Tema8_Ejercicio_reglas_asociacion_ECLAT

Fran Camacho

2025-03-13

Tema 8 - Ejercicio Reglas de Asociación (Usando ECLAT en lugar de Apriori)

Utilizando el dataset IncomeESL incluido con la librería arules, se pide generar reglas de asociación.

Para ello, previamente deberá depurar el dataset. En particular:

- Revisar que no haya valores omitidos.
- Transformar los factores en valores numéricos. ← no es necesario!!!
- Una vez depurado el dataset, crear la matriz de transacciones usando la función transactions.

A la hora de ejecutar el algoritmo para obtener las reglas, no olvide establecer los valores de los parámetros de la función apriori, justificando el motivo de su elección.

Por último, elabore un breve informe resumiendo las reglas obtenidas y analizando su significado.

Paso 1: Carga de los datos

```
if (!require(arules)) install.packages('arules', dependencies =
T)library(arules)

# to plot rules we use package arulesViz
if (!require(arulesViz)) {
  # package graph -and other packages too- must be installed first
(and it is not available anymore in CRAN)
  if (!require("BiocManager", quietly = TRUE)) {
    install.packages("BiocManager")
    BiocManager::install("Rgraphviz")
    BiocManager::install("graph")
  }
  install.packages('arulesViz', dependencies = T)
}
```

```r
data("IncomeESL")
#data("Income")
```

Descripción de las variables del dataset:

https://rdrr.io/cran/arules/man/Income.html

income: an ordered factor with levels: [0,10) < [10,15) < [15,20) < [20,25) < [25,30) < [30,40) < [40,50) < [50,75) < 75+

sex: a factor with levels: male female

marital status: a factor with levels: married cohabitation divorced widowed single

age: an ordered factor with levels: 14-17 < 18-24 < 25-34 < 35-44 < 45-54 < 55-64 < 65+

education: an ordered factor with levels: grade <9 < grades 9-11 < high school graduate < college (1-3 years) < college graduate < graduate study

occupation: a factor with levels: professional/managerial sales laborer clerical/service homemaker student military retired unemployed

years in bay area: an ordered factor with levels: <1 < 1-3 < 4-6 < 7-10 < >10

dual incomes: a factor with levels: not married yes no

number in household: an ordered factor with levels: 1 < 2 < 3 < 4 < 5 < 6 < 7 < 8 < 9+

number of children: an ordered factor with levels: 0 < 1 < 2 < 3 < 4 < 5 < 6 < 7 < 8 < 9+

householder status: a factor with levels: own, rent, live with parents/family

type of home: a factor with levels: house condominium apartment mobile Home other

ethnic classification: a factor with levels: american indian, asian black east indian hispanic pacific islander white other

language in home: a factor with levels: english spanish other

Paso 2: Explorar y preparar los datos

```r
income_raw <- IncomeESL

#structure
str(income_raw)
```

```
## 'data.frame':    8993 obs. of  14 variables:
##  $ income              : Ord.factor w/ 9 levels "[0,10)"<"[10,15)"<..: 9 9 9 1 1 8 1 6 2
4 ...
##  $ sex                 : Factor w/ 2 levels "male","female": 2 1 2 2 2 1 1 1 1 1 ...
##  $ marital status      : Factor w/ 5 levels "married","cohabitation",..: 1 1 1 5 5 1 5 3 1
1 ...
##  $ age                 : Ord.factor w/ 7 levels "14-17"<"18-24"<..: 5 5 3 1 1 6 2 3 6 7 ...
##  $ education           : Ord.factor w/ 6 levels "grade <9"<"grades 9-11"<..: 4 5 5 2 2 4 3 4
3 4 ...
##  $ occupation          : Factor w/ 9 levels "professional/managerial",..: 5 5 1 6 6 8 9 3 8 8
...
##  $ years in bay area   : Ord.factor w/ 5 levels "<1"<"1-3"<"4-6"<..: 5 5 5 5 3 5 4 5 5 4 ...
##  $ dual incomes        : Factor w/ 3 levels "not married",..: 3 3 2 1 1 3 1 1 3 3 ...
##  $ number in household : Ord.factor w/ 9 levels "1"<"2"<"3"<"4"<..: 3 5 3 4 4 2 3 1 3 2 ...
##  $ number of children  : Ord.factor w/ 10 levels "0"<"1"<"2"<"3"<..: 1 3 2 3 3 1 2 1 1 1 ...
##  $ householder status  : Factor w/ 3 levels "own","rent","live with parents/family": 1 1 2 3
3 1 2 2 2 2 ...
##  $ type of home        : Factor w/ 5 levels "house","condominium",..: 1 1 3 1 1 1 3 3 3 3 ...
##  $ ethnic classification: Factor w/ 8 levels "american indian",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ language in home    : Factor w/ 3 levels "english","spanish",..: NA 1 1 1 1 1 1 1 1 1 ...
```

Nos quedamos con las observaciones que estén completas:

```r
# only complete observations
income_complete <- income_raw[complete.cases(income_raw), ]
```

Y obtenemos las transacciones con ellas:

```r
# get transactions
income_transac <- transactions(income_complete)
```

Estructura y sumario de las transacciones:

```r
#See the structure
str(income_transac)
```

```
## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:96264] 8 9 11 20 27 33 42 45 50 57 ...
##   .. .. ..@ p       : int [1:6877] 0 14 28 42 56 70 84 98 112 126 ...
##   .. .. ..@ Dim     : int [1:2] 84 6876
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : NULL
##   .. .. ..@ factors : list()
##   ..@ itemInfo   :'data.frame':  84 obs. of  3 variables:
##   .. ..$ labels   : chr [1:84] "income=[0,10)" "income=[10,15)" "income=[15,20)"
"income=[20,25)" ...
##   .. ..$ variables: Factor w/ 14 levels "age","dual incomes",..: 6 6 6 6 6 6 6 6 6 12 ...
##   .. ..$ levels   : Factor w/ 73 levels "[0,10)","[10,15)",..: 1 2 3 4 5 6 7 8 29 54 ...
##   ..@ itemsetInfo:'data.frame':  6876 obs. of  1 variable:
##   .. ..$ transactionID: chr [1:6876] "2" "3" "4" "5" ...
```

Resumen estadístico

```r
#See the structure
summary(income_transac)
```

```
## transactions as itemMatrix in sparse format with
##  6876 rows (elements/itemsets/transactions) and
##  84 columns (items) and a density of 0.1666667
##
## most frequent items:
##    language in home=english ethnic classification=white
##                        6277                        4605
##       years in bay area=>10        number of children=0
##                        4446                        4276
##    dual incomes=not married                     (Other)
##                        4114                       72546
##
## element (itemset/transaction) length distribution:
## sizes
##   14
## 6876
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      14      14      14      14      14      14
##
## includes extended item information - examples:
##           labels variables  levels
## 1  income=[0,10)    income  [0,10)
## 2 income=[10,15)    income [10,15)
## 3 income=[15,20)    income [15,20)
##
## includes extended transaction information - examples:
##   transactionID
## 1             2
## 2             3
## 3             4
```

La matriz dispersa tiene 6876 filas y 84 columnas. Al no tratarse de un problema como el de la cesta de la compra, y al tener las filas originales todas 14 valores por haber eliminado las filas que tenían valores nulos, todas las transacciones tienen 14 valores. La densidad es el 16.67%. (No soy sociólogo, pero los items más frecuentes, dan una idea bastante clara de la población de la muestra).

Examinamos un par de transacciones con “inspect”:

```r
inspect(income_transac[1:2])
```

```
##     items                                   transactionID
## [1] {income=75+,
##      sex=male,
##      marital status=married,
##      age=45-54,
##      education=college graduate,
##      occupation=homemaker,
##      years in bay area=>10,
##      dual incomes=no,
##      number in household=5,
```

```
##      number of children=2,
##      householder status=own,
##      type of home=house,
##      ethnic classification=white,
##      language in home=english}                        2
## [2] {income=75+,
##      sex=female,
##      marital status=married,
##      age=25-34,
##      education=college graduate,
##      occupation=professional/managerial,
##      years in bay area=>10,
##      dual incomes=yes,
##      number in household=3,
##      number of children=1,
##      householder status=rent,
##      type of home=apartment,
##      ethnic classification=white,
##      language in home=english}                        3
```

Visualización de los datos

(Ver notebook con la versión Apriori).

Paso 3: Entrenamiento del modelo

Vamos a probar primero con los valores por defecto:

```
income_items_eclat <- eclat(income_transac)
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE     0.1      1     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 687
##
## create itemset ...
## set transactions ...[84 item(s), 6876 transaction(s)] done [0.01s].
## sorting and recoding items ... [42 item(s)] done [0.00s].
## creating bit matrix ... [42 row(s), 6876 column(s)] done [0.00s].
## writing  ... [1204 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

Con los parámetros por defecto (soporte 0.1), se obtienen 1204 itemsets.

Vamos a aumentar tanto el soporte como la confianza requeridas:

```
#income_items_eclat <- eclat(income_transac, parameter = list(support
= 0.25, minlen = 2))  # confidence = 0.5
```

```
income_items_eclat_02 <- eclat(income_transac, parameter =
list(support = 0.2, minlen = 2))  # confidence = 0.5
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen           target  ext
##     FALSE     0.2      2     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 1375
##
## create itemset ...
## set transactions ...[84 item(s), 6876 transaction(s)] done [0.01s].
## sorting and recoding items ... [21 item(s)] done [0.00s].
## creating bit matrix ... [21 row(s), 6876 column(s)] done [0.00s].
## writing  ... [189 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

Ahora obtenemos 189 itemsets:

```
inspect(income_items_eclat_02[1:10])
```

```
##      items                         support count
## [1]  {age=18-24,
##       language in home=english} 0.2063700  1419
## [2]  {age=18-24,
##       dual incomes=not married} 0.2017161  1387
## [3]  {marital status=married,
##       dual incomes=yes,
##       language in home=english} 0.2190227  1506
## [4]  {dual incomes=yes,
##       language in home=english} 0.2338569  1608
## [5]  {marital status=married,
##       dual incomes=yes}         0.2370564  1630
## [6]  {age=25-34,
##       language in home=english} 0.2374927  1633
## [7]  {householder status=rent,
##       type of home=apartment,
##       language in home=english} 0.2306574  1586
## [8]  {number of children=0,
##       householder status=rent,
##       type of home=apartment}   0.2052065  1411
## [9]  {type of home=apartment,
##       language in home=english} 0.2497091  1717
## [10] {number of children=0,
##       type of home=apartment}   0.2137871  1470
```

Obtenemos las reglas de esos 2 itemsets:

```
income_rules_eclat <- ruleInduction(income_items_eclat, confidence =
0.9)

income_rules_eclat
```

```
## set of 802 rules

income_rules_eclat_02 <- ruleInduction(x=income_items_eclat_02,
transactions=income_transac, confidence = 0.8)

income_rules_eclat_02

## set of 132 rules

inspect(sort(income_rules_eclat_02, by = "lift")[1:10])

##      lhs                           rhs                           support confidence     lift
itemset
## [1]  {dual incomes=yes,

##       language in home=english} => {marital status=married}   0.2190227  0.9365672 2.428294
3
## [2]  {dual incomes=yes}         => {marital status=married}   0.2370564  0.9357061 2.426061
5
## [3]  {number of children=0,

##       type of home=apartment}   => {householder status=rent}  0.2052065  0.9598639 2.290085
8
## [4]  {marital status=married,

##       type of home=house,

##       language in home=english} => {householder status=own}   0.2207679  0.8433333 2.244102
38
## [5]  {type of home=apartment,

##       language in home=english} => {householder status=rent}  0.2306574  0.9237041 2.203813
7
## [6]  {marital status=married,

##       type of home=house}       => {householder status=own}   0.2329843  0.8279070 2.203053
43
## [7]  {type of home=apartment}   => {householder status=rent}  0.2507272  0.9097625 2.170551
12
## [8]  {sex=male,

##       marital status=single}    => {dual incomes=not married} 0.2089878  1.0000000 1.671366
76
## [9]  {sex=female,

##       marital status=single}    => {dual incomes=not married} 0.2001163  1.0000000 1.671366
77
## [10] {marital status=single,

##       type of home=house,

##       language in home=english} => {dual incomes=not married} 0.2001163  1.0000000 1.671366
78
```

De todas maneras, mejor hacemos el mismo tratamiento de los ingresos que con Apriori:

```
income_complete["income3levels"] <- income_complete["income"]
levels(income_complete[["income3levels"]]) <- c("0-20k$","0-20k$","0-
20k$","20k-50k$","20k-50k$","20k-50k$","20k-50k$","50k+","50k+")
#test
income_complete[0:5,c("income","income3levels")]
```

```
##    income income3levels
## 2     75+          50k+
## 3     75+          50k+
## 4  [0,10)        0-20k$
## 5  [0,10)        0-20k$
## 6 [50,75)          50k+
```

```
# remove income variable with 9 levels
income3L_complete <- income_complete[,-1]
```

Y obtenemos las transacciones con este dataframe

```
# get transactions
income3L_transac <- transactions(income3L_complete)
```

Aplicamos ECLAT:

```
#default params
income3L_items_eclat <- eclat(income3L_transac)
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE     0.1      1     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 687
##
## create itemset ...
## set transactions ...[78 item(s), 6876 transaction(s)] done [0.01s].
## sorting and recoding items ... [40 item(s)] done [0.00s].
## creating bit matrix ... [40 row(s), 6876 column(s)] done [0.00s].
## writing  ... [1505 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

```
# support=0.1 (confidence=0.55 later)
income3L_items_eclat_01 <- eclat(income3L_transac, parameter =
list(support = 0.1, minlen = 2))
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE     0.1      2     10 frequent itemsets TRUE
```

```
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 687
##
## create itemset ...
## set transactions ...[78 item(s), 6876 transaction(s)] done [0.01s].
## sorting and recoding items ... [40 item(s)] done [0.00s].
## creating bit matrix ... [40 row(s), 6876 column(s)] done [0.00s].
## writing  ... [1465 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

Obtención de reglas de los itemsets:

```
#default parameters
income3L_rules_eclat <- ruleInduction(income3L_items_eclat)
income3L_rules_eclat

## set of 1389 rules

income3L_rules_eclat_01 <- ruleInduction(x=income3L_items_eclat_01,
transactions=income3L_transac, confidence = 0.55)
#income3L_rules_eclat_01 <- ruleInduction(income3L_items_eclat_01)
income3L_rules_eclat_01

## set of 3326 rules
```

[

Comparación con Apriori:

```
# support = 0.5, confidence = 0.5 -> same rules as with Apriori
income3L_items_eclat_05 <- eclat(income3L_transac, parameter =
list(support = 0.5, minlen = 2))

## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE     0.5      2     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 3438
##
## create itemset ...
## set transactions ...[78 item(s), 6876 transaction(s)] done [0.01s].
## sorting and recoding items ... [7 item(s)] done [0.00s].
## creating bit matrix ... [7 row(s), 6876 column(s)] done [0.00s].
## writing  ... [6 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

```
income3L_rules_eclat_05_05 <- ruleInduction(x=income3L_items_eclat_05,
transactions=income3L_transac, confidence = 0.5)

income3L_rules_eclat_05_05

## set of 12 rules

inspect(sort(income3L_rules_eclat_05_05, by = "lift"))
```

```
##      lhs                              rhs                              support confidence
lift itemset
## [1]  {language in home=english}    => {ethnic classification=white} 0.6595404  0.7224789
1.078776       6
## [2]  {ethnic classification=white} => {language in home=english}    0.6595404  0.9847991
1.078776       6
## [3]  {language in home=english}    => {number of children=0}        0.5801338  0.6354947
1.021904       4
## [4]  {number of children=0}        => {language in home=english}    0.5801338  0.9328812
1.021904       4
## [5]  {language in home=english}    => {years in bay area=>10}       0.6013671  0.6587542
1.018802       5
## [6]  {years in bay area=>10}       => {language in home=english}    0.6013671  0.9300495
1.018802       5
## [7]  {language in home=english}    => {sex=female}                  0.5122164  0.5610961
1.012890       1
## [8]  {sex=female}                  => {language in home=english}    0.5122164  0.9246521
1.012890       1
## [9]  {language in home=english}    => {type of home=house}          0.5446481  0.5966226
1.000092       2
## [10] {type of home=house}          => {language in home=english}    0.5446481  0.9129693
1.000092       2
## [11] {language in home=english}    => {dual incomes=not married}    0.5426120  0.5943922
0.993447       3
## [12] {dual incomes=not married}    => {language in home=english}    0.5426120  0.9069033
0.993447       3
```

Se obtienen las mismas reglas que con Apriori :)

]

Vamos a intentar encontrar ahora las reglas redundantes (con la función **is.subset()**), eliminarlas y simplicar así el conjunto de reglas:

```
income3L_rules_eclat_01_sortedByLift <- sort(income3L_rules_eclat_01, by =
"lift")
#inspect(income3L_rules_RHS_sortedByLift)

subset_matrix ← is.subset
(income3L_rules_eclat_01_sortedByLift,income3L_rules_eclat_01_sortedByLift)
#subset_matrix
```

[

3326 x 3326 sparse Matrix of class “ngCMatrix” [[suppressing 88 column names ‘{marital status=single,dual incomes=not married,householder status=live with parents/family,type of home=house,income3levels=0-20k$}’, ‘{marital status=single,householder status=live with parents/family,type of home=house,income3levels=0-20k$}’, ‘{marital status=single,occupation=student,dual incomes=not married,householder status=live with parents/family}’ ...]] [[suppressing 88 column names ‘{marital status=single,dual incomes=not married,householder status=live with parents/family,type of home=house,income3levels=0-20k$}’, ‘{marital status=single,householder status=live with parents/family,type of home=house,income3levels=0-20k$}’, ‘{marital status=single,occupation=student,dual incomes=not married,householder status=live with parents/family}’ ...]]

. . .

]

Eliminamos las ocurrencias que se producen en la diagonal principal y por debajo de ella:

```
subset_matrix[lower.tri(subset_matrix, diag = T)] <- F
#subset_matrix
```

Ahora podemos extraer las reglas de asociación redundantes, que serán aquellas que tengan una columna cuya suma sea igual o superior a 1:

```
redundant <- colSums(subset_matrix, na.rm = T) >= 1
#redundant
```

Las reglas marcadas como TRUE son consideradas redundantes por lo que podemos eliminarlas:

```
rules_pruned <- income3L_rules_eclat_01_sortedByLift[!redundant]
rules_pruned
```

```
## set of 818 rules

inspect(rules_pruned[1:20])

##      lhs                                                rhs
support confidence     lift itemset
## [1]  {marital status=single,

##       dual incomes=not married,

##       type of home=house,

##       income3levels=0-20k$}                          => {householder status=live with
parents/family} 0.1060209  0.7984666 3.893799     131
## [2]  {marital status=single,

##       type of home=house,

##       income3levels=0-20k$}                          => {householder status=live with
parents/family} 0.1060209  0.7984666 3.893799     135
## [3]  {marital status=single,

##       occupation=student,

##       dual incomes=not married}                      => {householder status=live with
parents/family} 0.1138743  0.7853561 3.829864      25
## [4]  {marital status=single,

##       occupation=student}                            => {householder status=live with
parents/family} 0.1138743  0.7853561 3.829864      27
## [5]  {occupation=student,

##       dual incomes=not married}                      => {householder status=live with
parents/family} 0.1147469  0.7471591 3.643593      26
## [6]  {marital status=single,

##       years in bay area=>10,

##       dual incomes=not married,

##       type of home=house}                            => {householder status=live with
parents/family} 0.1176556  0.7268643 3.544624     145
## [7]  {marital status=single,

##       years in bay area=>10,

##       type of home=house}                            => {householder status=live with
parents/family} 0.1176556  0.7268643 3.544624     148
## [8]  {dual incomes=not married,

##       type of home=house,

##       income3levels=0-20k$}                          => {householder status=live with
parents/family} 0.1119837  0.7129630 3.476832     136
## [9]  {householder status=live with parents/family} => {occupation=student}
0.1153287  0.5624113 3.428316      46
## [10] {marital status=single,

##       dual incomes=not married,

##       type of home=house}                            => {householder status=live with
```

```
parents/family} 0.1559046  0.6845466 3.338257     149
## [11] {marital status=single,

##       type of home=house}                          => {householder status=live with
parents/family} 0.1559046  0.6845466 3.338257     159
## [12] {marital status=married,

##       occupation=professional/managerial,

##       language in home=english}                    => {dual incomes=yes}
0.1242001  0.8087121 3.192138     257
## [13] {marital status=married,

##       occupation=professional/managerial}          => {dual incomes=yes}
0.1301629  0.8070334 3.185512     262
## [14] {type of home=house,

##       income3levels=0-20k$}                        => {householder status=live with
parents/family} 0.1140198  0.6484698 3.162325     140
## [15] {marital status=single,

##       dual incomes=not married,

##       income3levels=0-20k$}                        => {householder status=live with
parents/family} 0.1316172  0.5801282 2.829051     134
## [16] {marital status=single,

##       income3levels=0-20k$}                        => {householder status=live with
parents/family} 0.1316172  0.5801282 2.829051     141
## [17] {years in bay area=>10,

##       dual incomes=not married,

##       type of home=house}                          => {householder status=live with
parents/family} 0.1250727  0.5562743 2.712725     165
## [18] {marital status=married,

##       language in home=english,

##       income3levels=50k+}                          => {dual incomes=yes}
0.1151832  0.6863085 2.708988     249
## [19] {marital status=married,

##       income3levels=50k+}                          => {dual incomes=yes}
0.1194008  0.6847373 2.702786     252
## [20] {marital status=single,

##       years in bay area=>10,

##       dual incomes=not married}                    => {householder status=live with
parents/family} 0.1364165  0.5501466 2.682843     153
```

Pasamos a tener 818 reglas en total (mostramos solo 20 por brevedad).

Filtramos las que tengan en el consequente/RHS los niveles de ingresos.

Exactamente un nivel:

```
inspect(subset(rules_pruned, subset = rhs %in% "income3levels=0-
20k$"))
```

```
##      lhs                                                rhs                      support
confidence     lift itemset
## [1]  {occupation=student,

##       dual incomes=not married,

##       language in home=english}                    => {income3levels=0-20k$} 0.1025305
0.7730263 2.322118      29
## [2]  {marital status=single,

##       occupation=student,

##       dual incomes=not married}                    => {income3levels=0-20k$} 0.1118383
0.7713139 2.316975      28
## [3]  {marital status=single,

##       occupation=student}                          => {income3levels=0-20k$} 0.1118383
0.7713139 2.316975      32
## [4]  {occupation=student,

##       dual incomes=not married}                    => {income3levels=0-20k$} 0.1182373
0.7698864 2.312686      31
## [5]  {occupation=student,

##       language in home=english}                    => {income3levels=0-20k$} 0.1061664
0.7510288 2.256039      30
## [6]  {occupation=student}                          => {income3levels=0-20k$} 0.1231821
0.7508865 2.255612      45
## [7]  {dual incomes=not married,

##       householder status=live with parents/family} => {income3levels=0-20k$} 0.1394706
0.6979622 2.096631     139
## [8]  {householder status=live with parents/family} => {income3levels=0-20k$} 0.1423793
0.6943262 2.085709     184
## [9]  {sex=female,

##       marital status=single}                       => {income3levels=0-20k$} 0.1237638
0.6184593 1.857810     657
## [10] {age=18-24,

##       dual incomes=not married,

##       language in home=english}                    => {income3levels=0-20k$} 0.1022397
0.5738776 1.723889     200
## [11] {age=18-24,

##       dual incomes=not married}                    => {income3levels=0-20k$} 0.1156195
0.5731795 1.721792     202
## [12] {age=18-24}                                   => {income3levels=0-20k$} 0.1295812
0.5534161 1.662424     248
```

Todos los niveles:

```
rules_income3L <- subset(rules_pruned, subset = rhs %pin% "income3levels")
inspect(rules_income3L)

##      lhs                                                rhs                        support
confidence     lift itemset
## [1]  {dual incomes=yes,

##       householder status=own}                       => {income3levels=50k+}     0.1023851
```

```
0.6622766 2.513142     253
## [2]  {marital status=married,

##       occupation=professional/managerial,

##       language in home=english}                   => {income3levels=50k+}     0.1012216
0.6590909 2.501054     368
## [3]  {occupation=professional/managerial,

##       householder status=own,

##       language in home=english}                   => {income3levels=50k+}     0.1090750
0.6590510 2.500902     367
## [4]  {occupation=professional/managerial,

##       householder status=own}                     => {income3levels=50k+}     0.1122746
0.6547922 2.484741     375
## [5]  {marital status=married,

##       occupation=professional/managerial}         => {income3levels=50k+}     0.1048575
0.6501353 2.467070     374
## [6]  {occupation=student,

##       dual incomes=not married,

##       language in home=english}                   => {income3levels=0-20k$}   0.1025305
0.7730263 2.322118      29
## [7]  {marital status=single,

##       occupation=student,

##       dual incomes=not married}                   => {income3levels=0-20k$}   0.1118383
0.7713139 2.316975      28
## [8]  {marital status=single,

##       occupation=student}                         => {income3levels=0-20k$}   0.1118383
0.7713139 2.316975      32
## [9]  {occupation=student,

##       dual incomes=not married}                   => {income3levels=0-20k$}   0.1182373
0.7698864 2.312686      31
## [10] {occupation=student,

##       language in home=english}                   => {income3levels=0-20k$}   0.1061664
0.7510288 2.256039      30
## [11] {occupation=student}                         => {income3levels=0-20k$}   0.1231821
0.7508865 2.255612      45
## [12] {dual incomes=yes,

##       type of home=house}                         => {income3levels=50k+}     0.1009308
0.5827036 2.211187     251
## [13] {occupation=professional/managerial,

##       type of home=house,

##       language in home=english}                   => {income3levels=50k+}     0.1090750
0.5660377 2.147945     369
## [14] {marital status=married,

##       type of home=house,

##       ethnic classification=white,
```

```
##       language in home=english}                    => {income3levels=50k+}     0.1185282
0.5659722 2.147696     404
## [15] {marital status=married,

##       type of home=house,

##       ethnic classification=white}                 => {income3levels=50k+}     0.1194008
0.5654270 2.145627     406
## [16] {occupation=professional/managerial,

##       type of home=house}                          => {income3levels=50k+}     0.1128563
0.5578720 2.116958     373
## [17] {dual incomes=not married,

##       householder status=live with parents/family} => {income3levels=0-20k$}   0.1394706
0.6979622 2.096631     139
## [18] {householder status=live with parents/family} => {income3levels=0-20k$}   0.1423793
0.6943262 2.085709     184
## [19] {sex=female,

##       marital status=single}                       => {income3levels=0-20k$}   0.1237638
0.6184593 1.857810     657
## [20] {age=18-24,

##       dual incomes=not married,

##       language in home=english}                    => {income3levels=0-20k$}   0.1022397
0.5738776 1.723889     200
## [21] {age=18-24,

##       dual incomes=not married}                    => {income3levels=0-20k$}   0.1156195
0.5731795 1.721792     202
## [22] {age=18-24}                                  => {income3levels=0-20k$}   0.1295812
0.5534161 1.662424     248
## [23] {occupation=professional/managerial,

##       dual incomes=not married}                    => {income3levels=20k-50k$} 0.1016579
0.5964164 1.477823     747
```

Visualización de las reglas

Con la ayuda de la librería arulesViz, vamos a visualizar las reglas obtenidas, para ver si podemos deducir algo más:

```
plot(rules_income3L)
```

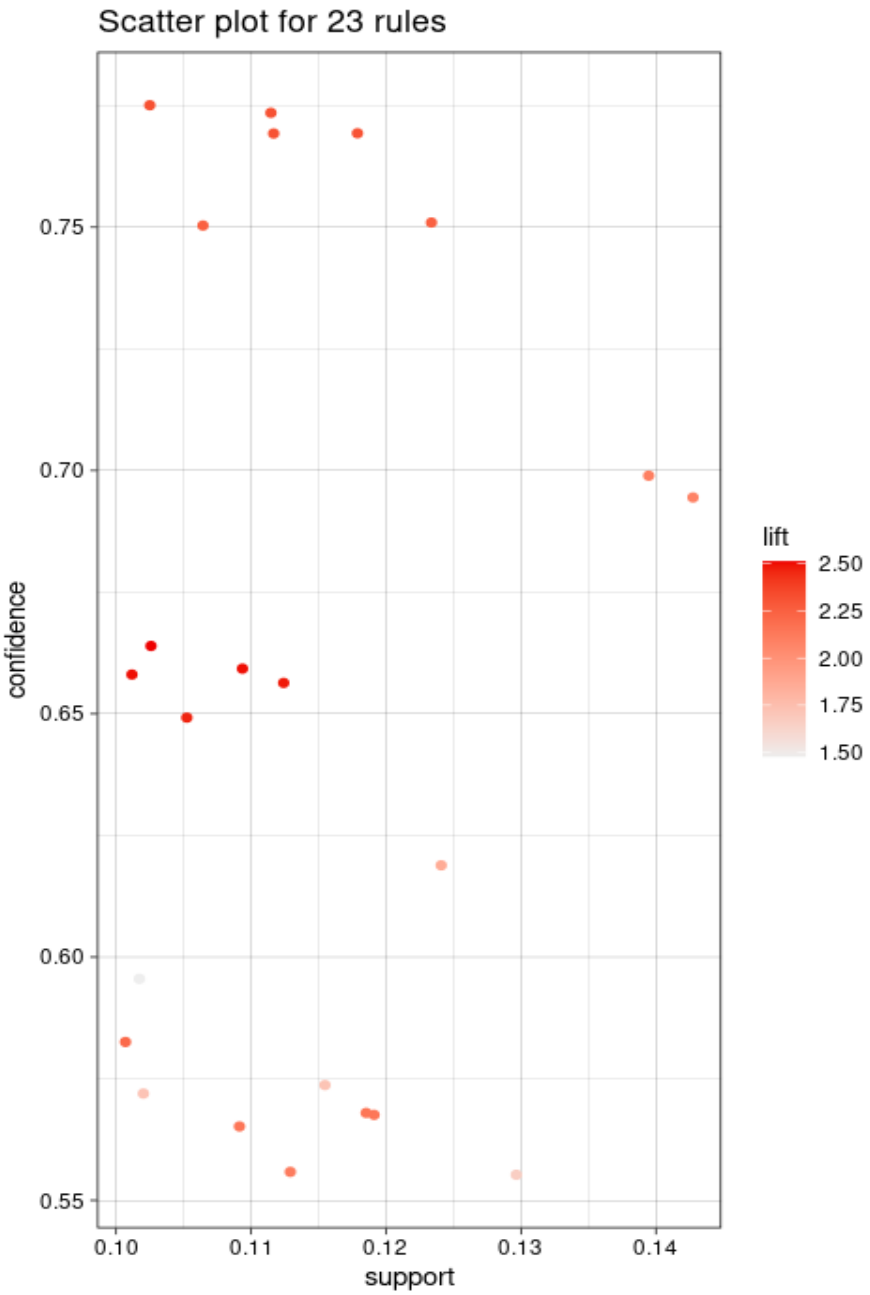

Gráfico de grupos de reglas:

```
plot(rules_income3L, method="grouped")
```

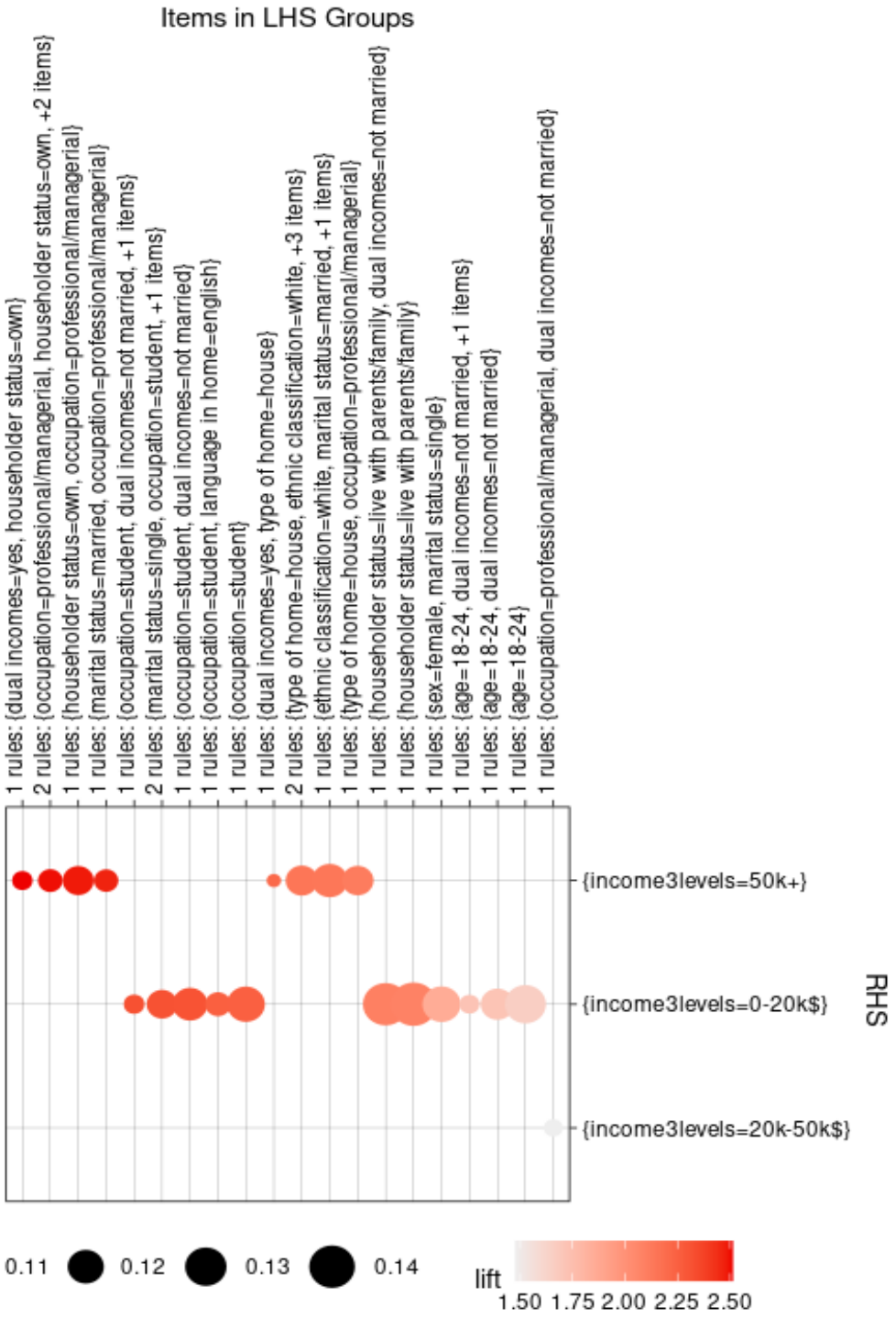

Gráfico de conexiones entre reglas:

```
plot(rules_income3L, method="graph")
```