TESIS DE MAESTRÍA

Ingeniería en Sistemas de información

Título:

“Estudio de pertinencia de algoritmos en procesos de descubrimiento de reglas de pertenencia a grupos”

Autor: Gabriel Ciciliani

Director de Tesis: Dr. Hernán Merlino - Dr(C). Sebastian Martins

Buenos Aires - 2019

DEDICATORIA

A mi esposa Angeles, por su apoyo incondicional

A mis hijos Mateo y Trini

A mis padres y hermanos

A mis amigos

A la memoria de Ramón

AGRADECIMIENTOS

A Sebastián, por brindarme su conocimiento, consejo y apoyo

A Hernán y a Sebastián, por tomarme como tesista aún en tiempos complicados para ellos

A José Panchuk, por su desinteresada colaboración

A la Facultad Regional Buenos Aires y a la Escuela de Posgrado de la Universidad Tecnológica Nacional

Resumen

En el campo de la Ingeniería de explotación de información, el proceso de descubrimiento de reglas de pertenencia a grupos se caracteriza por la utilización combinada de un proceso de descubrimiento de grupos (clustering) y uno de inducción de reglas. Dada la variedad de algoritmos de clustering e inducción de reglas disponibles en la actualidad, es de interés poder conocer a priori que pareja de algoritmos es más conveniente para un set de datos, en a base sus características. En esta tesis, se propone un proceso que permite validar el rendimiento de los algoritmos, en base a métricas internas, para distintos tipos de sets de datos, con características específicas, de forma tal que permita comprender bajo que características cada pareja de algoritmos ofrece mejor rendimiento.

Palabras clave

Minería de datos, clustering, inducción de reglas, descubrimiento de reglas de pertenencia a grupos, estudio de algoritmos

Abstract

In the data mining field, the group membership rules discovery process consists of utilizing a group discovery (clustering) and rules induction process combined. Due to the broad variety of clustering and rules induction algorithms currently available, it is considered of interest to know beforehand which pair of algorithms is more convenient for a given dataset, based just on its properties. In this thesis, a process that allows to validate algorithms performance, based on internal metrics, for different datasets, with specific characteristics, is proposed, so that it allows to understand under which characteristics, each algorithm pair offers better performance.

Keywords

Data mining, clustering, rules induction, membership rules discovery, algorithms study

INDICE DE TABLAS

INDICE DE FIGURAS

Tabla de contenidos

1. Introducción

En esta sección se presenta el contexto de esta tesis, presentando brevemente la situación disciplinar que dio origen al presente trabajo de investigación (sección 1.1), los objetivos definidos para la tesis (sección 1.2) y la metodología utilizada para el desarrollo del trabajo (sección 1.3). En la sección 1.4, se presenta producción científica vinculada con el desarrollo de la tesis. Finalmente, se presenta una descripción de la estructura de la tesis (sección 1.5).

1.1. Contexto

La Explotación de Información es una sub-disciplina de los Sistemas de Información, que brinda a la Inteligencia de Negocio las herramientas para la transformación de información en conocimiento (García-Martínez et al., 2015) y se define como la búsqueda de patrones interesantes y de reglas importantes, previamente desconocidas, en grandes cantidades de información almacenada en distintos medios (Martins, 2016). Un proceso de Explotación de Información consiste en un grupo de tareas relacionadas que se realizan con el objetivo de obtener información útil y significativa a partir de grandes cantidades de datos. Para esto, dichos procesos se valen de la utilización de algoritmos de Minería de Datos (García-Martínez et al., 2015; Martins, 2016).

Un procedimiento recurrente en la explotación de información es el proceso de descubrimiento de reglas de pertenencia a grupos (García-Martínez et al., 2015) consiste en tomar el conjunto de datos a estudiar y aplicar un algoritmo de agrupamiento o "clustering" para separarlo en distintos grupos (también llamados clases o clusters) y aplicar luego algoritmos de inducción de reglas para explicitar las características que definen la pertenencia a cada grupo descubierto (Kaski, 1997; Hall y Holmes, 2003).

Tanto para el procedimiento de clustering como para la de inducción de reglas existen varios algoritmos (Xu y Tian 2015; Sehgal y Garg 2014; Panchuk, 2015). Esto se explica por la necesidad de encontrar distintos tipos de patrones que expliquen el comportamiento de los datos. En relevante identificar en la diversidad de dominios de negocio, los algoritmos de explotación de información que mejor identifican los patrones ocultos en dicha masa de datos.

Numerosos trabajos demuestran que la performance de los algoritmos varía notablemente tanto con las características del set de datos a analizar como por los valores de los ejemplos de dicho set de datos (Kogan, 2007; López-Nocera, 2012; Panchuk, 2015; Sehgal y Garg 2014; Smith, Woo, Ciesielski y Ibrahim 2002). Podemos decir entonces que uno de los desafíos actuales de la ingeniería de explotación de información es encontrar los algoritmos que mejor describen el set de datos a analizar, para brindar un mayor valor a las demandas del cliente, obteniendo patrones

de mayor valor y novedad para dar soporte al proceso de toma de decisiones.

Varios investigadores se han volcado al estudio de la relación entre las características de un set de datos y la performance de algoritmos relevantes en el descubrimiento de grupos (Xu y Tian 2015), inducción de reglas (Smith et al., 2002) y el descubrimiento de reglas de pertenencia a grupos (Kogan, 2007; López-Nocera, 2012; Panchuk, 2015). Sin embargo, los trabajos citados proponen una clasificación de los dominios a los cuales pertenecen los datos analizados y demuestran empíricamente que pareja de algoritmos es más eficaz para cada tipo de dominio. Dicha clasificación se basa en parámetros del mismo y del set de datos a analizar, como la cantidad de atributos, la cantidad de clases (grupos), la cantidad de reglas por clase, etc. En este contexto, puede apreciarse que parte de las variables utilizadas para el estudio están estrechamente vinculadas con los resultados del proceso de descubrimiento de reglas de pertenencia a grupos en sí.

1.2. Objetivos

El objetivo de este trabajo consiste en definir un método que permita, en el marco de un proceso de descubrimiento de reglas de pertenencia a grupos, determinar cual es la combinación de algoritmos que mejor extrae los patrones ocultos en el set de datos y comprender las caracteristicas ante las cuales dicho modelo provee su mejor comportamiento. El método se enfocará en determinar los algoritmos apriori, con mayor rapidez y en un menor tiempo de cómputo que si se evaluaran todas las alternativas posibles.
Para alcanzar dicho objetivo general, se identifican los siguientes objetivos específicos:

1) Desarrollar una lista de parámetros y métricas objetivas para describir a un set de datos, en base a las cual se pueda tipificarlos.

2) Definir y desarrollar un ambiente experimental que permita (a) generar set de datos artificiales con características predefinidas en base a la lista de parámetros y métricas antes mencionada (b) Procesar dichos sets mediante los pares de algoritmos de interés (c) Calcular y registrar métricas de evaluación interna y el tiempo de procesamiento para cada combinación de tipo de set de datos y pareja de algoritmos posible.

3) Extraer patrones de los resultados que muestren correlación entre la calidad de los resultados obtenidos, en base a las métricas internas, el tipo de set de datos y las parejas de algoritmos utilizada para generar dichos resultados. Describir las condiciones bajo las cuales se obtienen mejores resultados con los distintos pares de algoritmos y evaluar la eficiencia del método, comparando el tiempo de cálculo de las métricas versus el de probar todas las combinaciones de algoritmos posibles.

4) Validar el método propuesto, aplicándolo a al menos tres set de datos reales

1.3. Metodología de la tesis

Para el desarrollo del proyecto se siguió un enfoque de investigación clásico [Rosas y Riveros, 1985; Creswell, 2002] con énfasis en la producción de tecnologías (Sábato y Mackenzie, 1982) identificandose a continuación los métodos necesarios para desarrollar el proyecto:

- Las revisiones sistemáticas (Argimón Pallás y Jiménez Villa 2004) de artículos científicos siguen un método explícito para resumir la información sobre determinado tema o problema. Se diferencia de las revisiones narrativas en que provienen de una pregunta estructurada y de un protocolo previamente realizado.
- El prototipado evolutivo experimental (Basili, 1993) consiste en desarrollar una solución inicial para un determinado problema, generando su refinamiento de manera evolutiva por prueba de aplicación de dicha solución a casos de estudio (problemáticas) de complejidad creciente. El proceso de refinamiento concluye al estabilizarse el prototipo en evolución.

1.4. Producción Científica

Como derivado de las actividades realizas en este proyecto, se realizó el siguiente articulo:

- Ciciliani, G., Martins, S., Merlino, H. (2018). Análisis Preliminar del Rendimiento de Algoritmos para el Procesos de Descubrimiento de Reglas de Pertenencia a Grupos. XXIV Congreso Argentino de Ciencias de la Computación. (Estado Pendiente de Aceptación)

1.5. Estructura de la Tesis

La tesis se estructura en 8 capítulos: Introducción, Estado del arte, Descripción del problema, solución propuesta, Resultados experimentales obtenidos, Conclusiones, aportes y futuras líneas de investigación y Referencias. A continuación se realiza una breve descripción de los mismos.

En el primer capítulo, Introducción, se plantea el contexto de la tesis y se mencionan los problemas abiertos identificados. Además se presentan los objetivos de la tesis y la metodología utilizada para el desarrollo del trabajo, se enumeran las producciones científicas vinculadas a esta

tesis y se realiza un breve resumen de la estructura de la misma.
En el segundo capítulo, Estado del arte, se brinda un panorama de la situación actual de cada uno de los conceptos, tecnologías y algoritmos sobre los que se construyó esta tesis.

El tercer capítulo, Descripción del problema, se detalla la problemática identificada, las oportunidades de extender estudios anteriores y se plantean las preguntas de investigación.

En el capítulo cuatro se presenta la solución propuesta en detalle: el diseño experimental y un detalle de las tareas ejecutadas en forma automática por el software creado para tal fin. También se proponen las características del set de datos a correlacionar con los algoritmos.

El capítulo quinto detalla los resultados obtenidos durante la experimentación, tanto para el proceso de agrupamiento o clustering como para el de inducción de reglas. Se presentan las tablas resumen vinculando los algoritmos con la combinación de métricas internas que los dieron por ganador y la cantidad de casos para cada combinación. Por último, se detalla el proceso de validación de los datos obtenidos mediante sets de datos reales.

El capítulo 6 presenta las conclusiones obtenidas en base a la información presentada en el capítulo anterior. Se detallan los aportes de esta tesis, las futuras líneas de trabajo y se da respuesta a las preguntas de investigación.

El capítulo 7 corresponde al anexo, donde se presentan detalles adicionales acerca de la solución software utilizada para la fase de experimentación.

Por último, en el capítulo 8 se presentan las referencias bibliográficas que dan sustento a esta tesis.

2. Estado del arte

En este capítulo se presenta el estado del arte de los conceptos que son base para los objetivos de la presente tesis. En la sección 2.1, se describe la disciplina de ingeniería de explotación de información en la cual se enmarca la tesis. En la sección 2.2, se presenta la estructura general de este tipo de proyectos (introduciéndose el modelo CRISP-DM). En la sección 2.3 se presentan los cinco procesos de explotación de información, señalandose aquel que es de interés para este proyecto. Luego, se presentan los algoritmos que conforman el proceso en análisis (sección 2.4), realizando una descripción detallada de los dos tipos de algoritmos (Clustering e Aprendizaje Inductivo) que intervienen en el proceso. Por último, en la sección 2.5 se describe la manera de evaluar ambos tipos de aprendizajes, presentando las métricas a utilizar en el proyecto.

2.1. Ingeniería de Explotación de Información

La Explotación de Información es una sub-disciplina de los Sistemas de Información, que brinda a la Inteligencia de Negocio las herramientas para la transformación de información en conocimiento (García-Martínez, Díez, García, Martins y Baldizzoni 2015) y se define como la búsqueda de patrones interesantes y de reglas importantes, previamente desconocidas, en grandes cantidades de información almacenada en distintos medios (Martins, 2013).

Con base en que la Ingeniería de Software ha sido definida en el SWEBOK (Abran et al., 2004) como: “la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software, y el estudio de estos enfoques, es decir, la aplicación de la ingeniería al software”; se conviene en definir a la Ingeniería de Explotación de Información como la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo de proyectos de explotación de información, y el estudio de este enfoque, es decir, la aplicación de la ingeniería a la explotación de información. La ingeniería de explotación de información entiende en los procesos y las metodologías utilizadas para: ordenar, controlar y gestionar la tarea de encontrar patrones de conocimiento en masas de información (García-Martínez et al. 2011).

Un proceso de Explotación de Información consiste en un grupo de tareas relacionadas que se realizan con el objetivo de obtener información útil y significativa a partir de grandes cantidades de datos. Para esto, dichos procesos se valen de la utilización de algoritmos de Minería de Datos (Data Mining) (García-Martínez et al., 2015; Martins, 2013)

Tradicionalmente, se denomina Minería de Datos (Data Mining) al conjunto de técnicas y

herramientas aplicadas al proceso de extraer y presentar conocimiento implícito, previamente desconocido, potencialmente útil y humanamente comprensible, a partir de grandes conjuntos de datos. Son sus objetivos predecir de forma automatizada tendencias y comportamientos y describir (también de forma automatizada) modelos previamente desconocidos (Michalski,1983; Holsheimer y Siebes 1991; Piatetski-Shapiro, Frawley y Matheus 1991; Chen, Han y Yu 1996; Evangelos y Han 1996; Mannila, 1997; Felgaer, Britos y García Martínez 2006; Kogan, 2007).

Sin embargo, en (Fayyad, Piatetsky-Shapiro y Smyth 1996) se señala que el término "minería de datos" está fuertemente ligado al concepto de base de datos y se remonta a la definición de algoritmos de búsqueda de patrones en grandes bases de datos, existiendo en la actualidad líneas de investigación en campos tales como: minería de textos (Tan, 1999), minería de imágenes (Hsu, Lee y Zhang 2002), minería de patrones en flujos de información (Gaber, Zaslavsky y Krishnaswamy 2010) y minería en la web (Kosala y Blockeel 2000), entre otras. En este contexto, se conviene utilizar el término "explotación de información" como referencia genérica a cualquiera de los tipos de minería precitados (Kruse y Borgelt 2003; Gopal, Marsden y Vanthienen 2011).

2.2. Modelo de Proceso

Según lo expuesto en (Cios, Kurgan 2005) y en (Rodríguez et al., 2010) la aplicación práctica de la explotación de información constituye un proyecto en sí mismo, que consta de una serie de etapas comunes. La estructura del proyecto más utilizada (Kdnuggets, 2014) es CRISP-DM (Chapman, et. al., 2000) (figura 2.1), la cual esta conformada por las siguientes fases.

1. **Comprensión del Negocio**, cuyo objetivo es comprender los objetivos y requerimientos del proyecto desde la perspectiva del negocio, así como identificar el problema de minería de datos y realizar la planificación del proyecto. Las actividades asociadas son: Determinar los Objetivos del Negocio, Evaluación de la Situación, Determinar las Metas de Minería de Datos y Producir el Plan del Proyecto.

2. **Comprensión de los Datos**, donde se realiza una recolección inicial de los datos y se los analiza con el objetivo de identificar problemas de calidad y posibles subconjuntos de interés para distintas hipótesis. Las actividades generales que lo integran son: Recolección Inicial de los Datos, Descripción de los Datos, Exploración de los Datos y Verificación de la Calidad de los Datos.

3. **Preparación de los Datos**, que conlleva la ejecución de todas las actividades necesarias para favorecer la calidad de los resultados. Las actividades generales que lo integran son: Seleccionar los Datos, Limpieza de los Datos, Construcción de los Datos, Integración de los Datos y Formateo de los Datos.

4. **Modelado**, donde se seleccionan y configuran las técnicas de modelado a utilizar. Las actividades generales que componen dicha fase son: Seleccionar de las Técnicas de Modelado, Generar el Diseño de las Pruebas, Construir el Modelo y Evaluar el Modelo.

5. **Evaluación**, donde se analiza el modelo generado para garantizar que este cumpla con los objetivos del negocio. Las actividades generales que lo integran son: Evaluar los Resultados, Revisar el Proceso y Determinar Próximos Pasos.

6. **Despliegue**, la cual abarca las actividades de integrar el conocimiento obtenido a algún sistema, o documentarlo para su posterior uso. Las actividades generales que conforman la última fase son: Planificar la Implementación, Planificar el Monitoreo y Mantenimiento, Producir el Reporte Final y Revisión del proyecto.

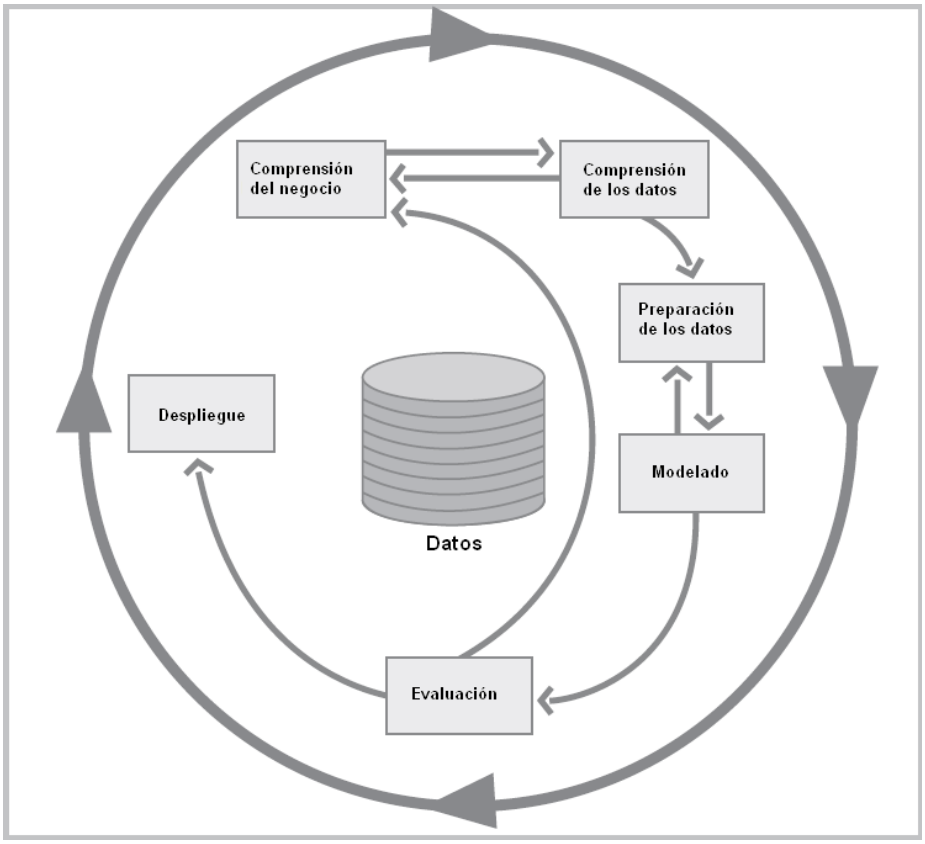

Figura 2.1. Proceso Estándar para Minería de Datos Multi Industria (CRISP-DM). Adaptado de [Chapman et al., 2000]

La etapa de modelado será de particular importancia para esta tesis ya que es donde se determinan las técnicas y algoritmos a utilizar para procesar los datos. Los algoritmos y técnicas utilizadas dependen, entre otras cosas, de la problemática abordada por el proyecto y de la capacidad del equipo de trabajo para encontrar los que mejor describan el dominio estudiado.

Existen casos de explotación de información que no pueden ser resueltos utilizando métodos de análisis tradicionales (numéricos y esencialmente cuantitativos), que en su mayoría están basados en el análisis estadístico. Algunos ejemplos de estos casos son la especificación de condiciones asociadas a diagnósticos técnicos o clínicos, identificación de características que permitan reconocimiento visual de objetos, descubrimiento de patrones o regularidades en estructuras de información (en particular en bases de datos de gran tamaño) (Britos, 2008).

Para abordar estas problemáticas se puede recurrir a métodos basados en sistemas inteligentes tales como la inducción de árboles de decisión, los mapas auto organizados o las redes neuronales, dando así lugar a la Explotación de información basada en Sistemas inteligentes (Britos, 2008).

2.3. Procesos de Explotación de Información

Los procesos de explotación de información definen las técnicas o algoritmos a utilizar en base a las características y necesidades del problema de explotación de información. En (Britos, 2008) y posteriormente en (García-Martínez, Britos y Rodríguez 2013) los autores proponen cinco procesos basados en sistemas inteligentes que dan solución a cinco problemáticas recurrentes en el ámbito de la inteligencia de negocios:

1) El descubrimiento de reglas de comportamiento: este proceso se aplica cuando se requiere identificar cuáles son las condiciones para obtener determinados resultados en el dominio del problema. Son ejemplos de problemas que requieren este proceso: la identificación de las características del local más visitado por los clientes, la identificación de factores que inciden en el alza de las ventas de un producto dado, establecimiento de las características o rasgos de los clientes con alto grado de fidelidad a la marca, el establecimiento de atributos demográficos y psicográficos que distinguen a los visitantes de un sitio web, entre otros.

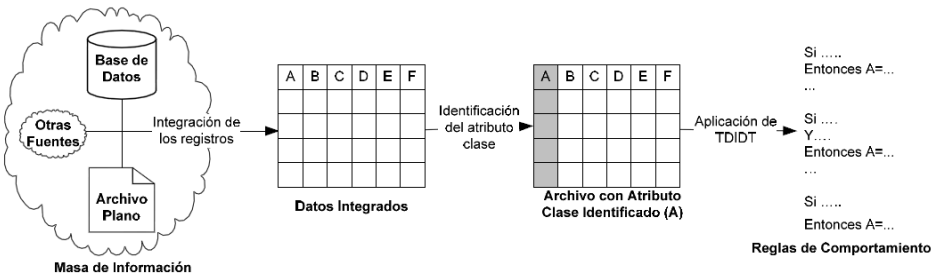

Figura 2.2. Proceso de descubrimiento de reglas de comportamiento. [García-Martínez et al., 2013]

2) El descubrimiento de grupos: este proceso es de utilidad cuando se requiere identificar una partición en la masa de información disponible sobre el dominio de un problema. Son ejemplos de problemas que requieren este proceso: la identificación de segmentos de clientes para bancos y financieras, la identificación de tipos de llamadas de los clientes para empresas de telecomunicación, la identificación de grupos sociales con las mismas características, la identificación de grupos de estudiantes con características homogéneas, entre otros.

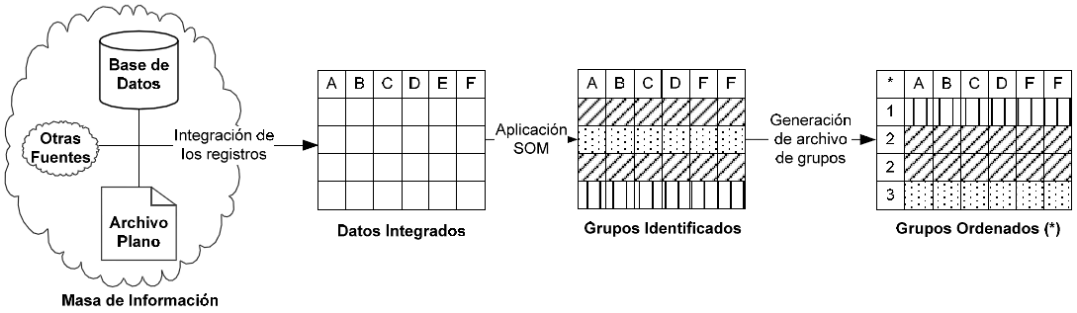

Figura 2.3. Proceso de descubrimiento de grupos [García-Martínez et al., 2013]

3) La ponderación de interdependencia de atributos: este proceso aplica cuando se requiere identificar cuáles son los factores con mayor incidencia (o frecuencia de ocurrencia) sobre un determinado resultado de un problema. Entre otros, son ejemplos de problemas de aplicabilidad de este proceso: la determinación de factores que poseen incidencia sobre las ventas, la determinación de los rasgos distintivos de clientes con alto grado de fidelidad a la marca, la individualización de los atributos claves que convierten en vendible a un determinado producto o las características sobresalientes que tienen los visitantes de un sitio web.

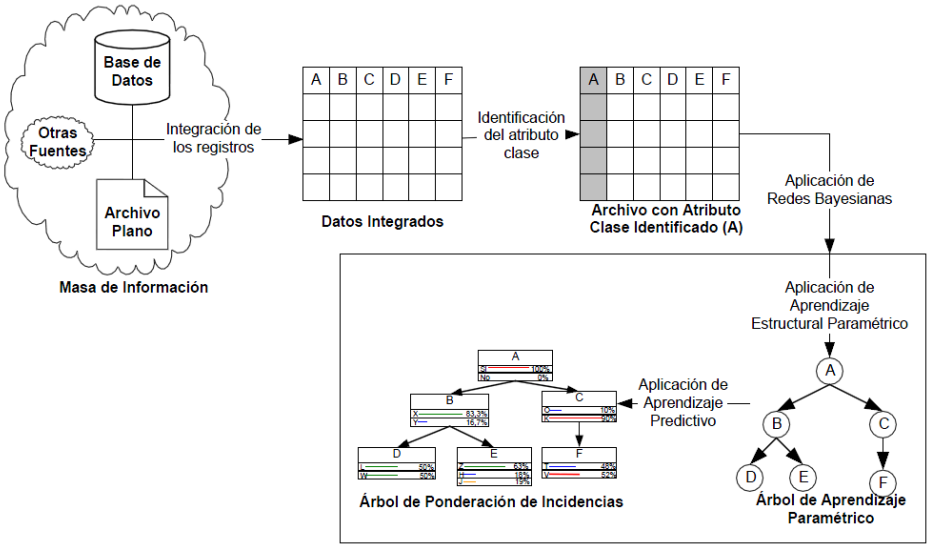

Figura 2.4. Proceso de ponderación de interdependencia de atributos [García-Martínez et al., 2013]

4) El descubrimiento de reglas de pertenencia a grupos: este proceso se utiliza cuando se busca identificar cuáles son las condiciones de pertenencia a cada una de las clases en una partición desconocida "a priori", pero que se encuentra presente en la masa de información disponible sobre el dominio de problema. Son ejemplos de problemas que requieren este proceso: el establecimiento de la tipología de perfiles de clientes y la caracterización de cada tipología, la distribución y estructura de datos de un sitio web, la segmentación etaria de estudiantes y el comportamiento de cada segmento, la determinación de las clases de llamadas telefónicas en una región y caracterización de cada clase, entre otros.

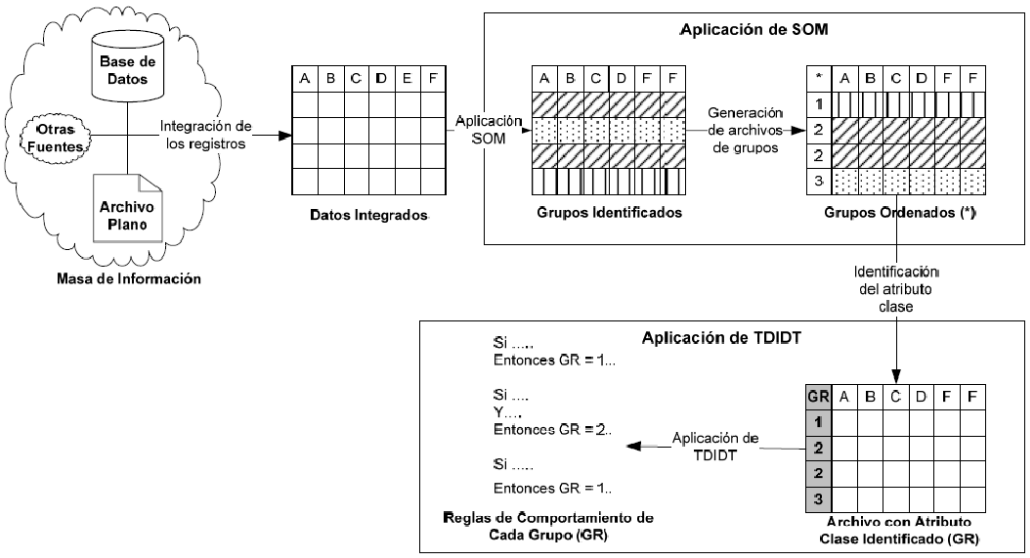

Figura 2.5. Proceso de descubrimiento de reglas de pertenencia a grupos [García-Martínez et al., 2013]

5) La ponderación de reglas de comportamiento o de pertenencia a grupos: este proceso es de utilidad cuando se requiere identificar cuáles son las condiciones con mayor incidencia (o frecuencia de ocurrencia) sobre la obtención de un determinado resultado en el dominio del problema, sean éstas las que en mayor medida inciden sobre un comportamiento o las que mejor definen la pertenencia a un grupo. Son ejemplos de problemas que requieren este proceso: la identificación del factor dominante que incide en el alza las ventas de un producto dado, el rasgo con mayor presencia en los clientes con alto grado de fidelidad a la marca, la frecuencia de ocurrencia de cada perfil de clientes, la identificación del tipo de llamada más frecuente en una región, entre otros.

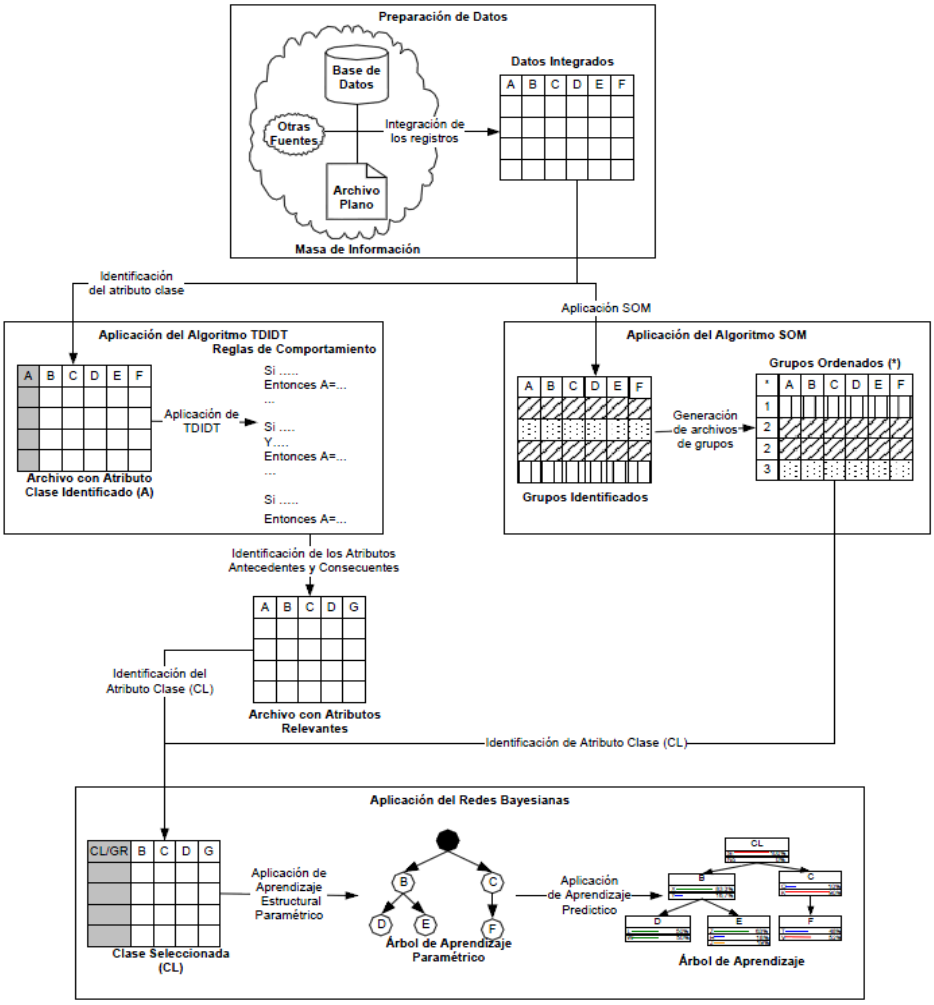

Figura 2.6. Proceso de ponderación de reglas de comportamiento o de la pertenencia a grupos [García-Martínez et al., 2013]

2.4. Algoritmos de Minería de Datos

Este trabajo se centrará en el proceso de descubrimiento de reglas de pertenencia a grupos. Como se definió en la sección anterior, este proceso consiste en tomar el conjunto de datos a estudiar y aplicar un algoritmo de agrupamiento o "clustering" para separarlo en distintos grupos (también llamados clases o clusters) y aplicar luego algoritmos de inducción de reglas para explicitar las que definen la pertenencia a cada grupo descubierto.

Tanto para el procedimiento de clustering como para la de inducción de reglas existen varios algoritmos (Xu y Tian 2015; Sehgal y Garg 2014; Panchuk, 2015) que permiten identificar distrintos tipos de patrones en los datos, y en los distintos tipos de datos. Es por esto quesurge la

necesidad de encontrar, en la diversidad de dominios, los algoritmos de minería de datos que mejor identifican los patrones ocultos en dicha masa de información, para obtener aquellos patrones que mejor expliquen el comportamiento del problema de negocio en estudio.

En este contexto, la sección actual describe aquellos algoritmos pertenecientes a la familia de Clustering (sección 2.4.1) y de Aprendizaje basado en Reglas de Inducción (sección 2.4.2) más relevantes considerados para el desarrollo de la tesis.

2.4.1. Algoritmo de Clustering

El análisis de grupos o *clustering* es la división de un conjunto de datos en grupos de elementos similares. Cada grupo o *cluster* agrupará elementos similares entre si, y disímiles respecto a elementos de otros grupos (Abbas, 2008). Distancia y alguna medida de "similitud" son las bases sobre las que se construyen los algoritmos de agrupamiento.

De acuerdo con el estudio realizado en (Xu y Tian 2015) los algoritmos de clustering *tradicionales* pueden clasificarse en las siguientes categorías, en base a la estrategia utilizada para construir los grupos: basados en partición, basados en jerarquía, basados en teoría difusa, basados en distribución, basados en densidad, basados en teoría de grafos, basados en grilla, basados en teoría fractal, y basados en modelo.

En este trabajo se analizarán algoritmos pertenecientes a las categorías: basados en partición (sección 2.4.1.1), basados en jerarquía (sección 2.4.1.2) y basados en densidad (2.4.1.3).

2.4.1.1. Clustering basado en partición

La idea básica detrás de este tipo de algoritmos es considerar el centro de la nube de puntos dato como el centro del cluster correspondiente. Entre las ventajas de este tipo de algoritmos podemos mencionar una complejidad en tiempo (*time complexity*) relativamente baja y la eficiencia computacional en general. Las desventajas observadas son :

- No son adecuados para datos no convexos
- La posibilidad de convergencia a un mínimo local, en función de los centros definidos inicialmente,
- El número de clusters deber ser especificado
- La dependencia entre los resultados del agrupamiento y el número de clusters especificado

Ejemplos de este tipo de algoritmos son: K-means, K-medoids, PAM, CLARA, CLARANS, Farthest First y Expectation-Maximization (EM). A continuación se describen los algoritmos

utilizados para este proyecto.

a) K-Means: También conocido como algoritmo de Lloyd, agrupa datos intentando separar los ejemplos en n grupos de igual varianza, minimizando un criterio conocido como inercia o suma de cuadrados interior al grupo (within-cluster sum-of-squares). Este algoritmo requiere que el número de clusters sea especificado.

K-means divide al set de datos de N ejemplos X en K clusters distintos C, cada uno descripto por la media μ_j de los ejemplos en el cluster. Dichos valores medios se denominan comúnmente "centroides". Cabe aclarar que los centroides en general no son puntos de X aunque si viven en el mismo espacio. El algoritmo tiende a elegir centroides que minimizan la inercia o suma de cuadrados interior al grupo:

$$J = \sum_{j=1}^{K} \sum_{i=1}^{N} \left(\left| x_i^{(j)} - c_j \right| \right)^2$$

La inercia o el criterio de la suma de cuadrados interior al grupo puede considerarse como una medida de cuan coherente son los clusters internamente. La misma sufre de varias limitaciones:

- La inercia asume que los clusters son convexos e isotropicos, aunque no siempre sea el caso
- Responde pobremente en clusters de formas alargadas o en variedades matemáticas de formas irregulares
- La inercia no es una métrica normalizada: se sabe que los valores bajos son mejores y que cero es el valor óptimo, pero en espacios de con alta cantidad de dimensiones, las distancias Euclidianas tienden a inflarse. Este fenómeno puede mitigarse ejecutando un proceso de reducción de dimensiones previo al proceso de clustering.

b) K-Means++: El algoritmo de Lloyd comienza eligiendo *k* datos puntos de la masa de datos como centros de partida uniformemente al azar. Cada dato punto es luego asignado al centro mas cercano y el centro es re-calculado como el centro de masa de todos los puntos asignados a ese cluster. Estos dos últimos pasos se repiten hasta que el proceso se estabiliza.
Generalmente, es la velocidad del algoritmo y no la precisión lo que lo hace atractivo. De hecho, existen varios ejemplos naturales en los cuales el algoritmo genera agrupaciones pobres arbitrariamente. Esto último no depende de una elección de centros inicial adversa y en particular esta situación se mantiene aun eligiendo los centros uniformemente al azar de la masa de datos.

En (Arthur, Vassilvitskii 2007) los autores proponen una variante al algoritmo de Lloyd donde en vez de elegir los centros iniciales en forma uniforme y al azar, se seleccionan uno a continuación de otro de forma en base a la distancia al cuadrado al centro mas cercano. En términos simples, es más probable que un dato punto sea elegido como centro si su distancia al centro más cercano es alta. La probabilidad de un dato *x* perteneciente al conjunto de datos *X* de convertirse en un nuevo centro esta dada por:

$$P = \frac{D(x)^2}{\sum_{x \in X} D(x)^2}$$

Siendo $D(x)$ la menor distancia de un punto dato al centro más cercano previamente elegido

c) K-Means elección inicial de centros mediante PCA: En (Alrabea, Senthilkumar, Al-Shalabi, Bader 2013) se propone la utilización del primer componente generado mediante análisis de componentes principales (PCA) para inicializar los centroides del algoritmo K-Means. Esta estrategia disminuye el tiempo de clustering y mejora la precisión comparado con el algoritmo original. El análisis de componentes principales es un método que reduce las dimensiones de los datos mediante el análisis de co-varianza entre factores. Por esta razón, es adecuado para sets de datos de múltiples dimensiones. Implica un proceso en donde un espacio de datos en transformado en un espacio de atributos, el cual posee una dimensión reducida.

c) Mini Batch K-Means: Este algoritmo es una variante de K-Means que utiliza mini-lotes ó *mini-batches* para reducir el tiempo de cómputo, mientras continúa intentando optimizar la misma función objetivo. Los mini-lotes son subgrupos del set de datos de entrada, muestreados en forma aleatoria en cada iteración de entrenamiento. Estos mini-lotes reducen drásticamente la cantidad de cómputo requerido para converger a una solución local. En contraste con otros algoritmos que reducen el tiempo de convergencia de K-Means, mini-batches K-Means produce resultados ligeramente peores que el algoritmo estándar (Sculley 2010).

El algoritmo itera entre dos grandes pasos, al igual que K-Means estándard. En el primer paso, se extraen *b* ejemplos del set de datos en forma aleatoria, para formar un mini-lote. Estos son luego asignados al centroide más cercano. En el segundo paso, los centroides son actualizados. En contraste con K-Means, es se realiza una vez por ejemplo. Para cada ejemplo en el mini-lote, se actualiza el centroide asignado tomando el promedio móvil del ejemplo y todos los ejemplos previos en el tiempo. Estos pasos se ejecutan hasta la convergencia o hasta alcanzar un número de iteraciones predeterminado

Mini-Batch K-Means converge más rápidamente que K-Means, pero la calidad de los resultados se ver reducida aunque en la práctica, esta diferencia en calidad puede ser muy pequeña (Sculley

2010).

2.4.1.2. Clustering basado en jerarquía

Esta categoría de algoritmos opera construyendo una relación jerárquica entre los datos con el fin de agruparlos. Inicialmente, consideran cada punto dato, un cluster individual y luego progresan fusionando los dos cluster más cercanos para formar un nuevo cluster, continuando hasta que solo quede uno. El proceso también puede ocurrir a la inversa. Estos algoritmos son adecuados para sets de datos de forma arbitraria y con atributos de tipos de datos también arbitrarios. La relación jerárquica entre clusters es fácilmente detectada y la escalibilidad de los algoritmos es relativamente alta en general. Las desventajas observadas son:

- Complejidad en tiempo (*time complexity*) relativamente alta en general
- El número de clusters debe ser especificado

Ejemplos de este tipo de algoritmos son: BIRCH, CURE, ROCK, Chameleon. A continuación se describe el algoritmo de esta familia seleccionado:

a) BIRCH: Para los datos dados, este algoritmo construye lo que se denomina un árbol de atributo característico o *Characteristic Feature Tree* (árbol CF). Basicamente, los datos son comprimidos con pérdidas en un grupo de nodos de atributo característico ó *Characteristic Feature nodes* (nodos CF). Los nodos CF poseen un número de sub clusters llamados sub clusters de atributo característico ó *Characteristic Feature Subclusters (subclusters* CF*)* y estos subclusters CF situados en un nodos CF no terminales pueden tener nodos CF como *hijos*.

Los subclusters CF contienen la información necesaria para el agrupamiento, lo que evita la necesidad de mantener el set de datos completo en memoria. Esta información incluye:

- Número de ejemplos en un sub cluster
- Suma lineal: Un vector n-dimensional conteniendo la suma de todos los ejemplos
- Suma al cuadrado: la suma de la norma L2 (cuadrados mínimos) al cuadrado de todos los ejemplos
- Centroides
- La norma al cuadrado de los centroides

BIRCH requiere dos parámetros: el umbral y el factor de ramificación o *branching factor*. Este último limita el número de subclusters en un nodo y el umbral limita la distancia entre los sub clusters existentes y el ejemplo a ser insertado.

Este algoritmo puede ser visto como un método de reducción de datos, ya que comprime los datos provistos a un grupo de sub clusters, que se obtienen directamente de las hojas del árbol

CF. El set de datos reducido puede ser procesado posteriormente en una etapa de agrupamiento global, por otros algoritmos conocidos.

Un nuevo ejemplo es insertado en la raíz del árbol CF, que es un nodo CF. Este es luego anexado al subcluster de la raíz que posea el menor radio luego de incluirlo, limitado por el umbral y el factor de ramificación. Si el subcluster posee algún nodo *hijo*, el proceso se repite hasta alcanzar una *hoja*. Una vez encontrado el subcluster más cercano en la *hoja,* las propiedades de este y los subclusters padre se actualizan en forma recursiva.
Si el radio del subcluster obtenido luego de anexar el nuevo ejemplo al subcluster más cercano es mayor que el cuadrado del umbral y si el número de subclusters es mayor que el factor de ramificación, se aloca un espacio temporal para el nuevo ejemplo. Los dos subclusters mas lejanos son divididos en dos grupos, en base a la distancia entre ellos. Si el nodo dividido tiene un subcluster *padre* y existe espacio para un nuevo subcluster, se divide al *padre* en dos. Si no hay espacio, entonces este nodo es dividido nuevamente en dos y el proceso continua en forma recursiva hasta llegar a la raíz.

2.4.1.3. Clustering basado en densidad

Este tipo de algoritmos opera bajo la premisa de que los datos en una región de alta densidad dentro del espacio de datos se consideran como parte de un mismo cluster. Estos algoritmos son adecuados para datos con forma arbitraria. Las desventajas observadas son:

- Los resultados del proceso de clustering presentan una baja calidad cuando la densidad del espacio de datos no es uniforme
- Los resultados del proceso de clustering son altamente sensibles a los parámetros.

Ejemplos de este tipo de algoritmos son DBSCAN, Mean-shift y OPTICS. A continuación se describen los algoritmos utilizados para este proyecto.

a) DBSCAN: Este algoritmo ve a los clusters como áreas de alta densidad separados por áreas de baja densidad. Debido a esta visión particularmente genérica, los clusters generados por DBSCAN pueden tener cualquier forma, a diferencia de K-Means, que asume que los clustes son convexos.

El componente central de DBSCAN es el concepto de datos centrales ó *core* que son aquellos situados en áreas de alta densidad. Un cluster es, por lo tanto, un conjunto de datos centrales, cercanos unos a otros (según una medida de distancia) y un conjunto de elementos no-centrales, cercanos a un dato central (pero sin ser un dato central).

El algoritmo posee dos parámetros: *min_samples* y *eps (Epsilon)*, que definen formalmente el concepto de densidad. Valores altos de *min_samples* o bajos de *eps* implican una mayor densidad necesaria para formar un cluster.

Más formalmente, definimos un dato central o *core* como aquel donde existe una cantidad de datos punto *min_samples* en un radio *eps.* A este espacio de radio *eps* se lo denomina *vecindario* del dato, y los datos en él contenidos se denominan *vecinos*. Aquellos datos que tienen menos de *min_samples* vecinos en su vecindario, pero poseen al menos un dato central se los denomina datos border. Por último, los datos que no cumplen con las condiciones para ser centrales o border son agrupados en un cluster denominado *noise* ó ruido. Todos los datos en este cluster al finalizar la ejecución del algoritmo son considerados *outliers* por no encontrarse lo suficientemente cercanos a ninguna zona de alta densidad.

Todo dato central pertenece a un cluster por definición y el mismo se forma partiendo de un dato central, encontrando todos los datos vecinos que son datos centrales, y analizando a su vez sus vecindarios en búsqueda de nuevos datos centrales, en forma recursiva. Los datos frontera o *border* encontrados son también anexados al cluster en este proceso.

En (Ester, Kriegel, Sander, Xu 1996) los autores del algoritmo también proponen una heurística para estimar el valor de *min_samples* y *eps* para el cluster menos denso del set de datos.

Para un valor de *k* dado, se define una función *k-distancia* mapeando cada dato punto con la distancia de su k-enesimo vecino más cercano. Cuando se ordenan los puntos del set de datos en orden descendente en base a los valores de la función *k-distancia* el gráfico de dicha función ofrece algunas pistas respecto a la distribución de densidad del set de datos. Este gráfico se denomina *k-distancia ordenado.* Si elegimos un punto arbitrario *p,* definimos el parámetro *eps = k-distancia(p)* y el parámetro *min_samples = k,* todos los datos punto con un valor de *k-distancia* igual o menor serán datos centrales. Si podemos encontrar un dato umbral con el valor máximo de *k-distancia* en el cluster menos denso del set de datos, obtendríamos los valores de los parámetro deseados. El dato umbral es el primer dato en el primer *valle* del gráfico *k-distancia ordernado.* Todos los datos punto con un valor de *k-distancia* mayor (a la izquierda del dato umbral) son considerados *outliers,* todos los demás puntos (a la derecha del dato umbral) son asignados a algún cluster.

En general es muy difícil detectar el primer *valle* automáticamente pero es relativamente fácil para una persona ver dicho *valle* en una representación gráfica. Es por esto que los autores recomiendan esta forma interactiva para determinar el dato umbral.

Aunque DBSCAN necesita dos parámetros (*eps* y *min_samples)* los experimentos realizados por los autores del algoritmo indican que los gráficos *k-distancia* con $k > 4$ no difieren significativamente del grafico *4-distancia* y además requiere un mayor tiempo de cómputo. Por esto recomiendan usar *min_samples* = 4, para sets de datos de dos dimensiones.

b) Meanshift: Este algoritmo apunta a detectar grupos en zonas de ejemplos de densidad

homogénea. Es un algoritmo basado en centroides que opera actualizando candidatos a centroide para que constituyan la media de los puntos contenidos en una región dada. Estos candidatos son luego filtrados en una etapa posterior para eliminar centroides similares y obtener los definitivos.

Dado un centroide candidato $x_{i,}$ para la iteración t, el candidato se actualiza de acuerdo a la siguiente ecuación:

$$x_i^{t+1} = x_i^t + m(x_i^t)$$

Donde m es el vector "variación de la media" o *mean shift* que se computa para cada centroide y apunta en la dirección de máximo crecimiento de la densidad (gradiente de la función de densidad). El mismo se calcula utilizando la siguiente ecuación, actualizando efectivamente un centroide de forma tal que sea la media de los ejemplos en su *vecindario* $N(x_i)$:

$$m(x_i) = \frac{\sum_{x_j \in N(x_i)} K(x_j - x_i)\, x_j}{\sum_{x_j \in N(x_i)} K(x_j - x_i)}$$

El algoritmo automáticamente define el número de clusters, en vez de basarse en un parámetro *bandwidth* o ancho de banda, que define el tamaño de la región a inspeccionar.

2.4.2. Algoritmos de Aprendizaje Inductivo

El aprendizaje inductivo se base en la generalización de reglas y conclusiones en base a la observación de conocimiento previo. Este permite identificar patrones y similitudes en datos de entrenamiento o conocimiento previo para construir reglas generalizadas (Almana y Aksoy 2014).

Según (Kotu, Deshpande, 2015) existen dos métodos para el análisis y extracción de reglas a partir de conocimiento previo: **el método directo**, que obtiene las reglas directamente del set de datos y se basa a la relación entre atributo y clase (o etiqueta) del set de datos. **El método indirecto,** en cambio, extrae las reglas de un modelo de clasificación (árboles de decisión, redes neuronales, etc) previamente construido a partir de los mismos datos.

Cabe mencionar que en (Panchuk, 2015) se utiliza el algoritmo Apriori para la inducción de reglas de pertenencia a grupos. Este algoritmo entra en la categoría de algoritmos de minería de patrones frecuentes. Los patrones frecuentes son aquellos ítems, secuencias o subestructuras recurrentes en bases de datos de transacciones, en base a una frecuencia especificada por el usuario. Un *itemset* con frecuencia mayor o igual al umbral mínimo se lo considerará como

patrón frecuente.
Agrawal et al. (1993) fue el primero en introducir el concepto de minería de patrones frecuentes en el análisis de compras en supermercados, para descubrir asociaciones entre los productos adquiridos. Este concepto utiliza bases de datos transaccionales y otros repositorios de datos con el fin de extraer estructuras de asociación casuales, correlaciones interesantes o patrones frecuentes entre diferentes grupos de datos. Otros ejemplos de este tipo de algoritmos son: FP-growth o RARM (Nasreen, Azam, Shehzad, Naeem & Ghazanfar 2014).

2.4.2.1. Método Directo

Entre los algoritmos que utilizan el método directo, los más utilizados en la disciplina son aquellos agrupados bajo la denominación Divide y vencerás (*Separate-and-Conquer).* Algunos ejemplos de algoritmos pertenecientes a esta familia, son: AQ15*,* CN2, RIPPER, etc. En la sección 2.4.2.1.1 se detallará el funcionamiento del algoritmo CN2, de interés para el desarrollo de esta tesis.

2.4.2.1.1 Algoritmo CN2

Este algoritmo incorpora ideas del algoritmo ID3 (Quinlan, 1983) y del algoritmo AQR, una reconstrucción del método AQ de Michalski realizada por los autores del algoritmo CN2 (Clark, Nibblet 1989).

Funciona en forma iterativa, buscando con cada iteración un conjunto de "complejos" que cubra un gran número de ejemplos para una única clase C y pocos para otras clases.

El término "complejo" se hereda de AQR y se refiere a una conjunción selectores. A su vez, los selectores son evaluaciones simples de atributos. Por ejemplo, (attr1=si), (attr2=A ó B) y (attr3 > 60) son todos ejemplos de selectores. Entonces, un ejemplo de complejo sería (attr1=si) AND (attr2=A ó B).

El complejo debe ser tanto predictivo como confiable, tal como como lo definen las funciones de evaluación de CN2. Una vez encontrar un buen complejo, el algoritmo remueve aquellos ejemplos que cubre del set de entrenamiento y agrega la regla "if [complejo] then predict C" al final de la lista de reglas. Este proceso continúa hasta no se puedan encontrar mas complejos satisfactorios.

El algoritmo busca complejos ejecutando una búsqueda de general a especifico podada. A cada etapa de la búsqueda, CN2 retiene un grupo de tamaño limitado S (star) conteniendo "los mejores complejos encontrados hasta ahora". El algoritmo solo evalúa especializaciones de este grupo mediante una búsqueda de haz sobre el espacio de complejos. Un complejo es especializado agregando un nuevo término conjuntivo (*AND*) o eliminando un elemento

disyuntivo (*OR*) de uno de sus selectores. Cada complejo puede ser especializado de varias formas y CN2 genera y evalúa todas ellas. El star es reducido al finalizar este paso removiendo sus elementos de menor rango de acuerdo con una función de evaluación.

CN2 maneja los atributos continuos dividiendo el rango de valores de cada atributo en rangos discretos. La evaluación de dichos atributos determina si un valor es mayor o menor (o igual) que los valores en los límites del subrango. El rango completo de valores y el tamaño de cada subrango es provisto por el usuario

CN2 utiliza un método simple para lidiar con valores de atributos desconocidos: reemplaza los valores desconocidos con los valores de dicho atributo que ocurren más frecuentes en el set de entrenamiento. En el caso de atributos numéricos, utiliza el valor medio del subrango con mayor frecuencia de ocurrencia.

2.4.2.2. Método Indirecto

Cómo se menciona en (Kotu, Deshpande, 2015) y en (Hailesilassie, 2016) los métodos indirectos son aquellos que pueden extraerse reglas a partir de modelos de clasificación. Algunos ejemplos son los algoritmos redes neuronales y árboles de decisión. A continuación se describiremos estos últimos para luego enfocarnos en los algoritmos que los generan.

2.4.2.2.1 Modelos utilizados en el método indirecto

Los árboles de decisión son estructuras de datos jerárquicas aplicables al aprendizaje supervisado tanto para problemas de clasificación como de regresión (Alpaydin, 2010). De acuerdo con (Mitchell, 1997) los árboles de decisión categorizan datos organizándolos desde la raíz hasta un nodo hoja particular, situado en el extremo inferior del árbol. Cualquier nodo del árbol representa la evaluación de algún atributo del dato mientras que cada ramificación del nodo representa un valor posible de dicho atributo. Los nodos hoja pueden contener valores de clase (clasificación), valores continuos (regresión), modelos no lineales (regresión) y hasta modelos producidos por otros algoritmos de aprendizaje automático (Barros et al, 2015).

Muchos árboles de decisión pueden construirse a partir de los mismos datos. La **inducción de un árbol de decisión** óptimo a partir de datos es considerado una tarea compleja. Por ejemplo Hyafil and Rivest demostraron que construir un árbol binario mínimo respecto al número esperado de pruebas requeridas para clasificar un objecto no visto es un problema NP (*nodeterministic polynomial time* o tiempo polinómico no determinístico) completo. En (Hancock et al) se prueba que encontrar un árbol de decisión mínimo consistente con el set de entrenamiento es NP-complejo, siendo también este el caso si se pretende encontrar el árbol de

decisión mínimo equivalente para un árbol de decisión dado o a partir de tablas de decisión.

En las últimas tres décadas se desarrollaron varias líneas de investigación con el propósito de resolver el problema de la construcción de árboles de decisión. En ese sentido, varias de las líneas desarrolladas son capaces de obtener arboles de decisión razonablemente precisos, aunque sub-óptimos en un tiempo reducido. Entre estas últimas existe una clara preferencia en la literatura por algoritmos que implementan una estrategia de particionado *greedy* (codiciosa), de arriba hacia abajo (*top-down*) y recursiva para la construcción del árbol (*top-down induction* o inducción de arriba hacia abajo).

***Concept Learning System framework (CLS)* ó Estructura para el Sistema de Aprendizaje de Conceptos** desarrollado por E.B. Hunt (Hunt et al, 1966) es mencionado como el trabajo que dió origen a la **inducción de árboles de decisión de arriba hacia abajo (TDIDT por sus siglas en inglés)**. CLS intenta minimizar el costo de clasificar un objeto. En este contexto, el costo se refiere a dos conceptos diferentes: el costo de medición para determinar el valor de cierto propiedad (atributo) exhibido por un objeto y el costo de clasificar un objeto como perteneciente a la clase *j* cuando en realidad pertenece a la clase *k.* A cada etapa, CLS explota el espacio de árboles de decisión posibles hasta una profundidad determinada, en este espacio limitado elige una acción para minimizar el costo y luego se mueve un nivel hacia abajo en el árbol.

El algoritmo de Hunt simplificado constituye la base de todos los algoritmos TDIDT. Sin embargo, sus presunciones son demasiado rigurosas para su aplicación práctica. Por ejemplo, solo funcionaría si todas las combinaciones de valores de los atributos están presentes en el set de entrenamiento y si los datos de entrenamiento no poseen inconsistencias (cada combinación tiene una clase única).

El algoritmo de Hunt fue mejorado en varios aspectos: su criterio de corte, por ejemplo, requería que todos los nodos hoja fueran puros (es decir, que pertenezcan a la misma clase). En la mayoría de los casos prácticos, esta limitación conduce a árboles de decisión enormes, que tienden a sufrir de ***overfitting*** (fenómeno que ocurre cuando un clasificador sobre-aprende los datos, es decir, cuando asimila todas las peculiaridades de los datos, incluyendo potencial ruido y patrones espurios son propios del set de entrenamiento y que no congenian con el set de prueba o no lo describen con precisión).

Otro problema de diseño es la selección de la condición de evaluación del atributo para particionar las instancias en grupos más pequeños. En la solución original de Hunt, una función del costo era la responsable de particionar el árbol. En algoritmos subsecuentes como ID3 and C4.5 se utilizan funciones basadas en teoría de la información para particionar los nodos en subgrupos más puros.

Además de los algoritmos TDIDT, existen otras estrategias de generación de árboles. La **inducción de árboles de decisión de abajo hacia arriba (*bottom-up*)** fue mencionada por primera vez en (Landeweerd et al. 1983). Los autores proponen una estrategia que se asemeja al clustering jerárquico aglomerativo. El algoritmo comienza conteniendo en cada hoja objetos de la misma clase. De esa forma, un problema de *k-clases* generaría un árbol de decisión con *k* hojas.
La operación clave de esta estrategia consiste en combinar en un nodo no terminal, las dos clases mas parecidas, en forma recursiva. Luego, un hiperplano es asociado a un nuevo nodo no terminal, en forma similar a la inducción de árboles oblicuos. A continuación, todos los objetos en el nuevo nodo no terminal son considerados miembros de la misma clase (una clase artificial que engloba las dos clases agrupadas), y el proceso evalúa una vez más cuales son las dos clases más parecidas. Repitiendo esta estrategia en forma recursiva, se obtiene un árbol de decisión en el cuál las clasificaciones más obvias son realizadas primero, y las distinciones más sutiles son relegadas a niveles inferiores.

Algunas desventajas obvias de la inducción de abajo hacia arriba son: (1) los problemas de clasificación binaria generan un árbol de decisión de un único nivel (el nodo raíz y dos hijos). Un árbol tan simple impide modelizar problemas complejos. (2) las instancias de una misma clase pueden situarse en regiones muy distintas del espacio de atributos, afectando la presunción inicial de que las instancias de una misma clase debe situarse en el mismo nodo hoja (3) el *clustering* jerárquico y la generación de hiperplanos son operaciones costosas.

(Barros et al, 2015) sostienen que probablemente sean estas las razones por la cual la inducción de abajo hacia arriba no es tan popular como la de arriba hacia abajo.

La **inducción de árboles de decisión híbrida** fue investigada en (Kim, Landgrebe 1991). La idea es combinar ambas estrategias (*top-down* y *bottom-up*) para construir el árbol de decisión final. Este algoritmo comienza ejecutando la estrategia *bottom-up* descripta anteriormente hasta obtener dos subgrupos. A continuación, se extraen dos centros (media vectorial) e información de covarianza de los subgrupos y se utiliza para dividir el set de entrenamiento al modo *top-down,* según la suma de error al cuadrado normalizado. Si las dos nuevas particiones inducidas corresponden a clases distintas, entonces se da por concluida la inducción híbrida. En caso contrario, para cada subgrupo que no corresponda a una clase, se ejecuta en forma recursiva la inducción híbrida comenzando nuevamente con el proceso *bottom-up.*

En (Kim, Landgrebe 1991) se sostiene que con la inducción híbrida “es más probable la convergencia a clases de valor informativo, ya que el comienzo basado en grupos apunta tempranamente en esa dirección, mientras que la estrategia *top-down* directa no garantiza dicha convergencia”

Varios estudios han intentando evitar la estrategia *greedy* ("codiciosa") empleada comúnmente para inducir árboles. Por ejemplo, se ha empleado *lookahead* para intentar mejorar la inducción *greedy* (Buntine, 1992)(Chou, 1991)(Dong, Kothari 2001)(Murthy, Salzberg 1995)(Norton, 1989). Murthy y Salzberg demuestran que *lookahead* de un nivel no colabora con producir mejores árboles y puede inclusive empeorar la calidad de los árboles inducidos.

Una estrategia más reciente para evitar la inducción de árboles de decisión *greedy* es generarlos mediante **algoritmos evolutivos** (R.C. Barros et al, 2012).

Otros ejemplos de estrategias no *greedy* para inducir árboles de decisión incluye (1) utilizar programación lineal para complementar el árbol inducido (Bennett, 1994) (2) reestructuración incremental y no incremental del árboles de decisión (Utgoff, Berkman, Clouse 1997) (3) sesgar los datos para simular una distribución alternativa con el objetivo de lidiar con casos problemáticos para los árboles de decisión (Page, Ray, 2003) (4) Aprendizaje de árboles de decisión "Anytime" (Esmeir, Markovitch 2007).

Aunque se ha empleado mucho esfuerzo en el diseño de algoritmos de inducción de árboles no *greedy*, es discutible el hecho de que estos intentos puedan obtener mejores resultados que una estrategia *greedy, top-down* en forma consistente. En la mayoría de los casos, la ganancia en performance obtenida por las estrategias no *greedy* no es suficiente para compensar el esfuerzo computacional extra.

2.4.2.2.1 Algoritmo CART

Este algoritmo es muy similar a C4.5 (Quinlan, 1993) pero difiere en que soporta valores numéricos para las clases (regresión) y no computa reglas. CART construye árboles binarios en base a el atributo y el umbral que produzca la mayor ganancia de información para cada nodo.

Cuando la variable objetivo o clase es discreta, el árbol de decisión se denomina "de clasificación" mientras que cuando es continua se lo llama "árbol de regresión".

La creación de un modelo CART implica seleccionar las variables de entrada y los puntos donde dividirlas hasta obtener un árbol adecuado. Al igual que otros algoritmos de inducción de árboles, la selección de variables y puntos de división ó corte se realiza mediante un algoritmo codicioso o greedy, que básicamente busca minimizar una función de costo.

Para modelos de regresión, la función de costo minimizada para seleccionar los puntos de división es la de suma de error cuadrático, mientras que para clasificación, se utiliza el coeficiente de Gini.

2.5. Evaluación de Calidad

Parte esencial de todo proceso de ingeniería de explotación de información es la evaluación de la calidad de los patrones detectados. Esta tiene como objetivo determinar cuan buenos son los resultados, ya sea como herramienta de distinción entre los aspectos relevantes de los patrones capturados, así como en su correcta generalización.

Las herramientas para evaluar la performance de un modelo, varia según el tipo de aprendizaje al que pertenece. Para este proyecto, se describirán las métricas más relevantes para los método de Clustering (sección 2.5.1) y de Clasificación (sección 2.5.2).

2.5.1. Evaluación de Calidad en Clustering

La evaluación de los resultados de un proceso de clustering o agrupamiento es tan compleja como el proceso de clustering en sí. Generalmente, las formas de abordar la evaluación se pueden agrupar en tres categorías: evaluación **interna**, donde el agrupamiento es reducido a un único indicador de "calidad", evaluación **externa**, donde los resultados del proceso son comparados con una clasificación de referencia preexistente, evaluación **manual**, en la que interviene un humano experto e **indirecta,** donde se evalúa la utilidad de los resultados respecto al caso de aplicación (Feldman y Sanger 2007).

A diferencia de los procesos de clustering (no supervisados), la performance de un proceso de inducción de reglas (supervisado) puede ser medida en forma más objetiva ya que se cuenta con los casos clasificados. Una vez entrenado el algoritmo y aplicado al set de datos se puede contrastar la clasificación obtenida con los datos originales y determinar coeficientes simples como la precisión, el recall, la medida "F" (F-measure) o incurrir en análisis más complejos, menos sesgados como ROC (Receiver operating characteristic) o la determinación del "informedness" y el "markedness" (Powers, 2007)

2.5.1.1 Métricas de evaluación interna

El objetivo de un algoritmo de clustering es generar particiones donde los elementos dentro de un cluster son similares y los elementos pertenecientes a diferentes clusters son disímiles. (Arbelaitz, Gurrutxaga, Muguerza, Pérez, & Perona 2013). No existe una forma óptima de particionar los datos, ya que diferentes algoritmos o inclusive diferentes configuraciones del mismo algoritmo generan particiones diferentes y ninguna ha probado ser la mejor en todas las

situaciones (Pal & Biswas 1997).

Una forma de validar los resultados de un algoritmo de clustering es utilizando unicamente el set de datos y las etiquetas (o clases) asignadas por el algoritmo: este tipo de evaluación se denomina interna. Existen diferentes índices de evaluación interna, cuyo fin es medir que tan compactos son los clusters y la separación que existe entre ellos (Arbelaitz et all, 2013):

a) Índice Davies-Bouldin
Este es probablemente uno de los índices más utilizados en estudios de comparación de índices de validez interna. Estima la cohesión basado en la distancia de los puntos del cluster a su centroide y la separación entre centroides.

b) Índice Dunn
Es un índice tipo ratio. En este caso la cohesión es estimada por la distancia del vecino más cercano y la separación por el máximo diámetro de cluster.

c) Índice Calinski-Harabasz
Es también un índice tipo ratio donde la cohesión se estima basada en la distancia de los puntos del cluster a su centroide. La separación se basa en la distancia de los centroides a un centroide global.

d) Índice silhouette
La cohesión para éste índice se mide en base a la distancia entre todos los puntos en el mismo cluster y la separación a partir de la distancia del vecino más cercano.

e) Suma de cuadrados
En el análisis de clusters, la varianza dentro del grupo y la varianza entre grupos puede ser calculada mediante la suma de cuadrados dentro del cluster y la suma de cuadrados entre los clusters respectivamente.

2.5.2 Evaluación de calidad en clasificación

Existen numerosas métricas que pueden utilizarse para evaluar la calidad de un clasificador. Un grupo de ellas esta basado en las diferentes celdas de la matriz de confusión, que denota los valores de Falsos-positivos, Falsos-negativos, Verdaderos-positivos y Verdaderos negativos, para un problema de clasificación dado: recall, accuracy, precision, F-score, sensitivity entre otras (Fawcett, 2006). El autor señala además que cualquier métrica que utilice ambas columnas de la matriz de confusión (accuracy, precision, F-score) sera sensible a cambios en la distribución de las clases. Debido a esto, sugiere la utilización de la curva ROC, que es inmune a los cambios en

la distribución de las clases.

2.5.2.1 Área bajo la curva ROC

La curva ROC (Receiver Operating Characteristic o Característica Operativa del Receptor) es bidimensional en el cual el ratio de Verdaderos-Positivos es graficado en el eje Y y el de Falsos-positivos en el eje X. Este tipo de gráficos describe el intercambio relativo entre beneficios (Verdaderos-Positivos) y el costo (Falsos-Positivos) (Fawcett, 2006)

Además de ser un método útil para graficar la performance, tiene propiedades que la hacen especialmente útil en dominios que poseen una distribución de clases sesgada y costos de error de clasificación desiguales. A diferencia de otras métricas generalmente utilizadas, permite contemplar los falsos positivos, factor vital en contextos de desbalanceo de datos.

Investigaciones recientes (Jurgovsky 2018) presentan estrategias para evaluar la performance de un clasificador basadas en el área bajo la curva (AUC, por sus siglas en ingleś) ROC. A mayor valor de área, mejor será la capacidad de las reglas generadas de clasificar los ejemplos correctamente (Bradley, A 1997).

3. Descripción del problema

En todo proyecto de explotación de información, es una etapa esencial el entendimiento del problema de negocio y su mapeo con el proceso (o conjunto de algoritmos) que da respuesta a dicho problema. Para el proceso de descubrimiento de reglas de pertenencia a grupos existen numerosas estrategias de combinación derivadas de la abundante cantidad de algoritmos tanto de descubrimiento de grupos como de inducción de reglas. Es por eso que al momento de diseñar un proceso de descubrimiento de reglas de pertenencia a grupos surge la problemática de determinar: ¿Cuál es la pareja de algoritmos más conveniente?. Numerosos estudios señalan que la respuesta a dicha pregunta tiene dependencia con las características del set de datos.

La cantidad aproximaciones y la variedad de algoritmos se explica tanto por la necesidad de encontrar soluciones más eficientes como por la diversidad de dominios de aplicación de la explotación de información. Numerosos trabajos demuestran que la performance de los algoritmos varía notablemente tanto con las características del set de datos a analizar como por los valores de los ejemplos de dicho set (Kogan, 2007; López-Nocera, 2012; Panchuk, 2015; Sehgal y Garg 2014; Smith et al., 2002). Se puede concluir entonces que uno de los desafíos en en la ingeniería de explotación de información es encontrar los algoritmos que mejor describen el conocimiento oculto en el set de datos a analizar.

Para reforzar esta noción de la dependencia entre el set de datos a analizar y los resultados de un proceso de explotación de información es menester mencionar que existe también una línea de investigación, tangencial a este trabajo, enfocada en el desarrollo de algoritmos destinados a reducir la dimensionalidad (mediante la selección de atributos) e incrementar la precisión a un bajo costo computacional (Goswami, Chakrabarti y Chakraborti 2016). Estos algoritmos se utilizan previo al subproceso de explotación de información principal (clustering, inducción de reglas, etc) en una etapa de preparación del set de datos.
A su vez, y al igual que los procesos de clustering y clasificación, también las características del set de datos influyen en la efectividad de estos algoritmos y la elección del más apropiado para un set de datos dado no es un proceso trivial (Goswami et al., 2016)

A partir de los estudios realizados en (Kogan, 2007; López-Nocera, 2012; Panchuk, 2015) y como resultado de las líneas de trabajos futuras definidas en ellos, se desprende la necesidad de extender el estudio en busca de un método que permita identificar a priori qué combinación de algoritmos de clustering e inducción de reglas es el más adecuado para un set de datos dado, sin recurrir a características que dependen en sí mismas de los resultados del proceso de descubrimiento de reglas de pertenencia a grupos. Esto implica la capacidad de determinar qué pareja de algoritmos es la más eficiente para las características del set de datos a utilizar.

3.1 Preguntas de investigación

Esta tesis pretende dar respuesta a las siguientes preguntas:

1) ¿Es posible definir un marco de referencia que permita identificar o seleccionar a priori la combinación de algoritmos más eficientes para la obtención de patrones según las características del set de datos?

2) De ser posible ¿qué características del dominio son relevantes para predecir el comportamiento de la pareja de algoritmos?

3) En base a las características estudiadas ¿cuáles son las condiciones sobre las cuales cada combinación de algoritmos presenta mejores resultados?

En síntesis, el presente trabajo pretende sentar las bases del estudio que determine qué pareja de algoritmos (Clustering + inducción de reglas) es la más eficiente según las características del set de datos a analizar, con el objetivo futuro de establecer un mecanismo que permita predecir la mejor conformación del proceso de acuerdo al set de datos en análisis. Asimismo se pretende extender el estudio realizado en (Kogan, 2007; López-Nocera, 2012; Panchuk, 2015), para incluir dos estrategias de definición de inicial de centros para el algoritmo de clustering K-Means, los algoritmos de clustering DBSCAN, Meanshift y Birch y el algoritmo CART de inducción de reglas.

4. Solución propuesta

En este capítulo se presenta una propuesta de solución aplicada en forma experimental al problema delimitado en la sección 3. La formulación del diseño experimental se detalla en la sección 4.1, introduciendo las tres etapas que lo componen . En la sección 4.2 se proponen una serie de características del set de datos, que se utilizaran para tipificarlos y así estudiar el comportamiento de cada tipo. La sección 4.3 brinda los detalles de las tareas de la etapa 1, siendo estas ejecutadas automáticamente por el software de experimentación. Se detalla el proceso para la generación y validación de los set de datos artificiales en la sección 4.3.1, se discute el proceso de descubrimiento de grupos, con sus diferentes etapas y algoritmos en la sección 4.3.2, mientras que en la sección 4.3.3 se describe el cálculo de métricas internas para dicho proceso y la determinación del algoritmo ganador. En forma análoga, se presenta una descripción del proceso de inducción de reglas en la sección 4.3.4, mientras que en la sección 4.3.5 se discute el cálculo de las correspondientes métricas y la determinación del algoritmo ganador.

4.1 Formulación del Diseño Experimental

Para abordar el problema definido, se realizará un diseño experimental que permita comprender el comportamiento de los algoritmos de acuerdo a las características del set de datos, generando una base de conocimiento que permita posteriormente definir las parejas más eficientes.

Esto implica, en primera instancia, definir la forma mediante la cual se determinará la eficiencia del proceso. En (Feldman y Sanger, 2007) se identifican tres alternativas posibles:

- **Evaluación externa**: consiste en partir de un set de datos conocido y, aplicando diferentes parejas de algoritmos, encontrar cual es el que mejor describe la realidad. La limitación de esta estrategia es evidenciada por el objetivo del proceso de descubrimiento de grupos en sí: la búsqueda de patrones ocultos en el set de datos. No siempre se cuenta con una verdad conocida en base a la cual se pueden evaluar los distintos algoritmos.

- **Evaluación manual**: un humano experto analiza los resultados generados por cada pareja de algoritmos. La debilidad de esta estrategia radica en los tiempos/costes y la subjetividad de dichas revisiones manuales, especialmente si se considera que en muchos casos los resultados obtenidos son contra-intuitivos.

- **Evaluación Interna**: se caracteriza por definir métricas o índices que de alguna forma describan la calidad de los resultados obtenidos. Esta estrategia cuenta también con cierto grado de subjetividad: buenos valores de métricas no siempre significan alto valor de

calidad en los resultados obtenidos. Además, existe una tendencia de ciertos algoritmos a generar buenos valores de métricas relacionadas (Van Craenendonck y Blockeel, 2015).

A pesar de dichas limitaciones, la evaluación interna no requiere conocimientos previos de los patrones que se intentan extraer de los datos, y el cálculo y evaluación de métricas puede automatizarse. De esta forma, se puede aplicar a cualquier problema de descubrimiento de reglas de pertenencia a grupos.

Para mitigar la tendencia de ciertos algoritmos de clustering a puntuar mejor en ciertas métricas, se utilizaron cinco índices de diferente naturaleza, detallados en la sección 4.3.3. Como criterios adicionales, se incluyeron también métricas secundarias que permiten una segunda instancia de evaluación en caso de empate.

Para la evaluación de los algoritmos de inducción de reglas, se aplica una estrategia similar, aunque utilizando una única métrica primaria y una única secundaria, descriptas en la sección 4.3.5

4.1.1 Descripción de las etapas del experimento

El objetivo del experimento es asociar características de un set de datos con una pareja de algoritmos que retorne la mejor separación en grupos y las mejores reglas que describen esos grupos, en base a métricas internas. Una vez generada la base de conocimiento, que asocia características del set de datos con algoritmos, sería posible inferir el más conveniente para un set de datos dado con solo conocer dichas características. Además de ser más simple en cuanto a las operaciones que involucra, es deseable que el proceso de cómputo de las características de un set de datos sea considerablemente más rápido en tiempo de cómputo respecto a obtener el mejor algoritmo por fuerza bruta, es decir, probando las diferentes opciones.

Con los objetivos descriptos en el párrafo anterior, se proponen las siguientes etapas, mencionando las secciones del documento asociadas:

4.1.1.1 ETAPA A: Generación de datos experimentales

1) Generar un set de datos artificialmente con valores específicos para cada una de las características a estudiar (sección 4.3.1)
2) Verificar que se cumplan los valores establecidos para cada característica (sección 4.3.1)
3) Someterlo a un proceso de clustering utilizando un algoritmo W. (sección 4.3.2)
4) Registrar el tiempo incurrido en el punto anterior y calcular métricas internas en base a los grupos generados en [3] (sección 4.3.3)

Las etapas [3] y [4] se repiten para todos los algoritmos de clustering abarcados por este trabajo.

5) En base a las métricas internas, buscar el algoritmo de clustering con mejores valores en mayor cantidad de métricas (sección 4.3.3)
6) Someter a los grupos encontrados en [3] a un proceso de inducción de reglas, utilizando un algoritmo Y (Sección 4.3.4)
7) Registrar el tiempo incurrido en el punto anterior y calcular métricas internas en base a las reglas generadas en [7] (Sección 4.3.5)

   Las etapas [6] y [7] se repiten para todos los algoritmos de inducción de reglas abarcados por este trabajo.
8) En base a las métricas internas, buscar el algoritmo de inducción de reglas con mejores valores en mayor cantidad de métricas (Sección 4.3.5)

La secuencia anterior se repite para cada set de datos, tipificado de la A a la Y (tabla 1) de acuerdo con la combinación de características (descriptas en la sección 4.3.1).

Para el proceso descripto anteriormente, se desarrollara un programa software encargado generar, validar y evaluar sets de datos artificiales en forma autónoma (pasos 1 a 10).

4.1.1.2 ETAPA B: Análisis de datos experimentales

Una vez completada la secuencia de pasos descripta en la Etapa 1 para todos los tipos de set de datos y para todos los algoritmos incluidos en este estudio, se analiza la información generada en forma manual. La búsqueda apunta a encontrar tendencias que muestren una mejor performance de una pareja de algoritmos dada (en términos de las métricas internas y el tiempo de cómputo) para un tipo de set de datos particular. Así, sería posible extraer relaciones tales como:

*El algoritmo de clustering **C** y el de inducción de reglas **R** generaron mejores resultados, en base a métricas internas, para sets de datos del tipo **A,** en un porcentaje de casos **P**.*

Con conclusiones similares a la anterior podría ser posible predecir qué pareja de algoritmos utilizar para un set de datos con características similares, sin necesidad de incurrir en un análisis comparativo de algoritmos.

4.1.1.3 ETAPA C: Validación de las predicciones

La etapa final del proceso consisten en validar las relaciones obtenidas en la etapa anterior, en forma empírica, utilizando sets de datos reales. Los pasos a seguir en esta etapa son:

1) Obtención de sets de datos real, de carácter público

2) Aplicación de la etapa de análisis de características relevantes en forma automática.

3) Predicción de los algoritmos de clustering e inducción de reglas óptimo en base a los valores obtenidos en (3)

4) Aplicación de los pasos de la etapa 1 al set de datos real para obtener la pareja de algoritmos ganadora

5) Comparación entre los valores predichos y los obtenidos en (4)

4.2 Propuesta de características del set de datos a utilizar para el estudio

Como se propuso en la sección 3 y en la sección 4.1, es central a este estudio vincular características objetivas de un set de datos con algoritmos de clustering e inducción de reglas. Para ello es necesario definir dichas características a considerar, teniendo en cuenta que deben aplicar a cualquier tipo de set de datos de tipo matricial formado por atributos numéricos.

Es también fundamental poder obtener dichas características en forma automática, sin ningún conocimiento adicional acerca del set de datos y en un tiempo de cómputo menor al que significaría evaluar las diferentes combinaciones de algoritmos. Las características del set de datos a considerar en este estudio son:

- **Cantidad de atributos (CA):** cantidad de columnas o valores que presenta cada ejemplo del set de datos.

- **Porcentaje de ejemplos con tendencia lineal (PL)**: para determinar si a un ejemplo se lo considera dentro de un grupo lineal se calculará la distancia del mismo a la recta de ajuste obtenida por el método de descomposición en valores singulares (Golub, Reinsch 1970). La distancia umbral está definida como un porcentaje del módulo de la recta contenida entre los planos correspondientes a los máximos valores de cada coordenada.

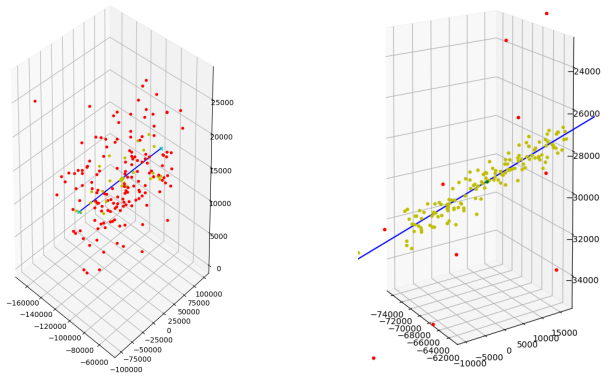

Fig 4.1. Ejemplos de set de datos con bajo (izq.) y alto (der.) porcentaje de ejemplos con tendencia lineal

- **Porcentaje de ejemplos repetidos (PR)**: cantidad de ejemplos similares, en términos de distancia que los separa con relación a otros ejemplos del set. Se consideran ejemplos repetidos a aquel grupo de ejemplos, mayor a un porcentaje del total de ejemplos (5% para este estudio), cuya distancia es menor ó igual a la mínima distancia no nula calculada entre todos los ejemplos del set de datos, más un porcentaje adicional (10% para este estudio)

- **Cantidad de grupos de ejemplos repetidos (GR)**: ligado directamente al punto anterior, esta característica indica cuántos grupos de ejemplos repetidos existen.

- **Porcentaje de outliers (PO)**: se consideran atípicos o outliers a aquellos ejemplos dentro del 20% más lejano al punto medio de la nube de puntos cuya distancia sea mayor a la distancia del ejemplo inmediatamente anterior a dicho 20%, multiplicado por un coeficiente. Considérese el siguiente ejemplo (Fig. 4): sobre un set de datos de 20 ejemplos, el 20% más lejano a **0** está constituido por los 4 puntos al final de la recta **D**. Para obtener el umbral **u**, se multiplica la distancia del ejemplo **p** (inmediato anterior al 20% más lejano) por un coeficiente. Los puntos cuya distancia sea mayor a ***u***, se consideran *outliers* (dos, en este caso).

Fig 4.2 Obtención de ejemplos atípicos en un set de datos de 20 elementos

4.3 Detalle de tareas en la etapa A

Como se mencionó en la sección 4.1.1.1, las tareas de esta etapa son ejecutadas en forma automática por un software experimental, desarrollado para esta tesis. En las siguientes secciones se brindan detalles acerca de la implementación de cada tarea de la etapa A en la solución software mencionada.

4.3.1 Generación y validación del set de datos

Los sets de datos para la etapa A son generados artificialmente y manipulados de forma que posean las características requeridas. La estructura es del tipo matricial, donde cada ejemplo del set está constituido por una cantidad fija de valores numéricos reales. La figura 4.3 ilustra la estructura del set de datos y sus características.

	A1	A2	A3	A4	…	An
E1	0,232	155	1,44	-6789	….	12,2
E2	3.54	-416	6.33	543	….	45
.						
.						
.						
Em	0.8	656	2.23	7853	….	98

Fig 4.3 Set de datos de *n* atributos por *m* ejemplos

La figura 4.4 presenta la estructura del subproceso de generación del set de datos.

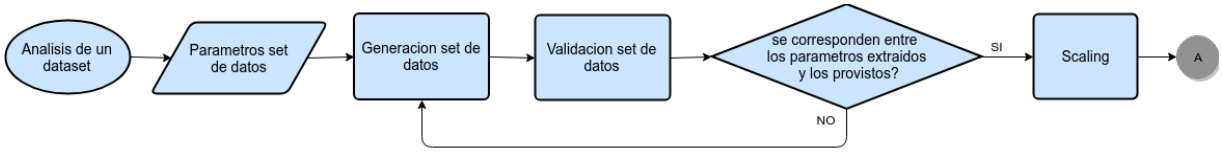

Fig 4.4. Diagrama de flujo subproceso de generación del set de datos

En cuanto a la distribución de los diferentes atributos, se optó por intercalar atributos con distribución normal y atributos uniformes, considerando que, en la práctica, los datos no son aleatorios y responden generalmente a distribuciones de probabilidad conocidas.

La generación de set de datos se realiza bajo las siguientes condiciones:

- Los sets de datos generados no pueden tener menos de 100 ejemplos

- Deberá existir al menos un porcentaje igual a (100 – `max_special_points_perc`) de valores aleatorios, es decir, que no correspondan a *outliers*, ni repetidos, ni lineales. Para la generación de datos experimentales, el porcentaje corresponde al 10%.

- Los atributos para los datos generados en forma aleatoria tendrán distribuciones normales o uniformes, en forma intercalada, comenzando con un atributo normal. A modo de ejemplo, para un set de datos de 5 atributos, las distribuciones de los mismos será la ilustrada en la digura 4.5.

Atributo 1	Atributo 2	Atributo 3	Atributo 4	Atributo 5
Normal	*Uniforme*	*Normal*	*Uniforme*	*Normal*

Fig 4.5. Detalle de la distribución de probabilidad de cada atributo para un set de datos de cinco atributos

- El porcentaje de *outliers* no puede superar el 20%.

- Los valores de los atributos oscilan entre `-value_limit` y `value_limit`. Para la generación de datos experimentales se utilizo `value_limit` = 100000.

Variando la composición de cada set en base a las características mencionadas en la sección 4.3, se obtienen los diferentes tipos de sets a estudiar (A-Y) . La tabla 1 ilustra las combinaciones estudiadas.

Tabla 4.1. Combinaciones de atributos utilizadas para la generación de datos.

Tipo	CA	CE	PL	PR	GR	PO
A	8	1000	N	N	0	N
B	8	1000	B	N	0	N
C	16	1000	B	N	0	N
D	24	1000	B	N	0	N
E	8	1000	M	N	0	N
F	8	1000	A	N	0	N
G	8	1000	N	B	2	N
H	8	1000	N	M	2	N
I	8	1000	N	A	2	N
J	8	1000	N	M	3	N
K	8	1000	N	A	3	N
L	8	1000	N	M	4	N
M	8	1000	N	A	4	N
N	8	1000	N	N	0	B
O	8	1000	N	N	0	M
P	8	1000	N	N	0	A
Q	8	1000	B	B	2	B
R	8	1000	B	B	2	M
S	8	1000	B	B	2	A
T	8	1000	B	A	2	B
U	8	1000	B	A	2	M
V	8	1000	B	A	2	A
W	8	1000	M	B	2	B
X	8	1000	M	B	2	M
Y	8	1000	M	B	2	A

Característica	B	M	A
PL	10%	40%	80%
PR	10%	20%	40%
PO	6%	12%	18%

Antes de someter al set de datos generado a los diferentes algoritmos, es necesario validar que las características solicitadas estén presentes utilizando el mismo método con el que luego se pretende caracterizar a los sets reales.

Si una o más características, deducidas por la rutina de análisis, difieren de las especificadas al generar el set de datos, se genera un set nuevo hasta que no haya discrepancias o se exceda una cantidad límite de intentos (100 intentos para este estudio). Cabe aclarar que la fase de análisis contempla un margen de error de +/- 5% para las métricas expresadas como porcentajes de la cantidad de ejemplos (PL, PR y PO).

Esta rutina de validación es el mismo que se utilizará posteriormente para predecir qué pareja de algoritmos es la más conveniente en base a las características del set de datos obtenidas.

Una vez que el set de datos fue validado, se lo somete a un proceso de scaling. El mismo consiste básicamente en estandarizar los atributos restando el valor medio y ajustando a varianza unitaria. El centrado y ajuste ocurre independientemente para cada atributo mediante el cómputo de las estadísticas pertinentes sobre los ejemplos del set de datos. La estandarización de atributos es una práctica común y permite la construcción de clusters con mejores formas (Berkhin, 2006).

Datos de análisis generados

Para un set de datos dado, se almacenan los parámetros provistos para su generación (atributos, cantidad de de ejemplos, porcentaje de ejemplos lineales, porcentaje de ejemplos repetidos, cantidad de grupos en caso de haber ejemplos repetidos, porcentaje de *outliers,* cantidad de atributos uniformes, cantidad de atributos standard, letra indicando el tipo de set de datos) y los resultados del proceso de validación (atributos, cantidad de de ejemplos, porcentaje de ejemplos lineales, porcentaje de ejemplos repetidos, cantidad de grupos en caso de haber ejemplos repetidos, porcentaje de *outliers*).

4.3.2 Descubrimiento de grupos

Una vez generado y validado el set de datos, se lo somete a los diferentes algoritmos de descubrimientos de grupo o clustering. La figura 4.6 ilustra el procedimiento aplicado.

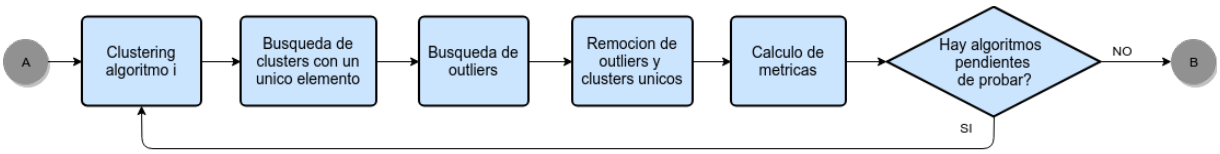

Fig 4.6. Diagrama de flujo subproceso de descubrimiento de grupos

Los algoritmos y variantes utilizados en este trabajo son:

1) K-Means en tres variantes de inicialización de centroides distintas:
   a) Definición de centros aleatoria,
   b) Variante k-means++ (Arthur, Vassilvitskii 2007)
   c) Determinación de centros mediante Análisis de componentes principales (Alrabea, Senthilkumar, Al-Shalabi, Bader 2013)
2) DBSCAN
3) Birch
4) Meanshift.

Para los algoritmos que no poseen una estrategia automática de configuración de sus parámetros, se utiliza una estrategia del tipo grid-search para determinar los valores de operación óptimos del estimador.

Dado que no todos operan de la misma forma, pueden darse situaciones donde el algoritmo ignore ciertos ejemplos por no poder asociarlos a un *cluster* o donde algunos de los clusters generados posea un único elemento. Estas dos situaciones se consideran indeseadas ya que se espera que ningún dato sea omitido en el análisis y un *cluster* de un único elemento no constituye un grupo en sí. Es por esto que tanto los grupos únicos como los ejemplos no clasificados son ignorados tanto en el cálculo de métricas como en la etapa de inducción de reglas, previo almacenamiento de la cantidad de ejemplos omitidos y el motivo, para ser considerado en la selección del algoritmo ganador, de ser necesario.

Si alguno de los algoritmos genere un único cluster, el mismo no es considerado para el cómputo de métricas, y en consecuencia, como candidato para este set de datos específico.

4.3.2.1 Determinación de centroides en K-Means y sus variantes

Para la determinación de la cantidad óptima de centroides a utilizar para los algoritmos K-Means y K-Means++, se ejecutan ocho iteraciones del proceso de clustering, incrementando el número de centroides desde 3 hasta 10. Para cada iteración, se computa la métrica suma de cuadrados y dicho valor se utiliza para determinar el valor óptimo del parámetro.

Para el caso de la determinación de los centros mediante análisis de componentes principales, se somete primero al set de datos a ocho iteraciones de un proceso de reducción de dimensiones con

número de componentes entre 3 y 10. Los ejes principales del espacio de atributos resultantes son pasados al estimador como centros iniciales.

4.3.2.2 Determinación de parámetros óptimos en DBSCAN

Este algoritmo requiere dos parámetros principales: epsilon y min_samples o cantidad mínima de ejemplos en el vecindario de un punto para considerarlo punto central o core. El valor de este último se configura como una fracción de la cantidad de ejemplos de set de datos y esa fracción esta especificada en el parámetro `dbs_min_samples_per_cluster_perc`. Para obtener los datos de análisis se utilizó un valor de `dbs_min_samples_per_cluster_perc = 0.1` indicando que min_samples se corresponderá con el 10% del número de ejemplos para el set de datos a procesar.

Para obtener el valor óptimo de epsilon, computa la matriz de vecinos más cercanos o Nearest Nieghbours, usando (min_samples -1) como número de vecinos. El vector de distancias obtenido se ordena y se toma el valor situado en la mitad del vector como valor de partida. Luego se ejecuta DBSCAN utilizando 20 valores de epsilon calculados como el valor de distancia medio multiplicado por el rango [1,01; 1,1] y luego dividido por el mismo rango. Si el valor medio es cero, se utiliza el valor de distancia mínimo mayor a cero.

4.3.2.3 Determinación de parámetros óptimos en Birch

Para la determinación del número de grupos óptimo en este algoritmo, se utiliza un proceso similar al utilizado para K-Means con la diferencia que se utilizan todas las métricas internas utilizadas para comparar los algoritmos.

Datos de análisis generados

Luego de que el set de datos es procesado por cada algoritmo, se almacena la siguiente información al respecto para cada set de datos, por cada algoritmo: cantidad de clusters generados, cantidad de clusters de un único elemento generados y la cantidad de ejemplos no considerados.

4.3.3 Cálculo de métricas y determinación del algoritmo de clustering ganador

Los grupos descubiertos por cada algoritmo y variante mencionados son puntuados utilizando las métricas siguientes métricas primarias:

a) Índice Davies-Bouldin
b) Índice Dunn
c) Índice Calinski-Harabasz

d) Índice silhouette
e) Suma de cuadrados.

Como métricas secundarias se utilizan: tiempo de cómputo, cantidad de clusters de un único elemento, y puntos descartados por el algoritmo. La figura 4.7 resume el proceso de determinación del algoritmo ganador.

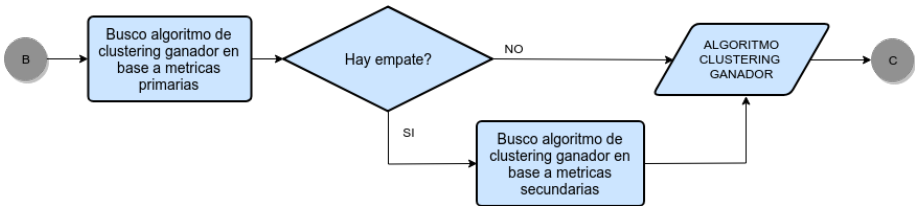

Fig 4.7. Diagrama de flujo subproceso determinación de algoritmo ganador (clustering)

Para obtener el algoritmo de descubrimiento de grupos ganador, se busca aquel que mejores valores tuvo para la mayor cantidad de métricas primarias. El valor de la métrica debe ser al menos un 5% mayor (o menor dependiendo de la misma) a la actual ganadora para tomar su lugar. Si existe un empate entre dos o más algoritmos, se utilizan las métricas secundarias.
Las valores de ejemplos ignorados y grupos de un único elemento se utilizan solo si algún algoritmo finalista incurrió en dichos comportamientos. En caso de que exista un empate nuevamente, se elige uno de los algoritmos ganadores al azar para proceder a la etapa de inducción de reglas, dejando registro de los finalistas.

Datos de análisis generados

Al concluir esta serie de tareas se almacena la siguiente información: valores de las métricas internas para cada uno de los algoritmos de clustering y el tiempo de ejecución del algoritmo. Para cada algoritmo finalista, se almacena también en que métricas superó a los demás finalistas, y si fue el ganador.

4.3.4 Inducción de reglas

En esta etapa se utilizan las etiquetas generadas por el algoritmo ganador para inducir reglas que describen las características de cada grupo. La figura 4.8 resume el subproceso de inducción de reglas de pertenencia a grupos. Los algoritmos de inducción de reglas utilizados para este trabajo son:

1) CART
2) CN2

Para la determinación del valor mínimo de ejemplos por hoja óptimo, se utiliza también una estrategia del tipo Grid-search, combinado con validación cruzada o cross-validation, para evitar sesgos producidos por la utilización del mismo set de datos tanto en el entrenamiento como para en la evaluación.

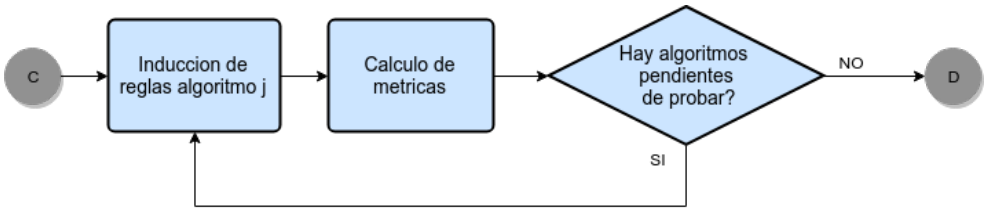

Fig 4.8. Diagrama de flujo subproceso inducción de reglas

El algoritmo CN2 devuelve directamente reglas vinculando uno o más valores de los atributos con el valor objetivo o clase (el identificador del cluster en este caso). El algoritmo CART en cambio, devuelve un objeto que representa un árbol de decisión, por lo que es necesario recorrerlo para obtener las reglas generadas. Los pasos para la extracción de reglas son los siguientes:

1) Buscar un nodo hoja
2) Buscar el nodo padre consultando los valores de `children_left` y `children_right` de los nodos
3) Extraer la condición del nodo padre mediante los atributos `feature` y `threshold`
4) Repetir el proceso hasta que el nodo 0 (raíz) sea alcanzado.

Los pasos anteriores se repiten para cada nodo hoja, hasta que todas las reglas son obtenidas.

Datos de análisis generados

Los datos almacenados para esta instancia del proceso son la cantidad de reglas generadas por cada algoritmo.

4.3.5 Cálculo de métricas y determinación del algoritmo de inducción ganador

Para determinar el algoritmo de inducción de reglas ganador se utiliza la curva ROC (Receiver Operating Characteristic o Característica Operativa del Receptor). En este trabajo, una vez calculados los puntos de la curva ROC, se computa el área bajo la misma utilizando la regla del trapecio. A mayor valor de área, mejor será la capacidad de las reglas generadas de clasificar los ejemplos correctamente (Bradley, A 1997). La figura 4.9 ilustra el flujo del subproceso determinación de algoritmo de inducción de reglas ganador.

El área bajo la curva ROC es la única métrica primaria utilizada para la evaluación de los algoritmos de inducción. El tiempo de cómputo será utilizado como métrica secundaria en caso de empate. Cabe aclarar que para esta métrica también se utiliza un margen ± 5% para considerar un valor como superior o inferior al del otro algoritmo.

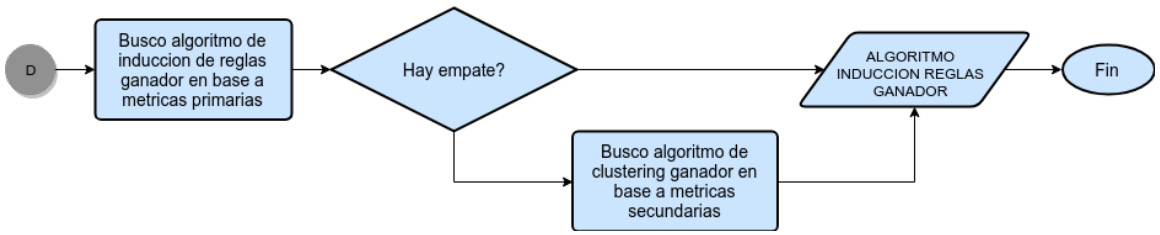

Fig 4.9. Diagrama de flujo subproceso determinación de algoritmo ganador (inducción de reglas)

Datos de análisis generados

En este punto se almacenan el tiempo de ejecución del algoritmo en cuestión, el valor del área bajo la curva ROC, cual fue el algoritmo ganador, y si hubo un empate.

5. Resultados experimentales obtenidos

Los resultados obtenidos de aplicar el proceso experimental, generados en base al análisis de 1300 sets de datos artificiales por cada tipo detallado en la tabla 4.1.
En esta sección se presentan los resultados obtenidos en la etapa intermedias de Clustering (sección 5.1) y de inducción de reglas (sección 5.2). Luego se describe la relación entre las métricas internas obtenidas y los resultados de los algoritmos ganadores (sección 5.3). Finalmente, en la sección 5.4, se presenta la validación de los resultados obtenidos en 11 set de datos reales.

5.1 Descripción de resultados para el proceso clustering

En la tabla 5.1 se puede observar un resumen de los resultados obtenidos para cada uno de los 25 tipos de set de datos, respecto a los algoritmos de clustering. Las filas resaltadas en gris corresponden a aquellos tipos que respondieron mejor al algoritmo ganador en más de un 70% de los casos.

En general puede observarse una baja cantidad de algoritmos ganadores por tiempo de cómputo, aunque para el tipo A y B (ver tabla 1), ocurre aproximadamente en el 20% de los caso, siendo estos el porcentajes los más alto del lote de datos. Específicamente, para los tipos de set de datos resaltados, se observa que en su mayoría los algoritmos se decidieron en base a la cantidad de métricas sobresalientes. Puede observarse también un porcentaje de empates de un dígito o menos para todos los sets de datos. En general, ni los clusters de un único elemento, ni los ejemplos ignorados fueron un factor relevante en la elección del algoritmo ganador, como puede inferirse de los valores de la columna %CUE y %IE.

De los resultados puede concluirse que para los tipos de datos resaltados en la tabla 5.1, existe una fuerte tendencia de los algoritmos ganadores a realizar una separación de los ejemplos en grupos de mejor calidad desde la perspectiva de las métricas internas utilizadas. Esto sugiere que el utilizar dicho algoritmo en un set de datos, con características similares a las tipificadas, tiene altas probabilidades de generar mejores grupos, comparado con los generados por los algoritmos restantes.

A continuación se describe el significado de cada columna indicada en la tabla 5.1:

- **Tipo**: corresponde a las combinaciones de atributos utilizadas para la generación del set de datos (tabla 4.1).
- **1er Alg**: indica cual es el algoritmo que obtuvo mejor rendimiento para cada tipo de set de datos generados .

- **% casos 1**: Porcentaje de set de datos que respondieron mejor al algoritmo ganador
- **2do Alg**: Segundo algoritmo en cantidad de set de datos mejor agrupados
- **% casos 2**: Porcentaje del set de datos correspondiente al segundo algoritmo
- **% mp**: Porcentaje de casos en los que el 1er algoritmo ganó en base a las métricas primarias
- **% tiempo**: Porcentaje de casos en los que el 1er algoritmo ganó por tiempo de cómputo
- **% CUE**: Porcentaje de casos en los que el 1er algoritmo ganó por haber generado menor cantidad de clusters de un único elemento
- **% EI**: Porcentaje de casos en los que el 1er algoritmo ganó por haber ignorado una menor cantidad de ejemplos
- **% emp**: Porcentaje del total de sets de datos analizado donde existió un empate entre dos o más algoritmos.

Tabla 5.1. Resultados de las ejecuciones para los algoritmos de descubrimiento de grupos.

Tipo	1er Alg	% casos 1	2do Alg	% casos 2	% mp	% tiempo	% CUE	% EI	% emp
A	kmeans_++	79,4	kmeans_random	11,1	79,8	20,2	0	0	0,14
B	kmeans_++	67,8	kmeans_random	14,8	79,9	20,1	0	0	0
C	kmeans_++	47,6	kmeans_random	33,6	97,7	2,3	0	0	0
D	kmeans_++	45,1	kmeans_random	33,1	97,5	2,5	0	0	0
E	meanshift	53	birch	31,8	100	0	0	0	0
F	dbscan	60,9	kmeans_++	17,9	100	0	0	0	2,07
G	kmeans_++	70,9	kmeans_random	11,6	84,9	14,8	0,3	0	0,74
H	kmeans_++	56,7	dbscan	29,7	86,9	13	0,1	0	0,59
I	dbscan	67,2	kmeans_++	26,9	99,9	0,1	0	0	0,14
J	kmeans_++	61,4	dbscan	20,6	84,1	11,7	4,2	0	5,79
K	dbscan	86,2	kmeans_++	10,3	100	0	0	0	0,44
L	kmeans_++	63	dbscan	15	83	11,3	5,7	0	8,76
M	dbscan	90,9	kmeans_++	7,5	100	0	0	0	6,53
N	meanshift	93,2	kmeans_++	3	100	0	0	0	0
O	meanshift	92.9	kmeans_++	3,6	100	0	0	0	0
P	meanshift	85.5	kmeans_++	7,1	100	0	0	0	0
Q	meanshift	89,0	birch	5,5	100	0	0	0	0
R	meanshift	87,1	kmeans_++	7,3	100	0	0	0	0,14
S	meanshift	82,1	kmeans_++	7,3	100	0	0	0	0,14
T	dbscan	99,9	meanshift	0,1	100	0	0	0	0
U	dbscan	99,6	meanshift	0,3	100	0	0	0	0,14
V	dbscan	99,7	meanshift	0,2	100	0	0	0	0
W	meanshift	54,7	birch	21,1	100	0	0	0	0
X	meanshift	55,2	birch	21,9	100	0	0	0	0,15
Y	meanshift	58,3	kmeans_++	19,9	100	0	0	0	0

5.2 Descripción de resultados para el proceso de inducción de reglas

La tabla 5.2 muestra los resultados para los algoritmos de inducción de reglas. En lo que a estos algoritmos respecta, CART supera a CN2 en entre el 93 y el 100 de los casos, dependiendo el tipo de set de datos. No obstante, la métrica decisiva en este caso es el tiempo de cómputo, ya que los valores de área bajo la curva ROC fueron muy similares para ambos algoritmos (diferencia menor al 5%). Los resultados indicarían que CART es superior a CN2 tanto en término de calidad de resultados como en tiempo cómputo, sin importar el tipo de set de datos o el algoritmo de clustering utilizado en la etapa anterior.

Tabla 5.2. Resultados de las ejecuciones para los algoritmos de inducción de reglas.

Tipo	Algoritmo ganador	% casos	% mp	% tiempo	% emp
A	cart	94.14	1.10	98.90	0
B	cart	97.11	0.61	99.39	0
C	cart	86.65	2.48	97.52	0
D	cart	82.57	4.67	95.33	0
E	cart	99.48	0	100.00	0
F	cart	99.85	0.30	99.70	0
G	cart	92.87	0.72	99.28	0
H	cart	93.39	0.40	99.60	0
I	cart	97.77	0.08	99.92	0
J	cart	92.80	0.32	99.68	0
K	cart	98.96	0.08	99.92	0
L	cart	93.76	0.32	99.68	0
M	cart	99.33	0.15	99.85	0
N	cart	99.48	0.07	99.93	0
O	cart	99.70	0.07	99.93	0
P	cart	98.74	0	100.00	0
Q	cart	100.00	0	100.00	0
R	cart	99.63	0.30	99.70	0
S	cart	99.78	0.15	99.85	0
T	cart	100.00	0	100.00	0
U	cart	100.00	0	100.00	0
V	cart	100.00	0	100.00	0
W	cart	99.85	0	100.00	0
X	cart	100.00	0.08	99.92	0
Y	cart	100.00	0.23	99.77	0

A continuación se describe el significado de cada columna indicada en la tabla 5.1:

- **% casos**: Porcentaje de set de datos que respondieron mejor al algoritmo ganador

- **% mp**: Porcentaje de casos en los que el algoritmo ganó en base a la métrica primaria
- **% tiempo**: Porcentaje de casos en los que el algoritmo ganó por tiempo de cómputo
- **% emp**: Porcentaje del total de sets de datos analizado donde existió un empate entre los dos algoritmos.

5.3 Relación entre métricas internas y algoritmos ganadores

A continuación puede observarse la relación entre métricas internas y algoritmos. Como se mencionó en la sección 4.6, el algoritmo ganador será aquel que posea mejores valores en mayor cantidad de métricas. Las tablas muestran el porcentaje de casos en los que cada algoritmo ganó con la combinación ganadora de las métricas marcadas con un rectángulo verde.

Tabla 5.3. Porcentaje de casos en los que el algoritmo K-Means con inicialización aleatoria de centros fue dado por ganador, para cada combinación de métricas

K-Means Random					
Sil	**C–H**	**Dunn**	**SdC**	**DV**	**% de casos**
■	■		■		34.00%
■				■	19.00%
■			■	■	8.00%
■	■				8.00%
■		■		■	5.00%
	■		■		5.00%
■			■		5.00%
■	■	■	■		3.00%
		■	■	■	3.00%
■	■	■			2.00%
	■	■	■		2.00%
			■	■	2.00%
■		■	■		2.00%
■		■	■	■	1.00%

Tabla 5.4. Porcentaje de casos en los que el algoritmo K-Means++ fue dado por ganador, para cada combinación de métricas

K-Means ++					
Sil	C–H	Dunn	SdC	DV	% de casos
■	■		■		21.00%
■	■		■	■	21.00%
■	■			■	18.00%
■			■	■	9.00%
■				■	8.00%
■	■				5.00%
	■		■	■	3.00%
■		■		■	3.00%
■	■	■	■		2.00%
■	■	■			2.00%
■	■	■		■	1.00%
		■	■	■	1.00%
	■			■	1.00%
	■		■		1.00%
■	■	■	■	■	1.00%
■			■		1.00%
■		■	■	■	1.00%

Tabla 5.5. Porcentaje de casos en los que el algoritmo K-Means con determinación de centros por análisis de componentes principales, fue dado por ganador, para cada combinación de métricas

Kmeans PCA					
Sil	C–H	Dunn	SdC	DV	% de casos
■	■		■		30.00%
■				■	23.00%
	■		■		14.00%
■	■				8.00%
■			■	■	6.00%
■		■		■	5.00%
■	■	■	■		2.00%
	■	■	■		2.00%
			■	■	2.00%
■			■		2.00%
■	■	■			1.00%
		■		■	1.00%
		■	■	■	1.00%
	■		■	■	1.00%
■		■	■	■	1.00%
	■			■	1.00%

Tabla 5.6. Porcentaje de casos en los que el algoritmo Meanshift fue dado por ganador, para cada combinación de métricas

Meanshift					
Sil	**C–H**	**Dunn**	**SdC**	**DV**	**% de casos**
■		■		■	88.00%
		■		■	12.00%

Tabla 5.7. Porcentaje de casos en los que el algoritmo Birch fue dado por ganador, para cada combinación de métricas

Birch					
Sil	**C–H**	**Dunn**	**SdC**	**DV**	**% de casos**
■		■			74.00%
■		■		■	14.00%
■		■	■		8.00%
		■	■		2.00%
		■		■	1.00%
■		■	■	■	1.00%
■	■		■		1.00%

Tabla 5.8. Porcentaje de casos en los que el algoritmo DBSCAN fue dado por ganador, para cada combinación de métricas

DBSCAN					
Sil	**C–H**	**Dunn**	**SdC**	**DV**	**% de casos**
■	■	■	■	■	57.00%
■	■	■		■	32.00%
■	■		■	■	3.00%
■		■		■	3.00%
■	■			■	3.00%
■		■	■	■	2.00%

***Referencias**: **Sil**: Silhouette; **C-H**: Calinski - Harabasz; **SdC**: Suma de cuadrados; **DV**: Davies-Bouldin*

5.4 Validación del método propuesto

En esta sección pretende validar los resultados generados durante la fase de experimentación y confirmar si es posible inferir los algoritmos que mejores resultados generaran, en término de las métricas internas seleccionadas, con solo conocer la cantidad de atributos, el porcentaje de ejemplos con tendencia lineal, la cantidad de ejemplos repetidos con la respectiva cantidad de grupos y la cantidad de outliers o puntos extraños.

Para esto se seleccionaron un grupo de set de datos reales, de acceso público, de forma de cubrir diferentes combinaciones de características del set de datos.

5.4.1 Selección de los set de datos reales para la validación

Los sets de datos reales elegidos poseen las siguientes características generales: la cantidad de atributos oscila entre 6 y 17, la cantidad de ejemplos fue limitada a 2000 para los sets de datos que la superaban y la distribución de los valores para cada atributo tiene una tendencia normal o uniforme en su mayoría con excepción del set contraceptive-5an-nn_a que se conservó por el número de ejemplos repetidos.

A continuación se detallan los sets de datos reales seleccionados, especificando sus atributos básicos y presentando un histograma con las distribución de valores de cada uno de sus atributos.

1) dataset_2175_kin8nm

Este dataset esta conformado por 9 atributos los cuales describen la cinemática directa de un brazo robot de 8 enlaces (theta 1 a 8) y la variable target (y). Entre las variantes existentes de este conjunto de datos, esta tiene la caracteristica de ser altamente no lineal y medianamente ruidosa. El set de datos se encuentra disponible en: https://www.openml.org/d/189. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.1):

- **Origen:** Mundo Real. Sensores
- **Cantidad de Instancias:** 8192
- **Cantidad de Atributos:** 9
- **Tipos de valores:** Reales

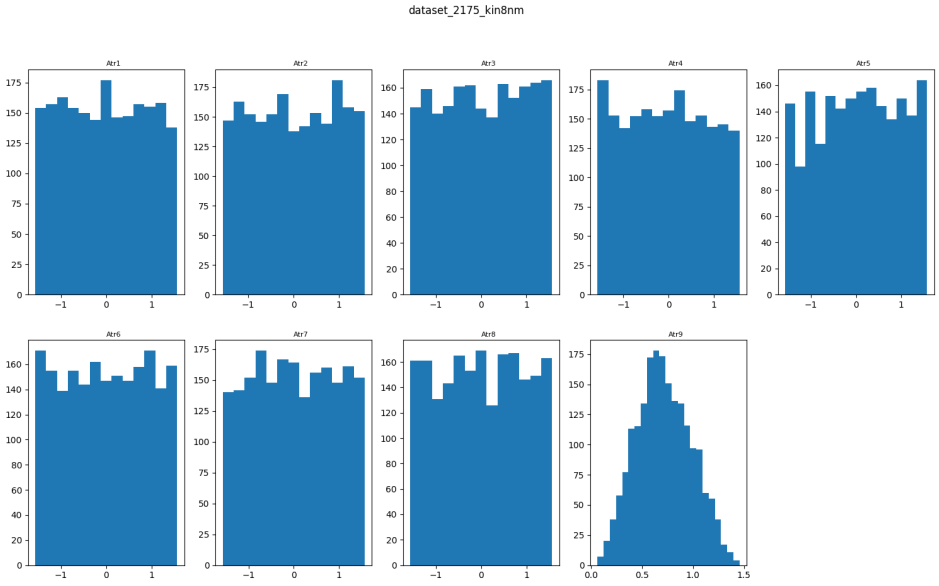

Fig 5.1. Distribución de los atributos para el set de datos dataset_2175_kin8nm

2) ColorTexture

Este dataset esta conformado por 17 atributos los cuales describen caracteristicas de las imagenes extraídas de una colección de imágenes del programa Corel. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset.php?cod=162. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.2):

- **Origen:** Mundo Real. Imágenes.
- **Cantidad de Instancias:** 68040
- **Cantidad de Atributos:** 17
- **Tipos de valores:** Reales y Enteros

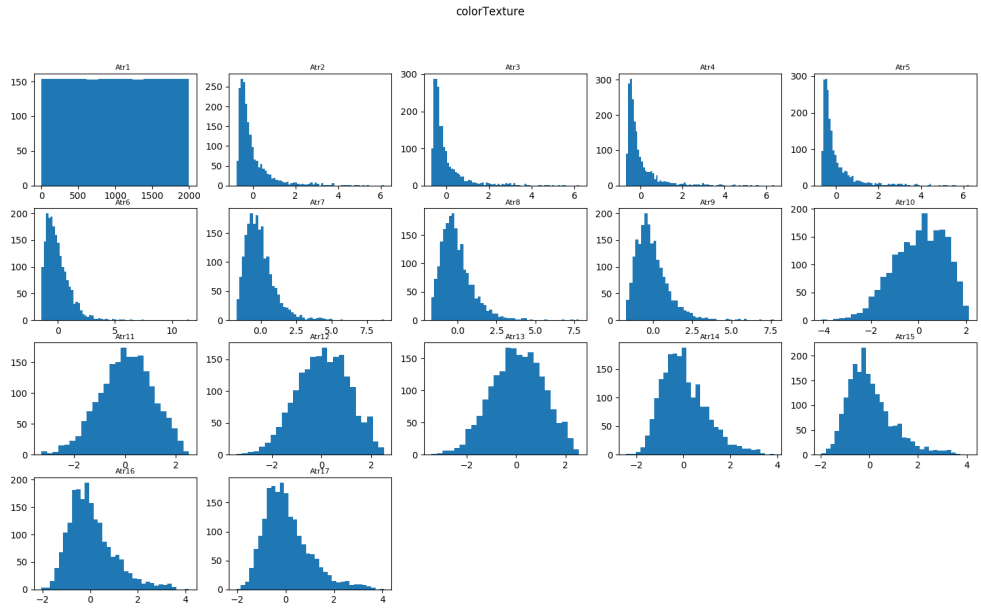

Fig 5.2. Distribución de los atributos para el set de datos ColorTexture

3) House16H

Este dataset esta conformado por 17 atributos los cuales describen características de las propiedades obtenidas mediante sensos con el objetivo de poder determinar el valor promedio de la propiedad. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset.php?cod=158. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.3):

- **Origen:** Mundo Real. Propiedades.
- **Cantidad de Instancias:** 22784

- **Cantidad de Atributos:** 17
- **Tipos de valores:** Reales y Enteros

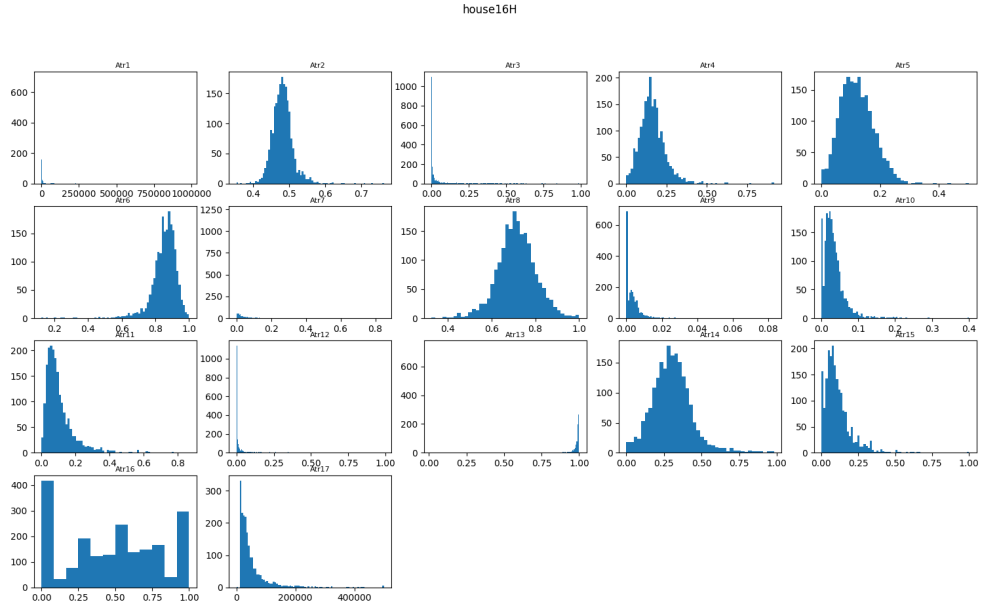

Fig 5.3. Distribución de los atributos para el set de datos House16H

4) NNGC1_dataset_F1_V1_002

Este dataset esta conformado por 6 atributos los cuales describen características datos del transporte incluyendo información del tráfico de autopistas, datos de tráfico de automóviles en túneles, tráfico en sistemas automáticos de pago en autopistas, tráfico de personas en sistemas de metro, vuelos de aeronaves nacionales, importaciones de embarcaciones, cruces fronterizos, flujos de tuberías y transporte ferroviario. Los datos contienen una serie temporal de frecuencia horaria. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset_smja.php?cod=947. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.4):

- **Origen:** Mundo Real. Transporte.
- **Cantidad de Instancias:** 1020
- **Cantidad de Atributos:** 6
- **Tipos de valores:** Reales

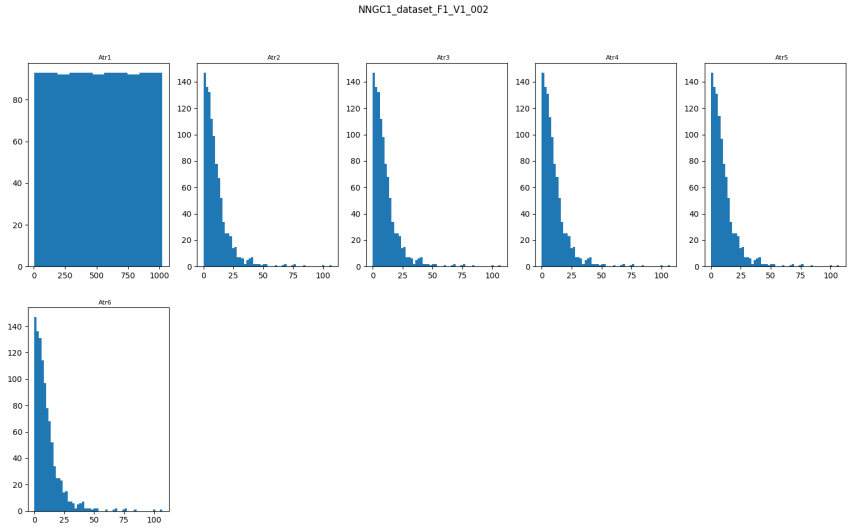

Fig 5.4. Distribución de los atributos para el set de datos NNGC1_dataset_F1_V1_002

5) yeast-5an-nn

Este dataset esta conformado por 8 atributos los cuales describen distintas características de las celulas de levadura con el objetivo de identificar distintios sitios de localización celular de proteínas. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset_smja.php?cod=595. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.5):

- **Origen:** Mundo Real. Biología.
- **Cantidad de Instancias:** 1484
- **Cantidad de Atributos:** 8
- **Tipos de valores:** Reales

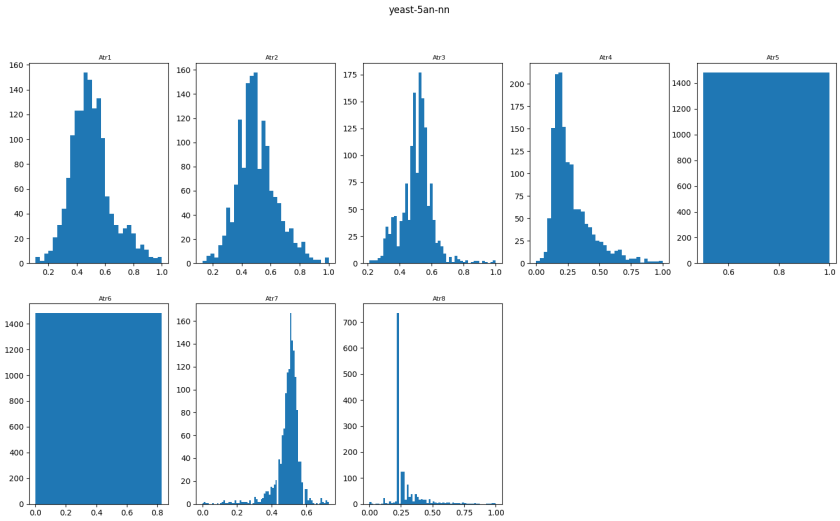

Fig 5.5. Distribución de los atributos para el set de datos yeast-5an-nn

6) DJIA

Este dataset esta conformado por 7 atributos los cuales describen la fluctuación de las acciones correspondientes a la empresa industrial Dow Jones. El primer atributo de este set de datos fue removido para el análisis ya que correspondía a la fecha de captura de cada ejemplo en formato texto. El set de datos se encuentra disponible en: https://introcs.cs.princeton.edu/java/data/DJIA.csv. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.6):

- **Origen:** Mundo Real. Finanzas.
- **Cantidad de Instancias:** 19450
- **Cantidad de Atributos:** 7
- **Tipos de valores:** Reales

7) Electricity_EBE

Este dataset esta conformado por 12 atributos los cuales describen mediciones de electricidad. El set de datos se encuentra disponible en: https://dataverse.harvard.edu/file.xhtml?persistentId=doi:10.7910/DVN/FIE0S4/H5A2CH. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.7):

- **Origen:** Mundo Real. Electrónica.
- **Cantidad de Instancias:** 1051200
- **Cantidad de Atributos:** 12

- Tipos de valores: Reales y Enteros

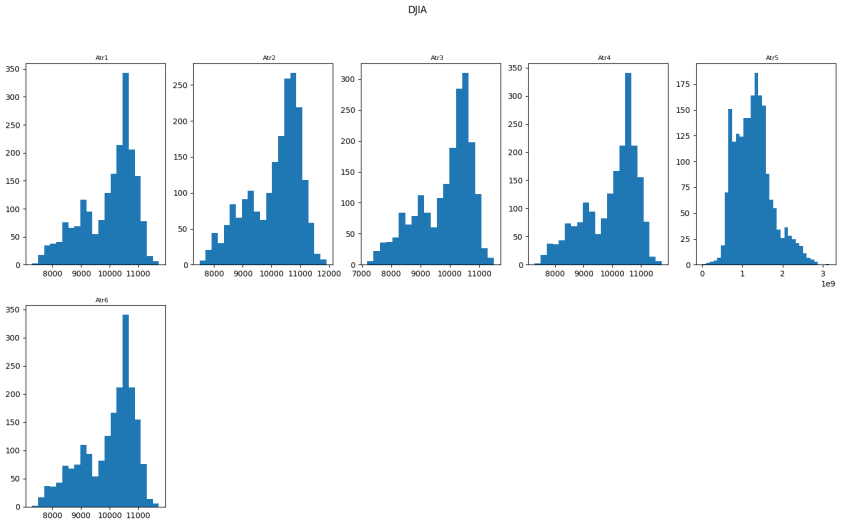

Fig 5.6. Distribución de los atributos para el set de datos DJIA

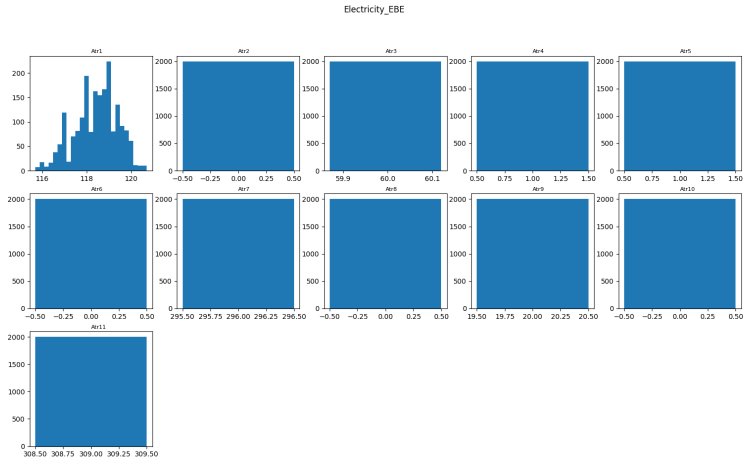

Fig 5.7. Distribución de los atributos para el set de datos Electricity_EBE

8) Acont_1_2000

Este dataset esta conformado por 6 atributos los cuales describen mediciones realizadas en un laboratorio experimental de física. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset_smja.php?cod=928. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.8):

- **Origen:** Mundo Real. Física.

- **Cantidad de Instancias:** 1995
- **Cantidad de Atributos:** 6
- **Tipos de valores**: Reales y Enteros

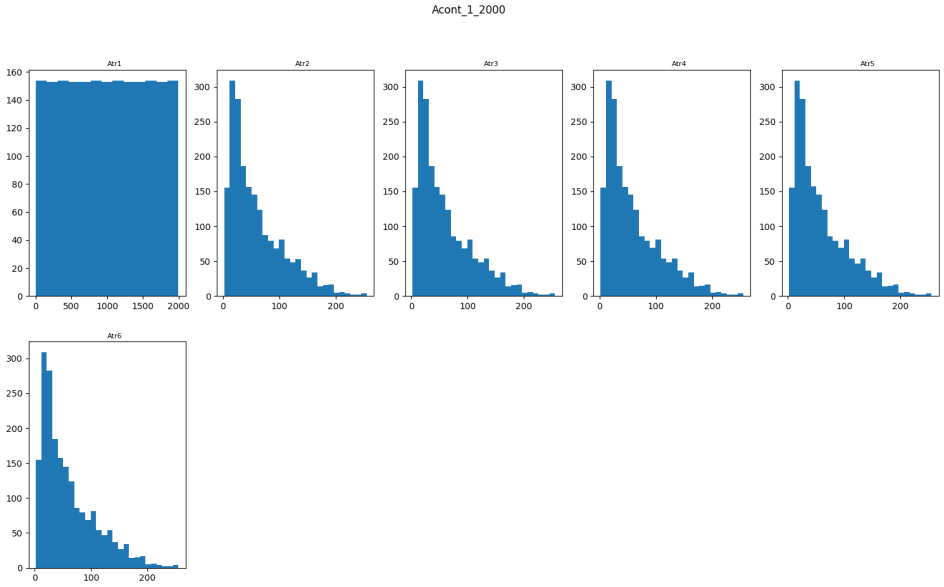

Fig 5.8. Distribución de los atributos para el set de datos Acont_1_2000

9) ColorMoments

Este dataset esta conformado por 10 atributos los cuales describen caracteristicas de los momentos de color de una imagen. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset.php?cod=161. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.9):

- **Origen:** Mundo Real. Imagen.
- **Cantidad de Instancias:** 68040
- **Cantidad de Atributos:** 10
- **Tipos de valores:** Reales y Enteros

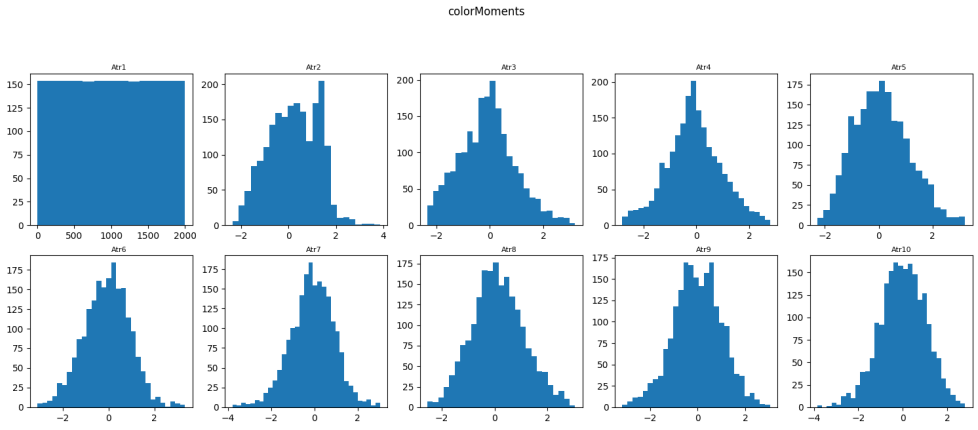

Fig 5.9. Distribución de los atributos para el set de datos ColorMoments

10) Edat_1_1661

Este dataset esta conformado por 4 atributos los cuales describe un conjunto de medidas de la curva de luz (variación de tiempo de la intensidad) de la estrella enana blanca variable PG1159-035 durante marzo de 1989. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset_smja.php?cod=932. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.10):

- **Origen:** Mundo Real. Astrofísica.
- **Cantidad de Instancias:** 1655
- **Cantidad de Atributos:** 4
- **Tipos de valores:** Reales.

11) contraceptive-5an-nn_a

Este dataset esta conformado por 4 atributos los cuales describen distintas características de la paciente con el objetivo de predecir el método anticonceptivo vigente. Solo se tomaron los atributos 3, 5, 6, 7, 9 y la clase. El set de datos se encuentra disponible en: http://sci2s.ugr.es/keel/dataset_smja.php?cod=565. A continuación se detallan sus principales caracteristicas y se presentan visualizaciones que describen la distribución de los datos (figura 5.11):

- **Origen:** Mundo Real. Salud.
- **Cantidad de Instancias:** 1473
- **Cantidad de Atributos:** 9
- **Tipos de valores:** Enteros

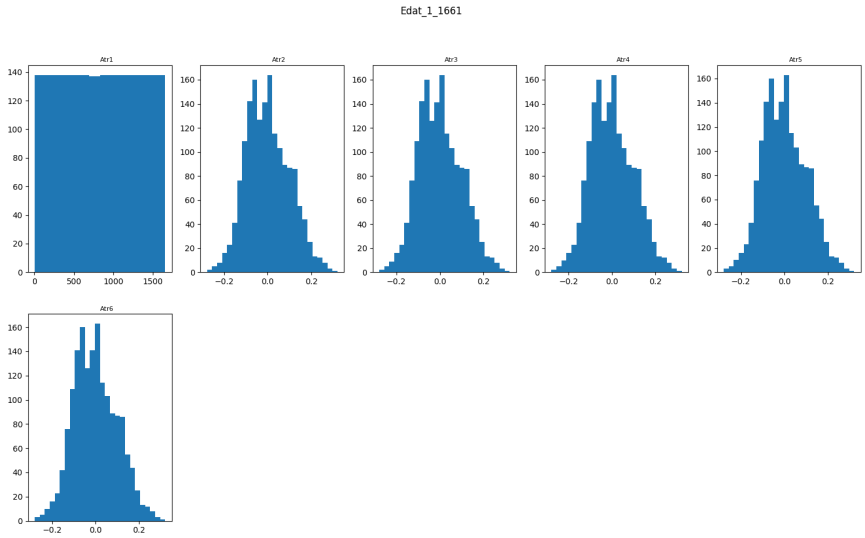

Fig 5.10. Distribución de los atributos para el set de datos Edat_1_1661

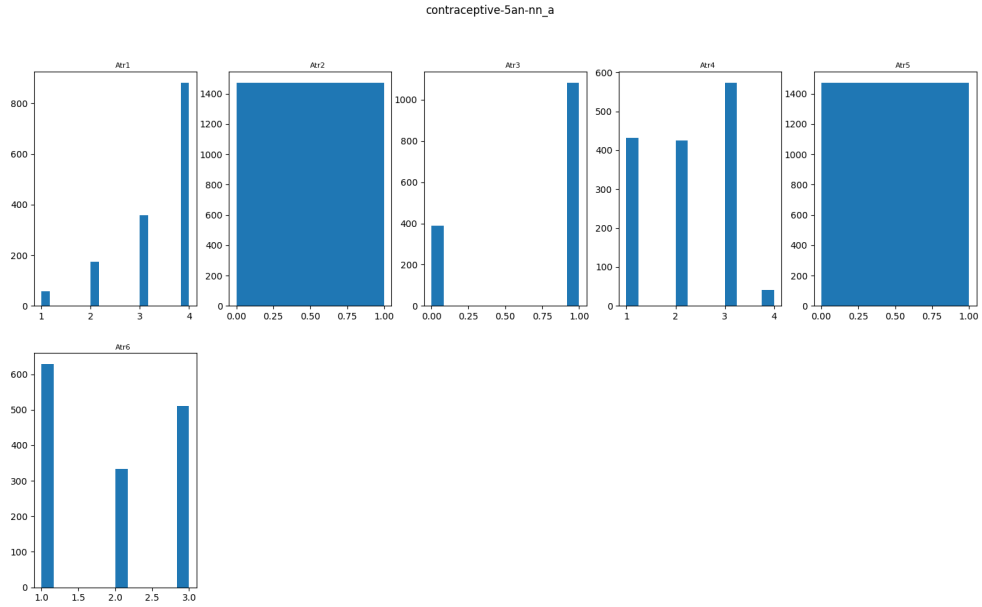

Fig 5.11. Distribución de los atributos para el set de datos contraceptive-5an-nn_a

5.4.2 Resultados de la validación

La siguiente tabla muestra las características del set de datos relevadas automáticamente por la rutina de análisis (Atrib, PL, PR, CG y PO). El tiempo insumido por la rutina de análisis se

observa en la columna Tc. Las columnas ACFB y AIRFB muestran el algoritmo de clustering y el algoritmo de inducción de reglas obtenidos por fuerza bruta, respectivamente. Por último, la columna Tfb muestra el tiempo insumido en la obtención de ambos algoritmos por fuerza bruta.

Tabla 5.9. Resumen de los resultados de la validación

	Set de datos	# ejem	Tc	Atrib	PL	PR	CG	PO	Tipo	ACFB	AIRFB	Tfb	RC	RIR
1	dataset_2175_kin8nm	2000	0.54	9	0	0	0	0	A	K-Means PCA	CART	54.82		
2	ColorTexture	2000	0.66	17	100	0	0	0	F	Meanshift	CART	27.94		
3	house16H	2000	0.53	17	54.65	0	0	11.15	X	Meanshift	CART	167.54		
4	NNGC1_dataset_F1_V1_002	1020	0.25	6	87.55	0	0	0	F	K-Means	CART	28.3		
5	yeast-5an-nn	1484	0.39	8	0	0	0	5.39	N	Birch	CART	27.25		
6	DJIA	2000	0.45	6	93.6	0	0	6.4	W	Birch / DBSCAN	CART	16.15		
7	Electricity_EBE	2000	0.41	11	95.2	0	0	3.2	W	Birch	CART	15.56		
8	Acont_1_2000	1995	0.47	6	11.58	0	0	0	B	K-Means++	CART	49.28		
9	ColorMoments	2000	0.55	10	100	0	0	0	F	K-Means++	CART	80.81		
10	Edat_1_1661	1655	0.43	6	100	0	0	0	F	K-Means++	CART	42.91		
11	contraceptive-5an-nn_a.csv	1473	0.48	6	0	65.10	8	1.77	M	K-Means++	CART	15.21		

La columna Tipo muestra la categoría correspondiente, en base a las métricas capturadas por la rutina de análisis. Las combinaciones de características que no se encuentran directamente representadas en los casos generados durante la experimentación, fueron asignadas al tipo más cercano.

Los colores utilizados en las columnas de resultados RC y RIR denotan la correspondencia entre los algoritmos predichos y los obtenidos por fuerza bruta. El color verde indica que el algoritmo obtenido por fuerza bruta corresponde con el ganador observado en la tabla 5.1. El color amarillo indica que el algoritmo obtenido por fuerza bruta se corresponde con el algoritmo en segunda posición, según la tabla antes mencionada. Por último el color rojo indica que el algoritmo obtenido no aparece ni en la primera ni en la segunda posición, para el tipo de set de datos atribuído.

6. Conclusiones, aportes y futuras líneas de investigación

En esta sección se analizan los resultados obtenidos y presentados en la sección anterior, detallando las conclusiones derivadas (sección 6.1), se listan las aportaciones realizadas como resultado del trabajo de investigación desarrollado en esta tesis (sección 6.2) y se delimitan las futuras líneas de trabajo (sección 6.3).

6.1 Conclusiones

En procesos de descubrimiento de reglas de pertenencia a grupos, los datos del dominio a estudiar son sometidos primero a un subproceso de separación en grupos o clustering, y luego, sobre los grupos obtenidos, un proceso de inducción de reglas. Este último, para descubrir características de los miembros de cada grupo obtenido, en términos de valores de los atributos.

Subproceso de descubrimiento de grupos

En base a los datos obtenidos durante la fase de experimentación, y según se puede observar en la tabla 5.1, existe una fuerte tendencia de ciertos algoritmos de clustering a generar buenos resultados relativos, en términos de las métricas internas utilizadas, para sets de datos artificiales tipo A, G, K, M, N, O, P, Q, R, S, T, U y V. Para estos tipos, los resultados son independientes de los valores de los atributos, mientras la distribución de dichos valores sea del tipo normal o uniforme.

En cuanto a la incidencia de la cantidad de atributos en los resultados, los valores para los sets de datos tipo B, C y D demuestran que existe un impacto negativo. El porcentaje de set de datos cubiertos por el algoritmo ganador se reduce en más de un 20% al duplicar el número de atributos, manteniendo las demás características constantes (variación entre B y C). Incrementando la cantidad de atributos en un 50% adicional, se observa una caída del 2% en los casos cubiertos. Por otro lado cabe destacar que los dos algoritmos en primer y segundo puesto se mantienen para los tres tipos de sets.

En cuanto a la generalización de estos resultados para cualquier tipo de set de datos que cumpla con las características del tipo en cuestión, sería prematuro confirmarlo en base a los resultados observados en la etapa de validación: de los 11 sets de datos reales evaluados, en tres oportunidades se predijo el algoritmo ganador, en 6 se predijo el algoritmo en segundo puesto, y en las dos restantes se obtuvo un algoritmo que no corresponde ni al ganador ni al segundo puesto. Por otro lado, cabe destacar que solo dos de los trece tipos presentaron alta correlación con un algoritmo fueron validados.

Las conclusiones derivadas en el párrafo anterior, se deben principalmente a la compleja tarea que es catalogar de manera precisa las variadas características de los set de datos en múltiples dominios. En este contexto, entendemos que un este trabajo sienta las bases para a partir de ello ampliar las variables a considerar para describir los set de datos, lo que permitiria generar datos experimentales más granulares, es decir, con mayor cantidad de rangos de variación de cada característica del set y cubriendo una mayor cantidad de combinaciones. Esto último también se ve dificultado por el proceso de generación de sets de datos, ya que ciertas combinaciones de porcentajes de ejemplos lineales, repetidos y outliers no eran correctamente detectadas por el proceso de análisis automático.

De las conclusiones desarrolladas en este trabajo, surge el interés por ampliar el estudio del impacto de la variación del número de atributos para todas las combinaciones de características del set de datos, asi como el impacto marginal de incrementar/reducir una característica específica. Entendemos que dicha ampliación permitiría comprender mejor las variaciones estructurales de los datos y por consiguiente obtener mejores resultados.

Subproceso de inducción de reglas

Los resultados obtenidos para esta etapa fueron concluyentes: el algoritmo CART superó al algortimo CN2 tanto durante la generación de datos experimentales como en la validación mediante set de datos reales. CART genera reglas de calidad similar a las que genera CN2, pero el tiempo promedio de inducción de reglas para este último esta en el orden de los 980ms, mientras que para CART esta en 2ms.

A partir de los resultados obtenidos y del análisis realizado, procedemos a continuación a responder las 3 preguntas de investigación planteadas en la sección 3:

1) ¿Es posible definir un marco de referencia que permita identificar o seleccionar a priori la combinación de algoritmos más eficientes para la obtención de patrones según las características del set de datos?

Los datos generados a partir de los experimentos demuestra una fuerte correlación entre la calidad de los grupos generados, en términos de métricas internas, y los algoritmos utilizados, para ciertos sets de datos, con características específicas. La cobertura de oscila entre el 45 y el 99,7% de los casos, para cada uno de los 25 tipos de set de datos estudiados. Desafortunadamente, los datos generados no son suficiente como para generalizar los resultados y aplicarlos a otros sets de datos con características similares, en forma consistente.

2) De ser posible ¿qué características del dominio son relevantes para predecir el

comportamiento de la pareja de algoritmos?

Se observó empíricamente el impacto negativo en la cobertura de casos al incrementar la **cantidad de atributos**, manteniendo las demás características constantes. Los sets de datos con porcentajes altos de **ejemplos repetidos** en 2 y 4 grupos respectivamente, mostraron muy buena cobertura para una pareja de algoritmos específica (87 y 90,4%). Los sets de datos con cantidades **bajas, medias y altas de outliers** presentaron una cobertura de casos por encima del 90%. Ciertas combinaciones de **cantidad de atributos, porcentaje de ejemplos lineales, porcentaje de repetidos y porcentaje de outliers**, generaron excelente respuesta por parte de parejas de algoritmos específicos (entre 82 y 99,8%). Como futura línea de investigación deberá estudiarse el impacto específico de cada característica, cuando se dan combinadas.

3) En base a las características estudiadas ¿cuáles son las condiciones sobre las cuales cada combinación de algoritmos presenta mejores resultados?

La respuesta a esta pregunta de investigación se responde por los resultados de la tabla 5.1 resaltados en gris y los resultados mostrados en la tabla 5.2.

6.2 Aportes

El aporte central de esta tesis consisten en sentar los bases de un método que permita identificar a priori que algoritmos de clustering e inducción de reglas son los más convenientes en términos de calidad relativa de los resultados generados y tiempo de cómputo, al momento de aplicar un proceso de descubrimiento de reglas de pertenencia a grupos. La calidad relativa se mide en términos de métricas internas y tiempo de ejecución de cada algoritmo, permitiendo el análisis automático.
La inducción de la pareja de algoritmos se realiza según una base de conocimiento generada en base a set de datos sintéticos, de características predefinidas. Esta base de conocimiento, vincula diferentes combinaciones de características de los sets de datos con los algoritmos que mejores resultados, en término de las métricas internas, generaron.

A partir de este método, se espera que los tiempos de desarrollo de la etapa de Modelado/Explotación de información se vean reducidos, siendo en la actualidad una de las fases que requiere mayor carga de trabajo (Cios, Kurgan 2005; Rodríguez et al., 2010).

Como aportes secundarios, reafirma en forma empírica la relación directa entre la performance de un algoritmo de clustering y las características del set de datos, y demuestra que en muchos casos, el mejor algoritmo en término de la calidad de los resultados generados depende de características del set de datos medibles a priori.

Se proponen también como características medibles el porcentaje de puntos con tendencia lineal, el porcentaje de puntos extraños o outliers y la cantidad de grupos de ejemplos particularmente cercanos o repetidos.

Por último, se presentan conclusiones respecto a la performance comparativa de los algoritmos CART y CN2 en términos de la calidad de las reglas generadas y el tiempo de cómputo, demostrando que el primero es más eficiente en la mayoría de los casos, generando calidad similar en tiempos menores.

6.3 Futuras líneas de trabajo

Durante el desarrollo del presente trabajo se detectaron los siguientes items como futuras líneas de trabajo:

- Extender el estudio realizado en cantidad y mayor cantidad de algoritmos pertenecientes a los tipos de algoritmos utilizados, así como ampliar las categorías.

- Ampliar el estudio introduciendo nuevas características del set de datos a vincular con los diferentes algoritmos, ampliando las posibilidades de represetar los distintos patrones en los set de datos y posiblemente los resultados obtenidos.

- Generar datos experimentales mas granulares, es decir, con mayor cantidad de rangos de variación para cada característica del set y cubriendo una mayor cantidad de combinaciones.

- Estudiar en profundidad la incidencia de la cantidad de atributos en la cantidad de casos cubiertos por un algoritmo. Incluir en el experimento variaciones en la cantidad de atributos para cada combinación de características considerada.

- Estudiar en profundidad la incidencia de la distribución de los atributos en la performance de los algoritmos de clustering

7. Anexos

7.1 Detalles del software de experimentación

El software utilizado para automatizar la generación, validación y evaluación de los sets de datos artificiales, así como la captura de parámetros de los sets de datos reales, esta escrito

integramente en el lenguaje Python version 3.5. La librería principal de aprendizaje automático utilizada es Scikit-learn (http://scikit-learn.org/stable/). También se utilizó la implementación del algoritmo CN2 provista por la librería Orange3 (https://orange.biolab.si/)

Para la construcción y manejo de matrices, se utilizó fuertemente la librería numpy (http://www.numpy.org/)

El código fuente de la solución esta publicado en el siguiente repositorio de Github (https://github.com/gabocic/python/tree/master/ML) y es de acceso público.

7.2 Librerías requeridas

Para el correcto funcionamiento del software es necesario, no solo contar con la versión 3.5 de Python, sino con las siguientes dependencias:

- wheel
- sklearn
- orange3
- cryptography
- matplotlib

Si se ejecuta la aplicación en un entorno Linux, los siguientes paquetes son necesarios para que las dependencias Python compilen correctamente:

- python3-dev
- libssl-dev
- python3-tk

7.3 Detalle de las tablas utilizadas para la base de conocimiento

A continuación se proveen un detalle de las tablas utilizadas para almacenar los resultados de los experimentos:

```
CREATE TABLE `clustering_metric` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `dataset_id` bigint(20) DEFAULT NULL,
```

```
  `algorithm` varchar(15) DEFAULT NULL,
  `total_clusters` int(11) DEFAULT NULL,
  `single_element_clusters` int(11) DEFAULT NULL,
  `samples_not_considered` bigint(20) DEFAULT NULL,
  `elap_time` decimal(19,4) DEFAULT NULL,
  `silhouette_score` decimal(19,4) DEFAULT NULL,
  `calinski_harabaz_score` decimal(19,4) DEFAULT NULL,
  `wb_index` decimal(19,4) DEFAULT NULL,
  `dunn_index` decimal(19,4) DEFAULT NULL,
  `davies_bouldin_score` decimal(19,4) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `dataset` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `run_id` bigint(20) DEFAULT NULL,
  `total_samples` bigint(20) DEFAULT NULL,
  `features` int(11) DEFAULT NULL,
  `linear_samples_perc` decimal(4,1) DEFAULT NULL,
  `repeated_samples_perc` decimal(4,1) DEFAULT NULL,
  `group_number` smallint(6) DEFAULT NULL,
  `outliers_perc` decimal(4,1) DEFAULT NULL,
  `uniform_features` int(11) DEFAULT NULL,
  `standard_features` int(11) DEFAULT NULL,
  `winner_clus_alg` varchar(15) DEFAULT NULL,
  `winner_ri_alg` varchar(15) DEFAULT NULL,
  `type` char(1) DEFAULT NULL,
  `winner_ri_alg_by` varchar(4) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `dataset_clus_finalists` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `dataset_id` bigint(20) DEFAULT NULL,
  `algorithm` varchar(15) DEFAULT NULL,
  `silhouette` tinyint(1) DEFAULT NULL,
  `calinski_harabaz` tinyint(1) DEFAULT NULL,
  `dunn` tinyint(1) DEFAULT NULL,
  `wb` tinyint(1) DEFAULT NULL,
  `davies_bouldin` tinyint(1) DEFAULT NULL,
  `time` tinyint(1) DEFAULT NULL,
  `sin_ele_clus` tinyint(1) DEFAULT NULL,
  `ignored_samples` tinyint(1) DEFAULT NULL,
  `winner` varchar(3) DEFAULT 'NO',
  PRIMARY KEY (`id`),
  KEY `dsid_algo` (`dataset_id`,`algorithm`)
)

CREATE TABLE `dataset_validation` (
```

```
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `run_id` bigint(20) DEFAULT NULL,
  `total_samples` bigint(20) DEFAULT NULL,
  `features` int(11) DEFAULT NULL,
  `linear_samples_perc` decimal(4,1) DEFAULT NULL,
  `repeated_samples_perc` decimal(4,1) DEFAULT NULL,
  `group_number` smallint(6) DEFAULT NULL,
  `outliers_perc` decimal(4,1) DEFAULT NULL,
  `outliersbyperp_perc` decimal(4,1) DEFAULT NULL,
  `dataset_id` bigint(20) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `rule_ind_metric` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `dataset_id` bigint(20) DEFAULT NULL,
  `clustering_metric_id` bigint(20) DEFAULT NULL,
  `algorithm` char(6) DEFAULT NULL,
  `total_rules` int(11) DEFAULT NULL,
  `elap_time` decimal(11,4) DEFAULT NULL,
  `auc` decimal(11,4) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `run` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `start_date` datetime DEFAULT NULL,
  `end_time` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
)
```

8. Referencias bibliográficas

Abran, A., Moore, J. W., Bourque, P., Dupuis, R., Tripp, L. 2004. Guide to the Software Engineering Body of Knowledge (2004 version). IEEE Computer Society Press. ISBN 0-7695-2330-7.

Almana, A. M., & Aksoy, M. (2014). An overview of inductive learning algorithms. International Journal of Computer Applications, 88(4).

Alrabea, A., Senthilkumar, A. V., Al-Shalabi, H., & Bader, A. (2013). Enhancing k-means algorithm with initial cluster centers derived from data partitioning along the data axis with PCA. Journal of Advances in Computer Networks, 1(2), 137-142.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I. (2013). An extensive comparative study of cluster validity indices. Pattern Recognition, 46(1), 243-256.

Argimón Pallás, J. M., & Jiménez Villa, J. (2004). Métodos de investigación clínica y epidemiológica. Métodos de investigación clínica y epidemiológica.

Arthur, D., Vassilvitskii S. (2007). k-means++: The advantages of careful seeding. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics

B. Kim, D. Landgrebe (1991) Hierarchical classifier design in high-dimensional numerous class cases. IEEE Trans. Geosci. Remote Sens. 29(4), 518–528

Barros et al. (2015), Automatic Design of Decision-Tree Induction Algorithms, SpringerBriefs in Computer Science, DOI 10.1007/978-3-319-14231-9_2

Basili, V. (1993). The experimental paradigm in software engineering. Experimental Software Engineering Issues: Critical Assessment and Future Directions, 1-12.

Basso, D. (2014). Propuesta de métricas para proyectos de explotación de información. Revista Latinoamericana de Ingeniería de Software, 2(4), 157-218.

Berkhin, P. (2006). A survey of clustering data mining techniques. I Grouping multidimensional data (pp. 25-71). Springer, Berlin, Heidelberg.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern recognition, 30(7), 1145-1159.

Britos, P. V. (2008). Procesos de explotación de información basados en sistemas inteligentes (disertación doctoral, Facultad de Informática, UNLP). http://sedici.unlp.edu.ar/bitstream/handle/10915/4142/Documento_completo.pdf?sequence=1

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C. & Wirth, R. (2000). CRISP-DM 1.0 Step-by-step data mining guide (). The CRISP-DM consortium

Chen, M., Han, J. y Yu, P. (1996) Data mining: An overview from database perspective. IEEE Transactions on Knowledge and Data Engineering

Cios, K. J., & Kurgan, L. A. (2005). Trends in data mining and knowledge discovery. In Advanced techniques in knowledge discovery and data mining (pp. 1-26). Springer London.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. Machine learning, 3(4), 261-283.

Cogliati, M., Britos, P. y García Martínez, R. (2006) Patterns in Temporal Series of Meteorological Variables Using SOM & TDITD. Lecture Notes in Artificial Intelligence, Springer Verlag

Creswell, J. W. (2002). Educational research: Planning, conducting, and evaluating quantitative (pp. 146-166). Upper Saddle River, NJ: Prentice Hall.

D. Page, S. Ray, Skewing: An efficient alternative to lookahead for decision tree induction, in 18th International Joint Conference on Artificial Intelligence (Morgan Kaufmann Publishers Inc., San Francisco, 2003), pp. 601–607

D. Sculley (2010). Web Scale K-Means clustering, Proceedings of the 19th international conference on World wide web

E. Alpaydin, Introduction to Machine Learning (2010). ISBN: 026201243X, 9780262012430

E.B. Hunt, J. Marin, P.J. Stone (1966), Experiments in Induction (Academic Press, New York)

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In Kdd (Vol. 96, No. 34, pp. 226-231).

Evangelos, S. y Han, J. (1996) Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. Portland, EEUU

Fawcett, T. (2006). An introduction to ROC analysis. Pattern recognition letters, 27(8), 861-874. http://people.inf.elte.hu/kiss/13dwhdm/roc.pdf

Fayyad U.,Piatetsky-Shapiro G., Smyth P. (1996). Knowledge Discovery and Data Mining: Towards a Unifying Framework. AI Magazine, 17(3)

Feldman, R., & Sanger, J. (2007). The text mining handbook: advanced approaches in analyzing unstructured data. Cambridge university press.

Felgaer, P., Britos, P. y García Martínez, R. (2006) Prediction in Health Domain Using Bayesian Network Optimization Based on Induction Learning Techniques. International Journal of Modern Physics, 17(3-C), 447-455, ISSN 0129-1831

G. Landeweerd et al. (1983), Binary tree versus single level tree classification of white blood cells. Pattern Recognit. 16(6), 571–577

Gaber, M. M., Zaslavsky, A., & Krishnaswamy, S. (2010). Data stream mining. Data Mining and Knowledge Discovery Handbook, 759-787.

García-Martínez, R., Britos, P., & Rodríguez, D. (2013, Junio). Information mining processes based on intelligent systems. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 402-410). Springer Berlin Heidelberg.

García-Martínez, R., Britos, P., Martins, S., & Baldizzoni, E. (2015). Explotación de Información. Ingeniería de Proyectos. Editorial Nueva Librería ISBN, 978-987.

García-Martínez, R., Britos, P., Pesado, P., Bertone, R., Pollo-Cattaneo, F., Rodríguez, D., Pytel, P., Vanrell. J. (2011). Towards an Information Mining Engineering. En Software Engineering, Methods, Modeling and Teaching. Sello Editorial Universidad de Medellín. ISBN 978-958-8692-32-6. Páginas 83-99.

García-Martínez, R., Diez, E., García, R., Martins, S., Baldizzoni, E. (2015) Modelos de Proceso para Ingeniería de Explotación de Información para Pymes: Abordaje Ágil y Abordaje Robusto. Proceedings XVII Workshop de Investigadores en Ciencias de la Computación, ISBN 978-987-633-134-0.

Golub, G. H.; Reinsch, C. (1970). "Singular value decomposition and least squares solutions". Numerische Mathematik. 14 (5): 403–420. doi:10.1007/BF02163027. MR 1553974.

Gopal, R., Marsden, J. R., & Vanthienen, J. (2011). Information mining—Reflections on recent advancements and the road ahead in data, text, and media mining.

Goswami S., Chakrabarti A., Chakraborti B. (2016) A Proposal for Recommendation of Features Selection Algorithm based on Data Set Characteristics

Grosser, H., Britos, P. y García Martínez, R. (2005) Detecting Fraud in Mobile Telephony Using Neural Networks. Lecture Notes in Artificial Intelligence, 3533: 613-615. Springer-Verlag

Hailesilassie, T. (2016). Rule extraction algorithm for deep neural networks: A review. arXiv preprint arXiv:1610.05267.

Hall, M. y Holmes, G. (2003) Benchmarking Attribute Selection Techniques for Discrete Class Data Mining. IEEE Transactions on Knowledge and Data Engineering, Vol. 15, Tomo 6, páginas 1437-1447

Holsheimer, M. y Siebes, A. (1991) Data Mining: The Search for Knowledge in Databases. Report CS-R9406, ISSN 0169-118X, Amsterdam, The Netherlands

Hsu, W., Lee, M. L., & Zhang, J. (2002). Image mining: Trends and developments. Journal of intelligent information systems, 19(1), 7-23.

K. Bennett, Global tree optimization: a non-greedy decision tree algorithm. Comput. Sci. Stat. 26, 156–160 (1994)

Kaski, S. (1997) Data exploration using self-organizing maps. Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series No. 82, 57 pp. Published by the Finnish Academy of Technology. ISSN 1238-9803

Kdnuggets. 2014. What main methodology are you using for your analytics, data mining, or data

science projects? Poll (Oct 2014).http://www.kdnuggets.com/

Kogan, A. (2007). Integración de Algoritmos de Inducción y Agrupamiento. Estudio del Comportamiento (Doctoral dissertation, Tesis de Ingeniería Informática. Laboratorio de Sistemas Inteligentes. Facultad de Ingeniería. Universidad de Buenos Aires).

Kosala, R., & Blockeel, H. (2000). Web mining research: A survey. ACM Sigkdd Explorations Newsletter, 2(1), 1-15.

Kruse, R., & Borgelt, C. (2003). Information mining. International Journal of Approximate Reasoning, 32(2), 63-66.

Leo, B., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees. Wadsworth International Group.

Lopez-Nocera M. (2012). Descubrimiento De Conocimiento Mediante La Integración De Algoritmos De Explotación De La Información. http://sistemas.unla.edu.ar/sistemas/gisi/tesis/lopez-nocera-tesisdemagister.pdf

Lyman, P., Varian, H., Dunn, J., Strygin, A., & Swearingen, K. (2000). How much information? school of information management and systems. Univ. of California at Berkeley.

M. Dong, R. Kothari, Look-ahead based fuzzy decision tree induction. IEEE Trans. Fuzzy Syst. 9(3), 461–468 (2001)

Maimon, O., & Rokach, L. (2010). Data Mining and Knowledge Discovery Handbook (2nd ed.). Springer US. doi:10.1007/978-0-387-09823-4

Mannila, H. (1997) Methods and problems in data mining. In Proc. of International Conference on Database Theory. Delphi, Greece

Martins S. (2013) Derivación del proceso de Explotación de Información desde el Modelado del Negocio. Departamento de Desarrollo Productivo y Tecnológico, Universidad Nacional de Lanús.

Martins, S., Pesado, P., & García-Martínez, R. (2016, Agosto). Intelligent Systems in Modeling Phase of Information Mining Development Process. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 3-15). Springer International Publishing.

Martins, S., Rodríguez, D., & García-Martínez, R. (2014, Junio). Deriving processes of information mining based on semantic nets and frames. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 150-159). Springer International Publishing.

Michalski, R. S. (1983) A Theory and Methodology of Inductive Learning. Artificial Intelligence, vol. 20, páginas 111-161

Moss, L. T. (2003). Nontechnical Infrastructure for BI Applications. DM REVIEW, 13, 42-45.

Nasreen, S., Azam, M. A., Shehzad, K., Naeem, U., & Ghazanfar, M. A. (2014). Frequent Pattern

Mining Algorithms for Finding Associated Frequent Patterns for Data Streams: A Survey. Procedia Computer Science, 37, 109-116.

Osama Abu Abbas " Comaprison between Data Clustering Algorithms" The International Arab Journal of Information Technology, Volume 5, July 2008.

P. Chou, Optimal partitioning for classification and regression trees. IEEE Trans. Pattern Anal. Mach. Intell. 13(4), 340–354 (1991)

P.E. Utgoff, N.C. Berkman, J.A. Clouse, Decision tree induction based on efficient tree restructuring. Mach. Learn. 29(1), 5–44 (1997)

Pal, N. R., & Biswas, J. (1997). Cluster validation using graph theoretic concepts. Pattern Recognition, 30(6), 847-857.

Panchuk J. (2015) Comportamiento De Integración De Algoritmos Para Descubrimiento De Reglas De Pertenencia A Grupos. http://sistemas.unla.edu.ar/sistemas/gisi/TFLS/Panchuk-TFL.pdf

Piatetski-Shapiro, G., Frawley, W.J. y Matheus, C.J. (1991) Knowledge discovery in databases: an overview. AAAI-MIT Press, Menlo Park, California.

Piatetsky-Shapiro, G. (1996). Advances in knowledge discovery and data mining (Vol. 21). U. M. Fayyad, P. Smyth, & R. Uthurusamy (Eds.). Menlo Park: AAAI press.

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Pyle, D. (2003). Business modeling and data mining. Morgan Kaufmann.

Pytel, P., Hossian, A., Britos, P., & García-Martínez, R. (2015). Feasibility and effort estimation models for medium and small size information mining projects. Information Systems, 47, 1-14.

Quinlan, J. R. (2014). C4. 5: programs for machine learning. Elsevier.

R.C. Barros et al., A survey of evolutionary algorithms for decision-tree induction. IEEE Trans. Syst. Man, Cybern. Part C: Appl. Rev. 42(3), 291–312 (2012)

Riveros, H., & Rosas, L. (1985). El Método Científico Aplicado a las Ciencias Experimentales. Editorial Trillas. México. ISBN 96-8243-893-4.

Rodríguez, D., Pollo Cattaneo, M. F., Britos, P. V., & García Martínez, R. (2010). Estimación Empírica de Carga de Trabajo en Proyectos de Explotación de Información. In XVI Congreso Argentino de Ciencias de la Computación.

Rudin, K., & Cressy, D. (2003). Will the Real Analytic Application Please Stand Up?. Dm Review, 13, 30-41.

S. Esmeir, S. Markovitch, Anytime learning of decision trees. J. Mach. Learn. Res. 8, 891–933 (2007)

S.K. Murthy, S. Salzberg, Lookahead and pathology in decision tree induction, in 14th International Joint Conference on Artificial Intelligence. (Morgan Kaufmann, San Francisco, 1995), pp. 1025–1031

S.W. Norton, Generating better decision trees, 11th International Joint Conference on Artificial Intelligence (Morgan Kaufmann Publishers Inc., San Francisco, 1989)

Sábato, J.,Mackenzie, M. (1982). La Producción de Tecnología. Editorial Nueva Imagen. México. ISBN 968- 429-348-8.

Sehgal, G., & Garg, D. K. (2014). Comparison of Various Clustering Algorithms. International Journal of Computer Science and Information Technologies, 5(3), 3074-307.

Smith K. A., Woo F., Ciesielske V., Ibrahim R. (2002) Matching Data Mining Algorithm Suitability to Data Characteristics Using a Self-Organizing Map

T. M. Mitchell, "Decision Tree Learning," in Machine Learning, Singapore: McGraw- Hill, 1997, pp. 52–80.

Tan, A. H. (1999, April). Text mining: The state of the art and the challenges. In Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases (Vol. 8, pp. 65-70).

Van Craenendonck, T., & Blockeel, H. (2015). Using internal validity measures to compare clustering algorithms. In Benelearn 2015 Poster presentations (online) (pp. 1-8).

Vijay Kotu, Bala Deshpande (2015) Predictive Analytics and Data Mining: Concepts and Practice with RapidMiner.

Vuori, V. (2006). The employees as a source of external business information. In Proceedings European Productivity Conference EPC (Vol. 6, pp. 29-36).

W. Buntine, Learning classification trees. Stat. Comput. 2, 63–73 (1992)

Xu, D., & Tian, Y. (2015). A comprehensive survey of clustering algorithms. Annals of Data Science, 2(2), 165-193.