Universidad Nacional Autónoma de México

Facultad de Ciencias, 2024 - 1

Almacenes y Minería de Datos

Práctica 09. Reglas de Asociación.

PostgresandoesoSQLazos

Integrantes

Adrian Aguilera Moreno	421005200
Marco Antonio Rivera Silva	318183583
Sebastían Alejandro Gutiérrez Medina	318287021
Israel Hernández Dorantes	318206604
Alejandra Ortega García	420002495
Luis Enrique García Gomez	315063880

1. Contesta lo siguiente:

1. Explica con tus propias palabras los algoritmos apriori y Eclat.

   R. Su objetivo principal es descubrir patrones de asociación entre diferentes conjuntos de elementos en grandes conjuntos de datos. El nombre "Apriori" proviene del hecho de que utiliza el principio de apriori, que implica que si un conjunto de elementos es frecuente, entonces todos sus subconjuntos también lo son. Este algoritmo calcula el conjunto potencia de un conjunto de items para crear reglas de asociación a partir de los itemsets de mayor frecuencia encontrados.

   A diferencia del algoritmo Apriori, Eclat no genera conjuntos de elementos de manera explícita. En su lugar, utiliza una estructura de datos llamada "Eclat tree" para representar la información sobre los itemsets frecuentes. En el caso de las reglas de asociación, al igual que en el algoritmo apriori las toma de los itemsets de mayor frecuencia encontrados en los recorridos del árbol.

2. Si el espacio de todas las reglas de asociación es exponencial, es decir, $O(2^m)$. Donde m es el número de elementos. En caso de ser cierto, ¿A que se debe dicha complejidad?

   R. Esto implicaría que todas los itemsets posibles son de gran frecuencia y es por esto que debemos considerarlos. Sin embargo es poco probable que en una muestra todos los itemsets tengan la misma frecuencia y además que esta sea alta (¿Respecto a qué itemsets?).

3. ¿Que implica que tengas Lift$(A \rightarrow X) = 0$?

   R. Lo anterior implica que la confianza en la regla de asociación es nula. En otras palabras, no hay una relación significativa entre A y X, ya que la probabilidad de que X ocurra dado que A ha ocurrido es la misma que la probabilidad de que X ocurra en general (sin tener en cuenta A).

2. Dado el siguiente BD realiza lo siguiente:

TID	Lista de elementos.
100	I1,I2,I4
101	I2,I4,I5
102	I2,I3,I4
103	I1
104	I1,I2,I3
105	I2,I3,I5
106	I1,I3,I4
107	I2,I3,I5
108	I2,I3

1. Si el soporte mínimo es de 2. ¿Cuál es el porcentaje?

   Coteo de soporte σ = 2

   Total de transacciones = 9

$$\text{soporte minimo} = \frac{2}{9} = 22.22\%$$

2. Si la confianza requerida es del 70%, debes encontrar los conjuntos de elementos frecuentes utilizando Aprori. Luego genera las reglas fuertes de asociación utilizando el soporte mínimo y la confianza mínima.

   Resultados de utilizar apriori:

```
    lhs         rhs  support   confidence coverage  lift      count
[1] {}       => {I2} 0.7777778 0.7777778  1.0000000 1.0000000 7
[2] {I5}     => {I2} 0.3333333 1.0000000  0.3333333 1.2857143 3
[3] {I4}     => {I2} 0.3333333 0.7500000  0.4444444 0.9642857 3
[4] {I3}     => {I2} 0.5555556 0.8333333  0.6666667 1.0714286 5
[5] {I2}     => {I3} 0.5555556 0.7142857  0.7777778 1.0714286 5
[6] {I3, I5} => {I2} 0.2222222 1.0000000  0.2222222 1.2857143 2
```

Código utilizado :

```r
install.packages( pkgs: "arules")
library(arules)

# Crea un vector para cada columna
TID ← c(100, 101, 102, 103, 104, 105, 106, 107, 108)
Lista_de_elementos ← c("I1,I2,I4", "I2,I4,I5", "I2,I3,I4", "I1", "I1,I2,I3", "I2,I3,I5", "I1,I3,I4", "I2,I3,I5", "I2,I3")

# Crea un dataframe con los vectores
data ← data.frame(TID, Lista_de_elementos)

# Convierte la columna Lista_de_elementos en una lista de listas
data$Lista_de_elementos ← strsplit(as.character(data$Lista_de_elementos), split: ",")
data$Lista_de_elementos

# Crea una transacción con la biblioteca arules
transacciones ← as(data$Lista_de_elementos, Class: "transactions")
inspect(transacciones)

soporte_minimo ← 0.22
confianza_minima ← 0.7

# Aplica el algoritmo Apriori
reglas ← apriori(transacciones, parameter = list(supp = soporte_minimo, conf = confianza_minima))

# Muestra las reglas generadas
inspect(reglas)
```

3. Dado la siguiente tabla realiza lo siguiente:

Donde se puede observar la información de algunas transacciones en formato horizontal. Utilizando Equivalente Class Transformation (ECLAT).

TID	Lista de elementos.
T1	I4,I5
T2	I2,I3,I4,I5
T3	I1,I3,I5
T4	I3,I4
T5	I1,I4
T6	I4,I5
T7	I2,I3,I5
T8	I2,I5
T9	I3,I4,I5

Considerando que el soporte mínimo es de 20 %.
Primero cambiamos el formato a vertical:

TID	Lista_de_elementos
I1	T3, T5
I2	T2, T7, T8
I3	T2, T3, T4, T7, T9
I4	T1, T2, T4, T5, T9
I5	T1, T2, T3, T6, T7, T8, T9

- Proporciona una tabla con 3 columnas donde identifiques los items (itemSet con k=1) que aparecen en el conjunto de transacciones y calcula su soporte.

items	support	count
{T2}	0.8	4
{T9}	0.6	3
{T7}	0.6	3
{T3}	0.6	3
{T1}	0.4	2
{T8}	0.4	2
{T4}	0.4	2
{T5}	0.4	2
{T6}	0.2	1

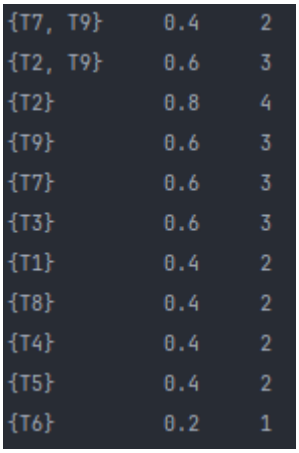
- Proporciona todas las posibles intersecciones de la columna Transacciones K=1 donde obtenemos los itemsets de longitud k+1.

items	support	count
{T2, T6}	0.2	1
{T6, T9}	0.2	1
{T6, T7}	0.2	1
{T3, T6}	0.2	1
{T1, T6}	0.2	1
{T6, T8}	0.2	1
{T2, T5}	0.2	1
{T5, T9}	0.2	1
{T3, T5}	0.2	1
{T1, T5}	0.2	1
{T4, T5}	0.2	1
{T2, T4}	0.4	2
{T4, T9}	0.4	2
{T4, T7}	0.2	1
{T3, T4}	0.2	1
{T1, T4}	0.2	1
{T2, T8}	0.4	2
{T8, T9}	0.2	1
{T7, T8}	0.4	2
{T3, T8}	0.2	1
{T1, T8}	0.2	1
{T1, T2}	0.4	2
{T1, T9}	0.4	2
{T1, T7}	0.2	1
{T1, T3}	0.2	1
{T2, T3}	0.4	2
{T3, T9}	0.4	2
{T3, T7}	0.4	2
{T2, T7}	0.6	3
{T7, T9}	0.4	2
{T2, T9}	0.6	3

- Proporciona las intersecciones de la tabla anterior (itemSet k=2) obteniendo los (itemset k=3). Hint: Piensa en el principio de downward closure te puede ahorrar trabajo.

items	support	count
{T2, T6, T8}	0.2	1
{T6, T8, T9}	0.2	1
{T6, T7, T8}	0.2	1
{T3, T6, T8}	0.2	1
{T1, T6, T8}	0.2	1
{T1, T2, T6}	0.2	1
{T1, T6, T9}	0.2	1
{T1, T6, T7}	0.2	1
{T1, T3, T6}	0.2	1
{T2, T3, T6}	0.2	1
{T3, T6, T9}	0.2	1
{T3, T6, T7}	0.2	1
{T2, T6, T7}	0.2	1
{T6, T7, T9}	0.2	1
{T2, T6, T9}	0.2	1
{T2, T6}	0.2	1
{T6, T9}	0.2	1
{T6, T7}	0.2	1
{T3, T6}	0.2	1
{T1, T6}	0.2	1
{T6, T8}	0.2	1
{T2, T4, T5}	0.2	1
{T4, T5, T9}	0.2	1
{T1, T4, T5}	0.2	1
{T1, T2, T5}	0.2	1
{T1, T5, T9}	0.2	1
{T2, T5, T9}	0.2	1
{T2, T5}	0.2	1
{T5, T9}	0.2	1
{T3, T5}	0.2	1
{T1, T5}	0.2	1
{T4, T5}	0.2	1
{T1, T2, T4}	0.2	1
{T1, T4, T9}	0.2	1
{T2, T3, T4}	0.2	1
{T3, T4, T9}	0.2	1
{T3, T4, T7}	0.2	1
{T2, T4, T7}	0.2	1

{T4, T7, T9}	0.2	1
{T2, T4, T9}	0.4	2
{T2, T4}	0.4	2
{T4, T9}	0.4	2
{T4, T7}	0.2	1
{T3, T4}	0.2	1
{T1, T4}	0.2	1
{T1, T2, T8}	0.2	1
{T1, T8, T9}	0.2	1
{T1, T7, T8}	0.2	1
{T1, T3, T8}	0.2	1
{T2, T3, T8}	0.2	1
{T3, T8, T9}	0.2	1
{T3, T7, T8}	0.2	1
{T2, T7, T8}	0.4	2
{T7, T8, T9}	0.2	1
{T2, T8, T9}	0.2	1
{T2, T8}	0.4	2
{T8, T9}	0.2	1
{T7, T8}	0.4	2
{T3, T8}	0.2	1
{T1, T8}	0.2	1
{T1, T2, T3}	0.2	1
{T1, T3, T9}	0.2	1
{T1, T3, T7}	0.2	1
{T1, T2, T7}	0.2	1
{T1, T7, T9}	0.2	1
{T1, T2, T9}	0.4	2
{T1, T2}	0.4	2
{T1, T9}	0.4	2
{T1, T7}	0.2	1
{T1, T3}	0.2	1
{T2, T3, T7}	0.4	2
{T3, T7, T9}	0.4	2
{T2, T3, T9}	0.4	2
{T2, T3}	0.4	2
{T3, T9}	0.4	2
{T3, T7}	0.4	2
{T2, T7, T9}	0.4	2
{T2, T7}	0.6	3



{T7, T9}	0.4	2
{T2, T9}	0.6	3
{T2}	0.8	4
{T9}	0.6	3
{T7}	0.6	3
{T3}	0.6	3
{T1}	0.4	2
{T8}	0.4	2
{T4}	0.4	2
{T5}	0.4	2
{T6}	0.2	1

- ¿En qué punto finaliza el algoritmo de ECLAT?

  El algoritmo Eclat finaliza cuando ya no es posible encontrar más conjuntos frecuentes que cumplan con los criterios de soporte mínimo. El proceso de encontrar conjuntos frecuentes en Eclat se realiza mediante una búsqueda recursiva, donde cada nodo en el árbol representa un elemento, y las ramas del nodo representan transacciones que contienen ese elemento. Al explorar y recorrer este árbol, el algoritmo identifica los conjuntos de elementos que cumplen con el umbral de frecuencia dado y continúa hasta que no hay más nodos o ramas que cumplan con los criterios de soporte mínimo.

- El algoritmo permite una identificación de itemSets frecuentes, pero no produce reglas de asociación. ¿Qué propones para concluir nuestra tarea?

  Podemos usar una de las subrutinas del algoritmo apriori para encontrar las reglas de asociación, en partícular podemos modificar el algoritmo 2 (en este documento) para que trabaje con el conjunto de itemsets generados por nuestro paso anterior (ECLAT).

- Menciona las principales diferencias entre el algoritmo Apriori y ECLAT.

  Las principales diferencias entre el algoritmo Apriori y ECLAT son:

  1. **Uso de conjuntos de datos**: Apriori se utiliza con conjuntos de datos grandes, mientras que ECLAT es más adecuado para conjuntos de datos pequeños y medianos.
  2. **Exploración de datos**: Apriori explora el conjunto de datos original, mientras que ECLAT explora el conjunto de datos que se genera actualmente.
  3. **Velocidad**: Apriori es más lento que ECLAT.
  4. **Enfoque de búsqueda**: Mientras que el algoritmo Apriori trabaja en un sentido horizontal imitando la Búsqueda en Anchura de un grafo, el algoritmo ECLAT trabaja de manera vertical, similar a la Búsqueda en Profundidad de un grafo.
  5. **Requerimientos de memoria**: Dado que el algoritmo ECLAT utiliza un enfoque de Búsqueda en Profundidad, utiliza menos memoria que el algoritmo Apriori.

6. **Número de cálculos**: El algoritmo ECLAT no implica el escaneo repetido de los datos para calcular los valores de soporte individuales.