TariyKDD: una herramienta genérica de Descubrimiento de Conocimiento débilmente acoplada con un SGDB

Ricardo Timarán Pereira, Ph. D.*, Andrés O. Calderón Romero**, Iván Ramírez Freyre**, Fernando Guevara**, Juan Carlos Alvarado**

**Universidad de Nariño, Facultad de Ingeniería, Departamento de Sistemas,*
Ciudad Universitaria Torobajo
San Juan de Pasto, Colombia

***Fundación Parque Tecnológico de Software de Pasto – ParqueSoft Pasto*
Cra 43 No. 18A - 130
San Juan de Pasto, Colombia

ritimar@udenar.edu.co, acalderon@parquesoftpasto.com, ivan.ramirezf@gmail.com, guevrim@mail.udenar.edu.co, endimeon777@gmail.com

Resumen

En este artículo se presenta la primera versión de TariyKDD, una herramienta genérica para el Descubrimiento de Conocimiento, débilmente acoplada con un SGBD. Esta herramienta fue desarrollada en el laboratorio de KDD del departamento de Ingeniería de Sistemas de la Universidad de Nariño (Colombia). TariyKDD, esta compuesta por cuatro módulos: el módulo de conexión que permite la recuperación de datos desde archivos planos y bases de datos relacionales, el módulo de utilidades con clases y librerías comunes, el módulo kernel donde se encuentran los filtros que permiten realizar los procesos de limpieza y transformación de datos, los algoritmos de minería de datos para las tareas de Asociación y Clasificación y los programas de visualización de datos, y el módulo de interfaz gráfica de usuario que facilita la interacción del usuario con la herramienta de una manera amigable. En TariyKDD se encuentran implementados los algoritmos Apriori, FPGrowth y EquipAsso para la tarea de Asociación y los algoritmos C4.5 y Mate-tree para la tarea de Clasificación.

Palabras Claves: *Minería de Datos, Algoritmos para Asociación y Clasificación, Herramientas de Minería de Datos, Software Libre.*

Abstract

In this paper the first version of TariyKDD, a generic tool for Knowledge Discovery in Databases loosely coupled with a SGBD, is presented. This tool was developed in the KDD laboratory of the Systems Engineering Department of the University of Nariño (Colombia). TariyKDD is composing by four modules: the connection module that allows to the data retrieval from plain files and relational databases, the utilities module with common classes and libraries, the kernel module where are the filters that allow to make data cleaning and data transformation processes, the data mining algorithms for the Association and Classification tasks and the data visualization programs, and the graphical user interface module that facilitates the user interaction with the tool of a friendly way. In TariyKDD are implemented the Apriori, FPGrowth and EquipAsso algorithms for the Association task and the C4.5 and Mate-tree algorithms for the Classification.

Keywords: Data Mining, Algorithms for Association and Classification Tasks, Data Mining Tools, Free Software.

1. Introducción

El Descubrimiento de Conocimiento en Bases de Datos (DCBD) es básicamente un proceso automático en el que se combinan desarrollo y análisis. El proceso consiste en extraer patrones en forma de reglas o funciones, a partir de los datos, para que el usuario los analice. Esta tarea implica generalmente preprocesar los datos, hacer minería de datos y presentar resultados [3] [9].

Una herramienta DCBD debe integrar una variedad de componentes (técnicas de minería de datos, consultas, métodos de visualización, interfaces, etc.), que juntos puedan eficientemente identificar y extraer patrones interesantes y útiles de los datos almacenados en las bases de datos. Las arquitecturas de estas herramientas se pueden ubicar en una de tres tipos:

sistemas débilmente acoplados, medianamente acoplados y fuertemente acoplados con un Sistema Gestor de Bases de Datos (SGBD) [17]. Por otra parte, de acuerdo a las tareas que desarrollen, las herramientas DCBD se clasifican en tres grupos: herramientas genéricas de tareas sencillas, herramientas genéricas de tareas múltiples y herramientas de dominio específico [12].

Algunas herramientas como Alice, C5.0 RuleQuest, Qyield, CoverStory ofrecen soporte únicamente en la etapa de minería de datos y requieren un pre y un post procesamiento de los datos. Hay una gran cantidad de este tipo de herramientas en [10], especialmente para clasificación apoyadas en árboles de decisión, redes neuronales y aprendizaje basado en ejemplos. El usuario de este tipo de programas puede integrarlos a otros módulos como parte de una aplicación completa [12]. Otros ofrecen soporte en más de una etapa del proceso de DCBD y una variedad de tareas de descubrimiento, típicamente, combinando clasificación, visualización, consulta y clustering, entre otras [12]. En este grupo están Clementine, DBMiner, Data Mine, IMACS, Intelligent Miner, Quest, entre otras. Una evaluación de un gran número de herramientas de este tipo se puede encontrar en [6]. Todas estas herramientas necesitan de la adquisición de costosas licencias para su utilización.

Por otra parte, existen algunas herramientas libres de minería de datos, desarrolladas por universidades y centros de investigación, tales como *Weka*[26], *ADaM*[1], *Orange* [4], *Tanagra* [15], *AlphaMiner* [5], *Yale* [11], que en su gran mayoría no son acopladas con un SGBD. Estas ofrecen diferentes formatos de archivos planos como ARFF, CSV, entre otros, para obtener los conjuntos de datos ha minar. Existen pocos algoritmos de minería de datos implementados, el más común para la tarea de Asociación es Apriori y para Clasificación el C4.5.

En este artículo se presenta la primera versión de *TariyKDD*, una herramienta genérica para el Descubrimiento de Conocimiento, débilmente acoplada con un SGBD, desarrollada en el laboratorio KDD del departamento de Ingeniería de Sistemas de la Universidad de Nariño (Colombia), bajo software libre. Esta herramienta esta compuesta por cuatro módulos: el módulo de *conexión* que permite la recuperación de datos desde archivos planos y bases de datos relacionales, el módulo de *utilidades* con clases y librerías comunes, el módulo *kernel* donde se encuentran los filtros que permiten realizar los procesos de limpieza y transformación de datos, los algoritmos de minería de datos para las tareas de Asociación y Clasificación y los programas de visualización de datos, y el módulo de *interfaz gráfica de usuario* que facilita la interacción del usuario con la herramienta de una manera amigable. En *TariyKDD* se encuentran implementados los algoritmos *Apriori* [2], *FPGrowth* [7] [8] *y EquipAsso* [19] [20] para la tarea de Asociación y los algoritmos *C4.5* [13] [14] *y Mate-tree* [21] para la tarea de Clasificación.

EquipAsso [19] [20][22] es un algoritmo, para el cálculo de los *itemsets* frecuentes basado en dos operadores del álgebra relacional para Asociación: *Associator* y *EquiKeep* [18] e implementado en el lenguaje SQL mediante las primitivas SQL *Associator Range* y *EquiKeep On* [18]. *Mate-tree* [21] es un algoritmo para la tarea de clasificación basado en el operador algebraico relacional *Mate* [23][24] que conjuntamente con los operadores agregados *Entro* y *Gain* [23][24] facilitan el cálculo de la *Ganancia de Información* y con el operador algebraico relacional *Describe Classifier* [23][24], la construcción del árbol de decisión.

El resto del artículo está organiza en secciones. En la sección 2, se describe la arquitectura de la herramienta TariyKDD y sus diferentes módulos. En la sección 3, se abordan los aspectos de la implementación de la herramienta. En la sección 4, se muestran las pruebas de funcionalidad de los algoritmos de Asociación y Clasificación implementados en TariyKDD. Finalmente, en la sección 5, se presentan las conclusiones y futuros trabajos.

2. Arquitectura de TariyKDD

La arquitectura de TariyKDD la componen cuatro módulos, el de *conexión,* el de *utilidades*, el *kernel* y la *interfaz gráfica de usuario*, que juntos soportan las etapas de DCBD: selección de datos, preprocesamiento de datos, minería de datos y visualización. La arquitectura de esta herramienta se muestra en la Figura 1.

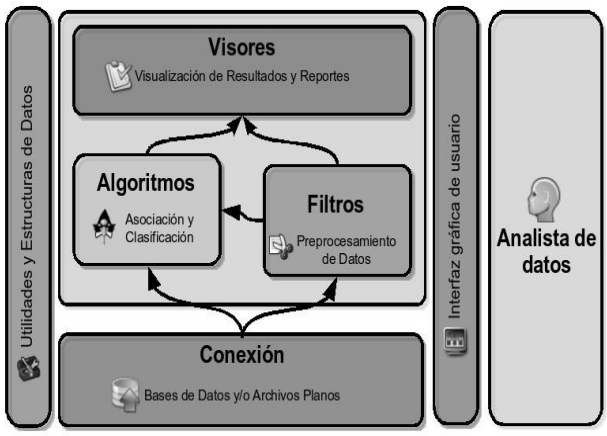

Figura 1. Arquitectura de TariyKDD

2.1. Módulo de conexión

Este módulo es el encargado de la conexión a una fuente de datos, esta puede ser una base de datos o un archivo plano. El objetivo de este módulo es proveer al usuario de herramientas amigables para la selección y

construcción de una vista o conjunto de datos minable. TariyKDD soporta las siguientes fuentes de datos:

- *DB:* A través de controladores JDBC y el diseño de una interfaz amigable se puede obtener conjuntos de datos a partir de bases de datos construidas con diferentes SGBD tales como PostgreSQL, MySQL y Oracle. Los controladores para la conexión a estos SGBD se incorporan dentro de la distribución libre de esta herramienta.
- *File:* Se dio soporte a formatos de archivos planos en disco como ARFF y CSV. Este último formato es el utilizado por el módulo de filtros para almacenar los cambios que se aplican al conjunto de datos durante el proceso de limpieza y transformación. El conjunto resultante puede ser almacenado en disco y recuperado de nuevo a través de esta opción.

La Figura 2 muestra la interfaz gráfica de conexión de esta herramienta.

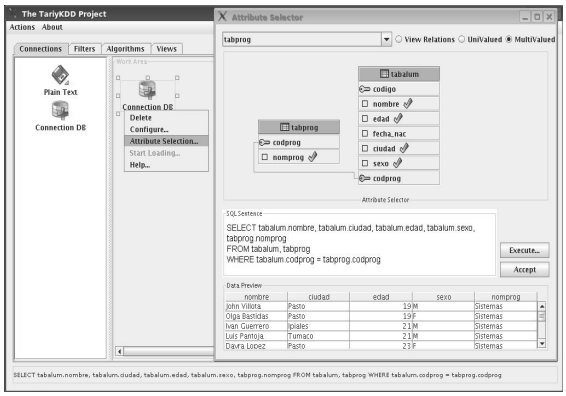

Figura 2. Interfaz gráfica de conexión a TariyKDD

2.2. Módulo de utilidades

En este módulo se mantiene una colección de clases principales y librerías que son utilizadas por otras clases a lo largo de la aplicación para el desarrollo de tareas comunes como visualización, manipulación e intercambio de datos.

El objetivo de este módulo es tener centralizada la administración de este tipo de clases y aprovechar la reutilización de código.

2.3. Módulo del Kernel TariyKDD

El objetivo del módulo de Kernel es permitir la interacción y comunicación entre los diferentes componentes de los módulos involucrados, así como de proveer soporte en el caso de no ser necesaria la utilización de alguno de ellos, por ejemplo, cuando una conexión se hace directamente hacia uno de los algoritmos de minería sin pasar por algún filtro. En este módulo reposan los paquetes primordiales para la ejecución de las tareas de descubrimiento de conocimiento. Esta compuesto por los submódulos de filtros, algoritmos y visores.

2.3.1 Módulo de filtros. El módulo de preprocesamiento contiene los diferentes filtros y rutinas necesarias para la transformación y adecuación de los datos, si es necesario, antes de aplicar las técnicas de minería de datos. A este módulo pertenecen los filtros:

-*RemoveMissing:* Elimina todas las transacciones que contengan campos vacíos.

-*UpdateMissing:* Remplaza los campos vacíos de un atributo especifico, por un valor otorgado por el analista.

-*Selection:* Hace una selección de atributos y del atributo objetivo sobre un conjunto de entrada.

-*Range:* Escoge una muestra sobre un conjunto de entrada de manera aleatoria o especificando un rango, especialmente útil para minería con algoritmos de clasificación.

-*Reduction:* Hace una reducción en el número de transacciones, manteniéndolas o eliminándolas, con diferentes técnicas y parámetros de selección.

-*Codification:* Realiza una codificación sobre el conjunto de datos de entrada.

-*ReplaceValue:* Remplaza uno o varios valores de un atributo seleccionado, por otro valor suministrado por el analista.

-*NumericRange:* Elimina los valores de un atributo numérico, que están por fuera de un rango determinado por el analista.

-*Discretize:* Efectua una transformación de los datos llevando un valor numérico a un formato categórico.

La figura 3 muestra la interfaz gráfica de filtros de esta herramienta.

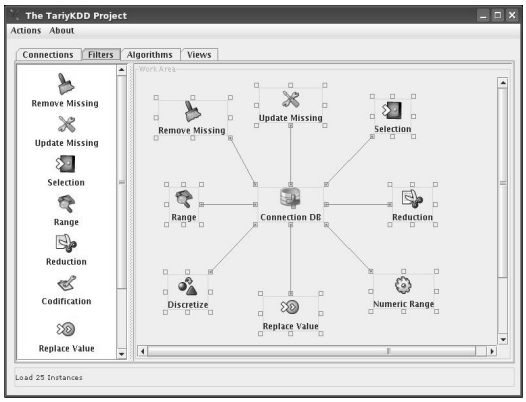

Figura 3. Interfaz gráfica de filtros de TariyKDD

2.3.2 Módulo de algoritmos. El módulo de algoritmos contiene las clases encargadas de aplicar las técnicas específicas de minería de datos. En TariyKDD se implementaron los siguientes algoritmos para las tareas de asociación y clasificación:

Para la tarea Asociación:
- *A priori*, propuesto en [2], tiene como objetivo reducir el número de conjuntos considerados, generando un conjunto de itemsets frecuentes a partir de itemsets candidatos.
- *FP – Growth,* propuesto en [7] [8], utiliza una estructura de datos llamada árbol de patrones frecuentes o FP-tree, la cual es una estructura que almacena información crucial y cuantitativa acerca de los patrones frecuentes.
- *EquipAsso*, propuesto en [19][20][22] es un algoritmo, para el cálculo de los itemsets frecuentes basado en dos operadores del álgebra relacional para Asociación: Associator y EquiKeep [18] e implementado en el lenguaje SQL mediante las primitivas SQL Associator Range y EquiKeep On [18].

Para la tarea Clasificación:
- *C4.5,* propuesto en [13] [14], construye el árbol de arriba hacia abajo recursivamente utilizando la manera de divide y vencerás. El algoritmo usa una métrica basada en la entropía, conocida como ganancia de información para construir el árbol.
- *Mate-tree*, propuesto en [21], es un algoritmo para la tarea de clasificación basado en el operador algebraico relacional *Mate* [23][24] que conjuntamente con los operadores agregados *Entro* y *Gain* [23][24] facilitan el cálculo de la *Ganancia de Información* y con el operador algebraico relacional *Describe Classifier* [23][24], la construcción del árbol de decisión.

La Figura 4 muestra la interfaz gráfica del módulo de algoritmos.

2.3.3. Módulo de visores. El módulo de visores contiene las clases necesarias para construir y desplegar las estructuras y reportes que permitirán visualizar de manera dinámica los resultados de la aplicación de técnicas de minería y preprocesamiento. Dentro de este módulo se desarrollaron las siguientes utilidades:

Rules: Este reporte organiza las reglas de asociación que se obtienen como resultado de los algoritmos *Apriori*, *FPGrowth* y *EquipAsso*. Posee criterios de ordenamiento como la confianza, el soporte y la relación entre el numero de atributos en el antecedente con respecto al consecuente. Se consideran más valiosas aquellas reglas donde un menor número de atributos en el antecedente implican la aparición de mayor número de atributos en el consecuente, especialmente en el problema de canasta de mercado.

Trees: Estas estructuras proveen un mecanismo dinámico para la visualización de los árboles de decisión resultado de los algoritmos *C4.5* y *Mate-tree*. Estos árboles, ofrecen reportes con diferentes criterios de ordenamiento como la confianza de la regla, el número de atributos y la categoría de la clase objetivo y permiten un despliegue dinámico de todas las reglas del árbol. Se implementaron en este módulo tres tipos de árboles: uno de tipo texto, útil a la hora de impresión, uno jerárquico, que permite la visualización paso a paso y uno dinámico, que permite un navegación más ágil a través de las reglas.

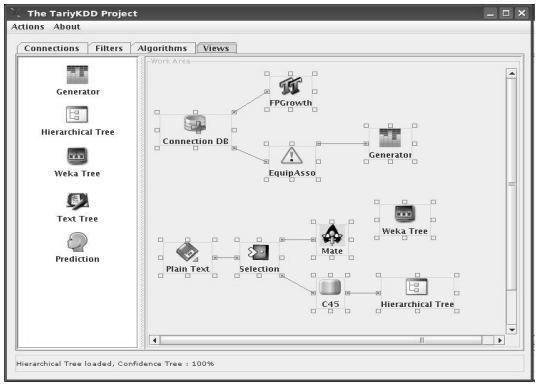

Figura 4. Interfaz gráfica de algoritmos de TariyKDD

Prediction: Esta utilidad permite la predicción de las clases de nuevos registros a partir de los modelos construidos con los algoritmos de clasificación.

2.4. Módulo de interfaz gráfica

Este módulo da soporte visual a los módulos de conexión, filtros, algoritmos y visores, y se encarga de brindar al usuario un medio agradable y fácil de usar para la construcción de experimentos que involucren la interacción entre los diferentes componentes de la herramienta.

3. Aspectos de Implementación de TariyKDD

La herramienta TariyKDD se desarrolló bajo el sistema operativo Fedora Core en sus versiones 3 y 5 bajo una arquitectura de procesador a 64 bits, utilizando el lenguaje de programación Java 5.0, actualización 09, lo que la convierte en una herramienta independiente a la plataforma donde se ejecute. Para su construcción, se usaron herramientas de software libre con aplicaciones como el IDE de desarrollo NetBeans 5.5, Subversion, para el control de versiones, The Gimp y KIconEdit, para la manipulación de imágenes e iconos, PostgreSQL 8.1, para las pruebas de conexión y evaluación, entre otros.

TariyKDD se diseño utilizando el análisis y diseño orientado a objetos con el lenguaje de modelamiento unificado UML. El diseño de TariyKDD es modular con el fin de permitir el crecimiento continuo de esta herramienta. En la Figura 5 se presenta el diagrama de paquetes de TariyKDD para ilustrar el desarrollo de la herramienta. Nuevos filtros, algoritmos y reportes pueden ser fácilmente implementados e incluidos en TariyKDD siguiendo este diseño. La estandarización de las entradas de los algoritmos, provenientes del módulo de conexión o de filtros, y sus salidas, hacia los esquemas de visualización, facilitó al grupo de desarrollo la implementación de los algoritmos,

trabajando de manera distribuida, haciendo uso de aplicaciones para el control de versiones.

Se implementó una estructura de datos en memoria que mejora el rendimiento de los algoritmos de minería de datos, como se aprecia en la Figura 6. Esta estructura se baso en un árbol n-ario cuyo uso permite que registros con elementos repetidos puedan compartir un mismo espacio en memoria. Si un registro completo se repite solo se incrementa un contador en la hoja de esa rama. Entre mayor sea el número de registros mejor se aprovechará esta estructura.

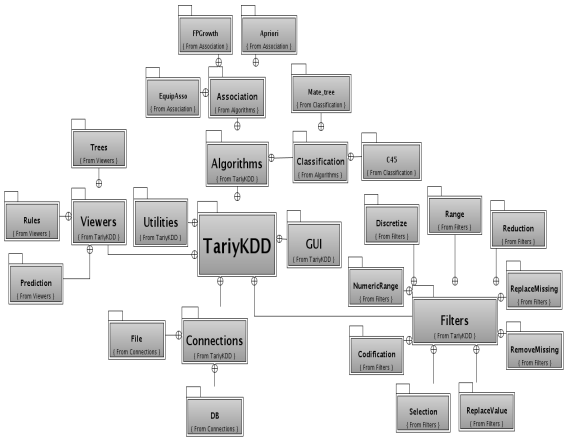

Figura 5. Diagrama de paquetes de TariyKDD

Tabla 1. Tabla de datos de ejemplo

Estado	Temperatura	Humedad	Viento	Jugar
Soleado	Caliente	Alta	Débil	NO
Soleado	Caliente	Alta	Fuerte	NO
Nublado	Caliente	Alta	Débil	SI
Lluvioso	Templado	Alta	Débil	SI
Lluvioso	Fresco	Normal	Débil	SI
Lluvioso	Fresco	Normal	Fuerte	NO

Por otra parte, este esquema permite el almacenamiento de registros de cualquier longitud, lo que facilita el trabajo con conjuntos de datos del modelo de la canasta de mercado, donde el tamaño de cada transacción es variable. Para esta estructura, previamente, se codifica cada entrada del registro a un entero corto, minimizando el uso de recursos por cada nodo.

Otra importante característica que se implementó en *TariyKDD*, fue la facilidad de uso, donde se adoptó la metodología de arrastrar y soltar tanto para la interfaz principal, a través de un grafo de nodos, como para el módulo de conexión a SGBD, donde, una vez establecida la conexión, se construye una vista a partir de la selección visual de los atributos y las tablas involucradas. Esto se logró aprovechando las funcionalidades del lenguaje de programación como *DnD*, *Swing* y *JDBC 3.0*.

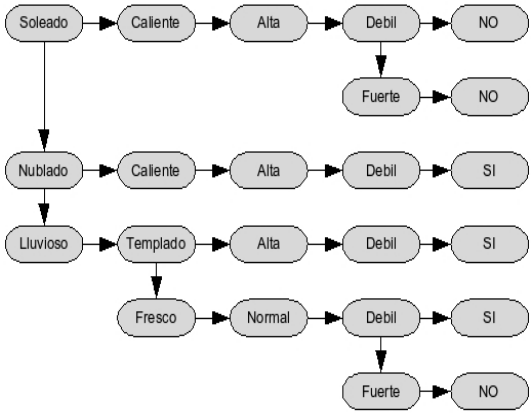

Figura 6. Estructura de datos para el almacenamiento de registros en memoria

La inclusión de los algoritmos *EquipAsso*, *FPGrowth* y *Mate-tree*, para las tareas de asociación y clasificación respectivamente, no presentes en otras herramientas, es una característica importante de *TariyKDD*. Se implementó el módulo de *predicción*, en el cual, una vez obtenido el árbol de decisión, se procede con un nuevo conjunto de datos (que se desconoce a que clase pertenecen) a predecir su clase, módulo que se encuentra en pocas herramientas de minería de datos.

En la etapa de visualización se desarrolló una serie de reportes utilizando tablas dinámicas que facilitaron el ordenamiento de los resultados por diferentes criterios. Se logró visualizar los árboles de decisiones ofreciendo diferentes alternativas al usuario. Una de ellas, fue la implementación del árbol dinámico de la herramienta WEKA, la cual, al ser software de código abierto, se estudió y extendió, añadiendo la funcionalidad de gráficos de pastel para cada nodo del árbol.

4. Pruebas de Funcionalidad

Las pruebas y evaluación del rendimiento de los algoritmos Apriori, FP-Growth y EquipAsso así como el de los algoritmos Mate y C4.5, se realizaron en un computador con un procesador AMD 64 bits a 2 Ghz, con una memoria RAM de 1Gb, con disco duro Serial ATA de 80Gb con una tasa de transferencia de 150 Mb/seg.

El conjunto de datos utilizados en las pruebas de asociación pertenecen a las transacciones de uno de los supermercados más importantes del departamento de Nariño (Colombia) durante un periodo determinado. El conjunto de datos contiene 10.757 diferentes productos. Los conjuntos de datos utilizados con TariyKDD se muestran en la Tabla 2.

Tabla 2. Descripción de los conjuntos de datos .

Nomenclatura	Número de registros	Número de transacciones	Promedio de ítems
BD85KT7	555.123	85.692	7
BD40KT5	197.337	40.256	5

Para cada conjunto de datos se realizó preprocesamiento y transformación de datos con el fin de eliminar los productos repetidos en cada transacción y transformar las tablas objeto a un modelo simple (i.e. una tabla con esquema Tid, Item).

Se evaluó el rendimiento de los algoritmos Apriori, FP-Growth y Equipasso, comparando los tiempos de respuesta para diferentes soportes mínimos. Los resultados de la evaluación del tiempo de ejecución de estos algoritmos, aplicados a los conjuntos de datos BD85KT7 y BD40KT5, se pueden observar en las tablas 3,4 y figuras 7 y 8 respectivamente.

En general, observando el comportamiento de los algoritmos FP-Growth y Equipasso con los diferentes conjuntos de datos, se puede decir que su rendimiento es similar, contrario al tiempo de ejecución de Apriori, que se ve afectado significativamente a medida que se disminuye el soporte.

Tabla 3. Resultados obtenidos para BD85KT7

BD85KT7			
Soporte (%)	Tiempo (ms)		
	Apriori	FP-Growth	EquipAsso
4.15	750	166	85
4.75	362	162	82
5.35	365	164	83
5.95	365	162	83
6.55	120	159	80

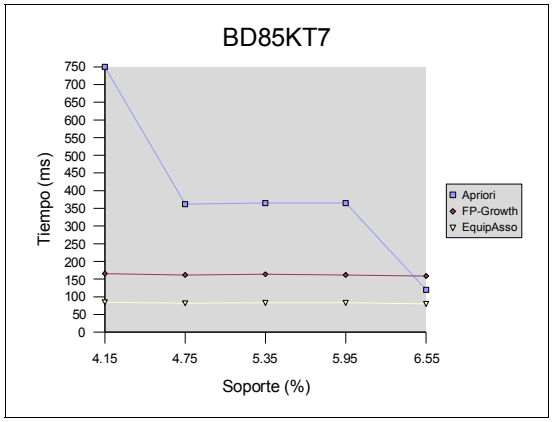

Figura 7. Resultados obtenidos para BD85KT7.

Analizando el tiempo de ejecución de únicamente los algoritmos FP-Growth y EquipAsso (Figuras 9 y 10) para los conjuntos de datos BD85KT7 y BD40KT5, con soportes más bajos, el comportamiento de estos algoritmos sigue siendo similar.

Tabla 4. Resultados obtenidos para BD40KT5

BD40KT5			
Soporte (%)	Tiempo (ms)		
	Apriori	FP-Growth	EquipAsso
1.90	268	66	29
2.00	265	64	29
2.10	132	63	27
2.20	45	61	27
2.30	44	61	27

Para realizar las pruebas de clasificación se seleccionó la base de datos histórica UDENAR[25] que contiene información personal y académica de 20.328 estudiantes de la Universidad de Nariño (Colombia) con un total de 22 atributos.

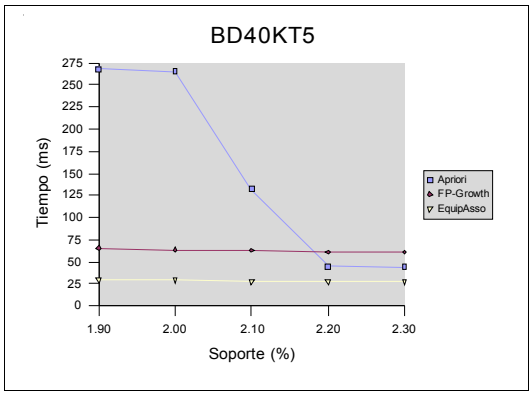

Figura 8. Resultados obtenidos para BD40KT5.

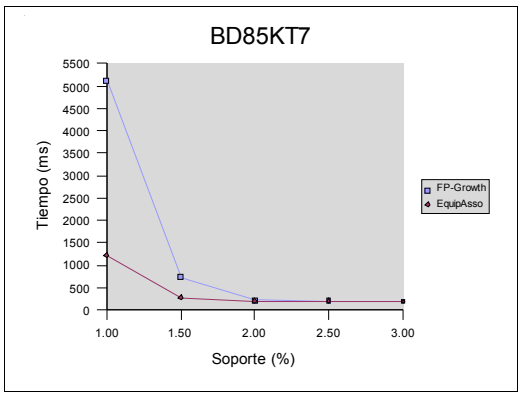

Figura 9. Resultados obtenidos para FP-Growth y EquipAsso en BD85KT7.

Se hicieron dos tipos de pruebas sobre el conjunto UDENAR evaluando los tiempos de ejecución de los algoritmos de clasificación implementados en TariyKDD (C4.5 y Mate). Las primeras pruebas variaban el número de atributos con el que se trabajaba y en las segundas se varió el número de registros con el que se construyó el modelo.

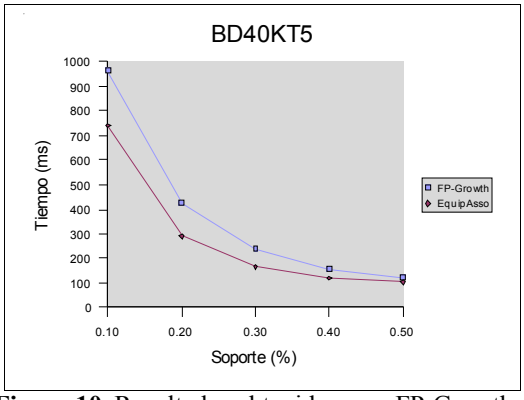

Figura 10. Resultados obtenidos para FP-Growth y EquipAsso en BD40KT5.

En la primera prueba se construyeron modelos seleccionando en primera instancia 22 atributos y el total de registros (20.328), la clase objetivo con la que se evaluó el modelo fue el rendimiento de los estudiantes. Posteriormente, se eliminó uno a uno los atributos hasta un limite de 4 columnas. Los resultados del análisis se pueden observar en la Figura 11.

A partir de una segunda prueba se determinó los atributos con mayor fuerza clasificatoria y se construyó una vista minable con los 5 atributos más relevantes (facultad, semestre, sexo, edad_ingreso, ponderado) más la clase (clase_rendimiento) y se tomaron muestras desde 20000 hasta 2000 registros disminuyendo el tamaño del conjunto en 2000 registros para cada medición. Un resumen de la prueba se puede ver a en la Figura 12.

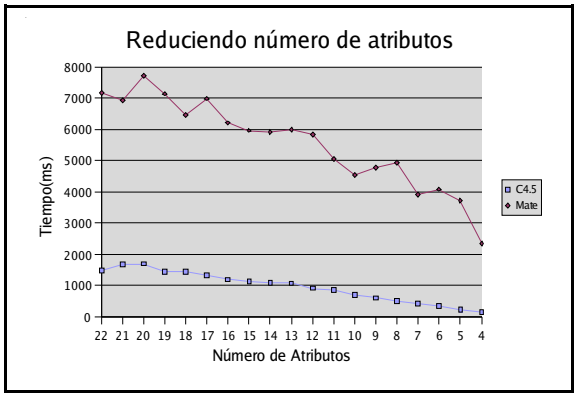

Figura 11. Rendimiento de algoritmos en el conjunto UDENAR al reducir atributos.

5. Conclusiones y Trabajos Futuros

Se cuenta con la primera versión de TariyKDD, una herramienta genérica para el Descubrimiento de reglas de Asociación y Clasificación, débilmente acoplada con un SGBD, desarrollada en el laboratorio de KDD del departamento de Ingeniería de Sistemas de la Universidad de Nariño (Colombia), bajo software libre. Soporta las etapas del proceso DCBD: selección, limpieza, transformación, minería de datos y visualización. En la etapa de minería de datos, soporta las tareas de Asociación con los algoritmos Apriori, FPGrowth y EquipAsso y la tarea de Clasificación con los algoritmos C4.5 y Mate-tree.

Las diferentes pruebas de funcionalidad con conjuntos de datos sintéticos como con conjuntos de datos reales, demuestran que la herramienta funciona correctamente y que sus resultados son fiables. Actualmente, TariyKDD es utilizada por los estudiantes de los últimos semestres del programa de Ingeniería de Sistemas de la Universidad de Nariño para realizar sus proyectos de minería de datos, en la asignatura de Descubrimiento de Conocimiento en Bases de Datos.

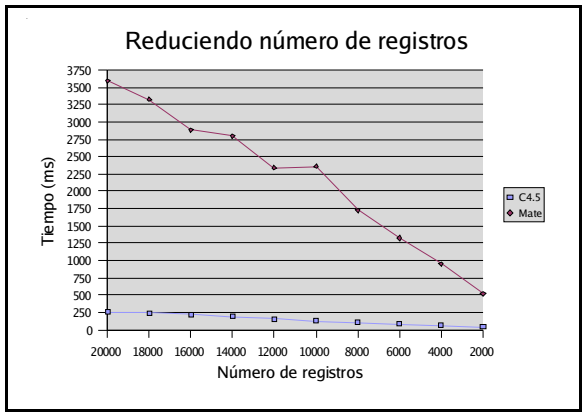

Figura 12. Rendimiento de algoritmos en el conjunto UDENAR al reducir registros.

TariyKDD es una herramienta de software libre, bajo licencia GPL v2.0, por tanto cualquier persona es libre de utilizarla y desarrollar y probar sus propios algoritmos reutilizando tanto código como le sea posible. Descargas, documentación, capturas de pantalla, conjuntos de datos, tutoriales y créditos pueden ser vistos en la página web del proyecto en [16].

Como trabajos futuros están, el de implementar en TariyKDD otras tareas y algoritmos de minería de datos, como clustering y patrones secuenciales. Implementar gráficos estadísticos a los conjuntos de datos cargados para obtener una información inicial de sus características. Utilizar la herramienta TariyKDD en proyectos reales de minería de datos con empresas colombianas, especialmente con las PYMES.

Referencias

[1] ADaM, The Algorithm Development and Mining System, http://datamining.itsc.uah.edu/adam/, consultado en septiembre de 2006.

[2] Agrawal, R., Srikant, R., Fast Algorithms for Mining Association Rules, VLDB Conference, Santiago, Chile, 1994.

[3] Chen, M., Han, J., Yu, P., Data Mining: An Overview from Database Perspective, IEEE Transactions on Knowledge and Data Engineering, 1996.

[4] Demsar, J., Zupan, B. Orange: From experimental machine learning to interactive data mining. Technical report, Faculty of Computer and Information Science, University of Liubliana, Slovenia, http://www.ailab.si/orange/wp/orange.pdf, consultado en septiembre 2006.
[5] E-Business Technology Institute, University of Hong Kong. http://www.eti.hku.hk, consultado en septiembre 2006.
[6] Goebel, M., Gruenwald, L., A Survey Of Data Mining And Knowledge Discovery Software Tools., In SIGKDD Explorations, volume 1 of 1, June 1999.
[7] Han, J., Pei, J., Yin, Y., Mining Frequent Patterns without Candidate Generation, Proc. ACM SIGMOD, Dallas, TX, 2000.
[8] Han, J., Pei, J., Mining Frequent Patterns by Pattern-Growth: Methodology and Implications, SIGKDD Explorations, 2:14-20, 2000.
[9] Han, J., Kamber, M. Data Mining Concepts and Techniques. Morgan Kaufmann Publishers, San Francisco,2001.
[10] Kdnuggets, http://www.kdnuggets.com/software, consultado en septiembre 2006.
[11] Mierswa, M., Wurst, R., Klinkenberg, M., Scholz,, Euler, T., Yale: Rapid prototyping for complex data mining tasks. 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006.
[12] Piatetsky-Shapiro, G., Brachman, R., Khabaza, T., An Overview of Issues in Developing Industrial Data Mining and Knowledge Discovery Applications. 1996.
[13] Quinlan, J.R., Induction of decision trees. Machine Learning, 81-106, 1986.
[14] Quinlan, J.R., C4.5: Programs for Machine Leraning, Morgan Kaufmann Publishers, 1993.
[15] Rakotomalala, R., Tanagra. In TANAGRA: a free software for research and academic purposes, volume 2, pages 697–702. EGC'2005, 2005.
[16] TariyKDD Project. Grupo de investigación aplicado a sistemas – GRiAS, Facultad de Ingeniería, UniversidaddeNariño,Colombia, http://tariykdd.berlios.de, consultado en abril de 2007.
[17] Timarán, R., Arquitecturas de Integración del Proceso de Descubrimiento de Conocimiento con Sistemas de Gestión de bases de datos: un Estado del Arte, en revista Ingeniería y Competitividad, Universidad del Valle, Volumen 3, No. 2, Cali, diciembre de 2001.
[18] Timarán R., Millán M., Machuca F., New AlgebraicOperators and SQL Primitives for Mining Association Rules, Proceedings of the IASTED International Conference Neural Networks and Computational Intelligence, Cancun, Mexico, 2003.
[19] Timarán, R., Millán, M., EquipAsso: un algoritmo para el descubrimiento de Reglas de Asociación basado en operadores algebraicos, en memorias de la 4ª Conferencia Iberoamericana en Sistemas, Cibernética e Informática CICI 2005, Orlando, Florida, EE.UU., julio 2005.
[20] Timarán, R., Millán, M., EquipAsso: an Algorithm based on New Relational Algebraic Operators for Association Rules Discovery, in proceedings of the Fourth IASTED International Conference on Computational Intelligence, Calgary, Alberta, Canada, july 2005.
[21] Timarán, R., Nuevas Primitivas SQL para el Descubrimiento de Conocimiento en Arquitecturas Fuertemente Acopladas con un Sistema de Gestión Bases de Datos, Tesis Doctoral, Escuela de Ingeniería de Sistemas y Computación, Universidad del Valle, Cali, Colombia, enero 2006.
[22] Timarán, R., Calderón, A., Ramirez, I., Alvardo, J., Guevara, F., Análisis de desempeño de EquipAsso: un algoritmo para el cálculo de Itemsets frecuentes basado en operadores algebraicos relacionales, en proceedings de la XXXII Conferencia Latinoamericana de Informatica (CLEI 2006), Santiago de Chile, Chile, agosto 2006.
[23] Timarán, R., Millán, M., New Algebraic Operators and SQL Primitives for Mining Classification Rules, in proceedings of the IASTED International Conference on Computational Intelligence (CI 2006), International Association of Science and Technology for Development, San Francisco, USA, november 2006.
[24] Timarán, R., Extensión del Lenguaje SQL con Nuevas Primitivas para el Descubrimiento de Reglas de Clasificación, VI Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento, ISBN 978-9972-2885-1-7, Facultad de Ciencias e Ingeniería, Pontificia Universidad Católica del Perú, Primera Edición, Lima, Perú, enero de 2007.
[25] Timarán, R., Lombana C, Narvaez R, Viteri G, Dulce E., Detección de patrones de bajo rendimiento y/o deserción de estudiantes de la Universidad de Nariño con técnicas de minería de datos, informe final VIII Convocatoria Alberto Quijano Guerrero, Universidad de Nariño, Pasto, Colombia, 2006.
[26] Waikato ML Group, The waikato environment for knowledge analisys, http://www.cs.waikato.ac.nz/ml/weka, consultado en septiembre de 2006.