Universidad de las Ciencias Informáticas

Facultad 6

Título: “Algoritmos de agrupamiento como extensión al gestor de base de datos PostgreSQL”

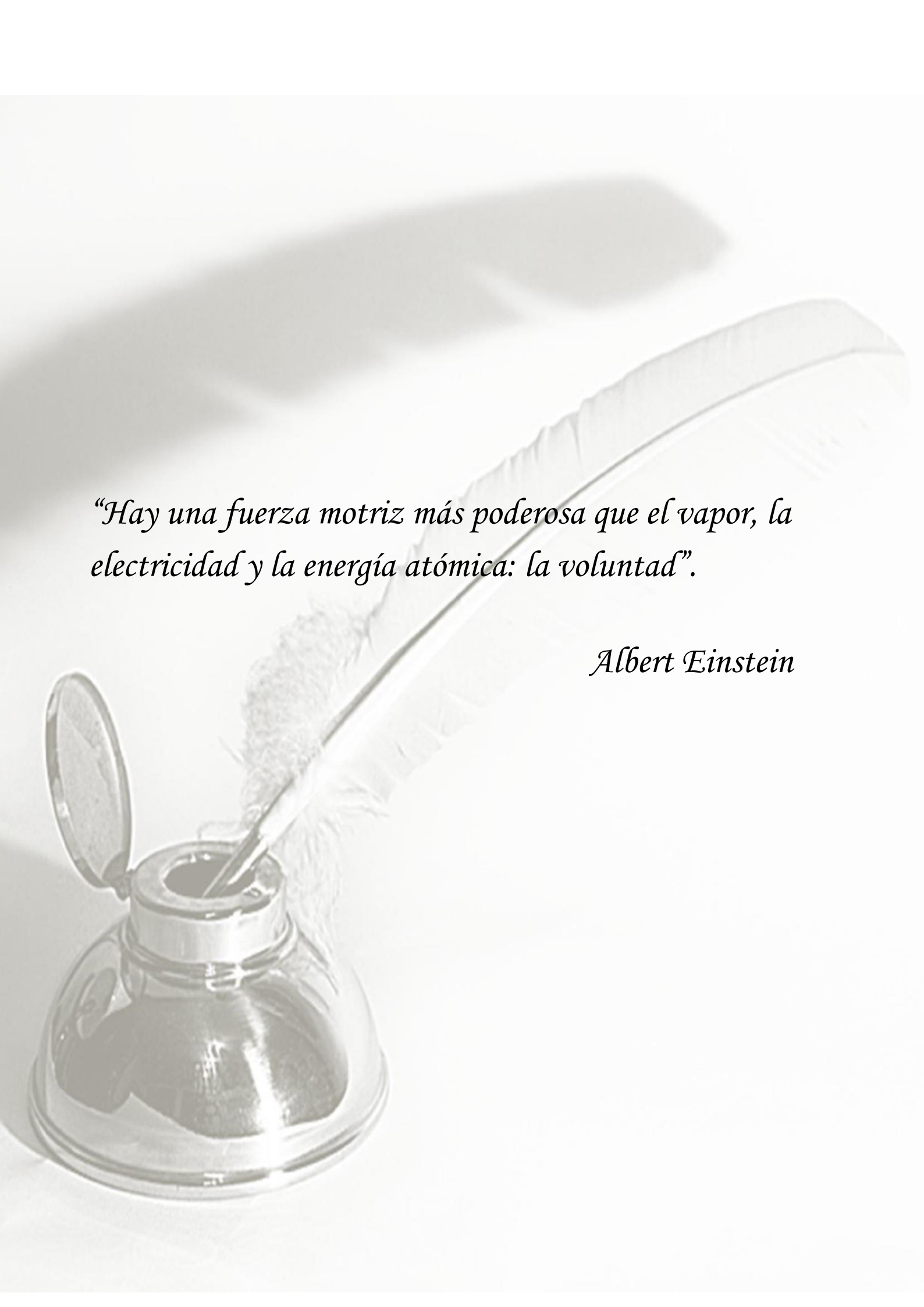
Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Nathalie Solano Trepeu

Tutores: MSc. Yadira Robles Aranda
Ing. Juan Manuel Ruiz Godoy

La Habana, Junio 2014

“Año 56 de la Revolución”



"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad".

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaro que soy la única autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Nathalie Solano Trepeu

_____

Firma del Autor

MSc. Yadira Robles Aranda

_____

Firma del Tutor

Ing. Juan Manuel Ruiz Godoy

_____

Firma del Tutor

DATOS DE CONTACTOS

Tutora:

MSc. Yadira Robles Aranda

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: yrobles@uci.cu

Tutor:

Ing. Juan Manuel Ruiz Godoy

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: jmruiz@uci.cu

AGRADECIMIENTOS

Quiero agradecer a Dios en primer lugar porque sin él no hubiera podido cumplir este sueño.

A mi mamá, por ser la luz de mis ojos, mi razón de ser, por hacerme ver el lado positivo de las cosas por muy difícil que fuese el camino, gracias por animarme cada vez que pensaba que todo estaba perdido.

A mi padrastro por todo su cariño y preocupación.

A mi abuelita por todo su amor y apoyo, quiero decirle que la amo mucho.

A mi papá por ser un buen padre para mí, por brindarme su preocupación y cariño y siempre estar ahí cuando lo necesito.

A mis hermanos por ser los mejores hermanos del mundo.

A mi novio por todo su amor y apoyo incondicional que siempre me ha brindado.

A mis tutores Juan Manuel y Yadira por todo el tiempo y esfuerzo dedicado a la realización de este trabajo.

A los profesores del departamento PostgreSQL que me extendieron su mano para ayudarme.

Agradecimientos

A todas las personas que me ayudaron no solo en el proceso de la tesis, sino también a lo largo de toda la carrera, gracias por su ayuda y amistad.

Por último, les agradezco también a todas aquellas personas lindas que la vida me dio la oportunidad de conocer en esta universidad, aquellas personas que se preocupan verdaderamente por mí, aquellas que saben mis tristezas y alegrías y siempre están ahí para ayudarme. Gracias a todas esas personas por formar parte de mí y hacer que mi vida sea más feliz.

A todos muchísimas gracias.

DEDICATORIA

Dedico este trabajo a la persona más importante en mi vida:
A mi mamá por servirme de guía procurando en todo momento mi superación educacional para ser en un futuro una persona independiente y profesional. A ti mamita te dedico mi tesis con todo mi amor para que te sientas orgullosa de mí y veas que toda la confianza que depositaste en mí no fue en vano.

RESUMEN

En la sociedad actual se ha producido un gran crecimiento de las bases de datos y una necesidad de aumento de las capacidades de almacenamiento que no pueden resolverse por métodos manuales. Por este motivo se hace necesario utilizar técnicas como la Minería de Datos que ayuden de forma automática, en el análisis de esas grandes cantidades de datos. En la presente investigación se analizó de las técnicas de Minería de Datos la de agrupamiento para integrar varios de sus algoritmos al Sistema Gestor de Base de Datos (SGBD) PostgreSQL debido a las deficiencias de las herramientas libres existentes para realizar el análisis de la información. Para el desarrollo de estos algoritmos se seleccionaron las tecnologías y herramientas a utilizar que más se adecuaron a las características del proyecto. Se generaron los artefactos correspondientes a cada fase del proceso de desarrollo de *software* guiado por la metodología XP. Además se comprobó la solución de la investigación mediante la aplicación de las técnicas en los juegos de datos *iris.arff* y *mfeat-fourier.arff* y se comprobó que los algoritmos integrados al gestor permiten aprovechar las potencialidades de PostgreSQL.

Palabras claves: Minería de Datos, PostgreSQL, Agrupamiento, Sistema Gestor de Base de Datos, Metodología XP.

ABSTRACT

Nowadays in our society there has been a large growth of databases and a need to increase the storage capacities that cannot be solved by manual methods. For this reason it is necessary to use techniques such as data mining to help automatically, in the analysis of such large amounts of data. In this research work it has been analyzed the data mining techniques grouping to integrate several of its algorithms to Database Management System (DBMS) PostgreSQL due to the shortcomings of the existing free tools to perform the analysis of information. For the development of these algorithms were selected the tools and technologies to use that were adapted the most to the characteristics of the project. The ertefacts corresponding to each phase of the software development process were generated guided by the XP methodology. The solution of the scientific problem of this investigation is ascertained throw the techniques application in data games iris.arff and mfeat-fourier.arff and proved that integrated algorithms allow the manager to exploit the potential of PostgreSQL.

KEYWORDS: Data mining, PostgreSQL, Grouping, Database Management System, XP methodology.

TABLA DE CONTENIDO

Tabla de contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

A lo largo de la vida de una empresa se acumulan grandes cantidades de datos que son almacenados para posteriormente realizar un análisis y administración de estos. Estos datos se clasifican en pequeños grupos que describan sus características principales basándose en la similitud o diferencia entre ellos. El análisis de grandes volúmenes de datos conduce a ventajas y avances en la ciencia, marketing, prevención de fraudes, vigilancia y otras áreas (1).

Investigaciones dentro de la inteligencia artificial han demostrado que cada vez más existe una diferencia entre la cantidad de datos existentes y la información útil extraída de ellos. Es por ello que la extracción del conocimiento válido y relevante de los datos para la toma de decisiones, requiere de la implementación de nuevas técnicas de la computación y la inducción de conocimiento en bases de datos.

A consecuencia de esta creciente necesidad ha aparecido un nuevo campo de interés: la Minería de Datos (DM, en inglés *Data Mining*). Esta es considerada como un campo que cubre numerosas áreas como la Estadística, la Inteligencia Artificial, la Computación Gráfica, las Bases de Datos, los Sistemas de Toma de Decisiones y otras áreas de la informática y la gestión de información (2).

El objetivo general del proceso de DM consiste en extraer información relevante de un conjunto de datos y transformarla en una estructura comprensible para su uso posterior, permitiendo obtener patrones o modelos a partir de los datos recopilados. Además hace uso de técnicas que aportan información útil como la inducción de reglas, los árboles de asociación, las redes neuronales artificiales y el agrupamiento. Estas son formas de representación del conocimiento utilizadas en sistemas de expertos.

Para la aplicación de estas técnicas existen numerosas herramientas tanto libres como propietarias, entre las libres más conocidas se encuentran Yale/Rapid Miner, Weka y Orange. Las mismas necesitan conectarse al Sistema Gestor de Base de Datos (SGBD), realizar la transformación de los datos para que pueda ser entendido por estas herramientas en sus respectivos formatos y si existe un gran volumen de datos para analizar, el proceso se vuelve engorroso. Por otro lado se conoce que el

SGBD Oracle posee un módulo con varias técnicas de DM como son: Naïve Bayes, Árbol de Decisión, Modelos Lineales Generalizados, Máquinas de Vectores Soporte *Clustering k-Means* mejorado, *Clustering* de Partición Ortogonal, Reglas de Asociación y Factorización de Matrices No Negativas. Sin embargo para hacer uso del mismo es necesario pagar un costo elevado de su licencia. El gestor SQL Server también posee técnicas de DM como: Árboles de Decisión, Naïve Bayes, Clústeres, Redes Neuronales, Serie Temporal, Regresión Lineal, Clústeres de Secuencia y Asociación, sin embargo al igual que Oracle es una herramienta propietaria e implica altas inversiones por el uso de herramientas comerciales y es muy costoso para la economía del país (3).

La Universidad de las Ciencias Informáticas es una institución que se caracteriza por formar profesionales comprometidos con la Revolución y altamente calificados en la rama de la informática. En la misma se han creado varios centros de desarrollo, uno de estos es el Centro de Tecnologías de Gestión de Datos (DATEC), en el cual se encuentra el departamento PostgreSQL. En dicho departamento se está potenciando el Gestor de Base de Datos PostgreSQL con la inclusión de extensiones que permitan realizar DM sobre los datos almacenados en las tablas.

Actualmente se han implementado un total de cinco algoritmos correspondientes a diferentes técnicas, de ellos dos son de técnicas de reglas de asociación y tres de técnicas de clasificación. Estos algoritmos fueron validados en el almacén de datos de ensayos clínicos Racotumumab del Centro de Inmunología Molecular y a la base de datos del Sistema de Genética Médica respectivamente. Esto permitió comprobar que los algoritmos integrados al gestor permiten aprovechar las potencialidades de PostgreSQL. Por otro lado hay que tener en cuenta que estos algoritmos limitan al usuario final a solo realizar análisis sobre tipos de datos nominales. Es por ello que en el departamento PostgreSQL, a pesar de contar con algoritmos de DM integrados al gestor, persiste la dificultad de realizar análisis de clasificación sobre otros tipos de datos.

De acuerdo a la problemática planteada se identifica como **problema científico**:

¿Cómo lograr el análisis de los datos en el SGBD PostgreSQL aplicando DM?

Se plantea como **objeto de estudio** la DM en PostgreSQL y como **campo de acción** las técnicas de agrupamiento de la DM en PostgreSQL.

Con el fin de resolver el problema planteado se ha trazado el siguiente **objetivo general:**

Integrar algoritmos de las técnicas de agrupamiento como extensión al gestor de base de datos PostgreSQL para analizar el comportamiento de los datos almacenados en las tablas. Del cual se derivan los siguientes **objetivos específicos:**

- Analizar las técnicas de agrupamiento de los datos.
- Desarrollar los algoritmos basados en técnicas de agrupamiento.
- Integrar los algoritmos implementados al SGBD PostgreSQL.
- Validar la solución propuesta.

Para lograr los objetivos específicos se formularon las siguientes **tareas de investigación:**

a) Caracterización de las técnicas de agrupamiento de los datos.

b) Caracterización de las herramientas de DM.

c) Análisis del lenguaje de programación y herramientas para la implementación en PostgreSQL.

d) Implementación de los algoritmos de agrupamiento de los datos.

e) Creación de la extensión de DM.

f) Incorporación de la extensión creada al SGBD PostgreSQL.

g) Aplicación del diseño de pruebas.

h) Validación del correcto funcionamiento de los algoritmos implementados.

En el desarrollo de la investigación los **métodos de trabajo científico** que fueron utilizados son:

Histórico – Lógico: permitió realizar un estudio para definir las tendencias actuales de las técnicas de DM y las herramientas utilizadas para la aplicación de estas.

Analítico – sintético: este método ha servido para analizar y comprender la teoría y documentación relacionada con el tema de investigación, permitiendo así, extraer los elementos más relacionados e importantes con el objeto de estudio.

Observación: es usado en varios momentos de la investigación pues a través del mismo es posible obtener conocimiento acerca del comportamiento del objeto de estudio tal y como este se da en la realidad, además de permitir el acceso a su información de forma directa e inmediata, en su manifestación externa, sin llegar a la esencia del mismo.

Entrevista: utilizada como medio para el conocimiento cualitativo sobre las características y propiedades del objeto de estudio a partir de la experiencia acumulada del entrevistado.

El presente trabajo está compuesto por introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y bibliografía. A continuación se muestra una breve descripción de cada capítulo.

Capítulo1: Fundamento Teórico.

En el Capítulo 1 se realiza un estudio sobre las herramientas usadas en el proceso de DM. Se define el lenguaje de programación y la herramienta a utilizar para implementar los algoritmos así como la metodología de desarrollo del *software* a utilizar. También se seleccionan los algoritmos a implementar de agrupamiento de los datos y se presenta una descripción de los mismos.

Capítulo 2: Descripción de la Solución.

En el Capítulo 2 se generan los artefactos que concibe la metodología escogida, se presenta una descripción de los algoritmos implementados incluyendo los pseudocódigos así como una breve descripción de todas las funciones implementadas. Además se muestra como fueron integradas al SGBD PostgreSQL.

Capítulo 3: Aplicación y Validación de la Solución Propuesta.

En el Capítulo 3 se realiza la aplicación de las técnicas de DM a los juegos de datos *iris.arff y mfeat-fourier.arff* y se validan los algoritmos implementados haciendo uso de las técnicas de caja negra.

También contiene las conclusiones donde se exponen los principales resultados de la investigación y se proponen recomendaciones para continuar desarrollando el objeto de la investigación. Además se exponen todos los materiales consultados y referenciados, quedando organizados en referencias bibliográficas y la bibliografía según corresponda en cada caso.

CAPÍTULO1. FUNDAMENTO TEÓRICO

Introducción

En este capítulo se exponen definiciones relacionadas con la DM para obtener una mejor comprensión de la misma. Se seleccionan los algoritmos de agrupamiento de los datos a implementar. Se realiza un estudio de las herramientas usadas en el proceso de DM y se escoge la más adecuada para solucionar el problema planteado. Además se seleccionan las herramientas, la metodología de desarrollo del *software* y el lenguaje de programación a utilizar.

1.1 Minería de Datos

El avance de los sistemas de cómputo ha incrementado el volumen de información de las empresas, por lo que es necesaria la creación de grandes almacenes de datos que permitan ser explorados en búsquedas de información refinada. Procesar estos grandes volúmenes de datos y convertirlos en conocimiento útil para la toma de decisiones constituye un reto colosal. El Descubrimiento de Conocimiento en Bases de Datos (KDD, en inglés *Knowledge Discovery in Data Bases*) ha emergido para abordar este problema. Dentro de sus etapas hay una que resulta fundamental, la DM (2).

La DM ha surgido por la necesidad del análisis de grandes volúmenes de información con el fin de obtener resúmenes y conocimiento que apoye la toma de decisiones y que pueda construir una experiencia a partir de millones de transacciones detalladas que registra una corporación en sus sistemas informáticos. Permite además descubrir pautas de comportamiento en bases de datos para tomar decisiones más confiables, además de construir modelos predictivos y en general extraer información no evidente utilizando métodos computacionales intensivos (4).

1.1.1 Definiciones de Minería de Datos

La DM consiste en la extracción de información que reside de manera implícita, desconocida o previamente ignorada que puede ser potencialmente útil de un conjunto de datos. Se puede considerar a la DM como una colección de diferentes técnicas que sirven para inducir el conocimiento e

información de una manera estructurada de un gran conjunto de datos (4). Existen varias definiciones de DM, a continuación se citan algunas de ellas:

“La DM es el proceso que tiene como propósito descubrir, extraer y almacenar información relevante de amplias bases de datos a través de programas de búsqueda e identificación de patrones y relaciones globales, tendencias, desviaciones y otros indicadores aparentemente caóticos que tienen una explicación que pueden descubrirse mediante diversas técnicas de esta herramienta” (5).

“Un proceso no trivial de identificación válida, novedosa, potencialmente útil y entendible de patrones comprensibles que se encuentran ocultos en los datos”. Desde el punto de vista empresarial, se define como: “La integración de un conjunto de áreas que tienen como propósito la identificación de un conocimiento obtenido a partir de las bases de datos que aporten un sesgo hacia la toma de decisiones” (6).

Después de exponer estos conceptos por los distintos autores se puede concluir que la DM es un proceso que apoyándose de técnicas y herramientas descubre información potencialmente útil y valiosa en grandes conjuntos de datos para la toma de decisiones.

1.1.2 Técnicas de Minería de Datos

Una técnica de DM constituye el enfoque conceptual para extraer la información de los datos y en general es implementada por varios algoritmos. Cada algoritmo representa en la práctica, la manera de desarrollar una determinada técnica paso a paso, de forma que es preciso un entendimiento de alto nivel de los algoritmos para saber cuál es la técnica más apropiada para cada problema (2).

Las técnicas de DM se encuentran en continua evolución como resultado de la colaboración entre campos de investigación tales como bases de datos, reconocimiento de patrones, inteligencia artificial, sistemas expertos, estadística, recuperación de información y computación de altas prestaciones. Estas técnicas se clasifican en dos grandes categorías:

- **Supervisados o predictivos:**

Predicen el valor de un atributo de un conjunto de datos, conocidos otros atributos. A partir de datos cuya etiqueta se conoce, se induce una relación entre dicha etiqueta y otra serie de atributos. Esas relaciones sirven para realizar la predicción de datos cuya etiqueta es desconocida (7).

- **No supervisados o del descubrimiento de conocimiento:**

Con estos algoritmos se descubren patrones y tendencias en los datos actuales. El descubrimiento de esa información sirve para llevar a cabo acciones y obtener un beneficio de ellas (7). A continuación se muestran algunas de las técnicas de DM de ambas categorías.

Técnicas de Minería de Datos	
Supervisado	No Supervisado
Clasificación ➢ Tabla de Decisión ➢ Árboles de Decisión ➢ Reglas ➢ Bayesianas ➢ Basado en Ejemplares ➢ Redes de Neuronas Predicción ➢ Regresión ➢ Árboles de Predicción ➢ Estimador de Núcleos	Agrupamiento ➢ Numérico K-MEDIAS ➢ Conceptual ➢ Probabilístico Asociación ➢ A Priori

Fig.1 Técnicas de Minería de Datos

Técnicas de agrupamiento

La DM contiene diversos tipos de técnicas, entre las que se destacan las técnicas de agrupamiento, que permiten la identificación de tipologías o grupos donde los elementos guardan gran similitud entre sí y muchas diferencias con los de otros grupos. El agrupamiento es una de las tareas más utilizadas en el proceso de DM para descubrir grupos e identificar interesantes distribuciones y patrones en los

datos (8). Este es un proceso de aprendizaje no supervisado ya que las clases no están predefinidas sino que deben ser descubiertas dentro de los datos y su objetivo fundamental es determinar el comportamiento de un nuevo grupo. A partir de las características del mismo, se define a qué grupo pertenecerá y que acción podrá realizar.

El agrupamiento de datos ayuda a discernir la estructura y simplifica la complejidad de cantidades masivas de datos. Es una técnica que se utiliza en diversos campos como: el aprendizaje automático, DM, reconocimiento de patrones, análisis de imágenes y bioinformática, donde la distribución de la información puede ser de cualquier tamaño y forma (8).

La principal característica de esta técnica es la utilización de una medida de distancia que en general, está basada en los atributos que describen a los objetos y se define usualmente por proximidad en un espacio multidimensional. Para datos numéricos suele ser preciso preparar los datos antes de realizar DM sobre ellos de manera que en primer lugar se someten a un proceso de estandarización. A través del cálculo de distancias se agrupan los elementos de acuerdo a los más cercanos. Las distancias más utilizadas hoy en día son: Euclidiana, Manhattan y Hamming (9).

Tipos de agrupamiento

A pesar de existir una gran cantidad de técnicas de agrupamiento, todas pueden ser clasificadas en uno de los siguientes cuatro tipos de agrupamiento:

- **Algoritmos de agrupamiento particionales:**

Son aquellos que obtienen como resultado una única partición de los datos iniciales, en lugar de una estructura de agrupamiento con varios niveles de particiones (9).

- **Algoritmos de agrupamiento jerárquicos:**

Organizan los datos en estructuras jerárquicas de acuerdo a la matriz de proximidades. Los resultados de estos algoritmos son, por lo general, mostrados en un árbol binario (9).

- **Algoritmos de agrupamiento probabilísticos:**

Desde el punto de vista probabilístico, se asume que los objetos son generados de acuerdo a algunas distribuciones probabilísticas. Objetos en distintos grupos son generados por distintas distribuciones de probabilidad o son derivados de distintos tipos de funciones de densidad o de las mismas familias pero con distintos parámetros (9).

- **Algoritmos de agrupamiento basados en densidades:**

Estos algoritmos aplican criterios locales de grupo. Los grupos son tenidos en cuenta como regiones en el espacio de datos de gran densidad de objetos y están separados por regiones de menor densidad (ruido). Estas regiones pueden tener cualquier forma y pueden estar distribuidas de cualquier manera (9).

Aplicaciones del agrupamiento

Los algoritmos de agrupamiento se han aplicado en una amplia variedad de campos. A continuación se muestran algunos de los que se consideran más importantes:

- **Ingeniería:**

Las aplicaciones típicas de la agrupación en el rango de ingeniería es el reconocimiento biométrico y reconocimiento de voz, para el análisis de señales de radar, la compresión de la información y la eliminación de ruido (10).

- **Ciencias de la computación:**

Las aplicaciones de la agrupación en el campo de la computación son múltiples, entre estas están la minería web, DM, análisis de bases de datos espaciales, la recuperación de información y la segmentación de imágenes (10).

- **Ciencias médicas y de la vida:**

Áreas como la genética, biología, microbiología, paleontología, psiquiatría, clínica, la filogenia y la patología, consisten en las principales aplicaciones de la agrupación en su etapa temprana y seguirá siendo uno de los principales campos de juego para los algoritmos de agrupamiento. Otras

aplicaciones importantes incluyen la definición de taxonomía y la identificación de genes en función de la proteína, el diagnóstico y el tratamiento de la enfermedad (10).

- **Astronomía y ciencias de la tierra:**

La agrupación en áreas como la geografía, geología y teledetección se puede utilizar para clasificar las estrellas y los planetas, investigar formaciones de la tierra y el estudio de los ríos y sistemas de montañas (10).

- **Las ciencias sociales:**

En la sociología, psicología, arqueología, antropología y educación se pueden encontrar aplicaciones interesantes en el análisis de patrones de comportamiento, identificación y relación entre las diferentes culturas, la construcción de la historia evolutiva de las lenguas, el análisis de las redes sociales, hallazgo arqueológico y artefactos de clasificación y el estudio de la psicología criminal (10).

- **Economía:**

Las principales aplicaciones dentro del marketing y el negocio son las características de los clientes, la agrupación de empresas y el análisis de los valores de tendencia que se benefician de la utilización de análisis de conglomerados (10).

Después de realizar un estudio de los principales campos en lo que se utiliza el agrupamiento se puede concluir que los resultados obtenidos en este trabajo se van a aplicar en los campos de las ciencias de la computación y la biología. Esto se puede afirmar ya que los algoritmos a implementar serán validados en bases de datos asociadas a estos dos campos.

1.2 Selección de los algoritmos a implementar

La eficiencia de los algoritmos de agrupamiento es extremadamente necesaria cuando se trabaja con enormes bases de datos y tipos de datos de grandes dimensiones. En este trabajo se comienza el estudio del análisis de datos con la propuesta de implementación de los algoritmos *K-means*, DBSCAN y *FarthestFirst*. Se decidió comenzar con la implementación de estos algoritmos teniendo en cuenta las ventajas que los mismos presentan. El algoritmo *K-means* es conveniente cuando existe un gran

volumen de datos a analizar, este puede ser considerable cuando se trata de grandes volúmenes de datos. En el caso de DBSCAN su mayor fortaleza radica que es potente ante la presencia de ruidos y valores atípicos. Por su parte el algoritmo *FarthestFirst* presenta como propiedad fundamental la rapidez en el análisis de los datos y da la posibilidad de obtener resultados diferentes para escoger el más adecuado.

1.2.1 K-means

El algoritmo *K-means* fue propuesto por MacQueen en el año 1967 y pertenece a los algoritmos de agrupamiento particionales. Toma un parámetro de entrada k y parte un conjunto de n objetos en k grupos tal que la similitud resultante dentro de un grupo es alta, pero la similitud con otros grupos es baja. Este método es simple, efectivo y fácil de probar, busca en su funcionamiento una partición óptima de los datos minimizando el criterio de la suma del error cuadrado con un procedimiento iterativo de optimización (11).

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2 \quad \text{Ec.1}$$

Donde E es la suma del error al cuadrado para todos los objetos del conjunto de datos; p es el punto en el espacio representando a un objeto dado; y mi es la media del clúster Ci.

Pasos para aplicar el algoritmo *K-means* (11).

1. Especificar la cantidad de grupos (K) que se van a crear y el número máximo de iteraciones.

2. Cada una de las instancias son distribuidas por los grupos de manera aleatoria.

3. Se calcula el valor del centroide de todos los grupos y se toma como el centro de sus respectivos grupos.

4. Cada una de las instancias son asignadas a su centroide más cercano formando los nuevos grupos.

5. Se repiten los pasos 3 y 4 hasta que el valor de los centroides no varíe más, después de cada iteración.

El pseudocódigo de *K-means* se muestra a continuación:

Require: Un conjunto de n elementos $X = \{x_1, \ldots, x_n\}$ y un número fijo de clusters k.
Ensure: Un conjunto de clusters $C = \{C_1, \ldots, C_k\}$ que divide X
1: Asigna k instancias siendo los centroides iniciales. Se define el conjunto de centroides como $Y = \{y_1, \ldots, y_k\}$ y $Y' = \emptyset$
2: **while** $Y \neq Y'$ **do**
3: Asigna todos $C_j = \emptyset$.
4: $Y' \leftarrow Y$.
5: **for all** $x_i \in X$ **do**
6: Calcula la distancia mínima del centroide a x_i. Siendo y_j la distancia mínima a x_i.
7: Introduce x_i en C_j.
8: **end for**
9: Calcula los centroides de C y establece $y_i \leftarrow centroide(C_i)$.
10: **end while**
11: **return** C

Fig.2 Pseudocódigo del algoritmo *K-means*

Las ventajas que presenta este algoritmo son su simplicidad y rapidez; la cual permite correr sobre largos conjuntos de datos. Esto es tenido en cuenta para minimizar el problema de tomar aleatoriamente los centros de clúster, permitiendo realizar varias corridas del algoritmo y tomar aquellos centros de clúster que favorezcan más a los agrupamientos (8).

La principal desventaja de este algoritmo radica en que el resultado obtenido es dependiente de la selección inicial de los centroides y puede converger a óptimos locales. Por lo tanto, la selección de los centroides iniciales afecta el proceso principal de *K-means* y la partición resultante de este proceso. No obstante, si se obtienen buenos centroides iniciales con alguna técnica alternativa, *K-means* refinaría esos centroides de los clústeres obteniendo mejores resultados. Además es muy sensible a los datos anómalos con valores extremos ya que distorsionan la media del clúster. Para disminuir esta sensibilidad, en vez de tomar un punto cualquiera como referencia del clúster, este deber ser un punto del clúster (12).

1.2.2 DBSCAN

El algoritmo DBSCAN fue desarrollado en el año 1996 por Martin Ester en la Universidad de Munich. Es un algoritmo simple basado en densidades que trata de ilustrar una serie de conceptos necesarios para cualquier algoritmo basado en densidades. La idea principal de DBSCAN es encontrar todos los puntos centrales, donde los puntos centrales de un grupo son aquellos que tienen una región de vecindad que contiene un número mínimo de puntos para un radio determinado. La forma de la región de vecindad viene determinada por la elección de la medida de la distancia entre dos puntos, dist (p; q) (9).

La región de vecindad de un punto *p* perteneciente a una base de datos *D* dado un radio *Eps* se define como:

$$N_{Eps}(p) = \{q \in D | dist(p,q)\} \leq Eps \quad \text{Ec.2}$$

El algoritmo tiene como parámetros de entrada el valor del radio de la región o del área de vecindad, Eps y el número mínimo de puntos, MinPts. Para encontrar los grupos DBSCAN comienza con un punto ***p*** arbitrario y se van recopilando todos los puntos que son densamente alcanzables desde ***p*** con respecto a Eps y MinPts. Si ***p*** es un punto central, entonces el proceso terminará formando un grupo con respecto a Eps y MinPts. Si por el contrario, ***p*** es un punto frontera o es ruido no hay puntos que sean densamente alcanzables desde ***p*** y en este caso, ***p*** se clasifica como ruido y se pasa al siguiente punto del conjunto total. Si un punto ***p*** es etiquetado inicialmente como ruido, pero después se descubre que es densamente alcanzable desde otro punto ***q***, entonces se le quita la etiqueta de ruido y se le asigna la de un punto frontera, clasificándolo en el mismo grupo que el punto ***q*** desde el que es densamente alcanzable (9).

El proceso completo para la agrupación usando DBSCAN se muestra a continuación:

Entrada: Conjunto de datos $D \in \mathbb{R}^d$, con n puntos, radio Eps y $MinPts$;

1: **while** $\exists\, p \in D$ sin clasificar **do**
2: coger $p \in D$;
3: **if** p sin clasificar **then**
4: $N_{Eps}(p) \leftarrow \{q \in D | dist(p,q) \leq Eps\}$;
5: **if** $|N_{Eps}(p)| > MinPts$ **then**
6: $type(p) \leftarrow core$;
7: $class(p) \leftarrow clusId$;
8: **end if**
9: **if** p es core **then**
10: crear lista $\Phi = \{x \in D | x \in N_{Eps}(p)\}$;
11: **while** Φ tiene objetos **do**
12: coger $o \in N_{Eps}(p)$;
13: $N_{Eps}(o) \leftarrow \{q \in D | dist(o,q) \leq Eps\}$;
14: **if** $|N_{Eps}(o)| > MinPts$ **then**
15: $type(o) \leftarrow core$;
16: **for all** objecto $i \in N_{Eps}(o)$ **do**
17: **if** i esta $sin\ clasificar$ o es $ruido$ **then**
18: **if** i esta $sin\ clasificar$ **then**
19: añadir objeto i a Φ;
20: **end if**
21: **if** i es $ruido$ **then**
22: $type(i) \leftarrow border$;
23: **end if**
24: $class(i) \leftarrow clustId$
25: **end if**
26: **end for**
27: **else**
28: $type(o) \leftarrow border$;
29: **end if**
30: $class(o) \leftarrow clusId$;
31: eliminar o de Φ;
32: **end while**
33: **else**
34: $type(p) \leftarrow noise$;
35: $class(p) \leftarrow noise$;
36: **end if**
37: **end if**
38: $clustId \leftarrow clusId + 1$;
39: **end while**

Salida: vector $class$ y vector $type$.

Fig.3 Pseudocódigo del algoritmo DBSCAN

Algunas de las aplicaciones de DBSCAN realizadas con éxito son la creación de perfiles de usuarios en *Internet* mediante la agrupación de sesiones *Web* o el agrupamiento de bases de datos de imágenes en histogramas en color facilitando la búsqueda de imágenes similares. El algoritmo DBSCAN presenta una ventaja sobre la mayoría de los algoritmos y es la capacidad de reconocer la presencia de ruido o de reconocer valores atípicos en los datos (9).

1.2.3 FarthestFirst

El algoritmo *FarthesFirst* fue creado por Hochbaum and Shmoys en el año 1985 y pertenece a los algoritmos de agrupamiento particionales. Tiene como parámetro de entrada el número de grupos que se desea obtener. Comienza seleccionando aleatoriamente una instancia del conjunto de datos que pasa a ser el centro del clúster. Se calcula la distancia entre cada una de las instancias y el centro. La distancia que se encuentre más alejada del centro, es seleccionada como el nuevo centro del clúster. Este proceso se repite hasta alcanzar el número de clústeres buscado (13). El pseudocódigo de *FarthesFirst* se muestra a continuación:

H denota o conjunto de objetos representativos do *cluster*, $\{h_1, ..., h_k\} \subset S$.
Seja $cluster(x_i)$ a identificação do *cluster* ao qual o elemento $x_i \in S$ pertence.
Seja $dist(x_i)$ a distância entre x_i e o objeto representativo mais próximo do cluster: $dist(x_i) = min_{h_j \in H} L(x_i, h_j)$
Selecione aleatoriamente um objeto x_j de S, seja $h_1 = x_j, H = h_1$.
for $j = 1$ até n **do**
  $dist(x_j) = L(x_j, h_1)$
  $cluster(x_j) = 1$
end for
for $i = 2$ até K **do**
  $D = \max_{x_j} : x_j \in S\, H dist(x_j)$
  Escolha $h_i \in S\, H s.t. dist(hi) = D$
  $H = H \in \{h_i\}$
  for $j = 1$ até n **do**
    if $L(x_j, , h_i) \leq dist(x_j j)$ **then**
      $dist(xj) = L(x_j, h_i)$
      $cluster(x_j) = i$
    end if
  end for
end for

Fig.4 Pseudocódigo del algoritmo *FarthesFirst*.

La principal ventaja de este algoritmo radica en su complejidad temporal la cual es O (nk), donde *n* es el número de objetos y *k* es el número de grupos deseados. *FarthesFirst* es rápido y adecuado para aplicaciones de DM a gran escala. La dificultad principal es que los resultados obtenidos en cada corrida son diferentes debido a que el resultado encontrado depende de la asignación aleatoria inicial de los centros de clústeres. Este puede quedar atrapado en mínimos locales y afectar la calidad final de la solución. Además el hecho de tener que fijar a priori el número de clústeres a obtener puede conllevar a resultados no óptimos (14).

1.3 Herramientas de Minería de Datos

Las herramientas de DM se están utilizando desde hace varios años para la obtención de patrones en los datos y para la extracción de información valiosa. Estas predicen futuras tendencias y comportamientos que permiten hacer una toma de decisiones y pueden responder a preguntas que tradicionalmente consumen mucho tiempo para poder ser resueltas. Además exploran las bases de datos en busca de patrones ocultos, encontrando información predecible que un experto no puede llegar a encontrar porque se encuentra fuera de sus expectativas. En la actualidad existe una gran cantidad de herramientas de *software* para el desarrollo de la DM. A continuación se realiza un estudio de algunas que se consideran más importantes.

Weka: es una herramienta de libre distribución (licencia GPL) desarrollada por miembros de la Universidad de Waikato y se destacada por la cantidad de algoritmos que presenta así como por la eficiencia de los mismos. Está escrita en lenguaje Java y permite realizar multitud de análisis. En ella se implementan las técnicas de clasificación, asociación, agrupamiento y predicción existentes en la actualidad. Weka también proporciona acceso a bases de datos vía SQL gracias a la conexión JDBC (*Java Database Connectivity*) y puede procesar el resultado devuelto por una consulta hecha a la base de datos. Sus principales características son:

- Es multiplataforma.
- Es fácil de usar gracias a su interfaz gráfica.

- Acceso a los datos desde un archivo en formato ARFF (es un archivo de texto plano organizado en filas y columnas).
- Preprocesado de datos (selección, transformación de atributos.)
- Modelos de Aprendizaje (reglas de asociación, modelos de agrupamiento, modelos combinados.)
- Contiene un grupo de herramientas que permiten el análisis de los datos (7).

RapidMiner/YALE: Es una herramienta prototipada creada en la Universidad de Dortmund bastante flexible para el descubrimiento del conocimiento y la DM. Es un *software* de código abierto GNU y con licencia GPL. YALE puede importar información a partir de sistemas de Bases de Datos como PostgreSQL y *Microsoft SQL Server*. Entre sus principales características se encuentran:

- Trabaja bajo las plataformas Windows y Linux.
- Posee alrededor de 400 operadores que pueden ser combinados.
- Usa el lenguaje de *scripting* XML para describir los operadores y su configuración.
- La característica más importante es la capacidad de jerarquizar cadenas del operador y de construir complejos árboles de operadores.
- Posee una gran cantidad de extensiones (plugins). (3)

Orange: Orange es una herramienta de libre distribución que permite realizar DM. Está implementada en C++, se pueden desarrollar módulos en Python y su interfaz gráfica de usuario se basa en el marco de Qt multiplataforma. Existen distribuciones para Windows, Linux y Macintosh (15). Cuenta con una programación visual fácil, potente, rápido y versátil para el análisis exploratorio de datos. Contiene además un juego completo de componentes para preprocesamiento de datos. Esta es una característica que anota el filtrado, modelado y la evaluación del modelo. Cuenta con técnicas como

método Bayesiano Naïve, árboles de decisión, árboles de regresión, vecinos más próximos y reglas de asociación. (16)

Oracle Data Mining: Oracle Data Mining es un módulo con técnicas de DM que posee el SGBD Oracle, esto le permite al gestor descubrir información valiosa en sus datos utilizando una amplia gama de algoritmos de avanzada. Permite además buscar patrones, identificar los atributos clave, descubrir nuevos clústeres y asociaciones y revelar conocimientos valiosos. Oracle Data Mining incluye una interfaz gráfica de usuarios para el análisis de datos llamada Oracle Data Miner, que tiene el fin de crear, evaluar y aplicar modelos de DM. Esta guía al analista de datos a través del proceso de DM con total flexibilidad y presenta los resultados en formatos gráficos y tabulares. Oracle Data Miner puede generar el código PL/SQL asociado con una Actividad de Recuperación de los Datos (17).

SQL Server Data Mining: El SGBD SQL Server ofrece un entorno integrado para crear modelos de DM y trabajar con ellos. La solución SQL Server Data Mining permite el acceso a la información necesaria para tomar decisiones inteligentes sobre problemas empresariales complejos. Entre sus características principales se encuentran:

- Oculta la complejidad de una tecnología avanzada.
- Incluye una suite completa de algoritmos para extraer automáticamente información de los datos.
- Puede procesar grandes volúmenes de datos y los datos con relaciones complejas.
- Los datos pueden ser obtenidos de bases de datos relacionales y OLAP.
- Utiliza interfaces de programación estándar XMLA y DMX.
- Proporciona un marco completo para crear y desplegar aplicaciones inteligentes.

Varios de los algoritmos implementados por SQL Server Data Mining son: Árboles de Decisión, Bayes Naive, Clústeres, Redes Neuronales, Regresión Logística y Lineal, Reglas de asociación, Clústeres de Secuencia y Asociación (18).

1.3.1 Fundamentación de la herramienta seleccionada

Después de un estudio de las herramientas de DM se puede llegar a la conclusión de que Yale/Rapid Miner, Weka y Orange a pesar de ser herramientas libres y poseer diversas ventajas tienen la desventaja de que el proceso es engorroso ya que requiere de tiempo para la preparación y la vinculación de los datos con el gestor, extendiendo así el tiempo de respuesta de los análisis de los datos.

Por otra parte ORACLE y SQL Server son herramientas muy potentes y una de sus mayores fortalezas radica en la integración con el Sistema Gestor de Base de Datos, pero ambas son herramientas propietarias e implican altas inversiones por el uso de herramientas comerciales y es muy costoso para la economía del país. Todas estas dificultades encontradas en este grupo de herramientas constituyen una problemática para la presente investigación. A raíz de ello se decide implementar algoritmos de agrupamiento para integrarlos al Gestor de Base de Datos PostgreSQL. De esta manera se aprovecha las propiedades propias del gestor y se evita utilizar las herramientas de DM debido al proceso engorroso de transformación de los datos desde el gestor PostgreSQL hacia estas.

1.4 Metodologías para aplicar la Minería de Datos. CRISP-DM

Las metodologías de DM permiten a los desarrolladores establecer una secuencia lógica a seguir para desarrollar proyectos de la mejor forma posible. Contar con una metodología se ha convertido hoy en día en un eslabón importante y necesario para las empresas.

Las iniciales CRISP-DM responden a *Cross Industry Standard Process for Data Mining* (según sus siglas en inglés), esta metodología está compuesta por una serie de etapas que son: análisis del problema, análisis de los datos, preparación de los datos, modelado, evaluación y despliegue (19). A continuación se explican cada una de ellas:

- **Análisis del problema:** Esta fase inicial está enfocada a entender los objetivos y requerimientos desde una perspectiva de negocio; para luego definirlos en términos de un problema de DM y diseñar un plan para satisfacerlos. Las principales tareas que componen esta fase son las siguientes:

1. Determinar los objetivos del negocio.
2. Evaluación de la situación.
3. Determinación de los objetivos de la DM (20).

- **Análisis de los datos:**

En esta fase se hace una recolección y exploración inicial de los datos para identificar problemas de calidad. Además se trata de descubrir o estimar las relaciones más evidentes para formular las primeras hipótesis sobre información oculta en ellos. Las principales tareas que componen esta fase son las siguientes:

1. Recopilación inicial de datos.
2. Descripción de los datos.
3. Exploración de los datos.
4. Verificación de calidad de datos (20).

- **Preparación de los datos:**

Esta fase cubre todas las actividades necesarias para construir la colección de datos que finalmente será minada a partir del grupo inicial. Incluye la colección, exploración, limpieza, transformación y construcción de datos. Las principales tareas que componen esta fase son las siguientes:

1. Selección de los datos.
2. Limpieza de datos.
3. Construcción de datos.
4. Integración de datos.
5. Formateo de datos (20).

- **Modelado:**

Durante esta fase se aplican varias técnicas de modelado. Comúnmente existen varias técnicas para resolver un problema de DM del mismo tipo. Incluye la evaluación desde el punto de vista de precisión de los modelos. Las principales tareas que componen esta fase son las siguientes:

1. Selección de la técnica de modelado.
2. Diseño de la evaluación.
3. Construcción del modelo.
4. Evaluación del modelo (20).

- **Evaluación:**

Al llegar a esta fase se tendrá los modelos de mayor calidad desde la perspectiva de la precisión. Se impone una evaluación de los modelos y de los pasos que se siguieron para su construcción, a fin de determinar si responden apropiadamente a los objetivos de negocio que se determinaron en la primera fase. Es de vital importancia analizar si alguna regla del negocio no fue tomada en cuenta con el suficiente peso. Las principales tareas que componen esta fase son las siguientes:

1. Evaluación de resultados.
2. Revisar el proceso.
3. Establecimiento de los siguientes pasos o acciones (20).

- **Explotación o Despliegue:**

En dependencia de los requerimientos y objetivos, la fase de despliegue puede ser tan simple como generar un reporte o tan compleja como emprender un proceso de KDD de mayor envergadura. En ocasiones son los clientes y no los desarrolladores quienes implementan esta fase; deben comprender cómo desarrollarla. Se hace imprescindible documentar y presentar los resultados de manera que todos los puedan entender. Las principales tareas que componen esta fase son las siguientes:

1. Planificación de despliegue.

2. Planificación de la monitorización y del mantenimiento.

3. Generación de informe final.

4. Revisión del proyecto (20).

Después de realizar un análisis de las principales características de la metodología CRISP-DM se puede observar que esta busca solucionar un problema que permanece en el negocio y que los resultados están ocultos dentro de toda la información. Se pudo comprobar que CRISP-DM ve el proyecto de forma global, enfatizando en el conocimiento del negocio, de esta forma la metodología está más apegada al proyecto que se realiza. Además ha sido diseñada como una metodología neutra respecto a la herramienta que se utilice para el desarrollo del proyecto de DM siendo su distribución libre y gratuita.

Dadas las características y ventajas expuestas se ha seleccionado CRISP-DM como metodología a utilizar en el proceso de MD de esta investigación. Esta elección está respaldada además porque se obtienen proyectos de minería de alta calidad. Es una metodología que brinda una facilidad de uso con cualquier herramienta que se utilice para el proceso de minería. Se opta además por esta metodología porque permite acercarse a las necesidades reales del proyecto e identificar qué es lo que se quiere encontrar, adentrándose a los requerimientos del negocio en cuestión.

1.5 Metodología de desarrollo de *software*. *Extreme Programming*

Las metodologías de desarrollo de *software* surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto *software*. Dichas metodologías pretenden guiar a los desarrolladores al crear un nuevo *software*, pero los requisitos de un *software* a otro son tan variados y cambiantes que ha dado lugar a que exista una gran variedad de metodologías para la creación del *software*. Las metodologías se dividen en dos grupos, tradicionales (pesadas) y ágiles (ligeras).

Las tradicionales se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar y requiere una extensa documentación, pretendiendo prever todo de antemano. En las ágiles es más importante lograr que un producto de *software* se realice con la calidad requerida que hacer

una buena documentación, en este tipo de metodología el cliente está presente en todo momento y colabora con el proyecto como un miembro más. Dentro de estas se encuentra *Extreme Programming* (XP), la cual es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software*. Promueve el trabajo en equipo preocupándose en todo momento del aprendizaje de los desarrolladores y estableciendo un buen clima de trabajo. Se basa en una realimentación continuada entre el cliente y el equipo de desarrollo con una comunicación fluida entre todos los participantes, también busca simplificar las soluciones implementadas y coraje para los múltiples cambios. Este tipo de programación es la adecuada para los proyectos con requisitos imprecisos, muy cambiantes y con un riesgo técnico excesivo. Una de las grandes primacías de XP es que garantiza a toda costa el cumplimiento de integrar todas las necesidades del cliente en el proyecto (21).

Características de la metodología XP

El desarrollo bajo XP tiene características que lo distinguen de otras metodologías, estas son:

- Comunicación efectiva en tiempo real entre el cliente y los desarrolladores.
- Se liberan varias entregas del *software* en la medida que se va desarrollando.
- Los objetivos planteados en características, tiempo y costos son reajustados inalterablemente en función del avance real obtenido.
- Añade funcionalidad con retroalimentación continua en correspondencia con la magnitud que va alcanzando el proyecto.
- Los cambios son parte sustantiva del proceso.
- El costo del cambio no depende de la fase o etapa.
- No introduce funcionalidades antes que sean necesarias.
- El cliente o el usuario se convierte en miembro del equipo (22).

Fases de la metodología XP

El ciclo de vida ideal consta de 5 fases:

- **Exploración:** los clientes plantean a grandes rasgos las Historias de Usuario que son de interés para la primera entrega del producto.

- **Planificación de Entregas:** se establece la prioridad de cada Historia de Usuario y los programadores realizan una estimación del esfuerzo necesario de cada una de ellas.

- **Iteraciones:** incluye varias iteraciones sobre el sistema antes de ser entregado. El plan de entrega está compuesto por iteraciones de no más de tres semanas.

- **Prueba:** requiere de pruebas adicionales y revisiones del rendimiento antes de que el sistema sea trasladado al entorno del cliente.

- **Mantenimiento:** mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones (22).

En el presente trabajo se decide utilizar XP porque es una metodología que se adecua perfectamente a las características que presenta el proyecto como son: requisitos muy cambiantes, existe un alto riesgo técnico, posee un equipo pequeño y poco tiempo de desarrollo. Además el cliente forma parte del equipo de desarrollo, lo que permite establecer un mejor vínculo de comunicación entre este y los desarrolladores, elevando así la calidad del sistema a desarrollar. Con el uso de esta metodología se consiguen productos fáciles de usar y con mayor rapidez, más fiables y robustos contra los fallos, gracias al diseño de las pruebas de forma previa a la codificación (22).

1.6 Herramientas y lenguaje de Programación

Hoy en día el uso de las herramientas informáticas en las empresas crea una ventaja competitiva al adaptarlas a sus funciones diarias logrando así una optimización en tiempo. Permite además evaluar fácilmente la calidad del producto o servicio que ofrece la empresa. A continuación se describen las

herramientas y el lenguaje de programación seleccionado para el desarrollo de los algoritmos de agrupamiento.

1.6.1 PostgreSQL

PostgreSQL engloba características que lo hacen considerablemente potente y robusto. Ofrece control de concurrencia multi-versión, soporta casi toda la sintaxis SQL, incluye además, subconsultas, transacciones y tipos y funciones definidas por el usuario. Permite agregar extensiones de DM. Soporta lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Además permite usar Perl, C, C++, Java, TCL y Python como lenguaje procedural (23).

La versión 9.3 de PostgreSQL incluye características que amplían la fiabilidad, disponibilidad y capacidad de integración con otras bases de datos:

- Sumas de verificación de páginas de datos: ayuda a administradores a detectar rápidamente discos con errores o hardware defectuoso que corrompe los datos.
- *Failover* más rápido en las réplicas para alta disponibilidad.
- *Streaming-only remastering*: reconfiguración de réplicas en cascada más fácil y rápida después de un *failover* (23).

PostgreSQL 9.3 incluye otras características para hacer el trabajo más fácil y flexible para los desarrolladores de aplicaciones, administradores y arquitectos de sistemas. Estas características incluyen:

- Métodos constructores y extractores adicionales para JSON.
- Vistas actualizables automáticas.
- Pg_dump en paralelo para acelerar el respaldo de bases de datos muy grandes.
- LATERAL JOINs.

PostgreSQL 9.3 es uno de los SGBD más potentes y robustos del mercado, funciona bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. La estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo (23).

1.6.2 PgAdmin III

PgAdmin III es un cliente gráfico que posee código abierto para la administración de bases de datos PostgreSQL. Está diseñado para responder a las necesidades de todos los usuarios. La interfaz gráfica soporta las características de PostgreSQL e incluye varios sistemas operativos como Windows, Linux, FreeBSD, Mac OSX y Solaris. Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como *Pervasive Postgres*, *EnterpriseDB*, *Mammoth Replicator* y *SRA PowerGres*. Entre sus principales funcionalidades se encuentran que posee herramienta de consulta SQL, editor de código procedural, agente de planificación SQL/shell/batch y soporte para el motor de replicación *Slony-I*. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix) y puede encriptarse mediante SSL para mayor seguridad (24).

1.6.3 PL/pgSQL

PL/pgSQL es un lenguaje procedural para el sistema de base de datos PostgreSQL. Este lenguaje tiene como propósito crear funciones y procedimientos, disparadores, adicionar estructuras de control al lenguaje SQL, realizar cálculos complejos, heredar todos los tipos, las funciones y los operadores definidos por el usuario, definirse como confiable por el servidor y ser fácil de usar. PL/pgSQL permite agrupar un bloque de cálculos y una serie de consultas dentro del servidor de la base de datos, obteniendo de esta manera el poder de un lenguaje procedural y la facilidad de uso de SQL. Permite además, ejecutar comandos SQL mediante un lenguaje de sentencias imperativas y uso de funciones, dando mucho más control automático que las sentencias SQL básicas. Sus principales ventajas son mayor rendimiento, soporte SQL y portabilidad (25).

Una de las principales ventajas de ejecutar programación en el servidor de Base de Datos es que las consultas y el resultado no tienen que ser transportadas entre el cliente y el servidor, ya que los datos

residen en el propio servidor. Además, el gestor de Base de Datos puede planificar optimizaciones en la ejecución de la búsqueda y actualización de datos (25).

1.7 Conclusiones del capítulo

En este capítulo se abarcó de manera detallada todo lo referente al proceso de DM y se plantearon definiciones fundamentales para lograr una mejor comprensión de la misma. Se realizó una explicación sobre las herramientas para trabajar con DM más usadas y se decidió que es necesario implementar algoritmos de agrupamiento de datos para integrarlos al Gestor de Base de Datos PostgreSQL 9.3. Se seleccionó CRISP-DM como metodología del proceso de DM ya que es la que más se adapta para dar solución al problema planteado. También se seleccionó XP como metodología de desarrollo del *software* porque es la que más se adecua a las características que presenta el proyecto. Además se seleccionaron los algoritmos a implementar, los cuales son *K-means*, DBSCAN y *FarthestFirst*, que pertenecen a la técnica de agrupamiento de DM. Se propone como herramienta de administración de base de datos PgAdmin III y como lenguaje de programación PL/pgSQL nativo del Gestor de Base de Datos PostgreSQL 9.3.

CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN

Introducción

En el presente capítulo se describen las principales características que tendrá la extensión de los algoritmos de agrupamiento a implementar. La misma cuenta con el modelo de dominio con el fin de proporcionar un mejor entendimiento de los principales conceptos del negocio, también se definen las Historias de Usuario, en las cuales se planifican las iteraciones y se realiza una estimación del tiempo de su realización. Además se incluye la lista de reserva del producto donde se muestran los requisitos tanto funcionales como los no funcionales. También se describen brevemente los algoritmos implementados y se mostrarán imágenes de la implementación. Para finalizar el capítulo se explica la integración de los algoritmos al SGBD PostgreSQL.

2.1 Identificación del problema

El departamento PostgreSQL tiene como misión potenciar las funciones del Gestor de Base de Datos PostgreSQL, para contribuir con esto se van a agregar algunos algoritmos de agrupamiento de la DM, pues a la hora de realizar análisis no existe una técnica que permita clasificar o agrupar los datos numéricos que se almacenan. Integrarle estos algoritmos al SGDB hace que el sistema sea más potente a la hora realizar análisis de DM.

2.2 Modelo de dominio

A pesar de que la metodología de desarrollo de *software* XP no precisa una técnica específica para definir el negocio, con el fin de proporcionar un mejor entendimiento de los principales conceptos que se manejan en este se decide realizar un modelo de dominio. Este artefacto de la disciplina de análisis puede utilizarse para capturar y expresar el entendimiento ganado mediante el análisis como paso previo al diseño de un sistema. Este es utilizado por el analista como un medio para comprender de una manera mucho más fácil el sistema que se va a realizar. También puede ser tomado como el punto de partida para el diseño del sistema.

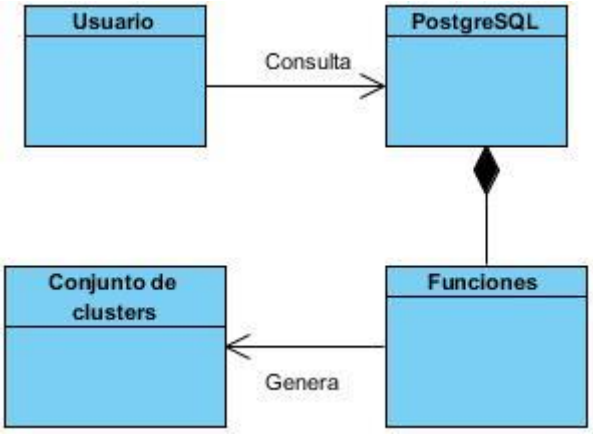


Fig.5 Modelo de dominio del sistema

Usuario: Persona que interactúa con la herramienta.
PostgreSQL: Servidor de bases de datos.
Funciones: Conjunto de funciones que se van a integrar al SGBD PostgreSQL.
Conjunto de clústeres: Conjunto de clústeres obtenido luego de ejecutar las consultas realizadas en el SGDB PostgreSQL.

2.3 Propuesta del componente a desarrollar

Se propone integrar algoritmos de agrupamiento al SGBD PostgreSQL, lo cual es de gran importancia pues no es necesario convertir los datos ni conectarse a otra herramienta. Además incorporando los algoritmos de agrupamiento se permite aprovechar las potencialidades del SGBD PostgreSQL para realizar el proceso de análisis de los datos.

2.4 Historias de Usuario

Las Historias de Usuario se utilizan para especificar los requisitos del *software*. En estas el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las Historias de Usuario es muy dinámico y flexible. Cada una de ellas es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Capítulo 2: Descripción de la Solución

En la metodología XP las Historias de Usuario sustituyen a los documentos de especificación funcional y a los casos de usos. Se usan para elaborar estimaciones de tiempo de desarrollo de la aplicación. Tienen la facilidad de permitir respuestas de forma rápida a los requisitos cambiantes que puedan surgir durante el desarrollo del *software*, gracias a esto no se derrocha tiempo en la gestión de los mismos y se pueden administrar cómodamente los requisitos de los usuarios. Además las Historias de Usuario deben tener un tiempo de desarrollo de programación estimado entre una y tres semanas, si esta estimación es mayor de tres semanas, esta historia debe ser dividida en dos o más, en el caso de que sea menos de una semana debe ser combinada con otra historia.

Para el desarrollo de los algoritmos de agrupamiento se definieron un total de cinco Historias de Usuario. A continuación se muestran en las siguientes tablas las historias de usuario definidas para el algoritmo *K-means*, las demás se encuentran en el expediente de proyecto.

Tabla.1 Historia de Usuario: Calcular los centroides y la distancia euclidiana del algoritmo *K-means*

Historia de Usuario	
Número: 1	**Nombre:** Calcular los centroides y la distancia euclidiana del algoritmo *K-means*
Cantidad de modificaciones: 1	
Usuario: Nathalie Solano Trepeu	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 semanas
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Se crea la misma cantidad de grupos que la cantidad de clúster pasada por parámetro, se calcula el centroide de cada grupo y la distancia euclidiana desde cada elemento a su centroide.	
Observaciones: N/A	

Prototipo de interfaz: N/A

Tabla.2 Historia de Usuario: Mostrar el resultado del algoritmo *K-means*

Historia de Usuario	
Número: 2	**Nombre:** Mostrar el resultado del algoritmo *K-means*
Cantidad de modificaciones: 1	
Usuario: Nathalie Solano Trepeu	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 semanas
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Se calcula iterativamente hasta converger los nuevos centroides y la distancia euclidiana y se muestran los resultados.	
Observaciones: N/A	
Prototipo de interfaz: N/A	

2.5 Lista de reserva del producto

La lista de Reserva del Producto es una tabla que contiene los requisitos funcionales que debe cumplir la aplicación, ordenados por la prioridad de implementación, así como los requisitos no funcionales del sistema a desarrollar. Además en esta tabla se refleja la estimación de cada uno de ellos y su implementación por semanas, definiendo de esta forma el rol que hizo la estimación.

Tabla.3 Lista de reserva del producto

Ítem	Descripción	Estimación	Estimado por
Requisitos Funcionales			

Prioridad: ALTA			
1	Calcular los centroides y la distancia euclidiana del algoritmo *K-means*	3 semanas	Analista
2	Mostrar el resultado del algoritmo *K-means*	3 semanas	Analista
Prioridad: MEDIA			
3	Encontrar los alcanzables de un punto seleccionado y clasificar dicho punto del algoritmo DBSCAN.	3 semanas	Analista
4	Expandir el grupo y mostrar el resultado del algoritmo DBSCAN.	3 semanas	Analista
5	Generar un conjunto de clústeres a través del algoritmo FarthestFirst	2 semanas	Analista
Requisitos No Funcionales			
1	*Software*: Para utilizar la extensión debe estar instalado, el gestor PostgreSQL a partir de la versión 9.3.		
2	*Hardware*: - El ordenador donde se utilice la extensión debe contar con 256 Mb de memoria RAM como mínimo, un microprocesador de 300MHz de frecuencia.		

2.6 Tareas de la ingeniería

Las Tareas de la Ingeniería (TI) se usan para describir las tareas que se realizan sobre el proyecto. Estas pueden ser de: desarrollo, corrección, mejora y otra. Las TI tienen relación con una Historia de Usuario, aunque de una Historia de Usuario se puede derivar más de una tarea de la ingeniería. En estas se especifica la fecha de inicio y fin de la tarea, se nombra al programador responsable de cumplirla, también aparece el tiempo estimado que se demorará en realizar las tareas, además se

realiza un pequeña descripción de lo que se tratará de hacer en la tarea. A continuación se muestran las tareas de ingeniería correspondiente a la Historia de Usuario: Calcular los centroides y la distancia euclidiana del algoritmo *K-means*, el resto se encuentra en el expediente de proyecto.

Tabla.4 Tareas de la ingeniería: Estudio de los pasos del algoritmo *K-means*

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea : Estudio de los pasos del algoritmo *K-means*	
Tipo de Tarea : Estudio	**Puntos Estimados:** 1 semana
Fecha Inicio: 12/02/2014	**Fecha Fin:** 17/02/2014
Programador Responsable: Nathalie Solano Trepeu	
Descripción: Realizar un estudio de la documentación del algoritmo de agrupamiento *K-means* de la DM.	

Tabla.5 Tareas de la ingeniería: Crear los grupos y asignarle los valores de la tabla

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 1
Nombre Tarea : Crear los grupos y asignarle los valores de la tabla	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 17/02/2014	**Fecha Fin:** 24/02/2014
Programador Responsable: Nathalie Solano Trepeu	

Descripción: Se cargan los datos de la tabla sobre la cual se va a trabajar, se introduce la cantidad de clúster y la cantidad de iteraciones máxima. Además se crean igual cantidad de grupos que la cantidad de clúster pasado por parámetro y se distribuye por los grupos los valores de la tabla.

Tabla.6 Tareas de la ingeniería: Calcular los centroides y la distancia euclidiana

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** 1
Nombre Tarea : Calcular los centroides y la distancia euclidiana	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 24/02/2014	**Fecha Fin:** 1/03/2014
Programador Responsable: Nathalie Solano Trepeu	
Descripción: Se calcula los centroides de cada grupo, se halla la distancia euclidiana desde cada elemento al centroide y se asigna el elemento al centroide más cercano.	

2.7 Plan de iteraciones

Una vez definidas las historias de usuario es necesario crear un plan donde se indiquen las Historias de Usuario que se implementarán para cada versión del programa y las fechas en las que se publicarán estas versiones. Para esto se crea el plan de iteraciones que es una planificación donde los

desarrolladores y clientes establecen los tiempos de implementación ideales de las Historias de Usuario, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa. Después de un plan de iteraciones tienen que estar claros estos cuatro factores: los objetivos que se deben cumplir (que son principalmente las historias que se deben desarrollar en cada versión), el tiempo que tardarán en desarrollarse y publicarse las versiones del programa, el número de personas que trabajarán en el desarrollo y cómo se evaluará la calidad del trabajo realizado.

Tabla.7 Plan de iteraciones

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
Iteración 1	En esta iteración se implementarán las Historias de Usuario que tengan la prioridad en el negocio ALTA.	1,2	6 semanas.
Iteración 2	En esta iteración se implementarán las Historias de Usuario que tengan la prioridad en el negocio MEDIA.	3,4,5	8 semanas.

2.8 Estándares de codificación

XP propone la definición de un estándar de programación para mantener un código legible. Esto beneficia la comunicación de los programadores a través del código y aún más si la implementación se realiza en pareja. Los estándares de codificación son modelos de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código.

Identación

La unidad de identado es de 4 espacios. El uso de la tabulación debe ser evitado pues no existe un estándar que determine con precisión el ancho que va a producir la tabulación.

```
WHILE(columna < cantidad_column+1)
LOOP
    EXECUTE ' SELECT AVG(c' ||columna|| ') FROM grupo_' ||var INTO centroide;
    EXECUTE ' UPDATE centroides SET c'||columna|| ' = '||centroide::numeric||' WHERE id_grupo='||var;
    columna:=columna+1;
END LOOP;
```

Fig.6 Ejemplo de identación en la implementación del algoritmo *K-means*

Comentarios

Es conveniente dejar información que pueda ser leída tiempo después por personas (posiblemente la misma persona que programó) que necesitan entender que fue lo que se hizo en el fragmento de código. Los comentarios deben ser escritos correctamente y claros. Generalmente deben usarse comentarios de una sola línea. Se debe reservar los comentarios de bloques para la documentación formal o para comentar porciones de código.

```
BEGIN

-- Cantidad de filas existentes en la tabla
EXECUTE 'SELECT COUNT(*) FROM '||$1 INTO cantidad_filas;

-- Cantidad de columnas existentes en la tabla
SELECT count(*) INTO cantidad_column FROM information_schema.columns WHERE table_name = $1;

-- Devuelve el nombre de las columnas existentes en la tabla
OPEN mi_cursor FOR SELECT column_name FROM information_schema.columns WHERE table_name = $1;
```

Fig.7 Ejemplo de comentarios en la implementación del algoritmo *K-means*

Declaración de variables

Cada variable debe de ser declarada en una línea y comentada. El nombre de las variables debe de comenzar con letras minúsculas y cada palabra relevante por la que esté compuesta debe ser con letra minúscula y separada por un guión bajo. Cada variable que sea declarada estará comentada para lograr un mejor entendimiento.

```
cantidad_filas INTEGER DEFAULT 0;      -- variable que contiene la cantidad de filas que tiene la tabla
cantidad_filas1 INTEGER DEFAULT 0;     -- variable que contiene la cantidad de filas que tiene los grupos
cantidad_column INTEGER DEFAULT 0;     -- variable que contiene la cantidad de columnas que tiene la tabla
centroide NUMERIC DEFAULT 0;           -- contiene el valor del centroide
distancia_euclideana NUMERIC DEFAULT 0;-- variable que contiene la distancia euclideana
nro INTEGER DEFAULT 0;                 --variable para incrementar el offset
```

Fig.8 Ejemplo de declaraciones de variables en la implementación del algoritmo *K-means*

Identificadores

Los identificadores pueden estar formados por cualesquiera de las 26 letras minúsculas o mayúsculas (A... Z, a... z), los 10 dígitos (0... 9) y el carácter subrayado "_". Debe evitarse el uso de caracteres internacionales (ej. ñ, ü) porque no siempre pueden ser leídos o entendidos correctamente en todos los lugares. No se debe usar el símbolo dólar "$" o la barra invertida "\" en los identificadores.

Sentencias Simples

Cada línea debe contener como máximo una sentencia. Debe poner un punto y coma ";" al final de cada sentencia simple. Tenga en cuenta que una sentencia de asignación puede resultar en la asignación de una función y en todos los casos como sentencia de asignación debe estar finalizada con un punto y coma.

2.9 Implementación de los algoritmos

En este epígrafe se muestra una pequeña descripción de los algoritmos implementados que serán integrados al SGBD PostgreSQL, así como un fragmento de código de dicha implementación.

2.9.1 Algoritmo K-means

La implementación del algoritmo *K-means* se realizó utilizando el pseudocódigo mostrado en la Fig.9. La función *k-means* sólo permite trabajar con tablas que tengan atributos numéricos y toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar, la cantidad de clúster y la

cantidad de iteraciones. Se devuelve como resultado un conjunto de grupos y sus respectivos centroides. A continuación se muestra un fragmento de código de la función implementada:

```
EXECUTE ' SELECT COUNT(*) FROM grupo_' ||var INTO cantidad_filas1;
WHILE(cantidad_filas1>0)
LOOP
    EXECUTE 'CREATE TEMP TABLE tabla (id INTEGER , distancia NUMERIC)';

    WHILE(a<>$2)
        LOOP
            CREATE TEMP TABLE distancia_euc( d NUMERIC);
            columna:=1;
            WHILE(columna < cantidad_column+1)
            LOOP
                EXECUTE ' SELECT c'||columna||' FROM  copia_grupo_' ||var|| ' OFFSET ' ||pos|| ' limit 1' INTO x;
                EXECUTE ' SELECT c'||columna||' FROM  centroides OFFSET '||nro|| ' limit 1 ' INTO y;

                valor:=power((x-y),2);
                INSERT INTO  distancia_euc VALUES (valor);
                columna:=columna+1;
            END LOOP;

            SELECT INTO distancia_euclideana sum(d)FROM distancia_euc;
            DROP TABLE distancia_euc;

            EXECUTE ' UPDATE  grupo_' ||var|| ' SET centroide_' ||i||' =  sqrt('||distancia_euclideana||') WHERE i
            EXECUTE 'INSERT INTO  tabla VALUES ('||i||',sqrt('||distancia_euclideana::NUMERIC||'))';
            nro:=nro+1;
            i:=i+1;
            a:=a+1;
        END LOOP;

        EXECUTE ' SELECT id FROM tabla WHERE distancia = (SELECT MIN(distancia) FROM tabla )' INTO id1;
        EXECUTE ' UPDATE  grupo_' ||var|| ' SET clasificacion= ''centroide_''||''||id1||' WHERE id_grupo = '||id;
        EXECUTE ' DROP TABLE tabla';
```

Fig.9 Fragmento de código de la implementación del algoritmo *K-means*

2.9.2 Algoritmo DBSCAN

La implementación del algoritmo DBSCAN se realizó utilizando el pseudocódigo mostrado en la Fig. 10. La función dbscan sólo permite trabajar con tablas que tengan atributos numéricos y toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar, el radio máximo de

vecindad de un punto y el número mínimo de puntos en un vecindario de ese punto. Se devuelve como resultado un conjunto de grupos y la clasificación de cada punto. A continuación se muestra un fragmento de código de la función implementada:

```
SELECT INTO colum * FROM columna;
FOR temptabla IN EXECUTE ' SELECT * FROM temp_tabla OFFSET ' ||nro1
LOOP
    IF(cantidad_column=1)THEN
        distancia_euclideana:=sqrt(power(abs((temptabla.c1-colum.c1)),2));
    ELSEIF(cantidad_column>1)THEN
        CREATE TEMP TABLE distancia( d NUMERIC);
        var:=1;
        WHILE(var<cantidad_column+1)
        LOOP
            EXECUTE ' SELECT c'||var||' FROM  temp_tabla OFFSET ' ||nro1 INTO a;
            EXECUTE ' SELECT c'||var||' FROM  columna ' INTO b;
            valor:=power((a-b),2);
            INSERT INTO  distancia VALUES (valor);
            var:=var+1;
        END LOOP;
        SELECT INTO distancia_euclideana sum(d)FROM distancia;

        DROP TABLE distancia;
    END IF;
    IF (sqrt(distancia_euclideana) <=$2 and sqrt(distancia_euclideana) > 0) THEN
        EXECUTE 'INSERT INTO columna (SELECT * FROM temp_tabla OFFSET ' ||nro1||' LIMIT 1 )';
    END IF;
    nro1:=nro1+1;
END LOOP;


EXECUTE ' SELECT COUNT (*) FROM  columna '  INTO cantidad_vecinos;
IF(cantidad_vecinos-1 >=$3) THEN
    EXECUTE ' UPDATE columna SET clasificacion= ''central'' WHERE id =1';

ELSEIF(cantidad_vecinos-1 < $3 AND cantidad_vecinos-1 > 0) THEN
    EXECUTE ' UPDATE columna SET clasificacion= ''frontera'' WHERE id=1';
ELSE
    EXECUTE ' UPDATE columna SET clasificacion= ''ruido'' WHERE id=1';
END IF;
EXECUTE ' SELECT clasificacion FROM columna WHERE id=1' INTO tipo;
```

Fig.10 Fragmento de código de la implementación del algoritmo DBSCAN

2.9.3 Algoritmo FarthestFirst

La implementación del algoritmo *FarthestFirst* se realizó utilizando el pseudocódigo mostrado en la Fig. 11. La función farthestfirst sólo permite trabajar con tablas que tengan atributos numéricos y toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar y la cantidad de

clúster deseado. Se devuelve como resultado un conjunto de grupos y los centroides de cada uno de ellos. A continuación se muestra un fragmento de código de la función implementada:

```
WHILE(var1< $2+1)
LOOP
    pos:=0;
    ident:=1;
    CREATE  TABLE tabla (id SERIAL , distancia NUMERIC);
    EXECUTE ' SELECT COUNT(*) FROM copia' INTO cantidad_filas; --CALCULAR DISTANCIAS
    WHILE(cantidad_filas >0)
    LOOP
        CREATE TEMP TABLE distancia_euc( d NUMERIC);
        var:=1;
        WHILE(var < cantidad_column+1)
        LOOP
            EXECUTE ' SELECT c'||var||' FROM  copia OFFSET ' ||pos|| ' LIMIT 1' INTO x;
            EXECUTE ' SELECT c'||var||' FROM  centroides OFFSET ' ||i|| ' LIMIT 1' INTO y;

            valor:=power((abs(x-y)),2);
            INSERT INTO  distancia_euc VALUES (valor);
            var:=var+1;
        END LOOP;
        SELECT INTO distancia_euclideana SUM(d)FROM distancia_euc;
        DROP TABLE distancia_euc;
        EXECUTE ' UPDATE  copia_original SET distancia_centroide_' ||i+1|| ' =  sqrt('||distancia_euclideana
        EXECUTE ' INSERT INTO tabla (distancia) VALUES (sqrt('||distancia_euclideana::NUMERIC||'))';
        ident:=ident+1;
        pos:=pos+1;
        cantidad_filas:=cantidad_filas-1;
    END LOOP;
    EXECUTE ' ALTER TABLE  copia ADD COLUMN id SERIAL';
    EXECUTE ' ALTER TABLE  centroides ADD COLUMN id SERIAL';
    EXECUTE ' SELECT id FROM tabla WHERE distancia = (SELECT MAX(distancia) FROM tabla )' INTO id1;
    EXECUTE ' SELECT COUNT(*) FROM centroides' INTO cantidad_filas1;
    IF(cantidad_filas1 < $2)THEN
        EXECUTE ' INSERT INTO centroides ( SELECT * FROM copia WHERE id =' ||id1||' )';
        DELETE FROM copia WHERE id=id1;
        DELETE FROM copia_original WHERE id=id1;
    END IF;
```

Fig.11 Fragmento de código de la implementación del algoritmo FarthestFirst

2.10 Integración de los algoritmos al SGBD PostgreSQL

La versión 9.3 de PostgreSQL permite añadir nuevas funcionalidades en el SGBD PostgreSQL, ahora los usuarios pueden crear, cargar, actualizar y administrar fácilmente las docenas de extensiones disponibles utilizando el objeto de base de datos EXTENSION (23).

Una extensión es un complemento que sirve para la integración de aplicaciones obteniéndose una nueva función, esto le permite a los desarrolladores interactuar con la aplicación y así aumentar la cantidad de funcionalidades que se puedan realizar.

La integración de los algoritmos implementados con el SGBD se va a realizar mediante la creación de una extensión por las ventajas que PostgreSQL brinda para su creación. Entre estas ventajas se encuentra, que en lugar de ejecutar un script SQL para cargar objetos que estén "separados" en su base de datos, se tendrá la extensión como un paquete que contendrá todos los objetos definidos en ella. Esto trae como beneficio que al actualizar o eliminar la extensión se pueden eliminar todos los objetos utilizando el comando DROP EXTENSION sin necesidad de especificar cada uno de los mismos definidos dentro de ella. Además se cuenta con un repositorio para obtener extensiones y contribuir con estas.

2.10.1 Creación de la extensión de los algoritmos de agrupamiento

Para la creación de la extensión se crean dos archivos, en el primero se definen las características de la extensión y en el segundo los objetos SQL que se desean agregar. Los mismos deben ser ubicados dentro del directorio de la instalación "C:\Archivos de programa\PostgreSQL\9.3\share\extension".
En el archivo "algoritmos_agrupamiento.CONTROL" creado para agregar la extensión donde se cargarán las funciones de los algoritmos implementados se definieron los siguientes parámetros:

- **Comment:** Una breve descripción sobre el contenido de la extensión creada.
- **Encoding:** El tipo de codificación utilizado.
- **Default_version:** La versión de la extensión.
- **Schema:** El esquema donde se almacenarán los objetos creados por la extensión.

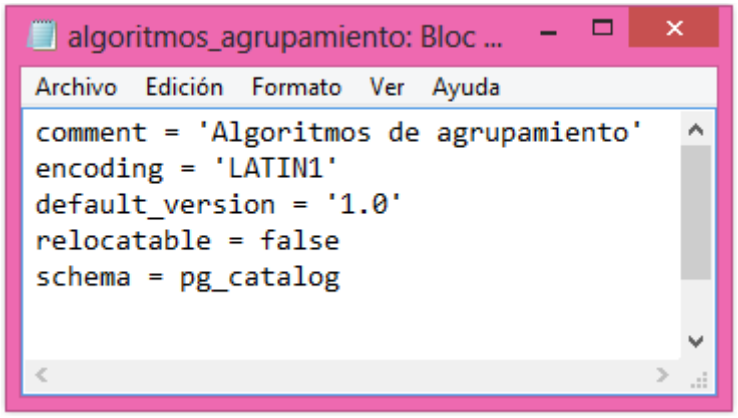


Fig.12 Archivo que contiene las características de la extensión

Una vez definido en el archivo "algoritmos_agrupamiento.control" se especifica el archivo que contendrá el código de las funciones desarrolladas "algoritmos_agrupamiento--1.0.sql".

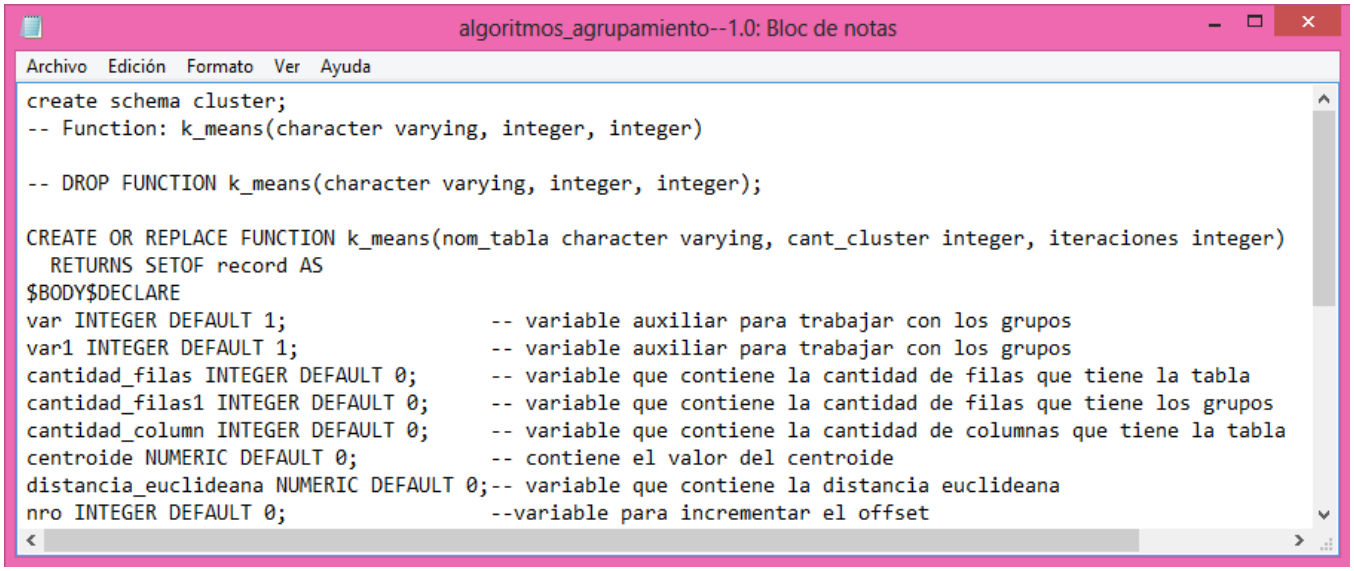


Fig.13 Archivo que contiene el código de la extensión

Para que los usuarios puedan utilizar la extensión de algoritmos de agrupamiento de DM simplemente deben ejecutar el comando "CREATE EXTENSION algoritmos_agrupamiento" que cargará la extensión como se muestra en la siguiente figura:

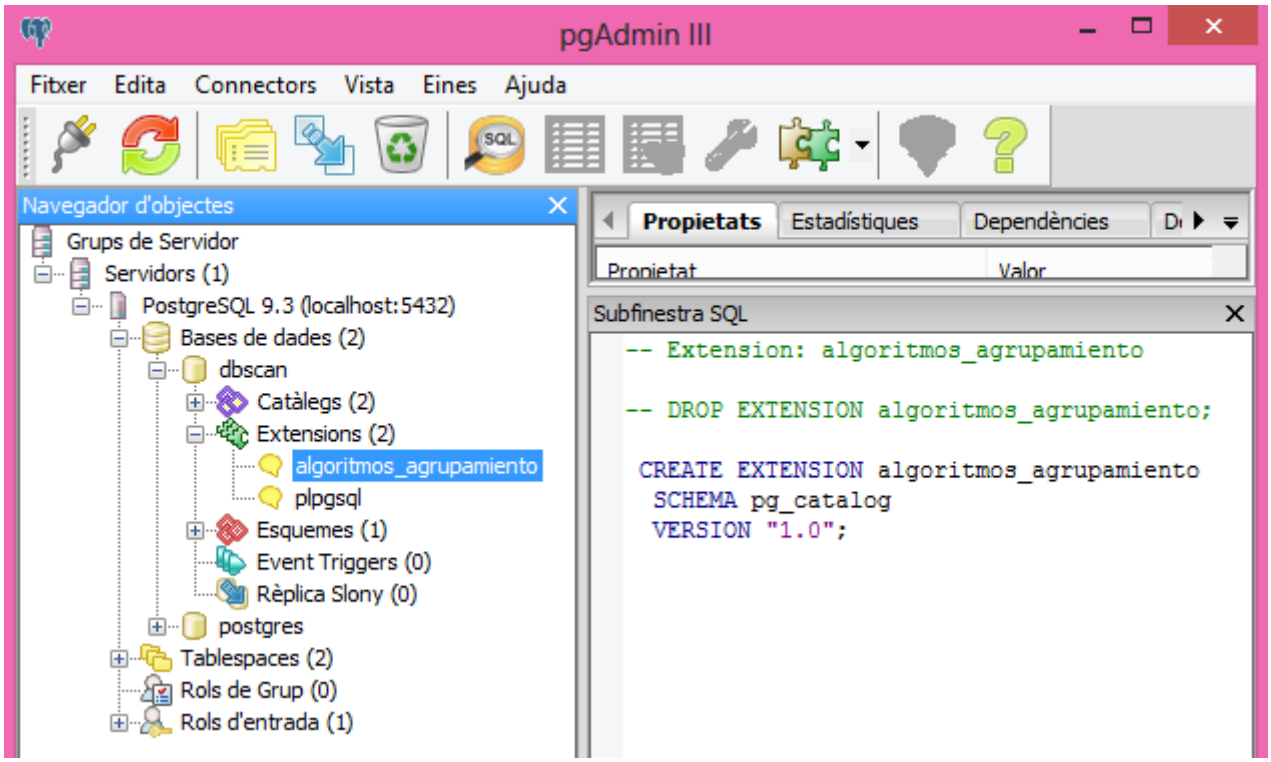


Fig.14 Extensión “algoritmos_agrupamiento” creada

2.10 Conclusiones del capítulo

En el presente capítulo se describió la solución propuesta para el problema de la investigación, para ello se elaboró el modelo de dominio, para así lograr un mejor entendimiento del desarrollo de la aplicación. Se definieron un total de catorce tareas de la ingeniería agrupadas en cinco Historias de Usuario. También se generaron otros artefactos que concibe la metodología XP como son: la lista de reserva del producto y el plan de iteraciones, los cuales son muy importantes ya que establecerán el avance de la aplicación y además se realizaron los estándares de codificación. Para finalizar el capítulo se analizaron todos los elementos que describen las características y diseño de la extensión y se describió cómo a través de su creación, se integraron los algoritmos al SGDB PostgreSQL 9.3.

CAPÍTULO 3: APLICACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Introducción

En la metodología XP se definen un grupo de normas para la validación de los productos, logrando que los mismos puedan ser probados para la verificación de su correcto funcionamiento. Se analizan en este capítulo las técnicas que definen la metodología XP para diseñar los casos de pruebas que guiarán la validación de la aplicación. Además se aplica el proceso de DM utilizando la metodología CRISP-DM y se compara los resultados de los algoritmos implementados con otra herramienta.

3.1 Métodos de Prueba

La fase de pruebas es fundamental en el desarrollo de una aplicación. El objetivo de cada una de ellas no es el de prevenir errores sino de detectarlos basándose en técnicas y estrategias empleadas en cada una de las pruebas. Existen disímiles estrategias de pruebas, de todas ellas serán usadas en la presente investigación las correspondientes a la metodología de desarrollo de *software* XP. La misma divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores y pruebas de aceptación o pruebas funcionales. Estas pruebas se encargan de evaluar el producto al final de una iteración, para así verificar que este cumpla con las condiciones diseñadas por el cliente final.

3.1.1 Pruebas unitarias o Caja Blanca

Las pruebas unitarias se centran en la estructura de control del programa, para ello realizan un seguimiento del código fuente según se van ejecutando los casos de prueba, de manera que se determinan de forma concreta las instrucciones y bloques en los que existen errores. Estas facilitan al desarrollador cambiar el código para mejorar su estructura y permite asegurarse que los cambios no han introducido errores, por esto se dice que fomentan el cambio. Es necesario aclarar que las

pruebas unitarias no sacarán a la luz todos los errores del código. Generalmente se utilizan para dar paso a las pruebas de integración y para que sean efectivas deben realizarse en conjunto con otras pruebas de *software*. Además que, después del más cuidadoso proceso de pruebas por parte del desarrollador, quedan una serie de errores que sólo aparecen cuando el cliente comienza a usarlo. Mediante los casos de prueba de caja blanca se pueden derivar casos de prueba que garanticen:

- Que se ejecutan las estructuras de datos internas para asegurar su validez.
- Que se ejercitan todas las decisiones lógicas en sus vértices verdaderos y falsos.
- Que se ejecutan todos los bucles en sus límites y con sus límites operacionales.
- Que se ejecutan por lo menos una vez todos los caminos independientes de cada módulo (26).

En ocasiones resultan necesarias realizar las pruebas de caja blanca para detectar errores que se producen ante las siguientes situaciones:

- Los errores lógicos y suposiciones, los cuales son proporcionales a la probabilidad de que se ejecute algún cambio en el programa.
- Cuando un camino lógico tiene pocas posibilidades de ejecutarse y de hecho se ejecuta de forma normal y esto significa o conlleva a que las suposiciones sobre el flujo de control y datos conlleven a errores de diseño y se encuentran solamente cuando comienza la prueba de camino básico.

También se pueden encontrar errores tipográficos mediante estas pruebas, lo cual es casi imposible de detectar mediante otro tipo de prueba (26).

3.1.2 Pruebas funcionales o Caja Negra

Las pruebas de caja negra se aplican a la interfaz del *software*, cuando se conoce bien la función específica del producto, para demostrar que cada funcionalidad del mismo es plenamente operacional. Es decir, este tipo de pruebas se ejecutan para examinar aspectos funcionales, tiene poca relación con la estructura interna del *software*. Además permiten obtener conjuntos de condiciones de entrada, que

ejerciten completamente todos los requisitos funcionales de un programa. Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del *software* son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene (26).

Se decide aplicar la técnica de pruebas de caja negra pues el mayor interés de los investigadores radica en identificar la mayor cantidad de defectos en las funcionalidades definidas para el sistema, con el objetivo de erradicarlos antes de que el producto sea entregado al cliente, logrando con esto, aumentar la calidad del producto y satisfacer las expectativas del cliente. La técnica de prueba a ejecutar puede descubrir errores de funciones incorrectas o ausentes entre las que se encuentran:

- Funciones que estén incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación (26).

A continuación se muestra un ejemplo de un caso de prueba que fue utilizado en el algoritmo *K-means* para la realización de las prueba.

Tabla.8 Caso de Prueba: Algoritmo *K-means*

Escenario	Descripción	Respuesta del sistema	Flujo central

EC 1.1 Ejecutar el algoritmo con datos correctos.	En este escenario se crean los grupos introduciendo datos correctos.	Se muestran los grupos creados.	1- El usuario selecciona la función k-means. 2- Introduce los datos requeridos para la función k-means. 3- Se generan los grupos para la función k-means.
EC 1.2 Ejecutar el algoritmo con datos incorrectos.	En este escenario se crean los grupos introduciendo datos incorrectos.	Se muestra un mensaje de error en la herramienta.	
EC 1.3 Ejecutar el algoritmo con datos vacíos.	En este escenario se crean los grupos dejando datos a insertar vacíos.	Se muestra un mensaje de error en la herramienta.	

3.2 Presentación de los resultados de las pruebas funcionales

Las pruebas han sido aplicadas a las cinco Historias de Usuario permitiendo detectar varios errores. Se encontraron en una primera iteración seis no conformidades las cuales fueron resueltas en siete días. Para una segunda iteración se encontraron dos no conformidades, la cual fue solucionada en un período de dos días. Finalmente para una tercera iteración se obtuvieron resultados satisfactorios, encontrándose cero no conformidades. En la siguiente figura se muestra una gráfica con la cantidad de no conformidades encontradas y resueltas en cada iteración.

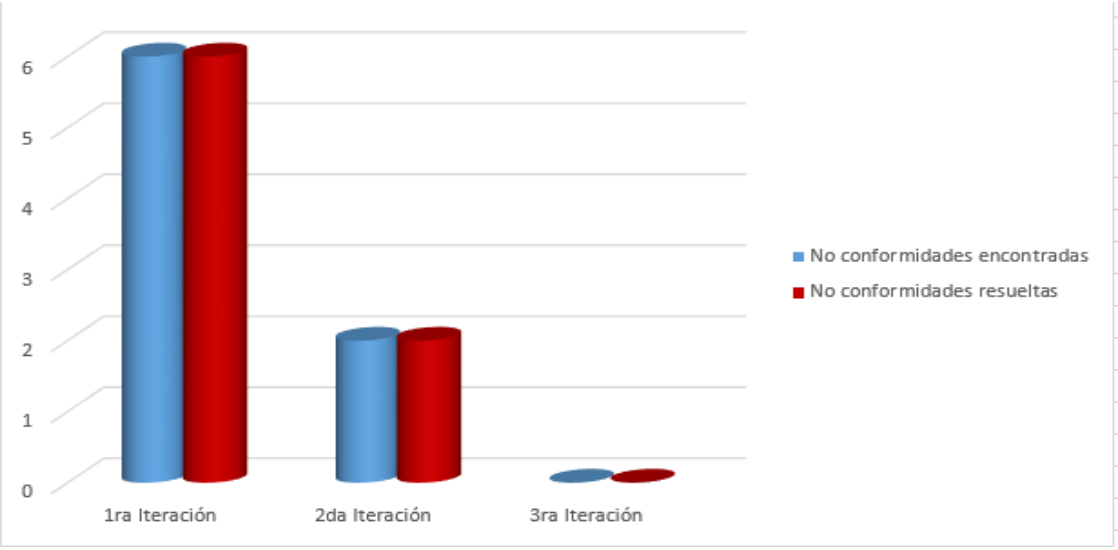


Fig.15 Resultado de las pruebas aplicadas

A continuación se muestra como se aplicó el proceso de DM utilizando la metodología CRISP-DM para validar los algoritmos integrados al SGBD PostgreSQL.

3.3 Análisis del problema

Los criterios para lograr el éxito de la investigación desde el punto de vista del **objetivo del negocio** son:

- Obtener conjuntos de grupos y comprobar su validez**.**
- Realizar un proyecto de minería de datos guiado por la metodología CRISP-DM.

Como **objetivo de Minería de Datos** se identificó:

Obtener grupos bien concentrados con alto nivel de aceptación de acuerdo a la herramienta Weka de la DM.

3.4 Análisis de los datos

El análisis de los datos está relacionado con la recolección y descripción de la información inicial con la que se comienza el proceso de obtener conocimiento, una vez establecidos los objetivos a seguir. Además, se desarrollan actividades que permiten su exploración, a fin de identificar problemas con su calidad (3).

La información recopilada son dos juegos de datos diferentes, por un lado se usa los datos del *iris.arff* y por otro *mfeat-fourier.arff*. Iris es quizás la base de datos más conocida que se encuentra en la literatura de reconocimiento de patrones, es un caso de prueba típica para muchas técnicas de clasificación. Esta contiene 3 clases de 50 casos cada una, donde cada clase se refiere a un tipo de planta de iris. Por su parte *mfeat-fourier* se compone de características de los números escritos a mano (0 - 9). Es extraído de una colección de mapas holandeses de servicios públicos, cuenta con 10 clases de 200 casos cada una y 77 atributos. A continuación se describen los atributos significativos de la tabla Iris para darle solución a los objetivos propuestos:

Tabla.9 Atributos significativos de los datos recopilados

Nombre del atributo	Tipo de dato	Descripción	Nombre de la tabla
sepallength	Real	Almacena la longitud en cm del sépalo del iris	Iris
sepalwidth	Real	Almacena el ancho en cm del sépalo del iris	Iris
petallength	Real	Almacena la longitud en cm del pétalo del iris	Iris
petalwidth	Real	Almacena el ancho en cm del pétalo del iris	Iris
clases	character varying	Almacena el tipo de iris: setosa, versicolor, virginica.	Iris

Debido a que se están utilizando juegos de datos empleados para probar los algoritmos de DM, en *mfeat-fourier* todos los atributos se consideran significativos para obtener un mejor resultado de los mismos.

Explorar y verificar la calidad de los datos

Esta tarea tiene un peso importante pues tiene como objetivo corroborar si la información recolectada es lo suficientemente sólida o no para satisfacer las necesidades del minero. Para decidir esta cuestión es necesario analizar si los datos contienen errores y si el contenido del campo describe realmente lo que este almacena. Para verificar la calidad de los datos se tuvo en cuenta que los mismos no tuvieran campos innecesarios, inconsistencias, campos vacíos y datos erróneos.

3.5 Preparación de los datos

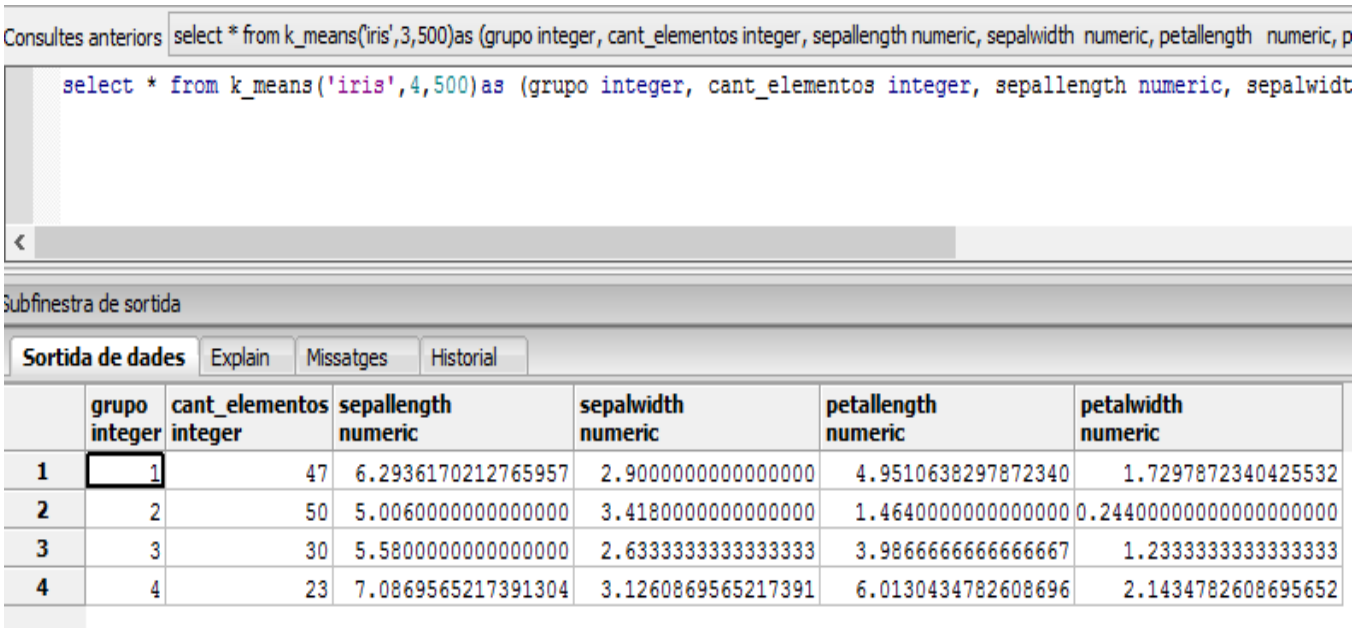
El objetivo de esta fase es listar los atributos que serán incluidos o excluidos del proceso de minería de datos. Los atributos seleccionados para el proyecto de minería en el caso de Iris son sepallength, sepalwidth, petallength y petalwidth. Se decidió excluir el atributo clases porque con este la separación

de los grupos es bastante obvia. En el caso de mfeat-fourier se consideran necesarios todos los atributos y por ende no se excluye ninguno.

3.6 Modelado y evaluación

La selección de las técnicas de DM y algoritmos a emplear depende de los objetivos de la DM propuestos en la fase de análisis del problema. La técnica seleccionada para darle cumplimiento a este objetivo es agrupamiento y de ella los algoritmos *K-means*, DBSCAN y *FarthestFirst*. A continuación se muestran los resultados obtenidos.

Presentación de los resultados obtenidos con el algoritmo *K-means*



	grupo integer	cant_elementos integer	sepallength numeric	sepalwidth numeric	petallength numeric	petalwidth numeric
1	1	47	6.2936170212765957	2.9000000000000000	4.9510638297872340	1.7297872340425532
2	2	50	5.0060000000000000	3.4180000000000000	1.4640000000000000	0.24400000000000000000
3	3	30	5.5800000000000000	2.6333333333333333	3.9866666666666667	1.2333333333333333
4	4	23	7.0869565217391304	3.1260869565217391	6.0130434782608696	2.1434782608695652

Fig.16 Resultado obtenido del algoritmo *K-means* en el SGBD PostgreSQL 9.3

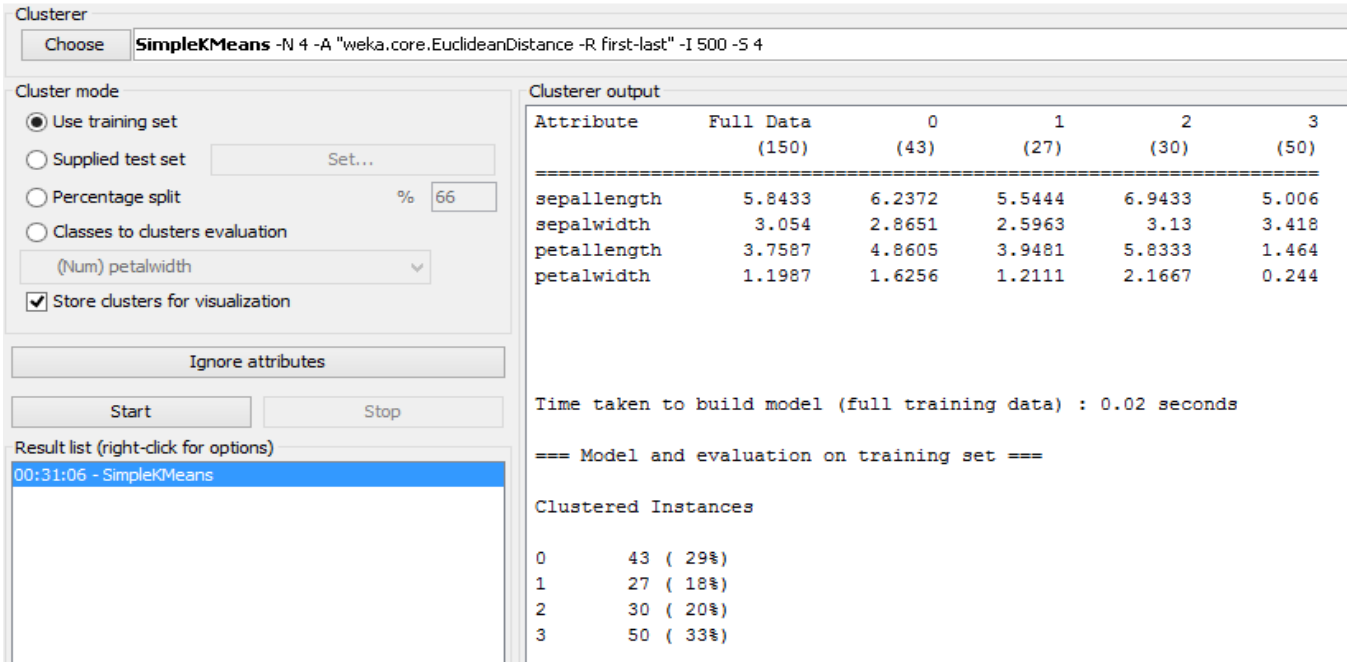


Fig.17 Resultado obtenido del algoritmo *K-means* en el Weka

Como se puede observar, los resultados obtenidos en ambas herramientas tienen una pequeña variación en dos de sus grupos, esto se debe al valor de la semilla que no es más que el criterio de selección inicial de los centroides que propone la herramienta Weka. En la misma los resultados cambian en dependencia de la semilla seleccionada, para esta prueba se utilizó con valor de 4. En la siguiente tabla se muestran los resultados obtenidos luego de ejecutar varias veces el algoritmo con diferentes semillas.

Tabla.10 Resultados obtenidos con varias semillas del algoritmo *K-means* en el Weka

S1	S3	S5	S7	S9	S11	S13	S15	S17
27	48	29	27	27	61	31	42	31
31	52	29	31	42	39	42	27	42
42	23	42	50	50	27	27	50	27

50	27	50	42	31	23	50	31	50

En la tabla se puede observar que a medida que cambian las semillas se obtienen una nueva distribución de los grupos, por lo que el resultado no siempre es el mismo. A continuación se muestra en la siguiente tabla una comparación de los resultados obtenidos del algoritmo *K-means* implementado en el SGBD PostgreSQL 9.3 y en la herramienta Weka.

Tabla.11 Comparación en PostgreSQL y Weka para el algoritmo *K-means*

Indicadores	*K-means* en PostgreSQL	*K-means* en Weka
Facilidad de uso	➢ Requiere de conocimientos de bases de datos para el análisis de los datos.	➢ Depende de especialistas para la manipulación de la herramienta además de conocimientos de bases de datos para el análisis de los datos.
Calidad de los datos	➢ Alta	➢ Alta
Manipulación de los datos	➢ El SGBD no necesita driver de conexión. ➢ Los datos están listos para ser analizados.	➢ Necesita un driver para la conexión al SGBD. ➢ Convertir los datos al formato arff.
Actualización de los datos	➢ Ejecutar el algoritmo.	➢ Necesita conectarse nuevamente a SGBD, volver a cargar los datos porque no actualiza o generar otra vez el archivo arff para después correr el algoritmo.
Comprensión de los resultados	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.

Como se muestra en la Tabla.11 el algoritmo *K-means* implementado en el SGBD PostgreSQL 9.3 es más eficiente en lo que se refiere a los parámetros evaluados, por lo tanto el uso de este algoritmo para el SGBD proporciona mejores ventajas para el análisis de los datos.

Presentación de los resultados obtenidos con el algoritmo DBSCAN

Consultes anteriors

```
select * from  dbscan('iris',0.9,6)as ( sepallength numeric, sepalw
```

Subfinestra de sortida

Sortida de dades | Explain | Missatges | Historial

	sepallength numeric	sepalwidth numeric	petallength numeric	petalwidth numeric	clasificacion character varying	grupo integer
89	6.4	2.8	5.6	2.1	central	0
90	6.0	2.2	4.0	1.0	central	0
91	5.9	3.2	4.8	1.8	central	0
92	5.5	2.3	4.0	1.3	central	0
93	6.5	3.0	5.8	2.2	central	0
94	6.3	3.4	5.6	2.4	central	0
95	6.2	3.4	5.4	2.3	central	0
96	5.7	3.0	4.2	1.2	central	0
97	6.1	2.8	4.0	1.3	central	0
98	6.9	3.1	4.9	1.5	central	0
99	6.4	2.9	4.3	1.3	central	0
100	5.0	3.4	1.6	0.4	central	1
101	4.6	3.1	1.5	0.2	central	1
102	5.1	3.5	1.4	0.2	central	1
103	5.1	3.5	1.4	0.3	central	1
104	4.9	3.1	1.5	0.1	central	1
105	4.7	3.2	1.3	0.2	central	1
106	4.5	2.3	1.3	0.3	central	1
107	4.6	3.6	1.0	0.2	central	1
108	5.0	3.2	1.2	0.2	central	1

Fig.18 Resultado obtenido del algoritmo DBSCAN en el SGBD PostgreSQL 9.3

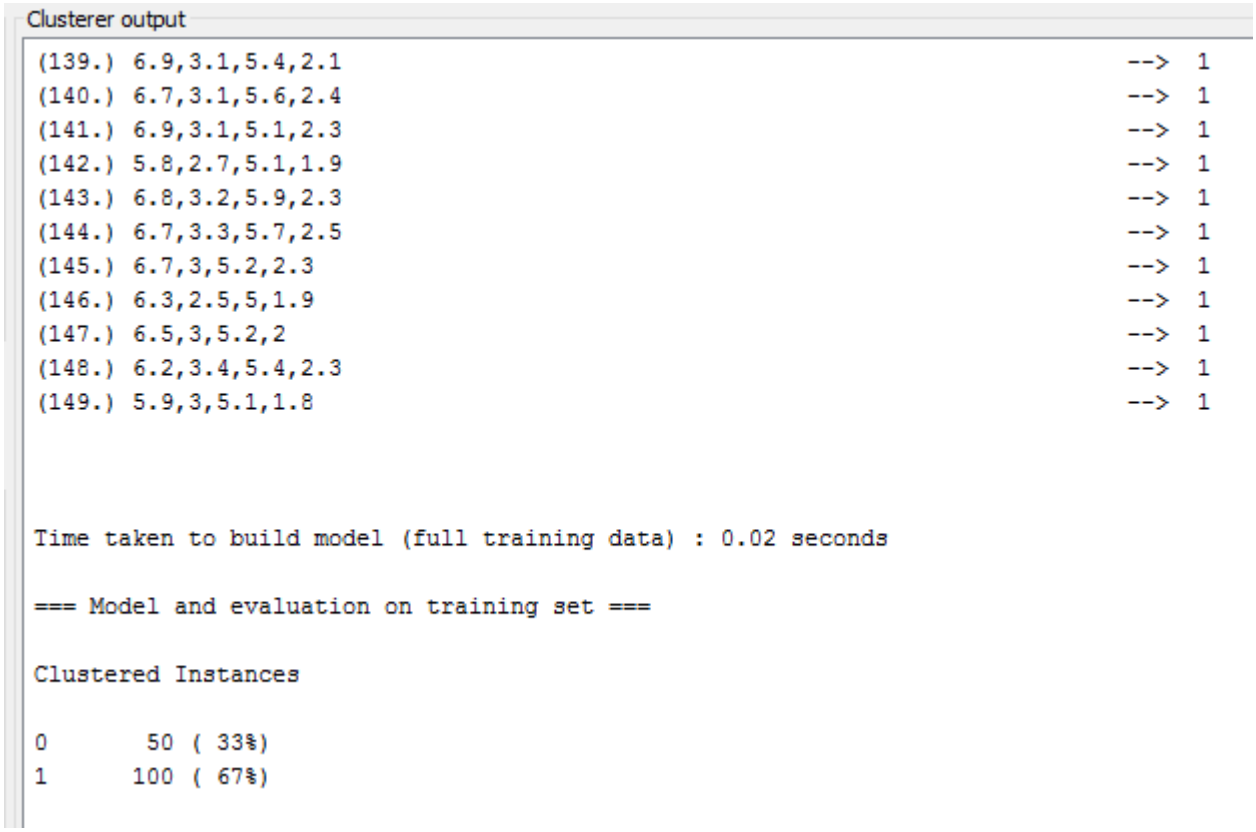


Fig.19 Resultado obtenido del algoritmo DBSCAN en el Weka

Como se puede observar los resultados obtenidos en ambas herramientas son bastantes aproximados, con la diferencia de 3 puntos. Esto se debe a que la herramienta Weka los clasifica todos aunque estén repetidos. El algoritmo implementado para PostgreSQL considera que por tratarse de puntos en un espacio multidimensional no es necesario clasificar un punto antes clasificado. Otra diferencia significativa en los resultados obtenidos es que el Weka no devuelve la clasificación de los puntos en caso de ser central o frontera al contrario del algoritmo implementado en PostgreSQL. Para devolver dicha clasificación fue necesario almacenar poco a poco los puntos ya analizados con su clasificación para una vez terminada la ejecución del algoritmo mostrar todos los resultados. Por otro lado se pudo comprobar que el Weka cuando analiza un juego de datos con muchas columnas, a la hora de mostrar el resultado no devuelve todas las columnas. A continuación se muestra en la siguiente tabla una comparación de los resultados obtenidos del algoritmo DBSCAN implementado en el SGBD PostgreSQL 9.3 y en la herramienta Weka.

Tabla.12 Comparación en PostgreSQL y Weka para el algoritmo DBSCAN

Indicadores	DBSCAN en PostgreSQL	DBSCAN en Weka
Facilidad de uso	➢ Requiere de conocimientos de bases de datos para el análisis de los datos.	➢ Depende de especialistas para la manipulación de la herramienta además de conocimientos de bases de datos para el análisis de los datos.
Calidad de los datos	➢ Clasifica los puntos sin repetirse por tratarse de puntos en un espacio multidimensional. ➢ Devuelve la clasificación de los puntos en central, frontera o ruido. ➢ Devuelve todas las columnas para todas las tablas.	➢ Clasifica todos los puntos aunque estén repetidos. ➢ No devuelve la clasificación de los puntos en caso de ser central o frontera. ➢ Devuelve todas las columnas para tablas con pocas columnas, no siendo así para tablas con muchas columnas.
Manipulación de los datos	➢ El SGBD no necesita driver de conexión. ➢ Los datos están listos para ser analizados.	➢ Necesita un driver para la conexión al SGBD. ➢ Convertir los datos al formato arff.
Actualización de los datos	➢ Ejecutar el algoritmo.	➢ Necesita conectarse nuevamente a SGBD, volver a cargar los datos porque no actualiza o generar otra vez el archivo arff para después correr el algoritmo.
Comprensión de los resultados	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.

Como se puede observar en la Tabla.12 el algoritmo DBSCAN implementado en el SGBD PostgreSQL 9.3 es más eficiente en lo que se refiere a los parámetros evaluados, por lo tanto el uso de este algoritmo para el SGBD proporciona mejores ventajas para el análisis de los datos.

Presentación de los resultados obtenidos con el algoritmo *FarthestFirst*

Consultes anteriors | select * from farthest_first('iris',2)as (grupo integer, cant_elementos integer, sepallength numeric, sepalwidth numeric, petallength numeric, petalwidth numeric);

```
select * from farthest_first('mfeat',3)as (grupo integer, cant_elementos integer, x1 numeric, x2 numeric, x3 numeric, x4 numeric, x5 nu
```

Subfinestra de sortida

Sortida de dades | Explain | Missatges | Historial

	grupo integer	cant_elementos integer	x1 numeric	x2 numeric	x3 numeric	x4 numeric	x5 numeric	x6 numeric	x7 numeric	x8 numeric	x9 numeric	x10 numeric	x11 numeric	x12 numeric	x13 numeric	x14 numeric	x15 numeric
1	1	988	0789708	0977862	8814819	6575213	6324775	6693826	3441319	4318433	4789570	9098078	1191542	5036510	9132002	8021290	9164338
2	2	599	5743762	0179676	0187702	8373414	6016089	0459125	7973476	4456054	3153460	0276810	9076306	9608331	1237516	6513815	2224581
3	3	413	3055488	9447936	2521719	9852279	2191089	4888564	6964591	7865175	7940718	5129826	6245323	7642687	1849766	4804020	1704264

Fig.20 Resultado obtenido del algoritmo *FarthestFirst* en el SGBD PostgreSQL 9.3

Clusterer

Choose | **FarthestFirst** -N 3 -S 5

Cluster mode

- (•) Use training set
- ( ) Supplied test set — Set...
- ( ) Percentage split — % 66
- ( ) Classes to clusters evaluation

(Nom) class

[✓] Store clusters for visualization

Ignore attributes

Start | Stop

Result list (right-click for options)

10:08:28 - FarthestFirst

Clusterer output

```
Cluster centroids:

Cluster 0
         0.34192221 0.43023383 0.46542399 0.28068626 0.4492498
Cluster 1
         0.01023823 0.11319968 0.01252211 0.28236089 0.6484381
Cluster 2
         0.11131312 0.57965844 0.16862359 0.34094011 0.2444561



Time taken to build model (full training data) : 0.06 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       982 ( 49%)
1       430 ( 22%)
2       588 ( 29%)
```

Fig.21 Resultado obtenido del algoritmo *FarthestFirst* en el Weka

Como se puede observar los grupos obtenidos en ambas herramientas tienen pequeñas variaciones. La diferencia está basada en una característica fundamental que posee algoritmo *FarthestFirst* que es el criterio de selección inicial de los centroides. Para esto Weka utiliza una semilla al igual que en el algoritmo *K-means*. Por su parte PostgreSQL usa una función *random()* que permite escoger aleatoriamente el centroide del conjunto de datos. Los grupos son conformados de acuerdo al centroide seleccionado y no necesariamente deben ser los mismos centroides escogidos por ambas herramientas. A continuación se muestra en la siguiente tabla una comparación de los resultados obtenidos del algoritmo *FarthestFirst* implementado en el SGBD PostgreSQL 9.3 y en la herramienta Weka.

Tabla.13 Comparación en PostgreSQL y Weka para el algoritmo *FarthestFirst*

Indicadores	*FarthestFirst* en PostgreSQL	*FarthestFirst* en Weka
Facilidad de uso	➢ Requiere de conocimientos de bases de datos para el análisis de los datos.	➢ Depende de especialistas para la manipulación de la herramienta además de conocimientos de bases de datos para el análisis de los datos.
Calidad de los datos	➢ Alta	➢ Alta
Manipulación de los datos	➢ El SGBD no necesita driver de conexión. ➢ Los datos están listos para ser analizados.	➢ Necesita un driver para la conexión al SGBD. ➢ Convertir los datos al formato arff.
Actualización de los datos	➢ Ejecutar el algoritmo.	➢ Necesita conectarse nuevamente a SGBD, volver a cargar los datos porque no actualiza o generar otra vez el archivo arff para después correr el algoritmo.
Comprensión de los resultados	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.	➢ Requiere de un especialista para realizar un análisis riguroso y con profundidad de los resultados obtenidos.

Como se puede observar en la Tabla.13 el algoritmo *FarthestFirst* implementado en el SGBD PostgreSQL 9.3 es más eficiente en lo que se refiere a los parámetros evaluados, por lo tanto el uso de este algoritmo para el SGBD proporciona mejores ventajas para el análisis de los datos.

3.7 Despliegue

El despliegue del proyecto no significa que el propósito haya terminado. Los resultados que presenta deben ser claros de tal manera que se puedan comprender fácilmente por personas que no tengan mucha intervención en el área de la informática, sino también de profesionales o no que pertenezcan a otras áreas.

La fase de despliegue puede ser tan compleja como los requisitos que se hayan propuesto, así que pueden ir desde la simple presentación de un informe hasta la demostración del sistema ya implementado. En este caso se presentará un informe con el resultado del trabajo realizado. El presente documento se considera el informe final.

3.8 Conclusiones del capítulo

En el presente capítulo con el objetivo de validar los algoritmos integrados al SGBD PostgreSQL 9.3 se realizaron pruebas funcionales a los algoritmos implementados para detectar y corregir los posibles errores antes de ser entregados al cliente final. Por otro lado se realizó la aplicación del proceso de DM sobre los juegos de datos *iris.arff* y *mfeat-fourier.arff*, donde se analizaron los campos importantes para darle cumplimiento a los objetivos de negocio y DM propuestos. Además se realizó una comparación de los resultados obtenidos al ejecutar los algoritmos en la herramienta Weka y en SGBD PostgreSQL 9.3 y se demostró que los algoritmos implementados en el gestor proporcionan mejores ventajas para el análisis de los datos.

CONCLUSIONES GENERALES

Con los resultados obtenidos en la investigación se concluye que:

- La técnica de agrupamiento de la DM permite obtener como resultado final el agrupamiento de los datos tanto conociendo el número de grupos a conformar a priori como la formación de los mismos sin saber la cantidad a formar. En ambos casos los resultados obtenidos son fáciles de entender por los usuarios finales.

- Las herramientas libres existentes de Minería de Datos tienen el inconveniente de ser independientes del SGBD por lo cual se implementaron los algoritmos *K-means*, DBSCAN y *FarthestFirst* de la técnica de agrupamiento y se integraron al SGBD PostgreSQL 9.3 a través de la creación de una extensión, lo que contribuye a la soberanía tecnológica del país y a que el gestor sea más competitivo.

- Las pruebas funcionales realizadas a los algoritmos implementados permitieron la validación de los mismos al retornar con ellas los resultados esperados. Para ello se aplicó la solución en los juegos de datos *iris.arff* y *mfeat-fourier.arf*f donde se realizó una comparación entre los resultados obtenidos en la herramienta Weka y el SGBD PostgreSQL 9.3. En dicha comparación se observó que la integración de los algoritmos a PostgreSQL permite aprovechar sus propiedades para el análisis de los datos.

RECOMENDACIONES

Con el objetivo de seguir potenciando el SGBD PostgreSQL se recomienda integrarle el algoritmo *HierarchicalClusterer* de Minería de Datos, el cual pertenece a los algoritmos jerárquicos dentro de la técnica de agrupamiento.

REFERENCIAS BIBLIOGRÁFICAS

1-Hanson, J. Jeffrey. 2011. *Agregue XML como una herramienta de minería de datos.* [En línea] 17 de julio de 2011. [Citado el: 7 de diciembre de 2013.] http://www.ibm.com/developerworks/ssa/xml/library/x-xmldatamine/.

2-José Manuel Molina López, Jesús García Herrero. *Técnicas de análisis de datos: Aplicaciones prácticas utilizando Microsoft Excel y Weka. 2011.*

3-Sánchez, Audrey Cordero. *Extensión de algoritmos de reglas de asociación para la Minería de Datos en PostgreSQL.* La Habana : s.n., 2013.

4-Académico, José María Caridad y Ocerin. 2011.*La Minería de Datos: Análisis de Bases de Datos en la Empresa.* 2011.

5-Gómez, María Isabel Ángeles Larrieta y Angélica María Santillán. 2010.*Minería de datos: Concepto, características, estructura y aplicaciones.* 2010.

6-Vallejos, Sofia J. *Minería de Datos.* Corrientes-Argentina : s.n., 2010

7-Yuniet Amador Rodríguez Suárez, Anolandy Díaz. *Herramientas de Minería de Datos.* 2009.

8-Andrea Villagra, Ana Guzmán, Daniel Pandolfi. *Análisis de medidas no-supervisadas de calidad en clústers obtenidos por K-means y Particle Swarm Optimization* . Patagonia Austral : s.n., 2011.

9-Campos, Margarita Gallardo. *Aplicación de técnicas de clustering para la mejora del aprendizaje.* Leganés : s.n., 2010.

10-Rui Xu, Donald C. Wunsch II. *Clustering, IEEE Press Series on Computational Intelligence.*

11- I. H. Witten, E. Frank & M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques.* s.l. : Second Edition.

12-García, Yenei Martínez.*Aplicación de la técnica de Minería de Datos agrupamiento sobre el área de Gestión Académica de la Universidad de las Ciencias Informáticas. .* La Habana : s.n., 2012.

13-Viviana R. Toledo Rivero, Yessica Quiñones Alvarez, Lissette Montero Herrera. *El uso de WEKA en la determinación de los estilos de aprendizaje de los estudiantes de ingeniería informática en Cienfuegos.* Cienfuegos, Cuba : s.n., 2013.

14- Narendra Sharma, Aman Bajpai , Mr. Ratnesh Litoriya. *Comparison the various clustering algorithms of weka tools.* Jaypee University of Engg. & Technology : s.n., 2010.

15-Guillermo Matos, Ricardo Chalmeta, Oscar Coltell. 2013. *Metodología para la Extracción del Conocimiento Empresarial a partir de los Datos* . [En línea] 8 de noviembre de 2013. [Citado el: 7 de diciembre de 2013.] http://www.scielo.cl/scielo.php?pid=S0718-07642006000200011&script=sci_arttext

16-Claudio Concepción Certad. 2010. *5 Programas libres para Data Mining.* [En línea] 25 de noviembre de 2010. [Citado el: 7 de diciembre de 2013.] http://fraterneo.blogspot.com/2010/11/5-programas-libres-para-data-mining.html.

17-Lumpkin, George. *Oracle Database para Data Warehousing e Inteligencia de Negocios. s.l. Informe Ejecutivo de Oracle.* 2010.

18-SolidQ, Dejan Sarka. *Data Mining in SQL Server.* 2012.

19-García, Yenei Martínez.*Aplicación de la técnica de Minería de Datos agrupamiento sobre el área de Gestión Académica de la Universidad de las Ciencias Informáticas. .* La Habana : s.n., 2012.

20-Kirenia Helen León Rodríguez, Frank Davila Hernández. *Técnicas de Minería de Datos aplicadas al estudio de la Hipertensión Arterial .* La Habana : s.n., 2011.

21-José H. Canós, Patricio Letelier y Mª Carmen Penadés. *Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia : s.n., 2011.

22-Evelyn Rodríguez Rojas, Yaciel López Pérez. *Plugin CRUD-PG para la herramienta de administración de bases de datos HABD.* Ciudad de la Habana : s.n., 2012.

23-PostgreSQL. PostgreSQL.org. [En línea] PostgreSQL 9.3 Press Kit, 12 de enero de 2013. [Citado el: 9 de septiembre de 2013.] http://www.postgresql.org/about/press/presskit93/es/.

24-Guía documentada para Ubuntu: PgAdminIII. Ubuntu, 2011.

25-Fonseca, Roberto Andrade.*PgSQL PL/pgSQL - SQL Procedural Language de la documentación de PostgreSQL.* 2010.

26-Pressman, Roger S. 2005. Ingeniería el software. Un enfoque práctico. 2005.

BIBLIOGRAFÍA

Hanson, J. Jeffrey. 2011. *Agregue XML como una herramienta de minería de datos.* [En línea] 17 de julio de 2011. [Citado el: 7 de diciembre de 2013.] http://www.ibm.com/developerworks/ssa/xml/library/x-xmldatamine/.

José Manuel Molina López, Jesús García Herrero. *Técnicas de análisis de datos: Aplicaciones prácticas utilizando Microsoft Excel y Weka. 2011.*

Sánchez, Audrey Cordero. *Extensión de algoritmos de reglas de asociación para la Minería de Datos en PostgreSQL.* La Habana : s.n., 2013.

Académico, José María Caridad y Ocerin. 2011.*La Minería de Datos: Análisis de Bases de Datos en la Empresa.* 2011.

Gómez, María Isabel Ángeles Larrieta y Angélica María Santillán. 2010.*Minería de datos: Concepto, características, estructura y aplicaciones.* 2010.

Vallejos, Sofia J. *Minería de Datos.* Corrientes-Argentina : s.n., 2010

Yuniet Amador Rodríguez Suárez, Anolandy Díaz. *Herramientas de Minería de Datos.* 2009.

Andrea Villagra, Ana Guzmán, Daniel Pandolfi. *Análisis de medidas no-supervisadas de calidad en clústers obtenidos por K-means y Particle Swarm Optimization* . Patagonia Austral : s.n., 2011.

Campos, Margarita Gallardo. *Aplicación de técnicas de clustering para la mejora del aprendizaje.* Leganés : s.n., 2010.

Rui Xu, Donald C. Wunsch II. *Clustering, IEEE Press Series on Computational Intelligence.*

I. H. Witten, E. Frank & M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques.* s.l. : Second Edition.

García, Yenei Martínez.*Aplicación de la técnica de Minería de Datos agrupamiento sobre el área de Gestión Académica de la Universidad de las Ciencias Informáticas. .* La Habana : s.n., 2012.

Viviana R. Toledo Rivero, Yessica Quiñones Alvarez, Lissette Montero Herrera. *El uso de WEKA en la determinación de los estilos de aprendizaje de los estudiantes de ingeniería informática en Cienfuegos.* Cienfuegos, Cuba : s.n., 2013.

Narendra Sharma, Aman Bajpai , Mr. Ratnesh Litoriya. *Comparison the various clustering algorithms of weka tools.* Jaypee University of Engg. & Technology : s.n., 2010.

Bibliografía

Guillermo Matos, Ricardo Chalmeta, Oscar Coltell. 2013. *Metodología para la Extracción del Conocimiento Empresarial a partir de los Datos* . [En línea] 8 de noviembre de 2013. [Citado el: 7 de diciembre de 2013.] http://www.scielo.cl/scielo.php?pid=S0718-07642006000200011&script=sci_arttext

Claudio Concepción Certad. 2010. *5 Programas libres para Data Mining.* [En línea] 25 de noviembre de 2010. [Citado el: 7 de diciembre de 2013.] http://fraterneo.blogspot.com/2010/11/5-programas-libres-para-data-mining.html.

Lumpkin, George. *Oracle Database para Data Warehousing e Inteligencia de Negocios.* s.l. Informe Ejecutivo de Oracle. 2010.

SolidQ, Dejan Sarka. *Data Mining in SQL Server*. 2012.

Aranda, Yadira Robles. *Propuesta de integración de las técnicas de minería de datos de Árboles de decisión y Reglas de inducción al Sistema Gestor de Base de Datos.* UCIENCIA. Ciudad de la Habana, 2012.

García, Yenei Martínez.*Aplicación de la técnica de Minería de Datos agrupamiento sobre el área de Gestión Académica de la Universidad de las Ciencias Informáticas. .* La Habana : s.n., 2012.

Kirenia Helen León Rodríguez, Frank Davila Hernández. *Técnicas de Minería de Datos aplicadas al estudio de la Hipertensión Arterial .* La Habana : s.n., 2011.

José H. Canós, Patricio Letelier y Mª Carmen Penadés. *Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia : s.n., 2011.

Evelyn Rodríguez Rojas, Yaciel López Pérez. *Plugin CRUD-PG para la herramienta de administración de bases de datos HABD.* Ciudad de la Habana : s.n., 2012.

PostgreSQL. PostgreSQL.org. [En línea] PostgreSQL 9.3 Press Kit, 12 de enero de 2013. [Citado el: 9 de septiembre de 2013.] http://www.postgresql.org/about/press/presskit93/es/.

Guía documentada para Ubuntu: PgAdminIII. Ubuntu, 2011.

Fonseca, Roberto Andrade.*PgSQL PL/pgSQL - SQL Procedural Language de la documentación de PostgreSQL.* 2010.

Pressman, Roger S. 2005. Ingeniería el software. Un enfoque práctico. 2005.

Domínguez Rodríguez, Karel Manuel y Téllez Sánchez, Lino. *Sistema de apoyo a la toma de decisiones en el proceso de negociación comercial*. Holguín, Cuba: s.n, 2011.

S. Richard, J. L. Vargas. *Revista Iberoamericana de Inteligencia Artificial: Minería de Datos Conceptos y Tendencias.* 25. 925, 2010.

Bibliografía

Soto Jaramillo, C. M. *Incorporación de Técnicas Multivariantes en un Sistema Gestor de Bases de Datos Incorporadas*, 2009.

Aranda, Yadira Robles. *Algoritmos de Minería de Datos: Árboles de decisión y reglas de Inducción Integrados a PostgreSQL*, 2012.

Ussama M, Fayyad. *Advances in Knowledge Discovery and Data Mining,* 1996.

Arencibia, José Alberto Gallardo. *Metodología para el desarrollo de proyectos en Minería de Datos CRISP-DM, 2009.*

1