Association Rules

Cynthia Mwadime

2022-10-06

Association Analysis

This section ivolves creation of association rules that allow us to identify relationships between variables in the dataset.

We are tasked with creating association rules that will allow us to identify relationships between variables in the dataset. We have been provided with a dataset that comprises of groups of items that will be associated with others. ### Importing the arules library

```
# Loading the arules library
library(arules, warn.conflicts = FALSE)
```

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

Loading the data

```
# loading
sm <- read.transactions("http://bit.ly/SupermarketDatasetII", sep = ",")
```

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

```
sm
```

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

```
# Previewing our first 5 transactionss
class(sm)
```

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

```
inspect(sm[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}
```

###Generating a summary of the supermarket dataset This gives us information on stuff like distribution of the item sets, most purchased items and number of items purchased in each transaction among other things

```
summary(sm)
```

```
## 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 the report we can see that the most sold items were mineral water], eggs,sphagetti,french fries and chocolate repectively.

```r
# Let's see transacations ranging from 6 to 10
# percentage of the total transactions
itemFrequency(sm[, 6:10],type = "absolute")
```

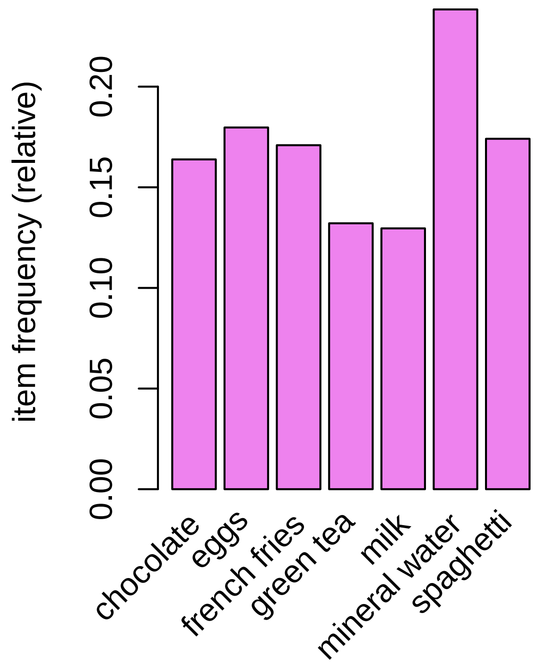
```
##          bacon barbecue sauce      black tea    blueberries     body spray
##             65             81            107             69             86
```

```r
round(itemFrequency(sm[, 6:10],type = "relative")*100,2)
```

```
##          bacon barbecue sauce      black tea    blueberries     body spray
##           0.87           1.08           1.43           0.92           1.15
```

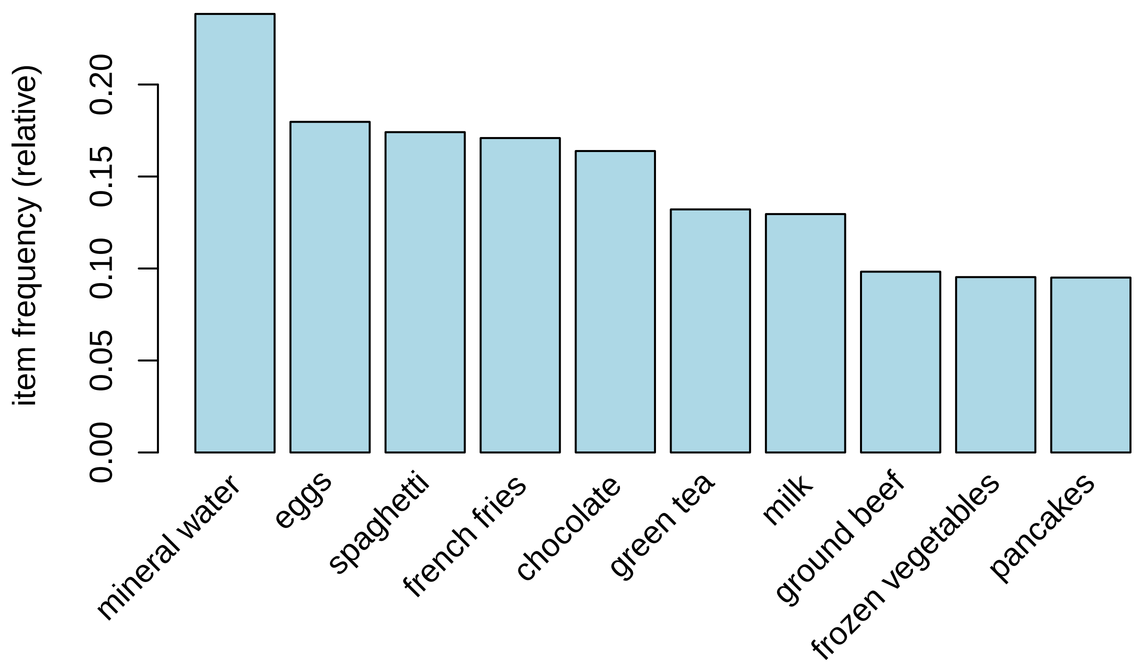
Displaying items whose relative importance is at least 10%

```r
par(mfrow = c(1, 2))
# plot the frequency of items
itemFrequencyPlot(sm, support = 0.1,col="violet")
```



Displaying top 10 most common items in the transactions dataset and the

```
itemFrequencyPlot(sm, topN = 10,col="lightblue")
```



Building a model based on association rules * We'll be using the apriori function

```r
# We use Min Support as 0.001 and confidence as 0.8
rules <- apriori (sm, parameter = list(supp = 0.001, conf = 0.7))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    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 ... [200 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
rules
```

```
## set of 200 rules
```

We obtained a set of 200 rules.

Tweking the parameters

Let's now see see happens if we increase the support or lower the confidence level.

```r
# Building a apriori model with Min Support as 0.002 and confidence as 0.6
ruls <- apriori (sm,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].
```

```r
ruls
```

```
## set of 43 rules
```

This gives us a set of 43 rules which is is not enough

```r
# Building the apriori model with Min Support as 0.002 and confidence as 0.6.
ruls2 <- apriori (sm, 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].
```

```
ruls2
```

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

This gives us 545 rules. That we can work with

Let's now get more information on support, lift and confidence

```
summary(ruls2)
```

```
## set of 545 rules
##
## rule length distribution (lhs + rhs):sizes
##   3   4   5   6
## 146 329  67   3
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   3.000   4.000   3.866   4.000   6.000
##
## summary of quality measures:
##     support           confidence        coverage             lift
##  Min.   :0.001067   Min.   :0.6000   Min.   :0.001067   Min.   : 2.517
##  1st Qu.:0.001067   1st Qu.:0.6250   1st Qu.:0.001600   1st Qu.: 2.797
##  Median :0.001200   Median :0.6667   Median :0.001866   Median : 3.446
##  Mean   :0.001409   Mean   :0.6893   Mean   :0.002081   Mean   : 3.889
##  3rd Qu.:0.001466   3rd Qu.:0.7273   3rd Qu.:0.002266   3rd Qu.: 4.177
##  Max.   :0.005066   Max.   :1.0000   Max.   :0.007999   Max.   :34.970
##      count
##  Min.   : 8.00
##  1st Qu.: 8.00
##  Median : 9.00
##  Mean   :10.57
##  3rd Qu.:11.00
##  Max.   :38.00
##
```

```
## mining info:
##  data ntransactions support confidence
##    sm          7501   0.001        0.6
##                                                           call
##  apriori(data = sm, parameter = list(supp = 0.001, conf = 0.6))
```

Most rules have 3 and 4 items

```
# Observing the first 5 rules built in our model
inspect(ruls2[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
```

If someone buys cookies and shallot they are 60% likely to buy low fat yogurt.

```
# Ordering the rules by the level of confidence then looking at the first five rules.
ruls2<-sort(ruls2, by="confidence", decreasing=TRUE)
inspect(ruls2[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 first four rules have a confidence of 100%.

```
Were interested in creating an ad relating to the sale of a particular item, we could
create a subset of rules concerning this product. This would inform us on what items the
customers bought before purchasing our target item. Let's use escalope and see our theory
in action
```

```r
escalope <- subset(rules, subset = rhs %pin% "escalope")
# Then order by confidence
escalope <-sort(escalope, by="count", decreasing=TRUE)
```

```r
inspect(escalope)
```

```
##     lhs                          rhs            support confidence    coverage     lift count
## [1] {mushroom cream sauce,
##      pasta}                   => {escalope} 0.002532996       0.95 0.002666311 11.97639    19
## [2] {french fries,
##      mushroom cream sauce,
##      pasta}                   => {escalope} 0.001066524       1.00 0.001066524 12.60672     8
```

- mushroom cream sauce and pasta were in 19 shopping basckets while frenchfries, mushroom creamsauce and pasta had been in 8 basket therefore it would motivate most mushroom cream sauce and pasta buyers to buy escalope as well were they promoted together.