m}{\sum w_m} \quad ,$$

$$V = \frac{\sum w_m^2 (x - \bar{x})(x - \bar{x})^\top}{\sum w_m^2 - 1} \quad .$$

(6) Berechne den ersten Eigenwert e_1 und den zugehörigen Eigenvektor u_1 von V.

(7) Wiederhole die Schritte (3) - (6) so lange, bis Konvergenz vorliegt.

Zur Bestimmung der weiteren Hauptkomponenten $u_i, 2 \leq i \leq p$, muss man die Daten auf den Raum orthogonal zu den bisherigen Eigenvektoren $u_1, \ldots, u_{i-1}$ projizieren und anschließend die Schritte (2) bis (7) wiederholen. Als Erweiterung ergibt sich hierfür:

(8) Bilde $x_{i,m} = (I - U_{i-1}U_{i-1}^\top)x_m$, wobei $U_{i-1} = (u_1, \ldots, u_{i-1})$.
Wähle als Startwert für den ersten Eigenvektor den zweiten Eigenvektor der letzten Iteration für den vorherigen Eigenvektor.

(9) Wiederhole die Schritte (2) bis (7) mit $x_{i,m}$ anstelle von x_m und bestimme den ersten Eigenvektor u, der dann u_i ist.

Die Schritte (7) bis (9) werden so lange wiederholt, bis alle p Eigenwerte e_i und Eigenvektoren u_i mit entsprechenden Gewichten bestimmt sind.
Schließlich kann eine robuste Schätzung der Kovarianz- bzw. Korrelationsmatrix durch $UEU^\top$ alternativ zu V gefunden werden, die positiv semidefinit ist. Dabei ist U die Matrix, die sich spaltenweise aus den Eigenvektoren u_i zusammensetzt und E eine Diagonalmatrix mit den Eigenwerten e_i als Einträge auf der Hauptdiagonalen.

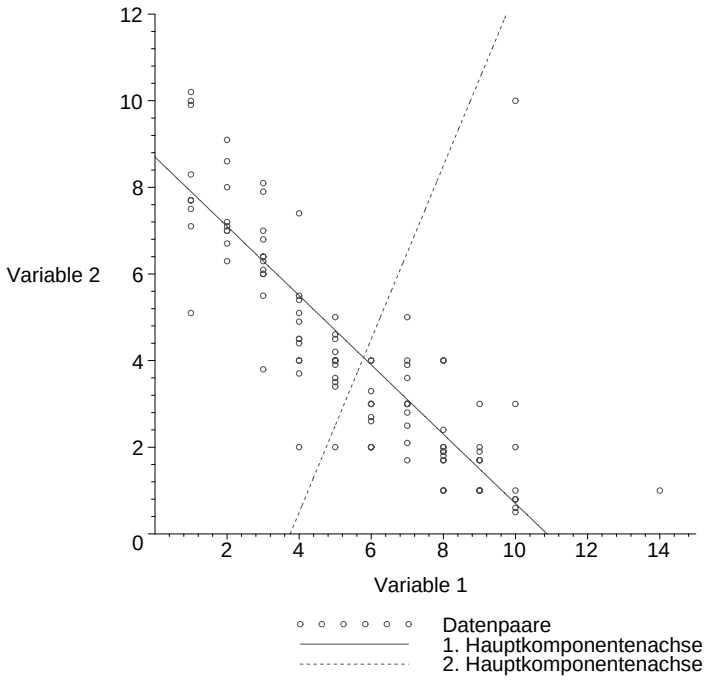


Abbildung 20: 108 von 109 Beobachtungen zweier Variablen und beide Hauptkomponentenachsen.

Für den Datensatz aus Tabelle 7 im Anhang mit 109 Beobachtungen (das Datenpaar $(1,0; 105,0)$ wird in der Grafik nicht berücksichtigt) ergibt sich nach Berechnung der robusten Hauptkomponentenanalyse die Situation wie in Abbildung 20 dargestellt. Das Verfahren zur robusten Hauptkomponentenanalyse nach Campbell wurde ebenfalls in der GALA-Software implementiert und kann als Alternative zur gewöhnlichen Hauptkomponentenanalyse mittels eines Optionsschalters beim Programmaufruf gewählt werden.

Anhang

Daten zur Korrelationsberechnung

Nr.	Var 1	Var 2	Nr.	Var 1	Var 2	Nr.	Var 1	Var 2	Nr.	Var 1	Var 2
1	6,0	2,0	29	4,0	7,4	57	10,0	0,8	85	14,0	1,0
2	7,0	3,0	30	9,0	1,0	58	4,0	4,5	86	10,0	2,0
3	6,0	3,0	31	6,0	2,0	59	4,0	4,5	87	10,0	3,0
4	3,0	6,0	32	6,0	2,0	60	9,0	1,7	88	9,0	2,0
5	4,0	4,9	33	6,0	4,0	61	2,0	7,0	89	4,0	5,5
6	6,0	4,0	34	10,0	1,0	62	1,0	7,7	90	8,0	1,8
7	5,0	5,0	35	1,0	105,0	63	3,0	6,0	91	2,0	8,0
8	8,0	1,9	36	7,0	1,7	64	4,0	2,0	92	8,0	1,7
9	8,0	1,9	37	6,0	3,0	65	4,0	5,4	93	7,0	5,0
10	7,0	2,1	38	4,0	5,1	66	7,0	2,8	94	3,0	7,9
11	5,0	4,2	39	7,0	3,0	67	4,0	3,7	95	2,0	9,1
12	4,0	4,0	40	2,0	8,6	68	5,0	3,6	96	2,0	7,1
13	2,0	7,2	41	3,0	6,3	69	6,0	2,0	97	6,0	3,3
14	1,0	8,3	42	3,0	6,4	70	10,0	10,0	98	5,0	4,5
15	1,0	10,0	43	3,0	5,5	71	5,0	4,0	99	2,0	6,7
16	2,0	7,0	44	1,0	7,1	72	7,0	3,0	100	5,0	4,6
17	2,0	6,3	45	9,0	1,7	73	10,0	0,8	101	10,0	0,6
18	3,0	7,0	46	5,0	2,0	74	4,0	4,4	102	9,0	1,9
19	5,0	4,0	47	3,0	6,8	75	9,0	3,0	103	3,0	8,1
20	1,0	9,9	48	9,0	1,0	76	8,0	2,0	104	5,0	3,4
21	8,0	1,0	49	8,0	4,0	77	7,0	2,5	105	1,0	10,2
22	8,0	2,0	50	9,0	1,0	78	3,0	3,8	106	1,0	7,7
23	8,0	1,0	51	6,0	2,6	79	8,0	1,7	107	1,0	5,1
24	4,0	4,0	52	5,0	3,5	80	9,0	1,0	108	1,0	7,5
25	7,0	3,0	53	6,0	2,7	81	6,0	3,0	109	3,0	6,4
26	3,0	6,1	54	7,0	3,6	82	5,0	4,0			
27	8,0	2,4	55	5,0	3,9	83	8,0	4,0			
28	10,0	0,5	56	7,0	3,9	84	7,0	4,0			

Tabelle 7: Datensatz (109 Beobachtungen)

Literaturverzeichnis

[1] V. Barnett and T. Lewis. *Outliers in statistical data.* John Wiley & Sons, New York, 1979.

[2] N. A. Campbell. *Robust procedures in multivariate analysis I: robust covariance estimation.* Journal of Applied Statistics 1980, 29:231–237.

[3] G. Dikta. *Operations research.* Unpublished script, FH Aachen, 2002.

[4] P. Érdi and J. Tóth. *Mathematical models of chemical reactions: theory and applications of deterministic and stochastic models.* Manchester University Press, 1989.

[5] F. R. Hampel.(1974) *The influence curve and its role in robust estimation.* Journal of the American Statistical Association 1974, 346:383–393.

[6] F. R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, W.A. Stahel. *Robust statistics: The approach based on influence functions.* John Wiley & Sons, New York, 1986.

[7] J. Hartung, B. Elpelt. *Statistik - Lehr- und Handbuch der angewandten Statistik.* R. Oldenbourg Verlag, München, 1995.

[8] D. C. Hoaglin, F. Mosteller, and J. W. Tukey (eds.). *Understanding robust and exploratory data analysis.* John Wiley & Sons, New York, 1983.

[9] P. J. Huber. *Robust statistics.* John Wiley & Sons, New York, 1981.

[10] J. E. Jackson. *A user's guide to principal components.* Wiley-Interscience, 1991.

[11] I. T. Jolliffe. *Discarding variables in a principal component analysis. I: Artificial data.* Applied Statistics 1972, 21:160–173.

[12] I. T. Jolliffe. *Principal component analysis.* Springer-Verlag, New York, 1986.

[13] E. L. Lehmann. *Nonparametrics: statistical methods based on ranks.* Holden-Day, Inc., San Francisco, 1975.

[14] J. Lehn, H. Wegmann. *Einführung in die Statistik.* Teubner Verlag, Stuttgart, 1992.

[15] G. P. McCabe. *Principal variables.* Technometrics 1984, 26:137–144.

[16] A. Marazzi. *Algorithms, routines, and S functions for robust statistics: the FORTRAN library ROBETH with an interface to S-Plus.* Wadsworth & Brooks/Cole Publishing Company, 1993.

[17] L. A. Segel. *Mathematical models in molecular and cellular biology.* Cambridge University Press, New York, 1980.

[18] R. J. Serfling. *Approximation theorems of mathematical statistics.* John Wiley & Sons, New York, 1980.

Klassifikationsalgorithmen für Daten aus der Pharmaindustrie

Tatjana Eitrich

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: t.eitrich@fz-juelich.de

1 Einleitung

Support-Vektor-Maschinen sind moderne Verfahren, welche in das Gebiet der Künstlichen Intelligenz, der Erforschung und Simulation von natürlichem intelligenten Verhalten, eingeordnet werden. Sie realisieren Vorgänge des Maschinellen Lernens, einem zur Zeit sehr bedeutenden Teilgebiet der Künstlichen Intelligenz. Maschinelles Lernen versucht Systeme zu entwickeln, die in der Lage sind, durch Benutzung von Eingabeinformationen neues Wissen zu konstruieren und vorhandenes Wissen zu verbessern. Dieses Forschungsgebiet findet immer stärkeren Zuspruch, da dadurch beispielsweise sehr aufwändige Experimente verkürzt werden können. Noch vor wenigen Jahrzehnten wurde diese Vorstellung kritisch betrachtet, weil nicht abzusehen war, dass sich die Rechenleistung von Computern so rasant entwickeln würde. Beispiele für Realisierungen von Maschinellem Lernen sind Algorithmen zur Handhabung binärer und multipler Klassifikationsaufgaben sowie zur Umsetzung von Regressionsverfahren.

Support-Vektor-Maschinen sind gekennzeichnet durch die Kombination von linearen Lernalgorithmen und hochdimensionalen kerninduzierten Abbildungen. Diese Kombination führt auf mächtige nichtlineare Entscheidungsverfahren. Geeignete Algorithmen zur Implementierung ergeben sich aus der statistischen Lerntheorie und der konvexen Optimierungstheorie. Support-Vektor-Maschinen zeichnen sich dadurch aus, dass sie globale Lösun-

gen erzeugen, was auf dem Gebiet des Maschinellen Lernens nicht selbstverständlich ist. Außerdem ist ihre Generalisierungsfähigkeit sehr gut, weil sie eine besonders günstige Form der Risikominimierung umsetzen. Ihre Anwendungsmöglichkeiten in der Praxis sind sehr vielfältig. Dazu zählen beispielsweise Bild- und Schrifterkennung, Umsatzprognosen für Unternehmen und automatische Textklassifikationen.

Ein wichtiges Ziel des GALA-Projektes besteht in der Untersuchung und Implementierung von Support-Vektor-Maschinen zur binären Klassifikation. Die folgenden Abschnitte setzen sich nach einer allgemeinen Einführung mit dem Aufbau von Support-Vektor-Maschinen für binäre Klassifikationsaufgaben sowie mit den dafür notwendigen mathematischen Theorien auseinander. Ausgehend von diesen Betrachtungen werden Algorithmen zur Umsetzung von Support-Vektor-Maschinen vorgestellt. Diese werden innerhalb des Projektes implementiert, für die speziellen Fragestellungen optimiert und zur Untersuchung aktueller Datensätze verwendet.

2 Überwachtes Lernen

Die leistungsfähigen Computer der heutigen Zeit sollen sehr komplexe Aufgaben lösen. Dabei sorgen Algorithmen dafür, dass genau festgelegt wird, wie ein bestimmtes Ergebnis durch die einfließenden Informationen entstehen soll. Doch es gibt immer mehr Aufgaben, bei denen dieser Zusammenhang zwar besteht, jedoch nicht sofort durch einen Algorithmus formuliert werden kann. Er muss zuerst erkannt oder erlernt werden. Mit Aufgaben dieser Art beschäftigen sich Verfahren des überwachten Lernens, die man in das Gebiet des Maschinellen Lernens einordnet. Maschinelles Lernen umfasst drei unterschiedliche Methoden:

(1) Überwachtes Lernen (supervised learning) gibt eine bestimmte Menge von Beispielen vor, die einen funktionalen Zusammenhang widerspiegeln. Auf deren Grundlage soll eine Eingabe-Ausgabe-Beziehung erlernt werden.

(2) Unüberwachtes Lernen (unsupervised learing) versucht, eine bestimmte Struktur in gegebenen Daten zu erkennen. Im Unterschied zu überwachtem Lernen gibt es keine vorgegebenen Ausgabewerte, nur der Aufbau der Eingabedaten soll nachvollzogen werden.

(3) Verstärkungslernen (reinforcement learning) findet Anwendung in der Spieltheorie. Ein Agent soll in Experimenten gute Handlungen erlernen, indem er für Erfolge belohnt und für Misserfolge bestraft wird.

Die in diesem Abschnitt betrachtete Methode soll die des überwachten Lernens sein.

2.1 Einordnung und Definitionen

Zwischen 1955 und 1957 wurde von Allen Newell und Herbert Alexander Simon der Logic Theorist entwickelt, ein Programm, das von vielen als erste Form Künstlicher Intelligenz überhaupt betrachtet wird [20]. Der Logic Theorist stellte Entscheidungsprobleme in einem dafür generierten Baum dar und verfolgte dann gemäß der erlernten Vorschrift immer denjenigen Ast, der zur Lösung des Problems führte. Im Jahr 1956 entwarf Frank Rosenblatt ein Modell, welches ebenfalls einen überwachten Lernprozess durchlaufen konnte [5]. Sein Perceptron löste binäre Klassifikationsprobleme, indem es Parameter erlernte, mit denen die Eingabewerte gewichtet wurden, bevor man sie summierte. Dieses Modell wurde mehr als zwanzig Jahre später mit dem Begriff des neuronalen Netzes identifiziert. Die Theorie der neuronalen Netze begann sich rasant ab 1986 zu entwickeln. Das Jahr der Entstehung von Support-Vektor-Maschinen wird auf etwa 1992 datiert. Dabei wurden zwei Ideen aus den sechziger Jahren kombiniert – die implizite Berechnung von Skalarprodukten zwischen transformierten Daten über Funktionen im Eingaberaum sowie die Theorie der optimal trennenden Hyperebene [33]. Diese ließen ein Verfahren völlig neuer Form entstehen.

Lernalgorithmen für überwachtes Lernen haben die Aufgabe, sogenannte Hypothesenfunktionen zu entwickeln und können auf eine Vielzahl von Problemen angewandt werden. Je nach Struktur der Ausgabewerte, die eine solche Funktion liefert, kann man die Algorithmen in drei Gruppen einteilen:

(1) Ausgabewerte aus $\{-1, 1\} \rightarrow$ binäre Klassifikationsprobleme,

(2) Ausgabewerte aus endlich vielen Zahlen $\rightarrow$ multiple Klassifikationsprobleme,

(3) Ausgabewerte aus den reellen Zahlen $\rightarrow$ Regressionsprobleme.

Die Wahl eines bestimmten Modells ist also von den zur Verfügung stehenden Daten abhängig. Innerhalb des Projektes werden im Allgemeinen binäre Klassifikationsprobleme definiert, untersucht und gelöst. Die Erweiterung der Lösungskonzepte bis hin zur Regression ist möglich und wird in der Literatur ausführlich erläutert. Zu multiplen Klassifikationsaufgaben siehe zum Beispiel [13], Regression wird unter anderem in [32], Kapitel 9 vorgestellt. Die drei schon erwähnten Modelle - Entscheidungsbäume, neuronale Netze und Support-Vektor-Maschinen - sind für binäre Klassifikationen anwendbar. Sie alle implementieren überwachtes Lernen. Das folgende kleine Beispiel zur Textklassifikation soll kurz verdeutlichen, warum man Algorithmen braucht, welche diese Form der Künstlichen Intelligenz umsetzen.

Jeder für eine Sammlung wissenschaftlicher Arbeiten eintreffende Text soll auf Eignung für die Aufnahme in die Datenbank überprüft werden. Das kann unter Umständen sehr zeitaufwändig sein. Kann man aber einen Algorithmus entwickeln, der anhand bestimmter Wortkombinationen ungeeignete Texte sofort ablehnt oder bei Auftreten bestimmter Schlüsselworte eine weitere Prüfung empfiehlt, spart man sich viel Arbeit. Dazu benötigt

man eine repräsentative Sammlung aufgenommener sowie abgelehnter Texte. Diese stellt man einem Lernalgorithmus zur Verfügung, der dann eine Hypothese entwickelt, mit der alle Texte, die in Zukunft eintreffen, bewertet werden.

Mit $X \subseteq \mathbb{R}^n$ $(n \in \mathbb{N})$ wird im Folgenden immer der Raum der Eingabevektoren bezeichnet. Wir suchen ein Verfahren, welches Vektoren aus X genau einer der beiden Klassen 1 und -1 zuordnet. Wir sprechen dabei auch von positiver und negativer Klasse. Das trägt der Tatsache Rechnung, dass man oft gewisse gute und schlechte Daten identifizieren und trennen will. Was genau mit „gut“ und „schlecht“ gemeint ist, hängt immer vom vorgegebenen Problem ab. Ein für die Medizin typisches Klassifikationsproblem beschäftigt sich beispielsweise mit der Frage, ob bestimmte Medikamente oder Substanzen einer erkrankten Person helfen können oder nicht.

Für einen Eingabevektor $\boldsymbol{x}$ aus X sei $y \in \{-1, 1\}$ der vorgegebene Ausgabewert. Dann heißt $(\boldsymbol{x}, y)$ Eingabe-Ausgabe-Paar für binäre Klassifikationsaufgaben. Solche Eingabe-Ausgabe-Paare bilden die praktische Grundlage eines lernenden Systems und beeinflussen neben dem gewählten Modell die Güte einer erlernten Funktionalität. Sie sollten möglichst zahlreich vorhanden, repräsentativ und fehlerfrei sein.

Sei $l \in \mathbb{N}$. Ein l-Tupel von Eingabe-Ausgabe-Paaren $\left\{(\boldsymbol{x}^1, y_1), ..., (\boldsymbol{x}^l, y_l)\right\}$ wird im Folgenden als Datensatz S_l bezeichnet. Wird S_l zum Erlernen einer Funktion benutzt, spricht man vom sogenannten Trainingsdatensatz; die l Datenpaare bezeichnet man dann auch als Trainings- oder Lernpaare. Wird S_l jedoch dazu benutzt, für eine feste Funktion zu zeigen, dass diese sinnvolle Ergebnisse produziert, spricht man vom Testdatensatz. Ein Datensatz wird nur mit S bezeichnet, falls dessen Größe variiert.

Überwachte Lernvorgänge setzen die Existenz von Trainings- und Testdaten voraus, die voneinander verschieden sein sollten. Der Grund dafür ist, dass jeder Test von Lernalgorithmen unabhängig vom Lernvorgang sein muss. In der Praxis wird der Trainingsdatensatz zusätzlich gesplittet, damit Teile der Trainingsdaten der Anpassung wichtiger Modellparameter dienen können. Die Details dazu werden später beschrieben. Die Menge der Funktionen, welche einem Lernalgorithmus als Kandidaten zur Auswahl stehen, wird Raum der Zielfunktionen $\mathcal{H}$ genannt.

Die Wahl eines geeigneten Raumes $\mathcal{H}$ für Lernalgorithmen ist nicht trivial. Er darf nicht zu klein sein, denn sonst gibt es keine Funktion in $\mathcal{H}$, die in der Lage ist, komplizierte Zusammenhänge darzustellen. Wenn er aber zu reichhaltig ist, dauert die Suche zu lange oder es entsteht Überanpassung. Als überangepasstes Modell bezeichnen wir eine Trennfunktion, die auf den Trainingsdaten eine hervorragende Performance zeigt, jedoch auf neuen Testdaten versagt. Überanpassung ist ein großes Problem innerhalb des überwachten Lernens. Besteht $\mathcal{H}$ aus affin-linearen Funktionen, wie beispielsweise bei der linearen Regression, ist es unmöglich, nichtlineare Abhängigkeiten zwischen Eingaben und Ausgaben darzustellen. Oft hat man es mit Daten zu tun, die experimentell ermittelt wurden. Sie können fehlerhaft und unvollständig sein. Es kann auch vorkommen, dass der funktionale Zusammenhang nicht eindeutig ist oder nicht existiert, beispielsweise wenn Variablen

untersucht worden sind, die mit der Klasseneinteilung nichts zu tun haben. Dann wird es umso schwerer, gute Modelle zu produzieren. Es gibt aber Ansätze, die genau solche Probleme auffangen und unter abgeschwächten Bedingungen lernen. Derartige Methoden werden wir im Zusammenhang mit Support-Vektor-Maschinen im Abschnitt 3 vorstellen.

In den nächsten vier Abschnitten werden Modelle für die Implementierung überwachter Lernvorgänge vorgestellt. Mit diesen wurden in den letzten Jahren und Jahrzehnten dauerhaft gute Ergebnisse für binäre Klassifikationsaufgaben erzielt und sie sind in der Praxis weit verbreitet. Für Klassifikationsaufgaben innerhalb des Projektes ist unter anderem das mächtige Verfahren der Support-Vektor-Maschinen gewählt worden. Die Gründe dafür werden im Anschluss motiviert, eine detaillierte Beschreibung folgt im Abschnitt 3.

2.2 Lineare Klassifikationsalgorithmen

Lineare Verfahren für binäre Klassifikationsaufgaben definieren einen sehr einfachen Raum der Zielfunktionen $\mathcal{H}$. Dieser enthält nur Funktionen der Form

$$f_{\text{lin}}(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle_2 + b = \sum_{k=1}^{n} w_k x_k + b \quad (\boldsymbol{x} \in X) \,, \tag{1}$$

wobei $\boldsymbol{w} \in \mathbb{R}^n$ und $b \in \mathbb{R}$ zunächst unbekannt sind und vom Lernalgorithmus mit Hilfe der Lernpaare bestimmt werden sollen. Einem beliebigen Eingabevektor $\boldsymbol{x}$ aus dem Raum X kann dann mittels (1) ein Funktionswert zugeordnet werden. Die Zielfunktion f_{lin} ist reell, also $f_{\text{lin}} : X \to \mathbb{R}$. Wie entsteht daraus eine binäre Klassifikation? Ein oft angewendetes Verfahren wird jetzt definiert, zunächst folgt noch ein technisches Detail.

In diesem Bericht wird ausschließlich die abgewandelte Signumfunktion

$$\text{sgn}(t) := \begin{cases} 1 & \text{falls } t \geq 0 \\ -1 & \text{sonst} \end{cases} \qquad (t \in \mathbb{R})$$

verwendet.

Im Folgenden wird unter einer linearen Entscheidungsvorschrift $h_{\text{lin}}(\cdot)$ die Hintereinanderausführung einer linearen Zielfunktion f_{lin} und der Signumfunktion verstanden, formal:

$$h_{\text{lin}}(\cdot) := \text{sgn}(f_{\text{lin}}(\cdot)) \,.$$

Eine lineare Entscheidungsvorschrift ordnet $\boldsymbol{x} = (x_1, ..., x_n) \in X$ der Klasse 1 zu, falls $f_{\text{lin}}(\boldsymbol{x}) \geq 0$ gilt, anderenfalls wird angenommen, $\boldsymbol{x}$ gehörte in die Klasse -1. Die Hyperebene $f_{\text{lin}} \equiv 0$ teilt den Raum X in zwei Teile. Man hofft, dass dadurch auch die zwei Klassen gut getrennt sind. Überprüfen kann man das allerdings nur für vorhandene Eingabe-Ausgabe-Paare.

Die Datenpaare eines Trainingsdatensatzes heißen linear separabel (linear trennbar), falls eine lineare Entscheidungsvorschrift existiert, welche alle diese Paare der richtigen Klasse zuordnet, also falls $h_{\text{lin}}(\boldsymbol{x}^i) = y_i$ für alle $i = 1, ..., l$ gilt. Man sagt auch, es existiert eine trennende Hyperebene für diese Datenpunkte.

Sei $(\boldsymbol{x}^i, y_i)$ ein Trainingspaar. Als funktionalen Abstand dieses Paares bezüglich einer trennenden Hyperebene $f_{\text{lin}} \equiv 0$ mit Parametern $\boldsymbol{w}$ und b bezeichnet man den Wert

$$\gamma_i := y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i \rangle_2 + b) = y_i \cdot f_{\text{lin}}(\boldsymbol{x}^i) \; . \tag{2}$$

Für alle Lernpaare eines Trainingsdatensatzes gilt $\gamma_i \in \mathbb{R}$ $(i = 1, ..., l)$. Falls $\gamma_i > 0$, so gilt $\text{sgn}(f_{\text{lin}}(\boldsymbol{x}^i)) = y_i$ und die Funktion f_{lin} hat die Eingabe $\boldsymbol{x}^i$ richtig klassifiziert. Im Falle $\gamma_i = 0$ ist y_i zu prüfen.

Abbildung 1 zeigt einen Punkt, der durch eine erlernte Funktion f nicht korrekt klassifiziert wurde und deshalb einen negativen funktionalen Abstand hat.

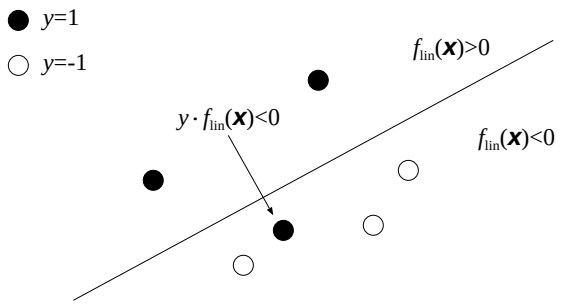


Abbildung 1: Lage eines Punktes mit negativem funktionalen Abstand γ.

Man betrachte nun die normalisierte Funktion f_{lin} mit Parametern $\left(\frac{1}{\|\boldsymbol{w}\|_2}\boldsymbol{w}, \frac{1}{\|\boldsymbol{w}\|_2}b\right)$. Dann kann man auch hier für alle Lernbeispiele die funktionalen Abstände berechnen. In diesem Fall werden sie als geometrische Abstände γ_i^g $(i = 1, ..., l)$ bezeichnet.

Als (funktionale) Marge oder auch Trenngüte γ einer Hyperebene bezüglich eines Trainingsdatensatzes S_l bezeichnet man das Minimum aller funktionalen Abstände γ_i, $1 \leq i \leq l$. Analog dazu ist die geometrische Marge oder auch geometrische Trenngüte γ^g einer Hyperebene bezüglich S_l definiert als das Minimum aller geometrischen Abstände γ_i^g, $1 \leq i \leq l$.

Wie man sieht, ist die Marge als Begriff nur dann sinnvoll, wenn alle Punkte richtig klassifiziert wurden. Dann ist sie positiv, und alle geometrischen Abstände können als Euklidische Distanzen zwischen Punkten und Hyperebene interpretiert werden.

Es ist möglich, dass man es mit Trainingsdaten zu tun hat, die nicht linear separabel sind. Man sucht dann intuitiv die bestmögliche Lösung, bei der so wenige Trainingspunkte wie möglich falsch zugeordnet werden. Auskunft über die Güte der Zuordnungen geben sogenannte Schlupfvariablen (slack variables). Man gibt sich eine gewünschte Marge $\gamma > 0$

vor und berechnet, wie stark diese von den funktionalen Abständen der Trainingspunkte verfehlt wird.

Sei $i \in \{1, ..., l\}$. Die Variable ξ_i, die definiert ist als

$$\xi_i := \max \{0, \gamma - y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i \rangle_2 + b)\}$$

mit vorgegebenem $\gamma > 0$, nennt man Schlupfvariable des Lernpaares $(\boldsymbol{x}^i, y_i)$.

Falls $\xi_i = 0$ gilt, liegt der i-te Punkt des Trainingsdatensatzes weit genug von der Hyperebene entfernt, andernfalls erfüllt er die Bedingung nicht. Falls dann zusätzlich $\xi_i > \gamma$ gilt, ist der i-te Eingabevektor falsch klassifiziert worden. Im Bereich $\xi_i \in (0, \gamma)$ liegen Punkte, die zwar richtig zugeordnet wurden, sich aber zu nah an der Hyperebene befinden.

Man beachte, dass während der Trainingsphase meist nur wenige Vertreter einer Gesamtheit von möglichen Punkten zur Verfügung stehen. Die trennende Hyperebene, die auch zur Theorie der Support-Vektor-Maschinen genutzt wird, stellt nur eine Annahme der Klassenaufteilung dar. Falsche Klassifikationen innerhalb eines Testdatensatzes durch ein lineares Modell können verschiedene Ursachen haben:

- Die Funktionalität ist nichtlinear, das Modell ist ungeeignet.
- Es gibt zu wenig Trainingsdaten, um die Klassen gut zu trennen.
- Die Klassen lassen sich auch mit vielen Daten nicht gut voneinander trennen, beispielsweise wenn die Daten stark verrauscht sind.
- Die eingeführten Variablen sind nicht für die beobachtete Klasseneinteilung der Trainingsdaten verantwortlich.
- Die Zuordnung der Trainingsdaten zu Klassen entspricht nicht der Realität. Beispiel: Wird an Testpersonen untersucht, ob ein bestimmter Wirkstoff Schmerzen lindert, können diese bei der Beurteilung dem Placebo- oder dem Nocebo-Effekt unterliegen und das Ergebnis stark verfälschen. Der Nocebo-Effekt ist das Gegenstück zum bekannten Placebo-Effekt. Es kommt oft vor, dass in Erwartung eines wirkungslosen oder schädigenden Medikaments auch nachweisbar wirksame, verträgliche Medikamente ihre Wirkung ganz oder teilweise einbüßen und andere Beschwerden auslösen.

Support-Vektor-Maschinen sowie die ersten einfachen Modelle neuronaler Netze basieren auf linearen Klassifikationsalgorithmen. Informationen zu den Erweiterungen, die dann zu diesen Verfahren führten, folgen später.

2.3 Entscheidungsbäume

Lernalgorithmen auf Basis von Entscheidungsbäumen sind einfach, erfolgreich und leicht zu implementieren. Sie waren die ersten Verfahren für die Umsetzung von Klassifikationsaufgaben mittels Computern. Abbildung 2 zeigt einen einfachen Entscheidungsbaum und soll den prinzipiellen Aufbau verdeutlichen.

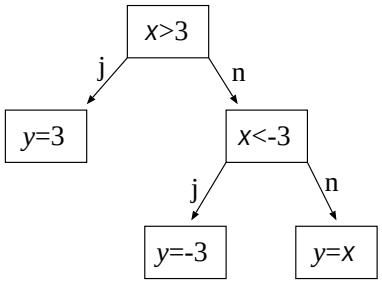


Abbildung 2: Entscheidungsbaum.

Dort soll

$$y = \max\ \{-3, \min\ \{3, x\}\} \quad (x \in \mathbb{R})$$

berechnet werden. Alle Knoten in einem Baum, hier sind es zwei, entsprechen einem Test. Je nach Ergebnis des Tests muss einer der möglichen Wege eingeschlagen werden. In diesem Beispiel gibt es immer nur zwei Wege, es können aber auch mehr sein. Jedes Blatt des Baumes stellt eine Entscheidung des Problems dar, das heißt, sobald ein Blatt erreicht wird, ist das Ausgangsproblem gelöst.

Hat ein Entscheidungsbaum seine endgültige Form schon angenommen, ist es nicht schwer, neue Daten zu klassifizieren. Man startet in der Wurzel und sucht den passenden Pfad. Dabei kann man immer einen solchen finden, da bei jedem Test alle möglichen Ergebnisse abgedeckt sein müssen. Handelt es sich zum Beispiel um den Test, welche Werte eine Variable annimmt, darf man nicht auf Alternativen *kleiner eins* oder *größer eins* testen, wenn es vorkommen kann, dass diese auch gelegentlich *genau eins* ist. Das muss bei Generierung jedes Baumes beachtet werden. Aber auch als Anwender eines festen Baumes kann man beispielsweise keine reellen Zahlen verwenden, wenn einer der Tests nur ganzzahlige Werte verarbeiten kann.

Bisher wurde nur von Bäumen mit fester Struktur gesprochen, die für ein bestimmtes Problem angepasst wurden. Aber wie generiert man einen solchen Baum? Wie schon beschrieben, stehen dafür die Eingabe-Ausgabe-Paare zur Verfügung. Aufgrund der Verzweigungen eines Baumes kann die Anzahl der Variablen in den Daten jedoch nichts darüber aussagen, wieviele Tests in einem Baum auftreten. Man weiß nur, dass an jedem Blatt einer der möglichen Ausgabewerte stehen muss, hier sind es zwei mögliche. Zunächst versucht man, einen Baum zu finden, sodass dieser für alle Paare des Trainingsdatensatzes bei erneuter Klassifikation die vorgegebenen Ausgabewerte liefert. Diese Aufgabe ist trivial. Man konstruiert einen Baum, der für jedes Paar einen Pfad zu einem Blatt mit dem gewünschten Ausgabewert besitzt. Dieser Entscheidungsbaum kann sehr aufgebläht sein. Außerdem ist intuitiv klar, dass er zu schlechten Ergebnissen führen kann. Dazu sei auf sogenannte übertrainierte Algorithmen verwiesen. Wie hier, können sich solche im Extremfall nur die Trainingsdaten „merken“. Von Künstlicher Intelligenz kann dann nicht mehr gesprochen werden. Daher sollte man zusätzlich versuchen, einen Baum aufzubauen, der eine Vielzahl von möglichen Fällen prägnant beschreiben kann, also bestimmte Varia-

blen auswählt, für die die oben erwähnten Tests durchgeführt werden sollen. Dabei muss man unter Umständen Fehler beim Training in Kauf nehmen.

Der auf Seite 105 angegebene Algorithmus verdeutlicht, wie man einen ersten sinnvollen Entscheidungsbaum generiert.

1.	S seien die aktuell betrachteten Lernpaare, $\mathcal{V}_{\text{aktuell}}$ die aktuelle Menge der Variablen und d_{aktuell} der zu verwendende Vorgabewert.
2.	S enthält keine Daten $\rightarrow$ Gib d_{aktuell} zurück, STOP. Alle y_i für Paare in S haben den gleichen Wert $y \rightarrow$ Gib y zurück, STOP. $\mathcal{V}_{\text{aktuell}} = \emptyset \rightarrow$ Gib den Ausgabewert zurück, der in S am häufigsten auftritt (Mehrheitsprinzip), STOP.
3.	Bestimme $\text{var}_{\text{aktuell}}$, die wichtigste Variable aus $\mathcal{V}_{aktuell}$. Benutze dafür S. Lege eine neue Wurzel mit Test auf $\text{var}_{\text{aktuell}}$ an.
4.	Bestimme alle κ unterschiedlichen Werte v_k von $\text{var}_{\text{aktuell}}$ in S.
5.	Führe 6. für alle $k = 1, ..., \kappa$ parallel und unabhängig aus.
6.	S_k bestehe nur aus den Lernpaaren $(\boldsymbol{x}^i, y_i)$ von S mit $\text{var}_{\text{aktuell}}(\boldsymbol{x}^i) = v_k$, $\mathcal{V}_{\text{aktuell}} = \mathcal{V}_{\text{aktuell}} \setminus \{\text{var}_{\text{aktuell}}\}$. Hänge einen Zweig mit Ausgabewert v_k an den Baum. Bestimme den aktuellen Teilbaum k an Wurzel $\text{var}_{\text{aktuell}}$ durch Start bei 1. mit S_k, $\mathcal{V}_{\text{aktuell}}$ und dem Vorgabewert d^*_{aktuell} als der Ausgabe, die in S_k die Mehrheit hat.

Tabelle 1: Basisalgorithmus zur Generierung eines Entscheidungsbaumes.

Der Algorithmus startet mit dem l-Tupel

$$S_l = \left\{(\boldsymbol{x}^1, y_1), ..., (\boldsymbol{x}^l, y_l)\right\}$$

von Lernpaaren, der Menge $\mathcal{V} := \{\text{var}_i,\ i = 1, ..., n\}$ aller Variablen sowie einem Vorgabewert d aus $\{-1, 1\}$ für die Entscheidung. Man beginne in Schritt 1. und arbeite al-

le weiteren Schritte ab. Die im dritten Schritt angegebene Bestimmung der sogenannten wichtigsten Variablen wurde noch nicht erklärt, es sollte immer die „beste“ der noch freien Variablen sein. Die Auswahl wichtiger Variablen als Wurzeln aller Teilbäume sowie die Suche einer geeigneten Variablen als Wurzel des gesamten Baumes werden vorgenommen, um die Tiefe des Baumes zu minimieren. Man versucht Variablen zu bestimmen, welche die Klassentrennung besonders gut erklären. Eine perfekte Variable würde also alle Trainingsdaten schon richtig trennen. Eine sehr schlechte Variable dagegen trägt zu keiner Verkleinerung des Problems bei. Aber wie kann man diese Wichtigkeit messen und ordnen? Dazu sollte man sich ein sinnvolles Maß definieren. Die beste Variable sollte dieses Maß maximieren, die schlechteste minimieren. Ein denkbares Maß kann zum Beispiel der erwartete Informationsgewinn sein, den man von einer Variablen und ihren Realisierungen erwarten kann. Als Vorschlag für die Suche nach guten Variablen wird im Folgenden [31] zitiert. Alle bisher benutzten Symbole gelten auch weiterhin.

Die wichtigste Variable in Schritt drei des Basisalgorithmus kann man bestimmen durch

$$\mathrm{var}_{\mathrm{aktuell}} = \underset{\mathrm{var}_i \in \mathcal{V}_{\mathrm{aktuell}}}{\arg\max}\, G(\mathrm{var}_i)$$

mit

$$G(\mathrm{var}_i) = I\left(\frac{S^+}{S^+ + S^-}, \frac{S^-}{S^+ + S^-}\right) - R(\mathrm{var}_i)\ .$$

Die Funktionen I und R seien definiert als

$$I(a,b) := -a\log_2(a) - b\log_2(b)$$

mit $a,\ b \in \mathbb{R}$ sowie

$$R(\mathrm{var}_i) := \sum_{k=1}^{\kappa} \frac{S_k^+ + S_k^-}{S^+ + S^-} \cdot I\left(\frac{S_k^+}{S_k^+ + S_k^-}, \frac{S_k^-}{S_k^+ + S_k^-}\right)\ .$$

Dabei haben die Symbole folgende Bedeutungen:

- S^+ (S^-) ist die Anzahl der Paare in S mit $y = 1$ $(y = -1)$ und
- S_k^+ (S_k^-) ist die Anzahl der Paare in S_k mit $y = 1$ $(y = -1)$.

Man beachte, dass ein Tupel S_k immer von der gewählten Variablen und deren Realisierungen auf den an dieser Stelle betrachteten Lernpaaren abhängig ist, siehe dazu die Schritte $4.$ und $5.$ im Algorithmus.

Welchen Hintergrund haben I, R und G? Man stelle sich einen Punkt vor, den man klassifizieren will. Welche Informationen benötigt man für eine korrekte Antwort? Vor den Tests kann man nur Schätzer für die Wahrscheinlichkeiten, in eine bestimmte Klasse zu fallen, angeben. Diese bauen auf den Trainingsdaten auf und sind einfach die relativen Häufigkeiten positiver und negativer Punkte, denn weitere Informationen sind zu diesem

Zeitpunkt nicht verfügbar. Betrachtet man dann eine Variable $\mathrm{var}_{\mathrm{aktuell}}$, kann diese nur einen Teil der benötigten Informationen liefern, denn es wird angenommen, dass perfekte Variablen nicht vorkommen. Die genaue Höhe der durch $\mathrm{var}_{\mathrm{aktuell}}$ zur Verfügung gestellten Informationen bestimmt man als Differenz der Informationen, welche vor und nach dem Test für eine korrekte Klassifikation benötigt werden und nennt diese G wie Gewinn. Dazu berechnet die Funktion R, wieviel Information im Durchschnitt nach einem Test auf eine bestimmte Variable für eine korrekte Klassifikation vorhanden sein muss. R steht für Rest und stellt die Summe der Informationen eines jeden Zweiges gewichtet mit dem Anteil der Lernpaare, die vom Testknoten aus diesen Zweig besuchen, dar. Eine besonders gute Variable zeichnet sich durch einen hohen Gewinn aus.

Ist ein Entscheidungsbaum generiert, kann man diesen Baum auch noch sinnvoll verkleinern. Um wie schon erwähnt nicht alle Variablen in die Entscheidung einzubauen, entfernt man diejenigen, die als irrelevant erkannt werden können. So kann man auch vermeiden, einen zu stark an die Trainingsdaten angepassten Baum zu erhalten. Dieses sogenannte Beschneiden (pruning) ist beispielsweise in der Software *C4.5* umgesetzt. Zu den Details lese man in [31] oder in [30] nach. Dabei werden nur Teilbäume abgeschnitten, die Grundstruktur des Baumes wird nicht verändert.

Wie man im vierten Schritt des Basisalgorithmus erkennen kann, wird zunächst davon ausgegangen, dass die Realisierungen aller Variablen nur eine bestimmte überschaubare Anzahl von Werten annehmen. In der Realität gibt es aber viele Experimente, in denen Variablen für jede Messung eine andere reelle Zahl annehmen. Um derartige Daten mit Entscheidungsbäumen bearbeiten zu können, muss der Basisalgorithmus geändert werden. Realisierungen stetiger Variablen kann man beispielsweise in sinnvolle Gruppen zusammenfassen [31] und somit diskretisieren.

2.4 Neuronale Netze

Neuronen sind Zellen, die elektrische Signale im menschlichen Gehirn verarbeiten. Diese Strukturen der Biologie wurden in die Informatik übernommen. Die Neuroinformatik rechtfertigt dieses Vorgehen damit, dass das Gehirn des Menschen das einzige existierende lernende intelligente System ist und deshalb intelligente Systeme des maschinellen Lernens nur analog zum Gehirn funktionieren können. Neuronen, mit denen man es bei Lernalgorithmen zu tun hat, sind informationsverarbeitende Einheiten.

Nach der allgemeinsten Definition sind neuronale Netze gerichtete Graphen von Neuronen. Die Kanten solcher Graphen stellen gewichtete Verbindungen zwischen Ausgaben eines Neurons und Eingaben eines anderen dar. Lernalgorithmen auf dieser Grundlage versuchen, diese Gewichte sinnvoll anzupassen. Neuronen eines Netzes sind immer in mindestens zwei Schichten angeordnet. Man unterscheidet Eingabeneuronen zur Signalaufnahme, Ausgabeneuronen zur Signalabgabe und Zwischenneuronen zur Signalübertragung. Letztere liegen in sogenannten verborgenen Schichten und sind nicht an den Ein-

und Ausgaben des Modells beteiligt. Je nach Richtung der Signalübertragungen werden zwei Gruppen von Netzen unterschieden:

- Rückgekoppelte neuronale Netze (feedback neural networks) und
- Vorwärtsgerichtete neuronale Netze (feedforward neural networks).

Die meisten Vertreter der zweiten Gruppe sind sogenannte mehrschichtige Netze (multi layer networks), bei denen es keine Verbindungen innerhalb der Schichten gibt. Neuronale Netze für binäre Klassifikation können mehrschichtig sein, müssen aber genau zwei Ausgabewerte zur Verfügung stellen. Die Begriffe neuronales Netz und Perceptron werden teilweise im gleichen Kontext verwendet, siehe dazu die Ausführungen in [10]. Im Allgemeinen gelten Perceptrons jedoch als diejenigen Netze, die keine Zwischenschichten besitzen. So wird es in [31] erklärt und auch hier verwendet.

Was geschieht mit den Eingabewerten in den Zwischenschichten? In jedem Neuron werden die eintreffenden gewichteten Eingabedaten summiert. Danach wird diese Summe zur Auswertung einer nichtlinearen Komponente, der sogenannten Aktivierungsfunktion a, benutzt. Der so berechnete Wert verlässt das Neuron als Ausgabe. Ein Netz mit fester Struktur kann im Gegensatz zu einem Baum durch Variierung der Funktion a unterschiedliche Ergebnisse liefern, innerhalb eines Netzes ist diese dann aber für alle Neuronen bindend. Oft verwendete Aktivierungsfunktionen sind

$$a(t) = \begin{cases} 1 & \text{falls } t \geq c \ (c \text{ konstant}) \\ -1 & \text{sonst} \end{cases} \qquad \text{und} \quad a(t) = \frac{1}{1 - e^{-t}} \ .$$

Abbildung 3 stellt ein Perceptron für binäre Klassifikation dar.

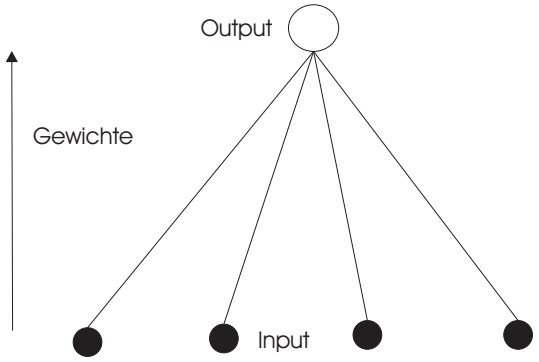


Abbildung 3: Graphische Darstellung eines Perceptrons.

Die vier Knoten im unteren Teil der Abbildung bedeuten, dass immer genau vier Variablen in den Algorithmus eingehen sollen. Im oberen Teil wird die berechnete Klasse ausgegeben. Jedes Element einer Eingabe $\boldsymbol{x} \in X$ soll in genau ein Eingabeneuron eingehen. Im einfachen Fall des Perceptrons wird der Trainingsdatensatz dazu verwendet, Gewichte

$w_j \in \mathbb{R}$ $(j = 1, ..., n)$ und eine Konstante $b \in \mathbb{R}$ zu bestimmen, die zu einer Hypothesenfunktion

$$h(\boldsymbol{x}) = \mathrm{sgn}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle_2 + b)$$

führen.

Anfang der sechziger Jahre des letzten Jahrhunderts wurde untersucht, wie man alle

$$w_j \quad (1 \leq j \leq n)$$

simultan bestimmen könnte. Frank Rosenblatt schlug den in Tabelle 2 angegebenen Algorithmus vor. Gegeben sei dafür S_l sowie ein $\eta \in \mathbb{R}_+$. η ist ein Modellparameter und wird Lernrate genannt. Das Verfahren bestimmt iterativ den Vektor $\boldsymbol{w}$ sowie die Konstante b.

1.	$\boldsymbol{w}^0 = 0,\ b_0 = 0,\ k = 0,\ R = \max\limits_{1 \leq i \leq l} \|\boldsymbol{x}^i\|$
2.	$i = 1$, $\mathrm{err} = 0$
3.	Falls $y_i \cdot (\langle \boldsymbol{w}^k, \boldsymbol{x}^i \rangle_2 + b_k) \leq 0$, dann gehe zu 4., sonst gehe zu 6.
4.	$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \eta y_i \boldsymbol{x}^i,\ b_{k+1} = b_k + \eta y_i R^2$
5.	$\mathrm{err} = 1,\ k = k + 1$
6.	$i = i + 1$, falls $i \leq l$ gehe zu 3., sonst gehe zu 7.
7.	Falls $\mathrm{err} = 1$ gehe zu 2., sonst STOP.

Tabelle 2: Lernen im Perceptron.

Die Struktur des so generierten Perceptrons reicht nicht aus, um komplizierte Zusammenhänge zu erlernen, zum Beispiel um Punkte zu trennen, die nicht linear trennbar sind. Deshalb wendet man sich den schon erwähnten mehrstufigen Netzen zu. In Abbildung 4 ist ein zweischichtiges neuronales Netz (two layer feedforward network) dargestellt.

Eingabeneuronen werden bei Angabe der Schichten im Allgemeinen nicht gezählt, denn sie verarbeiten keine Daten. Sowohl die Struktur als auch die Aktivierungsfunktion sollten zu Beginn des Trainings fest gewählt sein. Damit besteht die Aufgabe eines Lernalgorithmus auf Grundlage eines neuronalen Netzes nur in der Anpassung der Gewichte. Aber wer bestimmt diese Struktur? Die Bestimmung der Zahl der Neuronen und deren Anordnung in Schichten ist nicht einfach. Dazu gibt es zwei praktische Ansätze:

- Man startet mit einem sehr aufgeblähten Netz und verkleinert es iterativ.
- Man startet mit einem sehr einfachen Netz und vergrößert es iterativ.

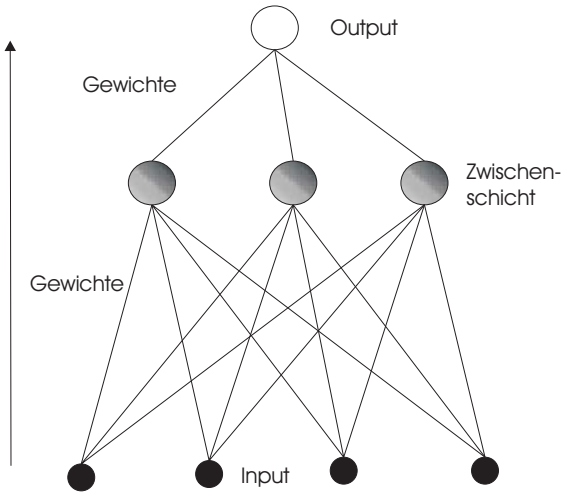


Abbildung 4: Zweischichtiges neuronales Netz.

Beim zweiten Ansatz kann man im Extremfall mit nur einem Neuron starten und von dort aus ein Netz aufbauen. Diese Vorgehensweise ähnelt dem vorgestellten Algorithmus zur Generierung eines Entscheidungsbaumes. Nach der Festlegung der Struktur agiert der eigentliche Lernalgorithmus. Der bekannteste Algorithmus zum Trainieren mehrschichtiger Netze wird zum Abschluss kurz vorgestellt.

Die Methode des back propagation (Rückwärts-Weitergabe) [31] wurde schon im Jahre 1969 entwickelt. Aufgrund zu schwacher Rechenleistung der damaligen Computer wurde sie jedoch abgelehnt und nicht genutzt. Ab dem Jahr 1986 erhielt der Algorithmus verstärkt Zuspruch. Zu dieser Zeit waren die Rechenkapazitäten soweit gestiegen, dass diese rechenintensive Methode gute Ergebnisse liefern konnte. Der Lernalgorithmus sucht das globale Minimum einer Fehlerfunktion auf den Trainingsdaten mittels Gradientenverfahren [9]. Begonnen wird dabei mit Startgewichten. Auch bei dieser Methode wird der Modellparameter $\eta \in \mathbb{R}_+$ für die Aktualisierungen benötigt und muss je nach Problem vorgegeben werden. Es wird sich zeigen, dass dieser Nachteil der Parameterwahl für neuronale Netze auch bei den Support-Vektor-Maschinen erhalten bleibt. Oft wird die Tatsache bemängelt, dass Lösungen des Gradientenverfahrens nur lokale Minima liefern und man das Training mehrmals mit unterschiedlichen Startparametern durchführen muss. Dieses Problem können Support-Vektor-Maschinen gezielt umgehen, mehr dazu folgt in Abschnitt 3.

Im Allgemeinen gelten neuronale Netze als black boxes, das bedeutet, im Gegensatz zu Entscheidungsbäumen erhält man keine Informationen darüber, wie die Eingaben auszusehen haben, um einen gewünschten Ausgabewert zu generieren.

2.5 Klassifikation mit Support-Vektor-Maschinen

An dieser Stelle soll kurz der Abschnitt 3 motiviert werden. Support-Vektor-Maschinen sind Lernalgorithmen für große Datenmengen, mit denen sich komplizierte Zusammenhän-

ge trainieren lassen. Sie wählen eine Zielfunktion aus linearen Abbildungen, da diese sehr gut handhabbar sind. Um Nichtlinearität zu ermöglichen, wird die Technik der Kerne benutzt. Die Wahl einer Zielfunktion erfolgt mit Hilfe der nichtlinearen, konvexen Optimierung. Die Qualität dieser Funktion sichert die sogenannte Generalisierungstheorie. Sie untersucht, wann eine Funktion in der Lage ist, gute Ergebnisse zu erzielen, wenn keine Beispiele, sondern nur noch unbekannte Daten in den Algorithmus eingehen. Support-Vektoren, die dem Verfahren den Namen geben, sind einige wenige der Beispieldaten, die während des Lernvorgangs ausgesucht werden und deren Eigenschaften bei allen späteren Klassifikationen neuer Punkte genutzt werden. Die bisher kurz erwähnten Komponenten werden in diesem Bericht detailliert vorgestellt, um das Verfahren der Support-Vektor-Maschinen zu verdeutlichen.

Auch wenn man auf den ersten Blick viele Gemeinsamkeiten finden kann, unterscheiden sich Support-Vektor-Algorithmen in vielen Details von neuronalen Netzen. Der erste Unterschied betrifft die Struktur. Wie im letzten Abschnitt zu sehen war, können Netze verschiedene Anzahlen von Schichten aufweisen. Für alle Verbindungen, die sich dann aus der Architektur ergeben, müssen Gewichte erlernt werden. Die Struktur von Support-Vektor-Verfahren ist fest und nicht durch die Biologie geprägt. Man wird im Abschnitt 3 sehen können, dass Support-Vektor-Maschinen keine Probleme mit lokalen Minima haben. Das ist ein klarer Vorteil für den Anwender.

Support-Vektor-Algorithmen zur Klassifikation finden verstärkt Zuspruch. Sie können für sehr viele Probleme und Forschungsgebiete angewendet werden. Dazu gehören beispielsweise:

(1) Klassifikation von Textdokumenten [15],

(2) Bild- und Schrifterkennung [28],

(3) Bioinformatik [22].

Dabei ist jedoch noch unklar, wie die Nutzung von Support-Vektor-Maschinen für binäre Modelle, wie sie beispielsweise im nächsten Abschnitt beschrieben werden, effizient auf multiple Klassifikationsaufgaben ausgeweitet werden kann. Sowohl die theoretische Formulierung dieses Problems als auch die daraus resultierenden Optimierungsprobleme sind an vielen Stellen besprochen worden, siehe zum Beispiel [13]. Allerdings konnte noch nicht geklärt werden, welcher Ansatz zur Umsetzung dieses Problems der günstigste ist. Aus diesem Grund werden oft die sehr erfolgreichen neuronalen Netze weiterbenutzt, da man den Nutzen von Support-Vektor-Maschinen für mehrklassige Fragestellungen nicht immer abschätzen kann.

3 Support-Vektor-Maschinen

Als Support-Vektor-Maschinen bezeichnet man Algorithmen, welche in kerninduzierten, hochdimensionalen Merkmalsräumen Parameter affin-linearer Funktionen mit Hilfe der

nichtlinearen Optimierung erlernen. Zunächst werden die nötigen Werkzeuge in den beiden folgenden Abschnitten bereitgestellt, bevor im Anschluss die zu lösenden Aufgaben diskutiert werden. Die Generalisierungstheorie wird in Abschnitt 3.4 Erkenntnisse über die Güte erlernbarer Funktionen liefern. Alle vier Abschnitte sind stark miteinander verknüpft und bilden die Grundlage für weitere Betrachtungen, insbesondere für die Beschreibung der implementierten Algorithmen.

In vielen Arbeiten wird das Lernverfahren der Support-Vektor-Maschinen losgelöst von den Theorien des Merkmalsraumes und der Kerne vorgestellt ([5], Kapitel 6). In diesem Bericht werden der Merkmalsraum und die Kerne als Werkzeuge interpretiert und rechtzeitig in die Lerntheorie eingebettet. Dieses Vorgehen kann zunächst zu Verwirrungen führen, ermöglicht es jedoch zu erkennen, an welchen Stellen Kerne benutzt werden können. Es sollte noch erwähnt werden, dass die Theorie der Kerne nicht einfach ein Modell ist, welches die Support-Vektor-Maschinen verfeinert. Sie ist eine ihrer Grundlagen und unterscheidet sie von linearen Lernverfahren.

3.1 Nichtlineare Optimierung

Wie noch zu sehen sein wird, führt das Lernverfahren einer Support-Vektor-Maschine auf das Problem einer konvexen Optimierungsaufgabe. Deshalb soll schon an dieser Stelle geklärt werden, wie man solche Aufgaben lösen kann. Dazu wird in diesem Abschnitt untersucht, wie eine bestimmte Klasse von Funktionen unter linearen Nebenbedingungen zu minimieren ist. Die Optimierungstheorie bietet gute Verfahren, um solche Probleme zu lösen. Die Minimierung einer beliebigen Funktion kann schwierig sein und auf viele lokale Minima führen. Schränkt man die Optimierung aber wie hier auf konvexe Funktionen ein, erhält man nur globale Lösungen. In diesem Abschnitt wird auf die Theorie von Lagrange eingegangen und gezeigt, in welchem günstigen Verhältnis primale und duale Optimierungsprobleme im konvexen Fall stehen.

Gegeben seien stetig differenzierbare, konvexe Funktionen

$$f : \mathbb{R}^n \to \mathbb{R}, \quad g_i : \mathbb{R}^n \to \mathbb{R} \ \ (i = 1, .., m_1)$$

sowie affin-lineare Funktionen

$$h_j : \mathbb{R}^n \to \mathbb{R} \ \ (j = 1, ..., m_2) \ .$$

Dann bezeichnet man die Aufgabe

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) \tag{3}$$

unter den Nebenbedingungen

$$g_i(\boldsymbol{x}) \leq 0 \ \ (i = 1, ..., m_1) \ ,$$
$$h_j(\boldsymbol{x}) = 0 \ \ (j = 1, ..., m_2)$$

als primale konvexe Optimierungsaufgabe.

Der sogenannte zulässige Bereich Z einer solchen Aufgabe ist definiert als

$$Z := \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}\} \; . \tag{4}$$

Globale Lösung einer Aufgabe der Form (3) ist ein Punkt $\boldsymbol{x}^*$ aus dem zulässigen Bereich Z, sodass es keinen anderen Punkt $\boldsymbol{x} \in Z$ gibt mit $f(\boldsymbol{x}) < f(\boldsymbol{x}^*)$.

Es gilt, dass jedes lokale Minimum von Aufgaben der Form (3) auch stets ein globales Minimum ist. Dazu sei angenommen, $\boldsymbol{x}^* \in Z$ sei ein lokales, aber kein globales Minimum von (3). Dann existiert ein $\tilde{\boldsymbol{x}} \in Z$ mit $f(\tilde{\boldsymbol{x}}) < f(\boldsymbol{x}^*)$. Deshalb und wegen der Konvexität von f folgt für alle Punkte $\boldsymbol{x}^\lambda = \lambda \boldsymbol{x}^* + (1-\lambda)\tilde{\boldsymbol{x}}$ mit $\lambda \in (0,1)$ die strikte Ungleichung

$$\begin{aligned} f(\boldsymbol{x}^\lambda) &\leq \lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\tilde{\boldsymbol{x}}) \\ &< \lambda f(\boldsymbol{x}^*) + (1-\lambda) f(\boldsymbol{x}^*) \\ &= f(\boldsymbol{x}^*) \; . \end{aligned}$$

Für $\lambda \to 1$ wäre die Eigenschaft von $\boldsymbol{x}^*$, lokales Minimum zu sein, verletzt.

Man nennt die Nebenbedingung g_i aktiv im Punkt $\boldsymbol{x}$, falls $g_i(\boldsymbol{x}) = 0$ gilt. Andernfalls nennt man sie inaktiv. Die Menge der aktiven Restriktionen eines Punktes $\boldsymbol{x}$ ist eine Teilmenge von $\{1, ..., m_1\}$ und sei mit A bezeichnet.

Der Tangentialkegel $\mathcal{T}_Z(\boldsymbol{x})$ von Z in $\boldsymbol{x}$ hat die Form

$$\mathcal{T}_Z(\boldsymbol{x}) := \{\boldsymbol{d} \in \mathbb{R}^n \mid \exists \{\boldsymbol{x}^k\} \subseteq Z, \{t_k\} \subseteq \mathbb{R} : t_k \downarrow 0, \boldsymbol{x}^k \to \boldsymbol{x}, (\boldsymbol{x}^k - \boldsymbol{x})/t_k \to \boldsymbol{d}\} \; \forall \boldsymbol{x} \in Z \; .$$

Der linearisierte Tangentialkegel $\mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x})$ von Z in $\boldsymbol{x}$ hat die Form

$$\mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) := \{\boldsymbol{d} \in \mathbb{R}^n \mid \nabla g_i(\boldsymbol{x})^T \boldsymbol{d} \leq 0 \; \forall i \in A \; , \nabla h_j(\boldsymbol{x})^T \boldsymbol{d} = 0 \; \forall j = 1, ..., m_2\} \; \forall \boldsymbol{x} \in Z \; .$$

Mit ∇ sei hier und auch im Folgenden immer der Gradient einer differenzierbaren Funktion bezeichnet.

Für differenzierbare Funktionen $g : \mathbb{R}^n \to \mathbb{R}^{m_1}$ und $h : \mathbb{R}^n \to \mathbb{R}^{m_2}$ gilt

$$\mathcal{T}_Z(\boldsymbol{x}) \subseteq \mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) \; \forall \boldsymbol{x} \in Z \; . \tag{5}$$

Zum Nachweis siehe [8].

Für einen zulässigen Punkt $\boldsymbol{x} \in Z$ eines Optimierungsproblems der Form (3) gelte

$$\mathcal{T}_Z(\boldsymbol{x}) = \mathcal{T}_{Z_{\text{lin}}}(\boldsymbol{x}) \; . \tag{6}$$

Dann sagt man, $\boldsymbol{x}$ erfüllt die Constraint-Qualification.

Diese Bedingung ist nicht leicht nachprüfbar. Deswegen werden häufig sogenannte Regularitätsbedingungen angegeben, unter denen die Constraint-Qualification für alle zulässigen Punkte erfüllt ist. Von den vielen vorhandenen Ansätzen seien zwei aus [8] zitiert.

Betrachte das Optimierungsproblem (3). Dann ist jede der folgenden Bedingungen hinreichend dafür, dass (6) für alle $\boldsymbol{x} \in Z$ gilt:

(1) Es gilt die Slater-Regularitätsbedingung, die definiert ist als

$$Z^0 := \{\boldsymbol{x} \in Z \mid g_i(\boldsymbol{x}) < 0,\ i = 1, ..., m_1\} \neq \emptyset\ . \tag{7}$$

(2) Neben den h_j $(j = 1, ..., m_2)$ sind auch alle Funktionen g_i $(i = 1, ..., m_1)$ affinlinear.

$\boldsymbol{x}^*$ sei Minimalstelle von (3) und erfülle die Constraint-Qualification. Dann existieren $\boldsymbol{\alpha}^* \in \mathbb{R}^{m_1}$ und $\boldsymbol{\beta}^* \in \mathbb{R}^{m_2}$, sodass die sogenannten Karush-Kuhn-Tucker-Bedingungen (KKT)

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m_1} \alpha_i^* \nabla g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j^* \nabla h_j(\boldsymbol{x}^*) = 0\ , \tag{8}$$

$$h_j(\boldsymbol{x}^*) = 0\ \ (j = 1, ..., m_2)\ , \tag{9}$$

$$g_i(\boldsymbol{x}^*) \leq 0\ \ (i = 1, ..., m_1)\ , \tag{10}$$

$$\alpha_i^* g_i(\boldsymbol{x}^*) = 0\ \ (i = 1, ..., m_1)\ , \text{ und} \tag{11}$$

$$\alpha_i^* \geq 0\ \ (i = 1, ..., m_1) \tag{12}$$

gelten. Die Forderungen (9) und (10) ergeben sich sofort aus der Zulässigkeit von $\boldsymbol{x}^*$. Die übrigen Bedingungen (8), (11) und (12) folgen aus der Anwendung des bekannten Farkas-Lemmas, siehe dazu [8].

Nun soll noch gezeigt werden, dass die KKT-Bedingungen sogar ohne weitere Bedingungen hinreichend für die Existenz einer Lösung sind.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Tripel, welches die KKT-Bedingungen (8) - (12) erfüllt (KKT-Punkt). Dann muss gelten

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)\ \forall \boldsymbol{x} \in Z.$$

Wegen (9) und (10) gilt $\boldsymbol{x}^* \in Z$.
Für den Beweis werden neben den KKT-Bedingungen noch die Voraussetzungen

a) f ist konvex ,

b) alle g_i $(i = 1, ..., m_1)$ sind konvex , und

c) alle h_j $(j = 1, ..., m_2)$ sind affin-linear
verwendet. Diese führen auf

$$\begin{aligned} f(\boldsymbol{x}) &\overset{a)}{\geq} f(\boldsymbol{x}^*) + \langle \nabla f(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \\ &\overset{(8)}{=} f(\boldsymbol{x}^*) - \sum_{i=1}^{m_1} \alpha_i^* \langle \nabla g_i(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 - \sum_{j=1}^{m_2} \beta_j^* \langle \nabla h_j(\boldsymbol{x}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \\ &\overset{b),c),(4)}{\geq} f(\boldsymbol{x}^*) - \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}) + \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) \\ &\overset{(10),(11),(12)}{\geq} f(\boldsymbol{x}^*) \, . \end{aligned} \tag{13}$$

Damit ist $\boldsymbol{x}^*$ eine Minimalstelle der Optimierungsaufgabe (3).

Wir werden später zeigen, dass Optimierungsaufgaben, wie sie beim Verfahren Support-Vektor-Maschinen entstehen, nur affin-lineare Nebenbedingungen haben. Für diese Form konvexer Optimierungsaufgaben kann man die bisherigen Aussagen zusammenfassen, denn die Voraussetzungen implizieren die Constraint-Qualification. Es folgt: Für Aufgaben der Form (3) mit ausschließlich affin-linearen Nebenbedingungen ist die Tatsache, dass $\boldsymbol{x}^*$ globale Lösung ist, äquivalent dazu, dass es Multiplikatoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ gibt, die zusammen mit $\boldsymbol{x}^*$ einen KKT-Punkt bilden.

3.1.1 Theorie von Lagrange

Die Abbildung $L : \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \to \mathbb{R}$, definiert durch

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &:= f(\boldsymbol{x}) + \sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}) \\ &= f(\boldsymbol{x}) + \langle \boldsymbol{\alpha}, \boldsymbol{g}(\boldsymbol{x}) \rangle_2 + \langle \boldsymbol{\beta}, \boldsymbol{h}(\boldsymbol{x}) \rangle_2 \, , \end{aligned} \tag{14}$$

heißt Lagrange-Funktion, wobei $\boldsymbol{\alpha}$ und $\boldsymbol{\beta}$ Vektoren von Lagrange-Multiplikatoren genannt werden.

Ein Tripel $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ mit $\boldsymbol{\alpha}^* \geq 0$ wird Sattelpunkt von L genannt, falls für alle $(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ mit $\boldsymbol{\alpha} \geq 0$ die Ungleichungen

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \leq L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \leq L(\boldsymbol{x}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \tag{15}$$

erfüllt sind.

Wir betrachten erneut die primale Aufgabe (3) und die zugehörige Lagrange-Funktion. Dann gilt, dass das Tripel $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \in \mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ die KKT-Bedingungen erfüllt, genau dann, wenn $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt von L ist. Diese Aussage ist als Sattelpunkttheorem bekannt. Wir beweisen die Aussage kurz.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein KKT-Punkt. Dann ist $\boldsymbol{x}^*$ ein stationärer Punkt von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$, denn wegen (8) gilt

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \mathbf{0} \,. \tag{16}$$

Lineare Funktionen sind konvex. Somit ist die Funktion $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ als Linearkombination konvexer und linearer Funktionen ebenfalls konvex. In $\boldsymbol{x}^*$ liegt ein globales Minimum von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ vor, denn für alle $\boldsymbol{x} \in \mathbb{R}^n$ gilt:

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(16)}{=} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) + \langle \nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*), (\boldsymbol{x} - \boldsymbol{x}^*) \rangle_2 \overset{(13)}{\leq} L(\boldsymbol{x}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,.$$

Der erste Teil der Sattelpunktungleichung wird direkt über die KKT-Eigenschaften gezeigt:

$$L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(9),(11)}{=} f(\boldsymbol{x}^*) \overset{(9),(10),(12)}{\geq} f(\boldsymbol{x}^*) + \sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}^*) = L(\boldsymbol{x}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \,.$$

Damit ist $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt der Lagrange-Funktion.

Sei $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt von L, dann ist $\boldsymbol{x}^*$ globales Minimum von $L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ und somit gelten (8) sowie die Ungleichung

$$\sum_{i=1}^{m_1} \alpha_i g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j h_j(\boldsymbol{x}^*) \leq \sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) + \sum_{j=1}^{m_2} \beta_j^* h_j(\boldsymbol{x}^*) \,.$$

Wegen $\boldsymbol{\alpha} \geq \mathbf{0}$ und $\boldsymbol{\beta} \in \mathbb{R}^{m_2}$ könnte der linke Teil der Ungleichung im Falle $\boldsymbol{g}(\boldsymbol{x}^*) > \mathbf{0}$ oder $\boldsymbol{h}(\boldsymbol{x}^*) \neq \mathbf{0}$ beliebig wachsen. Das impliziert $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$ sowie $\boldsymbol{h}(\boldsymbol{x}^*) = \mathbf{0}$ und sichert die Bedingungen (9) und (10) an einen KKT-Punkt.

Für die zulässigen Werte $\boldsymbol{\alpha} = \mathbf{0}$ und $\boldsymbol{\beta} = \mathbf{0}$ folgt schließlich aus obiger Ungleichung $\sum_{i=1}^{m_1} \alpha_i^* g_i(\boldsymbol{x}^*) \geq 0$ und wegen $\boldsymbol{\alpha}^* \geq \mathbf{0}$ sowie $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$ sind alle Komponenten der Summe genau Null.

Alle bisherigen Ergebnisse des Abschnittes 3.1 werden an dieser Stelle kurz zusammengefasst, bevor bestimmte Dualitätsaussagen beleuchtet werden.

Konvexe Optimierungsaufgaben der Form (3) haben die folgenden Eigenschaften:

(1) Ist $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ein Sattelpunkt der Lagrange-Funktion, dann ist $\boldsymbol{x}^*$ globale Minimalstelle der Optimierungsaufgabe.

(2) Ist $\boldsymbol{x}^*$ globale Minimalstelle und erfüllt es die Slater-Regularitätsbedingung, dann existieren Vektoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ von Lagrange-Multiplikatoren, sodass $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ Sattelpunkt der Lagrange-Funktion ist.

(3) Für Aufgaben mit ausschließlich affin-linearen Nebenbedingungen ist $\boldsymbol{x}^*$ genau dann globale Minimalstelle, wenn Vektoren von Lagrange-Multiplikatoren $\boldsymbol{\alpha}^*$ und $\boldsymbol{\beta}^*$ existieren, sodass $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ Sattelpunkt von L ist.

3.1.2 Dualität

Die Dualitätstheorie transformiert die bisher betrachteten primalen Optimierungsaufgaben und gibt Auskunft darüber, in welchem Verhältnis Lösungen dieser transformierten Aufgaben zu den ursprünglichen Lösungen stehen.

Die Funktion θ sei definiert als

$$\theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) := \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \; . \tag{17}$$

Dann hat die zum Ausgangsproblem duale Aufgabe in Lagrange-Form die Darstellung

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^{m_1},\, \boldsymbol{\beta} \in \mathbb{R}^{m_2}} \theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{18}$$

unter den Nebenbedingungen

$$\boldsymbol{\alpha} \geq \mathbf{0} \; .$$

Im Folgenden soll der Zusammenhang zwischen primalen und dualen Aufgaben gezeigt werden. Dieser führt in der konvexen Optimierung dazu, dass ohne Einschränkung die duale anstelle der primalen Aufgabe gelöst werden kann. Für die Anwendung von SV-Maschinen wird diese Tatsache von besonderer Wichtigkeit sein. Zunächst gilt der sogenannte schwache Dualitätssatz: Sei $\boldsymbol{x}^0$ ein zulässiger Punkt der primalen Aufgabe (3) und $(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0)$ ein zulässiger Punkt der dualen Aufgabe (18), dann ist

$$\theta(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \leq f(\boldsymbol{x}^0) \; ,$$

denn nach Definition gilt

$$\begin{aligned} \theta(\boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) &= \inf_{\boldsymbol{u} \in \mathbb{R}^n} L(\boldsymbol{u}, \boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \\ &\leq L(\boldsymbol{x}^0, \boldsymbol{\alpha}^0, \boldsymbol{\beta}^0) \\ &= f(\boldsymbol{x}^0) + \langle \boldsymbol{\alpha}^0, \boldsymbol{g}(\boldsymbol{x}^0) \rangle_2 + \langle \boldsymbol{\beta}^0, \boldsymbol{h}(\boldsymbol{x}^0) \rangle_2 \\ &\leq f(\boldsymbol{x}^0) \; . \end{aligned}$$

Letztere Ungleichung gilt, da für alle zulässigen Punkte $\boldsymbol{g}(\boldsymbol{x}^0) \leq \mathbf{0}$, $\boldsymbol{h}(\boldsymbol{x}^0) = \mathbf{0}$ und $\boldsymbol{\alpha}^0 \geq \mathbf{0}$ erfüllt sein müssen.

Das Maximum der dualen Aufgabe liegt also nie über dem Minimum der primalen Aufgabe. Dadurch ist aber noch nicht gesichert, dass die Lösungen nah beieinander liegen. Die sogenannte Dualitätslücke, der Abstand der Zielfunktionswerte, kann unter Umständen groß sein. Die Übereinstimmung der optimalen Funktionswerte sichert erst der starke Dualitätssatz. Für konvexe Aufgaben, wie sie hier betrachtet werden, gilt folgende Aussage: Gegeben sei eine konvexe Optimierungsaufgabe, für die ein Karush-Kuhn-Tucker-Punkt $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ existiert, dann ist

$$f(\boldsymbol{x}^*) = \theta(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,, \tag{19}$$

das heißt, der Wert der Dualitätslücke beträgt Null. Diese Aussage läßt sich schnell zeigen. $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ ist offensichtlich ein Sattelpunkt der Lagrange-Funktion und somit ist $\boldsymbol{x}^*$ Minimalstelle der primalen Zielfunktion f. Daraus erhält man sofort:

$$f(\boldsymbol{x}^*) \overset{(9),(11)}{=} L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \overset{(15)}{=} \inf L(\cdot, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = \theta(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) \,.$$

Die Dualitätstheorie garantiert, dass es für die Lösung konvexer Optimierungsaufgaben hinreichend ist, die zugehörigen dualen Aufgaben zu betrachten. Diese weisen sehr einfache Nebenbedingungen auf, was sich positiv auf die Handhabbarkeit auswirkt. Zusätzlich ermöglicht die Dualitätstheorie den Einsatz sogenannter Kernfunktionen, denn erst die Formulierung der dualen Aufgabe einer Support-Vektor-Maschine lässt erkennen, welchen einfachen Weg man einschlagen kann. Weitere Informationen dazu folgen in den nächsten beiden Abschnitten. Wie sich noch zeigen wird, reicht es für die Anwendung von Support-Vektor-Maschinen sogar aus, quadratische Optimierungsprobleme zu betrachten. Das sind Aufgaben mit quadratischer Zielfunktion und linearen Nebenbedingungen. Selbstverständlich gilt die allgemeinere Theorie der konvexen Aufgaben, wie sie in diesem Abschnitt untersucht wurde, denn quadratische Aufgaben gehören in die Klasse konvexer Optimierungsprobleme.

3.2 Merkmalsräume und Kerne

Affin-lineare Funktionen zur Erklärung von Eingabe-Ausgabe-Zusammenhängen sind oft nicht geeignet, um komplizierte Abhängigkeiten zu erfassen. Support-Vektor-Maschinen nutzen lineare Verfahren zur Klassifikation. Da sie aber auch nichtlineare Zusammenhänge erlernen sollen, wenden sie sich zusätzlich einer anderen Methode zu.

Die Idee der Merkmalsräume ist, die Eingabedaten in einen hochdimensionalen Raum abzubilden und dort einfache lineare Funktionen zu trainieren. Man hofft, durch eine nichtlineare Transformation der Daten eine lineare Trennbarkeit zu erzeugen. Support-Vektor-Maschinen ersparen dem Anwender die mühsame Transformation der Daten, sie verwenden Kerne (kernels), um diese Arbeit zu umgehen. Diese Technik der impliziten Abbildung

in einen anderen Raum und die Nutzung von Kernen ist eine der wichtigen Eigenschaften von Support-Vektor-Maschinen, aber auch eine ihrer größten Herausforderungen, denn das Auffinden geeigneter Kerne bleibt oft Sache des Benutzers. Auf diese Aufgabe wird später noch eingegangen.

Zunächst soll in diesem Abschnitt gezeigt werden, welche Eigenschaften Kerne haben und warum man sie verwenden kann. Die wichtigste Grundlage dafür ist die Dualitätstheorie des letzten Abschnittes.

3.2.1 Grundlagen

Zu Beginn geht man davon aus, dass die Daten des Raumes X transformiert werden sollen, um lineare Trennbarkeit zu ermöglichen. Dazu wird eine Abbildung

$$\boldsymbol{\phi} : X \to F \tag{20}$$

definiert. $\boldsymbol{\phi}$ ist die sogenannte Merkmalsabbildung, wobei F ein Hilbertraum ist. Oft wählt man den Hilbertraum $\mathbb{R}^N,\ N \in \mathbb{N}$. Diese Transformation liefert einen Merkmalsvektor, also

$$\boldsymbol{x} = (x_1, .., x_n) \mapsto \boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), ..., \phi_N(\boldsymbol{x}))\ .$$

F kann auch unendlichdimensional sein. In diesem Fall wählt man den Hilbertraum l_2, den Raum aller Folgen $\boldsymbol{\psi} = \{\psi_1, ..., \psi_j, ...\}$, für die gilt

$$\|\boldsymbol{\psi}\|_{l_2}^2 = \langle \boldsymbol{\psi}, \boldsymbol{\psi} \rangle_{l_2} := \sum_{j=1}^{\infty} \psi_j^2 < \infty\ .$$

Der Raum

$$F \supseteq \{\boldsymbol{\phi}(\boldsymbol{x}) \mid \boldsymbol{x} \in X\}$$

wird Merkmalsraum genannt.

Für $N \in \mathbb{N} \cup \{\infty\}$ werden die $\phi_j(\boldsymbol{x}),\ j = 1, ..., N$, als Merkmale bezeichnet. Oft spricht man auch von Features und dem Featureraum, wobei diese Bezeichnung kritisch ist, da auch Variablenselektion oft als *feature selection* bezeichnet wird. Damit sind dann wiederum die Originalvariablen gemeint.

Das Skalarprodukt zwischen Merkmalsvektoren in F sei definiert als

$$\langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F := \sum_{j=1}^{N} \phi_j(\boldsymbol{x}) \phi_j(\boldsymbol{z})\ . \tag{21}$$

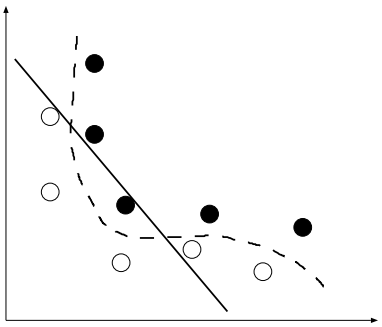


Abbildung 5: Nichtlineare Abhängigkeiten im Eingaberaum.

Abbildung 5 zeigt beispielhaft Daten, die im Eingaberaum nicht durch eine lineare Funktion separiert werden können.

Eine Transformation der Daten kann dazu führen, dass die Merkmale im Raum F linear trennbar sind, siehe dazu Abbildung 6. Man beachte, dass die Dimension der Räume X und F hier nur aus Darstellungsgründen immer zwei ist.

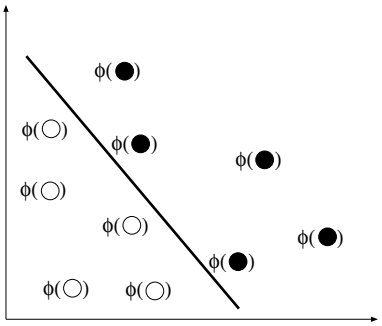


Abbildung 6: Lineare Trennbarkeit im Merkmalsraum.

Die wichtige Frage ist: Wie soll die Funktion ϕ gewählt werden und wie sieht der Wert für N aus? Als erste Idee kann man eine Dimensionsverkleinerung durchführen, also ein $N < n$ wählen. Dabei sollte die Anzahl der Merkmale N so klein wie möglich sein, wobei die wichtigsten Informationen, welche die Daten enthalten, nicht verloren gehen dürfen. Dieser Ansatz erscheint zu Beginn sinnvoll, denn:

(1) Die Performance von Algorithmen sinkt mit steigender Merkmalsanzahl, weil mehr Daten bearbeitet werden müssen.

(2) Bei wachsender Dimension des Raumes leidet möglicherweise die Generalisierungsfähigkeit. Alle Informationen dazu folgen im Abschnitt 3.4.

Problematisch bei diesem Ansatz ist die Tatsache, dass mit wachsendem N, also steigender Anzahl von Merkmalen, eine lineare Schätzfunktion auch immer bessere Ergebnisse erzielen wird. Diese Aussage geht zurück auf ein Theorem in [4]. Hier stehen sich also zwei Zielsetzungen konkurrierend gegenüber. Support-Vektor-Maschinen umgehen dieses

Problem, indem sie die Daten zwar in einen hochdimensionalen Raum abbilden, dies jedoch nur implizit tun, sodass alle Berechnungen im ursprünglichen Raum stattfinden und unabhängig von der Dimension des Merkmalsraumes sind. Als Mittel zur Realisierung dieses Ansatzes werden Kernfunktionen benutzt.

Jetzt soll untersucht werden, wie sich das Lernproblem verändert, falls man die Daten wie oben beschrieben transformiert. Zunächst kann man festhalten, dass es sich nicht mehr um ein lineares Problem handelt, da die Merkmalsabbildung $\boldsymbol{\phi}$ nicht linear sein muss. Dennoch stellt die gesuchte Funktion f auch hier eine Linearkombination dar - eine Kombination der Komponenten des Merkmalsvektors $\boldsymbol{\phi}$. Die zu lernende Funktion in X hat dann nach Abschnitt 2.2 folgende Gestalt:

$$f(\boldsymbol{x}) = \sum_{j=1}^{N} w_j \phi_j(\boldsymbol{x}) + b = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b \tag{22}$$

für zunächst unbekannte $\boldsymbol{w} \in F$ und $b \in \mathbb{R}$. Eine für einen gegebenen Datensatz optimale Funktion f, charakterisiert durch $(\boldsymbol{w}, b)$, kann, wie im Abschnitt 3.3 aufbauend auf Abschnitt 3.1 noch gezeigt wird (siehe Gleichung (42)), durch Substitution von

$$\boldsymbol{w} = \sum_{i=1}^{l} y_i \alpha_i \boldsymbol{\phi}(\boldsymbol{x}^i)$$

dual dargestellt werden als Auswertungsfunktion

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b \, . \tag{23}$$

Dabei müssen im Falle dieser Form der Zielfunktion gewisse α_i anstelle der w_j erlernt werden. Man sieht, dass für beliebige $\boldsymbol{x} \in X$ alle Trainingspaare an der Bestimmung von $f(\boldsymbol{x})$ beteiligt sind. Dabei ist wichtig, dass die Dimension des Raumes F nun für die Anzahl der Summenglieder keine Rolle mehr spielt. Jedoch sei festgehalten, dass F noch über das Skalarprodukt in die Berechnungen eingeht.

Kann man $\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F$ in (23) direkt über eine Funktion im Eingaberaum X berechnen, spart man sich die Abbildung in den Merkmalsraum und trainiert die Zielfunktion f direkt im Eingaberaum.

Mit der Wahl einer solchen Funktion umgeht man daher folgende Schritte:

(1) Suche einer geeigneten Abbildung $\boldsymbol{\phi}$,

(2) explizite Berechnung von Skalarprodukten in einem hochdimensionalen Merkmalsraum.

Man muss aber klären, welche Funktionen überhaupt dafür in Frage kommen und welche davon besonders geeignet sind. Gesucht sind also Funktionen, die Skalarprodukte in Merkmalsräumen implizit berechnen können.

Eine Funktion $K : X \times X \to \mathbb{R}$ wird Kern genannt, falls eine Abbildung $\boldsymbol{\phi}$ der Form (20) existiert mit

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \ \ \forall \boldsymbol{x}, \boldsymbol{z} \in X. \tag{24}$$

Gegeben seien ein Kern K und l Vektoren $\boldsymbol{x}^1, ..., \boldsymbol{x}^l \in X$. Dann nennt man die $l \times l$-Matrix $\boldsymbol{K}$ mit den Elementen $k_{ij} = K(\boldsymbol{x}^i, \boldsymbol{x}^j),\ 1 \leq i, j \leq l$, Grammatrix oder auch Kernmatrix.

Kerne sind geeignete Abbildungen, um Berechnungen im Merkmalsraum zu umgehen. Kombiniert man (23) und (24), erhält die Zielfunktion f die Form

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i K(\boldsymbol{x}^i, \boldsymbol{x}) + b \ . \tag{25}$$

f wird in diesem Fall Kernfunktion genannt um auszudrücken, dass nicht mit Merkmalen aus F gearbeitet wird.

Notwendige Eigenschaften, die jeder Kern erfüllen muss, ergeben sich aus den allgemeinen Eigenschaften von Skalarprodukten [2]:

(1)

$$\begin{aligned} K(\boldsymbol{x}, \boldsymbol{z}) &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \\ &\overset{\text{Symmetrie}}{=} \langle \boldsymbol{\phi}(\boldsymbol{z}), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F \\ &= K(\boldsymbol{z}, \boldsymbol{x}) \ , \end{aligned}$$

(2)

$$\begin{aligned} (K(\boldsymbol{x}, \boldsymbol{z}))^2 &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F^2 \\ &\overset{\text{Cauchy-Schwarz}}{\leq} \|\boldsymbol{\phi}(\boldsymbol{x})\|_F^2 \|\boldsymbol{\phi}(\boldsymbol{z})\|_F^2 \\ &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F \langle \boldsymbol{\phi}(\boldsymbol{z}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F \\ &= K(\boldsymbol{x}, \boldsymbol{x}) K(\boldsymbol{z}, \boldsymbol{z}) \ . \end{aligned}$$

Diese Eigenschaften sind aber nicht hinreichend. Es bleibt noch die Frage, für welche Funktionen K eine Abbildung $\boldsymbol{\phi}$ in einen Merkmalsraum F existiert, sodass K ein Kern ist. Folgende bekannte Aussagen sollen kurz in Erinnerung gerufen werden:

- Symmetrische Matrizen haben nur reelle Eigenwerte [2].
- Eine symmetrische Matrix ist genau dann positiv semidefinit, wenn alle ihre Eigenwerte nichtnegativ sind [2].

3.2.2 Theorem von Mercer

Man betrachte wieder den Fall $N \in \mathbb{N} \cup \{\infty\}$ und einen Merkmalsvektor $\boldsymbol{\phi}(\boldsymbol{x})$. Das Skalarprodukt in F sei wie bisher von der Form (21).

Das Theorem von Mercer beschäftigt sich mit der Frage, wann eine Funktion K die Darstellung

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{z}) \tag{26}$$

haben kann, also ein Kern ist.

Sei X eine kompakte Teilmenge des $\mathbb{R}^n$ und $K : X \times X \to \mathbb{R}$ eine stetige, symmetrische Funktion, für die der Integraloperator $T_K : L_2(X) \to L_2(X)$,

$$(T_K f)(\cdot) = \int_X K(\cdot, \boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x},$$

nichtnegativ ist, also

$$\int_{X \times X} K(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{x}) f(\boldsymbol{z}) d\boldsymbol{x} d\boldsymbol{z} \geq 0 \tag{27}$$

für alle $f \in L_2(X)$ gilt. Für $j = 1, ..., N$ seien $\phi_j^M \in L_2(X)$ die normierten orthogonalen Eigenfunktionen von T_K mit zugehörigen positiven Eigenwerten $\lambda_j^M > 0$. Dann gilt für alle $\boldsymbol{x},\ \boldsymbol{z} \in X$, dass K die Darstellung

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{N} \lambda_j^M \phi_j^M(\boldsymbol{x}) \phi_j^M(\boldsymbol{z}) \tag{28}$$

besitzt. Für $N = \infty$ konvergiert die Reihe gleichmäßig [23].

Die gleichmäßige Konvergenz kann mit Hilfe des Satzes von Dini gezeigt werden. Dieser sowie ein ausführlicher Beweis des Theorems von Mercer sind in [16], § 8 zu finden.

Die speziellen Abbildungen K in (28) nennt man Mercer-Kerne. Die zugehörigen

$$\phi_j^M(\boldsymbol{x})\ (j = 1, ..., N)$$

werden als Mercer-Merkmale bezeichnet. Man kann zeigen, dass die Bedingung (27) äquivalent dazu ist, dass für jede beliebige endliche Teilmenge $\boldsymbol{x}^1, ..., \boldsymbol{x}^l$ aus X die zugehörige Grammatrix positiv semidefinit ist. Diese Aussage wird in [5] hergeleitet.

Fazit: Falls die Abbildung $\boldsymbol{\phi}$ die Form

$$\phi_j(\boldsymbol{x}) = \sqrt{\lambda_j^M}\phi_j^M(\boldsymbol{x}) \;\; (j = 1, ..., N) \tag{29}$$

hat, folgt aus der Darstellung (28) unmittelbar, dass Mercer-Kerne Kerne nach unserer Definition sind.

3.2.3 Kernreproduzierende Räume

F sei ein durch eine abzählbare Menge linear unabhängiger Merkmalsabbildungen ϕ_j $(j = 1, ..., N)$ gegebener Merkmalsraum. Die Eigenschaften der Abbildung $\boldsymbol{\phi}^M$ aus (29) müssen nicht gelten.

Es sei

$$K(\boldsymbol{x}, \boldsymbol{z}) := \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle_F = \sum_{j=1}^{N} \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{z}) \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \; . \tag{30}$$

Man betrachte eine Abbildung T mit $T(F) = \mathcal{H}$. Der Raum $\mathcal{H}$ über der Menge X entsteht durch die Vorschrift

$$T : \boldsymbol{\psi} \mapsto \sum_{j=1}^{N} \psi_j\phi_j(\cdot) \quad (\boldsymbol{\psi} \in F, \; N \in \mathbb{N} \cup \{\infty\}) \; .$$

$\mathcal{H}$ entspricht damit bis auf die additive Konstante b dem gewünschten Raum der Zielfunktionen.

Bei Verknüpfung der dualen Aufgabe mit der Theorie der Kerne - siehe Gleichung (25) - erhält man die zulässige Darstellung der realisierbaren Auswertungsfunktionen als

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i\alpha_i K(\boldsymbol{x}^i, \boldsymbol{x}) + b \quad (l \in \mathbb{N}, \; \alpha_i \in \mathbb{R}) \; ,$$

die zunächst völlig unabhängig vom Merkmalsraum ist. Es stellt sich die folgende Frage, ob die Menge

$$\mathcal{G} = \left\{ \sum_{i=1}^{l} y_i\alpha_i K(\boldsymbol{x}^i, \cdot) \mid l \in \mathbb{N}, \; \alpha_i \in \mathbb{R}, \; \boldsymbol{x}^i \in X, \; y_i \in \{-1, 1\} \right\}$$

mit dem Raum $\mathcal{H}$ über X übereinstimmt.

Für den Fall $N < \infty$ liefern die Darstellung von $\boldsymbol{w}$ auf Seite 121 sowie die vorausgesetzte Unabhängigkeit der Merkmale ϕ_j, dass alle Funktionen aus $\mathcal{H}$ realisierbar sind, das heißt: $\mathcal{G} = \mathcal{H}$. Für $N = \infty$ ist diese Gleichheit nicht gesichert. Die Menge der Abbildungen in

$\mathcal{H}$ könnte nicht alle denkbaren Auswertungsfunktionen enthalten, beispielsweise wenn sie Bilder von Punkten ohne endliche Norm in F sind. $\mathcal{H}$ könnte auch zu viele Funktionen enthalten. Diese Sorge ist unbegründet, da $\mathcal{G} \subseteq \mathcal{H}$ gilt und jede Zielfunktion aus $\mathcal{H}$ beliebig genau durch ein Element aus $\mathcal{G}$ approximiert werden kann. Zum Nachweis benötigt man noch ein Skalarprodukt in $\mathcal{H}$, welches nun definiert wird.

Für Funktionen $f(\boldsymbol{x}) = \sum_{j=1}^{N} \psi_j \phi_j(\boldsymbol{x})$ und $g(\boldsymbol{x}) = \sum_{j=1}^{N} \psi'_j \phi_j(\boldsymbol{x})$ aus $\mathcal{H}$ sei das Skalarprodukt in $\mathcal{H}$ definiert als

$$\langle f(\cdot), g(\cdot) \rangle_{\mathcal{H}} := \sum_{j=1}^{N} \psi_j \psi'_j \; . \tag{31}$$

Für $\overline{\mathcal{G}}$, den Abschluss von $\mathcal{G}$, gilt $\overline{\mathcal{G}} = \mathcal{H}$. Wir beweisen das im Folgenden mittels Mengeninklusion.

Es gilt $K(\cdot, \boldsymbol{z}) \in \mathcal{H}$, denn

$$K(\cdot, \boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\cdot)\phi_j(\boldsymbol{z}) = \sum_{j=1}^{N} \phi_j(\boldsymbol{z})\phi_j(\cdot) = T(\boldsymbol{\phi}(\boldsymbol{z})) \; .$$

Daraus kann man direkt folgern, dass alle Funktionen der Form

$$f(\cdot) = \sum_{i=1}^{l} y_i \alpha_i K(\cdot, \boldsymbol{x}^i) \quad (l \in \mathbb{N})$$

ebenfalls in $\mathcal{H}$ liegen. Da $\mathcal{H}$ abgeschlossen ist, folgt $\overline{\mathcal{G}} \subseteq \mathcal{H}$.

Für $f(\cdot) = \sum_{j=1}^{N} \psi_j \phi_j(\cdot)$ und $K(\cdot, \boldsymbol{z})$ aus $\mathcal{H}$ liefert die Anwendung des in $\mathcal{H}$ eingeführten Skalarproduktes die sogenannte Reproduktionseigenschaft

$$\langle f(\cdot), K(\cdot, \boldsymbol{z}) \rangle_{\mathcal{H}} = \sum_{j=1}^{N} \psi_j \phi_j(\boldsymbol{z}) = f(\boldsymbol{z}) \; . \tag{32}$$

Sei nun $f \in \mathcal{H}$. Da $\mathcal{H}$ ein Hilbertraum ist, kann man die Funktion f darstellen als $f = f_1 + f_2$ mit $f_1 \in \overline{\mathcal{G}}$ und $f_2 \perp \overline{\mathcal{G}}$. Die Reproduktionseigenschaft (32) sichert, dass für alle $\boldsymbol{z} \in X$ gilt:

$$f_2(\boldsymbol{z}) = \langle f_2(\cdot), K(\cdot, \boldsymbol{z}) \rangle_{\mathcal{H}} = 0 \; .$$

Daraus folgt, dass f_2 die Nullfunktion ist und $f = f_1$ gilt. Dies liefert sofort $f \in \overline{\mathcal{G}}$. Daraus folgt $\mathcal{H} \subseteq \overline{\mathcal{G}}$.

3.2.4 Bildung neuer Kerne

Bevor wir zeigen, wie man Kerne aus anderen Kernen, Funktionen und Konstanten erzeugen kann, erinnern wir an einige bekannte Definitionen.

- Gegeben seien zwei reelle Matrizen $\mathbf{A} = (a_{ij})_{1 \leq i,j \leq m}$ und $\mathbf{B} = (b_{ij})_{1 \leq i,j \leq m}$. Dann wird die $m^2 \times m^2$-Matrix

$$\mathbf{C} := \mathbf{A} \otimes \mathbf{B} = (a_{ij} \cdot \mathbf{B})_{1 \leq i,j \leq m}$$

  als Kronecker-Produkt von $\mathbf{A}$ und $\mathbf{B}$ bezeichnet.

- Gegeben seien zwei reelle Matrizen $\mathbf{A} = (a_{ij})_{1 \leq i,j \leq m}$ und $\mathbf{B} = (b_{ij})_{1 \leq i,j \leq m}$. Dann wird die $m \times m$-Matrix

$$\mathbf{C} := (a_{ij} \cdot b_{ij})_{1 \leq i,j \leq m}$$

  als Hadamard-Produkt von $\mathbf{A}$ und $\mathbf{B}$ bezeichnet.

- Eine Hauptuntermatrix einer quadratischen Matrix entsteht durch Streichen von Zeilen und Spalten mit demselben Index.

Seien K_1 und K_2 Mercer-Kerne, κ eine reelle Zahl, $f : X \to \mathbb{R}$ eine reelle Funktion und p ein Polynom mit positiven Koeffizienten. Dann sind

(1) $K(\boldsymbol{x}, \boldsymbol{z}) := K_1(\boldsymbol{x}, \boldsymbol{z}) + K_2(\boldsymbol{x}, \boldsymbol{z})$,

(2) $K(\boldsymbol{x}, \boldsymbol{z}) := \kappa \cdot K_1(\boldsymbol{x}, \boldsymbol{z})$,

(3) $K(\boldsymbol{x}, \boldsymbol{z}) := K_1(\boldsymbol{x}, \boldsymbol{z}) \cdot K_2(\boldsymbol{x}, \boldsymbol{z})$,

(4) $K(\boldsymbol{x}, \boldsymbol{z}) := f(\boldsymbol{x}) \cdot f(\boldsymbol{z})$,

(5) $K(\boldsymbol{x}, \boldsymbol{z}) := p(K_1(\boldsymbol{x}, \boldsymbol{z}))$, und

(6) $K(\boldsymbol{x}, \boldsymbol{z}) := e^{K(\boldsymbol{x}, \boldsymbol{z})}$

ebenfalls Mercer-Kerne, denn offensichtlich sind alle Funktionen $(1). - (6).$ symmetrisch. Für alle Beispiele bleibt zu zeigen, dass die Grammatrix $\boldsymbol{K}$ einer endlichen Anzahl beliebiger Punkte $\boldsymbol{x}^1, ..., \boldsymbol{x}^l$ $(\boldsymbol{x}^i \in X)$ positiv semidefinit ist, also

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} \geq 0$$

für alle $\boldsymbol{a} \in \mathbb{R}^l$ gilt.

(1) $\boldsymbol{a}^T(\boldsymbol{K_1} + \boldsymbol{K_2})\boldsymbol{a} = \boldsymbol{a}^T \boldsymbol{K_1} \boldsymbol{a} + \boldsymbol{a}^T \boldsymbol{K_2} \boldsymbol{a} \geq 0$.

(2) $\boldsymbol{a}^T(\boldsymbol{\kappa K_1})\boldsymbol{a} = \kappa \boldsymbol{a}^T(\boldsymbol{K_1})\boldsymbol{a} \geq 0$.

(3) Man betrachte das Kronecker-Produkt von $\boldsymbol{K_1}$ und $\boldsymbol{K_2}$, es sei mit $\boldsymbol{A}$ bezeichnet. Dann ist $\boldsymbol{A}$ positiv semidefinit. Die zur Funktion $K_1 \cdot K_2$ gehörende Matrix $\boldsymbol{K}$ ist das Hadamard-Produkt von $\boldsymbol{K_1}$ und $\boldsymbol{K_2}$. Offensichtlich ist $\boldsymbol{K} \in \mathbb{R}^{l \times l}$ eine Hauptuntermatrix von $\boldsymbol{A} \in \mathbb{R}^{l^2 \times l^2}$ und somit existiert zu jedem $\boldsymbol{a} \in \mathbb{R}^l$ ein $\boldsymbol{\beta} \in \mathbb{R}^{l^2}$, so dass

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} = \boldsymbol{\beta}^T \boldsymbol{A} \boldsymbol{\beta} \geq 0$$

gilt. Damit ist $\boldsymbol{K}$ positiv semidefinit.

(4)

$$\boldsymbol{a}^T \boldsymbol{K} \boldsymbol{a} = \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j f(\boldsymbol{x}^i) f(\boldsymbol{x}^j)$$

$$= \sum_{i=1}^{l} a_i f(\boldsymbol{x}^i) \sum_{j=1}^{l} a_j f(\boldsymbol{x}^j) = \left(\sum_{i=1}^{l} a_i f(\boldsymbol{x}^i) \right)^2 \geq 0 \, .$$

(5) $\boldsymbol{K}$ ist positiv semidefinit nach $1., 2.$ und $4.$

(6) Exponentialfunktionen können beliebig genau durch Polynome mit positiven Koeffizienten approximiert werden [5]. Anwendung von $5.$ liefert den Beweis.

Diese Beispiele zeigen, dass man in einem SVM-Modell auch mehrere Kerne über eine Linearkombination verwenden kann. Die optimalen Gewichte müssen dann allerdings günstig geschätzt werden.

Ein wichtiger Mercer-Kern, der auch in der Software zur Verfügung steht, ist der Gauß-Kern, der definiert ist als

$$K(\boldsymbol{x}, \boldsymbol{z}) := e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|^2}{\sigma^2}} \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \, . \tag{33}$$

Dabei ist $\sigma > 0$ ein zu wählender Parameter. Dass der Gauß-Kern tatsächlich ein Mercer-Kern und damit ein Kern nach unserer Definition ist, kann man leicht zeigen. Dazu formt man den Ausdruck (33) um:

$$K(\boldsymbol{x}, \boldsymbol{z}) = e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|^2}{\sigma^2}} = e^{\frac{-\|\boldsymbol{x}\|^2}{\sigma^2}} \cdot e^{\frac{-\|\boldsymbol{z}\|^2}{\sigma^2}} \cdot e^{\frac{2\langle \boldsymbol{x}, \boldsymbol{z} \rangle}{\sigma^2}} = f_1(\boldsymbol{x}) \cdot f_1(\boldsymbol{z}) \cdot K_1(\boldsymbol{x}, \boldsymbol{z}) \, .$$

$f_1(\boldsymbol{x}) \cdot f_1(\boldsymbol{z})$ ist ein Kern (K_2). $K_2 \cdot K_1$ ist ein Kern und damit ist K ein Kern.

Der Gauß-Kern gehört zur Gruppe der Kerne, die sich für sehr viele Probleme eignen und wird neben dem Polynomial-Kern

$$K(\boldsymbol{x}, \boldsymbol{z}) := (\langle \boldsymbol{x}, \boldsymbol{z} \rangle_2 + 1)^d, \; d = 1, 2, ... \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \tag{34}$$

oft als Standardkern bezeichnet. Der Polynomial-Kern besitzt ebenfalls einen Parameter.

Als Ergänzung zum Gauß-Kern wird ein sogenannter Slater-Kern verwendet. Er basiert auf der bekannten Slater-Funktion und unterscheidet sich vom Gauß-Kern lediglich in der Berechnung der Norm. Der Slater-Kern hat die Form

$$K(\boldsymbol{x}, \boldsymbol{z}) := e^{\frac{-\|\boldsymbol{x}-\boldsymbol{z}\|}{\sigma^2}} \quad (\boldsymbol{x}, \boldsymbol{z} \in X) \,. \tag{35}$$

Zusätzlich ist ein Tanimoto-Kern implementiert worden. Er basiert auf dem Tanimoto-Koeffizienten und ist für binäre Datensätze geeignet. Er hat die Form

$$K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2)}{2I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) + 2I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) + I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2)} \,. \tag{36}$$

Die Funktion $I_{i,j}(\cdot, \cdot)$ berechnet, wie oft der Wert i in der ersten Komponente auftritt während **gleichzeitig** der Wert j in der zweiten Komponente zu verzeichnen ist. Die Reihenfolge ist von Bedeutung. Dieser Kern kann nur für $0/1$-wertige Daten sinnvoll ausgewertet werden. Für reelle Daten liefert er zwar auch Ergebnisse, da der Abgleich intern über Skalarprodukte implementiert ist, jedoch sind die Ergebnisse dabei nicht interpretierbar.

Anhand der Definition der Funktion $I_{i,j}(\cdot, \cdot)$ ist klar, dass folgende Gleichungen gelten:

- $I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_2$,
- $I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_1 - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2)$,
- $I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_2^T \boldsymbol{x}_2 - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2)$, und
- $I_{00}(\boldsymbol{x}_1, \boldsymbol{x}_2) = n - I_{11}(\boldsymbol{x}_1, \boldsymbol{x}_2) - I_{10}(\boldsymbol{x}_1, \boldsymbol{x}_2) - I_{01}(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Damit kann man die Darstellung (36) umformen:

$$\begin{aligned} K(\boldsymbol{x}_1, \boldsymbol{x}_2) &= \frac{\boldsymbol{x}_1^T \boldsymbol{x}_2 + n + \boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2}{n + \boldsymbol{x}_2^T \boldsymbol{x}_2 + \boldsymbol{x}_1^T \boldsymbol{x}_1 - 2\boldsymbol{x}_1^T \boldsymbol{x}_2} \\ &= \frac{n + 2\boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2}{n - 2\boldsymbol{x}_1^T \boldsymbol{x}_2 + \boldsymbol{x}_1^T \boldsymbol{x}_1 + \boldsymbol{x}_2^T \boldsymbol{x}_2} \\ &= \frac{n + t}{n - t} \,, \end{aligned}$$

wobei wir t definieren als

$$t = 2\boldsymbol{x}_1^T \boldsymbol{x}_2 - \boldsymbol{x}_1^T \boldsymbol{x}_1 - \boldsymbol{x}_2^T \boldsymbol{x}_2 \,.$$

Der Tanimoto-Kern besitzt keinen Parameter.

3.3 Klassifikation

In der Praxis werden je nach Zielsetzung und Möglichkeiten zwei Arten binärer Support-Vektor-Maschinen unterschieden. Diese werden als nächstes vorgestellt. Dabei wird sich auch zeigen, dass die darin entstehenden Optimierungsaufgaben immer quadratisch und somit konvex sind. Im Abschnitt 3.1 wurden Optimierungsaufgaben im $\mathbb{R}^n$ $(n \in \mathbb{N})$ betrachtet. Oft hat man es jedoch implizit mit einem unendlichdimensionalen Merkmalsraum zu tun, siehe dazu Abschnitt 3.2. Kann man die Theorie von Lagrange und alle Dualitätsaussagen dennoch nutzen?

Alle Konzepte der konvexen Optimierung sind auch im Raum F für $N = \infty$ anwendbar. Die Grundlagen dazu findet man in [21], Kapitel 8 - „Global theory of constrained optimization“. Im Einzelnen können dort die Theorie von Lagrange, Eigenschaften und Existenz von Sattelpunkten sowie Dualitätsaussagen nachgelesen werden. Die Problematik von $N = \infty$ innerhalb konvexer Optimierungsaufgaben wurde in der vorliegenden Literatur zu SV-Maschinen allerdings nicht erwähnt. Prinzipiell ist die Brücke zwischen Optimierung und maschinellem Lernen noch sehr klein und begrenzt sich im Allgemeinen auf theoretische Betrachtungen und die Nutzung von klassischen Optimierungsalgorithmen.

3.3.1 Klassifikation maximaler Trenngüte

Dieser Lernalgorithmus kann nur für linear trennbare Trainingsdaten verwendet werden. Er erzeugt eine Hyperebene mit größtmöglicher geometrischer Marge (maximal margin classifier). Oft wird dieses Verfahren als hard margin classifier bezeichnet [5, 33]. [34] reserviert diesen Namen jedoch für ein Modell, das für nichtseparierbare Daten angewendet wird. Der Vorteil solcher Hyperebenen liegt in der Tatsache, dass der Generalisierungsfehler bei wachsender geometrischer Trenngüte sinkt. Damit dieser Abschnitt nicht verwirrt, folgen die Gründe dafür erst im Abschnitt 3.4 nach einer Einführung in die Generalisierungstheorie. Es sei aber ab hier schon angenommen, dass eine große Marge für gute Qualität eines Klassifikators spricht.

Gegeben sei ein linear trennbarer Trainingsdatensatz S_l. Eine Lösung $(\boldsymbol{w}^*, b^*)$ des Optimierungsproblems

$$\min_{\boldsymbol{w} \in F, b \in \mathbb{R}} \quad \langle \boldsymbol{w}, \boldsymbol{w} \rangle_F \tag{37}$$

unter den Nebenbedingungen

$$y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b) \geq 1 \;\; (i = 1, ..., l)$$

realisiert die Hyperebene maximaler geometrischer Trenngüte. Die Marge hat dann den Wert $\gamma^g = 1/\|\boldsymbol{w}^*\|_F$. Beweis: Die Skalierung der Hyperebene $(\boldsymbol{w}^*, b^*)$ mit einem beliebigen $\lambda \in \mathbb{R}$ verändert deren geometrische Lage nicht, variiert aber die funktionale Marge.

Aus Abschnitt 2 ist bekannt, dass $\gamma^g = \gamma/\|\boldsymbol{w}^*\|_F$ gilt. Man kann daher, anstatt die geometrische Trenngüte zu maximieren, auch die funktionale Trenngüte konstant halten und die Norm des Parametervektors minimieren. Das obige Minimierungsproblem entsteht bei der Wahl einer funktionalen Trenngüte mit Wert 1.

An dieser Stelle geht man zur dualen Form der Aufgabe (37) über, denn im Abschnitt 3.2 hat sich gezeigt, dass diese den Einsatz von Kernen möglich macht. Außerdem vereinfachen sich die Nebenbedingungen, wie noch zu sehen sein wird.

Die Lagrange-Funktion für das Ausgangsproblem (37) hat die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) := \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F - \sum_{i=1}^{l} \alpha_i \cdot \Big(y_i\big(\langle \boldsymbol{w}\ , \boldsymbol{\phi}(\boldsymbol{x}^i)\rangle_F + b\big) - 1\Big)\ . \tag{38}$$

Der Faktor $1/2$ wurde zur Vereinfachung der weiteren Gleichungen gewählt. Er verändert als monotone Transformation das Ergebnis nicht.

Die notwendigen Bedingungen für einen stationären Punkt der Lagrange-Funktion in $(\boldsymbol{w}, b)$ bei festem $\boldsymbol{\alpha}$

$$\frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{l} y_i\alpha_i\boldsymbol{\phi}(\boldsymbol{x}^i) \stackrel{!}{=} 0, \qquad \frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial b} = -\sum_{i=1}^{l} y_i\alpha_i \stackrel{!}{=} 0 \tag{39}$$

werden in (38) eingesetzt und es ergibt sich die Funktion W, die der Funktion θ in (17) entspricht:

$$\begin{aligned} W(\boldsymbol{\alpha}) &:= \inf_{\boldsymbol{w}\in F, b\in\mathbb{R}} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) \\ &= \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j\langle\boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j)\rangle_F \\ &\stackrel{(24)}{=} \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j)\ . \end{aligned} \tag{40}$$

Die entstandene Funktion W ist nur noch von $\boldsymbol{\alpha}$ abhängig, denn alle anderen Informationen sind durch S_l und die Wahl einer impliziten Datentransformation über den Kern K gegeben.

Die bisher gewonnenen Erkenntnisse werden an dieser Stelle zusammengefasst. Sei S_l ein linear trennbarer Trainingsdatensatz und $\boldsymbol{\alpha}^*$ löse das duale Optimierungsproblem

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) \tag{41}$$

unter den Nebenbedingungen

$$\begin{aligned} \sum_{i=1}^{l} y_i\alpha_i &= 0 \ \ (i=1,..,l)\ , \\ \alpha_i &\geq 0 \ \ (i=1,...,l)\ . \end{aligned}$$

Dann realisiert der Vektor

$$\boldsymbol{w}^* = \sum_{i=1}^{l} y_i\alpha_i^*\boldsymbol{\phi}(\boldsymbol{x}^i) \in F \tag{42}$$

die Hyperebene maximaler geometrischer Trenngüte.

Um zu zeigen, wie Support-Vektor-Maschinen zu ihrem Namen kamen, werden erneut die wichtigen Karush-Kuhn-Tucker-Bedingungen von Seite 114 betrachtet. Diese führen zu den folgenden Forderungen:

$$\alpha_i^* \cdot \Big(y_i\big(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i)\rangle_F + b^*\big) - 1\Big) = 0 \quad (i=1,...,l)\ .$$

Sie sagen aus, dass nur diejenigen α_i^* streng positiv sein können, für welche die zugehörigen Merkmale $\boldsymbol{\phi}(\boldsymbol{x}^i)$ einen funktionalen Abstand zur Hyperebene von genau 1 haben. An dieser Stelle kann man den Begriff der Support-Vektoren erklären.

Für alle i mit $\alpha_i^* \neq 0$ bezeichnet man die Eingabevektoren $\boldsymbol{x}^i$ des Trainingsdatensatzes als Support-Vektoren.

Nach (42) tragen nicht alle Lernpaare, sondern nur die Support-Vektoren zur Bestimmung von $\boldsymbol{w}^*$ bei. Abbildung 7 stellt die Situation vereinfacht dar.

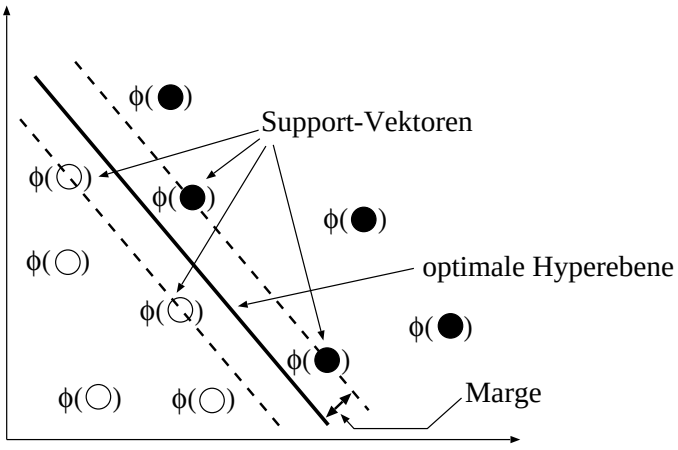


Abbildung 7: Lage von Support-Vektoren im Merkmalsraum.

Das Verfahren maximaler Trenngüte kann nur aus Hypothesen auswählen, welche alle Trainingsdaten korrekt zuordnen und dabei eine funktionale Marge von eins einhalten. Beispiele, die falsch klassifiziert werden oder sich zu nah an der Hyperebene befinden, haben funktionale Abstände, die kleiner als 1 sind. Im Fall falscher Klassifikationen sind diese sogar negativ, siehe Abschnitt 2.2. Für Algorithmen, die mit solchen Situationen arbeiten, sei auf den nächsten Abschnitt verwiesen.

Wenn die Menge aller i mit $\alpha_i^* \neq 0$ als $\mathcal{SV}$ bezeichnet wird, dann kann der Wert der Zielfunktion f an einer Stelle $\boldsymbol{x}$ berechnet werden als

$$\begin{aligned} f(\boldsymbol{x}) &\overset{(22)}{=} \sum_{j=1}^{N} w_j^* \phi_j(\boldsymbol{x}) + b^* \overset{(42)}{=} \sum_{j=1}^{N} \left(\sum_{i=1}^{l} y_i \alpha_i^* \phi_j(\boldsymbol{x}^i) \right) \phi_j(\boldsymbol{x}) + b^* \\ &= \sum_{i=1}^{l} \sum_{j=1}^{N} y_i \alpha_i^* \phi_j(\boldsymbol{x}^i) \phi_j(\boldsymbol{x}) + b^* = \sum_{i=1}^{l} y_i \alpha_i^* \sum_{j=1}^{N} \phi_j(\boldsymbol{x}^i) \phi_j(\boldsymbol{x}) + b^* \\ &\overset{(21)}{=} \sum_{i=1}^{l} y_i \alpha_i^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b^* = \sum_{i \in \mathcal{SV}} y_i \alpha_i^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}) \rangle_F + b^* \\ &\overset{(24)}{=} \sum_{i \in \mathcal{SV}} y_i \alpha_i^* K(\boldsymbol{x}^i, \boldsymbol{x}) + b^* \,. \end{aligned} \tag{43}$$

Als einzige Unbekannte bleibt noch der Wert von b^* übrig. Dieser kann theoretisch leicht bestimmt werden.

Es gilt

$$b^* = \frac{1}{y_i} - \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F \quad (i \in \mathcal{SV}) \,, \tag{44}$$

denn für $i \in \mathcal{SV}$ gilt $\alpha_i^* \cdot [y_i \left(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b^* \right) - 1] = 0$ (KKT-Bedingung). Umgestellt nach b^* folgt (44).

Praktisch hat (44) wenig Nutzen, da die Merkmalsabbildung $\boldsymbol{\phi}$ im Allgemeinen nicht explizit bekannt ist. Dualitätsaussagen und die Theorie der Kerne können aber auch hier helfen.

Die KKT-Bedingungen

$$\alpha_i^* \cdot \left[y_i \left(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}^i) \rangle_F + b^* \right) - 1 \right] = 0 \;\; (i = 1, .., l)$$

kann man umformulieren zu

$$\alpha_i^* \cdot \left[y_i \cdot f(\boldsymbol{x}^i) - 1 \right] \overset{(43)}{=} \alpha_i^* \cdot \left[y_i \left(\sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) + b^* \right) - 1 \right] = 0 \,.$$

Für $\alpha_i^* > 0$ folgt

$$y_i \cdot \left(\sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) + b^* \right) = 1 \, .$$

Wegen $y_i^2 = 1 \;\; \forall i = 1, .., l$ gilt

$$b^* = y_i - \sum_{j \in \mathcal{SV}} y_j \alpha_j^* K(\boldsymbol{x}^j, \boldsymbol{x}^i) \, , \tag{45}$$

falls $\boldsymbol{x}^i$ ein Support-Vektor ist. (45) kann direkt ausgewertet werden, ohne dass Kenntnis von $\mathcal{F}$ oder $\boldsymbol{\phi}$ nötig ist.

Als weitere Konsequenz aus den KKT-Bedingungen erhält man, dass die geometrische Marge der optimalen Hyperebene in Abhängigkeit von den positiven Lagrange-Multiplikatoren berechnet werden kann als:

$$\begin{aligned}
\gamma^g &= \frac{1}{\|\boldsymbol{w}^*\|_F} = \frac{1}{\sqrt{\langle \boldsymbol{w}^*, \boldsymbol{w}^* \rangle_F}} \overset{(42)}{=} \frac{1}{\sqrt{\sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i^* \alpha_j^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F}} \\
&= \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} y_i \alpha_i^* \sum_{j \in \mathcal{SV}} y_j \alpha_j^* \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F}} \overset{(44)}{\underset{y_i \cdot f(\boldsymbol{x}^i)=1}{=}} \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} y_i \alpha_i^* \left(\frac{1}{y_i} - b^* \right)}} \\
&= \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} \alpha_i^* - b^* \sum_{i \in \mathcal{SV}} y_i \alpha_i^*}} \overset{\sum y_i \alpha_i^* = 0}{=} \frac{1}{\sqrt{\sum_{i \in \mathcal{SV}} \alpha_i^*}} \, .
\end{aligned}$$

Für ein fertig trainiertes Modell kann man also - ohne Kenntnis der Merkmale - aus den Lagrange-Multiplikatoren die Marge im Merkmalsraum bestimmen. $\sum \alpha_i^*$ ist immer streng positiv, da Support-Vektoren streng positive Lagrange-Multiplikatoren haben müssen.

3.3.2 Klassifikation nichtseparierbarer Daten

Die Methode der maximalen Trenngüte hat einen entscheidenden Nachteil für Anwendungen aus der Praxis. Sie produziert nur Ergebnisse für linear trennbare Lerndaten. Diese Voraussetzung ist leider oft nicht erfüllt, beispielsweise wenn man verrauschte Daten nutzen muss. Deswegen werden in diesem Abschnitt ausgehend von der Klassifikation maximaler Trenngüte abgeschwächte Verfahren vorgestellt. Diese werden als soft margin classifier bezeichnet. Vladimir Vapnik führte zunächst einen sogenannten hard margin classifier ein, welcher aus Gründen der einfacheren Umsetzung in einen soft margin classifier umformuliert wurde und hier nicht vorgestellt werden soll. Details lese man in [34]

nach. Der Begriff des hard margin classifiers wird heutzutage fälschlicherweise gleichgesetzt mit dem sogenannten maximal margin classifier des letzten Abschnittes.

Unter nichtseparierbaren Daten werden auch diejenigen Daten verstanden, welche nur eine Klassifikationsfunktion maximaler Trenngüte erzeugen können, die durch Überanpassung gekennzeichnet ist, also für eine gute Klassifikation unbrauchbar sein wird.

Die im Folgenden beschriebene Methode der weichen Trennung setzt an den funktionalen Abständen der Trainingsdaten an und formuliert schwächere Bedingungen zur Datentrennung.

Bisher wurden für alle Trainingspaare $i = 1, ..., l$ funktionale Abstände von $\gamma_i \geq 1$ vorausgesetzt. Jetzt benutzt man die schon bekannten Schlupfvariablen aus Abschnitt 2 und formuliert neue Bedingungen

$$\gamma_i = y_i \cdot f(\boldsymbol{x}^i) \geq 1 - \xi_i.$$

Dabei wird gefordert, dass $\xi_i \geq 0$ für alle $i = 1, ..., l$ gelten. Die Schlupfvariablen drücken aus, dass nun auch kleinere funktionale Abstände toleriert werden. Damit können verrauschte Daten beziehungsweise Datensätze, die im gewählten Merkmalsraum nicht linear trennbar sind, klassifiziert werden. Als problematisch erweist sich jedoch die Größe der ξ_i, denn für jeden Lösungsvektor $\boldsymbol{\xi}$ gibt es einen Lösungsvektor $\boldsymbol{\xi'}$ mit $\xi_i \leq \xi_i'$. Es ist ersichtlich, dass $\boldsymbol{\xi}$ gewählt werden sollte. Alle i mit $\xi_i > 0$ werden als Fehler gewertet, deshalb sollten sie nicht unnötig vergrößert werden, sondern den Tatsachen entsprechen. Das muss noch in die Formulierung der Aufgabe aufgenommen werden, indem wachsende ξ_i auch die Zielfunktion wachsen lassen. Dafür wird eine Norm des Vektors $\boldsymbol{\xi}$, gewichtet mit einem noch unbekannten Faktor $C > 0$, zum Wert der Zielfunktion addiert.

In der Literatur zu Support-Vektor-Maschinen werden die beiden folgenden Verfahren vorgestellt, siehe beispielsweise [5]. Die Ansätze sind sehr ähnlich.

Betragssumme

$$\min_{\boldsymbol{w} \in F, b \in \mathbb{R}, \boldsymbol{\xi} \in \mathbb{R}^l} \left(\langle \boldsymbol{w}, \boldsymbol{w} \rangle_F + C \sum_{i=1}^{l} \xi_i \right) \tag{46}$$

unter den Nebenbedingungen

$$\begin{aligned} y_i \cdot f(\boldsymbol{x}^i) &\geq 1 - \xi_i \ \ (i = 1, .., l) \ , \\ \xi_i &\geq 0 \ \ (i = 1, ..., l) \ . \end{aligned}$$

Die Schlupfvariablen gehen hier mit der gewichteten Manhatten-Norm (1-Norm) in die Zielfunktion ein.

Die Lagrange-Funktion hat dann die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\alpha}') = \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i\Big(y_i\big(\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{w}\rangle_F + b\big) - 1 + \xi_i\Big) - \sum_{i=1}^{l}\alpha_i'\xi_i$$

mit $\alpha_i \geq 0$ und $\alpha_i' \geq 0$.

Bei Herleitung der dualen Aufgabe kommen zu den Gleichungen (39) aus dem letzten Abschnitt noch die folgenden durch Ableiten nach ξ_i dazu:

$$C - \alpha_i - \alpha_i' = 0 \;\; (i = 1, ..., l) \; .$$

Die duale Aufgabe hat nach Substitution der Variablen die Form von (41), jedoch unter den zusätzlichen Bedingungen $\alpha_i \leq C \;\; (i = 1, ..., l)$.

Euklidische Distanz

$$\min_{\boldsymbol{w}\in F, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^l} \left(\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + C\sum_{i=1}^{l}\xi_i^2 \right) \tag{47}$$

unter den Nebenbedingungen

$$y_i \cdot f(\boldsymbol{x}^i) \geq 1 - \xi_i \;\; (i = 1, ..., l) \; .$$

Die Schlupfvariablen gehen hier mit dem gewichteten Quadrat der Euklidischen Norm (2-Norm) in die Zielfunktion ein.

Die Lagrange-Funktion hat dann die Form

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle_F + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 - \sum_{i=1}^{l}\alpha_i\Big(y_i\big(\langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{w}\rangle_F + b\big) - 1 + \xi_i\Big)$$

mit $\alpha_i \geq 0$.

Die zusätzlichen Ableitungen haben jetzt die Form

$$C\xi_i - \alpha_i = 0 \;\; (i = 1, ..., l) \; .$$

Die Zielfunktion der dualen Aufgabe ergibt sich daraus als

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{\phi}(\boldsymbol{x}^i), \boldsymbol{\phi}(\boldsymbol{x}^j)\rangle_F - \frac{1}{4C}\langle \boldsymbol{\alpha}, \boldsymbol{\alpha}\rangle_2 \; . \tag{48}$$

Sie unterscheidet sich von (41) durch einen zusätzlichen Term, der von C und dem Vektor der Lagrange-Multiplikatoren abhängig ist.

Durch Zusammenfassung der beiden hinteren Summanden von (48) ergibt sich eine etwas elegantere Form der Zielfunktion:

$$\begin{aligned} W(\boldsymbol{\alpha}) &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(\langle \boldsymbol{\phi}(\boldsymbol{x}^i) \, , \boldsymbol{\phi}(\boldsymbol{x}^j) \rangle_F + \frac{1}{2C} \delta_{ij} \right) \\ &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C} \delta_{ij} \right) \, . \end{aligned}$$

Man hat analog zu (41) die Aufgabe

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^l} \quad W(\boldsymbol{\alpha}) \tag{49}$$

unter den Nebenbedingungen

$$\begin{aligned} \sum_{i=1}^{l} y_i \alpha_i &= 0 \, , \\ \alpha_i &\geq 0 \ \ (i = 1, ..., l) \end{aligned}$$

zu lösen.

Die Minimierungsaufgaben (46) und (47) unterscheiden sich nur dadurch, dass im ersten Fall alle ξ_i einfach aufsummiert werden, sie im zweiten Modell aber vorher quadriert werden. Welches Ziel wird mit der Quadrierung verfolgt? Eine positive Schlupfvariable ξ_i $(i \in \{1, ..., l\})$ verdeutlicht, dass der i-te Punkt den gewünschten funktionalen Abstand von mindestens eins verfehlt hat. Dabei kann $\boldsymbol{x}^i$ dennoch richtig klassifiziert sein, sodass ξ_i im Intervall $(0, 1]$ liegt, andernfalls gilt $\xi_i > 1$, der Punkt $\boldsymbol{x}^i$ ist also falsch klassifiziert worden. Quadrierung der Schlupfvariablen führt demnach dazu, dass Fehlklassifikationen $(\gamma_i < 0)$ stärker gewichtet werden als Klassifikationen mit positiven funktionalen Abständen γ_i $(0 \leq \gamma_i < 1)$. Bei beiden Ansätzen bleibt die Wahl des Parameters C offen. Klar ist, dass dieser eine Möglichkeit bietet, zu bestimmen, wie stark Klassifikationsfehler berücksichtigt werden sollen. Die Festlegung von C kann mit einer der Methoden, die unter dem Begriff Kreuzvalidierung zusammengefasst sind, erfolgen. Diese werden in Abschnitt 5.2 vorgestellt. Es ist auch möglich, die Schlupfvariablen mit einem Wert $k > 2$ zu potenzieren. Derartige Anwendungen sind aber bisher nicht bekannt.

3.4 Generalisierungstheorie

Ziel von Support-Vektor-Maschinen und anderen Methoden des überwachten Lernens ist immer, eine Funktion zu schätzen, die möglichst viele Daten richtig klassifiziert. Dabei handelt es sich entweder um Testdatensätze oder um bisher unbekannte Datensätze. Ein unbekannter Datensatz besteht nur aus Eingabewerten $\boldsymbol{x} \in X$, die einer Klasse zugeordnet werden sollen, ohne dass dabei weitere Informationen zur Verfügung stehen. Da man für solche Punkte nicht überprüfen kann, ob die Klassifikationen sinnvoll sind, ist es wichtig, bestimmte qualitative Merkmale der Entscheidungsfunktion zu untersuchen. Wie kann man die Qualität einer Schätzung bewerten?

Als Generalisierungs- oder Verallgemeinerungsfähigkeit bezeichnet man die Eigenschaft einer Hypothese, vom Training unabhängige Eingabedaten möglichst gut zu klassifizieren.

Diese Eigenschaft ist wünschenswert und kann daher als Maß für die Qualität von Lernalgorithmen betrachtet werden. Die Generalisierungstheorie untersucht also, ob und wann bestimmte Lernverfahren in der Lage sind, sinnvolle Ausgabewerte zu produzieren. Die schon vorgestellten Kernfunktionen erhöhen die Leistung von Lernalgorithmen, können aber auch dazu beitragen, dass Überanpassung entsteht. Ein Lernalgorithmus zeigt - wie schon auf Seite 100 erklärt wurde - überangepasstes Verhalten, wenn er die Trainingsdaten mit sehr wenigen Fehlern klassifiziert, jedoch bei Testdaten versagt.

Auch mit diesem Phänomen beschäftigt sich die Generalisierungstheorie. Zunächst wird die grundlegende Annahme dieser Theorie getroffen:

Der Trainingsdatensatz S_l ist eine zufällige Stichprobe aus dem Eingabe-Ausgabe-Raum $X \times Y$ gemäß einer festen aber unbekannten Verteilung P. Alle Paare $(\boldsymbol{x}^i, y_i),\ i = 1, ..., l$, sind somit Realisierungen unabhängiger und identisch verteilter Zufallsgrößen mit der Verteilung P. Diese Annahme gilt auch für alle Testdatensätze. Die Verteilung von S_l ist das l-fache Produktmaß von P. Es sei mit P_l bezeichnet.

Support-Vektor-Maschinen erlernen eine Zielfunktion der Form

$$f : X \to \mathbb{R}\ ,$$

welche dann im Fall binärer Klassifikation in die Entscheidungsfunktion

$$h : X \to Y,\ \ h := \operatorname{sgn}(f)$$

umgewandelt wird, siehe dazu Abschnitt 2.

Die Qualität einer Zielfunktion kann durch Einführung von Verlustfunktionen bewertet werden.

Eine Verlustfunktion ist eine Abbildung

$$c : X \times Y \times \mathbb{R} \to [0, \infty)\ . \tag{50}$$

Eine Verlustfunktion $c(\boldsymbol{x}, y, t)$ quantifiziert den „Verlust“ der entsteht, wenn die Zielfunktion f für ein Eingabe-Ausgabe-Paar $(\boldsymbol{x}, y)$ den Wert $t \in \mathbb{R}$ annimmt.

Die bekannteste Verlustfunktion ist die Funktion des $(0, 1)$-Verlustes

$$c(\boldsymbol{x}, y, t) = \begin{cases} 0 & \text{falls } \operatorname{sgn}(t) = y \\ 1 & \text{sonst} \end{cases} . \tag{51}$$

Hier werden alle Klassifikationsfehler gleich stark gewichtet. Der Verlust hängt nur davon ab, ob $\boldsymbol{x}$ durch die Zielfunktion in die richtige Klasse eingeordnet wurde oder nicht.

Etwas flexibler ist die Verlustfunktion

$$c(\boldsymbol{x}, y, t) = \begin{cases} 0 & \text{falls } \operatorname{sgn}(t) = y \\ \tilde{c}(\boldsymbol{x}, y) & \text{sonst} \end{cases} .$$

Durch eine Funktion $\tilde{c} : X \times Y \to [0, \infty)$ können bestimmte Fehler stärker gewichtet werden, zum Beispiel, wenn es von Interesse ist, in welcher Klasse der falsch eingeordnete Punkt liegt.

Die bisher vorgestellten Verlustfunktionen ordnen allen Entscheidungen, welche einen Eingabepunkt richtig klassifizieren, keinen Verlust zu. Das Konzept der Marge besagte aber, dass Klassifikationen innerhalb eines gewissen Grenzbereiches der trennenden Hyperebene unerwünscht sind. Deshalb gibt es auch Verlustfunktionen, die alle funktionalen Abstände, die kleiner als eins sind, als Verluste deklarieren:

$$c(\boldsymbol{x}, y, t) = \max\{0,\ 1 - y \cdot t\} = \begin{cases} 0 & \text{falls } y \cdot t \geq 1 \\ 1 - y \cdot t & \text{sonst .} \end{cases}$$

beziehungsweise

$$c(\boldsymbol{x}, y, t) = (\max\{0,\ (1 - y \cdot t)\})^2 .$$

Hat man sich entschieden, durch welche Verlustfunktion Fehler festgehalten werden sollen, kann man den erwarteten Verlust einer Zielfunktion betrachten.

Sei f eine Zielfunktion und c eine Verlustfunktion. Dann ist der erwartete Verlust, auch Risiko genannt, definiert als

$$R[f] := \mathbb{E}[c(\boldsymbol{x}, y, f(\boldsymbol{x}))] = \int_{X \times Y} c(\boldsymbol{x}, y, f(\boldsymbol{x}))\, dP(\boldsymbol{x}, y) . \tag{52}$$

Es ist klar, dass genau diese Größe Aussagen darüber macht, wie gut die Zielfunktion ist, aber man kann sie weder berechnen noch minimieren, da die Verteilung P unbekannt ist. Ein Ausweg aus diesem Dilemma wurde mit dem Prinzip der empirischen Risikominimierung zur Bewertung von Hypothesen (empirical risk minimization inductive principle) gefunden.

3.4.1 Empirische Risikominimierung

Das ERM-Verfahren setzt sich damit auseinander, zu überprüfen, wieviele Daten eines Trainingsdatensatzes durch eine feste Hypothese richtig klassifiziert werden. Die zu Grunde liegende Idee ist, dass aus einer Menge von Funktionen immer die gewählt werden sollte, die den Trainingsdatensatz am besten klassifiziert. Fehler, die bei Klassifikation des Trainingsdatensatzes mit der erlernten Funktion auftreten, liefern dieses empirische Maß. Es ist einfach zu bestimmen. Es wird davon ausgegangen, dass damit das Risiko von zukünftigen Fehlklassifikationen abgeschätzt werden kann. Es hat sich gezeigt, dass diese Methode nicht immer zu einer ausreichenden Generalisierungsfähigkeit führt und mit einer weiteren Idee kombiniert werden sollte [33]. Zunächst aber soll das ERM-Prinzip erklärt werden.

Für einen Trainingsdatensatz S_l, eine Zielfunktion f und eine Verlustfunktion c ist das empirische Risiko definiert als

$$R_{\text{emp}}[f] := \frac{1}{l} \sum_{i=1}^{l} c(\boldsymbol{x}^i, y_i, f(\boldsymbol{x}^i)) \; . \tag{53}$$

Das empirische Risiko erhält man also durch Einsetzen der empirischen Verteilung in (52).

Man betrachte die Verlustfunktion (51) und die Entscheidungsfunktion $h(\boldsymbol{x}) = \text{sgn}(f(\boldsymbol{x}))$. Dann ergibt sich für jedes Trainingspaar $(\boldsymbol{x}^i, y_i),\ i = 1, ..., l$, ein Verlust von

$$c_i = \frac{|h(\boldsymbol{x}^i) - y_i|}{2} \in \{0, 1\} \; .$$

Im Folgenden wird nur noch diese Verlustfunktion betrachtet. Sie ist typisch für das Problem der binären Klassifikation. Deshalb werden ab jetzt auch die Symbole $R[h]$ und $R_{\text{emp}}[h]$ anstelle von $R[f]$ und $R_{\text{emp}}[f]$ verwendet. Mit $\tilde{\mathcal{H}}$ sei der Raum aller Entscheidungsfunktionen $h = \text{sgn}(f),\ f \in \mathcal{H}$, bezeichnet. Damit entspricht $\tilde{\mathcal{H}}$ dem gewünschten Hypothesenraum.

Unter allen bisherigen Annahmen gilt für alle $\varepsilon > 0$, $h \in \tilde{\mathcal{H}}$ und $l \in \mathbb{N}$ die Ungleichung

$$P_l\{|R_{\text{emp}}[h] - R[h]| \geq \varepsilon\} \leq 2e^{-2l\varepsilon^2} \; . \tag{54}$$

Sie zeigt, wie nah das unbekannte Risiko R und das empirische Fehlermaß R_{emp} beieinander liegen. Zum Beweis wird ein Spezialfall der Hoeffding-Ungleichung [11], die Ungleichung von Chernoff [32], genutzt:

$$P_l\{|R_{\text{emp}}[h] - R[h]| \geq \varepsilon\} = P_l\left\{\left|\frac{1}{l}\sum_{i=1}^{l} c_i - \mathbb{E}[c]\right| \geq \varepsilon\right\} \overset{\text{Chernoff}}{\leq} 2e^{-2l\varepsilon^2} \; .$$

Die Minimierung des empirischen Risikos besteht darin, eine Funktion h zu suchen, die R_{emp} auf einem festen Trainingsdatensatz S_l minimiert. Das ERM-Prinzip schlägt diese Minimierung vor und geht davon aus, dass diese Methode ausreichend ist, um generalisierungsfähige Funktionen zu erlernen. Minimierung des empirischen Risikos kann unter Umständen zu Hypothesenfunktionen mit überangepasstem Verhalten führen, die im Extremfall alle Trainingspunkte korrekt klassifizieren, aber keine weiteren guten Klassifikationen hervorbringen. Das Prinzip ist daher nur dann sinnvoll, wenn einerseits das Risiko der gewählten Klassifikationsfunktion dem optimalen Risiko nahe kommt und wenn andererseits das empirische Risiko, welches diese Funktion verursacht, ein guter Schätzer für das wahre Risiko ist. Wie auch in [33] werden diese beiden Forderungen im Folgenden als Konsistenz der empirischen Risikominimierung bezeichnet, eine formale Definition dazu folgt nach einer kurzen Bemerkung zu zwei in diesem Abschnitt verwendeten Notationen.

(1) Eine Folge von Zufallsvariablen $\{X_n\}_{n\in\mathbb{N}}$ konvergiert stochastisch gegen eine Zufallsvariable X, wenn $\lim_{n\to\infty} P[\{|X_n - X| > \varepsilon\}] = 0 \ \forall\varepsilon \in (0,\infty)$; formal $X_n \xrightarrow[n\to\infty]{P} 0$.

(2) Für $a \in \mathbb{R}$ sei mit a_+ der Positivteil von a bezeichnet. Für reelle Zahlen a ist der Positivteil definiert als $\max\{0,\ a\}$.

$h^{\text{opt}} \in \tilde{\mathcal{H}}$ sei diejenige Funktion, die $R[h]$ minimiert. Dagegen sei $h^{S_l} \in \tilde{\mathcal{H}}$ die Funktion mit dem kleinsten empirischen Risiko für einen bestimmten Trainingsdatensatz S_l. Empirische Risikominimierung wird als konsistent bezeichnet, falls

$$R[h^{S_l}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0 \tag{55}$$

und

$$R_{\text{emp}}[h^{S_l}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0 \tag{56}$$

gelten.

Im Folgenden sollen hinreichende Bedingungen für die Konsistenz des ERM-Prinzips hergeleitet werden.

Für einen Hypothesenraum $\tilde{\mathcal{H}}$ sowie eine Verteilung P ist die Bedingung

$$\sup_{h\in\tilde{\mathcal{H}}}(R[h] - R_{\text{emp}}[h])_+ \xrightarrow[l\to\infty]{P} 0 \tag{57}$$

hinreichend für die Konsistenz der empirischen Risikominimierung.

Wir zeigen zunächst (55):

Für alle $h \in \tilde{\mathcal{H}}$ gelten die Ungleichungen

$$\begin{aligned} R[h] - R[h^{\text{opt}}] &\geq 0 \text{ und} \\ R_{\text{emp}}[h] - R_{\text{emp}}[h^{S_l}] &\geq 0 \,. \end{aligned}$$

Damit gelten für $h = h^{S_l}$ beziehungsweise $h = h^{\text{opt}}$

$$\underbrace{R[h^{S_l}] - R[h^{\text{opt}}]}_{(*)} \geq 0 \text{ und} \tag{58}$$

$$\underbrace{R_{\text{emp}}[h^{\text{opt}}] - R_{\text{emp}}[h^{S_l}]}_{(**)} \geq 0 \,. \tag{59}$$

Addition von (58) und (59) liefert

$$\begin{aligned} 0 &\leq R[h^{S_l}] - R[h^{\text{opt}}] + R_{\text{emp}}[h^{\text{opt}}] - R_{\text{emp}}[h^{S_l}] \\ &= R[h^{S_l}] - R_{\text{emp}}[h^{S_l}] + R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \\ &\leq \sup_{h \in \tilde{\mathcal{H}}}(R[h] - R_{\text{emp}}[h]) + R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \,. \end{aligned}$$

Aus (54) folgt: $R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}] \xrightarrow[l\to\infty]{P} 0$.

Durch zusätzliche Verwendung von (57) erhält man die Aussage, dass $(*)$ und $(**)$ jeweils stochastisch gegen 0 konvergieren, also ist (55) gesichert.

(56) folgt direkt aus:

$$R_{\text{emp}}[h^{S_l}] - R[h^{\text{opt}}] = \underbrace{R_{\text{emp}}[h^{S_l}] - R_{\text{emp}}[h^{\text{opt}}]}_{\xrightarrow[l\to\infty]{P} 0} + \underbrace{R_{\text{emp}}[h^{\text{opt}}] - R[h^{\text{opt}}]}_{\xrightarrow[l\to\infty]{P} 0 \text{ nach } (54)} \,.$$

3.4.2 Theorie von Vapnik und Chervonenkis

Diese Theorie beschäftigt sich mit der Frage, wie Lernalgorithmen aufgebaut sein müssen, damit sie zu Hypothesen mit guten Generalisierungsfähigkeiten führen. Vapnik und Chervonenkis haben gezeigt, dass die Bedingung (57) für die sogenannte nicht-triviale Konsistenz der empirischen Risikominimierung nicht nur hinreichend, wie für die Konsistenz

nach (55) und (56), sondern auch notwendig ist. Zur Definition der nicht-trivialen Konsistenz siehe Abschnitt 2.1 in [33], das Theorem zur Konsistenz folgt dann im Abschnitt 2.2. An dieser Stelle wird weiterhin versucht, hinreichende Bedingungen für die Konsistenz des ERM-Prinzips anzugeben. Deshalb muss eine sinnvolle obere Schranke für den Ausdruck

$$P_l \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R[h] - R_{\text{emp}}[h]) > \varepsilon \right\} \tag{60}$$

hergeleitet werden. Vapnik und Chervonenkis treffen für $l\varepsilon^2 \geq 2$ die folgende Abschätzung [34]:

$$P_l \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R[h] - R_{\text{emp}}[h]) > \varepsilon \right\} \leq 2P_{2l} \left\{ \sup_{h \in \tilde{\mathcal{H}}} (R^1_{\text{emp}}[h] - R^2_{\text{emp}}[h]) > \varepsilon/2 \right\} . \tag{61}$$

Dabei ist P_{2l} das $2l$-fache Produktmaß von P und bezieht sich auf Datensätze der Länge $2l$, die mit S_{2l} bezeichnet seien. R^1_{emp} bezieht sich dabei auf die ersten l Datenpaare, R^2_{emp} auf die andere Hälfte. Die Abschätzung ist nachvollziehbar: Liegen die empirischen Fehler zweier unabhängiger Datensätze nah beieinander, so muss das empirische Risiko nah am erwarteten Risiko liegen. Der rechte Teil von (61) muss weiter untersucht werden, um (60) nach oben abzuschätzen. Zunächst sollen einige Begriffe der VC-Theorie eingeführt werden.

Für eine Anzahl $l \in \mathbb{N}$ von Vektoren aus X gibt es maximal 2^l verschiedene Möglichkeiten, diese nach $\{-1, 1\}$ abzubilden. Es gilt also, dass die Funktionen $h \in \tilde{\mathcal{H}}$ eingeschränkt auf l Eingaben entweder genau 2^l oder weniger unterscheidbare Klassifikationen bereitstellen werden. Sei S_l ein Datensatz. Dann bezeichnet $\mathcal{N}^{S_l}(\tilde{\mathcal{H}})$ die maximale Anzahl der Funktionen aus $\tilde{\mathcal{H}}$, die man allein durch die Werte $h(\boldsymbol{x}^1), ..., h(\boldsymbol{x}^l)$ unterscheiden kann. Sei $l \in \mathbb{N}$ fest. Als Shattering-Koeffizient (shatter: zerrütten, zerschlagen) $\mathcal{N}^l(\tilde{\mathcal{H}})$ wird das Maximum von $\mathcal{N}^{S_l}(\tilde{\mathcal{H}})$ über alle möglichen S_l bezeichnet. Der Shattering-Koeffizient gibt an, wieviele verschiedene Ausgabefunktionen maximal auf einem Datensatz der Länge l durch $\tilde{\mathcal{H}}$ produziert werden können. Anders interpretiert gibt er die maximale Anzahl der Möglichkeiten an, einen Datensatz der Länge l durch die zur Verfügung stehenden Hypothesen in zwei Klassen einzuteilen. Existiert ein Datensatz S_l, sodass die Funktionen aus $\tilde{\mathcal{H}}$ in der Lage sind, die Eingabevektoren aller Datenpaare auf jede mögliche Art zu klassifizieren, dann gilt offensichtlich

$$\mathcal{N}^l(\tilde{\mathcal{H}}) = 2^l . \tag{62}$$

Unter allen bisherigen Annahmen gilt für das Produktmaß der rechten Seite von (61)

$$P_{2l}\left\{\sup_{h\in\tilde{\mathcal{H}}}(R^1_{\mathrm{emp}}[h]-R^2_{\mathrm{emp}}[h])>\varepsilon/2\right\}\leq 2\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\,e^{-\frac{l\varepsilon^2}{8}}\ .$$

Zum Beweis siehe [32]. Daraus und aus (61) folgt für $l\varepsilon^2\geq 2$ die Ungleichung

$$P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])>\varepsilon\right\}\leq 4\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\cdot e^{-\frac{l\varepsilon^2}{8}}\ . \tag{63}$$

Die folgende Abschätzung fasst alle bisherigen Ergebnisse zusammen und gibt eine praktische Aussage über das wahre Risiko bei Anwendung einer Klassifikationsfunktion.

Sei $\delta\in(0,1)$. Für Trainingsdatensätze der Länge l gelten mit einer Wahrscheinlichkeit von mindestens $(1-\delta)$ die Ungleichungen

$$R[h]\leq R_{\mathrm{emp}}[h]+\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)+\ln(4/\delta)\right)}\quad(h\in\tilde{\mathcal{H}})\ , \tag{64}$$

sofern l und $\varepsilon=\sqrt{(8/l)(\ln(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})])+\ln(4/\delta))}$ die Bedingung $l\varepsilon^2\geq 2$ erfüllen. Sei dazu $\delta\in(0,1)$ vorgegeben. Setze $\delta=4\,\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\cdot e^{-\frac{l\varepsilon^2}{8}}$, dann ergibt sich

$$\varepsilon=\sqrt{(8/l)(\ln(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})])+\ln(4/\delta))}\ .$$

Aus der Ungleichung (63) folgt für $l\varepsilon^2\geq 2$

$$\begin{aligned}
&P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])>\varepsilon\right\}\leq\delta\\
\Rightarrow\quad&P_l\left\{\sup_{h\in\tilde{\mathcal{H}}}(R[h]-R_{\mathrm{emp}}[h])\leq\varepsilon\right\}\geq 1-\delta\\
\Rightarrow\quad&P_l\left\{(R[h]-R_{\mathrm{emp}}[h])\leq\varepsilon\ \ \forall h\in\tilde{\mathcal{H}}\right\}\geq 1-\delta\\
\Rightarrow\quad&P_l\left\{R[h]\leq R_{\mathrm{emp}}[h]+\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)+\ln(4/\delta)\right)}\ \ \forall h\in\tilde{\mathcal{H}}\right\}\geq 1-\delta\ .
\end{aligned}$$

Aus diesem Satz folgt, dass für $\delta\in(0,1)$ und $l\varepsilon^2\geq 2$ mit einer Wahrscheinlichkeit von mindestens $(1-\delta)$ die Ungleichung

$$R[h^{S_l}] \leq R_{\text{emp}}[h^{S_l}] + \underbrace{\sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right) + \ln(4/\delta)\right)}}_{(KT)} \tag{65}$$

gilt.

Welche Vorteile oder Nachteile liefert die Abschätzung (65)?

- Vorteil: Die Abschätzung gilt für alle Funktionen des Hypothesenraumes in gleichem Maße. Die Ungleichung (65) ist daher nicht nur für Lernalgorithmen, die das ERM-Prinzip umsetzen, anwendbar, sondern bietet eine Abschätzung des Risikos für eine von einem beliebigen Lernalgorithmus gewählte Klassifikationsfunktion h^{S_l}.
- Nachteil: Kennt man von der gesuchten Hypothese h mehr als nur den Raum $\tilde{\mathcal{H}}$, aus dem sie stammt, wird man genauere Abschätzungen als (65) finden können.

Man betrachte nun die Struktur von (65). Der unbekannte Fehler wird durch zwei Terme additiv bestimmt. Wie verhalten sich die einzelnen Komponenten? Je reichhaltiger der Hypothesenraum, umso eher kann man eine Funktion mit kleinem empirischen Fehler wählen. Der zweite Ausdruck (KT) kann aber stark wachsen, wenn sich der Hypothesenraum vergrößert. Dazu sei an die Definition des Shattering-Koeffizienten erinnert. Im Folgenden sei dieser Teil als Kapazitätsterm bezeichnet.

Aus der Abschätzung (65) beziehungsweise (63) folgt die Bedingung (57), wenn ein geeigneter Hypothesenraum gewählt wird. Man kann also festhalten, dass man Konsistenz der empirischen Risikominimierung durch Festlegung geeigneter Räume $\tilde{\mathcal{H}}$ sichern kann. Was man unter einem geeigneten Hypothesenraum versteht und mit welchen weiteren Mitteln man das Risiko von Klassifikationsfunktionen abschätzen kann, wird am Ende dieses Abschnittes erklärt. Zunächst werden einige wichtige Werkzeuge der VC-Theorie vorgestellt, die dazu benutzt werden sollen, den Kapazitätsterm in (65) weiter nach oben abzuschätzen.

Die Höhe von $\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right)$ im Kapazitätsterm beeinflusst das Risiko. Für $l \in \mathbb{N}$ und einen Hypothesenraum $\tilde{\mathcal{H}}$ ist die Wachstumsfunktion definiert als

$$G_{\tilde{\mathcal{H}}}(l) := \max_{S_l \in (X \times \{-1,1\})^l} \ln \mathcal{N}^{S_l}(\tilde{\mathcal{H}}) \, .$$

Offensichtlich gilt $G_{\tilde{\mathcal{H}}}(l) = \ln \mathcal{N}^l(\tilde{\mathcal{H}})$, denn die natürliche Logarithmusfunktion ist im gesamten Definitionsbereich streng monoton wachsend. Daraus erhält man die wichtige Beziehung

$$\ln\left(\mathbb{E}[\mathcal{N}^{S_l}(\tilde{\mathcal{H}})]\right) \leq G_{\tilde{\mathcal{H}}}(l) \, . \tag{66}$$

Das heißt, ist eine Aussage über den Wert der Wachstumsfunktion möglich, so lässt sich die Ungleichung (64) auswerten. In [33] wird eine wichtige Eigenschaft der Wachstumsfunktion angegeben, die hier zitiert werden soll. Entweder, die Wachstumsfunktion erfüllt für alle $l > 0$ die Gleichung

$$G_{\tilde{\mathcal{H}}}(l) = l \cdot \ln(2) \tag{67}$$

oder es existiert ein $\mathcal{D} \in \mathbb{N}$, sodass für $l \leq \mathcal{D}$ die Gleichung (67) gilt und für $l > \mathcal{D}$

$$G_{\tilde{\mathcal{H}}}(l) \leq \mathcal{D}\left(\ln \frac{l}{\mathcal{D}} + 1\right) \tag{68}$$

erfüllt ist. Diese Funktion wächst also linear mit der Größe des Trainingsdatensatzes, bis $l = \mathcal{D}$ erreicht ist. Danach steigt ihr Wert nur noch sehr langsam. Den Wert $\mathcal{D}$ nennt man nach Vapnik und Chervonenkis VC-Dimension. Sie gibt die maximale Anzahl l der Punkte an, welche die Bedingung (62) erfüllen. Falls für alle $l > 0$ Bedingung (67) erfüllt ist, setzt man $\mathcal{D} = \infty$.

Sei $X \subseteq \mathbb{R}^2$. Die VC-Dimension des Raumes der trennenden Hyperebenen in X, in diesem Fall aller Geraden, ist die größte Zahl $l \in \mathbb{N}$, sodass alle l-Tupel von Punkten aus X durch Hyperebenen auf jede erdenkliche Weise getrennt werden können. Man kann sich schnell klarmachen, dass dies für drei Punkte immer möglich ist. Bei vier Punkten kann es Situationen geben, in denen die gewünschten Zuordnungen zu 1 und -1 nicht ausschließlich durch Geraden erreicht werden können. Ein Beispiel ist in Abbildung 8 dargestellt. Daraus folgt, dass die VC-Dimension trennender Hyperebenen im $\mathbb{R}^2$ genau drei beträgt. Im Allgemeinen ist die VC-Dimension (n-1)-dimensionaler Hyperebenen immer n+1.

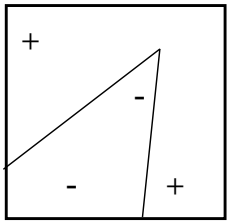


Abbildung 8: Nichtlineare Trennung von vier Punkten im $\mathbb{R}^2$.

Die Kenntnis der VC-Dimension ist ausreichend für eine Aussage über das Verhalten der Wachstumsfunktion und damit auch des Kapazitätsterms. Für eine bessere Übersicht soll die Ungleichung (64) mit diesen Erkenntnissen erweitert werden:

$$
\begin{aligned}
R[h] &\overset{(64)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(\ln\left(\mathbb{E}[\mathcal{N}^{S_{2l}}(\tilde{\mathcal{H}})]\right) + \ln(4/\delta)\right)} \\
&\overset{(66)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(G_{\tilde{\mathcal{H}}}(2l) + \ln(4/\delta)\right)} \\
&\overset{(68)}{\leq} R_{\text{emp}}[h] + \sqrt{\frac{8}{l}\left(\mathcal{D}\left(\ln\frac{2l}{\mathcal{D}} + 1\right) + \ln(4/\delta)\right)}. \qquad (69)
\end{aligned}
$$

Für $\mathcal{D} = \infty$ gilt (67), sodass der Kapazitätsterm wegen

$$
\frac{8}{l}\left(G_{\tilde{\mathcal{H}}}(2l)\right) \overset{(67)}{=} \frac{8}{l}\left(2l \cdot \ln(2)\right) = 16 \cdot \ln(2)
$$

eine Konstante enthalten würde. Bedingung (57) ist dann nicht notwendigerweise erfüllt. Zusätzlich existieren Abschätzungen nach unten [5], aus denen folgt, dass die lineare Lerntheorie in Merkmalsräumen sehr großer VC-Dimension versagt. Eben solche Räume sind laut Abschnitt 3.2 gewünscht. Der auf Seite 127 vorgestellte Kern führt sogar zu $\mathcal{D} = \infty$ [32], falls die Grammatrix vollen Rang besitzt, d.h. falls $\sigma \neq 0$ gilt und alle Trainingspunkte verschieden sind. Mit diesem Problem und dessen Lösung setzt sich der nächste Abschnitt auseinander. Im Folgenden werden Ansätze vorgestellt, mit denen das Risiko von Hypothesen genauer als bisher abgeschätzt werden kann.

3.4.3 Strukturelle Risikominimierung

Es wurde gezeigt, dass der Kapazitätsterm in (65) nicht von den Eigenschaften einer bestimmten Funktion in $\tilde{\mathcal{H}}$ abhängt, sondern durch Merkmale dieses Raumes bestimmt wird. Aus diesem Grund ist es nicht ausreichend, eine geeignete Funktion h zu wählen. Die Generalisierungsfähigkeit wird sich erst verbessern, wenn günstige obere Schranken für den Kapazitätsterm gefunden werden.

Ein erster Ansatz dafür ist die Einführung einer Struktur in $\tilde{\mathcal{H}}$, sodass die Summe aus empirischem Risiko und Kapazitätsterm über die Wahl der Struktur minimiert werden kann. Diese Methode wird als strukturelle Risikominimierung (structural risk minimization inductive principle) bezeichnet und erweitert das ERM-Prinzip. Strukturelle Risikominimierung innerhalb der VC-Theorie ist ein wichtiger Punkt, in dem sich Support-Vektor-Maschinen ganz klar von der Theorie der neuronalen Netze unterscheiden. Diese minimieren bestimmte Fehlermaße auf der Trainingsmenge, betrachten aber nicht die Eigenschaften der Hypothesenräume. Die Generalisierungsfähigkeit wird aber genau dadurch besser.

Idee: Man wähle einen Hypothesenraum $\tilde{\mathcal{H}}_1$, der sehr klein ist. Mit jedem Schritt vergrößere man diesen, also $\tilde{\mathcal{H}}_1 \subset \tilde{\mathcal{H}}_2 \subset ....$ Der Raum $\tilde{\mathcal{H}} := \tilde{\mathcal{H}}_k$ wird dann so gewählt, dass es eine Funktion $h \in \tilde{\mathcal{H}}$ mit einem kleinen empirischen Fehler gibt, wobei k möglichst klein sein soll. Gesucht sind also Hypothesenräume $\tilde{\mathcal{H}}_k \subset \tilde{\mathcal{H}}$ mit $\mathcal{D}_k \ll \infty$, in denen das ERM-Prinzip angewendet werden kann. Problematisch bei diesem Ansatz ist die Tatasche, dass im Allgemeinen für $\tilde{\mathcal{H}}$ keine Struktur bekannt ist, die die obige Idee realisieren könnte. Er ist also von geringem praktischen Nutzen. Sinnvoller ist es, wenn man Informationen nutzt, die durch die Trainingsdaten verfügbar sind. Damit beschäftigt sich der zweite Ansatz.

Wie man an (69) erkennen kann, liefert die VC-Theorie Schranken für das Risiko von Hypothesen, die von der Verteilung P unabhängig sind. Diese Schranken sind im Allgemeinen sehr groß, weil sie auch für ungünstige Verteilungen gelten müssen. Die Abschätzung des tatsächlichen Risikos wird damit schwierig. Kann man voraussetzen, dass die möglichen Verteilungen günstig sind, gibt es signifikant kleinere Schranken. Support-Vektor-Maschinen greifen diese Idee auf und suchen datenabhängige Fehlerschranken. Dabei werden Maße für die „Gutartigkeit“ von Verteilungen definiert, die dann angewendet auf den Trainingsdatensatz die zugehörigen empirischen Größen liefern. Diese Methode wird von SV-Maschinen umgesetzt und ist unter der Bezeichnung der datenabhängigen strukturellen Risikominimierung bekannt.

In Abschnitt 2.2 wurden die funktionalen Abstände der Punkte eines Trainingsdatensatzes von einer Hyperebene eingeführt. Die Güte einer Klassifikation kann deshalb über die Verteilung dieser Abstände bestimmt werden. Die Verfahren des Abschnittes 3.3 basieren auf Merkmalen dieser Verteilung. Die im Folgenden angegebenen datenabhängigen Risikoabschätzungen werden diese Verfahren rechtfertigen.

Die Klassifikation maximaler Trenngüte verwendet das Minimum der Verteilung der funktionalen Abstände. Es gilt [5]:

Sei $\mathcal{H}_1$ der Raum aller linearen Zielfunktionen f, die einen normierten Richtungsvektor $\boldsymbol{w}$ besitzen, $\gamma > 0$ sei eine reelle Zahl. Für jede Verteilung P auf $X \times \{-1, 1\}$, deren Träger in X in einer Kugel mit Radius R um den Ursprung liegt, gelten mit einer Wahrscheinlichkeit von $(1 - \delta)$ für Trainingsdatensätze S_l der Länge l die Ungleichungen

$$R[f] \leq \varepsilon(l, \mathcal{H}_1, \delta, \gamma) = \frac{2}{l}\left(\frac{64R^2}{\gamma^2}\log_2\frac{el\gamma}{8R^2}\log_2\frac{32l}{\gamma^2} + \log_2\frac{4}{\delta}\right) \quad \forall f \in \mathcal{H}_1 \,, \tag{70}$$

falls $l\varepsilon^2 \geq 2$ sowie $64R^2/\gamma^2 < l$ und falls eine Marge von mindestens γ eingehalten wird.

Man beachte, dass die Abschätzung (70) nicht mehr von der VC-Dimension abhängt. Kleine Fehlerschranken können realisiert werden, falls γ und l hinreichend groß sind. Damit ist die Marge direkt für die Fähigkeit der Generalisierung verantwortlich.

Verfahren für die Klassifikation nichtseparierbarer Trainingsdaten versuchen, eine vorgegebene funktionale Marge γ zu erreichen und berechnen dann das Minimum der Verteilung

einer Norm des Vektors der Schlupfvariablen $\boldsymbol{\xi}$. Es gelten die folgenden Abschätzungen, die ebenfalls aus [5] stammen:

Es sei $\mathcal{H}_1$ der Raum aller linearen Zielfunktionen f, die einen normierten Richtungsvektor $\boldsymbol{w}$ besitzen, $\gamma > 0$ sei eine reelle Zahl. Dann existiert eine Konstante c, sodass für jede Verteilung P auf $X \times \{-1, 1\}$, deren Träger in X in einer Kugel mit Radius R um den Ursprung liegt, mit einer Wahrscheinlichkeit von $(1 - \delta)$ für Trainingsdatensätze S_l der Länge l die folgenden Ungleichungen gelten:

$$R[f] \leq \frac{c}{l}\left(\frac{R^2 + \|\boldsymbol{\xi}\|_2^2}{\gamma^2}\log_2^2 l + \log_2\frac{1}{\delta}\right) \quad \forall f \in \mathcal{H}_1\ , \tag{71}$$

falls die 2-Norm verwendet wird, beziehungsweise

$$R[f] \leq \frac{c}{l}\left(\frac{R^2 + \|\boldsymbol{\xi}\|_1^2 \log_2(1/\gamma)}{\gamma^2}\log_2^2 l + \log_2\frac{1}{\delta}\right) \quad \forall f \in \mathcal{H}_1\ , \tag{72}$$

falls die 1-Norm gewählt wird.

In diesen Fällen ist der Generalisierungsfehler zusätzlich davon abhängig, wie stark jeder einzelne Trainingspunkt eine vorgegebene Marge γ verfehlt. An dieser Stelle soll noch gezeigt werden, wie man von diesen Abschätzungen zu den Verfahren des Abschnittes 3.3.2 gelangt. Im Fall der 2-Norm in (71) kann man erkennen, dass der Ausdruck

$$\frac{R^2 + \|\boldsymbol{\xi}\|_2^2}{\gamma^2} \tag{73}$$

direkten Einfluss auf die Fehlerschranke hat. Deshalb wäre es sinnvoll, diesen Ausdruck zu minimieren, wobei zu beachten ist, dass nach Voraussetzung $\|\boldsymbol{w}\| = 1$ gilt. Hebt man diese Einschränkung auf, erhält man den zu minimierenden Term

$$\frac{1}{\left(\gamma/\|\boldsymbol{w}\|\right)^2}\left(R^2 + \left(\frac{\|\boldsymbol{\xi}\|_2}{\|\boldsymbol{w}\|}\right)^2\right)\ .$$

Für $\gamma = 1$ und $C := 1/R^2$ ergibt sich die schon bekannte Darstellung

$$\min\left(\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_2^2\right)\ .$$

Die Abschätzung (72) kann in gleicher Weise untersucht werden und führt auf die Minimierung von

$$\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_1^2 \log_2(\|\boldsymbol{w}\|)\ .$$

Im Rahmen der Support-Vektor-Maschinen (siehe beispielsweise [5]) wird die leicht abgewandelte Aufgabe

$$\|\boldsymbol{w}\|^2 + C\|\boldsymbol{\xi}\|_1$$

betrachtet.

Zusammenfassend zu diesem Abschnitt über Generalisierungstheorie passt folgendes Zitat aus [15]:

„ ... we can generalize well in high dimensional spaces, if our hypothesis has a small weight vector ... “.

3.5 Zusammenfassung

In diesem Abschnitt geben wir die für die Implementierung zu Grunde liegenden Modelle in kompakter Darstellung an. Ausgewählt wurden die beiden in Abschnitt 3.3.2 vorgestellten Modelle, die Trainingsfehler zugunsten einer generalisierungsfähigen Funktion tolerieren.

Die primale Aufgabe des 1-Norm Ansatzes hat die Form

$$\min_{\boldsymbol{w}\in\mathcal{F}, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^l} \quad \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i\right) \tag{74}$$

mit den Nebenbedingungen

$$\begin{aligned} y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i\rangle + b) &\geq 1 - \xi_i \ \ (i = 1, ..., l)\ , \\ \xi_i &\geq 0 \ \ (i = 1, ..., l)\ . \end{aligned}$$

Die zugehörige duale Aufgabe lautet

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) - \sum_{i=1}^{l}\alpha_i \tag{75}$$

mit den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T\boldsymbol{y} &= 0\ , \\ \mathbf{0} &\leq \boldsymbol{\alpha}\ , \\ \boldsymbol{\alpha} &\leq \boldsymbol{C}\ . \end{aligned}$$

Die primale Aufgabe des 2-Norm Ansatzes hat die Form

$$\min_{\boldsymbol{w}\in\mathcal{F},b\in\mathbb{R},\boldsymbol{\xi}\in\mathbb{R}^l} \quad \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i^2\right) \tag{76}$$

mit den Nebenbedingungen

$$y_i(\langle \boldsymbol{w}, \boldsymbol{x}^i\rangle + b) \geq 1 - \xi_i \;\; (i = 1, ..., l) \; .$$

Die duale Aufgabe hat die Darstellung

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) &= \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left(K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C}\delta_{i,j}\right) - \sum_{i=1}^{l}\alpha_i \\ &= \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \tilde{K}(\boldsymbol{x}^i, \boldsymbol{x}^j) - \sum_{i=1}^{l}\alpha_i \end{aligned} \tag{77}$$

mit den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T\boldsymbol{y} &= 0 \, , \\ \boldsymbol{\alpha} &\geq \mathbf{0} \, . \end{aligned}$$

Wir haben hier die dualen Aufgaben als Minimierungsprobleme dargestellt, wobei sich keine Veränderung der Lösungen ergibt. Wie innerhalb dieses Abschnitts herausgearbeitet wurde, sind die dualen Darstellungen besonders wichtig für die Implementierung. Dazu seien die folgenden Punkte genannt.

- Die dualen Aufgaben sind konvex und besitzen deshalb keine lokalen Extrema.
- Jede duale Aufgabe besitzt genau einen globalen Extrempunkt.
- Die primalen Aufgaben arbeiten direkt mit Punkten des Merkmalsraumes. Innerhalb der Zielfunktionen der dualen Aufgaben treten ausschließlich Skalarprodukte zwischen Punkten im Merkmalsraum auf. Diese können durch Kernfunktionen die auf den Originaldaten arbeiten, ersetzt werden. Das ermöglicht die Nutzung hochdimensionaler Merkmalsräume.

Aus (40), (41) und (77) folgt direkt: Das $L2$-Norm-Modell entspricht dem *hard margin classifier* bis auf den modifizierten Kern

$$\tilde{K}(\boldsymbol{x}^i, \boldsymbol{x}^j) = K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \frac{1}{2C}\delta_{i,j} \; . \tag{78}$$

4 Algorithmen

In diesem Abschnitt werden die implementierten SVM-Algorithmen vorgestellt. Der hard margin classifier (37) ist für die Praxis nicht relevant, sodass die beiden soft margin Modelle umgesetzt worden sind. Sollte man im einen oder anderen Fall dennoch auf einer perfekten Trennung bestehen, kann das über die Wahl $C \approx \infty$, d.h. über einen sehr großen Wert für C erzwungen werden.

Abschnitt 4.1 wird sich mit der Implementierung des $L2$-Norm-Modells auseinandersetzen, wohingegen im Abschnitt 4.2 die Umsetzung einer $L1$-Norm Support-Vektor-Maschine vorgestellt wird. Wir werden im Anschluss daran erklären, welche Unterschiede zwischen diesen Modellen für die Praxis bestehen.

4.1 Nearest-Point-Algorithmen

Der iterative Nearest-Point-Algorithmus (NPA) wurde erstmals im Jahr 1999 vorgestellt [18]. Er löst SVM-Klassifikationsprobleme über folgende Aufgabe: Bestimme den minimalen Abstand zwischen zwei konvexen Polytopen, wobei die Polytope über die jeweiligen Trainingspunkte im Merkmalsraum definiert sind. Im Folgenden wird dieser Algorithmus vorgestellt. Im Anschluss daran wird gezeigt, wie der Algorithmus an die Projektaufgaben angepasst wurde.

Der Nearest-Point-Algorithmus löst das Trainingsproblem einer $L2$-Norm Support-Vektor-Maschine. Die zugehörige Optimierungsaufgabe ist in primaler (76) und dualer Form (77) bereits angegeben worden. NPA arbeitet unter zwei Bedingungen, die wir als erstes vorstellen, bevor im Anschluss auf die Details des Verfahrens eingegangen wird.

Es sei S_l ein Trainingsdatensatz. Seien $I := \{i \in \{1, ..., l\} \mid y_i = 1\}$ und $J := \{j \in \{1, ..., l\} \mid y_j = -1\}$.

Annahme 1: $I \neq \emptyset$ und $J \neq \emptyset$.

Diese Annahme stellt keine Einschränkung dar, denn ein Lernproblem mit ausschließlich Vertretern einer Klasse ist nicht sinnvoll. Dieser Fall könnte jedoch beim Aufspalten von Trainingsdaten für Validierungszwecke auftreten, siehe dazu Abschnitt 5.2. Insbesondere geschieht das oft beim Einsatz von Zufallsroutinen. Auch in den verkleinerten Trainingsdaten sollten immer Punkte beider Klassen vorhanden sein.

Aus den Eigenschaften von S_l folgen $I \cup J = \{1, ..., l\}$ und $I \cap J = \emptyset$.

Annahme 2: Es existieren $\boldsymbol{w}$ und b im zulässigen Bereich von (37).

Um auch nichtseparierbare Daten zu klassifizieren, wurden die abgeschwächten Klassifikationsverfahren eingeführt. Annahme 2 kann somit immer erfüllt werden. Es sei nochmals erwähnt, dass die Aufgabe (77) gelöst werden kann, indem man in der Ausgangsaufgabe (41) die Kernmatrix des Trainingsdatensatzes mit dem Parameter C anpasst. Der neue Kern $\tilde{K}$ wurde auf Seite 150 bereits definiert.

Im Folgenden wird deshalb nur (37) betrachtet und gezeigt, wie diese Aufgabe als Nearest-Point-Problem formuliert und gelöst werden kann.

Das in [18] vorgestellte Verfahren betrachtet Mengen und Punkte im Merkmalsraum F, dennoch wird die Merkmalsabbildung ϕ nicht als bekannt vorausgesetzt. Alle Berechnungen werden auch hier über einen Kern K realisiert. Das wird durch eine auf Skalarprodukten basierende Form ermöglicht. Details dazu folgen später.

4.1.1 Nearest-Point-Problem (NPP)

Sei $Z = \{\boldsymbol{z}^1, ..., \boldsymbol{z}^l\}$ eine endliche Menge von Punkten in F. Dann wird die konvexe Hülle $co(Z)$ von Z als

$$co(Z) := \left\{ \sum_{k=1}^{l} \beta_k \boldsymbol{z}^k \mid \boldsymbol{z}^k \in Z,\ \beta_k \geq 0,\ \sum_{k=1}^{l} \beta_k = 1 \right\}$$

definiert. Man betrachte jetzt die Mengen

$$\begin{aligned} U &:= co(\{\boldsymbol{\phi}(\boldsymbol{x}^i) \mid i \in I\}) \quad \text{und} \\ V &:= co(\{\boldsymbol{\phi}(\boldsymbol{x}^j) \mid j \in J\}) , \end{aligned}$$

die über die Merkmale in F definiert sind. Da I und J wegen $l \in \mathbb{N}$ endlich sind, stellen U und V konvexe Polytope in F dar.

Im Folgenden soll die Aufgabe

$$\min_{\boldsymbol{u} \in U,\ \boldsymbol{v} \in V} \quad \|\boldsymbol{u} - \boldsymbol{v}\|_F \tag{79}$$

betrachtet werden. Sie wird Nearest-Point-Problem genannt. Abbildung 9 veranschaulicht das Problem.

Das Paar $(\boldsymbol{w}^*, b^*)$ löst die Aufgabe (37) genau dann wenn $\boldsymbol{u}^* \in U$ und $\boldsymbol{v}^* \in V$ existieren, sodass $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ die Aufgabe (79) löst und

$$\boldsymbol{w}^* = \frac{2(\boldsymbol{u}^* - \boldsymbol{v}^*)}{\|\boldsymbol{u}^* - \boldsymbol{v}^*\|_F^2} \qquad b^* = \frac{\|\boldsymbol{v}^*\|_F^2 - \|\boldsymbol{u}^*\|_F^2}{\|\boldsymbol{u}^* - \boldsymbol{v}^*\|_F^2}$$

gelten ([18], Kapitel 2). Damit ist gesichert, dass ein Support-Vektor-Problem auch über die Minimierung eines Abstandes zwischen konvexen Polytopen gelöst werden kann.

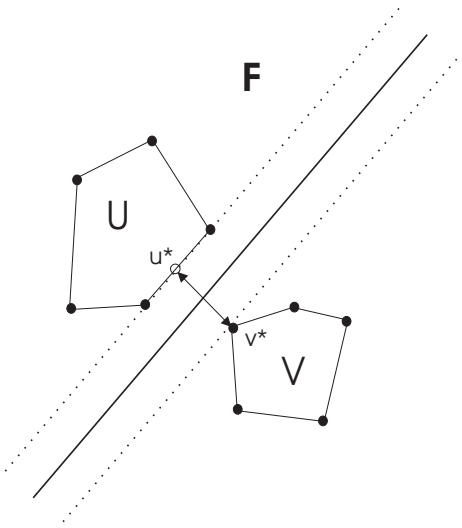


Abbildung 9: Nearest-Point-Problem im zweidimensionalen Merkmalsraum.

4.1.2 Optimalitätsbedingungen für NPP

Sei P eine kompakte Menge in F. Die sogenannte Support-Funktion $h_P : F \to \mathbb{R}$ wird definiert als

$$h_P(\boldsymbol{\vartheta}) := \max_{\boldsymbol{p} \in P} \quad \langle \boldsymbol{\vartheta}, \boldsymbol{p} \rangle_F \quad (\boldsymbol{\vartheta} \in F) \ .$$

Mit $s_P(\boldsymbol{\vartheta}) \in P$ wird eine zugehörige Lösung bezeichnet, d.h. $h_P(\boldsymbol{\vartheta}) = \langle \boldsymbol{\vartheta}, s_P(\boldsymbol{\vartheta}) \rangle_F$.

Sei nun P ein konvexes Polytop in F, also $P = co(Z)$ für eine Menge Z von Punkten in P. Es ist bekannt, dass das Maximum einer affin-linearen Funktion auf P in einem Punkt aus Z angenommen wird, genau genommen wird es in einem Punkt der minimalen Stützmenge angenommen, diese Menge ist aber in Z enthalten. Formal bedeutet das:

$$h_P(\boldsymbol{\vartheta}) = h_Z(\boldsymbol{\vartheta}) = \max_{\boldsymbol{z} \in Z} \quad \langle \boldsymbol{\vartheta}, \boldsymbol{z} \rangle_F \ ,$$

$$\exists\, \boldsymbol{z}^* \in Z : \ s_P(\boldsymbol{\vartheta}) = s_Z(\boldsymbol{\vartheta}) = \boldsymbol{z}^*, \quad \langle \boldsymbol{\vartheta}, \boldsymbol{z}^* \rangle_F = h_Z(\boldsymbol{\vartheta}) \ .$$

Die Funktion $g_P : F \times P \to \mathbb{R}$ wird definiert als

$$g_P(\boldsymbol{\vartheta}, \boldsymbol{p}) := h_P(\boldsymbol{\vartheta}) - \langle \boldsymbol{\vartheta} \ , \boldsymbol{p} \rangle_F .$$

Für alle $\boldsymbol{u} \in U$ und $\boldsymbol{v} \in V$ gelten

(1) $g(\boldsymbol{u}, \boldsymbol{v}) := g_U(\boldsymbol{v} - \boldsymbol{u}, \boldsymbol{u}) + g_V(\boldsymbol{u} - \boldsymbol{v}, \boldsymbol{v}) \geq 0$ **,**
denn $g_P(\boldsymbol{\vartheta}, \boldsymbol{p}) \geq 0 \ \ \forall\, \boldsymbol{\vartheta} \in F, \ \boldsymbol{p} \in P.$

(2) $\exists\, \overline{\boldsymbol{u}} \in U$ mit $\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{u}}\rangle_F < \langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{u}\rangle_F$
$\Rightarrow\ \exists\, \widetilde{\boldsymbol{u}} \in co(\{\boldsymbol{u}, \overline{\boldsymbol{u}}\})$ mit $\|\widetilde{\boldsymbol{u}}-\boldsymbol{v}\|_F < \|\boldsymbol{u}-\boldsymbol{v}\|_F$,
denn für eine Funktion ζ mit $\zeta(t) := \|\boldsymbol{u}+t(\overline{\boldsymbol{u}}-\boldsymbol{u})-\boldsymbol{v}\|_F^2\ (t\in\mathbb{R})$
gilt $\zeta'(t) = \sum 2(u_i + t(\overline{u}_i - u_i) - v_i)(\overline{u}_i - u_i)$.
Für $t=0$ folgt $\zeta'(0) = 2\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{u}}-\boldsymbol{u}\rangle_F < 0$.

(3) $\exists\, \overline{\boldsymbol{v}} \in V$ mit $\langle \boldsymbol{u}-\boldsymbol{v}, \overline{\boldsymbol{v}}\rangle_F > \langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}\rangle_F$
$\Rightarrow\ \exists\, \widetilde{\boldsymbol{v}} \in co(\{\boldsymbol{v}, \overline{\boldsymbol{v}}\})$ mit $\|\boldsymbol{u}-\widetilde{\boldsymbol{v}}\|_F < \|\boldsymbol{u}-\boldsymbol{v}\|_F$
Diese Aussage kann analog zu 2. gezeigt werden.

(4) $(\boldsymbol{u},\boldsymbol{v})$ löst $(79) \Leftrightarrow g(\boldsymbol{u},\boldsymbol{v}) = 0$
Gelte $g(\boldsymbol{u},\boldsymbol{v}) = 0$ und seien $\hat{\boldsymbol{u}} \in U,\ \hat{\boldsymbol{v}} \in V$ beliebig gewählt.
Wegen $g_U(\boldsymbol{v}-\boldsymbol{u}, \boldsymbol{u}) = 0$ und $g_V(\boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}) = 0$ folgen
$\langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{u}\rangle_F \le \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}\rangle_F$ und $\langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{v}\rangle_F \ge \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{v}}\rangle_F$.
Somit gilt $\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 \le \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\rangle_F$.
Zusammenfassend erhält man:

$$\begin{aligned}\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 &\le \|\boldsymbol{u}-\boldsymbol{v}\|_F^2 + \|(\boldsymbol{u}-\boldsymbol{v}) - (\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}})\|_F^2 \\ &= \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|_F^2 + 2\big(\|\boldsymbol{u}-\boldsymbol{v}\|_F^2 - \langle \boldsymbol{u}-\boldsymbol{v}, \hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\rangle_F\big) \\ &\le \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|_F^2\ .\end{aligned}$$

Daraus folgt, dass $(\boldsymbol{u},\boldsymbol{v})$ Lösung von (79) ist.
Sei nur $(\boldsymbol{u},\boldsymbol{v})$ optimal für (79). Weiterhin seien $\overline{\boldsymbol{u}} = s_U(\boldsymbol{v}-\boldsymbol{u})$ und $\overline{\boldsymbol{v}} = s_V(\boldsymbol{u}-\boldsymbol{v})$.
Unter Beachtung von 1. und 2. folgt $g(\boldsymbol{u},\boldsymbol{v}) = 0$.

Algorithmen, die nach endlich vielen Schritten eine Lösung $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ finden, müssen viele Matrixoperationen durchführen. Sie sind für große Punktmengen nicht geeignet. Deshalb werden iterative Verfahren genutzt, die pro Schritt wenig Speicher benötigen, sich der Lösung aber nur asymptotisch nähern. Hat man geeignete Abbruchkriterien, führen diese Algorithmen dennoch zu befriedigenden Näherungen für $(\boldsymbol{u}^*, \boldsymbol{v}^*)$.

Der NP-Algorithmus aus [18] basiert auf zwei traditionellen Algorithmen, die sich sinnvoll ergänzen. Sowohl deren Grundideen als auch die Art und Weise ihrer Verknüpfung werden kurz vorgestellt.

4.1.3 Algorithmus von Gilbert

Eine Lösung für (79) kann auch über die Lösung eines äquivalenten Problems gefunden werden. Auf dieser Überlegung basiert die Methode nach Gilbert aus dem Jahr 1966.

Sei $Z := U \ominus V = \{\boldsymbol{u} - \boldsymbol{v} \mid \boldsymbol{u} \in U,\ \boldsymbol{v} \in V\}$ die sogenannte Minkowski-Differenz der Polytope U und V.

Offensichtlich gilt, dass die Aufgaben

$$\min_{\boldsymbol{z} \in Z} \quad \|\boldsymbol{z}\|_F \tag{80}$$

und (79) äquivalent sind.

Seien $\boldsymbol{u} \in U$ und $\boldsymbol{v} \in V$ beliebig. Dann haben sie die Darstellungen

$$\boldsymbol{u} = \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i), \quad \sum_{i \in I} \beta_i = 1, \quad \beta_i \geq 0$$

und

$$\boldsymbol{v} = \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j), \quad \sum_{j \in J} \beta_j' = 1, \quad \beta_j' \geq 0\ .$$

Der Punkt $\boldsymbol{z} = \boldsymbol{u} - \boldsymbol{v}$ ist also von der Form

$$\begin{aligned} \boldsymbol{z} &= \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i) - \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j) \\ &= \left(\sum_{j \in J} \beta_j'\right) \sum_{i \in I} \beta_i \boldsymbol{\phi}(\boldsymbol{x}^i) - \left(\sum_{i \in I} \beta_i\right) \sum_{j \in J} \beta_j' \boldsymbol{\phi}(\boldsymbol{x}^j) \\ &= \sum_{i \in I} \sum_{j \in J} \beta_i \beta_j' \left(\boldsymbol{\phi}(\boldsymbol{x}^i) - \boldsymbol{\phi}(\boldsymbol{x}^j)\right)\ . \end{aligned}$$

Es gilt $\sum_{i \in I} \sum_{j \in J} \beta_i \beta_j' = \left(\sum_{i \in I} \beta_i\right) \left(\sum_{j \in J} \beta_j'\right) = 1$. Daraus folgt für die Menge Z als Differenz von Punkten aus U und V: $Z = co(Z_0)$, wobei

$$Z_0 := \{\boldsymbol{\phi}(\boldsymbol{x}^i) - \boldsymbol{\phi}(\boldsymbol{x}^j) \mid i \in I, j \in J\}\ .$$

Es gelten ([18], Theorem 2):

(1) Sei $\boldsymbol{z} \in Z$ beliebig. $\exists\ \overline{\boldsymbol{z}} \in Z$ mit $\|\boldsymbol{z}\|_F^2 - \langle \boldsymbol{z}, \overline{\boldsymbol{z}} \rangle_F > 0$, dann $\exists\ \widetilde{\boldsymbol{z}} \in co(\{\boldsymbol{z}, \overline{\boldsymbol{z}}\})$, sodass $\|\widetilde{\boldsymbol{z}}\|_F < \|\boldsymbol{z}\|_F$ gilt.

(2) Die Bedingung $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$ ist notwendig und hinreichend dafür, dass $\boldsymbol{z} \in Z$ die Aufgabe (80) löst.

Der Algorithmus von Gilbert nähert sich der Lösung $\boldsymbol{z}^*$ von (80) iterativ unter Zuhilfenahme dieser Erkenntnisse nach dem Schema in Tabelle 3.

Abbildung 10 stellt den Start und zwei Iterationen des Verfahrens im $\mathbb{R}^2$ dar.

1.	Wähle $\boldsymbol{z} \in Z$ beliebig.
2.	Berechne $h_Z(-\boldsymbol{z})$ und $g_Z(-\boldsymbol{z}, \boldsymbol{z})$.
3.	Falls $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$, dann $\boldsymbol{z}^* = \boldsymbol{z}$, STOP. Sonst setze $\overline{\boldsymbol{z}} = s_Z(-\boldsymbol{z})$.
4.	Berechne $\widetilde{\boldsymbol{z}}$ als den Punkt auf $\overline{\boldsymbol{z}\overline{\boldsymbol{z}}}$, der die kleinste Norm besitzt.
5.	Setze $\boldsymbol{z} = \widetilde{\boldsymbol{z}}$ und gehe zu 2.

Tabelle 3: Iterativer Algorithmus von Gilbert.

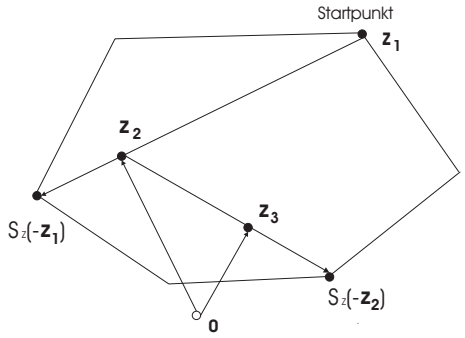


Abbildung 10: Zwei Iterationen nach Gilbert.

Der Algorithmus kann nur dann im Original verwendet werden, wenn die Merkmalsabbildung bekannt ist. Support-Vektor-Maschinen arbeiten jedoch mit einem Kern, der ohne Kenntnis des Merkmalsraumes gewählt wird. Deshalb ist es sinnvoll, alle $\boldsymbol{z} \in Z$ als konvexe Linearkombinationen der Elemente aus Z_0 darzustellen, das heißt:

$$\boldsymbol{z} := \sum_t \gamma_t \boldsymbol{z}_0^t, \quad \sum_t \gamma_t = 1, \quad \gamma_t \geq 0, \quad \boldsymbol{z}_0^t \in Z_0 \,, \tag{81}$$

wobei die Anzahl der Komponenten davon abhängt, wieviele Vertreter Z_0 hat, also wie sich die Trainingsdaten zusammensetzen. Die wesentliche Rolle im Algorithmus haben dann die Koeffizienten γ_t. Für weitere Details sei auf [18] verwiesen.

Der Algorithmus wird bei Annäherung an die Lösung sehr langsam. Den Grund dafür kann man sich an Abbildung 10 verdeutlichen. In jeder Iteration wird ein im Sinne der Aufgabe „guter“ Punkt $\boldsymbol{z}_0^{t'} = s_Z(-\boldsymbol{z})$ aus Z_0 bestimmt. Der zugehörige Koeffizient γ_t' in der Darstellung des aktuellen Punktes $\boldsymbol{z}$ wird vergrößert, betrachte zum Beispiel den Übergang von $\boldsymbol{z}_1$ nach $\boldsymbol{z}_2$ in Abbildung 10. Gleichzeitig werden alle anderen Koeffizienten in der Darstellung von $\boldsymbol{z}$ gleichmäßig verkleinert, unabhängig davon, ob sie zu relativ „guten“ oder „schlechten“ Eckpunkten gehören. Es wäre jedoch sinnvoller, wenn die Gewichte „schlechter“ Punkte schneller gegen Null gehen würden, als es hier der Fall ist. Die Methode nach Gilbert allein ist daher keine optimale Wahl für die Behandlung des Nearest-Point-Problems.

4.1.4 Algorithmus von Mitchell-Dem'yanov-Malozemov (MDM)

Der MDM-Algorithmus wurde erstmals im Jahr 1974 vorgestellt, siehe dazu [24]. Er löst ebenfalls die Aufgabe (80), wobei er die Darstellung (81) schon in seiner Ursprungsform voraussetzt.

Es kann Situationen geben, in denen der Algorithmus von Gilbert zu schlechter Performance führt. Man stelle sich vor, der aktuelle Punkt $\boldsymbol{z} \in Z$ liegt in der Nähe von $\boldsymbol{z}^* \in Z$. Trotzdem kann es $\gamma_t > 0$ geben, für die gilt

$$\langle \boldsymbol{z}, \boldsymbol{z}_0^t \rangle_F \gg \langle \boldsymbol{z}, \boldsymbol{z} \rangle_F \ . \tag{82}$$

Es ist einleuchtend, dass solche Koeffizienten γ_t den Punkt $\boldsymbol{z} \in Z$ nicht mitbestimmen, sondern eliminiert werden sollten. Dazu wird zunächst eine weitere Funktion eingeführt.

Die Funktion Δ über Z wird definiert als

$$\Delta(\boldsymbol{z}) := h_Z(-\boldsymbol{z}) - \underbrace{\min_{\gamma_t > 0} \langle -\boldsymbol{z}, \boldsymbol{z}_0^t \rangle_F}_{(*)} \ .$$

Der Punkt $\boldsymbol{z}_0^t$, der das Minimum $(*)$ realisiert, wird als $\boldsymbol{z}_0^{t_{min}}$ bezeichnet.

Es gelten $\Delta(\boldsymbol{z}) \geq 0$ auf ganz Z und $\Delta(\boldsymbol{z}) = 0$ genau dann wenn $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$. Aus diesem Grund ist es zulässig, $\Delta(\boldsymbol{z}) = 0$ anstelle von $g_Z(-\boldsymbol{z}, \boldsymbol{z}) = 0$ als Abbruchkriterium zu verwenden, siehe dazu Schritt drei im Algorithmus von Gilbert. Der MDM-Algorithmus bewegt sich ausgehend von einem $\boldsymbol{z}$ immer entlang der Richtung $s_Z(-\boldsymbol{z}) - \boldsymbol{z}_0^{t_{min}}$. Daraus leitet sich folgendes Prinzip ab: Für einen besonders „guten" Punkt $\boldsymbol{z}_0^{t'}$ aus Z_0 wird der Koeffizient in der Darstellung von $\boldsymbol{z}$ vergrößert, im Unterschied zu Gilbert aber auf Kosten eines einzigen „schlechten" Punktes $\boldsymbol{z}_0^{t_{min}}$. Die anderen Koeffizienten werden nicht angetastet. Somit ist es möglich, besonders unwichtige Punkte frühzeitig aus der Darstellung (81) zu eliminieren. Falls es also zu Phänomenen der Form (82) kommt, reagiert dieses Verfahren besser und führt schneller zu einem Abbruch, wie Tests im Zusammenhang mit der Arbeit [18] zeigten.

4.1.5 Nearest-Point-Algorithmus (NPA)

Auch wenn der MDM-Algorithmus in den meisten Fällen gute Ergebnisse liefert, ist es möglich, ihn zu verbessern. Das kann durch sinnvolle Kombinationen der beiden bisher vorgestellten Algorithmen erreicht werden.

Der Algorithmus vor Gilbert berechnet in seinen Iterationen hauptsächlich $h_Z(-\boldsymbol{z})$ und $s_Z(-\boldsymbol{z})$, siehe dazu nochmals Tabelle 3. Diese werden jedoch auch vom MDM-Algorithmus

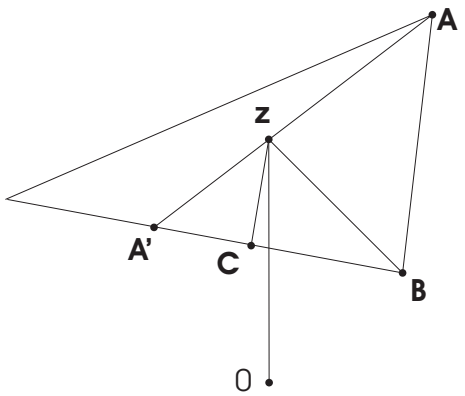


Abbildung 11: Mischalgorithmus für NPP.

zur Verfügung gestellt. Deshalb stellt es keinen Mehraufwand dar, die Ideen des Algorithmus von Gilbert im MDM-Algorithmus auszunutzen. Eine sehr günstige Variante soll noch kurz erklärt werden, siehe dazu Abbildung 11.

Die Bezeichnungen in Abbildung 11 haben folgende Bedeutungen:

- $\boldsymbol{A} := \boldsymbol{z}_0^{t_{min}}$ ist ein ungünstiger Punkt.
- $\boldsymbol{A'}$ realisiert die Verbindung von $\overline{\boldsymbol{Az}}$ mit dem Rand von Z.
- $\boldsymbol{B} := s_Z(-\boldsymbol{z})$ ist ein günstiger Punkt.
- $\boldsymbol{C}$ ist so gewählt, dass $\overline{\boldsymbol{zC}}$ parallel zu $\overline{\boldsymbol{AB}}$ ist und $\boldsymbol{C}$ auf der Strecke $\overline{\boldsymbol{A'B}}$ liegt.

Die Strecke $\overline{\boldsymbol{zB}}$ definiert den Bereich, auf dem Gilbert arbeitet, $\overline{\boldsymbol{zC}}$ hingegen jenen, auf dem MDM agiert. Eine Mischform wäre dann beispielsweise die Suche eines Punktes mit minimaler Norm im Dreieck $\boldsymbol{zCB}$. Falls der gefundene Punkt im Inneren dieses Dreiecks liegt, besagt diese Vorgehensweise:

- Der Koeffizient von $s_Z(-\boldsymbol{z})$ wird wie bei den ursprünglichen Verfahren vergrößert.
- Der Koeffizient von $\boldsymbol{z}_0^{t_{min}}$ wird deutlich stärker reduziert als bei Gilbert, aber er wird nicht so schnell eliminiert wie es beim MDM-Verfahren der Fall ist.
- Die restlichen Koeffizienten werden gleichmäßig reduziert und verzeichnen dadurch eine schwächere Abnahme als es beim Algorithmus von Gilbert der Fall ist. Sie werden jedoch im Gegensatz zum MDM-Algorithmus nicht völlig außer Acht gelassen.

Der in [18] vorgestellte Algorithmus geht sogar noch einen Schritt weiter. Er arbeitet auf dem Dreieck $\boldsymbol{zA'B}$. Ein durch eine solche Mischung entstandener Algorithmus ist schneller als alle in diesem Abschnitt vorgestellten Methoden. Details sollten in [18] nachgelesen werden.

Der Pseudocode in [18] sowie viele hilfreiche Anregungen von S. Sathiya Keerthi und Ricardo Henao trugen dazu bei, den NP-Algorithmus erfolgreich zu implementieren und einige Fehler der Originalarbeit zu korrigieren.

4.1.6 Gewichteter Nearest-Point-Algorithmus (GNPA)

Das allgemeine SVM-Modell unterscheidet grundsätzlich keine Fehlerarten. Klassifikationsfehler in den Trainingsdaten werden einfach oder quadratisch aufsummiert und mit dem Parameter C gewichtet. Damit gehen die Klasseninformationen komplett verloren. Ein Ausweg aus dieser Situation wird beispielweise in [26] vorgestellt.

Die Idee ist, Fehler erster Art härter zu bestrafen als Fehler zweiter Art. Das kann sehr einfach realisiert werden, indem der Parameter C auf zwei Parameter C^+ und C^- gesplittet wird. Bei einem Fehler erster Art wird mit C^+ gewichtet und bei einem Fehler zweiter Art mit C^-. Die Feststellung der Fehlerart ist trivial, da der Vektor $\boldsymbol{y}$ die Klassen jederzeit bereit hält.

Die Nutzung von C^+ und C^- führt im erfolgreichen Fall dazu, dass die optimale Hyperebene etwas näher an die Klasse -1 rückt (Abbildung 12). Anders ausgedrückt kann man sagen, dass sich die Gesamtmarge nicht mehr als Summe der beiden gleichen Einzelmargen m, sondern als $m_+ + m_-$ berechnet, wobei $m_+ > m_-$ gelten wird, falls $C^+ > C^-$. Der mit $\boldsymbol{x}$ gekennzeichnete Punkt in Abbildung 12 ist besonders interessant. Man sollte sich bewusst sein, dass bei sehr wenigen Punkten einer Klasse die Beeinflußung der Hypothese durch Ausreißer, beispielsweise durch falsche Angabe des Labels, viel höher ist als bei balancierten Daten, insbesondere wenn man mit einem großen Wert für C^+ arbeitet. Dieser führt dazu, dass um den Punkt $\boldsymbol{x}$ herum die Klasse 1 definiert wird. Ist dieser jedoch ein unerwünschter Ausreißer und gehört eigentlich in Klasse -1 oder als positiver Punkt in die Region der anderen Punkte der Klasse 1, ist die ursprüngliche Hyperebene die bessere.

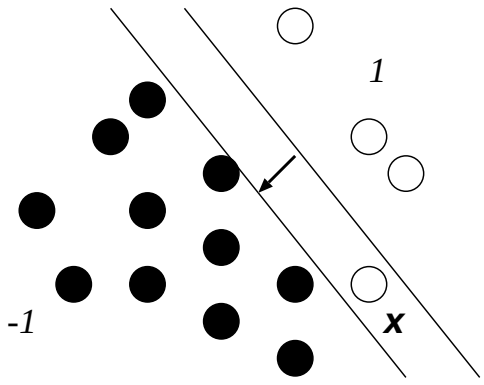


Abbildung 12: Verschiebung der Hyperebene durch C^-.

Die Konsequenzen daraus sind:

- Elemente des Klasse 1 werden verstärkt korrekt klassifiziert (true positive), auch wenn sie im Bereich der ursprünglichen Trennlinie liegen. Die Fehler erster Art nehmen ab.
- Elemente des Klasse -1 werden zunehmend falsch klassifiziert (false positive), wenn sie im Bereich der ursprünglichen Trennlinie liegen. Die Fehler zweiter Art nehmen zu.

Diesen Effekt kann man sich anhand der Abbildung 13 verdeutlichen, wobei sich die Trennfunktion auf der rechten Seite der Abbildung in Richtung der negativen Klasse verschoben hat.

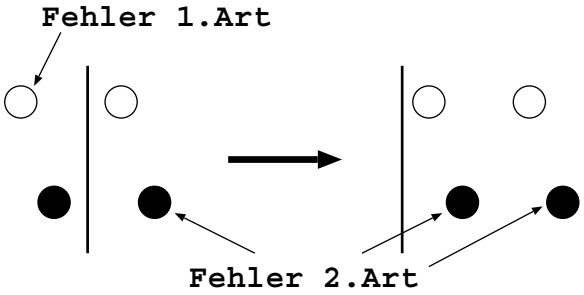


Abbildung 13: Fehler erster und zweiter Art in Konkurrenz.

Die Zielfunktion der ursprünglichen primalen Optimierungsaufgabe einer $L2$-Norm Support-Vektor-Maschine ändert sich wie folgt:

$$\min_{\boldsymbol{w}\in\mathcal{F},b\in\mathbb{R},\boldsymbol{\xi}\in\mathbb{R}^l} \quad \frac{1}{2}\|\boldsymbol{w}\|_{\mathcal{F}}^2 + C^+ \sum_{i:y_i=1} \xi_i^2 + C^- \sum_{j:y_j=-1} \xi_j^2 \,. \tag{83}$$

Die Nebenbedingungen bleiben identisch. Bei der Summierung der Fehler wird unterschieden, ob ein falsch negativer (erste Summe) oder falsch positiver Punkt (zweite Summe) betroffen ist.

Die Integration eines zweiten Fehlergewichtes war in diesem Fall sehr einfach. In der ursprünglichen dualen Optimierungsaufgabe (77) wird der Parameter C ausschließlich für den modifizierten Kern $\tilde{K}$, siehe (78), verwendet. Wir änderten diesen Kern wie in Abbildung 14 dargestellt. Weitere Änderungen waren nicht notwendig.

Die neue Zielfunktion der dualen Aufgabe hat dann wegen (41), (78) und den hier betrachteten Gewichten die Form

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^l} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \left[K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \delta_{i,j} \left(\frac{\delta_{y_i,1}}{2C^+} + \frac{\delta_{y_i,-1}}{2C^-} \right) \right] \tag{84}$$

```
function kernel(i, j)
    kernel = compute-normal-kernel(i, j)
    if (i == j) then
        if (y_i == 1) kernel = kernel + 1/2C^+
        if (y_i == −1) kernel = kernel + 1/2C^−
    end if
end function kernel
```

Abbildung 14: Anpassung eines Kerns für das $L2$-Norm-Modell mit Fehlergewichtung.

unter den Nebenbedingungen

$$\begin{aligned} \boldsymbol{\alpha}^T \boldsymbol{y} &= 0\,, \\ \boldsymbol{\alpha} &\geq \mathbf{0}\,. \end{aligned}$$

Die Nutzung dieser Gewichte ist für alle Kerne möglich, denn die ursprüngliche Kernfunktion wird nicht modifiziert, es werden lediglich die Ausgaben für Diagonalelemente der Kernmatrix geändert.

4.2 Sequential-Minimal-Optimization

Der Algorithmus Sequential-Minimal-Optimization (SMO) wurde 1998 von J. Platt entwickelt und vorgestellt [29]. Der SMO-Algorithmus gehört zu den sogenannten working set-Verfahren, welche die $L1$-Norm-Aufgabe (75) lösen. Diese Methoden verfahren nach dem Prinzip der iterativen Optimierung auf Teilmengen. Dazu wird in jeder Iteration ein reduziertes Optimierungsproblem gelöst. Dabei wird der Vektor $\boldsymbol{\alpha}$ in einen aktiven und einen nichtaktiven Teil zerlegt. Die Optimierungsarbeit findet dann nur auf dem aktiven Teil des Vektors (Arbeitsmenge, working set, chunk) statt. Die aktiven und inaktiven Indizes müssen nach jedem Schritt aktualisiert werden.

Das besondere am SMO-Algorithmus ist die Einschränkung der Größe des working set auf zwei Elemente. Im Folgenden werden wir zeigen, welche Aufgabe in jedem Teilschritt des Verfahrens gelöst werden muss und nach welcher Methode die globale Lösung generiert wird.

Die Matrix $\boldsymbol{Q}$ sei definiert als

$$Q_{ij} := y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) y_j \quad (1 \leq i, j \leq l)\,,$$

dann kann die Zielfunktion (75) kurz als

$$W(\boldsymbol{\alpha}) := \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Q}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{1} \tag{85}$$

angegeben werden.

4.2.1 Teiloptimierung

Wie schon erwähnt, wird die Aufgabe (85) iterativ auf einem Teil von $\boldsymbol{\alpha}$ gelöst. Wir definieren dazu eine Zerlegung von $\boldsymbol{\alpha}$, $\boldsymbol{y}$ und $\boldsymbol{Q}$:

$$\boldsymbol{\alpha} := \begin{pmatrix} \boldsymbol{\alpha}_{\mathrm{d}} \\ \boldsymbol{\alpha}_{\mathrm{f}} \end{pmatrix},$$

$$\boldsymbol{y} := \begin{pmatrix} \boldsymbol{y}_{\mathrm{d}} \\ \boldsymbol{y}_{\mathrm{f}} \end{pmatrix},$$

$$\boldsymbol{Q} := \begin{pmatrix} \boldsymbol{Q}_{\mathrm{dd}} & \boldsymbol{Q}_{\mathrm{df}} \\ \boldsymbol{Q}_{\mathrm{fd}} & \boldsymbol{Q}_{\mathrm{ff}} \end{pmatrix}.$$

Mit d bezeichnen wir jeweils den in der aktuellen Iteration dynamischen Teil von $\boldsymbol{\alpha}$. O.B.d.A. liegen die Daten jeweils sortiert nach dynamischen und festen Teilen f vor. Der Iterationsindex ist aus Gründen der Übersichtlichkeit nicht angegeben.

Diese Zerlegung kann in die Zielfunktion übernommen werden

$$W(\boldsymbol{\alpha}) = \frac{1}{2}\left(\boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{\mathrm{dd}}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{fd}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{df}\boldsymbol{\alpha}_{\mathrm{f}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{\mathrm{ff}}\boldsymbol{\alpha}_{\mathrm{f}}\right) - \boldsymbol{\alpha}_{\mathrm{d}}^T \mathbf{1} - \boldsymbol{\alpha}_{\mathrm{f}}^T \mathbf{1}. \tag{86}$$

Da der Vektor $\boldsymbol{\alpha}_{\mathrm{f}}$ für die spezielle Iteration konstant ist, können die Terme $\frac{1}{2}\boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{\mathrm{ff}}\boldsymbol{\alpha}_{\mathrm{f}}$ und $\boldsymbol{\alpha}_{\mathrm{f}}^T \mathbf{1}$ entfallen und wir erhalten eine neue Funktion von $\boldsymbol{\alpha}_{\mathrm{d}}$. Diese hat die Form

$$W_{\mathrm{d}}(\boldsymbol{\alpha}_{\mathrm{d}}) = \frac{1}{2}\boldsymbol{\alpha}_{\mathrm{d}}^T \boldsymbol{Q}_{\mathrm{dd}}\boldsymbol{\alpha}_{\mathrm{d}} + \boldsymbol{\alpha}_{\mathrm{f}}^T \boldsymbol{Q}_{fd}\boldsymbol{\alpha}_{\mathrm{d}} - \boldsymbol{\alpha}_{\mathrm{d}}^T \mathbf{1}\,. \tag{87}$$

Die angepassten Nebenbedingungen lauten

$$\begin{aligned} \boldsymbol{y}_{\mathrm{d}}^T \boldsymbol{\alpha}_{\mathrm{d}} &= -\boldsymbol{y}_{\mathrm{f}}^T \boldsymbol{\alpha}_{\mathrm{f}}\,, \\ \boldsymbol{\alpha}_{\mathrm{d}} &\geq \mathbf{0}\,, \\ \boldsymbol{\alpha}_{\mathrm{d}} &\leq \boldsymbol{C}\,. \end{aligned}$$

Sie sichern, dass

$$\boldsymbol{\alpha}^{k+1} := \begin{pmatrix} \boldsymbol{\alpha}_{\mathrm{d}}^{k+1} \\ \boldsymbol{\alpha}_{\mathrm{f}}^{k} \end{pmatrix}$$

in jeder Iteration zulässig ist.

Für Sequential-Minimal-Optimization gilt: $|\boldsymbol{\alpha}_\mathrm{d}| = 2$. Das ist die minimale Größe, die ein working set haben darf. Das kann sehr einfach erklärt werden: Im k-ten Optimierungsschritt muss zur Einhaltung der Nebenbedingungen gelten

$$\boldsymbol{y}_\mathrm{d}^T \boldsymbol{\alpha}_\mathrm{d}^k = \boldsymbol{y}_\mathrm{d}^T \boldsymbol{\alpha}_\mathrm{d}^{k+1} = c \quad (c \in \mathbb{R} \text{ konstant}) \,. \tag{88}$$

Sollte $\boldsymbol{\alpha}_\mathrm{d}^k$ nur ein Element haben, führt jede Änderung dieses Multiplikators zu einer Verletzung dieser Bedingung. Eine Optimierung auf zwei Elementen dagegen ist möglich. Der Vorteil von SMO gegenüber anderen working set-Methoden mit großen Arbeitsmengen liegt in der Tatsache, dass das Teilproblem mit der Zielfunktion (87) bei zwei Punkten analytisch gelöst werden kann. Das soll im Folgenden gezeigt werden, wobei auch die entsprechenden Lösungen mit angegeben werden.

Wir bezeichnen mit i und j die in der k-ten Iteration betrachteten Indizes von $\boldsymbol{\alpha}$. Aus der Bedingung (88) ergibt sich

$$\begin{aligned} y_i\alpha_i^k + y_j\alpha_j^k = y_i\alpha_i^{k+1} + y_j\alpha_j^{k+1} &= c \quad | \cdot y_i \\ \alpha_i^k + s(i,j)\alpha_j^k = \alpha_i^{k+1} + s(i,j)\alpha_j^{k+1} &= \tilde{c}\,, \end{aligned} \tag{89}$$

wobei $s(i,j) = y_i \cdot y_j$ und $\tilde{c} \in \mathbb{R}$ nicht von α_i und α_j abhängen. Außerdem muss

$$0 \leq \alpha_i^k, \alpha_j^k, \alpha_i^{k+1}, \alpha_j^{k+1} \leq C \,. \tag{90}$$

auch weiterhin gelten.

Die Optimierung verläuft sehr einfach. O.B.d.A. soll α_j zuerst berechnet werden. Aus (89) und (90) folgen

1. für $y_i = y_j$:

$$\alpha_i^k + \alpha_j^k = \alpha_i^{k+1} + \alpha_j^{k+1} \;\Rightarrow\; \alpha_j^{k+1} = \alpha_i^k + \alpha_j^k - \alpha_i^{k+1}$$

$$\Rightarrow \alpha_j^{k+1} \in \Big[\max\big\{0, \alpha_i^k + \alpha_j^k - C\big\},\ \min\big\{C, \alpha_i^k + \alpha_j^k\big\} \Big]$$

und

2. für $y_i \neq y_j$:

$$\alpha_i^k - \alpha_j^k = \alpha_i^{k+1} - \alpha_j^{k+1} \;\Rightarrow\; \alpha_j^{k+1} = \alpha_i^{k+1} + \alpha_i^k - \alpha_j^k$$

$$\Rightarrow \alpha_j^{k+1} \in \Big[\max\big\{0, \alpha_j^k - \alpha_i^k\big\},\ \min\big\{C, C - \alpha_i^k + \alpha_j^k\big\} \Big] \,.$$

Bezeichnet man jeweils mit UG die untere und mit OG die obere Grenze, kann man das Ergebnis symbolisch darstellen als

$$\alpha_j \in [\mathrm{UG}, \mathrm{OG}] \tag{91}$$

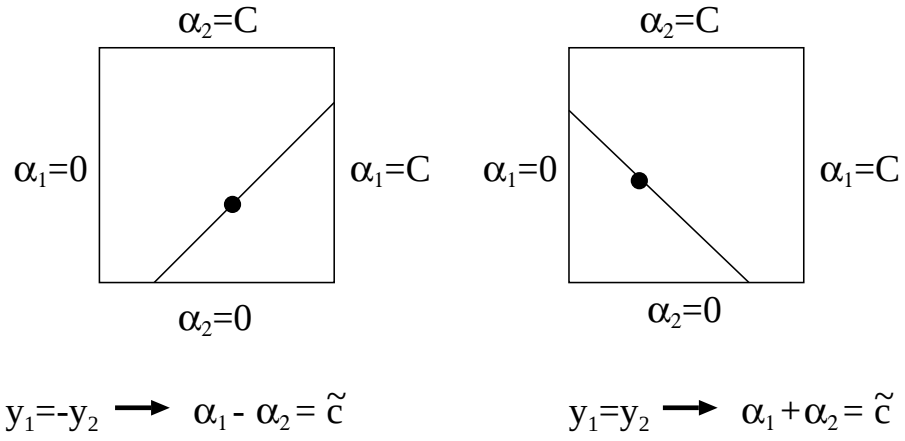


Abbildung 15: Boxen für zwei Lagrange-Multiplikatoren nach SMO.

und zeigt damit, dass der zweite Lagrange-Multiplikator immer zwischen bestimmten Grenzen liegt. Abbildung 15 stellt alle Bedingungen graphisch dar.

Mit $f(\boldsymbol{x})$ sei wie üblich der Wert der Klassifikationsfunktion bezeichnet, der sich bei der Auswertung an der Stelle $\boldsymbol{x}$ mit den aktuellen Parametern $\boldsymbol{\alpha}$ und b sowie einem fest gewählten Kern ergibt, also

$$f(\boldsymbol{x}) = \sum_{t=1}^{l} \alpha_t y_t K(\boldsymbol{x}^t, \boldsymbol{x}) + b. \tag{92}$$

Das bedeutet, der Wert für b muss ebenfalls immer wieder verändert werden, die Formeln dafür folgen später.

E_i symbolisiert die Differenz zwischen dem aktuellen Wert der Klassifikationsfunktion und der vorgegebenen Ausgabe im Punkt $\boldsymbol{x}^i$, also

$$E_i = f(\boldsymbol{x}^i) - y_i = \sum_{t=1}^{l} \alpha_t y_t K(\boldsymbol{x}^t, \boldsymbol{x}^i) + b - y_i \; .$$

Dieser Wert kann selbst bei richtiger Zuordnung von $\boldsymbol{x}^i$ groß werden.

Die α_t für $t \notin \{i, j\}$ sind in jeder Iteration konstant. Die Zielfunktion der dualen Aufgabe (85) kann man daher umformulieren zu

$$\begin{aligned} W(\alpha_i, \alpha_j) \;\; = \;\; & 0.5\alpha_i^2 K(\boldsymbol{x}^i, \boldsymbol{x}^i) + 0.5\alpha_j^2 K(\boldsymbol{x}^j, \boldsymbol{x}^j) + \alpha_i \alpha_j s(i,j) K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\ & + \sum_{t \notin \{i,j\}} \alpha_i \alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_j \alpha_t y_j y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) - \alpha_i - \alpha_j + c \; , \end{aligned}$$

wobei $c \in \mathbb{R}$ nicht von α_i und α_j abhängt, also konstant ist.

Ersetzt man α_i in der Zielfunktion gemäß (89) durch $\tilde{c} - s(i,j)\alpha_j$, ergibt sich eine Funktion von α_j:

$$\begin{aligned} W(\alpha_j) &= 0.5(\tilde{c} - s(i,j)\alpha_j)^2 K(\boldsymbol{x}^i, \boldsymbol{x}^i) + 0.5\alpha_j^2 K(\boldsymbol{x}^j, \boldsymbol{x}^j) \\ &\quad + s(i,j)(\tilde{c} - s(i,j)\alpha_j)\alpha_j K(\boldsymbol{x}^i, \boldsymbol{x}^j) + \sum_{t \notin \{i,j\}} (\tilde{c} - s(i,j)\alpha_j)\alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) \\ &\quad + \sum_{t \notin \{i,j\}} \alpha_j \alpha_t y_j y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) - \tilde{c} + s(i,j)\alpha_j - \alpha_j + c\,. \end{aligned}$$

Die notwendige Bedingung an einen stationären Punkt von W lautet

$$\begin{aligned} \frac{\partial W(\alpha_j)}{\partial \alpha_j} &= -s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^i)(\tilde{c} - s(i,j)\alpha_j) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)\alpha_j \\ &\quad + s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^j)\tilde{c} - 2K(\boldsymbol{x}^i, \boldsymbol{x}^j)\alpha_j - s(i,j) \sum_{t \notin \{i,j\}} \alpha_t y_i y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) \\ &\quad + y_j \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) + s(i,j) - 1 = 0\,. \end{aligned}$$

Deshalb und wegen $s(i,j)y_i = y_j$ muss für einen stationären Wert α_j^{st} von α_j die Gleichung

$$\begin{aligned} 0 &= s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^j)\tilde{c} - y_j\Bigg(\sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t)\Bigg) \\ &\quad - s(i,j)K(\boldsymbol{x}^i, \boldsymbol{x}^i)(\tilde{c} - s(i,j)\alpha_j^{st}) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)\alpha_j^{st} - 2K(\boldsymbol{x}^i, \boldsymbol{x}^j)\alpha_j^{st} \qquad (93) \\ &\quad - 1 + s(i,j) \end{aligned}$$

erfüllt sein.

Durch nochmaliges Differenzieren erkennt man, dass α_j^{st} Maximalstelle ist, falls

$$\frac{\partial^2 W(\alpha_j^{st})}{\partial^2 \alpha_j^{st}} = 2K(\boldsymbol{x}^i, \boldsymbol{x}^j) - K(\boldsymbol{x}^i, \boldsymbol{x}^i) - K(\boldsymbol{x}^j, \boldsymbol{x}^j) < 0$$

gilt.

Sei $\eta = -2K(\boldsymbol{x}^i, \boldsymbol{x}^j) + K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^j, \boldsymbol{x}^j)$. Einsetzen in (93) und Nutzung von $s(i,j) \in \{-1, 1\}$ liefert

$$\begin{aligned} -\alpha_2^{st} \cdot \eta &= -1 + s(i,j) - s(i,j) \cdot \tilde{c}(K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^i, \boldsymbol{x}^j)) \\ &\quad - y_j\Bigg(\sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) - \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t)\Bigg)\,. \end{aligned}$$

Multiplikation mit y_j ergibt

$$\begin{aligned}
-\alpha_j^{st} \cdot \eta \cdot y_j &= -y_j + y_i - y_i \cdot \tilde{c}(K(\boldsymbol{x}^i, \boldsymbol{x}^i) + K(\boldsymbol{x}^i, \boldsymbol{x}^j)) \\
&\quad - \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^i, \boldsymbol{x}^t) + \sum_{t \notin \{i,j\}} \alpha_t y_t K(\boldsymbol{x}^j, \boldsymbol{x}^t) \\
&\overset{(92)}{=} -y_j + y_i + \tilde{c} \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^i) + \tilde{c} \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad + y_i \alpha_i^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) + f(\boldsymbol{x}^i) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad - f(\boldsymbol{x}^j) + y_i \alpha_i^k K(\boldsymbol{x}^j, \boldsymbol{x}^i) + y_j \alpha_j^k K(\boldsymbol{x}^j, \boldsymbol{x}^j) \; .
\end{aligned}$$

Elimination von $\tilde{c}$ durch Nutzung von (89) führt zur Darstellung

$$\begin{aligned}
-\alpha_j^{st} \cdot \eta \cdot y_j &= y_j - y_i + (\alpha_i^k + s(i,j) \cdot \alpha_j^k) \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) \\
&\quad - (\alpha_i^a + s(i,j) \cdot \alpha_j^k) \cdot y_i K(\boldsymbol{x}^i, \boldsymbol{x}^j) + f(\boldsymbol{x}^i) \\
&\quad - y_i \alpha_i^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) - f(\boldsymbol{x}^j) \\
&\quad + y_i \alpha_i^k K(\boldsymbol{x}^j, \boldsymbol{x}^i) + y_j \alpha_j^k K(\boldsymbol{x}^k, \boldsymbol{x}^k) \\
&= y_j - y_i + f(\boldsymbol{x}^i) - f(\boldsymbol{x}^j) + y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^i) \\
&\quad - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) - y_j \alpha_j^k K(\boldsymbol{x}^i, \boldsymbol{x}^j) + y_j \alpha_j^k K(\boldsymbol{x}^j, \boldsymbol{x}^j) \\
&= -\alpha_j^k \eta y_j + \big(f(\boldsymbol{x}^i) - y_i\big) - \big(f(\boldsymbol{x}^j) - y_j\big)
\end{aligned}$$

und damit

$$\alpha_j^{st} = \alpha_j^k - \frac{y_j(E_i - E_j)}{\eta} \; .$$

Der Wert für α_j^{k+1} ergibt sich dann aus α_j^{st} oder aus einer der Grenzen aus (91), falls diese von α_j^{st} unter- oder überschritten werden. Die anschließende Berechnung von α_i^{k+1} ist trivial.

Der maximale Wert der Zielfunktion (85) unter der Bedingung, dass nur α_i und α_j geändert werden dürfen, wird also für die Werte

$$\begin{aligned}
\alpha_j^{k+1} &= \max\Big\{\mathrm{UG},\ \min\big\{\mathrm{OG}, \alpha_j^k - y_j(E_i - E_j)/\eta\big\}\Big\} \qquad (94) \\
&= \max\Big\{\max\big\{0, \alpha_j^k - \alpha_i^k\big\},\ \min\big\{C, C - \alpha_i^k + \alpha_j^k, \alpha_j^k - y_j(E_i - E_j)/\eta\big\}\Big\}
\end{aligned}$$

und

$$\alpha_i^{k+1} = \alpha_i^k + y_i y_j (\alpha_j^k - \alpha_j^{k+1})$$

angenommen, falls

$$\eta := -2K(\boldsymbol{x}^1, \boldsymbol{x}^2) + K(\boldsymbol{x}^1, \boldsymbol{x}^1) + K(\boldsymbol{x}^2, \boldsymbol{x}^2)$$

strikt positiv ist.

Wie auch in [29] nachzulesen ist, entsteht durch die Forderung $\eta > 0$ keine besondere Problematik. Folgende Fälle sollte man aber ausschließen:

(1) Erfüllt der Kern K nicht die Bedingungen des Theorems von Mercer, kann $\eta < 0$ auftreten.

(2) Sind zwei Eingabevektoren identisch, wird $\eta = 0$ auftreten, sobald diese beiden ausgewählt werden.

4.2.2 Heuristiken

Bisher wurde hergeleitet, wie zwei ausgewählte Lagrange-Multiplikatoren zu optimieren sind. Es bleibt zu klären, wie man in jedem Iterationsschritt des Verfahrens zwei passende Indizes auswählt und welche Größen zusätzlich angepasst werden müssen.

Die Konvergenzgeschwindigkeit des Verfahrens erhöht sich, wenn diejenigen Indizes i und j gewählt werden, welche die KKT-Bedingungen am stärksten verletzen. Diese Bedingungen für alle Punkte auszuwerten erfordert einen zu großen Rechenaufwand. Deshalb werden Heuristiken herangezogen, die die KKT-Bedingungen nur für einen Teil der Daten überprüfen und so den Gesamtaufwand reduzieren. Die Auswahl dieser aktiven Punkte erfolgt derart, dass die Zielfunktion einen möglichst großen Zuwachs erfährt.

- Heuristik für i: Dieser Index wird aus der Menge der Punkte ermittelt, welche die KKT-Bedingungen verletzen. Dafür wird eine Fehlertoleranz ε benutzt. Der Wert für ε sollte nach [29] zwischen 10^{-2} und 10^{-3} liegen. Dabei wird zunächst nicht darauf geachtet, in welchem Intervall die zugehörigen Lagrange-Multiplikatoren liegen, sondern der erste gefundene Punkt wird ausgewählt. Im Anschluss bestimmt die zweite Heuristik - siehe nächster Punkt - den anderen an der Optimierung beteiligten Index und ein Iterationsschritt kann durchgeführt werden. In den weiteren Iterationen sucht diese Heuristik nur noch nach Punkten, welche die KKT-Bedingungen verletzen **und** deren Lagrange-Multiplikatoren im offenen Intervall $(0, C)$ liegen, bis der erste gefunden ist. Erst wenn keine solchen Punkte mehr vorhanden sind, wird die Suche wieder auf alle Werte für α_i ausgedehnt.
- Heuristik für j: Der Index muss so gewählt werden, dass der während der Optimierung durchgeführte Schritt möglichst groß ist. Man betrachte nochmals die Darstellung (94). Offensichtlich ist die Änderung des zweiten Lagrange-Multiplikators von

η sowie von $|E_i - E_j|$ abhängig. Die Berechnung von η ist wegen der drei Kernauswertungen sehr zeitintensiv. Deshalb sollte man j so wählen, dass es den Ausdruck $|E_i - E_j|$ maximiert. Der Vorteil dieser Wahl ist, dass der Vektor $\boldsymbol{E}$ im Gegensatz zur Kernmatrix während der Laufzeit mitgeführt wird. Damit kann viel Rechenzeit eingespart werden.

Unter ungünstigen Voraussetzungen kann es dazu kommen, dass diese Heuristik keine Verbesserung liefert [29]. In diesen Fällen benutzt SMO bis zu zwei weitere Heuristiken. Zunächst werden alle Indizes $j \neq i$, deren Lagrange-Multiplikatoren α_j im Intervall $(0, C)$ liegen, der Reihe nach untersucht, wobei bei einem zufällig bestimmten Punkt angefangen wird. Liefert auch diese Suche keine Verbesserung, werden im letzten Schritt alle Punkte überprüft, also auch die Grenzfälle $\alpha_j = 0$ und $\alpha_j = C$, wobei auch hier bei einem zufälligen Punkt gestartet wird. Liefern diese drei Möglichkeiten zu einem bestimmten i kein passendes j, wird keine Optimierung und Aktualisierung durchgeführt, sondern SMO wählt ein neues i. Laut [29] tritt dieser Fall aber sehr selten auf.

4.2.3 Aktualisierungen

Der Wert von b muss nach jedem Iterationsschritt aktualisiert werden, denn die beiden betrachteten Punkte müssen nach dem Optimierungsschritt die KKT-Bedingungen erfüllen. Zu Beginn des Verfahrens gilt $b = 0$.

Zunächst betrachte man α_i:

$$\alpha_i^{k+1} \in (0, C) \qquad \Rightarrow f(\boldsymbol{x}^i) \stackrel{!}{=} y_i \; . \tag{95}$$

In diesem Fall ergibt sich ein b_1 aus

$$-b_1 = E_i^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^i) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^i, \boldsymbol{x}^j) - b^k \; .$$

Analog wird α_j ausgewertet:

$$\alpha_j^{k+1} \in (0, C) \qquad \Rightarrow f(\boldsymbol{x}^j) \stackrel{!}{=} y_j \; . \tag{96}$$

In diesem Fall ergibt sich b_2 aus

$$-b_2 = E_j^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^j) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^j, \boldsymbol{x}^j) - b^k \; .$$

Falls nur eine der Bedingungen erfüllt ist, wird $b^{k+1} = b_1$ (95), beziehungsweise $b^{k+1} = b_2$ (96) gesetzt. Falls beide erfüllt sind, gilt $b^{k+1} = b_1 = b_2$. Für den Fall, dass beide Lagrange-Multiplikatoren einen Grenzfall annehmen, also

$$\alpha_i^{k+1} \in \{0, C\} \text{ und } \alpha_j^{k+1} \in \{0, C\} \; ,$$

stellen alle Werte zwischen b_1 und b_2 zulässige Werte für b^{k+1} dar. Das Verfahren SMO wählt in diesem Fall die Mitte des Intervalls.

Der Fehlervektor $\boldsymbol{E}$ wird während der gesamten Laufzeit mitgeführt und muss für alle Indizes t $(t \neq i, j)$ mit $0 < \alpha_t < C$ angepasst werden durch

$$E_t^{k+1} = E_t^k + y_i(\alpha_i^{k+1} - \alpha_i^k)K(\boldsymbol{x}^i, \boldsymbol{x}^t) + y_j(\alpha_j^{k+1} - \alpha_j^k)K(\boldsymbol{x}^j, \boldsymbol{x}^t) - b^k + b^{k+1} \ .$$

Der SMO-Algorithmus nach J. Platt wurde im Rahmen des GALA Projektes implementiert. Der Pseudocode in [29] war dazu äußerst hilfreich. John Platt weist auf seiner Homepage auf einige Fehler im Paper hin. Seine Hinweise wurden berücksichtigt.

4.2.4 Gewichteter SMO-Algorithmus (GSMO)

Analog zum Algorithmus GNPA, der auf Seite 159 vorgestellt wurde, kann auch das SMO-Verfahren in einer gewichteten Variante definiert werden. Dabei ändert sich bei der Aufgabe (75) lediglich die dritte Nebenbedingung. Sie lautet dann

$$\forall i = 1, \ldots, l: \quad \alpha_i \leq \begin{cases} C^+ & \text{falls } y_i = 1 \ , \\ C^- & \text{sonst} \ . \end{cases} \tag{97}$$

Bei der Implementierung der gewichteten Variante von SMO muss an den jeweiligen Stellen im Quellcode, an denen der Parameter C verwendet wird, in Abhängigkeit der Klassenzugehörigkeit des betrachteten Trainingspunktes entweder C^+ oder C^- gewählt werden. Eine einfache Änderung der Kernfunktion analog zu der Vorgehensweise bei GNPA, siehe Abbildung 14, ist leider nicht möglich, da der Parameter C nicht in die Kernfunktion integriert werden kann. Details zum Quellcode von SMO können in [5] nachgelesen werden.

4.3 Zusammenfassung

In diesem Abschnitt sind die implementierten Verfahren zum Trainieren von Support-Vektor-Maschinen vorgestellt worden. Grob gesagt, stehen für die Algorithmen zwei unterschiedliche Modelle zur Verfügung – $L1$-Norm und $L2$-Norm. Das hard margin Modell kann über $C = \infty$ sowohl mit NPA als auch mit SMO trainiert werden. Fraglich für die Anwendung ist, welcher Algorithmus erfolgreicher arbeitet. Diese Frage läßt sich allgemein nicht beantworten. $L1$-Norm- und $L2$-Norm-Modell unterscheiden sich in der Formulierung der primalen Optimierungsaufgabe ausschließlich in der Art der Summierung der Schlupfvariablen ξ_i $(i = 1, ..., l)$. Für das $L2$-Norm-Modell werden diese Werte quadriert. Bei zahlreichen Tests hat sich gezeigt, dass die mittels NPA und SMO generierten Ergebnisse sehr ähnlich sind, wobei der Nearest-Point-Algorithmus weniger Rechenzeit in Anspruch nimmt. Im Gegenzug wirkt sich das $L1$-Norm-Modell positiver auf die Anzahl

der Support-Vektoren aus, ihre Anzahl ist geringer als mit NPA. Zusätzlich zu den Standardalgorithmen haben wir die Möglichkeit mittels C^+ und C^- Fehler zu gewichten und somit falsch positive oder falsch negative Klassifikationen zu verringern.

5 Optimierung

Die Algorithmen aus den Abschnitten 4.1 und 4.2 allein reichen nicht aus, um Support-Vektor-Maschinen erfolgreich zu trainieren. Für die Untersuchung von großen empirischen Datensätzen ist es außerdem erforderlich, hinreichend gutartige Trainingsdaten zu verwenden sowie ein geeignetes Modell mit optimalen Parameterwerten festzulegen. In den folgenden beiden Abschnitten werden wir kurz auf diese Aufgaben eingehen.

5.1 Datenvorverarbeitung

Datensäuberung und -reduktion als Bestandteile des Data Mining sind sehr wichtig als vorverarbeitende Schritte bei der Klassifikation von Daten. Diese Abhängigkeit wird oft unterschätzt. Auch das beste Klassifikationsverfahren erzielt schlechte Ergebnisse, wenn sich in den Daten zu viele Ausreißer und Fehler unerkannt verbergen.

Zur Datensäuberung zählen beispielsweise

- Entfernen von offensichtlichen Fehlern, z.B. nichtreelle Einträge,
- Ersetzen von Missing Values durch „sinnvolle“ Werte,
- Standardisierung oder Transformation von Variablen,
- Erkennen und Entfernen von Variablen ohne Varianz.

Diese Aufgaben werden durch die in Teil II dieses Bandes vorgestellten Methoden abgedeckt. Datensäuberung führt dazu, dass das SVM-Training einfacher und zuverlässiger wird; das konnte in verschiedenen Tests gezeigt werden.

Zusätzlich zur Säuberung fehlerhafter Daten beschäftigen wir uns mit Methoden zur Datenreduktion. Durch eine Verkleinerung des Datenraumes können beispielsweise die Rechenzeiten von Klassifikationsverfahren reduziert werden. Im Allgemeinen bezweckt man damit jedoch bessere Ergebnisse und Interpretationsmöglichkeiten der Klassifikationen.

Wir nutzen Methoden der Datenreduktion für die beiden Aufgaben

- Erkennen und Entfernen von unerwünschten Ausreißern und
- Entfernen unwichtiger Variablen.

Im Teil II dieses Berichtes sind diese Methoden detailliert beschrieben.

Beim Zusammenspiel von Datenreduktion und überwachtem Lernen mit Support-Vektor-Maschinen oder anderen Klassifikationsverfahren gibt es ein wichtiges Detail, welches bisher nicht erwähnt wurde. Das Identifizieren von unerwünschten Messungen und überflüssigen Variablen beruht auf statistischen Annahmen, die völlig frei von Informationen zur Verteilung der Klassen sind. Diese Methoden zählen dann zu den sogenannten unüberwachten Verfahren. Eine starke Datenreduktion ohne Betrachtung der Klassen kann dabei durchaus zum Verlust signifikanter Abhängigkeiten und nichtlinearer Funktionalitäten führen. Einen Ausweg aus diesem Dilemma stellt eine getrennte Untersuchung der beiden Klassen dar. Eine weitere Aufgabe innerhalb des GALA-Projektes bestand in der Untersuchung und Umsetzung von überwachten Methoden zur Variablenselektion. Diese werden in diesem Band jedoch nicht näher beschrieben. Einen kleinen Einblick zu dieser Arbeit gibt [19].

5.2 Modelloptimierung

Für das Training einer Support-Vektor-Maschine müssen neben der Wahl eines geeigneten Kerns verschiedene Parameterwerte an die jeweiligen Daten angepasst werden. Über diese Parameter und deren Funktion ist schon ausführlich in den Abschnitten 3.2.3 und 3.3 diskutiert worden. Die Generalisierungsfähigkeit einer SVM, d.h. die Qualität der Klassifikation, hängt von den gewählten Parameterwerten ab [25]. Ungünstige Parameter führen zum Scheitern einer Support-Vektor-Maschine, und, je besser man die Parameterwerte festlegt, umso verlässlicher ist auch die erlernte Klassifikationsfunktion. Das Problem der Parameteroptimierung für Support-Vektor-Maschinen ist ein in den letzten Jahren stark diskutiertes Thema [17, 25, 3, 14], wird aber dennoch zu oft unterschätzt. Oft werden Ergebnisse von neu entwickelten Methoden mit Ergebnissen von SVM-Software ohne Parametertuning verglichen [1].

Wir führen Parameteroptimierung auf der Grundlage der während einer Kreuzvalidierung entstandenen Fehler durch. Der Trainingsdatensatz S_l wird dafür in $p,\ 1 < p \leq l$, Teile möglichst gleicher Größe aufgeteilt. Der Testdatensatz wird vorerst nicht betrachtet. Abbildung 16 stellt die von uns implementierte Form der zehnfachen Kreuzvalidierung dar. Zusätzlich haben wir für moderate Datensätze die sogenannte leave one out Kreuzvalidierung $(p = l)$ implementiert.

Für eine Parameteroptimierung werden iterativ Kreuzvalidierungen mit verschiedenen Parameterwerten durchgeführt, siehe dazu Tabelle 4.

An dieser Stelle bleibt noch zu klären, welches Maß darüber entscheidet, ob ein Ergebnis zufriedenstellend ist, und wie die neu zu testenden Parameterwerte definiert werden (Schritt 5).

Wir stellen nun das implementierte Gütemaß vor. Dazu müssen zunächst einige grundlegende Kennzahlen definiert werden.

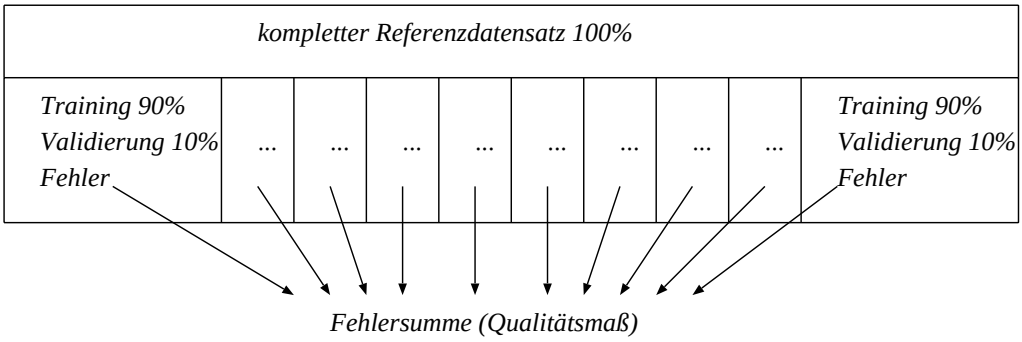


Abbildung 16: Zehnfache Kreuzvalidierung.

1.	Wähle Startwerte für Parameter, z.B. $\sigma = 1$ und $C = 1$; $i = 1$
2.	Führe das Lernverfahren mit aktuellen Parametern ohne Teil i der Daten durch.
3.	Klassifiziere Teil i von S_l.
4.	Falls $i < p$, dann $i = i + 1$ und gehe zu Schritt 2.
5.	Sind die Ergebnisse nicht zufriedenstellend, verändere die Parameterwerte, z.B. für σ und/oder C; i=1; gehe zu Schritt 2.
6.	Speichere die optimalen Parameterwerte, z.B: σ^* und C^*, STOP.

Tabelle 4: Algorithmus zur Parameterbestimmung mittels Kreuzvalidierung.

- Anzahl positiver und negativer Punkte:

$$\text{pos} := |\{i \in S_l : \ y_i = 1\}| \quad , \qquad \text{neg} := |\{i \in S_l : \ y_i = -1\}|$$

- Anzahl falscher Klassifikationen bei der Validierung:

$$\text{fk} := |\left\{i \in S_l : \ y_i \cdot h(\boldsymbol{x}^i) < 0\right\}|$$

- Anzahl richtig positiver und falsch positiver Klassifikationen:

$$\begin{aligned} \text{rp} &:= |\left\{i \in S_l : \ y_i = 1, \ h(\boldsymbol{x}^i) = 1\right\}| \, , \\ \text{fp} &:= |\left\{i \in S_l : \ y_i = -1, \ h(\boldsymbol{x}^i) = 1\right\}| \end{aligned}$$

- Anzahl richtig negativer und falsch negativer Klassifikationen:

$$\begin{aligned} \text{rn} &:= |\left\{i \in S_l : \ y_i = -1, \ h(\boldsymbol{x}^i) = -1\right\}| \, , \\ \text{fn} &:= |\left\{i \in S_l : \ y_i = 1, \ h(\boldsymbol{x}^i) = -1\right\}| \end{aligned}$$

- Genauigkeit (accuracy):
$$\mathrm{ac} = \frac{\mathrm{rp} + \mathrm{rn}}{\mathrm{pos} + \mathrm{neg}} \tag{98}$$
- Sensitivität (sensitivity):
$$\mathrm{se} = \frac{\mathrm{rp}}{\mathrm{pos}} \tag{99}$$
- Spezifität (specificity):
$$\mathrm{sp} = \frac{\mathrm{rn}}{\mathrm{neg}} \tag{100}$$
- Präzision (precision):
$$\mathrm{pr} = \frac{\mathrm{rp}}{\mathrm{rp} + \mathrm{fp}} \tag{101}$$

Eine der Kennzahlen (98) oder (101) könnte zur Parameteroptimierung herangezogen werden, was jedoch sehr einseitig wäre. Es gibt Einzelmaße, die aufgrund ihrer Struktur besser geeignet sind, um Ergebnisse zu bewerten. Dazu gehören die gewichtete Fehlersummierung mittels Kostenmatrix, der Anreicherungsfaktor und die sogenannten F-Maße. Ersteres gehört in die Kategorie der zu minimierenden Fehlermaße, alle anderen Funktionen zählen zu den Gütemaßen. Wir haben uns gegen die Arbeit mit Kostenmatrizen entschieden, denn diese stehen in den seltensten Fällen zur Verfügung.

Der Anreicherungsfaktor ef berechnet das Verhältnis zwischen dem Anteil tatsächlich positiver in den als positiv klassifizierten Testpunkten und dem Anteil positiver Punkte in den gesamtem Testdaten [19]. Er besagt demnach, um welchen Faktor die Wahrscheinlichkeit einen positiven Punkt zu ziehen ansteigt, wenn man anstelle des Gesamtdatenbestandes nur aus den Punkten mit Hypothese 1 auswählt. Formal ist dieser Faktor demnach definiert als

$$\mathrm{ef} := \frac{\mathrm{rp}}{\mathrm{rp} + \mathrm{fp}} \Big/ \frac{\mathrm{pos}}{l} = \frac{\mathrm{pr} \cdot l}{\mathrm{pos}} \ .$$

Es gilt $\mathrm{ef} \geq 0$ sowie

$$\mathrm{ef} \begin{cases} > 1 & \text{Anreicherung,} \\ = 1 & \text{kein Effekt der Anreicherung,} \\ < 1 & \text{Konzentrationsabnahme von Klasse } 1. \end{cases}$$

Der Anreicherungsfaktor ist nach oben beschränkt durch die Anzahl der Daten. Es gilt

$$\mathrm{ef} \leq \frac{l}{\mathrm{pos}},$$

wobei Gleichheit auftreten kann, wenn es keine falsch positiven Punkte gibt.

Der Anreicherungsfaktor kann Hinweise dazu geben, ob eine Klassifikationsfunktion geeignet ist, um positive, also interessante, Daten zu lokalisieren. Dennoch ist der Anreicherungsfaktor als Optimierungskennzahl zu einseitig, da er stark auf die Präzision abzielt. Zudem ist er für einen Performancevergleich auf verschiedenen Datensätzen nicht geeignet, da er nach oben beschränkt ist durch die Anzahl der Daten.

Eine andere Gruppe interessanter Performancekennzahlen stellen die sogenannten F-Maße (*f-measures*) dar. Ein F-Maß FM ist eine Abbildung von Präzision und Sensitivität auf einen einzelnen Wert. Es entspricht einer Gleichgewichtung von Präzision und Sensitivität (*harmonic mean*) zur besseren Bewertung von Klassifikationen auf möglicherweise unbalancierten Daten und wird im Allgemeinen definiert als [27]

$$\mathrm{FM} \;=\; 2\,\frac{\mathrm{pr}\cdot\mathrm{se}}{\mathrm{pr}+\mathrm{se}}\,. \tag{102}$$

Wegen $\mathrm{pr}\in[0,1]$ und $\mathrm{se}\in[0,1]$ gilt $\mathrm{FM}\in[0,1]$. Detaillierte Beschreibungen zum F-Maß und seinen Weiterentwicklungen innerhalb unserer Arbeit sind in [6, 7] zu finden. Das F-Maß wird im Anschluss an eine vollständige Kreuzvalidierung berechnet und ausgegeben.

In Tabelle 4 wird erwähnt, dass in jedem Optimierungsschritt die Parameterwerte neu zu wählen sind. Das bedeutet, eine iterative Suche wird durchgeführt. Unter einer iterativen Suche versteht man das einfache Durchprobieren einiger Parameterkombinationen, wobei nach jedem Durchlauf abhängig von der Performance entweder gestoppt oder wieder mit neu bestimmten Parameterwerten validiert wird. Diese Methode ist von den Ideen und der Flexibilität des Nutzers abhängig.

Wir haben zusätzlich zur iterativen Suche, die konkret nur aus mehrmaligem Aufruf der Kreuzvalidierung besteht, Standardmodelle und -parameter definiert sowie eine Rastersuche implementiert.

Die SVM-Software beinhaltet default-Werte für alle Modellparameter, um sicherzustellen, dass jeder Nutzer ein Ergebnis generieren kann. Wir verwenden als Standard das $L2$-Norm-Modell mittels NP-Algorithmus sowie den Gauß-Kern. Als Standardwerte sind

$$C=1 \qquad \text{und} \qquad \sigma=1$$

definiert. Sollten Slater- oder Polynomial-Kern gewählt werden, stehen ebenfalls Standardwerte zur Verfügung. Teilweise kann man mit diesen Voreinstellungen schon gute Ergebnisse erzielen.

Die Rastersuche, auch grid search oder Gittersuche genannt, stellt eine sehr einfache und weit verbreitete Methode der Parameteroptimierung dar. Dabei wird a priori für jeden Parameter eine endliche Anzahl von Werten angegeben, die getestet werden sollen. Es werden alle möglichen Parameterkombinationen verwendet. Diese Art der Parametersuche wird beispielsweise in [12] vorgeschlagen. Die Rastersuche wird von unserer Software zur Verfügung gestellt. Zu den Details sei auf Teil IV zur Softwaredokumentation hingewiesen.

Nach erfolgreich durchgeführter Parameteroptimierung verfügt man über alle benötigten Parameterwerte. Diese werden dann zusammen mit dem vollständigen Trainingsdatensatz dazu verwendet, ein einfaches Lernverfahren durchzuführen, um dann mit der erlernten Funktion einen Testdatensatz oder neue Daten zu klassifizieren. Stehen ausreichend viele Daten beider Klassen zur Verfügung, können, anstelle der Durchführung einer Kreuzvalidierung, auch einfach drei Datensätze - Trainings-, Validierungs-, und Testdatensatz - gebildet werden. Diese Variante steht ebenfalls zur Verfügung. Der Umweg über Validierungsdaten kann nur dann vollständig eingespart werden, wenn die Modellparameter bekannt oder fest vorgegeben sind.

Bei der Wahl der Parameter sei abschließend auf eine typische Fehlerquelle hingewiesen. Das in [32] auf Seite 128 beschriebene data snooping, bei dem die Anpassung der Modellparameter anhand des Testdatensatzes vollzogen wird, sollte unbedingt vermieden werden.

Literaturverzeichnis

[1] A. V. Anghelescu and I. B. Muchnik. *Optimization of SVM in a space of two parameters: weak margin and intercept - application in text classifier design.* Rutgers University, webpage, 2003.

[2] I. N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühlig. *Taschenbuch der Mathematik.* Verlag Harry Deutsch, 1999.

[3] O. Chapelle, V. N. Vapnik, O. Bousquet, and S. Mukherjee. *Choosing multiple parameters for support vector machines.* Machine Learning 2002, 46(1):131–159.

[4] T. M. Cover. *Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition.* IEEC 1965, 14:326–334.

[5] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods.* Cambridge University Press, Cambridge, UK, 2000.

[6] T. Eitrich and B. Lang. *Efficient optimization of support vector machine learning parameters for unbalanced datasets.* Journal of Computational and Applied Mathematics 2006, 196:425-436.

[7] T. Eitrich and B. Lang. *Parallel tuning of support vector machine learning parameters for large and unbalanced data sets.* In Proceedings of the First International Symposium on Computational Life, Science, Springer LNCS, Konstanz, Germany, September 2005
M. R. Berthold and R. Glen and K. Diederichs and O. Kohlbacher and I. Fischer (eds.), Springer Lecture Notes in Bioinformatics Vol. 3695, p. 253-264, 2005.

[8] A. Fischer. *Kontinuierliche Optimierung.* Unpublished script, TU Dresden, 2003.

[9] C. Großmann and J. Terno.4 *Numerik der Optimierung.* B. G. Teubner, Stuttgart, 1997.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2001.

[11] R. Herbrich. *Learning kernel classifiers: theory and algorithms.* MIT Press, Cambridge, MA, USA, 2001.

[12] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. *A practical guide to support vector classification.*
`http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`.

[13] C. Hsu and C. Lin. *A comparison of methods for multi-class support vector machines.* IEEE Transactions on Neural Networks 2002, 13:415–425.

[14] A. Jakulin, M. Mozina, J. Demsar, I. Bratko, and B. Zupan. *Nomograms for visualizing linear support vector machines.* Publications database, 2004.

[15] T. Joachims. *Text categorization with support vector machines: learning with many relevant features.* In Proceedings of ECML-98, 10th European Conference on Machine Learning, pages 137–142, Chemnitz, Germany, 1998. Springer Verlag.

[16] K. Jörgens. *Lineare Integraloperatoren.* B. G. Teubner, Stuttgart, 1970.

[17] S. S. Keerthi. *Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms.* IEEE Transactions on Neural Networks 2002, 13:1225–1229.

[18] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. *A fast iterative nearest point algorithm for support vector machine classifier design.* IEEE Transactions on Neural Networks 2000, 11(1):124–136.

[19] A. Kless and T. Eitrich. *Cytochrome p450 classification of drugs with support vector machines implementing the nearest point algorithm.* In J. A. López, E. Benfenati, and W. Dubitzky, editors, Knowledge Exploration in Life Science Informatics, International Symposium,KELSI 2004, Milan, Italy, November 25-26, 2004, Proceedings, volume 3303 of Lecture Notes in Computer Science, pages 191–205. Springer, 2004.

[20] J. E. Laird and P. Rosenbloom. *The evolution of the soar cognitive architecture.* In D. M. Steier and T. M. Mitchell, editors, Mind Matters: A Tribute to Allen Newell, pages 1–50. Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, 1996.

[21] D. G. Luenberger. *Optimization by vector space methods.* John Wiley & Sons, New York, 1969.

[22] F. Markowetz. *Support vector machines in bioinformatics.* Master's thesis, University of Heidelberg, 2001.

[23] J. Mercer. *Functions of positive and negative type and their connection with the theory of integral equations.* Philosophical Transactions of the Royal Society 1909, London A, 209:415–446.

[24] B. Mitchell, V. Demýanov, and V. Malozemov. *Finding the point of a polyhedron closest to the origin.* SIAM Journal on Control and Optimization 1974, 12(1):19–26.

[25] M. Momma and K. P. Bennett. *A pattern search method for model selection of support vector regression.* In R. L. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, editors, Proceedings of the Second SIAM International Conference on Data Mining, Arlington, VA, USA, April 11-13, 2002. SIAM, 2002.

[26] K. Morik, P. Brockhausen, and T. Joachims. *Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring.* In Proc. 16th International Conf. on Machine Learning, pages 268–277. Morgan Kaufmann, San Francisco, CA, 1999.

[27] D. R. Musicant, V. Kumar, and A. Ozgur. *Optimizing F-measure with support vector machines.* In I. Russell and S. M. Haller, editors, Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference, May 12-14, 2003, St. Augustine, Florida, USA, pages 356–360. AAAI Press, 2003.

[28] E. Osuna, R. Freund, and F. Girosi. *Support vector machines: training and applications.* AI Memo 1602, Massachusetts Institute of Technology, 1997.

[29] J. Platt. *Fast training of support vector machines using sequential minimal optimization.* In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning, pages 185–208, Cambridge, MA, 1999. MIT Press.

[30] J. R. Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann Publishers Inc., 1993.

[31] S. J. Russell and P. Norvig. *Artificial intelligence: a modern approach.* Prentice-Hall International, 1995.

[32] B. Schölkopf and A. J. Smola. *Learning with kernels.* MIT Press, Cambridge, MA, 2002.

[33] V. N. Vapnik. *The nature of statistical learning theory.* Springer, New York, 1995.

[34] V. N. Vapnik. *Statistical learning theory.* John Wiley & Sons, New York, 1998.

Datenanalyse

Claudia Druska, Tatjana Eitrich, Wolfgang Meyer

John von Neumann-Institut für Computing
Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
52425 Jülich
E-mail: {*c.druska, t.eitrich, w.meyer*}*@fz-juelich.de*

In diesem Teil stellen wir die konkreten Möglichkeiten der Datenanalyse vor, die sich aus den Implementierungen der in den bisherigen Teilen beschriebenen Methoden und weiteren Ergänzungen ergeben. Das innerhalb des Projektes entstandene Softwarepaket ermöglicht eine umfassende Säuberung und Analyse von pharmazeutischen Daten.

In Abbildung 1 ist der grobe Ablauf der Data Mining Pipeline dargestellt. Im Folgenden werden die einzelnen Schritte genau beschrieben. Sie enthalten eine Vielzahl von wichtigen Data Mining Methoden, die in ihrer Gesamtheit zu verlässlichen Informationen über Auffälligkeiten und Strukturen in Daten führen.

1 Datensatz

Grundlage der Data Mining Pipeline ist der zur Verfügung stehende Datensatz. Als einen Datensatz bezeichnen wir eine Menge von Rohdaten mit der folgenden Struktur:

- eine Matrix, deren Zeilen aus den jeweiligen Substanzen bestehen und deren Spalten die zu untersuchenden Attribute sind;
- eine Liste, welche die Namen der Substanzen enthält; sowie
- eine Liste, welche die Namen der Attribute beinhaltet.

Optional kann für die Erstellung von Klassifikatoren zusätzlich ein Klassenvektor bereitgestellt werden. Weitere Details dazu folgen im Abschnitt 4.2.

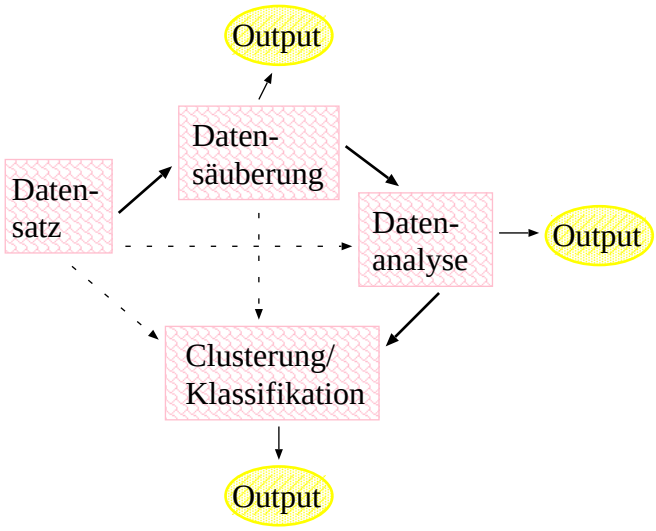


Abbildung 1: Data Mining Pipeline im GALA-Projekt.

2 Datensäuberung

In einem ersten Schritt nach dem Einlesen der Rohdaten und der zugehörigen Namenslisten für Substanzen und Attribute erfolgt eine Überprüfung der Datenmatrix auf Nullspalten. Da in der Regel große Datenmengen (bis zu 100000 Substanzen und 1000 Attribute) aus dem Data Warehouse zur weiteren Analyse gefiltert werden, kann es vorkommen, dass sich Nullspalten in die Rohdatenmatrix “einschleichen”. Diese werden vor Beginn der eigentlichen Datenaufbereitung entfernt. Die entsprechende Programmroutine erzeugt gegebenenfalls neue Eingabedaten, d.h. es wird eine um die Nullspalten bereinigte Rohdatenmatrix und die ebenfalls angepasste Liste der Attributnamen zur Verfügung gestellt. Im Anschluss daran können nun erste inhaltliche Datensäuberungen angestoßen werden.

2.1 Missing Value Behandlung

Neben der Entfernung von Nullspalten ist zu Beginn der Untersuchungen auch zu entscheiden wie Missing Values, d.h. fehlende oder offensichtlich falsche Einträge in der Rohdatenmatrix, zu behandeln sind. Es wurden zwei mögliche Behandlungsarten implementiert. Zum einen können unvollständige Beobachtungen, das heißt Beobachtungen, die mindestens einen fehlenden Wert aufweisen, unberücksichtigt bleiben. Sie werden aus dem Datensatz gestrichen. Der aus den vollständigen Beobachtungen bestehende verkleinerte Datensatz ist dann Basis der weiteren Betrachtungen. Eine andere Methode besteht darin, fehlende Beobachtungswerte durch die arithmetischen Mittelwerte der betroffenen Variablen zu ersetzen. Dadurch vermeidet man, auf wichtige Beobachtungen frühzeitig wegen des Fehlens einzelner Werte zu verzichten.

2.2 Ausreißerbestimmung

Die in den Rohdaten enthaltenen Informationen können durch extreme Werte beeinflusst sein. Dabei muss man zwischen fehlerbehafteten Daten und Daten mit überdurchschnittlichen Eigenschaften unterscheiden. Es wird empfohlen, erstere vor weiteren Analysen zu eliminieren, um spätere Ergebnisse nicht zu verfälschen. Ausreißer der zweiten Kategorie sollten identifiziert und kenntlich gemacht werden. Da diese Unterscheidung in der Regel pharmakologisches Fachwissen voraussetzt, identifiziert das Programm zunächst die Daten mit stark abweichenden Werten und der Anwender entscheidet über die weitere Behandlung.
Für eine Variable sind diejenigen Beobachtungen Ausreißer, deren standardisierte Werte betragsmäßig größer als ein kritischer Wert sind, für ein Variablenpaar diejenigen Beobachtungen, deren Mahalanobisdistanzen (3) (Teil II, S. 20) diese Bedingung erfüllen. Der kritische Wert wird entweder fest vorgegeben (Abschnitt 3.3.1, Teil II) oder durch Vorgabe des Niveaus des Diskordanztests für einen Ausreißer bei eindimensionaler (Abschnitt 6.4.1, Teil II) bzw. zweidimensionaler (Abschnitt 3.3.3, Teil II) Normalverteilung festgelegt. Für zweidimensionale Ausreißer besteht außerdem die Möglichkeit, den kritischen Wert so zu wählen, dass im Falle normalverteilter Daten der Anteil der als Ausreißer deklarierten Beobachtungen etwa $\sqrt{1/10n}$ beträgt, wobwi n die Anzahl der vollständigen Wertepaare ist (Abschnitt 3.3.2, Teil II).

2.3 Transformation

Ein Ziel von Transformationen ist es, nichtlineare Abhängigkeiten auf lineare Abhängigkeiten abzubilden. So können beispielsweise Volumendaten durch die Transformation mit dritter Wurzel linearisiert werden, um anschließend mit Hilfe der Korrelationsrechnung Abhängigkeiten aufdecken zu können.
Diese Programmoption transformiert die Eingabevariablen gemäß der Vorgaben durch eine benutzerdefinierte Eingabedatei. Enthält diese Datei das Paar $\{n \in \mathbb{N}, k \in \mathbb{N}_{<8}\}$, wird die n-te Variable wie folgt transformiert:

Wert des Parameters k	Transformationsregel
0	keine Transformation
1	Quadratwurzel, Missing Value für negative Werte
2	Quadrat
3	3. Wurzel
4	3. Potenz
5	Reziproker Wert, Missing Value für 0.0
6	Natürlicher Logarithmus, Missing Value für nichtpositive Werte
7	Exponentialfunktion

3 Datenanalyse

Mit den bisher beschriebenen Methoden ist es möglich, die Rohdaten aus dem Data Warehouse zu säubern und erste Informationen zur Datenstruktur zu sammeln. In diesem Abschnitt wird nun auf die eigentliche Datenanalyse eingegangen.

3.1 Korrelation

Die Programmoption zur Korrelationsberechnung ermöglicht die Untersuchung der Rohdaten oder transformierten Daten auf (lineare) Abhängigkeiten zwischen den Messreihen. Dabei stehen drei Arten der Korrelationsberechnung zur Verfügung. Man kann zum einen zwischen den klassischen Verfahren nach Pearson (Abschnitt 2, Teil II) oder Spearman (Abschnitt 5, Teil II) wählen oder man wendet eine robuste Methode (Abschnitt 8.6, Teil II) an.
Die Pearsonsche Produktmomentkorrelation aller Variablenpaare wird unter Verwendung aller vollständigen Wertepaare berechnet. Dabei werden die Mittelwerte und Standardabweichungen der Variablen berechnet. Wird das Verfahren nach Spearman ausgewählt, erfolgt eine Rangtransformation für alle Variablen, d.h. die Beobachtungswerte einer Variable werden durch die Ränge ersetzt, und anschließend wird die Pearsonsche Produktmomentkorrelation aller Variablenpaare unter Verwendung der vollständigen Wertepaare berechnet. Diese Vorgehensweise ermöglicht das Aufdecken monotoner Abhängigkeiten, die durch lineare Verfahren auf Originaldaten nicht entdeckt werden. Außerdem können auf Grundlage der Ergebnisse der Korrelationsberechnung Ausreißer identifiziert und eliminiert werden. Die Identifikation wird wie in Abschnitt 2.2 beschrieben durchgeführt. Über einen Parameter kann man die Anzahl der Korrelationsberechnungen steuern. Führt man mehr als eine Korrelationsberechnung durch, werden bei den aufeinanderfolgenden Rechnungen die zuvor entdeckten Ausreißer nicht mehr berücksichtigt. Auf diese Art und Weise können die Rohdaten iterativ von Ausreißern gesäubert werden. Dabei sollte die Anzahl der Läufe jedoch nicht zu groß sein, da sonst ein signifikanter Teil der Rohdaten entfernt werden müsste. Die Anzahl der Iterationen zur Ausreißereliminierung ist daher mit 2 zunächst klein voreingestellt, kann jedoch über eine Option gesteuert werden.

3.2 PCA

Diese Programmoption ermöglicht die Durchführung einer Hauptkomponentenanalyse. Dabei kann der Anwender zwischen folgenden drei Verfahren wählen:

(1) Hauptkomponentenanalyse einer Pearsonschen Produktmomentkorrelationsmatrix,

(2) Hauptkomponentenanalyse einer Rangkorrelationsmatrix und

(3) robuste Hauptkomponentenanalyse nach Campbell (iterative Bestimmung robuster Eigenwerte und Eigenvektoren, s. Abschnitt 8.7, Teil II).

Zusätzlich kann bestimmt werden, welche Scores der Hauptkomponenten berechnet und ausgegeben werden. Entweder werden die Scores der Hauptkomponenten zu den k ($k = 1, 2, \ldots$) größten Eigenwerten oder die Scores der Hauptkomponenten, deren zugehörige Eigenwerte größer als $1 + \frac{k}{10}$ ($k = 0, -1, \ldots, -10$) sind, berechnet. Da das Programm die Hauptkomponenten der Korrelationsmatrix bestimmt, werden die Scores mittels standardisierter Daten berechnet. Zur Berechnung der Scores der Hauptkomponenten einer Rangkorrelationsmatrix werden nicht die Rohdaten, sondern stets rangtransformierte Daten verwendet. Außerdem werden die Korrelationen zwischen den Variablen und den Hauptkomponenten berechnet und ausgegeben. Darüber hinaus wird für jede Variable der Anteil ihrer Varianz berechnet, der durch die ersten k ($k = 1, ...,$ Anzahl der der PCA zugrundeliegenden Variablen) Hauptkomponenten nicht erklärt wird.

3.2.1 Ausreißerbestimmung mittels Hauptkomponenten

Die Ergebnisse der Hauptkomponentenanalyse können ebenfalls der Ausreißersuche dienen. Wir unterscheiden dabei drei Szenarien.

Univariate Ausreißer
Bezüglich einer Hauptkomponente sind diejenigen Beobachtungen Ausreißer, deren standardisierte Scores für die betrachtete Hauptkomponente dem Betrag nach größer als ein kritischer Wert sind. Bestimmt werden die Ausreißer für die k Hauptkomponenten mit den größten zugehörigen Eigenwerten. Für $k = 0$ wird keine Ausreißerbehandlung durchgeführt.

Multivariate Ausreißer bzgl. der k Hauptkomponenten mit den größten Eigenwerten
Ausreißer sind die Beobachtungen, deren quadrierte Mahalanobis-Distanzen bzgl. der k Hauptkomponenten mit den größten Eigenwerten größer als ein kritischer Wert sind.

Multivariate Ausreißer bezüglich der (Anzahl der Variablen - k) Hauptkomponenten mit den kleinsten Eigenwerten
Ausreißer sind die Beobachtungen, für die die Quadratsumme der Scores der letzten (Anzahl der Variablen - k) Hauptkomponenten größer als ein kritischer Wert ist.

Der kritische Wert wird entweder fest vorgegeben oder durch Vorgabe des Niveaus des Diskordanztests für einen Ausreißer bei ein- bzw. mehrdimensionaler Normalverteilung festgelegt (Abschnitt 6.4, Teil II).

3.3 Variablenselektion

Neben der Hauptkomponentenanalyse bietet die Variablenselektion eine weitere Möglichkeit zur Dimensionsreduzierung an (Teil II, Abschnitt 7). Bei der Variablenselektion wird

eine fest vorgegebene oder eine durch Optimierungsmethoden berechnete Anzahl von Variablen für die weiteren Analyseverfahren ausgesucht. Die restlichen Variablen verlassen die Data Mining Pipeline. Die Variablenselektion stellt eine wichtige Schnittstelle zwischen den Datensäuberungs- bzw. -analysemethoden auf der einen und den Cluster- bzw. Klassifikationsverfahren auf der anderen Seite dar. Aus diesem Grund sind zwei grundsätzlich unterschiedliche Vorgehensweisen implementiert worden:

- unüberwachte Variablenselektionen ohne Verwendung eines Klassenvektors, sowie
- überwachte Variablenselektionen mit expliziter Auswertung des Klassenvektors.

Keines der Verfahren kann favorisiert werden, da die erzielten Ergebnisse stark daten- und parameterabhängig sind. Über Vor- und Nachteile muss individuell entschieden werden [5].

3.3.1 Unüberwachte Variablenselektion

Die Durchführung der Variablenselektion basiert auf einer Korrelationsmatrix, die man wie in Abschnitt 3.1 beschrieben näher spezifizieren kann. Innerhalb der Variablenselektion kann man zwischen fünf Verfahren wählen. Zur Auswahl steht zum einen die Methode der Principal Variables nach McCabe [7], bei der iterativ jeweils die Variable ausgewählt wird, welche die Summe der Residualvarianzen minimiert. Darüber hinaus wurden zwei auf einer PCA basierenden Selektionsverfahren implementiert. Bei der Selektionsvariante werden iterativ Variablen ausgewählt, die zusammen einen möglichst großen Anteil der Streuung aller Variablen erklären, während bei der Eliminationsvariante iterativ Variablen entfernt werden, deren Streuung von den verbleibenden Variablen möglichst gut erklärt wird. In beiden Verfahren wird iterativ jeder Hauptkomponente die Variable mit dem größten Eintrag in dem zugehörigen Eigenvektor (Faktorladung) unter den noch freien Variablen zugeordnet. Bei der Selektionsvariante werden die Hauptkomponenten in der Reihenfolge fallender zugehöriger Eigenwerte, in der Eliminationsvariante in der Reihenfolge wachsender zugehöriger Eigenwerte abgearbeitet. Das vierte und fünfte Verfahren unterscheidet sich in der Anzahl der durchzuführenden Hauptkomponentenanalysen. Hier wird in jedem Iterationsschritt eine neue PCA durchgeführt. Bei der Selektionsvariante erfolgt eine PCA der partiellen Korrelationsmatrix der verbliebenen Variablen bezüglich der ausgewählten Variablen. Bei der Eliminationsvariante handelt es sich um ein Verfahren, bei dem die PCA der Korrelationsmatrix der verbliebenen Variablen verwendet wird.

3.3.2 Überwachte Variablenselektion

Das implementierte Verfahren zur überwachten Variablenselektion ordnet alle Attribute in eine Rangliste, wobei diejenigen Attribute einen hohen Rang zugeordnet bekommen, die am besten dazu geeignet sind, die zugrundeliegende Klassenstruktur der Trainingsdaten zu

erklären. Das zugehörige Verfahren arbeitet auf den eindimensionalen Verteilungen der einzelnen Variablen. Für jede Variable k kann man die Menge der Realisierungen in zwei Gruppen einteilen: positive und negative Realisierungen. Die beiden daraus entstehenden eindimensionalen Verteilungen werden untersucht. Weichen diese in Lage und Struktur stark voneinander ab, wird ein funktionaler Zusammenhang zwischen der Variable k und der Klassenstruktur angenommen. Als Grundlage der Tests dient der Chi-Quadrat ($\mathcal{X}^2$) Homogenitätstest [6]. Der $\mathcal{X}^2$-Test ist geeignet für diskrete Daten, d.h. es wird eine Größe X mit c Kategorien und g Gruppen betrachtet ($c,\ g \in \mathbb{N}$). Die Implementierung arbeitet mit $g = 2$, da ausschließlich binäre Probleme vorliegen. Um diesen diskreten Test anwenden zu können, werden alle auf Varianz 1 skalierten Werte in Kategorien eingeordnet. Dazu wird mit einer vordefinierten Menge an Intervallen gestartet. Im Anschluss wird die Anzahl von Vertretern in jeder Kategorie geprüft und falls weniger als 5 Vertreter vorhanden sind, wird die jeweilige Kategorie mit einem Nachbarn zu einer gemeinsamen neuen Kategorie verbunden. Für jede Variable wird gesondert entschieden, ob diese zur Klasseneinteilung beitragen kann oder nicht.

Das beschriebene Verfahren zur überwachten Variablenselektion allein garantiert noch nicht eine optimale Auswahl von Attributen. Zwischen den Variablen mit hohen Rängen können große Korrelationen auftreten. Diese sind unerwünscht und sollten eliminiert werden. Aus diesem Grund schließt sich an das Ordnen der Variablen ein hierarchisches Clusterverfahren an [3]. Dazu wird die gut bekannte complete-link Methode eingesetzt. Diese definiert den Abstand zwischen zwei Clustern $\mathcal{C}_0$ and $\mathcal{C}_1$ als Maximum aller paarweisen Abstände zwischen Punkten in $\mathcal{C}_0$ und $\mathcal{C}_1$ [3]. Die Rangkorrelationen zwischen Variablenpaaren werden verwendet, um einen Baum aufzubauen, der es dann ermöglicht, hochkorrelierte Teilbäume zu erkennen und durch eine Variable mit hohem Rang zu ersetzen.

Resultat dieser zweistufigen Vorgehensweise ist eine Menge von Variablen, welche die Klassenzugehörigkeiten in den Daten erklärt, jedoch frei von redundanten Informationen ist.

Die beiden Methoden des überwachten und unüberwachten Selektierens von Variablen können auch hintereinandergeschaltet werden.

4 Clusteranalyse und Klassifikationsverfahren

Im Anschluss an die Schritte der Datensäuberung und -analyse können Methoden des überwachten und des unüberwachten Lernens eingesetzt werden. Diese Methoden ordnen Datenpunkte in Gruppen ein, wobei diese Gruppen entweder schon vorgegeben sind oder noch definiert werden müssen.

4.1 Clusteranalyse

Für das unüberwachte Lernen von Clustern stehen zwei übernommene Verfahren zur Verfügung:

- AutoClass [1] ist eine frei verfügbare Software zur unüberwachten Bayes-Klassifikation, der sogenannten Bayes-Clusterung. Ein sehr wichtiger Aspekt dieser Software ist, dass die Anzahl der Cluster automatisch bestimmt wird. Das ermöglicht dem Benutzer, weitere Trends in den im Allgemeinen nur in zwei Gruppen gegliederten Daten zu finden. Zusätzlich zur klassischen Clusteranalyse kann auch eine echte Klassifikation stattfinden. Dabei werden noch unbekannte Testfälle in die vorher generierten Cluster eingeordnet, ohne dass diese für die Modellierung eingesetzt worden sind. AutoClass beschreibt die Daten als Mischung unabhängiger Klassen, wobei jede Klasse über eine Wahrscheinlichkeitsverteilung über dem Raum, den die Variablen aufspannen, definiert ist. Gauß'sche Verteilungen für die stetigen Variablen und Bernoulli-Verteilungen für die diskreten Variablen werden angenommen. Das Verfahren bestimmt eine Menge von Klassen, die am wahrscheinlichsten zu den Daten und den weiteren Annahmen passen.
- Subset ist ein Clustering-Programm, welches sowohl zur Datenreduktion als auch zur Clusteranalyse verwendet werden kann. Subset arbeitet in der Originalversion ausschließlich mit binären Variablen. Aus diesem Grund steht nur der Tanimoto-Koeffizient als Abstandsmaß zur Verfügung. Um den Algorithmus auch für die Analyse stetiger Variablen verwenden zu können, wurde das Programm um ein auf dem Euklidischen Abstand basierendes Abstandsmaß erweitert und steht somit zur Reduktion und Clusterung von Daten mit stetigen sowie diskreten Variablen zur Verfügung. Die Grundlage der Clusteranalyse bildet eine in [9] beschriebene stochastische Clustermethode. Die Datenreduktionsfunktion innerhalb von Subset ist insbesondere für große Datensätze von Interesse, da das Programm eine repräsentative Teilmenge der Daten so bestimmen kann, dass aus jedem vorhandenen Cluster eine gleich große Anzahl von Substanzen erhalten bleibt. Vor allem für sehr unausgeglichende Daten bietet das die Möglichkeit, eine sehr stark aufgeblähte Gruppe so zu verkleinern, dass keine wichtigen Informationen verloren gehen. In diesem Sinne ist diese Methode insbesondere als Vorstufe zu echten Klassifikationsverfahren, die sehr empfindlich auf unausgeglichene Klassen reagieren, zu sehen.

Die vorhandenen Cluster-Methoden werden innerhalb des Projektes für folgende Fragenstellungen verwendet:

- für ungelabelte Daten
  - Clusteranalyse (klassisch) und
  - Ziehen einer repräsentativen Stichprobe aus einem großen Datensatz,
- für gelabelte Daten

– Clusteranalyse für zwei Gruppen auf einem Trainingsdatensatz zur Überprüfung der gegebenen Klassifikationen,

– Clusteranalyse auf einer Klasse des Trainingsdatensatzes, um zu prüfen, ob in dieser Klasse große strukturelle Unterschiede zwischen den Punkten vorhanden sind, und

– Verkleinerung von ganzen Datensätzen beziehungsweise von einzelnen stark aufgeblähten Klassen, um robuste Klassifikatoren erstellen zu können.

4.2 Klassifikationsverfahren

Klassifikation ist eine der wichtigsten Fragestellungen innerhalb des GALA-Projektes. Basierend auf gesäuberten und in beiden Dimensionen reduzierten Daten ist es die Aufgabe eines Klassifikationsverfahrens, eine Funktionalität zwischen Eingaben und gewünschten Ausgaben in einem Datensatz zu erkennen und mittels der zur Verfügung stehenden Funktionen zu formulieren. Diese Funktionalität kann dann für weitere Daten zur Klassenvorhersage verwendet werden. Im Teil III dieses Bandes ist dieses überwachte Lernen ausführlich beschrieben worden.

Bei dieser Form des Data Minings gibt es immer zwei Arten von Daten:

- die Trainingspunkte als Grundlage für die Modellierung, sowie
- die Punkte, die klassifiziert werden sollen.

Zusätzlich zu einer Datenmatrix ist es notwendig, jedem Trainingspunkt, der in das Modell einfließen soll, ein Klassenlabel zuzuordnen. Im Falle binärer Klassifikation, die am häufigsten als Problemstellung auftritt, bedeutet das, für jede Substanz i des Trainingsdatensatzes einen Klassenwert $y_i \in \{-1, 1\}$ anzugeben.

Die Erstellung von Klassifikatoren erfolgt innerhalb des Projektes hauptsächlich mit dem modernen Verfahren der Support-Vektor-Maschinen, welches im Teil III dieses Bandes detailliert vorgestellt wurde und zu den mächtigsten nichtlinearen Klassifikationsverfahren der heutigen Zeit zählt.

Die Anwendung einer Support-Vektor-Maschine gliedert sich in zwei hintereinanderfolgende Phasen:

- Zu Beginn wird basierend auf den Informationen des Trainingsdatensatzes eine nichtlineare Klassifikationsfunktion erlernt und gespeichert.
- Im Anschluss kann diese Funktion verwendet werden, um neue Daten zu labeln beziehungsweise um schon bekannte Labels zu überprüfen.

Mehrklassige Modelle können einfach über ein mehrmaliges binäres Lernen mit einem der Ansätze aus [2] generiert werden.

4.2.1 Trainingsphase

Das Erlernen einer Klassifikationsfunktion, das sogenannte Training, bedeutet im Wesentlichen, dass ein großes quadratisches Optimierungsproblem zu lösen ist. Dazu sei auf Abschnitt 3.5 in Teil III verwiesen. Es stehen dabei zwei verschiedene Modelle zur Verfügung:

- Das Modell der 1-Norm summiert schwache Klassifikationen und Fehler während des Trainings einfach auf. Siehe dazu Aufgabe (74) auf Seite 149.
- Das Modell der 2-Norm summiert schwache Klassifikationen und Fehler in quadrierter Form auf. Das bedeutet, Punkte, die in der Nähe der gewünschten Klassifikationsgüte liegen (Fehler im Intervall $[0, 1]$), werden etwas mehr toleriert, wohingegen echte Fehler im Intervall $(1, \infty)$ hart bestraft werden. Siehe dazu Aufgabe (76) auf Seite 150.

Die Algorithmen zur Umsetzung dieser Modelle sind in den Abschnitten 4.1.5 und 4.2 in Teil III vorgestellt worden. Der SMO-Algorithmus von J. Platt [8] implementiert das 1-Norm Verfahren, während der NP-Algorithmus von S. Keerthi et. al [4] die 2-Norm Methode umsetzt.

Für die Nutzung einer nichtlinearen Support-Vektor-Maschine ist es notwendig eine Kernfunktion zu definieren. Für den Lernvorgang stehen vier vordefinierte Kerne zur Verfügung:

- der Gauß-Kern (siehe (33) auf Seite 127),
- der Slater-Kern (siehe (35) auf Seite 128),
- der Polynomial-Kern (siehe (34) auf Seite 127), sowie
- der Tanimoto-Kern (siehe (36) auf Seite 128).

Zusätzlich kann auch ein selbst definierter Kern verwendet werden. Dafür muss zwar der Quellcode geändert werden, diese Änderung betrifft jedoch nur eine einzige Routine, die Funktion zur Berechnung des Kerns. Diese ist losgelöst von den Algorithmen implementiert worden. Eine Änderung wird damit sowohl für den SMO- als auch für den NP-Algorithmus wirksam.

4.2.2 Optimierungsstufen

Ein einzelnes SVM-Training generiert eine Hypothese, die für die verwendeten Parameter optimal ist. Die Schwierigkeit bei der Anwendung von Support-Vektor-Maschinen für komplizierte Datensätze ist die konkrete Wahl der Parameterwerte. Diese beeinflussen die Güte der Klassifikationsfunktion, sind aber nicht direkt berechenbar. Vielmehr müssen sie während einer Optimierungsphase erlernt werden.

Die hier betrachteten Parameter sind

- Kern-Parameter, zum Beispiel $\sigma > 0$ für den Gauß-Kern,

- Strafparameter C^+ für positive Trainingspunkte, und
- Strafparameter C^- für negative Trainingspunkte.

Zur Verfügung steht eine automatisierte Parameteroptimierung, die nach folgendem Schema abläuft:

(1) Für jeden zu optimierenden Parameter wird eine Datei mit gewünschten Werten zur Verfügung gestellt. Feste Parameterwerte können ebenfalls vorgegeben werden, falls sie schon bekannt sind. Das verkürzt die Optimierungszeit.

(2) Für alle möglichen Parameterkombinationen werden die Schritte (3) bis (5) durchgeführt.

(3) Zehn- oder l-fache Kreuzvalidierung mit den aktuellen Parameterwerten.

(4) Basierend auf den internen Testergebnissen, Berechnung des Gütemaßes.

(5) Aktualisierung des bisher besten Gütemaßes und Speicherung des zugehörigen Parametertupels.

(6) Im Anschluss wird das optimale Parametertupel für das reguläre SVM-Training auf dem kompletten Datensatz verwendet.

Zu den Details der Kreuzvalidierung verweisen wir auf Abschnitt 5.2 im Teil III. Das verwendete Gütemaß ist insbesondere auf Sensitivität ausgelegt. Es wurde bereits vorgestellt, siehe (102) auf Seite 174.

4.2.3 Recall

Zusätzlich zu den Kreuzvalidierungstests ist die Funktionalität des Recalls in die Software eingebettet worden. Unter Recall wird ein unabhängiger Test auf Teilen der Trainingsdaten verstanden. Dieser Test findet im Anschluss an das reguläre Training statt. Damit dieser Test unabhängig von den zur Parameteroptimierung und zum finalen Training verwendeten Daten ist, wird bei der Wahl der Recalloption die gewünschte Anzahl von Substanzen zurückgehalten und nicht zum Training freigegeben. In Anschluss an das Training werden diese Substanzen klassifiziert und der Benutzer erhält eine Information zu den dabei entstandenen Fehlklassifikationen in beiden Klassen.

4.2.4 Test und Klassifikation

Zusätzlich zu den Trainingsdaten, auf denen Parameteroptimierung, Validierung, Training und Recall durchgeführt werden können, kann eine weitere Klasse von Punkten bearbeitet werden. Je nach Bedarf werden die Punkte entweder als unbekannte Substanzen klassifiziert oder es findet eine Klassifikation gekoppelt an eine Auswertung statt. Letzterer Fall

setzt voraus, dass die Labels der Punkte bekannt sind. Dieser Test erscheint zunächst redundant, da die gleiche Support-Vektor-Maschine eingesetzt wird, welche auch schon auf den Recall-Punkten gearbeitet hat. Die Test-Option ist jedoch immer dann wichtig, wenn ganz bestimmte Substanzen getestet werden sollen. Bei der Recall-Option wird diese Möglichkeit nicht gegeben, da dort immer auf einer zufällig gewählten Datenmenge getestet wird, um die Verteilung der Klassen in etwa widerzuspiegeln. Durch die Test-Option gewinnt der Benutzer deutlich an Flexibilität. Die Auswertung erfolgt auch hier getrennt nach Klassen.

Literaturverzeichnis

[1] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. *AutoClass: a Bayesian classification system.* Proceedings of the Fifth International Conference on Machine Learning, Morgan Kaufmann Publishers, San Francisco, 1988, pages 54–64.

[2] C. Hsu and C. Lin. *A comparison of methods for multi-class support vector machines.* IEEE Transactions on Neural Networks 2002, 13:415–425.

[3] A. Jain, M. Murty, P. Flynn, P. *Data clustering: a review.* ACM Computing Surveys 1999, 31(3) 264–323.

[4] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. *A fast iterative nearest point algorithm for support vector machine classifier design.* IEEE Transactions on Neural Networks 2000, 11(1):124–136.

[5] A. Kless and T. Eitrich. *Cytochrome P450 classification of drugs with support vector machines implementing the nearest point algorithm.* In J. A. López, E. Benfenati, and W. Dubitzky, editors, Knowledge Exploration in Life Science Informatics, International Symposium, KELSI 2004, Milan, Italy, November 25-26, 2004, Proceedings, volume 3303 of Lecture Notes in Computer Science, pages 191–205, Springer, 2004.

[6] S. Kotz, N. L. Johnson. *Encyclopedia of Statistical Sciences (volume 3).* John Wiley & Sons, 1983.

[7] G. P. McCabe. *Principal variables.* Technometrics 1984, 26:137–144.

[8] J. Platt. *Fast training of support vector machines using sequential minimal optimization.* In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning, pages 185–208, Cambridge, MA, MIT Press, 1999.

[9] C. H. Reynolds, R. Druker, L. B. Pfahler. *Lead discovery using stochastic cluster analysis (SCA): a new method for clustering structurally similar compounds.* J. Chem. Inf. Comput. Sci. 1998, 38:305–312.

Anwendungsbeispiel: Cytochrom P450 Profilierung von neuen Wirkstoffen

Achim Kless

Biomedizinische Forschung, Drug Discovery
Grünenthal GmbH
52078 Aachen
E-mail: Achim.Kless@grunenthal.com

1 Einleitung

Der Prozess der Identifikation von neuen potentiellen Wirkstoffkandidaten hängt nicht nur von der Auswahl der potentesten Substanz für ein spezifisches Target ab. Durch unsere Erfahrungen in der Vergangenheit wissen wir, dass der Metabolismus und die Pharmakokinetik für eine Substanzauswahl gleichberechtigt berücksichtigt werden müssen. Cytochrom P450 Enzyme sind metabolisierende Enzyme, die in verschiedenen Geweben und Blut vorkommen und hauptsächlich in der Leber eine aktive Rolle im oxidativen Abbau der Substanzen spielen. Insbesondere sind die Isoformen 1A2, 2C19, 2C9, 2D6 und 3A4 wichtig, da diese am Metabolismus von 90% der Substanzen beteiligt sind [10, 11]. Wie von Flockhart [12] näher beschrieben, blocken Inhibitoren die Funktion dieser Enzyme, so dass die verbleibende Aktivität reduziert ist. In unserem Beispiel haben wir uns auf die 2D6-Isoform konzentriert, da bei dieser Isoform einige inaktive polymorphe Formen in der Bevölkerung existieren, was analog zu einer 2D6-Inhibition ist. Dies kann dann bei Wirkstoffen, die ein 2D6 Substrat sind, zu toxisch hohen Plasmaspiegeln führen [5]. CYP2D6 ist ein Enzym, das primär durch Substanzen mit einem Stickstoffatom und einem hydrophoben Part blockiert wird, was durch Pharmakophoranalysen gezeigt werden konnte. Wenn dieses Enzym blockiert wird, kann es auch mit einer zweiten Substanz zu unerwünschten Wirkstoff-Wechselwirkungen kommen, die für eine Anwendung am Menschen unerwünscht und bei der Selektion von neuen Wirkstoffkandidaten zu vermeiden

sind. Es ist deshalb wünschenswert, für eine neue chemische Struktur die Aktivität an diesem Enzym vorhersagen zu können, da hochqualitative Daten unter in vivo Bedingungen nur zu einem sehr späten Zeitpunkt in der präklinischen Forschung erhoben werden. Aus diesem Grund ist die in silico Vorhersage zu einer Schlüsseltechnologie in den frühen Phasen der Forschung geworden [4, 6]. Unsere Arbeiten sollen zeigen, dass unsere Methoden zur Erstellung von Klassifikatoren dieser Art geeignet sind und bereits etablierte Verfahren wie Pharmakophoranalyse, Homologiemodelle von Proteinen und quantenmechanische Berechnungen sinnvoll ergänzen. Der Bedarf an genauen Modellen, die eine hochoptimierte Analyse von virtuellen Substanzen erlauben, hat in der Vergangenheit dazu geführt, dass sich viele Arbeitsgruppen mit diesem Problem beschäftigt haben und ein ganzes Arsenal von Computermethoden zur Klassifikation von CYP450 Enzymen existiert [1, 2, 3]. Dazu gehören in der neueren Literatur Verfahren, die verschiedene Klassifikatoren miteinander verknüpfen. Man spricht dann auch von Konsensmodellen, die das Gesamtergebnis verbessern. Die zugrundeliegenden Algorithmen sind Entscheidungsbaumverfahren wie z.B. rekursives Partitionieren und Ensemble-Methoden wie AdaBoost oder auch die von uns in Teil III beschriebenen Support-Vektor-Maschinen. Hauptproblem in diesem Zusammenhang ist der Zugang zu genügend validierten, experimentellen Daten. Normalerweise ist die Generalisierungsfähigkeit einer Klassifikation mit der Anzahl der eingehenden Datenpunkte korreliert. Daraus kann aber nicht notwendigerweise gefolgert werden, dass ein Datensatz mit Daten aus verschiedenen Quellen auch zu einer besseren Klassifikation führt. Mit dem Fokus auf Datenqualität wurde erkannt, dass man mit unbalancierten Datensätzen trainieren muss, da die Anzahl an bekannten, validierten Datenpunkten von 2D6-Inhibitoren begrenzt ist. Umgekehrt führt die Reduktion von Nicht-Inhibitoren, um einen balancierten Datensatz zu erhalten, zu einem ungewollten Informationsverlust, da normalerweise mehr Informationen zu den Nicht-Inhibitoren vorliegen. Analog zur Anzahl der Trainingsinstanzen verhält sich die Reduktion von Eigenschaften, bei der irrelevante Deskriptoren zu einer verschlechterten Klassifikation führen. Wir haben deshalb unsere in den vorangegangenen Teilen beschriebenen Verfahren angewendet, um die metabolischen Eigenschaften von virtuellen Molekülen schnell vorherzusagen bzw. neue Moleküle mit gewünschtem metabolischem Verhalten zu synthetisieren.

2 Beschreibung der verwendeten Daten und die Vorgehensweise

Um unsere Algorithmen anwenden zu können, ist eine Darstellung in Feature-Vektoren notwendig, bei der jede chemische Struktur durch mehrere hundert physikochemische Deskriptoren oder Substruktureigenschaften, die in so genannten Fingerprints dargestellt werden, beschrieben wird. Ein Feature-Vektor ist eine normalisierte Darstellung der chemischen Struktur durch eine Anzahl von Eigenschaften, die unabhängig von der Anzahl der Atome ist. Im zweiten Schritt werden die Informationen zur 2D6-Inhibition hinzugefügt.

Wir wiederholen diesen Schritt für jedes Molekül im Datensatz, so dass wir letztendlich eine zweidimensionale Beziehung zwischen den Eigenschaften und der 2D6-Inhibition herstellen können.

Im Gegensatz zur visuellen Analyse der Klassifikationsergebnisse ist es für das Erstellen der Klassifikatoren notwendig, die zugrundeliegenden chemischen Strukturen, wie in Teil I näher beschrieben, durch physikochemische Eigenschaften zu repräsentieren. Zu diesem Zweck haben wir in unserem Fall das MOE (Molecular operating environment; chemical computing group) Programmpaket eingesetzt [13]. Zusammenfassend haben wir so 557 Deskriptoren zu unseren Molekülen berechnet, wie z.B. die Anzahl von bestimmten Atomtypen, Indizes von molekularen Graphen, topologische Deskriptoren, Autokorrelationen, Deskriptoren zur Aromatizität sowie 3D-Deskriptoren, die die Van der Waals Oberfläche der Moleküle in Kombination mit physikochemischen Eigenschaften enthalten.Für die Berechnung der 3D-Koordinaten der chemischen Strukturen haben wir CORINA (Arbeitsgruppe Prof. Gasteiger, Molecular Networks) verwendet. Die Berechnung der binären Vektoren, die eine Länge von 458 Bits haben, wurde mit dem Programm CACTVS (Dr. Ihlenfeldt, Xemistry) durchgeführt. Insgesamt wurden auf diese Weise vier Datensätze erzeugt. Die Ensemble-Datensätze (jeweils ein Training- und Testdatensatz) enthalten pro Molekül eine Zeile mit den entsprechenden Eigenschaften des gesamten Moleküls. Die Binärdatensätze (jeweils ein Training- und Testdatensatz) enthalten alle auf Substrukturen basierende Fragmente in binärer Form.

Auf der Basis eines Datensatzes mit 263 chemisch unterschiedlichen Wirkstoffen, die wir aus publizierten Daten extrahiert haben, wurde ein Trainingsdatensatz mit 185 Substanzen und ein Testdatensatz mit 78 Substanzen für die Validierung gewonnen. Weil Daten aus verschiedenen Quellen abhängig von der experimentellen Methode sind, mit denen sie erzeugt wurden, haben wir alle Substanzen mit mikromolarer Aktivität als Inhibitoren definiert. Für unseren Datensatz haben wir nur Substanzen ausgewählt, von denen entweder die Inhibition publiziert ist oder diese so weit charakterisiert sind, dass eine 2D6-Inhibition ausgeschlossen werden kann. Aus diesem Grund haben wir diejenigen Substanzen als Nicht-Inhibitoren festgelegt, bei denen die CYP2D6-Inhibition als unwahrscheinlich einzustufen ist. Der Trainingsdatensatz mit 185 Substanzen enthält 35 Substanzen, die die CYP2D6-Isoform inhibieren und 150 Nicht-Inhibitoren. Wir beziehen uns im ersten Fall auf die Klasse 1 und bei den negativen Trainingspunkten auf die Klasse 0. Der Testdatensatz enthält die gleichen Anteile von positiven und negativen Punkten wie der Trainingsdatensatz. Für die Oversampling-Methode [9] haben wir die Anzahl an Positiven im Trainingsdatensatz durch Wiederholungen der Feature-Vektoren verdreifacht. Der Testdatensatz wurde über alle angewendeten Methoden konstant gehalten. Der Split von Test- und Trainingsdatensatz wurde auf Basis von Tanimoto-Koeffizienten durchgeführt, die ein Maß für die Ähnlichkeit darstellen [7].

Die so erzeugten Datensätze wurden dann von redundanten Informationen, wie z.B. Nullspalten befreit. Zusätzlich wurde bei einem Teil der Datensätze die Anzahl der Merkmale mit dem im Teil II beschriebenen Variablenselektionsverfahren nach McCabe reduziert.

Für jeden Datensatz wurde anschließend die SVM-Methode mit unseren eigenen Kernen angewendet. Im Falle der Ensemble-Datensätze liegt eine Kombination von numerischen und kategorischen Eigenschaften vor. Die Erstellung der Modelle und die Vorhersagen wurden auf Basis von ASCII-Dateien durchgeführt. Die Optimierung der SVM-Parameter wurde durch eine Gridsuche realisiert, die im Algorithmus implementiert ist. Die Erzeugung der CYP2D6-Modelle beanspruchte auf einem PC wenige Sekunden pro Lauf während der Optimierung. Nachdem der Algorithmus trainiert war, konnten die Substanzen aus dem Testdatensatz klassifiziert werden. Die Klassifikation wurde dann anhand von Genauigkeitsmaßen bewertet [15].

3 Methoden für unbalancierte Trainingsdatensätze

Im Fall von unbalancierten Datensätzen überwiegt die Anzahl der negativen Trainingsbeispiele im Vergleich zu der Anzahl der positiven Beispiele. Zusätzlich zu der unbalancierten Klassenverteilung muss der Klassifikationsfehler berücksichtigt werden, der bei einer falsch negativen Vorhersage ungünstiger ist als bei einer falsch positiven Vorhersage [8]. Normalerweise sind die Ergebnisse von Verfahren des Maschinellen Lernens für unbalancierte Datensätze nicht überzeugend, da eine Tendenz zur Klassifikation neuer Substanzen in die überrepräsentierte Klasse besteht. Aus diesem Grund ist es notwendig, den Datensatz oder den Algorithmus so zu verändern, dass die unterrepräsentierte Klasse gleichwertig berücksichtigt wird. Innerhalb des SVM-Algorithmus kann das z.B. durch die Einführung eines Gewichtungsschemas erreicht werden. Dadurch wird allerdings die Datenverteilung nicht verändert und führt bei verrauschten, unbalancierten Datensätzen unter Umständen zu übertrainierten Modellen. Unbalancierte Datensätze können entweder durch Reduktion der überrepräsentierten Klasse bzw. durch ein wiederholtes Training von Datenpunkten der unterrepräsentierten Klasse modifiziert werden. Die Methode der Datenreduktion führt allerdings auch dazu, dass validierte negative Trainingsinformation nicht genutzt wird und damit die Generalisierungsfähigkeit des Modells reduziert wird. Aus diesem Grund haben wir die sogenannte Oversampling-Methode verwendet und die unterrepräsentierte Klasse dreifach trainiert. Die Klassifikationsergebnisse wurden anschließend noch durch Schwellwertverschiebung angepasst. Dabei wird der Schwellwert für die überrepräsentierte Klasse angehoben, damit eine Zuordnung von Testinstanzen zu dieser Klasse erschwert wird.

4 Diskussion der Ergebnisse

Nachfolgend werden die Ergebnisse mit unseren verwendeten Algorithmen diskutiert. Im Fall der Ensemble-Datensätze haben wir eine Kombination von numerischen und kategorischen Eigenschaften vorliegen. Im Fall der Binärdatensätze sind die vorhandenen strukturellen Merkmale durch sogenannte Fingerprints beschrieben, in denen die Anwesenheit ei-

ner strukturellen Einheit durch “1” bzw. bei Abwesenheit durch “0” repräsentiert wird. Um den Einfluss einer Merkmalsauswahl zu untersuchen, haben wir mit jeweils 5, 10 oder 20 bzw. allen berechneten Eigenschaften Modelle erstellt. Die Parameteroptimierung haben wir durch eine Gridsuche über einen jeweils methodenspezifischen Parameterraum durchgeführt. Die Schwellwertverschiebung wurde nach der Klassifikation mit einem Wert von 0.5 für die SVM durchgeführt. Die Erstellung der Modelle umfasst zunächst die Bestimmung der optimalen Parameter für den Algorithmus durch wiederholte 10-fache Kreuzvalidierung. Die daraus erhaltenen optimalen Parameterkombinationen wurden für die finale Erstellung des Klassifikators eingesetzt. Im nachfolgenden Schritt wurde dann ein Testdatensatz verwendet, der zu keiner Zeit im Optimierungszyklus eingesetzt wurde. Zur Diskussion der Ergebnisse wurden die Sensitivität (Hits) und die Anzahl der falsch positiven Klassifikationsergebnisse (falscher Alarm) herangezogen. Im Fall der SVM mit Gauß-Kern konnte eine dramatisch erhöhte Sensitivität beobachtet werden, die jedoch durch zusätzliche falsch positive Punkte beeinflusst wurde. Hingegen ergab die Oversampling-Methode keine deutliche Verbesserung der Sensitivität. Wurde jedoch zusätzlich der Schwellwert verschoben, wurden für alle Datensätze bessere Ergebnisse erzielt. Die besten Resultate ergab dann eine zusätzliche Merkmalsauswahl von fünf Eigenschaften. Im Vergleich dazu konnte mit dem Slater-Kern im Standardfall keine Verbesserung der Sensitivität erreicht werden. Allerdings war dort die Spezifität erhöht und zusammen mit einer Schwellwertverschiebung konnten bis zu 92% der 2D6-Inhibitoren im Testdatensatz erkannt werden. Die weitere Verwendung von Oversampling war ohne Einfluss auf die Klassifikationsergebnisse. Mit steigender Anzahl der Eigenschaften konnte ein Anstieg der Sensitivität beobachtet werden, der jedoch ebenfalls auf Kosten der Spezifität ging. Zusammenfassend kann gesagt werden, dass höhere Spezifität zu einer dramatischen Verringerung der Sensitivität geführt hat. Der Einfluss der Oversampling-Methode auf die Klassifikationsergebnisse führte zu geringeren Raten falschen Alarms, aber im Gegensatz zu unseren Annahmen war damit auch immer eine Abnahme der Sensitivität verbunden. Für die Klassifikation der Binärdatensätze haben wir den Tanimoto-Kern getestet. Besonders auffällig war die damit erzielte geringe Sensitivität im Fall der Datensätze mit 5 und 10 Eigenschaften. Da sich dieser Befund mit 20 beziehungsweise allen Eigenschaften verbesserte, kann abgeleitet werden, dass die Anzahl der verwendeten Merkmale entscheidend ist, um gute Klassifikationsergebnisse zu erhalten. Eine Merkmalsauswahl sollte deshalb bei Fingerprint-basierten Datensätzen nur vorsichtig eingesetzt werden. Die Oversampling-Methode erhöhte die Sensitivität zusammen mit geringer Abnahme der Spezifität. Nachfolgend werden die erhaltenen Ergebnisse auf Basis der chemischen Strukturen und selektierten Eigenschaften diskutiert. Basierend auf der unüberwachten Methode der Merkmalsauswahl nach McCabe selektierten wir die 20 besten Deskriptoren aus dem Ensembledatensatz. Die wichtigsten sind in Tabelle 1 mit der Beschreibung ihrer Bedeutung zusammengefasst. Die entsprechend selektierten Deskriptoren passen zu bereits in der Literatur beschriebenen Pharmakophor-, und Homologiemodellen. Diese lassen sich in Eigenschaften zusammenfassen, die die räumliche Gestalt und die Verknüpfung innerhalb der Moleküle beschreiben, wie z.B. die

chi0, chi0_C	Konnektivitätsindizes, die die Verknüpfung im Kohlenstoffgerüst beschreiben
TPSA, PEOE_VSA_HYD PEOE_PC+, Q_VSA_POS	Beschreibung der polaren und hydrophoben Anteile der Molekülоberfläche auf Basis der Gasteiger-Marsili-Ladungen
a_nN, b_single, MACCS Keys	Anzahl der N, S Atome, Einfachbindungen, SO Gruppen, Ringatome, Nitrogruppen, Heteroatome in 5-Ringen, Hydroxygruppen, Dreifachbindungen
SMR, SMR_VSA0 SMR_VSA7 bin 7	Molekularer Brechungsindex und deren Anteile
kS_sI	Elektrotopologische Eigenschaft von funktionellen Gruppen

Tabelle 1: Wichtige durch das McCabe-Verfahren identifizierte Eigenschaften.

Konnektivitätsindizes chi0 und chi0_C. Zusätzlich wurden einige Bins aus dem 166-Bin MACCS-Vektor, die Anzahl von bestimmten funktionellen Gruppen sowie elektrotopologische Indizes, die die Ladungsverteilung innerhalb der Moleküle beschreiben, selektiert. Zusammenfassend kann gesagt werden, dass die Merkmalsauswahl nur 2D-Eigenschaften selektiert hat, die somit die Basis unserer Modelle mit Merkmalsauswahl darstellen. Im Vergleich über alle verwendeten Methoden und Kerne stellte dies allerdings keine Einschränkung für die letztendlich erreichte Qualität und Genauigkeit der Modelle dar.

Abbildung 4 zeigt eine repräsentative Auswahl von diversen Verbindungen aus dem Trainingsdatensatz zusammen mit der CYP2D6-Inhibition. Wie aus dem direkten Strukturvergleich von Amitriptyline und Imipramine zu sehen ist, können kleine Abweichungen in der chemischen Struktur zu einer unterschiedlichen Klassifikation führen. Ein weiteres Beispiel dieser Art ist die gleiche Anzahl der Bindungen zwischen dem aromatischen Zentrum und dem Stickstoffatom in Venlafaxine und Dexfenfluramine aber unterschiedlicher CYP2D6-Inhibition. In den Abbildungen 1, 2 und 3 haben wir Verbindungen aus unserem Testdatensatz ausgewählt, die einige der gefundenen Unterschiede in der Klassifikation der verwendeten Algorithmen aufzeigen.

Verbindungen wie Yohimbine, Desmethylsertraline und Prevastatin wurden überwiegend richtig klassifiziert. Im Gegensatz dazu wurden Clemastine, Fentanyl und Perazine hauptsächlich falsch klassifiziert. Dieser Befund kann zum Teil durch einen Vergleich von Imi-

pramine aus dem Trainingsdatensatz und dem falsch negativ klassifizierten Perazine aus dem Testdatensatz erklärt werden. Beide Strukturen sind chemisch sehr ähnlich, aber unterscheiden sich in einem zusätzlichen Schwefelatom. Ein ähnliches Phänomen tritt bei den falsch positiven Clemastine und Fentanyl auf, bei denen Eigenschaften der 2D6-Inhibitoren aus dem Trainingsdatensatz wie O-N Abstände in Venlafaxine und Clemastine, zu einer falschen Klassifikation aufgrund einer funktionellen Untereinheit führen. Für eine Anzahl von Substanzen ist die Art der im Algorithmus verwendeten Eigenschaften von Bedeutung. Insbesondere die korrekte Klassifizierung von Nefazodone gelang mit Hilfe der numerischen Ensemble-Eigenschaften im Gegensatz zum Binärdatensatz, der zu einer falschen Klassifikation führte. Umgekehrt kann man aber auch Beispiele wie das Metoprolol finden, in denen der Binärdatensatz besser zur Klassifikation geeignet war. Die meisten der restlichen richtig klassifizierten Substanzen können auf Basis ihrer vergleichbaren Ähnlichkeit erklärt werden. Im Verlauf der Untersuchungen an diesen Datensätzen konnte generell festgestellt werden, dass nur diejenige Information im Testset abrufbar ist, die auch im Trainingsdatensatz enthalten ist. Die Möglichkeiten der Klassifikation sind durch den aufgespannten chemischen Raum der Trainingsdaten limitiert. Diese Aussage gilt unabhängig vom Algorithmus, der Art wie der Datensatz erstellt wurde und durch welche physikochemischen Eigenschaften die chemischen Strukturen beschrieben werden.

5 Zusammenfassung und Ausblick

Die humane Cytochrom P450 2D6-Isoform spielt eine Schlüsselrolle im Metabolismus von vielen neuen Wirkstoffen in der präklinischen Forschung. Wir haben einen Datensatz aus publizierten Quellen extrahiert und physikochemische Eigenschaften der Substanzen mit Methoden aus der Chemoinformatik berechnet. Auf der Basis dieser Daten haben wir mit unseren beschriebenen Methoden aus den vorangegangenen Teilen II und III Klassifikatoren erstellt. Zur Anwendung des Maschinellen Lernens haben wir unsere eigene Implementierung einer Support-Vektor-Maschine mit neuen Kernen eingesetzt. Die im Teil II beschriebene unüberwachte McCabe-Methode wurde dabei zur Merkmalsauswahl als vorgeschalteter Schritt für das überwachte Lernen eingesetzt. Nachfolgend wurde eine Analyse der klassifizierten Strukturen gezeigt und mit Beispielen aus dem Trainingsdatensatz verglichen. Es konnte gezeigt werden, dass die von uns umgesetzten Algorithmen eine effiziente Methode zur in silico Klassifikation von potentiellen neuen Wirkstoffen darstellen. Insbesondere die Verwendung von Mehrfachtraining und Schwellwertverschiebung in unseren CYP2D6-Datensätzen mit ungünstigem aktiv/nicht-aktiv Verhältnis führten so zu hochgenauen und sensiblen Klassifikatoren. Die eingesetzte Variablenselektion ergab Eigenschaften, die auf dem molekularen Level interpretiert werden können. Zusätzlich zu den bereits beschriebenen SVM-Methoden haben wir begonnen, die Maximum-Entropy-Methode (ME) [14], insbesondere den GIS-Algorithmus in Kombination mit Oversampling, zur Klassifikation einzusetzen. Diese wird normalerweise im Bereich der Spracher-

kennung genutzt. Dazu verwenden wir zur Zeit die Implementierung der frei verfügbaren openNLP-Software (sf.net/opennlp). Die ME-Methode ist ein elegantes Klassifikationsverfahren, welches auf der Tatsache beruht, dass Informationen über die Wahrscheinlichkeitsverteilung im Trainingsdatensatz bekannt sind und deshalb eine Verteilung mit der größten Entropie (nach Shannon) bestimmt werden kann. Erste Ergebnisse am Beispiel des CYP2D6-Datensatzes sind ebenfalls vielversprechend, so dass in Zukunft vergleichende Analysen vorgenommen werden.

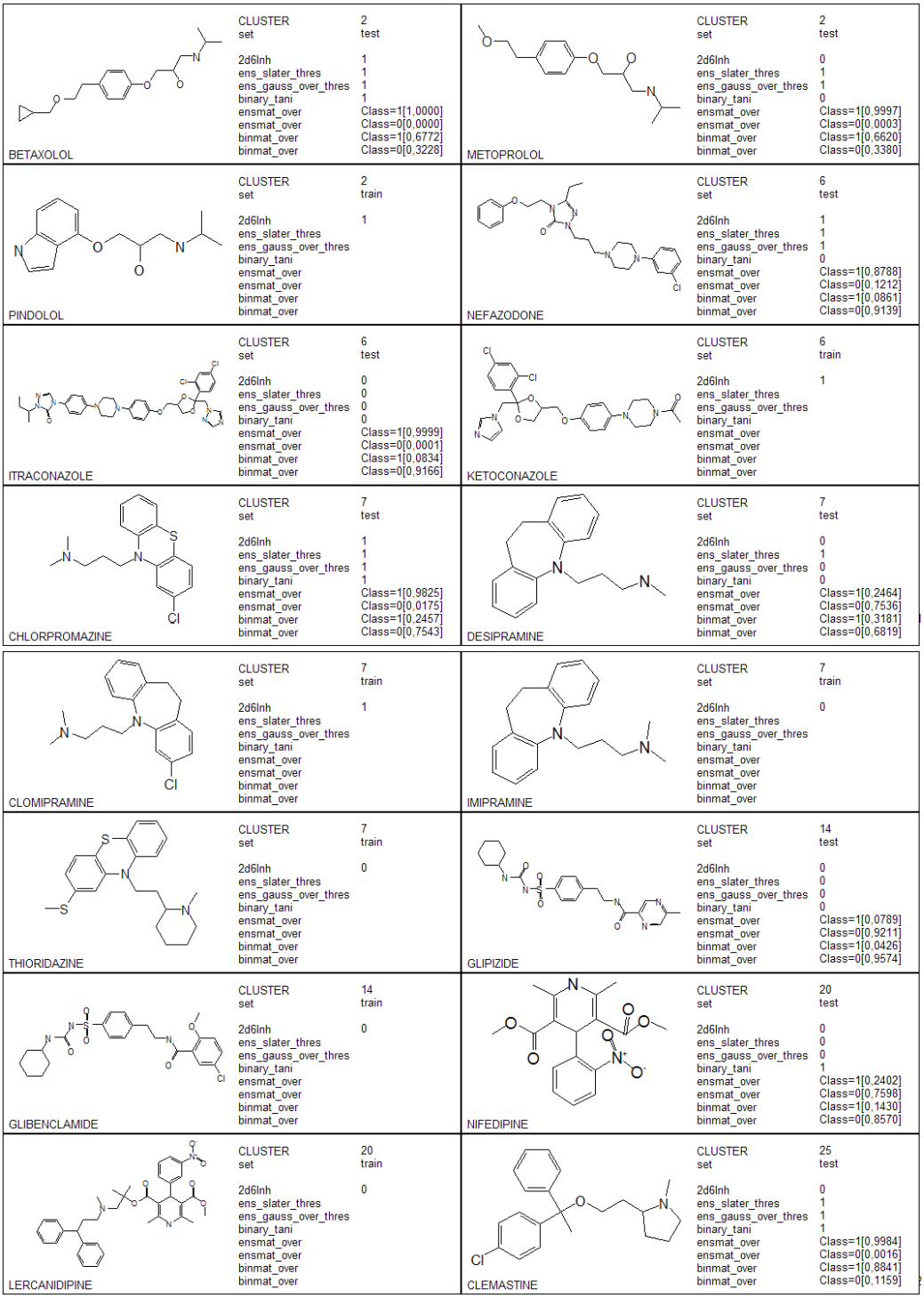


Abbildung 1: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (I).

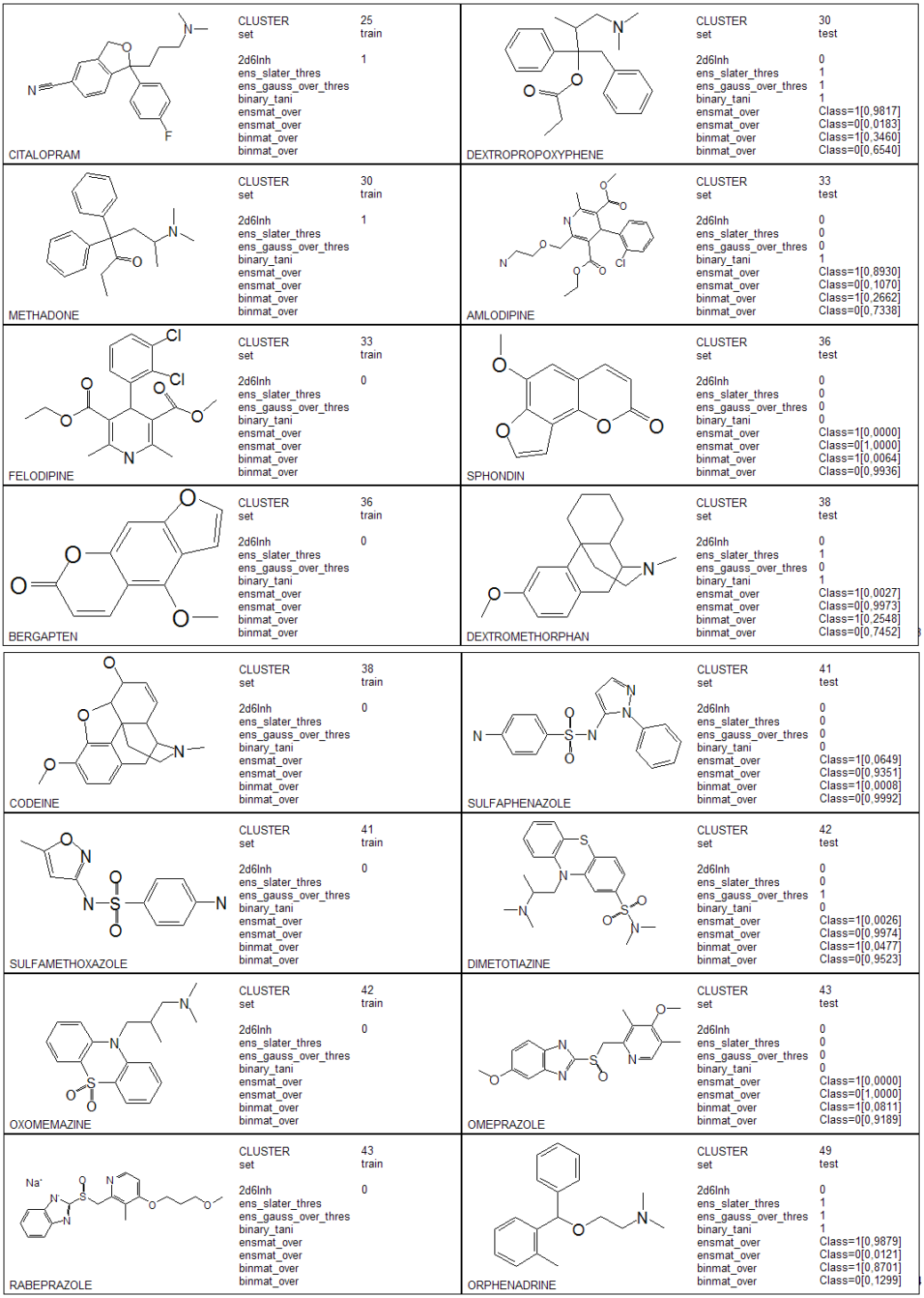


Abbildung 2: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (II).

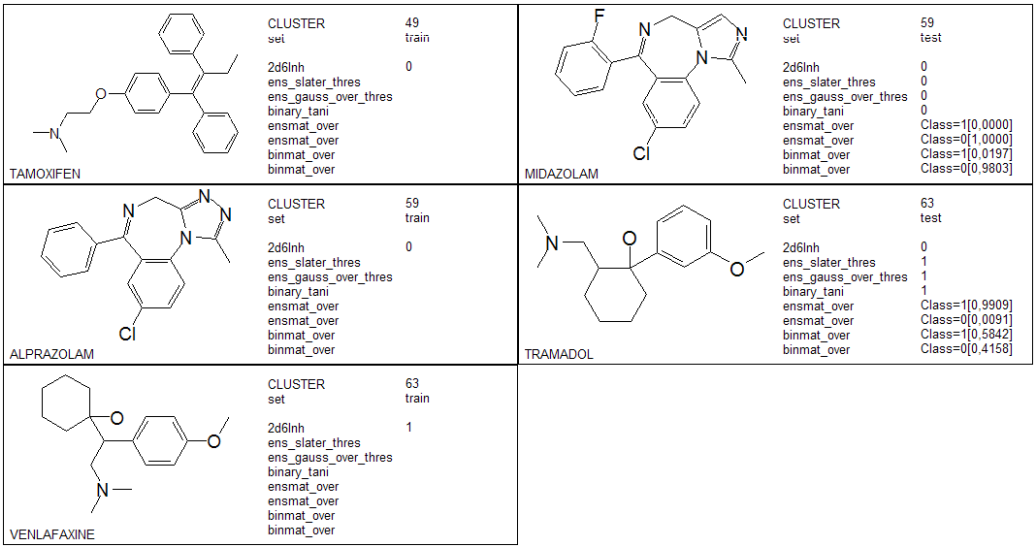


Abbildung 3: Ähnlichkeitsvergleich zwischen Test- und Trainingsdatensatz mit den Klassifikationsergebnissen (III).

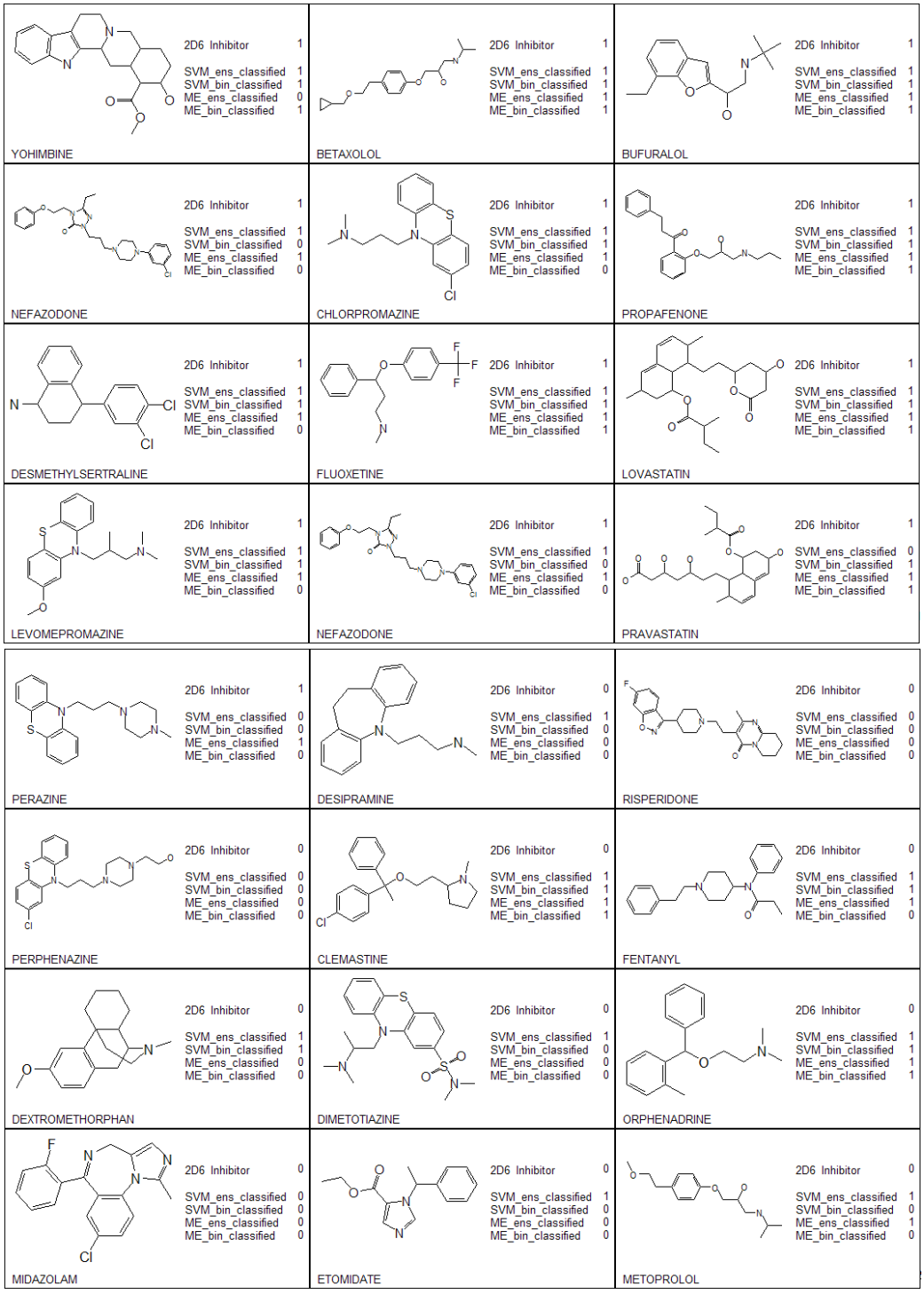


Abbildung 4: Ergebnisse aus dem Testdatensatz im Vergleich mit den verwendeten Methoden.

Literaturverzeichnis

[1] D. F. Lewis, S. Modi, M. Dickins. *Structure-activity relationship for human cytochrome P450 substrates and inhibitors.* Drug Metab. Rev. 2002, 34(1-2):69–82.

[2] C. de Graaf, N. P. Vermeulen, K. A. Feenstra. *Cytochrome P450 in silico: an integrative modeling approach.* J. Med. Chem. April 21, 2005, 48(8):2725–55.

[3] M. J. de Groot, S. Ekins. *Pharmacophore modeling of cytochromes P450.* AdV. Drug DeliVery ReV. 2002, 54(3):367–383.

[4] C. A. Kemp, J. U. Flanagan, A. J. van Eldik, J.-D. Marechal, C. R. Wolf, G. C. K. Roberts, M. J. I. Paine, M. J. Sutcliffe. *Validation of model of cytochrome P450 2D6: an in silico tool for predicting metabolism and inhibition.* Journal of Medicinal Chemistry, 2004, Vol. 47, No. 22.

[5] S. Ekins, J. Berbaum, R. K. Harrison. *Generation and validation of rapid computational filters for CYP2D6 and CYP3A4.* Drug Metab. Dispos. 2003, 31(9):1077–1080.

[6] J. M. Kriegl, T. Arnhold, B. Beck, T. Fox. *Prediction of human cytochrome P450 inhibition using support vector machines.* QSAR & Combinatorial Science 24, 2005, 491–502.

[7] P. Willett, J. M. Barnard, G. M. Downs. *Chemical similarity searching.* J. Chem. Inf. Comput. Sci. 1998, 38(6):983–996.

[8] S. Lessmann. *Solving unbalanced classification problems with support vector machines.* Proceedings of the International Conference on Artificial Intelligence, IC-AI'04 2004, 214–220.

[9] G. Rajarshi, P. C. Jurs. *Determining the validity of a QSAR model - a classification approach.* J. Chem. Inf. Model. 2005, Vol. 45, No. 1, 73.

[10] S. Rendic, F. J. Di Carlo. *Human cytochrome P450 enzymes: a status report summarizing their reactions, substrates, inducers and inhibitors.* Drug Metabolism Reviews 1997, 29, 413–580.

[11] S. Rendic. *Summary of information on human CYP enzymes: human P450 metabolism data.* Drug Metab. Rev. 2002, 34(1-2):83–448.

[12] D. Flockhart. *Cytochrome P450 drug interaction table.* `http://medicine.iupui.edu/flockhart`

[13] MOE (The Molecular Operating Environment) Version 2005.06. Software available from Chemical Computing Group Inc., 1010 Sherbrooke Street West, Suite 910, Montreal, Canada H3A 2R7. `http://www.chemcomp.com`

[14] A. Ratnaparkhi. *A simple introduction to maximum entropy models for natural language processing.* NSF Science and Technology Center for Research in Cognitive Science, Technical Report, 1997, 97–108.

[15] T. Eitrich, A. Kless, C. Druska, W. Meyer, J. Grotendorst. *Classification of highly unbalanced CYP450 data of drugs using cost sensitive machine learning techniques.* Journal of Chemical Information and Modeling, 2007 (first issue).

Bisher sind erschienen:

Modern Methods and Algorithms of Quantum Chemistry - Proceedings
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 1
ISBN 3-00-005618-1, Februar 2000, 562 Seiten
nicht mehr lieferbar

Modern Methods and Algorithms of Quantum Chemistry - Poster Presentations
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 2
ISBN 3-00-005746-3, Februar 2000, 77 Seiten
nicht mehr lieferbar

Modern Methods and Algorithms of Quantum Chemistry - Proceedings, Second Edition
Johannes Grotendorst (Hrsg.)
Winterschule, 21. - 25. Februar 2000, Forschungszentrum Jülich
NIC-Serie Band 3
ISBN 3-00-005834-6, Dezember 2000, 638 Seiten
nicht mehr lieferbar

Nichtlineare Analyse raum-zeitlicher Aspekte der hirnelektrischen Aktivität von Epilepsiepatienten
Jochen Arnold
NIC-Serie Band 4
ISBN 3-00-006221-1, September 2000, 120 Seiten

Elektron-Elektron-Wechselwirkung in Halbleitern: Von hochkorrelierten kohärenten Anfangszuständen zu inkohärentem Transport
Reinhold Lövenich
NIC-Serie Band 5
ISBN 3-00-006329-3, August 2000, 146 Seiten

Erkennung von Nichtlinearitäten und wechselseitigen Abhängigkeiten in Zeitreihen
Andreas Schmitz
NIC-Serie Band 6
ISBN 3-00-007871-1, Mai 2001, 142 Seiten

Multiparadigm Programming with Object-Oriented Languages - Proceedings
Kei Davis, Yannis Smaragdakis, Jörg Striegnitz (Hrsg.)
Workshop MPOOL, 18. Mai 2001, Budapest
NIC-Serie Band 7
ISBN 3-00-007968-8, Juni 2001, 160 Seiten

Europhysics Conference on Computational Physics - Book of Abstracts
Friedel Hossfeld, Kurt Binder (Hrsg.)
Konferenz, 5. - 8. September 2001, Aachen
NIC-Serie Band 8
ISBN 3-00-008236-0, September 2001, 500 Seiten

NIC Symposium 2001 - Proceedings
Horst Rollnik, Dietrich Wolf (Hrsg.)
Symposium, 5. - 6. Dezember 2001, Forschungszentrum Jülich
NIC-Serie Band 9
ISBN 3-00-009055-X, Mai 2002, 514 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Lecture Notes
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 10
ISBN 3-00-009057-6, Februar 2002, 548 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Poster Presentations
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 11
ISBN 3-00-009058-4, Februar 2002, 194 Seiten

Strongly Disordered Quantum Spin Systems in Low Dimensions: Numerical Study of Spin Chains, Spin Ladders and Two-Dimensional Systems
Yu-cheng Lin
NIC-Serie Band 12
ISBN 3-00-009056-8, Mai 2002, 146 Seiten

Multiparadigm Programming with Object-Oriented Languages - Proceedings
Jörg Striegnitz, Kei Davis, Yannis Smaragdakis (Hrsg.)
Workshop MPOOL 2002, 11. Juni 2002, Malaga
NIC-Serie Band 13
ISBN 3-00-009099-1, Juni 2002, 132 Seiten

Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms - Audio-Visual Lecture Notes
Johannes Grotendorst, Dominik Marx, Alejandro Muramatsu (Hrsg.)
Winterschule, 25. Februar - 1. März 2002, Rolduc Conference Centre, Kerkrade, Niederlande
NIC-Serie Band 14
ISBN 3-00-010000-8, November 2002, DVD

Numerical Methods for Limit and Shakedown Analysis
Manfred Staat, Michael Heitzer (Hrsg.)
NIC-Serie Band 15
ISBN 3-00-010001-6, Februar 2003, 306 Seiten

Design and Evaluation of a Bandwidth Broker that Provides Network Quality of Service for Grid Applications
Volker Sander
NIC-Serie Band 16
ISBN 3-00-010002-4, Februar 2003, 208 Seiten

Automatic Performance Analysis on Parallel Computers with SMP Nodes
Felix Wolf
NIC-Serie Band 17
ISBN 3-00-010003-2, Februar 2003, 168 Seiten

Haptisches Rendern zum Einpassen von hochaufgelösten Molekülstrukturdaten in niedrigaufgelöste Elektronenmikroskopie-Dichteverteilungen
Stefan Birmanns
NIC-Serie Band 18
ISBN 3-00-010004-0, September 2003, 178 Seiten

Auswirkungen der Virtualisierung auf den IT-Betrieb
Wolfgang Gürich (Hrsg.)
GI Conference, 4. - 5. November 2003, Forschungszentrum Jülich
NIC-Serie Band 19
ISBN 3-00-009100-9, Oktober 2003, 126 Seiten

NIC Symposium 2004
Dietrich Wolf, Gernot Münster, Manfred Kremer (Hrsg.)
Symposium, 17. - 18. Februar 2004, Forschungszentrum Jülich
NIC-Serie Band 20
ISBN 3-00-012372-5, Februar 2004, 482 Seiten

Measuring Synchronization in Model Systems and Electroencephalographic Time Series from Epilepsy Patients
Thomas Kreutz
NIC-Serie Band 21
ISBN 3-00-012373-3, Februar 2004, 138 Seiten

Computational Soft Matter: From Synthetic Polymers to Proteins - Poster Abstracts
Norbert Attig, Kurt Binder, Helmut Grubmüller, Kurt Kremer (Hrsg.)
Winterschule, 29. Februar - 6. März 2004, Gustav-Stresemann-Institut Bonn
NIC-Serie Band 22
ISBN 3-00-012374-1, Februar 2004, 120 Seiten

Computational Soft Matter: From Synthetic Polymers to Proteins - Lecture Notes
Norbert Attig, Kurt Binder, Helmut Grubmüller, Kurt Kremer (Hrsg.)
Winterschule, 29. Februar - 6. März 2004, Gustav-Stresemann-Institut Bonn
NIC-Serie Band 23
ISBN 3-00-012641-4, Februar 2004, 440 Seiten

Synchronization and Interdependence Measures and their Applications to the Electroencephalogram of Epilepsy Patients and Clustering of Data
Alexander Kraskov
NIC-Serie Band 24
ISBN 3-00-013619-3, Mai 2004, 106 Seiten

High Performance Computing in Chemistry
Johannes Grotendorst (Hrsg.)
Bericht des Verbundprojekts:
High Performance Computing in Chemistry - HPC-Chem
NIC-Serie Band 25
ISBN 3-00-013618-5, Dezember 2004, 160 Seiten

Zerlegung von Signalen in unabhängige Komponenten: Ein informationstheoretischer Zugang
Harald Stögbauer
NIC-Serie Band 26
ISBN 3-00-013620-7, April 2005, 110 Seiten

Multiparadigm Programming 2003
Joint Proceedings of the
3rd International Workshop on Multiparadigm Programming with Object-Oriented Languages (MPOOL'03)
and the
1st International Workshop on Declarative Programming in the Context of Object-Oriented Languages (PD-COOL'03)
Jörg Striegnitz, Kei Davis (Editors)
NIC-Serie Band 27
ISBN 3-00-016005-1, Juli 2005, 300 Seiten

Integration von Programmiersprachen durch strukturelle Typanalyse und partielle Auswertung
Jörg Striegnitz
NIC-Serie Band 28
ISBN 3-00-016006-X, Mai 2005, 306 Seiten

OpenMolGRID - Open Computing Grid for Molecular Science and Engineering
Final Report
Mathilde Romberg (Editor)
NIC-Serie Band 29
ISBN 3-00-016007-8, Juli 2005, 86 Seiten

Computational Nanoscience: Do It Yourself!
Lecture Notes
Johannes Grotendorst, Stefan Blügel, Dominik Marx (Hrsg.)
Winterschule, 14. - 22. Februar 2006, Forschungszentrum Jülich
NIC-Serie Band 31
ISBN 3-00-017350-1, Februar 2006, 528 Seiten

NIC Symposium 2006 - Proceedings
G. Münster, D. Wolf, M. Kremer (Hrsg.)
Symposium, 1. - 2. März 2006, Forschungszentrum Jülich
NIC-Serie Band 32
ISBN 3-00-017351-X, Februar 2006, 384 Seiten

Alle Bände stehen online zur Verfügung unter

www.fz-juelich.de/nic-series/.