Association Rule Mining

Mauro Travieso

Installing the required libraries

```
#install.packages("arules")
library("arules")
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
#create a sparse matrix
#grocery <- read.transactions(".\\demoData\\grocery.csv",  sep = ",")
#summary(grocery)
```

R has this dataset Groceries with 9835 rows

```
data("Groceries")
summary(Groceries)
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.02609146
##
## most frequent items:
##       whole milk other vegetables       rolls/buns             soda
##             2513             1903             1809             1715
##           yogurt          (Other)
##             1372            34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.409   6.000  32.000
##
## includes extended item information - examples:
##        labels  level2           level1
## 1 frankfurter sausage meat and sausage
## 2     sausage sausage meat and sausage
## 3  liver loaf sausage meat and sausage
```

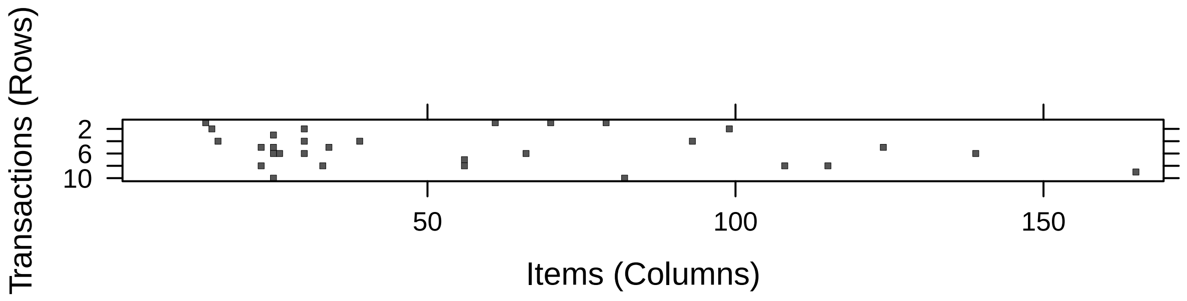
Inspecting the first five transactions

```r
inspect(Groceries[1:5])
```

```
##     items
## [1] {citrus fruit,
##      semi-finished bread,
##      margarine,
##      ready soups}
## [2] {tropical fruit,
##      yogurt,
##      coffee}
## [3] {whole milk}
## [4] {pip fruit,
##      yogurt,
##      cream cheese ,
##      meat spreads}
## [5] {other vegetables,
##      whole milk,
##      condensed milk,
##      long life bakery product}
```

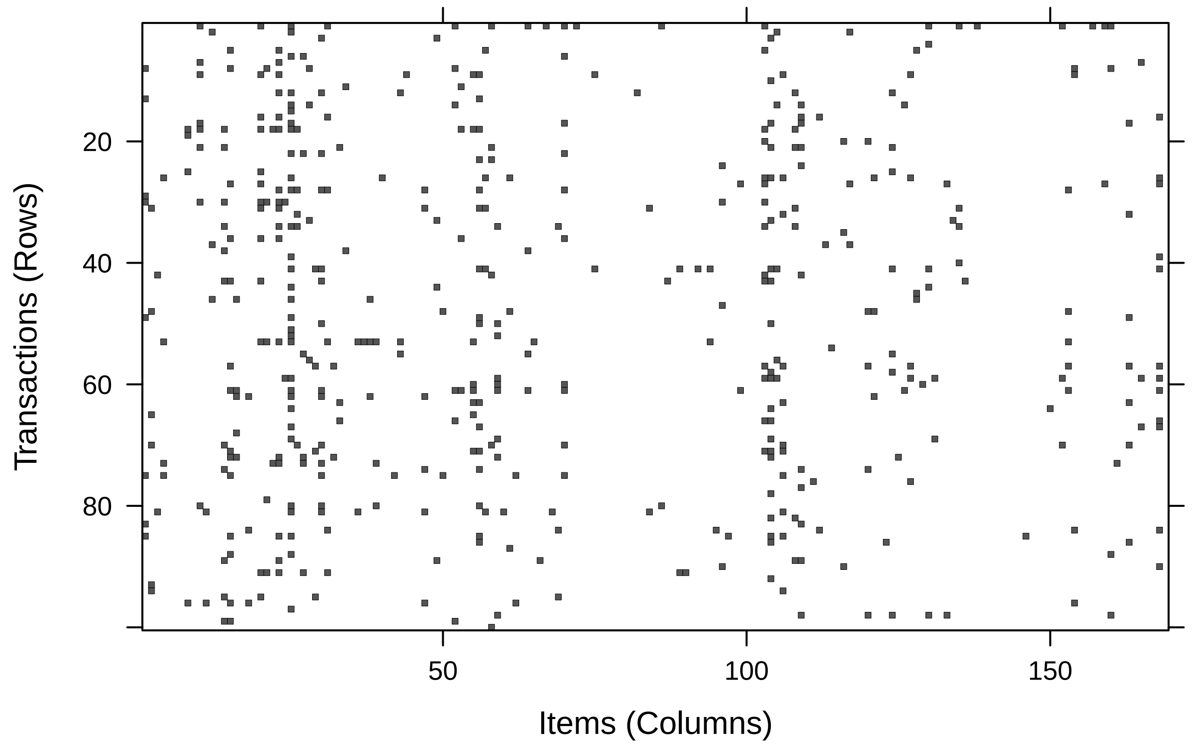
Visualizing the first 10 rows of sparse matrix

```r
image(Groceries[1:10])
```



Visualizing the randomly sampled 100 rows of sparse matrix

```
image(sample(Groceries,100))
```



Examining a particular item(a column of data)

Proportion of transactions that contain the item

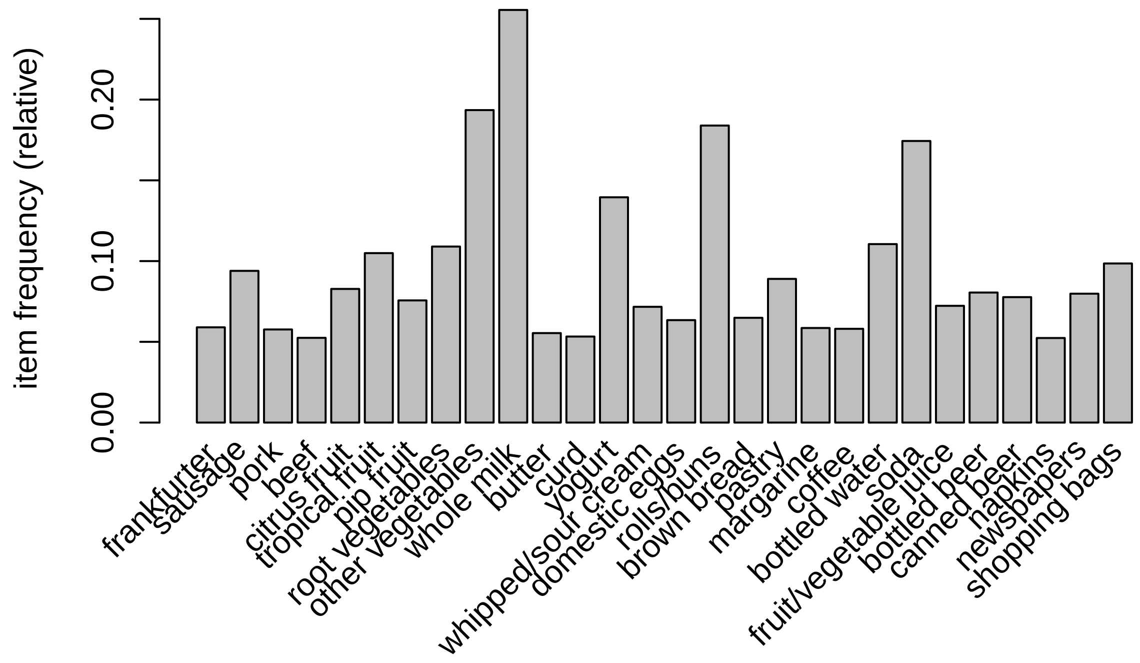
```
itemFrequency(Groceries[, 1:3])
```

```
## frankfurter     sausage  liver loaf
## 0.058973055 0.093950178 0.005083884
```

(a) Plot of interest

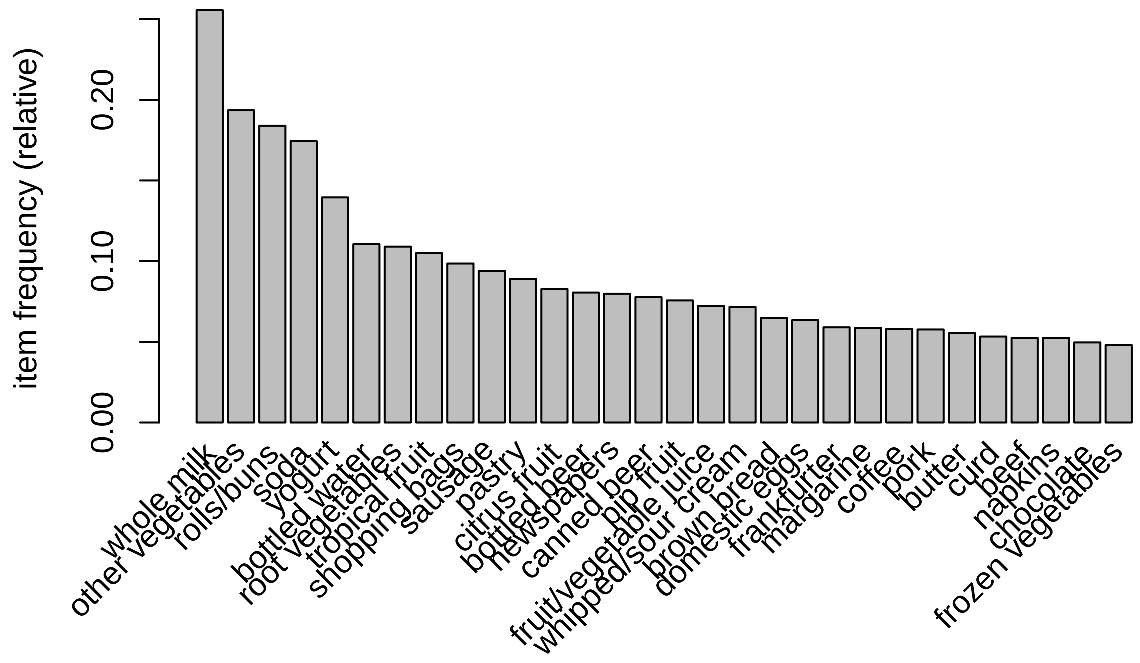
plot frequent items with min support = 0.05

```
itemFrequencyPlot(Groceries, support = 0.05)
```



Plot top 30 frequent items

```
itemFrequencyPlot(Groceries, topN = 30)
```



(b) Use apriori to generate rules

```r
rules <- apriori(Groceries,
                 parameter = list(support = 0.006, confidence = 0.25, minlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.25    0.1    1 none FALSE            TRUE       5   0.006      2
##  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: 59
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [109 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [463 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(rules)
```

```
## set of 463 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3   4
## 150 297  16
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   3.000   2.711   3.000   4.000
##
## summary of quality measures:
##     support           confidence          lift              count
##  Min.   :0.006101   Min.   :0.2500   Min.   :0.9932   Min.   : 60.0
##  1st Qu.:0.007117   1st Qu.:0.2971   1st Qu.:1.6229   1st Qu.: 70.0
##  Median :0.008744   Median :0.3554   Median :1.9332   Median : 86.0
##  Mean   :0.011539   Mean   :0.3786   Mean   :2.0351   Mean   :113.5
##  3rd Qu.:0.012303   3rd Qu.:0.4495   3rd Qu.:2.3565   3rd Qu.:121.0
##  Max.   :0.074835   Max.   :0.6600   Max.   :3.9565   Max.   :736.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.006       0.25
```

```
inspect(rules[1:5])
```

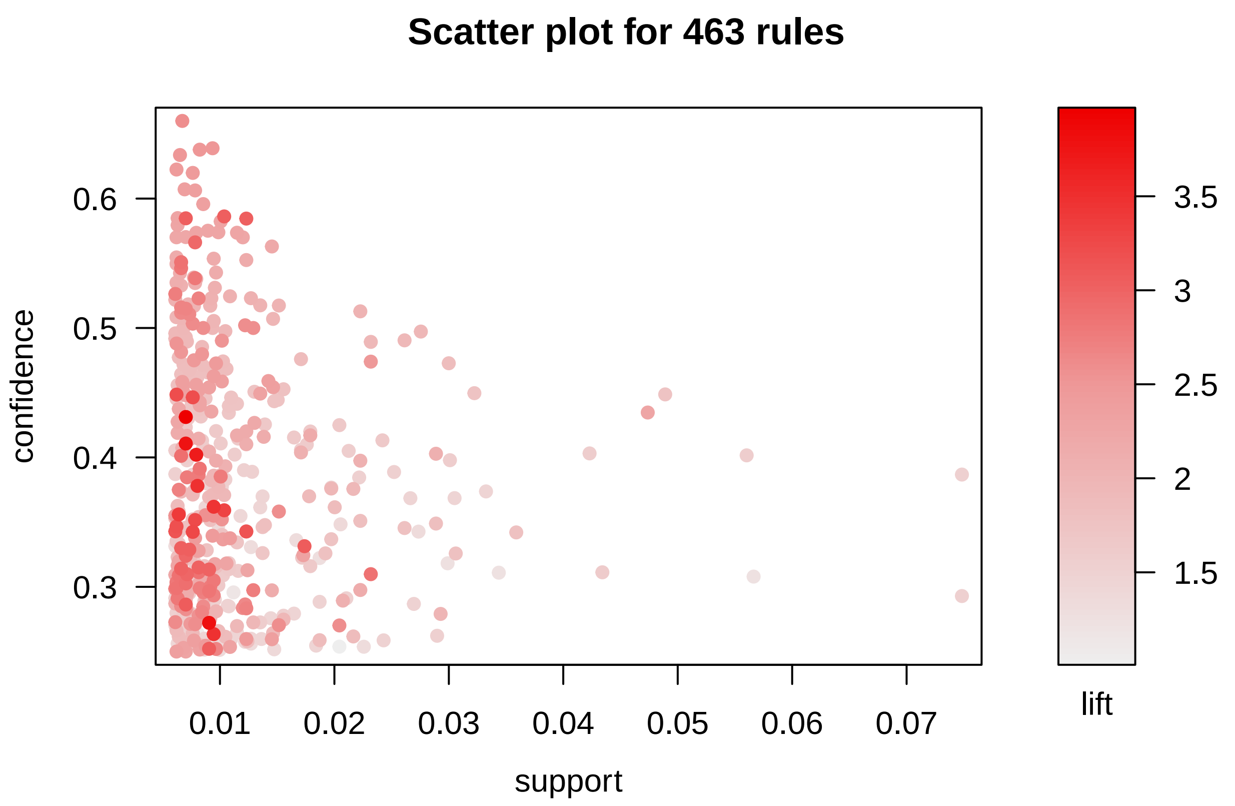
```
##     lhs              rhs                support     confidence lift     count
## [1] {pot plants} => {whole milk}       0.006914082 0.4000000  1.565460 68
## [2] {pasta}      => {whole milk}       0.006100661 0.4054054  1.586614 60
## [3] {herbs}      => {root vegetables}  0.007015760 0.4312500  3.956477 69
## [4] {herbs}      => {other vegetables} 0.007727504 0.4750000  2.454874 76
## [5] {herbs}      => {whole milk}       0.007727504 0.4750000  1.858983 76
```

```
library(arulesViz)
```

```
## Loading required package: grid
```

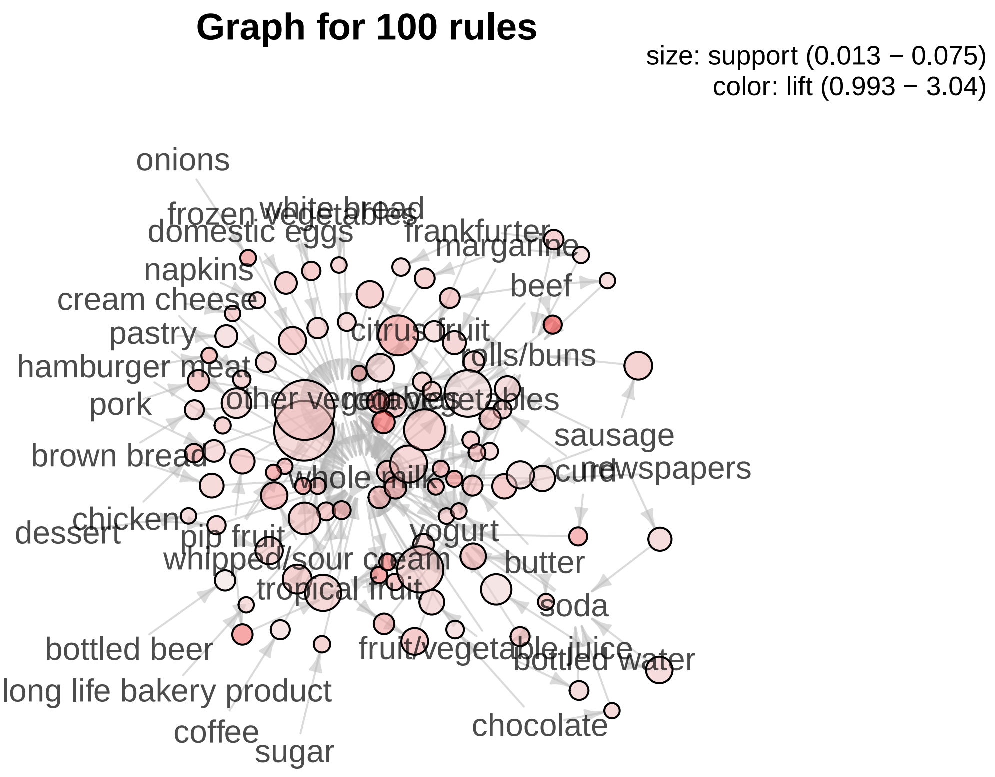
```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```
plot(rules, jitter = 0) # requires arulesViz
```



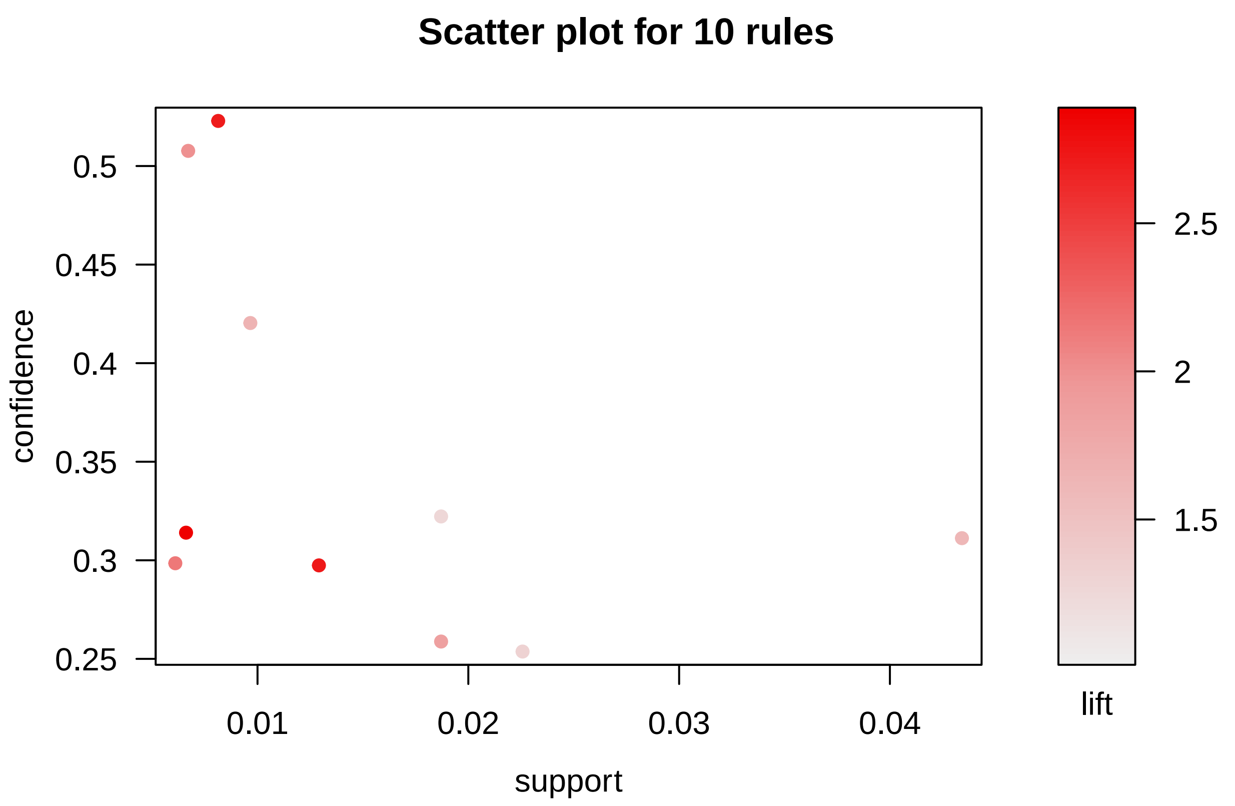
```r
plot(rules, method="graph", control=list(type="items"))
```

```
## Warning: Unknown control parameters: type

## Available control parameters (with default values):
## main  =  Graph for 100 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot  =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE

## Warning: plot: Too many rules supplied. Only plotting the best 100 rules using
## 'support' (change control parameter max if needed)
```



```r
sr <- sample(rules,10) #Selects 10 random rules
```

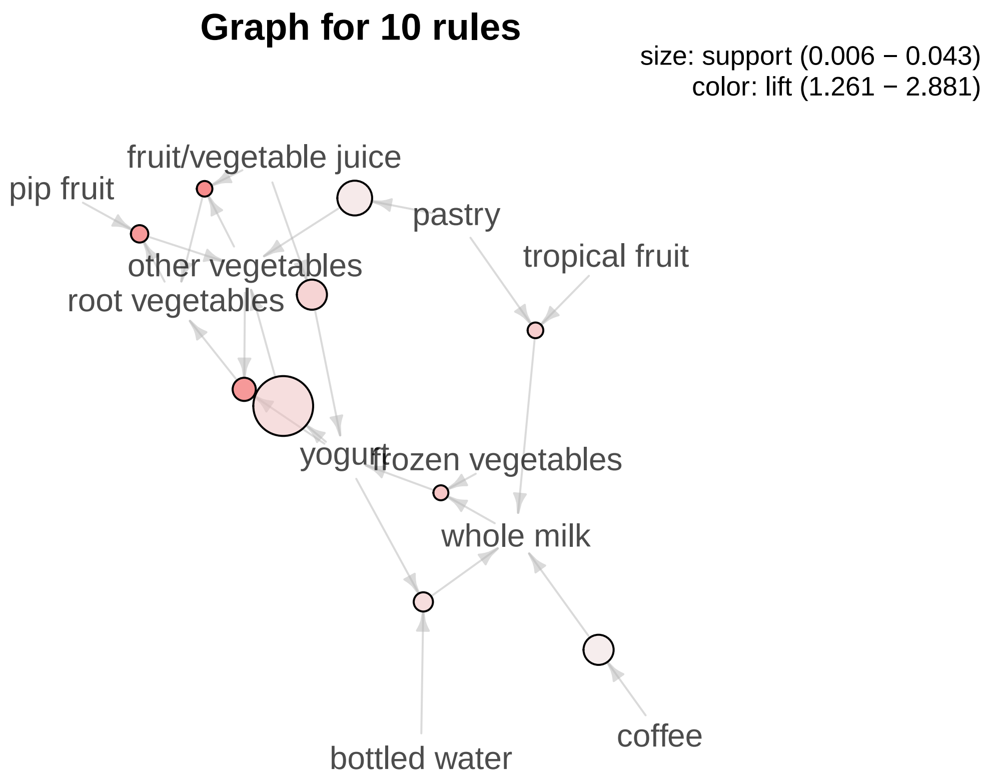
```r
plot(sr, jitter = 0) # requires arulesViz
```



```
plot(sr, method="graph", control=list(type="items"))
```

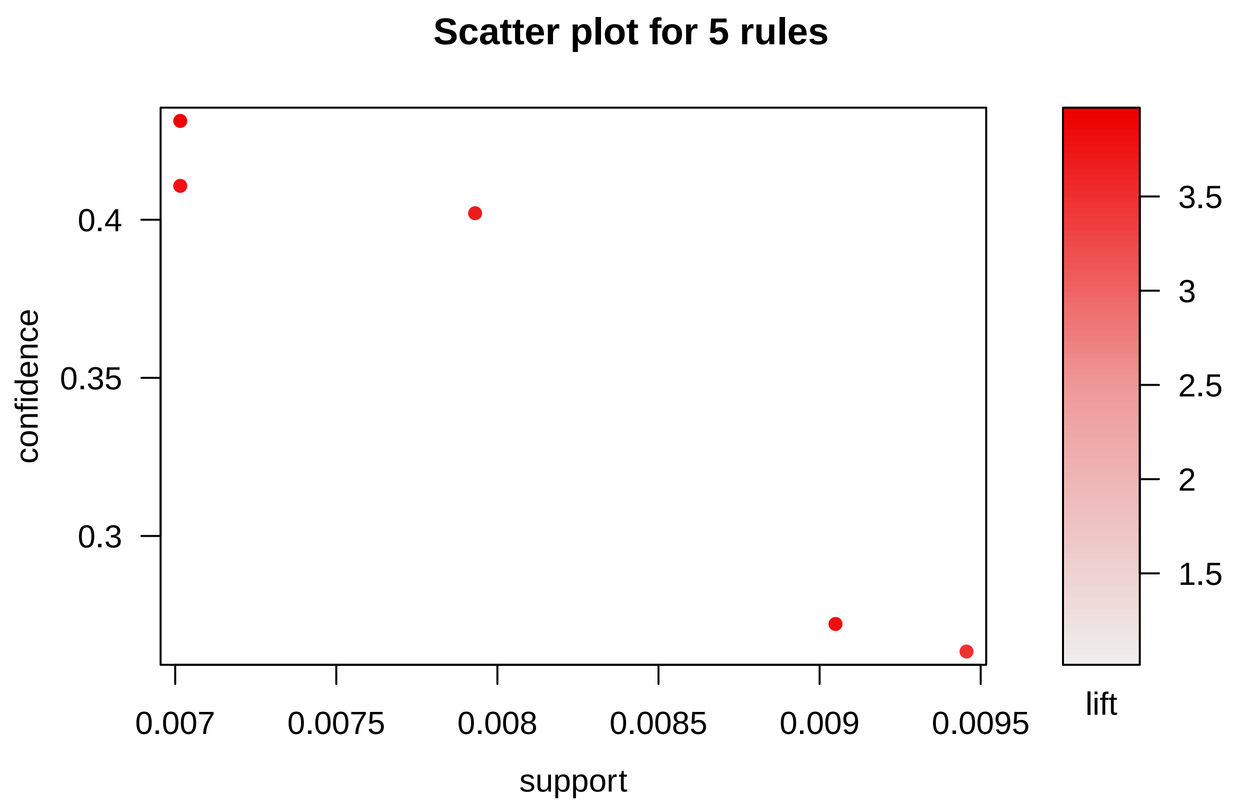
```
## Warning: Unknown control parameters: type
```

```
## Available control parameters (with default values):
## main  =  Graph for 10 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot  =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE
```

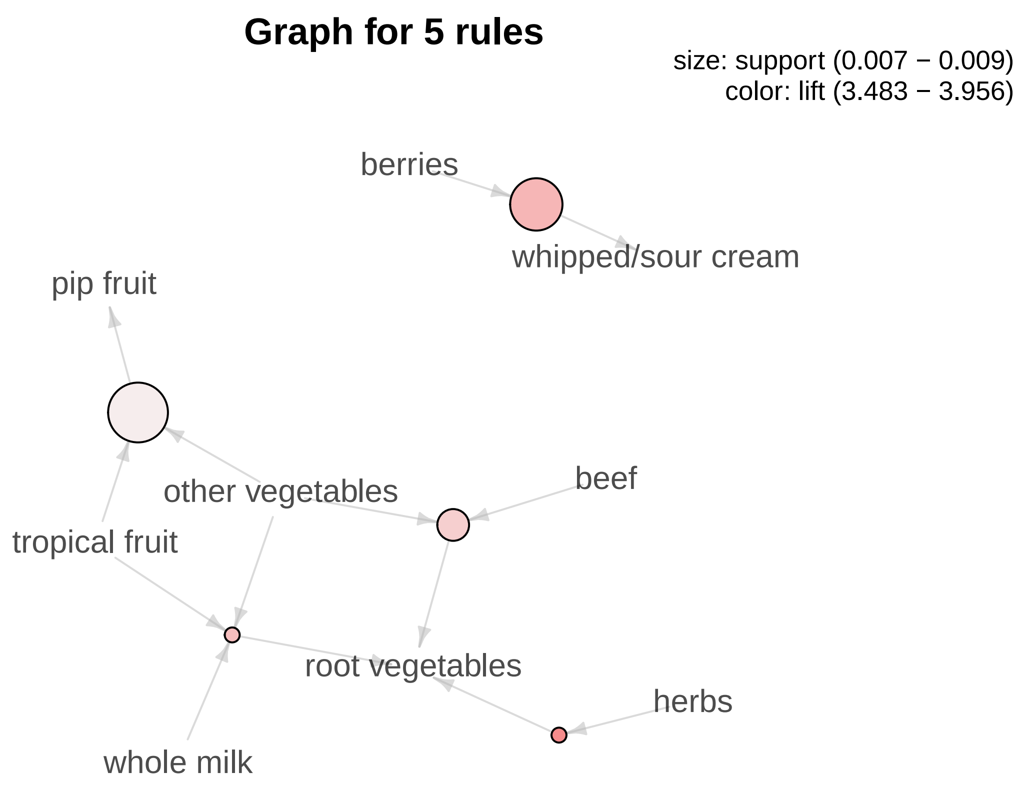


```r
sr1 <- sort(rules, by="lift")[1:5]
```

```r
plot(sr1, jitter = 0) # requires arulesViz
```



```
## Warning: Unknown control parameters: type

## Available control parameters (with default values):
## main   =  Graph for 5 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot   =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE
```



Get top five highest lift rules

```r
inspect(sort(rules, by="lift")[1:5])
```

```
##     lhs                      rhs                     support confidence     lift count
## [1] {herbs}               => {root vegetables}    0.007015760  0.4312500 3.956477    69
## [2] {berries}             => {whipped/sour cream} 0.009049314  0.2721713 3.796886    89
## [3] {tropical fruit,
##      other vegetables,
##      whole milk}          => {root vegetables}    0.007015760  0.4107143 3.768074    69
## [4] {beef,
##      other vegetables} => {root vegetables}    0.007930859  0.4020619 3.688692    78
## [5] {tropical fruit,
##      other vegetables} => {pip fruit}          0.009456024  0.2634561 3.482649    93
```

Find subset of the rules with "tropical fruit" appearing in the rule

```r
sub.rules <- subset(rules, rhs %in% "tropical fruit")
inspect(sub.rules)
```

```
##      lhs                          rhs                     support confidence      lift count
## [1]  {grapes}                 => {tropical fruit} 0.006100661  0.2727273 2.599101    60
## [2]  {pip fruit}              => {tropical fruit} 0.020437214  0.2701613 2.574648   201
## [3]  {other vegetables,
##       fruit/vegetable juice} => {tropical fruit} 0.006609049  0.3140097 2.992524    65
## [4]  {yogurt,
##       whipped/sour cream}    => {tropical fruit} 0.006202339  0.2990196 2.849668    61
## [5]  {other vegetables,
##       whipped/sour cream}    => {tropical fruit} 0.007829181  0.2711268 2.583849    77
## [6]  {pip fruit,
##       yogurt}                => {tropical fruit} 0.006405694  0.3559322 3.392048    63
## [7]  {pip fruit,
##       other vegetables}      => {tropical fruit} 0.009456024  0.3618677 3.448613    93
## [8]  {pip fruit,
##       whole milk}            => {tropical fruit} 0.008439248  0.2804054 2.672274    83
## [9]  {citrus fruit,
##       yogurt}                => {tropical fruit} 0.006304016  0.2910798 2.774002    62
## [10] {citrus fruit,
##       other vegetables}      => {tropical fruit} 0.009049314  0.3133803 2.986526    89
## [11] {citrus fruit,
##       whole milk}            => {tropical fruit} 0.009049314  0.2966667 2.827245    89
## [12] {yogurt,
##       bottled water}         => {tropical fruit} 0.007117438  0.3097345 2.951782    70
## [13] {other vegetables,
##       bottled water}         => {tropical fruit} 0.006202339  0.2500000 2.382510    61
## [14] {root vegetables,
##       yogurt}                => {tropical fruit} 0.008134215  0.3149606 3.001587    80
## [15] {root vegetables,
##       other vegetables}      => {tropical fruit} 0.012302999  0.2596567 2.474538   121
## [16] {yogurt,
##       rolls/buns}            => {tropical fruit} 0.008744281  0.2544379 2.424803    86
## [17] {other vegetables,
##       yogurt}                => {tropical fruit} 0.012302999  0.2833724 2.700550   121
## [18] {whole milk,
##       yogurt}                => {tropical fruit} 0.015149975  0.2704174 2.577089   149
## [19] {root vegetables,
##       other vegetables,
##       whole milk}            => {tropical fruit} 0.007015760  0.3026316 2.884091    69
## [20] {other vegetables,
##       whole milk,
##       yogurt}                => {tropical fruit} 0.007625826  0.3424658 3.263712    75
```

(c) Use apriori to generate rules

?apriori

lift(X->Y) = confidence(X->Y)/support(Y) = 0.25/0.083 = 3

```r
rules <- apriori(Groceries,
                 parameter = list(support = 0.006, confidence = 0.25, minlen = 2))
```

```
## Apriori
##
```

```
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.25    0.1    1 none FALSE            TRUE       5   0.006      2
##  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: 59
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [109 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [463 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules <- subset(rules, subset = lift > 3)
summary(rules)
```

```
## set of 25 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3  4
##  4 15  6
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    3.00    3.00    3.08    3.00    4.00
##
## summary of quality measures:
##     support           confidence          lift           count
##  Min.   :0.006101   Min.   :0.2521   Min.   :3.002   Min.   : 60.00
##  1st Qu.:0.007016   1st Qu.:0.3299   1st Qu.:3.040   1st Qu.: 69.00
##  Median :0.007829   Median :0.3516   Median :3.200   Median : 77.00
##  Mean   :0.008553   Mean   :0.3782   Mean   :3.277   Mean   : 84.12
##  3rd Qu.:0.009456   3rd Qu.:0.4107   3rd Qu.:3.449   3rd Qu.: 93.00
##  Max.   :0.017387   Max.   :0.5862   Max.   :3.956   Max.   :171.00
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.006       0.25
```

```
inspect(rules[1:5])
```

```
##     lhs                       rhs                        support confidence     lift count
## [1] {herbs}                => {root vegetables}    0.007015760  0.4312500 3.956477    69
## [2] {sliced cheese}        => {sausage}            0.007015760  0.2863071 3.047435    69
## [3] {berries}              => {whipped/sour cream} 0.009049314  0.2721713 3.796886    89
## [4] {beef}                 => {root vegetables}    0.017386884  0.3313953 3.040367   171
## [5] {other vegetables,
##      frozen vegetables} => {root vegetables}    0.006100661  0.3428571 3.145522    60
```

Get top five highest lift rules

```r
inspect(sort(rules, by="lift")[1:5])
```

```
##     lhs                     rhs                      support confidence     lift count
## [1] {herbs}              => {root vegetables}    0.007015760  0.4312500 3.956477    69
## [2] {berries}            => {whipped/sour cream} 0.009049314  0.2721713 3.796886    89
## [3] {tropical fruit,
##      other vegetables,
##      whole milk}         => {root vegetables}    0.007015760  0.4107143 3.768074    69
## [4] {beef,
##      other vegetables} => {root vegetables}    0.007930859  0.4020619 3.688692    78
## [5] {tropical fruit,
##      other vegetables} => {pip fruit}          0.009456024  0.2634561 3.482649    93
```

Find subset of the rules with "berries or yogurt" appearing in the rule

```r
sub.rules <- subset(rules, items %in% c("berries", "yogurt"))
inspect(sub.rules)
```

```
##     lhs                       rhs                      support confidence     lift count
## [1] {berries}              => {whipped/sour cream} 0.009049314  0.2721713 3.796886    89
## [2] {tropical fruit,
##      whipped/sour cream} => {yogurt}             0.006202339  0.4485294 3.215224    61
## [3] {pip fruit,
##      yogurt}               => {tropical fruit}     0.006405694  0.3559322 3.392048    63
## [4] {root vegetables,
##      yogurt}               => {tropical fruit}     0.008134215  0.3149606 3.001587    80
## [5] {tropical fruit,
##      other vegetables,
##      whole milk}           => {yogurt}             0.007625826  0.4464286 3.200164    75
## [6] {other vegetables,
##      whole milk,
##      yogurt}               => {tropical fruit}     0.007625826  0.3424658 3.263712    75
## [7] {other vegetables,
##      whole milk,
##      yogurt}               => {root vegetables}    0.007829181  0.3515982 3.225716    77
```