Association analysis

Daisy Lynn

2022-04-02

```
library(readr)
library(caret)
```

```
## Loading required package: ggplot2
```

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

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v purrr   0.3.4     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(dplyr)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(arules)
```

```
## 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
```

```
path <- "http://bit.ly/SupermarketDatasetII"
trans<-read.transactions(path, sep = ",")
```

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

```
trans
```

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

```
# Previewing our first 5 transactions
#
inspect(trans[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}
```

```
#previewing Items

items<-as.data.frame(itemLabels(trans))
colnames(items) <- "Item"
head(items, 12)
```

```
##                 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
```

```
#generating the summary
summary(trans)
```

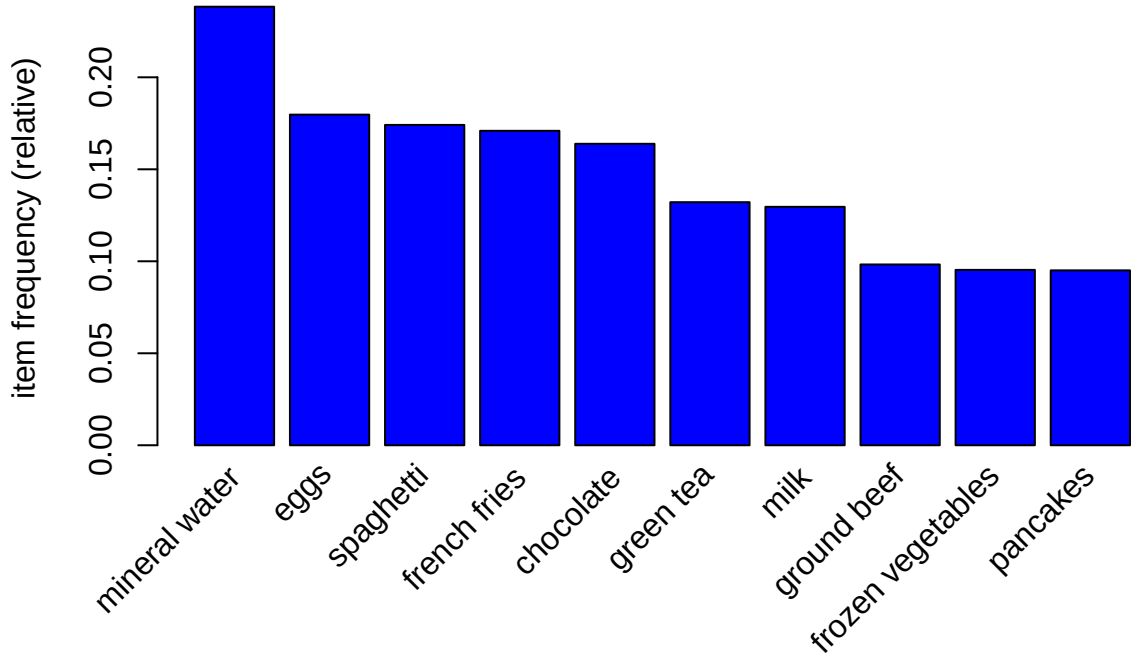
```
## transactions as itemMatrix in sparse format with
##  7501 rows (elements/itemsets/transactions) and
##  119 columns (items) and a density of 0.03288973
##
## most frequent items:
## mineral water          eggs     spaghetti  french fries     chocolate
##          1788          1348          1306          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
```

from above summary it can be seen that mineral water, spaghetti and eggs are the most popular items

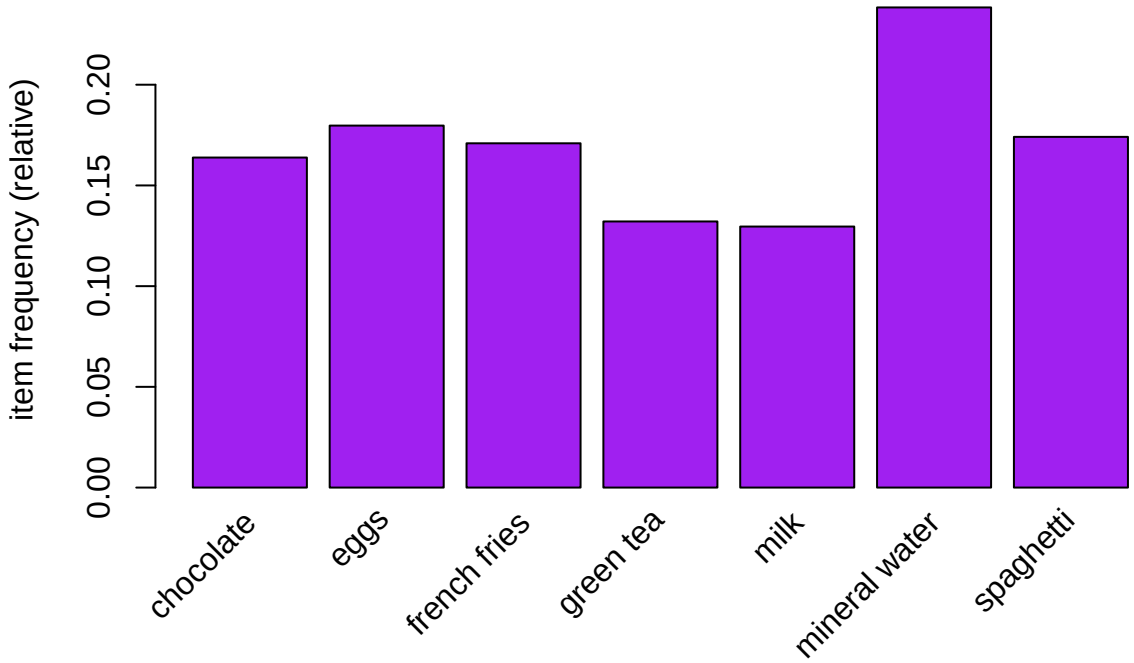
```
# Displaying top 10 most popular items among the transactions

itemFrequencyPlot(trans, topN = 10,col="blue")
```




```
#items whose relative importance is at least 10%

itemFrequencyPlot(trans, support = 0.1,col="purple")
```




```
#Building a model with a confidence of 0.8

rules <- apriori (trans, 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].
```

```
rules
```

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

above result shows we have 74 rules

```
# Building a apriori model with Min Support as 0.002 and confidence as 0.8.
rules2 <- apriori (trans,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].
```

we only got 2 rules using a support of 0.002,this means we lost important rules

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

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5   0.002      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 ... [43 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

using a confidence of 0.6 and support of 0.002 we have 43 rules,comparing this with the first model with a confidence 0.8 our rules decreased

we will use the first model since it has fair number of rules

```
summary(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
##  trans          7501   0.001        0.8
##                                                                 call
##  apriori(data = trans, parameter = list(supp = 0.001, conf = 0.8))
```

most rules 4 items followed by 3, with 1 rule having 6 items

```
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
```

In the first rule, if someone buys a frozen smoothie and spinach there is an 89% chance of them buying mineral water

In the second rule, if someone buys bacon and pancakes there is an 81% chance of the customer buying spaghetti

```
#ordering rules by confidence

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

```
##     lhs                         rhs                 support confidence    coverage      lift count
## [1] {french fries,
##      mushroom cream sauce,
##      pasta}                 => {escalope}      0.001066524       1.00 0.001066524 12.606723     8
## [2] {ground beef,
##      light cream,
##      olive oil}             => {mineral water} 0.001199840       1.00 0.001199840  4.195190     9
## [3] {cake,
##      meatballs,
##      mineral water}         => {milk}          0.001066524       1.00 0.001066524  7.717078     8
## [4] {cake,
##      olive oil,
##      shrimp}                => {mineral water} 0.001199840       1.00 0.001199840  4.195190     9
## [5] {mushroom cream sauce,
##      pasta}                 => {escalope}      0.002532996       0.95 0.002666311 11.976387    19
```

The above rules have a 95% and above confidence

assume carrefour want to boost chocolate sales

```
#lets see the most common items bought before a customer picks chocolate

chocolate <- subset(rules, subset = rhs %pin% "chocolate")

# Then order by confidence
chocolate <-sort(chocolate, by="confidence", decreasing=TRUE)
inspect(chocolate[1:2])
```

```
##     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 buying escalope, french fries, shrimp have the highest chances of picking chocolate, with a confidence of 89 %

```
# Well see the most common items that customers might buy who have previously bought chocolate

chocolate2 <- subset(rules, subset = lhs %pin% 'milk')
# sort by confidence
chocolate2 <- sort(chocolate2, by = 'confidence', decreasing = TRUE)

# Inspect first 10 commodities

inspect(chocolate2[15:19])
```

```
##     lhs                                    rhs             support
## [1] {chocolate, hot dogs, milk}        => {mineral water} 0.001066524
## [2] {avocado, burgers, milk}           => {spaghetti}     0.001066524
## [3] {cookies, green tea, milk}         => {french fries}  0.001066524
## [4] {cake, eggs, milk, turkey}         => {mineral water} 0.001066524
## [5] {chocolate, eggs, milk, olive oil} => {mineral water} 0.001066524
##     confidence coverage    lift     count
## [1] 0.8        0.001333156 3.356152 8
## [2] 0.8        0.001333156 4.594793 8
## [3] 0.8        0.001333156 4.680811 8
## [4] 0.8        0.001333156 3.356152 8
## [5] 0.8        0.001333156 3.356152 8
```

we see mineral water, spaghetti and french fries are common items that customers might buy who have previously bought chocolate

Recommendation

from the section where assumptions were made carrefour can boost their chocolate sale by placing the item near mineral water, french fries,shrimp,spaghetti this will gain customer attraction which will in turn boost its sales

The above model can also be used to determine how other items associates with each other, this knowledge can in turn help them find a good strategy on how to place their items and boost sales