Universidad de las Ciencias Informáticas
Facultad 2

Diseño de un Sistema Basado en Reglas aplicando el algoritmo conceptual RGC del reconocimiento lógico combinatorio de patrones

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DEINGENIERO EN CIENCIAS INFORMÁTICAS

Autor:
Suan López Cepero

Tutores:

Dra. C. Natalia Martínez Sánchez

Ms. C. Maidelis Milanés Luque

Ing. Bienvenido H. Roque Orfe

La Habana, junio 2019

“Año 61 de la Revolución”

"Una computadora puede ser llamada inteligente si logra engañar a una persona haciéndole creer que es un humano".

Alan Turing

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Firma del Autor

Suan López Cepero

_____ _____

Firma del Tutor Firma del Tutor

Dra.C. Natalia Martínez Sánchez MSc. Maidelis Milanés Luque

_____

Firma del Tutor

Ing. Bienvenido Hanley Roque Orfe

Datos de Contacto

Dra.C. Natalia Martínez Sánchez: Graduada de Licenciatura en Cibernética Matemática en la Universidad Central de Las Villas. Máster en Computación Aplicada y Doctora en Ciencias Técnicas. Profesora Titular. Investiga en el área de la Inteligencia Artificial e Informática Educativa. Ha impartido docencia tanto en pregrado como en postgrado en las ramas de la Inteligencia Artificial, las Matemáticas y Programación. Ha impartido conferencias de pregrado y postgrado en Universidades de Colombia, Perú y Mozambique. Ha participado en eventos nacionales e internacionales, publicando trabajos científicos en revistas y bases de datos de prestigio internacional. Vicerrectora de Formación en la Universidad de las Ciencias Informáticas. Correo electrónico: natalia@uci.cu

MSc. Maidelis Milanés Luque: Graduada de Ingeniera en Ciencias Informáticas en el 2007, actualmente realizando investigaciones en el área de la Inteligencia Artificial, específicamente en el enfoque lógico combinatorio del reconocimiento de patrones y los sistemas basado en el conocimiento. Máster en Ciencias en el año 2017. Profesora de la disciplina de Inteligencia Artificial. Jefa del Departamento de Programación de la Facultad 2. Correo electrónico: mmilanes@uci.cu

Ing. Bienvenido Hanley Roque Orfe: Graduado en Ciencias Informáticas en el 2018, actualmente realizando investigaciones en el área de la Inteligencia Artificial, específicamente en el enfoque lógico combinatorio del reconocimiento de patrones y los sistemas basado en el conocimiento. Profesor de la disciplina de Inteligencia Artificial del departamento de Sistemas digitales y técnicas de programación de la Facultad 2. Correo bhroque@uci.cu

Dedicatoria

El presente trabajo está dedicado a mis familiares, amigos y profesores que estuvieron presente a lo largo de la carrera, compartiendo conocimientos y tiempo, para lograr formar al profesional que seré mañana.

Agradecimientos

A mis tutores por servirme de guías y faros como profesionales.

A mi familia por todo el tiempo y apoyo dedicado a lo largo de la carrera.

A mis compañeros de la universidad.

¡Gracias por el recuerdo que me llevo de todos estos años en la universidad!

Resumen

Los Sistemas Basados en Reglas son sistemas expertos que basan su funcionamiento en las reglas de producción y un motor que las infieren para resolver problemas complejos. Dentro de ellos se encuentran los motores de reglas como almacenes que contienen las reglas y el motor de inferencia para una mejor organización del código fuente. En el sistema aplicado se representan dos problemas fundamentales, uno relacionado con el cálculo de los conceptos obtenidos a partir de la aplicación del algoritmo conceptual RGC y otro referente a la obtención de las reglas que serán usadas en el motor de reglas. El presente trabajo se propone como objetivo general generar reglas de forma automática para la clasificación en un Sistema Basado en Reglas aplicando el algoritmo RGC del reconocimiento lógico combinatorio de patrones. Se utilizaron como métodos científicos hipotético-deductivo, analítico-sintético, método histórico lógico y el dialéctico, inducción-deducción y preexperimento, los que permitieron dirigir la investigación. Se describen las dos etapas del algoritmo RGC: determinación extensional y determinación intencional. Para validar los resultados se utiliza el método de validación cruzada y la prueba no paramétrica de Friedman Se establecen comparaciones en cuanto a la eficacia entre un sistema aplicando el algoritmo conceptual RGC y otro sin aplicarlo, así como una comparación entre otros algoritmos como el LC-Conceptual, ID3, C4.5 y PART. Los resultados demuestran que la solución propuesta garantiza una mayor eficacia en la solución de problemas.

Palabras clave: algoritmos conceptuales, inteligencia artificial, Sistemas Basados en Reglas.

Abstract

Rule-Based Systems are expert systems that base their operation on production rules and an engine that infers them to solve complex problems. Within them are rule engines such as warehouses containing rules and the inference engine for better organization of the source code. In the applied system two fundamental problems are represented, one related to the calculation of the concepts obtained from the application of the conceptual algorithm RGC and another referring to the obtaining of the rules that will be used in the rules engine. The present work is proposed as a general objective to generate rules automatically for classification in a System Based on Rules applying the RGC algorithm of the combinatorial logical recognition of patterns. They were used as scientific methods hypothetical-deductive, analytic-synthetic, logical historical method and dialectic, induction-deduction and pre-experiment, which allowed directing the research. The two stages of the RGC algorithm are described: extensional determination and intentional determination. To validate the results, the cross-validation method and Friedman's nonparametric test are used. Comparisons are made as to the effectiveness between a system applying the RGC conceptual algorithm and another without applying it, as well as a comparison between other algorithms such as LC-Conceptual, ID3, C4.5 and PART. The results show that the proposed solution guarantees greater effectiveness in problem solving.

Keywords: conceptual algorithms, artificial intelligence, rule-based systems.

Índice

Introducción

La Inteligencia Artificial (IA) es una rama de la ciencia de la computación dedicada a la creación de hardware y software que intenta producir resultados similares a los expresados por los humanos. En (Nilsson 2014) se dice que la Inteligencia Artificial es una especialidad de la informática que se ocupa de cómo dar a las computadoras la sofisticación para actuar de forma inteligente, y hacerlo en ámbitos cada vez más amplios. Participa plenamente en la pasión de la informática por la abstracción, la programación y los formalismos lógicos, y el detalle de los algoritmos sobre los datos de comportamiento.

Le conciernen dos ideas básicas: la primera es que ésta involucra el estudio de los procesos del pensamiento de los humanos y la segunda que trata de representar estos procesos en una computadora. Conceptualizar estas ideas básicas condujo al desarrollo de los llamados Sistemas Basados en el Conocimiento o Sistemas Informáticos Inteligentes (Reyes González 2017).

Una característica distintiva de los sistemas basados en el conocimiento es la separación del conocimiento (base de conocimiento) del método de solución del problema (máquina de inferencia). La construcción de la base del conocimiento lleva implícito un arduo proceso de adquisición del conocimiento y es particular para cada sistema, por lo que es necesario construirla para cada aplicación. Sin embargo, la máquina de inferencia puede reusarse en la construcción de varios sistemas basados en el conocimiento siempre que el tipo de conocimiento y el tipo del razonamiento sea similar.

Diferentes tipos de conocimiento dan lugar a diferentes tipos de sistemas basados en el conocimiento, entre ellos Redes Neuronales Artificiales (Graupe 2013); los Sistemas Basados en Casos (Riesbeck, Schank 2013); (Kolodner 1992) y los Sistemas Basados en Reglas (Ignizio 1991), (Gálvez-Lio 1998); (Gutiérrez 2008).

La Ingeniería del Conocimiento surge como consecuencia de la necesidad de establecer principios metodológicos y científicos que permitan desarrollar sistemas basados en el conocimiento a partir de los fundamentos de la informática en general y de la inteligencia computacional en particular. En este aspecto puede verse como la especialización de Ingeniería de Software en su aplicación al desarrollo de Sistemas Inteligentes.

La Ingeniería del Conocimiento se enfoca al desarrollo de sistemas basados en el conocimiento, destacándose la necesidad de la adquisición del conocimiento, así como su especificación, verificación, validación, diseño e implementación en sistemas informáticos o lenguajes apropiados para la construcción de bases de conocimiento para la toma de decisiones.

Para la creación de la base de conocimiento es necesario realizar un arduo proceso de revisión del conocimiento público existente, así como el conocimiento que poseen los expertos en el dominio, conocimiento privado. El conocimiento público incluye las definiciones, hechos y teorías publicadas. Pero la experticidad usualmente incluye más que esta clase de conocimiento. Los expertos humanos generalmente poseen conocimiento privado. Sobre el conocimiento público hay consenso, el privado puede llevar a polémicas entre los expertos.

La adquisición del conocimiento a partir de expertos humanos, si bien necesaria e insustituible en muchas aplicaciones, ha presentado diversas dificultades que van desde la representación del sentido común hasta las excesivas demoras en la implementación y el mantenimiento de los sistemas.

El proceso de adquisición del conocimiento requerido en un sistema basado en el conocimiento puede ser automatizado o parcialmente automatizado (Jansen 1988). La idea radica en lograr eliminar el intermediario (ingeniero del conocimiento) entre el experto y el sistema que se desea construir.

La envergadura del proceso de adquisición del conocimiento depende del tipo de conocimiento. En los sistemas basados en reglas se desarrolla un proceso complejo y prolongado pues la extracción se refiere a la formalización de reglas y el pensamiento humano no siempre está regido conscientemente por las reglas de la lógica; en ocasiones es básicamente un procesamiento de información recuperada con el tiempo.

Por otro lado, el reconocimiento lógico combinatorio de patrones se ocupa del desarrollo de teorías, métodos, técnicas y dispositivos computacionales para la realización de procesos ingenieriles, computacionales y/o matemáticos, relacionados con objetos físicos o abstractos, que tienen el propósito de extraer información que permita establecer propiedades y/o vínculos de o entre conjuntos de dichos objetos sobre la base de los cuales se realiza una tarea de identificación o clasificación.

Por problemas de reconocimiento de patrones en este trabajo, siguiendo la definición descrita en (Ruiz-Shulcloper 2009) se entienden todos aquellos relacionados con la clasificación de objetos y fenómenos y con la determinación de los factores que inciden en los mismos. Así, se consideran cuatro familias de problemas que se denominan respectivamente: selección de rasgos; clasificación supervisada; clasificación no supervisada; clasificación parcialmente supervisada.

La selección de rasgos es uno de los pasos fundamentales en cualquier problema de clasificación debido a que la mayoría de los problemas de reconocimiento de patrones están basados en la descripción de los objetos en términos de un conjunto de rasgos. Relacionado con este problema en la literatura se identifican dos problemas diferentes pero muy vinculados: la selección de rasgos para la clasificación y la selección de rasgos para la descripción.

En el primer grupo de problemas, lo que se enfrenta es la determinación del mejor subconjunto de rasgos para la clasificación de nuevos objetos (no clasificados). Esto conlleva la reducción del conjunto de todos los posibles rasgos (reducción de la dimensionalidad) sobre la base de las diferencias que estos rasgos presentan en cuanto a mejor reconocer, clasificar a los nuevos objetos y otros problemas de optimización adicionales del subconjunto de rasgos a emplear, que también se deben tener en cuenta en muchos problemas de la realidad. Los problemas de selección de rasgos para la descripción son muy frecuentes en las ciencias poco formalizadas. Este problema conlleva la determinación de un subconjunto de rasgos que de una mejor manera caracteriza a los objetos de cada una de las clases.

Dentro del reconocimiento lógico combinatorio están los algoritmos de agrupamiento conceptual que tienen como objetivo, además de hacer una estructuración del universo de estudio en cuestión, decir qué significan cada una de las estructuras determinadas en términos de los rasgos que describen el problema. Proporcionan el concepto que está implícito en torno a las propiedades que cumplen los objetos contenidos en cada una de las estructuras. El concepto fundamental en el que se apoyan estos algoritmos es el principio de agrupamiento para los objetos, pues la pertenencia de un objeto a una estructura determinada de tal modo que refleje el concepto cualitativo que el especialista maneja dentro del área y contexto del problema que se pretende resolver y que será alrededor del cual se agrupen los objetos de estudio.

En particular resulta claro que la utilización de los algoritmos conceptuales en la etapa de ingeniería del conocimiento de un Sistema Basado en Reglas, apoya a la selección de las variables, sus valores, relevancia y conformación de las reglas utilizando los conceptos generados.

Por lo antes expuesto se plantea el siguiente **problema a resolver:** ¿Cómo generar las reglas para la clasificación en los Sistemas Basados en Reglas mediante los algoritmos conceptuales del reconocimiento lógico combinatorio de patrones?

Por tanto, el presente trabajo tiene como **objeto de estudio** los Sistemas Basados en Reglas.

El **objetivo general** del presente trabajo puede plantearse entonces de la manera siguiente: generar reglas de forma automática para la clasificación en un Sistema Basado en Reglas aplicando el algoritmo RGC del reconocimiento lógico combinatorio de patrones.

Como **campo de acción** la generación de reglas aplicando el algoritmo conceptual RGC del reconocimiento lógico combinatorio de patrones.

Para lograr este objetivo general se plantean los siguientes **objetivos específicos:**

1. Realizar un estudio del arte de la utilización de algoritmos del campo del reconocimiento lógico combinatorio de patrones en la elaboración de un Sistema Basado en Reglas.

2. Diseñar un Sistema Basado en Reglas que genere las reglas de forma automáticas aplicando el algoritmo conceptual RGC del reconocimiento lógico combinatorio de patrones.

3. Valorar el resultado en la práctica del diseño del Sistema Basado en Reglas aplicando el algoritmo conceptual RGC del reconocimiento lógico combinatorio de patrones.

Para el desarrollo de las tareas científicas se han combinado diferentes métodos y procedimientos teóricos y empíricos de la investigación científica en la búsqueda y procesamiento de la información. Los fundamentales son:

Métodos generales. el método hipotético-deductivo para elaborar la hipótesis de investigación y proponer líneas de trabajo a partir de resultados parciales; el método

sistémico para el desarrollo del sistema computacional y lograr que los elementos que formen parte del diseño sean un todo que funcione de manera armónica; el método histórico lógico y el dialéctico para el estudio crítico de los trabajos anteriores.

Métodos lógicos. el método analítico-sintético al descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución de la propuesta; el método inducción-deducción como vía de la constatación teórica durante el desarrollo de la tesis; el método de modelación para el desarrollo de los algoritmos.

Métodos empíricos. el método coloquial para la presentación y discusión de los resultados en sesiones científicas; el método preexperimental para comprobar la utilidad de los resultados obtenidos.

Después de la revisión de la literatura y el desarrollo consecuente del marco teórico, se formuló la siguiente **hipótesis de investigación**: la utilización de los algoritmos conceptuales del reconocimiento lógico combinatorio de patrones genera reglas, para mejorar la eficacia en la construcción de la base de conocimiento en un motor de reglas de los Sistema Basado en Reglas en problemas de clasificación.

Variable independiente: la utilización de los algoritmos conceptuales del reconocimiento lógico combinatorio de patrones genera reglas.

Variable dependiente: la eficacia en la construcción de la base de conocimiento en un motor de reglas de los Sistema Basado en Reglas en problemas de clasificación.

La tesis está conformada por tres capítulos. En el primero se realiza una caracterización de los Sistemas Basados en Reglas. Se analizan los algoritmos conceptuales del reconocimiento lógico combinatorio de patrones.

El segundo capítulo aborda concretamente la definición de la propuesta de solución para elaborar un Sistema Basado en Reglas aplicando los algoritmos conceptuales del reconocimiento lógico combinatorio de patrones en la base de conocimiento.

En el tercer capítulo se exponen los preexperimentos realizados para la validación de la de la propuesta de solución utilizando un caso de estudio.

Por último, se establecen las Conclusiones, se emiten Recomendaciones, se relacionan las Referencias Bibliográficas y se incluye un conjunto de Anexos que facilitan la comprensión de la memoria gráfica de la Tesis.

Capítulo 1. Marco teórico referencial sobre los motores de reglas y los algoritmos conceptuales

En el presente capítulo se abordan los principales referentes teóricos de la investigación relacionados con los Sistemas Basados en Reglas. Se realiza una caracterización de los requisitos necesarios para la implementación de la solución mediante las reglas de producción (o reglas de negocio). Se explican los fundamentos del Reconocimiento Lógico Combinatorio de Patrones que sirven de sustento teórico de la investigación. Se caracterizan los algoritmos de agrupamiento conceptual LC-Conceptual y RGC. Finalmente se describen las características fundamentales de las herramientas y tecnologías de software utilizadas para el desarrollo del sistema.

1.1 Sistemas Basados en Reglas

Los Sistemas Basado en Reglas son el producto del constante cambio y evolución de los Sistemas Basado en el Conocimiento, son modelos compuestos por arquitecturas, algoritmos y otros modelos, utilizados para resolver problemas como los son un sistema de control de tránsito, sistemas de seguridad, transacciones bancarias, problemas de clasificación, entre otros (Jiménez Builes 2006).

Como su nombre indica, estos sistemas están compuestos de reglas, las mismas que deben llegar una estructura dada por un antecedente y un consecuente. Mientras más reglas se tengan en la base de conocimiento, más precisa es la respuesta del sistema hacia un problema. Su implementación es mucha más sencilla ya que únicamente se debe tener claro todas las reglas que se va a implementar para su análisis (Almeida y Stalin 2018).

Estos sistemas trabajan mediante la aplicación de reglas, comparación de resultados y aplicación de las nuevas reglas basadas en situación modificada. También pueden trabajar por inferencia lógica dirigida, bien empezando con una evidencia inicial en una determinada situación y dirigiéndose hacia la obtención de una solución, o bien con hipótesis sobre las posibles soluciones y volviendo hacia atrás para encontrar una evidencia existente (o una deducción de una evidencia existente) que apoye una hipótesis en particular (Badaró, Ibañez y Agüero 2013).

Ellas han sido aplicadas con éxito en múltiples campos, tales como clasificación, regresión, control, debido a su capacidad para incluir conocimiento experto a priori, manejar la

imprecisión existente en los datos, y representar sistemas para los que no es posible obtener un modelo matemático por la tipología del problema y de los datos (Salazar Acosta 2012).

Teniendo en cuenta (Jiménez Builes 2006) se entiende en la presente investigación Sistemas Basados en Reglas como: modelos de representación del conocimiento de los sistemas de base de conocimiento. Que permite resolver problemas complejos mediante las reglas (estructuradas con un antecedente y una consecuente), la base de conocimiento y un motor de inferencia (encargado de activar las reglas).

A los sistemas basados en reglas se le denomina reglas de producción en la forma de representación del conocimiento, como método de inferencia utiliza la regla de Modus Ponens y Modus Tollens y como estrategias de inferencia el Encadenamiento de Reglas y el Encadenamiento de Reglas Orientado a un Objetivo. Las reglas utilizan un formato IF - THEN para representar el conocimiento, la parte IF de una regla es una condición (también llamada premisa o antecedente), y la parte THEN es la acción (conclusión o consecuente) permite inferir un conjunto de hechos nuevos si se verifican las condiciones establecidas en la parte IF (Galvez Lio 2006). Cada una de estas partes consiste en una expresión lógica con una o más afirmaciones objeto-valor conectadas mediante los operadores lógicos y, o, o no. Una regla se escribe normalmente como “Si premisa, entonces conclusión” (Salazar Acosta 2012).

Presenta ciertas limitaciones tales como:

- Al agregar nuevos objetos a la base de conocimiento, se necesitan de nuevas reglas para que abarque todo el dominio y del experto para que las gestione.
- La cantidad de reglas está en correspondencia con las variables que definen el problema, o sea, a problemas con más datos y mayor dificultad, aumenta el número de reglas y su complejidad.
- Alta dependencia del experto para gestionar las reglas.
- Encadenamiento infinito por falta de experticia del problema.
- Contradicción entre las reglas que ya existen.
- Modificación de las que ya existen.

1.1.1 Reglas de producción

Las reglas de producción son una de las Forma de Representar el Conocimiento que existen, consta de un par ordenado (A, B), representado en el cálculo proposicional como A → B, donde A es el antecedente y B el consecuente de la regla (Salazar Acosta 2012).

Una regla de producción se interpreta de la siguiente manera: si se satisface el antecedente, entonces se cumple el consecuente. Esta manera de interpretar una regla permite considerarla como una unidad relativamente independiente de conocimiento. Estas pueden adoptar varias formas según (Salazar Acosta 2012):

- Si condición P entonces conclusión C.
- Si situación S entonces acción A.
- Si condición C1 entonces no condición C2.

Los antecedentes de las reglas, independientemente de la forma que éstas adopten, pueden ser simples o compuestos. Los compuestos se forman uniendo varias condiciones simples por medio de las conectivas lógicas.

Las reglas de producción pueden ser comprendidas fácilmente y tienen fuerza expresiva para:

- Representar reglas de inferencia dependientes del dominio.
- Representar especificaciones del comportamiento.
- Almacenar el conocimiento que pueda ser expresado como heurística experimental.
- Expresa conocimiento orientado a un objeto.
- Expresa relaciones casuales.

Las reglas de producción han sido utilizadas con éxito como FRC para tareas de diagnóstico, diseño (configuración de computadoras), planificación, problemas deductivos, etc. Pero son inadecuadas para: definir términos, describir objetos y relaciones estadísticas entre ellos (Navarro Sánchez 2011).

1.1.2 Motor de inferencia

Los motores de inferencia o método de inferencia que utiliza las reglas de producción como se plantea al inicio del presente epígrafe son de Modus Ponens (*forward chaining*) y Modus Tollens (*backwards chaining*).

El método Modus Ponens es quizás para muchos la regla de inferencia más comúnmente aplicada, esta se utiliza para obtener soluciones sencillas. En ella, se examina la premisa de la regla, y si es cierta, la conclusión pasa a formar parte del conocimiento. Como ilustración supóngase que se tiene la regla. “Si A es cierto, entonces B es cierto”. Esta regla de inferencia, que parece trivial, debido a la familiaridad, es la base de un gran número de sistemas expertos (Gutiérrez 2008).

El funcionamiento de este método es *forward chaining* que consiste en el modo de razonamiento que parte de los hechos iniciales que describen un problema para llegar a una conclusión (Coles et al. 2010).

La regla de inferencia del Modus Tollens se utiliza también para obtener conclusiones simples, pero en este caso se examina primero la conclusión y si es falsa, se concluye que la premisa es falsa, supóngase de nuevo que se tiene la regla “Si A es cierto, entonces B es cierto”, pero se sabe que “B es falso”. Por tanto se obtiene aplicando la regla de inferencia Modus Tollens que “A es falso” (Gutiérrez 2008).

En este caso el método funciona con *backwards chaining* que es el modo de razonamiento que parte de una conclusión para volver a los hechos. Establecida una hipótesis se razona de manera retrospectiva para su comprobación (Coles et al. 2010).

Estos métodos son aplicados en los motores de reglas que son los que se encargan de general las reglas de producción para después tomar decisiones en las soluciones de problemas reales. En la presente investigación se propone utilizar los motores de reglas para integrarlos con el reconocimiento lógico combinatorio de patrones en la aplicación de algoritmos conceptuales.

1.1.3 Estrategias de inferencia y control

Las estrategias de inferencia son utilizadas por el motor de inferencia para obtener soluciones simples y compuestas según (Gutiérrez 2008):

El encadenamiento de Reglas: es aplicado cuando las premisas de ciertas reglas coinciden con las conclusiones de otras. Cuando se encadenan las reglas, los hechos pueden utilizarse para dar lugar a nuevos hechos. Esto se repite sucesivamente hasta que no se puedan obtener más conclusiones. El tiempo que consume este proceso hasta su terminación depende, por una parte, de los hechos conocidos, y, por otra, de las reglas que se activan.

Este algoritmo puede ser utilizado de muchas formas. Una de ellas comienza con las reglas cuyas premisas son conocidas. Estas reglas deben concluir y sus conclusiones dan paso a nuevos hechos. Estos nuevos hechos a su vez se añaden al conjunto de hechos conocidos, y el proceso se repite hasta no poder obtener nuevos hechos.

El Encadenamiento de Reglas Orientado a un Objeto: por otra parte, requiere del usuario para seleccionar una variable o nodo objetivo; para que el algoritmo navegue a través de las reglas en búsqueda de una conclusión para el nodo antes señalado. Si no se obtiene ninguna conclusión con la información existente, entonces el algoritmo se ve forzado a preguntar al usuario, en busca de nueva evidencia sobre los elementos que son relevantes para obtener información sobre el objetivo.

1.2 Motor de reglas

Un motor de reglas: es un sistema de información, que ejecuta reglas de producción o monitorea actividades de negocio en términos del cumplimiento de las reglas en tiempo de ejecución. Los motores de reglas de producción generalmente incluyen características para el desarrollo de reglas de producción que permiten una definición conveniente y la verificación de las reglas (Hitpass 2017).

Los motores de reglas son un componente que, partiendo de una información dada y un conjunto de reglas, detecta que reglas debe aplicarse en un instante determinado y cuáles son los resultados de esas reglas. Por lo tanto, un motor de reglas brinda una infraestructura desacoplada del código fuente de la aplicación para la definición, administración y ejecución de reglas de producción (DEL SUR 2008).

Un motor de reglas consiste en proporcionar un modelo computacional alternativo. En lugar del modelo imperativo habitual, que consiste en comandos en secuencia con condicionales y bucles, un motor de reglas se basa en un sistema de reglas de producción. Este es un conjunto de reglas de producción, cada una de las cuales tiene una condición y una acción;

de manera simplista, puede pensar que se trata de un grupo de declaraciones if-then, de tal manera que si se cumplen las condiciones del IF se ejecutan las acciones del THEN. El espacio de trabajo es donde se almacena el conocimiento (los hechos) que el motor utiliza para decidir qué reglas deben activarse. Por último, el procesador de reglas está basado en un mecanismo de inferencia, el cual permite a través de reglas definidas, determinar que reglas deben ejecutarse en base a los hechos que se van insertando en el espacio de trabajo (González y Ampuero 2015).

El uso de los motores de reglas como todo presenta ventajas y desventajas, dependiendo de la situación en la que se utilice, ya que no todos los problemas tienen una solución adecuada con el uso de esta tecnología. Algunas de ellas son:

Ventajas:

- Usan lenguajes declarativos, lo que permite modelar situaciones complejas con un código menos extenso y más entendible que su contraparte procedural.
- El conocimiento se concentra en un solo lugar y ya no es solo propiedad de la persona encargada, sino que está a disposición de cualquier persona.
- Tiene una buena escalabilidad ya que la curva de tiempo de respuesta tiene una tendencia logarítmica a medida que el número de reglas crece.
- Gran parte de los productos disponibles en el mercado poseen herramientas como lenguajes específicos de dominio, especificación grafica de flujos de reglas y otras que permiten lograr el entendimiento del código por gente que no necesariamente posee los conocimientos técnicos para hacerlo.

Desventajas:

- En aplicaciones donde las reglas son de carácter técnico y no tienen mayor variación en el tiempo, no es justificable el uso de un motor de reglas.
- Cuando la cantidad de reglas es baja, el uso de un motor de reglas es menos eficiente que el uso de un lenguaje procedural.
- Existe poco conocimiento de la tecnología, por lo que la mano de obra se torna escasa y por consistente de un costo mayor.
- Se evidencia gran dependencia de un experto para declarar las reglas.

1.2.1 Arquitecturas

Los autores (Ibarra y Bazán 2013), (DEL SUR 2008), (González y Ampuero 2015) y (Bañez, Prinse y Torres Utrilla 2016) plantearon diferentes arquitecturas para los motores de reglas, en las cuales pueden variar el modelo según el problema a resolver. La arquitectura básica está compuesta por una base de conocimiento, las reglas y el motor de inferencia, en otros casos a la arquitectura se le agrega una interfaz de administración de reglas, donde el experto gestiona las reglas del sistema. Por la fuerte dependencia que existe entre la base de conocimiento y las reglas estas pueden estar juntas en un proceso y el motor de inferencia por separado de ellas como se muestra en las figuras 1 y 2.

En la figura 1 se muestra una arquitectura desarrollada de forma sencilla compuesta por tres elementos principales, el motor de inferencia, la base de conocimiento y reglas y por último la interfaz de intercambio con el usuario.

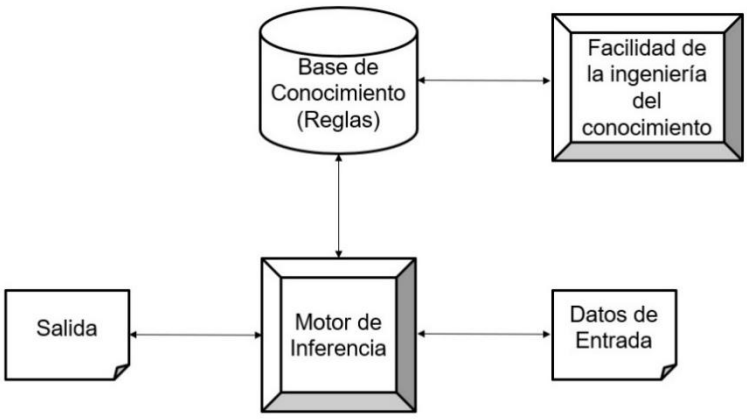
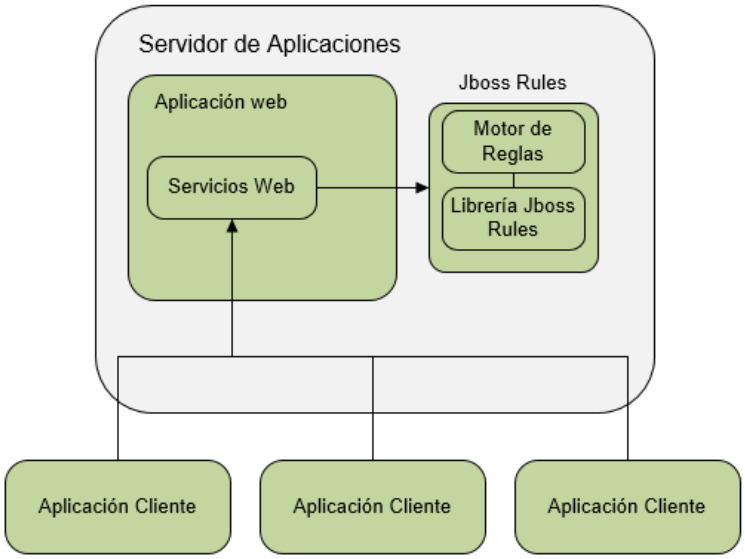


Figura 1 *Arquitectura de un motor de reglas (Ibarra y Bazán 2013)*

Los elementos que distinguen arquitectura de la figura 2 el servidor de aplicaciones a través del cual despliegan sus componentes, una interfaz para los sitios web, y las aplicaciones clientes desde donde se invocan los servicios web del motor JBoss Rules y (Bañez, Prinse y Torres Utrilla 2016)



.

Figura 2 *Arquitectura del motor de reglas JBoss Rules (Bañez, Prinse y Torres Utrilla 2016)*

En (González y Ampuero 2015) se dice que un motor de reglas está conformado por la base de conocimiento, las reglas de producción y el motor de inferencia, como se describe en la figura 3.

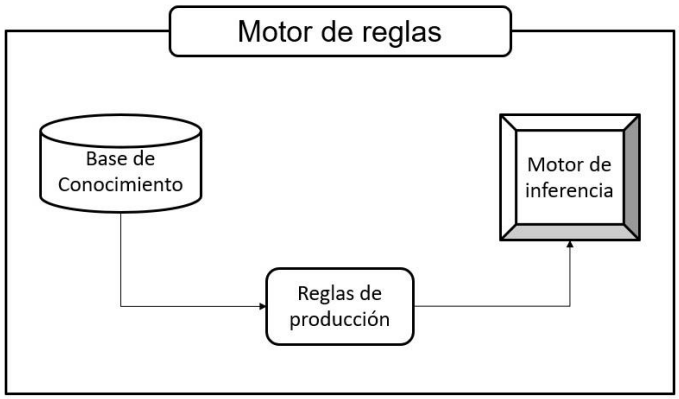


Figura 3 *Fuente (Elaboración Propia)*

Actualmente en la industria de software, existe una amplia variedad de productos que se constituyen como motores de reglas de producción. Los cuales pueden clasificarse de acuerdo al nivel de sofisticación de sus prestaciones y capacidades. Estos motores van desde sistemas que contemplan un conjunto integrado de herramientas para el modelado y ejecución de las reglas de producción, hasta sistemas gestores que cubren todo del ciclo

de vida de las reglas de producción (Bañez, Prinse y Torres Utrilla 2016). A continuación, se mencionan problemas que han encontrado solución con el uso de los mismo:

1. Tecnologías para implementar un marco integrador de SOA (arquitectura orientada a servicios) y BPM (Business Process Management o gestión de procesos de negocio) usando el motor de reglas JBoss Rules. (Bazán, Giandini y Díaz 2010).
2. Diseño y desarrollo de mecanismos de razonamiento multi-agente para la negociación energía eléctrica utilizando JESS (motor de reglas) y JADE (Arias et al. 2006).
3. Diseño y desarrollo de un juego de estrategia de cartas coleccionables (TCG) con inteligencia artificial utilizando el motor de reglas NRules (Herrerías Santos 2018).

Por lo antes planteado, en la presente investigación se entiende motores de reglas como: un componente que puede ser usado en la arquitectura de los Sistemas Basados en Reglas, estos suelen ser parcialmente o totalmente independizados (Bañez, Prinse y Torres Utrilla 2016). Por lo generar su uso es para almacenar el código referente a los Sistema Basados en Reglas (DEL SUR 2008), y su arquitectura varía en dependencia de los dos tipos mencionados anteriormente.

En el primero de los casos (independizado parcialmente) está compuesto por la base de conocimiento, las reglas y el motor de inferencia. (González y Ampuero 2015) Para el segundo caso (independizado totalmente) la arquitectura se mantiene igual que la antes mencionada, pero se le agrega una interfaz de usuario para su empleo por el experto (Ibarra y Bazán 2013).

1.3 Reconocimiento Lógico Combinatorio de Patrones

La selección de rasgos es uno de los pasos fundamentales en cualquier problema de clasificación debido a que la mayoría de los problemas de reconocimiento de patrones están basados en la descripción de los objetos en términos de un conjunto de rasgos. En la literatura se identifican dos problemas diferentes pero muy vinculados: la selección de rasgos para la clasificación y la selección de rasgos para la descripción (Ruiz-Shulcloper 2009).

El Reconocimiento Lógico Combinatorio de Patrones encuentra su basamento teórico matemático en la Lógica Matemática, la Teoría de Testores, la Teoría Clásica de Conjuntos, la Teoría de los Subconjuntos Difusos, la Teoría Combinatoria, y la Matemática Discreta en

general. Las ideas centrales de este enfoque consisten en suponer que los objetos se describen por medio de una combinación de rasgos numéricos y no numéricos, y los distintos valores pueden ser procesados por funciones numéricas (Ruiz, Guzman y Martínez 1999).

Este enfoque presupone que el espacio en que se modelan los objetos son en general productos cartesianos de los conjuntos de valores admisibles de las variables en términos de las cuales se describen todos los objetos, no espacios métricos. Además, tiene en cuenta el concepto de analogía o similitud que se refiere al parecido que poseen entre sí dos objetos en dependencia de los rasgos que los describen. En este sentido se considera también la semejanza o parecido que guardan dos valores de un mismo rasgo, lo que se conoce como criterios de comparación de valores de las variables o función de comparación por rasgos (Ruiz, Guzman y Martínez 1999).

En los problemas de selección de rasgos con datos mezclados e incompletos, la principal herramienta empleada es la Teoría de Testores. Se denomina testor de una matriz de entrenamiento (ME), al subconjunto de rasgos T⊆ R, tales que al eliminar todas las columnas de ME excepto las de T, no aparecen nuevas sub-descripciones semejantes en clases diferentes (Alba-Cabrera, Ibarra-Fiallo y Godoy-Calderon 2013); (Sanchez-Diaz et al. 2014).

Un testor se denomina irreducible (típico) si al eliminar alguna de sus columnas deja de ser testor (Alba-Cabrera, Ibarra-Fiallo y Godoy-Calderon 2013); (Sanchez-Diaz et al. 2014). En el cálculo de los testores típicos se utilizan algoritmos tales como el LEX, CT_Ext, BR, fast-CT_Ext y Fast-BR (Lias-Rodriguez y Sanchez-Diaz 2013).

El marcado carácter diferenciante e irreducible de los testores típicos, y por ende de los rasgos que lo conforman condujo a formular la definición de peso informacional de un rasgo (Ruiz-Shulcloper 2009), en función de la frecuencia relativa de aparición de ese rasgo en la familia de los testores típicos y la longitud de los testores típicos en los que aparece el rasgo, según la ecuación 1:

$$\varepsilon(x_i) = \alpha F(x_i) + \gamma L(x_i) \qquad (1)$$

Los parámetros α>0, >O y α+γ=1, α y γ ponderan la influencia de Fi (frecuencia de aparición) y Li (longitud de los testores) respectivamente y se determinan por el experto del área de aplicación.

Un problema de clasificación supervisada dentro del Reconocimiento Lógico Combinatorio de Patrones consiste en, construir un algoritmo que permita, a partir de una muestra no vacía de objetos estructurados en clases, decidir a cuál de las clases pertenece un objeto nuevo que se quiera clasificar. Los algoritmos basados en el ideal de la clase (AIC), en umbrales de exactitud (AUE), tipo votación (ATV), basados en la tipicidad y el contraste (ATV), en conjuntos de representantes (CR) y el modelo de algoritmos Kora-Ω son ejemplos de métodos de clasificación supervisada (Ruiz-Shulcloper 2009).

Resolver un problema de clasificación no supervisada consiste en determinar la estructura interna de un conjunto de descripciones de objetos en el espacio de representación. Esta estructura interna depende en una primera instancia, de la selección del propio espacio de representación y de la forma en que los objetos se comparen, es decir, del concepto de similitud que se emplee y del criterio de agrupamiento que se utilice. En este sentido se pueden encontrar dos situaciones diferentes: una en la que por determinadas razones se conoce que los objetos se agrupan en un número dado de clases, pero no se tiene muestra alguna de este (agrupamiento o estructuración restringida) y otra en la que no se cuenta con esa información (agrupamiento o estructuración libre).

Los algoritmos de agrupamiento, convencionales o tradicionales como también son conocidos en la literatura, presentan importantes limitaciones identificadas en Michalski y Stepp (1981): dejan el problema de la interpretación de los grupos al analista de datos. No tienen en cuenta los métodos que las personas emplean para agrupar objetos. La lógica humana tiende a agrupar objetos en categorías caracterizadas por conceptos, en grupos similares teniendo en cuenta algún atributo relevante o más importante que el resto. Los métodos tradicionales de agrupamiento no toman en consideración ningún concepto o construcciones lingüísticas que las personas usan para describir colecciones de objetos. Para enfrentar estas limitaciones a finales de los años 70 e inicios de los 80, Rysazard S. Michalski introdujo un conjunto de ideas que dieron origen al agrupamiento conceptual (Michalski y Stepp 1981).

1.3.1 Selección de rasgos

La selección de rasgos es uno de los pasos fundamentales en cualquier problema de clasificación debido a que la mayoría de los problemas de reconocimiento de patrones están basados en la descripción de los objetos en términos de un conjunto de rasgos. En la literatura se identifican dos problemas diferentes pero muy vinculados: la selección de

rasgos para la clasificación y la selección de rasgos para la descripción (Ruiz-Shulcloper 2009).

La selección de rasgos para la clasificación, es la determinación del mejor subconjunto de rasgos para la clasificación de nuevos objetos (no clasificados). Esto conlleva la reducción del conjunto de todos los posibles rasgos (reducción de la dimensionalidad) sobre la base de las diferencias que estos rasgos presentan para clasificar a los nuevos objetos y otros problemas de optimización adicionales del subconjunto de rasgos a emplear (Ruiz-Shulcloper 2009).

1.3.2 Clasificación supervisada

Un problema de clasificación supervisada consiste en, dado un universo de objetos estructurados en clases, de cada una de las cuales se tiene una muestra no vacía, que permite construir un algoritmo a partir de esta muestra, decidirá a cuál de las clases pertenece un objeto nuevo que se quiera clasificar.

Un problema de Reconocimiento de Patrones en la clasificación supervisada es un procedimiento efectivo donde la clasificación de un objeto $O \in U$ por un clasificador supervisado A, se entiende la acción de asignar un r-uplo de pertenencias a las clases de U, tomando dicho objeto como entrada de A. En algunos textos también se entiende por clasificación el resultado de esta acción. La clase que A, con matriz de entrenamiento M, le asigna a un objeto O se denotará por $\alpha A\ (M, O)$. A esto último se le llamará r-uplo de pertenencia del objeto asignado por A. En la descripción de los algoritmos se utiliza la notación funcional de un clasificador supervisado (Ruiz-Shulcloper 2009).

1.3.3 Clasificación no supervisada

En la clasificación no supervisada o agrupamiento, el propósito es juntar (agrupar) los objetos según su analogía (parecido, semejanza, cercanía si se está hablando de un espacio de representación con distancia definida). Los tres elementos esenciales que lo constituyen son: el espacio de representación de los objetos, la medida de similitud (β, función de semejanza) y el criterio de agrupamiento Π, es decir, la manera en que es utilizada la similitud para la solución del problema planteado (Reyes González 2014).

El objetivo de los algoritmos de agrupamiento es dado un conjunto de objetos definidos en términos de un conjunto de rasgos, intentar construir particiones o cubrimientos de este

conjunto, donde la semejanza intragrupo sea máxima y la semejanza inter-grupos sea mínima (Reyes González 2017).

Según (Reyes González 2014) los algoritmos de agrupamiento convencionales tienen las siguientes limitaciones:

1. Dejan el problema de la interpretación de los grupos al analista de datos.
2. No tienen en cuenta los métodos que los humanos emplean para agrupar objetos. Las personas tienden a agrupar objetos en categorías caracterizadas por conceptos, en grupos similares teniendo en cuenta algún atributo relevante o más importante que el resto.
3. No toman en consideración ningún concepto o construcciones lingüísticas que las personas usan para describir colecciones de objetos (Michalski y Stepp 1981).

1.3.4 Clasificación semisupervisada

Los problemas de clasificación semisupervisada combinan tanto la clasificación supervisada como la no supervisada, su definición formal plantea: la clasificación semisupervisada se considera una extensión de la clasificación supervisada donde el conjunto de entrenamiento está formado por un conjunto L de objetos clasificados y un conjunto U de objetos sin clasificar, donde se asume que el número de objetos no clasificados es mayor que los clasificados.

El objetivo de la clasificación semisupervisada es entrenar un clasificador f a partir de los conjuntos L y U, de manera que se podrá obtener una clasificación más exacta que la cantidad de objetos ya clasificados por los problemas de clasificación supervisados.

1.4 Algoritmos Conceptuales

Los algoritmos de agrupamiento conceptual se componen de dos tareas fundamentales, las cuales no tienen necesariamente que ser independientes ni realizarse en un orden determinado (Reyes-González 2014):

1. La estructuración o determinación extensional: se lleva a cabo el proceso de agrupar entidades, en el que se determinan grupos a partir de una colección de objetos, esto no es más que la enumeración de los objetos que lo componen los grupos.
2. La caracterización o determinación intencional: se determina el concepto de cada grupo de la estructuración, las propiedades que caracterizan el agrupamiento.

Los algoritmos de agrupamiento conceptual se pueden dividir en dos grandes grupos, a saber, los algoritmos incrementales y los no incrementales (Pérez-Valls 2013). Los algoritmos incrementales basan su funcionamiento en la adaptación de los agrupamientos (o conceptos) con los nuevos objetos que se le van presentando, es decir, cada vez que llega un nuevo objeto mediante una cierta estrategia éste es clasificado en los agrupamientos ya existentes o se crean nuevos agrupamientos. Por otro lado, los algoritmos no incrementales estructuran una muestra de objetos sin presuponer que éstos llegan de uno en uno (Ruiz-Shulcloper 2009).

En los trabajos de Ruiz-Shulcloper,(2009) y Reyes-González,(2014) se realiza un análisis crítico de diferentes algoritmos de agrupamiento conceptual, atendiendo a sus características y funcionamiento, destacándose entre sus principales resultados los siguientes:

1. La principal desventaja de los algoritmos conceptuales de tipo incremental es la dependencia del resultado (la estructuración) en función del orden de presentación de los objetos al algoritmo mientras que en los de tipo no incremental se determina el número de las agrupaciones de manera aleatoria, lo que constituye una dificultad en la vida real, debido al desconocimiento de los posibles agrupamientos en ciertos tipos de problemas.
2. Los algoritmos que no pertenecen al enfoque lógico combinatorio del reconocimiento de patrones construyen sus conceptos en función de criterios probabilísticos o estadísticos, por lo que su interpretación puede ser engorrosas para personas no especializadas en esas áreas del conocimiento.
3. Los algoritmos LC-Conceptual y RGC (ambos pertenecientes al RLCP) construyen sus conceptos en función de propiedades lógicas, basadas en los rasgos de los objetos en estudio. Además de que no requieren que sean especificados a priori el número de agrupamientos. La dificultad que poseen estos algoritmos, es la complejidad computacional en el cálculo de los testores típicos.

La descripción de los conceptos en función de propiedades lógicas basadas en los rasgos de los objetos, puede constituir una variante para la explicación del significado que puede tener una regla de producción en función de la premisa de la regla. En la bibliografía consultada se aprecia que se han utilizados los motores de reglas abordando diferentes técnicas de la Inteligencia Artificial, sin embargo, se ha demostrado que pueden producirse encadenamiento infinito e ineficiente en la inferencia de las reglas.

Un aspecto a tener en cuenta para afrontar esta situación, sería asociar al motor de reglas del reconocimiento lógico combinatorio de patrones aplicando los algoritmos de agrupamientos conceptuales definiéndose una nueva arquitectura del motor de inferencia.

1.4.1 Algoritmo LC-Conceptual

El algoritmo LC-Conceptual fue propuesto en el año 2001 por Martínez-Trinidad y Ruiz-Shulcloper,(1999), está basado en los conceptos de la clasificación no supervisada en el enfoque lógico-combinatorio y retoma algunas ideas propuestas por Michalski para generar conceptos, interpretables por los especialistas, en términos del conjunto de rasgos original.

El algoritmo LC-Conceptual incluye dos etapas:

Etapa de estructuración extensional: se construyen los agrupamientos de objetos basándose en la semejanza entre los mismos y utilizando un criterio de agrupamiento.

Etapa de estructuración intencional o conceptual: se construyen las propiedades (conceptos) que caracterizan a cada agrupamiento de objetos utilizando el operador de refunión condicionada y los testores típicos.

Paso 1: Se determinan, los criterios de comparación por rasgos, la función de semejanza y el criterio de agrupamiento. Se estructura la muestra inicial en función de estos parámetros, siendo considerado cada grupo formado como una clase. Posteriormente se procede a calcular todos los testores típicos para cada clase.

Paso 2: Se forma una matriz de entrenamiento ME o matriz de aprendizaje MA a partir de la matriz inicial MI y de los agrupamientos o clases K_1, …, K_c, luego se calcula el conjunto de los testores típicos de $ME_T = \{t_1, \ldots, t_p\}$. Para cada clase K_i, i = 1,.. , c se emplea el operador de Refunión Condicionada para construir los l-complejo (entiéndase por l-complejo como el producto lógico de todos los rasgos que cubren un concepto o el conjunto de todos los valores diferentes que pueden tomar los objetos de dicho concepto (Pons-Porrata 2004)) y a su vez construir los conceptos (Reyes-González 2014) (ver figura 4).

El operador de Refunión Condicionada (RUC): forma los l-complejos garantizando que los l-complejos de cada agrupamiento no satisfacen a ningún objeto de otros grupos, pero no de manera independiente, sino en combinación con el resto de las variables (Pons-Porrata 2004).

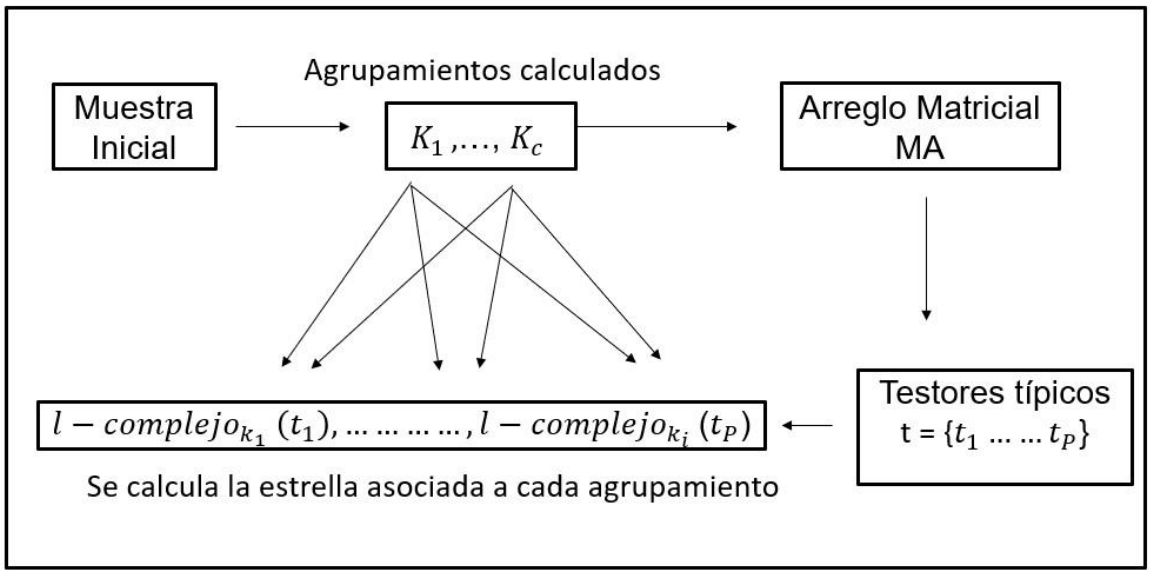


Figura 4 *Proceso del agrupamiento conceptual utilizando el algoritmo LC-Conceptual (Martínez-Trinidad y Sánchez-Díaz 2001)*

Este algoritmo presenta las siguientes limitaciones:

1. Se aplica el operador de refunión condicionada, el cual garantiza que el concepto de un agrupamiento no sea satisfecho por ningún objeto de otro agrupamiento, pero entonces, no logra que este concepto cubra a todos los objetos de dicho agrupamiento.
2. No permite el empleo de otro algoritmo de reducción de rasgo que no sea el testor típico.
3. No incluye la regla de generalización como parte del proceso del algoritmo.

1.4.2 Algoritmo RGC

El algoritmo conceptual RGC fue propuesto en el año 2004 por Aurora Pons Porrata (Pons-Porrata 2004) está basado en los conceptos de la clasificación no supervisada del Reconocimiento Lógico Combinatorio de Patrones y retoma algunas de las ideas planteadas por Michalski para generar los conceptos, en función del conjunto de rasgos original.

El algoritmo RGC cuanta con dos etapas como se muestra en la figura 5:

I Etapa de determinación extensional: Primeramente, se realizan los agrupamientos, usando los criterios de comparación entre los rasgos, la función de semejanza entre objetos

y un criterio de agrupamiento para generar la estructuración por clases. Para ello, podrá emplearse cualquier algoritmo de Clasificación no supervisada como grupo de apoyo.

II Etapa de determinación intencional: Después de haber aplicado la etapa extensional se emplea el operador de refunión extendida para el cálculo de los l-complejos, se aplican las

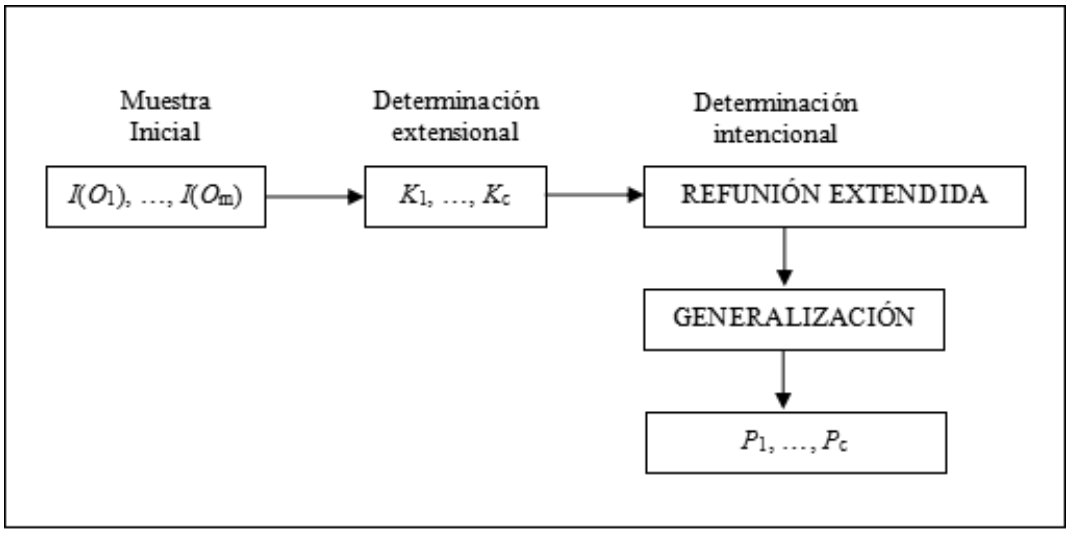


Figura 5 *Proceso del agrupamiento conceptual del RGC (Pons-Porrata 2004)*

reglas de generalización y se construyen los conceptos que corresponde a cada clase K_i.

Los pasos para aplicar el algoritmo RGC son:

1- Se construye las propiedades (conceptos) que caracterizan a cada grupo de objetos.

2- Se forma una matriz de entrenamiento a partir de MI y de los grupos K_1, …, K_c, luego se calcula el conjunto de apoyo en este caso es los testores típicos de $ME_T = \{t_1, \ldots, t_p\}$.

3- Para cada clase K_i, i = 1, …, c se calcula la estrella G_T ($K_i \backslash K_1$, ..., K_{i-1}, K_{i+1}, ..., K_c) que estará formada por l- complejos.

4- Para el cálculo de los l-complejos se emplea el operador de Refunión Extendida (RUE) (Pons-Porrata 2004). Luego se aplican las reglas de generalización.

5- Los conceptos que caracterizan a cada agrupamiento K_i serán los l-complejos obtenidos en el paso anterior.

El operador de refunión extendida (RUE): forma los conceptos garantizando que los l-complejos de cada agrupamiento no satisface a ningún objeto de otros grupos; pues si un nuevo objeto no puede ser añadido a ninguno de los l-complejos existentes se crea, al menos, un nuevo l-complejo que lo cubre (Pons-Porrata 2004).

1.4.3 Comparación entre los algoritmos conceptuales

En Pérez-Suárez y Medina-Pagola,(2014) y Reyes-González,(2017) se presenta un estado del arte en la temática de los algoritmos conceptuales, se realiza una comparación entre ellos. En esta investigación se presentan solo la comparación del LC- Conceptual y RGC teniendo en cuenta que estos son los algoritmos de referencia para el trabajo.

Los criterios que los autores tienen en cuenta para establecer la comparación entre los algoritmos coinciden con el de la investigación (Roque y Luna 2018):

La complejidad computacional: la métrica de medición se va a establecer en fácil, medio, alta y muy alta. En este caso los dos algoritmos tienen una complejidad computacional muy alta establecido expresado en la investigación (Reyes-González 2017).

La estructuración: la métrica se establece en conjunto o no conjunto. El algoritmo conjunto está definido por las instrucciones ordenadas y finitas que permite realizar una tarea (Montero-Montes 2012). El no conjunto es el que no cumple con las instrucciones ordenadas y tiene distintas salidas. En el caso de los dos algoritmos de objeto de comparación según sus autores (Martínez-Trinidad y Sánchez-Díaz 2001;Pons-Porrata 2004) son conjunto.

Tipo de atributo: se definen los tipos de atributos que puede tener los rasgos de la base de conocimiento que se puede trabajar. En Pérez-Suárez y Medina-Pagola,(2014) se considera que una de las características fundamentales, a los efectos de su investigación, radica en el tipo de atributo que admiten los algoritmos y en este sentido LC- Conceptual y RGC (basados en el Reconocimiento Lógico Combinatorio de Patrones) son los de mejores resultados pues permiten trabajar con datos mezclados e incompletos, lo cual es muy frecuente en los problemas prácticos.

Regla de generalización: se basa en tener en cuenta las reglas de generalización. En este caso según sus creadores (Martínez-Trinidad y Sánchez-Díaz 2001)(Pons-Porrata 2004), en el caso del LC-Conceptual no lo tiene incluido y el RGC sí lo tiene presente.

Excluyente: se analiza si en la forma de construir los conceptos el cual indica que no son satisfechos por ningún objeto de otra clase (Pons-Porrata 2004). Por lo que los dos algoritmos lo tienen presente cuando se construyen los conceptos.

Caracterizante: se refiere a que cubren a todos los objetos de la clase que describen (Pons-Porrata 2004). Esto ocurre en la forma de construir los l-complejos.

Sin embargo, la diferencia radica en que el RGC cumple con las propiedades de ser excluyente y caracterizante a diferencia del LC-Conceptual que solo puede ser excluyente, no garantiza que sea caracterizante (Pons-Porrata 2004), al igual el algoritmo RGC tiene incluida las reglas de generalización y el LC-Conceptual no, como se muestra en la Tabla 1. Por lo expuesto en la presente investigación se propone aplicar el algoritmo RGC para mejorar la eficacia en la clasificación de los objetos a través del motor de reglas de producción.

Tabla 1: Comparación de los algoritmos conceptuales

Criterios de Comparación	**Algoritmos Conceptuales**	
	LC- Conceptual	**RGC**
Complejidad	Muy alta	Muy alta
Estructuración	Conjunto	Conjunto
Atributos	Mezclados incompletos	Mezclados incompletos
Regla de generalización	No	si
Excluyentes	Si	si
Caracterizante	No	si

Fuente: (Roque y Luna 2018)

1.5 Herramientas y tecnologías utilizadas

Tanto a nivel nacional como internacional existen herramientas capaces de procesar información para dar paso a realizar tareas de aprendizaje automático, entre las que se considera a Weka como la aplicación más utilizada en la actualidad a nivel internacional para la minería de datos y CEPAR es una herramienta desarrollada en Cuba, la cual tiene como fin servir de apoyo a los investigadores del Reconocimiento Lógico Combinatorio de Patrones según propone (Ruiz-Shulcloper 2009).

1.5.1 Weka

Es un software desarrollado por la universidad de Waikato (Nueva Zelanda), implementado en el lenguaje de programación JAVA. Contiene las herramientas necesarias para realizar

transformaciones sobre los datos, tareas de clasificación, regresión, clustering, asociación y visualización. La licencia de WEKA es GPL, lo que significa que este programa es de libre distribución y difusión. Es independiente de la arquitectura (Sistema Operativo), ya que funciona en cualquier plataforma sobre la que haya una Máquina Virtual Java disponible. Está diseñado como una herramienta orientada a la extensibilidad, por lo que añadir nuevas funcionalidades es una tarea sencilla. La aplicación utiliza cubrimiento que se encuentran en documentos con extensión .arff, que son en texto plano (Pico Peña 1999).

El software en general es bastante atractivo con una interfaz visual clara, permite hacer pruebas sobre cubrimientos, brinda distintas opciones para la visualización de los resultados y permite la ausencia de información en los datos. Se distribuye como un software de código abierto, por lo que se puede modificar cualquier elemento.

WEKA posee entre sus inconvenientes, que no permite probar iterativamente nuevos conjuntos e incluir el resultado o el nuevo conjunto clasificado al ya clasificado inicialmente, para hacer esto, hay que exportar a alguna otra aplicación y desarrollar un programa que lo haga. La metodología sobre la cual se desarrolla WEKA, no permite el tratamiento de los datos como lo hace el Reconocimiento Lógico Combinatorio de Patrones, por tanto, no es muy útil a los usuarios que hagan uso este enfoque. La incorporación de algoritmos para el Reconocimiento Lógico Combinatorio de Patrones en WEKA deben traer aparejado un cambio en el mismo núcleo de la herramienta, haciendo que pierda parte de su objetivo inicial (Pico Peña 1999), (Ruiz, Guzman y Martínez 1999).

1.5.2 CEPAR

CEPAR (Entorno Cubano para el Reconocimiento Lógico Combinatorio de Patrones) es un Sistema Herramienta Universal (SHU) desarrollado según las teorías y presupuestos del Reconocimiento Lógico Combinatorio de Patrones. Tiene como objetivo fundamental apoyar en las labores cotidianas de los investigadores, docentes y estudiantes de estas teorías; además de proveer de una herramienta que pueda ser embebida en desarrollos propios de una investigación. (Reyes González y Martínez Sánchez 2014).

Implementado empleando JAVA como lenguaje de programación, dado sus comodidades como lenguaje multiplataforma, CEPAR solo requiere de la Máquina Virtual de Java (JVM) para poder ser utilizado.

CEPAR plantea un diseño modular, de manera que permite la incorporación de nuevos rasgos, funciones de semejanza. Cada módulo es independiente y se relaciona con los demás a través de las clases e interfaces definidas en su núcleo (cepar.core), que contiene las características más generales para modelar un problema de Reconocimiento Lógico Combinatorio de Patrones.

La herramienta permite el trabajo simultáneo con variables cualitativas y cuantitativas (todos los tipos de rasgos soportan la ausencia de información o valor "?"), y maneja de forma nativa los tipos:

- BooleanFeature: valores booleanos.

- DateFeature: valores de tipo fecha (día-mes-año).

- DoubleFeature: valores reales.

- FloatFeature: valores de punto flotante.

- IntegerFeature: valores enteros.

- StringFeature: valores alfanuméricos.

Esta cantidad de tipos de rasgos resulta mínima, en comparación con los que pueden aparecer en problemas reales del Reconocimiento Lógico Combinatorio de Patrones, por lo que resalta en CEPAR el permitir la extensión según las interfaces definidas de nuevos rasgos por los usuarios, pero limitando la herramienta sólo al trabajo con los seis rasgos nativos.

Por las potencialidades del uso del enfoque del Reconocimiento Lógico Combinatorio de Patrones que brinda la herramienta CEPAR y que es capaz de insertar nuevos conjuntos de datos clasificados a los ya clasificados inicialmente, en la presente investigación se utiliza la herramienta como librería para establecer las bases en la implementación del algoritmo conceptual RGC.

1.6 Lenguaje de programación

Los lenguajes de programación facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por las personas y a su vez resultan independientes del modelo de computador a utilizar (Pulido, Romero y Mendoza 2019).

Para la implementación del algoritmo conceptual RGC se propone la utilización de java en su versión 8 y para la realización de las pruebas estadísticas se propone el uso de R.

1.6.1 Java 8

Versión más reciente de Java que incluye nuevas características, mejoras y correcciones de bugs para mejorar la eficacia en el desarrollo y la ejecución de programas Java («Información sobre Java 8» 2019), (Córdova y José 2018). A continuación, se muestra un breve resumen de las mejoras que se incluyen en esta versión:

- Detectar y eliminar versiones de Java antiguas (Windows) a partir de Java 8 Update 20 (8u20), en los sistemas Windows, la herramienta de desinstalación de Java está integrada con el installer para contar con una opción para eliminar las versiones anteriores de Java del sistema. El cambio se aplica a plataformas Windows de 32 bits y 64 bits.
- Métodos de extensión virtual y expresión Lambda: una de las funciones destacables de Java SE 8 es la implantación de expresiones Lambda y funciones adyacentes a la plataforma y el lenguaje de programación Java.
- API de fecha y hora: esta nueva API permitirá a los administradores gestionar datos de fecha y hora de forma mucho más natural y fácil de comprender.
- Motor de JavaScript Nashhorn: nueva implantación ligera de alto rendimiento del motor de JavaScript integrada en JDk y disponible en las aplicaciones Java mediante las API existentes.
- Seguridad mejorada: sustitución de la lista de métodos sensibles al emisor mantenida a mano existente por un mecanismo que identifica con mayor precisión dichos métodos y permite detectar a los emisores de forma fiable.

1.6.2 R

R es un lenguaje y un entorno para computación y gráficos estadísticos que proporciona una amplia variedad de técnicas estadísticas y gráficos, y es altamente extensible. El lenguaje R suele ser el vehículo de elección para la investigación en metodología estadística y proporciona una ruta de código abierto para la participación en esa actividad (Team 2015).

Uno de los puntos fuertes de R es la facilidad con la que se pueden producir parcelas de calidad de publicación bien diseñadas, que incluyen símbolos matemáticos y fórmulas

cuando es necesario. Se ha tenido mucho cuidado con los valores predeterminados para las opciones menores de diseño en los gráficos, pero el usuario conserva el control total. R está disponible como software libre bajo los términos de la Licencia Pública General GNU de la Free Software Foundation en forma de código fuente. Se compila y se ejecuta en una amplia variedad de plataformas UNIX y sistemas similares (incluidos FreeBSD y Linux), Windows y MacOS (Team 2015).

R permite a los usuarios agregar funciones adicionales definiendo nuevas funciones. Gran parte del sistema está escrito en el dialecto R, lo que facilita a los usuarios seguir las elecciones algorítmicas realizadas. Para tareas intensivas en cómputo, el código C, C ++ y Fortran se puede vincular y ejecutar en tiempo de ejecución. Los usuarios avanzados pueden escribir código C para manipular objetos R directamente. (Team 2015)

1.7 Entorno de desarrollo integrado

Un Entorno de Desarrollo Integrado (IDE), es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Como entorno de desarrollo integrado se utilizará NetBeans en su versión 8.2 para la implementación del algoritmo conceptual y RStudio con el objetivo de realizar las pruebas no paramétricas de Friedman.

1.7.1 NetBeans

Está escrito en Java, pero puede servir para lenguajes de programación como C++, HTML5, JavaScript y otros. Existe además un número importante de módulos para extender el Netbeans IDE es un producto libre y gratuito sin restricciones de uso («NetBeans IDE - Overview» 2019).

A continuación, algunas ventajas de este IDE que se consideran importantes en la investigación:

- Es un IDE multilenguaje y adaptable.
- Es software libre y gratuito.
- Intuitivo y fácil de utilizar.
- Se puede desarrollar todo tipo de aplicaciones.

- Es poderoso y extensible.

1.7.2 RStudio

Basado en el lenguaje de programación R, dedicado a la computación estadística y gráficos. Incluye una consola, editor de sintaxis que apoya la ejecución de código, así como herramientas para el trazado, la depuración y la gestión del espacio de trabajo. Es multiplataforma para los sistemas operativos Windows, Mac y Linux o para navegadores conectados a RStudio Server o RStudio Server Pro. RStudio tiene la misión de proporcionar el entorno informático estadístico R. Permite un análisis y desarrollo para que cualquiera pueda analizar los datos con R (Team 2015).

1.8 Conclusiones parciales

Como resultado del estudio para establecer los referentes teóricos de mayor interés en el área de conocimiento de los Sistemas Basados en Reglas queda claro que:

- En los Sistemas Basados en Reglas, la arquitectura puede variar en dependencia del problema a resolver y la complejidad del mismo, además los motores de reglas pueden ayudar en una mejor organización de los Sistemas antes mencionados.
- La importancia informacional de rasgos y conceptos, puede ser un elemento a tener en cuenta para establecer las distintas reglas de producción.
- El RGC-Conceptual es un algoritmo que puede resultar útil para abordar la problemática planteada por su propiedad de ser caracterizante.
- Se determinó a CEPAR como herramienta que trabaja con el RLCP la cual se va a utilizar como librería para la realización del algoritmo RGC. Para el desarrollo de la solución se determinó emplear Java y R como lenguajes de programación y NetBeans en su versión 8.0.2 como entorno de desarrollo integrado para la implementación de la solución.

Capítulo 2: Diseño del Sistema Basado en Reglas

En el presente capítulo se describe la propuesta de solución al aplicar un Sistema Basado en Reglas usando un motor de Reglas diseñado a partir del uso del algoritmo conceptual RGC. Se realiza una descripción detallada de cada una de las etapas de dicho algoritmo, y se definen las reglas correspondientes para cada grupo, mediante la importancia informacional de rasgos y conceptos obtenida por la clasificación de algoritmo RGC.

2.1 Descripción de la propuesta de solución

La propuesta de solución está conformada por dos elementos con sus fases respectivamente. Se determinó el uso y diseño de un motor de reglas de tipo parcialmente independizado (abordado en el subepígrafe 1.2.1), separando la BC de las reglas de producción y el motor de inferencia, para su tratamiento con el uso del algoritmo conceptual RGC.

La primera fase se realizan los procesos relacionados con las etapas de determinación extensional e intencional del algoritmo conceptual RGC para establecer los conceptos dada una base de conocimiento. En la segunda fase se transforman los conceptos en formatos de reglas y se determina la cantidad de reglas a usar para su posterior uso por el motor de inferencia. Se muestra en la figura 6.

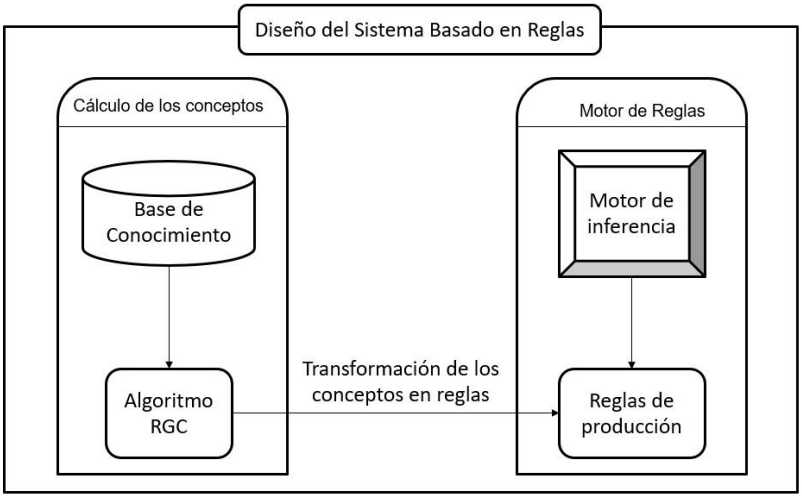


Figura 6 *Propuesta del Sistema Basado en Reglas. Fuente (Elaboración Propia)*

El punto de partida es la utilización de una base de conocimiento, esta será procesada por el algoritmo conceptual RGC, el cual permite realizar un análisis profundo sobre las características de los datos y su significado agrupándolos por sus características similares. Estos se determinan a través de los testores típicos, así como la importancia informacional de los rasgos que lo conforman, y se selecciona el testor típico de mayor relevancia con el que se construyen los conceptos distintivos para cada grupo.

Los procesos de la segunda etapa, comienzan con la obtención de los conceptos devueltos por el algoritmo RGC, y la transformación de los mismos en reglas de producción para luego su uso por el motor de inferencia. Esto es esencial, dado que el formato de los conceptos en su forma natural es imposible de usar por el motor de reglas.

Fase 1: Cálculos para la determinación de los conceptos utilizando el RGC

En esta fase se aplica el algoritmo RGC para tomar el testor típico de mayor peso informacional y el número de conceptos formados.

Objetivo: Calcular los testores típicos y seleccionar el de mayor peso informacional y los conceptos asociados a cada clase.

Independientemente del tipo de problema a resolver lo constituye el planteamiento formal del mismo, con lo que se persigue precisar el objetivo, así como determinar las fuentes de información con las cuales se trabaja. La presencia del especialista del área de aplicación es de vital importancia, pues sus criterios deciden en gran medida varios de los aspectos a tener en cuenta, tales como: cuáles son los rasgos y su importancia, cómo se comparan las variables y los objetos, determinar la forma en que se representan los objetos de estudio, si se tienen clases, si son disjuntas o no, duras o difusas. Para este proceso se propone adoptar la metodología descrita en (Ruiz-Shulcloper 2009).

Del enfoque lógico-combinatorio para problemas de clasificación sin aprendizaje, se conoce que los objetos pueden ser representados en una matriz inicial (MI), $MI = \{I(O_1), I(O_2), \dots, IO_m\}$ conjunto de descripciones de los objetos $O_1, O_2, \dots, O_m$ de un universo U, dadas como $I(O_n) = \big(x_1, (O_i), \dots, x_n(O_n)\big)$, como se puede observar en la tabla 2. Para cada x_i se tiene asociado un conjunto de valores admisibles M_i $i = 1, \dots, n,$ consecuentemente el espacio de representación inicial de los objetos, no es otra

cosa que $M_1 \times M_2 \times, \ldots, \times M_n$ el producto cartesiano de los conjuntos admisibles de valores de los rasgos $x_1, .. x_n$. Sobre M_i se define un criterio de comparación de valores:

C_i: donde G es un conjunto dado ($G = [0,1], G = \{0,1\}, G = \{1, \ldots, k\}\ y\ otros$)

Tabla 2 Matriz inicial (Roque y Luna 2018).

	x_1	x_2	...	x_n
O_1	$x_1\ (O_1)$	$x_2\ (O_1)$	...	$x_n\ (O_1)$
O_2	$x_1\ (O_2)$	$x_2\ (O_2)$	...	$x_n\ (O_2)$
⋮		⋮		
O_m	$x_1\ (O_m)$	$x_2\ (O_m)$	...	$x_n\ (O_m)$

Cada rasgo constituye una variable aleatoria con valores discretos (nominales u ordinales) o continuos, y se admite la ausencia de información, representada por el símbolo *. En dependencia de la naturaleza del rasgo pueden utilizarse diferentes criterios de comparación tales como los mostrados en las ecuaciones 2 a 5 ó pueden construirse nuevas funciones que no necesariamente son booleanas.

Ecuación 2: Criterio de comparación igualdad estricta

$$C_s\left(X_s(O_i), X_s(O_j)\right) = \begin{cases} 1 \quad si\ X_s(O_i) = X_s(O_j)\ \vee\ X_s(O_i) = *\ \vee\ X_s(O_j) = * \\ 0\ en\ otro\ caso \end{cases}$$

Ecuación 2: Intervalos de valores semejantes

$$C_s\left(X_s(O_i), X_s(O_j)\right) = \begin{cases} 1 \quad si\ X_s(O_i), X_s(O_j)\ \in\ [a_p, a_{p+1}] \vee\ X_s(O_i) = *\ \vee\ X_s(O_j) = * \\ 0\ en\ otro\ caso \end{cases}$$

Ecuación 3: Umbral de error admisible de semejanza

$$C_s\left(X_s(O_i), X_s(O_j)\right) = \begin{cases} 1 \quad si\ \left|X_s(O_i) - X_s(O_j)\right| \le \varepsilon_s\ \vee\ X_s(O_i) = *\ \vee\ X_s(O_j) = * \\ 0\ en\ otro\ caso \end{cases}$$

Ecuación 4: Conjunto de valores semejantes

$$C_s\left(X_s(O_i), X_s(O_j)\right) = \begin{cases} 1 & si\ X_s(O_i), X_s(O_j) \in A_p \vee X_s(O_i) = * \vee X_s(O_j) = * \\ 0 & en\ otro\ caso \end{cases}$$

Entre las descripciones de objetos se define una función de semejanza:

$$MS = \left|\left|\beta(I(O_i), I(O_n))\right|\right|_{m \times n}$$

A partir de MI y β se puede construir una matriz que refleje las relaciones de semejanza entre todos los objetos sujetos a estudio. A esta matriz se le llama matriz de semejanza (MS) y es:

MS es simétrica y $\beta(I(O_i), I(O_n) = 1, i = 1, \dots, m$.

La función de Semejanza β determina una medida numérica del grado de similaridad de un objeto con respecto al otro teniendo en cuanta las similitudes entre los rasgos. Esta función unida a la MI son usadas para hallar la matriz de semejanza.

Para determinar la similitud entre rasgos, es importante considerar la naturaleza de los mismos, en dependencia de esta, se utilizan diferentes criterios de comparación.

Umbral de Semejanza: El umbral es la cota inferior de los posibles valores de semejanza existentes para cada uno de los casos recuperados con respecto al nuevo caso. Su función consiste en restringir el intervalo en caso de que se desee lograr una mayor precisión en la obtención de los casos más semejantes.

La magnitud $\beta_0 \in \Delta$ (Δ=[0,1], Δ={0,1}, Δ={1,..,k}, y otros) se denomina umbral de semejanza y puede ser calculada, por ejemplo, según las ecuaciones 6, 7 y 8 .

Ecuación es 6 ,7 y 8 Cálculo del umbral de semejanza:

$$a)\ \beta_0 = \frac{2}{\mathrm{m}(\mathrm{m}-1)} \sum_{\mathrm{i}=1}^{\mathrm{m}-1} \sum_{\mathrm{j}=\mathrm{i}+1}^{\mathrm{m}} \beta(\mathrm{I}(\mathrm{O_i}), \mathrm{I}(\mathrm{O_j}))$$

$$b)\ \beta_0 = \max_{\substack{i=1\dots m-1 \\ i \neq j}} \left\{ \min_{j=i+1\dots m} \{\beta(I(O_i), I(O_j))\} \right\}$$

$$c)\ \beta_0 = \min_{\substack{i=1\ldots m-1 \\ i\neq j}} \left\{ \max_{j=i+1\ldots m} \{\beta(I(O_i), I(O_j))\} \right\}$$

El criterio de agrupamiento unido a la función de semejanza y a la existencia de otros objetos, es la razón por la cual un objeto va a pertenecer a un agrupamiento o por qué dos objetos pertenecerán a una misma agrupación. De esta manera se puede apreciar que la selección del criterio a usar, es determinante en la calidad de la solución del problema de clasificación no supervisado.

La definición del criterio de agrupamiento debe estar basada en el conocimiento que se tenga sobre el problema en concreto que se está tratando, para poder definir así el tipo de comportamiento entre los objetos a partir de sus semejanzas, que resulte significativo, según el problema en particular. Por tanto, al seleccionar algún criterio de agrupamiento, dado un conjunto de objetos y la función de semejanza, se ha definido indirectamente, la familia de agrupaciones, es decir, la estructura del universo ha sido conformada.

El planteamiento formal de la estructuración de universos, de la clasificación no supervisada, consiste en encontrar un criterio de agrupamiento que responda a los intereses del problema en cuestión.

β_0-semejantes

Dos descripciones (objetos) $I(O_n), I(O_j)$ se denominan β_0-semejantes si $\beta(I(O_i), I(O_j)) \geq \beta_0$.Sea un espacio de representación $\Phi = \{\Xi, \beta\}$, sea dado un conjunto de descripciones de objetos $MI = \{I(O_1), I(O_2), \ldots, IO_m\}$, una función de semejanza$\beta: MI \times MI \longrightarrow \Delta$, y un umbral $\beta_0 \in \Delta$ que define la β_0-semejanza entre los elementos de MI.

Criterio agrupacional β_0-conexo

Se dice que $C \subseteq MI, C \neq \emptyset$ es una componente β_0-conexa si y sólo si:

a) $\forall\, O_i\, O_j \in C \in\ O_{i_1}, \ldots, O_{i_q} \in C\ [O_i = O_{i_1} \wedge O_j = O_{i_q} \wedge \forall\, p \in \{1, \ldots, q-1\}\, \beta\ (O_{i_p}, \ldots, O_{i_{p+1}}) \geq \beta_0]$
b) $\forall O_i \in MI\left[\left(O_j \in C, \beta\left(O_i, O_j\right) \geq \beta_0\right) \Rightarrow O_i \in C\right]$
c) Todo elemento β_0-aislado es una componente β_0-conexa (degenerada).

La condición a) expresa que para cualquier par de elementos de C existe una sucesión de elementos en C, que empieza en O_i y termina en O_j tales que uno es β_0-semejante al siguiente, b) significa que no existe fuera de C un elemento β_0-semejante a un elemento de C.

Criterio Agrupacional β_0-compacto

Se dice que *B* (F 9) cMI, *B* ≠o es un conjunto β_0-compacto si y sólo si:

a) $\forall \mathrm{O_j} \in \mathrm{MI}[\mathrm{O_i} \in \mathcal{B} \wedge \max_{\substack{O_t \in MI \\ O_t \neq O_i}} \{\beta(O_i, O_t)\} = \beta(O_i, O_j) \geq \beta_0\ ] \Rightarrow O_i \in \mathcal{B}$

b) $[\max_{\substack{O_t \in MI \\ O_t \neq O_i}} \{\beta(O_P, O_i)\} = \beta(O_P, O_t) \geq \beta_0 \wedge O_t \in \mathcal{B}] \Rightarrow O_i \in \mathcal{B}$

c) $|\mathcal{B}|$ es la mínima

d) Todo elemento β_0-aislado constituye un conjunto β_0-compacto (degenerado).

La condición a) expresa que todo elemento de *B* tiene en *B* al elemento que más se le parece que es β_0-semejante con él. La condición b) significa que no existe fuera de *B* un elemento cuyo elemento más parecido que sea β_0-semejante esté en *B*, la tercera condición c) especifica que *B* debe ser el conjunto más pequeño de cardinalidad mayor que 1.

El algoritmo 1 realiza el agrupamiento de los objetos semejantes, a partir de la MI se obtiene la MS y posteriormente se utiliza un umbral de semejanza $\beta_0 \in \Delta$ y un criterio de agrupamiento para conformar la ME.

Algoritmo 1: "Agrupamiento de los objetos"

Entrada: MI // Matriz inicial

β // Función de semejanza

Π // Criterio de agrupamiento

$C_s\left(X_s(O_i), X_s\left(O_j\right)\right)$ // Funciones de comparación por rasgos

Salida: ME // Matriz estructurada en c agrupamientos (K_c).

Paso 1: Construir la matriz de semejanza utilizando la función de semejanza β.

Paso 2: Calcular el umbral de semejanza utilizando un criterio β_0.

Paso 3: Agrupar siguiendo un criterio agrupacional Π .

Como dato de salida del algoritmo 1 se obtiene la ME, la cual está estructurada en c agrupamientos (K_c) de objetos similares, culminando así la primera etapa con la determinación extensional.

Para la construcción de los conceptos, es necesario seleccionar en primer lugar los rasgos determinantes en el problema, esto es realizado mediante el cálculo de los testores, lo que permite la reducción eficiente de la cantidad de rasgos que describen los objetos. A partir de la Matriz de Entrenamiento (ME) se obtiene la Matriz de Diferencias (MD) y posteriormente la Matriz Básica (MB), mediante la cual se hallan los testores. Se aplica un algoritmo para obtener los testores típicos, luego se obtienen los l-complejos de cada agrupamiento con el operador Refunión Extendida (Martínez-Trinidad y Ruiz-Shulcloper 1999), después el operador Generalización (GEN) y finalmente se forman los conceptos para cada clase perteneciente al problema.

La ME es una matriz estructurada en m clases de objetos similares. La MD una matriz booleana que se obtiene de la ME comparando los respectivos valores de los rasgos en objetos de clases diferentes por medio de los criterios de comparación de valores de las variables (Shulcloper et al. 1995).

La MB es una matriz formada únicamente por filas básicas de la MD, las cuales constituyen el conjunto de testores. Se entiende por fila básica según (Ruiz-Shulcloper 2009).

Se entiende por fila básica según (Ruiz-Shulcloper 2009).

La fila i_t es básica sí y sólo sí en MD no existe fila i_p alguna que sea subfila de i_t.

Sea i_p, i_t filas de MD Se dice que i_p es subfila de $_{it}$ sí y sólo sí:

a) $\forall\forall j\ (ai_tj\ =\ 0 \rightarrow\ ai_pj = 0)$

b) $\forall\exists j_0\ (ai_tj_0 = 1 \wedge ai_pj_0 = 0)$

Algoritmo 2: "Cálculo de los testores típicos"

Entrada: ME// Matriz de Entrenamiento/

Salida: TT//Conjunto de Testores Típicos

Paso 1: Calcular la matriz de diferencias.

Paso 2: Calcular la matriz básica.

Paso 3: Aplicar algoritmo para el cálculo de los TT.

El número de testores típicos para ciertos problemas puede ser muy grande. Esto da como resultado que cada clase podría tener asociado una gran cantidad de conceptos o propiedades, por lo cual, el algoritmo 3 calcula la utilidad de cada testor típico utilizando las propuestas descritas en (Ruiz-Shulcloper 2009). (Reyes-González 2014) para seleccionar el mejor.

Algoritmo 3. Cálculo del testor típico de mayor peso informacional.

Entrada: TT//Conjunto de Testores Típicos

Salida: TT' // Testor típico de mayor peso informacional

Paso 1: Calcular el peso ε_i de los rasgos X_i que aparecen en la familia de testores típicos según:

$$\varepsilon(x_i) = \alpha F(x_i) + \gamma L(x_i)$$

α>O, β>O y α+β=1. α y β parámetros que ponderan la participación o influencia de frecuencia de aparición y la longitud de los testores típicos respectivamente.

$$\alpha = \beta = 0.5 \qquad p(x_i) = \frac{T_i}{|T|} \qquad L(X_i) = \frac{\sum_{t \in T} \frac{1}{|T_i|}}{|T|}$$

T_i : número de testores donde aparece rasgo i.

$|T|$: número de testores.

$|T_i|$: número de rasgos que forman el testor T_i

Paso 2: Seleccionar los testores típicos de menor longitud.

Paso 3: $\Psi(t_i) = \sum_{x \in I} \frac{\varepsilon(x)}{|TT|}$ Calcular el peso de los testores típicos de menor longitud según:

Paso 4: Ordenar descendentemente los testores típicos según su peso.

Paso 5: Seleccionar el testor típico de mayor peso informacional (Ψ), a partir de un umbral previamente definido.

La importancia de utilizar el testor típico de mayor peso informacional radica en que el mismo no contiene todos los rasgos, sino los más relevantes al problema en cuestión, por tanto cuando se haga referencia al conjunto de rasgos, debe entenderse que son sólo aquellos rasgos presentes en el testor típico con que se trabaje (Reyes-González, Yunia et al. 2016).

En el algoritmo 4, se obtienen los l-complejos de cada agrupamiento con el operador Refunión Extendida (Pons-Porrata 2004) luego se aplica el operador Generalización (GEN) y finalmente se forman los conceptos para cada clase perteneciente al problema.

Algoritmo 4. Construcción de los conceptos

Entrada: TT' // Testor típico de mayor peso informacional

ME //Matriz de entrenamiento

Salida: l-complejos //conceptos para cada clase

Paso 1: Calcular l-complejos

Paso 2: Calcular la estrella G_τ ($K_i \backslash K_1$, ..., K_{i-1}, K_{i+1}, ..., K_c) para cada clase K_i, i = 1, …, *c.*

Paso 3: Aplicar reglas de generalización en dependencia del tipo de variable.

La complejidad computacional de este algoritmo depende de la naturaleza de las variables que describen a los objetos, de los criterios de comparación por rasgos y la función de semejanza entre objetos seleccionados, del criterio de agrupamiento que se utilice y de las características que posean los objetos de un mismo agrupamiento. El operador de mayor complejidad computacional es el de Refunión Extendida. En el caso peor la complejidad es exponencial. No obstante, los pasos del operador de Refunión

Extendida pudieran optimizarse explotando las características propias de la función de semejanza y del criterio de agrupamiento empleados(Pons-Porrata 2004).

Fase 2: Composición del motor de reglas

En esta fase corresponde establecer cada uno de los elementos que conforman al motor de reglas; así como determinar la cantidad de reglas correspondientes al mismo, y el motor de inferencia a usar.

Objetivo: Establecer las reglas a partir de los conceptos.

Aspectos a examinar:

Reglas de producción: las reglas están asociadas a los conceptos construidos en la fase 1.

Base de conocimiento: la base de conocimiento a tener en cuenta para resolver el problema de clasificación y determinar las reglas será la obtenida luego de aplicar el algoritmo RGC.

Motor de inferencia: el motor de inferencia a utilizar será uno elaborado bajo el método Modus Ponens, que verificará que se cumplan las condiciones para asignar al grupo al que satisfaga dichas condiciones.

2.2 Ejemplo al aplicar el Sistema Basado en Reglas en la base de datos zoo

A continuación, se muestra un ejemplo al aplicar la base de datos zoo tomado del repositorio de bases de datos internacionales.

La figura 7 muestra la matriz inicial al cargar la base de datos zoo que cuenta con 101 objetos descritos por 18 rasgos:

animal, hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, venomous, fins, legs, tail, domestic, catsize, y type.

El ultimo rasgo ¨*type*¨ es el que agrupa por clases, pero la base de datos se analizó como no supervisada.

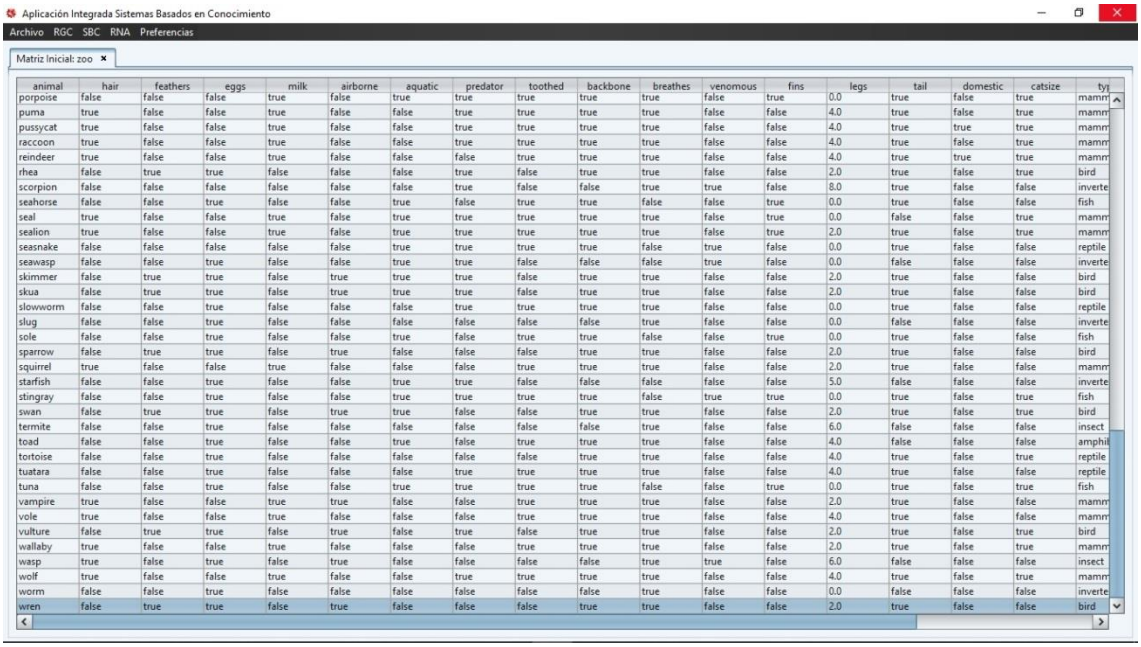


Figura 7 *Matriz inicial. Fuente (Elaboración propia)*

Si la base de datos es no supervisada se comienza con la determinación extensional para agrupar los objetos por clases como se muestra en la figura 8.

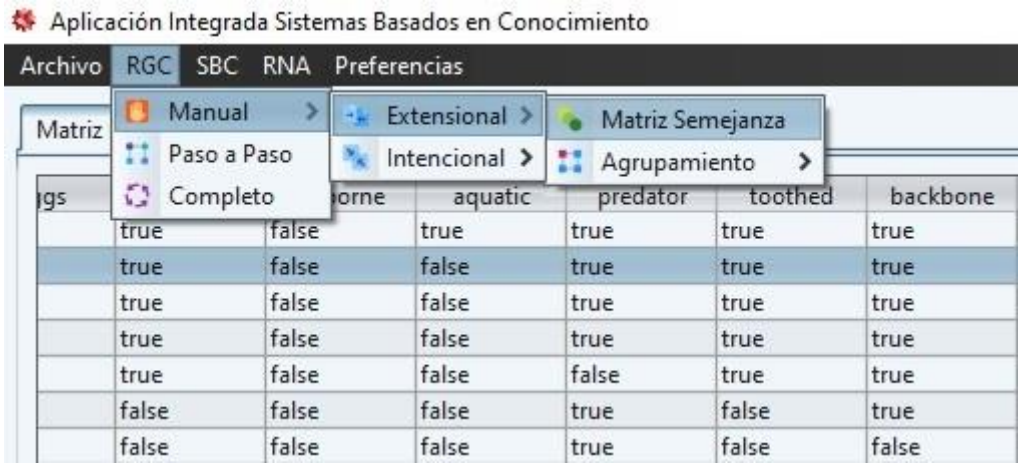


Figura 8 *Calculo de la Diferencial. Fuente (Elaboración propia)*

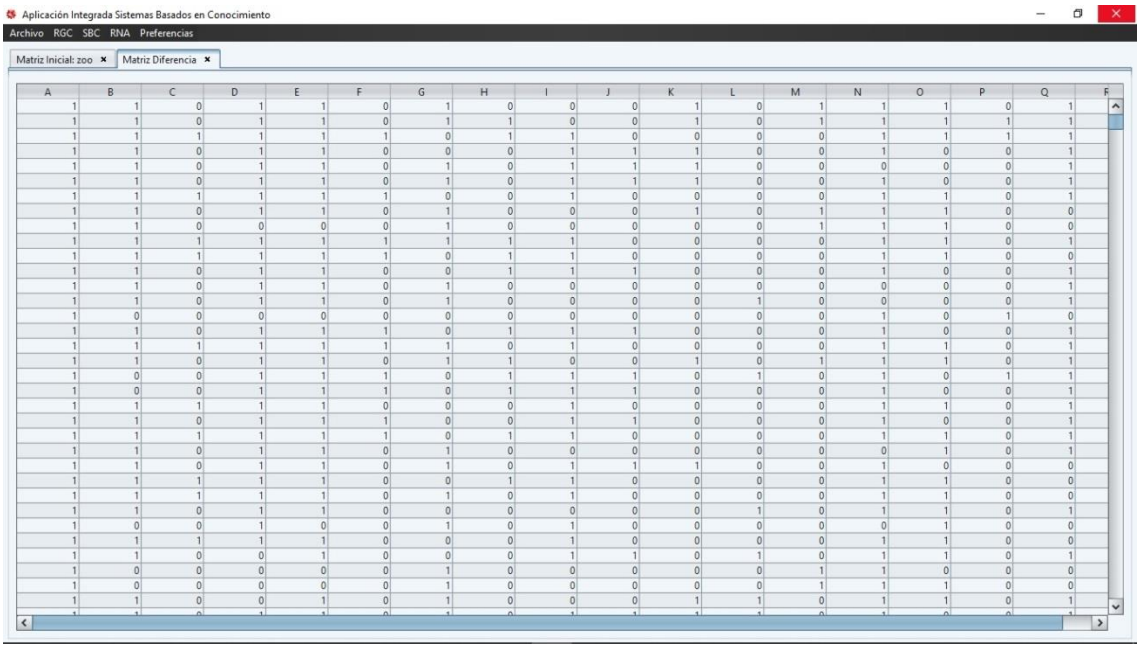


Figura 9 *Ejemplo de la Matriz Diferencial. Fuente (Elaboración propia)*

La selección del criterio de agrupamiento se deja abierta según las características del dominio de aplicación y la valoración de los expertos, pudiendo decidirse entre aquel que calcula las componentes βo-Conexas o los conjuntos βo-Compactos. Debe tenerse en cuenta que en el caso de los conjuntos compactos estos son más restrictivos en cuanto a la formación de los agrupamientos porque los construye con los objetos de máxima semejanza, y por tanto al utilizar este criterio se obtiene un mayor número de conjuntos que con el conexo. Se recomienda utilizar el criterio βo-Conexo y si es necesario volver a aplicar dentro de algún agrupamiento otro criterio entonces se sugiere el βo-Compacto, aunque esta elección depende del problema en particular. (Cabrera Frene, 2018). Ver figuras 10 y 11.

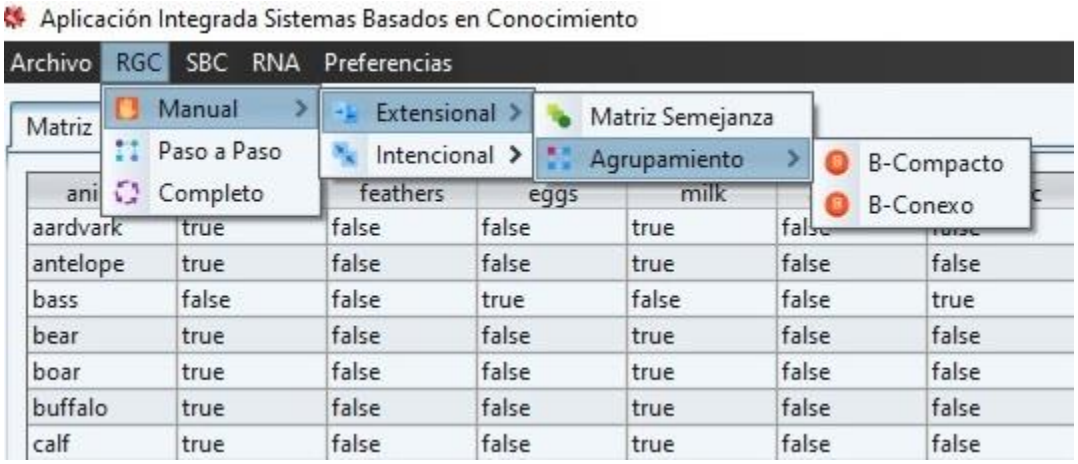


Figura 10 *Variantes para el criterio de agrupamiento. Fuente (Elaboración Propia)*

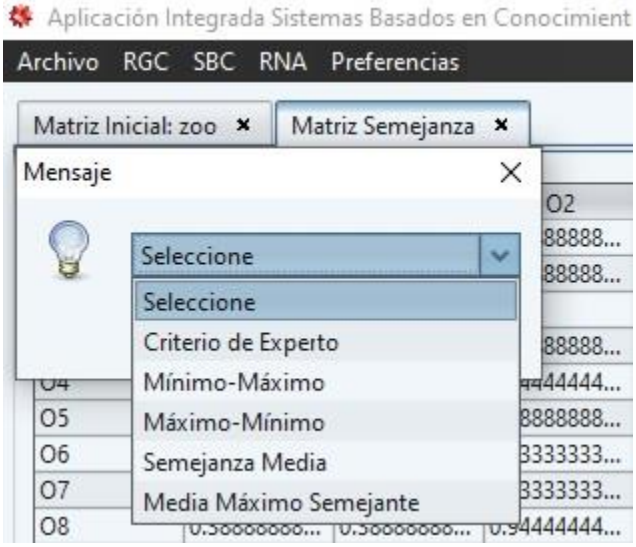


Figura 11 *Criterios de Agrupamiento. Fuente (Elaboración propia)*

Una vez que se obtienen los agrupamientos es necesario determinar el conjunto de apoyo el cual se forma utilizando la teoría de testores como herramienta matemática para la selección de rasgos. De este modo se reduce la cantidad de atributos con los cuales se deben describir los casos y se determinan los que inciden de manera determinante en el problema.

En caso de estar en presencia de datos supervisados se parte de la determinación intencional del algoritmo RGC puesto que ya los objetos están agrupados.

En las siguientes figuras 12 y 13 se presentan los pasos a seguir para obtener finalmente los datos necesarios para completar los elementos a tener en cuenta para la conformación del motor de reglas.

Figura 12 *Calculo de la Matriz Básica. Fuente (Elaboración Propia)*

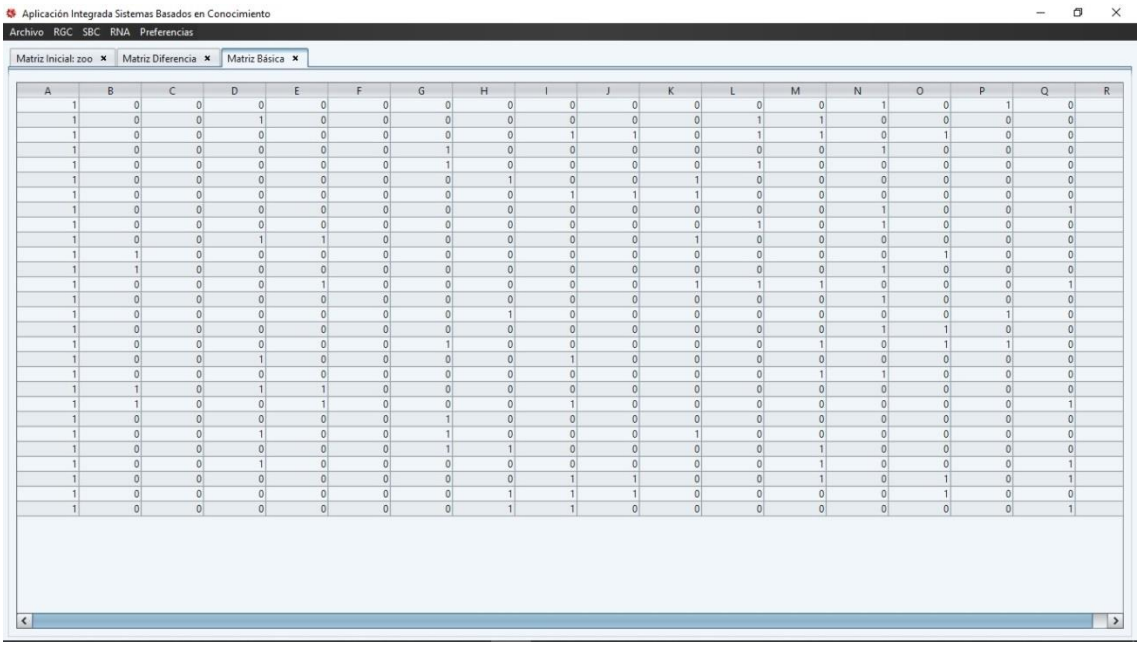


Figura 13 *Ejemplo de la Matriz de Semejanza. Fuente (Elaboración Propia)*

Para el cálculo de los testores típicos se debe seleccionar uno de los algoritmos propuestos (Ruíz-Shulcloper, Alba y Lazo 1994) como se muestra en las figuras 14 y 15.

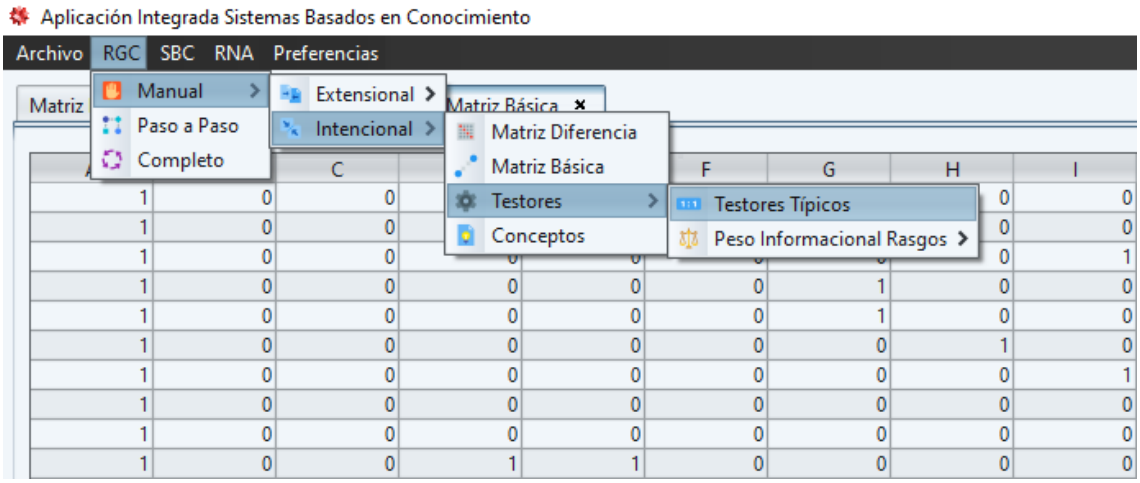


Figura 14 *Testores Típicos. Fuente (Elaboración Propia)*

Según el experimento realizado por (Reyes González 2017) muestra que al aplicar el algoritmo Fast-BR se logra una mayor reducción de los testores típicos, por lo que se recomienda el uso del algoritmo Fast-BR por defecto para realizar el cálculo de los testores típicos. Ver figuras 15 y 16.

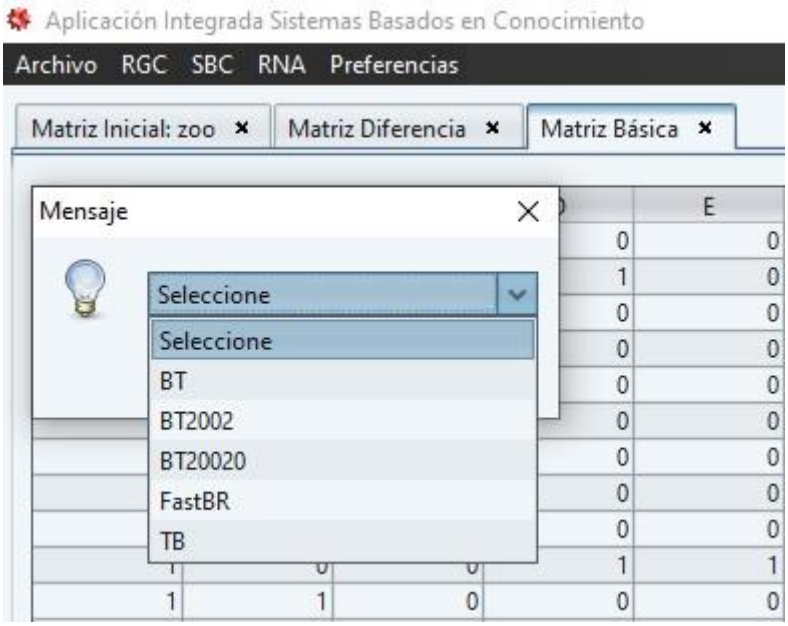


Figura 15 *Algoritmos para el Cálculo de los Testores Típicos. Fuente (Elaboración Propia)*

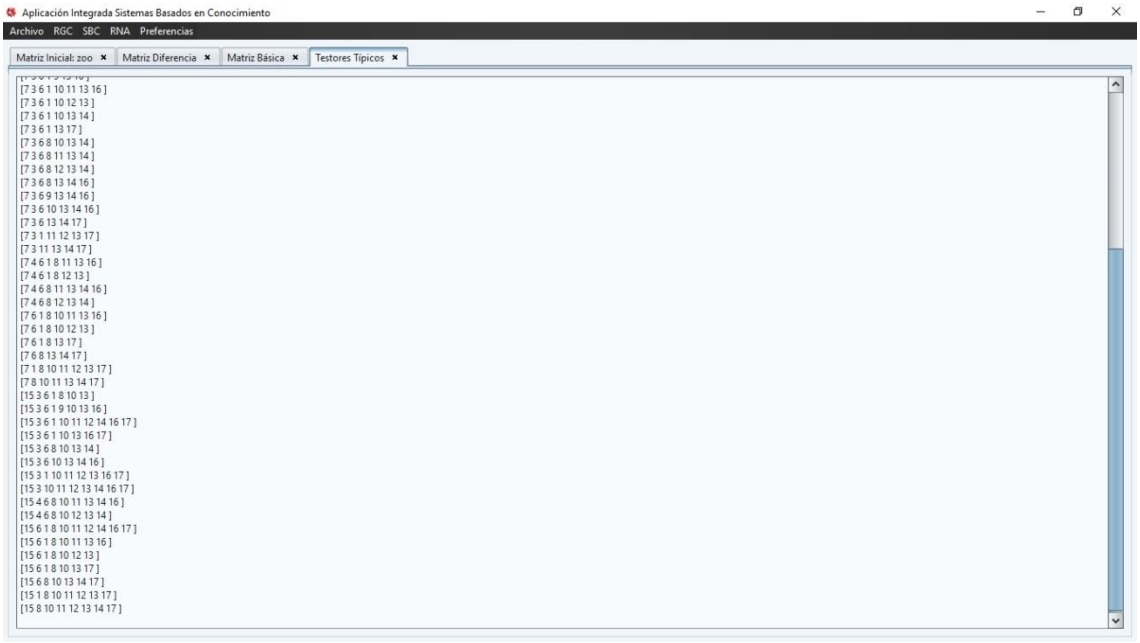


Figura 16 *Ejemplo de los Testores Típicos. Fuente (Elaboración Propia)*

En la figura 17 se presenta cómo calcular el peso informacional de los rasgos. Para ello se cuenta con tres opciones: Frecuencia, Longitud y Frecuencia +Longitud, de las cuales el experto debe escoger una de ellas considerando su criterio para la solución del problema.

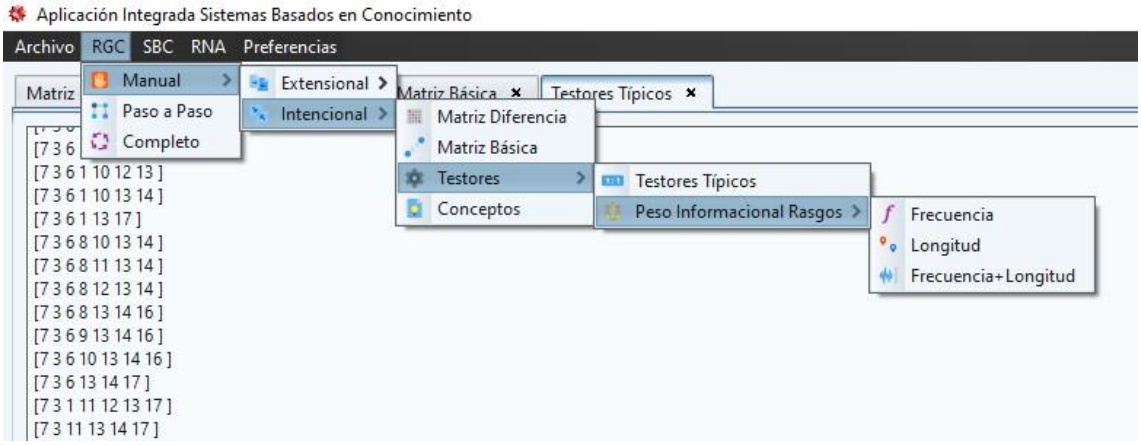


Figura 17 *Peso Informacional. Fuente (Elaboración Propia)*

En el cálculo de la estrella para cada l-complejo de cada clase Ki se generan todas las combinaciones posibles con los valores que toman los rasgos que los conforman y se seleccionan aquellos l-complejos que sólo representan a objetos de la clase Ki, siendo estos excluyentes y caracterizantes.

Los l-complejos son considerados excluyentes cuando no existe ningún objeto fuera del agrupamiento K_i que esté cubierto por los conceptos generados de dicho agrupamiento y se consideran caracterizantes cuando el concepto generado cubre a todos los objetos existentes en K_i.

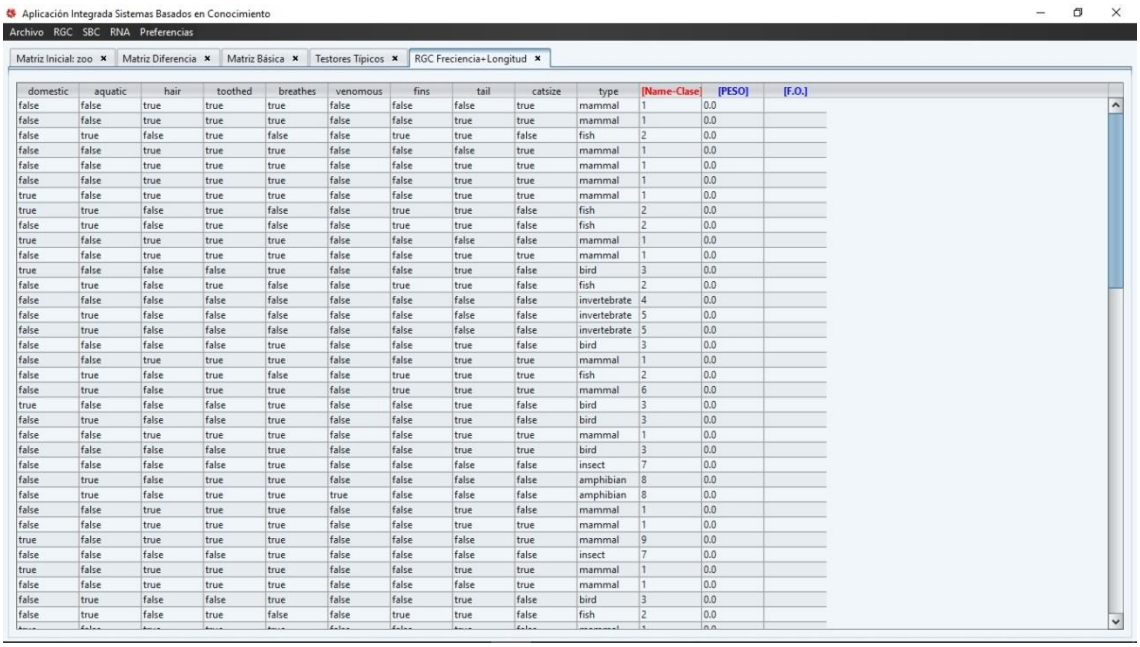


Figura 18 *Ejemplo de la Matriz RGC. Fuente (Elaboración Propia)*

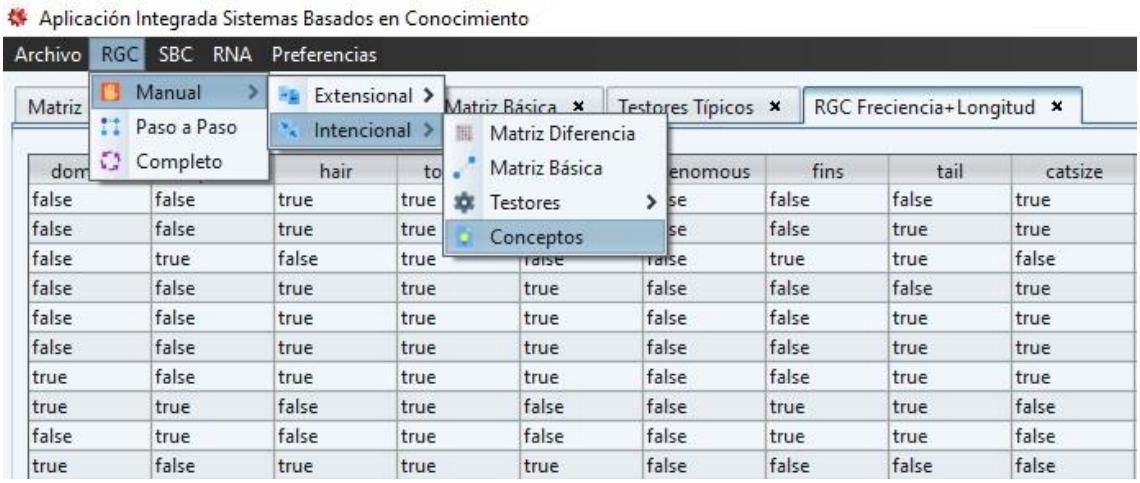


Figura 19 *Obtención de los Conceptos. Fuente (Elaboración Propia)*

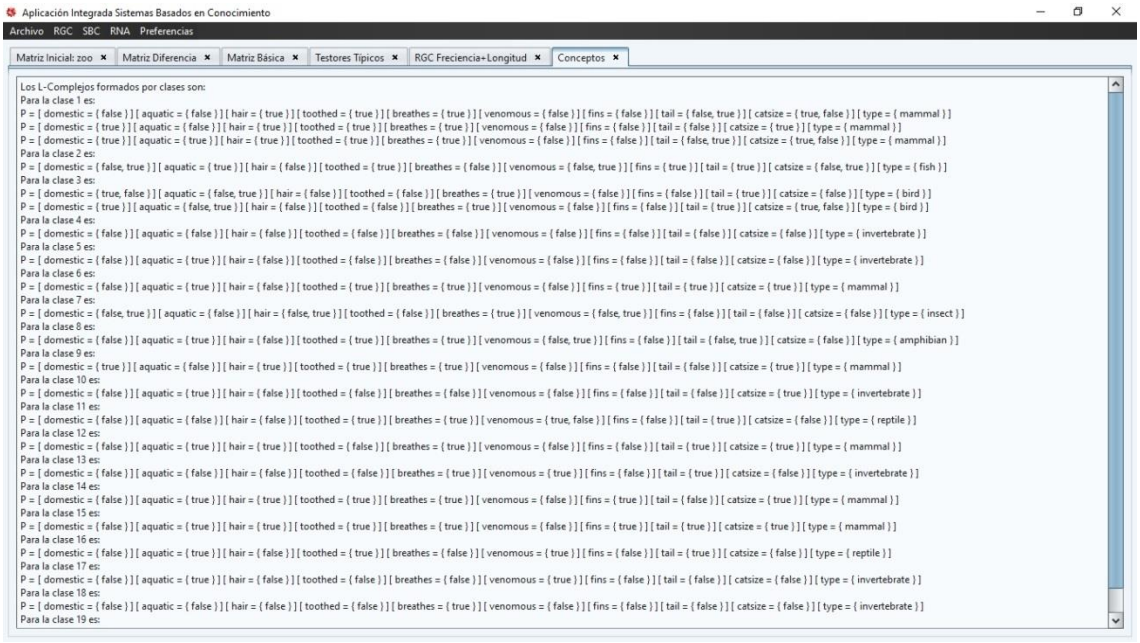


Figura 20 *Ejemplos de los Conceptos. Fuente (Elaboración Propia)*

Una vez obtenidos los conceptos se realiza la transformación de los mismos en reglas como se puede apreciar en la figura 21 y 22.

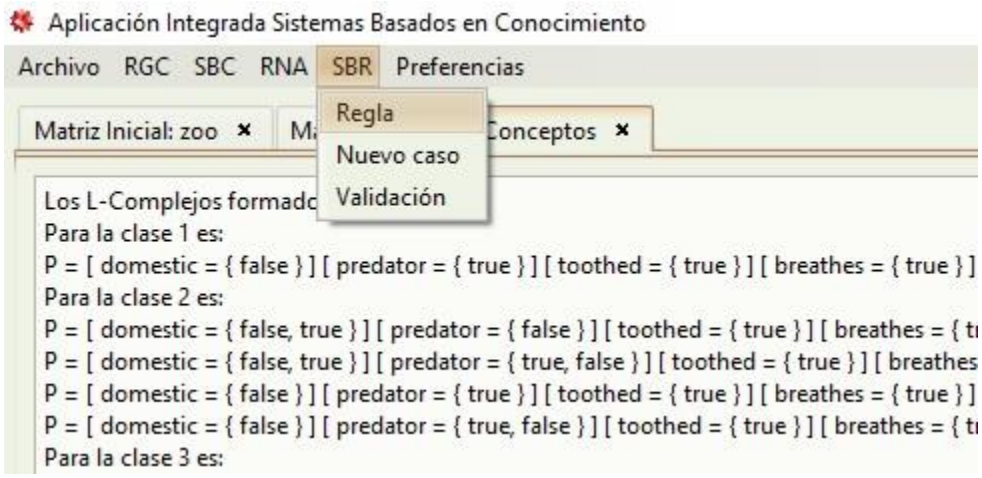


Figura 21 *Reglas obtenidas. Fuente (Elaboración Propia)*

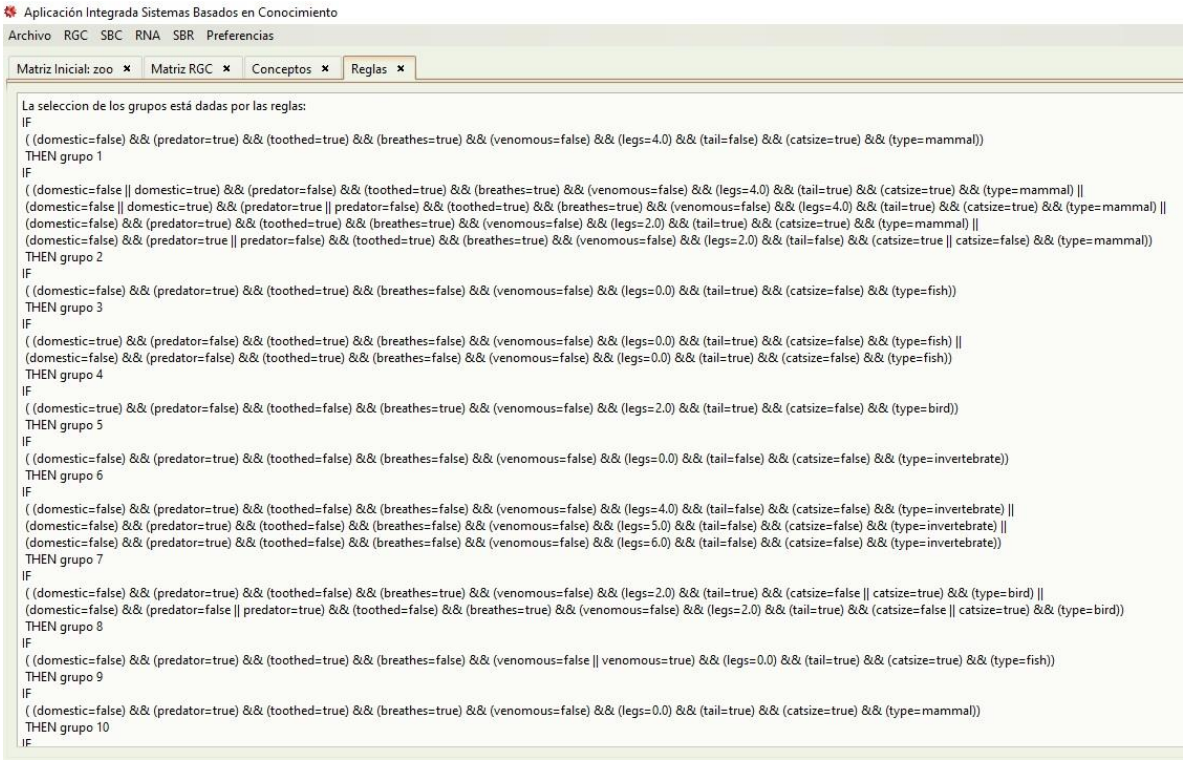


Figura 22 *Ejemplo de las reglas obtenidas. Fuente (Elaboración Propia)*

Como se muestra en la figura 23, obtenidas las reglas se procede al clasificar el nuevo objeto, primero introduciendo los rasgos correspondientes al mismo para luego agruparlo según las reglas determinadas anteriormente.

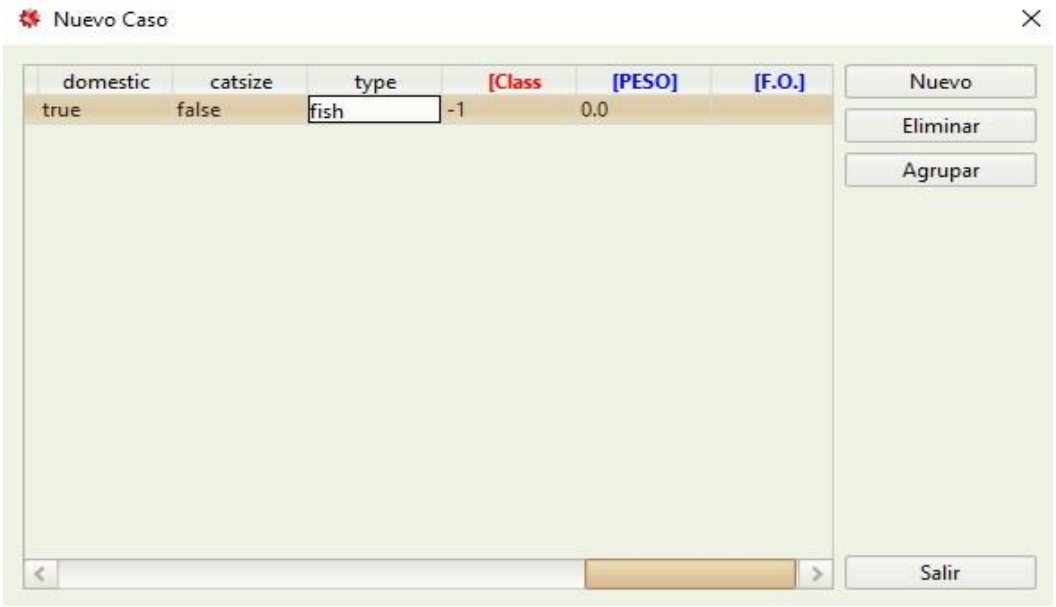


Figura 23 *Introducción del nuevo objeto. Fuente (Elaboración Propia)*

Inicialmente como el nuevo objeto no se encuentra clasificado, su valor en el número de clase al que pertenece es -1, pero luego de realizar el agrupamiento, se asigna al grupo devolviendo así el numero de la clase que recibió. Ver figura 24.

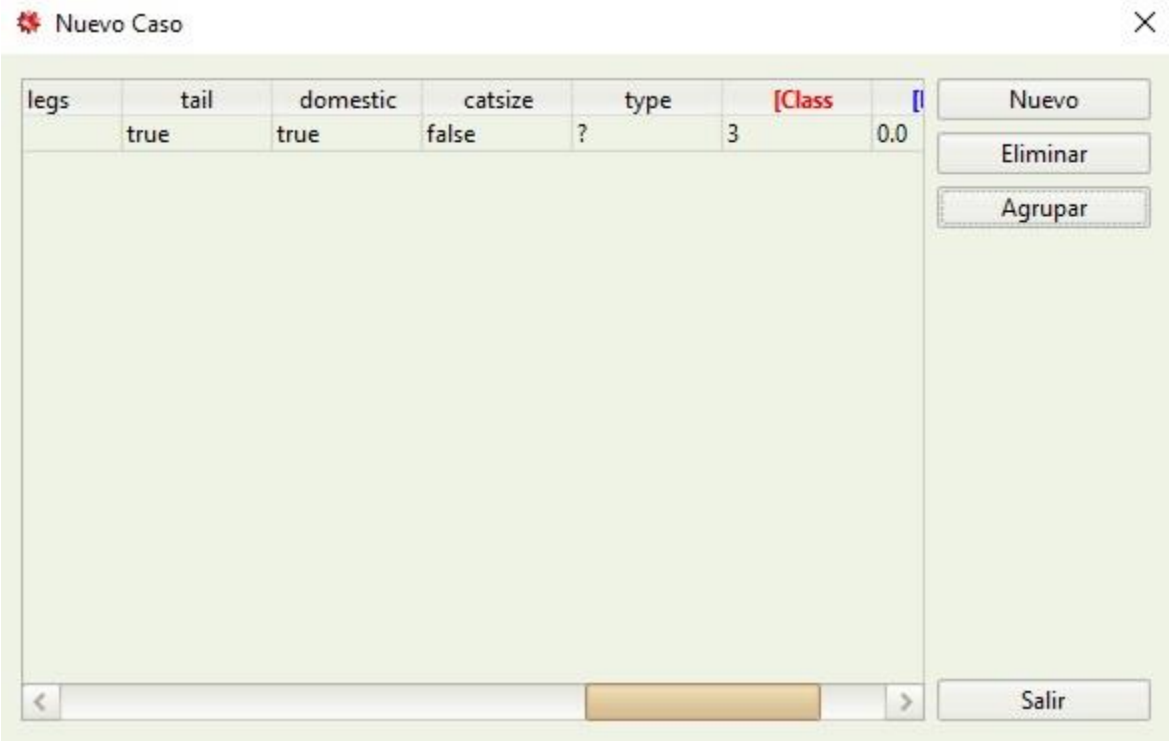


Figura 24 *Objeto clasificado. Fuente (Elaboración Propia)*

2.3 Conclusiones parciales

Como resultado de aplicar el algoritmo RGC en el diseño de la arquitectura del motor de regla se obtienen las siguientes conclusiones:

- Con la aplicación de los testores típicos se lograron seleccionar las variables de entrada, reduciendo la dimensión de los rasgos iniciales presentes en el problema.
- El cálculo de los conceptos permite tratarlo como una base de conocimiento para generar las reglas para el Sistema Basados en Reglas.

Capítulo 3

En este capítulo se realiza una descripción de las bases de casos utilizadas y se describen cada uno de los preexperimentos a realizar. Además, se muestran los resultados de aplicar el algoritmo conceptual RGC en un Sistema Basado en Reglas. Se aplica el método de validación cruzada para separar los conjuntos en entrenamiento y prueba, además se aplica el test estadístico de Friedman como prueba no paramétrica.

3.1 Descripción de los preexperimentos

Para el análisis del modelo propuesto se utilizaron 10 bases de datos de reconocimiento internacional disponibles en el repositorio para aprendizaje automatizado de la Universidad de Irvine, California (Merz y Murphy 1998). En la selección se consideraron conjuntos de datos con variadas características como: la presencia de rasgos numéricos y no numéricos, la ausencia de información, y diversos en cuanto a la cantidad de rasgos y de objetos. Para medir la validez de la propuesta de solución se establece como criterio la eficacia en la clasificación (aumento de la cantidad de patrones clasificados correctamente). Ver Anexo I.

Se compararon tres modelos de sistemas basado en reglas: el primero donde se generan las reglas a través del motor de reglas con una base de conocimiento sin aplicar ningún tratamiento previo, se le denomina SBR1. El segundo se generan las reglas después de aplicar el algoritmo conceptual utilizando el motor de reglas propuesto y tendrá el termino SBR-LC. Por último, el tercero es se realiza la generación de reglas después de aplicar el algoritmo conceptual RGC a través del motor de reglas y tendrá la nomenclatura SBR-RGC, esta última utiliza el modelo propuesto en la investigación.

Para la prueba del modelo propuesto se aplicó el método de validación cruzada (*k-fold cross validation*). Se realizan 10 iteraciones siguiendo el principio del método para un k igual 10 como se recomiendan (Yu et al. 2005), (Nematzadeh, Ibrahim y Selamat 2015) y (Pérez Planells et al. 2015). Al obtener estos resultados se utiliza el paquete del software RStudio llamado “scmamp” (*Statistical Comparison of Multiple Algorithms in Multiple Problems*) descrito en (Calvo y Santafé Rodrigo 2016) para realizar pruebas estadísticas.

Para la definición de los preexperimentos se propone realizar la prueba no paramétrica de Friedman. Este *test* no asume distribución normal ni homogeneidad en las varianzas de los datos (Demšar 2006). En particular en esta investigación se utiliza la prueba de Iman y

Davenport como una extensión de la prueba de Friedman según lo sugerido en García et al. (2010).

Una vez cargada una base de conocimiento es necesario establecer un criterio de agrupamiento para cada rasgo, un umbral de semejanza, una función de semejanza, una función objetivo y la forma de calcular los testores típicos; se procede a la construcción de los conceptos de la base de conocimiento y por último se procede a realizar la generación de las reglas como se describe en el capítulo anterior. La validación se divide en tres preexperimentos:

1. Determinar si existen diferencias significativas entre la clase de los nuevos objetos respecto a la variable por ciento de clasificación correcta para los conjuntos de datos seleccionados supervisados utilizando una los modelos SBR1 y SBR-RGC.
2. Determinar si existen diferencias significativas entre la clase de los nuevos objetos respecto a la variable por ciento de clasificación correcta para los conjuntos de datos seleccionados no supervisados utilizando una los modelos SBR1 y SBR-RGC.
3. Determinar si existen diferencias significativas entre la clase de los nuevos objetos respecto a la variable por ciento de clasificación correcta para los conjuntos de datos seleccionados supervisados y no supervisados en comparación con algoritmos como el SBR-LC, ID3, y SBR-RGC.

3.2 Descripción de las bases de datos utilizadas

Para los preexperimentos se escogieron 10 bases de datos, 5 supervisadas y 5 no supervisadas con diferentes características en cuanto a cantidad de rasgos. La selección se realiza en función de analizar los resultados del diseño propuesto en problemas con diferentes dominios en las variables de entrada, como se muestra en las tablas 3 y 4.

Tabla 2 Bases de datos supervisadas.

Base de Datos	Cantidad de objetos	Cantidad de rasgos	Cantidad de clases
zoo	101	18	7
wine	178	14	3
glass	210	10	7
ecoli	336	8	8
lymphography	148	19	5

(Fuente: Elaboración propia)

Tabla 3 Bases de datos no supervisadas.

Base de Datos	Cantidad de objetos	Cantidad de rasgos
hayes-roth_test	28	4
Cmc	1473	10
Haberman	306	4
shuttle-landing-control	15	7
Autos	205	26

(Fuente: Elaboración propia)

3.3 Eficacia de en el modelo del sistema basado en reglas según la cantidad de objetos clasificados correctamente

El análisis de la eficacia se realiza en función de la cantidad de patrones clasificados correctamente por el modelo propuesto con respecto al modelo sin aplicar el agrupamiento 3de (Merz y Murphy 1998) que cuenta con 177 objetos descritos por 13 rasgos y clasificados en tres tipos de clases.

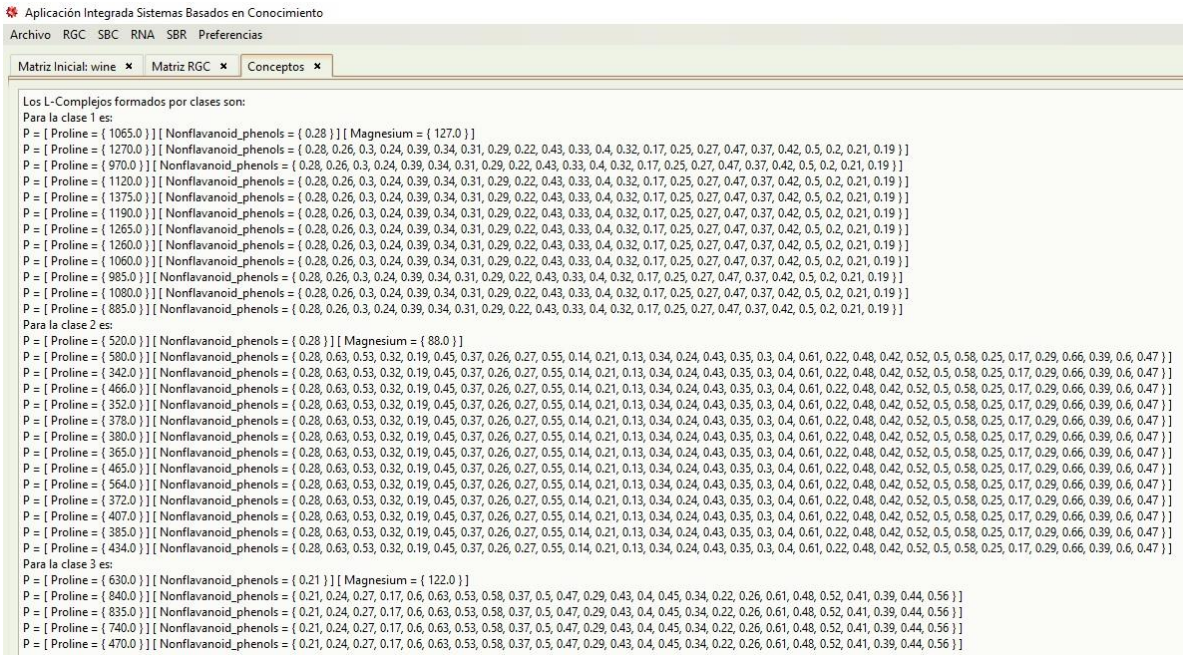


Figura 25 *Grupos obtenidos en wine de forma supervisada. Fuente (Elaboración Propia)*

3.3.1 Preexperimento 1

Se realizó el preexperimento 1 que se basa en la comparación de la eficacia del Sistema Basado en Regla sin aplicar el agrupamiento conceptual (SBR1) y el Sistema aplicando el algoritmo RGC (SBR-RGC) para bases de datos supervisadas. Se evidencia en la tabla 5 y figura 26 que el comportamiento de la eficacia en la clasificación de los objetos, se comporta mejor en el SBR-RGC en el 100% de los casos en las bases de datos utilizadas.

Tabla 5 Datos del preexperimento 1 para bases de datos no supervisadas

Base de Datos	Eficacia SBR 1 (%)	Eficacia SBR-RGC (%)
hayes-roth_test	95.02	100
cmc	98.51	98.99
haberman	90.23	93.33
shuttle-landing-control	96.34	100
autos	98.32	100

Fuente: Elaboración propia

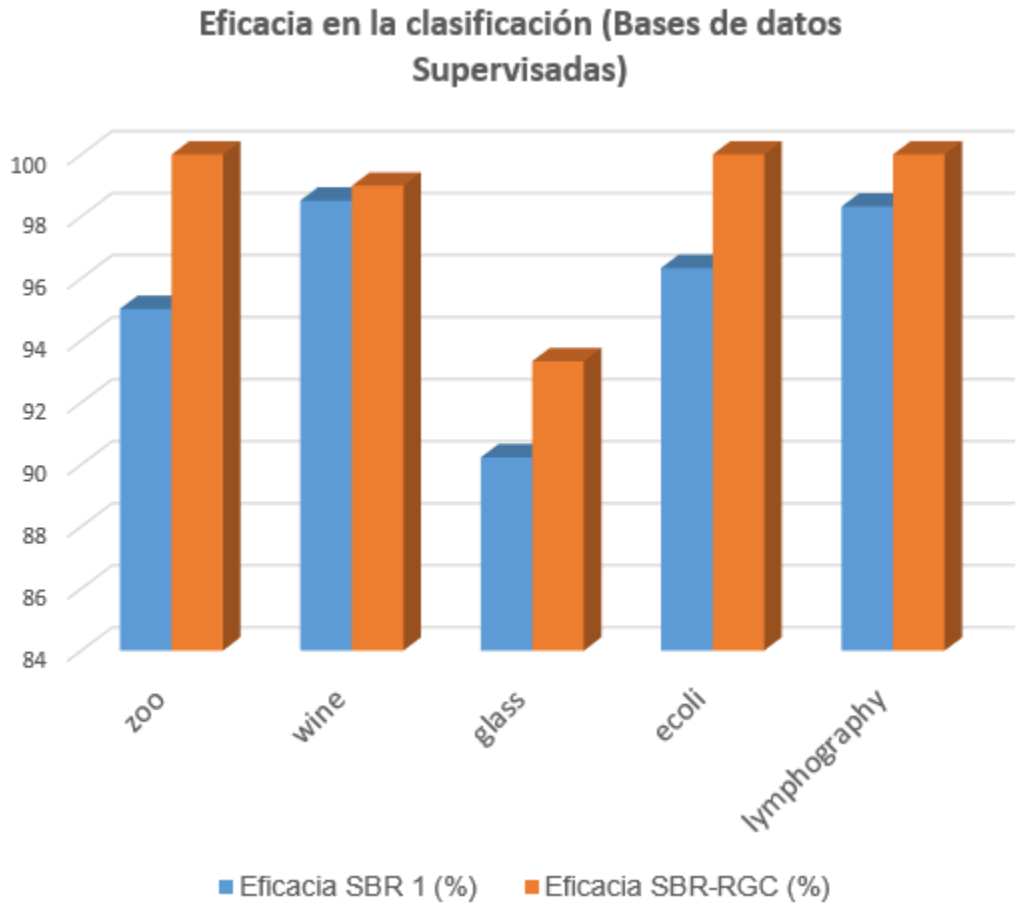


Figura 26 *Resultados obtenidos en la clasificación en cuanto a eficacia. Fuente (Elaboración Propia)*

3.3.2 Preexperimento 2

El segundo preexperimento se basa en la comparación de la eficacia del SBR sin aplicar el agrupamiento conceptual (SBR1) y el SBR aplicando el RGC (SBR-RGC) para bases de datos no supervisadas. Se observa en la tabla 6 y figura 27 que el comportamiento de la eficacia en la clasificación de los objetos, se comporta mejor en el SBR-RGC en el 80% de las bases de datos de prueba.

Tabla 4 Datos del preexperimento 2 para bases de datos no supervisadas

Base de Datos	Eficacia SBR 1 %	Eficacia SBR-RGC %

hayes-roth_test	92.50	97.33
Cmc	78.33	92.14
Haberman	89.50	88.60
shuttle-landing-control	85.20	99.11
Autos	98.24	99.45

(Fuente: Elaboración propia)

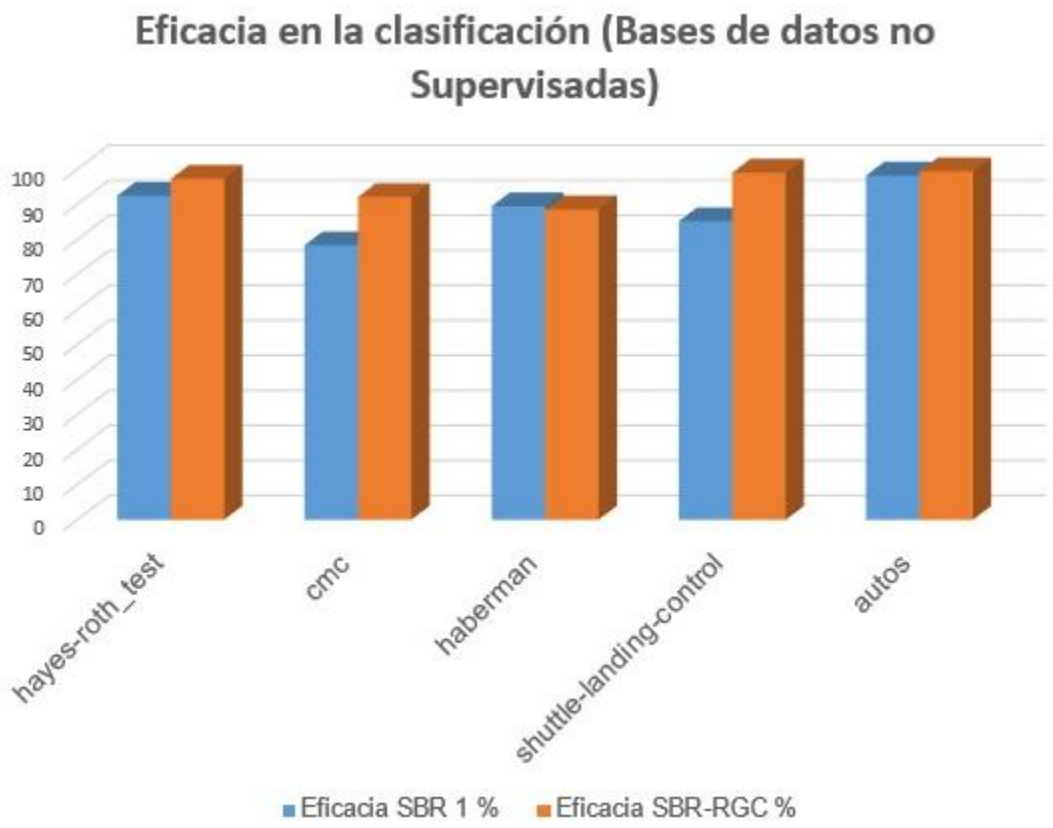


Figura 27 *Resultados obtenidos en la clasificación en cuanto a eficacia. Fuente (Elaboración Propis)*

3.3.3 Preexperimento 3

El tercero preexperimento se basa en la comparación de la eficacia del SBR aplicando el LC-Conceptual (SBR-LC), RGC (SBR-RGC); también comparado con los generadores de reglas ID3, C4.5 y PART para todas las bases de datos. Se observa en la tabla 6 y figura 28 que el comportamiento de la eficacia en la clasificación de los objetos, se comporta mejor

en el SBR-RGC en el 80% de las bases de datos de prueba. Los datos de los algoritmos ID3, C4.5 y PART se realizaron en la herramienta informática Weka.

Tabla 6

Base de Datos	C4.5	ID3	PART	Eficacia SBR-LC %	Eficacia SBR-RGC %
hayes-roth_test	76.63	75.98	75.53	96.48	97.33
cmc	84.98	87.6	87.6	83.5	92.14
haberman	81.5	81.5	82.03	90.05	88.6
shuttle-landing-control	79.68	79.53	79.46	83.3	99.11
autos	90.3	90.3	90.3	98.94	99.45
zoo	93.96	93.7	93.03	99.02	100
wine	85.79	85.79	86.36	98.91	98.99
glass	91.46	90.99	90.99	95.23	93.33
ecoli	84.89	84.8	84.03	99.48	100
lymphography	94.69	93.95	95.68	99.92	100

(Fuente: Elaboración propia)

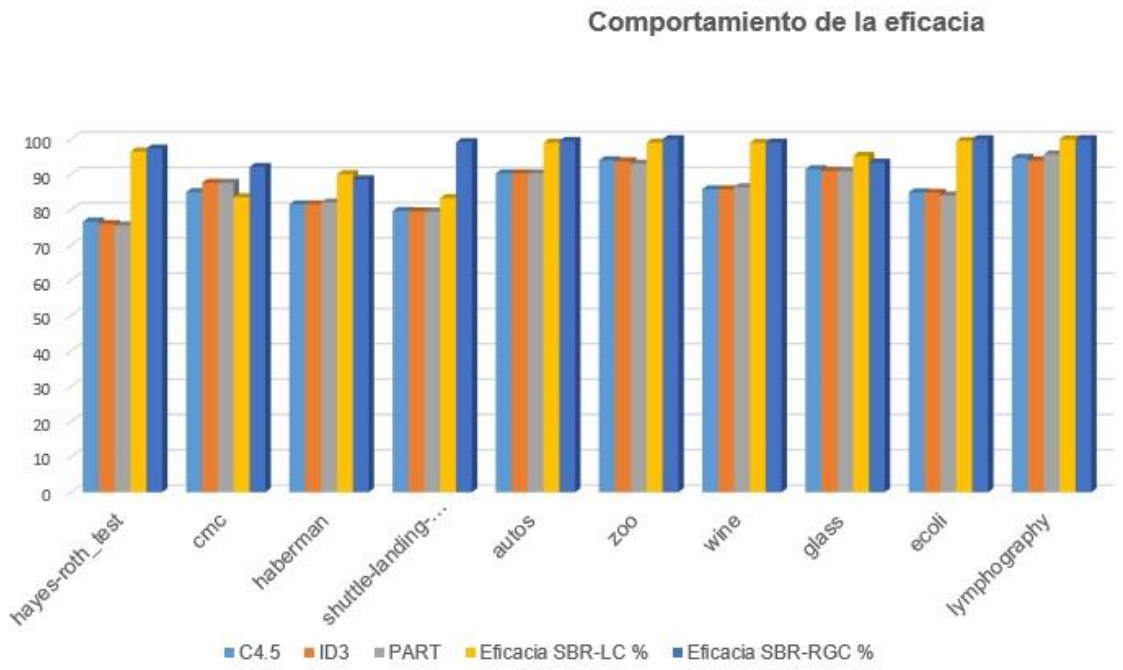


Figura 28 *Comportamiento de la eficacia. Fuente (Elaboración Propia)*

Se aplica la prueba de Friedman para la variable por ciento de clasificaciones correctas, con un intervalo de confianza de 95%. La hipótesis nula consiste en asumir que todos los algoritmos son estadísticamente equivalentes. La prueba se realiza con un p-value = $1.51486e^{-5}$ como este valor está por debajo de 0.05 se debe rechazar la hipótesis nula y por tanto se puede afirmar que existen diferencias significativas entre los algoritmos aplicados. Para detectarlas se aplica la prueba post hoc con la corrección de Finner (García et al. 2010), que arroja como resultado que el modelo propuesto en esta investigación es estadísticamente significativo en comparación con los algoritmos anteriormente plateados. Ver anexo 58.

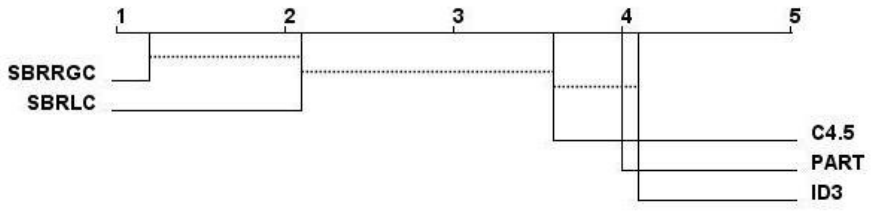


Figura 29 *Grafico de diferencia significativa. Fuente (Elaboración Propia)*

3.4 Conclusiones parciales

El desempeño de la variable eficacia, se comporta mejor en bases de datos supervisadas que, en bases de datos no supervisadas.

Con la aplicación de los preexperimentos a las bases de datos de prueba se puede afirmar que la utilización del Motor de Reglas aplicando el algoritmo conceptual RGC garantiza una mayor eficacia en la clasificación, con respecto a los sistemas planteados SBR1 y SBR-LC y los algoritmos ID3, C4.5 y PART.

Se aprecia una diferencia significativa entre los sistemas SBR antes mencionados y los algoritmos respectivamente en las bases de datos utilizadas.

Conclusiones

A partir de la sistematización de los principales referentes teóricos que sustentan la investigación se evidencia que aplicar un motor de reglas ayuda en la organización, ejecución y gestión de las reglas y que el algoritmo conceptual RGC mejora la clasificación por su capacidad de caracterizante, por lo que se evidencia la necesidad de utilizar dicho algoritmo en un motor de reglas.

Con el análisis y desarrollo del diseño teórico se logró una descripción legible de las técnicas y algoritmos utilizados para comprender mejor su funcionamiento y facilitar su implementación. Se definió como trabajar con las bases de conocimiento clasificadas y no clasificadas y cuáles eran sus métodos para lograrlo además de cómo se obtendrías las reglas para implementar el Motor de Reglas, con el fin de lograr un modelo computacional de fácil entendimiento.

Los métodos científicos empleados para la validación del modelo computacional permitieron comprobar que la solución propuesta contribuye a mejorar la eficacia en la clasificación de objetos con respecto a los otros algoritmos analizados en la investigación.

Recomendaciones

Una vez concluida la investigación, la implementación y las pruebas correspondientes al diseño del Sistema Basado en Reglas aplicando el algoritmo conceptual RGC del reconocimiento lógico combinatorio de patrones se recomienda:

- Se aborde la variable eficiencia en la clasificación para el diseño del Sistema Basado en Reglas utilizando algoritmo conceptual RGC.
- Implementar al algoritmo RGC otros algoritmos de selección de rasgos.

Referencias Bibliográficas

ALMEIDA, Y. y STALIN, J., 2018. Sistema experto web y móvil para la determinación de rutinas y dietas en el gimnasio fuerza extrema utilizando la herramienta libre clips. S.l.: s.n.

ARIAS, F.J., MARULANDA, J.A., CADAVID, J.M. y CARRANZA, D.A.O., 2006. Diseño y Desarrollo de Mecanismos de Razonamiento Multi-Agente para la Negociación de Energía Eléctrica Utilizando JESS Y JADE. RASI, vol. 3, no. 1, pp. 51-56.

BADARÓ, S., IBAÑEZ, L.J. y AGÜERO, M.J., 2013. Sistemas expertos: fundamentos, metodologías y aplicaciones. Ciencia y tecnología, no. 13, pp. 349-364.

BAÑEZ, R., PRINSE, D. y TORRES UTRILLA, J.D., 2016. Aplicación del paradigma de reglas de negocio al proceso de personalización de las aplicaciones en SAP, en los módulos: gestión de Materiales y Finanzas de una Empresa. ,

BAZÁN, P., GIANDINI, R. y DÍAZ, F., 2010. Tecnologías para implementar un marco integrador de SOA y BPM. LINTI (Laboratorio de Investigación en Nuevas Tecnologías Informáticas), Facultad de Informática Universidad,

COLES, A.J., COLES, A.I., FOX, M. y LONG, D., 2010. Forward-chaining partial-order planning. Twentieth International Conference on Automated Planning and Scheduling. S.l.: s.n.,

CORDERO, J.G., CORDERO, F.G., GUZMÁN, E. y MUÑOZ, R.C., 2007. Un sistema inteligente para el aprendizaje de fundamentos de programación orientada a objetos. CAEPIA-TTIA 2007: actas. S.l.: s.n., pp. 329-338. ISBN 84-611-8846-2.

DEL SUR, P.T., 2008. DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE ADMINISTRACIÓN DE REGLAS DE NEGOCIO. S.l.: Universidad Austral de Chile.

GARCÍA, S., FERNÁNDEZ, A., LUENGO, J. y HERRERA, F., 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, vol. 180, no. 10, pp. 2044–2064. ISSN 0020-0255.

GONZÁLEZ, J.C.G. y AMPUERO, M.A., 2015. Implementación del enfoque de reglas de negocio utilizando motores de reglas en el desarrollo de aplicaciones Java. Ciencias de la Información, vol. 46.

GUTIÉRREZ, J.M., 2008. Sistemas expertos basados en reglas. Recurso personal. Dpto. de Matemática Aplicada. Universidad de Cantabria, pp. 1-12.

HERRERÍAS SANTOS, J.M., 2018. Diseño y desarrollo de un de juego de estrategia de cartas coleccionables (TCG) con inteligencia artificial. ,

HITPASS, B., 2017. BPM: Business Process Management: Fundamentos y Conceptos de Implementación 4a Edición actualizada y ampliada. S.l.: Dr. Bernhard Hitpass. ISBN 956-345-977-6.

IBARRA, G.M. y BAZÁN, P., 2013. Análisis y comparación de plataformas BRMS a través de una prueba de concepto. XV Workshop de Investigadores en Ciencias de la Computación. S.l.: s.n.,

GÁLVEZ LIO, D.D., 1998. *Sistemas Basados en el Conocimiento*. Universidad Central «Martha Abreu» de Las Villas 1998: s.n.

JIMÉNEZ BUILES, J.A., 2006. Un modelo de planificación instruccional usando razonamiento basado en casos en sistemas multi-agente para entornos integrados de sistemas tutoriales inteligentes y ambientes colaborativos de aprendizaje. S.l.: Universidad Nacional de Colombia, Sede Medellín.

BADARÓ, S., IBAÑEZ, L.J. y AGÜERO, M.J., 2013. Sistemas expertos: fundamentos, metodologías y aplicaciones. *Ciencia y tecnología*, no. 13, pp. 349-364.

CÓRDOVA, P. y JOSÉ, R., 2018. *Evaluación del rendimiento de la transferencia de imágenes entre los lenguajes de programación Java y Python.* S.l.: Escuela Superior Politécnica de Chimborazo.

Información sobre Java 8. [en línea], 2019. [Consulta: 8 junio 2019]. Disponible en: https://www.java.com/es/download/faq/java8.xml.

NetBeans IDE - Overview. [en línea], 2019. [Consulta: 8 junio 2019]. Disponible en: https://netbeans.org/features/index.html.

NILSSON, N.J., 2014. *Principles of artificial intelligence*. S.l.: Morgan Kaufmann. ISBN 1-4832-9586-9.

PULIDO, K.R., ROMERO, M.S. y MENDOZA, J.E., 2019. Lenguajes de Programación, 2019-2. ,

TEAM, Rs., 2015. RStudio: integrated development for R. *RStudio, Inc., Boston, MA URL http://www. rstudio. com*, vol. 42, pp. 14.

JURADO, F. y ASBUSAC, J., 2015. *Desarrollo de Videojuegos: Desarrollo de Componentes*. S.l.: Cursos en Español.

MARSHALL, J.S. y THOMSON, M.A.E.P.J.C., 2015. *The Pandora software development kit for pattern recognition*. S.l.: s.n.

MARTÍNEZ-TRINIDAD, J.. y RUIZ-SHULCLOPER, J., 1999. Algoritmo LC Conceptual Duro. IV Simposio Iberoamericano de Reconocimiento dePatrones. La Habana, Cuba: s.n.,

MARTÍNEZ-TRINIDAD, J.. y RUIZ-SHULCLOPER, J., 1999. Algoritmo LC Conceptual Duro. *IV Simposio Iberoamericano de Reconocimiento de Patrones*. La Habana, Cuba: s.n.,

MARTÍNEZ-TRINIDAD, J.F. y SÁNCHEZ-DÍAZ, G., 2001. LC: a conceptual clustering algorithm. *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. S.l.: Springer, pp. 117–127.

MENDOZA, G., FERNANDA, J. y ZAPATA-SARZOSA, M.B., 2018. *Desarrollo de una red neuronal para la clasificación y reconocimiento del lenguaje de signos ecuatoriano.* S.l.: Universidad de las Fuerzas Armadas ESPE. Extensión Latacunga. Carrera de Ingeniería en Electrónica e Instrumentación.

MERZ, C.J. y MURPHY, P.M., 1998. UCI repository of machine learning datasets. , pp. 92697–3425.

MICHALSKI, R., 1980. Knowledge acquisition through conceptual clustering: A theoretical framework and algorithm for partitioning data into conjunctive concepts. *International Journal of Policy Analysis and Information Systems*, vol. 4, pp. 219-243.

MICHALSKI, R.S. y STEPP, R.E., 1981. *Concept-based clustering versus numerical taxonomy*. S.l.: Department of Computer Science, University of Illinois at Urbana-Champaign.

MONTERO-MONTES, G., 2012. *Desarrollo de un simulador de tráfico ferroviario sobre cartografía descargada dinámicamente*. S.l.: s.n.

MONTERO-MONTES, G., 2012. Desarrollo de un simulador de tráfico ferroviario sobre cartografía descargada dinámicamente. S.l.: s.n.

NAVARRO SÁNCHEZ, Y., 2011. Sistema basado en conocimiento para el diagnóstico de los riesgos asociados con el uso de sustancias químicas. S.l.: Universidad Central “Marta Abreu” de Las Villas.

PÉREZ-SUÁREZ, A. y MEDINA-PAGOLA, J.E., 2014. Algoritmos para el agrupamiento conceptual de objetos. ,

PÉREZ-VALLS, J., 2013. HERRAMIENTA MATLAB PARA LA SELECCIÓN DE ENTRADAS Y PREDICCIÓN NEURONAL DE VALORES DE BOLSA [en línea]. Tesis. S.l.: Universidad de Sevilla. Disponible en: http://bibing.us.es/proyectos/abreproy/12166/fichero/Volumen+1+-+Memoria+descriptiva+del+proyecto%252F3+-+Perceptron+multicapa.pdf.

PONS-PORRATA, A., 2004. DESARROLLO DE ALGORITMOS PARA LA ESTRUCTURACIÓN DINÁMICA DE INFORMACIÓN Y SU APLICACIÓN A LA DETECCIÓN DE SUCESOS. Trabajo de Tesis en opción al grado científico de Doctor en Ciencias Técnicas. S.l.: s.n.

PÉREZ-SUÁREZ, A. y MEDINA-PAGOLA, J.E., 2014. Algoritmos para el agrupamiento conceptual de objetos. ,

PÉREZ-VALLS, J., 2013. *HERRAMIENTA MATLAB PARA LA SELECCIÓN DE ENTRADAS Y PREDICCIÓN NEURONAL DE VALORES DE BOLSA* [en línea]. Tesis. S.l.: Universidad de Sevilla. Disponible en: http://bibing.us.es/proyectos/abreproy/12166/fichero/Volumen+1+-+Memoria+descriptiva+del+proyecto%252F3+-+Perceptron+multicapa.pdf.

PONS-PORRATA, A., 2004. *DESARROLLO DE ALGORITMOS PARA LA ESTRUCTURACIÓN DINÁMICA DE INFORMACIÓN Y SU APLICACIÓN A LA DETECCIÓN DE SUCESOS.* Trabajo de Tesis en opción al grado científico de Doctor en Ciencias Técnicas. S.l.: s.n.

RABUÑAL, J.R., 2002. Metodología para el desarrollo de sistemas de extracción de concimiento en RNA. ,

REYES-GONZÁLEZ, Y., 2014. *Modelo para la adaptación de las soluciones en un Sistema Basado en Casos utilizando el agrupamiento conceptual*. S.l.: Tesis de Maestría.

REYES-GONZÁLEZ, Y., 2017. *Modelo basado en casos utilizando algoritmos conceptuales del Reconocimiento Lógico Combinatorio de Patrones*. Tesis en opción al grado científico de Doctor en Ciencias Técnicas. La Habana: Universidad de las Ciencias Informáticas.

REYES-GONZÁLEZ, Y., ARCEO, A.C., MARTÍNEZ-SÁNCHEZ, N. y HERNÁNDEZ-DOMINGUEZ, A., 2016. Agrupamiento conceptual lógico combinatorio: Una alternativa para la toma de decisiones. *INTELIGENCIA ARTIFICIAL*, vol. 19, pp. 82-96.

REYES-GONZÁLEZ, Y., RODRÍGUEZ-VALLEJO, L., MARTÍNEZ-SÁNCHEZ, N. y YERO-OSES, E.A., 2016. Métricas para la validación de los conceptos en el Reconocimiento Lógico Combinatorio de Patrones. ,

RUIZ SHULCLOPER, J., 2013. Acerca del surgimiento del Reconocimiento de Patrones en Cuba. , vol. 7, no. 2, pp. 169–192.

RUIZ-SHULCLOPER, J., 2009. Reconocimiento lógico combinatorio de patrones: teoría y aplicaciones (Tesis en opción al grado científico de Doctor en Ciencias). ,

RUÍZ-SHULCLOPER, J., ALBA, E. y LAZO, M., 1994. Introducción a la teoría de testores. *Serie Verde Cinvestav-IPN*, no. 50.

REYES-GONZÁLEZ, Y., 2014. Modelo para la adaptación de las soluciones en un Sistema Basado en Casos utilizando el agrupamiento conceptual. S.l.: Tesis de Maestría.

REYES-GONZÁLEZ, Y., 2017. Modelo basado en casos utilizando algoritmos conceptuales del Reconocimiento Lógico Combinatorio de Patrones. Tesis en opción al grado científico de Doctor en Ciencias Técnicas. La Habana: Universidad de las Ciencias Informáticas.

RUIZ-SHULCLOPER, J., 2009. Reconocimiento lógico combinatorio de patrones: teoría y aplicaciones (Tesis en opción al grado científico de Doctor en Ciencias). ,

SALAZAR ACOSTA, I.N., 2012. UCShell versión 2.1: un ambiente para la construcción de sistemas expertos. S.l.: Universidad Central “Marta Abreu” de Las Villas.

SANTANA, A. y NIEVES-HERNÁNDES, C., 2018. *Presentación del Curso: El entorno estadístico R (R4ULPGC)* [en línea]. Gran Canaria: Universidad de las Palmas. [Consulta: 2 junio 2018]. Disponible en: http://www.dma.ulpgc.es/profesores/personal/stat/cursoR4ULPGC/1-presentacion.html.

SARDIÑAS, A.D. y PÉREZ, R.B., 2001. Generalización del uso de prototipos para el diseño de redes neuronales tipo MLP con una capa oculta. *Revista Facultad de Ingeniería*, no. 22, pp. 126-133. ISSN 2422-2844.

SHULCLOPER, J.R., ALBA-CABRERA, E., LAZO-CORTÉS, M.S. y GRUPO DE RECONOCIMIENTO DE PTRONES CUBA-MÉXICO, 1995. Introducción al Reconocimiento de Patrones. Grupo de Reconocimiento de Patrones Cuba-México. , ISSN CINVESTAV-IPN.

TALUKDAR, J. y MEHTA, B., 2017. Human Action Recognition System using Good Features and Multilayer Perceptron Network. *arXiv preprint arXiv:1708.06794*,

TIPAN, C. y FERNANDO, M., 2018. *Ubicación de estabilizadores de potencia y el control realimentación de estados para amortiguar oscilaciones electromecánicas de baja frecuencia utilizando redes neuronales*. S.l.: Quito, 2018.

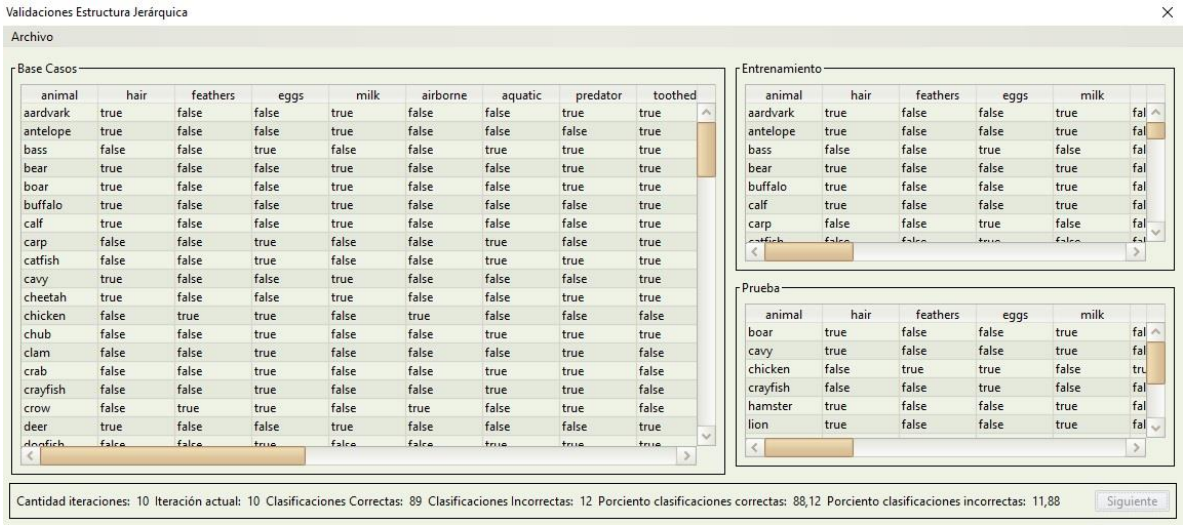
VARGAS-BEJARANO, C.I., 2017. Evaluación del desempeño sísmico de puentes continuos. ,

YERO-OSES, E.A., REYES-GONZÁLEZ, Y. y MARTÍNEZ-SANCHEZ, N., 2016. *CEPAR:Un sistema herramienta de apoyo al docente del reconocimiento lógico combinatorio de patrones.* S.l.: s.n.

Anexos

Anexo I

Validación de los objetos clasificados con *k* =10.



Anexo II

Resultados obtenidos por el test de Friedman.

R-Studio

```
      dataLong.value  r
C4.5               24 10
ID3                19 10
PART               20 10
SBRLC              39 10
SBRRGC             48 10

Friedman's Test
===============
Adjusted for ties
Critical Value: 27.58333
P.Value Chisq: 1.51486e-05
F Value: 19.99329
P.Value F: 9.605507e-09

Post Hoc Analysis

Comparison between treatments
Sum of the ranks

               difference pvalue signif.    LCL    UCL
C4.5 - ID3              5 0.2271          -3.25  13.25
C4.5 - PART             4 0.3321          -4.25  12.25
C4.5 - SBRLC          -15 0.0007     *** -23.25  -6.75
C4.5 - SBRRGC         -24 0.0000     *** -32.25 -15.75
ID3 - PART             -1 0.8073          -9.25   7.25
ID3 - SBRLC           -20 0.0000     *** -28.25 -11.75
ID3 - SBRRGC          -29 0.0000     *** -37.25 -20.75
PART - SBRLC          -19 0.0000     *** -27.25 -10.75
PART - SBRRGC         -28 0.0000     *** -36.25 -19.75
SBRLC - SBRRGC         -9 0.0334       * -17.25  -0.75
```