Aim: Create Association Rules for the Market Basket Analysis for the given Threshold. (Using R)

Association Rules

There are many ways to see the similarities between items. These are techniques that fall under the general umbrella of association. The outcome of this type of technique, in simple terms, is a set of rules that can be understood as "if this, then **that"**.

Applications

There are many applications of association:

- Product recommendation – like Amazon's "customers who bought that, also bought this"
- Music recommendations – like Last FM's artist recommendations
- Medical diagnosis – like with diabetes really cool stuff
- Content optimisation – like in magazine websites or blogs

In this post we will focus on the retail application – it is simple, intuitive, and the dataset comes packaged with R making it repeatable.

The Groceries Dataset

Imagine 10000 receipts sitting on your table. Each receipt represents a transaction with items that were purchased. The receipt is a representation of stuff that went into a customer's basket – and therefore 'Market Basket Analysis'.

That is exactly what the Groceries Data Set contains:

a collection of receipts with each line representing 1 receipt and the items purchased. Each line is called a transaction and each column in a row represents an item.You can download the Groceries data set to take a look at it, but this is not a necessary step.

Background:

We can represent our items as an item set as follows:

I = { i1, i2, i3 .. in }

Therefore a transaction is represented as follows:

tn = { ij, ik,…..in}

This gives us our rules which are represented as follows:

{i1, i2} => ik

which can be read as "**if a user buys an item in the item set on the left hand side, then the user will likely buy the item on the right hand side too".**

A more human readable example is:

{Coffee, Sugar} => milk

If a customer buys coffee and sugar, then they are also likely to buy milk.

With this we can understand three important ratios; the support, confidence and lift. We describe the significance of these in the following bullet points.

- Support: The fraction of which our item set occurs in our dataset.
- Confidence: probability that a rule is correct for a new transaction with items on the left.
- Lift: The ratio by which by the confidence of a rule exceeds the expected confidence.

Note: if the lift is 1 it indicates that the items on the left and right are Independent

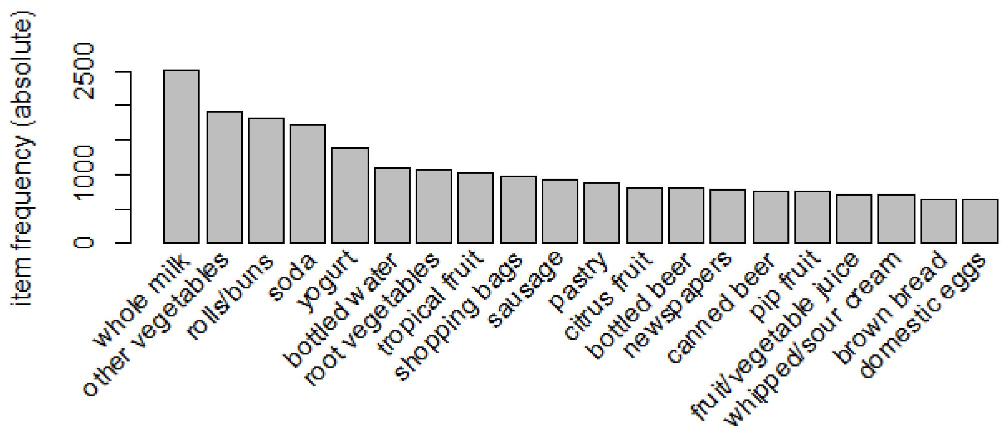
Apriori Recommendation with R

So lets get started by loading up our libraries and data set.

```
# Load the libraries
library(arules)
library(arulesViz)
library(datasets)
# Load the data set
data(Groceries)
```

Lets explore the data before we make any rules:

 # Create an item frequency plot for the top 20 items
itemFrequencyPlot(Groceries,topN=20,type="absolute")



We are now ready to mine some rules!
You will always have to pass the minimum required support and confidence.

- We set the minimum support to 0.001
- We set the minimum confidence of 0.8
- We then show the top 5 rules

```
# Get the rules
rules <- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8))
# Show the top 5 rules, but only 2 digits
options(digits=2)
inspect(rules[1:5])
```

The output we see should look something like this

```
lhs rhs support confidence lift
1 {liquor,red/blush wine} => {bottled beer} 0.0019 0.90 11.2
```

```
2 {curd,cereals} => {whole milk} 0.0010 0.91 3.6
3 {yogurt,cereals} => {whole milk} 0.0017 0.81 3.2
4 {butter,jam} => {whole milk} 0.0010 0.83 3.3
5 {soups,bottled beer} => {whole milk} 0.0011 0.92 3.6
```

This reads easily, for example: if someone buys yogurt and cereals, they are 81% likely to buy whole milk too.

We can get summary info. about the rules that give us some interesting information such as:

- The number of rules generated: 410
- The distribution of rules by length: Most rules are 4 items long
- The summary of quality measures: interesting to see ranges of support, lift, and confidence.
- The information on the data mined: total data mined, and minimum parameters

```
set of 410 rules
rule length distribution (lhs + rhs): sizes
3 4 5 6
29 229 140 12
summary of quality measures:
support conf. lift
Min. :0.00102 Min. :0.80 Min. : 3.1
1st Qu.:0.00102 1st Qu.:0.83 1st Qu.: 3.3
Median :0.00122 Median :0.85 Median : 3.6
Mean :0.00125 Mean :0.87 Mean : 4.0
3rd Qu.:0.00132 3rd Qu.:0.91 3rd Qu.: 4.3
Max. :0.00315 Max. :1.00 Max. :11.2
mining info:
data n support confidence
Groceries 9835 0.001 0.8
```

Sorting stuff out

The first issue we see here is that the rules are not sorted. Often we will want the most relevant rules first.Lets say we wanted to have the most likely rules. We can easily sort by confidence by executing the following code.

```
rules<-sort(rules, by="confidence", decreasing=TRUE)
```

Now our top 5 output will be sorted by confidence and therefore the most relevant rules appear.

```
lhs rhs support conf. lift
1 {rice,sugar} => {whole milk} 0.0012 1 3.9
2 {canned fish,hygiene articles} => {whole milk} 0.0011 1 3.9
3 {root vegetables,butter,rice} => {whole milk} 0.0010 1 3.9
4 {root vegetables,whipped/sour cream,flour} => {whole milk} 0.0017 1 3.9
5 {butter,soft cheese,domestic eggs} => {whole milk} 0.0010 1 3.9
```

Rule 4 is perhaps excessively long. Lets say you wanted more concise rules. That is also easy to do by adding a “maxlen” parameter to your apriori function

```
rules <- apriori(Groceries, parameter = list(supp = 0.001, conf =
0.8,maxlen=3))
```

Redundancies

Sometimes, rules will repeat. Redundancy indicates that one item might be a given. As an analyst you can elect to drop the item from the dataset. Alternatively, you can remove redundant rules generated. We can eliminate these repeated rules using the follow snippet of code:

```
subset.matrix <- is.subset(rules, rules)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1
rules.pruned <- rules[!redundant]
rules<-rules.pruned
```

Targeting Items

Now that we know how to generate rules, limit the output, lets say we wanted to target items to generate rules. There are two types of targets we might be interested in that are illustrated with an example of "whole milk":

- What are customers likely to buy before buying whole milk
- What are customers likely to buy if they purchase whole milk?

This essentially means we want to set either the Left Hand Side and Right Hand Side. This is not difficult to do with R! Answering the first question we adjust our apriori() function as follows:

```
rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.08),
appearance = list(default="lhs",rhs="whole milk"),
control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])
```

The output will look like this:

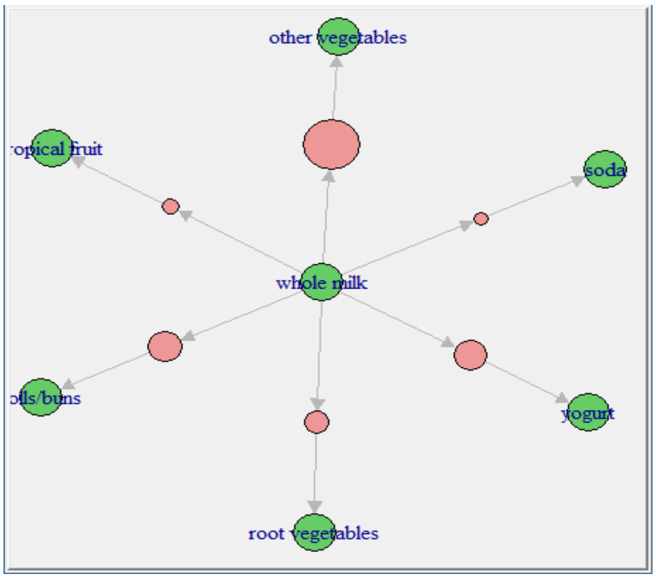
```
lhs rhs supp. conf. lift
1 {rice,sugar} => {whole milk} 0.0012 1 3.9
2 {canned fish,hygiene articles} => {whole milk} 0.0011 1 3.9
3 {root vegetables,butter,rice} => {whole milk} 0.0010 1 3.9
4 {root vegetables,whipped/sour cream,flour} => {whole milk} 0.0017 1 3.9
5 {butter,soft cheese, domestic eggs} => {whole milk} 0.0010 1 3.9
```

Visualization

The last step is visualization. Lets say you wanted to map out the rules in a graph. We can do that with another library called "arulesViz".

```
library(arulesViz)
plot(rules,method="graph",interactive=TRUE,shading=NA)
```

You will get a nice graph that you can move around to look like this:



Conclusion:

Thus we have created Association Rule for the Market Basket Analysis for the given Threshold Using Rstudio

```r
# Load the libraries for apriori algorithm, visulizations and for required data set
library(arules)
library(arulesViz)
library(datasets)
# Load the data set
data(Groceries)
#Lets explore the data before we make any rules:
# Create an item frequency plot for the top 20 items
itemFrequencyPlot(Groceries,topN=20,type="absolute")
#You will always have to pass the minimum required support and confidence.
# We set the minimum support to 0.001
# We set the minimum confidence of 0.8
# Get the rules
rules <- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8))
# Show the top 5 rules, but only 2 digits
options(digits=2)
inspect(rules[1:5])
#Sorting Rules by confidence
rules<-sort(rules, by="confidence", decreasing=TRUE)
rules <- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8,maxlen=3))
# Redundancies
subset.matrix <- is.subset(rules, rules)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1
rules.pruned <- rules[!redundant]
rules<-rules.pruned
#Targeting items
rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.08),
appearance = list(default="lhs",rhs="whole milk"),
control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])
rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.15,minlen=2),
appearance = list(default="rhs",lhs="whole milk"),
control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])
#Visualization
library(arulesViz)
plot(rules,method="graph",interactive=TRUE,shading=NA)
```

```
library(arules)
library(arulesViz)
```

```
Loading required package: grid
 library(datasets)

data(Groceries)

itemFrequencyPlot(Groceries,topN=20,type="absolute")


rules <- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8))
```

```
Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5   0.001      1
10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 9

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.02s].
sorting and recoding items ... [157 item(s)] done [0.00s].
creating transaction tree ... done [0.02s].
checking subsets of size 1 2 3 4 5 6 done [0.07s].
writing ... [410 rule(s)] done [0.01s].
creating S4 object  ... done [0.01s].
```

```
options(digits=2)
inspect(rules[1:5])
```

```
  lhs                          rhs            support confidence coverage lift count
[1] {liquor,red/blush wine} => {bottled beer} 0.0019  0.90       0.0021   11.2 19
[2] {curd,cereals}          => {whole milk}   0.0010  0.91       0.0011    3.6 10
[3] {yogurt,cereals}        => {whole milk}   0.0017  0.81       0.0021    3.2 17
[4] {butter,jam}            => {whole milk}   0.0010  0.83       0.0012    3.3 10
[5] {soups,bottled beer}    => {whole milk}   0.0011  0.92       0.0012    3.6 11
```

```
rules<-sort(rules, by="confidence", decreasing=TRUE)
rules <- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8,maxl
en=3))
Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5   0.001      1
3  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 9
```

```
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.02s].
sorting and recoding items ... [157 item(s)] done [0.00s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 done [0.02s].
writing ... [29 rule(s)] done [0.00s].
creating S4 object  ... done [0.01s].
```

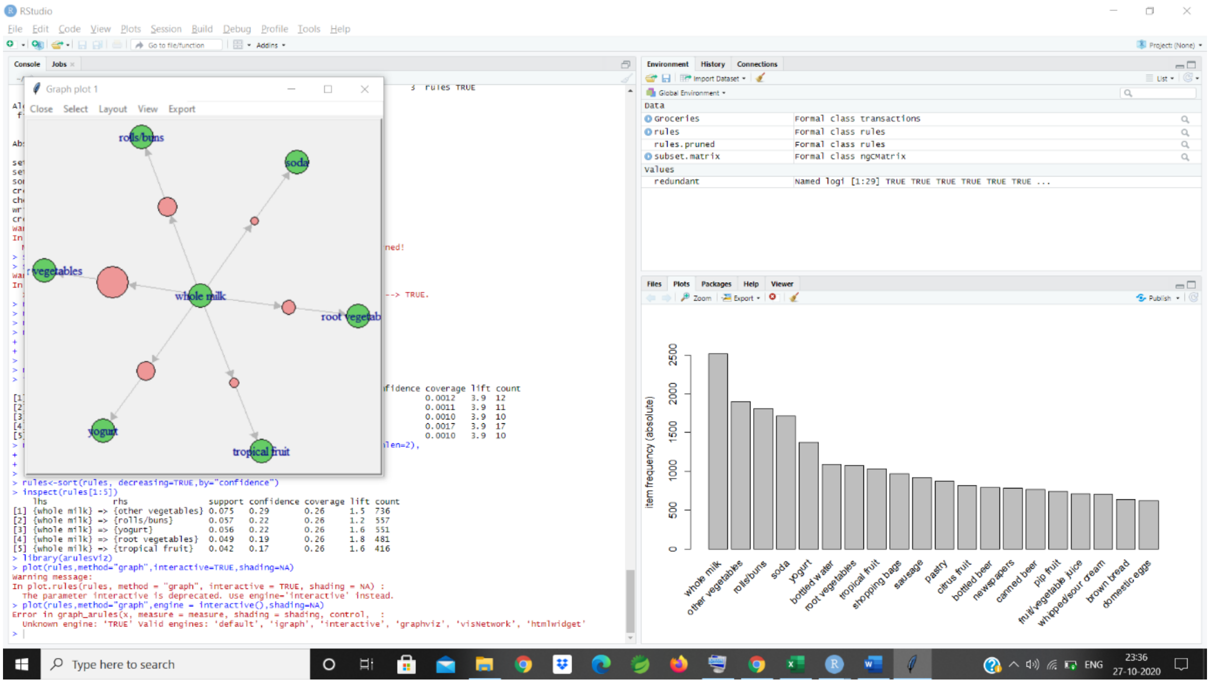
```
subset.matrix <- is.subset(rules, rules)

subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1

rules.pruned <- rules[!redundant]

rules<-rules.pruned

rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.08),
+                  appearance = list(default="lhs",rhs="whole milk"),
+                  control = list(verbose=F))

rules<-sort(rules, decreasing=TRUE,by="confidence")

inspect(rules[1:5])
```

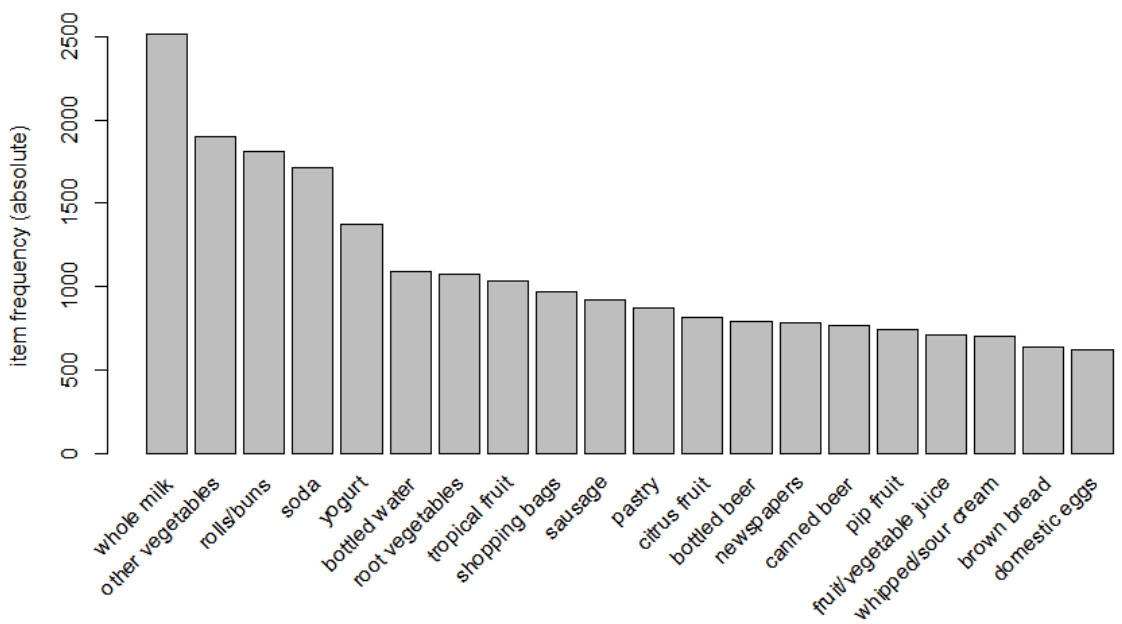
```
    lhs                                            rhs          support confidence coverage lift count
[1] {rice,sugar}                                => {whole milk} 0.0012  1          0.0012   3.9  12
[2] {canned fish,hygiene articles}              => {whole milk} 0.0011  1          0.0011   3.9  11
[3] {root vegetables,butter,rice}               => {whole milk} 0.0010  1          0.0010   3.9  10
[4] {root vegetables,whipped/sour cream,flour} => {whole milk} 0.0017  1          0.0017   3.9  17
[5] {butter,soft cheese,domestic eggs}          => {whole milk} 0.0010  1          0.0010   3.9  10
```

```
rules<-apriori(data=Groceries, parameter=list(supp=0.001,conf = 0.15,minle
n=2),
+                  appearance = list(default="rhs",lhs="whole milk"),
+                  control = list(verbose=F))

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

```
    lhs             rhs                support confidence coverage lift count
[1] {whole milk} => {other vegetables} 0.075   0.29       0.26     1.5  736
[2] {whole milk} => {rolls/buns}       0.057   0.22       0.26     1.2  557
[3] {whole milk} => {yogurt}           0.056   0.22       0.26     1.6  551
[4] {whole milk} => {root vegetables}  0.049   0.19       0.26     1.8  481
[5] {whole milk} => {tropical fruit}   0.042   0.17       0.26     1.6  416
```

```
library(arulesViz)

plot(rules,method="graph",interactive=TRUE,shading=NA)
```


RStudio
Graph plot 1
rolls/buns
soda
whole milk
root vegetab
yogurt
tropical fruit
> rules<-sort(rules, decreasing=TRUE,by="confidence")
> inspect(rules[1:5])
lhs rhs support confidence coverage lift count
[1] {whole milk} => {other vegetables} 0.075 0.29 0.26 1.5 736
[2] {whole milk} => {rolls/buns} 0.057 0.22 0.26 1.2 557
[3] {whole milk} => {yogurt} 0.056 0.22 0.26 1.6 551
[4] {whole milk} => {root vegetables} 0.049 0.19 0.26 1.8 481
[5] {whole milk} => {tropical fruit} 0.042 0.17 0.26 1.6 416
> library(arulesviz)
> plot(rules,method="graph",interactive=TRUE,shading=NA)
Warning message:
In plot.rules(rules, method = "graph", interactive = TRUE, shading = NA) :
The parameter interactive is deprecated. Use engine='interactive' instead.
> plot(rules,method="graph",engine = interactive(),shading=NA)
Error in graph_arules(x, measure = measure, shading = shading, control, :
Unknown engine: 'TRUE' Valid engines: 'default', 'igraph', 'interactive', 'graphviz', 'visNetwork', 'htmlwidget'
Groceries Formal class transactions
rules Formal class rules
rules.pruned Formal class rules
subset.matrix Formal class ngCMatrix
redundant Named logi [1:29] TRUE TRUE TRUE TRUE TRUE TRUE ...
Type here to search
23:36
27-10-2020


item frequency (absolute)
0
500
1000
1500
2000
2500
whole milk
other vegetables
rolls/buns
soda
yogurt
bottled water
root vegetables
tropical fruit
shopping bags
sausage
pastry
citrus fruit
bottled beer
newspapers
canned beer
pip fruit
fruit/vegetable juice
whipped/sour cream
brown bread
domestic eggs