UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

MÉTODOS DE SELECCIÓN DE ATRIBUTOS BASADOS EN UTILIDADES PARA LA PREDICCIÓN DE FUGA DE CLIENTES EN TELECOMUNICACIONES.

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN GESTIÓN DE OPERACIONES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO

ÁLVARO FLORES RÍOS

PROFESOR GUÍA:

RICHARD WEBER HAAS

MIEMBROS DE LA COMISIÓN:

AXEL OSSES ALVARADO
SEBASTIÁN MALDONADO ALARCÓN

SANTIAGO DE CHILE
2014

A mi familia, amigos y esposa

RESUMEN DE TESIS PARA
OPTAR AL GRADO DE MAGÍSTER
EN GESTIÓN DE OPERACIONES
POR: ALVARO FLORES RIOS
PROF. RICHARD WEBER

MÉTODOS DE SELECCIÓN DE ATRIBUTOS BASADOS EN UTILIDADES PARA LA PREDICCIÓN DE FUGA DE CLIENTES EN TELECOMUNICACIONES.

Hoy en día, respondiendo a la gran capacidad de almacenaje de datos, y procesamiento de éstos, existe un interés creciente en descubrir patrones en la información recabada y delegar problemas de negocio a procesos de aprendizaje automatizado, ayudando a tomar decisiones empresariales fundamentadas desde un sustento teórico, y con precisión que un conjunto de individuos no sería de capaz alcanzar dado el volumen de datos y la dificultad de descubrir de manera eficiente los patrones ocultos en las bases de datos.

Una rama importante de estos problemas de negocios, es la clasificación binaria y una problemática que ha despertado gran interés es la predicción de fuga de clientes, debido a la facilidad con que estos últimos pueden cambiarse de empresa de servicios, en particular en el sector de Telecomunicaciones. En este caso los modelos son orientados a identificar el conjunto de clientes con mayor tendencia a dejar su empresa de servicios actual, basándose en sus características individuales descritas en forma de atributos numéricos o cualitativos, como por ejemplo variables socio-demográficas, comportamiento de pago, etc..

Uno de los grandes desafíos en esta línea, es la selección de atributos para el modelo de clasificación. La mayoría de las técnicas de selección de atributos, sin embargo, son basadas en criterios de validación estadística, perdiendo en muchos casos el objetivo del negocio en si mismo, lo que no necesariamente lleva a modelos que optimicen las metas definidas por la empresa.

En este trabajo de Tesis se propone un enfoque basado en utilidades para la construcción del modelo de clasificación y selección de atributos, usando la herramienta de Support Vector Machines, en donde las métricas basadas en utilidades simulan la realización de una campaña de retención de clientes enfocada al grupo objetivo que el clasificador determine, considerando beneficios y costos (*Maximum Profit Criterion (MPC)* y *Expected Maximum Profit Criterion (EMPC)*) o bien sólo costos, como es el caso de *H-measure*. Este enfoque consiste en un método de selección de atributos empotrado en la construcción del modelo clasificador, que apunta a la eliminación secuencial de atributos removiendo los que tienen menor relevancia de acuerdo a las métricas basadas en utilidades recién descritas, reduciendo así la dimensionalidad del problema original.

Los resultados experimentales indican que estos métodos de selección de atributos y evaluación de modelos son más estables (al reducir atributos) y obtienen mejores resultados tanto en términos de métricas usuales de evaluación de modelos predictivos (AUC), como en métricas de desempeño basadas en utilidades orientadas al negocio de Telecomunicaciones (en el contexto de fuga de clientes). Lo que deja planteada la posibilidad de extender esta metodología a otros rubros, definiendo de forma conveniente las funciones de utilidades propias de cada negocio en particular.

AGRADECIMIENTOS

Es necesario hacer una pausa para agradecer a todas las personas que me acompañaron en esta hermosa travesía que fue mi estadía en la Universidad de Chile... Una etapa llena de vivencias que serían imposibles de reflejar en sólo una página, agradezco mucho a las buenas y hermosas personas que llenaron de cariño este viaje, pintándolo de alegrías y de un día a día que difícilmente pudo haber sido mejor. Se hace difícil identificar personas en un marco tan estrecho, y en ningún caso pretendo hacer una lista exhaustiva, así que usted señor lector si está leyendo esto y no lo mencioné, le pido mis sinceras disculpas.

Quiero partir agradeciendo a mi hermosa familia, que apoyó cada una de mis locuras desde el momento cero, y entendió que mi felicidad radica en el aprendizaje, y la superación. A mi padre, por esas largas conversaciones en la cocina, luego de compartir un plato de comida luego de un largo día. A mi madre, que es una fuente inagotable de cariño, una mujer admirable que pone en pausa cada vez que puede su felicidad por sobre la de otros. A mis hermanas, que son tres ángeles que me han tratado como un hijo más, y que son ejemplos de mujer en cada dimensión posible. A cada una de ellas les tengo un cariño muy especial, por sus diferencias inherentes. A María Elena, que contribuyó en mi formación y construcción de una personalidad curiosa, y autosuficiente. A Ruth, por ser la persona más ordenada del mundo y una de las personas que no son capaces de negar una mano en momentos de necesidad. A Claudia, quién con su carácter enérgico y alegre, inunda mi vida con ganas de superación, y es un ejemplo de fortaleza en mi día a día. A todos y cada uno de ustedes, los AMO.

Agradecer también, a mis amigos de infancia, eternos compañeros de juegos e historias, que forjaron en gran medida lo que soy ahora. Personas que son capaces de llevar al borde mis aptitudes, sin siquiera ser concientes de ello. Dentro de este hermoso conjunto, destaco a la gente de mi villa, Nahuelhuapi, o simplemente Nawer, que se convirtió con el paso del tiempo en una fábrica de excelentes personas, y que agradezco haber tenido la suerte de compartir gran parte de mi tiempo con ellas. A mi mejor amigo Mauro, por ser un pilar importantísimo en mi vida, una persona a mi parecer intachable, con un mindset muy especial, condicionado para ser una gran persona por definición. A este selecto grupo se suman demasiadas personas, que conocí a lo largo de mi vida académica y laboral, y que dejaron su huella en mí y saben que ocupan un lugar en mi mente y corazón... a todos y cada uno de ustedes, GRACIAS!.

En mi Universidad, al equipo de Basket (Slam Drunk y coach), a todos los funcionarios (incluyo al personal de las fotocopiadora y por supuesto a los negocios aledaños), profesores, y los innumerables amigos que conocí en esta estadía... A todos ustedes gracias por ser parte de mi formación y apoyarme cada vez que lo necesité.

Y si de agradecer se trata, amor, te doy las gracias por presentarte en esta etapa de mi vida, y hacer que la perfección no sea una palabra inalcanzable. Gracias por el apoyo y la comprensión en estos dos años de felicidad, por saber las justas palabras de motivación en la adversidad y sobre todo por acceder a forjar un futuro juntos. Daniela, esposa mía, Te amo.

A todos ustedes, les dedico esto. Que no es más que el principio, de un gran viaje. Gracias.

Tabla de Contenido

Capítulo 1

Introducción

La clasificación es una tarea relevante en muchas aplicaciones orientadas a mejorar las utilidades, tales como Credit Scoring o Predicción de Fuga de Clientes Baesens (2014), adicionalmente se ha mostrado que el desempeño de un clasificador puede ser mejorado enfocándose en los atributos más relevantes usados para la construcción de este. Esto a su vez tiene importantes ventajas:

a. Una representación de menor dimensionalidad en los atributos realza el poder predictivo de los modelos de clasificación disminuyendo su complejidad, reduciendo de esta manera el riesgo de *Overfitting (sobreajuste)*, causado por la *Maldición de la Dimensión* Guyon et al. (2006); Maldonado et al. (2011).

b. Permite una mejor interpretación del clasificador, lo que es particularmente importante en *Business Analytics*, puesto que muchos profesionales consideran que los enfoques de *Machine Learning* son cajas negras y se rehusan a emplear estos métodos debido a esto Carrizosa et al. (2011).

c. El entendimiento del proceso que generar los datos es también de crucial importancia en *Business Analytics* para la toma de decisiones, por ejemplo, identificar los atributos más relevantes de los cliente conlleva a un mejor entendimiento de las decisiones de éstos.

En este trabajo se propone abordar la problemática de fuga de clientes en telecomunicaciones, desde un punto de vista proactivo, generando un listado de clientes candidatos a ser contactados para una campaña de retención, maximizando las utilidades que esto genera, comparando las utilidades de un estado basal sin realizar ninguna acción comercial, versus realizar una campaña de retención focalizada en el conjunto de clientes que el clasificador determina como potenciales fugas. Esquemáticamente, esto se puede ver

reflejado en el siguiente diagrama:

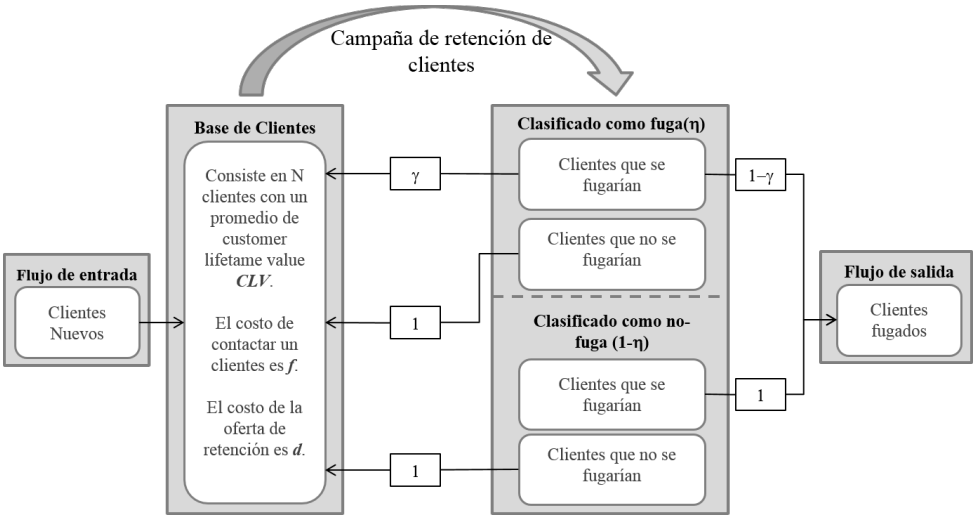

Figura 1.1: Esquema campaña de retención de clientes, (Fuente : Verbraken et al. (2012))

En la figura 1.1, se puede ver como el recuadro de **Base de Clientes** es dividido al realizar la campaña de retención en 4 grupos principales, dependiendo de su clasificación y respuesta a la oferta realizada por la empresa. Los valores señalados en los recuadros situados entre las flechas del flujo, corresponden a las probabilidades estimadas de cada uno de estos flujos, más explícitamente, γ es la tasa de éxito de la campaña ($1-\gamma$ la probabilidad que un cliente rechace la oferta), y el resto de los flujos son deterministas dada la naturaleza del problema[1]. Adicionalmente, el valor η corresponde a la fracción de clientes sobre la cuál se realiza la campaña de retención, valor que se determina maximizando las utilidades de la campaña.

Se propone un método de selección de atributos que se incluye en la construcción del modelo de clasificación en si mismo, usando la herramienta *Support Vector Machines* y proponiendo métricas de desempeño asociadas a utlidades que permiten discriminar las variables que más aportan a discernir de forma eficiente los clientes prospectos a fugarse de la compañía.

1.1. Objetivos

A continuación se detallan los objetivos perseguidos en este trabajo de Tesis, todos éstos en el contexto de Fuga de Clientes en Telecomunicaciones.

[1] A saber, un cliente que no pretendía fugarse y fue contactado, no se fugará, y los clientes no contactados por la campaña de retención seguirán su decisión, ya sea fugarse o mantenerse en la empresa según corresponda

1.1.1. Objetivo General

Proponer una metodología de construcción de modelos de clasificación, que use métricas de desempeño basadas en utilidades, tanto para la construcción del modelo, como para la selección de atributos, para el problema de fuga de clientes en Telecomunicaciones.

1.1.2. Objetivos Específicos

En aras de alcanzar objetivo general, se definen objetivos específicos, que representan segmentos importantes que se quieren lograr en este trabajo, y cuyo alcance colectivo constituyen el objetivo general. Los objetivos específicos propuestos son:

a. Probar que el uso de métricas de desempeño basadas en utilidades tanto para la evaluación de modelos de clasificación como para la selección de atributos, genera resultados competitivos respecto de las metodologías actuales

b. Probar estas métricas con diferentes configuraciones de Support Vector Machines y diferentes bases de clientes de Telecomunicaciones.

c. Comparar los desempeños obtenidos por las métricas usuales (en particular AUC) con las nuevas métricas basadas en utilidades.

d. Analizar cómo afecta en el rendimiento de los clasificadores las ténicas para corregir el desbalance de datos (Undersampling y Oversampling).

1.2. Alcances y Limitaciones

La cobertura de este trabajo de Tesis, tiene las siguientes restricciones:

a. Este trabajo de Tesis se enfoca en el desempeño final de los clasificadores en términos de indicadores clásicos en la literatura y las métricas definidas en este documento, y no en analizar la evolución de los conjuntos de atributos que son elegidos por el modelo luego de la seleción de parámetros

b. Se cuenta con 3 bases de datos de Telecomunicaciones para testear los modelos de clasificación propuestos

c. Se toman ciertos parámetros como conocidos, en particular los hiperpárametros de las distribuciones de probabilidad de aceptación de la campaña de retención (Verbraken et al. (2012))

1.3. Resultados Esperados

Se espera crear un método de selección de atributos, empotrado en la construcción del modelo, que obtenga un desempeño competitivo o mejor en términos de utilidades e indicadores habituales respecto de las metodologías usadas en el estado del arte.

1.4. Contribuciones y Publicaciones

Lo realizado en este trabajo de Tesis, fue finalmente enviado para su revisión a la ASOC (Applied Soft Computing), bajo el título *Profit-based Feature Selection using Support Vector Machines - General Framework and an Application for Customer Churn Prediction.*

1.5. Estructura de la Tesis

Para facilitar su lectura y comprensión, la estructura de este documento sigue la siguiente lógica: una vez terminado este capítulo introductorio, en el **Marco Teórico** (Capítulo 2), se revisan los tópicos utilizados para llevar a cabo este trabajo de Tesis, comenzando por la descripción de un problema de Clasificación Binaria e introduciendo los conceptos claves de KDD, Machine Learning, y Support Vector Machines con sus variantes y fijando notación para el resto del documento.

Luego, se revisan distintas métricas habituales de desempeño de clasificadores, y se introducen las nuevas métricas de desempeño ligadas a utilidades, que serán usadas posteriormente para la construcción y evaluación de los clasificadores considerados en este trabajo de Tesis.

Adicionalmente, en este capítulo se revisan los conceptos de *Undersampling* y *Oversampling*, como estrategias para manejar el fuerte desbalance de clases que presentan las bases de datos en cuestión.

Para cerrar este capítulo, se explican los métodos de selección de parámetros utilizados en este trabajo de Tesis, describiendo en detalle como funciona cada uno de ellos, y describiendo la construcción de los métodos de selección de atributos originales de este trabajo de tesis, que basan su paso de selección de atributos en las utilidades que generan al ser usados en un clasificador para predecir fuga de clientes.

En **Metodología** (Capítulo 3), se describe brevemente cómo fue aplicada la metodología KDD y adicionalmente se detalla paso a paso la forma de realizar los experimentos para comparar los diferentes clasificadores, y los algoritmos propuestos para selección de atributos, que usan a cabalidad las herramientas vistas en el Capítulo anterior.

Luego, en **Resultados** (Capítulo 4), se describen las bases de datos empleadas en este trabajo de Tesis, y posteriormente se realiza una vista general y análisis detallado a los resultados de los experimentos realizados según la metodología propuesta.

Finalmente, en **Conclusiones y Trabajo Futuro** (Capítulo 5), se pueden apreciar las principales conclusiones de este trabajo y potenciales alternativas de Trabajo Futuro.

Capítulo 2

Marco Teórico

Para poder abordar el problema detallado en la introducción, es necesario tener algunos conocimientos previos sobre metodologías existentes para trabajar bases de datos, y adicionalmente algunos elementos tanto de *Data Mining* como *Machine Learning*, en el presente capítulo, se comienza con una descricpción general de un problema de clasificación binaria, en la Sección 2.1, y explicando también la problemática puntual a tratar, que es la identificación de los clientes mas propensos a la fuga de una compañia de Telecomunicaciones. Una vez definida la problemática, en la Sección 2.2 se describe la metodología KDD (*Knowledge Discovery in Databases*) y posteriormente, en la Sección 2.3, se describe el concepto de *Machine Learning* y se ahonda precisamente en la descripción exhaustiva de la metodología usada en este trabajo, *Support Vector Machines*, para entender conceptualmente el uso de esta herramienta posteriormente en el diseño experimental (ver Capítulo 3). Luego de esto, se explican las medidas de rendimiento habituales, y se presentan las nuevas métricas de desempeño basadas en utlidades. El capítulo concluye con una breve vista al problema de desbalance de clases, y con una revisión de los métodos de selección de atributos que se usarán posteriormente.

2.1. Problema de Clasificación Binaria

En general, un problema de clasificación, se entiende como la identificación de personas, objetos o cualquier elemento generalizado, y tener la facultad de asignarle una etiqueta de acuerdo a su comportamiento o atributos (Kennedy et al. (2009)), y en función de esta clasificación obtener ventajas para entender cada uno o el conjunto de los objetos en cuestión.

Si adicionalmente, se tiene conocimiento del conjunto de clases existentes, y en cierto

modo cuáles son sus características o atributos que definen cada clase, es posible predecir, en cierta medida la clase de un nuevo objeto comparando sus atributos con los de las clases pre-existentes.

Lo anterior es utilizado en varios rubros, como telecomunicaciones o empresas crediticias, con el fin de obtener ventajas comparativas por sobre sus competidores y tomar medidas preventivas para evitar que los casos no deseados sucedan. A modo de ejemplo se puede mencionar que las empresas crediticias utilizan este tipo de enfoques para predecir si un cliente, pagará o no un crédito en el caso que éste sea concedido, comparando los atributos del cliente en cuestión, como sueldo, género, edad, profesión, etc.; con los atributos de clientes que con anterioridad realizaron una solicitud crediticia (Bravo et al. (2013)).

Del mismo modo que el ejemplo anteriormente descrito, este trabajo de investigación consiste en la utilización de conocimiento previo de un conjunto de observaciones para predecir el comportamiento de una nueva instancia (en este caso un cliente de Telecomunicaciones) en el futuro, en particular, si es un candidato a irse de la compañía o no.

Para resolver este tipo de problemáticas, es frecuente el uso de una función clasificadora, capaz de aprender del comportamiento de un conjunto de observaciones y etiquetar observaciones nuevas con una clase específica. Es por esto, que es necesario contar con datos históricos, ya que dicha función aprende de ellos para luego aplicar este "aprendizaje"sobre un nuevo objeto. Lo que será tratado en la sección 2.3.

2.2. Metodología KDD

Para poder obtener conclusiones útiles y válidas en minería de datos, es importante acompañar al proceso de análisis y minería de datos con una preparacón previa de los datos y posteriormente un análisis recabado de los resultados obtenidos. De esta manera, es posible catalogar a la minería de datos como parte de un proceso más amplio, que se puede ver en la Figura 2.1, que es conocido como extracción o descubrimiento de conocimiento en bases de datos (KDD, *Knowledge Discovery in Databases*, Fayyad (1996)). KDD es conocido como un campo altamente multidisciplinario, en donde las principales áreas de impacto son el aprendizaje automático (Sección 2.3), la estadística, y las bases de datos.

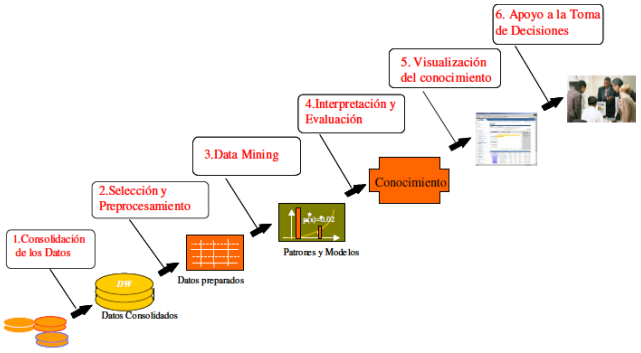

Figura 2.1: Proceso KDD

Como se puede observar en la figura 1, el proceso KDD está compuesto por varias etapas como se puede ver en Fayyad (1996); Ruiz Sánchez (2006), comenzando por la consolidación de la base de datos hasta la toma de decisiones per se. A continuación se especifican los aspectos más relevantes bajo el contexto de este trabajo de investigación.

1. **Recopilación y consolidación de datos:** Para poder utilizar los algoritmos de minería de datos, es necesario en primera instancia construir el set de datos correspondiente. Adicionalmente, dado que el objetivo de fondo es revelar patrones presentes en los datos, el conjunto generado debe ser lo suficientemente grande para contener los patrones, y acotado para poder ser minado en un tiempo considerable. Las fuentes más comunes de datos, son las bases de datos transaccionales *data marts* y *data warehouses*.

2. **Pre-procesamiento de datos:** Para garantizar utilidad de los datos, es necesario que los datos sean de la mejor calidad posible. En esta etapa el objetivo primordial es trasformar los datos en bruto, de manera de facilitar la extracción de información contenida en estos. Las principales actividades de esta etapa son: Limpieza de datos (en donde se eliminan valores pérdidos e inconsistencias), transformación (proceso donde se adecúan los datos para el proceso de aprendizaje, en aras de mejorar la capacidad predictiva de este último) y reducción (en donde se eliminan las observaciones o atributos que no sean relevantes para el problema en cuestión).

3. **Minado de datos:** En esta etapa se realizan pricipalmente tareas asociadas al aprendizaje supervisado (clasificación y regresión), como también aprendizaje no supervisado (dependiendo de la naturaleza del problema en cuestión) y aprendizaje por reglas de asociación (por ejemplo búsqueda de relaciones entre variables, donde un caso habitual es *Basket Market Analysis*), ver Fayyad et al. (1996).

4. **Evaluación de los resultados:** Una vez conocidos los resultados del minado de datos, se hace necesario evaluar el desempeño de los modelos predictivos considerados. Para este efecto se disponen de diversas estrategias de validación, según como se particione el conjunto de datos disponibles. La forma más básica de realizar esto, es la validación

simple, que cosiste en utilizar un subconjunto de las muestras para construir el modelo de clasificación, y otro diferente para estimar el error realizado, para con esto eliminar el efecto de *overfitting* (sobreajuste). Dentro de la variedad de formas de realizar la partición, una de las más usadas en la literatura es el proceso de *holdout* (Yang and Liu (2002)), que consiste en entrenar aproximadamente con dos tercios de las observaciones, y validar con el tercio restante. El principal inconveniente de esta técnica es que se existe una potencial pérdida de información útil al utilizar una fracción de los datos para entrenar y no la base completa. Este efecto se torna más relevante cuando la cantidad de observaciones es muy reducida, ya sea por que el porcentaje para construir el modelo es muy pequeño, o bien por que se dispone de pocos datos en primera instancia.

Adicionalmente a la técnica de *Holdout* recién descrita, existe otra llamada *Cross-Validation* (Validación Cruzada), que se genera para evitar la pérdida de información. La forma de realizar esta técnica, es particionar el conjunto de datos en k partes mutuamente excluyentes, conteniendo una cantidad similar de datos cada una, siendo denominada en muchos casos como validación cruzada en k partes (*k-fold crossvalidation* Yang and Liu (2002)). En cada evaluación, se deja uno de los subconjuntos para el testeo, y se entrena con los $k-1$ restantes. De esta manera los indicadores de desempeño considerados pasan a ser la media del desempeño de los k conjuntos de testeo. El principal beneficio de esta técnica, es que todos los casos son utilizados tanto en el proceso de aprendizaje como en el testeo, lo que lleva a obtener un estimador con sesgo muy pequeño. Un caso particular de esta técnica, es la validación cruzada dejando una instancia fuera (*Leave One Out (LOO)*, en cuyo caso k lógicamente corresponde al tamaño completo de la base. A continuación se presenta un esquema gráfico del método de *Cross-Validation*:

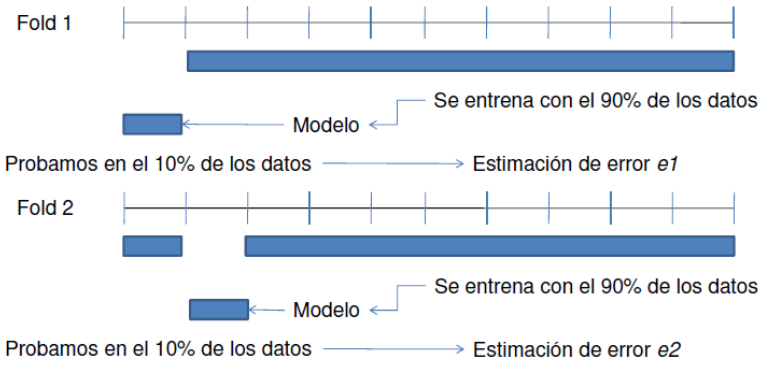

Figura 2.2: Esquema de Cross-Validation

A continuación, revisaremos una forma habitual de minado de datos, conocidad como *Machine Learning*, que será el sustento del desarrollo de este trabajo de Tesis.

2.3. Machine Learning

2.3.1. Introducción al Machine Learning

Se conoce como Machine Learning o aprendizaje automático al proceso en el cual se utiliza un sistema que puede aprender de la información (Mitchell (1997)) y de esta manera poder predecir que sucederá en el futuro. Para poder implementar aprendizaje automático debe existir un patrón subyacente en los datos, ya que es necesario que la pertenencia de los datos a una clase u otra esté dada por un patrón que explique el por qué un objeto pertenece o no a una clase particular. Esto es, que el hecho que un objeto pertenezca a una clase está relacionado con el hecho que otro objeto distinto pertenezca a la misma. Además se debe contar con información, ya que el aprendizaje automático aprende de los datos del pasado.

Algunos ejemplos clásicos de *Machine Learning*, son:

- **Reconocimiento de caracteres**: Categorizar imágenes de caracteres escritos a mano e identificarlas con las letras que representan
- **Detección de rostros**: Encontrar caras en imágenes, o bien indicar si existe alguna cara en estas
- **Filtros Anti-spam**: Identificar los potenciales correos como SPAM y seprarlos de correos habituales
- **Reconocimiento de temas**: Categorizar noticias, artículos, e incluso libros de acuerdo a su contenido, y determinar si son sobre deportes, política, entretenimiento, etc.
- **Diagnóstico médico**: Dados los atributos y los resultados de los exámenes de un paciente médico, establecer si padece o no alguna condición clínica
- **Detección de fraudes**: Identificar las transacciones (por ejemplo de tarjetas de crédito) que podrían ser eventualmente fraudulentas
- **Predicción de clima**: Predecir condiciones climáticas para los días futuros, si lloverá o no el día siguiente, o bien con qué probabilidad lloverá.
- **Segmentación de clientes**: Determiniar que conjunto de clientes será más suscep-

tible a aceptar una campaña o promoción, o bien predecir los potenciales candidatos a fuga de la compañía.

En el mundo del *Machine Learning* se distinguen dos ramas: el **aprendizaje supervisado**, que construye una función a partir de los datos de entrenamiento, que consisten en pares de objetos con los datos de entrada y sus etiquetas correspondientes. El resultado de la evaluación de la función en una observación puede ser tanto un valor numérico (como en los problemas de regresión) o una etiqueta de clase (como en los de clasificación). El objetivo principal del aprendizaje supervisado es crear una función capaz de predecir el valor correspondiente a cualquier objeto de entrada después de haber visto una serie de ejemplos Mitchell (1997). Por otro lado, el **aprendizaje no supervisado**, difiere del aprendizaje supervisado puesto que no hay un conocimiento *a priori*, y su objetivo es describir ciertas características del conjunto de datos de entrada, y entender como éstos se encuentran organizados. En este trabajo el aprendizaje será entendido como supervisado, y nos orientaremos a problemas de clasificación, en particular fuga de clientes en telecomunicaciones.

Los componentes del aprendizaje (en el contexto de fuga de clientes, visto como un problema de Clasificación Binaria) son:

a) Datos de entrada (x): Corresponde a un vector que representa los atributos o características que posee un cliente en particular.

b) Etiquetas (y): Variable binaria (1 ó -1) que representa si un cliente pertenece a la clase 1 o si pertenece a la clase -1.

c) Función Objetivo Ideal ($f : X \to Y$): Corresponde a la fórmula ideal, la cual aplicada a cada cliente predice de manera perfecta la clase a la que pertenece. No se tiene conocimiento de esta función, ya que no se puede determinar con seguridad qué cliente pertenecerá a qué clase.

d) Información Histórica $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$: Representa la información histórica que se ha podido almacenar en una cierta empresa en el tiempo, conteniendo las características de cada cliente que ha tenido y el comportamiento (la etiqueta) de dichos clientes.

e) Hipótesis ($g : X \to Y$): La hipótesis es una función que aproxima a la función ideal de la cual no se tiene conocimiento e intenta explicar cómo se comportarán los clientes en el futuro basado en el comportamiento de los clientes del pasado.

   Para poder crear la función g, los métodos de aprendizaje automático utilizan los datos históricos (tanto los atributos de los clientes como sus clases) y a través de un algoritmo

de aprendizaje va internalizando cómo se comportan los datos según sus atributos.

El algoritmo de aprendizaje basal utilizado en este trabajo de investigación corresponde a *Support Vector Machines*, con diferentes variantes tanto en la definición de hiperparámetros, en la función de Kernel utilizada (ver Sección 2.3.2.4), y la metodología usada para realizar selección de atributos (ver Sección 2.6)

2.3.2. Support Vector Machines (SVM)

Support Vector Machines (formulado explícitamente en Vapnik (1998)) es un modelo de clasificación basado en minimizar el error cuadrático de la clasificación, a la vez que se construye un hiperplano que separa los datos de la forma mas precisa posible. Es considerado uno de los modelos más precisos y robustos posibles dentro de los algoritmos de clasificación binaria (Wu et al. (2008)), siendo muy utilizado y considerado dentro de los más versátiles y efectivos a la hora de clasificar, principalmente por que considera la minimización del error estructural al clasificar nuevos objetos, que está directamente relacionado con uno de los principales objetivos de un clasificador, tener la posibilidad de enfrentar tanto problemas lineamente separables, como aquellos que no lo son, y tener la habilidad de generalizar de forma correcta como vimos en el apartado anterior.

A modo de ejemplo, se presenta el siguiente ejemplo de clasificación binaria, que es lineamente separable:

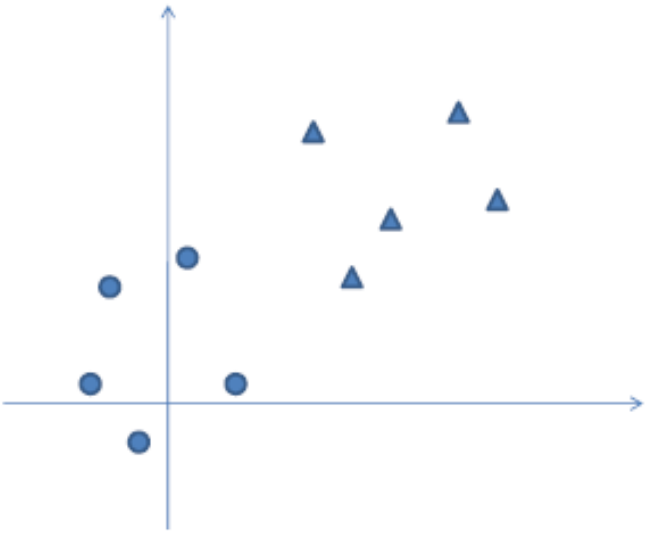

Figura 2.3: Problema de Clasificación binaria

Como se puede notar en la figura 2.3, se pueden ver a simple vista dos clases linealmente separables, por un lado los círculos representan una clase y los triángulos representan la otra. En el presente problema, existen infinitas soluciones (hiperplanos) que separan ambas clases, y a modo de ejemplo en la figura 2.4 se pueden ver gráficamente dos de ellas:

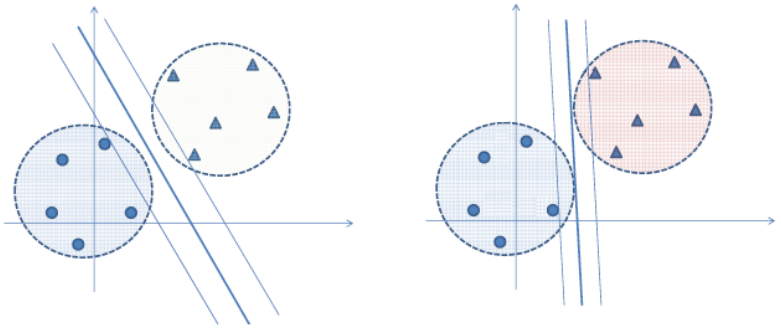

Figura 2.4: Problema de Clasificación binaria, ejemplos de soluciones

Support Vector Machines **(SVM)** no sólo es capaz de separar estas dos clases, sino que además minimiza el posible error en la clasificación de un nuevo objeto. Dicho esto el objetivo de éste modelo consiste en encontrar un hiperplano de separación optimal dividiendo el espacio en dos regiones conteniendo idealmente en cada una de ellas a una clase, y para este efecto minimiza el riesgo estructural *maximizando* la distancia que separa el hiperplano de los objetos más cercanos a él.

En la mayoría de los casos de la vida real (si no en todos), **no** es posible realizar una separación perfecta de las clases de manera lineal, por lo que este método construye un hiperplano caracterizado por una función objetivo cuadrática y un conjunto de restricciones lineales, que en cierta medida permiten holguras y a su vez restringen los errores.

En las siguientes secciones se explican los fundamentos principales de esta metodología, comenzando por su caso más simple.

2.3.2.1. SVM lineal, caso separable

En general, el modelo se construye sobre un subconjunto de los datos, a saber *conjunto de entrenamiento*, y en este caso particular, se asume que los datos son linealmente separables. Esto es, existe un hiperplano en $\mathbb{R}^M$ donde M es la cantidad de atributos, que separa completamente una clase de la otra (ver figura 2.4). Matemáticamente hablando, existe un $(\vec{w}, b) \in \mathbb{R}^{M+1}$ tal que si $\vec{w} \cdot x_i + b \geq 0$, el objeto (en este caso el *cliente*), pertenece a la clase positiva (posible fuga), o bien si $\vec{w} \cdot x_i + b < 0$ el cliente pertenece a la clase negativa. Luego para poder clasificar de manera correcta a la base completa (al menos al conjunto

de entrenamiento) se deben incluir todas estas restricciones al problema de minimzación. Además, existe un conjunto de clientes "especial", y son aquellos que cumplan $\mid \vec{w} \cdot x_i + b \mid = 1$, que se encuentran en los *hiperplanos canónicos*, y se definen de la siguiente manera:

$$\vec{w} \cdot x_i + b = 1 \tag{2.1}$$

$$\vec{w} \cdot x_i + b = -1 \tag{2.2}$$

Los hiperplanos canónicos son paralelos al hiperplano optimal dado por el par $(\vec{w}, b)$ y su distancia a ellos viee dada por $\frac{1}{\|\mathbf{w}\|}$ (ver figura 2.5).

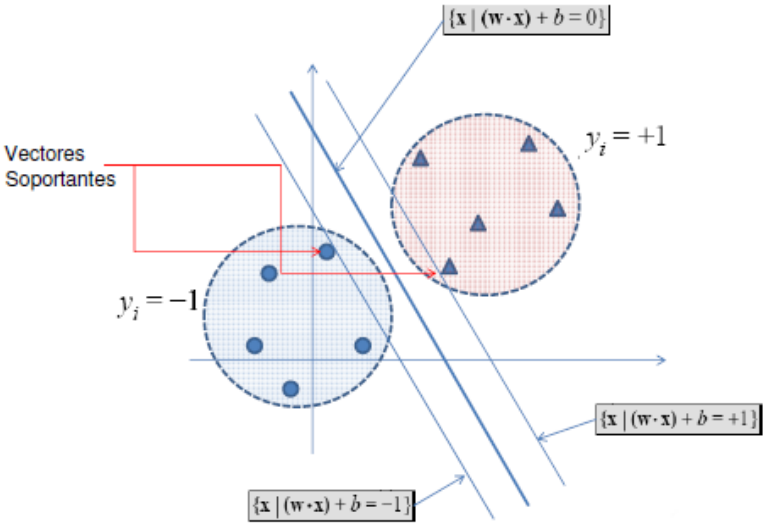

Figura 2.5: SVM Lineal, caso separable

Los objetos que pertenecen a cada una de las clases y cumplen las igualdades de los hiperplanos canónicos son llamados *Support Vectors (Vectores de Soporte)*, y dado que el objetivo es maximizar la distancia entre el hiperplano optimal y los hiperplanos canónicos, se minimiza la norma euclidiana de $\vec{w}$ dando origen al siguiente problema de minimización como formulación primal:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.a.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq 1, \qquad i = 1, \ldots, N. \end{aligned} \tag{2.3}$$

Formulación de la cual es posible construir el problema dual, mediante la técnica de los *Multiplicadores de Lagrange*, donde el Lagrangeano toma la siguiente forma:

$$\mathcal{L}(w, b, a) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \cdot [y_i(\mathbf{w}^\top \cdot \mathbf{x}_i + b) - 1] \tag{2.4}$$

De donde se desrprende, usando las condiciones de *Karush-Kuhn-Tucker*, para $\vec{w}$ y para b:

$$\frac{\partial \mathcal{L}(w, b, a)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{\mathbf{i=1}}^{\mathbf{N}} \alpha_\mathbf{i} \cdot \mathbf{y_i} \cdot \mathbf{x_i} = \mathbf{0} \tag{2.5}$$

$$\frac{\partial \mathcal{L}(w, b, a)}{\partial b} = \sum_{i=1}^{N} \alpha_i \cdot y_i = 0 \tag{2.6}$$

De las cuales se obtiene la formulación Dual:

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{s=1}^{N} \alpha_i \cdot \alpha_s \cdot y_i \cdot y_s \cdot \mathbf{x}_i \cdot \mathbf{x}_s \\ &\text{s.a.} \quad \sum_{i=1}^{N} \alpha_i \cdot y_i = 0, \qquad i = 1, \ldots, N. \\ &\qquad\qquad \alpha_i \geq 0, \qquad\qquad i = 1, \ldots, N. \end{aligned} \tag{2.7}$$

Una vez resuelto este problema de maximización, de esta descripción se deduce que los vectores con multiplicadores asociados $\alpha_i > 0$, son denominados *Support Vectors*, puesto que son los únicos que participan en la construcción del hiperplano explícitamente, es decir, se puede construir el mismo hiperplano considerando sólo estas observaciones.

Además con esto, para predecir la etiqueta de un nuevo (y desconocido) objeto $\vec{z}$, basta con evaluarlo en la función de clasificación (con los parámetros obtenidos de resolver la formulación dual) y determinar en qué lado de la frontera de clasificación pertenece simplemente tomando el signo de la evaluación.

2.3.2.2. SVM lineal, caso no separable

Como se menciona con anterioridad, la mayoría de los problemas que se presentan en la vida real, dada su naturaleza, no son separables (al menos en la representación de datos que se puede obtener) por lo que no es posible construir una función de clasificación lineal como la descrita capaz de separar los datos inequívocamente. Es por esto que para estos casos SVM no llegará a una solución factible dado que no existirá ningún hiperplano que cumpla con *todas* las restricciones descritas anteriormente.

Dicho esto, se le permite al método poder equivocarse en *algunos* elementos, introduciendo un costo por objeto mal clasificado ξ_i con $i = 1, \ldots, N$. Estas variables de holgura tienen por objetivo relajar las restricciones, permitiendo que ocurran errores, pero penalizándolos en la función objetivo. Con esto en mente, la nueva formulación toma la siguiente forma:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.a.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, N, \\ & \xi_i \geq 0, \qquad i = 1, \ldots, N. \end{aligned} \tag{2.8}$$

Siguiendo la línea argumental de la sección anterior, la forma Dual se expresa como sigue:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{s=1}^{N}\alpha_i \cdot \alpha_s \cdot y_i \cdot y_s \cdot \mathbf{x}_i \cdot \mathbf{x}_s \\ \text{s.a.} \quad & \sum_{i=1}^{N}\alpha_i \cdot y_i = 0, \qquad i = 1, \ldots, N. \\ & C \geq \alpha_i \geq 0, \qquad i = 1, \ldots, N. \end{aligned} \tag{2.9}$$

Cuando las soluciones brindadas en los formatos SVM anteriormente mencionados, no obtienen resultados satisfactorios, se puede intentar *mapear* los datos en espacios de mayor dimensión, mediante funciones no necesariamente lineales, y separarlos en el nuevo espacio de mayor complejidad. A continuación presentaremos los detalles de la formulación no lineal de SVM.

2.3.2.3. SVM no lineal

Supongamos el siguiente caso expuesto en la figura 2.6, el ideal es tomar los datos en el plano y enviarlos al otro espacio para entonces poder separarlos de manera lineal:

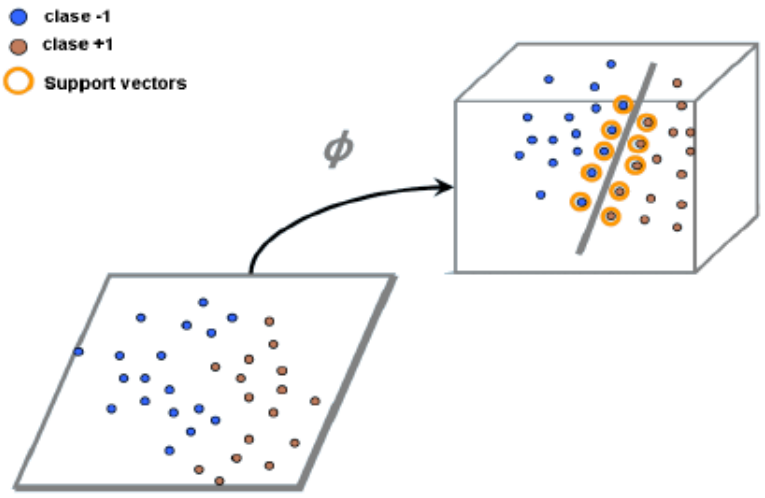

Figura 2.6: SVM no Lineal, representación gráfica

De la figura anterior se puede apreciar cómo una función ϕ proyecta los datos (no separables en $\mathbb{R}^2$) a un espacio $\mathbb{R}^3$, donde los datos pueden ser separados de manera lineal, encontrando un hiperplano en $\mathbb{R}^3$ que puede ser representado luego en $\mathbb{R}^2$ como se puede ver a continuación:

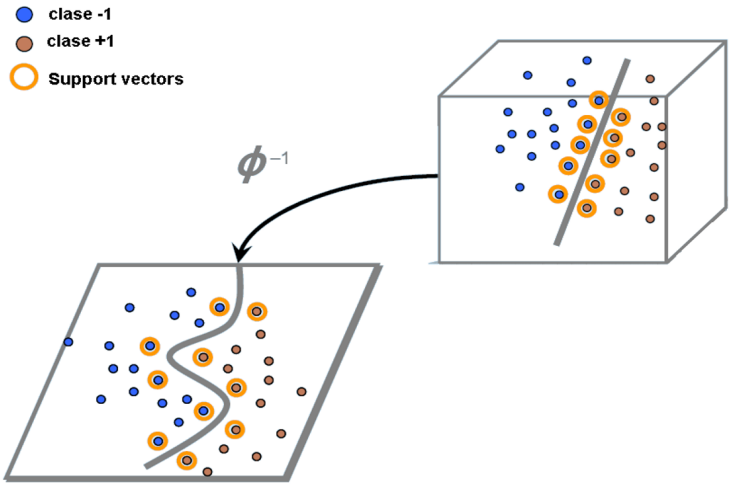

Figura 2.7: SVM no Lineal, superficie en el dominio original

La formulación previa vista en la sección anterior (ver ecuación 2.8), puede ser extendida

a clasificadores no necesariamente lineales, usando el *kernel trick*: Los datos de entrenamiento son transformados en un espacio de mayor dimensionalidad $\mathscr{H}$, a través de una función $\phi : \mathbf{x} \to \phi(\mathbf{x}) \in \mathscr{H}$ (Schölkopf and Smola (2002)). Una función de Kernel $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \cdot \phi(\mathbf{y})$ define un producto interno en el espacio de $\mathscr{H}$ (*Teorema de Mercer*), lo que nos lleva a la siguiente formulación:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,s=1}^{N} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s) \\ \text{s.a.} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \\ & 0 \leq \alpha_i \leq C, \qquad i = 1, \ldots, N. \end{aligned} \tag{2.10}$$

De donde, resolviendo este último problema, es posible testear la pertenencia de un nuevo objeto a las clases en cuestión, transformándolo hacia $\mathscr{H}$ y usando los mismos criterios de pertenencia que en las seccione anteriores.

2.3.2.4. Funciones de Kernel

Asumiendo que los datos de entrada siempre pertenecen a $\mathbb{R}^M$, los Kernels más utilizados en la literatura son, :

a) Kernel Polinomial Homogéneo de grado d:

$$K(\mathbf{x}_i, \mathbf{x}_s) = \langle \mathbf{x}_i, \mathbf{x}_s \rangle^d \tag{2.11}$$

b) Kernel Gaussiano de Base Radial, sugerido por Boser et al. (1992), con $\sigma > 0$ controlando el ancho de kernel:

$$K(\mathbf{x}_i, \mathbf{x}_s) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_s||^2}{2\sigma^2} \right) \tag{2.12}$$

c) Kernel Sigmoidal, con $\kappa > 0$ y $\nu > 0$. No es definido positivo para algunos valores de κ, ν sin embargo se ha utilizado en la práctica a pesar de esto:

$$K(\mathbf{x}_i, \mathbf{x}_s) = tanh(\kappa \langle \mathbf{x}_i, \mathbf{x}_s \rangle + \nu) \tag{2.13}$$

d) Kernel Polinomial No Homogéneo de grado d, con $c \geq 0$:

$$K(\mathbf{x}_i, \mathbf{x}_s) = (\langle \mathbf{x}_i, \mathbf{x}_s \rangle + c)^d \tag{2.14}$$

e) Kernel Racional Cuadrático, con $c > 0$:

$$K(\mathbf{x}_i, \mathbf{x}_s) = 1 - \frac{||\mathbf{x}_i - \mathbf{x}_s||^2}{||\mathbf{x}_i - \mathbf{x}_s||^2 + c} \tag{2.15}$$

Notar además que todos los Kernels mencionados con anterioridad son unitariamente invariantes, lo que es interesante, es decir: $(K(\mathbf{x}_i, \mathbf{x}_s) = K(\mathbf{U}\mathbf{x}_i, \mathbf{U}\mathbf{x}_s) \quad si \quad \mathbf{U^T} = \mathbf{U^{-1}})$. Los kernel de base radial (RBF), pueden ser escritos de la forma $K(\mathbf{x}_i, \mathbf{x}_s) = f(d(\mathbf{x}_i, \mathbf{x}_s)$, con $\mathbf{d}$ es una métrica en $\mathbb{R}^M$ y $\mathbf{f}$ una función en $\mathbb{R}^0_+$. Adicionalmete, los kernels RBF son invariantes a la traslación, esto es $K(\mathbf{x}_i, \mathbf{x}_s) = K(\mathbf{x}_i - \mathbf{x}_0, \mathbf{x}_s - \mathbf{x}_0)$ para todo $\mathbf{x}_0 \in \mathbb{R}^M$.

2.4. Medidas de desempeño

2.4.1. Motivación

Tratar de predecir qué clientes van a dejar una compañia, fugarse o simplemente *churn* (en este trabajo se hace referencia indistintamente a fuga o *churn*) como se conoce en la literatura, es una de las tareas mas importantes de las empresas de servicio, principalmente en banca y telecomunicaciones. La importancia de la predicción de fuga de cliente es potenciada con la creciente cantidad de clientes dispuestos a cambiar sus proveedores, y por la fuerte competencia por captar a clientes nuevos. Es por esto que nace la necesidad urgente de crear y desarrollar modelos precisos capaces de identificar clientes actuales que de alguna manera tengan tendencia a dejar la compañia en un período dado de tiempo. *Churn* puede a su vez ser observado de dos maneras diferentes, **voluntario**, en donde el cliente decide terminar el contrato, o bien **involuntario**, donde la compañia en cuestión decide terminar el contrato con el cliente (Blattberg et al. (2008)). En este trabajo estudiaremos el *churn* como un fenómeno voluntario.

El objetivo más importante cuando se desarrollan modelos de predicción de fuga, es establecer estrategias orientadas a la retención del cliente. Si la compañia es capaz de identificar los posibles *churners*, el siguiente paso es desarrollar campañas comerciales y estrategias de retención enfocadas en este grupo en particular, potenciando de esta manera la lealtad del cliente y obteniendo mayores beneficios, como por ejemplo:

- Un mejor estándar de servicio y estrategias comerciales enfocadas en mejorar la lealtad del cliente

- Un incremento en la cantidad de clientes fidelizados. Los clientes fidelizados, generan 1.7 más ingreso que los otros clientes (Fleming and Asplund (2007))

- Un impacto directo en la rentabilidad: un 5 % de incremento en la tasa de retención al cliente, puede llevar a un 18 % de reducción en costos operacionales (Fleming and Asplund (2007)).

- Una disminución del gasto innecesario, enfocando los recursos en candidatos reales a fuga en vez de a la base de clientes completa, reduciendo costos operacionales y de marketing (Van den Poel and Larivière (2004)).

Atendiendo a estos hechos, la tasa de fuga es puesta explícitamente en la formula para el *Customer Life Value* (en adelante **CLV**), que toma la siguiente forma considerando periódos anuales (ver Blattberg et al. (2008)):

$$CLV = \sum_{t=1}^{\infty} \frac{m(1-c)^{t-1}}{(1+r)^{t-1}} = m\frac{(1+r)}{(r+c)} \tag{2.16}$$

donde c es la tasa anual de fuga y m es el retorno esperado medio por cliente. El Parámetro r es la tasa de descuento anual. Existen dos maneras clasicas de determinar este último valor. La primera es obtener directamente el *Weighted Average Cost of Capital (WACC)* de la compañia. La segunda, es usar la tasa de descuento del sector industrial en particular. Dada la estructura de esta fórmula, el entendimiento del CLV como en valor presente neto del del beneficio por cliente, luego un descenso en la tasa de fuga de clientes impacta de manera directa en las ganancias de la compañia.

El fenómeno de la fuga de clientes puede ser modelado con técnicas dependientes del tiempo (Blattberg et al. (2008)), o bien como predicciones sobre simplemente el siguiente período futuro. En el primer caso este tipo de modelos no asume que la fuga ocurrirá explícitamente en un período de tiempo, proponiendo proabilidades de fuga hasta un número fijo de períodos desde el origen de los datos, incluso variando con el tiempo (ver Blattberg et al. (2008)). De la otra manera, encontramos formas de enfocarnos a predecir si un cliente decide o no fugarse en el período siguiente, donde los enfoques más tradicionales son regresión logística (ver Burez and Van den Poel (2009); Lemmens and Croux (2006); Neslin et al. (2006)), modelos estadísticos no paramétricos como *K-nearest neighbor (KNN)* (Datta et al. (2000)), árboles de decisión (Wei and Chiu (2002)), redes neuronales (Hung et al. (2006)), y obviamente técnicas de *Machine Learning*, como se puede apreciar en Verbeke et al. (2012). Una revisión mas completa sobre la modelación

de fuga de clientes, puede ser encontrado en Verbeke et al. (2011b). En este trabajo de Tesis, usaremos clasificadores basado en SVM, prediciendo la fuga de los clientes en el siguiente período de tiempo.

Incluso cuando en muchos sectores de la idustria de servicios se encuentran tasas de fuga anuales en torno al 20 % y 50 % (Neslin et al. (2006)), cuando se usan modelos de predicción de fuga mensual, las tasa de fuga se mantienen usualmente bajo el 5 % (ver Verbeke et al. (2012)), lo que nos lleva naturalmente al problema de desbalance de clases.

Para poder evaluar el desempeño de un clasificador, es necesario usar alguna medida de rendimiento que permita conocer el grado de asertividad que tiene dicho clasificador. Adicionalmente, esto permite comparar un clasificador con otro, en el sentido de su certeza al momento de predecir nuevas observaciones. En la literatura se proponen muchas métricas de rendimiento para evaluar el desempeño de un clasificador. Una revisión mas exhaustiva en el tema puede ser encontrada en Chawla (2010).

Dado un objeto $\mathbf{x}$ (observación, evento, cliente,etc.) un clasificador $\mathcal{C}$ producirá una puntuación s, donde por convención una puntuación más alta implica que tiene mayor tendencia a ser etiquetado con (+1), es decir, *churn*. Se fija un valor de **umbral** t para proveer una clasificación de la base completa basada en sus puntajes. De esta manera todas las instancias que tengan una puntuación s menor que t son clasificados como *no-churners* (-1), en tanto que clientes con s mayor que t son considerados como *churners* (+1).

Siendo coherentes con la notación presentada en Verbeke et al. (2011a), explicitamos las siguientes definiciones:

- **Probabilidades a Priori**: π_{-1} y π_1 son las probabilidades a priori de que una observación posea la etiqueta -1 o 1, respectivamente. Notemos que $\pi_{-1} + \pi_1 = 1$, es decir, son las únicas posibilidades para una observación.

- **Distribuciones de Probabilidades**: Dado una puntuación s, la función de densidad de probabilidad para los *non-churners* y los *churners* son respectivamente $f_{-1}(s)$ y $f_1(s)$, en donde las funciones de densidad acumlativa son denotadas por $F_{-1}(s)$ y $F_1(s)$.

- **Terminos de Costo-Beneficio**: Definimos b_{-1} (b_1) como el beneficio obtenido de clasificar correctamente a un *non-churner (churner)*, y c_{-1} (c_1) al costo de clasificar incorrectamente un *non-churner (churner)*. Así mismo definimos $\theta = (b_1 + c_1)/(b_{-1} + c_{-1})$ como el **Ratio Costo Beneficio** para simplificar notación. Tanto el beneficio esperado de realizar la campaña, cómo el umbral óptimo dependeran de este ratio de costos y beneficios, por lo que es importante definirlo en si mismo.

Con la ayuda de la notación recién presentada, es posible construir la siguiente matriz, conocida como la **Matriz de Confusión**:

		Clasificado cómo	
		Clase -1	Clase 1
Pertenece a	Clase -1	**True Negative (TN)** $[c(-1\|-1) = b_{-1}]$	**False Positive (FP)** $[c(1\|-1) = c_{-1}]$
	Class 1	**False Negative (FN)** $[c(-1\|1) = c_1]$	**True Positive (TP)** $[c(1\|1) = b_1]$

Cuadro 2.1: Matriz de confusión, que indica los posibles resultados de la clasificación de una observación, y los beneficios o costos que esto genera

Esta matriz se encuentra construida en base a las predicciones sobre un conjunto de entrenamiento, es por esto que se cada observación posee dos etiquetas, la predicción del clasificador y la clase donde realmente pertenece. De esta manera es posible asignar cada observación de manera única en una de las cuatro casillas descritas en la tabla 2.1. El detalle de cada casilla es el siguiente:

- **TN (True Negative)**: Observaciones negativas etiquetadas de manera correcta por el método.
- **FP (False Positive)**: Observaciones negativas erróneamente clasificadas como clase positiva.
- **FN (False Negative)**: Observaciones positivas clasificadas erróneamente como clase negativa.
- **TP (True Positive)**: Observaciones positivas etiquetadas correctamente como positivas.

2.4.2. Indicadores de rendimiento habituales

En esta sección, y usando los conceptos definidos en la tabla 2.1, definimos los indicadores más usados en la literatura.

a) **Accuracy**: El *Accuracy*, es definido como el porcentaje de aciertos sobre la totalidad de la base, explicitando e identificando términos definidos en la matriz de confusión

señalada en la tabla 2.1, podemos usar la siguiente expresión:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{2.17}$$

Esta métrica es una de las más utilizadas, pero presenta el grave problema de no diferenciar los tipos de errores que pueden cometerse. Puesto que un clasificador orientado a maximixar este indicador, cuando la base tiene alto desbalance, tendría tendencia a clasificar todas las observaciones como clase mayoritaria.

b) **Balanced Accuracy**: A diferencia del *Accuracy*, este indicador pondera la tasa de acierto de cada una de las clases, esto es, la cantidad de aciertos sobre la totalidad de los elementos de esa clase, siendo más explícitos, podemos definirlo de la siguiente manera:

$$Balanced\ Accuracy = \alpha_1 \cdot \frac{(TP)}{(TP + FN)} + \alpha_2 \cdot \frac{(TN)}{(TN + FP)} \tag{2.18}$$

Donde α_i con $i = 1, 2$ son los ponderadores asociados a cada clase, cumpliendo lógicamente $\alpha_1 + \alpha_2 = 1$.

c) **Recall**: Corresponde a la proporción de observaciones pertenecientes a la clase positiva que son correctamente predichos como positivos:

Este indicador consiste en la proporción de observaciones de la clase positiva correctamente predichos como positivos:

$$Recall = \frac{(TP)}{(TP + FN)} \tag{2.19}$$

d) **Precison**:Este indicador denota la proporción de las observaciones etiquetadas como positivas que corresponden a la clase positiva (Powers, 2011):

$$Precision = \frac{(TP)}{(TP + FP)} \tag{2.20}$$

Precision suele calcularse en conjunto con *Recall*, puesto que ambas miden el desempeño de los aciertos predictivos de la clase positiva, pero desde distinta perspectiva (al

considerar distintos denominadores).

e) **F-Measure**: este indicador, es una combinación de los dos últimos indicadores descritos (*Precision* y *Recall*), tomando la media ármonica entre ellos:

$$F - Measure = 2 \cdot \frac{(Precision * Recall)}{(Precision + Recall)} \tag{2.21}$$

O bien:

$$F - Measure = \frac{TP}{(TP + \frac{FN+FP}{2})} \tag{2.22}$$

Esta medida es un poco más robusta que cada una de las anteriores por si sola, puesto que considera explícitamente los dos tipos de errores en su definición.

Adicionalmente a las medidas recién mencionadas, una medida usada frecuentemente en la literatura para evaluar el desempeño es el AUC, que corresponde al área bajo la curva de ROC. La curva de ROC (*Receiver Operating Characteristic* es una representación del desempeño del clasificador en la medida que se cambia el valor umbral t, que marca el límite para discernir si una observación pertenece a la clase positiva o negativa dado el score obtenido por el clasificador para esa observación. Dicho de otro modo, la curva de ROC corresponde a un gráfico que muestra la **Sensibilidad** vs $1 -$ **Especificidad**, es decir $F_{-1}(t)$ como función de $F_1(t)$:

$$\begin{aligned}
&\text{Sensibilidad} = F_{-1}(t), \\
&\text{Especificidad} = 1 - F_1(t), \\
&\text{AUC} = \int F_{-1}(s) f_1(s) \mathrm{d}s.
\end{aligned}$$

El área bajo la curva de ROC es usada a menudo para evaluar el desempeño de los clasificadores. En términos simples, el AUC de un método de clasificación es la probabilidad de que una observación positiva elegida de forma aleatoria sea puntuada más alta que una clasificación negativa escogida al azar (Fawcett (2006)). Lo que implica que a mayor AUC, mejor capacidad predictiva.

2.4.3. Indicadores de rendimiento asociados a Utilidades

2.4.3.1. Conceptos preeliminares

Al momento de montar una campaña de retención de fuga de clientes, una fracción η de los clientes mas propensos a fuga es contactada, con un costo de f por persona, para ofrecerles un incentivo con un costo monetario d para la empresa. Entre este conjunto de clientes, lógicamente habrán clientes que eventualmente se fugarían como clientes que no tenían intenciones de hacerlo. Dado esto, se asume que a priori los clientes del último grupo aceptan el incentivo, y no se fugan de la compañía, puesto que no tenían la intención de hacerlo en primera instancia (ver Verbraken et al. (2012)). Para el primer grupo, se estima que una fracción γ acepta la oferta, lo que resulta en una ganancia directa en CLV (Ecuación 2.16), y dado esto una fracción $1-\gamma$ efectivamente deja la compañía a pesar del incentivo. De la otra fracción no contactada de clientes, todos los clientes que se fugarían, se asume que lo harán, y los que no se quedarán en la empresa. Este proceso se puede ver gráficamente , en el siguiente diagrama:

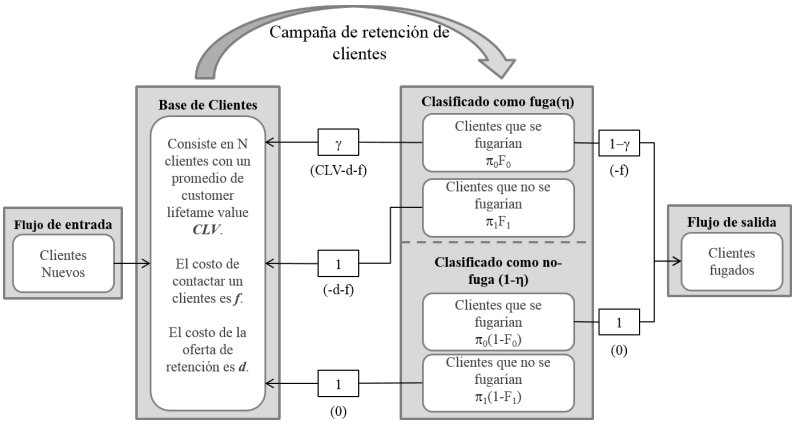

Figura 2.8: Esquema de evaluación de desempeño de una campaña de retención de clientes. (Fuente: Verbraken et al. (2012))

Y este proceso recién descrito en la figura 2.8, se sintetiza en la siguiente ecuación para el *Profit* (Neslin et al. (2006)):

$$\text{Profit } = N\eta\left[(\gamma CLV + d(1-\gamma))\,\pi_{-1}\lambda - d - f\right] - A, \tag{2.23}$$

donde η es la fracción de clientes contactada, CLV es el *Customer Lifetime Value* (ver ecuación 2.16), d es el costo del incentivo, f es el costo de contactar al cliente, y A son los costos administrativos fijos para la campaña. El coeficiente de *lift* λ es la fracción de clientes que se fugarían en la fracción η de clientes dividido por la tasa base de fuga para

todos los clientes (a saber, π_{-1} Verbeke et al. (2011a)). Finalmente, γ es interpretado como la probabilidad de que un cliente que tenía pensado fugarse acepte el incentivo y no se vaya de la empresa. Para todo efecto, CLV, A, f y d son positivos, y $CLV > d$ para que esto sea coherente en primera instancia. Es importante notar que claramente η depende de la selección del valor umbral t, lo que permite a la empresa tener el control sobre que fracción de clientes contactar para ofrecer el incentivo de retención. Haciendo un pequeño ajuste a la matriz de confusión vista el la Tabla 2.1, se obtiene lo siguiente:

		Clasificado cómo	
		Clase -1	Clase 1
Pertenece a	Clase -1	$\pi_{-1}F_{-1}(t)\,N$ $[c(-1\|-1)=b_{-1}]$	$\pi_{-1}(1-F_{-1}(t))\,N$ $[c(1\|-1)=c_{-1}]$
	Clase 1	$\pi_1 F_1(t)\,N$ $[c(-1\|1)=c_1]$	$\pi_1(1-F_1(t))\,N$ $[c(1\|1)=b_1]$

Cuadro 2.2: Matriz de confusión, en este caso se detalla la proporción de la base total que corresponde en cada casilla, y su costo o beneficio correspondiente

Considerando los datos de esta matriz, que entrega las probabilidades requeridas y si además nos remitimos a estudiar el *Profit* promedio en vez de el *Profit* total, se puede descartar A puesto que es un costo fijo que no depende del clasificador. Dicho esto, se define el **Profit medio generado por un clasificador para fuga de clientes** como la siguiente expresión:

$$P_{\mathcal{C}}(t;\gamma,CLV,\delta,\phi) = CLV\,(\gamma(1-\delta)-\phi)\,\pi_{-1}F_{-1}(t) - CLV(\delta+\phi)\cdot\pi_1F_1(t). \quad (2.24)$$

donde $\delta = \frac{d}{CLV}$ and $\phi = fCLV$. Y es posible notar que $b_{-1} = CLV\,(\gamma(1-\delta)-\phi)$, y $c_1 = CLV(\delta+\phi)$.

También es posible deducir las siguientes relaciones, identificando términos y viendo la estructura de la ecuación 2.24:

- $\eta(t) = \pi_{-1}F_{-1}(t) + \pi_1F_1(t)$
- $\lambda(t) = \frac{F_{-1}(t)}{\pi_{-1}F_{-1}(t)+\pi_1F_1(t)}$.

Dicho esto, y teniendo un conjunto de entrenamiento de clientes $\mathcal{T}$ con atributos $\mathcal{F}$, se estudian tres métricas diferentes, descritas a continuación:

2.4.3.2. H- Measure

The H-Measure: En Hand (2009) se propone esta métrica como una alternativa al AUC. La diferencia principal entre este indicador y las MP que serán descritas posteriormente, es que la medida H sólo se enfoca en costos. Dicho esto, el foco de esta medida no es en maximizar beneficios, si no en minimizar costos esperados. De esta manera, definiendo la perdida media de clasificación Q como:

$$Q_{\mathcal{C}}(t; c, b) = b \cdot \left[c\pi_{-1}\left(1 - F_{-1}(t)\right) + (1 - c)\pi_1 F_1(t)\right], \tag{2.25}$$

donde $c = c_{-1}/(c_{-1} + c_1)$ y $b = c_{-1} + c_1$. Para esta medida, el ratio de costos del cual T depende, es $\theta = (1 - c)/c$.

Ahora bien, calcular el valor de la pérdida mínima esperada requiere hacer supuestos sobre la densidad de probabilidad tanto de b como de c. Si se asume independencia de estos valoresm y definiendo $w(b, c)$ como la distribución conjunta de b y c, asi como $u(c)$ y $v(b)$ las densidades de probabilidad marginal de c y b respectivamente, la relación explícita entre ellas es $w(b, c) = u(c)v(b)$, usando esto último, la pérdida mínima esperada L toma la siguiente forma:

$$\mathrm{L} = \mathrm{E}[b] \int_0^1 Q_{\mathcal{C}}(T(c); b, c) \cdot u(c)\mathrm{d}c, \tag{2.26}$$

con $\mathrm{E}[b] = 1$ seleccionando convenientemente la unidad en la que se mide b. Se asume además que c sigue una distribución Beta con parametros α y β, cuya forma funcional es:

$$u_{\alpha,\beta}(x) = \begin{cases} \frac{x^{\alpha-1}\cdot(1-x)^{\beta-1}}{B(1,\alpha,\beta)} & \text{si } x \in [0, 1], \\ 0 & \text{si no,} \end{cases} \tag{2.27}$$

donde α y $\beta \in \mathbb{R}$ y mayores que 1, y además:

$$B(x, \alpha, \beta) = \int_0^x t^{\alpha-1} \cdot (1 - t)^{\beta-1}\mathrm{d}t. \tag{2.28}$$

Una vez definido esto, para llegar a una métrica final (*H-measure*), se normaliza lo obtenido, para obtener una métrica acotada entre cero y uno:

$$\mathrm{H} = 1 - \frac{\int_0^1 Q_{\mathcal{C}}(T(c); b, c) \cdot u(c)\mathrm{d}c}{\pi_0 \int_0^{\pi_1} c \cdot u(c)\mathrm{d}c + \pi_1 \int_{\pi_1}^1 (1 - c) \cdot u(c)\mathrm{d}c}, \tag{2.29}$$

Acá $u(c)$ es una abreviación de notación de $u_{\alpha,\beta}(c)$. El denominador corresponde a la pérdida ocasionada por el peor clasificador, que es un clasificador aleatorio. En este punto es interesante notar además que la integración sobre $c = [0, 1]$ corresponde a una integra-

ción sobre $\theta = [0, +\infty)$, lo que es equivalente a variar la tangente de la curva de ROC yendo desde infinito a cero.

2.4.3.3. Maximum Profit

***Maximum Profit Criterion* para fuga de clientes (MPC)**: Si se asume que todos los parámetros de la ecuación 2.24 son conocidos, para un clasificador $\mathcal{C}$ es posible generar una medida determinista. Tomando el valor máximo para todos los posibles umbrales, tenemos la siguiente medida de rendimiento (ver Verbeke et al. (2012)):

$$\text{MPC} = \max_{\forall t} P_{\mathcal{C}}(t; \gamma, CLV, \delta, \phi). \tag{2.30}$$

De esta manera es posible obtener la fracción de clientes que debe ser contactada, $\bar{\eta}_{\text{mpc}}$, para poder maximizar el beneficio generado por la campaña de retención. y esta dado por:

$$\bar{\eta}_{\text{mpc}} = \pi_{-1} F_{-1}(T) + \pi_1 F_1(T), \tag{2.31}$$

en donde

$$T(\gamma) = \arg\max_{\forall t} P_{\mathcal{C}}(t; \gamma, CLV, \delta, \phi). \tag{2.32}$$

2.4.3.4. Expected Maximum Profit

***Expected Maximum Profit Criterion* para fuga de clientes (EMPC)**: Para este caso particular, se modela γ, que corresponde a la probabilidad de que un cliente que se iba de la empresa acepte el incentivo y se quede, como una variable aleatoria destribuida como una función Beta, lo que conduce a la siguiente expresión::

$$\text{EMPC} = \int_{\gamma} P_{\mathcal{C}}(T(\gamma); \gamma, CLV, \delta, \phi) \cdot h(\gamma) \mathrm{d}\gamma, \tag{2.33}$$

Donde $T(\gamma)$ es el corte optimal según la ecuación (2.32) y $h(\gamma)$ la densidad de probabilidad para γ. Los parámetros α y β, relacionados a la distribución Beta de γ fueron obtenidos de un trabajo previo de fuga de clientes, que puede ser revisado en detalle en Verbraken et al. (2012). De manera Análoga al **MPC**, el porcentaje o fracción de clientes contactados en la campaña de retención sugerida por esta métrica es:

$$\bar{\eta}_{\text{empc}} = \int_{\gamma} \left[\pi_{-1} F_{-1}\left(T(\gamma)\right) + \pi_1 F_1(T\left(\gamma\right))\right] \cdot h(\gamma) \mathrm{d}\gamma, \tag{2.34}$$

Es importante notar la influencia que tiene γ en esta última ecuación, que tiene un rol fundamental en el ratio costo beneficio, y este último valor es importante, puesto que define el umbral óptimo:

$$\theta = \frac{b_1 + c_1}{b_{-1} + c_{-1}} = \frac{\delta + \phi}{\gamma(1-\delta) - \phi} \tag{2.35}$$

2.5. Métodos para tratar el desbalance de clases

Una de las grandes complejidades que enfrentan los algoritmos de clasificación es el desbalance de clases, es decir, una de las clases posee una cantidad muy pequeña de objetos (en este caso, clientes), con respecto de la otra clase, como se puede ver en Barandela et al. (2003).

Dependiendo de la construcción del algoritmo, es posible que este último disminuya su efectividad cuando es puesto a prueba en este tipo de problemas,puesto que tienden a sobreajustarse a la clase mayoritaria, tendiendo un sesgo a clasificar un nuevo objeto en la clase más numerosa (Maloof (2003)), esto es debido a que los algoritmos sacrifican el rendimiento en la predicción de la clase minoritaria buscando obtener mejor acierto global (Ertekin et al. (2007)). Por ejemplo en un problema en el cual la clase mayoritaria representa un 95 % del total de observaciones (y la clase minoritaria el 5 % restante), los métodos de clasificación sólo errarán en un 5 % global al clasificar todos los datos como clase mayoritaria. Esto se da principalmente si la métrica que se usa para evaluar el desempeño del algoritmo de clasificación depende fuertemente del acierto global y no de métricas mas adecuadas para el problema en cuestión, como veremos más adelante en la sección 2.4.3.

El problema de fondo en esta clasificación sesegada es que en muchos casos la clase minoritaria es la clase en la que se tiene un mayor interés de predecir de manera correcta, puesto que en general figuran como los casos de interés en la clasificación. A modo de ejemplo, se listan algunos casos en los que la predicción tiene esta característica:

- **Detección de fraudes**: Para muchas empresas del rubro financiero es necesaria la detección oportuna de transacciones fraudulentas para tomar medidas correctivas, como por ejemplo los fraudes en tarjetas de crédito. Para estos casos es aplicable un modelo de minería de datos para identificar cuando una transacción, o una ca-

dena de estas es sospechosa, por sus atributos (montos, horas, ventanas de tiempo entre ellas, etc). Como es de esperar, en estos casos, el número de transacciones fraudulentas corresponden a un porcentaje muy bajo respecto de la cantidad total de transacciones.

- **Credit Scoring**: Este problema consiste principalmente en la evaluación del riesgo crediticio de los clientes, es decir, estimar la probabilidad que un cliente, no necesariamente nuevo en la empresa, no pague el crédito, o como es conocido en el rubro, caiga en *default*, basándose en el comportamiento de clientes anteriores y las realizaciones de sus créditos asignados. Este problema se da con mucha frecuencia en las entidades financieras que intentan predecir si un postulante a crédito es buen o mal pagador según sus carácteristicas demográficas y financieras (edad,sexo,años de antigüedad en su trabajo actual, sueldo, etc.) para decidir si resulta rentable o no otorgar el crédito. Actualmente la cantidad de clientes considerados buenos pagadores es ampliamente mayor a los clientes con mal comportamiento de pago, y precisamente el interés radica en identificar a estos últimos.

- **Fuga de clientes**: Análogo a lo que ocurre en el caso de *Credit Scoring*, la fuga de clientes intenta identificar de forma anticipada a los clientes que tienen mayor tendencia a abandonar la compañía en el siguiente período (algunos modelos consideran más de un período para la predicción, ver Blattberg et al. (2008)). Esta problemática se replica en varios rubros, como telecomunicaciones, entidades bancariaes, proveedores de internet, y empresas de servicios en general. El objetivo que persigue la predicción oportuna es evitar que la fuga ocurra. En este caso también se da que los casos de fuga son ampliamente superados en número por los casos en donde los clientes permanecen en la misma compañía por un largo período de tiempo.

  Similarmente a lo que sucede en credit scoring, fuga de clientes intenta anticipar cuáles son los clientes que abandonarán la compañía en el periodo siguiente. Se debe mencionar que esta problemática se replica en variados rubros, como las telecomunicaciones, los bancos, proveedores de internet, etc; y el objetivo final es aplicar metodologías correctivas para evitar que estos casos ocurran. Nuevamente, los casos de fugas son ampliamente superados por los casos en los que los clientes permanecen en la misma compañía por muchos años. Además según plantean varios autores, captar un cliente nuevo resulta alrededor de 6 veces más costoso que retener un cliente antiguo.

Una forma de atacar el problema de desbalance de clases, es alterar de manera controlada el desbalance de clases en el conjunto den entrenamiento. Acá se pueden apreciar dos opciones principales, eliminar un porcentaje de observaciones de la clase mayoritaria, o bien crear artificialmente observaciones en la clase minoritaria. Estas estrategias son llamadas *Undersampling* y *Oversampling* respectivamente, y serán detalladas a continuación.

2.5.1. Undersampling

El objetivo de fondo de esta técnica es la modificación del conjunto de entrenamiento, de modo que que algoritmo de clasificación aprenda mejor de la clase minoritaria, y con esto generar una capacidad de predicción depurada sobre la clase de interés.

Básicamente esta técnica consiste en eliminar de manera aleatoria observaciones de la clase mayoritaria, para en cierto modo equilibrar la distribución de clases (Batista et al, 2004) En la figura 2.9 se ve de manera explícita lo que realiza esta técnica, logrando un balance en la proporción de las clases.

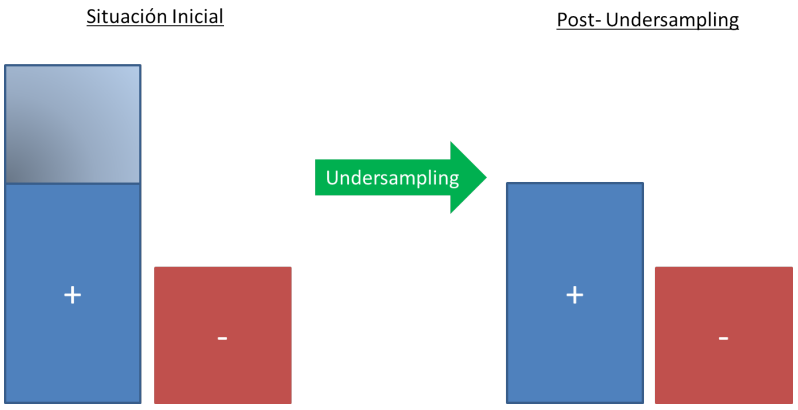

Figura 2.9: Esquema Undersampling, elaboración propia

En la figura anterior, en el lado izquierdo de la imagen, se muestra un conjunto de observaciones en el cual la clase positiva supera de manera amplia a la clase negativa. Luego de aplicar la técnica de *Undersampling* se puede apreciar al lado derecho el conjunto de datos restante que presenta una cantidad de observaciones por clase más equilibrada.

Es importante notar que el objetivo final del *Undersampling* no es equiparar la cantidad de observaciones, si no más bien balancear el tamaño de ambas mediante la eliminación de elementos de la clase mayoritaria.

Esta técnica no está exenta de inconvenientes, ya que se está eliminando información potencialmente útil para la efectividad predictiva del modelo de clasificación (ver Batista et al. (2004)), esto debido a que el modelo aprende sólo de un subconjunto de los datos originales. Por otro lado, al realizar *undersampling* deja de ser considerada aleatoria (Kotsiantis and et al. (2006)) lo que se introduce un sesgo en la muestra finalmente obtenida.

2.5.2. Oversampling (SMOTE)

Análogamente al *Undersampling*, el *Oversampling* tiene como objetivo inducir un balance entre las clases del conjunto de entrenamiento, de forma que el clasificador pueda aprovechar de mejor manera la información de la clase minorataria, lo señalado puede ser visto en el siguiente esquema:

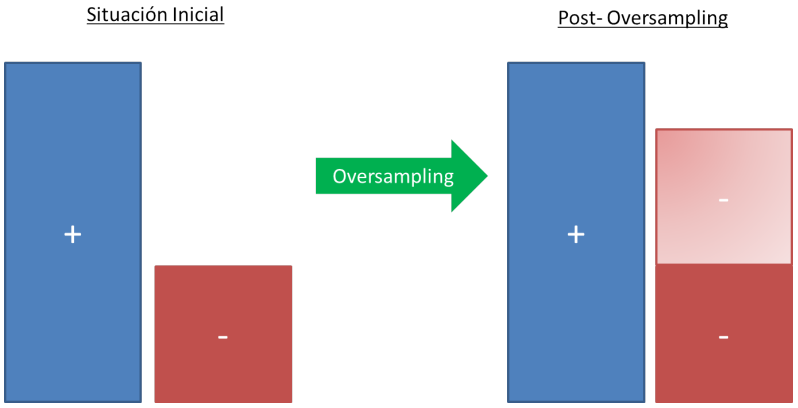

Figura 2.10: Esquema Oversampling, elaboración propia

En la figura anterior, se observa como se produce un aumento de la clase minoritaria entre el lado izquierdo al lado derecho a raíz del *Oversampling*. Una manera práctica de realizar esta técnica consiste en crear observaciones sintéticas pertenecientes a la clase minoritaria interpolando los atributos de de un subonjunto de observaciones de dicha clase (Chawla (2010)), para este efecto, se usa ***SMOTE Oversampling***, cuya representación gráfica se puede apreciar a continuación:

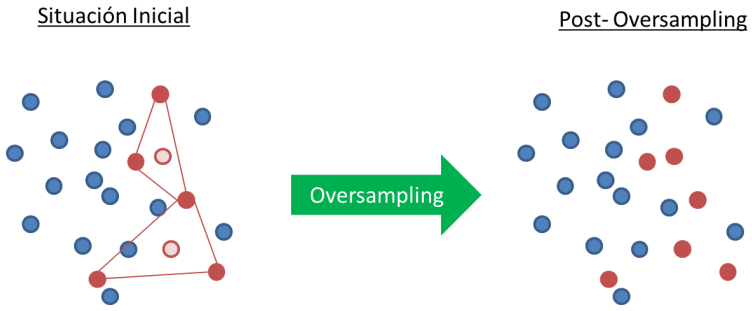

Figura 2.11: SMOTE. Se puede ver en el lado izquierdo la base original, y con trazos rojos la envoltura convexa generada por dos tríos de conjuntos de elementos de la clase minoritaria. En el lado derecho, se creó una observación sintentica para cada uno de esos tríos, lo que incrementa la cantidad de observaciones de la clase minoritaria.

En el esquema anterior, se crean observaciones siguiendo la siguiente lógica: se escogen

tres observaciones de manera aleatoria de la clase minoritaria, de las cuales mediante interpolación con algún criterio para cada atributo se crea una nueva observación sintética perteneciente a la clase minoritaria. Este proceso se repite hasta tener la cantidad deseada de observaciones de la clase minoraitaria. Usualmente en la literatura la cantidad de vecinos utilizados para crear una nueva observación es k=5.

Al igual que en el caso de *Undesampling*, esta técnica induce un sesgo en la muestra, y adicionalmente los elementos sintéticos podrían en principio solo introdcur ruido a la construcción del modelo de aprendizaje. Adicionalmente, dependiendo de la naturaleza de la base y del método de interpolación, la complejidad y el tiempo de aprendizaje puede incrementar de manera significativa.

2.6. Selección de Atributos

A lo largo de este documento, se ha enfatizado la importancia de la selección de atributos, tanto para mejorar el desempeño de los clasificadores, como para un mejor entendimiento del modelo final. Pero no se había explicitado las formas de llevar a cabo esta importante tarea. En esta sección se hará una revisión del estado del arte en lo que a selección de parámetros se refiere, y se explican los tres enfoques principales Guyon et al. (2006), así como los métodos que serán usados en este trabajo de tesis.

2.6.1. Métodos de Filtro (*Filter Methods*)

La aplicación de este enfoque, ocurre antes de aplicar cualquier algoritmo de clasificación, y usa propiedades estadísticas de los atributos, con el objetivo de dejar fuera los que aportan menos información (en algún sentido) al modelo. Ejemplos clásicos de la literatura, son **el estadístico de** χ^2, que mide que tan independiente es la distribución de cada atributo, respecto de las etiquetas de las observaciones (Van Hulse et al. (2009)), la **la Ganancia de Información** que usa la entropía para decidir que tan relevante es un atributo, como se puede ver en Van Hulse et al. (2009), y finalmente el **Fisher Score (**F**)**, que estima la relevanciia de cada atributo independiente del resto, calculando de manera explícita la diferencia absoluta entre las medias del valor del atributo en ambas clases, y normaliza según la suma de las varianzas intra-clases, para poder comparar los indicadores obtenidos para cada atributo (ver Guyon et al. (2006)), lo que se resume en la siguiente ecuación:

$$F(j) = \left| \frac{\mu_j^+ - \mu_j^-}{(\sigma_j^+)^2 + (\sigma_j^-)^2} \right| \tag{2.36}$$

en donde μ_j^+ (μ_j^-) es la media del j-ésimo atributo en la clase positiva (negative) y σ_j^+ (σ_j^-) es la correspondiente desviación estándar.

A raíz de este indicador, es posible observar que atributos difieren más entre clases, luego el **Método de Fisher** para Selección de atributos consiste en obtener el indicador señalado en la ecuación 2.36, y elegir la cantidad de atributos deseada que tengan mejor *Fisher Score*, o bien decidir un valor umbral para el score obtenido por cada atributo y sobre ese valor dejar el atributo en el modelo.

2.6.2. Métodos de Envultura (Wrapper Methods)

Estos métodos recorren el conjunto de atributos de manera completa, buscando entre todos los posbiles subconjuntos de atributos evaluando su potencial predictivo, lo que es altamente demandante en términos computacionales, puesto que tiene un tamaño exponencial en el tamaño de la entrada, pero en la mayoría de los casos los resultados son mejores que los métodos de Filtro (Guyon et al. (2006)). Las estrategias más populares para llevar a cabo esta tarea son *Sequential Forward Selection (SFS)* y *Sequential Backward Elimination (SBE)*. SFS empieza con un conjunto vaciío de atributos, y luego intenta agregar atributos de manera secuencial, donde en cada etapa se agrega el atributo mas relevante (de acuerdo a algún método de clasificación en particular) del conjunto de atributos pendientes por agregar. SBS es exactamente lo contrario, empieza con el espacio completo de atributos, y calcula la significancia estadística de cada uno, eliminado en cada iteración el menos significativo.

2.6.3. Métodos Empotrados (Embedded Methods)

Estos modelos realizan la selección de atributos de manera simultánea con la construcción del clasificador. Lo que implica considerar en la construcción el espacio combinado de hipótesis y atributos. De manera similar a los métods *Wrapper*, este enfoque es específico para cada método de clasificación, y por esta razón incluye la relación de dependencia de los atributos en el clasificador. Sin embargo, los métodos *embedded* son computacionalmente menos intensivos que los métodos *wrapper* (ver Guyon et al. (2006)).

Estos métodos también pueden ser vistos como un problema de optimización. Esto se puede hacer introduciendo la selección de atributos en los parámetros del modelo de manera directa, por ejemplo considerando un término de dispersión. A modo de ejemplo, la norma Euclidiana en la formulación usual de SVM (detallada en la ecuación 2.10) puede ser cambiada por la norma l_1 , es decir $\Omega(\mathbf{w}) = \sum_i |w_i|$, como se puede apreciar en el enfoque l_1-SVM presentado en Bradley and Mangasarian (1998).

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \sum_{j=1}^{n} |w_j| + C \sum_{i=1}^{m} \xi_i \\ \text{s.a.} \quad & y_i \cdot (\mathbf{w}^T \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, m, \\ & \xi_i \geq 0, \qquad i = 1, \ldots, m. \end{aligned} \tag{2.37}$$

Otra alternativa es considerar minimizar la "norma cero": $\Omega(\mathbf{w}) = \|\mathbf{w}\|_0 = |\{i : w_i \neq 0\}|$. Cabe notar que $\|\cdot\|_0$, a diferencia de la norma l_1 , no es una norma puesto que la desigualdad triangular no se cumple (Bradley and Mangasarian (1998)). En Weston et al. (2003) se propone un enfoque para minimizar la "norma cero" (l_0-SVM) escalando las variables de manera iterativa, multiplicándolas por el valor absoluto del peso para cada una de ellas, $\mathbf{w}$ que se obtiene de la formulación de SVM, hasta lograr convergencia. De esta manera, las variables pueden ser ordenadas y remover los atributos cuyos pesos se aproximan a cero durante el algoritmo iterativo. Este método considera la siguiente aproximación de la norma l_0:

$$\Omega(\mathbf{w}) = \sum_{j=1}^{n} log(\epsilon + |w_j|). \tag{2.38}$$

con ϵ una constante positiva pequeña.

2.6.3.1. RFE-SVM, y sus variantes

Un método muy popular, que es relevante para el desarrollo de este trabajo, es *recursive Feature Elimnation* (RFE-SVM, ver Guyon et al. (2002)). El objetivo de este enfoque es encontrar un subconjunto de tamaño r entre n variables (con $r < n$), eliminando aquellos atributos cuya extracción contribuye a alcanzar el mayor margen de separación entre clases. Esto puede ser logrado usando una estrategia *backward elimination*, eliminado de manera secuencial basándose en las componentes del vector de pesos en SVM, $\overrightarrow{w}$. Que en el caso lineal toma la siguiente forma:

Algoritmo 1 *Recursive Feature Elimination, SVM* - Caso Lineal

1. **repetir**
2. $\mathbf{w} \leftarrow$ Entrenamiento SVM (formulación primal).
3. Eliminar el atributo p con el valor más pequeño de $|w_p|$.
4. **hasta** reducir los atributos a r.

Cabe notar que el caso RFE-SVM lineal, expuesto en Guyon et al. (2002), puede ser generalizado al caso no-lineal (es decir, usando funciones de Kernel), y este hecho es la base

de las metodologías que se propondrán en el siguiente capítulo. Dado que la distancia de los hiperplanos canónicos que separan ambas clases (el **margen**) es inversamente proporcional a la norma del vector de pesos $\overrightarrow{w}$, este valor puede ser escrito en términos de las variables duales del modelo generado por SVM, tomando la siguiente forma funcional:

$$W^2(\boldsymbol{\alpha}) = \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s) \tag{2.39}$$

El atributo removido en cada iteración, es aquel cuya eliminación minimice la variación de $W^2(\boldsymbol{\alpha})$. El algoritmo queda descrito como sigue:

Algoritmo 2 Recursive Feature Elimination, SVM - Caso no Lineal

1. **repetir**
2. $\mathbf{w} \leftarrow$ Entrenamiento SVM (formulación dual).
3. Eliminar el atributo p con menor valor de $|W^2(\boldsymbol{\alpha}) - W^2_{(-p)}(\boldsymbol{\alpha})|$.
4. **hasta** reducir los atributos a r.

Si bien es posible remover una variable a la vez en cada iteración, esto es altamente ineficiente cuando el problema presenta una gran dimensionalidad (por ejemplo datos de *microarray*), en donde las bases de datos cuentan con miles de atributos. En estos casos usualmente se remueve la mitad de los atributos en cada iteración (Guyon et al. (2002)). Un caso de éxito de la utilización de RFE-SVM que vale la pena destacar, se puede observar en Hidalgo-Munoz et al. (2013), donde los autores confirman que este método entrega información útil y precisa para realizar diagnósticos médicos.

2.6.3.2. BFE-SVM

A continuación se detalla un algoritmo base que permitirá hacer diferentes métodos de selección de parámetros, básicamente usando diferentes funciones de pérdida. Siguiendo la notación empleada en Song et al. (2012), denotamos como $\mathcal{F}$ el conjunto completo de atributos. Queremos encontrar un subconjunto $\mathcal{K}$ ($\mathcal{K} \subseteq \mathcal{F}$) de atributos, de tal manera que el desempeño del clasificador basado en SVM se maximice, considerando un conjunto de entrenamiento $\mathcal{T}$.

Para poder comparar el desempeño de las diferentes estrategias de selección de atributos, se genera una lista de los atributos en un orden creciente de relevancia ($\mathcal{F}^{\dagger}$). Esto es logrado por el siguiente algoritmo iterativo (Maldonado et al. (2014a)):

Algoritmo 3 Algoritmo *Backward Feature Elimination* (BFE-SVM)

Input: El conjunto original de atributos $\mathcal{F}$
Output: Un vector ordenado de atributos $\mathcal{F}^\dagger$

1. $\mathcal{F}^\dagger \leftarrow \emptyset$
2. **repetir**
3. $\quad \boldsymbol{\alpha} \leftarrow$ Entrenamiento SVM usando $\mathcal{T}$
4. $\quad \mathcal{I} \leftarrow \text{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \text{LOSS}(\boldsymbol{\alpha}, \mathcal{F} \setminus \{j\}),\ \mathcal{I} \subset \mathcal{F}$
5. $\quad \mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{I}$
6. $\quad \mathcal{F}^\dagger \leftarrow (\mathcal{F}^\dagger, \mathcal{I})$
7. **hasta** $\quad \mathcal{F} = \emptyset$

En el cuarto paso, el agoritmo determina un conjunto $\mathcal{I}$ de atributos a ser eliminados.Al igual que en el caso anterior, remover un sólo atributo el conjunto $\mathcal{F}$, es ía ineficiente cuando hay un gran número de atributos irrelevantes. Por otro lado, remover muchos atributos a la vez aumenta el riesgo de perder atributos relevantes para la clasificacón (Guyon et al. (2002)), en este trabajo una buena combinación entre velocidad y calidad de elección de atributos, fue remover en torno al 10 % de los atributos presentes en cada iteración.

Es posible utilizar diferentes funciones de pérdida (**LOSS**). En este trabajo se usaron las siguientes:

- **Función de pérdida binaria:** Esta medida está basada en el conteo del total de errores que ocurren en el conjunto de entrenamiento. Asume que los costos de los errores son iguales y por esta razón no es adecuada para conjuntos con alto desbalance de clases. Usando esta métrica queremos validar que la relevancia de atributos debe ser medida considerando difrentes costos para diferentes tipos de errores. Más formalmente, dada una solución $(\boldsymbol{\alpha}, b)$, obtenida al aplicar SVM al conjunto de entrenamiento $\mathcal{T}$, y co un conjunto de atributos $\mathcal{F}$, la función de pérdida binaria (*0-1 loss*) para un atributo $j \in \mathcal{F}$, se define como:

$$\text{LOSS}_{0-1}((\boldsymbol{\alpha}, b), \mathcal{F} \setminus \{j\}, \mathcal{T}) = \sum_{s \in \mathcal{T}} \left| y_s - sgn\left(\sum_{i \in \mathcal{T}} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right| \tag{2.40}$$

  en donde $\mathbf{x}_i^{(-j)}$ el la observación i en donde el atributo j es removido y sgn corresponde a la función signo.

- **Función de pérdida balanceada:** Para poder hacer comparables los errores en una base de datos desbalanceada, se usa una función de perdida balanceada, que toma el peso ponderado de los errores de cada tipo para generar un indicador único. Dicho esto, la

función de pérdida balanceada (*balanced loss*) para un atributo $j \in \mathcal{F}$, se define como:

$$\begin{aligned}\text{LOSS}_{bl}((\boldsymbol{\alpha}, b), \mathcal{F} \setminus \{j\}, \mathcal{T}) = &\frac{\sum\limits_{s \in \mathcal{T}^-} \left| y_s - sgn\left(\sum\limits_{i \in \mathcal{T}^-} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right|}{|T^-|} + \\ &\frac{\sum\limits_{s \in \mathcal{T}^+} \left| y_s - sgn\left(\sum\limits_{i \in \mathcal{T}^+} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right|}{|T^+|}\end{aligned} \tag{2.41}$$

en donde $|T^+|$ ($|T^-|$) representan la cantidad de observaciones de T^+ (T^-).

- **Función de pérdida predefinida**: Se puede usar adicionalmente una función de costos o pérdida predefinida que rescate la relación de costos y beneficios del problema en cuesti'on, o bien usar alguno de los indicadores usados en la literatura de forma incremental como se señala en los dos casos anteriores, eligiéndo el atributo que maximice (o minimice según corresponda) la diferencia entre considerar o no cada atributo. En este trabajo de Tesis consideraremos funciones de pérdida basadas en utilidades, como fueron vistas en la sección 2.4.3, la explicación y justificación detallada se puede ver en el siguiente capítulo (Capítulo 3).

En el siguiente capítulo, se usarán estas funciones de pérdida con algunas modificaciones que permiten incluir tanto en la selección de atributos como en la construcción del modelo las funciones basadas en utilidades, para definir un método de selección de atributos, parámetros y construcción de un modelo robusto.

Capítulo 3

Metodología

En este capítulo se detalla la metodología propuesta para realizar los experimentos, en primera instancia utilizaremos y describiremos como aplica el proceso KDD (Sección 2.2), que detalla los pasos a seguir para extraer de forma eficiente conocimiento a partir de bases de datos de gran dimensión. Luego se explica el diseño experimental en si mismo (Sección 3.1), y finalmente el algoritmo propuesto para evaluar el rendimiento de cada uno de los métodos de selección de atributos mencionados en la sección 2.6 del capítulo anterior, y definiremos los métodos asociados a las métricas basadas en utilidades.

A continuación, se detalla el diseño experimental, que comprende el proceso KDD y las técnicas revisadas en el Capiítulo 2.

3.1. Diseño experimental

El diseño experimental es directo de lo expuesto en la sección anterior, se usa la metodología KDD, que es una técnica con éxito comprobado en *Business Analytics*, tanto en predicción de fuga como en *credit scoring*, ver Bravo et al. (2013), siguiendo los pasos descritos en la sección previa, se detalla la realización de estos en este trabajo de Tesis.

- **Recopilación y consolidación de datos:** El primer paso consiste en identificar las fuentes relevantes de datos y consolidarlos en un repositorio único creado para la construcción de los modelos de clasificación. Este paso fue directo puesto que todas las bases de datos ya estaban consolidadas.

- **Pre-procesamiento de datos:** El siguiente paso, es la remoción de valores perdi-

dos, y transformación de datos. Estos pasos también fueron directos puesto que las bases ya estaban preprocesadas para el análisis.

- **Minado de datos:** Se siguió el siguiente procedimiento para realizar el ranking de atributos y la selección de hiperparámetros del modelo: Para cada una de las bases, los conjuntos de entrenamiento y testeo fueron generados usado una Validación Cruzada en 10 partes, que es una técnica común en la predicción de fuga de clientes (Verbeke et al. (2012)). Dicho esto, se realiza tanto el ranking de atributos como la clasificación de las observaciones en el conjunto de entrenamiento. Y el desempeño de la clasificación final es calculado promediando los resultados de la clasificación de los diferentes conjuntos de testeo, usando tanto métricas usuales como indicadores de rendimiento basados en utilidades. El conjunto de entrenamiento a su vez fue procesado con técnicas para corregir en cierta medida el desbalance natural que presentan las bases de datos. Para este efecto, se estudiaron dos alternativas, *Undersampling* aleatorio, y una combinación de *Undersampling* y *SMOTE Oversmapling*. La selección del modelo fue realizada a través de una búsqueda en grilla tomando diferentes valores de hiperparámetros, se consideraron los siguientes valores asociados al modelo SVM, $C \in \{2^{-7}, ..., 2^{7}\}$ and $\sigma \in \{2^{-7}, ..., 2^{7}\}$, usando un Kernel RBF.

- **Evaluación de los resultados:** El desempeño de todos los métodos fue estudiado para diferentes valores de los parámetros y comparados con diferentes indicadores entre ellos para ver cuál es el impacto en el clasificador final, para mayor detalle de este análisis revisar el capítulo 4

Para los diferentes enfoques de SVM se usó LIBSVM (Chang and Lin (2011)) en Matlab principalmente.

3.2. Algoritmo Propuesto

3.2.1. Algoritmo basal

Adicionalmente a los métodos señalados en la sección 2.6 del capítulo 2, se generan nuevos métodos de selección de atributos usando *Support Vector Machines* e inspirados por HOSVM (Maldonado and Weber (2009)). El concepto fundamental que está detrás de estos enfoques es eliminar aquellos atributos cuya extracción tenga menor impacto en la solución final, considerando los costos y beneficios respectivos que un clasificador puede ocasionar (dependiendo de la medida de desempeño considerada). Para lograr esto, nos valemos del objetivo principal, que es encontrar el mejor desempeño predictivo en un conjunto desconocido del conjunto de entrenamiento, usando métricas basadas en utilidades (sean beneficios o costos), a saber MPC, EMPC, y H-Measure , todas vistas de forma

exhaustiva en el capítulo 2, en la sección 2.4.3. A raíz de esto, los métodos de selección de atributos propuestos toman los nombres de SVM_{MPC}, SVM_{EMPC}, y SVM_{H}, respectivamente, de acuerdo a qué métrica de desempeño es utilizada. Para esto, es necesario recordar el algoritmo visto en la sección 2.6.3.2:

Algoritmo 4 Algoritmo *Backward Feature Elimination* (BFE-SVM)

Input: El conjunto original de atributos $\mathcal{F}$
Output: Un vector ordenado de atributos $\mathcal{F}^{\dagger}$

1. $\mathcal{F}^{\dagger} \leftarrow \emptyset$
2. **repetir**
3. $\quad \boldsymbol{\alpha} \leftarrow$ Entrenamiento SVM usando $\mathcal{T}$
4. $\quad \mathcal{I} \leftarrow \text{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \text{LOSS}(\boldsymbol{\alpha}, \mathcal{F} \setminus \{j\}),\ \mathcal{I} \subset \mathcal{F}$
5. $\quad \mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{I}$
6. $\quad \mathcal{F}^{\dagger} \leftarrow (\mathcal{F}^{\dagger}, \mathcal{I})$
7. **hasta** $\quad \mathcal{F} = \emptyset$

En donde es posible usar una función predefinida para llevar a cabo la elección de los atributos.

En este trabajo, se apuesta por usar las métricas de desempeño basadas en utilidades para llevar a cabo este propósito. Más explicitamente, en el paso 3 del Algoritmo 4, el resultado de entrenar SVM en el conjunto de entrenamiento $\mathcal{T}$, consiste en $\Lambda = (\boldsymbol{\alpha}, b)$, y esta información es usada en la expresión $\text{LOSS}^{(-j)}(\boldsymbol{\Lambda}, \mathcal{T})$, en donde proponemos calcular alguna de las 3 métricas propuestas en la sección 2.4.3, a saber MPC, EMPC, y H-Measure. Intuitivamente, el atributo cuya eliminación lleva a obtener un mejor *Profit* (o un costo menor en el caso de la H-measure) debe ser eliminado de la base de datos. Para adaptar estas métricas hay que notar que los scores obtenidos por el clasificador toman un rol fundamental, puesto que cambian las distrbuciones de probabilidades lo que implica un cambio en el Profit o costo obtenido según corresponda. Dado esto, se define el score para la observación $k \in \mathcal{T}$ cuando el atributo j es removido como:

$$s_k^{(-j)}(\boldsymbol{\Lambda}, \mathcal{T}) = \sum_{i \in \mathcal{T}} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_k^{(-j)}) + b \tag{3.1}$$

donde $\mathbf{x}_i^{(-j)}$ y $\mathbf{x}_k^{(-j)}$ corresponden a las observaciones i y k sin considerar el atributo j. Para evitar cálculos adicionales, y reducir la complejidad computacional, se asume que el vector α es igual a la solución obtenida en el el paso 3 del Algoritmo 3 como es sugerido en Guyon et al. (2002).

Usando esta notación y entendiendo $\mathcal{C}^{(-j)}$ como el clasificador sin el atributo j, considerando los scores descritos en la eucación 3.1 y $\mathcal{F}^{(-j)}$ el espacio de atributos sin considerar el atributo j, se proponen las siguientes funciones de pérdida ($LOSS$):

- Primero, para el *Maximum Profit* tenemos:

$$\text{LOSS}_{MPC}((\mathcal{C}, \mathcal{F} \setminus \{j\}, \mathcal{T}) = \text{MPC}(\mathcal{C}^{(-j)}, \mathcal{T}, \mathcal{F} \setminus \{j\}) \tag{3.2}$$

Luego para el *Expected Maximum Profit*:

$$\text{LOSS}_{EMPC}((\mathcal{C}, \mathcal{F} \setminus \{j\}, \mathcal{T}) = \text{EMPC}(\mathcal{C}^{(-j)}, \mathcal{F} \setminus \{j\})) \tag{3.3}$$

Donde se entiende MPC y $EMPC$ como sigue (ver sección 2.4.3):
Maximum Profit (MPC):

$$\text{MPC} = \max_{\forall t} P_{\mathcal{C}}(t; \gamma, CLV, \delta, \phi). \tag{3.4}$$

Expected Maximum Profit (EMPC):

$$\text{EMPC} = \int_{\gamma} P_{\mathcal{C}}(T(\gamma); \gamma, CLV, \delta, \phi) \cdot h(\gamma) \text{d}\gamma, \tag{3.5}$$

En donde se usa P definido de la siguiente manera (extraído desde la sección 2.4.3):

$$P_{\mathcal{C}}(t; \gamma, CLV, \delta, \phi) = CLV\left(\gamma(1-\delta) - \phi\right)\pi_{-1}F_{-1}(t) - CLV(\delta + \phi) \cdot \pi_1 F_1(t). \tag{3.6}$$

En donde para cualquier precisión de notación, basta revisar la sección 2.4.3.
Y finalmente para la *H-measure*

$$\text{LOSS}_{H}((\mathcal{C}, \mathcal{F} \setminus \{j\}, \mathcal{T}) = \text{H}(\mathcal{C}^{(-j)}, \mathcal{F} \setminus \{j\})) \tag{3.7}$$

Donde se usa la siguiente notación:
H-measure:

$$\text{H} = 1 - \frac{\int_0^1 Q_{\mathcal{C}}(T(c); b, c) \cdot u(c) \text{d}c}{\pi_0 \int_0^{\pi_1} c \cdot u(c) \text{d}c + \pi_1 \int_{\pi_1}^1 (1-c) \cdot u(c) \text{d}c}, \tag{3.8}$$

En donde $u(c)$ es una abreviación de notación de $u_{\alpha,\beta}(c)$. El denominador corresponde a la peŕdida ocasionada por el peor clasificador, que es un clasificador aleatorio y Q está definida por la ecuación 2.25.

3.2.2. Algoritmo usando conjuntos de validación, y resampling de clases

Ahora bien, dado que estamos tratando con bases de datos con alto desbalance de clases, primero redefinimos el algoritmo de *Holdout SVM* para incorporar un paso donde se realizan técnicas de *resampling* para mitigar el efecto del desbalance. En este trabajo se proponen dos estrategias: *Undersampling* aleatorio, y una combinación de *Undersampling* aleatorio y *SMOTE Oversampling* para aumentar el tamaño de la clase minoritaria. El propósito final del algoritmo es encontrar u subconjunto $\mathcal{K}$ ($\mathcal{K} \subseteq \mathcal{F}$) de atributos, de tal manera que el desempeño del clasificador usando este subconjunto de atributos sea maximizado, considerando un conjunto de entrenamiento $\mathcal{T}$. Este conjunto a su vez es particionado en un conjunto de entrenamiento $\mathcal{TR}$ y uno de validación $\mathcal{V}$. Al conjunto de entrenamiento $\mathcal{TR}$ se le realizan las técnicas de *resampling* dando origen a un nuevo conjunto $\mathcal{TR}'$ en donde se construye el clasificador, para lidiar con un conjunto más balanceado que la base original. El conjunto de validación es finalmente usado para construir una función de pérdida no sobre ajustada y adecuada para el problema de predicción de fuga. Esquemáticamente lo recién expuesto se puede ver en el siguiente recuadro:

Algoritmo 5 Algoritmo de *Holdout* para *Backward Feature Elimination* usando SMOTE

Input: El conjunto original de atributos $\mathcal{F}$

Output: Un vector ordenado de atributos $\mathcal{F}^\dagger$

1. $\mathcal{F}^\dagger \leftarrow \emptyset$
2. **repetir**
3. $(\mathcal{TR}, \mathcal{V}) \leftarrow$ Holdout usando $\mathcal{T}$
4. $\mathcal{TR}' \leftarrow$ Resampling($\mathcal{TR}$)
5. $\mathbf{\Lambda} \leftarrow$ Entrenamiento SVM usando $\mathcal{TR}'$
6. $\mathcal{I} \leftarrow \text{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \text{LOSS}^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V}),\ \mathcal{I} \subset \mathcal{F}$
7. $\mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{I}$
8. $\mathcal{F}^\dagger \leftarrow (\mathcal{F}^\dagger, \mathcal{I})$
9. **hasta** $\mathcal{F} = \emptyset$

El clasificador entrenado sobre el conjunto $\mathcal{TR}'$ en el paso 5 del algoritmo recién descrito corresponde al par ordenado $\Lambda = (\boldsymbol{\alpha}, b)$, y esta información se usa para computar la pérdida en el siguiente paso, a saber $\text{LOSS}^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V})$, lo que a diferencia de el algoritmo 3, se realiza sobre un conjunto de validación. Proponemos entonces calcular las medidas MPC, EMPC y H-Measure usando el subconjunto $\mathcal{V}$ cuando el atributo j es eliminado. Luego de manera intuitiva, el atributo cuya eliminación nos lleve a obtener el beneficio mayor (o menor costo en el caso de H-Measure) debe ser eliminado de la base. Para adaptar estas métricas, notemos que nuestra propuesta solo difiere de las versiones originales de MPC, EMPC y H-Measure en el cálculo de los *scores* para cada observación, lo que cambia las distribuciones de probabilidad. Mientras que los costos y beneficios de una solución dada, como la definición de γ no se ven afectados. Usando esto, y de manera análoga a lo expuesto en la ecuación 3.1, definimos $s_k^{(-j)}$ como el *score* de la observación $k \in \mathcal{V}$ cuando el atributo j es eliminado:

$$s_k^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V}) = \sum_{i \in \mathcal{TR}'} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_k^{v(-j)}) + b \tag{3.9}$$

donde $\mathbf{x}_i^{(-j)}$ corresponde a una observación del conjunto de entrenamiento originado por el *resmapling* cuando el atributo j es eliminado y $\mathbf{x}_k^{v(-j)}$ corresponde a un objeto de validación k con el atributo j eliminado. Análogamente a lo expuesto la ecuación 3.1, para reducir la complejidad computacional del algoritmo, el vector α se asume que es igual a la solución obtenida en el paso 5 del Algoritmo 5 como se sugiere en Guyon et al. (2002).

Luego de lo anterior, se proponen las siguientes métricas de desempeño basadas en utlidades para el paso 6 del algoritmo 5, donde la única diferencia respecto a la definición original de las métricas es la inclusión de la fórmula para $s^{(-j)}$:

- **H measure**:

$$\text{LOSS}^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V}) = \text{H}(s^{(-j)}) \tag{3.10}$$

- **Maximum Profit (MPC)**:

$$\text{LOSS}^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V}) = \text{MPC}(s^{(-j)}) \tag{3.11}$$

- **Expected Maximum Profit (EMPC)**:

$$\text{LOSS}^{(-j)}(\mathbf{\Lambda}, \mathcal{TR}', \mathcal{V}) = \text{EMPC}(s^{(-j)}) \tag{3.12}$$

Usar estas funciones de pérdida en el paso 6 del algoritmo 5 genera tres variantes del enfoque propuesto, explicitamente HOSVM$_H$, HOSVM$_{MPC}$, y HOSVM$_{EMPC}$, que serán evaluadas en su desempeño de manera comparativa con las otras metodologías de selección de atributos descritas en la sección 2.6 del capítulo 2.

Finalmente, en el paso 6 el algoritmo determina un conjunto de atributos a ser eliminado ($\mathcal{I}$). Si bien es posible eliminar un sólo elemento en cada iteración, esto es ineficiente dado que en general existe un número considerable de atributos irrelevantes. Por otro lado, remover muchos atributos a la vez aumenta el riesgo de eliminar atributos relevantes (Guyon et al. (2002)).

Capítulo 4

Resultados

En este capítulo se analizan los resultados de tres problemas de predicción de fuga de clientes, y diversas técnicas de selección de atributos. Primero describiremos las bases de datos que usamos, que fueron previamente trabajadas como se explica en el capiítulo 3. Luego se procede a presentar los resultados directamente.

4.1. Descripción de las Bases de Datos

Las tres bases de datos que se usaron usadas para clasificación binaria sobre conjuntos con desbalance de clases se describen a continuación:

- **UCI-Telecom**: Esta base de datos de clientes fue extraída desde el repositorio UCI y contiene la información de 5,000 clientes de una compañía de telecomunicaciones, descritos por 20 atributos.
- **Operator 1**: Esta base de cliente de telecomunicaciones fue originalmente estudiada por Mozer et al. (2000), y contiene 47,761 clientes descritos por 47 variables. Esta base de datos fue además usada para realizar *benchmark* de métodos de *Machine Learning* en Verbeke et al. (2012) bajo el nombre de Operator 1 (O1).
- **Cell2Cell**: Esta base de datos fue propuesta en Duke University (2014) como caso de estudio, contiene información de 20,406 clientes descritos por 73 variables, también fue usada para realizar *benchmark* de métodos de *Machine Learning* en Verbeke et al. (2012) bajo el nombre de D2.

La siguiente tabla 4.1 resume la información relevante de cada base de datos

Base de Datos	#variables	#obs(min.,may.)	tasa de churn
UCI-Telecom	20	(707;4,293)	16.5 %
Operator 1	47	(1,761;46,000)	3.8 %
Cell2Cell	73	(406;20,000)	2.0 %

Cuadro 4.1: Número de variables, número de observaciones de cada clase y tasa de *churn* para cada una de las 3 bases.

4.2. Presentación de Resultados

En esta sección se presenta un resumen de los resultados para facilitar la evaluación de la mejor instancia de cada uno de los distintos enfoques. En las tablas 4.2, 4.3, y 4.4 se resume el desempeño medio entre diferentes conjuntos de atributos para cada método en las bases de datos Telecom1, Cell2Cell, y Operator1 respectivamente.

Para este efecto, consideramos las siguientes métricas: AUC, EMPC, MPC y H-measure. Las métricas EMPC y MPC están medidas en Euros (€) por cliente. El mejor desempeño entre todos los métodos se encuentra en negrita. Adicionalmente se destaca con un arterisco en donde el desempeõ es significativamente más bajo que el mejór método a un 10 % de nivel de significancia estadística, con doble arterisco a un 5 % de significancia estadística, y con 3 arteriscos a 1 %. Un test-t es usado para hacer comparaciones entre las medias de pares de enfoques y el mejor método para cada base de datos. Los resultados son mostrados para la mejor estrategia de *resampling*, que fue al fin de cuentas *undersampling* aleatorio en cada base de datos.

	Fisher	RFE	HOSVM$_{EMPC}$	HOSVM$_{H}$	HOSVM$_{MPC}$
AUC	62.4**	63.5*	64.5	64.4	**64.6**
EMPC	2.21**	2.45	2.58	2.55**	**2.61**
MPC	2.06**	2.36	2.51	2.49**	**2.55**
H	0.064**	0.089	0.092	0.085	**0.094**

Cuadro 4.2: Desempeño medio para todos los métodos e indicadores, para la base Telecom1

	Fisher	RFE	HOSVM$_{EMPC}$	HOSVM$_{H}$	HOSVM$_{MPC}$
AUC	64.81***	94.09	94.09	94.13	**94.13**
EMPC	0.224***	**0.879**	0.860	0.860	0.859
MPC	0.223***	**0.876**	0.859	0.860	0.859
H	0.097***	**0.462**	0.429	0.381	0.385

Cuadro 4.3: Desempeño medio para todos los métodos e indicadores, para la base Cell2Cell

	Fisher	RFE	HOSVM$_{EMPC}$	HOSVM$_{H}$	HOSVM$_{MPC}$
AUC	49.65**	54.35	54.49	**55.18**	54.47
EMPC	0.006	0.006	**0.008**	0.007	0.007
MPC	0.005	0.006	0.007	**0.007**	0.006
H	0.001***	0.001	**0.002**	0.001	0.002

Cuadro 4.4: Desempeño medio para todos los métodos e indicadores, para la base Operator1

De la Tabla 4.2, para la base Telecom1, observamos que el método propuesto usando la métrica MPC para eliminación de atributos y construcción del clasificador tiene mejor desempeño para todos los indicadores considerados. El método superó a la selección vía *Fisher Score* con un nivel de significancia del 5 % en todos los indicadores y al método RFE-SVM a un nivel de 10 % de significancia en AUC. Mientras que el método HOSVM con EMPC nunca es significativamente peor que el mejor de los métodos para todas las métricas consideradas.

Para la Tabla 4.3, que corresponde a la base Cell2Cell, el método propuesto HOSVM con MPC tiene mejor AUC en general, mientras que RFE usando SVM tiene mejor desempeño para el resto de las métricas. Una vez más *Fisher Score* fue superado por todos los métodos con un nivel de significancia del 1 % en todas las métricas, mientras que los otros métodos nunca se comportaron significativamente peor que el mejor método.

Adicionalmente, para la Tabla 4.4, que corresponde al la base Operator1, el método propuesto de HOSVM basado en EMPC tiene mejor desempeño para las métricas EMPC y H-measure, mientras que HOSVM basado en H-measure alcanza mejores resultados tanto en AUC como en MPC. Una vez más, notamos que *Fisher Score* es superado en AUC (significancia del 5 %) y en el caso de H-measure alcanza un 1 % de significancia, mientras que los otros métodos nunca están significativamente por debajo del mejor métodos para todas las métricas.

Ahora bien, para poder analizar el comportamiento de la selección de atributos para diferentes subconjuntos de variables, presentamos los gráficos 4.1, 4.2 y 4.3, que resumen el mejor desempeño para un número creciente de atributos elegidos para las 3 bases de datos. Para cada subconjunto de atributos, la media de AUC para los métodos *Fisher Score*, RFE-SVM y el mejor método propuesto para cada métrica (a saber, HOSVM basado en MPC para Telecom1 y Cell2Cell, y HOSVM basado en H-measure para Operator1). Los resultados son mostrados también para la mejor estrategia de resampleo.

En el gráfico 4.1 (Telecom1), se puede observar que el método de selección de atributos propuesto (HOSVM basado en MPC), alcanza el mejor desempeño AUC 0.648 con 18 atributos, y luego suavemente disminuye su desempeño. A diferencia de *Fisher Score* o bien RFE-SVM, que disminuyen abruptamente su desempeño en la medida que se van removiendo atributos.

En el caso de Cell2Cell, en el gráfico 4.2, podemos apreciar un comportamiento similar entre RFE-SVM y el método propuesto para esta base (HOSVM basado en MPC), sin embargo el mejor

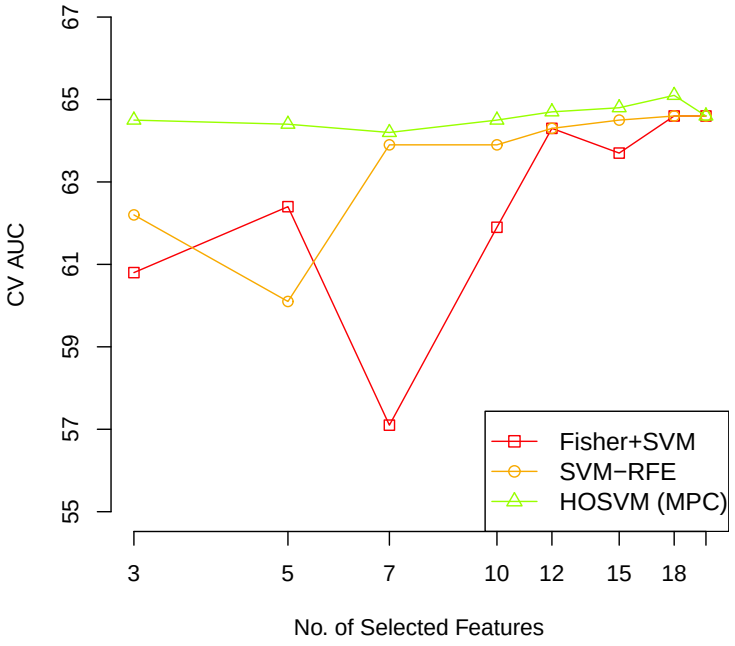

Figura 4.1: AUC versus el número de variables rankeadas para diferentes enfoques de selección de atributos. Para la base Telecom1.

desempeño es obtenido usando este último, llegando a un AUC de 0.948 con 55 atributos. *Fisher Score*, por otro lado remueve atributos relevantes luego de los 55 atributos, lo que disminuye su rendimiento de manera significativa.

Finalmente, en el gráfico 4.3, que corresponde a Operator1, se observa una ganancia importante en el desempeño comparado con el modelo con todos los atributos. En este caso el mejor desempeño es alcanzado usando el método recomendado (HOSVM basado en H-measure), con un AUC de 0.595, mientras que los métodos alternativos si bien mejoran su rendimiento al hacer selección de atributos, lo hacen con menor precisión y de manera poco estable.

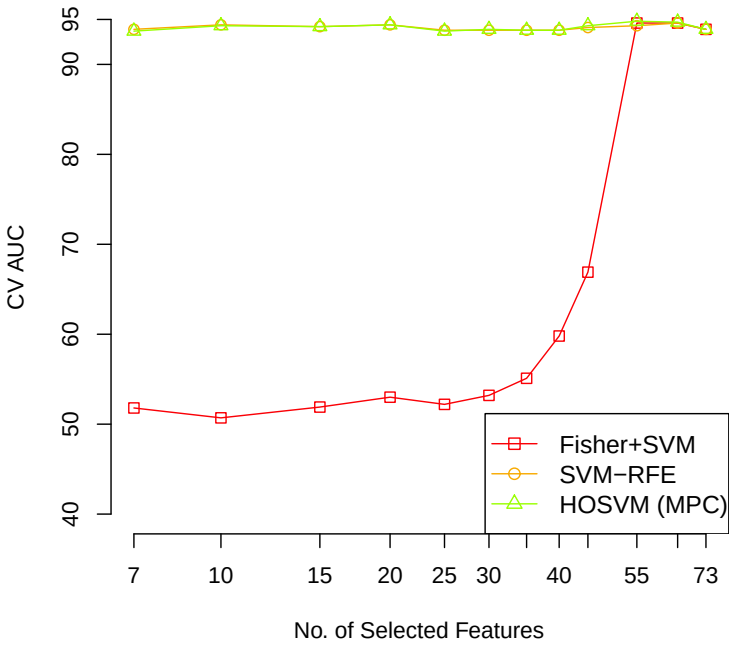

Figura 4.2: AUC versus el número de variables rankeadas para diferentes enfoques de selección de atributos. Para la base Cell2Cell.

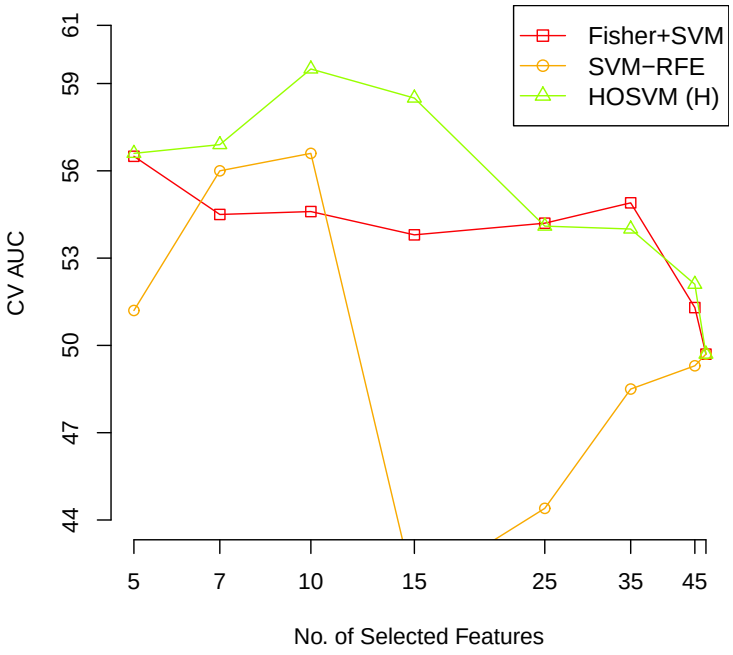

Figura 4.3: AUC versus el número de variables rankeadas para diferentes enfoques de selección de atributos. Para la base Operator1.

Capítulo 5

Conclusiones y Trabajo Futuro

5.1. Conclusiones

En este trabajo se presenta un enfoque de eliminación de atributos de manera recursiva, y empotrado en la construcción misma del modelo de clasificación usando SVM. El método propuesto estudia tres diferentes variantes, y a su vez tres medidas de desempeño adecuadas para problemas con desbalance de clases, y en particular, aplicados a la fuga de clientes: la *H-measure*, el *Maximum Profit Measure for Costumer Churn (MPC)*, y el *Expected Maximum Profit Measure for Costumer Churn (EMPC)*. Mientras que *H-measure* provee una estructura capaz de considerar de manera explícita los costos de clasificar de forma incorrecta como medida de la capacidad predictiva de un modelo, como se puede ver en Hand (2009), la medida MPC Verbeke et al. (2012) y el EMPC Verbraken et al. (2012) van un paso adicional e incorporan los potenciales beneficios de una campaña de retención realizada para evitar una eventual fuga de clientes, lo que genera una medida muy robusta y con una visión de negocios mas recabada para evaluar el desempeño de un modelo de clasificación. La principal diferencia entre MPC y EMPC, es que este último considera la decisión de un cliente candidato a fuga como una variable aleatoria, y luego calcula el valor esperado del beneficio de una campaña de retención realizada orientada a los clientes que fueron señalados por el clasificador.

A diferencia de la literatura disponible en esta área, que se enfoca en seleccionar el mejor modelo entre varios métodos de clasificación, el objetivo es proporcionar un sustento teórico que permita la correcta selección de hiperparámetros y los atributos correctos en la construcción del clasificador en si mismo, en particular enfocándos en un método, *Support Vector Machines*. Este enfoque además presenta las siguientes ventajas:

- El método propuesto permite la incorporación explícita de los costos y beneficios obtenidos al clasificar en priblemas de predicción de fuga de clientes, llevándo a un proceso de selección de atributos especialmente diseñado para este problema en particular.

- El enfoque propuesto alcanza mejores resultados que otras técnicas de selección de atributos en problemas de predicción de fuga, considerando tanto métricas usuales de desempeño (como por ejemplo AUC), e indicadores basados en utilidades.
- La estrategia en si misma es muy flexible y permita elegir diferentes funciones de kernel para selección de atributos en entornos no lineales, y clasificación usando SVM. Incluso el enfoque permite ser extendido a otros métodos de clasificación, no necesariemente SVM.

Dicho esto, y en vista de los resultados vistos en el capítulo anterior, se puede concluir lo siguiente:

- Los métodos propuestos claramente tuvieron mejor rendimiento que las propuestas alternativas usadas en este trabajo, puesto que para todas las bases de datos la estrategia propuesta alcanza el máximo desempeño, para algún subconjunto de atributos, y adicionalmente el mejor desempeño global, entendiéndolo como el desempeño medio sobre todos los conjuntos de atributos, con la única excepción de Cell2Cell, en donde RFE-SVM tiene un mejor desempeño global que el resto de los métodos.
- Estos experimentos prueban que usar selección de atributos es útil en términos de mejorar el poder predictivo de un clasificacor, incluso en aplicaciones de baja dimensionalidad como predicción de fuga de clientes, puesto que la mejor solución fue siempre encontrada con un subconjunto de la totalidad de los atributos.
- Finalmente, las técnicas de resampling probaron su efectividad, puesto que mejoró los resultados en todas las bases de datos, demostrando su efectividad al enfrentar problemas con desbalance de clases, particularmente en Undersampling, que era de esperar dado el gran tamaño de las bases de datos.

5.2. Trabajo Futuro

Existen muchas oportunidades de trabajo futuro, a modo de ejemplo consideramos las siguientes:

- El proceso de selección de atributos puede ser extendido a otras aplicaciones de *Business Analytics*, como por ejemplo *credit scoring* (ver Bravo et al. (2013); Thomas et al. (2002)). El EMPC puede adaptarse para incorporar los costos y los beneficios de aceptar o rechazar a los solicitantes de créditos, la regresión logística se puede establecer como el clasificador de referencia, puesto que es el método de clasificación más común para esta tarea debido a razones regulatorias Thomas et al. (2002).
- También es posible incorporar el costo de la adquisicón de las variables en el modelo, enriqueciendo de esta manera el proceso de selección de atributos. Es posible ver este enfoque en Maldonado et al. (2014b), en donde el costo de los atributos se considera explícitamente en el modelo a través de variables binarias y una restricción presupuestaria explícita.

Bibliografía

Baesens, B. (2014). *Analytics in a Big Data World*. John Wiley and Sons.

Barandela, R., Sánchez, J., García, V., and Rangel, E. (2003). Strategies for learning in class imbalance problems. *Pattern recognition society*.

Batista, G., Prati, R., and Monard, M. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations newsletter*, 6:20–29.

Blattberg, R., Kim, B., and Neslin, S. (2008). *Database Marketing: Analyzing and Managing Customers*. New York: Springer Science+Business Meida, LLC.

Boser, B., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifer. In *ACM Workshop on Computational Learning Theory*, volume 5, pages 144–152.

Bradley, P. and Mangasarian, O. (1998). Feature selection via concave minimization and support vector machines. In *Machine Learning proceedings of the fifteenth International Conference (ICML'98) 82-90, San Francisco, California, Morgan Kaufmann.*

Bravo, C., Maldonado, S., and Weber, R. (2013). Methodologies for granting and managing loans for micro-entrepreneurs: New developments and practical experiences. *European Journal of Operational Research*, 227(2):358–366.

Burez, J. and Van den Poel, D. (2009). Handling class imbalance in customer churn prediction. *Expert Systems with Applications*, 36(3):4626–4636.

Carrizosa, E., Martín-Barragán, B., and D., R.-M. (2011). Detecting relevant variables and interactions in supervised classification. *European Journal of Operational Research*, 213(1):260–269.

Chang, C. and Lin, C. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chawla, N. (2010). *Data mining for imbalanced datasets: An overview*. Springer, Berlin.

Datta, P., Masand, B., Mani, D., and Li, B. (2000). Automated cellular modeling and prediction on a large scale. *Artificial Intelligence Review*, 14:485–502.

Duke University, C. f. C. R. M. (2014).

Ertekin, S., Huang, J., Bottou, L., and Giles, L. (2007). Learning on the border active: Learning in imbalanced data classification.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874.

Fayyad, U. (1996). Data mining and knowledge discovery- making sense out of data. *IEEE Expert-Intelligent Systems and Their Aplications*, 11,:20–25.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54.

Fleming, J. and Asplund, J. (2007). *Human Sigma: Managing The Employee-Customer Encounter*. Gallup Press, New York.

Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. A. (2006). *Feature extraction, foundations and applications*. Springer, Berlin.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422.

Hand, D. (2009). Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning*, 77(1):103–123.

Hidalgo-Munoz, A. R., López, M. M., Santos, I. M., Pereira, A. T., Vázquez-Marrufo, M., Galvao-Carmona, A., and Tomé, A. M. (2013). Application of svm-rfe on eeg signals for detecting the most relevant scalp regions linked to affective valence processing. *Expert Systems with Applications*, 40(6):2102–2108.

Hung, S., Yen, D., and Wang, H. (2006). Applying data mining to telecom churn management. *Expert Systems with Applications*, 31:515–524.

Kennedy, K., Mac Namee, B., and Delany, S. (2009). Low-default portfolio one-class classification: A literature review. *School of Computing Technical Report*, 4.

Kotsiantis, S. B. and et al. (2006). Data preprocessing for supervised learning.

Lemmens, A. and Croux, C. (2006). Bagging and boosting classification trees to predict churn. *Journal of Marketing Research*, 43(2):276–286.

Maldonado, S., Famili, F., and Weber, R. (2014a). Feature selection for high-dimensional class-imbalanced data sets using support vector machines. *Information Sciences*, 286:228–246.

Maldonado, S., Pérez, J., Labbé, M., and Weber, R. (2014b). Feature selection for support vector machines via mixed integer linear programming. *Information Sciences*, 279:163–175.

Maldonado, S. and Weber, R. (2009). A wrapper method for feature selection using support vector machines. *Information Sciences*, 179:2208–2217.

Maldonado, S., Weber, R., and Basak, J. (2011). Kernel-penalized SVM for feature selection. *Information Sciences*, 181(1):115–128.

Maloof, M. (2003). Learning when data sets are imbalanced and when cost are unequal and unknown. Workshop on learning from imbalanced data sets, Washington DC.

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Mozer, M., Wolniewicz, R., Grimes, D., Johnson, E., and Kaushansky, H. (2000). Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on Neural Networks*, 11(3):690–696.

Neslin, S., Gupta, S., Kamakura, W., Lu, J., and Mason, C. (2006). Defection detection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of Marketing Research*, 43(2):204–211.

Ruiz Sánchez, R. (2006). *Heurísticas de selección de atributos para datos de gran dimensionalidad*. PhD thesis, Universidad de Sevilla.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. MIT Press, Cambridge, MA, USA.

Song, L., Smola, A., Gretton, A., Bedo, J., and Borgwardt, K. (2012). Feature selection via dependence maximization. *Journal of Machine Learning research*, 13:1393–1434.

Thomas, L., Crook, J., and Edelman, D. (2002). *Credit Scoring and its Applications*. SIAM.

Van den Poel, D. and Larivière, B. (2004). Customer attrition analysis for financial services using proportional hazard models. *European Journal of Operational Research*, 157(1):196–217.

Van Hulse, J., Khoshgoftaar, T., Napolitano, A., and Wald, R. (2009). Feature selection with high-dimensional imbalanced data. In *Proceedings of the IEEE International Conference on Data Mining Workshops*, pages 507–514.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley and Sons.

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., and Baesens, B. (2011a). New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. *European Journal of Operational Research*, In press.

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., and Baesens, B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218(1):211–229.

Verbeke, W., Martens, D., Mues, C., and Baesens, B. (2011b). Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert Systems with Applications*, 38:2354–2364.

Verbraken, T., Verbeke, W., and Baesens, B. (2012). A novel profit maximizing metric for measuring classification performance of customer churn prediction models. *IEEE Transactions on Knowledge and Data Engineering*, 25(5):961 – 973.

Wei, C. and Chiu, I. (2002). Turning telecommunications call details to churn prediction: a data mining approach. *Expert Systems with Applications*, 23:103–112.

Weston, J., Elisseeff, A., Schölkopf, B., and Tipping, M. (2003). The use of zero-norm with linear models and kernel methods. *Journal of Machine Learning research*, 3:1439–1461.

Wu, X., Kumar, V., Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G., Ng, A., Liu, B., Yu, P., Zhou, Z., Steinbach, M., Hand, D., and Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37.

Yang, J. and Liu, G. (2002). The evaluation of classification models for credit scoring. *T. Arbeitsbericht*, 02-2002. Institut für Wirtschaftsinformatik, Georg-August-Universität Göttingen.