

# CUSTOMER SEGMENTATION AND MARKET BASKET ANALYSIS FOR AN ONLINE RETAIL

BHAVYA SREE BINDELA

A20448208

GROUP: 320

# INTRODUCTION

**Customer segmentation** is the process of dividing the customers into different groups based on their behavior with the company. Using customer segments, the company can market their strategies to each group effectively.

**Market Basket Analysis** is a technique to identify the strength of association between pairs of products that are purchased together.

This technique is based on a concept that if a person buys a product A, how likely the person will buy a product B.

Customer Segmentation and Market Basket Analysis is to explore different types of customers and their granular purchase behavior patterns.

# DATA SET

- Dataset contains the Business data, and this data set belongs to a UK-based registered non-store online retail.
- This Data set contains all the transactions data occurred between 01/12/2010 and 09/12/2011.

Number of Instances : 541909

[Online Retail Dataset](#)

Attribute	Description	Data Type
InvoiceNo	Invoice number	Character
StockCode	Product Code	Character
Description	Product Name	Character
Quantity	Quantity of each product per transaction	Number
InvoiceDate	Invoice Date and Time	Number
UnitPrice	Price of the Product per unit	Number
CustomerID	Customer Number	Number
Country	Country Name	Character



## RESEARCH PROBLEMS

Identify the customers into different groups based on their behavior with the online retail. Identify the Loyal Customers.

Identify best 10 Association rules for the products purchased in the online retail.

# PROPOSED SOLUTIONS

**Identify the customers into different groups based on their behavior with the online retail.**

RFM Analysis is the data-driven customer segmentation technique.

Recency : Number of days from the last purchase

Frequency : Number of visits

Monetary : Total amount spent

we use both K-means Clustering and Hierarchical clustering algorithms on RFM data for customer segmentation.

**Identify best 10 Association rules for the products purchased in the online retail.**

Group the products based on InvoiceNo and apply Apriori algorithm to find the association rules for the products.

# Load the Data

```
# Load retail data into R session
install.packages("openxlsx")
library(openxlsx)

> data=read.xlsx("Online Retail.xlsx", 1)
>
> # Structure of the data
> str(data)
'data.frame': 541909 obs. of 8 variables:
 $ InvoiceNo : chr "536365" "536365" "536365" "536365" ...
 $ StockCode  : chr "85123A" "71053" "84406B" "84029G" ...
 $ Description: chr "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUPID HEARTS COAT HANGER" "KNITTED UNION FLAG HOT WATER BOTTLE" ...
 $ Quantity   : num 6 6 8 6 6 2 6 6 6 32 ...
 $ InvoiceDate: num 40513 40513 40513 40513 ...
 $ UnitPrice  : num 2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...
 $ CustomerID: num 17850 17850 17850 17850 17850 ...
 $ Country    : chr "United Kingdom" "United Kingdom" "United Kingdom" "United Kingdom" ...
> # Change the type of InvoiceDate
> data$InvoiceDate <- as.POSIXct(data$InvoiceDate * (60*60*24),
+                                     origin="1899-12-30", tz="GMT")
> head(data)
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom
2	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
3	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom
4	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
5	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom
6	536365	22752	SET 7 BABUSHKA NESTING BOXES	2	2010-12-01 08:26:00	7.65	17850	United Kingdom

- Observations = 541909
- Variables = 8

Change InvoiceDate type from  
Number to Date

# Pre-Processing the Data

```
> # Check for the sum of Null values in all columns
> colSums(is.na(data))
  InvoiceNo StockCode Description   Quantity InvoiceDate   UnitPrice CustomerID   Country
      0          0        1454         0          0          0       135080         0
...
> # Omit the rows with Null values in CustomerID.
> library(tidyverse)
> retail_data <- data %>%
+   drop_na(CustomerID)
>
> # Check for Null values
> colSums(is.na(retail_data))
  InvoiceNo StockCode Description   Quantity InvoiceDate   UnitPrice CustomerID   Country
      0          0        1454         0          0          0       135080         0
`-> # Summary of retail data
> summary(retail_data)
  InvoiceNo     StockCode     Description     Quantity     InvoiceDate     UnitPrice     CustomerID     Country
Length:406829 Length:406829 Length:406829 Min. :-80995.00 Min. :2010-12-01 08:26:00 Min. : 0.00 Min. :12346 Length:406829
Class :character Class :character Class :character 1st Qu.: 2.00 1st Qu.:2011-04-06 15:02:00 1st Qu.: 1.25 1st Qu.:13953 Class :character
Mode :character Mode :character Mode :character Median : 5.00 Median :2011-07-31 11:48:00 Median : 1.95 Median :15152 Mode :character
                                         Mean : 12.06 Mean :2011-07-10 16:30:57 Mean : 3.46 Mean :15288
                                         3rd Qu.: 12.00 3rd Qu.:2011-10-20 13:06:00 3rd Qu.: 3.75 3rd Qu.:16791
                                         Max. : 80995.00 Max. :2011-12-09 12:50:00 Max. :38970.00 Max. :18287
`-
```

## Handling Missing values

### Remove return Transactions - Summary of data

```
> ## Remove unrelated and return transactions
> retail_data <- retail_data[retail_data$Quantity >= 0,]
> retail_data <- retail_data[retail_data$UnitPrice > 0,]
>
> summary(retail_data)
  InvoiceNo     StockCode     Description     Quantity     InvoiceDate     UnitPrice     CustomerID     Country
Length:397884 Length:397884 Length:397884 Min. : 1.00 Min. :2010-12-01 08:26:00 Min. : 0.001 Min. :12346 Length:397884
Class :character Class :character Class :character 1st Qu.: 2.00 1st Qu.:2011-04-07 11:12:00 1st Qu.: 1.250 1st Qu.:13969 Class :character
Mode :character Mode :character Mode :character Median : 6.00 Median :2011-07-31 14:39:00 Median : 1.950 Median :15159 Mode :character
                                         Mean : 12.99 Mean :2011-07-10 23:41:23 Mean : 3.116 Mean :15294
                                         3rd Qu.: 12.00 3rd Qu.:2011-10-20 14:33:00 3rd Qu.: 3.750 3rd Qu.:16795
                                         Max. :80995.00 Max. :2011-12-09 12:50:00 Max. :8142.750 Max. :18287
`-
```

# Customer Segmentation – Calculate RFM

**Recency** : Number of days since last Purchase.

**Frequency**: Number of visits.

**Monetary**: Total amount spent.

```
> ## Recency
> max_date <- max(retail_data$InvoiceDate)
> max_date
[1] "2011-12-09 12:49:59 GMT"
>
> # Difference between the InvoiceDate and max_date
> retail_data$time_delta <- as.numeric(difftime(as.Date(max_date), as.Date(retail_data$InvoiceDate), units="days"))
> head(retail_data$time_delta, 2)
[1] 373 373
> # minimum time_delta (maxdate - recent InvoiceDate) of every customer
> Recency <- retail_data %>%
+   select(CustomerID, time_delta) %>%
+   group_by(CustomerID) %>%
+   slice(which.min(time_delta))
```

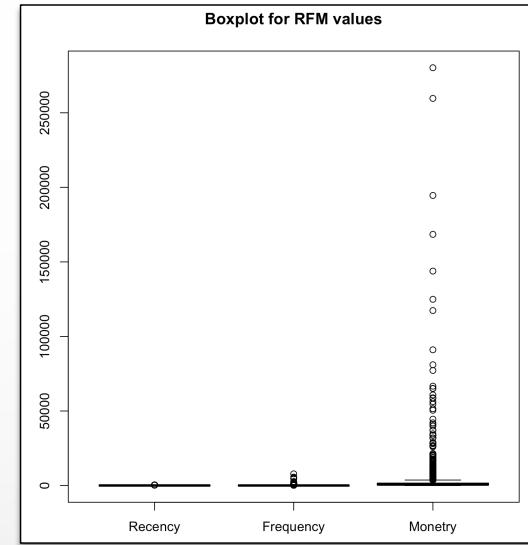
```
> # Frequency
> # Group the data by customerId and get the count of invoiceNo
> Frequency <- retail_data %>%
+   select(CustomerID, InvoiceNo) %>%
+   group_by(CustomerID) %>%
+   summarize(count = n())
`summarise()` ungrouping output (override with `.`groups` argument)
>
> ## Monetry
> # Group the data by CustomerID and get the sum of amount spent
> # Calculate total price i.e., quantity*unitprice
> retail_data$Total_Price <- (retail_data$Quantity)*(retail_data$UnitPrice)
>
> Monetry <- retail_data %>%
+   select(CustomerID, Total_Price) %>%
+   group_by(CustomerID) %>%
+   summarize(Total = sum(Total_Price))
`summarise()` ungrouping output (override with `.`groups` argument)
```

```
> RF <- merge(Recency, Frequency, by='CustomerID')
> RFM <- merge(RF, Monetry, by='CustomerID')
> colnames(RFM) <- c('CustomerID', 'Recency', 'Frequency', 'Monetry')
>
> str(RFM)
'data.frame': 4338 obs. of  4 variables:
$ CustomerID: num  12346 12347 12348 12349 12350 ...
$ Recency    : num  325 2 75 18 310 36 204 232 214 22 ...
$ Frequency  : int  1 182 31 73 17 85 4 58 13 59 ...
$ Monetry    : num  77183.60 4310.00 1797.24 1757.55 334.40 ...
> head(RFM)
  CustomerID Recency Frequency Monetry
1      12346     325          1 77183.60
2      12347      2         182  4310.00
3      12348      75         31  1797.24
4      12349     18          73  1757.55
5      12350     310         17   334.40
6      12352     36          85  2506.04
> summary(RFM)
  CustomerID        Recency       Frequency       Monetry
Min.   :12346   Min.   : 0.00   Min.   : 1.00   Min.   :  3.75
1st Qu.:13813   1st Qu.:17.00   1st Qu.: 17.00   1st Qu.: 307.42
Median :15300   Median :50.00   Median : 41.00   Median : 674.48
Mean   :15300   Mean   :92.06   Mean   : 91.72   Mean   :2054.27
3rd Qu.:16779   3rd Qu.:141.75  3rd Qu.:100.00  3rd Qu.:1661.74
Max.   :18287   Max.   :373.00   Max.   :7847.00  Max.   :280206.02
```

# RFM data Pre-Processing

```
> ## RFM data re-processing #####
>
> # Check for outliers
> boxplot(RFM[-1], main="Boxplot for RFM values")
>
> # Remove outliers
> Recency_outliers <- boxplot(RFM$Recency, plot=FALSE)$out
> RFM_data <- RFM[ -which(RFM$Recency %in% Recency_outliers), ]
>
> Frequency_outliers <- boxplot(RFM$Frequency, plot=FALSE)$out
> RFM_data <- RFM[ -which(RFM$Frequency %in% Frequency_outliers), ]
>
> Monetory_outliers <- boxplot(RFM$Monetory, plot=FALSE)$out
> RFM_data <- RFM[ -which(RFM$Monetory %in% Monetory_outliers), ]
>
> # Structure of RFM_data
> str(RFM_data)
'data.frame': 3911 obs. of 4 variables:
 $ CustomerID: num 12348 12349 12350 12352 12353 ...
 $ Recency    : num 75 18 310 36 204 232 214 22 1 52 ...
 $ Frequency   : int 31 73 17 85 4 58 13 59 19 129 ...
 $ Monetory    : num 1797 1758 334 2506 89 ...
```

## Removing Outliers



```
> # Scale the Numerical Variables
> vars <- c('Recency', 'Frequency', 'Monetory')
> Clust_Data <- data.frame(RFM_data[-1])
> Clust_Data[vars] <- lapply(RFM_data[vars], scale)
> head(Clust_Data)
      Recency   Frequency   Monetory
3 -0.2403706 -0.39472592  1.0990271
4 -0.8015356  0.14852823  1.0511234
5  2.0732046 -0.57581063 -0.6665412
6 -0.6243256  0.30374370  1.9545102
7  1.0296345 -0.74396072 -0.9627256
8  1.3052945 -0.04549111  0.2326333
```

## Scale the data

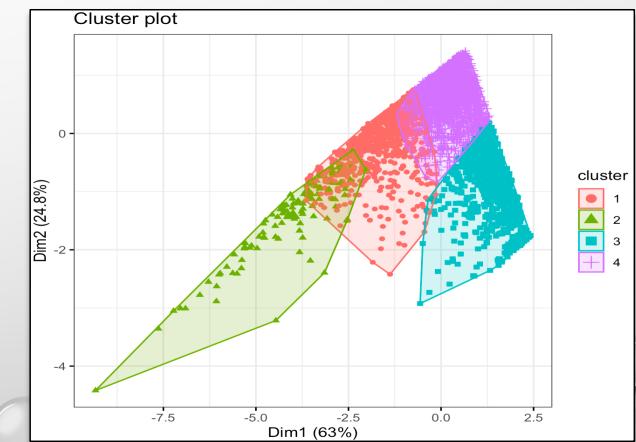
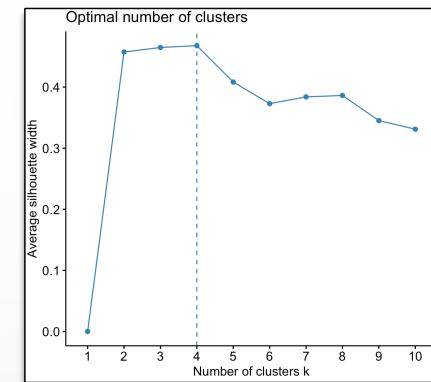
# K-Means Clustering – K with Silhouette Method

## Silhouette Method

- Silhouette Score is a measure of how similar an object is to its own cluster. Ranges from -1 to +1

```
> # Silhouette Score  
> set.seed(123)  
> fviz_nbclust(Clust_Data, kmeans, method = "silhouette")  
> # From the silhouette method, the optimal number of clusters is 4  
  
> # Compute K-Means with k as 4  
> set.seed(101)  
> KM_Modell <- kmeans(Clust_Data, 4, nstart = 25)  
> # nstart=25 will generate 25 random centroids and choose the best one for algorithm.  
>  
> fviz_cluster(KM_Modell, data=Clust_Data,  
+               geom="point",  
+               ellipse.type = "convex",  
+               ggtheme = theme_bw())  
>  
> # Size of Clusters  
> KM_Modell$size  
[1] 786 119 985 2021  
> # Center of clusters  
> KM_Modell$centers  
   Recency Frequency Monetary  
1 -0.5719567 0.7426322 1.4600967  
2 -0.7496632 4.0453278 1.8326463  
3  1.5205852 -0.4883279 -0.6109894  
4 -0.4745218 -0.2890153 -0.3779794
```

K = 4



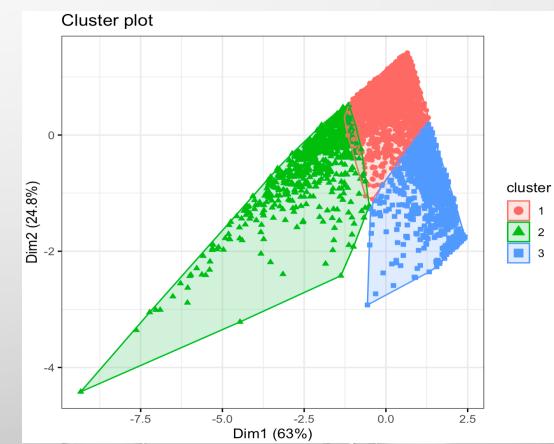
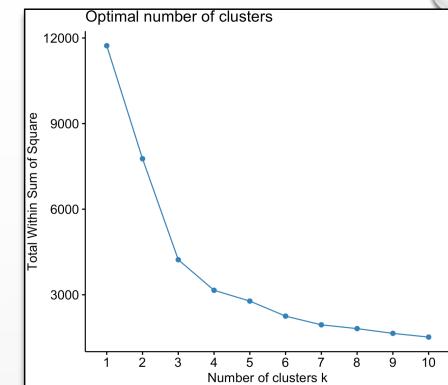
# K-Means Clustering – K with Elbow Method

## Elbow Method

- Elbow Method gives the total Within Sum of Squared Errors for different values of K.

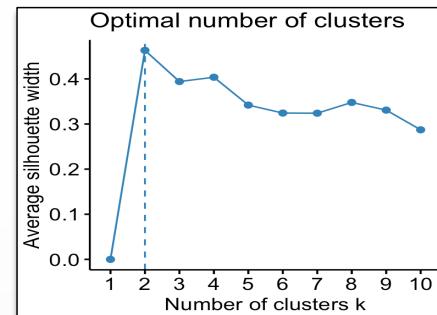
```
> # Elbow method
> set.seed(123)
> fviz_nbclust(Clust_Data, kmeans, method='wss')
> # From the elbow method, at k=3, the total within sum of squares will be less.
```

```
> # Compute K-Means with k as 3
> set.seed(101)
> KM_Model2 <- kmeans(Clust_Data, 3, nstart = 25)
> # nstart=25 will generate 25 random centroids and choose the best one for algorithm.
>
> fviz_cluster(KM_Model2, data=Clust_Data,
+               geom="point",
+               ellipse.type = "convex",
+               ggtheme = theme_bw())
>
> # Size of Clusters
> KM_Model2$size
[1] 2154 758 999
> # Center of clusters
> KM_Model2$centers
  Recency Frequency Monetary
1 -0.4781634 -0.2581498 -0.3034692
2 -0.6310794  1.3699627  1.6535819
3  1.5098321 -0.4828600 -0.6003427
```

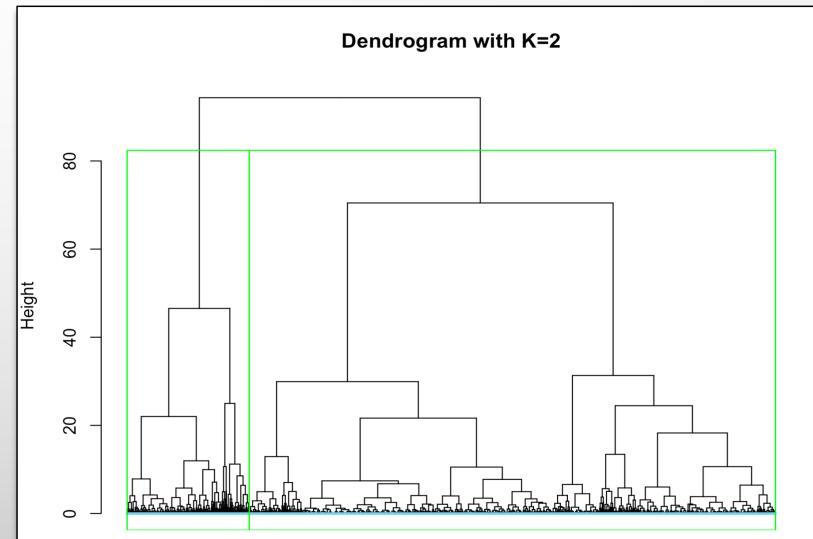


# Hierarchical Clustering – K with Silhouette Method

```
> ## Hierarchical Clustering #####
>
> # Silhouette Score
> set.seed(123)
> fviz_nbclust(Clust_Data, hcut, method = "silhouette")
> # From the silhouette method, the optimal number of clusters is 2
>
> # calculate distance between vectors of Clust_Data
> d <- dist(Clust_Data, method='euclidean')
> HC_modell <- hclust(d, method='ward.D2')
>
> # Dendrogram, customizing the plot to remove labels
> HC_modell_d <- as.dendrogram(HC_modell)
> nodePar <- list(lab.cex = 0.6, pch = c(NA, 19),
+                  cex = 0.2, col = "skyblue")
> plot(HC_modell_d, ylab = "Height", nodePar = nodePar, leaflab = "none",
+       main = "Dendrogram with K=2")
> rect.hclust(HC_modell, k=2, border="green")
>
> groups <- cutree(HC_modell, k=2)
> HC_data1 <- data.frame(Clust_Data, groups)
> head(HC_data1, 1)
      Recency Frequency Monetary groups
3 -0.2403706 -0.3947259 1.099027     1
>
> # size of groups
> HC_data1 %>%
+   group_by(groups) %>%
+   summarise(count=n())
#> `summarise()` ungrouping output (override with `.`groups` argument)
#> # A tibble: 2 x 2
#>   groups count
#>   <int> <int>
1       1  3174
2       2    737
```

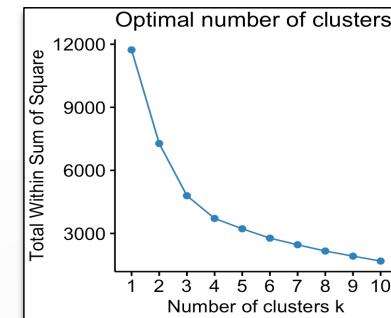


Considering  
K = 2

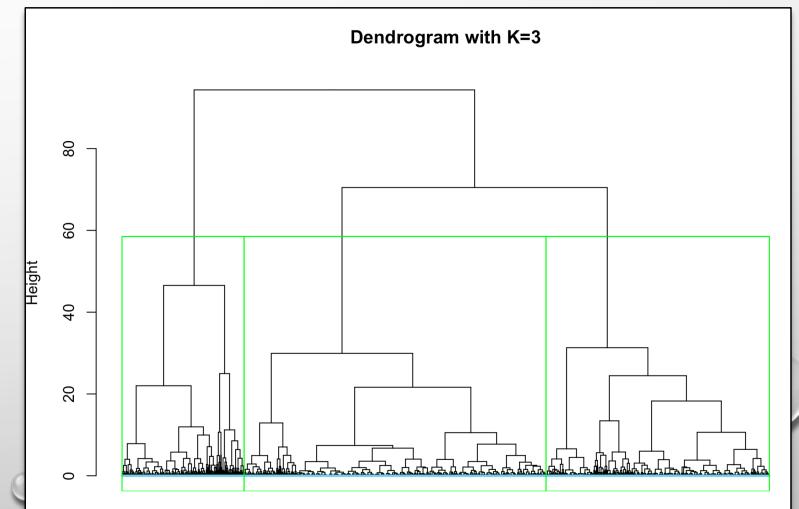


# Hierarchical Clustering - K with Elbow method

```
> # Elbow method
> set.seed(123)
> fviz_nbclust(Clust_Data, hcut, method='wss')
> # From the elbow method, at k=3, the total within sum of squares will be less.
>
> HC_model2 <- hclust(d, method='ward.D2')
>
> # Dendrogram, customizing the plot to remove labels
> HC_modell_d <- as.dendrogram(HC_modell)
> nodePar <- list(lab.cex = 0.6, pch = c(NA, 19),
+                   cex = 0.2, col = "skyblue")
> plot(HC_modell_d, ylab = "Height", nodePar = nodePar, leaflab = "none",
+       main = "Dendrogram with K=3")
> rect.hclust(HC_modell, k=3, border="green")
>
> groups <- cutree(HC_model2, k=3)
> HC_data2 <- data.frame(Clust_Data, groups)
> head(HC_data2, 1)
  Recency Frequency Monetary groups
3 -0.2403706 -0.3947259 1.099027      1
>
> # size of groups
> HC_data2 %>%
+   group_by(groups) %>%
+   summarise(count=n())
#> summarise() ungrouping output (override with `.`groups` argument)
#> A tibble: 3 x 2
  groups count
  <int> <int>
1      1    1349
2      2     737
3      3    1825
```



Considering  
K = 3



# Clustering - Evaluation

## Using Internal validation measures

**Connectivity:** Degree of connectedness of the clusters determined by KNN. This should be minimum.

**Silhouette width:** Measures the compactness of the clusters.

$$S_i = \frac{(b_i - a_i)}{\max(a_i, b_i)}$$

$a_i$  is the average distance of object i from all other objects of same cluster.

$b_i$  is the average distance of object i from all other objects of different cluster.

The range of  $S_i$  is [-1,1].

**Dunn Index:** Measures the separation of clusters.

$$D = \frac{\min. separation}{\max. diameter}$$

This should be maximum.

```
> ## Clustering Validation
> library(clValid)
> # Internal
> internal <- clValid(as.matrix(Clust_Data), nClust = 2:4,
+ +                         clMethods = c("hierarchical", "kmeans"),
+ +                         validation = "internal")

The number of items to be clustered is larger than 'maxitems'
The memory and time required may be excessive, do you wish to continue?
(y to continue, any other character to exit)
y
> summary(internal)

Clustering Methods:
  hierarchical kmeans

Cluster sizes:
  2 3 4

Validation Measures:
                                     2       3       4
hierarchical Connectivity  3.0290 15.8480 26.2401
                           Dunn     0.2803  0.0540  0.0540
                           Silhouette  0.8232  0.6360  0.4251
kmeans      Connectivity 153.5032 255.6778 276.3798
                           Dunn     0.0024  0.0031  0.0041
                           Silhouette  0.4574  0.4648  0.4688

Optimal Scores:
          Score Method Clusters
Connectivity 3.0290 hierarchical 2
Dunn         0.2803 hierarchical 2
Silhouette   0.8232 hierarchical 2
  ...
```

# Clustering - Evaluation

## Manual evaluation using centers of Clusters

### Centers of 4 clusters with K-Means

```
> # Centers of clusters from K-means Model with K= 4
> KM_Model1$centers
   Recency Frequency Monetary
1 -0.5719567  0.7426322 1.4600967
2 -0.7496632  4.0453278 1.8326463
3  1.5205852 -0.4883279 -0.6109894
4 -0.4745218 -0.2890153 -0.3779794
```

### Centers of 2 clusters with Hierarchical

```
> # Centers of clusters from Hierarchical clustering with K = 2
> apply (Clust_Data, 2, function (x) tapply (x, groups2, mean))
   Recency Frequency Monetary
1  0.1508643 -0.2854663 -0.4000176
2 -0.6497195  1.2294028  1.7227353
```

### Centers of 3 clusters with K-Means

```
> # Centers of clusters from K-means Model with K= 3
> KM_Model2$centers
   Recency Frequency Monetary
1 -0.4781634 -0.2581498 -0.3034692
2 -0.6310794  1.3699627  1.6535819
3  1.5098321 -0.4828600 -0.6003427
```

### Centers of 3 clusters with Hierarchical

```
> # Centers of clusters from Hierarchical clustering with K = 3
> apply (Clust_Data, 2, function (x) tapply (x, groups3, mean))
   Recency Frequency Monetary
1  1.1628303 -0.4485738 -0.4882853
2 -0.6497195  1.2294028  1.7227353
3 -0.5971588 -0.1649007 -0.3347721
```

# Customer Segmentation - Conclusion

```
> customers_1 <- head(RFM_data$CustomerID[RFM_data$Cluster == 1])
> filter(RFM_data, RFM_data$CustomerID %in% customers_1)
  CustomerID Recency Frequency Monetrey Cluster
1      12348      75          31   1797.24      1
2      12350     310          17   334.40      1
3      12353     204          4    89.00      1
4      12354     232          58  1079.40      1
5      12355     214          13   459.40      1
6      12358       1          19  1168.06      1
```

Cluster 1 is of the customers who are either with More Recency or Less Frequency or Less Monetary in the online Retail.

```
> customers_2 <- head(RFM_data$CustomerID[RFM_data$Cluster == 2])
> filter(RFM_data, RFM_data$CustomerID %in% customers_2)
  CustomerID Recency Frequency Monetrey Cluster
1      12349      18          73   1757.55      2
2      12352      36          85  2506.04      2
3      12356      22          59  2811.43      2
4      12360      52          129  2662.06      2
5      12370      51          167  3545.69      2
6      12371      44          63  1887.96      2
```

Cluster 2 is of the customers who are Recent, Frequent and spent more amount in the online Retail.

Customers in Cluster 2 are **Loyal Customers**.

# Market Basket Analysis – Association rules

## Pre-Processing the data

```
> # Pre-Processing #####
> colSums(is.na(data))
  InvoiceNo StockCode Description  Quantity InvoiceDate  UnitPrice CustomerID  Country
      0          0        1454         0            0          0       135080         0

> library(tidyr)
>
> # Remove rows with Description as Null
> rules_data <- data %>%
+   drop_na(Description)
> #remove the unrelated and return transactions
> rules_data <- rules_data[rules_data$Quantity >= 0,]
> rules_data <- rules_data[rules_data$UnitPrice > 0,]
>
> # str(rules_data)
>
> length(unique(rules_data$StockCode))
[1] 3922
> length(unique(rules_data$Description))
[1] 4026
>
> head(rules_data$Description, 3)
[1] "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN"           "CREAM CUPID HEARTS COAT HANGER"
>
> # trim the Description
> library(stringr)
> rules_data$Description <- str_trim(rules_data$Description)
> # Check for Special characters
> rules_data[grep('[!#$%&*+;<=>?@[]^~|~]', rules_data$Description),]
[1] InvoiceNo StockCode Description Quantity InvoiceDate UnitPrice CustomerID Country
<0 rows> (or 0-length row.names)
```

Handle Missing values

- Removed return transactions
- Trim the description
- Check for special characters in the description
- Check for the number of unique stock codes and unique descriptions

# Market Basket Analysis – Pre-Processing the data

```
> # Check for Description in lower case
> head(rules_data[grep('[:lower:]', rules_data$Description),])
   InvoiceNo StockCode          Description      Quantity InvoiceDate UnitPrice CustomerID Country
2060      536557    22686 FRENCH BLUE METAL DOOR SIGN No       1 2010-12-01 14:41:00     1.25    17841 United Kingdom
2231      536569            M                         Manual       1 2010-12-01 15:35:00     1.25    16274 United Kingdom
2242      536569            M                         Manual       1 2010-12-01 15:35:00    18.95    16274 United Kingdom
2558      536592    21594 Dr. Jam's Arouzer Stress Ball       1 2010-12-01 17:06:00     4.21        NA United Kingdom
3296      536620    22965 3 TRADITIONAL BISCUIT CUTTERS SET       6 2010-12-02 10:27:00     2.10    14135 United Kingdom
3400      536626    22965 3 TRADITIONAL BISCUIT CUTTERS SET       6 2010-12-02 10:46:00     2.10    13418 United Kingdom
> # Remove the data with description as Manual
> rules_data <- rules_data[rules_data$Description != 'Manual',]
> # Identify the StockCodes having two descriptions and use the descriptions to process the multi descriptions
> df <- data.frame(rules_data %>%
+   select(StockCode, Description) %>%
+   group_by(StockCode, Description) %>%
+   count(StockCode, Description))
>
> x <- df$StockCode[duplicated(df$StockCode)]
> sort(df$Description[df$StockCode %in% x])
[1] "16 PC CUTLERY SET PANTRY DESIGN"      "16 PIECE CUTLERY SET PANTRY DESIGN"      "3 DRAWER ANTIQUE WHITE WOOD CABINET"      "3 TRADITIONAL BISCUIT CUTTERS SET"
[5] "3 TRADITIONAL COOKIE CUTTERS SET"      "36 DOILIES DOLLY GIRL"                  "36 DOILIES VINTAGE CHRISTMAS"             "50'S CHRISTMAS GIFT BAG LARGE"
[9] "60 CAKE CASES VINTAGE CHRISTMAS"       "70'S ALPHABET WALL ART"                 "72 CAKE CASES VINTAGE CHRISTMAS"           "ACRYLIC JEWEL SNOWFLAKE,PINK"
[13] "ADULT APRON APPLE DELIGHT"              "ALUMINIUM HEART"                      "ALUMINIUM STAMPED HEART"                  "ANIMALS AND NATURE WALL ART"

```

- Handle data with no valid Description
- Handle data with typo errors in Description

```
> rules_data$Description <- ifelse(rules_data$Description == "WALL ART BICYCLE SAFTEY", "WALL ART BICYCLE SAFETY",
+ ifelse(rules_data$Description == "WALL ART,ONLY ONE PERSON", "WALL ART ONLY ONE PERSON",
+ ifelse(rules_data$Description == "WHITE WIRE PLANT POT HOLDER", "WHITE HEARTS WIRE PLANT POT HOLDER",
+ ifelse(rules_data$Description == "WHITE METAL LANTERN", "WHITE MOROCCAN METAL LANTERN",
+ ifelse(rules_data$Description == "WOODLAND MINI RUCKSACK", "WOODLAND MINI BACKPACK",
+ ifelse(rules_data$Description == "WRAP RED DOILEY", "WRAP RED VINTAGE DOILY" ,
+ ifelse(rules_data$Description == "WRAP VINTAGE PETALS DESIGN", "WRAP VINTAGE LEAF DESIGN",
+ ifelse(rules_data$Description == "ZINC PLANT POT HOLDER", "ZINC HEARTS PLANT POT HOLDER",
+ ifelse(rules_data$Description == "ZINC STAR T-LIGHT HOLDER", "ZINC STAR T-LIGHT HOLDER",
+ ifelse(rules_data$Description == "ZINC T-LIGHT HOLDER STAR LARGE", "ZINC T-LIGHT HOLDER STARS LARGE", rules_data$Description )))))))))
```

## Market Basket Analysis – Format the data

```

> # Use ddply function to get all the items bought together in a row separated by ,.
> # To get the items bought together, get Description by grouping the data on InvoiceNo.
> Association_data <- ddply(rules_data,c("InvoiceNo"),
+                               function(x) paste(x$Description,
+                                                 collapse = ","))
> str(Association_data)
'data.frame': 19868 obs. of 2 variables:
 $ InvoiceNo: chr "536365" "536366" "536367" "536368" ...
 $ V1       : chr "WHITE HANGING HEART T-LIGHT HOLDER,WHITE MOROCCAN METAL LANTERN,CREAM CUPID HEARTS COAT HANGER,RED POLKA DOT" "ASSORTED COLOUR BIRD ORNAMENT,POPPIY'S PLAYHOUSE BEDROOM,POPPIY'S PLAYHOUSE KITCHEN,FELTCRAFT PRINACK PARIS FASHION,YELLOW COAT RACK PARIS FASHION,BLUE COAT RACK PARIS FASHION" ...
>
> Association_data$InvoiceNo <- NULL
> colnames(Association_data) <- c("items")
> Association_data$items <- as.factor(Association_data$items)
>
> str(Association_data)
'data.frame': 19868 obs. of 1 variable:
 $ items: Factor w/ 18926 levels "10 COLOUR SPACEBOY PEN,BLUE CALCULATOR RULER,ASSORTED TUTTI FRUTTI PEN,PAINT YOUR 7575 1380 685 10606 6441 17919 17121 ...
>
> # To view the transactional_data in the better format
> write.csv(Association_data, "transactional_data.csv", quote=FALSE, row.names = FALSE)

```

# Association Rules – Apriori Algorithm

```
> transactional_data <- read.transactions('transactional_data.csv', format = 'basket', sep=',', quote="")  
Warning message:  
In asMethod(object) : removing duplicated items in transactions
```

```
> # Association rules  
> rules.1 <- apriori(transactional_data, parameter=list(support=0.1, confidence = 0.1))  
Apriori  
  
Parameter specification:  
confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  
0.1 0.1 1 none FALSE TRUE 5 0.1 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: 1986  
  
set item appearances ...[0 item(s)] done [0.00s].  
set transactions ...[3938 item(s), 19869 transaction(s)] done [0.40s].  
sorting and recoding items ... [3 item(s)] done [0.00s].  
creating transaction tree ... done [0.00s].  
checking subsets of size 1 2 done [0.00s].  
writing ... [3 rule(s)] done [0.00s].  
creating S4 object ... done [0.00s].
```

With min.confidence 0.1 and min.support 0.1  
- 3 Association Rules

```
> rules.05 <- apriori(transactional_data, parameter=list(support=0.05, confidence = 0.1))  
Apriori  
  
Parameter specification:  
confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  
0.1 0.1 1 none FALSE TRUE 5 0.05 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: 993  
  
set item appearances ...[0 item(s)] done [0.00s].  
set transactions ...[3938 item(s), 19869 transaction(s)] done [0.19s].  
sorting and recoding items ... [34 item(s)] done [0.00s].  
creating transaction tree ... done [0.00s].  
checking subsets of size 1 2 done [0.00s].  
writing ... [3 rule(s)] done [0.00s].  
creating S4 object ... done [0.00s].
```

With min.confidence 0.1 and Support 0.05  
- 3 Association Rules

# Inspect Association Rules

```
> rules.029 <- apriori(Transactional_data, parameter=list(support=0.0293, confidence = 0.1, minlen=2))
Apriori

Parameter specification:
confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target ext
          0.1      0.1     1 none FALSE           TRUE      5  0.0293      2    10   rules TRUE

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

Absolute minimum support count: 582

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[3938 item(s), 19869 transaction(s)] done [0.21s].
sorting and recoding items ... [139 item(s)] done [0.00s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [20 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
> inspect(sort(rules.029, by='confidence'))
      lhs                      rhs            support  confidence coverage lift      count
[1] {PINK REGENCY TEACUP AND SAUCER} => {GREEN REGENCY TEACUP AND SAUCER} 0.03180834 0.8261438 0.03850219 16.203999 632
[2] {PINK REGENCY TEACUP AND SAUCER} => {ROSES REGENCY TEACUP AND SAUCER} 0.03009714 0.7816993 0.03850219 14.583647 598
[3] {GREEN REGENCY TEACUP AND SAUCER} => {ROSES REGENCY TEACUP AND SAUCER} 0.03860285 0.7571570 0.05098394 14.125776 767
[4] {ROSES REGENCY TEACUP AND SAUCER} => {GREEN REGENCY TEACUP AND SAUCER} 0.03860285 0.7201878 0.05360109 14.125776 767
[5] {JUMBO BAG PINK POLKADOT}        => {JUMBO BAG RED RETROSPOT} 0.04152197 0.6773399 0.06130152 6.442349 825
[6] {ALARM CLOCK BAKELIKE GREEN}   => {ALARM CLOCK BAKELIKE RED} 0.03221098 0.6530612 0.04932307 12.346026 640
[7] {JUMBO BAG BAROQUE BLACK WHITE} => {JUMBO BAG RED RETROSPOT} 0.02944285 0.6270096 0.04695757 5.963645 585
[8] {GREEN REGENCY TEACUP AND SAUCER} => {PINK REGENCY TEACUP AND SAUCER} 0.03180834 0.6238894 0.05098394 16.203999 632
[9] {JUMBO STORAGE BAG SUKI}       => {JUMBO BAG RED RETROSPOT} 0.03643867 0.6114865 0.05959032 5.816000 724
[10] {ALARM CLOCK BAKELIKE RED}    => {ALARM CLOCK BAKELIKE GREEN} 0.03221098 0.6089439 0.05289647 12.346026 640
[11] {JUMBO SHOPPER VINTAGE RED PAISLEY} => {JUMBO BAG RED RETROSPOT} 0.03422417 0.5787234 0.05913735 5.504383 680
[12] {ROSES REGENCY TEACUP AND SAUCER} => {PINK REGENCY TEACUP AND SAUCER} 0.03009714 0.5615023 0.05360109 14.583647 598
[13] {LUNCH BAG PINK POLKADOT}      => {LUNCH BAG RED RETROSPOT} 0.03049977 0.5559633 0.05485933 7.058425 606
[14] {LUNCH BAG BLACK SKULL.}       => {LUNCH BAG RED RETROSPOT} 0.03226131 0.5035350 0.06406966 6.392803 641
[15] {LUNCH BAG RED RETROSPOT}      => {LUNCH BAG BLACK SKULL.} 0.03226131 0.4095847 0.07876592 6.392803 641
[16] {JUMBO BAG RED RETROSPOT}      => {JUMBO BAG PINK POLKADOT} 0.04152197 0.3949258 0.10513866 6.442349 825
[17] {LUNCH BAG RED RETROSPOT}      => {LUNCH BAG PINK POLKADOT} 0.03049977 0.3872204 0.07876592 7.058425 606
[18] {JUMBO BAG RED RETROSPOT}      => {JUMBO STORAGE BAG SUKI} 0.03643867 0.3465773 0.10513866 5.816000 724
[19] {JUMBO BAG RED RETROSPOT}      => {JUMBO SHOPPER VINTAGE RED PAISLEY} 0.03422417 0.3255146 0.10513866 5.504383 680
[20] {JUMBO BAG RED RETROSPOT}      => {JUMBO BAG BAROQUE BLACK WHITE} 0.02944285 0.2800383 0.10513866 5.963645 585
```

# Prune Redundant Association Rules

```
> # Prune the Redundant Rules #####
>
> # Check for the rules which are subset of other rules
> subset_matrix <- issubset(rules,rules)
>
> # Assigning FALSE to diagonal line (as every rule is a subset of it's own, neglect diagonal positions)
> subset_matrix[lower.tri(subset_matrix, diag=T)] <- F
>
> # Check the item sets which are redundant and which are not
> redundant <- apply(subset_matrix, 2, any)
>
> rules.pruned <- rules[!redundant]
>
> # Inspect the pruned rules
> inspect(rules.pruned)
```

lhs	rhs	support	confidence	coverage	lift	count
[1] {PINK REGENCY TEACUP AND SAUCER}	=> {GREEN REGENCY TEACUP AND SAUCER}	0.03180834	0.8261438	0.03850219	16.203999	632
[2] {PINK REGENCY TEACUP AND SAUCER}	=> {ROSES REGENCY TEACUP AND SAUCER}	0.03009714	0.7816993	0.03850219	14.583647	598
[3] {GREEN REGENCY TEACUP AND SAUCER}	=> {ROSES REGENCY TEACUP AND SAUCER}	0.03860285	0.7571570	0.05098394	14.125776	767
[4] {JUMBO BAG PINK POLKADOT}	=> {JUMBO BAG RED RETROSPOT}	0.04152197	0.6773399	0.06130152	6.442349	825
[5] {ALARM CLOCK BAKELIKE GREEN}	=> {ALARM CLOCK BAKELIKE RED}	0.03221098	0.6530612	0.04932307	12.346026	640
[6] {JUMBO BAG BAROQUE BLACK WHITE}	=> {JUMBO BAG RED RETROSPOT}	0.02944285	0.6270096	0.04695757	5.963645	585
[7] {JUMBO STORAGE BAG SUKI}	=> {JUMBO BAG RED RETROSPOT}	0.03643867	0.6114865	0.05959032	5.816000	724
[8] {JUMBO SHOPPER VINTAGE RED PAISLEY}	=> {JUMBO BAG RED RETROSPOT}	0.03422417	0.5787234	0.05913735	5.504383	680
[9] {LUNCH BAG PINK POLKADOT}	=> {LUNCH BAG RED RETROSPOT}	0.03049977	0.5559633	0.05485933	7.058425	606
[10] {LUNCH BAG BLACK SKULL.}	=> {LUNCH BAG RED RETROSPOT}	0.03226131	0.5035350	0.06406966	6.392803	641

# Association rules Evaluation

**Support** ( $\{X\} \rightarrow \{Y\}$ ) =

$$\frac{\text{Transactions containing both } X \text{ and } Y}{\text{Total Transactions}}$$

**Confidence** ( $\{X\} \rightarrow \{Y\}$ ) =

$$\frac{\text{Transactions containing both } X \text{ and } Y}{\text{Transactions containing } X}$$

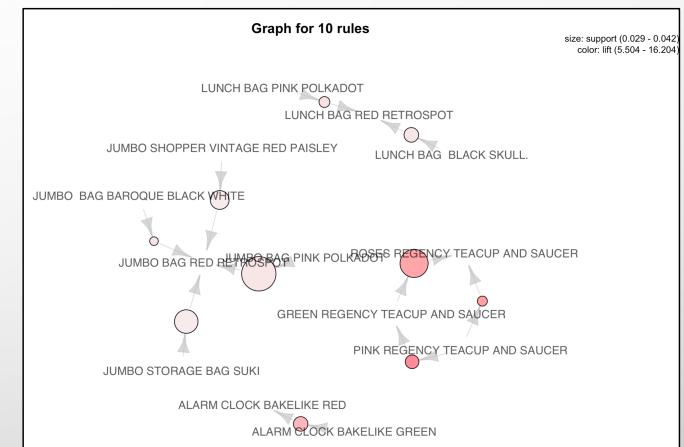
$$\text{Lift } (\{X\} \rightarrow \{Y\}) = \frac{(\text{Transactions containing both } X \text{ and } Y) / (\text{Transactions containing } X)}{\text{Fraction of transactions containing } Y}$$

```
> # We evaluate the association rules with support, confidence, lift values
> interestMeasure(rules.pruned, c("support", "confidence", "lift"), transactional_data)
      support confidence      lift
1  0.03180834  0.8261438 16.203999
2  0.03009714  0.7816993 14.583647
3  0.03860285  0.7571570 14.125776
4  0.04152197  0.6773399  6.442349
5  0.03221098  0.6530612 12.346026
6  0.02944285  0.6270096  5.963645
7  0.03643867  0.6114865  5.816000
8  0.03422417  0.5787234  5.504383
9  0.03049977  0.5559633  7.058425
10 0.03226131  0.5035350  6.392803
```

# Best 10 Association rules

```
> # Inspect the pruned rules ordered by confidence
> inspect(sort(rules.pruned, by='confidence'))
   lhs                               rhs          support  confidence coverage lift      count
[1] {PINK REGENCY TEACUP AND SAUCER} => {GREEN REGENCY TEACUP AND SAUCER} 0.03180834 0.8261438 0.03850219 16.203999 632
[2] {PINK REGENCY TEACUP AND SAUCER} => {ROSES REGENCY TEACUP AND SAUCER} 0.03009714 0.7816993 0.03850219 14.583647 598
[3] {GREEN REGENCY TEACUP AND SAUCER} => {ROSES REGENCY TEACUP AND SAUCER} 0.03860285 0.7571570 0.05098394 14.125776 767
[4] {JUMBO BAG PINK POLKADOT}        => {JUMBO BAG RED RETROSPOT}    0.04152197 0.6773399 0.06130152 6.442349 825
[5] {ALARM CLOCK BAKELIKE GREEN}   => {ALARM CLOCK BAKELIKE RED}  0.03221098 0.6530612 0.04932307 12.346026 640
[6] {JUMBO BAG BAROQUE BLACK WHITE} => {JUMBO BAG RED RETROSPOT}    0.02944285 0.6270096 0.04695757 5.963645 585
[7] {JUMBO STORAGE BAG SUKI}       => {JUMBO BAG RED RETROSPOT}    0.03643867 0.6114865 0.05959032 5.816000 724
[8] {JUMBO SHOPPER VINTAGE RED PAISLEY} => {JUMBO BAG RED RETROSPOT} 0.03422417 0.5787234 0.05913735 5.504383 680
[9] {LUNCH BAG PINK POLKADOT}      => {LUNCH BAG RED RETROSPOT}   0.03049977 0.5559633 0.05485933 7.058425 606
[10] {LUNCH BAG BLACK SKULL.}      => {LUNCH BAG RED RETROSPOT}   0.03226131 0.5035350 0.06406966 6.392803 641
```

```
> library(grid)
> library(arulesViz)
>
> plot(rules.pruned, method="graph")
```



# CONCLUSION

**Identify the customers into different groups based on their behavior with the online retail.**  
**Identify the Loyal Customers.**

Customers of online-retail have been segmented into two groups using their Recency, Frequency and Monetary values.

Group 1 – either with More Recency or Less Frequency or Less Monetary.

Group 2 – Less Recency, More Frequency and More Monetary.

Group 2 customers are identified as Loyal Customers.

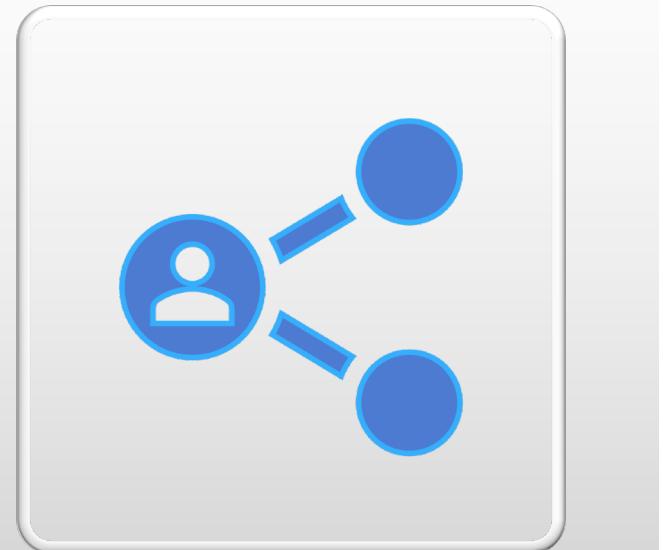
**Identify best 10 Association rules for the products purchased in the online retail.**

Best 10 Association rules ordered  
by confidence

{PINK REGENCY TEACUP AND SAUCER}	=> {GREEN REGENCY TEACUP AND SAUCER}
{PINK REGENCY TEACUP AND SAUCER}	=> {ROSES REGENCY TEACUP AND SAUCER}
{GREEN REGENCY TEACUP AND SAUCER}	=> {ROSES REGENCY TEACUP AND SAUCER}
{JUMBO BAG PINK POLKADOT}	=> {JUMBO BAG RED RETROSPOT}
{ALARM CLOCK BAKELIKE GREEN}	=> {ALARM CLOCK BAKELIKE RED}
{JUMBO BAG BAROQUE BLACK WHITE}	=> {JUMBO BAG RED RETROSPOT}
{JUMBO STORAGE BAG SUKI}	=> {JUMBO BAG RED RETROSPOT}
{JUMBO SHOPPER VINTAGE RED PAISLEY}	=> {JUMBO BAG RED RETROSPOT}
{LUNCH BAG PINK POLKADOT}	=> {LUNCH BAG RED RETROSPOT}
{LUNCH BAG BLACK SKULL.}	=> {LUNCH BAG RED RETROSPOT}

## FUTURE WORK

- Use Gap-Statistic Method to find the optimal number of clusters for clustering.
- Use Density based and K-medoids clustering algorithms for Customer segmentation.
- Inspect Association rules with confidence levels 0.8, 0.7 and so.. .Business can use these rules in designing lossless promotional offers for the products.



THANK YOU