Salem Marafi

Market Basket Analysis with R

Posted by Salem on March 19, 2014 105 Comments

Association Rules

Source: Paul's Health Blog

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

So what kind of items are we talking about?
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

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.

A little bit of Math

We already discussed the concept of Items and Item Sets.

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

$$I = \{ i_1,i_2,\ldots,i_n \}$$

Therefore a transaction is represented as follows:

$$t_n = \{ i_j,i_k,\ldots,i_n \}$$

This gives us our rules which are represented as follows:

$$\{ i_1,i_2\} => \{ i_k\}$$

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, but if you are interested in a formal mathematical definition you can find it on wikipedia.

- **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.

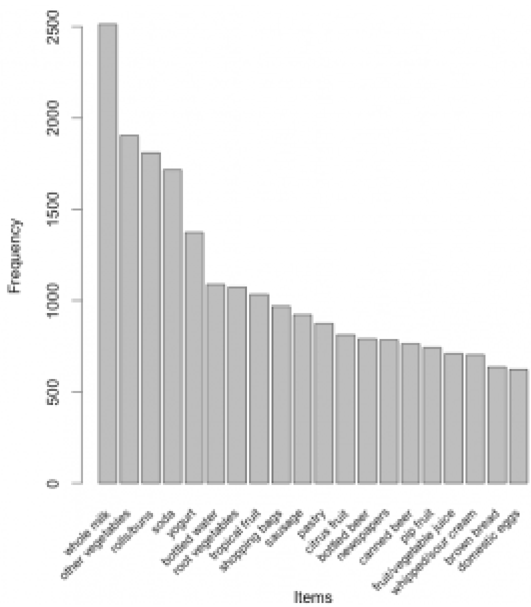
So lets get started by loading up our libraries and data set.

```
# 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 =

# Show the top 5 rules, but only 2 digits
options(digits=2)
inspect(rules[1:5])
```

```
1 {liquor,red/blush wine} => {bottled beer}  0.0019
2 {curd,cereals}          => {whole milk}    0.0010
3 {yogurt,cereals}        => {whole milk}    0.0017
4 {butter,jam}            => {whole milk}    0.0010
5 {soups,bottled beer}    => {whole milk}    0.0011
```

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)
```

```
  lhs                                         rhs
1 {rice,sugar}                             => {whc
2 {canned fish,hygiene articles}           => {whc
3 {root vegetables,butter,rice}            => {whc
4 {root vegetables,whipped/sour cream,flour} => {whc
5 {butter,soft cheese,domestic eggs}       => {whc
```

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
```

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.
               appearance = list(default="lhs",rhs="w
               control = list(verbose=F))
```

The output will look like this:

```
  lhs                                              rh
1 {rice,sugar}                                  => {wh
2 {canned fish,hygiene articles}                => {wh
3 {root vegetables,butter,rice}                 => {wh
4 {root vegetables,whipped/sour cream,flour}    => {wh
5 {butter,soft cheese, domestic eggs}           => {wh
```

Likewise, we can set the left hand side to be “whole milk” and find its antecedents.

Note the following:

- We set the confidence to 0.15 since we get no rules with 0.8
- We set a minimum length of 2 to avoid empty left hand side items

```
rules<-apriori(data=Groceries, parameter=list(supp=0.
               appearance = list(default="rhs",lhs="w
               control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])
```


Now our output looks like this:

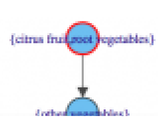
```
  lhs             rhs                  support confiden
1 {whole milk} => {other vegetables}   0.075         0.
2 {whole milk} => {rolls/buns}         0.057         0.
3 {whole milk} => {yogurt}             0.056         0.
4 {whole milk} => {root vegetables}    0.049         0.
5 {whole milk} => {tropical fruit}     0.042         0.
6 {whole milk} => {soda}               0.040         0.
```

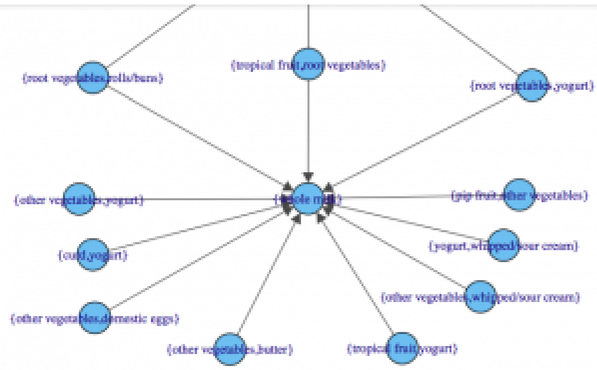
Visualization

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:





References

- Snowplow Market Basket Analysis
- Discovering Knowledge in Data: An Introduction to Data Mining
- RDatamining.com

Resources

- Groceries Dataset

Category: Code

Tags: apriori, arules, arulesViz, association, data, data mining, mining, R

105 Comments on “Market Basket Analysis with R”

Comment navigation

← Older Comments

August 17, 2016 by Arfin

Hello,

I have selected topic – Data mining in MBA using apriori algorithm, for my m.tech cse project. I am very new in data mining

do in it, i mean do i have to make an application for doing MBA using programing or something else.
I know this can be a foolish question!!!!!!!!!
But please guide me to understand about what should i do actually in this.

Waiting for reply…..

Reply

September 9, 2016 by Shashank

Hi Salem, great work !! Are you planning to do market basket analysis using python as well ? Keep up the good work.

Reply

September 30, 2016 by Carlos

Hi, I'm working on a project and I found very useful your code! Thanks!
But I have a question, how am I supposed to do this very analysis with my own data?
When I read a .csv and transform to data frame, I'm in trouble since the itemFrequencyPlot command…

Thanks again

Carlos

Reply

October 25, 2016 by Bárbara Olave

Thanks a lot!

Reply

been processed?using data(Groceries) followed by class(groceries) show as transactions attr (“package”). can anyone share how to start from data in a csv file and pre-process before using arules? thanks in advance

Reply

January 17, 2017 by Mike

The arules documentation indicates that data in the function apriori(data, parameter = NULL, appearance = NULL, control = NULL), must be an object of class transactions.

The documentation also includes information on the transactions class, and how to coerce objects of other classes (e.g. lists) into transactions.

Reply

June 11, 2017 by krupa kapadia

I believe its need to be T/F or 0/1 Matrix. With each item as a column name and transactions as rows

Reply

November 14, 2016 by shantala

Hi Salem,

Thanks for the article. I was working on a similar dataset and your article was very informative.
Very beautifully and neatly explained all the steps and concepts.

Reply

November 16, 2016 by Ged

Reply

Pingback: Affinity Analysis – Big Data Analytic by True

January 23, 2017 by Riz

Awesome explanation… Salem.

Reply

February 9, 2017 by Jasper

Thnx for the code. I found it very usefull for my project. Mainly the interactive graphs are great!

Reply

April 25, 2017 by Rui

Hi Salem, great work. Thank you for your explanation and code.
Can you please give more details about the dataset (groceries.csv)? Where did you get it?

Thanks
Rui

Reply

May 25, 2017 by Gopal Shah,Gujrat,India

Sir,at present i am working on Association Rule.I am confusing that in given data say groceries, i want to find no. of association rules with Support and Confidence.It is possible in R language.Ex. say
Suppprt confidence No. opf association Rule
23 77.56 33
… … ..

Heartily request.

Reply

July 18, 2017 by Tajdar Khan

Very nice post thanks a lot.

Reply

July 19, 2017 by saravana

Hi, Thanks for article, I have a question that How to gather data or query data which is vry appropriate to apriori algorithm. It would be useful if anyone give me some advice and thought.

Thanks.

Reply

July 21, 2017 by Prat

My R does not have groceries data set , could anyone upload it here?

Reply

July 22, 2017 by Salem

Hi Prat,

Sure, I added the attachment under resources. Good luck!

Salem

Reply

writing and code examples are very clear. This really helped me understand market basket analysis. What are your favorite applications of market basket analysis?
Best – Scott

Reply

September 23, 2017 by vishal

Hi Salem,

I have a doubt concerning the code output for checking redundancy. I got an output that said "set of 0 rules", after executing : rules <- rules.pruned. Does it mean there are no redundancies in my dataset?

Thanks in advance and oh, I love reading your blog!

Reply

September 26, 2017 by Nandha

After running the redundancy section of the code, I get the output which says that rules is empty.
How do you fix this?

Reply

November 1, 2017 by Tyler

Would you possibly know why when I use the code above to remove redundant rules, it ends up saying all of my rules are redundant? This leads me to having an empty set of rules.

Thanks!

Reply

before, and previously I just took support and multiplied it by the total transactions count that I fed into the algorithm to get the number of transactions for that rule. I'm running this exercise on a large dataset (hundreds of thousands) and I'm not getting the total transaction count that I expected. Is it possible I'm not getting enough decimal precision back from arules?

Thanks,

Reply

November 9, 2017 by Erick

Also I am using as(rules_1, "data.frame"); to export that dataset to another file. I see now there is a options(digits=x) option. I might try that.

Reply

Pingback: Some Data Processing and Analysis with Python | sandipanweb

Comment navigation

← Older Comments

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

RSS Feed · Log in

Top