

PECL2 - Fundamentos de la ciencia de datos

Mario Adán Herrero Alberto González Martínez
Branimir Stefanov Yanev Diego Gutiérrez Marco

12 de noviembre de 2020

Resumen

En el siguiente documento se presentan los resultados y solución de la PECL2 del laboratorio de Fundamentos de la Ciencia de Datos. Se realizará un análisis de asociación de datos siguiendo el ejercicio realizado en teoría. Utilizaremos el algoritmo Apriori y obtendremos aquellas asociaciones cuyo soporte sea mayor o igual que 50 % y cuya confianza sea mayor o igual que 80 %. Se mostrarán los resultados en este pdf utilizando las herramientas Sweave y TinyTex/MikTeX.

Índice

1. Introducción	3
2. Ejercicio 1- Análisis de asociación. Cesta de la compra	3
2.1. Elementos	3
2.2. Sucesos	3
2.3. Paquete Arules	4
2.4. Matriz de asociaciones	4
2.5. Análisis de asociación	5
3. Ejercicio 2- Análisis de asociación. Componentes	6
3.1. Productos	6
3.2. Elementos	6
3.3. Paquete Arules	6
3.4. Lectura de datos	6
3.5. Analisis de asociacion	7
4. Conclusiones	8

1. Introducción

La práctica consta de dos partes:

En la primera parte, se va a realizar un análisis de asociación, utilizando R, aplicando conceptos vistos en la teoría y haciendo uso del algoritmo Apriori, se resolverá el mismo problema visto en teoría.

En la segunda parte, el grupo desarrollará un enunciado, y su posterior solución, de un ejercicio que contenga modificaciones del ejercicio hecho en clase, en el que se realice un análisis de asociación con R.

2. Ejercicio 1- Análisis de asociación. Cesta de la compra

Para resolver este ejercicio haremos uso del paquete “arules”. Este paquete de R nos proporciona las funciones necesarias para realizar nuestro análisis de asociación.

En este caso en particular analizaremos las compras de varios usuarios para concluir qué productos son comprados juntos habitualmente.

El objetivo es obtener las asociaciones cuyo soporte sea igual o superior al 50 % y cuya confianza sea igual o superior al 80 %.

2.1. Elementos

Los distintos productos que podemos comprar son los siguientes.

1. Pan
2. Agua
3. Café
4. Leche
5. Naranjas

2.2. Sucesos

Los usuarios han realizado las siguientes compras.

1. Pan, Agua, Leche, Naranjas
2. Pan, Agua, Café, Leche
3. Pan, Agua, Leche
4. Pan, Café, Leche
5. Pan, Agua
6. Leche

2.3. Paquete Arules

Primero, debemos cargar la librería *arules* que contiene las funciones necesarias para nuestro análisis de asociación con el algoritmo Apriori.

```
> library("arules")
```

2.4. Matriz de asociaciones

Tras esto, cargamos nuestra matriz de asociaciones, que contiene la información de las compras realizadas.

```
> muestra<- Matrix(  
+   c(1,1,0,1,1,1,1,1,1,0,1,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0),  
+   6, 5, byrow=T, dimnames=list(  
+     c("compra1","compra2","compra3","compra4","compra5","compra6"),  
+     c("Pan","Agua","Cafe","Leche","Naranjas")),  
+   sparse=T)  
> #Imprimimos la matriz  
> muestra
```

```
6 x 5 sparse Matrix of class "dgCMatrix"  
      Pan Agua Cafe Leche Naranjas  
compra1  1   1   .   1       1  
compra2  1   1   1   1       .  
compra3  1   1   .   1       .  
compra4  1   .   1   1       .  
compra5  1   1   .   .       .  
compra6  .   .   .   1       .
```

Para emplear el algoritmo Apriori, primero convertimos la matriz a un objeto de transacciones a través de una matriz dispersa.

```
> #Matriz dispersa  
> muestrangC <- as(muestra,"nsparseMatrix")  
> muestrangC
```

```
6 x 5 sparse Matrix of class "ngCMatrix"  
      Pan Agua Cafe Leche Naranjas  
compra1 |   |   .   |       |  
compra2 |   |   |   |       .  
compra3 |   |   .   |       .  
compra4 |   .   |   |       .  
compra5 |   |   .   .       .  
compra6 .   .   .   |       .
```

```
> #Matriz transpuesta  
> muestrangC_transpuesta <- t(muestrangC)  
> muestrangC_transpuesta
```

```

5 x 6 sparse Matrix of class "ngCMatrix"
      compra1 compra2 compra3 compra4 compra5 compra6
Pan      |      |      |      |      |      .
Agua     |      |      |      .      |      .
Cafe     .      |      .      |      .      .
Leche    |      |      |      |      .      |
Naranjas |      .      .      .      .      .

> #Transacciones
> transacciones <- as(muestrangC_transpuesta , "transactions")
> transacciones

transactions in sparse format with
  6 transactions (rows) and
  5 items (columns)

```

2.5. Análisis de asociación

Aplicamos ahora el algoritmo Apriori con soporte mayor o igual al 50% y confianza mayor o igual al 80%, y mostramos el resultado por pantalla para ver qué asociaciones cumplen este criterio.

```
> asociaciones <- apriori(transacciones, parameter=list(support=0.5, confidence=0.8))
```

Apriori

Parameter specification:

```

confidence minval smax arem aval originalSupport maxtime support minlen
      0.8    0.1    1 none FALSE              TRUE      5    0.5    1
maxlen target  ext
      10  rules TRUE

```

Algorithmic control:

```

filter tree heap memopt load sort verbose
  0.1 TRUE TRUE  FALSE TRUE    2    TRUE

```

Absolute minimum support count: 3

```

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[5 item(s), 6 transaction(s)] done [0.00s].
sorting and recoding items ... [3 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [7 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].

```

```
> inspect(asociaciones)
```

```

      lhs      rhs      support      confidence coverage lift count
[1] {}      => {Leche} 0.8333333 0.8333333 1.0000000 1.00 5
[2] {}      => {Pan}   0.8333333 0.8333333 1.0000000 1.00 5

```

```
[3] {Agua}      => {Pan}      0.6666667 1.0000000 0.6666667 1.20 4
[4] {Pan}       => {Agua}      0.6666667 0.8000000 0.8333333 1.20 4
[5] {Leche}     => {Pan}       0.6666667 0.8000000 0.8333333 0.96 4
[6] {Pan}       => {Leche}     0.6666667 0.8000000 0.8333333 0.96 4
[7] {Agua,Leche}=> {Pan}      0.5000000 1.0000000 0.5000000 1.20 3
```

3. Ejercicio 2- Análisis de asociación. Componentes

Para el segundo ejercicio, procederemos de igual manera que anteriormente, pero aplicando los cambios necesarios para la modificación del ejercicio.

En este caso, nos pondremos en el lugar de una tienda de ordenadores por lo que se desea estudiar las asociaciones entre las ventas de sus productos para poder elaborar paquetes de oferta.

3.1. Productos

Los productos son los siguientes:

3.2. Elementos

1. Periféricos
2. Tarjeta gráfica
3. Monitor
4. Disco duro
5. Procesador
6. Placas base
7. Ventiladores

3.3. Paquete Arules

Posteriormente, para comenzar importamos de nuevo *arules*.

```
> library(arules)
```

3.4. Lectura de datos

Cargamos los datos del csv en una variable que llamaremos “datos”.

```
> datos <- scan("./data/componentes.csv", sep=",")
> #MATRIZ DATOS
>
> matriz_datos <- Matrix(datos,
+ ncol=7,
+ nrow=5,
+ byrow=T,
+ sparse=T,
```

```

+ dimnames=list(c(1:5), c("Perifericos",
+       "Tarjeta Grafica",
+       "Monitor",
+       "Disco duro",
+       "Procesador",
+       "Placas base",
+       "Ventiladores")))
> print(matriz_datos)

5 x 7 sparse Matrix of class "dgCMatrix"
  Perifericos Tarjeta Grafica Monitor Disco duro Procesador Placas base
1           1              1      .           1           .           1
2           1              .      1           1           1           1
3           1              1      1           .           1           .
4           .              1      .           1           1           1
5           1              1      1           1           .           1
  Ventiladores
1           1
2           .
3           1
4           1
5           .

```

Observamos la matriz generada.

3.5. Analisis de asociacion

Convertimos nuestra matriz en una matriz dispersa para poder aplicarle el algoritmo Apriori a los datos que tenemos, con un soporte mayor o igual a 60 % y una confianza mayor o igual a 80 %.

Mostramos las asociaciones entre los distintos datos.

```

> asociaciones <- apriori(
+   as(t(as(matriz_datos,"nsparseMatrix")), "transactions"),
+   parameter=list(support=0.6,confidence=0.8))

```

Apriori

Parameter specification:

```

confidence minval smax arem aval originalSupport maxtime support minlen
      0.8    0.1    1 none FALSE              TRUE      5    0.6    1
maxlen target  ext
      10  rules TRUE

```

Algorithmic control:

```

filter tree heap memopt load sort verbose
  0.1 TRUE TRUE  FALSE TRUE    2    TRUE

```

Absolute minimum support count: 3

set item appearances ...[0 item(s)] done [0.00s].

```

set transactions ...[7 item(s), 5 transaction(s)] done [0.00s].
sorting and recoding items ... [7 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [12 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].

```

```
> inspect(asociaciones)
```

	lhs	rhs	support	confidence
[1]	{}	=> {Tarjeta Grafica}	0.8	0.8
[2]	{}	=> {Disco duro}	0.8	0.8
[3]	{}	=> {Placas base}	0.8	0.8
[4]	{}	=> {Perifericos}	0.8	0.8
[5]	{Ventiladores}	=> {Tarjeta Grafica}	0.6	1.0
[6]	{Monitor}	=> {Perifericos}	0.6	1.0
[7]	{Disco duro}	=> {Placas base}	0.8	1.0
[8]	{Placas base}	=> {Disco duro}	0.8	1.0
[9]	{Tarjeta Grafica,Disco duro}	=> {Placas base}	0.6	1.0
[10]	{Tarjeta Grafica,Placas base}	=> {Disco duro}	0.6	1.0
[11]	{Perifericos,Disco duro}	=> {Placas base}	0.6	1.0
[12]	{Perifericos,Placas base}	=> {Disco duro}	0.6	1.0

	coverage	lift	count
[1]	1.0	1.00	4
[2]	1.0	1.00	4
[3]	1.0	1.00	4
[4]	1.0	1.00	4
[5]	0.6	1.25	3
[6]	0.6	1.25	3
[7]	0.8	1.25	4
[8]	0.8	1.25	4
[9]	0.6	1.25	3
[10]	0.6	1.25	3
[11]	0.6	1.25	3
[12]	0.6	1.25	3

4. Conclusiones

Hemos realizado un análisis de asociación de datos con R, usando el algoritmo Apriori proporcionado por el paquete *arules*.