Práctica 8 - Reglas de asociación

Almacenes y Minería de Datos
Grupo 7094

Cruz Ramírez Nicolás
García Beltrán Axel David

15 de mayo de 2022

1. Investiga y responde correctamente los siguientes comandos o conceptos:

   - Imagina que estás con un(a) familiar que quiere entrar a la facultad de Derecho y para irte de party loca te piden lo siguiente: explica con TUS PROPIAS PALABRAS los algoritmos Apriori y Eclat.

     Respuesta. El algoritmo Apriori es un algoritmo que nos ayuda a obtener los conjuntos de ítems frecuentes en una base de datos de transacciones (por ejemplo, compras en un supermercado). Primero definimos una frecuencia o soporte mínimo, digamos un 20%, entonces nos van a interesar todos los productos y conjuntos de productos que aparezcan más de 20% en la BD. Después se comienza a trabajar con una tabla en formato horizontal, en cada renglón se tendrá el conjunto de ítems y su frecuencia. Comenzamos con conjuntos de ítems de tamaño 1 y calculamos su frecuencia haciendo un barrido sobre la BD, luego, si su frecuencia es mayor al soporte dado (20% es nuestro ejemplo) los conservamos, si no, los descartaremos. Cuando descartamos los conjuntos con soporte menor al 20%, usamos el principio downward closure, el que nos dice que si un conjunto de ítems no es frecuente, al "meterlo" en otro conjunto de ítems más grande, este tampoco será frecuente, esto tiene bastante sentido si ponemos un ejemplo, tenemos el producto {Lechuga} que sólo aparece una vez en las transacciones, es obvio que el conjunto de productos {Lechuga,Mostaza} puede aparecer a los más una sola vez, ya que está limitado cuantas veces aparece la {Lechuga} sola. Usando esta idea, en la siguiente iteración trabajaremos con los conjuntos que no hayamos descartado en el paso anterior, y con ellos crearemos conjuntos de ítems de tamaño 2, calcularemos su soporte haciendo un barrido a la base, y descartamos aquellos que tengan soporte (frecuencia) menor a nuestro 20%. Nuestro algoritmo termina cuando ya no encontremos conjuntos de ítems con frecuencia mayor al 20%.
     El algoritmo ECLAT también nos ayuda a obtener conjuntos de ítems frecuentes, pero con ya no haremos un barrido a la BD en cada iteración, ya que este se basa en intersección de conjuntos. Primero comenzamos a trabajar con una tabla con una columna con el ítemset, otra columna con el Id de las transacciones donde aparece el ítemset y otra columna con su frecuencia. Para la iteración 1 ponemos los ítemsets de tamaño 1, ponemos en qué Id transacción aparecen y para calcular su frecuencia contamos cuantos valores distintos aparecen en la columna de

Id Transacciones. Hecho esto, usando el principio downward closure descartaremos todos los ítemsets con frecuencia menor al 20%. Para la siguiente iteración, nos fijamos en los ítemsets que no descartamos en el paso anterior, y vamos a tomar las intersecciones sobre la columna Id Transacciones para formar nuevos ítemsets de tamaño 2, estos ítemset los ponemos en la primera columna, en la segunda columna ponemos el resultado de la intersección de Id Transacciones, y finalmente calculamos la frecuencia, donde solo basta con contar cuantos conjuntos distintos tenemos en columna de Id transacciones, en vez de hacer un barrido sobre toda la BD como en el algoritmo Apriori. Nuestro algoritmo termina cuando ya no encontremos conjuntos de ítems con frecuencia mayor al 20%.

Finalmente, con estos 2 algoritmos obtuvimos conjuntos de ítems frecuentes, pero nos falta obtener las reglas de asociación. Para esto debemos tomar todos los ítemsets frecuentes encontrados, y para cada uno vamos a partirlo en 2 de todas las formas posibles (Por ejemplo el ítemset {A,B} lo partimos en {A} y {B}) y cada forma posible nos dará 2 reglas al permutar el antecedente y consecuente en cada regla (por ejemplo obtenemos las reglas $\{A\} \Rightarrow \{B\}$ y la regla $\{B\} \Rightarrow \{A\}$).

■

- En tu clase de Complejidad la morra de los plumones dice que el espacio de todas las reglas de asociación es exponencial, es decir, O(m). Donde m es el número de elementos. En caso de ser cierto, ¿a qué se debe dicha complejidad?

  Respuesta. Es correcto, ya que para obtener reglas de asociación debemos hacer particiones binarias para cada unos de los conjuntos de ítemsets frecuentes, por lo cual para cada uno de los k-ítemsets se pueden obtener $2^k - 2$ posibles reglas, por lo cual si tuviéramos un ítemset de tamaño 100 podríamos obtener en total $2^{100} - 2$ posibles reglas. Lo cual tiene un crecimiento exponencial.

  ■

- ¿Qué implica que tengas $lift(A \to B) = 0$?

  Respuesta. Por definición de lift se tiene que:

$$lift(A \to B) = \frac{Soporte(A \cup B)}{Soporte(A) * Soporte(B)}$$

Por lo cual:

$$0 = \frac{Soporte(A \cup B)}{Soporte(A) * Soporte(B)}$$

Es decir que:

$$0 = Soporte(A \cup B)$$

Por definición de soporte, nos indica que el conjunto $A \cup B$ no aparece en ninguna transacción de la BD. Por lo tanto, si tenemos que $lift(A \rightarrow B) = 0$ implica que esta regla se obtuvo a partir de un ítemset $A \cup B$ que no aparece en la BD de transacciones.

■

2. Supón que tienes una BD como la imagen de abajo, dicha base consiste de 7 transacciones a realizar.

   - Demos un apoyo mínimo de 2. ¿Cuál es el porcentaje?

     Respuesta. El porcentaje sería $\frac{2}{7}(100) = 28.5714\%$; es decir, los itemsets que consideraremos frecuentes serán los que aparezcan en el 28.57 por ciento del total de transacciones.

     ■

   - Si la confianza requerida es del 70% debes encontrar los conjuntos de elementos frecuentes utilizando Apriori. Luego genera las reglas fuertes de asociación utilizando el apoyo mínimo y la confianza mínima.

     Respuesta. Siguiendo el algoritmo a priori, primero debemos obtener C_1, el conjunto de los 1-itemsets, y calcular su frecuencia. Tenemos primero a $I = \{I1, I2, I3, I4\}$ como el conjunto de todos los items de la base de datos. Luego, Vemos que todos los 1-itemsets son frecuentes; es decir, $F_1 = C_1$.

     Después, a partir de F_1, se forman los 2-itemsets posibles, y se determina su frecuencia. Entonces:

     De estos itemsets, vemos que {I1, I3} no es frecuente, por lo que lo eliminamos:

C_1 :

1-itemset	Frec
I1	3
I2	6
I3	3
I4	4

C_2 :

2-itemset	Frec
{I1, I2}	3
{I1, I3}	**1**
{I1, I4}	2
{I2, I3}	3
{I2, I4}	4
{I3, I4}	3

Luego, a partir de F_2, generamos C_3:

Donde ambos itemsets son frecuentes, por lo que $F_3 = C_3$. Finalmente, como la base de datos no tiene 4-itemsets o superiores, termina el algoritmo. Así, obtenemos todos los itemsets frecuentes en la tabla $\bigcup_{k=1}^{3} F_k$.

Ahora, después de obtener todos los itemsets frecuentes y sus frecuencias, debemos calcular las reglas de asociación fuertes; es decir, las reglas que satisfagan tener soporte y confianza mayores a los mínimos establecidos. Como estos itemsets ya son frecuentes, solo hay que calcular la confianza de las reglas posibles Siguiendo el algoritmopara generar reglas de asociación fuertes, obtenemos:

Donde las reglas en negritas son las que no cumplen con la confianza mínima. Así, eliminándolas, nos quedamos con el conjunto de reglas fuertes:

F_2 :

2-itemset	Frec
{I1, I2}	3
{I1, I4}	2
{I2, I3}	3
{I2, I4}	4
{I3, I4}	3

C_3 :

3-itemset	Frec
{I1, I2, I4}	2
{I2, I3, I4}	2

■

- Con lo realizado te diste cuenta de algunos problemas al emplear Aprior, algunas optimizaciones pudieran ser:
  (a) Conteo de elementos usando tablas Hash.
  (b) Reducción en el número de transacciones.
  (c) Particionamiento.
  (d) Muestreo.

  Explica como adoptas al menos 2 de optimizaciones de los incisos anteriores del ejercicio.

  Respuesta. El algoritmo apriori se puede optimizar con reducción en el número de transacciones. Por ejemplo, al pasar de crear F_2 a C_3, lo que hicimos fue crear los itemsets a partir de las distintas combinaciones de lso 4 items. Sin embargo, como por ejemplo, el itemset {I1, I3} no aparece en F_2, es innecesario incluir cualquier combinación de 3 items que tenga a este como subconjunto; como este no es frecuente, sabemos que todos sus superconjuntos tendrán, a lo más, su frecuencia, por lo que tampoco serán frecuentes. Así, es innecesario seguir iterando sobre estas transacciones en pasos siguientes, por lo que pudimos eliminarlas en cuanto nos dimos cuenta de que ya no contienen más itemsets frecuentes, evitando así tener que recorrer toda la base de datos en cada iteración del algoritmo.

  Otra estrategia que podemos usar es la de particionamiento. En ella, buscaremos itemsets frecuentes en particiones del conjunto de transacciones, subiendo en

$\bigcup_{k=1}^{3} F_k$:

1-itemset	Frec	2-itemset	Frec	3-itemset	Frec
I1	3	{I1, I2}	3	{I1, I2, I4}	2
I2	6	{I1, I4}	2	{I2, I3, I4}	2
I3	3	{I2, I3}	3		
I4	4	{I2, I4}	4		
		{I3, I4}	3		

Regla	Confianza	Regla	Confianza
I1 ⇒ I2	3/3 = 100%	**I1 ⇒ {I2, I4}**	**2/3 = 66.66%**
I2 ⇒ I1	**3/6 = 50%**	**I2 ⇒ {I1, I4}**	**2/6 = 33.33%**
I1 ⇒ I4	**2/3 = 66.66%**	**I4 ⇒ {I1, I2}**	**2/4 = 50%**
I4 ⇒ I1	**2/4 = 50%**	**{I2, I4} ⇒ I1**	**2/4 = 50%**
I2 ⇒ I3	**3/6 = 50%**	**{I1, I4} ⇒ I2**	**2/6 = 66.66%**
I3 ⇒ I2	3/3 = 100%	{I1, I2} ⇒ I4	2/2 = 100%
I2 ⇒ I4	**4/6 = 66.66%**	**I2 ⇒ {I3, I4}**	**2/6 = 33.33%**
I4 ⇒ I2	4/4 = 100%	**I3 ⇒ {I2, I4}**	**2/3 = 66.66%**
I3 ⇒ I4	3/3 = 100%	**I4 ⇒ {I2, I3}**	**2/4 = 50%**
I4 ⇒ I3	3/4 = 75%	**{I3, I4} ⇒ I2**	**2/3 = 66.66%**
		{I2, I4} ⇒ I3	**2/4 = 50%**
		{I2, I3} ⇒ I4	**2/3 = 66.66%**

Regla	Confianza
I1 ⇒ I2	3/3 = 100%
I4 ⇒ I2	4/4 = 100%
I3 ⇒ I2	3/3 = 100%
I3 ⇒ I4	3/3 = 100%
I4 ⇒ I3	3/4 = 75%
{I1, I2} ⇒ I4	2/2 = 100%

tamaño hasta llegar al conjunto total. Con esta optimización, nos deshacemos de itemsets que no podrían ser frecuentes sin la necesidad de iterar sobre toda la base de datos en cada paso; como no son frecuentes en un trozo del conjunto total, es imposible que lo sean en todo el conjunto. Sin embargo, no por el hecho de que haya frecuencia en una partición de la base de datos implica que

ese itemset particular será frecuente en el total; simplemente se descartan los que definitivamente no lo son.

■

3. Corría el año de 2000 cuando Zaki propone un nuevo algoritmo para encontrar patrones frecuentes llamado Equivalente Class Transformation o ECLAT:

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

     Respuesta. Una de las principales diferencias es que el algoritmo Apriori hace un barrido de la base de datos para calcular el soporte de cada nuevo ítemset que se vaya obteniendo en cada iteración. En cambio al usar ECLAT, como para cada ítemset ya se tiene el conjunto de Id-Transacciones donde éste aparece, únicamente basta con realizar las intersecciones y calcular la cardinalidad de estos subconjuntos, luego al dividir la cardinalidad del conjunto intersección entre total de transacciones obtenemos el soporte, lo cual es menos costoso computacionalmente que hacer otro barrido a la BD.
     Otra de las principales diferencias es que en el algoritmo ECLAT desde un inicio se requiere conocer los Id-Transacciones donde aparece cada ítemset, mientras que el el algoritmo Apriori no se utiliza. Otra diferencia es que el Apriori se trabaja con los datos en una forma llamada horizontal, donde cada fila representa una transacción y contienen los ítems que la conforman, mientras que en ECLAT se trabaja en forma vertical donde cada fila representa un ítemset y se tiene las transacciones donde aparece.

     ■

- En la tabla de la izquierda podemos observar la información de algunas transacciones en formato horizontal. Consideremos el soporte mínimo para un itemset del 20%.

:

Transacción	Ítems
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

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

Respuesta. Iniciamos generando la tabla con los ítemset de k = 1.
De lado derecho le ponemos un ✓ a aquellos que tenga soporte ≥ 20

$k = 1$:

Ítemset	Transacción	Soporte	
{I1}	T3, T5	$2/9 = 0.22$	✓
{I2}	T2, T7, T8	$3/9 = 0.33$	✓
{I3}	T2, T3, T4, T7, T9	$5/9 = 0.61$	✓
{I4}	T1, T2, T4, T5, T6, T9	$6/9 = 0.66$	✓
{I5}	T1, T2, T3, T6, T7, T8, T9	$7/9 = 0.77$	✓

■

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

Respuesta. Generamos la tabla con los ítemset de tamaño k = 2.
De lado derecho le ponemos un ✓ a aquellos que tenga soporte ≥ 20

■

	Ítemset	Transacción	Soporte	
	{I1,I2}	$\emptyset$	$0/9 = 0$	
	{I1,I3}	T3	$1/9 = 0.11$	
	{I1,I4}	T5	$1/9 = 0.11$	
	{I1,I5}	T3	$1/9 = 0.11$	
$k = 2:$	{I2,I3}	T2, T7	$2/9 = 0.22$	✓
	{I2,I4}	T2	$1/9 = 0.11$	
	{I2,I5}	T2, T7, T8	$3/9 = 0.33$	✓
	{I3,I4}	T2, T4, T9	$3/9 = 0.33$	✓
	{I3,I5}	T2, T3, T7, T9	$4/9 = 0.44$	✓
	{I4,I5}	T1, T2, T6, T9	$4/9 = 0.44$	✓

- 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.)*

  Respuesta. Generamos la tabla con los ítemset de tamaño k = 3 haciendo las intersecciones en la tabla k = 2.
  Por el principio downward closure ya sólo usamos los conjuntos de ítemsets con ✓ es decir con soporte ≥ 20 en la iteración anterior.
  Y en nuestra nueva tabla le ponemos un ✓ a aquellos que tengan soporte ≥ 20

	Ítemset	Transacción	Soporte	
	{I2,I3,I5}	T2, T7	$2/9 = 0.22$	✓
$k = 3:$	{I2,I3,I4}	T2	$1/9 = 0.11$	
	{I2,I4,I5}	T2	$1/9 = 0.11$	
	{I3,I4,I5}	T2, T9	$2/9 = 0.22$	✓

■

- ¿En qué punto finaliza el algoritmo de ECLAT?

  Respuesta. ECLAT finaliza cuando al realizar las intersecciones sobre los conjuntos de transacciones para crear ítemsets de tamaño k+1 ya no se obtengamos ítemsets con soporte mayor al umbral definido previamente (en este caso 20%). En nuestro caso al hacer las intersecciones para obtener los ítemsets de tamaño

k=4 se tiene la siguiente tabla, la cual ya no contiene ningún conjunto de ítemsets frecuentes. Por lo que el algoritmo ECLAT termina.

$k = 4$:

Ítemset	Transacción	Soporte	
{I2,I3,I4,I5}	T2	$1/9 = 0.11$	

Ahora bien, podemos mostrar a manera de resumen todos los conjuntos de ítemsets con soporte ≥ 20 que encontramos.

:

1-itemset	Frec	2-itemset	Frec	3-itemset	Frec
{I1}	2	{I2, I3}	2	{I2, I3, I5}	2
{I2}	3	{I2, I5}	3	{I3, I4, I5}	2
{I3}	5	{I3, I4}	3		
{I4}	6	{I3, I5}	4		
{I5}	7	{I4, I5}	4		

■

- Dada la nota* no cerrar el proceso, ¿qué propones para concluir nuestra tarea?

Respuesta. Una vez que finalizó el algoritmo y obtuvimos el conjunto de ítemset frecuentes, podemos obtener el conjunto de reglas de asociación y calcular su confianza. Para obtener las reglas debemos hacer una partición binaria para cada ítemset frecuente y permutar el antecedente y consecuente en cada regla. Posteriormente calculamos la confianza y dado umbral de confianza podemos conservar sólo aquellas reglas cuya confianza supere el umbral determinado.

Por ejemplo, si se pide una confianza del 70% conservaríamos únicamente las reglas con confianza mayor a ese umbral, las marcamos con ✓.

■

Regla	Confianza		Regla	Confianza	
I2 ⇒ I3	2/3 = 0.66		I2 ⇒ {I3, I5}	2/3 = 0.66	
I3 ⇒ I2	2/5 = 0.40		I3 ⇒ {I5, I2}	2/5 = 0.40	
I2 ⇒ I5	3/3 = 1.00	✓	I5 ⇒ {I2, I3}	2/7 = 0.28	
I5 ⇒ I2	3/7 = 0.42		{I3, I5} ⇒ I2	2/4 = 0.50	
I3 ⇒ I4	3/5 = 0.60		{I5, I2} ⇒ I3	2/3 = 0.66	
: I4 ⇒ I5	3/6 = 0.50		{I2, I3} ⇒ I5	2/2 = 1.00	✓
I3 ⇒ I5	4/5 = 0.80	✓	I3 ⇒ {I4, I5}	2/5 = 0.40	
I5 ⇒ I3	4/7 = 0.57		I4 ⇒ {I5, I3}	2/6 = 0.33	
I4 ⇒ I5	4/6 = 0.66		I5 ⇒ {I3, I4}	2/7 = 0.28	
I5 ⇒ I4	4/7 = 0.57		{I4, I5} ⇒ I3	2/4 = 0.50	
			{I5, I3} ⇒ I4	2/4 = 0.50	
			{I3, I4} ⇒ I5	2/3 = 0.66	

4. Analiza el script `arules.r` y comenta el comportamiento debajo de cada línea que tiene un símbolo de comentario, una vez que hayas logrado su ejecución agrega evidencia de las ejecuciones o gráficas resultantes.

Respuesta. La primer gráfica del script muestra un histograma relacionando las transacciones con su tamaño, ordenadas ascendentemente por tamaño. Notamos que hay muchas (más de 2000) transacciones de un solo item, y el número va bajando conforme sube el tamaño, hasta un máximo de tamaño de 32.

La segunda gráfica definida en el script se efectúa ya realizado el algoritmo apriori sobre los datos. La gráfica muestra el top 20 de los itemsets que encontró apriori, ordenados descendentemente por soporte.

■

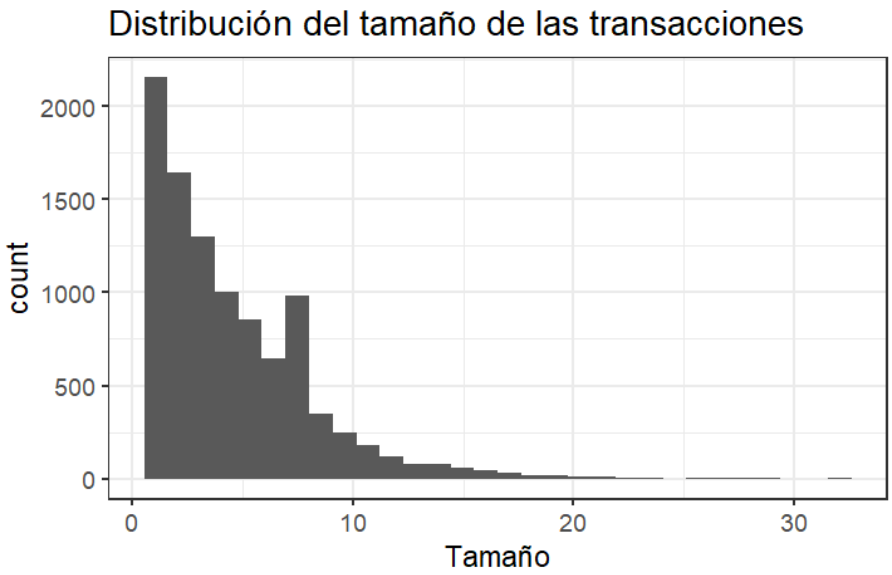

Figure 1: Distribución del tamaño de las transacciones leídas.

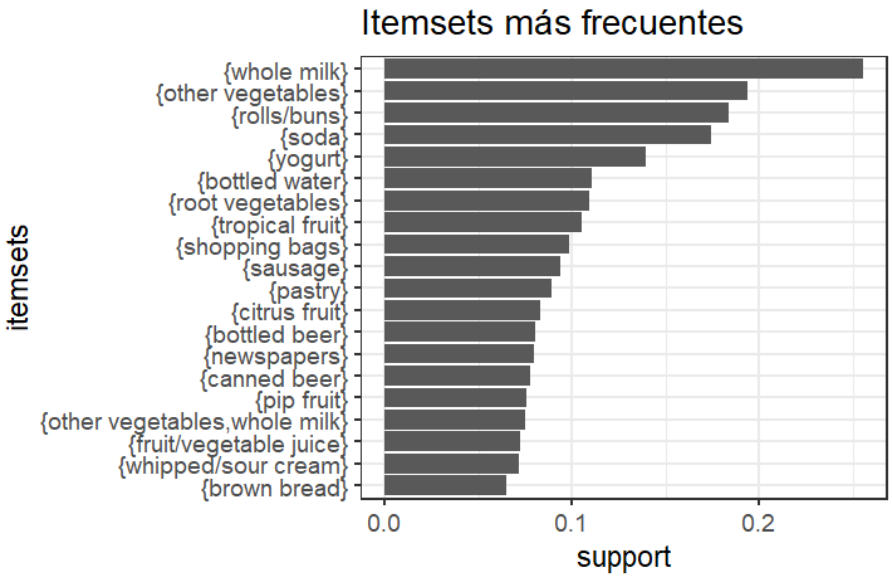

Figure 2: Los primeros 20 itemsets más frecuentes.