Part 3: Association Rules

This section will require that you create association rules that will allow you to identify relationships between variables in the dataset. You are provided with a separate dataset that comprises groups of items that will be associated with others. Just like in the other sections, you will also be required to provide insights for your analysis.

```
# Loading Libraries
library(data.table)
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse
1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## Warning: package 'ggplot2' was built under R version 4.0.5

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::transpose() masks data.table::transpose()

library(dplyr)
library(tibble)
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa

library(caret)

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift

library(arules)
```

```
## Warning: package 'arules' was built under R version 4.0.5

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

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

library(relaimpo)

## Warning: package 'relaimpo' was built under R version 4.0.5

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: boot

##
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':
##
##     melanoma

## Loading required package: survey

## Warning: package 'survey' was built under R version 4.0.5

## Loading required package: grid

## Loading required package: survival

##
## Attaching package: 'survival'
## The following object is masked from 'package:boot':
##
##     aml
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
##
## Attaching package: 'survey'
```

```
## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
## Loading required package: mitools
```

```
## Warning: package 'mitools' was built under R version 4.0.5
```

```
## This is the global version of package relaimpo.
```

```
## If you are a non-US user, a version with the interesting additional metric
pmvd is available
```

```
## from Ulrike Groempings web site at prof.beuth-hochschule.de/groemping.
```

```
# Loading the dataset
smk <-read.csv('http://bit.ly/SupermarketDatasetII', header= TRUE)
# Loading the dataset as a dataframe
data = as.data.frame(smk)
head(df)
```

```
##
## 1 function (x, df1, df2, ncp, log = FALSE)
## 2 {
## 3     if (missing(ncp))
## 4         .Call(C_df, x, df1, df2, log)
## 5     else .Call(C_dnf, x, df1, df2, ncp, log)
## 6 }
```

```
# Previewing the first five rows of the dataframe
head(data)
```

```
##               shrimp      almonds    avocado   vegetables.mix green.grapes
## 1            burgers    meatballs       eggs
## 2            chutney
## 3             turkey      avocado
## 4      mineral water         milk energy bar whole wheat rice    green tea
## 5     low fat yogurt
## 6 whole wheat pasta french fries
##   whole.weat.flour yams cottage.cheese energy.drink tomato.juice
low.fat.yogurt
```

```
## 1
## 2
## 3
## 4
## 5
## 6
##   green.tea honey salad mineral.water salmon antioxydant.juice
frozen.smoothie
## 1
## 2
## 3
## 4
## 5
## 6
##   spinach olive.oil
## 1                NA
## 2                NA
## 3                NA
## 4                NA
## 5                NA
## 6                NA
```

```
path <-"http://bit.ly/SupermarketDatasetII"

data<-read.transactions(path, sep = ",")
```

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

```
data
```

```
## transactions in sparse format with
##  7501 transactions (rows) and
##  119 items (columns)
```

```
# Structure of the dataframe
str(data)
```

```
## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:29358] 0 1 3 32 38 47 52 53 59 64 ...
##   .. .. ..@ p       : int [1:7502] 0 20 23 24 26 31 32 34 37 40 ...
##   .. .. ..@ Dim     : int [1:2] 119 7501
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : NULL
##   .. .. ..@ factors : list()
##   ..@ itemInfo   :'data.frame':  119 obs. of  1 variable:
##   .. ..$ labels: chr [1:119] "almonds" "antioxydant juice" "asparagus"
"avocado" ...
##   ..@ itemsetInfo:'data.frame':  0 obs. of  0 variables
```

```
# Changing column names to lower case, and replacing spaces with underscores
colnames(data) = tolower(str_replace_all(colnames(data), c(' ' = '_')))
# Checking column names.
colnames(data)
```

```
##   [1] "almonds"              "antioxydant_juice"    "asparagus"
##   [4] "avocado"              "babies_food"          "bacon"
##   [7] "barbecue_sauce"       "black_tea"            "blueberries"
##  [10] "body_spray"           "bramble"              "brownies"
##  [13] "bug_spray"            "burger_sauce"         "burgers"
##  [16] "butter"               "cake"                 "candy_bars"
##  [19] "carrots"              "cauliflower"          "cereals"
##  [22] "champagne"            "chicken"              "chili"
##  [25] "chocolate"            "chocolate_bread"      "chutney"
##  [28] "cider"                "clothes_accessories"  "cookies"
##  [31] "cooking_oil"          "corn"                 "cottage_cheese"
##  [34] "cream"                "dessert_wine"         "eggplant"
##  [37] "eggs"                 "energy_bar"           "energy_drink"
##  [40] "escalope"             "extra_dark_chocolate" "flax_seed"
##  [43] "french_fries"         "french_wine"          "fresh_bread"
##  [46] "fresh_tuna"           "fromage_blanc"        "frozen_smoothie"
##  [49] "frozen_vegetables"    "gluten_free_bar"      "grated_cheese"
##  [52] "green_beans"          "green_grapes"         "green_tea"
##  [55] "ground_beef"          "gums"                 "ham"
##  [58] "hand_protein_bar"     "herb_&_pepper"        "honey"
##  [61] "hot_dogs"             "ketchup"              "light_cream"
##  [64] "light_mayo"           "low_fat_yogurt"       "magazines"
##  [67] "mashed_potato"        "mayonnaise"           "meatballs"
##  [70] "melons"               "milk"                 "mineral_water"
##  [73] "mint"                 "mint_green_tea"       "muffins"
##  [76] "mushroom_cream_sauce" "napkins"              "nonfat_milk"
##  [79] "oatmeal"              "oil"                  "olive_oil"
##  [82] "pancakes"             "parmesan_cheese"      "pasta"
##  [85] "pepper"               "pet_food"             "pickles"
##  [88] "protein_bar"          "red_wine"             "rice"
##  [91] "salad"                "salmon"               "salt"
##  [94] "sandwich"             "shallot"              "shampoo"
##  [97] "shrimp"               "soda"                 "soup"
## [100] "spaghetti"            "sparkling_water"      "spinach"
## [103] "strawberries"         "strong_cheese"        "tea"
## [106] "tomato_juice"         "tomato_sauce"         "tomatoes"
## [109] "toothpaste"           "turkey"               "vegetables_mix"
## [112] "water_spray"          "white_wine"           "whole_weat_flour"
## [115] "whole_wheat_pasta"    "whole_wheat_rice"     "yams"
## [118] "yogurt_cake"          "zucchini"
```

```
# Verifying the object's class
class(data)
```

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

```
# Generating a summary of the dataset
summary(data)
```

```
## transactions as itemMatrix in sparse format with
##  7501 rows (elements/itemsets/transactions) and
##  119 columns (items) and a density of 0.03288973 
## 
## most frequent items:
## mineral_water          eggs     spaghetti  french_fries     chocolate 
##          1788          1348          1306          1282          1229 
##       (Other) 
##         22405 
## 
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15 
16 
## 1754 1358 1044  816  667  493  391  324  259  139  102   67   40   22   17 
4 
##   18   19   20 
##    1    2    1 
## 
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   1.000   2.000   3.000   3.914   5.000  20.000 
## 
## includes extended item information - examples:
##              labels
## 1           almonds
## 2 antioxydant_juice
## 3         asparagus
```

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

```
##                 Item
## 1            almonds
## 2  antioxydant_juice
## 3          asparagus
## 4            avocado
## 5        babies_food
## 6              bacon
## 7     barbecue_sauce
## 8          black_tea
## 9        blueberries
## 10        body_spray
```

```
# Exploring the frequency of some articles
#itemFrequency(data[, 8:10],type = "absolute")
#round(itemFrequency(Transactions[, 8:10],type = "relative")*100,2)

# 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 transactions dataset
# and the items whose relative importance is at least 10%
#
par(mfrow = c(1, 2))
# plot the frequency of items
#itemFrequencyPlot(Transactions, topN = 10,col="darkgreen")
#itemFrequencyPlot(Transactions, support = 0.1,col="darkred")

# 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 (Transactions, parameter = list(supp = 0.001, conf = 0.8))
#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 (Transactions,parameter = list(supp = 0.002, conf = 0.8))

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

#rules2

#rules3

# We can perform an exploration of our model
# through the use of the summary function as shown
# ---
# Upon running the code, the function would give us information about the
model
```

```r
# i.e. the size of rules, depending on the items that contain these rules.
# In our above case, most rules have 3 and 4 items though some rules do have
upto 6.
# More statistical information such as support, lift and confidence is also
provided.
# ---
#
#summary(rules)

# Observing rules built in our model i.e. first 5 model rules
# ---
#
#inspect(rules[1:5])


# Interpretation of the first rule:
# ---
# If someone buys liquor and red/blush wine, they are 90% likely to buy
bottled beer too
# ---

# Ordering these rules by a criteria such as the level of confidence
# then looking at the first five rules.
# We can also use different criteria such as: (by = "lift" or by = "support")
#
#rules<-sort(rules, by="confidence", decreasing=TRUE)
#inspect(rules[1:5])

# Interpretation
# ---
# The given five rules have a confidence of 100
# ---

# If we're interested in making a promotion relating to the sale of yogurt,
# we could create a subset of rules concerning these products
# ---
# This would tell us the items that the customers bought before purchasing
yogurt
# ---
#
#bottled beer <- subset(rules, subset = rhs %pin% "bottled beer")

# Then order by confidence
#bottled beer<-sort(bottled beer, by="confidence", decreasing=TRUE)
#inspect(bottled beer[1:5])
```