Association_Analysis.R

Paola

2019-05-18

```
#### Association Analysis ####

setwd("D:/R_2")
library(magrittr)

set.seed(270)
faults <- data.frame(
  serialNumber = sample(1:20, 100, replace = T),
  faultCode = paste("fc", sample(1:12, 100, replace = T),
                    sep = "")
)
faults <- unique(faults)

# table(faults$serialNumber)
# table(faults$faultCode)

str(faults)
```

```
## 'data.frame':    80 obs. of  2 variables:
##  $ serialNumber: int  9 8 1 18 11 20 2 16 10 20 ...
##  $ faultCode   : Factor w/ 12 levels "fc1","fc10","fc11",..: 2 5 1 12 1 3 6 10 11 1 ...
```

```
# Create a column where all values are true
faults$indicator = TRUE
# Reshape the data into the wide format
faults_wide <- tidyr::spread(faults, key = faultCode,
                             value=indicator)
#View(faults_wide)

# Turn the data into a matrix while dropping the ID
faults_matrix <- as.matrix(faults_wide[, -1])
#View(faults_matrix)

# Turn NA values into something understood such as FALSE
faults_matrix[is.na(faults_matrix)] <- FALSE
#View(faults_matrix)

# Turn this data into transactions class
library(Matrix)
#install.packages("arules")
library(arules)
```

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

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

```
##
##     abbreviate, write
```

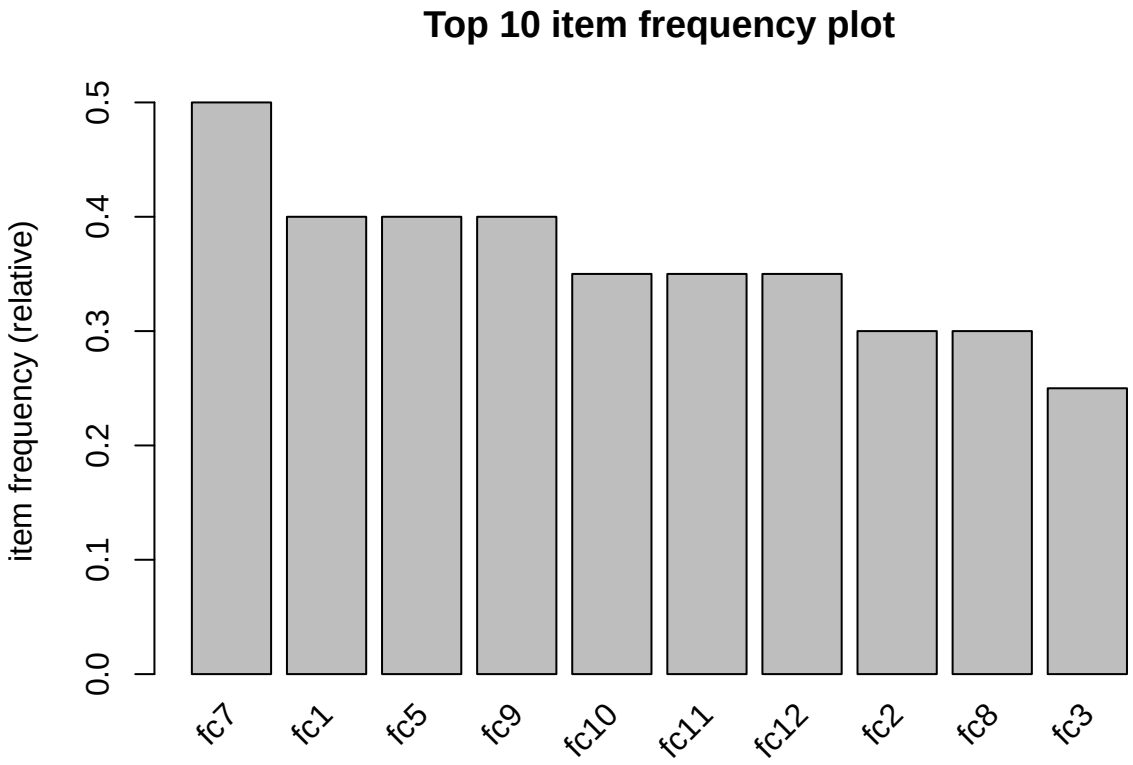
```
faults_transactions <- as(faults_matrix, "transactions")
faults_transactions
```

```
## transactions in sparse format with
##  20 transactions (rows) and
##  12 items (columns)
```

```
summary(faults_transactions)
```

```
## transactions as itemMatrix in sparse format with
##  20 rows (elements/itemsets/transactions) and
##  12 columns (items) and a density of 0.3333333
##
## most frequent items:
##     fc7     fc1     fc5     fc9    fc10 (Other)
##      10       8       8       8       7      39
##
## element (itemset/transaction) length distribution:
## sizes
## 1 2 3 4 6 7 8
## 1 4 5 4 3 2 1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    2.75    3.50    4.00    6.00    8.00
##
## includes extended item information - examples:
##   labels
## 1    fc1
## 2   fc10
## 3   fc11
```

```
# Create a plot of the top 10 item frequency
arules::itemFrequencyPlot(faults_transactions, topN = 10,
                          main="Top 10 item frequency plot")
```

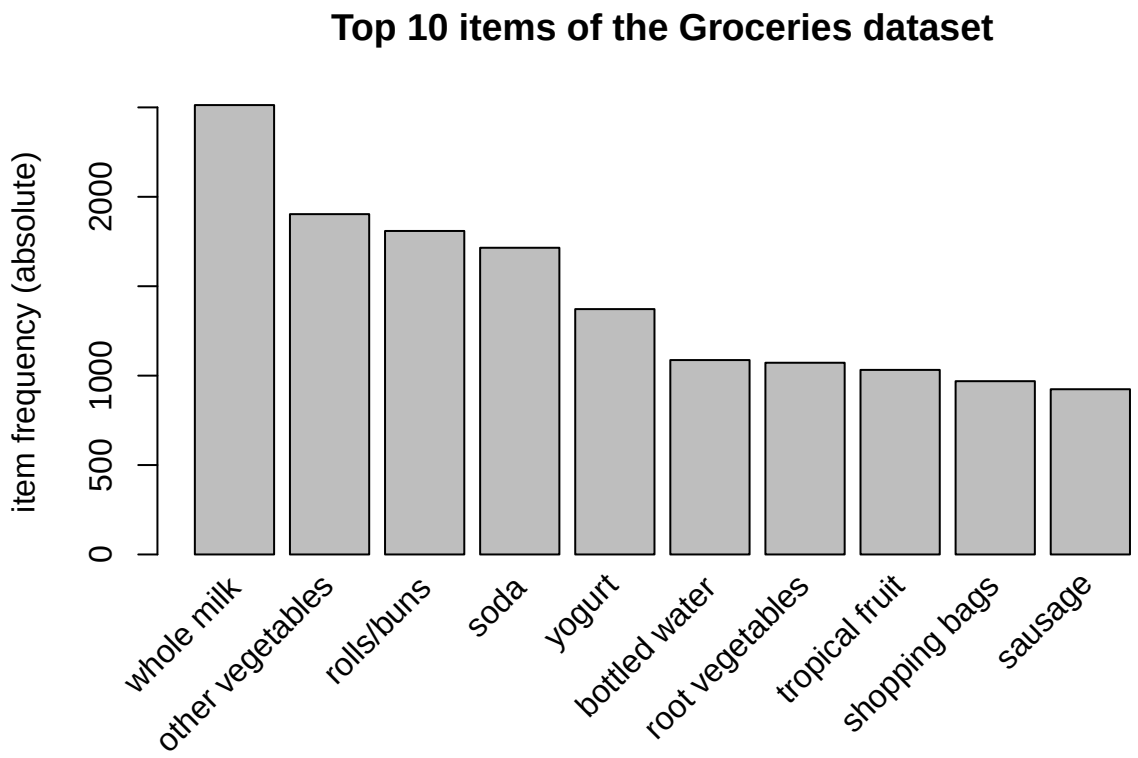



```
#install.packages("arulesViz")

# Load the in-built dataset from "arules" package
data(Groceries)
str(Groceries)
```

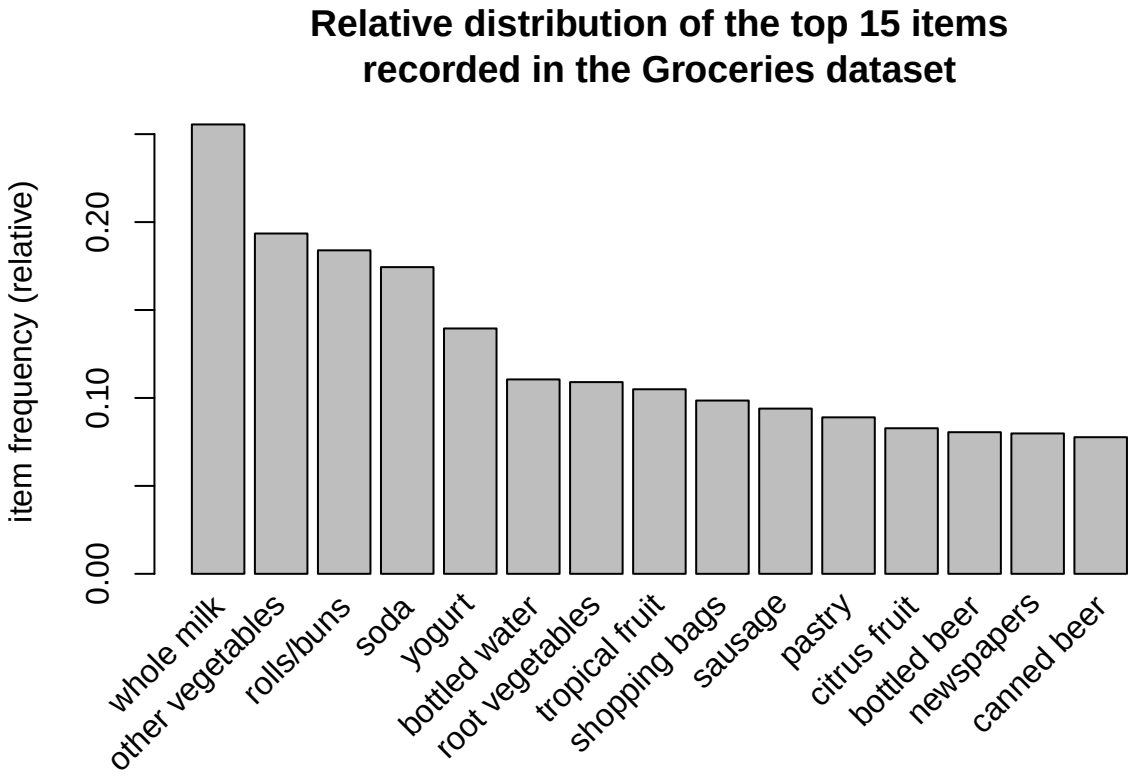
```
## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:43367] 13 60 69 78 14 29 98 24 15 29 ...
##   .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
##   .. .. ..@ 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
```

```
# This dataset is structured as a sparse matrix object,
# known as the trasaction class
# Since the structure is that of the transaction class,
# we cannot explore the data with standard technique, and
```

```r
# so we use those of "arules" package

# Use the item frequency plot
arules::itemFrequencyPlot(Groceries, topN = 10,
                          type = "absolute",
                          main = "Top 10 items of the Groceries dataset")
```




```r
# ?Groceries
# So we have 9835 transactions per 169 categories of products
# By looking at the previous plot the top item purchsed
# is whole milk with roughly 2500 of the 9835 transactions
# in the basket.

# Plot the relative distribution of the top 15 items
arules::itemFrequencyPlot(Groceries, topN = 15,
                          main = "Relative distribution of the top 15 items\nrecorded in the Groceries
```




```
# Note that Beer shows up as the 13th and 15th most
# purchased item at this store

#### Mining the data for the overall association rules #####

# Set the rules by establishing:
# - the minimum support at 1 in 1,000 transactions
# - minimum confidence at 90%
rules <-
  arules::apriori(Groceries, parameter = list(
    supp = 0.001,
    conf =
      0.9,
    maxlen = 4
  ))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.9    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target   ext
##       4  rules FALSE
##
## 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
```

```
## Warning in arules::apriori(Groceries, parameter = list(supp = 0.001, conf
## = 0.9, : Mining stopped (maxlen reached). Only patterns up to a length of 4
## returned!
```

```
##  done [0.01s].
## writing ... [67 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
# either change maxtime (add to the parameter list same
# as minlen: maxtime=10 or maxtime=20 etc)
# or as in most cases - ignore the warning.
# This is not an error. Is it really important for you
# to find rules longer than 4 items? I think not.
# You did not specify this it is only a default value
# Anyway, I keep it.

rules
```

```
## set of 67 rules
```

```
# Examine the rules
options(digits = 2)
rules <- arules::sort(rules, by = "lift",
                      decreasing = TRUE)
arules::inspect(rules[1:5])
```

```
##     lhs                     rhs                support confidence lift count
## [1] {liquor,
##      red/blush wine}     => {bottled beer}      0.0019       0.90 11.2    19
## [2] {root vegetables,
##      butter,
##      cream cheese }      => {yogurt}            0.0010       0.91  6.5    10
## [3] {citrus fruit,
##      root vegetables,
##      soft cheese}        => {other vegetables}  0.0010       1.00  5.2    10
## [4] {pip fruit,
##      whipped/sour cream,
##      brown bread}        => {other vegetables}  0.0011       1.00  5.2    11
## [5] {butter,
##      whipped/sour cream,
##      soda}               => {other vegetables}  0.0013       0.93  4.8    13
```

```
# we can see that the rule that offers the best overall
# lift is the purchase of liquor and the red wine on the
# probabilities of purchasing bottled beer.
# Although it's still not a very common transaction with
# support of only 1.9 per 1,000.

# Sort by confidence
rules <- arules::sort(rules, by = "confidence",
                      decreasing = TRUE)
arules::inspect(rules[1:5])
```

```
##     lhs                     rhs                support confidence lift count
## [1] {citrus fruit,
##      root vegetables,
##      soft cheese}        => {other vegetables}  0.0010          1  5.2    10
## [2] {pip fruit,
##      whipped/sour cream,
##      brown bread}        => {other vegetables}  0.0011          1  5.2    11
## [3] {rice,
##      sugar}              => {whole milk}        0.0012          1  3.9    12
## [4] {canned fish,
##      hygiene articles}   => {whole milk}        0.0011          1  3.9    11
## [5] {root vegetables,
##      butter,
##      rice}               => {whole milk}        0.0010          1  3.9    10
```

```
# we can see that the confidence for these transactions is
# 100%.

# Analyse rules using crossTable which allows to
# interrogate partially the products per transactions.
tab <- arules::crossTable(Groceries)

# Investigate the first 3 rows and columns
tab[1:3, 1:3]
```

```
##             frankfurter sausage liver loaf
## frankfurter         580      99          7
## sausage              99     924         10
## liver loaf            7      10         50
```

```
# Liver loaf is selected only 50 time out of the 9835 transactions.

# See specifically one genre of product such as
# bottled beer
tab["bottled beer", "bottled beer"]
```

```
## [1] 792
```

```
# There were 792 transactions of bottled beer.

# See what the joint occurrence between bottled beer and
```

```r
# canned beer is:
tab["bottled beer", "canned beer"]
```

```
## [1] 26
```

```r
#### Mining for a specific association rule ####
# Derive specific rules for bottled beer
beer.rules <- arules::apriori(
  data = Groceries,
  parameter = list(support = 0.0015,
                   confidence = 0.3),
  appearance = list(default = "lhs",# lhs stands for left hand side
                    rhs = "bottled beer")#specify that bottled beer is on the right hand side
)
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.3    0.1    1 none FALSE            TRUE       5  0.0015      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: 14
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [153 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 ... [4 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
beer.rules
```

```
## set of 4 rules
```

```r
# we found ourself 4 association rules

beer.rules <- arules::sort(beer.rules,
                           decreasing = TRUE, by = "lift")
arules::inspect(beer.rules)
```

```
##     lhs                               rhs             support confidence
## [1] {liquor,red/blush wine}        => {bottled beer} 0.0019  0.90
## [2] {liquor}                       => {bottled beer} 0.0047  0.42
## [3] {soda,red/blush wine}          => {bottled beer} 0.0016  0.36
```

```
## [4] {other vegetables,red/blush wine} => {bottled beer} 0.0015  0.31
##     lift count
## [1] 11.2 19
## [2]  5.2 46
## [3]  4.4 16
## [4]  3.8 15
```

```
# In all of the instances the bottled beer is associated
# with booze.
# What is interesting is that white wine isn't in the mix
# here.

# Compare the joint occurrences of bottled beer
# and types of wine
tab["bottled beer", "red/blush wine"]# 48
```

```
## [1] 48
```

```
tab["red/blush wine", "red/blush wine"]# 189
```

```
## [1] 189
```

```
48/189
```

```
## [1] 0.25
```

```
tab["white wine", "white wine"]# 187
```

```
## [1] 187
```

```
tab["bottled beer", "white wine"]# 22
```

```
## [1] 22
```

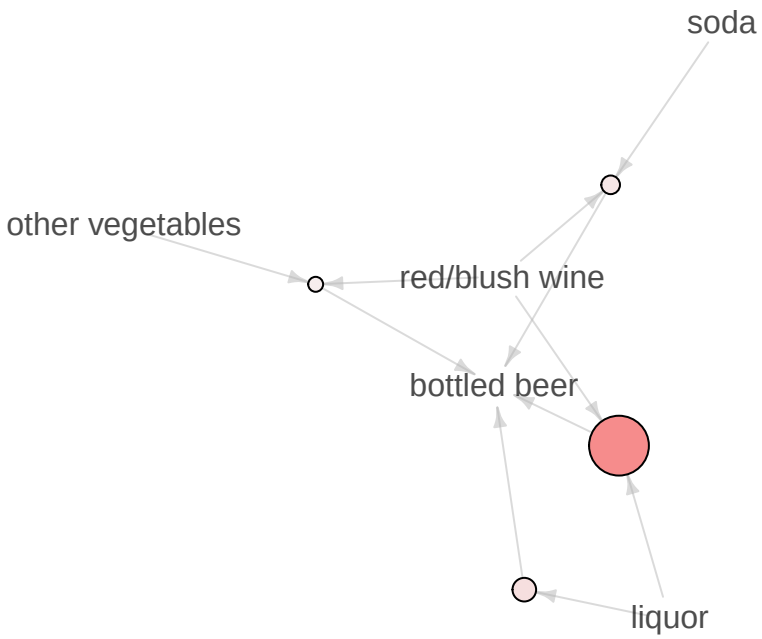
```
22/187
```

```
## [1] 0.12
```

```
# 25% of the time when someone purchased red wine,
# the also purchased bottled beer.
# Instead, with white wine, a joint purchased only
# happened in the 12% of the instances.
# This information could be useful to determine how we
# should position our product in this grocery store.

# Plot the rules with "arulesViz" package
library(arulesViz)
```

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

```
plot(beer.rules,
     method = "graph",
     measure = "lift",
     shading = "confidence")
```




```
# The graph shows that liquor and red wine provide
# the best lift and the highest level of confidence
# with both the size of the circle and its shading.
```