Herramienta *software* para el análisis de canasta de mercado sin selección de candidatos

Software tool for analysing the family shopping basket without candidate generation

Roberto Carlos Naranjo Cuervo[1] y Luz Marina Sierra Martínez[2]

RESUMEN

Actualmente en el entorno del comercio electrónico es necesario contar con herramientas que permitan obtener conocimiento útil que brinde soporte a la toma de decisiones de marketing; para ello se necesita de un proceso que utiliza una serie de técnicas para el procesamiento de los datos, entre ellas se encuentra la minería de datos, que permite llevar a cabo un proceso de descubrimiento de información automático. Este trabajo tiene como objetivo presentar la técnica de reglas de asociación como la adecuada para descubrir cómo compran los clientes en una empresa que ofrece un servicio de comercio electrónico tipo B2C, con el fin de apoyar la toma de decisiones para desarrollar ofertas hacia sus clientes o cautivar nuevos. Para la implementación de las reglas de asociación existe una variedad de algoritmos como: A priori, DHP, Partition, FP-Growth y Eclat y para seleccionar el más adecuado se define una serie de criterios (Danger y Berlanga, 2001), entre los que se encuentran: inserciones a la base de datos, costo computacional, tiempo de ejecución y rendimiento, los cuales se analizaron en cada algoritmo para realizar la selección. Además, se presenta el desarrollo de una herramienta software que contempla la metodología CRISP-DM constituida por cuatro submódulos, así: Preprocesamiento de datos, Minería de datos, Análisis de resultados y Aplicación de resultados. El diseño de la aplicación utiliza una arquitectura de tres capas: Lógica de presentación, Lógica del Negocio y Lógica de servicios; dentro del proceso de construcción de la herramienta se incluye el diseño de la bodega de datos y el diseño de algoritmo como parte de la herramienta de minería de datos. Las pruebas hechas a la herramienta de minería de datos desarrollada se realizaron con una base de datos de la compañía FoodMart[3]. Estas pruebas fueron de: rendimiento, funcionalidad y confiabilidad en resultados, las cuales permiten encontrar reglas de asociación igualmente. Los resultados obtenidos facilitaron concluir, entre otros aspectos, que las reglas de asociación como técnica de minería de datos permiten analizar volúmenes de datos para servicios de comercio electrónico tipo B2C, lo cual es una ventaja competitiva para las empresas.

Palabras clave: minería de datos, comercio electrónico B2C, análisis de la canasta de mercado.

ABSTRACT

Tools leading to useful knowledge being obtained for supporting marketing decisions being taken are currently needed in the e-commerce environment. A process is needed for this which uses a series of techniques for data-processing; data-mining is one such technique enabling automatic information discovery. This work presents the association rules as a suitable technique for discovering how customers buy from a company offering business to consumer (B2C) e-business, aimed at supporting decision-making in supplying its customers or capturing new ones. Many algorithms such as A priori, DHP, Partition, FP-Growth and Eclat are available for implementing association rules; the following criteria were defined for selecting the appropriate algorithm: database insert, computational cost, performance and execution time. The development of a software tool is also presented which involved the CRISP-DM approach; this software tool was formed by the following four sub-modules: data pre-processing, data-mining, results analysis and results application. The application design used three-layer architecture: presentation logic, business logic and service logic. Data warehouse design and algorithm design were included in developing this data-mining software tool. It was tested by using a FoodMart company database; the tests included performance, functionality and results' validity, thereby allowing association rules to be found. The results led to concluding that using association rules as a data mining technique facilitates analysing volumes of information for B2C e-business services which represents a competitive advantage for those companies using Internet as their sales' media.

Keywords: data-mining, B2C e-business, family shopping basket analysis.

Recibido: abril 18 de 2008
Aceptado: marzo 2 de 2009

[1] Ingeniero de Sistemas, Universidad Industrial de Santander, Colombia. Docente en Planta Tiempo completo categoría Asociado, Universidad del Cauca, Colombia. rnaranjo@unicauca.edu.co
[2] Ingeniera de Sistemas, Universidad Industrial de Santander UIS, Colombia. Especialización en Gerencia de Proyectos, Universidad del Cauca, Colombia. Docente en Planta Tiempo completo categoría Asociado. Universidad del Cauca. lsierra@unicauca.edu.co
[3] FoodMart: es una gran cadena de tiendas de ultramarinos dispersas por EE.UU., México y Canadá

Introducción

Hoy en día para las empresas se ha convertido en una necesidad y oportunidad el conocer la información y analizarla en pro de tomar decisiones que en el momento apropiado apoyen su gestión y supervivencia en la actual y competitiva economía (Tapscott, Lowy y Ticoll, 2000). De ahí surge la necesidad de incorporar en su dinámica herramientas informáticas que permitan procesar y obtener de los volúmenes de información almacenados los elementos suficientes para tomar decisiones.

Es necesario tener clara y precisa comprensión de que para una empresa tomar una decisión sin el conocimiento profundo de la información implica la posibilidad de errar en la toma de decisiones, dado que conlleva el costo requerido para poner en marcha un plan que busque la fidelidad de los clientes o capturar nuevos, o cautivar a un nuevo nicho de mercado. Según sea el fin que se pretende alcanzar con la toma de la decisión, si este no se logra se habrá perdido el esfuerzo de dicha estrategia.

Adicionalmente, y teniendo en cuenta que la forma de hacer negocios en las empresas ha migrado a ambientes web, como es el caso del comercio electrónico, donde ellas pueden desde publicitar sus productos y comercializarlos hasta hacer sus propias compras vía Internet, es allí donde encuentran un punto vital y estratégico, y por lo tanto, no pueden conformarse sólo con establecer una infraestructura tecnológica para ofrecer productos y servicios a través de un sitio de comercio electrónico (Castañeda y Rodríguez, 2005), sino que deben contar con herramientas que permitan aprovechar y potencializar las ventajas ofrecidas por la Internet, como es el caso de mejorar la efectividad en las ventas de productos online, al analizar los datos para apoyar sus decisiones de marketing, dado que ese análisis deberá proveer respuestas a preguntas como: ¿qué se debe hacer para entender cómo compran los clientes? Para responder esta pregunta, es necesario partir del análisis de canasta de mercado, ya que una canasta de mercado tipica contiene los datos de la compra de productos de un cliente, en qué cantidad cada uno, y en qué época lo hace. Por lo tanto, es necesario descubrir patrones interesantes ocultos, no triviales, y de interés para las empresas alrededor de los mismos, lo cual es el objetivo principal de la minería de datos (CRISP-DM Consortium, 2000).

Las herramientas de minería de datos predicen futuras tendencias y comportamientos, permitiendo tomar decisiones conducidas por un conocimiento acabado de la información; para conseguirlo, hace uso de diferentes tecnologías que resuelven problemas típicos de agrupamiento automático, clasificación, asociación de atributos y detección de patrones secuenciales (Kimbal y Ross, 2002).

Este trabajo presenta un aporte al proponer el desarrollo de un prototipo software, accesible a pequeñas y medianas empresas, que permita apoyar el proceso de selección de estrategias publicitarias y comerciales para venta de productos o servicios a través de la web utilizando una técnica de minería de datos que mejor se ajuste a esta necesidad y cumpla requerimientos computacionales tales como eficiencia y eficacia, su implementación sea viable y los resultados arrojados sean iguales o mejores que otras herramientas hechas para el mismo fin.

Existen otros proyectos, tales como el "Plan para enfocar campañas bancarias utilizando Dataminig" (DeLuca, 2006), y el trabajo "Mining interesting knowledge from weblogs: a survey. Data and knowledge" (Facca y Lanzi, 2004). El primero presenta un plan para enfocar la estrategia comercial basado en los datos transaccionales de los bancos y la metodología CRISP-DM, utilizando la herramienta Clementine, de SPSS (Clementine-SPSS, 2008), y el segundo exhibe un survey donde despliega la importancia de realizar análisis de los datos presentes en los logs de los servidores web hoy en día. Ambos trabajos muestran la importancia de efectuar análisis de datos mediante un proceso de minería de datos, pero adicionalmente este trabajo plantea el análisis de la canasta de mercado de una empresa que ofrece venta de productos mediante la selección de la técnica de reglas de asociación como la mejor alternativa para realizar dichos análisis (Naranjo, Montenegro, 2007), teniendo en cuenta además, para encontrar el mejor algoritmo que la implemente, criterios tales como inserciones a la base de datos, costo computacional, tiempo de ejecución, rendimiento, y que resuelva el problema de selección de candidatos. Al final del artículo se plantea la construcción de una herramienta software que soporte la técnica de análisis, permitiendo facilitar la toma de decisiones.

Para este último proyecto se siguió la metodología CRIPS-DM (CRISP-DM Consortium, 2000), por ser ampliamente utilizada en proyectos de minería de datos (DeLuca, 2006). Siguiendo tal metodología, en la primera fase se hizo una comprensión del negocio, donde se estableció como objetivo conocer cuál era el comportamiento de los clientes que compran en una empresa dedicada al comercio electrónico B2C; en la segunda fase se exploraron los datos disponibles; para este caso, se estudiaron los datos de la compañía foodMart, que contaba con cerca de 250.000 registros de ventas; en la tercera fase se diseñó una bodega de datos, ya que se contaba con una base de datos transaccional, que se cargaron a la bodega de datos; en la cuarta fase nos centramos en la selección del algoritmo de reglas de asociación que mas se adecuara de acuerdo a los criterios definidos; el diseño y desarrollo de la herramienta software fue en la quinta fase, y se evaluó el resultado del análisis arrojado por la herramienta construida. Para corroborar sus resultados se hizo el contraste con otras herramientas que llevaban a cabo análisis similar (Cabena y Stadler, 1998).

En este artículo se encuentran los conceptos sobre las reglas de asociación requeridos para el trabajo que se trata en el presente documento; seguidamente se describe cómo se hizo la selección del algoritmo para implementar las reglas de asociación, luego se presenta la herramienta software de minería de datos desarrollada, su arquitectura, el diseño de la bodega de datos y la implementación del algoritmo en la aplicación; posteriormente, se presentan las pruebas a las que fue sometida la herramienta de minería desarrollada. Finalmente, se dan las conclusiones asociadas al trabajo aquí presentado.

Reglas de asociación en la transacción de negocios

Conceptos

Actualmente, con la masiva cantidad de datos que las organizaciones recolectan en sus procesos de negocio el descubrimiento de asociaciones interesantes en los registros de transacciones puede ayudar para la toma de decisiones en los procesos de marketing (Han y Kamber, 2002). En el ejemplo típico para reglas de asociación, "el análisis de canasta de mercado", supóngase que un granjero local ha puesto un stand de verduras y está ofreciendo los siguientes artículos: {espárragos, fríjoles, brócoli, maíz, pimientas

verdes, calabazas, tomates}, a este conjunto de artículos lo denotaremos I, y en la Tabla 1 se mostrarán los artículos comprados.

En el conjunto D de transacciones representadas en la Tabla 1, cada transacción (T) en D representa un conjunto de artículos contenidos en I. Suponga que se tiene un conjunto particular de artículos A (e. g., fríjoles y calabazas), y otro conjunto de artículos B (e.g., espárragos). Luego una regla de asociación toma la forma de (A =>B), donde el antecedente A y el consecuente B son subconjuntos propios de I, y A y B son mutuamente exclusivos.

Existen dos medidas asociadas a una regla de asociación: soporte y confianza, que le dan validez a la misma. El soporte para una regla de asociación particular A=>B es la proporción de transacciones en D que contienen A y B (Larose, 2004).

Tabla 1. Artículos comprados por los clientes

No	*Contenido canasta*
1	Brócoli, pimienta, maíz
2	Espárragos, calabaza, maíz
3	Maíz, Tomates, frijoles, calabazas
4	Pimienta, maíz, tomates, frijoles
5	Frijoles, espárragos, frijoles, tomates
6	Calabaza, espárragos, frijoles, tomates
7	Tomates, maíz
8	Brócoli, tomates, pimienta
9	Calabaza, espárragos, frijoles
10	Frijoles, maíz
11	Pimienta, brócoli, frijoles, calabaza
12	Espárragos, frijoles, calabaza
13	Calabaza, maíz, espárragos, frijoles
14	Maíz, pimienta, tomates, frijoles, brócoli

$$Soporte = P(A \cap B) = \frac{No.transacciones\ cont\ A\ y\ B}{No.total\ transacciones}$$

La confianza c de la regla de asociación A=>B es una medida de exactitud de la regla, determinada por el porcentaje de transacciones en D que contienen A y B (Larose, 2004).

$$Confianza = P(B/A) = \frac{P(A \cap B)}{P(A)}$$

El analista puede preferir reglas que tengan alto soporte o alta confianza, o usualmente ambas. Las reglas fuertes son las que reúnen o superan ciertos soportes mínimos y criterios de confianza. Por ejemplo, un analista interesado en encontrar qué artículos del supermercado se compran juntos, puede establecer un nivel de soporte mínimo de 20% y un nivel de confianza mínimo del 70%. Por otro lado, en detección de fraude o de terrorismo, se necesitaría reducir el nivel de soporte mínimo a 1% o menos, ya que comparativamente pocas transacciones son fraudulentas o relacionadas con terrorismo.

Un itemset es un conjunto de artículos contenidos en I, y un k-itemset es un itemset que contiene k artículos; por ejemplo, {fríjoles, calabazas} es un 2-itemset, y {brócoli, pimienta verde, maíz} es un 3-itemset, cada uno de los estantes de vegetales puestos en I. La frecuencia Φ del conjunto de artículos (itemset) es simplemente el número de transacciones que contienen el conjunto de artículos particular. Un conjunto de artículos frecuente es aquel que ocurre al menos un cierto mínimo número de veces, teniendo una frecuencia de conjunto de artículos, por ejemplo: suponiendo que Φ = 4, los conjuntos de artículos que ocurren más de cuatro veces se dice que son frecuentes; denotamos el conjunto de k-itemsets como Fk.

Las reglas de asociación para minería de grandes bases de datos son procesos de dos pasos:

1. Encontrar todos los conjuntos de artículos frecuentes, es decir, aquellos con frecuencia ≥ Φ.

2. Del conjunto de artículos frecuentes, generar reglas de asociación que satisfagan condiciones mínimas de soporte y confianza.

Se observa que la técnica de reglas de asociación es la que más se adecúa al problema que queremos resolver, ya que se quiere descubrir el comportamiento de compra de los clientes con respecto a los productos ofrecidos (Han y Kamber, 2002). Además esta técnica sugiere una búsqueda por toda la base de datos, realizando una clasificación en cada barrido, por lo tanto no hay límite establecido para la cantidad de datos que puede manejar, busca las características presentes en las transacciones realizadas, las cuales pueden tener atributos de diferentes tipos, por lo tanto no es necesario hacer una conversión a un tipo de datos específico. La capacidad predictiva de la técnica depende de las medidas establecidas de confianza y soporte, ya que esta técnica se basa en el conteo de ocurrencias posibles entre las combinaciones de ítems en la tabla de transacciones, y posee gran escalabilidad ya que realiza un barrido por la base de datos, por lo que puede operar sin mayores problemas con un número grande de datos (Naranjo y Montenegro, 2007).

Búsqueda de ítemsets frecuentes

La identificación de itemsets frecuentes es computacionalmente costosa ya que requiere considerar todas las combinaciones de los distintos ítems, resultando en una búsqueda exponencial. La figura 1, muestra el lattice de espacio de búsqueda resultante de E= {a, b, c, d} (Ceglar y Roddick, 2006). Para la búsqueda de los ítemsets frecuentes se emplean dos formas comunes de búsqueda en árbol: primero, a lo ancho (Figura 2) (BFS, por sus siglas en inglés), y segundo, en profundidad (DFS, por sus siglas en inglés), sobre árboles similares (Figura 3). Estos algoritmos trabajan por lo general de la siguiente manera: buscan un conjunto Ck de k-itemsets con alta probabilidad de ser frecuentes, llamémosles en lo sucesivo k-itemsets candidatos, comenzando por k=1. Se comprueba cuáles son frecuentes y a partir de estos se genera nuevamente un conjunto de candidatos de tamaño k+1, Ck+1. Este proceso se re-pite hasta que no se pueda generar un nuevo conjunto candidato. Tal estrategia garantiza que sean visitados todos los itemsets frecuentes, al mismo tiempo que se reduce el número de itemsets infrecuentes visitados (Danger y Berlanga, 2001).

Con la estrategia BFS el valor del soporte de los (k-1) itemsets son determinados antes de contar el soporte de todos los k-itemsets, ello le permite utilizar la propiedad arriba enunciada. Con la estrategia DFS no son conocidos todos los (k-1) itemsets, pero sí los necesarios (k-1) itemsets cuando se generan cada uno de los k-itemsets, pues trabaja recursivamente descendiendo por el árbol y siguiendo la estructura del segundo (Figura 3) (Danger y Berlanga, 2001).

Para contar el soporte de todos los conjuntos de ítems también se emplean por lo general dos mecanismos:

1. Determinar el valor del soporte contando directamente sus ocurrencias en la base de datos.

2. Determinar el soporte empleando la intersección entre conjuntos. Un tid es un identificador único de una transacción.

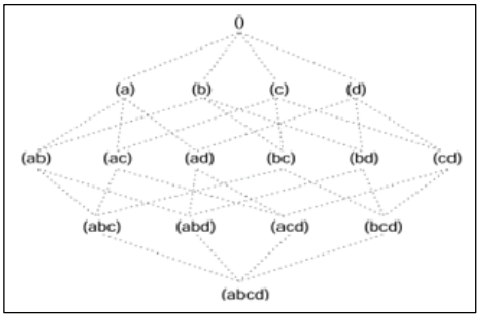

Figura 1. Lattice del espacio de búsqueda

Para cada ítem se genera un tidlist, el conjunto de identificadores que se corresponden con las transacciones que contienen a este ítem. Existe también para cada conjunto de ítems X un tidlist denotado como X.tidlist. El tidlist de un candidato C = X U Y es obtenido por la intersección de los tidlist de los conjuntos de ítems de X e Y, o sea, C.tidlist = X.tidlist ∩ Y.tidlist. Los algoritmos más comunes para el cálculo de los ítemsets frecuentes se muestran en la Figura 4.

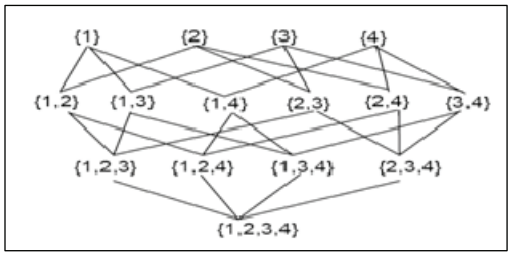

Figura 2. Árbol utilizado en la estrategia BFS

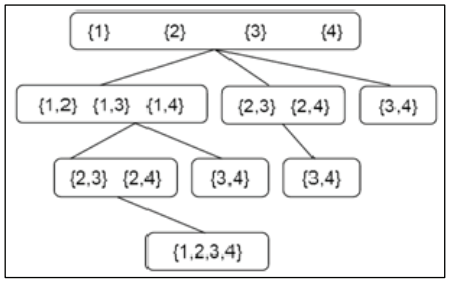

Figura 3. Árbol usado en los algoritmos con estrategia DFS

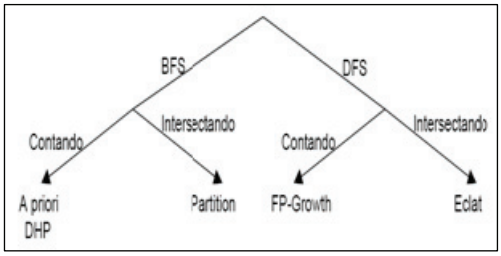

Figura 4. Algoritmos para el cálculo de itemsets frecuentes

Selección del algoritmo

Para la técnica de reglas de asociación existen una serie de algoritmos tales como: A priori, DHP, Partition, FP-Growth y Eclat, de los cuales se seleccionó el más adecuado teniendo en cuenta los siguientes criterios (Danger y Berlanga, 2001):

Inserciones a la base de datos: es importante que los algoritmos minimicen el recorrido por la base o bodega de datos, pues el número de reglas crece exponencialmente con el de ítems considerados, lo cual afecta el rendimiento del algoritmo cuando se accede constantemente a la base o bodega de datos.

Costo computacional: es importante que el algoritmo no realice un gran número de operaciones.

Tiempo de ejecución: se desea que el tiempo utilizado para la generación de reglas sea razonable.

Rendimiento: es importante que el algoritmo realice las operaciones y procesos de forma eficiente.

Se revisó cada uno de los algoritmos mencionados con los criterios definidos:

A priori

Este algoritmo busca primero todos los conjuntos frecuentes uñitarios (contando sus ocurrencias directamente en la base de datos), se mezclan estos para formar los conjuntos de ítems candidatos de dos elementos y seleccionan entre ellos los frecuentes. Considerando la propiedad de los conjuntos de ítems frecuentes, se vuelve a mezclar estos últimos y se seleccionan los frecuentes (hasta el momento ya han sido generados todos los conjuntos de ítems frecuentes de tres o menos elementos). Así sucesivamente se repite el proceso hasta que en una iteración no se obtengan conjuntos frecuentes (Agrawal y Srikant, 1994).

Inserciones en la base de datos: este algoritmo busca todos los conjuntos frecuentes unitarios contando sus ocurrencias directamente en la base de datos, por lo tanto se realizan varias pasadas en dicha base.

Costo computacional: el conteo de soporte de los candidatos es costoso debido a que el número de subconjuntos frecuentes en cada candidato es cada vez mayor y el de niveles en el árbol hash de candidatos se incrementa.

Tiempo de ejecución: hay que hacer tantos recorridos como sea necesario para encontrar todos los ítems frecuentes, por lo que no solo es costosa la solución en memoria, sino además en tiempo.

Rendimiento: este algoritmo tiene algunas mejoras para el rendimiento, entre ellas está la de reducir el número de ítems que contienen subconjuntos infrecuentes, aunque posteriormente al mezclar pares de conjuntos frecuentes con k-2 elementos iguales hay que verificar si todos los subconjuntos de k-1 elementos pertenecen al conjunto de itemsets frecuentes, con lo cual mejora el rendimiento (Danger y Berlanga, 2001).

DHP (Poda y hashing directa)

Este algoritmo emplea una técnica de hash para eliminar los conjuntos de ítems innecesarios para la generación del próximo conjunto de ítems candidatos (Park, Chen y Yu, 1997). Cada (k+1)-itemset es añadido a una tabla hash en un valor hash dependiente

de las ocurrencias en la base de datos de los conjuntos candidatos de k elementos que lo formaron, o sea, dependiente del soporte de los conjuntos candidatos de k elementos. Estas ocurrencias son contadas explorando en las transacciones de la base de datos. Si el soporte asociado a un valor hash es menor que el soporte mínimo entonces todos los conjuntos de ítems de k+1 elementos con este valor hash no serán incluidos entre los candidatos de k+1 elementos en la próxima pasada.

Inserciones en la base de datos: este algoritmo emplea una tabla hash con un valor hash dependiente de las ocurrencias en la base de datos de los conjuntos candidatos, por lo cual se requiere hacer varias inserciones en la base de datos.

Costo computacional: se emplea una tabla hash para reducir el número de candidatos; el espacio de memoria empleado por la tabla compite con el necesitado por al árbol hash, de ahí que, con tablas hash muy grandes (para reducir la cantidad de falsos positivos) la memoria se vuelve insuficiente.

Tiempo de ejecución: este algoritmo requiere de varias pasadas a la base de datos para su funcionamiento, en cada pasada cuenta el soporte de cada ítem y coloca en la tabla hash los conjuntos de K - ítems de acuerdo al valor del soporte de cada uno de dichos conjuntos, lo que representa un costo en tiempo.

Rendimiento: en este algoritmo el número de candidatos que tienen igual valor hash está directamente relacionado con el tamaño de la tabla, por tanto el espacio de memoria empleado por la tabla compite con el necesitado por al árbol hash y la memoria se vuelve insuficiente, afectando el rendimiento del algoritmo (Danger y Berlanga, 2001).

Partition

Este algoritmo propone fraccionar la base de datos en tantas partes como fueren necesarias para que todas las transacciones en cada partición estén en la memoria (Savesere, Omiecinski, y Navatie, 1995). En contraste con los vistos hasta el momento, este algoritmo recorre la base de datos sólo dos veces. En la primera, cada partición es minada independientemente para encontrar los conjuntos de ítems frecuentes en la partición y luego se mezclan estos para generar el total de los conjuntos de ítems candidatos. Muchos de estos pueden ser falsos positivos, pero ninguno falso negativo (notemos que si existen m particiones, para que un itemset tenga soporte s debe poseer un soporte no menor que s/m al menos en una de las m particiones, los conjuntos candidatos serán por tanto los que cumplan esta condición). En la segunda pasada se cuenta la ocurrencia de cada candidato, aquellos cuyo soporte es mayor que el mínimo soporte especificado se retienen como conjuntos frecuentes. Este algoritmo emplea el mecanismo de intersección entre conjuntos para determinar su soporte, en este caso cada ítem en una partición mantiene la lista de los identificadores de las transacciones que contienen a dicho ítem.

Inserciones en la base de datos: sólo requiere dos pasadas a través de la base de datos, para el cálculo de los ítems frecuentes.

Costo computacional: es relativamente más eficiente que el A priori pero tiene dos problemas: el costo en memoria es mayor, pues requiere almacenar para cada ítem el conjunto de transacciones que lo contiene; y además el cálculo del soporte de un candidato obtenido por la unión de dos conjuntos frecuentes obliga a intersectar los dos conjuntos.

Tiempo de ejecución: este algoritmo mantiene la base de datos en memoria y evita las operaciones de E/S en disco, divide la base de datos en tantas partes como sean necesarias para que todas las transacciones queden en la memoria, al reducir las operaciones de entrada/salida disminuye el tiempo de ejecución.

Rendimiento: este algoritmo, al igual que el A priori, mejora el rendimiento al reducir el número de ítems que contienen subconjuntos infrecuentes, aunque posteriormente al mezclar pares de conjuntos frecuentes con k-2 elementos iguales hay que verificar si todos los subconjuntos de k-1 elementos pertenecen al conjunto de itemsets frecuentes (Danger y Berlanga, 2001).

Eclat

Los algoritmos del tipo Eclat, fueron introducidos en (Zari, Parthasabathy, Oghiara y Li, 1998), al igual que el Partition, reducen la cantidad de operaciones de E/S, aunque esta vez atravesando la base de datos sólo una vez. Se basan en realizar un agrupamiento (*clustering*) entre los ítems para aproximarse al conjunto de ítems frecuentes maximales y luego emplean algoritmos eficientes para generar los ítems frecuentes contenidos en cada grupo. Para el agrupamiento proponen dos métodos que son empleados después de descubrir los conjuntos frecuentes de dos elementos: el primero, por clases de equivalencia: esta técnica agrupa los itemsets que tienen el primer ítem igual. El segundo, por la búsqueda de cliques maximales: se genera un grafo de equivalencia cuyos nodos son los ítems, y los arcos conectan los ítems de los 2-itemsets frecuentes, se agrupan los ítems por aquellos que forman cliques maximales.

Inserciones en la base de datos: este algoritmo reduce la cantidad de operaciones de entrada/salida atravesando la base de datos sólo una vez.

Costo computacional: es más eficiente que el A priori, sin embargo presenta el mismo problema que el Partition: el costo en memoria es mayor, pues requiere almacenar para cada ítem el conjunto de transacciones que lo contiene; y además el cálculo del soporte de un candidato obtenido por la unión de dos conjuntos frecuentes obliga a intersectar los dos conjuntos.

Tiempo de ejecución: este algoritmo se basa en realizar un agrupamiento (*clustering*) entre los ítems, lo que infiere en el tiempo de ejecución.

Rendimiento: el realizar tareas de agrupamiento requiere de pasos adicionales en su funcionamiento, sin embargo computacionalmente es más eficiente que el A priori (Danger y Berlanga, 2001).

Los algoritmos mencionados se basan en la estrategia del algoritmo A priori, por lo tanto todos ellos presentan generación de candidatos para seleccionar las reglas de asociación. En el A priori, cuando la base de datos presenta gran cantidad de ítems frecuentes, grandes patrones, o mínimas medidas de soporte, el algoritmo presenta los siguientes problemas (Han, Pei, Yin, 2000):

-Es costoso manejar una gran cantidad de conjuntos candidatos. Por ejemplo, para describir patrones de 100 ítems tal como $\{a_1, a_2, \ldots a_{100}\}$, es necesario crear cerca de 10^{30} candidatos, que representa un alto costo computacional sin importar la técnica aplicada.

-Es tedioso repetir este proceso para comparar los candidatos en búsqueda de concordancia en la base de datos, especialmente aquellos patrones considerados como largos.

Es por esto que para solucionar el problema de generación de candidatos se plantea un algoritmo que no requiere generación de candidatos y mejora el rendimiento de esta técnica, llamado FP-Growth.

FP-Growth

Este algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones llamada Frequente Pattern Tree (Borgelt, 2005) (Han, Pei, Yin, 2000). La idea básica del algoritmo FP-Growth puede ser descrita como un esquema de eliminación recursiva: en un primer paso de preprocesamiento se borran todos los ítems de las transacciones que no son frecuentes individualmente o no aparecen en el mínimo soporte de transacciones, luego se seleccionan todas las transacciones que contienen al menos un ítem frecuente, se realiza esto de manera recursiva hasta obtener una base de datos reducida. Al retorno, se remueven los ítems procesados de la base datos de transacciones en la memoria y se empieza otra vez, y así con el siguiente ítem frecuente. Los ítems en cada transacción son almacenados y luego se ordena descendentemente su frecuencia en la base de datos.

Después de que se han borrado todos los ítems infrecuentes de la base de datos de transacciones, se pasa al árbol FP. Un árbol FP es básicamente de prefijos para las transacciones, esto es: cada camino representa el grupo de transacciones que comparten el mismo prefijo, cada nodo corresponde a un ítem. Todos los nodos que referencian al mismo ítem son referenciados juntos en una lista, de modo que todas las transacciones que contienen un ítem específico pueden encontrarse fácilmente y contarse al atravesar la lista. Esta lista puede ser accesada a través de la cabeza, lo cual también expone el número total de ocurrencias del ítem en la base de datos.

Inserciones en la base de datos: este algoritmo no requiere de la generación de candidatos, por lo tanto, precisa de pocos accesos a la base de datos (Borgelt, 2005).

Costo computacional: el algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones, por lo tanto no necesita de la creación de un árbol de prefijos; sin embargo, la creación de dicho árbol no requiere de un costo computacional elevado (Han y Kamber, 2002).

Tiempo de ejecución: este algoritmo busca patrones frecuentes con una corta búsqueda recursiva de prefijos, lo que en tiempo de ejecución es muy superior al del A priori, ya que no requiere de constantes accesos a la base de datos (Borgelt, 2005).

Rendimiento: puede generar un árbol FP-Tree de una base de datos proyectada si el árbol inicial no se puede alojar completamente en la memoria principal, lo que le permite adecuarse a los recursos disponibles (Han y Kamber, 2002).

De acuerdo a lo anterior, se decide implementar el algoritmo FP-Growth ya que tiene ventajas operacionales sobre los otros al no necesitar de la generación de ítems candidatos y ser computacionalmente más rápido.

Entre las razones por la que se seleccionó este algoritmo tenemos que requiere de pocos accesos a la base o bodega de datos. Este algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones; sin embargo, la creación de dicho árbol no requiere de un costo computacional elevado. El algoritmo busca patrones frecuentes con una corta búsqueda recursiva de prefijos, lo que en tiempo de ejecución es muy superior al A priori, ya que no requiere constantes accesos a la base o bodega de datos (Han y Kamber, 2002).

Herramienta *software*

El problema a resolver es que el análisis de información basado en la canasta de mercado sea generado automáticamente por una herramienta que busque información relevante sobre el repositorio de datos de compras de los clientes (Kimbal y Ross, 2002) y que a partir de esas transacciones se pueda determinar la probabilidad de que un producto pueda ser comprado, a partir de otros productos relacionados.

Esta aplicación está principalmente constituida por: *Submódulo de preprocesamiento de datos*, en el que se hace la conversión de los datos a un modelo analítico para su posterior uso en el algoritmo de minería de datos; un *Submódulo de minería de datos*, el cual implementa el algoritmo de minería de datos seleccionado; un *Submódulo de análisis de resultados* encargado de la interpretación de los resultados obtenidos por el algoritmo y la visualización de dichos resultados; y un *Submódulo de aplicación de resultados*, que se encarga de aplicar las sugerencias escogidas en la base de datos de estrategias de publicidad.

Diseño de la aplicación

La aplicación se desarrolló utilizando la tecnología Microsoft .NET (Microsoft, 2003). La arquitectura planteada se basa en aplicaciones por capas. De las capas presentadas se han definido esencialmente tres: lógica de presentación, lógica del negocio y lógica de servicios (Figura 5).

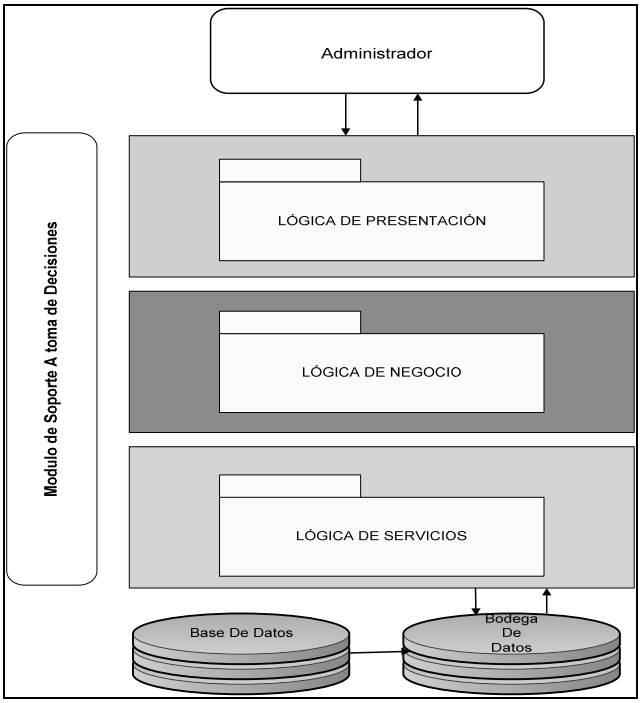

Figura 5. Arquitectura de la aplicación

Capa de lógica de presentación

Esta capa contiene los elementos de interfaz, que permite al usuario de la aplicación interactuar con la capa de lógica del negocio.

Entre los componentes de la capa lógica de presentación están las interfaces para la administración básica de la bodega a través de la aplicación; las interfaces para el análisis de información, en las cuales se configuran las medidas y parámetros para el algoritmo de minería de datos; las interfaces de análisis de resultados, en las cuales se despliegan los resultados obtenidos, y las interfaces de sesión para la aplicación.

Capa de lógica del negocio

Esta capa se encarga de manejar los detalles lógicos de cada uno de los servicios, dependiendo de las acciones del usuario sobre la aplicación. Básicamente, los componentes de la lógica de negocio de la aplicación son: Administración DW, encargada de la carga de las dimisiones y la tabla de hechos en el momento en que el administrador estime conveniente; el componente de análisis de información contiene las clases y métodos necesarios para el proceso de minería de datos y los métodos necesarios para la configuración de las medidas y los rangos de fechas a analizar por parte de la aplicación; el componente de análisis de resultados contiene los métodos de visualización y aplicación de resultados producto del proceso de minería de datos; y finalmente, el componente de sesión contiene la lógica referente a la autenticación del administrador para el uso de las funcionalidades de la aplicación desarrollada.

Capa de lógica de servicios

En esta capa encontramos los elementos que permiten interactuar con el servidor de base o bodega de datos, esta capa contiene los métodos necesarios para el acceso a datos y el preprocesamiento de los mismos. Sus componentes son: el de acceso a datos, el cual contiene los métodos y clases necesarios para el acceso al motor de base de datos; el de acceso a DW contiene los métodos necesarios para el acceso a la bodega de datos, y el de preprocesamiento contiene los métodos de preprocesamiento de los datos para su posterior uso en el algoritmo.

Diseño de la bodega de datos

Para este caso se asumió un modelo multidimensional, donde los datos se organizan en torno a hechos que tienen unos atributos o medidas que pueden verse según ciertas dimensiones (Hernández, Ramírez y Ferri, 2004). Cada dimensión debe tener una tabla asociada, llamada tabla dimensión; un modelo de datos multidimensional está organizado alrededor de un tema central, como ventas, por ejemplo.

En la Figura 6 se observa el modelo estrella realizado para la bodega de datos, el cual almacena la información correspondiente al registro de compras realizado, en ella se observa la tabla de hechos, en la cual se registran los pedidos de compra; además se observan las tablas dimensiones, en las cuales se almacenan los datos correspondientes a la fecha de compra e información del cliente, de los productos y las empresas.

Diseño del algoritmo en la aplicación

En la Figura 7 se visualiza el diagrama de clases (Rumbaugh, Jacobson y Booch, 1999) que nos muestra su relación para una consulta general. La aplicación funciona de la siguiente forma: se empieza con la clase **PreProcess**, que hace uso del método *RetornarFechas*, el cual retorna la fecha inicial del rango de fechas necesario para la consulta, luego continúa con el método *RetornarFechas2* que como parámetro de entrada la fecha inicial de la consulta y retorna un valor de fecha mayor a la de inicio; dichos valores son desplegados en los componentes de la lógica de presentación correspondientes.

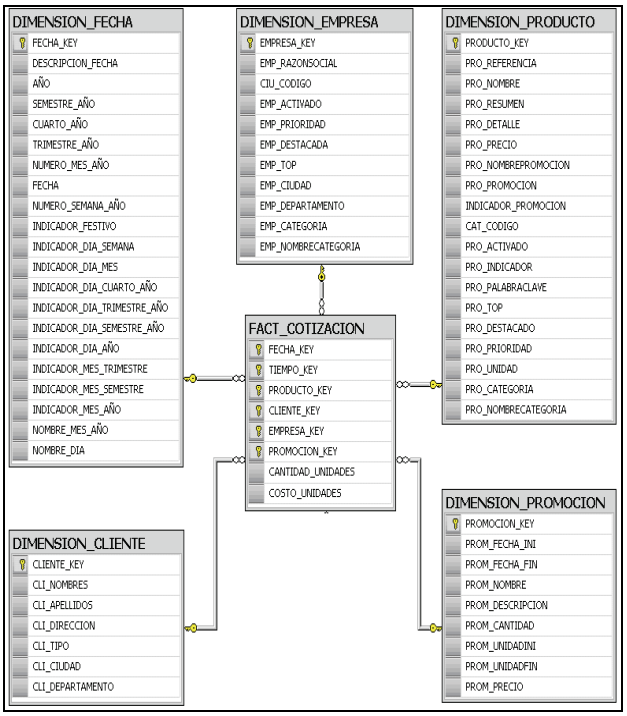

Figura 6. Diseño de la bodega de datos

Posteriormete se establecen los valores de confianza y soporte para la consulta; una vez que se ha realizado esto se invoca al método *SeleccionRangoFechas* de la clase **Preprocess**, encargado del preprocesamiento, luego se acude al método *ordenarDatosEntrada* de la misma clase, el cual se encarga de ordenar los datos de una forma específica para el proceso de minería de datos respectivo.

Enseguida se procede al ingreso de los datos preprocesados para el proceso de minería de datos, invocando al método de la clase **FPgrowth** llamado *InitFPgrowth*, y luego el de la clase **FPtree** *SetConfianza* para la confianza respectiva; acto seguido, a un método llamado *SetSoporte*, para el soporte respectivo, *Entrada DataSet* para la carga de los datos, *EntradaDatosOrden* para el proceso de minería de datos; *EntradaDatosYSoporte* de la clase **ReglaMining** para seleccionar los ítems que cumplen con el soporte mínimo; *NumeroItemSets* de la clase **ReglaMining**, el cual llena una estructura de búsqueda de reglas del proceso de minería de datos y establece sus dimensiones; *CrearFPtree* de la clase **FPtree** encargado de la creación de la estructura de árbol correspondiente; *startMining,* de la clase **FPtree**, dentro del que se despliegua el método *generarRAs* de la clase **FPtree**, encargado de la generación de las reglas de asociación para el proceso de minería de datos; y el método *SalidaRAs* de la clase **ReglaMining**, encargado de almacenar las reglas generadas en una estructura de tipo vector.

Ejecución de pruebas

Para el desarrollo de las pruebas se optó por la escogencia de una base de datos de prueba que contenía cerca de 250.000 registros de compra de los clientes en los años 1997 y 1998 de la compañía FoodMart. Con esta base de datos se realizaron pruebas de rendimiento, funcionalidad y confiabilidad con los registros de compras que contenía. Los registros estaban de la forma número de factura,

fecha de compra, código de los artículos, nombre de los artículos, cantidad y precio.

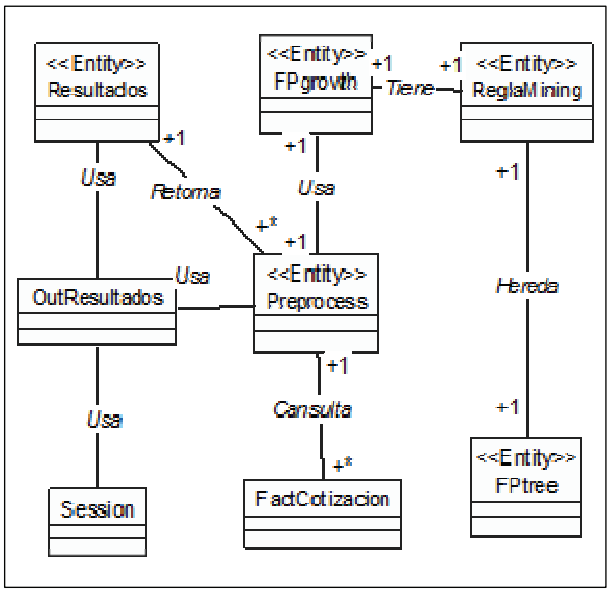

Figura 7. Diagrama de clases consulta de minería de datos

Pruebas de funcionalidad

Se corrigieron los errores presentes en las clases, realizando una prueba de funcionalidad en la aplicación (Pressman, 2005). Luego se efectuó una prueba general tomando la base de datos de prueba, con una confianza del 20% y un soporte del 1%, generando 120 reglas de asociación (ejemplo: si salsa de tomate Plato, entonces, queso bajo en grasa Booker). Los resultados los podemos apreciar en la Figura 8.

Figura 8. Resultados del análisis

Pruebas de rendimiento

Se hizo un contraste con el algoritmo A priori que se encuentra implementado en la herramienta Weka (Waikato, 2007), para observar el rendimiento que tenían ambos algoritmos frente a los mismos datos. El punto central de esta prueba era el de confrontar los tiempos de ejecución de un algoritmo que genera candidatos frente a otro que no lo hace. Se observó que FP-Growth demostró ser más rápido que el algoritmo A priori (Figura 9). El eje de las X muestra el número de registros de la base de datos que se procesa, y en el eje de las Y, el tiempo gastado en la ejecución por las dos herramientas.

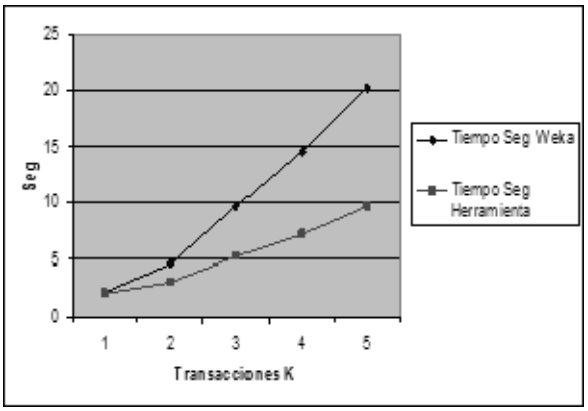

Figura 9. Prueba de rendimiento

Prueba de confiabilidad

Posteriormente se llevó una prueba de confiabilidad de los resultados con Weka y TariyKDD (BerliOS Developer, 2007) que permite encontrar reglas de asociación. Esta prueba se hace con el objetivo de determinar el grado de confiabilidad de las reglas generadas que tiene la herramienta desarrollada, frente a otras herramientas que hacen lo mismo. Para determinar el grado de efectividad de la herramienta desarrollada se hizo uso de la siguiente fórmula:

CDOHP = Cantidad de datos obtenidos de la herramienta de prueba.

CDOHD = Cantidad de datos obtenidos por la herramienta desarrollada que son similares a las demás.

Efectividad = (CDOHD/ CDOHP)*100

La prueba fue practicada con la base de datos de prueba, con un soporte del 20% y una confianza del 40%, donde se obtuvieron los siguientes resultados: con la herramienta desarrollada se obtuvieron 62 reglas; con la herramienta TariyKDD, 64; y con Weka, 64 reglas.

Las pruebas reflejaron que con TariyKDD se halla un mayor grado de similaridad de reglas encontradas que con la herramienta Weka (Tabla 2).

Tabla 2. Comparación de los resultados

Herramientas de Prueba	*Weka*	*TariyKDD*
Reglas Encontradas	64	64
Reglas similares Herramienta desarrollada	49	55
Efectividad	76.5625	85.9375

Conclusiones

Las reglas de asociación como técnica de minería de datos ofrecen muy buena alternativa para el análisis de canasta de mercado, per-

mitiendo encontrar patrones de conducta para apoyar la toma de decisiones en marketing.

Obtener un modelo soportado en las tecnologías de la información y las comunicaciones que simule o permita predecir el comportamiento de los clientes al hacer sus compras, propone una clara ventaja competitiva para las empresas.

La sistematización de técnicas de minería de datos como las reglas de asociación permite abordar problemas generales a entornos particulares.

El uso del algoritmo FP-Growth, uno de los más rápidos y eficientes para establecer reglas de asociación, facilita la realización del proceso de selección de dichas reglas en forma rápida, ya que permite encontrarlas sin selección de candidatos.

El desarrollo rápido de tecnologías como .NET provee muchas ventajas en desarrollo de aplicaciones web.

El desarrollo de herramientas software para entornos locales nos van a permitir abrir camino para que más empresas con pocos recursos utilicen estas tecnologías, convirtiéndose en una ventaja competitiva para las mismas.

Como trabajo futuro se plantea realizar un análisis de sensibilidad de las reglas encontradas, para verificar que ciertos elementos suyos no sean de gran aporte para el soporte y la confianza de las mismas y por tanto se puedan descartar, con el fin de facilitar su análisis. Por otro lado, se podría tomar en cuenta el estudio de técnicas híbridas para determinar si las reglas generadas son mejores.

Bibliografía

Agrawal, R., Srikant, R., Fast Algorithms for Mining Association Rules,Proc. 20th Int. Conf. Very Large Data Bases, VLDB,1994.

BerliOS Developer., Plataforma para soportar el proceso de Descubrimiento de Conocimiento en Bases de Datos-TariyKDD., febrero, 2007, https://developer.berlios.de/projects/tariykdd/

Borgelt, C., Frequent Pattern Mining., Department of Knowledge Processing and Language Engineering -School of Computer Science, OttovonGuericke - University of Magdeburg, 2005.

Castañeda G, J. A., Rodríguez M, M. A., La Minería de Datos como herramienta de Marketing: Delimitación y Evaluación del resultado., Facultad de CC. EE., Departamento de Comercialización e Investigación de mercados, Universidad de Granada, España, 2005.

Cabena, H., Stadler, V. Z., Discovering Data mining From Concept To Implementation., Prentice Hall PTR (ed), Upper Saddle River, 1998.

Ceglar, A., Roddick, J., Association Mining., Flinders University of South Australia, ACM Computing Survey, Vol. 38, No. 2, Julio, 2006.

Clementine-SPSS., Descubra soluciones con Clementine que de otra manera no podría., 2008. http://www.spss.com/la/productos/clementine/clementine.htm.

Danger M., R., Berlanga Ll, R., Informe técnico: Búsqueda de Reglas de Asociación en bases de datos y colecciones de textos., Departamento de Computación, Universidad de Oriente, Santiago de Cuba, 2001.

DeLuca, M. P., Plan para enfocar las campañas Bancarias utilizando Datamining., tesis presentada a la Universidad de Chile, Santiago de Chile, para optar al grado de Magíster en Gestión y Dirección de Empresas., 2006. http://www.cybertesis.cl/ tesis/uchile/2006/deluca_m/html/index-frames.html.

Facca F.M., Lanzi P. L., Mining interesting knowledge from weblogs: a survey., Data and knowledge Engineering, 2004, www.elsevier.com/locate/datak.

Han, J., Pei, J., Yin, Y., Mining Frequent Patterns without Candidate Generation., ACM SIGMOD Record, Vol. 29, No. 2, 2000.

Han J., Kamber M., Data Mining: Concepts and Techniques., Simon Fraser University – Morgan Kaufmann Publishers (ed), 2002.

Hernandez, J., Ramirez M. J., Ferri, C., Introducción a la Minería de Datos., Pearson Prentice Hall, (ed), ISBN: 84 205 4091 9, 2004.

Larose, D. T., Discovering Knowledge in Data: An Introduction to Data Mining., John Wiley & Sons(ed), 2004.

Kimball, R., Ross, M., The Data Warehouse Toolkit The Complete Guide to Dimensional Modeling., Second Edition, McGraw Hill (ed.), 2002.

Microsoft., Introducción a la Tecnología .NET., 2003. http://www.microsoft.com/latam/windowsserver2003/evaluation/overview/dotnet/default.mspx

Naranjo, R., Montenegro, R., Selección de una Técnica de Minería de Datos para la correlación de productos en el Comercio Electrónico tipo B2C., Revista Gerencia Tecnológica Informática, Vol. 6, No. 13, 2007.

Park, J. S., Chen, M. S., Yu, P. S., Using a Hash-Based Method with Transaction Trimming for Mining Association Rules., IEEE Transactions on Knowledge and Data Engineering, Vol. 9, No. 5, Sept./Oct, 1997.

Pressman, R., Ingeniería del Software, un enfoque práctico., McGraw Hill (ed), ISBN: 970-10-5473-3, 2005.

Rumbaugh, J., Jacobson, I., Booch, G., El Lenguaje Unificado de Modelado, Manual de Referencia, Addison Wesley (ed), ISBN. 0-201-30998-X, 1999.

Savesere, A., Omiecinski, E., Navatie, S., An efficient algorithm for mining association rules in large databases., In Proceedings of the 21st International Conference On Very Large Data Bases, 1995.

Tapscott, D., Lowy, A., Ticoll, D., La Era de los Negocios Electrónicos., McGraw Hill (ed), ISBN: 958-600-975-0, 2000.

The CRISP-DM Consortium., CRISP-DM Step by step data mining guide., 2000 -. Documento. http://www.crisp-dm.org/CRISPWP-0800.pdf ,2007.

Waikato ML Group., The waikato environment for knowledge analysis., The University of Waikato, http://www.cs.waikato.ac.nz/ml/weka, 2007.

Zari, M. J., Parthasabathy, S., Oghiara, M., Li, W., New Algorithms for fast discovery of association rules., In 3rd International Conference on Knowledgement Discovery and Data Mining, 1998.