Algoritmo Apriori

Prof. Dr. Jorge Zavaleta
zavaleta.jorge@gmail.com

Algoritmo Apriori - introducción

- El algoritmo **Apriori**, es fundamental en la minería de datos para descubrir **itemsets frecuentes** y **reglas de asociación** en grandes conjuntos de transacciones.
- Se utiliza en diversos campos para descubrir **conjuntos de elementos frecuentes** y **reglas de asociación** en grandes conjuntos de transacciones.
- Su versatilidad lo convierte en una poderosa herramienta en áreas como:
  - **Minorista**: Análisis de cesta de compras. Recomendación de productos. Análisis de segmentación de clientes.

Algoritmo Apriori – introducción … cont.

- **Salud**: Descubrimiento de Asociaciones en Registros Médicos. Análisis de farmacovigilancias.
- **Finanzas**: Detección de fraude. Análisis de riesgo de crédito. Gestión de portafolios.
- **Manufactura**: Análisis de procesos de producción. Análisis de control de calidad.
- **Gobierno**: Análisis de seguridad pública. Análisis de evasión fiscal. Análisis de fraude electoral.
- **Telecomunicaciones**: Análisis de usado de red.
- **Telemarketing**

Algoritmo Apriori - definición

- El principio de **Apriori** se basa en una propiedad fundamental: **Si un conjunto de elementos (itemsets) es frecuente, entonces todos sus subconjuntos también lo serán**.
- Formalmente:
  - **$\forall X \subseteq I$, si X es frecuente, entonces $\forall Y \subseteq X$, Y es frecuente**
- El nombre del algoritmo se basa en el hecho de que el *algoritmo utiliza conocimiento **previo** de las propiedades frecuentes de los itemsets*.

Algoritmo Apriori – generación frecuente

- Apriori emplea un enfoque iterativo conocido como búsqueda por niveles, donde se utilizan k-itemsets para explorar (k + 1)-itemsets.
  - El conjunto frecuente de 1-itemsets es encontrado escaneando la base de datos para acumular el conteo de cada elemento y recopilando aquellos elementos que satisfacen el soporte mínimo.
  - El conjunto resultante se denota por L1. Luego L1 se usar para encontrar L2. El conjunto de 2-itemsets frecuentes, que se usa para encontrar L3, y así sucesivamente, hasta que no se pueda encontrar k-itemsets frecuentes.
  - Encontrar Lk frecuentes requiere escaneo completo de la base de datos.

Algoritmo Apriori - teorema

- Para mejorar la eficiencia de la generación nivelada de itemsets frecuentes, se utiliza una propiedad importante llamada **propiedad Apriori** para reducir el espacio de búsqueda.
- Teorema **a priori (principio apriori)**:
  - **Si un conjunto de ítems es frecuente, entonces todos los subconjuntos también deben ser frecuentes**.
- Propiedad **monotónica**
  - Suponga que I sea un itemset y $J= 2^I$ sea el conjunto de potencia de I.
  - Una medida f es monotónica: Si X fuera un subconjunto de Y, entonces f(X) no debe exceder f(Y).

Algoritmo Apriori – propiedad monotónica

- f es anti monotónica se

$$\forall X, Y \in J: (X \subseteq Y) \rightarrow f(X) \leq f(Y),$$

$$\forall X, Y \in J: (X \subseteq Y) \rightarrow f(Y) \leq f(X),$$

- Lo que significa que, se X fuera un subconjunto de Y, entonces, f(Y) no debe exceder f(X).
- Cualquier medida que tenga una propiedad anti monotónica puede ser incorporada directamente al algoritmo de minería de datos para podar efectivamente el espacio de investigación exponencial de conjuntos de ítems candidatos.

Algoritmo Apriori – candidatos y poda

- El algoritmo de generación de candidatos y poda:
  - **Generación de candidatos**: Esta operación genera nuevos conjuntos de candidatos de k ítems, basada en los conjuntos frecuentes de (k-a) ítems encontrados en la iteración anterior.
  - **Poda de candidatos**: Esta operación elimina algunos de los conjuntos candidatos de k ítems, usando a estrategia de poda basada en soporte.
- Procedimientos de generación de candidatos:
  - **Método de la Fuerza Bruta**: Analiza cada conjunto de k ítems como un potencial candidato y después aplica el paso de poda de

Algoritmo Apriori – candidatos y poda … cont.

- **Método de la Fuerza Bruta**: … candidato para remover cualquier candidato desnecesario.
- El número de conjuntos de ítems candidatos en el nivel k es igual a $\binom{d}{k}$, donde *d* es el número total de ítems.

- A pesar de la generación de candidatos sea bastante trivial, la poda de candidatos se torna extremadamente costosa debido al grande número de conjuntos de ítems que deben ser examinados.

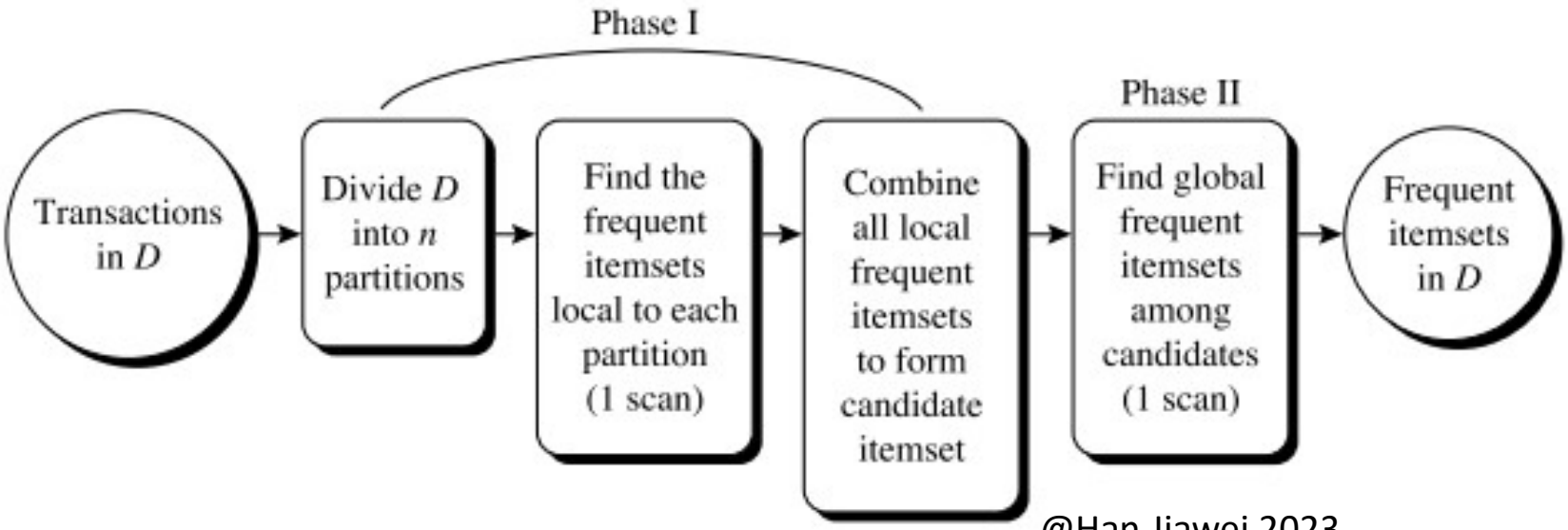
$$O\left(\int_{k=1}^{d} k \times \binom{d}{k}\right) = O(d.2^{d-1})$$

Algoritmo Apriori – mejorando a eficiencia

- **Técnica basada en HASH** (hash de itemsets en las canastas correspondientes). Se puede utilizar una técnica basada en **hash** para reducir el tamaño de los k-itemsets candidatos, C_k, para $k > 1$.
- **Reducción de transacciones** (reduciendo la cantidad de transacciones escaneadas en futuras iteraciones). Una transacción que no contiene ningún k-itemsets frecuentes no puede contener ningún $(k + 1)$-itemsets frecuentes. Por lo tanto, dicha transacción se puede marcar o eliminar de una mayor consideración porque los escaneos posteriores de la base de datos en busca de j-itemsets, donde $j > k$, no necesitarán considerar dicha transacción.

Algoritmo Apriori – mejorando a eficiencia

- **Partición** (particionar los datos para encontrar itemsets candidatos). Se puede utilizar una técnica de **partición,** eso requiere solo dos escaneos de la base de datos para extraer los itemsets frecuentes.



@Han,Jiawei 2023

Algoritmo Apriori – mejorando a eficiencia

- **Muestreo** (minería de un subconjunto de los datos dados). La idea básica del enfoque de muestreo es elegir una muestra aleatoria S de los datos dados D y luego buscar itemsets frecuentes en S en lugar de D. De esta manera, compensamos cierto grado de precisión con eficiencia.
- **Conteo dinámico de itemsets** (agregar itemsets candidatos en diferentes puntos durante un escaneo). Se propone una técnica dinámica de conteo de itemsets en la que la base de datos se divide en bloques marcados por puntos de inicio. En esta variación, se pueden agregar nuevos itemsets candidatos en cualquier punto inicial, a diferencia de **Apriori**, que determina nuevos itemsets candidatos solo después de cada escaneo completo de la base de datos.

Algoritmo FP-Growth

- El algoritmo FP-Growth es una técnica frecuente de minería de datos que tiene como objetivo **descubrir patrones relevantes en grandes conjuntos de datos**.
- El algoritmo está basado en:
  - **Reglas de asociación**: una regla de asociación se compone de un **antecedente** y un **consecuente**, y ambos representan itemsets. Estas reglas ayudan a descubrir relaciones entre elementos de una base de datos.
  - **Soporte**: Indica la frecuencia con la que ocurre un conjunto de elementos en la base de datos.
  - **Confianza**: Mide la validez de la regla.

Algoritmo FP-Growth ... Cont.

- **Incremento (Lift)**: Representa el incremento en las ventas del consecuente dado el antecedente.

- El algoritmo **FP-Growth**:
  - Utiliza una estructura de árbol llamada **FP-Tree** para representar elementos frecuentes de forma condensada y eficiente.
  - Superó el **rendimiento** del algoritmo Apriori.
  - **Pasos**:
    - **Generación de conjuntos frecuentes**: busca itemsets significativos con soporte mínimo.
    - **Generación de reglas**: identifica reglas por encima de un umbral mínimo de confianza.

Algoritmo ECLAT

- El algoritmo ECLAT (Equivalence Class Clustering and bottom-up Lattice Traversal) es una técnica frecuente de minería de datos que se centra en descubrir patrones de asociación en grandes conjuntos de datos.
- **Funcionamiento básico**:
  - ECLAT está diseñado para resolver el problema del **análisis de la cesta de compra**. Su objetivo es comprender qué productos se compran juntos con frecuencia.
  - A diferencia de los algoritmos **Apriori** y **FP-Growth**, **ECLAT** maneja datos almacenados en formato orientado a columnas (o “formato vertical”). Esto lo hace más rápido ya que sólo escanea la base de datos una vez.

Algoritmo ECLAT ... Cont.

- Los pasos de ECLAT incluyen:
  - **Transformación de datos**: convertir datos de formato horizontal a vertical.
  - **Generación de conjuntos frecuentes**: identificar combinaciones frecuentes de elementos.
  - **Generación de reglas de asociación**: crear reglas que relacionen estos elementos.

Algoritmo Apriori - aplicaciones

- **Análisis de la cesta de la compra**: Identifique los productos que se compran juntos con frecuencia.
- **Descubrimiento de asociaciones**: encontrar relaciones entre diferentes entidades en una base de datos.
- **Detección de fraude**: Identifique patrones de transacciones fraudulentas.
- **Recomendación de producto**: recomiende productos que los clientes puedan estar interesados en comprar.

Algoritmo Apriori - Herramientas

- **SPMF**: Herramienta de minería de datos de código abierto que implementa el algoritmo Apriori.
- **RapidMiner**: Herramienta comercial de minería de datos que incluye el algoritmo Apriori.
- **Weka**: herramienta de código abierto para aprendizaje automático que implementa el algoritmo Apriori.
- **KNIME**: es una plataforma analítica de código abierto para ciencia de datos
- **Mlxtend**: biblioteca Python para ciencia de datos y asociación de reglas – algoritmos (a priori, fp-growth, ECLAT).

Algoritmo Apriori - ejemplos

- Ejemplos de conjuntos de datos utilizados por el algoritmo a priori:
  - **Conjunto de datos de cesta de la compra**: registros de compras de clientes en un supermercado.
  - **Conjunto de datos de transacciones con tarjeta de crédito**: Transacciones realizadas con tarjeta de crédito.
  - **Conjunto de datos de registros médicos**: registros de pacientes en un hospital.

Algoritmo Apriori - referencias

- Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proc. of the 20th int. conf. on very large data bases (pp. 487-499). VLDB Endowment.
- Han, J., Kamber, M., & Pei, J. (2011). Data mining: Concepts and techniques (3rd ed.). Morgan Kaufmann.