TITLE: CARREFOUR SUPERMARKET PRODUCT ASSOCIATION ANALYSIS

AUTHOR: JOSEPH NJUGUNA

DATE: 10/6/22

1.Defining the question

a) Specifying the question

Create association rules that will allow you to identify relationships between variables in the dataset.

b) Defining the metric for success

Attainemnet of a model that gives meaningful insights on association rules.

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). Your project has been divided into four parts where you'll explore a recent marketing dataset by performing various unsupervised learning techniques and later providing recommendations based on your insights.

d) Recording the experimental design

- Exploratory data analysis

- Implementing the solution

2. Reading the data

```
library(arules)
```

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

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

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

```
products <- read.transactions("Supermarket_Sales_Dataset II part(2).csv", sep = ",")
```

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

```
products
```

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

3. Exploring the data

```
### inspecting first 5 elements
inspect(products[1:5])
```

```
##     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}
## [5] {energy bar,
##      green tea,
##      milk,
##      mineral water,
##      whole wheat rice}
```

```
### class of our dataset
class(products)
```

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

Dataset is of class: transactions

```
dim(products)
```

```
## [1] 7501  119
```

Total of 119 instances in our dataset.

```
### data types
str(products)
```

```
## 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
```

Dataset structure consists of integers, characters etc.

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

```
##                    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
## 11                bramble
## 12               brownies
## 13              bug spray
## 14           burger sauce
## 15                burgers
## 16                 butter
## 17                   cake
## 18             candy bars
## 19                carrots
## 20            cauliflower
## 21                cereals
## 22              champagne
## 23                chicken
## 24                  chili
## 25              chocolate
## 26        chocolate bread
## 27                chutney
## 28                  cider
## 29   clothes accessories
## 30                cookies
## 31           cooking oil
## 32                  corn
## 33         cottage cheese
## 34                  cream
## 35          dessert wine
## 36              eggplant
## 37                  eggs
## 38            energy bar
## 39           energy drink
## 40              escalope
## 41  extra dark chocolate
## 42             flax seed
## 43          french fries
## 44           french wine
## 45           fresh bread
## 46            fresh tuna
## 47         fromage blanc
## 48       frozen smoothie
## 49     frozen vegetables
## 50       gluten free bar
## 51         grated cheese
## 52            green beans
## 53           green grapes
## 54              green tea
## 55            ground beef
## 56                  gums
## 57                   ham
## 58      hand protein bar
## 59         herb & pepper
## 60                 honey
```

```
## 61              hot dogs
## 62               ketchup
## 63           light cream
## 64            light mayo
## 65        low fat yogurt
## 66             magazines
## 67         mashed potato
## 68            mayonnaise
## 69             meatballs
## 70                melons
## 71                  milk
## 72         mineral water
## 73                  mint
## 74        mint green tea
## 75               muffins
## 76  mushroom cream sauce
## 77               napkins
## 78           nonfat milk
## 79               oatmeal
## 80                   oil
## 81             olive oil
## 82              pancakes
## 83       parmesan cheese
## 84                 pasta
## 85                pepper
## 86              pet food
## 87               pickles
## 88           protein bar
## 89              red wine
## 90                  rice
## 91                 salad
## 92                salmon
## 93                  salt
## 94              sandwich
## 95               shallot
## 96               shampoo
## 97                shrimp
## 98                  soda
## 99                  soup
## 100            spaghetti
## 101      sparkling water
## 102              spinach
## 103        strawberries
## 104        strong cheese
## 105                  tea
## 106         tomato juice
## 107         tomato sauce
## 108             tomatoes
## 109           toothpaste
## 110               turkey
## 111       vegetables mix
## 112          water spray
## 113           white wine
## 114     whole weat flour
```

```
## 115    whole wheat pasta
## 116     whole wheat rice
## 117                 yams
## 118          yogurt cake
## 119            zucchini
```

Carrefour sells a total of 119 different items.

```
### brief summary of dataset
summary(products)
```

```
## 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
```

Most frequent items in descending order are: mineral water, eggs, spaghetti, french fries, chocolate etc.

```
### transactions ranging from 8-10
itemFrequency(products[, 8:10],type = "absolute")
```

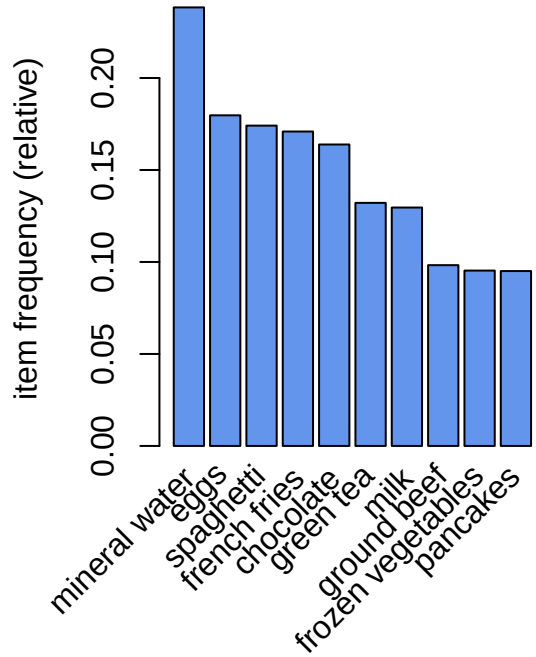
```
##   black tea blueberries  body spray
##         107          69          86
```

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

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

4. Implementing the Solution

```
### plot of 10 most common items
### top 10 most common items sold in Carrefour
par(mfrow = c(1, 2))
# plot the frequency of items
itemFrequencyPlot(products, topN = 10,col="cornflowerblue")
```



Mineral water and eggs are the top most purchased items.

```
### modeling based on association rules
### hyperparameters are Min Support as 0.001 and confidence as 0.8
rules <- apriori (products, 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 [0.00s].
## writing ... [74 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
### number of rules
rules
```

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

With the set hyperparametrs, 74 rules were created.

```
### preview of first 5 rules
inspect(rules[1:5])
```

```
##     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
##     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
```

A customer who buys frozen smoothie or spinach are more likely to buy mineral water.

A customer who buys mushroom cream sauce or pasta are more likely to buy escalope. Escalope is a meat thus, consumers often like accompanying their pasta dipped in mushroom sauce with escalope on the side.

```
### creating promotion related to the sale of groundbeef
groundbeef <- subset(rules, subset = rhs %pin% "ground beef")

### order by confidence
groundbeef <-sort(groundbeef, by="confidence", decreasing=TRUE)
inspect(groundbeef)
```

```
##     lhs                                    rhs           support
## [1] {herb & pepper, mineral water, rice} => {ground beef} 0.001333156
```

```
## [2] {grated cheese, mineral water, rice} => {ground beef} 0.001066524
##     confidence coverage    lift     count
## [1] 0.9090909  0.001466471 9.252498 10
## [2] 0.8888889  0.001199840 9.046887  8
```

Customers who buy herb, pepper, mineral water and rice are more likely to buy groundbeef

```
### creating promotion related to the sale of french fries
fries <- subset(rules, subset = rhs %pin% "french fries")

### order by confidence
fries <-sort(fries, by="confidence", decreasing=TRUE)
inspect(fries)
```

```
##     lhs                           rhs            support     confidence
## [1] {cookies, green tea, milk} => {french fries} 0.001066524 0.8
##     coverage    lift     count
## [1] 0.001333156 4.680811 8
```

Customers who buy cookies, green tea or milk are more likely to buy french fries. Odd! I know.

```
### creating promotion related to the sale of chocolate
chocolate <- subset(rules, subset = rhs %pin% "chocolate")

### order by confidence
chocolate <-sort(chocolate, by="confidence", decreasing=TRUE)
inspect(chocolate)
```

```
##     lhs                                 rhs         support     confidence
## [1] {escalope, french fries, shrimp} => {chocolate} 0.001066524 0.8888889
## [2] {red wine, tomato sauce}         => {chocolate} 0.001066524 0.8000000
##     coverage    lift     count
## [1] 0.001199840 5.425188 8
## [2] 0.001333156 4.882669 8
```

Customers who buy red wine or tomato sauce are more likely to buy chocolate.

Customers who buy escalope, french fries or shrimp are more likely to buy chocolate. Odd! I know.

Hyperparameter Tuning

```
### To illustrate the sensitivity of the model to these two parameters, we will see what happens if we
### Min Support as 0.002 and confidence as 0.8.
rulesa <- apriori (products,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].
```

```
### number of rules
rulesa
```

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

With the set hyperparameters, rules created are 2.

```
### preview rules
inspect(rulesa)
```

```
##     lhs                                             rhs         support
## [1] {mushroom cream sauce, pasta}            => {escalope}  0.002532996
## [2] {frozen vegetables, olive oil, tomatoes} => {spaghetti} 0.002133049
##     confidence coverage    lift      count
## [1] 0.9500000  0.002666311 11.976387 19
## [2] 0.8421053  0.002532996  4.836624 16
```

Only 2 rows are captured in this model.

As observed a set of 2 rules is not insightful at all, and therefore we discard this model.

```
rulesb <- apriori (products,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 [0.00s].
## writing ... [545 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rulesb
```

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

Compared to our first model, our second model (rulesb) has 545 rules.

```
### preview of first 5 rules
inspect(rulesb[1:5])
```

```
##     lhs                          rhs              support     confidence
## [1] {cookies, shallot}        => {low fat yogurt} 0.001199840 0.6000000
## [2] {low fat yogurt, shallot} => {cookies}        0.001199840 0.6923077
## [3] {cookies, shallot}        => {green tea}      0.001199840 0.6000000
## [4] {cookies, shallot}        => {french fries}   0.001199840 0.6000000
## [5] {low fat yogurt, shallot} => {french fries}   0.001066524 0.6153846
##     coverage    lift     count
## [1] 0.001999733 7.840767 9
## [2] 0.001733102 8.611940 9
## [3] 0.001999733 4.541473 9
## [4] 0.001999733 3.510608 9
## [5] 0.001733102 3.600624 8
```

We observe that our first 5 rules have changed.

A customer who buys cookies or shallot is 60% likely to buy low fat yoghurt or green tea.

Lets carry out promotion strategies on the same items(ground beef, chocolate and french fries) to see if change of parameters has any effect.

```
### creating promotion related to the sale of groundbeef
groundbeef <- subset(rulesb, subset = rhs %pin% "ground beef")
```

```
### order by confidence
groundbeef <-sort(groundbeef, by="confidence", decreasing=TRUE)
inspect(groundbeef)
```

```
##      lhs                    rhs              support confidence    coverage     lift count
## [1]  {herb & pepper,
##       mineral water,
##       rice}              => {ground beef} 0.001333156  0.9090909 0.001466471 9.252498    10
## [2]  {grated cheese,
##       mineral water,
##       rice}              => {ground beef} 0.001066524  0.8888889 0.001199840 9.046887     8
## [3]  {burgers,
##       herb & pepper,
##       spaghetti}         => {ground beef} 0.001333156  0.7692308 0.001733102 7.829037    10
## [4]  {green tea,
##       spaghetti,
##       tomato sauce}      => {ground beef} 0.001333156  0.7142857 0.001866418 7.269820    10
## [5]  {frozen vegetables,
##       herb & pepper,
##       mineral water,
##       spaghetti}         => {ground beef} 0.001199840  0.6923077 0.001733102 7.046133     9
## [6]  {herb & pepper,
##       shrimp,
##       spaghetti}         => {ground beef} 0.001466471  0.6875000 0.002133049 6.997201    11
## [7]  {cereals,
##       green tea,
##       spaghetti}         => {ground beef} 0.001066524  0.6666667 0.001599787 6.785165     8
## [8]  {burgers,
##       herb & pepper,
##       milk}              => {ground beef} 0.001066524  0.6666667 0.001599787 6.785165     8
## [9]  {chocolate,
##       eggs,
##       herb & pepper,
##       mineral water}     => {ground beef} 0.001066524  0.6666667 0.001599787 6.785165     8
## [10] {herb & pepper,
##       oil}               => {ground beef} 0.001199840  0.6428571 0.001866418 6.542838     9
## [11] {light cream,
##       mineral water,
##       olive oil}         => {ground beef} 0.001199840  0.6428571 0.001866418 6.542838     9
## [12] {chicken,
##       herb & pepper,
##       spaghetti}         => {ground beef} 0.001199840  0.6428571 0.001866418 6.542838     9
## [13] {burgers,
##       herb & pepper,
##       mineral water}     => {ground beef} 0.001199840  0.6428571 0.001866418 6.542838     9
## [14] {herb & pepper,
##       rice}              => {ground beef} 0.001866418  0.6363636 0.002932942 6.476748    14
## [15] {green tea,
##       tomato sauce}      => {ground beef} 0.001599787  0.6315789 0.002532996 6.428051    12
## [16] {frozen vegetables,
##       herb & pepper,
##       spaghetti}         => {ground beef} 0.001599787  0.6315789 0.002532996 6.428051    12
## [17] {chicken,
```

```
##       herb & pepper,
##       mineral water}      => {ground beef} 0.001333156  0.6250000 0.002133049 6.361092    10
## [18] {salmon,
##       tomato sauce}       => {ground beef} 0.001066524  0.6153846 0.001733102 6.263229     8
## [19] {grated cheese,
##       mineral water,
##       shrimp}             => {ground beef} 0.001066524  0.6153846 0.001733102 6.263229     8
```

```
### creating promotion related to the sale of french fries
fries <- subset(rulesb, subset = rhs %pin% "french fries")

### order by confidence
fries <-sort(fries, by="confidence", decreasing=TRUE)
inspect(fries)
```

```
##     lhs                           rhs            support     confidence
## [1] {cookies, green tea, milk} => {french fries} 0.001066524 0.8000000
## [2] {eggs, ham, milk}          => {french fries} 0.001066524 0.7272727
## [3] {low fat yogurt, shallot}  => {french fries} 0.001066524 0.6153846
## [4] {cookies, shallot}         => {french fries} 0.001199840 0.6000000
##     coverage    lift     count
## [1] 0.001333156 4.680811 8
## [2] 0.001466471 4.255283 8
## [3] 0.001733102 3.600624 8
## [4] 0.001999733 3.510608 9
```

```
### creating promotion related to the sale of chocolate
chocolate <- subset(rulesb, subset = rhs %pin% "chocolate")

### order by confidence
chocolate <-sort(chocolate, by="confidence", decreasing=TRUE)
inspect(chocolate)
```

```
##      lhs                    rhs             support confidence    coverage     lift count
## [1]  {escalope,
##       french fries,
##       shrimp}             => {chocolate} 0.001066524  0.8888889 0.001199840 5.425188     8
## [2]  {red wine,
##       tomato sauce}       => {chocolate} 0.001066524  0.8000000 0.001333156 4.882669     8
## [3]  {burgers,
##       olive oil,
##       pancakes}           => {chocolate} 0.001199840  0.7500000 0.001599787 4.577502     9
## [4]  {almonds,
##       olive oil,
##       spaghetti}          => {chocolate} 0.001066524  0.7272727 0.001466471 4.438790     8
## [5]  {almonds,
##       milk,
##       spaghetti}          => {chocolate} 0.001066524  0.7272727 0.001466471 4.438790     8
## [6]  {frozen vegetables,
##       mineral water,
##       pancakes,
##       shrimp}             => {chocolate} 0.001066524  0.7272727 0.001466471 4.438790     8
## [7]  {shrimp,
```

```
##       tomato sauce}       => {chocolate} 0.001066524  0.6666667 0.001599787 4.068891     8
## [8]  {butter,
##       escalope}           => {chocolate} 0.001066524  0.6666667 0.001599787 4.068891     8
## [9]  {burgers,
##       chicken,
##       french fries}       => {chocolate} 0.001066524  0.6666667 0.001599787 4.068891     8
## [10] {french fries,
##       tomato sauce}       => {chocolate} 0.001466471  0.6470588 0.002266364 3.949217    11
## [11] {butter,
##       salmon}             => {chocolate} 0.001466471  0.6470588 0.002266364 3.949217    11
## [12] {frozen vegetables,
##       mineral water,
##       olive oil,
##       shrimp}             => {chocolate} 0.001199840  0.6428571 0.001866418 3.923573     9
## [13] {eggs,
##       spaghetti,
##       tomato sauce}       => {chocolate} 0.001066524  0.6153846 0.001733102 3.755899     8
## [14] {eggs,
##       escalope,
##       green tea}          => {chocolate} 0.001066524  0.6153846 0.001733102 3.755899     8
## [15] {escalope,
##       oil}                => {chocolate} 0.001466471  0.6111111 0.002399680 3.729816    11
## [16] {cooking oil,
##       eggs,
##       pancakes}           => {chocolate} 0.001199840  0.6000000 0.001999733 3.662002     9
## [17] {cake,
##       eggs,
##       olive oil}          => {chocolate} 0.001199840  0.6000000 0.001999733 3.662002     9
```

We observe that even with a different model, preference of customers remains the same.

This affirms the validity of our first model.

5. Conclusions

Most frequent items in descending order are: mineral water, eggs, spaghetti, french fries, chocolate etc

A customer who buys frozen smoothie or spinach are more likely to buy mineral water.

A customer who buys mushroom cream sauce or pasta are more likely to buy escalope. Escalope is a meat thus, consumers often like accompanying their pasta dipped in mushroom sauce with escalope on the side.

Customers who buy cookies, green tea or milk are more likely to buy french fries. Odd! I know.

Customers who buy red wine or tomato sauce are more likely to buy chocolate.

Customers who buy escalope, french fries or shrimp are more likely to buy chocolate. Odd! I know.

As expected, data analysis gives us more questions than answers in some situations.

6. Recommendations

Carry out bundle promotions for most common items bought.

To realize high rate of stock turnover on the less frequent purchased items, offer discounts.

Most commonly bought items to be shelved strategically near each other.

Red wine and chocolate should be advertised together.

Breakfast products precisely yoghurt, milk, shallow etc., should be shelved next to each other.

7. Follow up questions

a) Did we have right data?

```
Yes.
```

b) Do we need other data to answer our question?

```
No, however other data is need if we were to focus on certain brands sold.
```

c) Did we have the right question?

```
Yes.
```