# Association Analysis

The

```
# Loading the arules library
#
library(arules)
```

```
## Warning: package 'arules' was built under R version 3.6.3
```

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

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

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

```
#Loading our dataset

sale <- read.transactions("http://bit.ly/SupermarketDatasetII")
```

```
## Warning in asMethod(object): removing duplicated items in transactions
```

```
head(sale)
```

```
## transactions in sparse format with
##  6 transactions (rows) and
##  5729 items (columns)
```

```
# Verifying the object's class
# ---
# This should show us transactions as the type of data that we will need
# ---
#
class(sale)
```

```
## [1] "transactions"
## attr(,"package")
## [1] "arules"
```

```
# Previewing our first 5 sales
#
inspect(sale[1:5])
```

```
##     items
## [1] {cheese,energy,
##      drink,tomato,
##      fat,
##      flour,yams,cottage,
##      grapes,whole,
##      juice,frozen,
##      juice,low,
##      mix,green,
##      oil,
##      shrimp,almonds,avocado,vegetables,
##      smoothie,spinach,olive,
##      tea,honey,salad,mineral,
##      water,salmon,antioxydant,
##      weat,
##      yogurt,green}
## [2] {burgers,meatballs,eggs}
## [3] {chutney}
## [4] {turkey,avocado}
## [5] {bar,whole,
##      mineral,
##      rice,green,
##      tea,
##      water,milk,energy,
##      wheat}
```

```
items<-as.data.frame(itemLabels(sale))
colnames(items) <- "Item"
head(items, 10)
```

```
##                                       Item
## 1                                        &
## 2                              accessories
## 3                  accessories,antioxydant
## 4              accessories,champagne,fresh
## 5            accessories,champagne,protein
## 6                    accessories,chocolate
## 7  accessories,chocolate,champagne,frozen
## 8             accessories,chocolate,frozen
## 9                accessories,chocolate,low
## 10       accessories,chocolate,pasta,salt
```

```
# Generating a summary of the transaction dataset
# ---
# This would give us some information such as the most purchased items,
# distribution of the item sets (no. of items purchased in each transaction), etc.
# ---
#
summary(sale)
```

```
## transactions as itemMatrix in sparse format with
##  7501 rows (elements/itemsets/transactions) and
##  5729 columns (items) and a density of 0.0005421748
##
## most frequent items:
##     tea   wheat mineral     fat  yogurt (Other)
##     803     645     577     574     543   20157
```

```
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   15   16
## 1603 2007 1382  942  651  407  228  151   70   39   13    5    1    1    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.106   4.000  16.000
##
## includes extended item information - examples:
##                    labels
## 1                       &
## 2             accessories
## 3 accessories,antioxydant
```

```
# Exploring the frequency of some articles
# i.e. transacations ranging from 8 to 10 and performing
# some operation in percentage terms of the total transactions
#
itemFrequency(sale[, 8:10],type = "absolute")
```

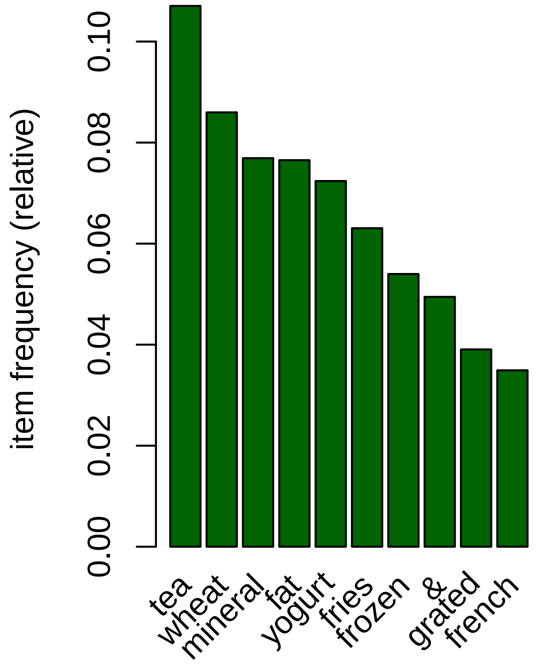
```
##     accessories,chocolate,frozen        accessories,chocolate,low
##                                1                                1
## accessories,chocolate,pasta,salt
##                                1
```

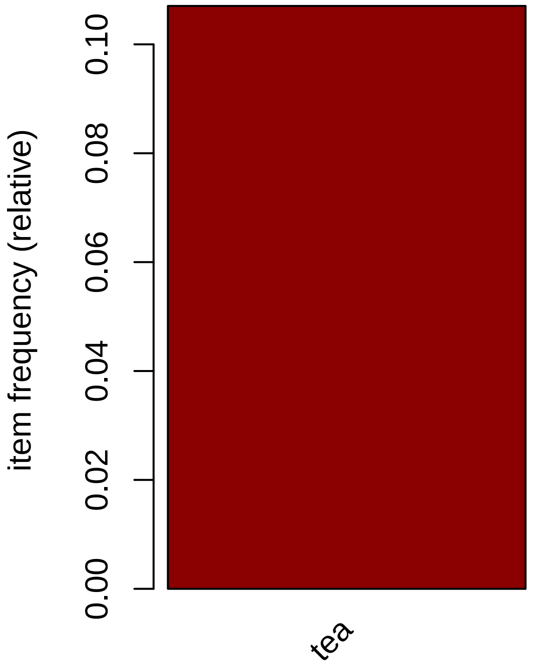
```
round(itemFrequency(sale[, 8:10],type = "relative")*100,2)
```

```
##     accessories,chocolate,frozen        accessories,chocolate,low
##                             0.01                             0.01
## accessories,chocolate,pasta,salt
##                             0.01
```

```
# Producing a chart of frequencies and fitering
# to consider only items with a minimum percentage
# of support/ considering a top x of items
# ---
# Displaying top 10 most common items in the sale dataset
# and the items whose relative importance is at least 10%
#
par(mfrow = c(1, 2))

# plot the frequency of items
itemFrequencyPlot(sale, topN = 10,col="darkgreen")
itemFrequencyPlot(sale, support = 0.1,col="darkred")
```





```r
# Building a model based on association rules
# using the apriori function
# ---
# We use Min Support as 0.001 and confidence as 0.8
# ---
#
rules <- apriori (sale, parameter = list(supp = 0.001, conf = 0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    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: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5729 item(s), 7501 transaction(s)] done [0.01s].
## sorting and recoding items ... [354 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [271 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules
```

```
## set of 271 rules
```

```
# We use measures of significance and interest on the rules,
# determining which ones are interesting and which to discard.
# ---
# However since we built the model using 0.001 Min support
# and confidence as 0.8 we obtained 410 rules.
# However, in order to illustrate the sensitivity of the model to these two parameters,
# we will see what happens if we increase the support or lower the confidence level
#

# Building a apriori model with Min Support as 0.002 and confidence as 0.8.
rules2 <- apriori (sale,parameter = list(supp = 0.002, conf = 0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5   0.002      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: 15
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5729 item(s), 7501 transaction(s)] done [0.01s].
## sorting and recoding items ... [189 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [99 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
# Building apriori model with Min Support as 0.002 and confidence as 0.6.
rules3 <- apriori (sale, parameter = list(supp = 0.001, conf = 0.6))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    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: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5729 item(s), 7501 transaction(s)] done [0.01s].
## sorting and recoding items ... [354 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [319 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules2
```

```
## set of 99 rules
```

```
rules3
```

```
## set of 319 rules
```