Association Rules and Anomaly detection

Wangechi Mungai

01/04/2022

1.Defining the Question

a) Specifying the Question create association rules that will allow you to identify relationships between variables in the dataset. check whether there are any anomalies in the given sales dataset

b) Defining the Metric for Success The project will be appraised successful if it will be able to detect anomalies correctly and also sho association between variables using confidence. **c) Understanding the context**

You are a Data analyst at Carrefour Kenya and are currently undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax).

d) Recording the Experimental Design

The following are the experimental design i took in order to complete this project:

1.Importing all the necessary libraries

2.Loading the dataset

3.Reading the dataset

4.Performing Association rules among thee variables

5.Performing Anomaly detection

e)Reading the Data

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.

```
library(arules)
```

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

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

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

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:arules':
##
##     intersect, recode, setdiff, setequal, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(relaimpo)
```

```
## Loading required package: MASS

##
## Attaching package: 'MASS'

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

## Loading required package: boot

## Loading required package: survey

## Loading required package: grid

## Loading required package: survival

##
## Attaching package: 'survival'

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

##
## Attaching package: 'survey'

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

```
## Loading required package: mitools

## 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.
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(ggplot2)
library(ggthemes)
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

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

## The following object is masked from 'package:stats':
##
##     filter

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

```
library(psych)
```

```
##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

## The following object is masked from 'package:boot':
##
##     logit
```

```r
# reading our data
path <-"http://bit.ly/SupermarketDatasetII"
carr<-read.transactions(path, sep = ",")
```

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

```r
carr
```

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

```r
# verifying the class of the data
class(carr)
```

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

```r
# previewing the column names
colnames(carr)
```

```
##   [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"
```

```
# Previewing our first 4 transactions
inspect(carr[1:4])
```

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

```
# preview the items that make up our dataset
carritems<-as.data.frame(itemLabels(carr))
colnames(carritems) <- "Item"
head(carritems, 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
```

There are 10 items in the dataset

```r
# previewing the summary of the dataset
summary(carr)
```

```
## 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
```

```r
# checking the frequency of some articles
itemFrequency(carr[, 8:10],type = "absolute")
```

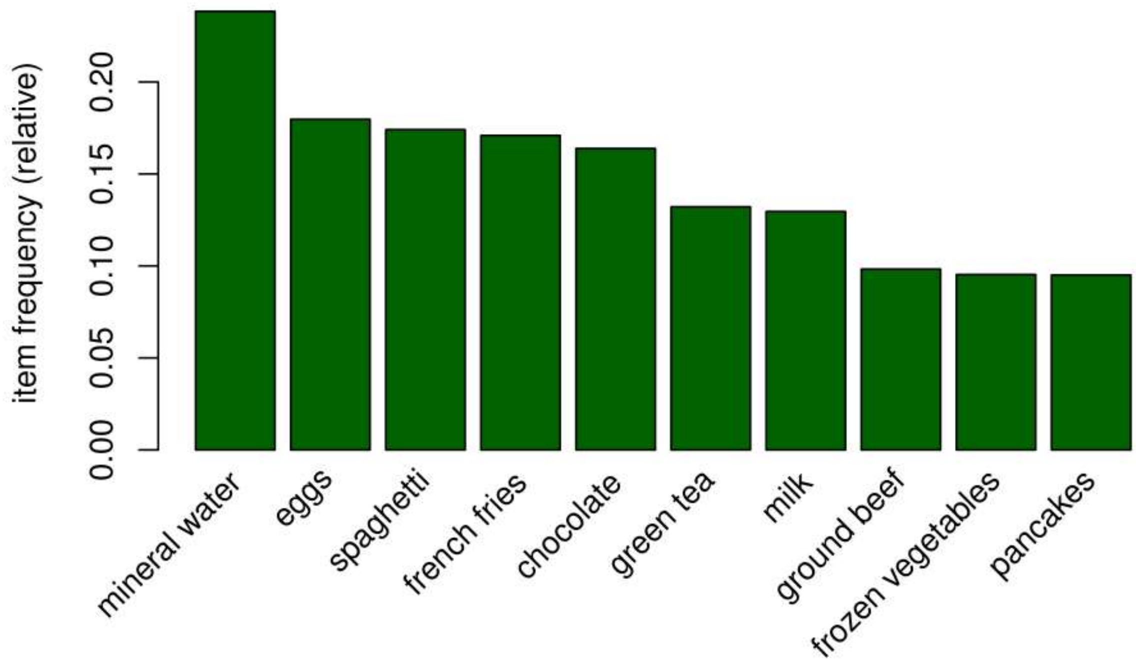
```
##   black tea blueberries  body spray
##         107          69          86
```

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

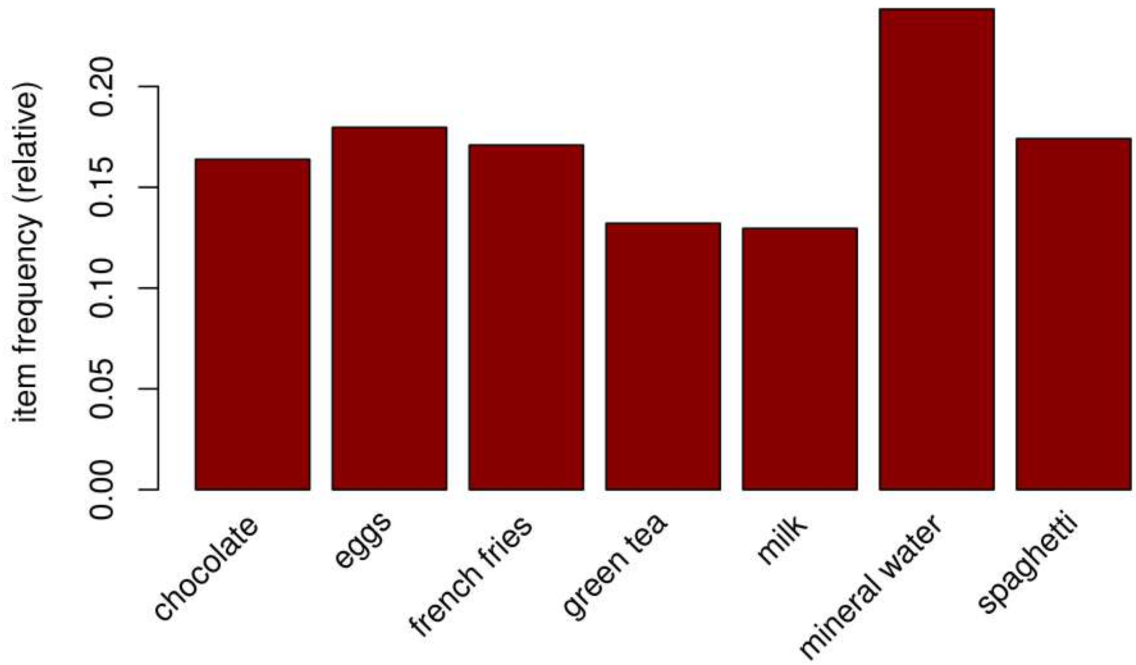
```
##   black tea blueberries  body spray
##        1.43        0.92        1.15
```

```r
# visualizing top 10  items in the transactions dataset
# and the items whose relative importance is at least 10%
par(mfrow = c(1, 2))
```

```r
# plot the frequency of items
itemFrequencyPlot(carr, topN = 10,col="darkgreen")
```




```r
itemFrequencyPlot(carr, 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
carr_rules <- apriori (carr, 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 ...[119 item(s), 7501 transaction(s)] done [0.00s].
## sorting and recoding items ... [116 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 ... [74 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
carr_rules
```

```
## set of 74 rules
```

There are 74 rules observed

```r
# Building a apriori model with Min Support as 0.002 and confidence as 0.8.
carr_rules1 <- apriori (carr,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 ...[119 item(s), 7501 transaction(s)] done [0.00s].
## sorting and recoding items ... [115 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 done [0.00s].
## writing ... [2 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
carr_rules1
```

```
## set of 2 rules
```

```r
# Building apriori model with Min Support as 0.002 and confidence as 0.6.
carr_rules2 <- apriori (carr, 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 ...[119 item(s), 7501 transaction(s)] done [0.00s].
## sorting and recoding items ... [116 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 ... [545 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
carr_rules2
```

```
## set of 545 rules
```

The first model has 74 rules while the second one has 2.They have a confidence level of 0.8 but different minimum supports. The third has 545 rules. We can conclude that when the support level is high, is equal to a loss in the rules while a low confidence level equals higher number of rules.

```
#checking descriptive statistics of the rules
summary(carr_rules)
```

```
## set of 74 rules
##
## rule length distribution (lhs + rhs):sizes
##  3  4  5  6
## 15 42 16  1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   4.000   4.000   4.041   4.000   6.000
##
## summary of quality measures:
##     support           confidence        coverage             lift
##  Min.   :0.001067   Min.   :0.8000   Min.   :0.001067   Min.   : 3.356
##  1st Qu.:0.001067   1st Qu.:0.8000   1st Qu.:0.001333   1st Qu.: 3.432
##  Median :0.001133   Median :0.8333   Median :0.001333   Median : 3.795
##  Mean   :0.001256   Mean   :0.8504   Mean   :0.001479   Mean   : 4.823
##  3rd Qu.:0.001333   3rd Qu.:0.8889   3rd Qu.:0.001600   3rd Qu.: 4.877
##  Max.   :0.002533   Max.   :1.0000   Max.   :0.002666   Max.   :12.722
##      count
##  Min.   : 8.000
##  1st Qu.: 8.000
##  Median : 8.500
##  Mean   : 9.419
##  3rd Qu.:10.000
##  Max.   :19.000
##
## mining info:
##  data ntransactions support confidence
##  carr          7501   0.001        0.8
##                                                                      call
##  apriori(data = carr, parameter = list(supp = 0.001, conf = 0.8))
```

```r
# previewing rules built in our model i.e. first 6 model rules
inspect(carr_rules[1:6])
```

```
##     lhs                                rhs             support     confidence
## [1] {frozen smoothie, spinach}      => {mineral water} 0.001066524 0.8888889
## [2] {bacon, pancakes}               => {spaghetti}     0.001733102 0.8125000
## [3] {nonfat milk, turkey}           => {mineral water} 0.001199840 0.8181818
## [4] {ground beef, nonfat milk}      => {mineral water} 0.001599787 0.8571429
## [5] {mushroom cream sauce, pasta} => {escalope}      0.002532996 0.9500000
## [6] {milk, pasta}                   => {shrimp}        0.001599787 0.8571429
##     coverage    lift      count
## [1] 0.001199840  3.729058  8
## [2] 0.002133049  4.666587 13
## [3] 0.001466471  3.432428  9
## [4] 0.001866418  3.595877 12
## [5] 0.002666311 11.976387 19
## [6] 0.001866418 11.995203 12
```

```r
#ordering by confidence for the 6 rules
carr_rules<-sort(carr_rules, by="confidence", decreasing=TRUE)
inspect(carr_rules[1:6])
```

```
##     lhs                         rhs                 support confidence    coverage      lift count
## [1] {french fries,
##      mushroom cream sauce,
##      pasta}                  => {escalope}      0.001066524  1.0000000 0.001066524 12.606723     8
## [2] {ground beef,
##      light cream,
##      olive oil}              => {mineral water} 0.001199840  1.0000000 0.001199840  4.195190     9
## [3] {cake,
##      meatballs,
##      mineral water}          => {milk}          0.001066524  1.0000000 0.001066524  7.717078     8
## [4] {cake,
##      olive oil,
##      shrimp}                 => {mineral water} 0.001199840  1.0000000 0.001199840  4.195190     9
## [5] {mushroom cream sauce,
##      pasta}                  => {escalope}      0.002532996  0.9500000 0.002666311 11.976387    19
## [6] {red wine,
##      soup}                   => {mineral water} 0.001866418  0.9333333 0.001999733  3.915511    14
```

the first four have a confidence of 100% while the 5th has 95% and the 6th has 93%

```r
#creating a subset of milj
milk <- subset(carr_rules, subset = rhs %pin% "milk")

# Then order by confidence
milk<-sort(milk, by="confidence", decreasing=TRUE)
milk
```

```
## set of 5 rules
```

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

```
##     lhs                                     rhs    support     confidence
## [1] {cake, meatballs, mineral water}     => {milk} 0.001066524 1.0000000
## [2] {escalope, hot dogs, mineral water} => {milk} 0.001066524 0.8888889
## [3] {meatballs, whole wheat pasta}      => {milk} 0.001333156 0.8333333
## [4] {black tea, frozen smoothie}        => {milk} 0.001199840 0.8181818
## [5] {burgers, ground beef, olive oil}   => {milk} 0.001066524 0.8000000
##     coverage    lift     count
## [1] 0.001066524 7.717078  8
## [2] 0.001199840 6.859625  8
## [3] 0.001599787 6.430898 10
## [4] 0.001466471 6.313973  9
## [5] 0.001333156 6.173663  8
```

Intrepretation: if one bought milk, they is 100% confidence they will buy cake,meatballs and mineral water if one bought milk, they is 89% confidence they will buy escalope, hot dogs, mineral water if one bought milk, they is 83% confidence they will buy meatballs, whole wheat pasta if one bought milk, they is 81% confidence they will buy black tea, frozen smoothie if one bought milk, they is 80% confidence they will buy burgers, ground beef, olive oil

Part 4: Anomaly Detection

You have also been requested to check whether there are any anomalies in the given sales dataset. The objective of this task being fraud detection.

```r
library(anomalize)
```

```
## == Use anomalize to improve your Forecasts by 50%! =============================
## Business Science offers a 1-hour course - Lab #18: Time Series Anomaly Detection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```r
anom <- read.csv("http://bit.ly/CarreFourSalesDataset")
head(anom)
```

```
##        Date    Sales
## 1  1/5/2019 548.9715
## 2  3/8/2019  80.2200
## 3  3/3/2019 340.5255
## 4 1/27/2019 489.0480
## 5  2/8/2019 634.3785
## 6 3/25/2019 627.6165
```

```r
#previewing the dataset
dim(anom)
```

```
## [1] 1000    2
```

The dataset has 1000 rows and 2 columns

```r
#checking descriptive statistics
summary(anom)
```

```
##      Date               Sales        
##  Length:1000        Min.   :  10.68  
##  Class :character   1st Qu.: 124.42  
##  Mode  :character   Median : 253.85  
##                     Mean   : 322.97  
##                     3rd Qu.: 471.35  
##                     Max.   :1042.65
```

Implementing the solution

```r
library(data.table)
library(psych)
```

```r
library(mvtnorm)
library(caret)
```

```
## Loading required package: lattice
```

```
## 
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
## 
##     melanoma
```

```
## 
## Attaching package: 'caret'
```

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

```r
library(PRROC)
```

```r
anomm <- sum(as.numeric(anom$Class))/nrow(anom)
sprintf('Fraud transactions in the dataset %f', anomm*100)
```

```
## [1] "Fraud transactions in the dataset 0.000000"
```

conclusion There are no anomalies in the dataset