UNIVERSIDAD NACIONAL AGRARIA LA MOLINA

FACULTAD DE ECONOMÍA Y PLANIFICACIÓN

"IDENTIFICACIÓN DE LAS REGLAS DE ASOCIACIÓN UTILIZANDO LOS ALGORITMOS SECUENCIALES SPADE Y GSP"

PRESENTADO POR

FERNANDO THOMAS LÉVANO CHIROQUE

TESIS PARA OPTAR EL TÍTULO DE

INGENIERO ESTADÍSTICO INFORMÁTICO

Lima – Perú

2019

UNIVERSIDAD NACIONAL AGRARIA LA MOLINA

FACULTAD DE ECONOMÍA Y PLANIFICACIÓN

"IDENTIFICACIÓN DE LAS REGLAS DE ASOCIACIÓN UTILIZANDO LOS ALGORITMOS SECUENCIALES SPADE Y GSP"

PRESENTADO POR

FERNANDO THOMAS LÉVANO CHIROQUE

TESIS PARA OPTAR EL TÍTULO DE

INGENIERO ESTADÍSTICO INFORMÁTICO

SUSTENTADA Y APROBADA ANTE EL SIGUIENTE JURADO

.....	.....
MS. Grimaldo José Febres Huamán Presidente	Mg. Jesús Walter Salinas Flores Asesor
.....	.....
MS. Jaime Carlos Porras Cerrón Miembro	MS. Carlos López de Castilla Vasquez Miembro

Lima – Perú
2019

.ÍNDICE GENERAL

I. Introducción 1
II. Revisión de Literatura 4
2.1 Patrones 4
2.2 Reconocimiento Estadístico de Patrones 5
2.3 Minería de Datos 6
2.4 Market Basket 8
2.5 Reglas de Asociación 9
2.6 Medidas de las Reglas de Asociación 9
2.7 Árbol Lexicográfico 12
2.8 Datos Secuenciales 12
2.9 Medidas de ls Reglas de Asociación Secuencial 14
2.9.3 Comtrastes de Ocurremcias de una Secuencia “GAP” 15
2.10 Algoritmos Secuenciales 16
2.11 Antecedemtes de la Investigación 42
III. Materiales y Métodos 45
3.1 Materiales 45
3.2 Metodología de la Investigación 45
3.2.1 Tipo de Investigación 45
3.2.2 Diseño de la Investigación 45
3.2.3 Formulación de la Hipótesis 45
3.2.4 Población 46
3.2.5 Identificación de las Variables 46
3.2.6 Metodología Aplicada 46
IV. Resultados y Discusión 47
V. Conclusiones 57
VI. Recomendaciones 59
VII. Referencias Bibliográficas 60
VIII. Anexos 63

ÍNDICE DE FIGURAS

Figura 1. Modelo para un reconocimiento estadístico de patrones 6
Figura 2. Minería de datos y el proceso KDD 6
Figura 3. Árbol lexicográfico 12
Figura 4. Efectos del contrase de máxia ocurrencia Gap 16
Figura 5. Patrones de secuencias a través de la técnica GSP 18
Figura 6. Árbol GSP formado con el primer escaneo 20
Figura 7. Árbol GSP formado con el segundo escaneo 22
Figura 8. Árbol GSP formado con el tercer escaneo 23
Figura 9. Transacciones en formato vertical 24
Figura 10. Algoritmo SPADE 25
Figura 11. Transformación del conjunto de datos de vertical a horizontal 26
Figura 12. Clases equivalentes del algoritmo SPADE 27
Figura 13. Descomposición recursiva de la clase [$\mathbf{30}$] en pequeñas sub-clases vía $\boldsymbol{\theta k}$ 28
Figura 14. Representación en formato vertical 30
Figura 15. Árbol SPADE formado con el primer escaneo 32
Figura 16. Árbol SPADE formado con el segundo escaneo 40
Figura 17. Árbol SPADE formado con el tercer escaneo 41
Figura 18. Comparación entre los algoritmos en estudio según las reglas generadas 55

RESUMEN

Hoy en día los datos secuenciales son de gran importancia, debido a que pueden encontrarse en distintas aplicaciones como: registros de ventas, registros médicos de pacientes, registros webs, bolsa de valores, base de datos en geofísica, etc. Es por esta razón que se han estudiado las tendencias o patrones a través del tiempo con el algoritmo secuencial. En esta investigación se estudia con mayor profundidad el algoritmo secuencial Sequential Pattern Discovery using Equivalent Class (SPADE, por sus siglas en inglés), debido a que usa una eficiente búsqueda de las reglas que se generan por el algoritmo, reduciendo el número de reglas y costos de la memoria. En primer lugar, se ilustra el procedimiento para obtener las reglas de asociación y luego con un conjunto de datos se identifican las reglas de asociación computacionalmente. Por último se compara los resultados obtenidos con el algoritmo Generalized Sequential Patterns (GSP, por sus siglas en inglés), debido a que ambos algoritmos tienen el mismo enfoque. Uno de los resultados más resaltantes fue "el cliente compra pavo en un tiempo máximo de tres meses, dado que compró antes costilla de cordero". Los resultados que se obtuvieron sirven para incrementar las ventas del establecimiento a través de ventas cruzadas. El algoritmo SPADE permitió obtener reglas más completas que el GSP.

Palabras claves: Algoritmo SPADE, Algoritmo GSP, Minería de datos, Patrón Secuencial, Data Secuencial.

SUMMARY

Today, sequential data are of great importance, because they can be found in various applications such as: sales records, patient medical records, web registries, stock exchange, geophysics database, etc. Trends have been studied over time with the sequential algorithm. Sequential discovery of the pattern using the equivalent class (SPADE), by an efficient search of the rules that are generated by the algorithm, reducing the number of rules and costs of memory. First, Illustrates the procedure for obtaining the association rules, and then with a set of data, they identify the association rules computationally. Generalized sequential schemes (GSP), because both algorithms have the same approach. One of the most remarkable results was "the customer buys turkey in a maximum time of three months, since he bought before lamb rib". The results obtained are used to increase sales through cross-selling. The SPADE algorithm allowed to obtain more complete rules than the GSP.

Keywords: Data Mining, GSP Algorithm, Sequential Pattern, Sequential Data, SPADE Algorithm.

I. INTRODUCCIÓN

Un problema muy frecuente en el reconocimiento de patrones es la dificultad en la identificación de las tendencias en el tiempo con el fin de encontrar relaciones entre eventos. Un tipo especial de patrón en el tiempo es el denominado patrón secuencial que consiste en el descubrimiento de relaciones antes y después, en un conjunto de variables o atributos a través del tiempo. Un ejemplo de una regla de patrón secuencial en el contexto de las ventas al por menor es que el 65% de los clientes que compró una TV LCD, comprará un equipo de sonido después de dos meses.

El problema de descubrir todas las secuencias frecuentes en grandes conjuntos de datos se encuentra en que el espacio de búsqueda es extremadamente grande debido al número de reglas que se generan. Esto ocasiona un alto costo computacional debido a la gran cantidad de memoria en uso. Se conoce que en un supermercado existen miles de productos por lo que identificar todas las reglas de asociación ocasionaría que falle el sistema o el programa que procesa los datos. Por lo que se tiene la necesidad de encontrar un algoritmo que simplifique el espacio de búsqueda y encuentre solo las principales reglas de asociación secuencial.

Dentro de las reglas de asociación secuencial existen una gran cantidad de algoritmos que permiten obtenerlas, cada una con un procedimiento diferente, que sirven para /posteriormente poder calcular las reglas de asociación. Entre ellas se encuentran: A priori all, GSP, SPADE, FreeSpan, PrefixSpan, SSMiner, SPAM y MILE. Estas reglas obtenidas tienen que cumplir con los requisitos mínimos de soporte definidos por el investigador o usuario. El algoritmo que se estudió para identificar las reglas de asociación en la presente investigación es el SPADE propuesto por Mohammed Zaki.

Se comparó el algoritmo secuencial GSP y SPADE, utilizando el indicador: Número de reglas de asociación generadas. Se identificaron las reglas de asociación para encontrar patrones entre ítems en las transacciones de compras de un supermercado para ambos

conjuntos de datos: el utilizado por Agrawal & Srikant (1995) y el caso usado por Solano (2011). Se escogieron estos conjuntos de datos porque el primer caso de prueba fue propuesto por uno de los creadores de la rama del algoritmo secuencial y se usó para ilustrar el algoritmo. El motivo de uso del segundo caso de prueba fue porque se le aplicó el algoritmo GSP en una investigación pasada (Rojas, 2011) y se esperó obtener mejores resultados con el algoritmo propuesto.

1.1 JUSTIFICACIÓN DE LA INVESTIGACIÓN

En la actualidad se analizan las transacciones por cantidad de ítems vendidos y generalmente ofertan estos ítems cruzando con ítems complementarios, aquellos ítems que deben usarse con otros para safisfacer una necesidad, dejando de lado ítems que no son complementarios pero son altamente correlacionados y también el tiempo en que se hizo la transacción. Debido a esto, existe una gran cantidad de datos secuenciales que no son explotados, como por ejemplo: las transacciones de negocios, registros de telecomunicaciones, datos climáticos, registros clínicos de enfermedades y procesos de producción, es necesario analizar, explotar y tomar las mejores decisiones con este tipo de datos (Han & Kamber, 2006). Incluso aprender a predecir sub-secuencias de eventos poco frecuentes pero altamente relacionados es importante en el campo de la medicina donde es posible encontrar relaciones entre síntomas y enfermedades a través del tiempo.

SPADE no solo minimiza costos reduciendo los datos, también minimiza costos computacionales debido a que realiza una búsqueda eficiente. Adicionalmente, hay estudios donde corroboran lo planteado, demostrando que SPADE es más eficiente que otros algoritmos, como por ejemplo el GSP (Zaki M. , 2001).

1.2 ALCANCES

En este sentido, el presente trabajo de investigación contribuirá al conocimiento de la metodología de las reglas de asociación secuencial y poder así descubrir patrones secuenciales con menores costos computacionales. Dicho procedimiento responde a la necesidad de seleccionar transacciones, donde las secuencias de estos se encuentran generalmente ordenados a través del tiempo.

OBJETIVO GENERAL

- Desarrollar la metodología de las reglas de asociación utilizando el algoritmo SPADE y GSP.

OBJETIVOS ESPECÍFICOS

a. Identificar las principales reglas de asociación utilizando el algoritmo SPADE mediante ocurrencias de secuencias para un conjunto de transacciones de compra de un supermercado.
b. Comparar los resultados obtenidos de las transacciones de compra de un supermercado aplicando el algoritmo SPADE y el algoritmo GSP en cuanto a la cantidad de reglas generadas.

II. REVISIÓN DE LITERATURA

2.1 PATRONES

De manera general se define patrón como un conjunto de características de un objeto. El objetivo del reconocimiento de patrones es determinar las características de un conjunto de elementos.

Watanabe (1985), citado por Jain, P.W. Duin & Mao (2000) define patrón "como la oposición al caos; es una entidad, vagamente definida, que se le pudiera dar un nombre." Por ejemplo, un patrón podría ser una asociación de productos de un supermercado, una palabra manuscrita cursiva, un rostro humano, o una señal de voz. Dado un patrón, su reconocimiento / clasificación puede consistir en una de las dos tareas siguientes:

1) Clasificación supervisada: El patrón de entrada se identifica como un miembro de una clase predefinida (por ejemplo, el Análisis Discriminante).
2) Clasificación no supervisada: El patrón se asigna a una clase hasta ahora desconocida (por ejemplo, el Análisis de Conglomerados).

El interés en el área de reconocimiento de patrones ha tomado mayor importancia por la creación de aplicaciones a causa de los avances de la tecnología y presentan un gran reto computacional (ver Tabla 1). Estas aplicaciones incluyen la minería de datos, clasificación de documentos (buscar de manera eficiente documentos de texto), la previsión financiera, organización y recuperación de bases de datos multimedia y la biometría (identificación personal sobre la base de diversos atributos físicos tales como la cara y las huellas dactilares) (Jain et al., 2000).

Tabla 1. Ejemplos de aplicaciones en el reconocimiento de patrones

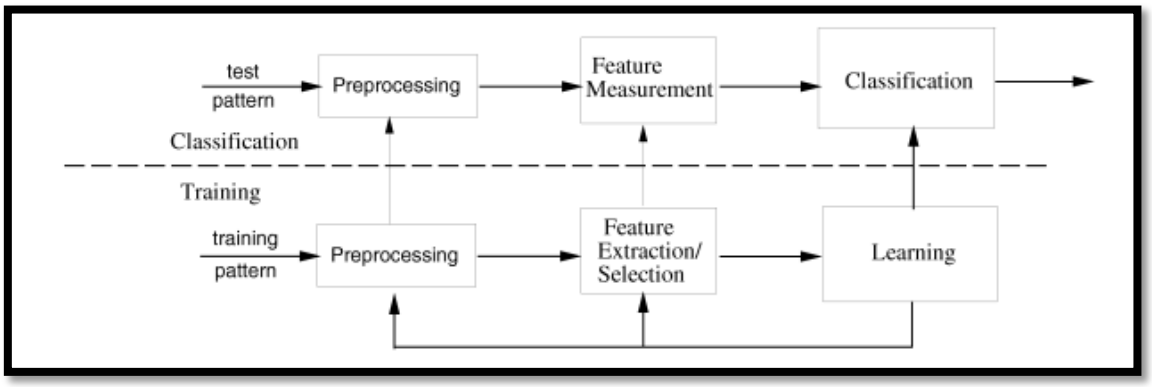
Problema de dominio	Aplicación	Patrones de entrada	Patrón de clase
Bioinformática	Análisis de secuencia	ADN/ secuencia de proteína	Conocer el tipo de patrón de genes
Minería de datos	Búsqueda de patrones significativos	Puntos en un espacio Multidimiensional	Clusters compactados y bien separados
Clasificación de documentos	Búsqueda en internet	Documentos de texto	Categorías semánticas (por ejemplo: negocios, deportes, etc.)
Análisis de imágenes de documento	Máquinas de lectura para ciegos	Documentos de imágenes	Caracteres alfanumércos, palabras
Automatización industrial	Inspección de plaza de circuitos impreso	Intensidad o rango de la imager	Productos naturales defectuosos/ no defectuosos
Database Multimedia	Búsqueda en internet	Videoclip	Género de videos (acción, diálogos, ect.)
Reconocimiento Biométrico	Identificación personal	Rostro, iris, huella digital	Autorización a los usuarios para acceder el control
Teledetección	Previsión de rendimiento de los cultivos	Imagen multiespectral	Patrón de crecimiento de cultivos
Reconocimiento de voz	Directoro telefónico, asistente operador	Onda de voz	Palabras habladas

Fuente: (Jain et al., 2000)

2.2 RECONOCIMIENTO ESTADÍSTICO DE PATRONES

El reconocimiento estadístico de patrones ha sido utilizado con éxito para diseñar una serie de sistemas de reconocimiento comercial. En reconocimiento estadístico de patrones, un patrón es representado por un conjunto de "d" características o atributos que se interpretan como un vector d dimensional, los conceptos de la teoría de decisión estadística se utilizan para establecer los límites de decisión entre las clases de patrones. El sistema de reconocimiento se hace funcionar en dos modos: formación (aprendizaje) y clasificación (pruebas) (ver Figura 1). El proceso comienza con el modo aprendizaje, donde se realiza la función del módulo del pre-procesamiento, que es segmentar el patrón de interés desde el fondo, eliminar el ruido, normalizar el patrón, y cualquier otra operación que contribuirá en la definición de una representación compacta del patrón. Luego en el modo de entrenamiento, se refiere a la de extracción de características, a continuación en la selección de modelos se descubren las características apropiadas para la representación de los patrones de entrada y el clasificador está entrenado para dividir el espacio de características. La trayectoria de retroalimentación permite a un diseñador optimizar las estrategias de pre-procesamiento y la extracción / selección de característica. A continuación se ubica en el modo de prueba, el clasificador entrenado asigna el patrón de entrada a una de las clases de patrones bajo consideración en base a las características medidas (Jain et al., 2000).

Figura 1. Modelo para un reconocimiento estadístico de patrones



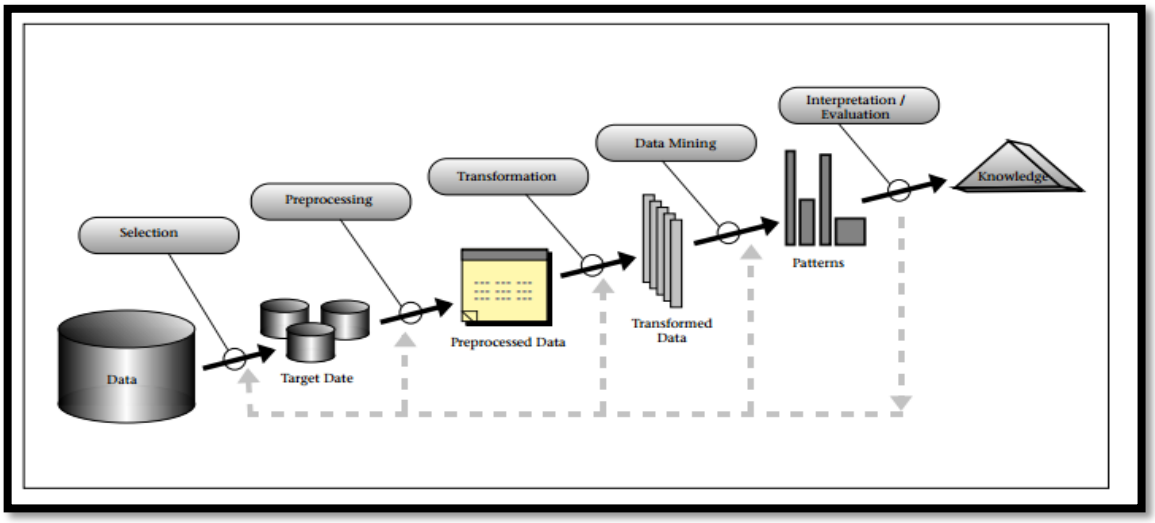
Fuente: (Jain et al., 2000)

2.3 MINERÍA DE DATOS

A continuación se muestran algunas definiciones:

La minería de datos se define como un paso esencial en knowledge discovery in databases (KDD) proceso que produce patrones o modelos de utilidad a partir de datos, como se muestra en la Figura 2. Los términos de KDD y minería de datos son diferentes. KDD se refiere al proceso general de descubrimiento de conocimiento útil a partir de datos. La minería de datos se refiere a descubrir nuevos patrones de una gran cantidad de datos en bases de datos, centrándose en los algoritmos para extraer conocimientos útiles (Fayyad et. al, 1996).

Figura 2. Minería de datos y el proceso KDD



Fuente: (Usama Fayyad, 1996)

La minería de datos es el descubrimiento de un modelo en datos; también se llama análisis exploratorio de datos, y descubre el conocimiento útil, válido, inesperado, y comprensible de los datos. Algunos de los objetivos son compartidos con otras ciencias, como: estadística, inteligencia artificial, aprendizaje de máquinas y reconocimiento de patrones (Gabrys, 2006).

La minería de datos se refiere al análisis de grandes cantidades de datos que se guardan en espacios de almacenamiento. Por ejemplo, los supermercados almacenan grandes cantidades de datos generados por nuestras compras. El código de barras proporciona a los establecimientos minoristas grandes cantidades de datos. Los encargados de las tiendas de comestibles y otras tiendas minoristas son capaces de procesar rápidamente nuestras compras, y el uso de las computadoras para determinar con exactitud los precios del producto. Estos mismos equipos pueden ayudar a las tiendas con su gestión de inventario, determinando instantáneamente la cantidad de unidades de cada producto en la mano. También son capaces de aplicar la tecnología informática para contactar a sus proveedores para que no se quede fuera de las cosas que queremos comprar. Las computadoras permiten contar con un sistema contable en la tienda para medir con mayor precisión los costos y determinar el beneficio que almacenan los accionistas. La minería de datos no se limita a los negocios y se ha utilizado mucho en el campo de la medicina, por ejemplo para incluir el diagnóstico de los registros de los pacientes. También es ampliamente utilizado por las empresas bancarias en otorgar tarjetas de créditos a los clientes, por las compañías de seguro y telecomunicaciones en la detección del fraude, por las compañías telefónicas y en lo emisores de tarjeta de crédito en la identificación de fuga de clientes. Pero la minería de datos no se limita a un análisis automatizado. El descubrimiento de conocimiento por los seres humanos se puede mejorar mediante herramientas gráficas y la identificación de patrones inesperados a través de la combinación del humano y la interacción del equipo (Olson & Delen, 2008).

La minería de datos puede ser usada en los negocios en varias ramas. Tres ejemplos son:

1) Perfiles de los clientes: Identificación de un subconjunto de los clientes más rentables para el negocio.
2) *Targeting*: Determinar las características de los clientes más rentables que han sido capturados por los competidores.

3) Market-baket análisis: Determinar los productos comprados por los consumidores que pueden ser utilizados para el posicionamiento de productos y para la venta cruzada.

2.4 MARKET BASKET

El market basket es el enfoque que se toman en las reglas de asociación en un contexto de ventas en un supermercado. El market basket se refiere a la metodología de la composición de compras de las canastas de productos adquiridos durante un evento dado. Esta técnica ha sido ampliamente aplicada a tiendas de comestibles (así como en otras tiendas de ventas al por menor, incluyendo restaurantes). Los datos del market basket en su forma más cruda podrían ser una lista de transacciones de compras de los clientes, lo que indica solo los ítems comprados (Olson & Delen, 2008). Las características de estos datos son:

- La gran cantidad de registros.
- Escasez (cada market basket contiene solo una pequeña porción de ítems distribuidos).
- Heterogeneidad (aquellos con gustos diferentes tienden a comprar un específico subconjunto de ítems).

El objetivo del análisis del market basket es identificar que productos se compran juntos. Analizando las transacciones se pueden identificar patrones de compras, como el que los útiles escolares y juguetes se compran en los meses de febrero y marzo. Esta información puede ser utilizada para determinar dónde colocar los productos en elalmacén, así como la gestión de inventario de ayudas. Presentaciones de productos y la dotación de personal se pueden planificar de forma más inteligente para momentos específicos del día, días de la semana, o días festivos.

Otra aplicación comercial es de cupones electrónicos, cupón sastrería valor nominal y el calendario de distribución utilizando la información obtenida del Market basket (Olson & Delen, 2008).

Según Svetina & Zupančič (2005) con el análisis del market basket se puede encontrar la respuesta a la siguiente pregunta ¿Qué mercancías se venden juntas en la misma transacción o al mismo cliente? Mediante este análisis, se intentará encontrar patrones recurrentes con el fin de ofrecer bienes relacionados entre sí y por lo tanto aumentar nuestras ventas. Se puede rastrear las ventas relacionadas en los diferentes niveles de clasificación de mercancía o en diferentes segmentos de clientes.

2.5 REGLAS DE ASOCIACIÓN

Se debe entender un ítem como un elemento, puede ser un evento, un producto, un síntoma, etc. Mientras que el ítemset se entiende como un conjunto de elementos. Por ejemplo un ítemset puede estar compuesto por {cerveza, yogurt, queso}. Este ítemset está conformado por los siguientes ítems: cerveza, yogurt y queso.

La regla de asociación es una de las técnicas más populares de minería de datos que busca encontrar tendencias entre los ítems y detectar cuándo la ocurrencia de un ítem está asociado a otro ítem en el mismo conjunto de datos, mediante la siguiente expresión:

$$\{\text{Pañal}, \text{Snacks}\} \rightarrow \{\text{Cerveza}\}$$

La declaración formal del problema de la regla de asociación se afirmó en primer lugar por Agrawal et al. (1993). Sea I = $\{i_1, i_2, i_3, \ldots, i_m\}$ un conjunto finito de m ítems distintos, T es la transacción que contiene al conjunto de elementos tales que T $\subseteq$ I, D es un conjunto de datos con diferentes registros de transacciones Ts. Una regla de asociación es una implicación en la forma de $X \rightarrow Y$, donde:

- $X, Y \subset I$ son conjuntos de ítems llamados ítemsets
- $X \cap Y = \phi$

En las reglas de asociación X es llamado antecedente, mientras que Y es llamado consecuente En el ejemplo mostrado {Pañal, Snacks} representan el antecedente, mientras que {cerveza} representa el consecuente.

2.6 MEDIDAS DE LAS REGLAS DE ASOCIACIÓN

2.6.1 DATOS TRANSACCIONALES

Según lo mostrado en las reglas de asociación, los datos se tienen que encontrar en formato transaccional. El formato transaccional muestra a cada ítem con su respectivo id del elemento en estudio por ejemplo: cliente, paciente, etc. Si el elemento está vinculado a más de un ítem, se repetirá su id. La Tabla 2 representa un conjunto de datos transaccionales que se empleó en Agrawal & Srikant (1995) que representa a cinco clientes comprando productos en un supermercado, los ítems están representado con su respectivo código. Por el momento no se considerará el tiempo como variable.

Tabla 2 Datos en formato transaccional

Elemento	Tiempo	Ítem
1	5	30
1	6	90
2	1	10
2	1	20
2	3	30
2	4	40
2	4	60
2	4	70
3	5	30
3	5	50
3	5	70
4	5	30
4	6	40
4	6	70
4	7	90
5	2	90

Fuente: Srikant & Agrawal (1995)

2.6.2 SOPORTE

El soporte se define como el porcentaje (fracción) de los registros que contienen $X \cup Y$ con el número total de registros del conjunto de datos. El recuento de cada elemento se incrementa en uno cada vez que el ítem se encuentre en diferentes T transacciones del conjunto de datos D durante el proceso de escaneado (Qiankun & Bhowmick, 2003).

$$Soporte\ (XY) = \frac{Número\ \ de\ veces\ que\ aparece\ XY}{Número\ total\ de\ transacciones\ \ en\ D} \quad (1)$$

Del conjunto de datos de la Tabla 1, se quiere hallar el soporte para el conjunto de ítems $\{30, 70\}$:

$$Soporte\ (30{,}70) = \frac{3}{5} = 0.6$$

2.6.3 CONFIANZA

Según Qiankun y Bhowmick (2003) la confianza se define como el porcentaje (fracción) del número de transacciones que contienen $X \cup Y$ al número total de registros que contienen X, en la que si el porcentaje supera el umbral de confianza, regla de asociación puede ser generado.

$$Confianza\ (X/Y) = \frac{Soporte(XY)}{Soporte(X)} \qquad (2)$$

Del conjunto de datos de la Tabla 1, se quiere hallar la confianza para la regla ítems $\{30, 70\} \rightarrow \{50\}$:

$$Confianza\ (30{,}70/50) = \frac{Soporte(\{30{,}70\} \cup \{50\})}{Soporte(\{30{,}70\})} = \frac{0.2}{0.6} = 0.33$$

Esto quiere decir que el 33% de las reglas del conjunto de datos que anteceden con los ítems 30 y 70, el consecuente resultará ser el ítem 50.

2.6.4 LEVANTAMIENTO

Según Kaur & Singh (2013) el levantamiento (lev) se define como el grado en el que la confianza es mayor (o menor) a lo esperado (umbral). Esto quiere decir que para valores mayores a uno, indica que ese conjunto aparece una cantidad de veces superior a lo esperado, mientras que para valores menores a uno, indica que esa conjunto aparece una cantidad de veces menor a lo esperado. La elevación de $X \rightarrow Y$ es:

$$Lev(X \rightarrow Y) = \frac{confianza(X \rightarrow Y)}{Soporte(Y)} \qquad (3)$$

Del conjunto de datos de la Tabla 1, se quiere hallar el levantamiento para la regla ítems $\{30, 70\} \rightarrow \{50\}$:

$$Lev(30{,}70 \rightarrow 50) = \frac{confianza(\{30{,}70\} \rightarrow \{50\})}{Soporte(\{50\})} = \frac{1}{0.2} = 5$$

2.6.5 SOPORTE MÍNIMO

Es el umbral mínimo de soporte, este umbral se fija por el usuario o investigador. Es la mínima probabilidad que una transacción contenga al antecedente X.

Se denota por la expresión MinSupp. Esto quiere decir, que se tomará en cuenta los subconjuntos cuyos soportes son al menos mayores que el soporte mínimo. A estos subconjuntos se les denomina ítemset frecuentes.

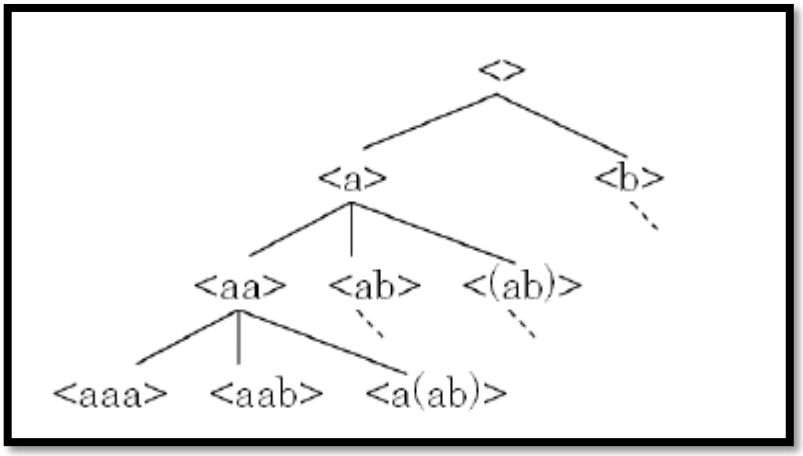
$$Supp(X \rightarrow Y) \geq MinSupp \qquad (4)$$

En el caso que se fije un soporte mínimo de 0.5 para el conjunto de datos de la Tabla 1, entonces el ítemset $\{30,70\}$ resulta ser un ítemset frecuente.

2.7 ÁRBOL LEXICOGRÁFICO

Según Bayardo (1998) y Ayres et al. (2002) citados por Yang (2008), el árbol lexicográfico es un tipo de dato abstracto que se utiliza para representar relaciones binarias (ver Figura 3), como las reglas de asociación, donde indican el orden de la clase paterna e hija, en el caso de las reglas de asociación indican los patrones y subpatrones. Estos árboles se pueden representar de manera ascendente o descendente, dependiendo donde se ecuentre la raiz (inicio) del árbol. La raíz se representa con el símbolo vacío: "{}" o " <> ".

Figura 3. Árbol lexicográfico



Fuente: (Yang, 2008)

2.8 DATOS SECUENCIALES

Se entiende como datos secuenciales a un listado de ítems ordenados la cual pueden ser productos, eventos, artículos y otros. Incluso no es necesario que los datos cuenten con el tiempo, basta que los ítems se encuentren ordenados, sin importar cuando, esta es una gran

diferencia entre una data secuencial y una data de series de tiempo (Solano, 2011).

Según Slimani & Lazzez (2013), un conjunto de datos de secuencia es un conjunto de elementos o acontecimientos ordenados, almacenados con o sin una noción concreta de tiempo. Cada ítemset contiene un conjunto de ítems que incluyen el mismo valor de tiempo de transacción.

Según Agrawal & Srikant (1995) se denota un ítemset I= ($I_1 I_2 \ldots I_m$), donde cada I_j representa un ítem. Una secuencia α es denotada como $<\alpha_1 \rightarrow \alpha_2 \ldots \rightarrow \alpha_n >$ donde α_i es un evento. Una secuencia con k ítems (k=$\sum_j |\alpha_j|$) es llamada una k-secuencia. Por ejemplo, ($A \rightarrow BCD$) es una 4-secuencia.

Una secuencia $<a_1 a_2 \ldots a_n >$ está contenida en otra secuencia $<b_1 b_2 \ldots b_n>$ si existe un conjunto de enteros $i_1 < i_2 < \ldots < i_n$ tales que $a_1 \subseteq b_1$, $a_2 \subseteq b_2$,…, $a_n \subseteq b_{in}$. Un conjunto de datos secuencial D es un conjunto de tuplas <Sid, S>, donde Sid representa el código del cliente y S es la secuencia del cliente. Los datos secuenciales pueden estar representados de forma transaccional, con las variables código del cliente "Sid", tiempo de la transacción "Tiempo" y el ítem vinculado "ítem" (ver Tabla 2). También se pueden representar de forma secuencial, con las variables código del cliente (Sid), Tiempo y la secuencia del cliente "S" que representa el orden de los ítems vinculados según el Sid. Para una visualización de los datos transaccionales y secuencial (ver Tabla 3) se empleó el conjunto de datos de Agrawal & Srikant (1995).

El análisis de datos secuencial juega un papel importante en el área de los itemsets frecuentes, porque gracias a los itemsets frecuentes se puede descubrir sub-secuencias frecuentes de un conjunto de datos de secuencias. En un conjunto de datos secuencial, una secuencia registra una serie de acciones o valores generados por una entidad. Por ejemplo, una secuencia puede registrar los artículos respectivos comprado por un cliente en el transcurso de varias operaciones realizadas en diferentes puntos en el tiempo (Ying-Ho, 2015).

Tabla 3 Datos en formato secuencial

Sid	Tiempo	S
1	5,6	<(30)(90)>
2	1,3,4	(<(10 20) (30) (40 60 70)>
3	5	<(30 50 70)>
4	5,6	<(30) (40 70) (90)>
5	2	<(90)>

Fuente: Agrawal & Srikant (1995)

2.9 MEDIDAS DE LAS REGLAS DE ASOCIACION SECUENCIAL

2.9.1 SOPORTE DE UNA SECUENCIA

Según Gouda (2011) y Sakurai et al. (2008) el soporte de una secuencia S, es el número total de tuplas del conjunto de datos que contienen esta secuencia, es decir, el soporte evalúa la frecuencia de los patrones. Este soporte se llama soporte absoluto, mientras que el soporte relativo se define como el porcentaje de tuplas en el conjunto de datos que contiene S y este soporte se utilizará a largo de esta investigación.

$$soporte(s) = \frac{fs_{(s)}}{N} \qquad (5)$$

Donde s es un patrón secuencial, $fs_{(s)}$ es la frecuencia del patrón s y N es el número total de datos secuenciales.

Del conjunto de datos de la Tabla 2, se quiere hallar el soporte para la secuencia $< \{30\}, \{90\} >$:

$$soporte(< (30)(90) >= \frac{2}{5} = 0.4$$

El soporte para la secuencia $< \{30\}, \{90\} >$ es 0.4 debido a los clientes 1 y 4

2.9.2 CONFIANZA DE UNA SECUENCIA

La confianza evalúa frecuencias de patrones en el caso de que se generen sub-patrones. La confianza puede evaluar las relaciones entre los patrones secuenciales y sus sub-patrones secuenciales. El método, basado en la confianza, descubre patrones secuenciales cuyas frecuencias están cerca de las frecuencias de sus sub-patrones secuenciales. Sin embargo, la confianza no satisface la propiedad Apriori. El método no puede descubrir de manera

eficiente patrones secuenciales con alta confianza. Por lo general, los métodos de extracción secuencial descubren patrones secuenciales con alto soporte y extraen patrones secuenciales con alto grado de confianza de los patrones. Estos métodos no aseguran el descubrimiento de todos los patrones secuenciales con alta confianza. Se tiene que establecer un umbral menor para el soporte para evitar perder los patrones secuenciales con alta confianza y los analistas tienden a gastar mucho tiempo examinando los patrones (Sakurai et al., 2008).

$$conf(s/s_p) = \frac{f_s(s,sp)}{f_s(s)} \qquad (6)$$

Donde s es un patrón secuencial, sp es un sub-patrón secuencial del patrón s y $fs_{(s)}$ es la frecuencia del patrón s. Del conjunto de datos de la Tabla 2, se quiere hallar la confianza para la regla de secuencia $< \{10,20\} > \rightarrow < \{30\} >$:

$$Confianza\ (< \{10,20\} > / < \{30\} >) = \frac{Soporte(\{< \{10,20\} \cup \{30\} >)}{Soporte(< \{10,20\} >)} = \frac{0.2}{0.2} = 1$$

Esto quiere decir que el 100% de las reglas de secuencia del conjunto de datos que anteceden con la secuencia $< \{10\},\{20\} >$, el consecuente resultará ser la secuencia $< \{30\} >$.

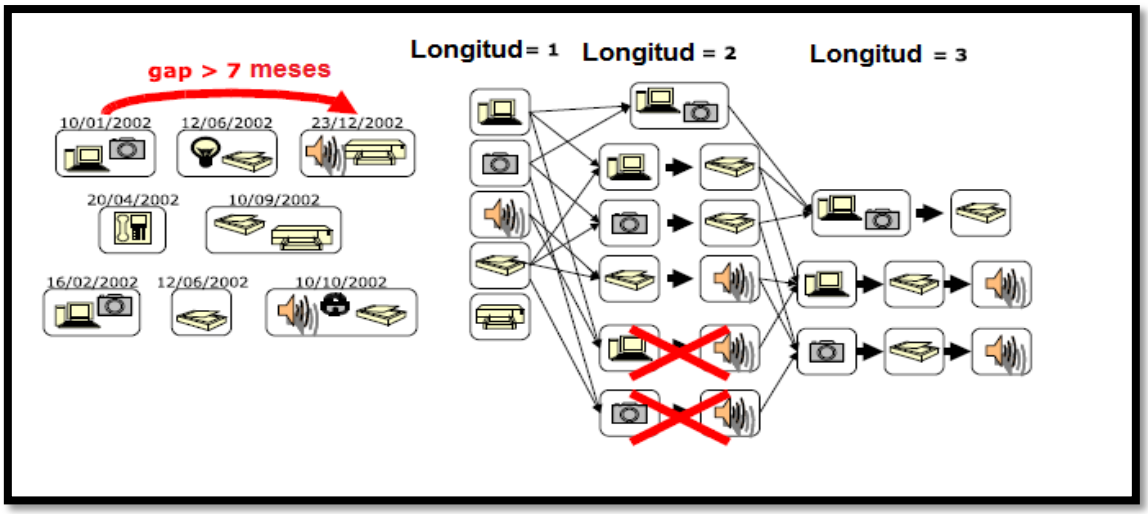
2.9.3 CONTRASTES DE OCURRENCIAS DE UNA SECUENCIA "GAP"

De manera general, se define gap como la distancia de tiempo excesivo que existe entre secuencias, este parámetro puede ser mínimo o máximo y es fijado por el investigador para centrarnos en eventos que ocurrirán en determinado tiempo, por ejemplo en la Figura 4 se fija un máximo gap de 7 meses, que nos obtendrá los patrones de compras que ocurrieron en un tiempo menor o igual a 7 meses, por lo que los eventos que excedan este parámetro serán eliminados de nuestra búsqueda.

De manera formal sea β una secuencia de entrada, cuyos eventos ($\beta_1 \rightarrow \beta_2 \ldots \rightarrow \beta_m$) están estampados por los tiempos con ($eid_1, eid_2 \ldots eid_m$). El gap entre dos eventos consecutivos β_i y β_{i+1} se define como ($eid_{i+1} - eid_i$).

Una secuencia α= eventos ($\alpha_1 \rightarrow \alpha_2 \ldots \rightarrow \alpha_k$) ocurre en β bajo un mínimo y máximo gap, si $\alpha \subseteq \beta$, si existen un conjunto de enteros $i_1 < i_2 < \ldots i_k < i_n$, tal que ∀j, $1<j\leq k$, min_gap ≤ (eid_{i_j}-$eid_{i_{j-1}}$) ≤ max_gap, donde min_gap y max_gap son los umbrales fijados por el usuario (Silvestri, 2006).

Figura 4. Efectos del contrase de máxia ocurrencia Gap



Fuente: (Silvestri, 2006)

2.10 ALGORITMOS SECUENCIALES

2.10.1 ENFOQUE APRIORI

Este enfoque en las reglas de asociación es llamado así porque se basa en el conocimiento previo o "apriori" de los itemset frecuentes, esto sirve para reducir el espacio de búsqueda. La primera introducción de algoritmos de minería de patrones secuenciales basados en apriori fue en Agrawal & Srikant (1995). El proceso fue descompuesto en cinco pasos:

a) Orden: Ordenar los datos transaccionales de acuerdo al ID cliente.
b) L-ítemset: El objetivo es obtener los ítemset de longitud 1 (1-ítemset) de la data ordenada, basado por el soporte fijado.
c) Transformación: Este paso reemplaza las secuencias por aquellos grandes ítemset que contienen. Para una eficiente minería, todas los grandes ítemset son registradas en una serie de enteros. Por último, el conjunto de datos original será transformada en un conjunto de secuencias de los clientes representados por aquellos grandes ítemset.
d) Secuencias: A partir del conjunto de datos secuencial transformado, este paso genera todos los patrones secuenciales frecuentes.

e) Máximo: Este paso poda los patrones secuenciales que están contenidos en otros súper-patrones secuenciales. Esto determina una poda de gran alcance que soluciona los problemas de búsqueda.

Este proceso se itera hasta que cualquier k-itemset frecuente pueda ser generado para algún "k" (Slimani & Lazzez, 2013). De los distintos algoritmos de búsqueda que se originan de este enfoque, se estudiaron los algoritmos SPADE y GSP.

2.10.2 ALGORITMO SECUENCIAL GSP

El algoritmo GSP fue propuesto por Srikant y Agrawal (1996). El nombre de este algoritmo se debe a las iniciales del término en inglés *Generalized Sequential Pattern* (GSP). Este algoritmo crea múltiples escaneos o pasos por encima del conjunto de datos. El primer escaneo es determinar el soporte de cada ítem, que es, el número de secuencias que incluye el conjunto de datos. Al finalizar el primer paso, el algoritmo conoce qué ítems son frecuentes, esto se calcula fijando un mínimo soporte. Cada uno de esos ítems produce un elemento de secuencia frecuente (1-secuencia). Cada escaneo subsiguiente comienza con un conjunto de semillas, que son las secuencias frecuentes encontradas en el escaneo anterior. La semilla es utilizada para generar nuevas secuencias potencialmente frecuentes, llamadas secuencias candidatas. El soporte de estas secuencias candidatas se encuentran durante el paso de los datos. Al final del escaneo, el algoritmo determina cuál de las secuencias candidatas en realidad son frecuentes. Estos candidatos frecuentes se convierten en la semilla para el siguiente paso. El algoritmo termina cuando no hay secuencias frecuentes al final del escaneo, o cuando no se generan secuencias candidatas.

2.10.2.1 GENERACIÓN DE CANDIDATOS

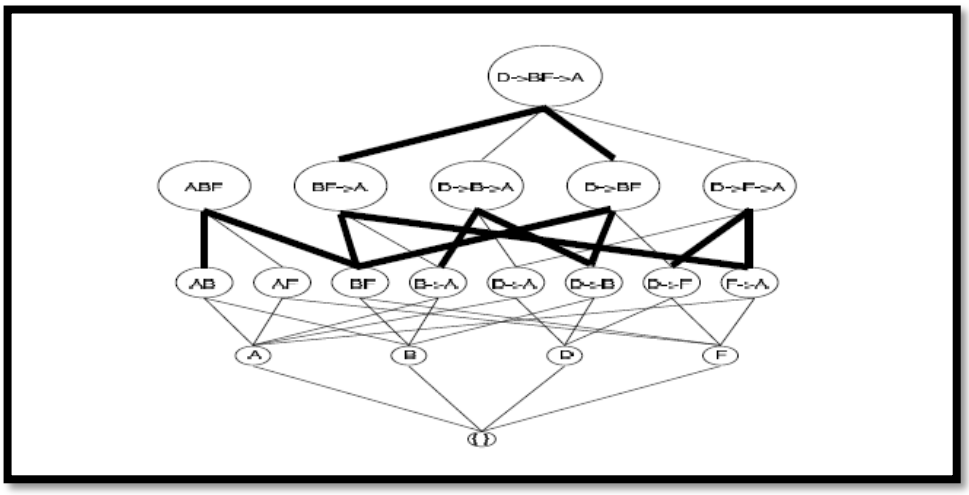
Según Srikant y Agrawal (1996) se refiere a una secuencia con k artículos como k-secuencia. Si un ítem aparece varias veces en diferentes elementos de una secuencia, cada ocurrencia contribuye al valor de k. Se define L_k como el conjunto de todas las k-secuencias frecuentes, y C_k como el conjunto de las k-secuencias candidatas.

Teniendo L_{k-1}, el conjunto de todas las (k-1)-secuencias frecuentes, se quiere generar un súper-conjunto del conjunto de todas las k-secuencias frecuentes. Los candidatos son generados en dos pasos:

- Fase Unión: Se generan secuencias candidatas uniendo L_{k-1} con L_{k-1}, una secuencia s_1 se une con s_2 si la sub-secuencia es obtenida dejando caer los primeros ítems de s_1 es la misma que la sub-secuencia obtenida bajando el última ítem de s_2.
- Fase Poda: Se borran las secuencias candidatas que tengan (k-1) sub-secuencias contiguas cuyo recuento de soporte sea menor que el mínimo establecido.

El conjunto de secuencias candidatos C_k se representa mediante un árbol lexicográfico. Cada nodo en el diagrama es un nodo hoja, que contiene secuencias. Con el fin de encontrar el contador para un patrón, el árbol está atravesado empezando desde la raíz. La siguiente rama para visitar es escogido mediante una función hash en el p^{th} ítem en la secuencia, donde una función hash es una función matemática que transforma los datos de entrada en una serie de caracteres de longitud fija, reduciendo así el espacio de memoria. La Figura 5 representa el árbol lexicográfico con las secuencias obtenidas, cada elipse representa a un patrón, la línea representa la relación de inclusión.

Figura 5. Patrones de secuencias a través de la técnica GSP



Fuente: (Silvestri, 2006)

Los parámetros que se ingresan en este algoritmo son: el conjunto de transacciones (D) y el soporte mínimo especificado por el usuario (S), para tener como resultado a las secuencias frecuentes (F). A continuación se puede observar la estructura de este algoritmo:

C_k: Secuencias candidatos de tamaño k

F_k: Secuencias frecuentes de tamaño k

D: Conjunto de transacciones

S: Soporte mínimo

El pseudocódigo del algoritmo GSP:

Input
D // Lista-ID de secuencias
S// Soporte
Salida
F // Secuencias frecuentes
Comienzo
-Obtener una secuencia en forma de <x> candidata de longitud 1;
-Encontrar F1 (El conjunto de patrones secuenciales de longitud 1);
-Sea K=1;
-Mientras F_K no esté vacío;
-Formar C_{K+1}, el conjunto de candidatos de tamaño k+1 de
- Si C_{K+1}, no está vacío, escanear la base de datos, encontrar los F_{K+1} (El conjunto de patrones secuenciales de tamaño K+1);
Sea K= K+1;
Fin Mientras

2.10.2.2 APLICACIÓN DEL ALGORITMO SECUENCIAL GSP

Se aplicará el algoritmo GSP al conjunto de transacciones que aparecen en la Tabla 2 con un soporte mínimo 0.3

Escaneo 1

Se buscan los ítemsets de tamaño uno, los cuales deben tener un soporte mayor o igual al mínimo fijado por el usuario (0.3)

$$Soporte\ (10) = \frac{1}{5} = 0.2$$

$$Soporte\ (20) = \frac{1}{5} = 0.2$$

$$Soporte\ (30) = \frac{4}{5} = 0.8$$

$$Soporte\ (40) = \frac{2}{5} = 0.4$$

$$Soporte\ (50) = \frac{1}{5} = 0.2$$

$$Soporte\ (60) = \frac{1}{5} = 0.2$$

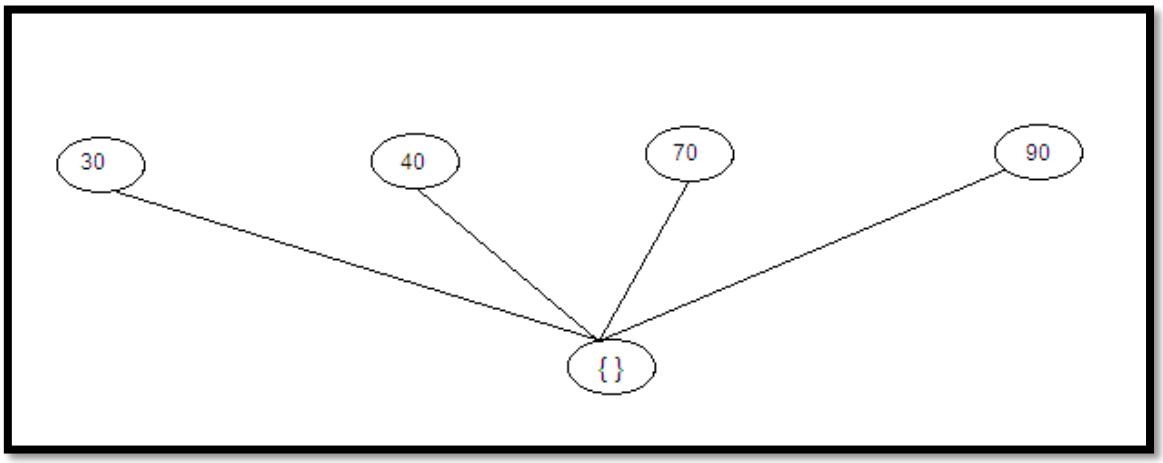
$$Soporte\ (70) = \frac{3}{5} = 0.6$$

$$Soporte\ (90) = \frac{3}{5} = 0.6$$

Luego de obtener los soportes, se compara cada uno de estos con el soporte mínimo especificado por el usuario (0.3), quedando los 4 itemsets frecuentes $F_1 = \{(30), (40), (70), (90)\}$, los cuales tienen un soporte superior al especificado por el usuario.

La Figura 6 muestra el árbol lexicográfico considerando los ítems que sean frecuentes.

Figura 6. Árbol GSP formado con el primer escaneo



Fuente: Elaboración Propia

Escaneo 2

Una vez que se tienen las 1-secuencias, estos se van juntando de a dos, de modo que van formando los siguientes candidatos, con la condición de que k-2 ítems deben ser comunes, en este caso como k=2, no hay restricción. Luego de generar los candidatos, se contabiliza el recuento de cada 2-secuencias, para poder obtener sus respectivos soportes (ver Tabla 4).

Tabla 4: Candidatos de 2-secuencias

Ítems	Soporte
30 , 40	0
30 , 70	0.2
30 , 90	0
40 , 70	0.4
40 , 90	0
70 , 90	0
30-->30	0
30-->40	0.4
30-->70	0.4
30-->90	0.4
40-->30	0
40-->40	0
40-->70	0
40-->90	0.2
70-->30	0
70-->40	0
70-->70	0
70-->90	0.2
90-->30	0
90-->40	0
90-->70	0
90-->90	0

Fuente: Elaboración Propia

Luego de obtener los soportes, se compara cada uno de estos con el soporte mínimo especificado por el usuario (0.3), quedando las 2-secuencias frecuentes, que se muestra en la Tabla 5 las cuales tienen un soporte superior al mínimo.

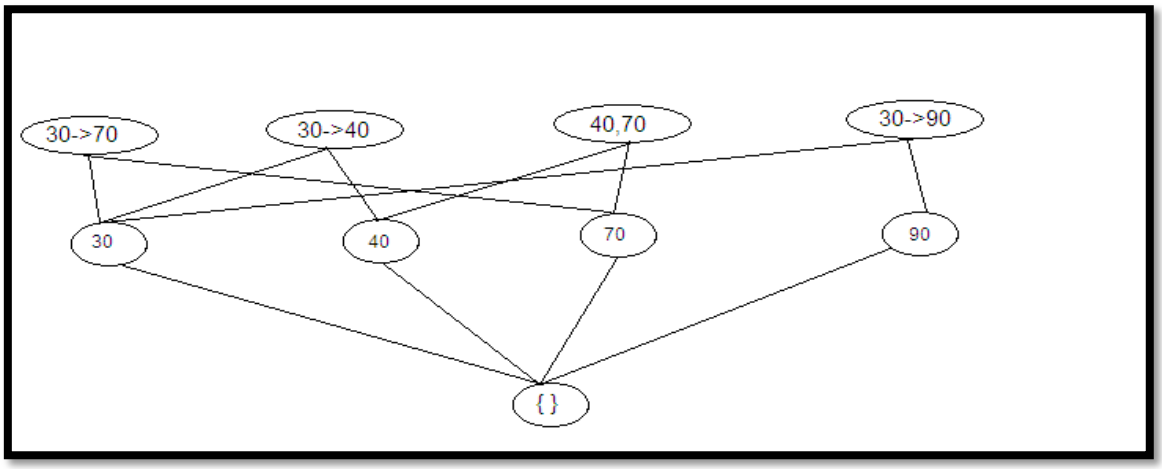
Tabla 5: 2-secuencias frecuentes

Ítems	Soporte
40, 70	0.4
30 → 40	0.4
30 → 70	0.4
30 → 90	0.4

Fuente: Elaboración Propia

De igual modo que en el paso anterior se pude ir armando el árbol (ver Figura 7), la cual es la continuación de la obtenida en la Figura 6, solo considerando las secuencias frecuentes.

Figura 7. Árbol GSP formado con el segundo escaneo



Fuente: Elaboración Propia

Escaneo 3

Una vez calculado las 2-secuencias, estos se van juntando de a dos, de modo que van formando los siguientes candidatos, con la condición de que k-2 ítems deben ser comunes, en este caso como k=3, las secuencias deben tener un ítem en común. Luego de generar los candidatos, se contabiliza el recuento de cada 3-secuencias, para poder obtener sus respectivos soportes (ver Tabla 6).

La única combinación que se puede obtener es $30 \rightarrow 40,70$. Debido a la unión de las secuencias $30 \rightarrow 40$ y $40,70$.

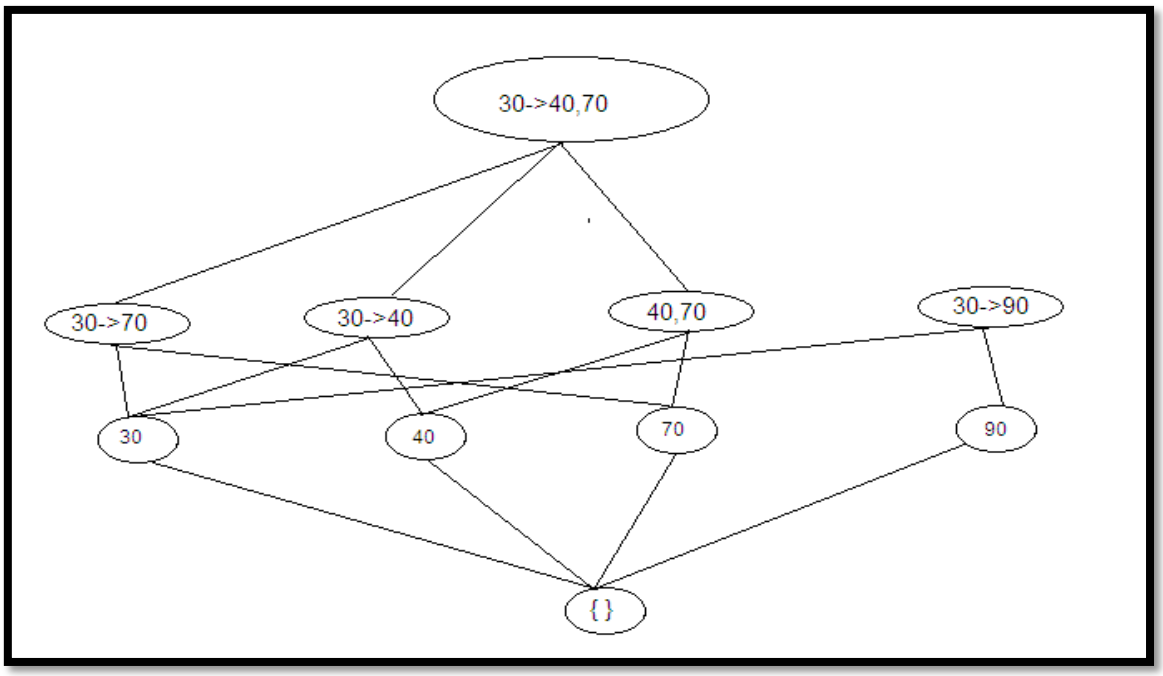
Tabla 6: 3-secuencias frecuentes

Ítems	Soporte
$\mathbf{30 \rightarrow 40,70}$	0.4

Fuente: Elaboración Propia

Luego de obtener el soporte, se compara con el soporte mínimo especificado por el usuario (0.3), quedando la 3-secuencia frecuentes debido a que el soporte superior al mínimo.
El árbo lexicográfico con la 3-secuencia frecuente se muestra en la Figura 8.

Figura 8. Árbol GSP formado con el tercer escaneo



Fuente: Elaboración Propia

Debido a que ya no se puede generar una secuencia de tamaño cuatro, entonces se debe detener el algoritmo y mostrar las secuencias obtenidas, las cuales se muestran en la Tabla 7:

Tabla 7: Reglas obtenidas con el algoritmo GSP

Secuencia	Soporte
30	0.8
40	0.4
70	0.4
90	0.4
40,70	0.4
30 → 40	0.4
30 → 70	0.4
30 → 90	0.4
30 → 40, 70	0.4

Fuente: Elaboración Propia

2.10.3 ALGORITMO SECUENCIAL SPADE

SPADE fue propuesto por Zaki M. J (2000). El nombre de este algoritmo se debe a las iniciales del término en inglés *Sequential Pattern Discovery using Equivalence classes* (SPADE). Es un método de formato vertical con enfoque a priori. Según Slimani & Lazzez (2013) el espacio de búsqueda en SPADE se representa como una estructura de uniones y utiliza la noción de clases equivalentes para reducir el espacio de búsqueda. Descompone la red original en más ligeras sub-uniones, de modo que cada sub-unión se pueda procesar por completo utilizando una estrategia de búsqueda más eficiente. En resultados experimentales muestran que SPADE es dos veces más rápido GSP. La razón detrás de esto es que SPADE utiliza un método de conteo de apoyo más eficiente basado en la estructura de la lista-id. Además, SPADE muestra una escalabilidad lineal con respecto al número de secuencias.

Este algoritmo utiliza un formato vertical sobre el conjunto de datos, diferenciándose de los demás algoritmos secuenciales donde adoptan un formato horizontal, donde mantiene su lista-id basada en disco para cada ítem, como se muestra en la Figura 9. Cada entrada de la lista-id es un par (Sid, Tiempo) donde los ítems ocurren, esto permite hacer las uniones.

Se describe el diseño e implementación del SPADE. La Figura 10 muestra la estructura del algoritmo. Los pasos principales incluyen las secuencias frecuentes de tamaño 1 o 1-secuencia frecuente y 2-secuencia frecuente, la descomposición en la equivalencia de los padres basado en el prefijo clases y la enumeración de todas las demás secuencias frecuentes a través de la búsqueda BFS dentro de cada clase.

Figura 9. Transacciones en formato vertical

A	
SID	EID
1	15
1	20
1	25
2	15
3	10
4	25

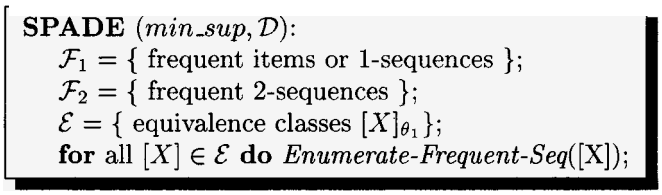
B	
SID	EID
1	15
1	20
2	15
3	10
4	20

D	
SID	EID
1	10
1	25
4	10

F	
SID	EID
1	20
1	25
2	15
3	10
4	20

Fuente: (Zaki M. , 2001)

Figura 10. Algoritmo SPADE



Fuente: (Zaki M. , 2001)

2.10.3.1 CÁLCULO DE 1-SECUENCIA Y 2-SECUENCIAS FRECUENTES

Cálculo de F_1: Teniendo en cuenta el conjunto de datos y la lista-id vertical, todas las 1-secuencias frecuentes pueden calcularse en una sola exploración del conjunto de datos. Para cada elemento del conjunto de datos, se lee su lista-id desde el disco a la memoria. Luego, se escanea la lista-id, incrementando el soporte a cada nuevo Sid encontrado.

Cálculo de F_2: Sea $N = |F_1|$ el número de ítems frecuentes, y A el tamaño promedio de bytes de la lista-id. Una implementación ingenua para calcular la 2-secuencia frecuente requiere $\binom{N}{2}$ listas-id unidas para todos los ítems. La cantidad de datos leída es $A * N * N(N-1)/2$ que corresponde alrededor de N/2 de datos escaneadas. Esto es ineficiente. En vez de este método ingenuo se proponen dos soluciones alternativas:

1. Usar un paso de procesamiento previo para reunir los conteos de todas las 2-secuencias por encima de un límite inferior. Dado que esta información es invariante, tiene que ser calculada una vez, y el costo puede ser amortizado sobre el número de veces que se extrae los datos.

2. Realizar una transformación de vertical a horizontal. Esto se puede hacer con bastante facilidad, con muy poca sobrecarga. Para cada elemento i, se escanea su lista-id en la memoria. Para cada par (Sid, Tiempo), se denominará (S, T) en $L(i)$, se inserta (I, T) en la lista de secuencias de entrada s. Por ejemplo, considerando la lista-id del ítem A, mostrado en la Figura 9. Se escanea el par (1,15), y luego se inserta (A, 15) en la lista de 1-secuencia de entrada. La Figura 11 muestra el conjunto completo de datos de forma horizontal recuperado de los ítems de la lista-id vertical. El cálculo de F_2 se realiza de forma directa a partir del conjunto de datos recuperado de forma horizontal. Se forma

una lista de todas las 2-secuencias en la lista por cada sid, y se actualiza los conteos en una matriz de 2 dimensiones indexada por los ítems frecuentes.

Figura 11. Transformación del conjunto de datos de vertical a horizontal

sid	(*item*, *eid*) pairs
1	(A 15) (A 20) (A 25) (B 15) (B 20) (C 10) (C 15) (C 25) (D 10) (D 25) (F 20) (F 25)
2	(A 15) (B 15) (E 20) (F 15)
3	(A 10) (B 10) (F 10)
4	(A 25) (B 20) (D 10) (F 20) (G 10) (G 25) (H 10) (H 25)

Fuente: (Zaki M. , 2001)

2.10.3.2 DESCOMPOSICION DE RED: CLASES BASADOS EN EL PREFIJO

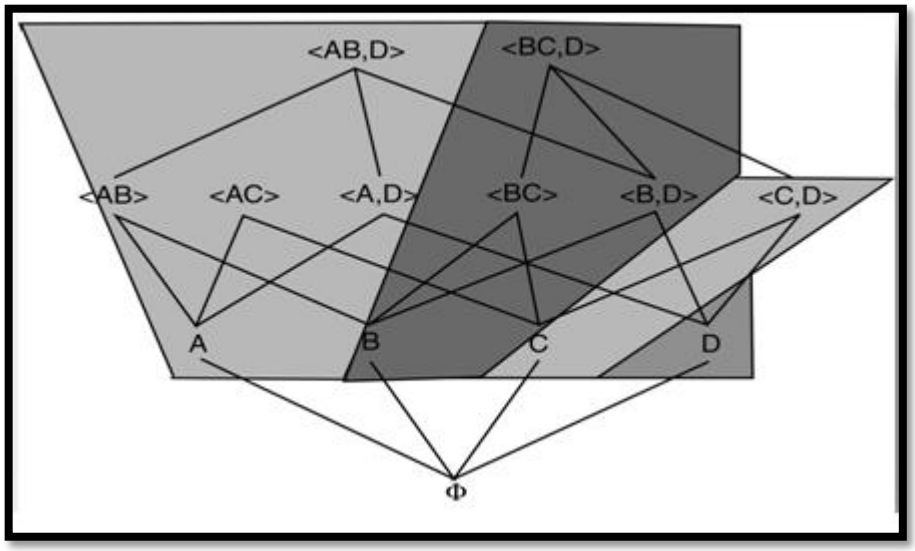
Si se tiene suficiente memoria principal, se podrían enumerar todas las secuencias frecuentes por la que atraviesa la red, y con las uniones temporales se obtendrían los soportes de secuencias. En la práctica, sin embargo, sólo tenemos una cantidad limitada de memoria principal, y todas las listas-id intermediarias no caben en la memoria. Esto nos lleva a la búsqueda de una solución donde la red original se pueda descomponer en piezas más pequeñas de tal manera que cada pieza se pueda resolver de forma independiente en la memoria principal.

Se define la función $p: (S, N) \rightarrow S$ donde S es el conjunto de secuencias, N es el conjunto de números enteros no negativos $p(X, k) = X[1: k]$. En otras palabras, $p(X, k)$ retorna la longitud k prefijo de X. Se define una relación de equivalencia θ_k en la red de la siguiente manera $\forall X, Y \in S$, se dice que X está relacionado a Y bajo θ_k, denotado como $X \equiv_{\theta_k} Y$ si y solo si $p(X, k) = p(Y, k)$. Es decir, dos secuencias son de la misma clase si comparten un prefijo común de tamaño k.

La Figura 12 muestra la partición inducida por la relación equivalente θ_1 en S, donde se derrumban todas las secuencias con un ítem prefijo común en una clase de equivalencia. Resultando un conjunto de clases equivalentes como $\{[A], [B], [C], [D]\}$. Cada espacio

sombreado representa la partición de todas las reglas de asociación, es decir, se tiene un espacio de búsqueda para A, B, C y D de manera independiente. Así ahorrando costos computacionales (Zaki M. , 2001).

Figura 12. Clases equivalentes del algoritmo SPADE

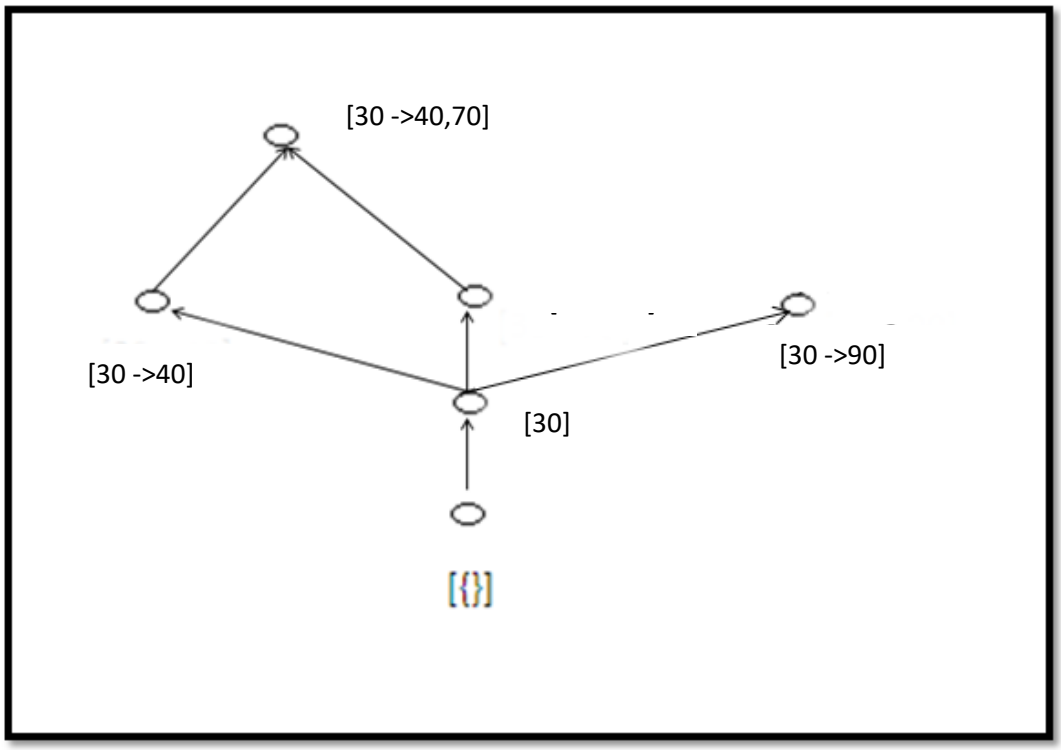


Fuente: (Slimani & Lazzez, 2013)

2.10.3.3 BUSQUEDA DE SECUENCIAS FRECUENTES

Para encontrar una búsqueda eficiente de secuencias frecuentes dentro de cada clase paterna existen dos estrategias principales: Breadth-first (Amplitud primero) y Depth-first (Profundidad primero). Ambos métodos se basan en una descomposición recursiva de cada clase paterna en clases más pequeñas inducidas por la relación de equivalencia θ_1. La Figura 13 muestra la descomposición de $[D]_{\theta_k}$ en clases cada vez más pequeñas, y la red de equivalencia de clase.

Figura 13. Descomposición recursiva de la clase $[30]$ en pequeñas sub-clases vía θ_k



Fuente: Elaboración Propia

Breadth-First Search (BFS): En BFS la red de clases equivalentes es generada por una aplicación recursiva de θ_k es explorada de manera ascendente. Se procesa todas las clases hijas de un nivel antes de pasar al siguiente nivel. Por ejemplo en la Figura 13 se procesa las clases equivalentes $\{[30 \rightarrow 40], [30 \rightarrow 70], [30 \rightarrow 90]\}$ antes de moverse a la clase $\{[30 \rightarrow 40{,}70]\}$.

Depth-First Search (DFS): En DFS se resuelve por completo todas las clases equivalentes hijas a lo largo de una ruta de acceso antes de pasar a la siguiente ruta. Por ejemplo se procesa las clases en el siguiente orden $[30 \rightarrow 40], [30 \rightarrow 70], [30 \rightarrow 40{,}70]$, y así sucesivamente.

La ventaja del BFS sobre DFS es que se tiene más información disponible para la poda. Por ejemplo, en BFS se sabe el conjunto de 2-secuencias antes de construir el conjunto de 3-secuencias, mientras esta información no está disponible en DFS. Por otro lado DFS requiere menos memoria principal que BFS. DFS solo necesita mantener las listas-id intermedias para dos clases consecutivas a lo largo de un único camino, mientras que en BFS se debe llevar un registro de identificación de las listas de todas las clases en dos

niveles consecutivos. Consecuentemente, cuando el número de frecuencias es muy alta, por ejemplo en los casos que el valor del soporte mínimo es muy bajo, DFS puede ser el único enfoque viable, ya que usando el enfoque BFS se puede quedar sin memoria virtual (Zaki M. , 2001).

2.10.3.4 ENUMERACIÒN DE LAS SECUENCIAS FRECUENTES DE LAS CLASES

Se escanea el disco antes de procesar cada clase paterna equivalente de la descomposición inicial, todos los ítems relevantes de la lista-id de dicha clase son escaneadas desde el disco a la memoria. Las listas-ID de los átomos (que son 2-secuencias) de cada clase inicial son construidas por la unión de los ítems de la lista-id. Todas las otras secuencias frecuentes se enumeran como se describió anteriormente. Si todas las clases iniciales tienen un conjunto disjunto de ítems, id-lista de cada elemento se escanea desde el disco sólo una vez durante todo el proceso de enumeración de las secuencias frecuentes en todos los sub-celosías. En el caso general habrá algún grado de superposición de elementos entre los diferentes sub-celosías. Sin embargo, sólo la parte del conjunto de datos correspondiente a los temas frecuentes necesitan ser escaneados, lo que puede ser mucho más pequeño que el conjunto de datos. Además, sub-redes que comparten muchos elementos comunes se pueden procesar en un modo por lotes para minimizar el acceso al disco. De este modo pretendemos que nuestros algoritmos por lo general requieren una sola exploración del conjunto de datos después de calcular F_2, en contraste con los enfoques actuales que requieren varias exploraciones (Zaki M. , 2001).

El Pseudo código para la búsqueda Breadh-First y Depth-First:

Enumerate-Frequent-Seq(S):

for *all atoms* $A_i \in S$ ***do***
$T_i = \emptyset$;

for *all atoms* $A_j \in S$, *with* $j \geq i$ ***do***

$R = A_i \vee A_j$;

if *(Prune (R) =FALSE)* ***then***

$L(R) = L(A_i) \cap L(A_j)$;

$if\ \sigma(R) \geq \min\_sup$ ***then***

$T_i = T_i \cup \{R\};\ F_{|R|} = F_{|R|} \cup \{R\}$;

2.10.3.5 UNIONES TEMPORALES DE LA LISTA-ID

Se describe la forma en que se realiza las uniones de la lista-id para dos secuencias. Considere una clase equivalente $[B \rightarrow A]$ con el conjunto de átomos $\{B \rightarrow AB,\ B \rightarrow AD,\ B \rightarrow A \rightarrow A, B \rightarrow A \rightarrow D,\ B \rightarrow A \rightarrow F\}$. Sea P el prefijo $B \rightarrow A$, entonces se puede re-escribir la clase obteniendo $[P] = \{PB, PD, P \rightarrow A, P \rightarrow D, P \rightarrow F\}$. Se puede observar que la clase tiene dos tipos de átomos: los átomos de eventos $\{PB, PD\}$, y los átomos de secuencia $\{P \rightarrow A, P \rightarrow D, P \rightarrow F\}$. Se puede asumir que los átomos de eventos siempre preceden a los átomos de secuencia. Sin embargo, dependiendo de la unión de los pares de átomos, estos pueden tener hasta tres posibles resultados de secuencias frecuentes (estos son las tres posibles súper-secuencias comunes mínimas) (Zaki M. , 2001):

- Átomo de evento con átomo de evento: Si se une PB con PD, el único resultado posible es PBD.
- Átomo de evento con átomo de secuencia: Si se une PB con $P \rightarrow A$, el único resultado posible es $PB \rightarrow A$.
- Átomo de secuencia con átomo de secuencia: Si se une $P \rightarrow A$ con $P \rightarrow F$, se obtienen tres posibles resultados: un nuevo átomo de evento $P \rightarrow AF$, y dos nuevas secuencias de átomo $P \rightarrow A \rightarrow F$ y $P \rightarrow F \rightarrow A$. Un caso especial surge cuando se une $P \rightarrow A$ consigo misma, que sólo puede producir el nuevo átomo de secuencia $P \rightarrow A \rightarrow A$.

2.10.3.6 APLICACIÓN DEL ALGORITMO SECUENCIAL SPADE

Se aplicará el algoritmo SPADE al conjunto de transacciones que aparecen en la Tabla 2 con un soporte mínimo 0.3.

En la Figura 14 se representa el conjunto de datos de la Tabla 3 en un formato vertical.

Figura 14. Representación en formato vertical

10	
Sid	Tiempo
2	1

20	
Sid	Tiempo
2	1

30	
Sid	Tiempo
1	5
2	3
3	5
4	5

40	
Sid	Tiempo
2	4
4	6

50	
Sid	Tiempo
3	5

60	
Sid	Tiempo
2	4

70	
Sid	Tiempo
2	4
3	5
4	6

90	
Sid	Tiempo
1	6
4	7
5	2

Elaboración Propia

Escaneo 1

Se buscan los ítemsets de tamaño uno, los cuales deben tener un soporte mayor o igual al mínimo fijado por el usuario (0.3)

$$Soporte\ (10) = \frac{1}{5} = 0.2$$

$$Soporte\ (20) = \frac{1}{5} = 0.2$$

$$Soporte\ (30) = \frac{4}{5} = 0.8$$

$$Soporte\ (40) = \frac{2}{5} = 0.4$$

$$Soporte\ (50) = \frac{1}{5} = 0.2$$

$$Soporte\ (60) = \frac{1}{5} = 0.2$$

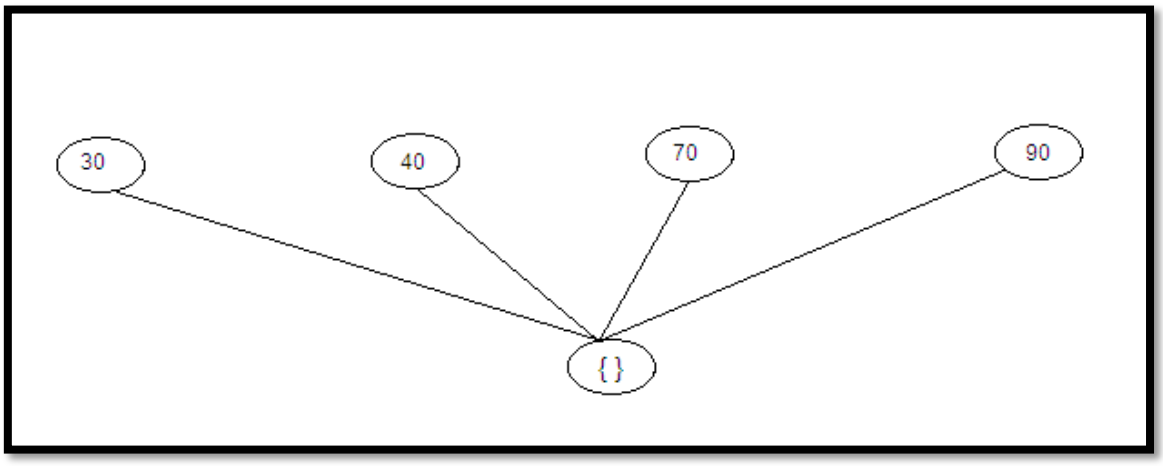
$$Soporte\ (70) = \frac{3}{5} = 0.6$$

$$Soporte\ (90) = \frac{3}{5} = 0.6$$

Luego de obtener los soportes, se compara cada uno de estos con el soporte mínimo especificado por el usuario (0.3), quedando los 4 itemsets frecuentes $F_1 = \{(30), (40), (70), (90)\}$, los cuales tienen un soporte superior al especificado por el usuario.

Se puede ir llenando el gráfico en el árbol lexicográfico considerando los ítems que sean frecuentes, como se observa en la Figura 15.

Figura 15. Árbol SPADE formado con el primer escaneo



Fuente: Elaboración Propia

Escaneo 2

Una vez que se tienen las 1-secuencias, estos se van juntando de a dos, de modo que van formando los siguientes candidatos. Luego de generar los candidatos, se contabiliza el recuento de cada 2-secuencias en, para poder obtener sus respectivos soportes.

La Tabla 8 representa los soportes para la secuencia 30, 40

.

Tabla 8: Lista-id para 30 y 40 (Unión no temporal)

30, 40		
Sid	Tiempo 30	Tiempo 40
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30,40 es 0, así que el soporte es 0 y no satisface el mínimo soporte.

La Tabla 9 representa los soportes para la secuencia 30 y 70.

Tabla 9: Lista-id para 30 y 70 (Unión no temporal)

30, 70		
Sid	Tiempo 30	Tiempo 70
3	5	5

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30,70 es 1, así que el soporte es 0.2 y no satisface el mínimo soporte.

La Tabla 10 representa los soportes para la secuencia 30 y 70.

Tabla 10: Lista-id para 30, 90 (Unión no temporal)

30 ∧ 90		
Sid	Tiempo 30	Tiempo 90
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30,90 es 0, y no satisface el mínimo soporte.

La Tabla 11 representa los soportes para la secuencia 40 y 70.

Tabla 11: Lista-id para 40, 70 (Unión no temporal)

40, 70		
Sid	Tiempo 40	Tiempo 70
2	4	4
4	6	6

Fuente: Elaboración Propia

Para la secuencia 40, 70, el número de ocurrencias es 2, por lo que el soporte es 0.4 y satisface el mínimo soporte.

La Tabla 12 representa los soportes para la secuencia 40 y 90.

Tabla 12: Lista-id para 40, 90 (Unión no temporal)

40 ∧ 90		
Sid	Tiempo 40	Tiempo 90
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 40,90 es 0, y no satisface el mínimo soporte.

La Tabla 13 representa los soportes para la secuencia 70 y 90.

Tabla 13: Lista-id para 70, 90 (Unión no temporal)

70 ∧ 90		
Sid	Tiempo 70	Tiempo 90
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 70,90 es 0, y no satisface el mínimo soporte.

La Tabla 14 representa los soportes para la secuencia $30 \rightarrow 30$.

Tabla 14: Lista-id para $30 \rightarrow 30$ (Unión temporal)

$30 \rightarrow 30$		
Sid	Tiempo 30	Tiempo 30
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia $30 \rightarrow 30$ es 0, y no satisface el mínimo soporte.

La Tabla 15 representa los soportes para la secuencia $30 \rightarrow 40$.

Tabla 15: Lista-id para 30 → 40 (Unión temporal)

30 → 40		
Sid	Tiempo 30	Tiempo 40
2	3	4
4	5	6

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30 → 40 es 2, por lo que el soporte es 0.4 y satisface el mínimo soporte.

La Tabla 16 representa los soportes para la secuencia 30 → 70.

Tabla 16: Lista-id para 30 → 70 (Unión temporal)

30 → 70		
Sid	Tiempo 30	Tiempo 70
2	3	4
4	5	6

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30 → 70 es 2, por lo que el soporte es 0.4 y satisface el mínimo soporte.

La Tabla 17 representa los soportes para la secuencia 30 → 90.

Tabla 17: Lista-id para 30 → 90 (Unión temporal)

30 → 90		
Sid	Tiempo 30	Tiempo 90
1	5	6
4	5	7

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia 30 → 90 es 2, por lo que el soporte es 0.4 y satisface el mínimo soporte.

La Tabla 18 representa los soportes para la secuencia 40 → 30.

Tabla 18: Lista-id para 40 → 30 (Unión temporal)

40 → 30		
Sid	Tiempo 40	Tiempo 30
-	-	-

Fuente: Elaboración Propia

El número de ocurrencias para la secuencia $40 \rightarrow 30$ es 0, por lo que no satisface el mínimo soporte.

La Tabla 19 representa los soportes para la secuencia $40 \rightarrow 40$.

Tabla 19: Lista-id para 40 → 40 (Unión temporal)

40 → 40		
Sid	Tiempo 40	Tiempo 40
-	-	-

Fuente: Elaboración Propia

Para la secuencia $40 \rightarrow 40$, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 20 representa los soportes para la secuencia $40 \rightarrow 70$.

Tabla 20: Lista-id para 40 → 70 (Unión temporal)

40 → 70		
Sid	Tiempo 40	Tiempo 70
-	-	-

Fuente: Elaboración Propia

Para la secuencia $40 \rightarrow 70$, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 21 representa los soportes para la secuencia $40 \rightarrow 90$.

Tabla 21: Lista-id para 40 → 90 (Unión temporal)

40 → 90		
Sid	Tiempo 40	Tiempo 90
4	6	7

Fuente: Elaboración Propia

Para la secuencia $40 \rightarrow 90$, el número de ocurrencias es 1, por lo que el soporte es 0.2 y no satisface el mínimo soporte.

La Tabla 22 representa los soportes para la secuencia $70 \rightarrow 30$.

Tabla 22: Lista-id para $70 \rightarrow 30$ (Unión temporal)

$70 \rightarrow 30$		
Sid	Tiempo 70	Tiempo 30
-	-	-

Fuente: Elaboración Propia

Para la secuencia $70 \rightarrow 30$, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 23 representa los soportes para la secuencia $70 \rightarrow 40$.

Tabla 23: Lista-id para $70 \rightarrow 40$ (Unión temporal)

$70 \rightarrow 40$		
Sid	Tiempo 70	Tiempo 40
-	-	-

Fuente: Elaboración Propia

Para la secuencia $70 \rightarrow 40$, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 24 representa los soportes para la secuencia $70 \rightarrow 70$.

Tabla 24: Lista-id para $70 \rightarrow 70$ (Unión temporal)

$70 \rightarrow 70$		
Sid	Tiempo 70	Tiempo 70
-	-	-

Fuente: Elaboración Propia

Para la secuencia $70 \rightarrow 70$, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 25 representa los soportes para la secuencia $70 \rightarrow 90$.

Tabla 25: Lista-id para 70 → 90 (Unión temporal)

70 → 90		
Sid	Tiempo 70	Tiempo 90
4	6	7

Fuente: Elaboración Propia

Para la secuencia 70 → 90, el número de ocurrencias es 1, por lo que el soporte es 0.2 y no satisface el mínimo soporte.

La Tabla 26 representa los soportes para la secuencia 90 → 30.

Tabla 26: Lista-id para 90 → 30 (Unión temporal)

90 → 30		
Sid	Tiempo 90	Tiempo 30
-	-	-

Fuente: Elaboración Propia

Para la secuencia 90 → 30, el número de ocurrencias es 0 y no satisface el mínimo soporte.
La Tabla 27 representa los soportes para la secuencia 90 → 40.

Tabla 27: Lista-id para 90 → 40 (Unión temporal)

90 → 40		
Sid	Tiempo 90	Tiempo 40
-	-	-

Fuente: Elaboración Propia

Para la secuencia 90 → 40, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 28 representa los soportes para la secuencia 90 → 70.

Tabla 28: Lista-id para 90 → 70 (Unión temporal)

90 → 70		
Sid	Tiempo 90	Tiempo 40
-	-	-

Fuente: Elaboración Propia

Para la secuencia 90 → 70, el número de ocurrencias es 0 y no satisface el mínimo soporte.

La Tabla 29 representa los soportes para la secuencia 90 → 90.

Tabla 29: Lista-id para 90 → 90 (Unión temporal)

90 → 90		
Sid	Tiempo 90	Tiempo 40
-	-	-

Fuente: Elaboración Propia

Para la secuencia 90 → 90, el número de ocurrencias es 0 y no satisface el mínimo soporte.

De las uniones no temporales y temporales obtenidas arriba, la Tabla 30 muestra el recuento de ocurrencia de secuencias de elementos.

Tabla 30: 2-secuencias candidatas

Ítems	**Número de ocurrencia del ítem**
30, 40	0
30, 70	0.2
30, 90	0
40, 70	0.4
40, 90	0
70 ∧ 90	0
30 → 30	0
30 → 70	0.4
30 → 90	0.4
40 → 30	0
40 → 40	0
40 → 70	0
40 → 90	0.4
70 → 30	0
70 → 40	0
70 → 70	0
70 → 90	0.2
90 → 30	0
90 → 40	0.2
90 → 70	0.2
90 → 90	0

Fuente: Elaboración Propia

Luego de obtener los soportes, se compara cada uno de estos con el soporte mínimo especificado por el usuario (0.3), quedando las 2-secuencias frecuentes, que se muestra en la Tabla 31 las cuales tienen un soporte superior al mínimo.

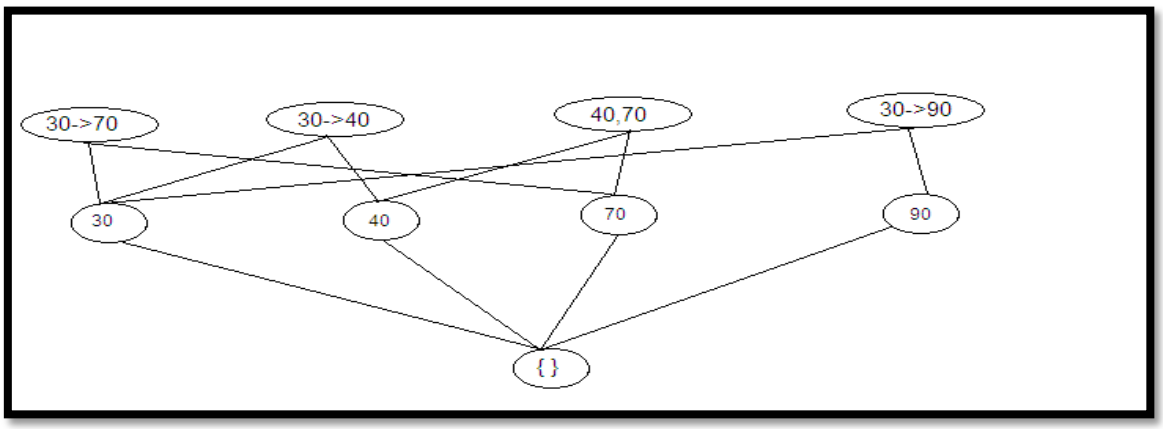
Tabla 31: 2-secuencias

Ítems	Número de ocurrencia del ítem
40, 70	0.4
30 → 40	0.4
30 → 70	0.4
30 → 90	0.4

Fuente: Elaboración Propia

De igual modo que en el paso anterior se pude ir armando árbol lexicográfico (ver Figura 16), la cual es la continuación de la obtenida en la Figura 15, solo considerando las secuencias frecuentes.

Figura 16. Árbol SPADE formado con el segundo escaneo



Fuente: Elaboración Propia

Escaneo 3

Una vez que se tienen las 2-secuencias, estos se van juntando de a dos, de modo que van formando los siguientes candidatos. Luego de generar los candidatos, se contabiliza el recuento de cada 3-secuencias (ver Tabla 32), para poder obtener sus respectivos soportes.

La única unión de secuencias que se puede obtener es:

Tabla 32: Lita-id para $30 \rightarrow 40, 70$ (Unión temporal)

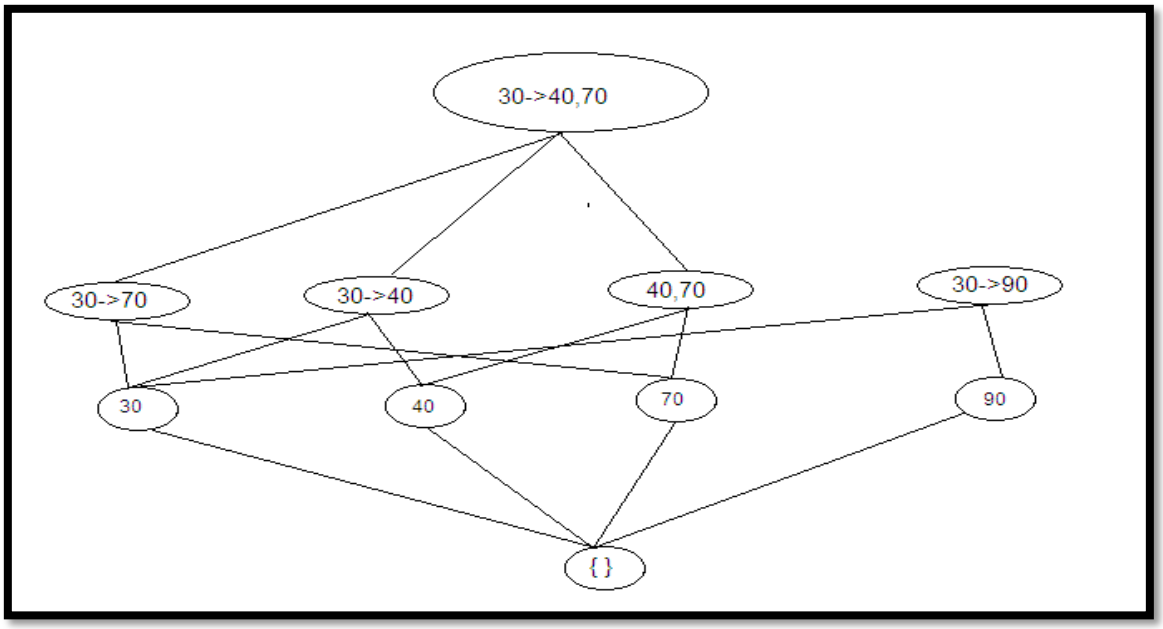
$30 \rightarrow 40, 70$			
SID	Tiempo 30	Tiempo 40	Tiempo 70
2	3	4	4
4	5	6	6

Fuente: Elaboración Propia

Para la secuencia 30 → 40 70, el número de ocurrencias es 2, por lo que el soporte es 0.4 y satisface el mínimo soporte.

El árbol lexicográfico con la 3-secuencia frecuente se muestra en la Figura 17.

Figura 17. Árbol SPADE formado con el tercer escaneo



Fuente: Elaboración Propia

Debido a que ya no se puede generar una secuencia de tamaño cuatro, entonces se debe detener el algoritmo y mostrar las secuencias obtenidas (ver Tabla 33), las cuales son:

Tabla 33: Reglas obtenidas con el algoritmo SPADE

Secuencia	Soporte
30	0.8
70	0.4
90	0.4
40	0.4
40,70	0.4
$\mathbf{30 \rightarrow 40}$	0.4
$\mathbf{30 \rightarrow 70}$	0.4
$\mathbf{30 \rightarrow 90}$	0.4
$\mathbf{30 \rightarrow 40,70}$	0.4

Fuente: Elaboración propia

2.11 ANTECEDENTES DE LA INVESTIGACIÓN

En un estudio relacionado a la salud en el Reino, Unido Reps & Garibaldi (2011) tuvieron como objetivo mejorar la atención de salud. El principal problema fue pronosticar que enfermedad tendría el paciente dado que se presentaron enfermedades previas. Para dar solución a este problema se utilizaron todos los registros de las historias clínicas de los pacientes del hospital y se aplicaron las reglas de asociación secuencial SPADE para encontrar asociaciones entre enfermedades. Entre los resultados que se presentaron se llegaron a las siguientes conclusiones: se pudo observar que es más probable que una persona sufra de Hipertensión esencial cuando esta nace en el año 1943 que una que nazca en el 1944. La probabilidad que un paciente padezca Hipertensión esencial aumenta cuando la edad aumenta. Por último un paciente puede tener menor probabilidad de repetir la enfermedad si se les da consejos.

Uno de los primeros estudios de algoritmos secuenciales se realizó en Zaki M. (2001) debido al gran tamaño de reglas de asociación que realiza la técnica GSP y al tiempo de procesamiento, se trataron de buscar mejores técnicas que sean más eficientes, es decir, que generen solo las reglas más importantes y reglas con menos tiempo de proceso computacional. Por lo que se buscó una nueva técnica donde el espacio de búsqueda sea más reducido. La técnica de denominó SPADE y la única diferencia es que toma los datos de forma vertical mientras que el GSP lo toma de forma horizontal**.**

En otro estudio Yang (2008) comparó tres técnicas: Prefix Span, SPADE y SPAM, debido a las numerosas técnicas de algoritmos secuenciales. Para la comparación se tomaron los indicadores de tiempo de proceso y el uso de memoria, donde se probaron dos conjuntos de datos. En el primer conjunto de datos la técnica Prefix SPAN logró ser el más eficiente algoritmo respecto al tiempo de proceso y uso de memoria, mientras que en el segundo conjunto de datos la ´técnica SPADE obtuvo menos tiempo de proceso, pero respecto a la memoria usada Prefix SPAN logró ser más eficiente.

Verma & Mehta (2014) se realizaron una comparación de los algoritmos secuenciales, debido a la gran cantidad de algoritmos que existen, donde el objetivo fue determinar la técnica más eficiente respecto al número de reglas generadas, tiempo de proceso y uso de la memoria. Las técnicas de estos algoritmos escogidos fueron: SPADE, GSP y Prefix SPAN. Los resultados que se obtuvieron fueron que según el tiempo de proceso en primer lugar resultó Prefix SPAN, seguido de SPADE y por último GSP. Según el uso de memoria empleada en primer lugar resultó Prefix SPAN, seguido de GSP y por último SPADE. Por último aspecto en reglas generadas el Prefix SPAN resultó con menores reglas.

En otro estudio, que pretendía encontrar asociaciones entre palabras recogidas del Haddith, donde el Haddith son las citas que dijo el profeta islámico Muhammad sobre la fe, ciencia, adzan, entre otros. Con el algoritmo SPADE encontraron fuertes asociaciones entre palabras, empleando un apoyo mínimo de 0.03 y una confianza de 0.5. Las reglas que se generaron fueron las siguientes:

- {Urwah bin Az Zubair} => {Hisyam bin 'Urwah bin Az Zubair bin Al 'Awwam}
- {Syu'aib bin Abi Hamzah Dinar} => {Al Hakam bin Nafi'}
- {Muhammad bin Muslim},{Syu'aib bin Abi Hamzah Dinar} =>{Al Hakam bin Nafi'}

La contribución de esta investigación es la extensión de las reglas de asociación ya no en el campo de transacciones sino en textos (Septiani, Febriani Astuti, & Fajriya Hakim, 2015)

Por último en el campo del marketing se utilizó el algoritmo secuencial GSP para poder determinar las secuencias de compras de transacciones en un centro de ventas. Los resultados obtenidos mostraron que los clientes compraron bio cola y luego compran pavo

y en otro resultado los clientes compran bio cola y luego la vuelven a comprar. Estos dos patrones se pueden interpretar como “el cliente compra bio cola y alguna vez en el futuro compra pavo” y como “el cliente compra bio cola y alguna vez en el futuro vuelve a comprar bio cola” respectivamente. El primer patrón podría ser interpretado como una relación entre ambos productos y el segundo patrón puede ser interpretado como una satisfacción con el producto. Ambos patrones pueden ser utilizados en un proceso de toma de decisiones como, por ejemplo, la producción de campañas publicitarias (Solano, 2011).

III. MATERIALES Y MÉTODOS

3.1 MATERIALES

Los materiales y equipos utilizados en la presente investigación son los siguientes

a) Una computadora laptop marca ACER, con un procesador Intel® Core(TM) i5 CPU 2430M @ 2.40GHz 2.40GHz, con una memoria RAM de 4.00 GB y un sistema operativo Windows 7 Professional de 64 bits.

b) El programa estadístico R versión 3.3.3 con los paquetes arules y arulesSequences

c) El programa para minería de datos rapidminer versión 5.3 con los módulos Data Transformation y Modelling

3.2 METODOLOGÍA DE LA INVESTIGACIÓN

3.2.1 TIPO DE INVESTIGACIÓN

La investigación es de tipo exploratoria y descriptiva, ya que se describirán los datos con el objetivo de generar las reglas o patrones ocultos entre los productos vendidos.

3.2.2 DISEÑO DE LA INVESTIGACIÓN

Es una investigación no experimental con diseño transversal, dado que la recolección de los elementos de información de la muestra de la población se hizo solo una vez.

3.2.3 FORMULACIÓN DE LA HIPÓTESIS

El algoritmo secuencial SPADE obtiene resultados más eficientes en cuanto al número de reglas generadas que el algoritmo secuencial GSP.

3.2.4 POBLACIÓN

Se trabajó con todos los datos correspondiente de los clientes que compraron en un supermercado en el periodo Julio 2009 – Junio 2010, extraídos de Solano (2011). El conjunto de datos está formado por 2738 transacciones, 246 ítems y 199 clientes que compraron en un supermercado. La localización exacta del conjunto de datos es: http://sourceforge.net/projects/chronos-tmw/files/synthetic_data/synthetic_data_generated_using_KNIME_with_007008_ShoppingBasket_out_of_the_box.csv/download.

3.2.5 IDENTIFICACIÓN DE LAS VARIABLES

Las variables utilizadas en esta investigación son:

- Código del cliente: Variable de tipo numérica que registra el número de identificación única de cada cliente.

- Código del tiempo: La variable código del tiempo es una variable cualitativa. Esta variable se encuentra en formato Tiempo Unix, que nos permitirá saber en qué instante el cliente realizó la transacción.

- Elemento: Esta variable nos permite saber el o los ítems que compró el cliente en una determinada transacción.

3.2.6 METODOLOGÍA APLICADA

Los pasos que se realizaron para poder contrastar la hipótesis de la siguiente investigación son las siguientes:

I. Análisis exploratorio de las variables del supermercado
II. Identificación de las reglas de asociación secuencial SPADE más importantes que promueven la oferta de otros productos.
III. Identificación de las reglas de asociación secuencial GSP más importantes que promueven la oferta de otros productos.
IV. Identificación de las reglas de asociación más importantes que promueven la oferta de otros productos.
V. Comparación de las reglas obtenidas según las ventas con los algoritmos SPADE y GSP mediante la cantidad de reglas generadas y las similitudes.

IV. RESULTADOS Y DISCUSIÓN

4.1 ANÁLISIS EXPLORATORIO DE LOS DATOS

Antes de realizar los análisis respectivos, se tuvo que limpiar los datos debido a que existían ítems que se repetían en una misma transacción y al evaluarlo con el algoritmo secuencial SPADE, este elimina los ítems duplicados. Por esta razón, eliminando los ítems duplicados, se estandarizaron los resultados y procesos.

DISTRIBUCIÓN DE PRODUCTOS

La Tabla 34 muestra la distribución de los productos utilizados en el estudio, según la clase que corresponda. La lista de todos los productos se encuentran en el Anexo 1

Tabla 34: Distribución de Productos

Clase de Producto	Cantidad	Porcentaje
Vegetales	58	24%
Frutas	42	17%
Leche, huevo y otros productos	28	11%
Bebidas alcohólicas	25	10%
Lujos	19	8%
Productos de panadería	16	7%
Gaseosas	14	6%
Comida preparada	12	5%
Carne	12	5%
Pescados	10	4%
Caramelos	10	4%
Total	246	100%

Fuente: Elaboración propia

Se puede observar que el producto leche, huevo y otros productos fue el producto ofertado con ventas cruzadas debido a que se encuentran en una sola categoría. Además, el mayor porcentaje de la Clase de producto fue el de vegetales (24%), seguido de frutas (17%), leche, huevo y otros productos (11%), entre otros.

4.2 OBTENCIÓN DE LAS REGLAS DE ASOCIACIÓN SECUENCIAL SPADE

Primero se buscarán reglas de asociación con soportes mínimos de 70% y 75% para los algoritmos SPADE.

Para poder hacer uso del algoritmo SPADE en el programa R 3.3.1, es necesario ingresar los filtros de soporte para el algoritmo (ver Anexo 2 y 3). La Tabla 35 muestra las reglas obtenidas con el algoritmo secuencial SPADE con un soporte mínimo 75%.

Tabla 35: Reglas obtenidas con el algoritmo SPADE

Regla	Soporte
<{152}>	0.92
<{23}>	0.895
<{92}>	0.87
<{43}>	0.86
<{3}>	0.84
<{70}>	0.83
<{90}>	0.83
<{34}>	0.825
<{84}>	0.815
<{122}>	0.81
<{123}>	0.81
<{124}>	0.81
<{22}>	0.81
<{122,124}>	0.81
<{123,124}>	0.81
<{122,123,124}>	0.81
<{122,123}>	0.81
<{152},{152}>	0.765
<{152},{92}>	0.75

Fuente: Elaboración propia

Para un soporte de 75% el algoritmo SPADE encontró un total de 19 patrones. Se podría eliminar los patrones con un solo tamaño, como por ejemplo el patrón $< \{152\} >$.

Para la interpretación de los patrones es necesario conocer los elementos que se encuentran con códigos. Por ejemplo el elemento 92 se refiere al Pavo (carne blanca) y el elemento 152 hace referencia a bio coke (una bebida carbonatada). Con esta información, el patrón $< \{152\}, \{92\} >$ se puede interpretar como que "existe un 75% de probabilidad que el cliente compre la gaseosa bio coke y alguna vez en un futuro compre pavo". Para el siguiente patrón, se podría interpretar como "existe un 76.5% de probabilidad que el cliente compra bio coke y alguna vez en el futuro lo vuelve a comprar".

Además del soporte, también se pueden mostrar más indicadores como la confianza y el levantamiento. La Tabla 36 muestra los resultados con un soporte mínimo del 70% y una confianza mínima del 70%.

Tabla 36: Reglas obtenidas con el algoritmo SPADE con los indicadores

Regla	Soporte	Confianza	Levantamiento
<{43}> => <{92}>	0.740	0.860	0.989
<{43}> => <{3}>	0.705	0.820	0.976
<{92}> => <{3}>	0.705	0.810	0.965
<{92}> => <{92}>	0.725	0.833	0.958
<{43}> => <{23}>	0.725	0.843	0.942
<{152}> => <{92}>	0.750	0.815	0.937
<{92}> => <{43}>	0.700	0.805	0.936
<{92}> => <{152}>	0.745	0.856	0.931
<{152}> => <{3}>	0.715	0.777	0.925
<{152}> => <{43}>	0.730	0.793	0.923
<{23}> => <{92}>	0.715	0.799	0.918
<{43}> => <{152}>	0.725	0.843	0.916
<{152}> => <{152}>	0.765	0.832	0.904
<{152}> => <{23}>	0.735	0.799	0.893
<{23}> => <{152}>	0.725	0.810	0.880

Fuente: Elaboración propia

Según los resultados, al momento de insertar indicadores como la confianza y el soporte hacen que los patrones de tamaño uno se eliminen ya que no poseen un antecedente y un consecuente. Según el levantamiento y la confianza, el patrón $< \{43\} > => < \{92\} >$ es el más fuerte. El elemento 43 se refiere a costillas de cordero y el elemento 92 se refiere al pavo. El patrón se puede interpretar a que "la probabilidad que un cliente compre pavo en un futuro, dado que compró antes costillas de cordero es de 0.86". El indicador del levantamiento es cercano a uno, por lo que esta regla es más certera que el patrón $< \{152\}, \{92\} >$.

En caso que se quiera determinar las reglas de asociación de acuerdo a un producto, el algoritmo SPADE facilita la visualización utilizando clases equivalentes. La Tabla 37 muestra las primeras 15 reglas de asociación secuencial SPADE utilizando como clase equivalente el producto costilla de cordero (código 43).

Tabla 37: Las primeras 15 reglas obtenidas con el algoritmo SPADE utilizando clases equivalentes

Regla	Soporte	Confianza	Levantamiento
<{43}> => <{92}>	0.740	0.860	0.989
<{43}> => <{23}>	0.725	0.843	0.942
<{43}> => <{152}>	0.725	0.843	0.916
<{43},{152}> => <{152}>	0.610	0.841	0.915
<{43},{124}> => <{152}>	0.505	0.835	0.907
<{43},{123}> => <{152}>	0.505	0.835	0.907
<{43},{122}> => <{152}>	0.505	0.835	0.907
<{43},{123,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,123,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,123}> => <{152}>	0.505	0.835	0.907
<{43,152}> => <{152}>	0.565	0.831	0.903
<{43},{43}> => <{152}>	0.575	0.827	0.899
<{43,152}> => <{92}>	0.560	0.824	0.947
<{43},{70}> => <{152}>	0.535	0.823	0.895

Fuente: Elaboración propia

Utilizando las clases equivalentes se puede obtener más información de un producto particionando la red. Así, poder tener una información específica por cada ítem. El elemento 23 hace referencia a coke. Por lo que se podría interpretar como "la probabilidad que un cliente compre pavo en un futuro, dado que compró antes costillas de cordero es de 0.86". La interpretación para el patrón 9 es "la probabilidad que un cliente compre bio coke en un futuro, dado que compró antes los tres tipos de caramelos, dado que compró aún antes costillas de cordero es 0.835". El algoritmo SPADE encontró un total de 154 reglas de asociación. Los patrones descubiertos se encuentran en el Anexo 4.

4.3 OBTENCIÓN DE LAS REGLAS DE ASOCIACIÓN SECUENCIAL GSP

Para poder hacer uso del algoritmo GSP en el programa Rapidminer 5.3, es necesario ingresar los filtros de soporte para el algoritmo. Se utilizaron los mismos filtros que en el

algoritmo SPADE. La Tabla 38 muestra las reglas obtenidas con el algoritmo GSP con un soporte mínimo de 70% (ver Anexo 5).

Tabla 38: Reglas obtenidas con el algoritmo GSP con un soporte de 70%

Soporte	Regla	
0.81	122.0,123.0>	
0.81	122.0,124.0>	
0.81	123.0,124.0>	
0.81	122.0,123.0, 124.0>	
0.765	152.0>	152.0>
0.75	152.0>	92.0>
0.745	92.0>	152.0>
0.74	43.0>	92.0>
0.735	152.0>	23.0>
0.73	152.0>	43.0>
0.725	23.0>	152.0>
0.725	43.0>	23.0>
0.725	43.0>	152.0>
0.725	92.0>	92.0>
0.715	23.0>	92.0>
0.715	152.0>	3.0>
0.705	43.0>	3.0>
0.705	92.0>	3.0>
0.7	92.0>	43.0>

Fuente: Elaboración propia

Los patrones encontrados por el algoritmo GSP coinciden con los encontrados en SPADE ya que provienen del mismo enfoque a priori. La única diferencia que se tiene es que en Rapidminer no considera los patrones de tamaño uno. Además, GSP no cuenta con la opción de mostrar más indicadores como la confianza y el levantamiento. Por último en caso que se encuentre un elevado número de patrones, resultaría difícil encontrar patrones específicos como se utilizó en el algoritmo SPADE.

4.4 IDENTIFICACIÓN DE LAS REGLAS MÁS IMPORTANTES QUE PROMUEVEN LA OFERTA DE OTROS

La Tabla 39 muestra los productos respectivos que promueven la oferta de otros.

Tabla 39: Principales reglas entre productos

Regla	Soporte	Confianza	Levantamiento
<(Costilla de cordero)> => <(Pavo)>	0.74	0.860	0.989
<(Costilla de cordero)> => <(Manzana)>	0.705	0.820	0.976
<(Pavo)> => <(Manzana)>	0.705	0.810	0.965
<(Pavo)> => <(Pavo)>	0.725	0.833	0.958
<(Costilla de cordero)> => <(Coke)>	0.725	0.843	0.942
<(Bio Coke)> => <(Pavo)>	0.75	0.815	0.937
<(Pavo)> => <(Costilla de cordero)>	0.700	0.805	0.936
<(Pavo)> => <(Bio Coke)>	0.745	0.856	0.931
<(Bio Coke)> => <(Manzana)>	0.715	0.777	0.925
<(Bio Coke)> => <(Costilla de cordero)>	0.73	0.793	0.923
<(Coke)> => <(Pavo)>	0.715	0.799	0.918
<(Costilla de cordero)> => <(Bio Coke)>	0.725	0.843	0.916
<(Bio Coke)> => <(Bio Coke)>	0.765	0.832	0.904

Fuente: Elaboración propia

- El producto pavo tiene como antecedente a la costilla de cordero con un soporte, confianza y levantamiento de 74%, 86% y 98,9% respectivamente.
- El producto manzana tiene como antecedente a costilla de cordero con un soporte, confianza y levantamiento de 70.5%, 82% y 97.6% respectivamente.
- El producto pavo tiene como antecedente a pavo con un soporte, confianza y levantamiento de 72.5%, 83.3% y 95.8% respectivamente.
- El producto coke tiene como antecedente a costilla de cordero con un soporte, confianza y levantamiento de 72.5%, 84.3% y 94.2% respectivamente.
- El producto pavo tiene como antecedente a bio coke con un soporte, confianza y levantamiento de 75%, 81.5% y 93.7% respectivamente.
- El producto manzana tiene como antecedente a costilla de cordero con un soporte, confianza y levantamiento de 70.5%, 82% y 97.6% respectivamente.
- El producto costilla de cordero tiene como antecedente a pavo con un soporte, confianza y levantamiento de 70%, 80.5% y 93.6% respectivamente.
- El producto manzana tiene como antecedente a bio coke con un soporte, confianza y levantamiento de 71.5%, 77.7% y 92.5% respectivamente.

- El producto costilla de cordero tiene como antecedente a bio coke con un soporte, confianza y levantamiento de 73%, 79.3% y 92.3% respectivamente.
- El producto pavo tiene como antecedente a coke con un soporte, confianza y levantamiento de 71.5%, 79.9% y 91.8% respectivamente.
- El producto bio coke tiene como antecedente a costilla de cordero con un soporte, confianza y levantamiento de 72.5%, 84.3% y 91.6% respectivamente.
- El producto bio coke tiene como antecedente a bio coke con un soporte, confianza y levantamiento de 76.5%, 83.2% y 90.4% respectivamente

4.5 IDENTIFICACIÓN DE LAS REGLAS UTILIZANDO OCURRENCIA DE SECUENCIA

Las reglas identificadas anteriormente relacionaban ítems a través de un tiempo futuro indeterminado, el problema es que no especifican dentro de cuánto volverán a comprar el siguiente ítem por lo que debe aplicar ocurrencia de secuencias para determinar el periodo de tiempo. Este periodo puede ser máximo o mínimo y es especificado por el investigador y pueden ser: un mes, dos meses, tres meses, un año, etc. Debido a que los algoritmos SPADE y GSP muestran resultados similares, se presentan las reglas por el algoritmo SPADE. En la Tabla 40 se identifican las 10 primeras reglas entre ítems para un tiempo máximo de tres meses y con un soporte de 60%. Las reglas identificadas se muestran en el Anexo 4.

Tabla 40: Principales reglas utilizando periodos de tiempo de tres meses

Regla	Soporte
<{152},{92}>	0.675
<{152},{152}>	0.665
<{92},{152}>	0.660
<{152},{43}>	0.655
<{152},{23}>	0.650
<{43},{152}>	0.65
<{43},{92}>	0.645
<{23},{152}>	0.635
<{23},{92}>	0.630
<{92},{43}>	0.630

Fuente: Elaboración propia

Utilizando ocurrencia de secuencia se puede obtener información más detallada y precisa. La primera regla que aparece en la Tabla 40 se puede interpretar de la siguiente manera "la probabilidad que un cliente compre pavo en un tiempo máximo de tres meses, dado que compró antes costillas de cordero es de 0.675". La interpretación para el cuarto patrón es "la probabilidad que un cliente compre costilla de cordero en un tiempo máximo de tres meses, dado que compró antes costilla bio coke es 0.68". El algoritmo SPADE encontró un total de 17 reglas de asociación.

A continuación se utiliza las ocurrencias de secuencia para un periodo de tiempo máximo de uno, dos y tres meses y la búsqueda de clases equivalentes para obtener resultados más precisos que lo mostrado en la Tabla 37 para el producto costilla de cordero (código 43) con un soporte mínimo de 60%.

Tabla 41: Reglas obtenidas con un periodo de tiempo de un mes

Regla			Soporte	Confianza	Levantamiento
<{43}>	=>	<{92}>	0.605	0.703	0.801
<{43}>	=>	<{152}>	0.630	0.733	0.796

Fuente: Elaboración Propia

Tabla 42: Reglas obtenidas con un periodo de tiempo de dos meses

Regla			Soporte	Confianza	Levantamiento
<{43}>	=>	<{92}>	0.62	0.721	0.829
<{43}>	=>	<{152}>	0.64	0.744	0.809

Fuente: Elaboración Propia

Tabla 43: Reglas obtenidas con un periodo de tiempo de tres meses

Regla			Soporte	Confianza	Levantamiento
<{43}>	=>	<{92}>	0.645	0.750	0.862
<{43}>	=>	<{43}>	0.610	0.709	0.824
<{43}>	=>	<{23}>	0.610	0.709	0.792
<{43}>	=>	<{152}>	0.650	0.756	0.822

Fuente: Elaboración Propia

Se puede observar de las Tablas 41, 42 y 43 que el soporte, confianza y levantamiento de la regla $< \{43\}, \{92\} >$ van aumentando a medida que el tiempo avanza. La interpretación para la primera regla obtenida de la Tabla 41 es: "existe un 60.5% de probabilidad que el cliente compre costilla de cordero y en un tiempo máximo de un mes compre pavo".

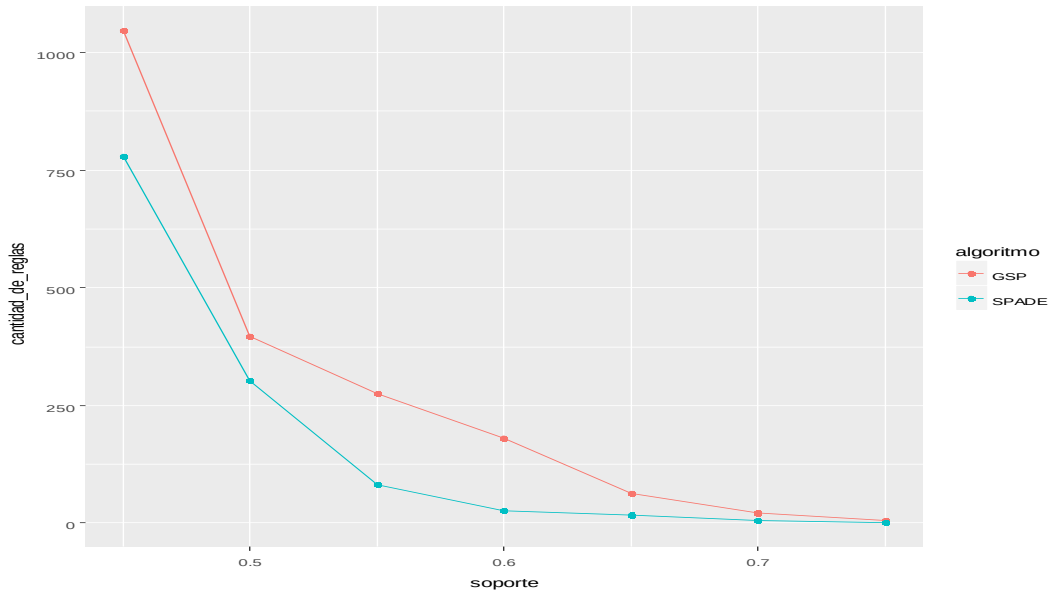
4.6 COMPARACIONES ENTRE EL ALGORITMO SPADE Y GSP

El algoritmo SPADE tiende a presentar relativamente más reglas que el SPADE, debido a que presenta los patrones de tamaño uno, esto se soluciona al momento de añadir otros indicadores como la confianza y el levantamiento. Fuera de este detalle, los dos algoritmos presentan los mismos resultados.

El algoritmo SPADE presenta una búsqueda más eficiente que el GSP debido a las clases equivalentes porque se puede visualizar los patrones de acuerdo al patrón paterno que se busque, ahorrando tiempo y cantidad de patrones generados. Adicionalmente se utilizaron las estrategias de búsqueda DFS para soporte menores que 0.6 y BFS para soportes mayores o iguales a 0.65, con el fin de reducir costos computacionales.

La Figura 18 muestra la comparación que se realizó tomando en cuenta la cantidad de reglas generadas entre el algoritmo secuencial SPADE utilizando clases equivalentes al ítem 43 y GSP.

Figura 18. Comparación entre los algoritmos en estudio (SPADE y GSP) según las reglas generadas



Fuente: Elaboración propia

El algoritmo SPADE resultó ser más eficiente en la cantidad de reglas generadas, debido a que utiliza una búsqueda por clases equivalentes reduciendo el universo de las reglas generadas. Por ejemplo, para un soporte mínimo de 0.65 el algoritmo GSP encontró 181 reglas, mientras que el algoritmo SPADE encontró 27.

V. CONCLUSIONES

1. Este trabajo de investigación surgió por la necesidad de encontrar una herramienta que pueda encontrar patrones a través del tiempo con una eficiente búsqueda, así facilitando la visualización de los patrones que se soliciten. Siendo una extensión de las reglas de asociación, mientras las reglas de asociación muestran un antecedente y un consecuente, en las reglas de asociación secuencial existe más de un consecuente, dependiendo de las transacciones de los clientes.

2. Se cumplió con el objetivo de estudiar las reglas de asociación secuencial y poder así entender esta rama de la minería de datos que es poco explotada. Se pudo complementar y añadir nuevos conocimientos de los algoritmos secuenciales presentado en Solano (2011), debido a que en su investigación se estudió el algoritmo secuencial GSP, y en esta investigación se estudiaron y compararon los algoritmos secuenciales GSP y el SPADE, añadiendo indicadores como la confianza y el levantamiento.

3. Los resultados mostrados tanto con SPADE y GSP resultaron muy similares, debido a que los dos utilizan el mismo enfoque a priori, siendo la mayor diferencia en el tipo de búsqueda. Mientras el GSP utiliza solo un tipo de búsqueda, el SPADE utiliza dos tipos de búsquedas dependiendo del soporte mínimo que se inserte. Resultando ser más eficiente.

4. Los resultados que se generan al utilizar las reglas de asociación secuencial son conjuntos de secuencias utilizando el tiempo para ordenar los eventos realizados. Una desventaja es la interpretación de los tiempos entre ítems debido a que no se cuantifica el tiempo entre ítems. Por ello se utiliza periodos de tiempo para cuantificar los tiempos entre ítems. Entre las reglas principales que se encontraron fueron: el patrón $< (43)(92) >$ que hace referencia a "la probabilidad que un cliente compre pavo en un futuro, dado que compró antes costillas de cordero es de 0.86", seguido del patrón $< (43)(23) >$ que hace referencia a "la probabilidad que un cliente compre Coke en

5. un futuro, dado que compró antes Costillas de Cordero es de 0.843". A diferencia de la investigación de Solano (2011), que determinó como el patrón más importante a $< \{152\}, \{92\} >$ tomando en cuenta solo el soporte.

6. Utilizando periodos de tiempo de tres meses se pudieron identificar 17 reglas de asociación, entre las cuales destaca $< (152)(92) >$, que se puede interpretar como "la probabilidad que un cliente compre pavo en un tiempo máximo de tres meses, dado que compró antes costillas de cordero es de 0.675" y el patrón $< (92)(152) >$, que se interpreta como "la probabilidad que un cliente compre bio coke en un tiempo máximo de tres meses, dado que compró antes pavo es 0.66".

7. Utilizando clases equivalentes y periodos de tiempo se pudo determinar que a medida que el tiempo transcurra, el soporte, confianza y el levantamiento de los productos fuertemente relacionados con las costillas de cordero incrementa. Estos productos son: pavo, bio coke y coke. Un patrón que se encontró fue que existe un 60.5% de probabilidad que el cliente compre costilla de cordero y en un tiempo máximo de un mes compre pavo.

VI. RECOMENDACIONES

1. El presente trabajo presenta un primer paso en el uso de técnicas de minería de datos con patrones secuenciales, estudiando los algoritmos SPADE y GSP. Quedaría como trabajo futuro utilizar otros enfoques o algoritmos que resulten más eficientes que los mostrados en esta investigación.

2. Trabajar los datos con un repositorio y previamente relacionados para una mayor rapidez a la hora de extraer los datos, brindando una mayor calidad y precisión de estos.

3. Utilizar variables geo-localizadores y enfocarlo en la rama de la Geo-Estadística encontrando patrones temporales de datos climáticos, temperaturas o fenómenos meteorológicos. Visualizando los resultados en un interfaz, mostrando los principales patrones, dependiendo de los soportes mínimos y confianza.

4. Es una realidad que en el Perú se integrarán los datos de la historia clínica, siendo importante encontrar los patrones de las enfermedades más raras. Encontrando asociaciones de síntomas previos a la enfermedad, por lo cual el algoritmo SPADE resulta fundamental a través de su búsqueda por clases equivalentes.

5. Aplicar los algoritmos secuenciales en la web, encontrando relaciones entre páginas de internet de los usuarios. Este enfoque es llamado Web Mining y recién se está explotando, por lo que estos algoritmos son fundamentales.

6. Aplicar los algoritmos a datos de instituciones peruanas como por ejemplo: Hospitales, Supermercados, etc para determinar posibles relaciones entre enfermedades/ ventas para la toma de decisiones.

VII. REFERENCIAS BIBLIOGRÁFICAS

Agrawal, R; Imielinski, T; Swami,A. (1993). Mining association rules between sets of items in large databases. ACM SIGMOD International Conference, Washington, p. 207–216.

Agrawal, R; Srikant, R. (1995). Mining sequential. ICDE '95 Proceedings of the Eleventh International Conference on Data Engineering, p. 3-14

Fayyad, U; Piatetsky-Shapiro, G; Smyth,P. (1996). From Data Mining to Knowledge Discovery in Databases. American Association for Artificial Intelligence., p. 37-54.

Gabrys, B; Howlett, RJ; Jain, LC. (eds.). (2006). Knowledge-Based Intelligent Information Systems: 10th International Conference, KES 2006, Bournemouth, UK, October 9-11 2006, Springer. 1324 p.

Gouda, K.; Hassan, M. (2011). Mining Sequential Pattern in Dense Databases. International Journal of Database Management Systems, p. 179-194.

Han, J; Kamber, M. (2006). Data mining: concepts and techniques. Kaufmann. 2 ed.

Jain, AK; Duin, RP; Mao, J. (2000). Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence, p. 4-37.

Kaur, H., & Singh, K. (2013). Market Basket Analysis of Sports Store using Association Rules. International Journal of Recent Trends in Electrical & Electronics Engineering, p.81-85.

Olson, DL; Delen, D. (2008). Advanced Data Mining Techniques. Springer. 180 p.

Qiankun, Z., & Bhowmick, S. S. (2003). Association Rule Mining: A Survey (No. 2003116). Technical Report, CAIS, Nanyang Technological University, Singapore.

Reps, J; Garibaldi, JM; Aickelin, U; Soria, D, Gibson, JE; Hubbard, RB (2012). Discovering Sequential Patterns in a UK General Practice Database. Biomedical and Health Informatics, p. 960-963.

Sakurai, S., Kitahara, Y., & Orihara, R. (2008). A sequential pattern mining method based on sequential interestingness. International Journal of Computational Intelligence, 4(4), 252-260.

Septiani, A., Febriani, A. & Fajriya, H. Sequential Pattern Mining of Hadith Narrator with Sequential Pattern Discovery Using Equivalent Classes (SPADE) Algorithm. *Conference on Waste Managemen, Ecology and Biological Sciencs*. Kuala Lumpur, Malasia, p. 147-154

Sha, P; Dua, A. (2014). CSPADE-UE: Algorithm for Sequence Mining for Unstructured Elements Using Time Gap Constraints. International Journal of Emerging Technology and Advanced Engineering, p. 592-596.

Silvestri, C. (2006). Distributed and Stream Data Mining Algorithms for Frequent Pattern Discovery. PhD. Thesis.Venezia: Universit`a Ca' Foscari di Venezia.

Slimani, T.; Lazzez, A. (2013). Sequential Mining: Patterns and Algorithm analysis. International Journal of Computer and Electronics Research, p. 639-647.

Solano, B. (2011). Descubrimiento de Patrones Secuenciales Utilizando Razonamiento Lógico Temporal.Tesis Mag. Rodrigo Facio,CR.

Srikant, R; Agrawal, R. (1996). Mining Sequential Patterns: Generalizations and Performance Improvements. Springer, p. 1-17.

Svetina, M; Zupančič, J. (2005). How to Increase Sales in Retail with Market Basket Analysis. Systems Integration, p. 418-428.

Verma, M; Mehta, D. (2014). Sequential Pattern Mining: A Comparison between GSP, SPADE and Prefix SPAN. IJEDR, p. 3016-3036.

Yang, Z. (2008). Fast Algorithms for Sequential Pattern Mining. PhD. Thesis. Tokio.

Ying-Ho, L. (2015). Mining time-interval univariate uncertain sequential patterns. Data & Knowledge Engineering, p. 1-24.

Zaki, M. (2000). Secuence Mining in Categoral Domains: Incorporating Constraints. In Proceedings of the ninth international conference on Information and knowledge management. ACM, p. 422-429.

Zaki, M. (2001). SPADE: An Efficient Algorithm for Mining: Frequent Sequences. Machine Learning, p. 31-60.

VIII. ANEXOS

ANEXO 1:

Lista de productos utilizados:

Categoría	Producto	Código
Fish	clam chowder	p_22
Fish	lobster	p_45
Fish	mussels	p_52
Fish	octopus	p_53
Fish	plaice	p_67
Fish	salmone	p_75
Fish	shrimps	p_79
Fish	swordfish	p_84
Fish	trout	p_88
Fish	Tuna	p_90
Meat	pork haunch	p_109
Meat	chicken	p_17
Meat	duck	p_27
Meat	lamb	p_42
Meat	lamm careé	p_43
Meat	beef	p_6
Meat	pork	p_68
Meat	spare ribs	p_81
Meat	T-bone steak	p_85
Meat	turkey	p_92
Meat	chicken breast	p_173
Meat	flank steak of lamb	p_190
Milk and egg products	yoghurt	p_100
Milk and egg products	yoghurt with fruit	p_118
Milk and egg products	cheese	p_16
Milk and egg products	egg	p_28
Milk and egg products	fried egg	p_30
Milk and egg products	ice cream	p_37
Milk and egg products	milk	p_49
Milk and egg products	milk fatfree	p_50
Milk and egg products	milk lactosefree	p_51
Milk and egg products	omlett	p_54
Categoría	Producto	Código

Milk and egg products	pancake	p_59
Milk and egg products	parmisano	p_60
Milk and egg products	scrambled egg	p_78
Milk and egg products	sour cream	p_80
Milk and egg products	American cheese	p_144
Milk and egg products	cheddar	p_170
Milk and egg products	Colby cheese	p_177
Milk and egg products	gorgonzola	p_193
Milk and egg products	gouda	p_194
Milk and egg products	gruyere	p_197
Milk and egg products	parmesan	p_218
Milk and egg products	pizza cheeze	p_225
Milk and egg products	provolone	p_230
Milk and egg products	romano	p_237
Milk and egg products	swiss cheese	p_248
Milk and egg products	tofu	p_250
Milk and egg products	yoghurt lowfat	p_259
Milk and egg products	yoghurt with fruit lowfat	p_260
Sweets		p_122
Sweets		p_123
Sweets		p_124
Sweets	choclate	p_20
Sweets	choclate peanuts	p_21
Sweets	cookies	p_24
Sweets	gummibears	p_34
Sweets	jelly-gums	p_39
Sweets	liquorice	p_44
Sweets	pralines	p_70
Vegetables	alium	p_1
Vegetables	zuccini	p_101
Vegetables	peanuts	p_110
Categoría	Producto	Código
Vegetables	hazelnuts	p_111

Vegetables	china kohl	p_18
Vegetables	cucumber	p_25
Vegetables	garlic	p_31
Vegetables	ginger	p_32
Vegetables	Aubergene	p_4
Vegetables	onions	p_55
Vegetables	potatoes	p_69
Vegetables	red herb	p_73
Vegetables	red patches	p_74
Vegetables	sauerkraut	p_77
Vegetables	tomato	p_87
Vegetables	blue herb	p_9
Vegetables	white kohl	p_97
Vegetables	Artichoke	p_146
Vegetables	Asparagus	p_147
Vegetables	Beans	p_149
Vegetables	Beet	p_150
Vegetables	Broccoli	p_159
Vegetables	Brussels sprouts	p_161
Vegetables	Cabbage	p_162
Vegetables	Carrot	p_163
Vegetables	Cauliflower	p_164
Vegetables	Celeriac	p_165
Vegetables	Celery	p_166
Vegetables	Chard	p_168
Vegetables	Chicory	p_174
Vegetables	Collards	p_178
Vegetables	Corn	p_179
Vegetables	Cress	p_183
Vegetables	Cucumbers	p_185
Vegetables	Gourds	p_195
Vegetables	Kales	p_200
Vegetables	Kohlrabi	p_203
Vegetables	Leek	p_204
Vegetables	Lettuce	p_206
Vegetables	Mushrooms	p_214
Vegetables	Okra	p_216
Vegetables	Parsnips	p_219
Vegetables	Peas	p_220
Vegetables	Peppers	p_221
Vegetables	Potatoes	p_229
Vegetables	Pumpkins	p_231
Vegetables	Radicchio	p_233
Vegetables	Radish	p_234
Categoría	Producto	Código
Vegetables	Rhubarb	p_235
Vegetables	Shallots	p_239

Vegetables	Spinach	p_242
Vegetables	Squash	p_244
Vegetables	Swede	p_246
Vegetables	Sweetcorn	p_247
Vegetables	Turnips	p_251
Vegetables	Watercress	p_253
Vegetables	Watermelon	p_254
Vegetables	Yams	p_258
alcoholic drinks	brandy	p_10
alcoholic drinks	campari	p_12
alcoholic drinks	chachasa	p_14
alcoholic drinks	champagne	p_15
alcoholic drinks	alkopops(wodky/strawberry)	p_2
alcoholic drinks	long island ice trea	p_46
alcoholic drinks	margerithas	p_48
alcoholic drinks	pisco	p_65
alcoholic drinks	beer	p_7
alcoholic drinks	uso	p_93
alcoholic drinks	Pinot Blanc(White wine)	p_98
alcoholic drinks	wodka	p_99
alcoholic drinks	alkopops(rum/cherry)	p_140
alcoholic drinks	alkopops(rum/lime)	p_141
alcoholic drinks	alkopops(tequilla/orange)	p_142
alcoholic drinks	alkopops(wodka/citron)	p_143
alcoholic drinks	Bordeaux (Red Wine)	p_156
alcoholic drinks	Chardonnay(White wine)	p_169
alcoholic drinks	Dornfelder(Red wine)	p_186
alcoholic drinks	Merlot(Red wine)	p_211
alcoholic drinks	Müller-Thurgau(White wine)	p_213
alcoholic drinks	Pinot Noir(Red wine)	p_223
alcoholic drinks	Riesling(White wine)	p_236
alcoholic drinks	Sauvignon Blanc(White wine)	p_238
alcoholic drinks	sparkling wine	p_241
bakery products	Choclate cake	p_102
bakery products	Vanilla cake	p_103
bakery products	Apple cake	p_104
bakery products	Brezels	p_11
bakery products	Donuts	p_26
bakery products	Toast	p_86
bakery products	Wedding cake	p_96
bakery products	bread	p_157
bakery products	brown bread	p_160
Categoría	Producto	Código
bakery products	corn pine	p_180
bakery products	crispbread	p_184
bakery products	French bread	p_191

bakery products	pita bread	p_224
bakery products	sunflower seed bread	p_245
bakery products	unleavened bread	p_252
bakery products	whole-grain bread	p_257
convenience foods	Ham sandwich	p_108
convenience foods	Chinese Food	p_19
convenience foods	French fries	p_29
convenience foods	Hamburger	p_35
convenience foods	Hot Dog	p_36
convenience foods	Ketchup	p_40
convenience foods	Pizza	p_66
convenience foods	Ranch Dressing Mix	p_71
convenience foods	Ravioli	p_72
convenience foods	Sauce bolognese	p_76
convenience foods	Tuna Sandwich	p_91
convenience foods	Egg sandwich	p_188
fruits	rasberries	p_119
fruits	peach	p_120
fruits	cherries	p_121
fruits	apples	p_3
fruits	grapes	p_33
fruits	kiwi	p_41
fruits	mango	p_47
fruits	bananas	p_5
fruits	orange	p_56
fruits	pears	p_62
fruits	pineapple	p_63
fruits	blackberries	p_8
fruits	strawberries	p_82
fruits	Apricot	p_145
fruits	Avocado	p_148
fruits	Blackcurrant	p_154
fruits	Blueberry	p_155
fruits	Breadfruit	p_158
fruits	Cherimoya	p_171
fruits	Clementine	p_175
fruits	Coconut	p_176
fruits	Cranberry	p_182
fruits	Durian	p_187
fruits	Fig	p_189
fruits	Grapefruit	p_196
fruits	Guava	p_198
Categoría	Producto	Código
fruits	Jackfruit	p_199
fruits	Kiwi	p_201
fruits	Lemon	p_205
fruits	Lime	p_207

fruits	Loganberry	p_208
fruits	Mandarin	p_209
fruits	Melon	p_210
fruits	Nectarine	p_215
fruits	Papaya	p_217
fruits	Persimmon	p_222
fruits	Plum	p_226
fruits	Pomegranate	p_227
fruits	Quince	p_232
fruits	Sharon Fruit (Persimmon)	p_240
fruits	Tamarillo	p_249
fruits	Watermelon	p_255
luxus	lobster	p_127
luxus	safran	p_128
luxus	spiny lobster	p_129
luxus	caviar	p_13
luxus	quail eggs	p_130
luxus	snails	p_131
luxus	oysters	p_132
luxus	ostrich eggs	p_133
luxus	paté de foie gras	p_61
luxus	Sushi	p_83
luxus	truffles	p_89
luxus	Best white tea	p_151
luxus	Champagne Esprit du Siècle	p_167
luxus	crabs	p_181
luxus	glenfiddich whisky	p_192
luxus	Kobe beef	p_202
luxus	Mouton Rothschild 1945	p_212
luxus	potato Bonnotte	p_228
luxus	white truffles	p_256
soft drinks	pure rock water	p_115
soft drinks	tap water	p_116
soft drinks	Apple Juice	p_117
soft drinks	Coke	p_23
soft drinks	Ice Tea	p_38
soft drinks	Orange Juice	p_57
soft drinks	Orange lemonade	p_58
soft drinks	pink lemonade	p_64
soft drinks	water	p_94
soft drinks	water with gas	p_95
Categoría	Producto	Código
soft drinks	Bio Coke	p_152
soft drinks	Bio lemonade	p_153
soft drinks	Cherry coke	p_172
soft drinks	spritzer	p_243

ANEXO 2:

Códigos R para la generación de reglas de asociación secuencial SPADE fijando el soporte

Cargar el paquete arulesSequences y arules del CRAN en R

```
library(arules)
library(arulesSequences)
#Introducir las listas
tmp_data=list(instertar las transacciones)
#Convertir las listas en transacciones
names(tmp_data) = paste("Tr",c(1:2738), sep = "")
trans = as(tmp_data,"transactions")
#Introducir la variable tiempo en las transacciones
transactionInfo(trans)$eventID=c(insertar los tiempos)
#Introducir la variable tiempo en las transacciones
transactionInfo(trans)$sequenceID=c(inserter los ID de los clientes)
#Reglas generadas con la estrategia de búsqueda BFS
m1 <- cspade(trans, parameter = list(support = 0.75),control = list(verbose = TRUE,
tidLists = TRUE, BFS=TRUE))
m2 <- cspade(trans, parameter = list(support = 0.7),control = list(verbose = TRUE, tidLists
= TRUE))
m3 <- cspade(trans, parameter = list(support = 0.65),control = list(verbose = TRUE,
tidLists = TRUE, BFS=TRUE))
#Reglas generadas con la estrategia de búsqueda DFS
m4<- cspade(trans, parameter = list(support = 0.6),control = list(verbose = TRUE, tidLists
= TRUE, bfstype=FALSE))
m5 <- cspade(trans, parameter = list(support = 0.55),control = list(verbose = TRUE,
tidLists = TRUE, bfstype= FALSE))
m6 <- cspade(trans, parameter = list(support = 0.5),control = list(verbose = TRUE, tidLists
= TRUE, bfstype=FALSE))
m7<- cspade(trans, parameter = list(support = 0.45),control = list(verbose = TRUE,
tidLists = TRUE, bfstype=FALSE))
```

```
m8<- cspade(trans, parameter = list(support = 0.4),control = list(verbose = TRUE, tidLists = TRUE, bfstype=FALSE))

Mostrar las reglas por cada soporte
m11<-as(m1, "data.frame")
m22<-as(m2, "data.frame")
m33<-as(m3, "data.frame")
m44<-as(m4, "data.frame")
m55<-as(m5, "data.frame")
m66<-as(m6, "data.frame")
m77<-as(m7, "data.frame")
m88<-as(m8, "data.frame")
```

ANEXO 3:

Obtener las reglas de asociación SPADE con el soporte, la confianza, el levantamiento y contrastes de tiempo

#Reglas generadas con confianza de 0.7

```
m7 <- cspade(trans, parameter = list(support = 0.75),control = list(verbose = TRUE,
tidLists = TRUE))
r2 <- ruleInduction(m7, confidence = 0.7, control = list(verbose = TRUE))
summary(r2)
as(r2, "data.frame")
```

#Reglas generadas con confianza 0

```
m0 <- cspade(trans, parameter = list(support = 0.75),control = list(verbose = TRUE,
tidLists = TRUE))
m00<-as(m0, "data.frame")
r0 <- ruleInduction(m0, confidence = 0.0,control = list(verbose = TRUE))
m1 <- cspade(trans, parameter = list(support = 0.7),control = list(verbose = TRUE, tidLists
= TRUE))
m11<-as(m1, "data.frame")
r1 <- ruleInduction(m1, confidence = 0.0,control = list(verbose = TRUE))

m2 <- cspade(trans, parameter = list(support = 0.65),control = list(verbose = TRUE,
tidLists = TRUE))
m22<-as(m2, "data.frame")
r2<- ruleInduction(m2, confidence = 0.0,control = list(verbose = TRUE))
m3<- cspade(trans, parameter = list(support = 0.6),control = list(verbose = TRUE, tidLists
= TRUE))

m33<-as(m3, "data.frame")
r3<- ruleInduction(m3, confidence = 0.0,control = list(verbose = TRUE))
m4 <- cspade(trans, parameter = list(support = 0.55),control = list(verbose = TRUE,
tidLists = TRUE))

m44<-as(m4, "data.frame")
r4<- ruleInduction(m4, confidence = 0.0,control = list(verbose = TRUE))
```

```
m5 <- cspade(trans, parameter = list(support = 0.5),control = list(verbose = TRUE, tidLists
= TRUE))

m55<-as(m5, "data.frame")

r5<- ruleInduction(m5, confidence = 0.0,control = list(verbose = TRUE))

m6 <- cspade(trans, parameter = list(support = 0.45),control = list(verbose = TRUE,
tidLists = TRUE))

m66<-as(m6, "data.frame")
r6<- ruleInduction(m6, confidence = 0.0, control = list(verbose = TRUE))
```

#Reglas generadas con contrastes de tiempo de tres meses y un soporte de 0.6

```
s2 <- cspade(trans, parameter = list(support = 0.6, maxgap = 93))
as(s2, "data.frame")
```

ANEXO 4:

Todas las reglas generadas de la Tabla 37

Regla	Soporte
<{152},{92}>	0.675
<{152},{152}>	0.665
<{92},{152}>	0.66
<{152},{43}>	0.655
<{152},{23}>	0.65
<{43},{152}>	0.65
<{43},{92}>	0.645
<{23},{152}>	0.635
<{23},{92}>	0.63
<{92},{43}>	0.63
<{70},{152}>	0.615
<{92},{92}>	0.61
<{43},{43}>	0.61
<{152},{3}>	0.61
<{92},{3}>	0.61
<{43},{23}>	0.61
<{152},{90}>	0.605

ANEXO 5:

Todas las reglas generadas de la Tabla 38

Regla	Soporte	Confianza	Levantamiento
<{43}> => <{92}>	0.740	0.860	0.989
<{43}> => <{23}>	0.725	0.843	0.942
<{43}> => <{152}>	0.725	0.843	0.916
<{43},{152}> => <{152}>	0.610	0.841	0.915
<{43},{124}> => <{152}>	0.505	0.835	0.907
<{43},{123}> => <{152}>	0.505	0.835	0.907
<{43},{122}> => <{152}>	0.505	0.835	0.907
<{43},{123,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,123,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,124}> => <{152}>	0.505	0.835	0.907
<{43},{122,123}> => <{152}>	0.505	0.835	0.907
<{43,152}> => <{152}>	0.565	0.831	0.903
<{43},{43}> => <{152}>	0.575	0.827	0.899
<{43,152}> => <{92}>	0.560	0.824	0.947
<{43},{70}> => <{152}>	0.535	0.823	0.895
<{152},{43}> => <{92}>	0.600	0.822	0.945
<{70},{43}> => <{92}>	0.530	0.822	0.944
<{92},{43}> => <{92}>	0.575	0.821	0.944
<{43}> => <{3}>	0.705	0.820	0.976
<{152},{43}> => <{152}>	0.595	0.815	0.886
<{70},{43}> => <{152}>	0.525	0.814	0.885
<{3},{43}> => <{92}>	0.500	0.813	0.934
<{124},{43}> => <{92}>	0.500	0.813	0.934
<{123},{43}> => <{92}>	0.500	0.813	0.934
<{122},{43}> => <{92}>	0.500	0.813	0.934
<{123,124},{43}> => <{92}>	0.500	0.813	0.934
<{122,124},{43}> => <{92}>	0.500	0.813	0.934
<{122,123,124},{43}> => <{92}>	0.500	0.813	0.934
<{122,123},{43}> => <{92}>	0.500	0.813	0.934
<{3},{43}> => <{152}>	0.500	0.813	0.884
<{43,152}> => <{43}>	0.550	0.809	0.940
<{43}> => <{43}>	0.695	0.808	0.940
<{92},{43}> => <{152}>	0.565	0.807	0.877
<{43},{152}> => <{3}>	0.585	0.807	0.961
<{43},{23}> => <{152}>	0.585	0.807	0.877

Continuación

Regla	Soporte	Confianza	Levantamiento
<{43},{90}> => <{92}>	0.515	0.805	0.925
<{43},{92}> => <{152}>	0.595	0.804	0.874
<{43,152}> => <{3}>	0.545	0.801	0.954
<{43,152}> => <{23}>	0.545	0.801	0.895
<{43},{70}> => <{92}>	0.520	0.800	0.920
<{23},{43}> => <{92}>	0.555	0.799	0.918
<{43},{90}> => <{3}>	0.510	0.797	0.949
<{43},{34}> => <{152}>	0.505	0.795	0.864
<{90},{43}> => <{92}>	0.520	0.794	0.913
<{43},{22}> => <{152}>	0.520	0.794	0.863
<{43},{23}> => <{92}>	0.575	0.793	0.912
<{43},{84}> => <{152}>	0.535	0.793	0.862
<{23},{43}> => <{152}>	0.550	0.791	0.860
<{152},{43}> => <{23}>	0.575	0.788	0.880
<{43},{3}> => <{152}>	0.555	0.787	0.856
<{43},{23}> => <{3}>	0.570	0.786	0.936
<{43},{152}> => <{92}>	0.570	0.786	0.904
<{92},{43}> => <{3}>	0.550	0.786	0.935
<{43}> => <{84}>	0.675	0.785	0.963
<{70},{43}> => <{3}>	0.505	0.783	0.932
<{43},{43}> => <{3}>	0.540	0.777	0.925
<{43},{43}> => <{92}>	0.540	0.777	0.893
<{92},{43}> => <{43}>	0.540	0.771	0.897
<{43},{43}> => <{43}>	0.535	0.770	0.895
<{43},{43}> => <{23}>	0.535	0.770	0.860
<{152},{43}> => <{3}>	0.560	0.767	0.913
<{43},{3}> => <{92}>	0.540	0.766	0.880
<{43},{152}> => <{43}>	0.555	0.766	0.890
<{90},{43}> => <{3}>	0.500	0.763	0.909
<{43},{84}> => <{3}>	0.515	0.763	0.908
<{23},{43}> => <{3}>	0.530	0.763	0.908
<{43}> => <{22}>	0.655	0.762	0.940
<{43},{152}> => <{23}>	0.550	0.759	0.848
<{92},{43}> => <{90}>	0.530	0.757	0.912
<{92},{43}> => <{23}>	0.530	0.757	0.846

Continuación

Regla	Soporte	Confianza	Levantamiento
<{43},{92}> => <{3}>	0.560	0.757	0.901
<{43},{92}> => <{92}>	0.560	0.757	0.870
<{43}> => <{70}>	0.650	0.756	0.911
<{43},{43}> => <{90}>	0.525	0.755	0.910
<{152},{43}> => <{70}>	0.550	0.753	0.908
<{43,152}> => <{90}>	0.510	0.750	0.904
<{43},{43}> => <{70}>	0.520	0.748	0.901
<{43},{23}> => <{43}>	0.540	0.745	0.866
<{43}> => <{90}>	0.640	0.744	0.897
<{92},{43}> => <{84}>	0.520	0.743	0.911
<{92},{43}> => <{70}>	0.520	0.743	0.895
<{43,152}> => <{84}>	0.505	0.743	0.911
<{23},{43}> => <{43}>	0.515	0.741	0.862
<{23},{43}> => <{23}>	0.515	0.741	0.828
<{43},{84}> => <{43}>	0.500	0.741	0.861
<{43},{84}> => <{92}>	0.500	0.741	0.851
<{152},{43}> => <{84}>	0.540	0.740	0.908
<{152},{43}> => <{43}>	0.540	0.740	0.860
<{43}> => <{34}>	0.635	0.738	0.895
<{43},{152}> => <{90}>	0.535	0.738	0.889
<{43},{3}> => <{43}>	0.520	0.738	0.858
<{43},{92}> => <{23}>	0.545	0.736	0.823
<{43,152}> => <{34}>	0.500	0.735	0.891
<{43,152}> => <{70}>	0.500	0.735	0.886
<{43},{43}> => <{84}>	0.510	0.734	0.900
<{152},{43}> => <{90}>	0.535	0.733	0.883
<{43},{92}> => <{43}>	0.540	0.730	0.849
<{43},{23}> => <{23}>	0.525	0.724	0.809
<{23},{43}> => <{90}>	0.500	0.719	0.867
<{43},{152}> => <{84}>	0.520	0.717	0.880
<{43},{152}> => <{70}>	0.520	0.717	0.864
<{92},{43}> => <{34}>	0.500	0.714	0.866
<{152},{43}> => <{22}>	0.520	0.712	0.879
<{152},{43}> => <{34}>	0.520	0.712	0.863
<{43},{152}> => <{22}>	0.515	0.710	0.877
<{43},{3}> => <{3}>	0.500	0.709	0.844
<{43}> => <{124}>	0.605	0.703	0.869
<{43}> => <{123,124}>	0.605	0.703	0.869
<{43}> => <{122,123,124}>	0.605	0.703	0.869
<{43}> => <{122,124}>	0.605	0.703	0.869
<{43}> => <{123}>	0.605	0.703	0.869

Regla	Soporte	Confianza	Levantamiento
<{43}> => <{122,123}>	0.605	0.703	0.869
<{43}> => <{122}>	0.605	0.703	0.869
<{152},{43}> => <{124}>	0.510	0.699	0.863
<{152},{43}> => <{123,124}>	0.510	0.699	0.863
<{152},{43}> => <{122,123,124}>	0.510	0.699	0.863
<{152},{43}> => <{122,124}>	0.510	0.699	0.863
<{152},{43}> => <{123}>	0.510	0.699	0.863
<{152},{43}> => <{122,123}>	0.510	0.699	0.863
<{152},{43}> => <{122}>	0.510	0.699	0.863
<{43},{152}> => <{34}>	0.505	0.697	0.844
<{43},{23}> => <{90}>	0.505	0.697	0.839
<{43},{152}> => <{124}>	0.500	0.690	0.851
<{43},{152}> => <{123,124}>	0.500	0.690	0.851
<{43},{152}> => <{122,123,124}>	0.500	0.690	0.851
<{43},{152}> => <{122,124}>	0.500	0.690	0.851
<{43},{152}> => <{123}>	0.500	0.690	0.851
<{43},{152}> => <{122,123}>	0.500	0.690	0.851
<{43},{152}> => <{122}>	0.500	0.690	0.851
<{43},{92}> => <{22}>	0.510	0.689	0.851
<{43},{92}> => <{84}>	0.505	0.682	0.837
<{43}> => <{45}>	0.580	0.674	0.977
<{43}> => <{45,127}>	0.580	0.674	0.977
<{43}> => <{127}>	0.580	0.674	0.977
<{43}> => <{29}>	0.555	0.645	0.978
<{43}> => <{256}>	0.550	0.640	0.962
<{43}> => <{40}>	0.545	0.634	0.975
<{43}> => <{61}>	0.545	0.634	0.925
<{43}> => <{35}>	0.545	0.634	0.899
<{43}> => <{43,152}>	0.540	0.628	0.923
<{43}> => <{167}>	0.540	0.628	0.917
<{43}> => <{188}>	0.535	0.622	0.980
<{43}> => <{23,92}>	0.530	0.616	0.971
<{43}> => <{92,152}>	0.530	0.616	0.913
<{43}> => <{2}>	0.525	0.610	0.969
<{43}> => <{67}>	0.525	0.610	0.954
<{43}> => <{71}>	0.520	0.605	0.923
<{43}> => <{76}>	0.520	0.605	0.896
<{43}> => <{87}>	0.515	0.599	0.936
<{43}> => <{66}>	0.515	0.599	0.921
<{43}> => <{20}>	0.515	0.599	0.881
<{43}> => <{72}>	0.510	0.593	0.879
<{43}> => <{130}>	0.505	0.587	0.932
<{43}> => <{19}>	0.500	0.581	0.894

ANEXO 6:

Obtener las reglas utilizando clases equivalentes al ítem 43

#Soporte de 0.75

```
m000<-as(subset(r0, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 0.7

```
m111<-as(subset(r1, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 0.65

```
m222<-as(subset(r2, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 0.6

```
m333<-as(subset(r3, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 0.55

```
m444<-as(subset(r4, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 0.5

```
m555<-as(subset(r5, lhs(x) %ain% c("152")), "data.frame")
```

#Soporte 045

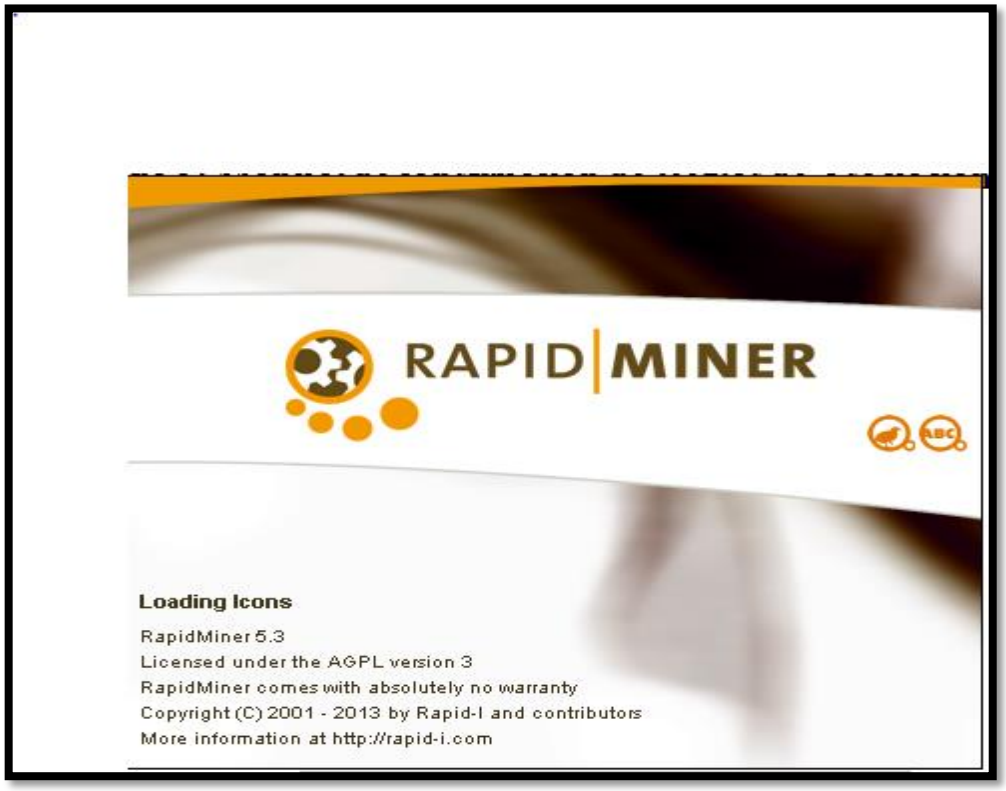
```
m666<-as(subset(r6, lhs(x) %ain% c("152")), "data.frame")
```

#Grafico 17 en ggplot2

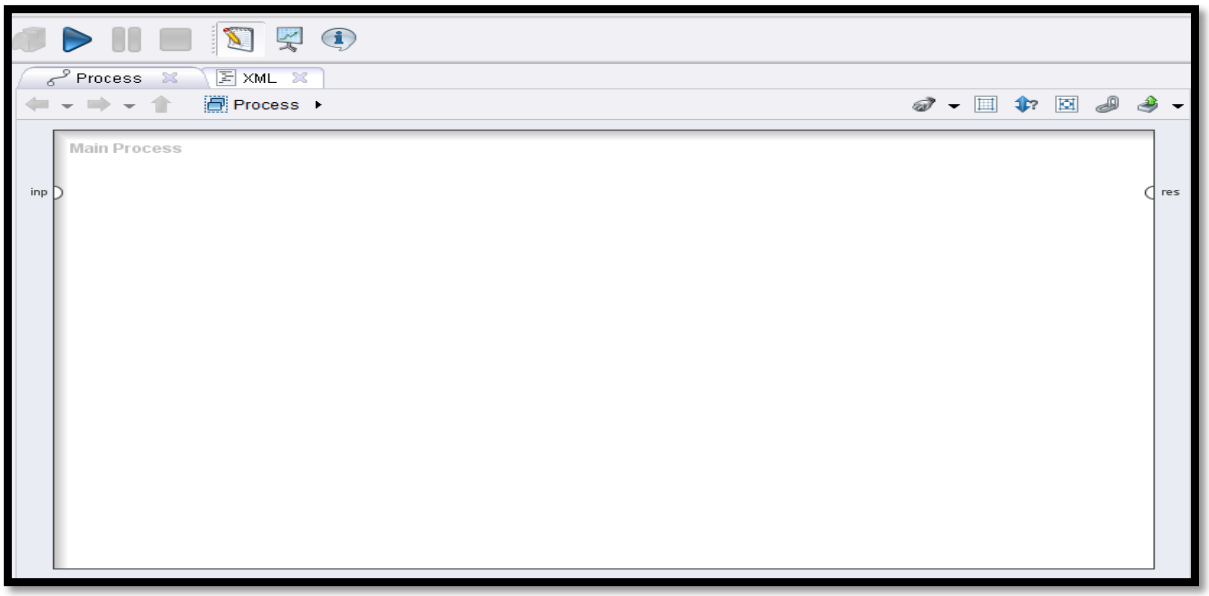
```
library(ggplot2)
data<-data.frame(reglas=reglas,algoritmo=algoritmo,soporte=soporte)
ggplot(data, aes(x = soporte, y = cantidad_de_reglas)) +
+   geom_line(aes(color = algoritmo, group=algoritmo)) +geom_point(aes(color =
algoritmo))
```

ANEXO 7:

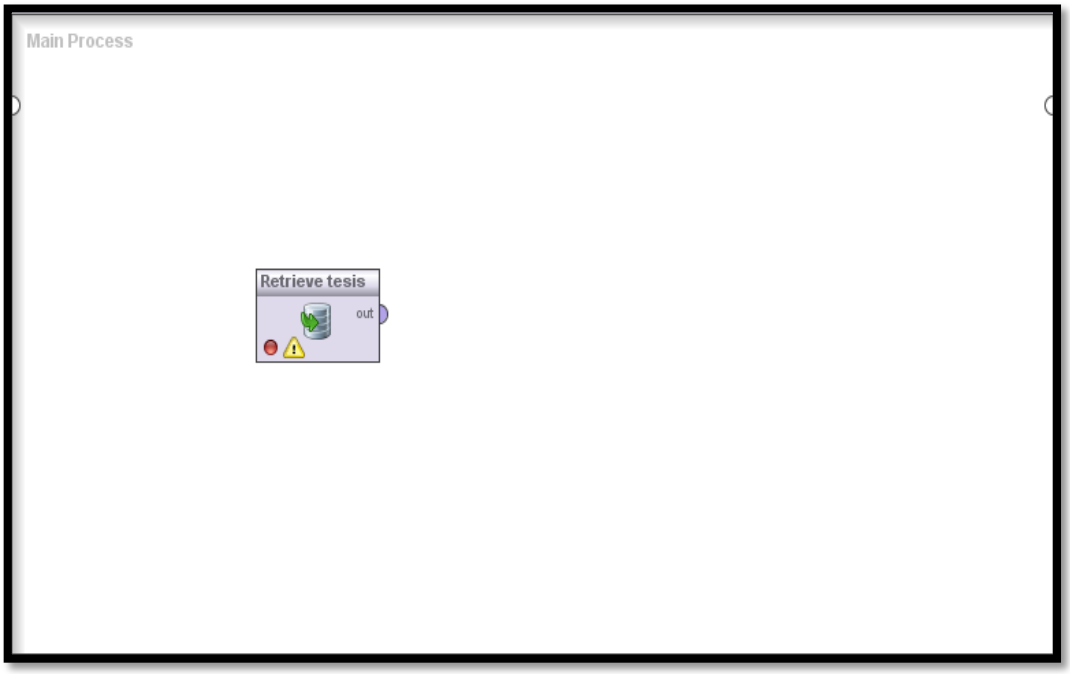
Manual de construcción de Reglas de Asociación con Rapidminer versión 5.3



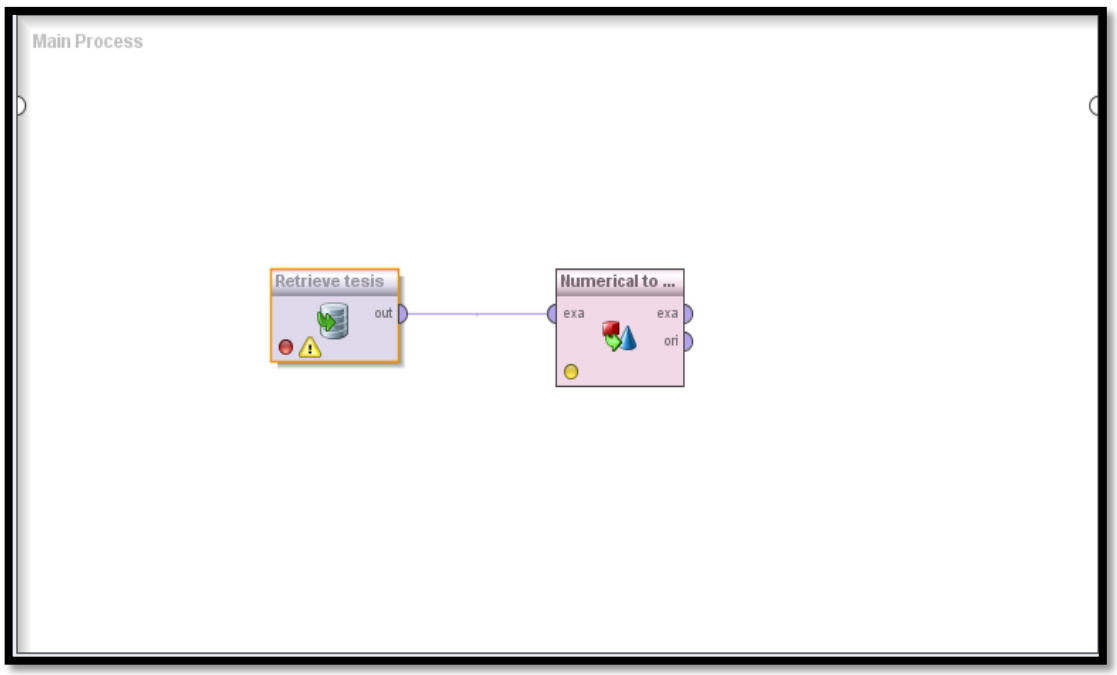
Paso 1: Abrir el programa Rapidminer 5.3



Paso 2: Seleccionar la paleta file, luego import data y dentro de ella escoger el tipo de archivo en la cual se encuentra los datos (en este caso se seleccionó Archivo Tesis).



Paso 3: Seleccionar la paleta Operaciones, luego hacer click en data transformation y dentro de ellas seleccionar type convertion. Finalmente colocar la opción Numerical to Binomial y unir el nodo con la data tesis.



Paso 4: Seleccionar la paleta Operaciones, luego hacer click en modeling y dentro de ellas seleccionar Association and Itemset mining .Finalmente colocar la opción Generalized Sequential Pattern y unir el nodo con Numerical to Binomial. En este paso colocar las variables Customer Id, Tiem atrribute y colocar el mínimo soporte que se desea emplear. Posteriormente click en "run"