Asociation Rules

association rules with R Groceries data

```
library(arules)
```

```
## Warning: package 'arules' was built under R version 4.0.2
```

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

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

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

```
library(arulesViz)
```

```
## Warning: package 'arulesViz' was built under R version 4.0.2
```

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

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

```
library(datasets)
data("Groceries")
# summary statistics
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
```

```
inspect(head(Groceries,10))
```

```
##      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}
## [6]  {whole milk,
##       butter,
##       yogurt,
##       rice,
##       abrasive cleaner}
## [7]  {rolls/buns}
## [8]  {other vegetables,
##       UHT-milk,
##       rolls/buns,
##       bottled beer,
##       liquor (appetizer)}
## [9]  {pot plants}
## [10] {whole milk,
##       cereals}
```

```
str(Groceries)
```

```
## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:43367] 13 60 69 78 14 29 98 24 15 29 ...
##   .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
##   .. .. ..@ Dim     : int [1:2] 169 9835
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : NULL
##   .. .. ..@ factors : list()
##   ..@ itemInfo   :'data.frame':  169 obs. of  3 variables:
##   .. ..$ labels: chr [1:169] "frankfurter" "sausage" "liver loaf" "ham" ...
##   .. ..$ level2: Factor w/ 55 levels "baby food","bags",..: 44 44 44 44 44 44 44 42 42 41 ...
##   .. ..$ level1: Factor w/ 10 levels "canned food",..: 6 6 6 6 6 6 6 6 6 6 ...
##   ..@ itemsetInfo:'data.frame':  0 obs. of  0 variables
```

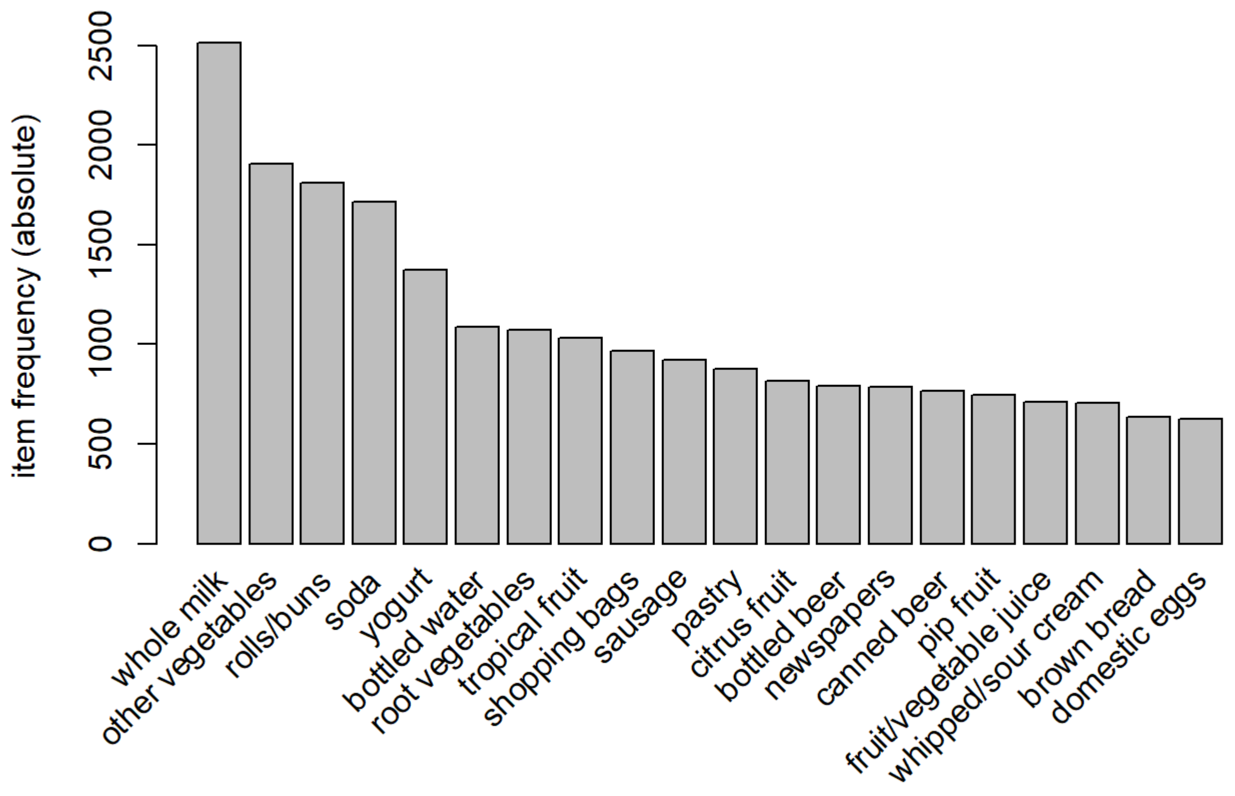
```
head(Groceries)
```

```
## transactions in sparse format with
##  6 transactions (rows) and
##  169 items (columns)
```

```
#rules <- apriori(Groceries,parameter=list(supp = 0.001, conf=0.8))
```

plot the first 20 ‘count’ of each grocery item appearing in the dataset

```
itemFrequencyPlot(Groceries,topN=20,type="absolute")
```



RETRIEVAL OF ASSOCIATION RULES

Use ‘apriori’ to generate association rules. Output to ‘rules’, which is a data frame.

```
rules <- apriori(Groceries,parameter=list(supp = 0.001, conf=0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5   0.001      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: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.01s].
## writing ... [5668 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
#specify to two decimal places for any numeric output
options(digits=2)

# summarize the set of rules which tells the number of rules generated by length (number of item
s),amongst other things
inspect(rules[1:5])
```

```
##     lhs                    rhs          support confidence coverage lift count
## [1] {honey}             => {whole milk} 0.0011  0.73       0.0015   2.9  11
## [2] {tidbits}           => {rolls/buns} 0.0012  0.52       0.0023   2.8  12
## [3] {cocoa drinks}      => {whole milk} 0.0013  0.59       0.0022   2.3  13
## [4] {pudding powder}    => {whole milk} 0.0013  0.57       0.0023   2.2  13
## [5] {cooking chocolate} => {whole milk} 0.0013  0.52       0.0025   2.0  13
```

```
summary(rules)
```

```
## set of 5668 rules
##
## rule length distribution (lhs + rhs):sizes
##    2    3    4    5    6
##   11 1461 3211  939   46
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     3.0     4.0     3.9     4.0     6.0
##
## summary of quality measures:
##     support         confidence      coverage          lift          count
##  Min.   :0.0010   Min.   :0.50   Min.   :0.001   Min.   : 2.0   Min.   : 10
##  1st Qu.:0.0011   1st Qu.:0.55   1st Qu.:0.002   1st Qu.: 2.5   1st Qu.: 11
##  Median :0.0013   Median :0.60   Median :0.002   Median : 2.9   Median : 13
##  Mean   :0.0017   Mean   :0.62   Mean   :0.003   Mean   : 3.3   Mean   : 16
##  3rd Qu.:0.0017   3rd Qu.:0.68   3rd Qu.:0.003   3rd Qu.: 3.7   3rd Qu.: 17
##  Max.   :0.0223   Max.   :1.00   Max.   :0.043   Max.   :19.0   Max.   :219
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.001        0.5
```

top 3 rules along with their measures of support, confidence and lift.

```
inspect(head(sort(rules, by ="support"),3))
```

```
##     lhs                                       rhs          support confidence
## [1] {other vegetables,yogurt}             => {whole milk} 0.022   0.51
## [2] {tropical fruit,yogurt}               => {whole milk} 0.015   0.52
## [3] {other vegetables,whipped/sour cream} => {whole milk} 0.015   0.51
##     coverage lift count
## [1] 0.043    2    219
## [2] 0.029    2    149
## [3] 0.029    2    144
```

```
inspect(head(sort(rules, by ="confidence"),3))
```

```
##     lhs                                rhs          support confidence coverage
## [1] {rice,sugar}                    => {whole milk} 0.0012  1          0.0012
## [2] {canned fish,hygiene articles} => {whole milk} 0.0011  1          0.0011
## [3] {root vegetables,butter,rice}  => {whole milk} 0.0010  1          0.0010
##     lift count
## [1] 3.9  12
## [2] 3.9  11
## [3] 3.9  10
```

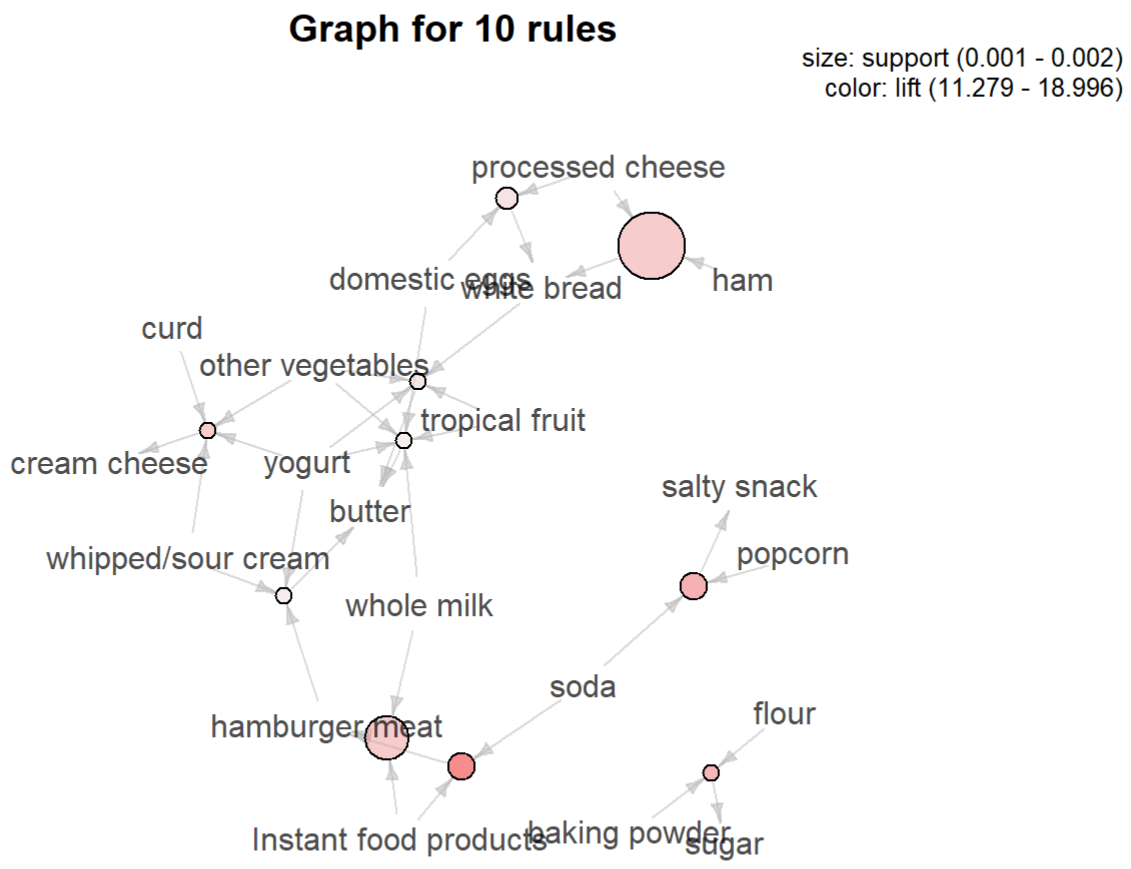
```
inspect(head(sort(rules, by ="lift"),3))
```

```
##     lhs                              rhs              support confidence
## [1] {Instant food products,soda}  => {hamburger meat} 0.0012  0.63
## [2] {soda,popcorn}                => {salty snack}    0.0012  0.63
## [3] {flour,baking powder}         => {sugar}          0.0010  0.56
##     coverage lift count
## [1] 0.0019   19   12
## [2] 0.0019   17   12
## [3] 0.0018   16   10
```

```
#generate rules where we fix some values "whole milk"
#inspect(subset(rules, subset = lhs %pin% "whole milk"))
```

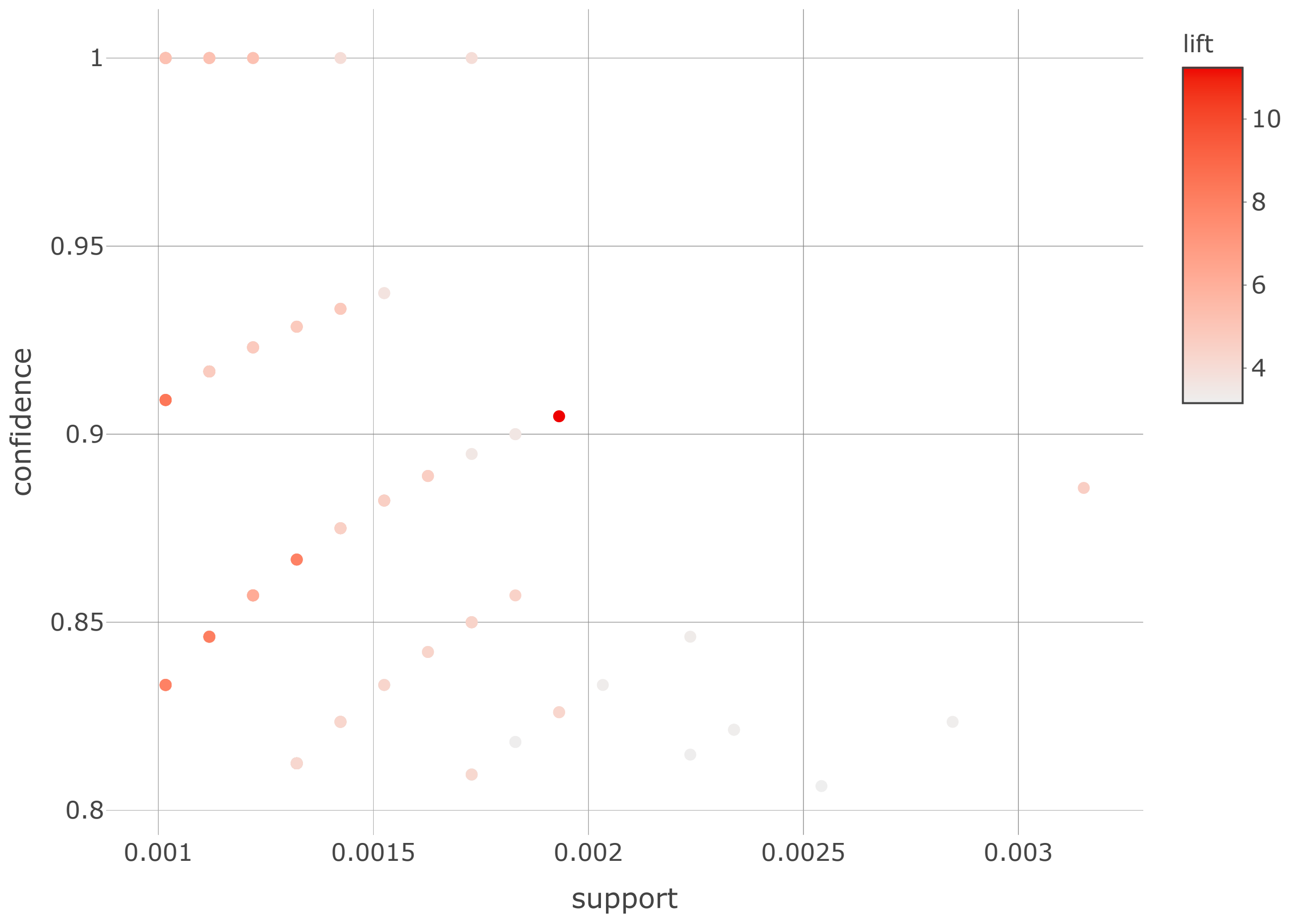
VISUALIZATION

```
#extract subsets
subrules1 <-rules[quality(rules)$confidence > 0.8]
subrules2<-head(sort(rules,by="lift"),10)
# plotting
plot(subrules2,method="graph")
```



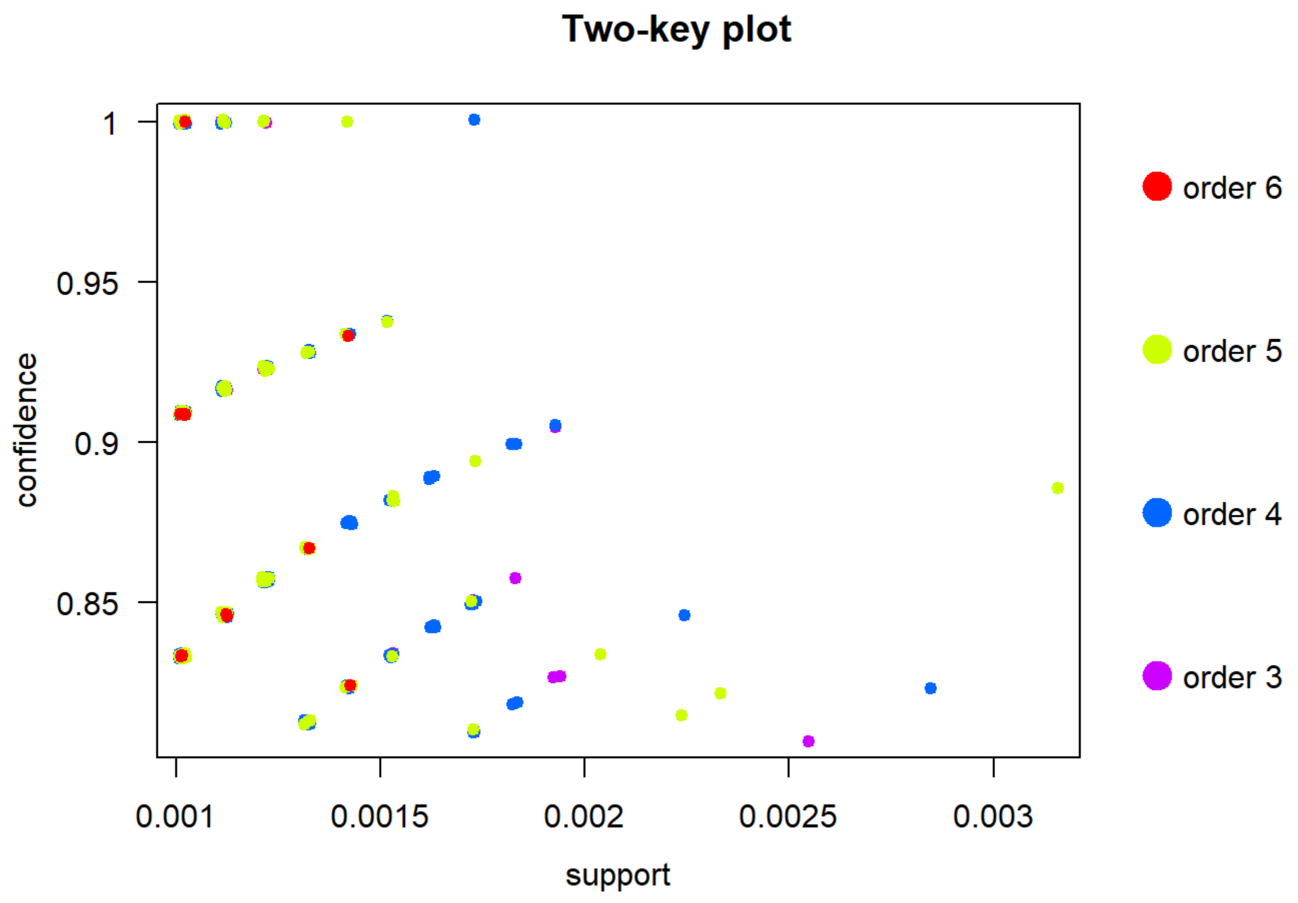
```
plot(subrules1,jitter = 0,engine = "plotly")
```

```
## Warning: `arrange_()` is deprecated as of dplyr 0.7.0.
## Please use `arrange()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```



```
plot(subrules1,method="two-key plot")
```

```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```


Two-key plot
confidence
support
1
0.95
0.9
0.85
0.001
0.0015
0.002
0.0025
0.003
order 6
order 5
order 4
order 3