Universidad de las Ciencias Informáticas

Facultad 6

Título: “Desarrollo de árboles de decisión como extensión al gestor de bases de datos PostgreSQL”

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

Autor: Yudaisy Rivera Barrios

Tutores: MSc. Yadira Robles Aranda

Ing. Juan Manuel Ruiz Godoy

La Habana, junio de 2014

DECLARACIÓN DE AUTORÍA

Declaro ser autora de la presente tesis que tiene por título: **"**Desarrollo de árboles de decisión como extensión al gestor de bases de datos PostgreSQL" 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 _____.

Yudaisy Rivera Barrios

_____

Firma del Autor

MSc. Yadira Robles Aranda

_____

Firma del Tutor

Ing. Juan Manuel Ruiz Godoy

_____

Firma del Tutor

DATOS DE CONTACTO.

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

Muchas son las personas que a lo largo de estos cinco años han colaborado por hacer realidad mis sueños por lo que hoy les estoy muy agradecida:

A mi mamá, por ser mi amiga, compañera, la luz de mis ojos, mi razón de ser, mi todo.

A mi padre por ser el mejor padre y aunque no esté aquí le agradezco por encaminarme en mi educación.

A mis queridos hermanos por estar siempre allí en todo momento y en especial a mi hermana Yudaimy por sus consejos y apoyo en todo.

A mi querida sobrina Laurita, por llenar de felicidad mi vida con sus travesuras.

A toda mi familia en general por estar siempre apoyándome, en especial a mis tíos: Margarita, Maisdevis, Tita, Yazmín, Frank y a mi abuela Silvia que fueron parte de este gran logro de hoy.

A mis amigas del barrio 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, gracias por su apoyo, sus consejos y por ser mis hermanas.

A mis compañeros de aula que aunque algunos se han ido, su amistad ha perdurado a pesar de la distancia.

A mis compañeras de apartamento en especial a Tania que siempre me daba paz con sus palabras y consejos y a mi compañera de cuarto Ailín por aguantar mis resabios.

A todas las amistades de la escuela que con el tiempo fueron marcando mi vida por su sinceridad y por estar allí en las buenas y en las malas.

A mis tutores por apoyarme y guiarme durante el desarrollo de la tesis.

Al tribunal y al oponente por sus críticas constructivas.

Y a mi novio Yordan por estar presente en esta última etapa de la universidad, la cual era la más importante y difícil, y él supo estar ahí queriéndome, amándome, apoyándome y ayudándome en todo lo posible para poder alcanzar la meta.

DEDICATORIA

Dedico este logro a mis padres:

A mi querido padre por ser el motor impulsor de mi educación y superación, siempre me decía que para triunfar había que ser alguien en la vida y no uno más. A él por ser el mejor padre del mundo, por estar siempre ahí vigilando mis pasos, por su confianza y la fe que siempre tuvo en mí. Su sueño más deseado era que me graduara en la universidad y aunque no esté hoy aquí debe sentirse orgulloso de mí porque acabo de realizar su sueño.

Y a mi madre querida, la cual sin ella no lo hubiese logrado, por estar siempre presente, por darme fuerzas y ánimos para seguir adelante, por toda la confianza que depositó en mí y por creer siempre que si podía lograrlo.

RESUMEN

La alta disponibilidad de información almacenada en bases de datos producida por el desarrollo de las tecnologías de la informática y las comunicaciones, ha dificultado el proceso de análisis de los datos recopilados manualmente. Es por ello que surge la técnica de Minería de Datos con el objetivo de identificar patrones de comportamiento y extraer conocimientos ocultos en grandes volúmenes de información de forma automática. La siguiente investigación presenta el análisis de los algoritmos de árboles de decisión ID3, C4.5 y *Decision Stump* como mejor solución a integrar al Sistema Gestor de Bases de Datos (SGBD) PostgreSQL, debido a las deficiencias que presentan las herramientas libres existentes para realizar el análisis de la información y con el fin de aprovechar sus potencialidades e incrementar las funcionalidades del mismo. El desarrollo de los algoritmos estuvo guiado por la metodología de desarrollo de *software Extreme Programming* (XP), la cual posibilitó obtener los resultados esperados. También se realizaron las pruebas de caja negra a la solución, así como el proceso de Minería de Datos, aportando una valoración acerca de la calidad de los algoritmos implementados y verificando el cumplimiento de los objetivos propuestos. Con el desarrollo de la extensión integrada al Sistema Gestor de Bases de Datos mencionado se contribuye a un mejor análisis de los datos.

PALABRAS CLAVE: árboles de decisión, minería de datos, PostgreSQL, sistema gestor de bases de datos.

ABSTRACT

The high availability of information stored in databases produced by the development of information and communications technologies has difficulted the process of analyzing the manually collected data. That is why Data Mining techniques were created, with the objective of identifying patterns and automatically extract knowledge hidden in large volumes of information. The following research presents the analysis of the decision tree algorithms ID3, C4.5 and Decision Stump as the better solution to integrate with the Database Management System (DBMS) PostgreSQL, considering the deficiencies that open source tools presents at the time of analyzing data and with the objective of using its potential and increase its functionalities. The development of the algorithms was guided by the *software* development methodology Extreme Programming (XP), which allowed to obtain the expected results. Black box tests were carried out on the solution, as well as the Data Mining process, providing an assessment of the quality of the implemented algorithms, and verifying compliance with the objectives. With the development of the extension integrated to the mentioned Database Management System a contribution was made to the better analysis of the data.

Keywords: decision trees, data mining, PostgreSQL, database management system.

INDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Con el desarrollo incremental de las tecnologías en el mundo, ha surgido un aumento en la información almacenada en Bases de Datos, implicando que todas las instituciones y empresas se han hecho dependientes de los medios informáticos para su funcionamiento habitual. Gracias a este volumen de información muchas entidades han podido predecir su futuro comercial siguiendo patrones estadísticos de su pasado.

Para lograr esto se hace necesario analizar minuciosamente toda esta información, inicialmente se hacía a través de expertos, pero esto se convertía en una tarea tediosa y casi imposible de lograr cuando el volumen de datos era muy grande. Por este motivo surgen técnicas y herramientas automatizadas, capaces de hallar patrones estadísticos en un conjunto de información, garantizando que entre más grande sea la fuente de datos más exacta sería la predicción.

Dentro de estas técnicas se encuentra la Minería de Datos (MD), que actualmente es una de las más usadas para encontrar patrones, tendencias, comportamientos y conocimiento útil que están ocultos en los datos recopilados. La misma se aplica en diversas ramas tales como la educación, la medicina, las finanzas y los negocios de mercado, obteniendo resultados satisfactorios. Dentro de la MD existen disímiles técnicas entre las cuales se encuentran la de árboles de decisión la cual es una de las más utilizadas, según diversos estudios realizados. (1)

Esta técnica se caracteriza por la sencillez de su representación y de su forma de actuar, además de la fácil interpretación dado a que puede ser expresada en forma de reglas de decisión. Una de las grandes ventajas de los árboles de decisión es que, en su forma más general, las opciones posibles a partir de una determinada condición son excluyentes. Esto permite analizar una situación y siguiendo el árbol de decisión apropiadamente, llegar a una sola acción o decisión a tomar. (1)

Variadas son las investigaciones que se han realizado sobre la MD debido a la importancia que tiene su uso, y es por ello que en Cuba siguiendo una política de avance científico enfocada a alcanzar la soberanía tecnológica se están desarrollando soluciones libres de costo de licencia que apoyan la acertada toma de decisiones haciendo uso de la técnica antes mencionada. Un ejemplo vigente es la Universidad de las Ciencias Informáticas (UCI), la cual ha sido reconocida como centro de avanzada de

desarrollo y exportación de *software* en el país (2). En esta institución se encuentra el Centro de Tecnologías de Gestión de Datos (DATEC), el cual posee proyectos que actualmente aplican la MD.

Existen numerosas herramientas para realizar el proceso de MD como YALE/Rapid Miner y WEKA, las cuales necesitan conectarse al Sistema Gestor de Bases de Datos (SGBD) y si en estos existe un gran volumen de datos para analizar, el proceso se vuelve engorroso. También están los gestores Oracle y SQL Server que cuentan con un módulo de técnicas de MD. Estos módulos permiten ganar en rapidez en los tiempos de respuesta, ya que no sería necesario transformar los datos. Aunque estos gestores cuentan con la implementación de algoritmos de MD, son herramientas cuyas licencias de uso y el soporte las hacen altamente costosas. Es por ello que en el departamento PostgreSQL, se está potenciando el uso del SGBD PostgreSQL con la inclusión de extensiones que permitan realizar MD. Ya se han desarrollado algunas de estas extensiones que incluyen algoritmos como reglas de asociación, reglas de inducción y el ID3 el cual es un algoritmo de árboles de decisión, pero este presenta como deficiencias que del conjunto de reglas que se obtienen como resultado derivadas del árbol de decisión solo el 83% coincidían con el resultado mostrado por la herramienta Weka. Además de solo permitir trabajar con tablas que las clases tuvieran los siguientes valores “si, +, p, Sí, no, -, n, No” de lo contrario era necesario hacerle modificaciones al código para poder realizar el análisis. La validación de estos algoritmos implementados fue aplicada al almacén de datos de ensayos clínicos Racotumumab del Centro de Inmunología Molecular y a la base de datos del Sistema de Genética Médica respectivamente, obteniendo resultados satisfactorios y así comprobar que con la integración de los algoritmos al gestor se permiten aprovechar las potencialidades del mismo para el análisis de los datos.

A pesar de la existencia de algoritmos de MD integrados a PostgreSQL no son suficientes si se tiene en cuenta que existen muchos de ellos con objetivos específicos en dependencia de los datos que se analizan. Al contar con un único algoritmo de MD de la técnica de árboles de decisión y teniendo en cuenta que este presenta algunas deficiencias siendo esta una de las técnicas más utilizadas (ver anexo1), se desaprovechan las ventajas del gestor PostgreSQL dado sus potencialidades.

De acuerdo a la problemática planteada se identifica como **problema científico**:
¿Cómo lograr el análisis de los datos en el gestor de Bases de Datos PostgreSQL aplicando MD?

Se plantea como **objeto de estudio** la MD en PostgreSQL y como **campo de acción** las técnicas de árboles de decisión de la MD en PostgreSQL.

Para dar solución al problema científico se traza como **objetivo general**: Desarrollar una extensión de árboles de decisión para el gestor de base de datos PostgreSQL que permita analizar el comportamiento de los datos almacenados.

Objetivos específicos:

- Analizar la técnica de MD: árboles de decisión para la selección de los algoritmos a implementar.
- Implementar los algoritmos de la técnica de MD: árboles de decisión para agregarlos como una extensión al SGBD PostgreSQL.
- Integrar los algoritmos implementados al SGBD PostgreSQL para el análisis del comportamiento de los datos.
- Validar la solución propuesta para la verificación del correcto funcionamiento de los algoritmos implementados.

Guiados por lo expuesto anteriormente y para apoyar la investigación se plantean las siguientes **preguntas científicas**:

1. ¿Cuál es la concepción actual sobre las herramientas utilizadas para realizar minería de datos?
2. ¿Qué herramientas se deben utilizar para el desarrollo de la extensión?
3. ¿Cómo se deben desarrollar las funcionalidades identificadas?
4. ¿Cómo integrar los algoritmos de minería de datos con el SGBD PostgreSQL?
5. ¿Cómo validar los algoritmos integrados al SGBD PostgreSQL?

Para cumplir con los objetivos planteados se trazaron las siguientes **tareas de la investigación**:

1. Selección de los algoritmos de árboles de decisión a implementar.
2. Selección de las tecnologías a utilizar para la implementación de los algoritmos.
3. Análisis de los lenguajes de programación y herramientas para la implementación de los algoritmos.
4. Implementación de los algoritmos para la integración al SGBD PostgreSQL.

5. Creación de la extensión de MD para la integración al SGBD PostgreSQL.
6. Incorporación de la extensión creada al SGBD PostgreSQL para el análisis del comportamiento de los datos.
7. Diseño de los casos de pruebas para validar los algoritmos implementados.
8. Validación de los algoritmos implementados para la verificación del correcto funcionamiento de los mismos.

Para la realización del presente trabajo de diploma se pusieron en práctica varios **métodos de investigación:** Teóricos y Empíricos. Los métodos teóricos son aquellos que permiten las relaciones esenciales del objeto de investigación, son fundamentales para la comprensión de los hechos y para la formulación de la hipótesis de investigación. Los métodos teóricos potencian la posibilidad de realización del salto cualitativo que permite ascender del acondicionamiento de información empírica a describir, explicar, determinar las causas y formular la hipótesis investigativa. (3)

Dentro de los métodos teóricos utilizados se encuentra el **Analítico-Sintético** ya que fue necesario el análisis de las teorías y los documentos referenciados con el objetivo de extraer los elementos más importantes que se relacionan con la MD en PostgreSQL, y de plasmar la información encontrada en las diferentes bibliografías con el fin de lograr una alta comprensión del contenido.

También se hizo tangible el uso del método **Análisis Histórico- Lógico** para evaluar la trayectoria real de un conjunto de algoritmos de árboles de decisión existentes en el ámbito de la MD. Los cuales tienen la capacidad de encontrar patrones estadísticos en un cúmulo de información extremadamente grande. Asimismo, permitió constatar teóricamente cómo han evolucionado estos algoritmos en busca de los más idóneos para la solución a brindar.

Se emplean además otros métodos tales como la **Observación** y las **Entrevistas**, ambos pertenecientes a los métodos empíricos. El primero se evidencia en la observación de otras herramientas en funcionamiento que poseen algoritmos de MD, para clasificar y analizar la efectividad y eficiencia de estos en la búsqueda de criterios en la información contenida en las bases de datos. Las entrevistas se le realizaron a un total de 2 especialistas de la facultad 6 (MSc. César Raúl García Jacas y MSc. Asnay Guirola González) con amplios conocimientos en el tema de investigación, con el fin de conocer el objeto de investigación desde un punto de vista externo, sin

que se requiera aún la profundización en la esencia del fenómeno. Se efectúa una entrevista informativa y no estructurada focalizada al tema de los algoritmos de árboles de decisión como técnica de MD.

El presente documento se encuentra estructurado en tres capítulos.

Capítulo1: Fundamentación Teórica.

En el Capítulo 1 se realiza un estudio sobre las herramientas y metodologías usadas en el proceso de MD. Se define el lenguaje de programación y la herramienta a utilizar para implementar los algoritmos así como la metodología de desarrollo de *software* para la realización de la extensión. También se seleccionaron los algoritmos a implementar de árboles de decisión y se muestra una descripción de los mismos.

Capítulo 2: Descripción de la Solución.

En el Capítulo 2 se describen las principales características que tendrá la extensión. Se presenta una descripción de los algoritmos implementados incluyendo un fragmento del código así como una breve explicación de todas las funciones implementadas. Además se muestra como fueron integradas al SGBD PostgreSQL 9.3.

Capítulo 3: Aplicación y Validación de la Solución Propuesta.

En el Capítulo 3 se analizan las técnicas que define la metodología XP para diseñar los casos de pruebas que guiarán la validación de la extensión. Se muestra el desarrollo de la validación de los algoritmos implementados mediante el uso de las pruebas de caja negra. Además se aplica el proceso de MD a la base de datos de Genética Médica.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

Introducción

En este capítulo se explica la relación de la Inteligencia Artificial con la MD. Además, se analizan los árboles de decisión dentro de las técnicas que realizan MD, con el fin de seleccionar los algoritmos a implementar. También, se tratan las herramientas de MD propietarias y las desarrolladas bajo licencia libre. Asimismo se caracterizan las metodologías utilizadas tanto en el proceso de desarrollo de *software* como en el proceso de MD, y las herramientas y el lenguaje de programación a utilizar para el desarrollo de la extensión.

1.1 Minería de datos

La Inteligencia Artificial (IA) es un campo que por sus investigaciones trata de ser independiente de la informática, y se define como la técnica de *software* que los programas utilizan para dar solución a algún tipo de problema, pero tratando de asemejar el comportamiento inteligente que se observa en la naturaleza; es decir, trata de resolver problemas y tomar decisiones similares a las que toman los seres humanos al afrontar la vida diaria, realizando programas de computadora que aumenten la capacidad o “inteligencia“ de las mismas; el objetivo de las investigaciones de la IA es, aumentar la utilidad de las máquinas y sus procesos. (4)

Dentro de la IA existen un conjunto de técnicas para la extracción de conocimiento implícito en las bases de datos, que se conoce como MD, esta es una de las fases más importantes dentro del proceso: descubrimiento de conocimiento en bases de datos (KDD, por sus siglas en inglés: *Knowledge Discovery from Databases*). Este proceso se ha desarrollado en los últimos años y consta de una secuencia iterativa de etapas o fases, las cuales son: preparación de los datos (selección y transformación), MD, evaluación, interpretación y toma de decisiones. (5)

La MD básicamente consiste en extraer información de un conjunto de datos y transformarla en una estructura comprensible para su uso posterior mediante algoritmos de predicción, clasificación y segmentación. Por lo que se pudiera afirmar que la MD tiene su base en la IA y en el análisis estadístico. (6)

En muchas situaciones, el método tradicional de convertir los datos en conocimiento consiste en un análisis e interpretación realizada de forma manual. El especialista en la materia, analiza los datos y elabora un informe o hipótesis que refleja las tendencias de los mismos. Esta forma de actuar es lenta, cara y altamente subjetiva, de hecho, el análisis manual es impracticable en dominios donde el volumen de los datos crece exponencialmente, debido a que la enorme abundancia de datos desborda la capacidad humana de comprenderlos sin la ayuda de herramientas potentes. Consecuentemente, muchas decisiones importantes se realizan, no sobre la base de la gran cantidad de datos disponibles, sino siguiendo la propia intuición del usuario al no disponer de las herramientas necesarias. Ésta es la principal función de la MD: resolver problemas analizando los datos presentes en las bases de datos.

Entre las múltiples definiciones que identifican a la MD se encuentran:

La Minería de Datos por las siglas en inglés *Data Mining* es el proceso de extraer conocimiento útil y comprensible, previamente desconocido, desde grandes cantidades de datos almacenados en distintos formatos. Las herramientas de la MD predicen futuras tendencias y comportamientos, permitiendo en los negocios la toma de decisiones. (7)

Una definición tradicional es la siguiente: 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. (8)

"...el proceso de descubrir conocimientos interesantes, como patrones, asociaciones, cambios, anomalías y estructuras significativas a partir de grandes cantidades de datos almacenadas en Bases de Datos, *Data-Warehouse,* o cualquier otro medio de almacenamiento de información". (9)

Algunos ejemplos de la aplicación de la MD son:

- Detección de hábitos de compra en supermercados.
- Detección de patrones de fuga.
- Detectar en las industrias aquellos clientes que puedan estar pensando en terminar sus contratos para pasarse a la competencia.

- Detección de transacciones de blanqueo de dinero, fraude en el uso de tarjetas de crédito o de servicios de telefonía móvil.
- Análisis del comportamiento de los visitantes en una página de Internet, o la utilización de la información sobre ellos para ofrecerles propaganda adaptada específicamente a su perfil.

De manera general, puede afirmarse que la MD es un proceso apoyado en técnicas y herramientas que descubren, a partir de datos almacenados, información útil que brinda algún beneficio a una organización.

1.1.1 Técnicas de MD

Las técnicas de MD intentan obtener patrones o modelos a partir de los datos recopilados, donde la valoración del usuario suele decidir si los modelos obtenidos son útiles o no. Dichas técnicas no son más que algoritmos sofisticados que se aplican sobre un conjunto de datos para obtener resultados, patrones o modelos a partir de los datos recopilados. Se clasifican en dos grandes categorías: supervisadas o predictivas y no supervisadas o descriptivas.

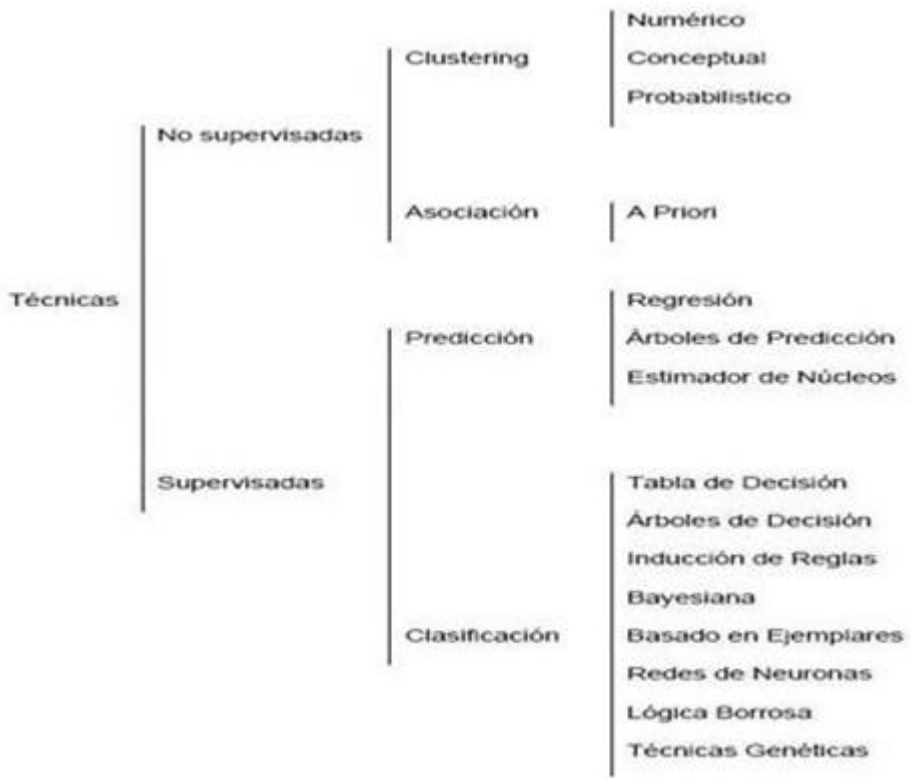

Fig. 1: Técnicas de MD. *(10)*

Entre los diversos tipos de técnicas que contiene la MD, se destacan las técnicas de clasificación que se encargan de dividir un conjunto de datos en grupos mutuamente excluyentes, de tal forma que cada miembro de un grupo esté lo más cerca posible de otros y que grupos diferentes estén lo más lejos posible de otros, donde la distancia se mide con respecto a las variables especificadas, que se quieren predecir. Dentro de este grupo se encuentra la técnica árboles de decisión.

1.2 Árboles de decisión

Los árboles de decisión son una de las técnicas de aprendizaje inductivo supervisado, se utilizan para la predicción y se emplea en el campo de inteligencia artificial, donde a partir de una base de datos se construyen diagramas de construcción lógica (11). Los modelos de árboles de decisión son comúnmente usados en la MD para examinar los datos e inducir las reglas para realizar predicciones. Constituyen una de las principales técnicas de MD. Son fáciles de usar, tolerantes a atributos no significativos y a valores faltantes.

Entre las facilidades de utilizar un árbol de decisiones se encuentra que permite plantear claramente el problema de tal manera que todas las opciones sean analizadas, es decir, hacer un análisis rápido de todas las consecuencias de las posibles decisiones. Ya que utiliza un esquema que cuantifica el costo de los resultados y las probabilidades de que los diferentes resultados aparezcan, eso ayuda a tomar decisiones adecuadamente. (12)

Su utilización cotidiana se puede dar en diagnósticos médicos, predicciones meteorológicas, controles de calidad, y otros problemas que necesiten de análisis de datos y toma de decisiones. Sin embargo, se tiene un uso amplio en la toma de decisiones de inversión, reinversión, políticas de créditos y financiamiento a corto y largo plazo. Dentro de la Gerencia y la Administración financiera serán de gran ayuda pues se logrará tener un mapa que pueda medir el riesgo y beneficios de las decisiones tomadas, claramente será de mayor precisión en cuanto se pueda contar con la mayor cantidad de información posible que nos permitan elegir las opciones que minimicen el riesgo y maximicen los beneficios. (12)

Permiten además reducir la cantidad de variables para realizar el análisis. Representan una gran ventaja respecto a las demás técnicas de clasificación, ya que permiten representar el conocimiento extraído en un conjunto de reglas. (13)

Dentro de la técnica de árboles de decisión se encuentran los algoritmos ID3, C4.5 y *Decision Stump*, los cuales son la propuesta de implementación, dado a las características que estos presentan. En el caso del primer algoritmo ID3 es capaz de tomar decisiones con gran precisión, trabaja con atributos cuyos valores sean discretos y es iterativo, lo que permite hallar el árbol correcto en unas pocas iteraciones. El C4.5 trabaja además con atributos que presentan valores continuos y valores desconocidos. *Decision Stump* es un algoritmo sencillo que genera un árbol de decisión de un único nivel, pero la implementación es muy completa, dado que admite tanto atributos numéricos como simbólicos y clases de ambos tipos también. Debido a las funciones antes mencionadas, estos algoritmos son los más utilizados en el proceso de realización de MD (14) y además tienen gran utilidad en diferentes ramas.

Algoritmo ID3 integrado al SGBD PostgreSQL

Dado a las diversas ventajas que el algoritmo ID3 presenta, fue integrado al SGBD PostgreSQL en el año 2012 por la MSc. Yadira Robles Aranda. Aunque su realización por si solo representaba una gran ventaja, se detectó que no era una solución genérica. El mismo trabajaba con valores fijos en el clasificador, limitando el proceso de análisis de los datos para cualquier tipo de información, pues en la mayoría de los casos los datos analizados eran distintos. Esta desventaja traía consigo que había que hacer modificaciones a las tablas o al código para poder analizar dichos datos, donde se necesitaba tener conocimientos de base de datos. Por otra parte el resultado obtenido no era totalmente confiable, pues no mostraba la misma cantidad de reglas obtenidas cuando se comparaba con otras herramientas como Weka[1], coincidiendo solamente el 83% como promedio en cada análisis, lo que no le permitía a los especialistas tomar correctamente decisiones importantes, y afectaba aún más cuando los datos contenían información sensible.

Por tales motivos es que se propone reimplementar dicho algoritmo para extender su aplicación a diversas ramas de la sociedad y ayudar a una correcta toma de decisiones relacionada con problemas afines.

A continuación se muestra un fragmento de la función implementada para dicho algoritmo, donde se puede evidenciar las deficiencias que este presenta.

[1] Weka: es una colección de algoritmos de aprendizaje automático para tareas de minería de datos. (17)

```
while finc > 0
 loop
    delete from aa;
    execute 'select count(*) from ejemplo where ' ||$2||' in (''Sí'', ''+'',
''Si'', ''p'')' into p;
    execute 'select count(*) from ejemplo where ' ||$2||' in (''No'',
''-'',''n'')' into n;
    pn:=p+n;
    if p=0 or n=0 then
     i:=0;
          else
     i:= -(p/pn)*(log(2,p)-log (2,pn))-(n/pn)* (log (2, n) - log
(2,pn));--informaci'on
     end if;
          EXECUTE 'select count(*) from ejemplo where ' ||atrib.column_name|| ' =
' || '''' ||valores|| ''' and ' ||$2||' in (''Sí'', ''+'',''Si'',''p'')'  into p1;
          EXECUTE 'select count(*) from ejemplo where ' ||atrib.column_name|| ' =
' || '''' ||valores|| ''' and ' ||$2||' in (''No'', ''-'',''n'')' into n1;

          pn1:=p1+n1;
          if p1=0 or n1=0 then
           i1:=0;
```

Fig. 2: Fragmento de código del algoritmo ID3 existente. *(15)*

1.2.1 ID3

El algoritmo ID3 que significa "inducción mediante árboles de decisión" es capaz de tomar decisiones con gran precisión. Es un sistema de aprendizaje supervisado que aplica la estrategia "divide y vencerás" para hacer la clasificación, implementando métodos y técnicas para la realización de procesos inteligentes, representando así el conocimiento y el aprendizaje, con el propósito de automatizar tareas. (11) (16)

El ID3 es un algoritmo simple y a la vez potente, cuya misión es la elaboración de un árbol de decisión. El procedimiento para generar un árbol de decisión consiste en seleccionar un atributo como raíz del árbol y crear una rama con cada uno de los posibles valores de dicho atributo. Con cada rama resultante (nuevo nodo del árbol), se realiza el mismo proceso, esto es, se selecciona otro atributo y se genera una nueva rama para cada posible valor del atributo. Este procedimiento continúa hasta que los ejemplos se clasifiquen a través de uno de los caminos del árbol. El nodo final de cada camino será un nodo hoja, al que se le asignará la clase correspondiente. Así, el objetivo de los árboles de decisión es obtener reglas o relaciones que permitan clasificar a partir de los atributos. (11) (16)

En cada nodo del árbol de decisión se debe seleccionar un atributo para seguir dividiendo, y el criterio que se toma para elegirlo es: se selecciona el atributo que mejor separe (ordene) los ejemplos de acuerdo a las clases. Para ello se emplea la entropía, que es una medida de cómo está ordenado el universo. La teoría de la información (basada en la entropía) calcula el número de bits (información, preguntas sobre atributos) que hace falta suministrar para conocer la clase a la que pertenece un

ejemplo. Cuanto menor sea el valor de la entropía, menor será la incertidumbre y más útil será el atributo para la clasificación. Se define como entropía: dado un conjunto de eventos $A = \{A_1, A_2, \dots, A_n\}$, con probabilidades $\{p_1, p_2, \dots, p_n\}$, la información en el conocimiento de un suceso A_i (bits) se define en la ecuación de la figura 3 (Fig. 3). En esta figura se muestra la propiedad del logaritmo aplicada en la ecuación de la información para conformar la ecuación de la información media. (11) (16)

$$I(A_i) = log_2\left(\frac{1}{p_i}\right) = -log_2(p_i)$$

Fig. 3: Ecuación de la información. *(16)*

La información media de A (bits) se muestra en la ecuación de la figura 4 (Fig. 4). Esta consiste en la sumatoria de la multiplicación de la probabilidad por el logaritmo de dicha probabilidad dado un atributo A_i.

$$I(A) = \sum_{i=1}^{n} p_i I(A_i) = -\sum_{i=1}^{n} p_i log_2(p_i)$$

Fig. 4: Ecuación de la información media. *(16)*

Si se aplica la entropía a los problemas de clasificación se puede medir lo que se discrimina (se gana por usar) un atributo A_i empleando para ello la ecuación de la figura 5 (Fig. 5), en la que se define la ganancia de información.

$$G(A_i) = I - I(A_i)$$

Fig. 5: Ecuación de la ganancia de información. (16)

Siendo I la información antes de utilizar el atributo, e $I(A_i)$ la información después de utilizarlo. Se definen ambas en las ecuaciones de la figura 5 (Fig. 5). Una vez explicada la heurística empleada para seleccionar el mejor atributo en un nodo del árbol de decisión, se muestra el pseudocódigo del algoritmo ID3:

1. Seleccionar el atributo A_i que maximice la ganancia $G(A_i)$.
2. Crear un nodo para ese atributo con tantos sucesores como valores tenga.
3. Introducir los ejemplos en los sucesores según el valor que tenga el atributo A_i.
4. Por cada sucesor:
    a. Si sólo hay ejemplos de una clase, C_k, entonces etiquetarlo con C_k.
    b. Si no, llamar a ID3 con una tabla formada por los ejemplos de ese nodo, eliminando la columna del atributo A_i.

Fig. 6: Pseudocódigo del algoritmo ID3. *(10)*

El ID3 es capaz de tratar con atributos cuyos valores sean discretos, es iterativo que elige al azar un subconjunto de datos a partir del conjunto de datos de "entrenamiento" y construye un árbol de decisión a partir de ello. De esta forma se puede hallar el árbol correcto en unas pocas iteraciones, procesando un conjunto de datos. (16)

1.2.2 C4.5

El algoritmo C4.5 fue desarrollado por JR Quinlan en 1993, como una extensión del algoritmo ID3 que desarrolló en 1986, que permite:

1. Empleo del concepto razón de ganancia (GR, [Gain Ratio])

2. Construir árboles de decisión cuando algunos de los ejemplos presentan valores desconocidos para algunos de los atributos.

3. Trabajar con atributos que presenten valores continuos.

4. Obtención de Reglas de Clasificación. (10) (16)

Razón de Ganancia

El test basado en el criterio de maximizar la ganancia tiene como ventaja la elección de atributos con muchos valores. Esto es debido a que cuanto más fina sea la participación producida por los valores del atributo, normalmente, la incertidumbre o entropía en cada nuevo nodo será menor, y por lo tanto también será menor la media de la entropía a ese nivel. C4.5 modifica el criterio de selección del atributo empleando en lugar de la *ganancia* la *razón de ganancia*, cuya definición se muestra en la ecuación de la figura 7 (Fig. 7).

$$GR(A_i) = \frac{G(A_i)}{I(\text{División } A_i)} = \frac{G(A_i)}{-\sum_{j=1}^{nv(A_i)} \frac{n_{ij}}{n} log_2\left(\frac{n_{ij}}{n}\right)}$$

Fig. 7: Ecuación de la razón de ganancia del algoritmo C4.5. *(16)*

Al término $I(División\ A_i)$ se le denomina información de ruptura. En esta medida cuando n_{ij} tiende a n, el denominador se hace *0*.

PSEUDOCODIGO DE C4.5

```
Función C4.5
R: conjunto de atributos no clasificadores,
C: atributo clasificador,
S: conjunto de entrenamiento, devuelve un árbol de decisión
Comienzo
    Si S está vacío,
        Devolver un único nodo con Valor Falla;      'para formar el nodo raiz
    Si todos los registros de S tienen el mismo valor para el atributo
clasificador,
        Devolver un único nodo con dicho valor;   'un unico nodo para todos
    Si R está vacío,
        Devolver un único nodo con el valor más frecuente del atributo
        Clasificador en los registros de S [Nota: habrá errores, es decir,
        Registros que no estarán bien clasificados en este caso];
    Si R no está vacío,
        D ←atributo con mayor Proporción de Ganancia (D,S) entre los
atributos de R;
        Sean {dj | j=1,2,...., m} los valores del atributo D;
        Sean {dj | j=1,2,...., m} los subconjuntos de S correspondientes a los
valores de dj respectivamente;
            Devolver un árbol con la raíz nombrada como D y con los arcos
nombrados d1, d2,....,dm, que van respectivamente a los árboles
            C4.5(R-{D}, C, Sl), C4.5(R-{D}, C, S2), C4.5(R-{D}, C, Sm);
Fin
```

Fig. 8: Pseudocódigo del algoritmo C4.5. *(16)*

Atributos Continuos

El tratamiento que realiza C4.5 de los atributos continuos está basado en la ganancia de información, al igual que ocurre con los atributos discretos. Si un atributo continuo A_i presenta los valores ordenados $v_1, v_2, \dots, v_n$, se comprueba cuál de los valores $z_i = \frac{v_i + v_{i+1}}{2}; 1 \leq j < n$, supone una ruptura del intervalo $[v_1, v_n]$ en dos subintervalos $[v_1, z_j]$ y $(z_j, v_n]$ con mayor ganancia de información. El atributo continuo, ahora con dos únicos valores posibles, entrará en competencia con el resto de los atributos disponibles para expandir el nodo.

Obtención de Reglas de Clasificación

Cualquier árbol de decisión se puede convertir en reglas de clasificación, entendiendo como tal una estructura del tipo Si <Condición> Entonces <Clase>. El algoritmo de generación de reglas consiste básicamente en, por cada rama del árbol de decisión, las preguntas y sus valores estarán en la parte izquierda de las reglas y la etiqueta del nodo hoja correspondiente en la parte derecha (clasificación). En la figura 9 (Fig. 9) se muestra el algoritmo completo de obtención de reglas. (10)

```
ObtenerReglas (árbol) {
   Convertir el árbol de decisión (árbol) a un conjunto de reglas, R
   error = error de clasificación con R
   Para cada regla Ri de R Hacer
      Para cada precondición pj de Ri Hacer
         nuevoError = error al eliminar pj de Ri
         Si nuevoError <= error Entonces
            Eliminar pj de Ri
            error = nuevoError
      Si Ri no tiene precondiciones Entonces
         Eliminar Ri
}
```

Fig. 9: Pseudocódigo del algoritmo de obtención de reglas de C4.5. *(16)*

1.2.3 Decision Stump

El algoritmo *Decision Stump* que significa “árbol de decisión de un único nivel”, utiliza un único atributo para construir el árbol de decisión. La elección del único atributo que formará parte del árbol se realizará basándose en la ganancia de información, y a pesar de su simplicidad, en algunos problemas puede llegar a conseguir resultados interesantes. El árbol de decisión tendrá tres ramas: una de ellas será para el caso de que el atributo

sea desconocido, y las otras dos serán basados en dos casos. Para el caso de los atributos simbólicos la segunda rama será cuando el valor del atributo sea igual a un valor concreto y la tercera distinta a dicho valor. En el caso de atributos numéricos la segunda rama será cuando el valor sea mayor y la tercera cuando el valor sea menor a un determinado valor.

En el caso de los atributos simbólicos se considera cada valor posible del mismo y se calcula la ganancia de información con el atributo igual al valor, distinto al valor y valores desconocidos del atributo, en este caso se busca el mejor punto de ruptura. Deben tenerse en cuenta cuatro posibles casos al calcular la ganancia de información: que sea un atributo simbólico y la clase sea simbólica o que la clase sea numérica, o que sea un atributo numérico y la clase sea simbólica o que la clase sea numérica. A continuación se comenta cada caso por separado.

Atributo Simbólico y Clase Simbólica

Se toma cada vez un valor v_x del atributo simbólico A_i como base y se consideran únicamente tres posibles amas en la construcción del árbol: que el atributo A_i sea igual a v_x, que el atributo A_i sea distinto a v_x o que el valor del atributo A_i sea desconocido. Con ello, se calcula la entropía del atributo tomando como base el valor escogido tal y como se muestra en la siguiente figura.

$$I(A_{iv_x}) = \frac{\sum_{j=1}^{3} n_{ij} log(n_{ij}) - I_{ij}}{n\, log(2)};\ I_{ij} = \sum_{k=1}^{nc} n_{ijk} log(n_{ijk})$$

Fig. 10: Ecuación de la entropía del atributo del algoritmo *Decision Stump*. *(16)*

En la ecuación mostrada en la anterior figura el valor de j en la sumatoria va desde 1 hasta 3 porque los valores del atributo se restringen a tres: igual a v_x, distinto a v_x o valor desconocido. En cuanto a los parámetros, n_{ij} es el número de ejemplos con valor j en el atributo i, n el número total de ejemplos y n_{ijk} el número de ejemplos con valor j en el atributo i y que pertenece a la clase j.

Atributo Numérico y Clase Simbólica

Se ordenan los ejemplos según el atributo A_i y se considera cada z_x, definido como el punto medio entre los valores v_x y v_{x+1}, del atributo como posible punto de corte. Se consideran entonces como posibles valores del

atributo el rango menor o igual a z_x, mayor a z_x y valor desconocido. Se calcula la entropía figura 10 (Fig. 10) del rango tomando como base esos tres posibles valores restringidos del atributo.

Atributo Simbólico y Clase Numérica

Se vuelve a tomar como base cada vez un valor del atributo, tal y como se hacía en el caso Atributo Simbólico y Clase Simbólica, pero en este caso se calcula la varianza de la clase para los valores del atributo mediante la ecuación se muestra en la figura 11 (Fig. 11).

$$Varianza(A_{iv_x}) = \sum_{j=1}^{3} \left(SS_j - \frac{S_j}{W_j} \right)$$

Fig. 11: Ecuación de la varianza del algoritmo *Decision Stump*. *(16)*

En la ecuación de la figura 11 (Fig. 11), S_j es la suma de los valores de la clase de los ejemplos con valor j en el atributo i, SS_j es la suma de los valores de la clase al cuadrado y W_j es la suma de los pesos de los ejemplos (número de ejemplos si no se incluyen pesos) con valor j en el atributo.

Atributo Numérico y Clase Numérica

Se considera cada valor del atributo como punto de corte tal y como se hacía en el caso Atributo Numérico y Clase Simbólica. Posteriormente, se calcula la varianza tal y como se muestra en la ecuación de la figura 11 (Fig. 11).

En cualquiera de los cuatro casos que se han comentado, lo que se busca es el valor mínimo de la ecuación calculada, ya sea la entropía o la varianza. De esta forma se obtiene el atributo que será raíz del árbol de decisión y sus tres ramas. Lo único que se hará por último es construir dicho árbol: cada rama finaliza en un nodo hoja con el valor de la clase, que será la media o la moda de los ejemplos que se clasifican por ese camino, según se trate de una clase numérica o simbólica. (10)

1.3 Herramientas de Minería de Datos

Las herramientas de la MD son utilizadas para encontrar patrones y comportamientos donde los volúmenes de datos son muy grandes y se dificulta hacer el análisis de forma manual. Estas herramientas extraen de las bases de datos conocimientos ocultos, que a un experto se le haría imposible encontrar debido a la magnitud de los datos. De estas predicciones se puede hacer tomas de decisiones y sacar conclusiones estadísticas del futuro de una institución o empresa, permiten resolver situaciones que tradicionalmente tomarían mucho tiempo solucionarlas. Actualmente existe una gran variedad de estas herramientas, y a continuación se harán algunas caracterizaciones de las más utilizadas.

1.3.1 Herramientas Libres

Yale / Rapid Miner:

Las iniciales Yale responden a Yet Another Learning Environment, fue desarrollada en el lenguaje de programación Java, el cual integra completamente los códigos de Weka y además permite realizar MD. Tiene dos tipos de licencia, la gratuita bajo la licencia GPL y la propietaria. YALE puede importar información a partir de sistemas de Base de Datos como PostgreSQL y Microsoft SQL Server. Se requiere tener instalado con anterioridad el Java Runtime Environmentde Sun.

Características de YALE / Rapid Miner:

- Es un sistema para el descubrimiento del conocimiento y MD.
- Es un *software* de tipo código abierto con licencia GNU GPL, basado en java.
- Trabaja bajo las plataformas Windows y Linux.
- Posee alrededor de 400 operadores que pueden ser combinados.
- La característica más importante es la capacidad de jerarquizar cadenas del operador y de construir complejos árboles de operadores.
- Su lenguaje de encriptación permite automáticamente una gran cantidad de experimentos.
- Permite gran cantidad de extensiones. (7)

Weka:

Weka acrónimo de Waika to Enviroment for Knowledge Analysis es una herramienta para el análisis de datos que permite aplicar, analizar y evaluar las técnicas más relevantes de análisis de datos. Es una herramienta

desarrollada en Java en la universidad de Waikato, Nueva Zelanda. Constituida por paquetes de código abierto con técnicas de pre-procesado, clasificación, agrupamiento, asociación y visualización. Cuenta con una interfaz gráfica que permite al usuario acceder y configurar las diferentes herramientas integradas.

La herramienta dispone de cuatro interfaces distintas:

1. **Interfaz en modo texto:** Permite la introducción de todo tipo de comandos, pero no es posible realizar representaciones gráficas, el interfaz en modo texto permite instanciar las distintas clases Java definidas en el programa Weka.

2. **Interfaz Explorer:** Es el interfaz gráfico básico, en él se pueden mostrar gráficamente tanto las características de los datos de partida como los resultados de los análisis. Permite introducir los comandos con ayuda del mouse, seleccionando los operadores adecuados en menús desplegables.

3. **Interfaz Experimenter:** Se trata de un interfaz gráfico más avanzado, en el que no solo se pueden realizar análisis sobre los datos, sino que además es posible comparar el funcionamiento de diferentes algoritmos (por ejemplo, diferentes clasificadores) o bien comparar distintos ficheros de datos.

4. **Interfaz Knowledge Flow:** Este último interfaz permite representar como una red de operadores en cascada los procesos a realizar sobre los datos (pre-procesado, selección de características, ajuste de un clasificador y evaluación de los porcentajes de acierto esperables). (17)

Extensiones de MD en PostgreSQL:

En el Centro de Tecnologías de Gestión de Datos (DATEC) que pertenece a la facultad 6 existen proyectos que se dedican a potenciar las funciones del gestor de bases de datos PostgreSQL. Los proyectos antes mencionados conforman el departamento PostgreSQL, el cual posee extensiones integradas al SGBD con un total de 5 algoritmos de MD de los cuales 2 son de la técnica de reglas de asociación, 2 de la técnica de reglas de inducción y 1 de árboles de decisión. La validación de estos algoritmos implementados fue aplicada al almacén de datos de ensayos clínicos Racotumumab del Centro de Inmunología Molecular y a la base de datos del Sistema de Genética Médica respectivamente, donde se obtuvieron resultados satisfactorios en las dos

primeras técnicas. Dicha validación permitió comprobar que los algoritmos integrados al gestor con resultados exitosos posibilitan aprovechar las potencialidades de PostgreSQL para el análisis de los datos. (1) (18)

1.3.2 Herramientas Propietarias

Oracle Data Mining:

Oracle Data Mining es un módulo del SGBD Oracle que proporciona algoritmos de extracción de datos muy potentes que permite descubrir conocimientos ocultos en los datos, estos algoritmos se ejecutan como funciones SQL nativas (19). Mediante el uso y comprensión de estos algoritmos se pueden construir modelos predictivos de patrones y comportamientos para el futuro.

Las técnicas en MD se dividen en aprendizaje supervisado y aprendizaje no supervisado, la primera examina minuciosamente los datos para buscar patrones y relaciones, en el caso de Oracle Data Mining incluye Bayes Naive, Árbol de Decisión, Modelos Lineales Generalizados y Máquinas de Vectores Soporte (19). En el caso de aprendizaje no supervisado se pretende encontrar asociaciones entre los datos, sin importar el objetivo de negocio definido, para este tipo incluyen el algoritmo de clúster k-Means mejorado y el algoritmo de clúster Partición Ortogonal, Reglas de Asociación y Factorización de Matrices No Negativas.

Esta herramienta incluye una interfaz gráfica con el objetivo de que el usuario pueda crear, evaluar y aplicar los distintos modelos de MD. Es una aplicación flexible y fácil de entender, mostrando los datos en forma gráfica.

SQL Server Data Mining:

SQL Server Data Mining es un módulo del SGBD SQL Server desarrollado por Microsoft, determinado específicamente a analizar y comprender el conocimiento oculto que existe entre un conjunto de datos. La misma tiene como característica, que después de realizado el proceso de MD, incorpora el resultado nuevamente en el modelo, obteniéndose un nuevo modelo de análisis, es capaz de realizar proceso de MD sobre documentos de Microsoft Excel, entiende cuándo, cómo y dónde aplicar sus algoritmos según el tipo de datos a analizar en el servidor SQL, realiza un procesamiento analítico en línea para la extracción de datos (OLAP), para acceder y proteger los objetos de MD utiliza el módulo *Management Studio* y para crear y gestionar sus proyecto el *Business Intelligence Development* (1).

Esta herramienta tiene gran integración con la plataforma de Base de Datos SQL Server, aprovechando su desempeño, seguridad y características de optimización. Este modelo puede ser ampliado con nuevos

algoritmos que respondan a necesidades particulares de sus clientes, y sus técnicas y algoritmos pueden ser ejecutados en tiempo real.

1.3.3 Fundamentación de la herramienta seleccionada

Después de realizada la investigación exhaustiva sobre las herramientas de MD, la autora concuerda con la investigación previamente realizada por la Ing. Audrey Cordero Sánchez la cual afirma que YALE/Rapid Miner y Weka liberadas ambas bajo licencia libre, presentan como desventaja el proceso de integración con el gestor, siendo de esta manera algo engorroso y extenso el análisis de los datos. En el caso de los gestores Oracle y SQL Server los cuales son unos de las más potentes en la MD y son dependientes del SGBD, presentan como desventaja ser herramientas propietarias, bajo licencias comerciales, y constituye un gran gasto económico su utilización.

La autora de la presente investigación seleccionó como herramienta a utilizar PostgreSQL, herramienta libre que en el trabajo con las extensiones de MD que se han realizado, una vez aplicadas, mostraron resultados satisfactorios. Además, permite aprovechar las potencialidades del gestor para el análisis de los datos. Por lo antes planteado se decide desarrollar una extensión para el SGBD PostgreSQL mediante la implementación de algoritmos de MD de árboles de decisión.

1.4 Metodología para aplicar la MD.

Para realizar un desarrollo tecnológico se hace necesario tener en cuenta ciertas técnicas y procedimientos que permitan llevar una adecuada documentación sobre este desarrollo. Hoy, las metodologías son las que recogen este conjunto de procedimientos, permitiendo de una forma organizada guiar al equipo de trabajo en la implementación. En la MD existen diversas metodologías encargadas de guiar el proceso de planificación y ejecución de los proyectos como son: CRISP-DM (*Cross Industry Standard Process for Data Mining*), SEMMA (*Sample, Explore, Modify, Model, Asses*), Metodología de las cinco A‟s (*Asses, Access, Analyze, Act, Automate*), Modelo de proceso de MD de *Two Crows*, CRITIKAL (*Client-Server Rule Induction Technology for Industrial Knowledge Acquisition from Large Database*) y Metodología SQL Server- 2005.

Entre las más conocidas y empleadas, según un estudio realizado por el *SAS Institute*, está: CRISP-DM.

1.4.1 CRISP-DM

CRISP-DM acrónimo de *Cross Industry Standard Process for Data Mining* (según sus siglas en Inglés) es una metodología que consiste en un conjunto de tareas descritas en cuatro niveles de abstracción: fase, tarea genérica, tarea especializada, e instancia de proceso; organiza el desarrollo de forma jerárquica en tareas que van desde el nivel más general hasta los casos más específicos en una serie de seis fases.

Niveles de abstracción:

- ✓ **Fase**: Se le denomina fase al asunto o paso dentro del proceso. CRISP-DM consta de 6 fases: comprensión del negocio, comprensión de los datos, preparación de los datos, modelación, evaluación y explotación.

- ✓ **Tarea genérica**: Cada fase está formada por tareas genéricas, o sea, la tarea genérica es la descripción de las actividades que se realizan dentro de cada fase. Por ejemplo, la tarea "Limpiar los datos" es una tarea genérica.

- ✓ **Tarea especializada**: La tarea especializada describe cómo se pueden llevar a cabo las tareas genéricas en situaciones específicas. Por ejemplo, la tarea "Limpiar los datos" tiene tareas especializadas, como limpiar valores numéricos, y limpiar valores categóricos.

- ✓ **Instancias de proceso**: Las instancias de proceso son las acciones y resultados de las actividades realizadas dentro de cada fase del proyecto. Las fases del proyecto de Minería de acuerdo a lo establecido por la metodología CRISP-DM interactúan entre ellas de forma iterativa durante el desarrollo del proyecto. La secuencia de las fases no siempre es ordenada, o en ocasiones si se determina al realizar la evaluación que los objetivos del negocio no se cumplieron se debe regresar y buscarlas causas del problema para redefinirlo. (20)

A continuación se describen cada una de las fases en que se divide CRISP-DM:

- ✓ **Fase # 1- Comprensión del negocio:**

La primera fase es probablemente la más importante, ya que es muy importante la capacidad de poder convertir el conocimiento adquiriendo del negocio, en un problema de *Data Mining* y en un plan preliminar cuya meta sea el alcanzar los objetivos del negocio. 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 MD.

- ✓ **Fase # 2- Comprensión de los datos:**

Esta fase comprende la recolección inicial de datos, en esta junto a las próximas dos fases, son las que demandan el mayor esfuerzo y tiempo en un proyecto de MD. Las principales tareas que componen esta fase son las siguientes:

- Recolección de datos iniciales.
- Descripción de los datos.
- Exploración de datos.
- Verificación de la calidad de los datos.

- ✓ **Fase # 3- Preparación de los datos:**

En esta fase una vez efectuada la recolección inicial de datos, se procede a su preparación para adaptarlos a las técnicas de MD. Las principales tareas que componen esta fase son las siguientes:

- Selección de datos.
- Limpieza de los datos.
- Estructuración de los datos.
- Integración de los datos.
- Formateo de los datos.

- ✓ **Fase # 4- Modelado:**

Las técnicas a utilizar en esta fase se eligen en función de los siguientes criterios:

- ✓ Ser apropiada al problema.
- ✓ Disponer de datos adecuados.
- ✓ Cumplir los requisitos del problema.
- ✓ Tiempo adecuado para obtener un modelo.

- ✓ Conocimiento de la técnica.

Las principales tareas que componen esta fase son las siguientes:

1. Selección de la técnica de modelado.
2. Generación del plan de prueba.
3. Construcción del Modelo.
4. Evaluación del modelo.

- ✓ **Fase # 5- Evaluación:**

En esta fase se evalúa el modelo, teniendo en cuenta el cumplimiento de los criterios de éxito del problema. Las principales tareas que componen esta fase son las siguientes:

1. Evaluación de los resultados.
2. Proceso de revisión.
3. Determinación de futuras fases.

- ✓ **Fase # 6- Implementación:**

Las tareas que se ejecutan en esta fase son las siguientes:

1. Plan de implementación.
2. Monitorización y Mantenimiento.
3. Informe Final.
4. Revisión del proyecto. (21)

Las ventajas encontradas en esta metodología son:

- ➢ El proyecto de MD es visto de forma global y estrechamente relacionado al negocio en cuestión.
- ➢ Fue diseñada de forma neutra a la herramienta que se utilice para el desarrollo del proyecto, brindando la facilidad de uso con cualquiera de ellas.
- ➢ Es una metodología de distribución libre.
- ➢ Muchas de las metodologías que se pueden encontrar en la actualidad se basan en este estándar.

➢ Es la que cuenta con mayor aceptación por parte de los desarrolladores de procesos de extracción de conocimientos a partir de datos. (21)

Se selecciona la metodología CRISP-DM a utilizar en la presente investigación, dadas las características y ventajas expuestas, además de que haciendo uso de ella se obtienen proyectos de MD de alta calidad. Es una metodología que brinda una facilidad de uso con cualquier herramienta que se utilice para el proceso de MD.

1.5 Metodología de desarrollo

Las metodologías de desarrollo tienen como objetivo facilitar la organización y planificación de todas las actividades a realizar en el desarrollo de *software*, definiendo los responsables y el tiempo que debe demorar su cumplimiento. Son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos *software.* (22)

Las metodologías de desarrollo se dividen en ágiles y tradicionales, las ágiles valoran al individuo y las iteraciones en el equipo más que a las actividades y herramientas. Disminuye la documentación y modelado del desarrollo. Es menos resistente a los cambios, no sigue una estricta planificación. Este tipo de metodología es ideal, cuando se desea una alta calidad en el producto.

XP acrónimo de *Extreme Programming* o Programación Extrema es una metodología ágil diseñada para entornos dinámicos, pensada para equipos pequeños, orientada fuertemente hacia la codificación, haciendo énfasis en la comunicación informal y verbal. Fomenta los valores de comunicación, simplicidad y la retroalimentación. (23)

El ciclo de vida de XP está compuesto por las siguientes fases:

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

- **Pruebas:** 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.
- Fase de cierre del proyecto. (24)

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

- Integración del equipo de programación con el cliente.
- Simplicidad.
- 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. (24)

XP posee gran adaptabilidad a los diferentes tipos de proyectos, y gracias a la falta de guías prescriptivas se puede experimentar total o parcialmente con los métodos y prácticas que propone. Su utilización garantizará que queden incluidas explícitamente las actividades y herramientas que tienen que estar presente para el correcto desarrollo del sistema. Dicha metodología está orientada a la productividad y demuestra tener un conjunto de prácticas y métodos eficientes que junto a la continua interacción con el cliente y la mejora incremental del producto en cada iteración influyen significativamente a la hora de conseguir un producto de calidad.

Por lo antes expuesto se selecciona la metodología de desarrollo de *software* XP a partir de que sus características se ajustan a las del proyecto, las cuales son: requisitos muy cambiantes, existe un alto riesgo técnico. Se cuenta con un equipo de desarrollo pequeño, se necesita una implementación en un

corto tiempo. 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.

1.6 Herramientas y lenguaje de Programación

Uno de los aspectos fundamentales en la elaboración de un *software* es seleccionar las tecnologías y herramientas que mayores beneficios aporten. Para llevar a cabo la implementación y documentación de este trabajo, se realizó un estudio de las mismas, teniéndose en cuenta las tendencias actuales, novedades y facilidades de cada una de ellas. A continuación se describen las herramientas y el lenguaje de programación seleccionado para el desarrollo de los algoritmos de árboles de decisión.

1.6.1 PostgreSQL

PostgreSQL es un SGBD objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es un SGBD de código abierto potente y en sus últimas versiones ha ido mejorando llegando a igualarse con otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (25)

PostgreSQL posee un conjunto de características maduras que lo hace superior a sistemas de bases de datos propietarios, ya que los supera en extensibilidad, potencia, robustez, facilidad de administración, seguridad y estabilidad.

Algunas de sus características son:

Replicación Sincrónica: permitiendo alta disponibilidad con consistencia sobre múltiples servidores.

Regionalización por columna: soportando correctamente el ordenamiento por lenguaje en las bases de datos, tablas o columnas.

Multiplataforma: disponible para Linux y UNIX.

Algunas ventajas que presenta PostgreSQL:

- Ideal para tecnologías Web.
- Fácil de Administrar.

- Soporta el almacenamiento de objetos binarios grandes (gráficos, videos, sonido).
- Su sintaxis SQL es estándar y fácil de aprender.
- Capacidades de replicación de datos.
- Numerosos tipos de datos y posibilidad de definir nuevos tipos.

Extensión en PostgreSQL

PostgreSQL permite agregar nuevas funcionalidades en el SGBD PostgreSQL utilizando el objeto de base de datos EXTENSION para crear una extensión, esta 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.

Una extensión de PostgreSQL incluye típicamente múltiples objetos de SQL, por ejemplo, un nuevo tipo de datos requerirá nuevas funciones, nuevos operadores, y probablemente nuevas clases de operador de índice. Es útil para recoger todos estos objetos en un solo paquete, para simplificar la gestión de bases de datos. PostgreSQL llama a dicho paquete una extensión. (26)

1.6.2 PgAdmin III

PgAdmin III es una aplicación gráfica para interactuar con el gestor de Bases de Datos PostgreSQL y sus derivados (*Enterprise DBPostgres Plus Advanced Server y Greenplum Database*), siendo esta una herramienta completa y popular con licencia *Open Source*. Está escrita en C++ usando la librería gráfica multiplataforma *wxWidgets*, lo que permite que se pueda usar en *Linux*, *FreeBSD, Solaris, Mac OS X* y *Windows*. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas.

La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. También incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación *Slony-I* y mucho más. 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. (27)

1.6.3 PL/pgSQL

PL/pgSQL (*Procedural Language/PostgreSQL Structured Query Language*) es un lenguaje imperativo provisto por el gestor de Bases de Datos PostgreSQL. Permite ejecutar comandos SQL mediante un lenguaje de sentencias imperativas y uso de funciones, dando mucho más control automático que las sentencias SQL básicas. (28)

Desde PL/pgSQL se pueden realizar cálculos complejos y crear nuevos tipos de datos de usuario. Como un verdadero lenguaje de programación, dispone de estructuras de control repetitivas y condicionales, además de la posibilidad de creación de funciones que pueden ser llamadas en sentencias SQL normales o ejecutadas en eventos de tipo disparador. (28)

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 Bases de Datos puede planificar optimizaciones en la ejecución de la búsqueda y actualización de datos. (28)

Las funciones escritas en PL/pgSQL aceptan argumentos y pueden devolver valores de tipo básico o de tipo complejo (por ejemplo, registros, vectores, conjuntos o incluso tablas), permitiéndose tipificación polimórfica para funciones abstractas o genéricas (referencia a variables de tipo objeto). (28)

Conclusiones parciales.

Con el estudio de la MD, mediante las definiciones y caracterizaciones planteadas, se logró una mejor comprensión del objeto de estudio. El análisis de las herramientas para trabajar con MD permitió decidir que es necesario implementar los algoritmos de árboles de decisión para integrarlos al SGBD PostgreSQL 9.3. Mediante la técnica de árboles de decisión como parte de la MD, se orientó la investigación a la selección de los algoritmos a implementar: ID3, *Decision Stump* y C4.5. Se seleccionaron la metodología de desarrollo de *software* XP para guiar el ciclo de vida de la propuesta de solución, la metodología de MD CRISM-DM, así como las herramientas y lenguajes de programación a utilizar.

CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN

Introducción

En el presente capítulo se describen las principales características que tendrá la extensión “arboles_decision”. Para ello se representa primeramente el modelo de dominio con el fin de incorporar los conceptos clave del problema y alcanzar un mejor entendimiento del negocio. Se definen además las historias de usuario, que describen brevemente lo que el sistema debe realizar. Posteriormente se puede encontrar la lista de reserva que contiene los requisitos funcionales y no funcionales que el sistema debe cumplir. Se procede a puntualizar cada una de las tareas que se van a elaborar dentro de cada historia de usuario. Se planifican las iteraciones y se realiza una estimación de la realización de cada tarea. Se describen los estándares de codificación y se abarca concisamente la implementación de los algoritmos desarrollados incluyendo imágenes del código.

2.1 Modelo de dominio

Aunque la metodología de desarrollo de *software* XP no cuenta con una técnica para describir el negocio, con el fin de proporcionar un mejor entendimiento de los principales conceptos que intervienen y que describen el flujo y manejo de la información, se decide realizar un modelo de dominio. Este modelo permite definir el alcance de un sistema, quiénes y qué elementos están involucrados. Representa un modelo conceptual de las entidades que participan del negocio y sus relaciones. Fue construido haciendo uso de las reglas de UML para lograr un entendimiento común entre la autora de la presente investigación y futuros especialistas que la consulten, como un medio para comprender el sistema que se va a realizar. También puede ser tomado como el punto de partida para el diseño del sistema.

En la siguiente figura se muestra la descripción del negocio, donde un usuario accede al SGBD PostgreSQL el cual incluye funciones de la técnica de árboles de decisión de MD que generan reglas de decisión. Estas son analizadas para la toma de decisiones.

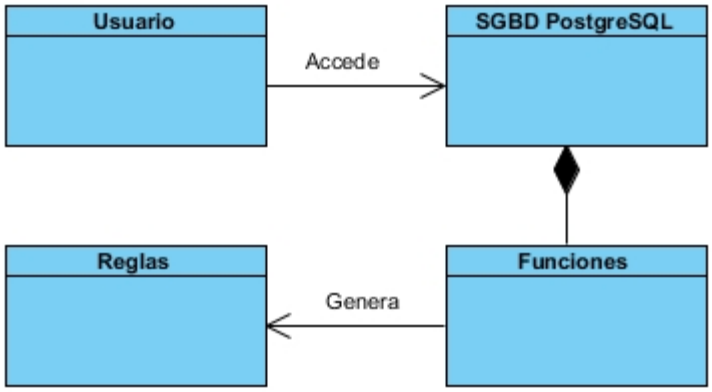

Fig. 12: Modelo de dominio del sistema.

Usuario: Persona que interactúa con la herramienta.

SGBD PostgreSQL: Sistema Gestor de Bases de Datos.

Funciones: Conjunto de funciones que se van a integrar al SGBD PostgreSQL.

Reglas: Conjuntos de reglas que genera el SGDB PostgreSQL.

2.3 Propuesta del componente a desarrollar

Lo que se propone realizar con el siguiente trabajo de diploma es una extensión con el objetivo de integrar los algoritmos de árboles de decisión al SGBD PostgreSQL. Esta permitirá a los usuarios a través de consultas analizar la información contenida en las bases de datos, lo cual es de gran importancia pues incorporando estos algoritmos se aprovechan las potencialidades del SGBD PostgreSQL, el cual brindará una variedad de algoritmos a la hora de realizar cualquier análisis de MD con grandes volúmenes de información.

2.4 Historias de usuario

En la metodología XP las Historias de usuario (HU) sustituyen a los documentos de especificación funcional, y a los casos de uso. Estas historias son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Las historias de usuario deben poder ser programadas en un tiempo entre una y tres

semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia. (29)

Se definieron para el desarrollo de los algoritmos de árboles de decisión 6 historias de usuarios, a continuación se muestran las tablas de las 3 primeras que corresponden al algoritmo ID3 de la técnica de árboles de decisión:

Tabla 1: Historia de usuario: Calcular la entropía y la ganancia de información del algoritmo ID3 de la técnica de árboles de decisión.

Historia de usuario	
Número: 1	**Nombre:** Calcular la entropía y la ganancia de información del algoritmo ID3 de la técnica de árboles de decisión.
Cantidad de modificaciones: 1	
Usuario: Yudaisy Rivera Barrios	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 semanas
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Se calcula por cada uno de los valores de las variables la entropía dada la variable clasificadora. Luego con esa entropía se calcula la ganancia de información de cada variable. Estos resultados se guardan en una tabla para analizar cuál será la variable que mejor clasifica.	
Observaciones: N/A	
Prototipo de interfaz: N/A	

Tabla 2: Historia de usuario: Definir la raíz, las ramas y las hojas para conformar las reglas del algoritmo ID3 de la técnica de árboles de decisión.

Historia de usuario	
Número: 2	**Nombre:** Definir la raíz, las ramas y las hojas para conformar las reglas del algoritmo ID3 de la técnica de árboles de decisión.
Cantidad de modificaciones: 1	
Usuario: Yudaisy Rivera Barrios	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 semanas
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Se selecciona la variable de mayor ganancia de información y se crea una rama para cada uno de sus valores. De cada valor se analiza si todos tienen el mismo clasificador para definirlo como nodo hoja. Si no se modifica la tabla original con los datos correspondientes a esa rama y se vuelve a calcular la ganancia de información para realizar el mismo proceso. Esto concluye cuando no queden más variables para el análisis o que todos los nodos sean hojas.	
Observaciones: N/A	
Prototipo de interfaz: N/A	

Tabla 3: Historia de usuario: Generar reglas de decisión a través del algoritmo ID3 de la técnica de árboles de decisión.

Historia de usuario	
Número: 3	**Nombre:** Generar reglas de decisión a través del algoritmo ID3 de la técnica de árboles de decisión.
Cantidad de modificaciones: 0	

Usuario: Yudaisy Rivera Barrios	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 3 semanas
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Permite al usuario generar a partir de la raíz, ramas y nodos, reglas de decisión a través del algoritmo ID3 de la técnica de árboles de decisión.	
Observaciones: N/A	
Prototipo de interfaz: N/A	

2.5 Lista de Reserva del Producto

Una vez definidos los requisitos del *software* estos son agrupados en la Lista de Reserva del Producto (LRP). La cual es una tabla que contiene los requisitos funcionales y no funcionales que el sistema debe cumplir, clasificados por una prioridad, según las necesidades del cliente. 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.

Se definieron para el desarrollo de los algoritmos de árboles de decisión 6 requisitos funcionales y 3 requisitos no funcionales, los cuales se muestran en la siguiente tabla:

Tabla 4: Lista de reserva del producto.

Ítem	Descripción	Estimación	Estimado por
Requisitos Funcionales			
Prioridad: ALTA			
1	Calcular la entropía y la ganancia de información del algoritmo ID3 de la técnica de árboles de decisión.	3 semanas	Analista
2	Definir la raíz, las ramas y las hojas para conformar las reglas del algoritmo ID3 de la técnica de árboles de decisión.	3 semanas	Analista
3	Generar reglas de decisión a través del algoritmo ID3	3 semanas	Analista

	de la técnica de árboles de decisión.		
4	Calcular la ganancia y la razón de ganancia del algoritmo C4.5 de la técnica de árboles de decisión.	3 semanas	Analista
5	Generar reglas de decisión a través del algoritmo C4.5 de la técnica de árboles de decisión.	3 semanas	Analista
Prioridad: MEDIA			
6	Generar reglas de decisión a través del algoritmo *Decision Stump* de la técnica de árboles de decisión.	3 semanas	Analista
Requisitos No Funcionales			
7	Facilidad de uso: Para utilizar la extensión es necesario poseer conocimientos elementales del Gestor PostgreSQL.		
8	*Software*: Para utilizar la extensión debe estar instalado, el gestor PostgreSQL a partir de la versión 9.1.		
9	Hardware: El ordenador donde se utilice la extensión debe contar con 1 Gb de memoria RAM como mínimo, y un microprocesador de doble núcleo de 1.6 GHz de frecuencia.		

2.6 Tareas de la ingeniería

Una vez descritas las historias de usuarios, se procede a describir cada una de las tareas que se van a elaborar dentro de cada historia de usuario, cada una contendrán los objetivos que se deben cumplir, el tiempo en que tardará en realizar la tarea y el responsable asignado.

Las tareas de la ingeniería se consideran como las entradas de trabajo para el equipo de programadores. Es la ficha que contiene el número identificador de la tarea, el identificador de la historia de usuario con la que está relacionada, el nombre de la tarea, la fecha de inicio, la fecha de fin, el equipo responsable y la descripción.

Se definieron un total de 16 tareas de la ingeniería de 2 a 3 por cada historia de usuario. A continuación se muestra 3 ejemplos de la tarea de ingeniería asociada a la historia de usuario: Definir la raíz, las ramas y las hojas para conformar las reglas del algoritmo ID3 de la técnica de árboles de decisión.

Tabla 5: Tareas de la ingeniería 1 de la historia de usuario 2.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 2
Nombre Tarea: Estudio de los pasos para definir la raíz, las ramas y hojas para conformar la reglas de decisión.	
Tipo de Tarea: Estudio	**Puntos Estimados:** 1 semana
Fecha Inicio: 9/02/2014	**Fecha Fin:**15/02/2014
Programador Responsable: Yudaisy Rivera Barrios	
Descripción: Realizar un estudio de los pasos para definir la raíz, las ramas y hojas para conformar la reglas de decisión del algoritmo ID3 de la técnica de árboles de decisión de MD.	

Tabla 6: Tarea de la ingeniería 2 de la historia de usuario 2.

Tarea de Ingeniería	
Número Tarea:2	**Número Historia de Usuario:** 2
Nombre Tarea: Definir la raíz, las ramas y las hojas para conformar las reglas de decisión.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 16/02/2014	**Fecha Fin:**22/02/2014
Programador Responsable: Yudaisy Rivera Barrios	

Descripción: Se llama a la función que calcula la ganancia de información de cada atributo, con la tabla y clase entradas por parámetro. A partir del resultado de la ganancia de información de cada atributo, se selecciona el atributo con mayor ganancia de información como raíz y las ramas van a ser todos los valores distintos del mismo. Luego para cada rama se analiza si su clasificador tiene el mismo valor para crear la hoja con el valor del clasificador al que corresponde. Estos elementos son guardados en una tabla para luego general las reglas de decisión.

Tabla 7: Tarea de la ingeniería de la historia de usuario 1.

Tarea de Ingeniería	
Número Tarea:3	**Número Historia de Usuario:** 2
Nombre Tarea: Definir los nuevos valores de la raíz, las ramas y las hojas para seguir conformando nuevas reglas de decisión.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1 semana
Fecha Inicio: 23 /02/2014	**Fecha Fin:** 1/03/2014
Programador Responsable: Yudaisy Rivera Barrios	
Descripción: Para cada rama que su clasificador no tiene el mismo valor se calcula la ganancia de información modificando la tabla entrada por parámetro con los valores correspondientes a dicha rama. Para ello se llama a la función que calcula la ganancia de información de cada atributo, con la tabla modificada y la misma clase. A partir del resultado de la ganancia de información de cada atributo, se selecciona el atributo con mayor ganancia de información como raíz y las ramas van a ser todos los valores distintos del mismo. Este proceso se seguirá realizando hasta que no queden atributos por analizar o que todas las ramas sean hojas.	

2.7 Plan de iteraciones

El Plan de iteraciones permite definir el número de iteraciones en que se desarrollará el sistema de modo que se puedan precisar con exactitud las entregas inmediatas y la entrega final. Este artefacto posee una estrecha relación con las Historias de Usuario anteriormente detalladas, ya que se tiene en cuenta la

prioridad definida por el cliente para cada historia y se colocan en un orden con prioridad. La implementación del sistema propuesto tendrá una duración de 19 semanas, tiempo en que se realizarán 2 iteraciones

Iteración 1

Esta iteración tiene como objetivo la implementación de las historias de usuario de mayor prioridad o prioridad alta. Al finalizar se contará con las funcionalidades descritas en las historias de usuario 1, 2, 3, 4 y 5, que hacen referencia a las implementaciones de los algoritmos de árboles de decisión ID3 y C4.5.

Iteración 2

El objetivo de esta iteración es la implementación de las funcionalidades con prioridad media. Con la culminación de las mismas, se tendrán implementadas las peticiones del cliente descritas en la historia de usuario 6 que hace referencia a la implementación del algoritmo de árboles de decisión *Decision Stump*.

Tabla 8: 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, 3,4 y 5	16 semanas.
Iteración 2	En esta iteración se implementarán las Historias de Usuario que tengan la prioridad en el negocio MEDIO.	6	2 semanas.

2.8 Estándares de codificación

Extreme Programing (XP) promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo, y que facilite la recodificación. Los estándares de codificación son pautas 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.

Un estándar de codificación abarca todo lo referente al proceso de generación de código. Establecer el

estándar es necesario al iniciar cualquier proyecto de *software* para asegurar que todos los miembros del equipo de desarrollo trabajen de forma similar.

Declaración de variables:

Cada variable debe ser declarada en una línea y comentada. El nombre de las variables debe 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.

```
DECLARE
atrib VARCHAR;    -- variable que contiene los nombres de las columnas de la tabla
valores VARCHAR;  -- variable que contiene los nombres de los valores de las columnas
vc VARCHAR;       -- variable que contiene los nombres de los valores de la clase
p NUMERIC;        -- variable que contiene la cantidad de cada tipo de la clase
t NUMERIC;        -- variable que contiene el total de cada valor existente en las columna
i NUMERIC;        -- variable que contiene la entropia de cada valor de la columna
e NUMERIC;        -- variable que contiene la entropia de cada columna
total NUMERIC;    -- variable que contiene el total de tuplas de la tabla
tg NUMERIC;       -- variable que contiene el total de cada tipo de la clase
eg NUMERIC;       -- variable que contiene la entropia general
sume numeric;     -- variable que contiene la sumatoria de la entropia de cada columna
sumeg numeric;    -- variable que contiene la sumatoria de la entropia general
gi NUMERIC :=0;   -- variable que contiene la ganancia de informacion
```

Fig. 13: Ejemplo de declaraciones de variables en la implementación del algoritmo ID3.

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 (ejemplo: 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. Teniendo 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.

Identación:

La unidad de identado es de 2 espacios. El uso de la tabulación debe ser evitado porque (tal como se

escribía en el siglo pasado) no existe un estándar que determine con precisión el ancho que va a producir la tabulación.

```
For vc in execute 'select distinct ' ||$2|| ' from ejemplo ' -- ciclo para recorrer cada valor distinto del segundo parametro de entrada
loop
  if atrib <> $2  then -- condicion para no tomar encuenta la columna del segundo parametro de entrada q es la clase
    execute 'SELECT COUNT(*) FROM ejemplo WHERE ' ||atrib|| ' = ' || '''' ||valores||''''|| ' and ' ||$2|| ' = '|| '''' ||vc||'''' into p;
    -- cuenta la cantidad de cada tipo del segundo parametro de entrada (si y no) de los valores existente en las columnas
    execute 'select count(*) from ejemplo where ' ||atrib|| ' = ' || '''' ||valores|| ''''  into t;
    -- cuenta el total de cada valor existente en las columnas
    if p=0 or t=0 then
      i:=0;-- como el logaritmo de cero no existe la entropia es cero
    else
      i:=-(p/t) * (log(2,p)-log(2,t));-- calcula la entropia de cada valor de la columna
    end if;
```

Fig. 14: Ejemplo de identación en la implementación del algoritmo ID3.

***Comentarios*:**

Es conveniente dejar información que pueda ser leída tiempo después y se necesita entender que fue lo que se hizo en el fragmento de código, por lo que se recomienda usar comentarios, estos deben ser escritos correctamente y claros. Generalmente deben usarse comentarios de una sola línea. Se debe reservarlos comentarios de bloques para la documentación formal o para comentar porciones de código.

```
sumi:=sumi+i;-- sumatoria de la entropia de cada valor de la columna
sume:=sume+e;-- sumatoria de la entropia de cada columna
sumeg:=sumeg+eg;-- sumatoria de la entropia general
insert into temporall values (atrib,valores,$2,vc,p,tg,i,e,eg);
-- inserta una tabla con los parametros q se le pasa
/*select v1 from temporal where v2 =(select max(v2) from temporal)into raiz;
  execute 'ALTER TABLE ejemplo DROP COLUMN ' ||raiz;
  raiz:='';*/
```

Fig. 15: Ejemplo de comentarios en la implementación del algoritmo ID3.

2.9 Implementación de los algoritmos

A continuación se realiza una breve descripción de los algoritmos de árboles de decisión implementados para la integración al SGBD PostgreSQL, posibilitándole al usuario un mejor entendimiento. También se explican las funciones creadas para el funcionamiento de dichos algoritmos, el tipo de atributos que permiten y el resultado que devuelven.

2.9.1 Algoritmo ID3

La implementación del algoritmo ID3 se realizó utilizando el pseudocódigo mostrado en la figura 6 (Fig. 6).

La función “algoritmo_id3” solo permite trabajar con tablas que tengan atributos nominales y en la misma no debe haber atributos con valores desconocidos para obtener el resultado deseado, en esta función se realizará el cálculo de la ganancia de información utilizando las ecuaciones mostradas anteriormente en las figuras 3, 4 y 5. La misma toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar y el nombre de la clase (columna) por la que se va a clasificar. La función “principal_id3”, va a ser la encargada de llamar iterativamente a la función “algoritmo_id3” la cual calculará la ganancia de información para cada atributo. Dicha función tomará por cada llamada el mejor clasificador y sus valores correspondientes y elaborará las reglas. Esta función devuelve como resultados una tabla con las reglas creadas las cuales permitirán apoyar el proceso de toma de decisiones.

A continuación se muestran fragmentos de la implementación:

```
For vc in execute 'select distinct ' ||$2|| ' from ejemplo '
loop
  -- ***condicion para no tomar encuenta la columna del segundo parametro de entrada que es la clase***
  if atrib <> $2  then
    -- ***cuenta la cantidad de cada tipo del segundo parametro de entrada (si y no) de los valores existente
    execute 'SELECT COUNT(*) FROM ejemplo WHERE ' ||atrib|| ' = ' || '''' ||valores||''''|| ' and ' ||$2|| ' =
    -- ***cuenta el total de cada valor existente en las columnas***
    execute 'select count(*) from ejemplo where ' ||atrib|| ' = ' || '''' ||valores|| ''''  into t;
    --***se calcula la entropia de cada valor***
    if cc=0 or t=0 then
      i:=0;-- como el logaritmo de cero no existe la entropia es cero
    else
      i:=-(cc/t) * (log(2,cc)-log(2,t));-- calcula la entropia de cada valor de la columna
    end if;
    -- ***cuenta total de tuplas***
    execute 'SELECT COUNT(*) FROM ejemplo' into total;
    -- ***cuenta el total de cada tipo del 2do parametro de entrada***
    execute 'select count(*) from ejemplo where ' ||$2|| ' = ' || '''' ||vc|| ''''  into tc;
    --***calcula la entropia general***
    if tc=0 then
      eg:=0;
    else
      eg:=-(tc/total)*(log(2,tc)-log(2,total));
    end if;
```

Fig. 16: Fragmento de código de la función “algoritmo_id3” de la implementación del algoritmo ID3.

```
a = arboles_decision.algoritmo_id3('ejemplo',$2);--se llama a la funcion
select vl from ta_ganancia where v2 =(select max(v2) from ta_ganancia)into raiz;
--se busca el atributo de mayor ganancia y se guarda en raiz
atrib := raiz;
--*** para insertar los valores a ta_arbol***
For valores in  EXECUTE 'select v2 from ta_entropia where vl ='||''''||raiz||''''
loop
  select vl from ta_entropia where vl = raiz into vvl;--raiz
```

Fig. 17: Fragmento de código de la función “principal_id3” de la implementación del algoritmo ID3.

2.9.2 Algoritmo C4.5

La implementación del algoritmo C4.5 se realizó utilizando el pseudocódigo mostrado en la figura 8 (Fig. 8). La función “algoritmo_c45” permite trabajar con tablas que tengan atributos tanto nominales como continuos y también puede existir atributos con valores desconocidos, en esta función se realizará el cálculo de la razón de ganancia utilizando la ecuación mostrada anteriormente en la figura 7 (Fig. 7). La misma toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar y el nombre de la clase (columna) por la que se va a clasificar. La función “principal_c45”, va a ser la encargada de llamar iterativamente a la función “algoritmo_c45” la cual calculará la razón de ganancia para cada atributo. Dicha función tomará por cada llamada el mejor clasificador y sus valores correspondientes y elaborará las reglas. Esta función devuelve como resultados una tabla con las reglas creadas las cuales van a permitir la toma de decisiones.

A continuación se muestra un fragmento de la implementación:

```
For columna in select column_name from information_schema.columns where table_name = 'ejemplo'
loop
  --***calcular el promedio de la columna***
  execute 'SELECT Avg( '||columna ||' ) FROM ejemplo' into prom  ;
  nueva_columna:=columna||'_n';
  nueva_prom:=''''||prom||'''';
  --***modificar la tabla entrada***
  execute ' alter TABLE  ejemplo add column '|| nueva_columna ||' varchar(350)';
  execute '  update ejemplo set '|| nueva_columna ||' = case
  when '||columna||' <= '||prom||' then
    ''menor''
  else
    ''mayor''
  end';
  execute 'ALTER TABLE ejemplo DROP COLUMN ' ||columna; --se elimina la columna con valor cont
end loop
```

Fig. 18: Fragmento de código de la función “algoritmo _c45” de la implementación del algoritmo C4.5.

```
a = arboles_decision.algoritmo_c45('ejemplo',$2);--se llama a la funcion
select vl from ta_ganancia where v2 =(select max(v2) from ta_ganancia)into raiz;
--se busca el atributo de mayor razon de ganancia y se guarda en raiz
atrib := raiz;
--*** para insertar los valores a ta_arbol***
For valores in  EXECUTE 'select v2 from ta_entropia where vl ='||''''||raiz||''''
loop
```

Fig. 19: Fragmento de código de la función “principal _c45” de la implementación del algoritmo C4.5.

2.9.3 Algoritmo Decision Stump

La función “decision_stump” permite trabajar con tablas que tengan atributos tanto nominales como continuos y clases de ambos tipos, también pueden existir atributos con valores desconocidos. En esta función se realizará el cálculo de la ganancia de información donde se trabajará con la ecuación de la entropía para el caso de atributos simbólicos y la de la varianza para atributos numéricos, utilizando las ecuaciones mostradas anteriormente en la figura 10 (Fig. 10) y figura 11 (Fig. 11) respectivamente. La misma toma como parámetros de entrada el nombre de la tabla sobre la cual se va a trabajar y el nombre de la clase (columna) por la que se va a clasificar. Una vez definido el atributo y su valor correspondiente se crean las tres reglas: una de ellas será para el caso de que el atributo sea desconocido, otra para el caso de que el atributo sea igual al valor y la última cuando el atributo es distinto a dicho valor. Esta función devuelve como resultados una tabla con las reglas que permitirán apoyar el proceso de toma de decisiones.

A continuación se muestra un fragmento de la implementación:

```
  ii:=0;-- como el logaritmo de cero no existe la entropia es cero
else
ii:=ni *log(2,ni);-- calcula la entropia del valor
end if;
end if;
if raiz <> $2 and raiz <> valores then
-- ***cuenta la cantidad de cada tipo del segundo parametro de entrada (si y no) de los valores existente en las columnas***
execute 'SELECT COUNT(*) FROM ejemplo WHERE ' ||raiz|| ' <> ' || '''' ||valores||''''|| ' and ' ||raiz|| ' <> '''' '|| 'and '
execute 'select count(*) from ejemplo where ' ||raiz|| ' <> ' || '''' ||valores|| ''''|| ' and ' ||raiz|| ' <> '''' ' into td
probd:=nd/td;
if nd=0 then
  id:=0;-- como el logaritmo de cero no existe la entropia es cero
else
id:=nd *log(2,nd);-- calcula la entropia del valor
end if;

end if;
if raiz <> $2 and raiz = '' then
-- ***cuenta la cantidad de cada tipo del segundo parametro de entrada (si y no) de los valores existente en las columnas***
execute 'SELECT COUNT(*) FROM ejemplo WHERE ' ||raiz|| ' = '''' '|| ' and ' ||$2|| ' = '|| '''' ||vc||'''' into nn;
execute 'select count(*) from ejemplo where ' ||raiz|| ' <> '''' ' into tn;
probn:=nn/tn;
--***se calcula la entropia de cada valor***
if nn=0 then
  en:=0;-- como el logaritmo de cero no existe la entropia es cero
else
```

Fig. 20: Fragmento de código de la implementación del algoritmo *Decision Stump*.

2.10 Integración de los algoritmos al SGBD PostgreSQL.

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. La principal ventaja de usar una extensión, es 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. (26)

2.10.1 Creación de la extensión reglas de árboles de decisión.

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 "reglas_ arboles_ decision.control" creado para agregar la extensión donde se cargarán las funciones de los algoritmos implementados se definieron los siguientes parámetros:

- Comment: 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.

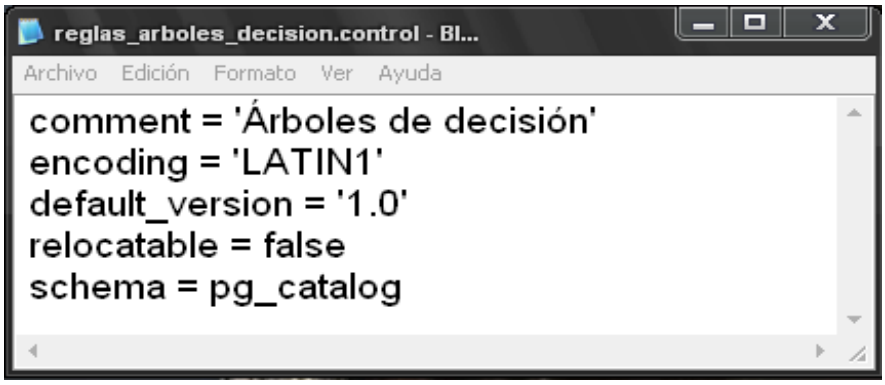

Fig. 21: Archivo que contiene las características de la extensión.

Una vez definido en el archivo “reglas_arboles_ decision.control” se especifica el archivo que contendrá el código de las funciones desarrolladas “reglas_ arboles_ decision--1.0.sql”.

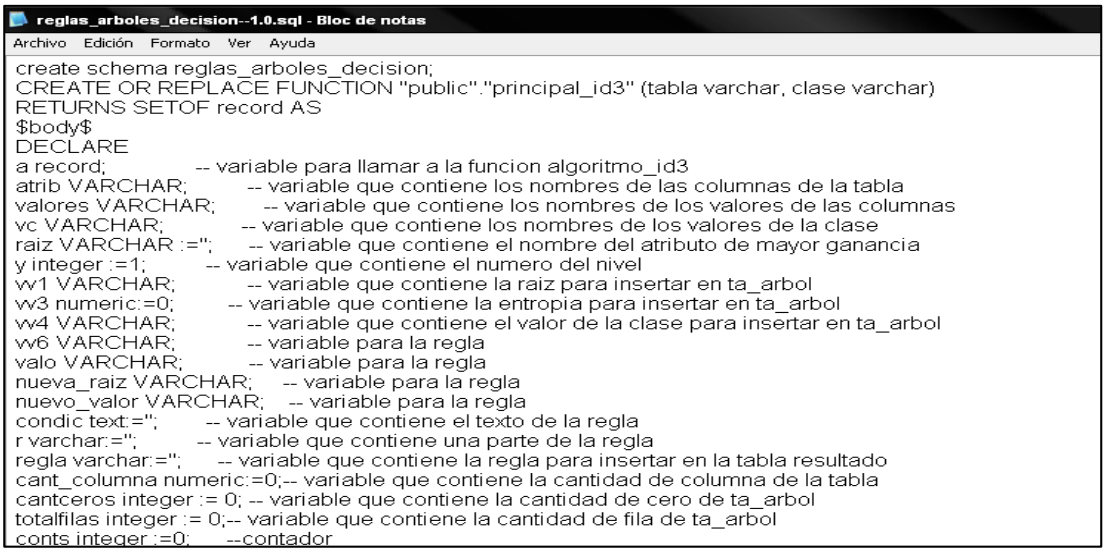

Fig. 22: Archivo que contiene el código de la extensión.

Para que los usuarios puedan utilizar la extensión de reglas de MD simplemente deben ejecutar el comando “CREATE EXTENSION arboles_decision” que cargará la extensión como se muestra en la siguiente figura:

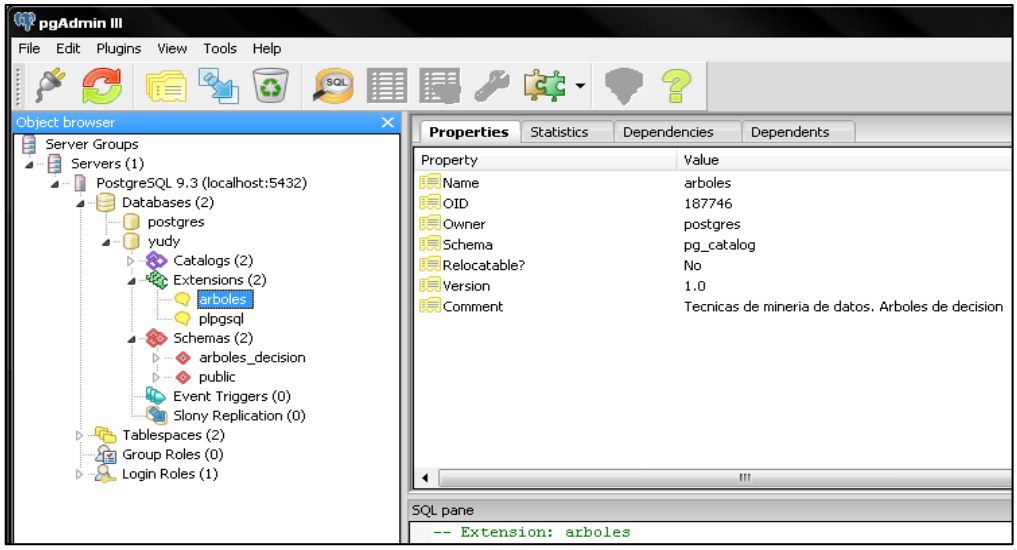

Fig. 23: Extensión “arboles_decision” creada.

Conclusiones parciales

Durante el desarrollo del capítulo se evidenció que con el uso de la metodología de desarrollo de *software* XP se presentaron los elementos descriptivos adecuados a los requerimientos del cliente. Por otra parte, aunque XP no exige la realización de un Modelo de Dominio, se consideró necesario plantearlo, debido a la importancia que reviste para la organización, control y un mejor entendimiento de la información de la aplicación. La creación de las Historias de Usuario y la Lista de Reserva del Producto, permitieron definir los requisitos necesarios para lograr el objetivo propuesto y facilitaron la planificación del tiempo de desarrollo, el cual fue realizado guiándose por el estándar de codificación detallado. Los elementos anteriores dieron paso a una correcta implementación e integración en el SGBD PostgreSQL.

CAPÍTULO 3: APLICACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Introducción

XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. Es por ello que en este capítulo se analizan las técnicas que define la metodología XP para diseñar los casos de pruebas que guiarán la validación de la aplicación. Asimismo se realiza el proceso de MD siguiendo la metodología CRISP-DM y se comparan los resultados de los algoritmos implementados con la herramienta Weka.

3.1 Pruebas del Sistema

Una de las prácticas de la metodología XP es el uso de las pruebas para garantizar el funcionamiento de los códigos que se vayan implementando. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección (30).
La metodología XP divide las pruebas en dos grupos:

- ✓ Pruebas unitarias.
- ✓ Pruebas de aceptación.

Pruebas Unitarias

Las pruebas unitarias o pruebas de unidad consisten en comprobaciones (manuales o automatizadas) desarrolladas por los programadores. Las mismas se realizan para verificar que el código correspondiente a un módulo concreto se comporta de manera esperada. Las pruebas unitarias proporcionan beneficios tales como (30):

- ✓ Brindan al programador una inmediata retroalimentación de cómo está realizando su trabajo.
- ✓ El programador puede realizar cambios de forma segura respaldada por efectivos casos de pruebas.
- ✓ Permite saber si una determinada funcionalidad se puede agregar al sistema existente sin alterar el funcionamiento actual del mismo.

Pruebas de Aceptación

Las pruebas de aceptación se elaboran a lo largo de la iteración, en paralelo con el desarrollo del sistema y adaptándose a los cambios que el sistema sufra (31). Las pruebas de aceptación pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepta de forma adecuada y que la salida producida es correcta, ya que no es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar (32). Estas pruebas son más importantes que las pruebas unitarias dado que significan la satisfacción del cliente con el producto desarrollado y el final de una iteración y el comienzo de la siguiente (31). Los clientes son los principales responsables de verificar que los resultados de estas pruebas sean correctos, de forma que se cubran, sino todas, si la mayor cantidad posible de las funcionalidades registrada en una historia de usuario (30). Existen dos tipos de pruebas de aceptación:

- ✓ La prueba alfa: Se lleva a cabo por un cliente, en el lugar de desarrollo. Se usa el *software* de forma natural con el desarrollador como observador del usuario.
- ✓ La prueba beta: Se lleva a cabo por los usuarios finales del *software*, en los lugares de trabajo de los clientes. A diferencia de la prueba alfa, el desarrollador no está presente normalmente.

Pruebas alfa

Con el fin de encontrar la mayor cantidad de errores y defectos posibles, y que estas sean realizadas por el propio usuario final y no por el desarrollador del producto, serán aplicadas a la extensión desarrollada pruebas alfa.

Las pruebas alfa se llevan a cabo por un cliente en el lugar de desarrollo. Se usa el *software* de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso. Las pruebas alfa se hacen en un entorno controlado. Se realizan después de todos los procedimientos de prueba básicos, y se producen después de las pruebas del sistema. Por lo general emplea a cualquiera de las pruebas de caja blanca o caja negra para probar el *software*. (30)

Después de realizado un estudio de las pruebas de *software* que propone la metodología XP, se decide llevar a cabo la estrategia de pruebas de aceptación, debido a que van dirigidas a evaluar el trabajo desarrollado desde una perspectiva visual, centrándose en el perfecto funcionamiento de las funcionalidades a las cuales el cliente se enfrenta. Esta estrategia significa la satisfacción del cliente con el producto desarrollado, el final de una iteración donde se han cumplido las expectativas y el comienzo de la siguiente. Es casi imposible obtener una implementación libre de errores, es por ello que se debe tener bien definido un criterio de aprobación, para

saber el momento en el que el *software* está listo para ser liberado. A continuación se detalla el método seleccionado para realizar las pruebas de aceptación.

3.1.1 Método seleccionado: Prueba de Caja Negra.

Para determinar la calidad de la extensión desarrollada se aplicaron pruebas de caja negra, las cuales se centran en los requisitos funcionales del sistema sin tener en cuenta su estructura lógica interna. Tienen como objetivo demostrar que las funciones del *software* son operativas, que las entradas acepten de forma adecuada y se produzca un resultado correcto, teniendo en cuenta que la integridad de la información externa se mantenga. Estas pruebas permiten descubrir diferentes tipos de errores que no se encuentran con los métodos de caja blanca como son: errores de rendimiento, errores en la interfaz, funciones incorrectas o ausentes, errores en estructuras de datos o en accesos a bases de datos externas, errores de inicialización y de terminación (32). Durante las iteraciones, las HU seleccionadas serán traducidas a pruebas. En ellas se especifican desde la perspectiva del cliente, los escenarios para probar que una HU ha sido implementada correctamente. El objetivo final de las pruebas es garantizar que los requisitos han sido cumplidos y que el sistema es aceptable.

Técnicas de pruebas de Caja Negra:

✓ *Técnica de prueba basada en grafos***:** En la técnica se debe entender los objetos que se modelan en el *software* y las relaciones que conectan a estos, tales como objetos de datos, objetos de programa como módulos o colecciones de sentencias del lenguaje de programación. (32)

✓ *Partición equivalente:* Divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. En otras palabras, este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia, de tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada clase es equivalente, a una prueba realizada con cualquier otro valor de dicha clase. (32)

✓ *Análisis de Valores Límites:* El análisis de valores límite (AVL) es una técnica de diseño de casos de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de la clase. (32)

Una vez analizadas las técnicas anteriores se decidió utilizar la técnica de partición de equivalencia, por la gran importancia que posee la misma permitiendo encontrar gran número de errores en pocas iteraciones, por estar dirigidas a una definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

Un caso de prueba especifica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse. Se pueden realizar muchos casos de prueba para determinar que una HU es completamente satisfactoria. Con el propósito de comprobar que todas las HU de una aplicación son revisadas, debe haber al menos un caso de prueba para cada una de ellas.
Los casos de prueba según la metodología XP deben cumplir las siguientes características:

- ✓ Los casos de prueba (CP) deben escribirse para realizar las pruebas desde el punto de vista del usuario.
- ✓ Un error aunque sea en un solo paso de un caso hace que se considere que falló el caso entero.
- ✓ Brindar un *feedback*[2] rápido y concreto de cómo se está desarrollando la iteración y el proyecto.
- ✓ Los casos de prueba exitosos de una iteración deben repetirse con éxito en las siguientes iteraciones.

3.1.2 Casos de pruebas basados en HU

Con el propósito de obtener una retroalimentación rápida y concreta de cómo se están desarrollando las funcionalidades, se realizaron los casos de prueba correspondiente a las 6 HU descritas en la fase de Exploración.

A continuación se muestra el caso de prueba para la HU “Generar reglas de decisión a través del algoritmo ID3 de la técnica de árboles de decisión” con la descripción de las variables correspondientes. Se muestra una tabla donde se especifican las no conformidades detectadas por el CP y la solución a cada una de ellas. Finalmente se disponen los resultados de las pruebas en general para cada iteración.

[2] **Feedback**: característica de XP basada en el desarrollo incremental de pequeñas partes, con entregas y pruebas frecuentes y continuas, proporciona un flujo de retro-información valioso para detectar los problemas o desviaciones. (24)

Tabla 9: Generar reglas de decisión a través del algoritmo ID3 de la técnica de árboles de decisión.

Escenarios	Descripción	Respuesta del Sistema	Flujo Central
EC1.1 Ejecutar función con datos correctos.	En este escenario se crean las reglas de árboles de decisión introduciendo datos correctos.	Se muestran las reglas de árboles de decisión creadas.	1- El usuario selecciona la función “principal_id3”.
EC1.2 Ejecutar función con datos incorrectos.	En este escenario se crean las reglas de árboles de decisión introduciendo datos incorrectos.	Se muestra un mensaje de error en la herramienta.	2- Introduce los datos requeridos para la función “principal_id3”.
EC1.3 Ejecutar función con datos vacíos.	En este escenario se crean las reglas de árboles de decisión dejando datos a insertar vacíos.	Se muestra un mensaje de error en la herramienta.	3- Se generan las reglas de asociación para la función “principal_id3”.

3.2 Presentación de los resultados de las pruebas funcionales

Para llevar a cabo un proceso satisfactorio de pruebas se realizaron tres iteraciones. En la primera iteración se efectúan todo los CP, las no conformidades detectadas por ellos, son aceptadas y resueltas por el equipo de desarrollo. Luego para verificar la adecuada rectificación y solución de los problemas revelados, se procede a una segunda iteración en la que se pueden descubrir nuevas no conformidades y detectar una ineficiente solución a ellas. En la tercera iteración se deben haber corregido todas las no conformidades, y luego se aborta el proceso de pruebas.

Las pruebas han sido aplicadas a las 6 historias de usuario permitiendo detectar varios errores. Fueron encontradas en una primera iteración 7 no conformidades las cuales fueron resueltas en una semana. Para una segunda iteración se encontró 3 no conformidad, las cuales fueron solucionadas en un período de 3 días. Finalmente para una tercera iteración no se encontraron no conformidades. A continuación se

muestra una gráfica donde se recogen la cantidad de no conformidades encontradas y resueltas en cada iteración.

Fig. 24: Resultados de las pruebas.

3.3 Proceso de MD utilizando la metodología CRISP-DM

A continuación se muestra cómo se aplicó el proceso de MD a la base de datos de la aplicación de Genética Médica, utilizando la metodología CRISP-DM para validar los algoritmos integrados al SGBD PostgreSQL.

Genética médica

Entre las ramas de la medicina se encuentra la genética médica, la cual tiene como objetivo principal en Cuba estudiar cómo garantizar la reducción del impacto de las enfermedades genéticas sobre la salud y el bienestar de los individuos a través de estrategias de prevención. Ello permite ayudar a las personas con "desventajas" genéticas a vivir y a reproducirse de forma tan normal como sea posible, así como reducir la frecuencia y las manifestaciones clínicas de los defectos congénitos severos (33). Una de las acciones realizadas por el Centro Nacional de Genética Médica para alcanzar este objetivo ha sido la informatización de varios estudios realizados junto a la Universidad de las Ciencias Informáticas, pero debido al volumen de la información recopilada se decidió analizar el comportamiento a través de las técnicas de análisis de datos. (1)

Comprensión del negocio

Los objetivos del negocio son enmarcados a:

- Conocer las relaciones que se establecen entre las características de los pacientes:
- Determinar cuáles son las características de las personas que pueden influir en el grado de la discapacidad intelectual.

Como objetivo de MD se identificó:

Obtener reglas que permitan descubrir la influencia que tienen en el grado de la discapacidad intelectual parámetros como el sexo del paciente, color de la piel, zona donde nació, conducta del paciente, área cognitiva del paciente, edad en que la madre quedó embarazada, si la madre tuvo enfermedades infecciosas y si tuvo movimientos fetales durante el embarazo.

Comprensión de los datos

La comprensión de datos está relacionada 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. (34)

La información recopilada fue obtenida de una única fuente, que de manera centralizada es la empleada para recopilar los datos de los pacientes. Esta base de datos cuenta con 142 tablas y se encuentra sobre el gestor PostgreSQL. A continuación se describen los atributos significativos para darle solución a los objetivos propuestos en el epígrafe 3.4.

Tabla 10: Resumen descriptivo de la información relevante recopilada.

Nombre del atributo	Tipo de datos	Descripción	Nombre de la tabla
sexo	character	Almacena el sexo del paciente	tpersona
color_piel	character	Almacena el color de piel del paciente	tcolor_piel
zona	character varying	Almacena la zona del paciente	tzona
area_conducta	character varying	Almacena la conducta del paciente	tarea_conducta
area_cognocitiva	character varying	Almacena el nivel de comprensión del	tarea_cognocitiva

		paciente	
rango_edad_emb	character	Almacena el rango de edad que tenía la madre del paciente cuando se embarazó	trango_edad_embarazo
enferm_infec_embar	character	Almacena si la madre durante el embarazo tuvo enfermedades infecciosas	tmadre_emb
mov_fetales	character	Almacena si la madre tuvo movimientos fetales durante el embarazo	tmadre_emb
grado_de_rm	character varying	Almacena el grado de la discapacidad intelectual del paciente	tgrado_de_rm

Explorar y verificar la calidad de los datos

Para realiza el análisis, el juego de datos utilizado para mostrar en la investigación es ficticio, debido a la sensibilidad de los datos almacenados en la base de datos sobre la cual se está realizando el estudio.
Los resultados de la exploración realizada a los datos para verificar su calidad:

Los atributos sexo, color_piel, rango_edad_emb, zona, area_conducta, area_cognocitiva, enferm_infec_embar, mov_fetales y grado_de_rm no tienen ningún valor vacío.

Preparación de los datos

Una vez efectuada la recolección inicial de datos, se procede a su preparación para adaptarlos a las técnicas de MD que se utilicen posteriormente. La preparación de datos incluye las tareas generales de selección de datos a los que se va a aplicar una determinada técnica de modelado, limpieza de datos, generación de variables adicionales, integración de diferentes orígenes de datos y cambios de formato.

Esta fase se encuentra relacionada con la fase de modelado, puesto que en función de la técnica de modelado elegida, los datos requieren ser procesados de diferentes formas. Es así que las fases de preparación y modelado interactúan de forma permanente. (35)

Limpieza y transformación de los datos

Se realizaron un conjunto de consultas en las tablas seleccionadas para determinar las tuplas que no aportan ninguna información provechosa. En las mismas no se obtuvieron campos con valores desconocidos, ni tuplas inconsistentes.

No se tuvo que realizar la limpieza de los datos ya que no existían instancias con valores vacíos.

Para facilitar el posterior uso de estos atributos y simplificar el entendimiento, se sustituyeron los valores 1 por Masculino y 2 por Femenino pertenecientes al atributo sexo. También se realizaron transformaciones en los atributos que tenían tildes, caracteres como “ñ” y espacios entre las palabras por las que estaba compuesto (estos fueron sustituidos por guiones bajos “_”).

Modelado

La selección de las técnicas de MD y los algoritmos a emplear depende de los objetivos de MD propuestos en la fase de comprensión del negocio. La técnica seleccionada para darle cumplimiento a este objetivo es árboles de decisión y de ellas los algoritmos ID3, C4.5 y *Decision Stump*.
A continuación se realiza una comparación de los resultados obtenidos por los algoritmos integrados al SGBD PostgreSQL y los de la herramienta Weka, en la misma se utiliza la BD de Genética Médica.

Presentación de los resultados obtenidos por ambas herramientas del algoritmo ID3:

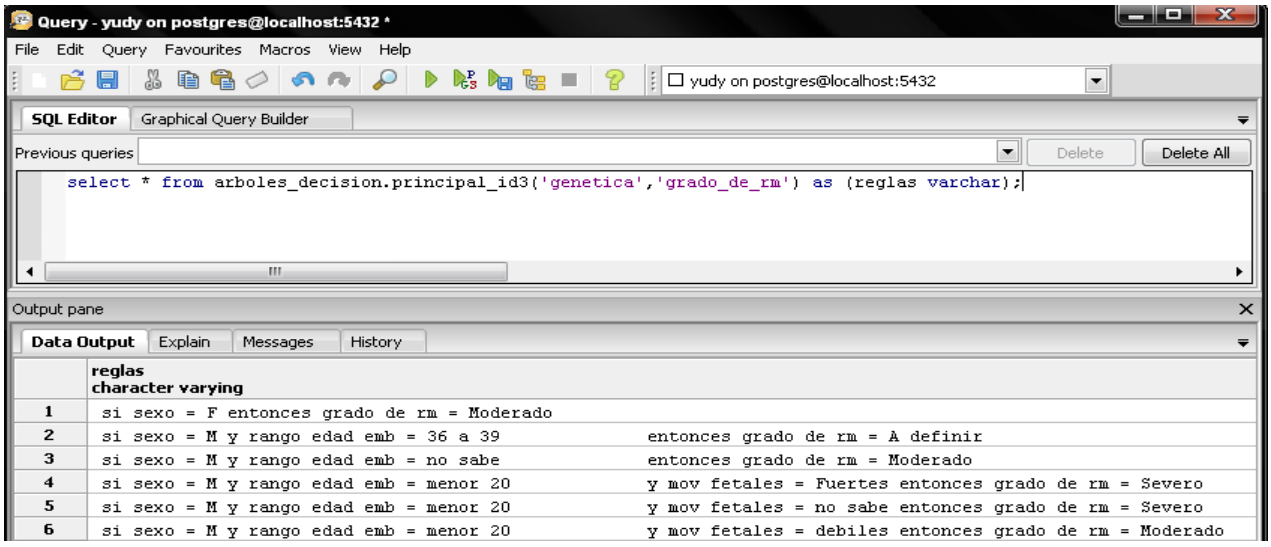

Fig. 25: Resultados obtenidos del algoritmo ID3 en el SGBD PostgreSQL 9.3.

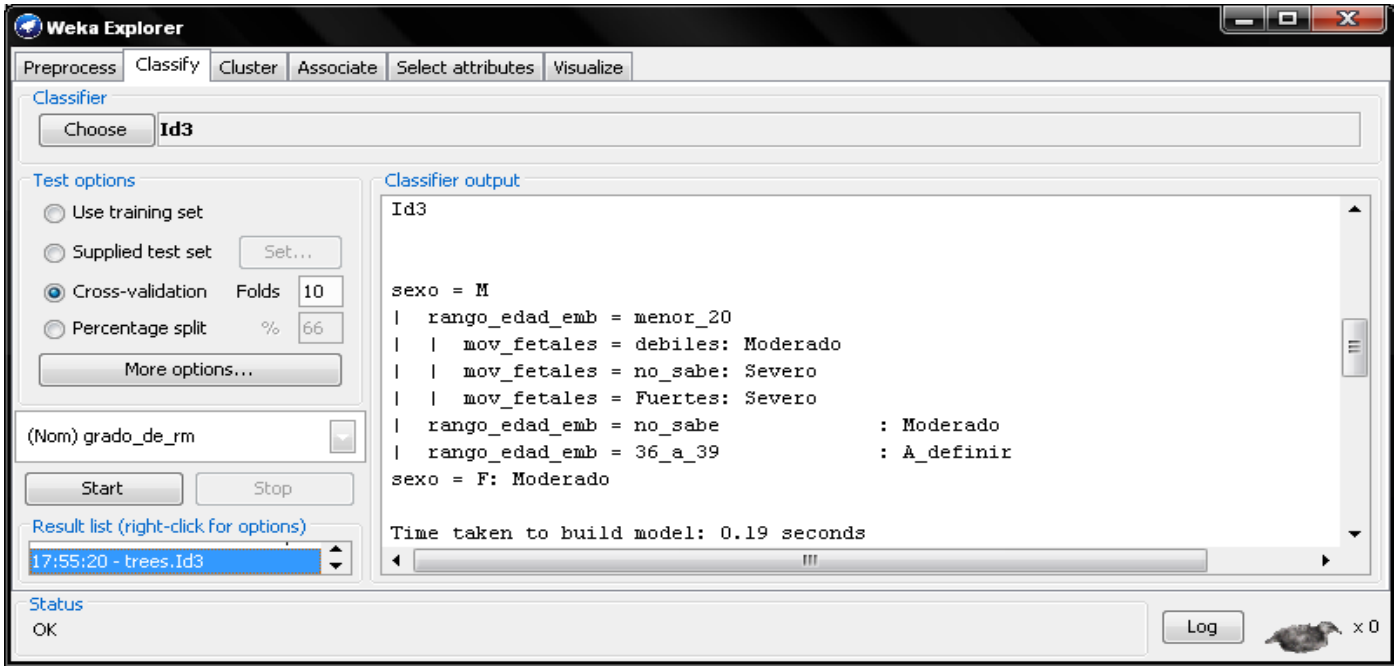

Fig. 26: Resultados obtenidos del algoritmo ID3 en el Weka.

Presentación de los resultados obtenidos por ambas herramientas del algoritmo C4.5:

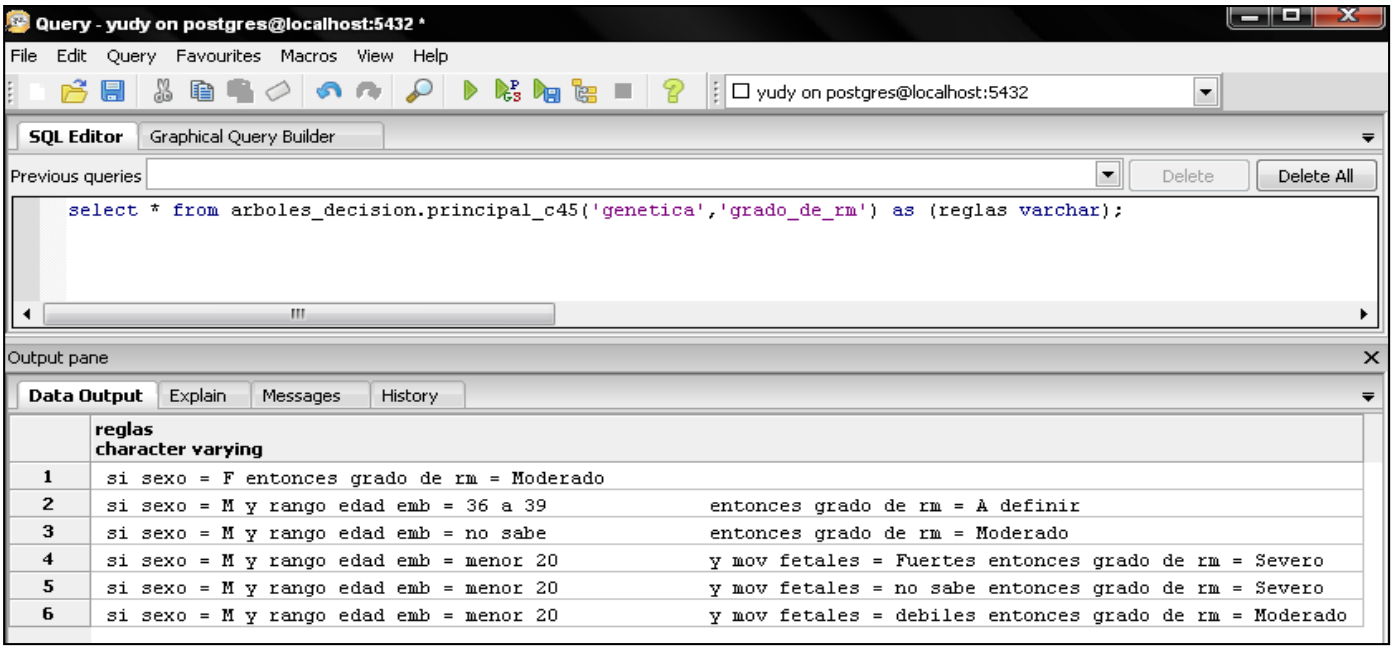

Fig. 27: Resultados obtenidos del algoritmo C4.5 en el SGBD PostgreSQL 9.3.

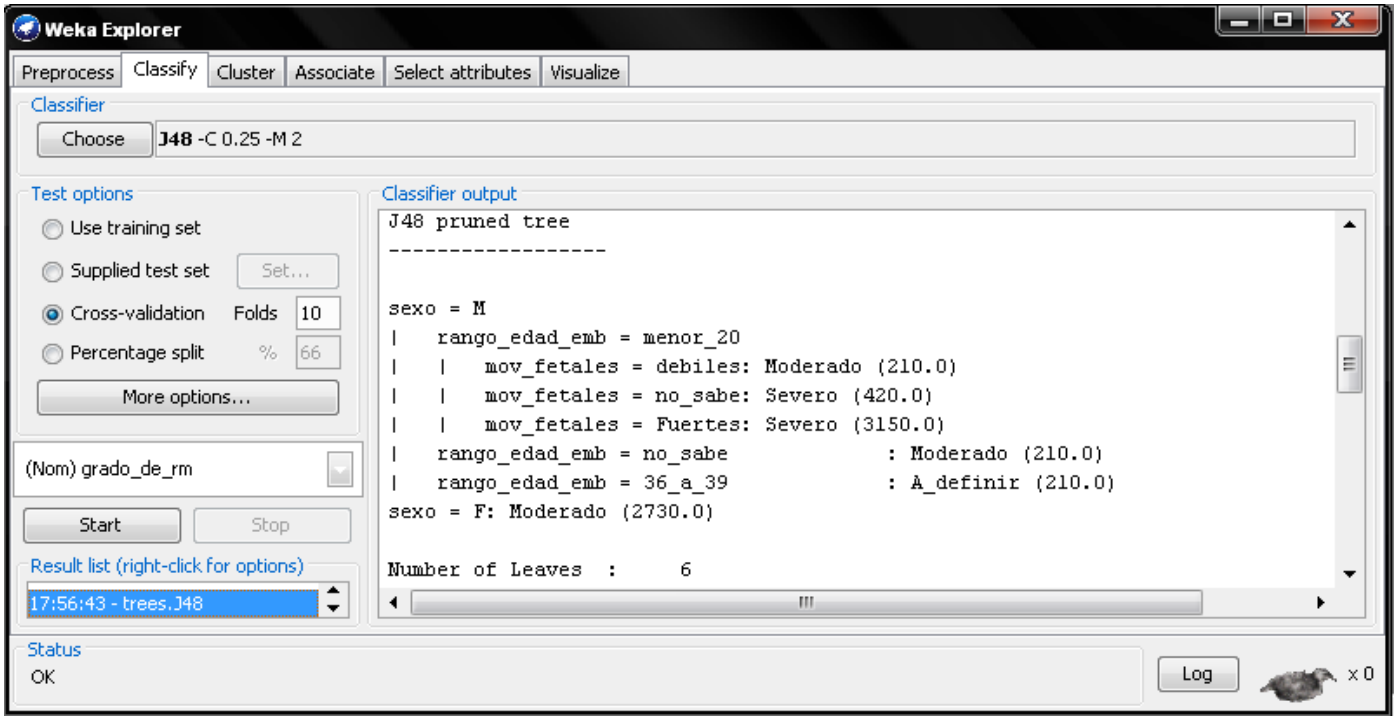

Fig. 28: Resultados obtenidos del algoritmo C4.5 en el Weka.

Presentación de los resultados obtenidos por ambas herramientas del algoritmo Decision Stump:

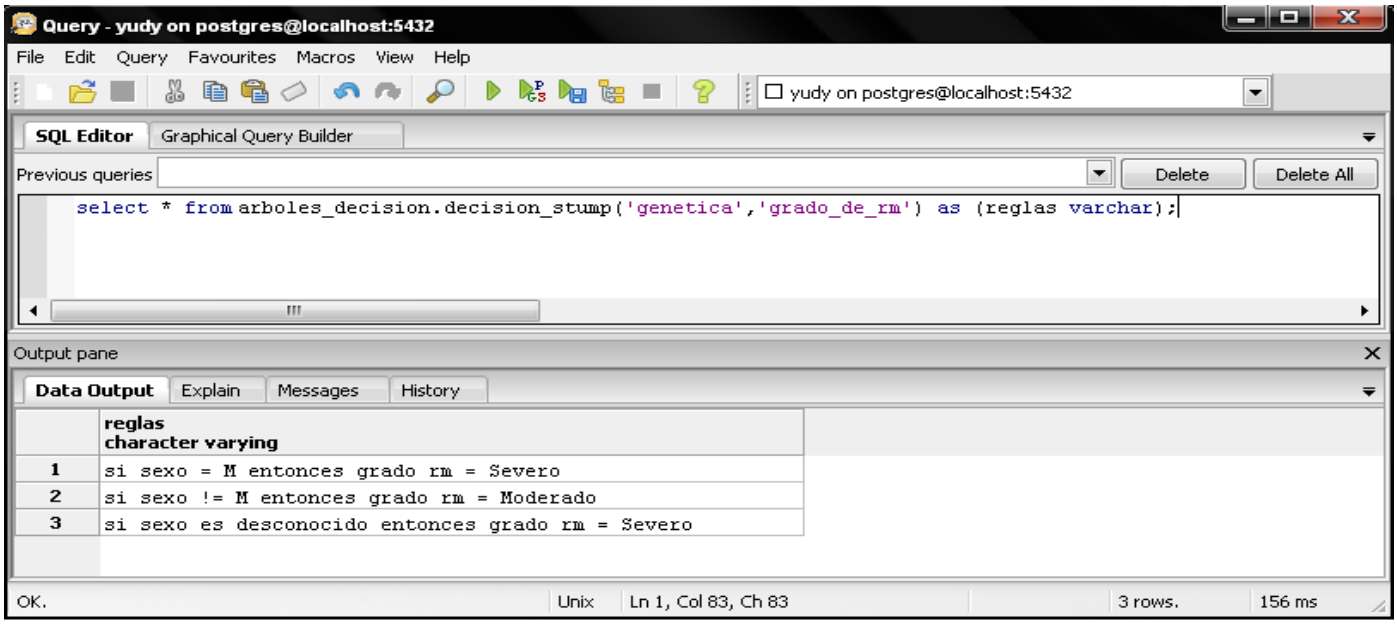

Fig. 29: Resultados obtenidos del algoritmo *Decision Stump* en el SGBD PostgreSQL 9.3.

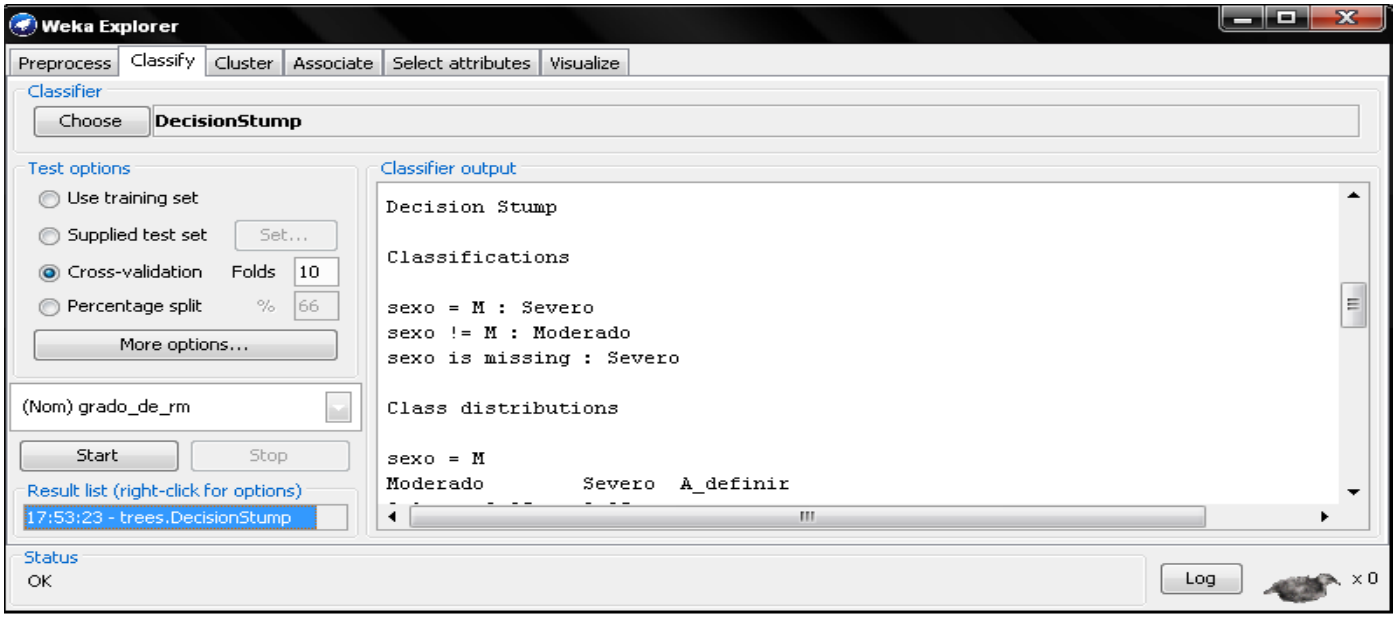

Fig. 30: Resultados obtenidos del algoritmo *Decision Stump* en el Weka.

Después de haber presentado los resultados obtenidos por ambas herramientas se procede a mostrar una comparación por indicadores entre las mismas.

Tabla 11: Comparación de los indicadores en PostgreSQL y Weka.

Indicadores	Extensión en PostgreSQL	Algoritmos en Weka
Facilidad de uso	• Requiere de conocimientos de base de datos para el análisis de los datos.	• Depende de especialistas para la manipulación de la herramienta además de conocimientos de base de datos para el análisis de los datos.
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, para posteriormente configurarlo. • Convertir los datos al formato arff en caso de no cargar los datos del SGBD.
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	• Los resultados se muestran claro	• Requiere de un especialista con

resultados	y completos, así facilitando el análisis de los especialistas. • Pueden ser entendibles por cualquier usuario.	conocimientos de la herramienta para realizar el análisis de los resultados obtenidos.
Recursos de Hardware	• No necesita de una computadora con elevadas prestaciones.	• Para grandes cantidades de datos requiere computadoras con altas prestaciones.

Como se pudo observar en las figuras 25, 26, 27, 28, 29 y 30, en ambas herramientas se obtuvieron la misma cantidad de reglas. Las reglas generadas por la extensión del PostgreSQL coinciden en su totalidad con las generadas por el Weka, lo cual tiene un nivel de aceptación por el cliente de un 100%. Además en lo que se refiere a los parámetros evaluados, el uso de esta extensión proporciona mejores ventajas para el análisis de los datos.

Evaluación e implementación

El objetivo del negocio correspondiente al descubrimiento de patrones ocultos en los datos; que permite clasificar el grado de discapacidad intelectual, basado en las relaciones que se establecen entre los atributos seleccionados fue cumplido, por lo que pueden considerarse los modelos como aceptados, desde el punto de vista analítico, para apoyar la toma de decisiones administrativas del Centro de Genética Médica.

Para la implementación del proyecto los directivos del Centro de Genética Médica son los encargados de emprender acciones y determinar, si así lo estiman conveniente, una estrategia a seguir, basada en la información descubierta por los algoritmos. Además se generará un informe con todo el proceso de minería, que servirá de apoyo o consulta para el proceso administrativo y la toma de decisiones.

Conclusiones Parciales

La validación mediante las pruebas de caja negra de los algoritmos propuestos integrados al SGBD PostgreSQL 9.3 permitió corregir los errores que podrían afectar el correcto funcionamiento de la extensión logrando la aceptación del cliente. La correcta aplicación del proceso de MD utilizando la metodología CRISP-DM sobre la Base de Datos de Genética Médica comprobó que los algoritmos implementados reflejan resultados similares a la herramienta Weka. Los resultados arrojados permitieron comprobar que la integración de los algoritmos al SGBD contribuye a un mejor análisis de los datos.

CONCLUSIONES GENERALES

Una vez concluida la investigación, la autora arriba a las siguientes conclusiones:

- Los métodos de investigación utilizados garantizaron analizar las técnicas de MD: árboles de decisión y de ella seleccionar los algoritmos a ID3, C4.5 y *Decision Stump*. Estos algoritmos generan reglas de decisión que por sus características son unas de las formas de representar más utilizadas por su fácil comprensión.

- El análisis de las herramientas que trabajan con las técnicas de MD permitió decidir que es necesario implementar los algoritmos de Árboles de Decisión para integrarlos al Gestor de Bases de Datos PostgreSQL 9.3 como mejor opción para realizar MD.

- Se creó una extensión para integrar los algoritmos implementados al SGBD PostgreSQL permitiendo aprovechar las potencialidades del gestor para el análisis de los datos.

- La validación de la extensión propuesta, mediante las pruebas de aceptación confirmó el correcto funcionamiento de los algoritmos implementados.

RECOMENDACIONES

Se recomienda:

- Continuar integrando algoritmos que realicen Minería de Datos al SGBD PostgreSQL con el objetivo de enriquecer el mismo y potenciar sus funcionalidades.

REFERENCIAS BIBLIOGRÁFICAS.

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

2. **Gulín, Jorge González.** Universidad de las Ciencias Informáticas. [En línea] 17 de Enero de 2013. [Citado el: 22 de Febrero de 2013.] http://www.uci.cu/la-uci-un-salto-historico-en-la-ciencia-video.

3. **Sánchez, José Cegarra.** *Los métodos de investigación.* s.l. : Ediciones Díaz de Santos, 2012. 8499693911/9788499693910.

4. *Usos y aplicaciones de la inteligencia artificial.* **Fernández, Luis Alberto García.** 3, 2010, Ciencia Hombre, Vol. 17.

5. *Aplicación del proceso de kdd en el contexto de bibliomining: El caso Elogim.* **Quiroz, Nohra Ledis.** 1, 2012, Vol. 35.

6. **Oded Maimon, Lior Rokach.** *Data Mining and Knowledge Discovery Handbook.* New York : Springer, 2010. ISBN 978-0-387-09823-4.

7. **Jaramillo Monsalve, Jorge Humberto y López Noreña, John Alexander.** *Datamining.* Medellín : s.n., 2010.

8. **Vallejos, Sofia J.** *Minería de Datos.* Corrientes, Argentina : s.n., 2006.

9. **Calderón, Montero, Alberto.** Información sobre HTA para pacientes y familiares. [En línea] [Citado el: 10 de Octubre de 2013.] http://www.medynet.com/hta/3.htm#1-2.

10. **José Manuel Molina López, Jesús García Herrero.** *Aplicaciones prácticas utilizando Microsoft Excel y WEKA.* 2010.

11. **Solarte Martinez, Guillermo Roberto y Soto Mejía, José A., Pereira,.** Árboles de decisiones en el diagnóstico de enfermedades cardiovasculares. Pereira, Colombia : s.n., 2011.

12. **Calancha Zuniga, Niefar Abgar, y otros.** *Breve aproximación a la Técnica de Árbol de Decisiones.* Cusco : s.n., 2010.

13. *Descubrimiento de conocimiento en lecciones aprendidas documentadas en los procesos de cierre de proyectos informáticos.* **Valentin, Eliana Bárbara Ril.** La Habana : s.n., 2013, Vol. 7.

14. **García Jiménez, María y Álvarez Sierra, Aránzazu.** *Análisis de Datos en WEKA – Pruebas de.* 2010.

15. *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.* **Aranda, Yadira Robles.** Habana : s.n., 2012, UCIENCIA.

16. **Garzon, Paula Andrea Vizcaino.** *Aplicación de técnicas de Inducción de Árboles de Decisión a problemas de Clasificación mediante el uso de WEKA.* Bogotá : s.n., 2011.

17. **University of Waikato.** Machine Learning Group at the University of Waikato. [En línea] [Citado el: 16 de Marzo de 2014.] http://www.cs.waikato.ac.nz/ml/weka/.

18. **Cordero Sánchez, Audrey.** *Extensión de Algoritmos de Reglas de Asociación para Minería de Datos en PostgreSQL.* La Habana, Cuba : s.n., 2013.

19. **Oracle Corporation.** Oracle Advanced Analytics. [En línea] [Citado el: 12 de Octubre de 2013.] http://www.oracle.com/es/products/database/options/advanced-analytics/index.html.

20. **P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth.** *CRISP-DM Step by step data mining guide.* 2009.

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

22. **Informática Aplicada a la Gestión Pública. Facultad Derecho UMU. Ingeniería del software. Metodologías de desarrollo. [En línea] [Citado el: 4 de Noviembre de 2013.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.**

23. Letelier, Patricio. *Metodologías Ágiles y XP.* Valencia : s.n., 2010.

24. Baumeister, Hubert y Marchesi, Michele. *Extreme Programming and Agile Processes in Software Engineering: 6th International Conference, XP 2005, Sheffield, UK, June 18-23, 2005, Proceedings Volumen 3556 de Lecture Notes in Computer Science / Programming and Software Engineering.* s.l. : Springer, 2005. ISBN: 3540262776, 9783540262770.

25. PostgreSQL Corporation. PostgreSQL en Español. [En línea] [Citado el: 20 de Octubre de 2013.] http://www.postgresql.org.es/sobre_postgresql.

26. Group, The PostgreSQL Global Development. *PostgreSQL 9.3.0 Documentation.* 2013.

27. Guía documentada para Ubuntu: PgAdminIII. Ubuntu. 2011.

28. Postgresql Global Development Group. *PostgreSQL 9.0 Official Documentation - Volume V. Internals and Appendixes.* s.l. : Fultus Corporation, 2011. ISBN 1596822503, 9781596822504.

29. Joskowicz, Ing. José. *Reglas y Prácticas en eXtreme Programming.* España : s.n., 2008.

30. Lisa Crispin, Tip House. *Testing Extreme Programming.* 2003. ISBN 0321113551, 9780321113559.

31. Ken Auer, Roy W. Miller. *Extreme Programming Applied: Playing to Win.* s.l. : ADDISON WESLEY Publishing Company Incorporated, 2002. 0201616408, 9780201616408.

32. Pressman, Roger S. ***Ingeniería del Software: Un enfoque práctico.*** **Séptima Edición. 2006.**

33. Teruel, Marcheco. ***Genetica comunitaria: la principal prioridad para la genetica medica en Cuba. Revista Cubana Genetica Comunitaria.*** **La Habana : s.n., 2010.**

34. Sarasa, Brito. ***Minería de Datos aplicada a la Gestión Docente del Instituto Superior Politécnico Jose Antonio Echeverría.*** **2008.**

35. Silveira Martineaux, Karina y Fernández Pérez, Reid. ***Comparación de algoritmos de clasificación y agrupamiento aplicando técnicas de minería de datos.*** **Habana : s.n., 2009.**

36. Polls, KDnuggets. Data mining methods. [En línea] Marzo de 2007. [Citado el: 21 de Febrero de 2014.] http://www.kdnuggets.com/polls/2007/data mining methods.htm.

37. Takeyas, Bruno López. Ingeniería en Sistemas Computacionales, Inteligencia Artificial. Nuevo Laredo, Tamaulipas : s.n., 2005.

38. Hidalgo, Israel Cueva. Aprobación de créditos bancarios. 2010.

39. ***Descubrimiento de conocimiento en lecciones aprendidas documentadas en los procesos de cierre de proyectos informáticos.*** **Ril Valentin, Eliana Bárbara, y otros. 3, La Habana : s.n., 2013, Vol. 7.**

40. F. Valenga, I. Perversi, E. Fernández, H. Merlino, D. Rodríguez, P. Britos y R. APLICACION DE MINERIA DE DATOS . Buenos Aires, Argentina : s.n.

41. ***Revista de Administracion Tributaria.*** **Verdi, Márcio Ferreira. 32, 2011. ISSN 1684-9434.**

42. Sutter, Herb y Alexandrescu, Andrei. ***C++ Coding Standards: 101 Rules, Guidelines, and Best Practices.C++ In-Depth Series.*** **2004.**

43. Ortiz Ocaña, Alexander. ***Temas pedagógicos, didácticos y metodológicos.*** **2006.**

44. J. J. Gutiérrez, M. J. Escalona, M. Mejías, J. Torres. ***Pruebas del Sistema en ProgramaciónExtrema.*** **Universidad de Sevilla.**

45. Garzón, Jesus María. ***Herramientas Case El mejor soporte para el proceso de desarrollo de software.*** **Instituto Nacional de Estadística e Informática. 1999. Colección Cultura Informática.**

46. Letelier, Patricio. ***Metodologías Ágiles y XP.*** **Valencia : s.n.**

47. Pressman, Roger S. ***Ingeniería del software: un enfoque práctico.*** **s.l. : Mikel Angoar, 1997.**

48. Datametrics, Equipo. ***Los mejores 10 algoritmos en minería de datos.*** **2013.**

49. Pressman, Roger S. ***Ingeniería del Software: Un enfoque práctico.*** **Séptima Edición.**

50. Waikato, University of. Machine Learning Group at the University of Waikato. [En línea] [Citado el: 16 de Marzo de 2014.] http://www.cs.waikato.ac.nz/ml/weka/.

BIBLIOGRAFÍA.

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

2. **Gulín, Jorge González.** Universidad de las Ciencias Informáticas. [En línea] 17 de Enero de 2013. [Citado el: 22 de Febrero de 2013.] http://www.uci.cu/la-uci-un-salto-historico-en-la-ciencia-video.

3. **Sánchez, José Cegarra.** *Los métodos de investigación.* s.l. : Ediciones Díaz de Santos, 2012. 8499693911/9788499693910.

4. *Usos y aplicaciones de la inteligencia artificial.* **Fernández, Luis Alberto García.** 3, 2010, Ciencia Hombre, Vol. 17.

5. *Aplicación del proceso de kdd en el contexto de bibliomining: El caso Elogim.* **Quiroz, Nohra Ledis.** 1, 2012, Vol. 35.

6. **Oded Maimon, Lior Rokach.** *Data Mining and Knowledge Discovery Handbook.* New York : Springer, 2010. ISBN 978-0-387-09823-4.

7. **Jaramillo Monsalve, Jorge Humberto y López Noreña, John Alexander.** *Datamining.* Medellín : s.n., 2010.

8. **Vallejos, Sofia J.** *Minería de Datos.* Corrientes, Argentina : s.n., 2006.

9. **Calderón, Montero, Alberto.** Información sobre HTA para pacientes y familiares. [En línea] [Citado el: 10 de Octubre de 2013.] http://www.medynet.com/hta/3.htm#1-2.

10. **José Manuel Molina López, Jesús García Herrero.** *Aplicaciones prácticas utilizando Microsoft Excel y WEKA.* 2010.

11. **Solarte Martinez, Guillermo Roberto y Soto Mejía, José A., Pereira,.** Árboles de decisiones en el diagnóstico de enfermedades cardiovasculares. Pereira, Colombia : s.n., 2011.

12. **Calancha Zuniga, Niefar Abgar, y otros.** *Breve aproximación a la Técnica de Árbol de Decisiones.* Cusco : s.n., 2010.

13. *Descubrimiento de conocimiento en lecciones aprendidas documentadas en los procesos de cierre de proyectos informáticos.* **Valentin, Eliana Bárbara Ril.** La Habana : s.n., 2013, Vol. 7.

14. **García Jiménez, María y Álvarez Sierra, Aránzazu.** *Análisis de Datos en WEKA – Pruebas de.* 2010.

15. *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.* **Aranda, Yadira Robles.** Habana : s.n., 2012, UCIENCIA.

16. **Garzon, Paula Andrea Vizcaino.** *Aplicación de técnicas de Inducción de Árboles de Decisión a problemas de Clasificación mediante el uso de WEKA.* Bogotá : s.n., 2011.

17. **University of Waikato.** Machine Learning Group at the University of Waikato. [En línea] [Citado el: 16 de Marzo de 2014.] http://www.cs.waikato.ac.nz/ml/weka/.

18. **Cordero Sánchez, Audrey.** *Extensión de Algoritmos de Reglas de Asociación para Minería de Datos en PostgreSQL.* La Habana, Cuba : s.n., 2013.

19. **Oracle Corporation.** Oracle Advanced Analytics. [En línea] [Citado el: 12 de Octubre de 2013.] http://www.oracle.com/es/products/database/options/advanced-analytics/index.html.

20. **P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth.** *CRISP-DM Step by step data mining guide.* 2009.

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

22. **Informática Aplicada a la Gestión Pública. Facultad Derecho UMU. Ingeniería del software. Metodologías de desarrollo. [En línea] [Citado el: 4 de Noviembre de 2013.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.**

23. Letelier, Patricio. *Metodologías Ágiles y XP.* Valencia : s.n., 2010.

24. Baumeister, Hubert y Marchesi, Michele. *Extreme Programming and Agile Processes in Software Engineering: 6th International Conference, XP 2005, Sheffield, UK, June 18-23, 2005, Proceedings Volumen 3556 de Lecture Notes in Computer Science / Programming and Software Engineering.* s.l. : Springer, 2005. ISBN: 3540262776, 9783540262770.

25. PostgreSQL Corporation. PostgreSQL en Español. [En línea] [Citado el: 20 de Octubre de 2013.] http://www.postgresql.org.es/sobre_postgresql.

26. Group, The PostgreSQL Global Development. *PostgreSQL 9.3.0 Documentation.* 2013.

27. Guía documentada para Ubuntu: PgAdminIII. Ubuntu. 2011.

28. Postgresql Global Development Group. *PostgreSQL 9.0 Official Documentation - Volume V. Internals and Appendixes.* s.l. : Fultus Corporation, 2011. ISBN 1596822503, 9781596822504.

29. Joskowicz, Ing. José. *Reglas y Prácticas en eXtreme Programming.* España : s.n., 2008.

30. Lisa Crispin, Tip House. *Testing Extreme Programming.* 2003. ISBN 0321113551, 9780321113559.

31. Ken Auer, Roy W. Miller. *Extreme Programming Applied: Playing to Win.* s.l. : ADDISON WESLEY Publishing Company Incorporated, 2002. 0201616408, 9780201616408.

32. Pressman, Roger S. ***Ingeniería del Software: Un enfoque práctico.*** **Séptima Edición. 2006.**

33. Teruel, Marcheco. ***Genetica comunitaria: la principal prioridad para la genetica medica en Cuba. Revista Cubana Genetica Comunitaria.*** **La Habana : s.n., 2010.**

34. Sarasa, Brito. ***Minería de Datos aplicada a la Gestión Docente del Instituto Superior Politécnico Jose Antonio Echeverría.*** **2008.**

35. Silveira Martineaux, Karina y Fernández Pérez, Reid. ***Comparación de algoritmos de clasificación y agrupamiento aplicando técnicas de minería de datos.*** **Habana : s.n., 2009.**

36. Polls, KDnuggets. Data mining methods. [En línea] Marzo de 2007. [Citado el: 21 de Febrero de 2014.] http://www.kdnuggets.com/polls/2007/data mining methods.htm.

37. Takeyas, Bruno López. Ingeniería en Sistemas Computacionales, Inteligencia Artificial. Nuevo Laredo, Tamaulipas : s.n., 2005.

38. Hidalgo, Israel Cueva. Aprobación de créditos bancarios. 2010.

39. ***Descubrimiento de conocimiento en lecciones aprendidas documentadas en los procesos de cierre de proyectos informáticos.*** **Ril Valentin, Eliana Bárbara, y otros. 3, La Habana : s.n., 2013, Vol. 7.**

40. F. Valenga, I. Perversi, E. Fernández, H. Merlino, D. Rodríguez, P. Britos y R. APLICACION DE MINERIA DE DATOS . Buenos Aires, Argentina : s.n.

41. ***Revista de Administracion Tributaria.*** **Verdi, Márcio Ferreira. 32, 2011. ISSN 1684-9434.**

42. Sutter, Herb y Alexandrescu, Andrei. ***C++ Coding Standards: 101 Rules, Guidelines, and Best Practices.C++ In-Depth Series.*** **2004.**

43. Ortiz Ocaña, Alexander. ***Temas pedagógicos, didácticos y metodológicos.*** **2006.**

44. J. J. Gutiérrez, M. J. Escalona, M. Mejías, J. Torres. ***Pruebas del Sistema en ProgramaciónExtrema.*** **Universidad de Sevilla.**

45. Garzón, Jesus María. ***Herramientas Case El mejor soporte para el proceso de desarrollo de software.*** **Instituto Nacional de Estadística e Informática. 1999. Colección Cultura Informática.**

46. Letelier, Patricio. ***Metodologías Ágiles y XP.*** **Valencia : s.n.**

47. Pressman, Roger S. ***Ingeniería del software: un enfoque práctico.*** **s.l. : Mikel Angoar, 1997.**

48. Datametrics, Equipo. ***Los mejores 10 algoritmos en minería de datos.*** **2013.**

49. Pressman, Roger S. ***Ingeniería del Software: Un enfoque práctico.*** **Séptima Edición.**

50. Waikato, University of. Machine Learning Group at the University of Waikato. [En línea] [Citado el: 16 de Marzo de 2014.] http://www.cs.waikato.ac.nz/ml/weka/.

ANEXOS

Anexo 1: Técnicas de minería de datos más empleadas.

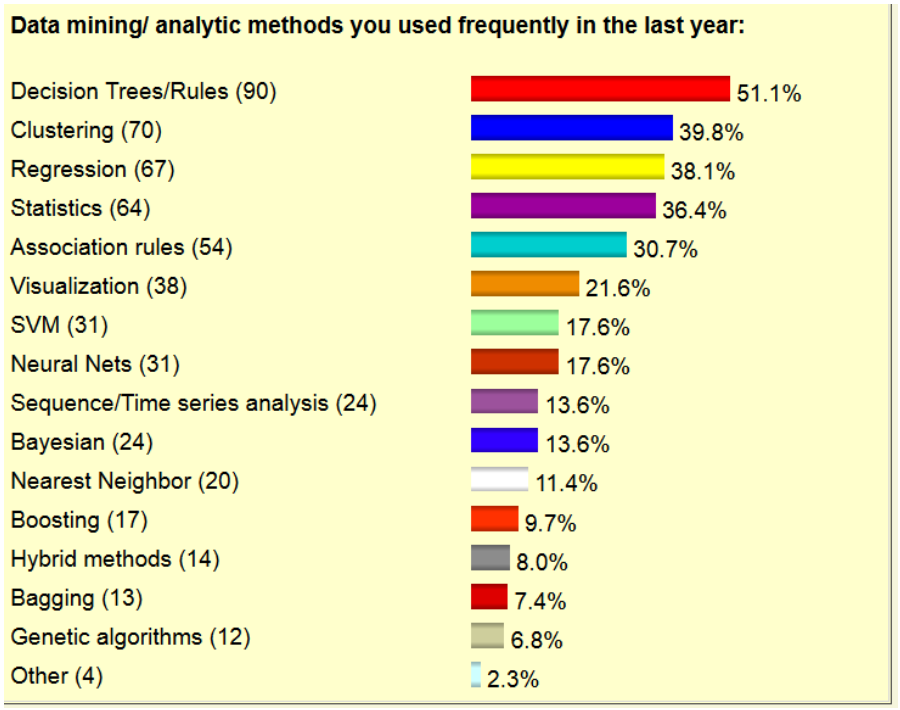

Fig. 31: Técnicas de minería de datos más empleadas. *(36)*