Market Basket Analysis with R

listendata.com/2015/12/market-basket-analysis-with-r.html

Market basket analysis explains the combinations of products that frequently co-occur in transactions. For example, people who buy bread and eggs, also tend to buy butter as many of them are planning to make an omelette. Marketing team should target customers who buy bread and eggs with offers on butter, to encourage them to spend more on their shopping basket.

> It is also known as **"Affinity Analysis"** or **"Association Rule Mining"**.

Basics of Market Basket Analysis (MBA)

Example

In a retail shop 400 customers had visited in last month to buy products. It was observed that out of 400 customers, 200 of them bought Product A, 160 of them bought Product B and 100 of them buy both Product A and Product B; We can say 50% (200 out of 400) of the customer buy Product A, 40% (160 out of 400) customers buy Product B and 25% (100 out of 400) buy both Product A and B.

1. Items (Products)

Items are the objects that we are identifying associations between. For an online retailer, each item is a product in the shop. A group of items is an **item set** (set of products).

2. Support

The support of a product or set of products is the fraction of transactions in our data set that contain that product or set of products.

In our example,

1. Support(Product A) = 50%
2. Support(Product B) =40%
3. Support(Product A and B) = 25%

Practical Application -

Support of a product or set of products implies the popularity of the product or set of products in the transaction set. Higher the support, more popular is the product or product bundle.

3. Confidence

Confidence is conditional probability that customer buy product A will also buy product B. Out of 200 customers who bought Product A, 100 bought Product B too.

> Confidence (Product A, Product B) = 100/200 = 50%.

It implies if someone buys Product A, they are 50% likely to buy Product B too.

> Formula - Confidence (A ==> B) = Support (A and B) / Support (A)

Practical Application -

It measures how often items in B appear in transactions that contain A.

4. Lift

If someone buys Product A, what % of chance of buying product B would increase.

> A lift greater than 1 indicates that the presence of A has increased the probability that the product B will occur on this transaction.
> A lift smaller than 1 indicates that the presence of A has decreased the probability that the product B will occur on this transaction

> **Formula** - Lift (A ==> B) = Confidence (A ==> B) / Support (B)

% increase of chance of buying other product(s) = (Lift - 1) * 100

A lift value of 1.25 implies that chance of buying product B (on the right hand side) would increase by 25%.

Practical Application -

Lift indicates the strength of an association rule over the random co-occurrence of Item A and Item B, given their individual support. Lift provides information about the change in probability of Item A in presence of Item B.

Drawback of Confidence
Confidence does not measure if the association between A and B is random or not. Whereas, Lift measures the strength of association between two items.

Association Rule:
Bread => Butter
Prob(Butter) = 0.9
Confidence (Bread => Butter) = Prob(Butter **|** Bread) = 0.75
Lift(Bread => Butter) = 0.8333. It is less than 1, which means negative association between them.

Desired Outcome

> In market basket analysis, we pick rules with a lift of more than one because the presence of one product increases the probability of the other product(s) on the same transaction. Rules with higher confidence are ones where the probability of an item appearing on the RHS is high given the presence of the items on the LHS.

Data Preparation

I. Continuous variables need to be binned / discretized

```
dat$age2 = discretize(dat$age, method = "frequency", 3)
```

II. Convert raw and demographic data to transaction class

```
df <- data.frame(
  age   = as.factor(c(6, 6, 8, 8, NA, 9, 16)),
  grade = as.factor(c("A", "C", "C", "C", "F", NA, "C")),
  pass  = c(TRUE, TRUE, TRUE, TRUE, FALSE, TRUE, TRUE))
```

```
## convert to transaction
trans3 <- as(df, "transactions")
summary(trans3)
inspect(trans3)
```

Transactions Detail

1. 7 rows (elements/itemsets/transactions) and
2. 8 columns (items) and a density of 0.3214286

8 columns (items) - 8 Unique products or categories

Remove Unnecessary Rules

When a rule (let's say rule A) is a super rule of another rule (rule B) and the rule A has the same or a lower lift, the former rule (rule A) is considered to be redundant.

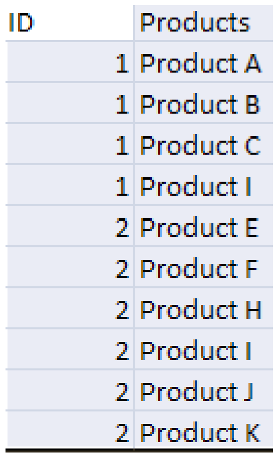
Let's say two rules - {A,B,C} → {D} and {A,B} → {D}.
{A,B,C} → {D} is redundant as lift value of both the rules are almost same.

Example - milk => bread (12% support , 85% confidence, 1.2 lift)
milk, flavor = Chocolate => bread (1% support, 84% confidence, 1.15 lift)

In this case, we are not interested if some % of milk is chocolate milk as it contains fewer % of transactions.

Sample Data

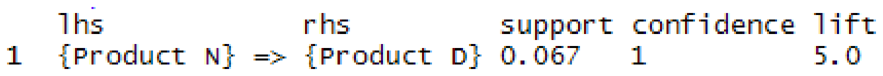
Download Data Set



ID	Products
1	Product A
1	Product B
1	Product C
1	Product I
2	Product E
2	Product F
2	Product H
2	Product I
2	Product J
2	Product K

Market Basket Analysis

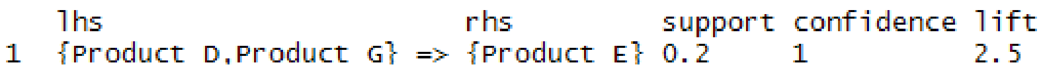
Output



```
    lhs              rhs          support confidence lift
1   {Product N} => {Product D} 0.067   1          5.0
```

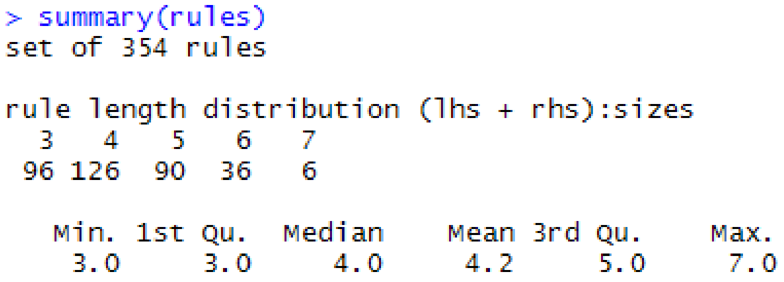
Confidence value of 1 indicates If someone buys Product N, they are 100% likely to buy Product D.

The **support value** of 0.067 indicates that 6.7% of the transaction in the data involve Product N purchases. Hence the support indicates goodness of the choice of rule and confidence indicates the correctness of the rule.



```
    lhs                        rhs          support confidence lift
1   {Product D,Product G} => {Product E} 0.2     1          2.5
```

Confidence value of 1 indicates If someone buys Product D and G, they are 100% likely to buy Product E. The support 0.2 indicates that 20% of the transaction in the data involve both Product D and G purchases. Hence the support indicates goodness of the choice of rule and confidence indicates the correctness of the rule.



```
> summary(rules)
set of 354 rules

rule length distribution (lhs + rhs):sizes
  3   4   5   6   7 
 96 126  90  36   6 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    3.0     3.0     4.0     4.2     5.0     7.0 
```

- The number of rules generated: 354
- The distribution of rules by length: Most rules are 4 items long

R Code : Market Basket Analysis

```
# Read CSV data file
mydata = read.csv("C:\\Users\\Deepanshu\\Desktop\\mba.csv")

# See first 10 observations
head(mydata, n=10)

# Split data
dt <- split(mydata$Products, mydata$ID)

# Loading arules package
if(!require(arules)) install.packages("arules")

# Convert data to transaction level
dt2 = as(dt,"transactions")
summary(dt2)
```

```
inspect(dt2)
# Most Frequent Items
itemFrequency(dt2, type = "relative")
itemFrequencyPlot(dt2,topN = 5)


# aggregated data
rules = apriori(dt2, parameter=list(support=0.005, confidence=0.8))
rules = apriori(dt2, parameter=list(support=0.005, confidence=0.8, minlen = 3))
rules = apriori(dt2, parameter=list(support=0.005, confidence=0.8, maxlen = 4))

#Convert rules into data frame
rules3 = as(rules, "data.frame")
write(rules, "C:\\Users\\Deepanshu\\Downloads\\rules.csv", sep=",")

# Show only particular product rules
inspect( subset( rules, subset = rhs %pin% "Product H" ))

# Show the top 10 rules
options(digits=2)
inspect(rules[1:10])

# Get Summary Information
summary(rules)

# Sort by Lift
rules<-sort(rules, by="lift", decreasing=TRUE)

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

#Clean Rules
rules3$rules=gsub("\\{", "", rules3$rules)
rules3$rules=gsub("\\}", "", rules3$rules)
rules3$rules=gsub("\"", "", rules3$rules)
#Split the rule
library(splitstackshape)
Rules4=cSplit(rules3, "rules","=>")
names(Rules4)[names(Rules4) == 'rules_1'] <- 'LHS'
Rules5=cSplit(Rules4, "LHS",",")
Rules6=subset(Rules5, select= -c(rules_2))
names(Rules6)[names(Rules6) == 'rules_3'] <- 'RHS'
```

```
# What are customers likely to buy before they purchase "Product A"
rules<-apriori(data=dt, parameter=list(supp=0.001,conf = 0.8),
            appearance = list(default="lhs",rhs="Product A"),
            control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])

# What are customers likely to buy if they purchased "Product A"
rules<-apriori(data=dt, parameter=list(supp=0.001,conf = 0.8),
            appearance = list(default="rhs",lhs="Product A"),
            control = list(verbose=F))
rules<-sort(rules, decreasing=TRUE,by="confidence")
inspect(rules[1:5])
```