PONTIFICIA UNIVERSIDAD CATOLICA MADRE Y MAESTRA

CAMPUS SANTO TOMÁS DE AQUINO

Nombres:

Andrés Andino - 10138676

Eliam Pimentel - 10138836

SC446 - 201

Minería de Datos

Tema:

Trabajo de investigación grupal sobre algoritmos de minería de datos/Machine Learning

Profesor/a:

Luis Rafael Reyes Castillo

Santo Domingo, D.N

03/02/2024

Índice

Introducción

Los **algoritmos de asociación** son herramientas de minería de datos que se utilizan para descubrir relaciones entre diferentes elementos o atributos dentro de un conjunto de transacciones. Su objetivo principal es identificar patrones de co-ocurrencia que indiquen la probabilidad de que ciertos elementos aparezcan juntos con cierta frecuencia.

En este contexto, el término transacción hace referencia a cada grupo de eventos que están asociados de alguna forma, por ejemplo:

- El carrito de compra en un supermercado.
- Los libros que compra un cliente en una librería.
- Las páginas web visitadas por un usuario.

Estos algoritmos buscan asociaciones entre elementos que se compran juntos, se visitan juntos en un sitio web o se presentan juntos en un conjunto de datos.

- **¿Cómo funcionan estos algoritmos?**

Los algoritmos de reglas de asociación generalmente se basan en dos pasos:

1. Encontrar conjuntos de elementos frecuentes.

Se analiza el conjunto de datos para identificar grupos de elementos que aparecen juntos con una frecuencia superior a un umbral predefinido, llamado soporte mínimo. Estos grupos de elementos se denominan conjuntos frecuentes.

2. Generar reglas de asociación.

A partir de los conjuntos frecuentes, se generan reglas que expresan la relación entre dos o más conjuntos de elementos. Estas reglas se evalúan utilizando dos métricas:

- Confianza: Indica la probabilidad de que el antecedente de la regla implique el consecuente.
- Levantamiento: Mide si la asociación entre los elementos es más fuerte de lo que cabría esperar por casualidad.

- **¿Para qué se utilizan?**

Los algoritmos de reglas de asociación se pueden utilizar en una amplia variedad de aplicaciones, como:

a) Análisis de mercado:
   - Identificar productos que se compran juntos con frecuencia (por ejemplo pan y leche).
   - Descubrir patrones de compra de diferentes segmentos de clientes.

b) Análisis web:
   - Encontrar páginas web que se visitan juntas con frecuencia.
   - Recomendar contenido a los usuarios basado en su historial de navegación.

c) Análisis de datos médicos:
   - Identificar factores de riesgo para enfermedades.
   - Descubrir nuevos tratamientos para enfermedades.

- **Ventajas de los algoritmos de asociación:**

    - Son fáciles de entender e interpretar.
    - Se pueden aplicar a una amplia variedad de conjuntos de datos.
    - Son relativamente eficientes desde el punto de vista computacional.

- **Desventajas de los algoritmos de asociación:**

    - Pueden generar un gran número de reglas, lo que dificulta su análisis.
    - No son sensibles al orden de los elementos.
    - Pueden ser sensibles al ruido en los datos.

A cada uno de los elementos que forman parte de una transacción se le conoce como item y a un conjunto de ellos itemset. Una transacción puede estar formada por uno o varios items, en el caso de ser varios, cada posible subconjunto de ellos es un itemset distinto.

Por ejemplo, la transacción T = {A,B,C} está formada por 3 items (A, B y C) y sus posibles itemsets son: {A,B,C}, {A,B}, {B,C}, {A,C}, {A}, {B} y {C}.

Una regla de asociación se define como una implicación del tipo “si X entonces Y” (X⇒Y), donde X e Y son items individuales. El lado izquierdo de la regla recibe el nombre de antecedente y el lado derecho el nombre de consecuente.

Por ejemplo, la regla {A,B} => {C} significa que, cuando ocurren A y B, también ocurre C.

- **Ámbitos de aplicación**

Las reglas de asociación son una herramienta poderosa que va más allá del análisis de cestas de compra en el sector minorista. Su capacidad para identificar relaciones entre diferentes elementos las convierte en una herramienta invaluable para diversos sectores que buscan optimizar sus procesos y tomar decisiones estratégicas.

Más allá del retail:

- **Medicina:** Las reglas de asociación basadas en datos de salud pueden ayudar a mejorar el diagnóstico y la prevención de enfermedades. Por ejemplo, al identificar patrones de síntomas que predicen la aparición de una enfermedad, se pueden tomar medidas tempranas para prevenirla.

- **Diseño UX:** Las reglas de asociación permiten analizar el comportamiento de los usuarios y adaptar la interfaz para que sea más intuitiva y fácil de usar. Por ejemplo, se pueden modificar la ubicación de los botones o la organización de los contenidos.

- **Gestión de almacenes:** En la logística, las reglas de asociación permiten optimizar la ubicación de los productos en un almacén. Al identificar qué productos se suelen comprar juntos, se pueden colocar estratégicamente para facilitar su picking y aumentar la eficiencia de la operación.

Existen varios algoritmos diseñados para identificar elementos frecuentes. Estos son los siguientes:

- Algoritmo Apriori
- Algoritmo FP-Growth
- Algoritmo FilteredAssociator
- Algoritmo Eclat
- Algoritmo SETM

Algoritmo Apriori

El algoritmo Apriori fue desarrollado en 1994 por Rakesh Agrawal y Ramakrishnan Srikant, investigadores del IBM Almaden Research Center. Su objetivo era encontrar una forma eficiente de identificar conjuntos de items frecuentes en conjuntos de datos transaccionales.

Antes del desarrollo de Apriori, los algoritmos existentes para la minería de reglas de asociación eran muy lentos e ineficientes. Esto limitaba su uso a conjuntos de datos pequeños.

Apriori introdujo un enfoque iterativo para la búsqueda de conjuntos frecuentes. Este enfoque comienza por encontrar conjuntos de items de un solo item que son frecuentes. Luego, se van extendiendo estos conjuntos un item por vez, hasta que no se encuentren más conjuntos frecuentes.

El desarrollo del algoritmo Apriori tuvo un gran impacto en el campo de la minería de datos. Permitió el análisis de conjuntos de datos mucho más grandes y complejos de lo que era posible antes.

Desde su desarrollo, el algoritmo Apriori ha sido objeto de numerosas mejoras y modificaciones. Se han desarrollado algoritmos más eficientes, como FP-Growth y Eclat, que se basan en el enfoque de Apriori.

Este algoritmo sigue siendo uno de los algoritmos más utilizados para la minería de reglas de asociación. Es una herramienta fundamental para el análisis de datos transaccionales en una amplia variedad de aplicaciones.

★ Ejemplo

Supongamos que se tiene un conjunto de datos que consiste en transacciones, donde cada transacción contiene un conjunto de elementos…

Transacción 1: A, B, C
Transacción 2: A, C, D, E
Transacción 3: B, C, E
Transacción 4: A, B, E

- **Paso 1: Generación de conjuntos de 1-elemento (items frecuentes)**

    - Se enumeran todos los elementos individuales presentes en las transacciones.
    - Se cuenta la frecuencia de cada elemento.
    - Se eliminan los elementos que no cumplen con un umbral mínimo de frecuencia (umbral de soporte).

Si establecemos un umbral de soporte mínimo en 2, los elementos frecuentes de 1-elemento podrían ser A, B, C y E.

- **Paso 2: Generación de conjuntos de 2-elementos (pares frecuentes)**

    - Se combinan los elementos frecuentes de 1-elemento para formar conjuntos de 2-elementos (pares).
    - Se cuenta la frecuencia de cada conjunto de 2-elementos.
    - Se eliminan los conjuntos que no cumplen con el umbral mínimo de frecuencia.

Si establecemos un umbral de soporte mínimo en 2, los conjuntos frecuentes de 2-elementos podrían ser {A, B}, {A, C}, {A, E}, {B, C}, {B, E}, {C, E}.

- **Paso 3: Generación de conjuntos de k-elementos (k > 2)**

Similar al paso 2, pero ahora se combinan los conjuntos frecuentes de k-1 elementos para formar conjuntos de k-elementos.

    - Se cuenta la frecuencia de cada conjunto de k-elementos.
    - Se eliminan los conjuntos que no cumplen con el umbral mínimo de frecuencia.
    - El proceso se repite hasta que no se pueden generar conjuntos de k-elementos frecuentes.

Usando el ejemplo anterior y estableciendo un umbral de soporte mínimo en 2, podríamos continuar generando conjuntos de 3-elementos, 4-elementos, y así sucesivamente.

Algoritmo FP-Growth

El algoritmo FP-Growth (Frequent Pattern Growth) se utiliza para encontrar conjuntos frecuentes y reglas de asociación en conjuntos de datos transaccionales. Es una alternativa más eficiente al algoritmo Apriori.

- **Funcionamiento del algoritmo FP-Growth:**

Construcción del árbol FP:

1. Se crea un árbol FP-Tree (Frequent Pattern Tree) a partir del conjunto de datos transaccionales. Este árbol es una estructura compacta que representa los items y sus frecuencias.
2. Se recorre el árbol FP de forma recursiva para encontrar los conjuntos frecuentes.
3. A partir de los conjuntos frecuentes, se generan reglas de asociación que expresan la relación entre dos o más conjuntos de items.

- **Ventajas del algoritmo FP-Growth:**

    - Es más eficiente que el algoritmo Apriori, especialmente para conjuntos de datos grandes.
    - Requiere menos memoria que el algoritmo Apriori.
    - Es más fácil de implementar que el algoritmo Apriori.

- **Desventajas del algoritmo FP-Growth:**

    - Puede ser menos eficiente que el algoritmo Apriori para conjuntos de datos pequeños.
    - No es sensible al orden de los items.
    - Puede ser sensible al ruido en los datos.

★ **Ejemplo**

Supongamos que se tiene un conjunto de datos que consiste en transacciones, donde cada transacción contiene un conjunto de elementos…

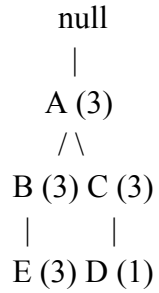
Transacción 1: A, B, C
Transacción 2: A, C, D, E
Transacción 3: B, C, E
Transacción 4: A, B, E

- **Paso 1: Construcción del Árbol FP-Growth**

    - Se enumeran todos los ítems únicos junto con su frecuencia de aparición en todas las transacciones y ordena los ítems en orden descendente según su frecuencia.

Ítem	Frecuencia
A	3
B	3
C	3
E	3
D	1

    - Para cada transacción, ordena los ítems según su frecuencia en la tabla de encabezados y construye el Árbol FP-Growth. Para el ejemplo, el Árbol FP-Growth sería algo así:



- **Paso 2: Extracción de conjuntos frecuentes**

    - Se comienza con el ítem menos frecuente (D en este caso) y construimos conjuntos frecuentes siguiendo las rutas en el Árbol FP-Growth.
    - Se construyen conjuntos frecuentes recursivamente: Por ejemplo, para el ítem D, las rutas serían D -> C y D -> C -> E. Estas rutas forman conjuntos frecuentes {D}, {C, D}, {C, D, E}.
    - Se repite este proceso para el ítem siguiente en la tabla de encabezados (E en este caso) y construir conjuntos frecuentes.

Resultado Final:

Los conjuntos frecuentes obtenidos serían los mismos que en el ejemplo Apriori:
{A}, {B}, {C}, {E}, {A, B}, {A, C}, {A, E}, {B, C}, {B, E}, {C, E}.

Algoritmo FilteredAssociator

El algoritmo FilteredAssociator fue introducido por primera vez en el año 1995 por los investigadores Agrawal y Srikant en su artículo "Fast Algorithms for Mining Association Rules". En este artículo, los autores presentan el algoritmo como una técnica eficiente para descubrir reglas de asociación en grandes conjuntos de datos.

El desarrollo del algoritmo FilteredAssociator fue motivado por la necesidad de mejorar la eficiencia de los algoritmos de minería de reglas de asociación existentes. Los algoritmos tradicionales, como el algoritmo Apriori, podían ser computacionalmente costosos cuando se aplicaban a grandes conjuntos de datos.

El algoritmo FilteredAssociator es una técnica de minería de datos que se utiliza para descubrir reglas de asociación en grandes conjuntos de datos. Se basa en la idea de filtrar las transacciones que no son relevantes para la búsqueda de reglas, lo que reduce el espacio de búsqueda y mejora la eficiencia del proceso.

Funcionamiento:

1. **Definición del problema:** Se define el problema de minería de datos, incluyendo los atributos relevantes, el soporte mínimo y la confianza mínima para las reglas.
2. **Filtrado de transacciones:** Se eliminan las transacciones que no contienen ningún elemento de interés o que no cumplen con los criterios de frecuencia mínima.
3. **Generación de candidatos:** Se generan candidatos a reglas de asociación utilizando diferentes estrategias, como la generación de itemsets frecuentes o la búsqueda exhaustiva.
4. **Evaluación de candidatos:** Se evalúan los candidatos a reglas utilizando medidas como el soporte, la confianza y el lift.
5. **Selección de reglas:** Se seleccionan las reglas que cumplen con los criterios de soporte mínimo y confianza mínima.

Ventajas:

- **Eficiencia:** El algoritmo FilteredAssociator es más eficiente que otros algoritmos de minería de reglas de asociación, ya que reduce el espacio de búsqueda.
- **Escalabilidad:** El algoritmo es escalable a grandes conjuntos de datos.
- **Flexibilidad:** El algoritmo se puede adaptar a diferentes tipos de problemas de minería de datos.

Desventajas:

- **Pérdida de información:** El filtrado de transacciones puede eliminar información relevante para la búsqueda de reglas.
- **Complejidad:** La elección de la estrategia de filtrado y la evaluación de candidatos pueden ser tareas complejas.

Ejemplo.

Supongamos que tenemos un conjunto de datos de transacciones de una tienda de comestibles. Queremos descubrir reglas de asociación que indiquen qué productos se compran juntos con frecuencia.

- **Definición del problema:** Los atributos relevantes son los productos que se venden en la tienda. El soporte mínimo se establece en 5% y la confianza mínima se establece en 70%.
- **Filtrado de transacciones:** Se eliminan las transacciones que no contienen ningún producto de interés.
- **Generación de candidatos:** Se generan candidatos a reglas de asociación utilizando la generación de itemsets frecuentes.
- **Evaluación de candidatos:** Se evalúan los candidatos a reglas utilizando el soporte, la confianza y el lift.
- **Selección de reglas:** Se seleccionan las reglas que cumplen con los criterios de soporte mínimo y confianza mínima.

Reglas:

- **Leche -> Pan (soporte: 10%, confianza: 80%)**
- **Pan -> Mantequilla (soporte: 8%, confianza: 75%)**
- **Leche -> Mantequilla (soporte: 6%, confianza: 70%)**

Estas reglas indican que los clientes que compran leche también suelen comprar pan y mantequilla.

Algoritmo Eclat

El algoritmo Eclat, presentado por Zaki y Aggarwal en 1997, ha revolucionado la minería de reglas de asociación al ofrecer una técnica más eficiente para descubrir patrones relevantes en grandes conjuntos de datos.

Los algoritmos tradicionales, solían ser costosos y consumir grandes cantidades de memoria, limitando su aplicabilidad. Eclat buscaba superar estas limitaciones mediante la generación de candidatos a partir de itemsets cerrados, reduciendo significativamente el espacio de búsqueda.

Funcionamiento:

1. **Generación de itemsets cerrados:** Eclat identifica conjuntos de elementos frecuentes que no son subconjuntos de ningún otro conjunto de elementos frecuente, utilizando un algoritmo específico.
2. **Generación de candidatos a reglas:** A partir de los itemsets cerrados, se generan candidatos a reglas utilizando diferentes estrategias, como la generación por pares o por subconjuntos.

3. **Evaluación de candidatos:** Se evalúan las reglas utilizando medidas como el soporte, la confianza y el lift.
4. **Selección de reglas:** Se seleccionan las reglas que cumplen con los criterios de soporte mínimo y confianza mínima.

Ventajas:

- **Eficiencia:** Eclat reduce significativamente el tiempo de procesamiento y el uso de memoria en comparación con algoritmos tradicionales.
- **Escalabilidad:** Eclat es capaz de manejar grandes conjuntos de datos con eficiencia.
- **Precisión:** Eclat genera reglas de asociación con mayor precisión que otros algoritmos.

Desventajas:

- **Complejidad:** La generación de itemsets cerrados puede ser una tarea compleja para conjuntos de datos grandes.
- **Memoria:** El algoritmo Eclat puede requerir una cantidad significativa de memoria para almacenar los itemsets cerrados.

Ejemplo.

Supongamos que tenemos un conjunto de datos de transacciones de una tienda de comestibles. Queremos descubrir reglas de asociación que indiquen qué productos se compran juntos con frecuencia.

- **Generación de itemsets cerrados:** El algoritmo Eclat genera los siguientes itemsets cerrados:
    - {Leche}
    - {Pan}
    - {Mantequilla}
    - {Leche, Pan}
    - {Leche, Mantequilla}
- **Generación de candidatos a reglas:** A partir de los itemsets cerrados, el algoritmo Eclat genera los siguientes candidatos a reglas:
    - Leche -> Pan
    - Pan -> Leche
    - Leche -> Mantequilla
    - Mantequilla -> Leche
    - Leche, Pan -> Mantequilla
- **Evaluación de candidatos a reglas:** Los candidatos a reglas se evalúan utilizando el soporte, la confianza y el lift.
- **Selección de reglas:** Se seleccionan las reglas que cumplen con los criterios de soporte mínimo y confianza mínima.

Reglas:

- Leche -> Pan (soporte: 10%, confianza: 80%)
- Pan -> Leche (soporte: 8%, confianza: 75%)
- Leche -> Mantequilla (soporte: 6%, confianza: 70%)

Estas reglas indican que los clientes que compran leche también suelen comprar pan y mantequilla.

Algoritmo SETM

El algoritmo SETM, presentado por Pei et al. en 2001, se ha convertido en una herramienta fundamental para la minería de patrones secuenciales. Este algoritmo ofrece un enfoque eficiente para descubrir patrones de comportamiento en secuencias de eventos, como las compras de un cliente en una tienda online o las acciones de un usuario en un sitio web.

SETM adopta una estrategia de "minería sin candidatos", donde los patrones se generan directamente a partir de la base de datos sin necesidad de generar candidatos intermedios.

Funcionamiento:

1. **Construcción de la tabla sufijo:** SETM crea una tabla que registra las ocurrencias de cada sufijo en la secuencia.
2. **Identificación de patrones frecuentes:** Se utiliza un algoritmo de barrido vertical para identificar los patrones que cumplen con el criterio de frecuencia mínima.
3. **Generación de reglas de asociación:** A partir de los patrones frecuentes, se generan reglas de asociación que establecen relaciones entre los elementos de la secuencia.

Ventajas:

- **Eficiencia:** SETM es significativamente más eficiente que los algoritmos tradicionales, especialmente al trabajar con secuencias largas o datasets extensos.
- **Escalabilidad:** SETM es capaz de manejar grandes conjuntos de datos con eficiencia.
- **Precisión:** SETM genera patrones precisos y relevantes para el análisis de la secuencia.

Desventajas:

Sensibilidad al orden:

- SETM es sensible al orden de los elementos en la secuencia. Un cambio en el orden puede afectar la frecuencia de los patrones y las reglas de asociación generadas.

Dificultad para interpretar reglas:

- Las reglas de asociación generadas por SETM pueden ser difíciles de interpretar, especialmente para datasets con muchos elementos o secuencias largas.
- La identificación de los patrones relevantes y la comprensión de su significado puede requerir un análisis adicional y conocimiento del dominio específico.

Ejemplo.

Supongamos que tenemos un conjunto de datos de secuencias de acciones de usuarios en una aplicación de música:

- **Usuario 1:** {Reproducir canción A, Reproducir canción B, Agregar canción C a la lista de reproducción}
- **Usuario 2:** {Reproducir canción B, Buscar canción C, Reproducir canción D}
- **Usuario 3:** {Agregar canción A a la lista de reproducción, Reproducir canción B, Buscar canción C}
- **Usuario 4:** {Reproducir canción C, Agregar canción D a la lista de reproducción, Buscar canción A}

Objetivo:

Descubrir patrones de comportamiento frecuentes en las secuencias de acciones de los usuarios.

Ejecución del Algoritmo SETM:

1. **Construcción de la tabla sufijo:**

Sufijo	**Frecuencia**
Reproducir canción	4
Agregar canción a la lista de reproducción	3
Buscar canción	3
Canción A	2
Canción B	2
Canción C	2
Canción D	1

2. **Identificación de patrones frecuentes:**
   - **Reproducir canción** (frecuencia > 2)
   - **Agregar canción a la lista de reproducción** (frecuencia > 2)
   - **Buscar canción** (frecuencia > 2)

3. **Generación de reglas de asociación:**
   - **Reproducir canción -> Agregar canción a la lista de reproducción** (confianza: 75%)
   - **Reproducir canción -> Buscar canción** (confianza: 66%)
   - **Buscar canción -> Reproducir canción** (confianza: 50%)

Interpretación de los resultados:

- **Reproducir canción -> Agregar canción a la lista de reproducción:** El 75% de los usuarios que reproducen una canción la agregan a la lista de reproducción en la misma secuencia.
- **Reproducir canción -> Buscar canción:** El 66% de los usuarios que reproducen una canción buscan otra canción en la misma secuencia.
- **Buscar canción -> Reproducir canción:** El 50% de los usuarios que buscan una canción la reproducen en la misma secuencia.

Conclusión

La naturaleza de estos algoritmos como herramientas poderosas en la minería de datos nos ayudan a descubrir relaciones entre elementos en conjuntos de transacciones. Además, se destacan varios ámbitos de aplicación, como la medicina, el diseño UX y la gestión de almacenes, donde las reglas de asociación pueden ser utilizadas para mejorar el diagnóstico médico, la experiencia del usuario y la eficiencia logística.

El algoritmo Apriori, se presenta como un hito importante en este campo, al introducir un enfoque iterativo para encontrar conjuntos frecuentes de items en datos transaccionales. Su impacto se refleja en la capacidad para analizar conjuntos de datos más grandes y complejos, aunque desde entonces han surgido algoritmos más eficientes basados en el enfoque de Apriori, como FP-Growth y Eclat.

A medida que se han producido avances en la minería de datos, las reglas de asociación se pueden utilizar en una gama más amplia de casos de uso. Cada día se están generando más datos, lo que significa más aplicaciones para reglas de asociación.