Desarrollo de herramientas para mineria de datos "UDMiner"

Development of tools for data mining "UDMiner"

Jorge Enrique Rodríguez Rodríguez[1]

Fecha de recepción: 8 de diciembre de 2011
Fecha de aceptación: 25 de enero de 2012

Resumen

En este artículo se muestra el informe final del proyecto de investigación "Desarrollo de Herramientas para Minería de Datos - UDMiner"; el cual tiene como objetivo principal la implementación de técnicas de Inteligencia Artificial y Estadísticas, en las tareas de: preprocesamiento, asociación, clasificación y agrupación de datos. Se presenta una breve descripción de cada una de las tareas mencionadas, se documentan las técnicas, y se plasman los resultados obtenidos.

Palabras clave: Minería de datos, agrupación de datos, reglas de asociación, clasificación de datos, preprocesamiento, redes neuronales artificiales, inteligencia artificial, estadística.

1 Magíster en Ingeniería de Sistemas. Especialista en Ingeniería de Software. Especialista en Diseño y Construcción de Soluciones Telemáticas. Ingeniero de Sistemas. Docente investigador de la Universidad Distrital Francisco José de Caldas. Director del Grupo de Investigación en Inteligencia Artificial de la misma Universidad. email: jrodri@udistrital.edu.co

Abstract

In this paper I show the final report of the research project, Development of Tools for Data Mining - UDMiner"; which has as main goal the implementation of methods of Artificial Intelligence and Statistical, in the tasks: data preprocessing, association, classification and clustering of data. A description is presented of each one of the mentioned tasks; the techniques are documented, and show the obtained results with conclusions.

Keywords: Data mining, clustering, association rules, data classification, preprocessing data, neural networks, artificial intelligence, statistical.

Introducción

Las primeras investigaciones sobre Minería de Datos se remontan aproximadamente a finales de la década de los 80's. Se impulso en gran parte por el desarrollo de áreas como la inteligencia artificial, el aprendizaje automático, las bases de datos relaciones y avances en la microelectrónica e informática.

Al hablar de Minería de Datos es necesario hacer referencia a las áreas con las cuales esta tiene relación; la estadística tradicional y el análisis de datos son algunos de estas. Los métodos estadísticos y el análisis sobre los datos, no proporcionan conocimiento como tal, debido a esto, fue necesario fomentar una práctica más profunda, para utilizar los datos y extraer beneficios de estos. La respuesta a estas necesidades y a muchas otras, como el almacenamiento de gran cantidad de datos y la necesidad de herramientas adecuadas e innovadoras que apoyen la toma de decisiones, está reflejada en una de las áreas de investigación más recientes, la Minería de Datos. A continuación, se dan algunas definiciones de Minería de Datos:

La Minería de Datos es la exploración de forma automática o semiautomática de grandes cantidades de datos para el descubrimiento de reglas y patrones [1].

La Minería de Datos es la búsqueda para nueva y valiosa información no trivial en grades volúmenes de datos [9].

La Minería de Datos puede definirse como un proceso iterativo de detección y extracción de patrones a partir de grandes bases de datos: esto es modelo-reconocimiento [11].

La Minería de Datos es el análisis de un conjunto de datos para encontrar relaciones desconocidas y resumir los datos de nuevas formas entendibles para el minero [4].

En la práctica, los modelos para extraer patrones pueden ser de dos tipo: predictivos y descriptivos. Los modelos predictivos pretenden estimar valores futuros o desconocidos de variables de interés, que se denominan variables objetivo o dependientes, usando otras variables o campos de la base de datos, llamadas variables independientes o predictivas. Los

modelos descriptivos en cambio identifican patrones que explican o resumen los datos, es decir sirven para explorar las propiedades de los datos examinados, no para predecir nuevos datos [6]. En UDMiner, se desarrollan las tareas de clasificación, asociación y agrupación, y también se incluye la fase de preprocesamiento de datos. En la figura 1 se muestra la arquitectura planteada.

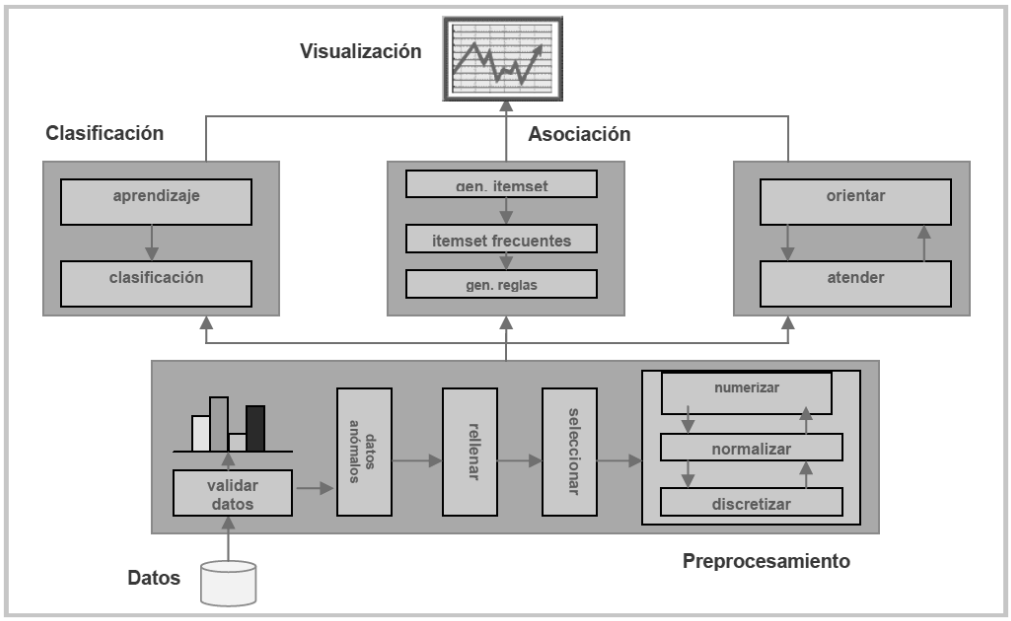
2. Tareas de minería de datos

En este apartado se describen de forma general las tareas de minería de datos realizadas por UDMiner.

2.1 Preprocesamiento de datos

Previo ha aplicar minería de datos, es necesario realizar una preparación de estos, comúnmente a esta preparación se le denomina preprocesamiento de datos. El propósito fundamental del preprocesamiento de datos es manipular y transformar cada conjunto de datos haciendo que la información contenida dentro de ellos sea más accesible y coherente [14]. La recopilación de los datos de diferentes fuentes y aun de una sola, implica la ejecución de una limpieza exhaustiva de los datos para un buen análisis, que en ocasiones se convierte en una tarea bastante tediosa debido a que se pueden tener muchas inconsistencias en los datos que impide un buen aprendizaje de los mismos, estas inconsistencias se verán reflejadas a la hora de tomar decisiones. El preprocesamiento cumple un papel fundamental en todo el proceso de la Minería de Datos. La tarea de extraer patrones o conocimiento útil y veraz, está estrechamente relacionada con la condición de los datos; es decir, los datos son fiables cuando tienen cierto grado calidad y significado, siendo esta la función del preprocesamiento: transformar los datos de forma tal que posean condiciones aceptables para mejorar el proceso de Minería de Datos.

Figura 1. Arquitectura de UDMiner



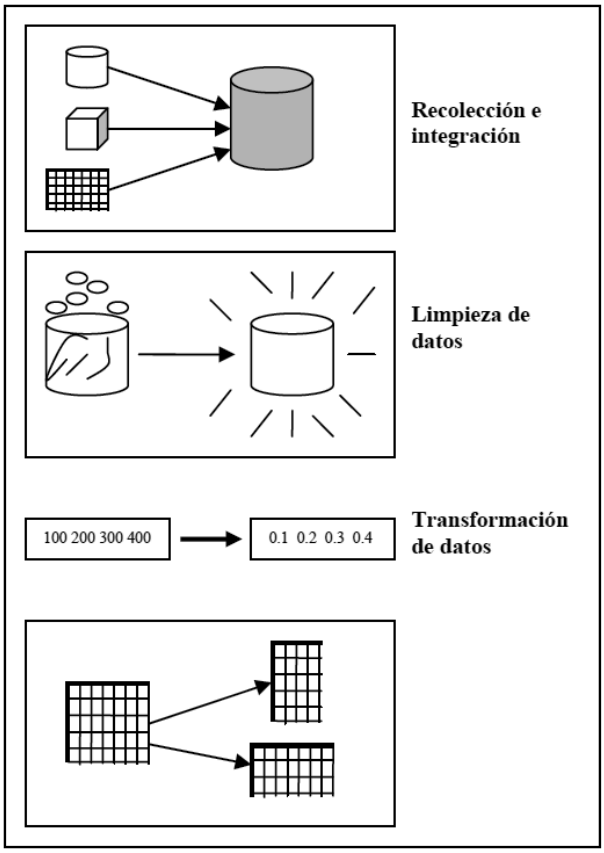
El preprocesamiento puede incluir las siguientes tareas (figura 2): recolección e integración, limpieza de los datos, transformación, y reducción de los datos (selección de atributos, selección de instancias, discretización).

Las tareas a nivel de preprocesamiento de datos, implementadas en UDMiner son: detección de valores anómalos (outliers), relleno de valores faltantes, selección de atributos, discretización, normalización y numerización.

a) Detección de valores anómalos

Los datos anómalos son aquellos valores que están por encima de los datos normales pero que estadísticamente son correctos, es decir, que son datos tomados de la realidad. Suele suceder que un valor erróneo caiga en la normalidad y por consiguiente no puedan ser detectados, para está situación existen varios métodos estadísticos encargados de determinar si un dato es anómalo o no, y es el minero de datos el que determina si esté se toma en cuenta. Los valores anómalos suelen deberse a errores en el procedimiento a la hora de introducir los datos o de codificarlos. Una vez detectados los casos atípicos el analista debe saber elegir con criterio entre eliminarlos del análisis o evaluar toda la información incluyéndolos [13]. Una técnica utilizada en la detección de valores anómalos es el de los k-vecinos que consiste en establecer una distancia y ver los valores con mayor distancia media entre el resto de los valores.

Figura 2. Tareas de la fase de preprocesamiento de datos



b) Relleno de valores faltantes

Dentro de la integración de los datos se pueden presentar diversos inconvenientes que afectan a la calidad de los mismos. Con el fin de solucionar este problema es indispensable contar con un procedimiento adecuado, para ello se consideran los posibles inconvenientes que los datos pueden presentar, uno de estos es la existencia de valores faltantes. Los valores faltantes en un conjunto de datos pueden presentar inconsistencias al momento de crear modelos o patrones para la Minería de Datos. En la mayoría de los casos los valores faltantes, perdidos o ausentes (missing values), son reemplazados por varias razones; bien sea porque el método de Minería de Datos a implementar no haga un buen tratamiento de estos o se quiera realizar una mejor vista minable, y estos no permitan agregar correctamente.

Los valores faltantes se presentan por varios factores, como: la recopilación, ya que cuando se realiza está de diferentes bases de datos frecuentemente se hace la unión pero no la integración. Otro caso puede ser los valores no existentes, por ejemplo; un paciente no tiene registro de accidentes. Una vez identificados los valores faltantes se procede a su solución.

c) Selección de atributos

Comúnmente las bases de datos contienen millones de registros y miles de atributos (variables), es poco probable que todas las variables sean independientes sin una estructura de correlación entre los datos [10]. Del mismo modo, las bases de datos pueden contener atributos irrelevantes para la fase de Minería de Datos, o también puede existir redundancia de las variables. Por ejemplo, si la tarea es clasificar los clientes en compradores y no compradores de CD's de música popular, atributos tales como, el número telefónico y el número de hijos pueden ser irrelevantes, mientras que la edad y el nivel académico pueden ser características o atributos relevantes para clasificar los clientes.

La selección de atributos relevantes es el proceso más importante dentro del preprocesamiento, ya que es la etapa en la cual se dejan los atributos más significativos dentro del conjunto de datos. En algunos casos la selección de atributos se aplica por prueba y error hasta conseguir un modelo o un patrón más eficiente, pero esto no es lo más adecuado ya que el tiempo y el costo computacional crece con respecto al número de variables existentes en la colección de datos. Con la selección de atributos se busca dejar únicamente los atributos con los cuales se pueda llega a realizar una predicción o descripción lo más exacta posible.

d) Discretización de datos

La discretización consiste en transformar un atributo numérico y representarlo como un atributo categórico. Un atributo numérico se representa en nominal por medio de intervalos o bins; a la discretización, también se le conoce como "binning". Un bin o intervalo es un grupo que representa un valor de atributo, hablando ya de atributos nominales que acoge a los datos numéricos que se encuentran dentro de sus límites.

e) Numerización de datos

La tarea de numerización es el proceso inverso a la discretización, se transforman los atributos categóricos en numéricos. Este caso no es tan común, pero existen situaciones en el que se convierte en un proceso útil, como por ejemplo que el método de Minería de Datos no acepte datos categóricos.

f) Normalización de datos

Una transformación muy útil empleada en la Minería de Datos es la denominada normalización. Esta se realiza con dos fines: - establecer una relación de equidad entre los atributos; esto quiere decir que todos estén representados en una misma escala, - representar los datos en una escala menor, con el propósito de satisfacer los requerimientos de las técnicas utilizadas para minar.

2.2 Clasificación de datos

La clasificación de datos es un proceso de dos pasos (figura 3). En el primer paso, se construye un modelo, el cual describa el conjunto preliminar de clases. El modelo es construido analizando los registros ejemplos. Cada registro pertenece a una clase específica conocida, debido a esto, esta técnica de clasificación se conoce como aprendizaje supervisado. En contraste con el aprendizaje no supervisado (también conocido como clustering), en el cual la clase a la que pertenece cada registro es desconocida, y el número de clases por aprender tampoco puede ser conocido.

Generalmente, el modelo aprendido es representado en la forma de reglas de clasificación, árboles de decisión, o fórmulas matemáticas.

El primer paso a seguir, consiste en estimar la precisión del modelo o clasificador. La precisión de un modelo en un conjunto dado de datos es el porcentaje de ejemplos, del conjunto de entrenamiento, que fueron correctamente clasificados. Si la precisión del modelo es considerada aceptable, el modelo puede ser usado para clasificar futuros conjuntos de datos para los cuales la etiqueta de clase es desconocida.

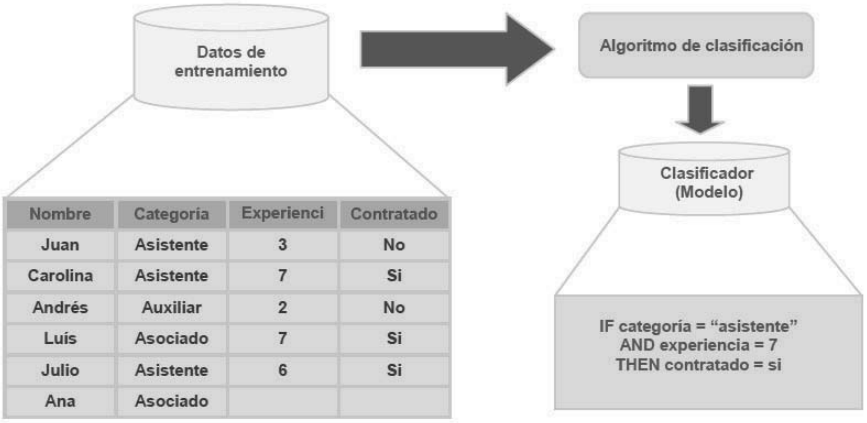
2.3 Agrupación de datos (clustering)

El proceso de agrupar una colección de objetos físicos y abstractos dentro de clases de objetos similares es llamado agrupamiento (clustering). Un grupo (en adelante, se utilizará el término cluster, para referirse a grupo) es una colección de datos que son similares a otros dentro de un mismo grupo y son diferentes a los objetos en otros grupos. Un cluster de datos puede ser tratado como un cluster en muchas aplicaciones. El análisis de clusters es un conjunto de metodologías para clasificación automática de muestras entre un número de grupos usando medidas de asociación, es decir, las muestras en un grupo son similares, y las pertenecientes a otros grupos son diferentes [9].

En el segundo paso (Figura 4), el modelo es usado para generar la clasificación de datos desconocidos.

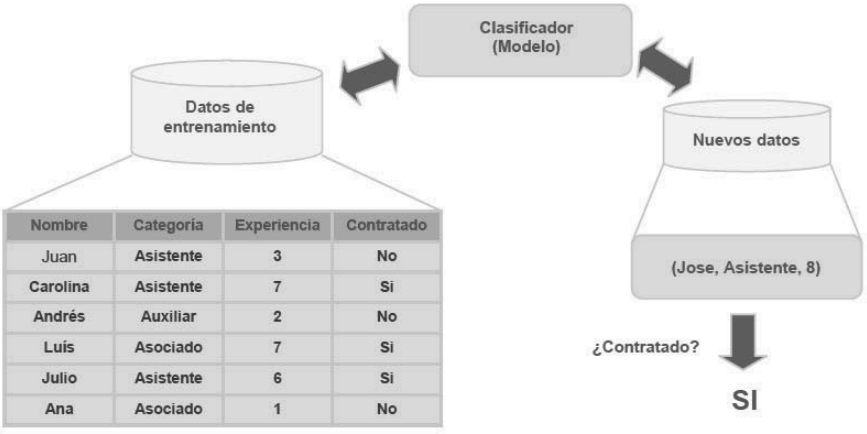
El análisis de cluster, también llamado segmentación de datos, tiene una gran variedad de metas. Todas referidas a una colección de grupos o segmentos de objetos entre subconjuntos o "clusters", tal que, los clusters de un mismo objeto están estrechamente relacionados entre sí, y difieren notablemente con clusters de otros objetos. Un objeto puede ser descrito por un conjunto de medidas, o por sus relaciones con otros objetos [5].

Figura 3. Fase de aprendizaje [2]



Nombre	Categoría	Experienci	Contratado
Juan	Asistente	3	No
Carolina	Asistente	7	Si
Andrés	Auxiliar	2	No
Luis	Asociado	7	Si
Julio	Asistente	6	Si
Ana	Asociado		

Figura 4. Clasificación / Predicción [2]



Nombre	Categoría	Experiencia	Contratado
Juan	Asistente	3	No
Carolina	Asistente	7	Si
Andrés	Auxiliar	2	No
Luis	Asociado	7	Si
Julio	Asistente	6	Si
Ana	Asociado	1	No

En Minería de Datos, los esfuerzos se han enfocado en encontrar métodos para un eficiente y efectivo análisis cluster en las grandes bases de datos. Activos temas de investigación se han enfocado en la escalabilidad de los métodos de agrupación, la mayor eficiencia de los métodos para la agrupación de formas complejas y tipos de datos, técnicas de agrupación de alta dimensionalidad, y métodos de agrupación para datos mixtos numéricos y categóricos en grandes bases de datos.

2.4 Asociación de datos

La tarea de asociación se lleva a cabo, a través de las reglas de asociación, las cuales son una de las mejores técnicas de Minería de Datos y tal vez, la más común en el descubrimiento de patrones en sistemas de aprendizaje no supervisado [9]. El aprendizaje de reglas de asociación se basa en su confidencia y soporte, los algoritmos de aprendizaje trabajan en la búsqueda de reglas que satisfagan los parámetros de mínimo soporte y confidencia.

Dado el gran volumen de datos de los problemas con el que los algoritmos de aprendizaje de reglas de asociación trabajan, la tarea de buscar patrones que cumplan estos requisitos puede parecer muy costosa computacionalmente, ya que el análisis de los conjuntos de ítems crece notablemente con respecto al número de variables de los datos. Sin embargo, en los casos reales existen pocos conjuntos frecuentes y los métodos que existen con una confidencia mínima se benefician de este hecho. Por ejemplo, la mayoría de los clientes de un supermercado suelen comprar un número bastante limitado de productos.

La mayoría de las técnicas de asociación necesitan una métrica adecuada para poder extraer el grado de dependencia que existe entre las variables asociadas a un conjunto de datos. Hay trabajos donde se ha evaluado y comparado diferentes medidas de reglas de asociación, unas que provienen del campo de la estadística y otras definidas de forma específica en Minería de Datos para la evaluación de reglas obtenidas en un entorno educativo mediante algoritmos de programación genética en gramáticas [7].

3. Técnicas implementadas

En el desarrollo de UDMiner se seleccionó e implementó diferentes técnicas de aprendizaje computacional y de estadística para cada una de las tareas de minería antes mencionadas. A continuación, se dará una introducción a cada técnica.

3.1 Preprocesamiento de datos

a) Detección de valores anómalos

Para la detección de valores anómalos se empleó una técnica basada en una aproximación estadística (ecuación 1), con esta técnica, es asumir la distribución a priori de los datos. La ecuación 1 establece un umbral máximo y mínimo utilizando la media y la desviación estándar de los datos. Cualquier valor que sobrepase estos umbrales es considerado anómalo.

$$umbral = media \pm 3 \times desviacion \qquad (1)$$

b) Relleno de valores faltantes

El primer paso para rellenar valores faltantes en UDMiner es construir la red bayesiana, en este caso el clasificador Naïve Bayes. La independencia entre atributos puede ser algo arriesgado, pero se ha demostrado la efectividad de este clasificador. La idea del Naïve Bayes radica en tener una estructura fija, en donde el único nodo padre es la clase (que se tiene que conocer), y sus hijos son los demás atributos (atributos independientes). Es decir, aprender los parámetros (distribuciones de probabilidad) de la siguiente forma: de acuerdo a la hipótesis de independencia que asume este clasificador la tabla de probabilidad $P(A_1, A_2, ..., A_n | c)$, donde la A representa los atributos, y c representa la clase, se factoriza y quedan n tablas de probabilidad, una para cada atributo de la forma $P(A_1 | c)$. De tal manera que hay que estimar la tabla de probabilidad para cada atributo y la distribución a priori de la variable clase P(c). Como se trabajan dos tipos de atributos, nominales y numéricos, existen diferentes métodos para estimar las distribuciones.

Para el caso de atributos categóricos se emplea la ley de sucesión de La Place (ver ecuación 2) y para atributos número se emplea una función de distribución de probabilidad[2] (tabla 1).

2 Se recomienda utilizar una herramienta estadística, como por ejemplo SPSS, con el fin de determinar la distribución de los datos.

$$P(x_i \mid Pa(x_i)) = \frac{n(x_i, Pa(x_i)) + 1}{n(Pa(x_i)) + \mid \Omega x_i \mid} \qquad (2)$$

Tabla 1. Funciones de distribución de probabilidad [12].

Nombre del parámetro	Función de distribución	Parámetro
Normal	$f(x) = \frac{1}{\sqrt{(2\pi)}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2} \right]$	$-\infty < \mu < \infty$ $\sigma > 0$
Exponencial	$f(x) = \lambda e^{-\lambda x} \mathrm{I}_{(0,\infty)}(x)$	$\lambda > 0$
Gamma	$f(x) = \frac{\lambda^r}{\Gamma(r)} x^{r-1} e^{-\lambda x} \mathrm{I}_{(0,\infty)}(x)$	$\lambda > 0$ $r > 0$
Beta	$f(x) = \frac{1}{\mathrm{B}(a,b)} x^{a-1} (1-x)^{b-1} \mathrm{I}_{(0,1)}(x)$	$a > 0$ $b > 0$
Cauchy	$f(x) = \frac{1}{\pi\beta\left\{1 + \left[(x-\alpha)/\beta\right]^2\right\}}$	$-\infty < \mu < \infty$ $\beta > 0$
Lognormal	$f(x) = \frac{1}{x\sqrt{2\pi}\sigma} \exp\left[-\frac{(\log_e x - \mu)^2}{2\sigma^2} \right] \mathrm{I}_{(0,\infty)}(x)$	$-\infty < \mu < \infty$ $\sigma > 0$

Una vez construida la red se hace uso del algoritmo EM (Maximización de la Esperanza). La implementación de este algoritmo tiene dos fases: una etapa de inicialización y una etapa iterativa. En la etapa de inicialización se provee al algoritmo de las distribuciones de probabilidad condicional que se encuentran en la red[3], la segunda etapa es donde se maximiza el valor esperado hasta su convergencia. Una vez inicializados los parámetros e inicializado el contador de etapas, se procede a la fase iterativa que se da hasta su convergencia.

c) Selección de atributos

Un clasificador usado en la Minería de Datos, para la selección de atributos, son los árboles de decisión. Un árbol de decisión es una representación en el que cada conjunto de posibles conclusiones se establece implícitamente mediante una lista de muestras de clase conocida [20].

Los pasos fundamentales para seleccionar atributos con árboles de decisión son tres, generar en primera instancia el árbol, luego a partir de este generar las reglas y posteriormente seleccionar los atributos observando los que más se utilizan en las reglas.

La idea de construir árboles de decisión tiene que ver en gran medida en construirlos de forma eficiente pero corta, es decir que genere el árbol de decisión más pequeño posible. La teoría de la información proporciona una fórmula para medir el desorden total en una base de datos, se utilizó la fórmula descrita en la ecuación 3, aunque no garantice que ayudará a construir el árbol más pequeño posible.

3 La red utilizada para UDMiner es una red bayesiana tipo Naïve, en la cual se asume independencia entre los atributos del conjunto de datos, pero estos tienen dependencia con una clase.

$$desorden = \sum_b \left(\frac{n_b}{n_t} \right) \times \left(\sum_c - \frac{n_b}{n_b} \log_2 \frac{n_b}{n_b} \right) \quad (3)$$

d) Discretización de datos

La técnica empleada se denomina "simple binning", en esta técnica se establece un número de bins estáticos en los cuales se discretizará, y los intervalos se calculan utilizando información de los datos como el máximo y el mínimo (ver ecuación 4).

$$Intervalo = \frac{Maximo - Minimo}{\#bins} \quad (4)$$

e) Numerización

La solución que se plantea para transformar un atributo categórico en uno numérico es convertirlo por medio de variables Dummy. El trabajar con variables Dummy implica construir un nuevo atributo para cada valor posible, es decir, si el atributo categórico "estatura" tiene como posibles valores {alto, medio, bajo} habrá un nuevo atributo para cada uno de estos valores. Otro aspecto importante es la forma de presentarlos, cuando en la instancia original el atributo estatura contiene por ejemplo un "alto", entonces en el nuevo atributo construido correspondiente con ese valor tendrá en esa instancia un 1, que representa que ese es el valor original. Los otros dos atributos tendrán en esa instancia un valor de 0 que representan que el dato original no corresponde con ese valor.

f) Normalización de datos

La técnica más frecuente y utilizada para normalizar datos se denomina Normalización Lineal Uniforme. Con esta técnica es posible representar cualquier atributo numérico en una escala deseada, por lo general entre cero y uno. La fórmula utilizada la mostrada en la ecuación 5.

$$v' = \frac{(v - \min)}{(\max - \min)} (new\_\max - new\_\min) + new\_\min \quad (5)$$

3.2 Clasificación de datos

Para la clasificación de datos se utiliza una red neuronal artificial de tipo feedforward, el algoritmo empleado es el de retropropagación, el cual se resume en seis (6) pasos, a modo de síntesis junto con las fórmulas utilizadas por el algoritmo [8].

Paso 1
Inicializar los pesos (conexiones) de la red con valores pequeños aleatorios[4].

Paso 2
Presentar un patrón de entrada y especificar la salida deseada que debe generar la red.

Paso 3
Calcular la salida actual de la red, para esto se presentan las entradas a la red y se calcula la salida que presenta cada capa hasta llegar a la capa de salida, esta será la salida de la red. Lo anterior se logra realizando los siguientes subpasos:

a. Se calculan las entradas netas para las neuronas ocultas procedentes de las neuronas de entrada.

b. Para una neurona *j* oculta:

4 El intervalo de los valores de los pesos generalmente es una medida empírica, para este proyecto se tomará un intervalo entre -1 y 1.

$$net_{pj}^{h} = \sum_{i=1}^{N} w_{ji}^{h} x_{pi} + \Theta_{j}^{h} \tag{6}$$

en donde el índice h se refiere a magnitudes de la capa oculta; el subíndice p, al p-ésimo vector de entrenamiento, y j a la j-ésima neurona oculta. El símbolo Q puede ser opcional, dado que actúa como una entrada más.

c. Se calculan las salidas de las neurona ocultas:

$$y_{pj} = f_{j}^{h}(net_{pj}^{h}) \tag{7}$$

d. Se realizan los mismos cálculos para obtener las salidas de las neuronas de salida

$$net_{pk}^{o} = \sum_{j=1}^{L} w_{kj}^{o} y_{pj} + \Theta_{k}^{o} \tag{8}$$

$$y_{pk} = f_{k}^{o}(net_{pk}^{o}) \tag{9}$$

Paso 4
Calcular los términos de error para todas las neuronas.

Si la neurona k es una neurona de la capa de salida, el valor de la delta es:

$$\delta_{pk}^{o} = (d_{pk} - y_{pk}) f_{k}^{o}(net_{pk}^{o}) \tag{10}$$

La función f debe ser derivable, para lo cual se utilizará la función sigmoidal definida por la ecuación 13:

$$f_{k}(net_{k}) = \frac{1}{1 + e^{-net_{k}}} \tag{11}$$

Los términos de error para las neuronas de salida quedan:

$$\delta_{pk}^{o} = (d_{pk} - y_{pk}) y_{pk} (1 - y_{pk}) \tag{12}$$

Si la neurona j no es de salida, entonces la derivada parcial del error no puede ser evaluada directamente. Por tanto, se obtiene el desarrollo a partir de valores que son conocidos y otros que pueden ser evaluados. La expresión obtenida en este caso es:

$$\delta_{pj}^{h} = f_{j}^{b}(net_{pj}^{h}) \sum_{k} \delta_{pk}^{o} w_{kj}^{o} \tag{13}$$

Donde se observa que el error en las capas ocultas depende de todos los términos de la capa de salida. De aquí surge el término de Backpropagation. En particular, para la función sigmoidal:

$$\delta_{pj}^{h} = x_{Ypj}(1 - x_{Ypj}) \sum_{k} \delta_{pk}^{o} w_{kj}^{o} \tag{14}$$

Donde k se refiere a todas las neuronas de la capa superior a la de la neurona j.

Paso 5
Actualización de pesos

Para los pesos de las neuronas de la capa de salida:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \Delta w_{kj}^{o}(t+1);$$
$$\Delta w_{kj}^{o}(t+1) = \alpha \delta_{pk}^{o} y_{pj} \tag{15}$$

Y para los pesos de las neuronas de la capa oculta:

$$w_{ji}^{h}(t+1) = w_{j}^{h}(t) + \Delta w_{j}^{h}(t+1);$$
$$\Delta w_{j}^{h}(t+1) = \alpha \delta_{j}^{h} x_{ip} \quad (16)$$

Paso 6
El proceso se repite hasta que el término de error

$$E_{p} = \frac{1}{2}\sum_{k=1}^{M} \delta_{jp}^{2} \quad (17)$$

resulta aceptablemente pequeño para cada uno de los patrones aprendidos.

3.3 Agrupación de datos

Como técnica para agrupar datos se seleccionó la red neuronal artificial ART2. Una red ART2 básicamente consta de dos capas entre las que se establecen conexiones hacia adelante y hacia atrás (feedforward/feedback). La estructura general de una red ART2 se muestra en la figura 5.1.

A continuación se hará un resumen del funcionamiento de la red ART2 [19].

La capa F1 se encuentra dividida en seis subcapas, w, x, u, v, p y q. Todos los nodos que están marcados con una G son unidades de control de ganancia, que envían una señal inhibitoria no específica a todas las unidades de la capa a la que llegan. Todas las subcapas de F1 así como la capa r del subsistema de orientación, tiene el mismo número de unidades. Las subcapas individuales de F1 están conectadas de unidad a unidad; esto es, las capas no están completamente interconectadas, con la excepción de las conexiones ascendentes que llegan a F2 y de las conexiones descendentes de F2.

El subsistema de orientación es el responsable de detectar falta de coincidencia entre las tramas ascendentes y descendentes de la capa F1. Esta utiliza para determinar la coincidencia una magnitud que recibe el nombre de parámetro de vigilancia y suele identificarse mediante el símbolo p. El valor del parámetro de vigilancia mide hasta que grado discrimina el sistema entre distintas clases de tramas de entrada.

El control de ganancia se utiliza cuando se implementa una red ART2 donde la capa F2 podría recibir entradas de otra capa por encima de ella (dentro de una jerarquía de redes pertenecientes a un sistema mayor), así como de la capa F1 que esta situada más abajo. Este control impide que una trama que entre por encima de la capa F2, se cruce o se compare con otra trama que a entrado al mismo tiempo por la capa F1.

El subsistema de atención esta compuesto por las dos capas de elementos de procesamiento, F1 y F2, y un sistema de control de ganancia.

3.4 Asociación de datos

Para la generación de reglas de asociación, se empleó el algoritmo a priori, el cual busca ítemsets frecuentes usando generación de candidatos. Su nombre se debe a que usa conocimiento a priori para la generación de ítemsets frecuentes. Este algoritmo se resume en dos pasos [3]:

I. Generación de todos los ítemsets que contienen un solo elemento, utilización de estos para generar ítemsets que contengan dos elementos, y así sucesivamente. Se toman todos los posibles pares

de ítems que cumplen con las medidas mínimas de soporte inicialmente preestablecidas; esto permite ir eliminando posibles combinaciones: aquellas que no cumplan con los requerimientos de soporte no entrarán en el análisis.

II. Generación de las reglas revisando que cumplan con el criterio mínimo de confianza. Es interesante observar que si una conjunción de consecuentes de una regla cumple con los niveles mínimos de soporte y confianza, sus subconjuntos (consecuentes) también los cumplen; en el caso contrario, si algún ítem no los cumple no tiene caso considerar sus superconjuntos. En la tabla 2 se muestra el algoritmo a priori.

Tabla 2. Algoritmo a priori

Algoritmo: A priori. Encuentra los ítemsets frecuentes. (Fuente tomada y adaptada de HAN, 2001)
Entradas: Transacciones de una base de datos D; min_sop
Salidas: L, ítemsets frecuentes de la base de datos D.
Método:
(1) L_1 = encontrar_1-ítemsets_frecuente(D);
(2) Para (k=2; $L_{k-1}\neq\emptyset$; k++){
(3) C_k = generar_apriori(L_{k-1}, min_sop);
(4)Para cada transacción t ∈ D{ // examinar D para el contador
(5)C_t = subconjuntos(C_{k-t}) // obtener los subconjuntos t que son candidatos
(6) Para cada candidato c ∈ C_t
(7) c.contador++
(8) fin-para
(9)L_k = {c ∈ C_k | c.contador ≥ min_sop}
(10) fin-para
(11) retornar L = $U_k L_k$;

función generar_apriori(L_{k-1} : (k-1)-ítemsets frecuentes; min_sop : mínimo soporte)
(1) Para cada ítemset $l_1 \in L_{k-1}$
(2)Para cada ítemset $l_2 \in L_{k-1}$
(3)Si ($l_1[1] = l_2[1] \wedge (l_1[2] = l_2[2]) \wedge \ldots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$)
Entonces{
(4)c = l_1 X l_2; //generar candidatos
(5) Si subconjunto_infrecuente(c, L_{k-1}) Entonces
(6) eliminar c;
(7) Si No
(8) adicionar c a C_k
(9) fin-para
(10) retornar C_k;

función subconjunto_infrecuente(c : k-ítemset candidato; L_{k-1} : (k-1)-ítemset frecuente)
(1) Para cada (k-1)-subconjunto s de c
(2) Si ($s \notin L_{k-1}$) Entonces
(3) retornar Verdadero;
(4) retornar Falso;

4. Análisis de pruebas y resultados

Para validar los resultados obtenidos por UDMiner, se presentaron diferentes conjuntos de datos tomados de http://www.ics.uci.edu/~mlearn/MLRepository.html.

4.1 Preprocesamiento de datos [15]

a) Relleno de valores faltantes

De los tres ejemplos seleccionados para realizar las pruebas, dos contienen valores faltantes, Soybean y Census Soybean contiene exactamente 2337 valores faltantes, distribuidos en todos los atributos. Census tiene 4262, distribuidos en solo tres atributos. Por razones de espacio en el artículo, solo se relacionan los valores de relleno para el ejemplo Census (Tabla 3), ya que los valores faltantes en Soybean están distribuidos en todos los atributos y en 5 de las 19 clases que contiene.

La Tabla 3 muestra con que valor se rellenaron los datos faltantes dependiendo su clase. Por ejemplo, para el atributo "workclass" el valor de relleno fue "private" en ambas clases, mientras que en el atributo "Ocupation" existen diferentes valores para cada clase. Se tiene que realizar la distinción entre clases debido a que el Algoritmo EM es una técnica supervisada, es decir, su proceso de aprendizaje esta basado en muestras de la clase conocida.

Tabla 3. Relleno de valores faltantes en Census

Atributo			
Clase	woksclass	Occupation	Native-country
>50K	Private	exec-managerial	United Status
<=50K	Private	adm-clerical	United Status

El atributo "Native-country" en un 90% tiene el valor "United States", por lo tanto este será el valor de relleno. Lo anterior, podría establecerse trivialmente, ya que el algoritmo EM trabaja con probabilidades, es decir que existe un 90% de probabilidad de que el valor de relleno sea "United States", debido a que es el que más se presenta.

b) Selección de atributos

Para la selección de atributos se empleó los conjuntos de datos Soybean, Chess y Census, a continuación se describe la selección hecha a cada conjunto de datos, producto de la construcción de un árbol de decisión:

- Soybean: este conjunto de datos muestra un comportamiento en el que se selecciona un bajo porcentaje de atributos a mayor porcentaje de selección, y el máximo de atributos seleccionados no excede el 70%. La explicación se de a que el conjunto de datos no cubre el total de los atributos, es decir, este conjunto cuenta con 35 atributos y 683 instancias, por consiguiente, las 683 instancias no son suficientes para cubrir todas las posibles combinaciones que se pueden presentar. A este problema se le ha llamado *la maldición de la dimensionalidad* [6], y debido a este, los patrones que posteriormente se extraen en la etapa de Minería de Datos pueden ser incorrectos o poco útiles, ya que no tienen la cantidad de datos en donde apoyarse para tomar determinada forma.

- Chess: el comportamiento que presenta estos datos es totalmente opuesto al conjunto de datos Soybean, pues en este se cubre la mayor cantidad de espacio de los atributos. Lo que significa que la mayoría de datos son relevantes y solo se empieza a eliminar con un porcentaje de

selección alto. Este conjunto de datos teóricamente provee todos los datos necesarios para crear modelos robustos y útiles que puedan extraer patrones y conocimiento relevante.

- Census: el ejemplo presenta un comportamiento equilibrado debido en parte a que son datos extraídos de bases de datos reales y con una cantidad considerable de instancias. Este seria el conjunto de datos escogido para realizar selección y posteriormente Minería de Datos, ya que en la practica, formar un conjunto de datos que cubra todo el espacio de dimensiones es bastante costoso computacionalmente, a la hora de generar un modelo a partir de ellos, teniendo en cuenta el volumen creciente de las bases de datos.

- Por otro lado, La complejidad computacional del algoritmo EM, esta dada por el número de iteraciones requeridas para que se de la convergencia [4]. En las pruebas hechas la convergencia se logró en promedio entre la quinta y décima iteración, para cada uno de los ejemplos. El ejemplo Census, contiene considerablemente mayor cantidad de instancias que Soybean; sin embargo, el tiempo empleado en el relleno es relativamente bajo comparándolo con este. Esto se debe a que construir la red bayesiana esta estrechamente relacionado al número de clases existentes, por lo tanto, se equilibran los tiempos, dado que Soybean tiene 19 clases, mientras que Census tiene solo 2.

- Se podría pensar que a mayor número de instancias, mayor tiempo se emplearía en la selección. Si se toma en cuenta por ejemplo el conjunto de datos Chess, este tarda más tiempo que los otros; aunque, el conjunto de datos Census posee más instancias y más atributos, este tarda aproximadamente el 28% del tiempo empleado para la selección en Chess. Esto se presente, pues el número de clases del ejemplo Chess es más alto que el del ejemplo Census; el primero contiene 18 clases y el segundo contiene solo dos, lo que implica que en la construcción del árbol exista más desorden por el alto número de clases.

- Otro factor que puede explicar esta situación, es cuando el total de instancias cubre todo el espacio de dimensiones, el árbol generado es más equilibrado con respecto a su profundidad y amplitud, lo que obviamente implica más tiempo en su generación por el número de ramificaciones y nodos creados.

4.2 Clasificación de datos [16]

En la tabla 4, se muestra una síntesis de los resultados obtenidos por UDMiner (la efectividad de clasificación mostrada por UDMiner, corresponde al promedio de 10 corridas) frente a los obtenidos por WEKA (Entorno para el análisis de conocimiento de Waikato)[5]. En esta se observa que la efectividad de clasificación de UDMiner, en general es buena; en los ejemplos mostrados se obtuvo una clasificación correcta por encima del 70% (8 de los diez ejemplos), solo en uno (chess) la clasificación es definitivamente deficiente, esto se debe a la incompletitud de los datos. Con WEKA se obtuvo una clasificación del más del 70%, en cuatro de los diez ejemplos.

Por otro lado, para cada ejemplo se probó con diferentes estructuras de redes neuronales, variando el número de neuronas en la

5 Para la clasificación de datos con WEKA, se probó con el perceptrón multicapa.

capa oculta, el número de capas ocultas, la tasa de aprendizaje, el momentum, el valor inicial de los pesos (aleatoriamente), hasta identificar la estructura de red que generaba menor error. Agregar más de una capa oculta puede llevar a una mejora del desempeño de la red, aunque no es significativo frente al incremento de la complejidad y el tiempo empleado en la clasificación. Incrementar el número de neuronas en la capa oculta mejora el desempeño de la red hasta cierto grado, luego disminuye su desempeño.

Otro aspecto a analizar es la inclusión de diferentes heurísticas para determinar la tasa de aprendizaje (a) durante el entrenamiento de la red; para tal fin se utilizó el conjunto de datos SOYBEAN, y se logró establecer que la mejor heurística para el valor de la tasa de aprendizaje, consiste en disminuir la tasa proporcionalmente luego de haberse superado el 50% de las épocas en el entrenamiento.

Tabla 4. Efectividad de entrenamiento y prueba (clasificación) de UDMiner y WEKA[6]

Conjuntos de datos	UDMiner		WEKA	
	Entrenamiento	Test	Entrenamiento	Test
Soybean	98.22%	82.15%	97.39%	79.38%
Agaricus	99.68%	99.69%	98.98%	*
Chess	98.42%	3.98%	*	*
House	95.68%	84.11%	?	?
Nourse	100%	80.62%	85%	85.16%
ADN	99.96%	75.82%	73.35%	69.84%
Shuttle	99.76%	80.41%	?	?
Tic-Tac	100%	78.53%	100%	92.91%
Bolsa	97.42%	83.73%	86%	78%
Connect_4	79.64%	57.74%	?	?

4.3 Agrupación de datos [17]

Se hace una comparación del modelo implementado frente al modelo de mapas auto-organizativos de Kohonen y el algoritmo EM (Expectation Maximization), para este último se utilizó la herramienta WEKA, ver tabla 5.

En cuanto a la configuración de la red ART2, luego de realizar las respectivas pruebas con cada uno de los conjuntos de datos, se obtiene: - el valor medio para el parámetro de vigilancia es de 0.4, y - para la constate e es 0.2, para las demás constantes no se pudo establecer un valor medio. En términos generales la efectividad de agrupación de la red ART2 es superior a los mapas auto-organizativos de kohonen y al algoritmo EM, obteniendo una efectividad promedio del 69.12%.

6 En la tabla 4 se utilizan dos caracteres especiales: - asterisco (*) indica que WEKA terminó el proceso de clasificación pero no se logró obtener ningún resultado, - interrogación (?) nunca terminó de procesar.

Tabla 5. Efectividad de la agrupación

Conjuntos de datos	Algoritmo EM		Mapas auto-organizativos de Kohonen		ART2	
	Efectividad de Agrupación	Tiempo	Efectividad de agrupación	Tiempo	Efectividad de Agrupación	Tiempo
Soybean	73%	181 seg	57%	100 seg	94.7%	11 seg
Contact-lences	33%	1 seg	0%	1 seg	0%	1 seg
Led7_data	100%	4 seg	80%	8 seg	70%	3 seg
Vehicle_data	0%	75 seg	75%	15 seg	100%	6 seg
SatImage	83.3%	110 seg	83%	290 seg	50%	102 seg
Wine_data	100%	2 seg	67%	2 seg	100%	2 seg
Promedio de la agrupación	64.88%		60.33%		69.12%	

Existen casos particulares en los cuales no se logra obtener una buena agrupación, tal es el caso del conjunto de datos CONTACT-LENCES, en el cual el porcentaje de agrupación es del 0%; la razón está en que para este ejemplo no se logró determinar una configuración aceptable para la red ART2, siendo esta una limitante del modelo implementado.

Por otro lado, se observa que la calidad de agrupación obtenida con ART2 del conjunto de datos SATIMAGE es inferior a la obtenida con los otros dos modelos, alcanzándose tal solo un 50% de agrupación. La explicación a lo anterior radica en que el funcionamiento de la red ART2 tiende a degradarse cuando el volumen de instancias es alto (La degradación del funcionamiento se puede contrarrestar con una buena configuración de la red).

4.4 Asociación de datos [18]

Para medir la efectividad y el tiempo de respuesta del algoritmo para asociación de datos implementado en UDMiner, se comparó con las herramientas WEKA y CBA (Asociación Basada en Clasificación). Una de las diferentes pruebas se realizó con el archivo ASSOCSAM, en el que se almacenan datos de aleaciones (color, textura, compuestos químicos, …). Este archivo tiene 39 atributos y 898 patrones. Se probó con un soporte del 90% y una confidencia del 90%. En este ejemplo se observa que el comportamiento (figuras 5, 6, 7) de las 3 herramientas es igual en cuanto a la generación de ítemsets frecuentes y reglas, salvo las reglas generadas por CBA.

Las 3 herramientas (UDMiner, WEKA y CBA) utilizan para asociación el algoritmo A priori, el cual se basa en la generación de ítemsets frecuentes, condicionado por las medidas de soporte y confidencia. Sin embargo WEKA, utiliza otras medidas, como son: una constante delta, un parámetro de convicción, mínimo y máximo contador de soporte, y permite definir el número de reglas a generar. Por lo anterior, en algunos ejemplos los resultados, difieren en cierta medida con respecto a los resultados generados por UDAssociate.

Figura 5. Gráfica de tiempo para el archivo ASSOCSAM

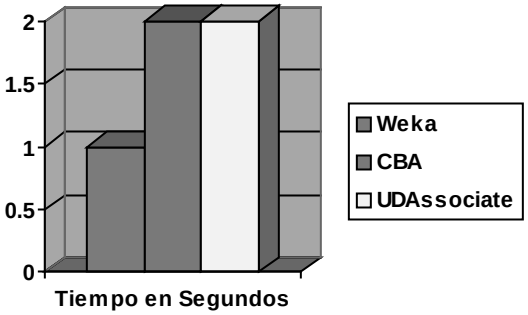


Figura 6. Gráfica de ítemsets frecuentes para el archivo ASSOCSAM

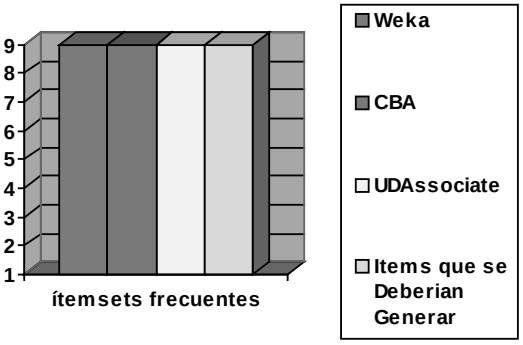
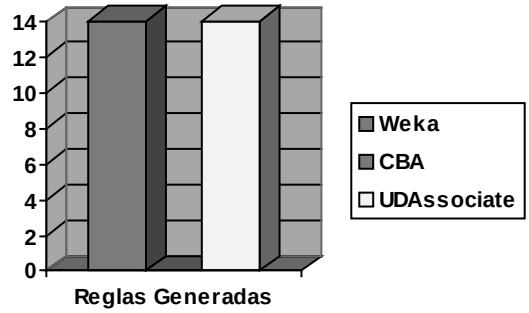


Figura 7. Gráfica de regla para el archivo ASSOCSAM



Para más información consultar [18].

5. Conclusiones

Se estableció un flujo básico operacional para realizar preprocesamiento de datos, que consiste en primera medida en tratar datos anómalos y faltantes, posteriormente seleccionar atributos o reducir dimensionalidad, y por último transformar los datos de forma adecuada para la utilización de diferentes modelos de Minería.

En cuanto a la técnica seleccionada para la clasificación de datos, se destaca, entre otras, la parsimonia de esta, puesto que mediante una red neuronal se puede abordar tanto un problema de clasificación como un problema de regresión, mientras que desde la perspectiva estadística clásica se han necesitado dos modelos tan diferentes como el análisis discriminante (en cuanto a clasificación se refiere) y las series de tiempo (para el caso de predicción/regresión). Lo anterior, se corrobora en las pruebas, donde se observa que el algoritmo backpropagation es una buena elección, cuando de clasificar/predecir datos se trata.

La red neuronal ART2 es una técnica viable para la agrupación de datos, dada su alta efectividad; sin embargo, la configuración del parámetro de vigilancia debe hacerse utilizando heurísticas, o en el peor de los casos a través de la experimentación, lo cual implica que si el parámetro empleado no es adecuado, la efectividad de agrupación es baja.

6. Trabajos futuros

Se tiene previsto analizar seleccionar e implementar más técnicas en cada una de las fases de preprocesamiento y minería de datos. Para la fase de preprocesamiento se analizarán algoritmos para la reducción de la dimensionalidad, relleno de valores faltantes, detección de valores anómalos y algoritmos

de transformación; del mismo modo, se contemplarán diferentes fuentes de datos (URL, Bases de Datos y Hojas de Cálculo). En cuanto a la tarea de clasificación se espera implementar algunos métodos bayesianos con el fin de medir su efectividad y complejidad computacional. Para la tarea de agrupación se analizará e implementará el algoritmo GTM (The Generative Topographic Mapping). Por otro lado, se utilizará lógica difusa con el fin de generar reglas de asociación. Por último, se analizarán y utilizarán algoritmos para minar datos sobre la WEB.

Bibliografía

[1] BERRY, M and LINOFF, G. Data Mining Techniques. John Wiley & Sons, Inc, USA: 1997, p. 5.

[2] GONZÁLEZ, L., y RODRÍGUEZ, J. Prototipo de software para la clasificación de datos mediante el método bayesiano TAN - UDTan". Revista Vínculos - Universidad Distrital Francisco José de Caldas No.1 ISSN 1794-211X de la Facultad Tecnológica. II semestre de 2006 Volumen 3 Número 1.

[3] HAN, J., y KAMBER, M. Data Mining: Concepts and Techniques. Morgan Kaufmann Publisher, USA: 2001, p.230

[4] HAND, D., MANNILA, H., and SMYTH, P. Principles of Data Mining. The MIT Press, USA: 2001, pp. 1.

[5] HASTIE, T., TIBSHIRANI, R., and FRIEDMAN, J. The Elements of Statistical Learning. Springer, Canadá: 2001, p. 453.

[6] HERNÁNDEZ, J., RAMÍREZ, M., Y FERRI, C. Introducción a la Minería de Datos. Prentice Hall, España: 2004, pp. 12.

[7] HERVAS-MARTÍNEZ, C., ROMERO, C. y VENTURA, S. Comparación de medidas de evaluación de reglas de asociación. Departamento de Informática y Análisis Numérico, Universidad de Córdoba, España: 2004. p.6.

[8] HILERA, J. y MARTÍNEZ, V. Redes Neuronales Artificiales "Fundamentos, Modelos y Aplicaciones". Addison Wesley Iberoamericana, USA: 1995. p. 138 – 142.

[9] KANTARDZIC, Mehmed. Data Mining: concepts, models, methods, and algorithms. Wiley – Interscience, USA: 2001, p. 2, 117, 165.

[10] LAROSE, D. Data Mining: Methods and Models. Wiley-Interscience, USA: 2006. p. 1-2

[11] MENA, J. Data Mining Your Website. Digital Press, USA: 1999, pp. 5.

[12] MOOD, A., GRAYBILL, F., and BOES, D. Introduction to the theory of statistics. McGraw Hill, USA: 1974. p. 540-541

[13] PÉREZ, C. Técnicas de análisis multivariante de datos. Aplicaciones con SPSS. Editorial Pearson Prentice-Hall, España: 2004. p. 39-40.

[14] PYLE, D. Data Preparation for Data Mining. Morgan Kaufmann, USA: 1999. p. 90-95

[15] RODRÍGUEZ, J., CORREA, J., y BARRERA, H. Prototipo de software para el preprocesamiento de datos - UDClear. IV Simposio Internacional de Sistemas de Información e Ingeniería de Software en la Sociedad del Conocimiento, libro de actas volumen 1, ISBN: 84-690-0258-9.

[16] RODRÍGUEZ, J. Software para la clasificación/predicción de datos. Revista TECNURA – Universidad Distrital Francisco José de Caldas ISSN 0123921X de la Facultad Tecnológica. AÑO 11 No. 21.

[17] RODRÍGUEZ, J. Red neuronal artificial para la agrupación de datos. Revista Científica. Centro de Investigaciones y Desarrollo Científico – Universidad

Distrital Francisco José de Caldas. ISSN 0124-2253 No. 9.

[18] RODRÍGUEZ, J., RODRÍGUEZ, M., y AMAYA, A. Prototipo de software para la asociación de datos - UDAssociate. Revista Científica. Centro de Investigaciones y Desarrollo Científico – Universidad Distrital Francisco José de Caldas. ISSN 0124-2253 No. 7.

[19] SKAPURA, D. Y FREEMAN J. "Redes neuronales, algoritmos, aplicaciones y técnicas de programación". España: Diaz de Santos, 1993; p.335-343

[20] WINSTON, P. Inteligencia artificial. Addison Wesley, USA: 1994. p. 457