PECL2 - Fundamentos de la ciencia de datos

Marcos Barranquero Adrián Montesinos
Eduardo Graván

October 21, 2019

1 Apartado 1 - Análisis de asociación con la muestra de clase

1.1 Introducción

En este apartado realizamos un ejercicio en clase de analisis de asociación de datos en R con el algoritmo apriori. Para esto empleamos la misma muestra que se ha empleado en clase, sobre cestas de la compra.

1.2 Procedimiento

Primero, cargamos librería *arules* que contiene algoritmo apriori.

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

Procedemos ahora a cargar la matriz de asociaciones, etiquetada con sus cabeceras.

```
> matriz_datos <- Matrix(
+     c(1,1,0,1,1,1,1,1,1,0,1,1,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0),
+     6, 5, byrow=T, dimnames=list(
+         c("suceso1","suceso2","suceso3","suceso4","suceso5","suceso6"),
+         c("Pan","Agua","Cafe","Leche","Naranjas")),
+     sparse=T)
```

Antes de poder emplear el algoritmo apriori, tenemos que convertir la matriz a un objecto de transacciones a través de una matriz dispersa.

```
> muestra <- as(matriz_datos,"nsparseMatrix")
> transpuestaMatriz <- t(muestra)
> transacciones <- as(transpuestaMatriz, "transactions")
```

Aplicamos ahora el algoritmo apriori con $s \geq 50\%$ y $c \geq 80\%$, y mostramos el resultado por pantalla.

```
> 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 FALSE

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 lift count
[1] {}           => {Leche} 0.8333333 0.8333333  1.00 5
[2] {}           => {Pan}   0.8333333 0.8333333  1.00 5
[3] {Agua}       => {Pan}   0.6666667 1.0000000  1.20 4
[4] {Pan}        => {Agua}  0.6666667 0.8000000  1.20 4
[5] {Leche}      => {Pan}   0.6666667 0.8000000  0.96 4
[6] {Pan}        => {Leche} 0.6666667 0.8000000  0.96 4
[7] {Agua,Leche} => {Pan}   0.5000000 1.0000000  1.20 3
```

Podemos así ver qué asociaciones cumplen con nuestro criterio.

2 Apartado 2.1 - Análisis de asociación con distinta muestra

2.1 Introducción

En este apartado realizamos los mismos pasos para el uso de apriori con una muestra distinta dada, esta vez sobre ventas de extras en coches.

2.2 Procedimiento

De nuevo, cargamos la librería *arules*.

```
> library("arules") # Libreria arules
```

Construimos la matriz de asociaciones. Esta vez, la cargamos de un archivo usando la función read.transactions(), parte de *arules*, para cargar los datos a partir de una lista de extras vendidos.

```
> datos <- read.transactions("datos.txt", sep=" ")
> print(as(datos, "ngCMatrix"))

6 x 8 sparse Matrix of class "ngCMatrix"

A . . . . . . . |
B | | . | | . | .
C | | | . | . | .
N | . | | . | . .
T . | . | . . . |
X | | | | | . | .
```

Realizamos ahora el algoritmo apriori para obtener asociaciones, esta vez con $s \geq 40\%$ y $c \geq 90\%$.

```
> soporte <- apriori(datos, parameter=list(support=0.4, confidence=0.9))

Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.9    0.1    1 none FALSE            TRUE       5     0.4      1
 maxlen target   ext
     10  rules FALSE

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 ...[6 item(s), 8 transaction(s)] done [0.00s].
sorting and recoding items ... [4 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [3 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].

> inspect(soporte)

    lhs      rhs support confidence lift     count
[1] {C}   => {X} 0.625   1          1.333333 5
[2] {B}   => {X} 0.625   1          1.333333 5
[3] {B,C} => {X} 0.500   1          1.333333 4
```

Obtenemos así las asociaciones que cumplen nuestro criterio.

3 Apartado 2.2 - Ejercicio de clase modificado

3.1 Introducción

Finalmente, realizamos de nuevo el procedimiento original, pero aplicando los cambios que hemos visto convenientes. Hemos tomado el punto de vista de una tienda de servicios por cable que desea estudiar las asociaciones entre las ventas de sus servicios para poder elaborar paquetes de oferta. Los servicios en concreto son:

- Línea móvil
- Línea fija
- ADSL
- Fibra
- Televisión
- Películas
- Fútbol

Puesto que no tenemos los datos de estas compras, para realizar el ejercicio generaremos al azar una muestra de compras en un csv. Después, la estudiaremos con el algoritmo apriori.

3.2 Procedimiento

De nuevo, importamos *arules*.

```
> library(arules)
```

Para generar los datos, creamos una función encargada de generar datos de muestra aleatorios y guardarlos en un csv en la ruta dada. Esto nos servirá para usar usar distintas muestras.

```
> # nombre_archivo: nombre del archivo .CSV guardado
> # lista_articulos: posibles servicios de la tienda
> # tamano_muestra: nº de filas a generar
> generar_datos <- function(nombre_archivo, lista_articulos, tamano_muestra)
+ {
+     tamano_lista_articulos <- length(lista_articulos)
+
+     # Creamos tabla de la muestra
+     muestra <- matrix(
+         sample(0:1, length(tamano_muestra) * tamano_lista_articulos, replace=TRUE),
+         tamano_muestra,
+         tamano_lista_articulos)
+
+     # Escribimos en el csv
+     write.table(
```

```
+         muestra,
+         file=paste("./", nombre_archivo, ".csv", sep=""),
+         row.names=FALSE,
+         col.names=FALSE,
+         sep=",")
+ }
```

Generamos ahora el csv con muestra de servicios comprados a partir de una lista de posibles productos.

```
> lista_articulos <- c("Linea movil",
+                      "Linea fija",
+                      "ADSL",
+                      "Fibra",
+                      "Television",
+                      "Peliculas",
+                      "Fútbol")
> generar_datos("datos_aleatorios", lista_articulos, 5)
```

Leemos ahora los datos generados a una tabla y la convertimos en una matriz.

```
> datos <- scan("datos_aleatorios.csv", sep=",")
> matriz_datos <- Matrix(
+     datos,
+     ncol=length(lista_articulos),
+     nrow=5,
+     byrow=T,
+     sparse=T,
+     dimnames=list(c(1:5), lista_articulos))
> print(matriz_datos)

5 x 7 sparse Matrix of class "dgCMatrix"
  Linea movil Linea fija ADSL Fibra Television Peliculas Fútbol
1           .          .    1     1          .         1      1
2           1          .    1     1          .         .      1
3           1          .    .     1          1         .      1
4           1          1    .     1          1         .      .
5           1          1    .     .          1         1      .
```

Podemos ver por tanto la matriz generada. Convirtiendo la matriz a una matrix dispersa, podemos ya aplicar el algoritmo apriori sobre la muestra, con $s \geq 60\%$ y $c \geq 80\%$.

```
> 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 FALSE

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 ... [4 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 done [0.00s].
writing ... [4 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].

> inspect(asociaciones)

    lhs             rhs           support confidence lift count
[1] {}           => {Fibra}       0.8     0.8        1.00 4
[2] {}           => {Linea movil} 0.8     0.8        1.00 4
[3] {Fútbol}     => {Fibra}       0.6     1.0        1.25 3
[4] {Television} => {Linea movil} 0.6     1.0        1.25 3
```

Obtenemos finalmente asociaciones entre nuestros servicios.