RFM Analysis

library(plyr)  
library(dplyr)  
library(ggplot2)  
library(RColorBrewer)  
library(data.table)  
library(scales)  
  
setwd("C:/Users/cherring/Documents")

## Data Ingestion and Transformations

data <- read.csv("train\_clubmahindra.csv")  
data <- data[c(1,2,7,8,13,14,15,18,20,24)]  
data$booking\_date <- as.character(data$booking\_date)  
data$booking\_date <-as.POSIXct(data$booking\_date, format = "%d/%m/%y")  
  
# Subset to 2017  
data <- data[substr(data$booking\_date,1,4) == "2017",]  
  
# Calculate Total Amount Spent   
data$total\_amount\_spent <- data$amount\_spent\_per\_room\_night\_scaled \* 100 \* data$roomnights  
  
# Initialize customer data frame  
customers <- as.data.frame(unique(data$memberid))  
names(customers) <- "memberid"

## Recency

# Calculate number of days since booking date  
data$recency <- as.Date("2018-01-01") - as.Date(data$booking\_date)  
  
# Obtain number of days since most recent booking  
data = data.table(data)  
recency = data[,list(recency=min(recency)),by = 'memberid']  
  
# Add recency to customer data  
customers <- merge(customers, recency, by="memberid", all=TRUE, sort=TRUE)  
remove(recency)  
customers$recency <- as.numeric(customers$recency)

## Frequency

# Obtain list of distinct invoices by customer  
customer.invoices <- subset(data, select = c("memberid","reservation\_id"))  
customer.invoices <- customer.invoices[!duplicated(customer.invoices), ]  
customer.invoices <- customer.invoices[order(customer.invoices$memberid),]  
row.names(customer.invoices) <- NULL  
customer.invoices$rescount <- 1  
  
# Calculate frequency by taking sum of distinct invoices  
frequency = customer.invoices[,list(frequency=sum(rescount)),by = 'memberid']  
  
# Add frequency to customer data  
customers <- merge(customers, frequency, by="memberid", all=TRUE, sort=TRUE)  
remove(frequency)  
customers$frequency <- as.numeric(customers$frequency)

## Monetary

# Calculate monetary by taking sum of total amounts  
monetary = data[,list(monetary=sum(total\_amount\_spent)),by = 'memberid']  
  
# Add monetary value to customers dataset  
customers <- merge(customers, monetary, by="memberid", all.x=TRUE, sort=TRUE)  
remove(monetary)  
customers$monetary <- as.numeric(customers$monetary)

#### Apply Pareto Principle (80/20 rule)

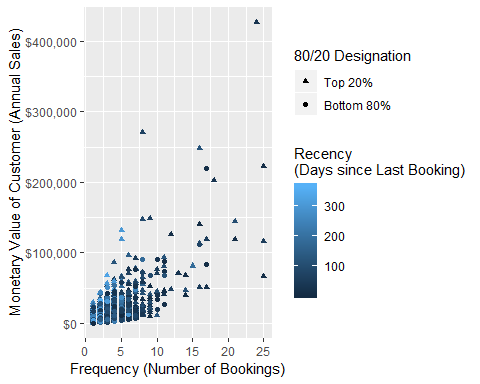
pareto.cutoff <- 0.8 \* sum(customers$monetary)  
customers$pareto <- ifelse(cumsum(customers$monetary) <= pareto.cutoff, "Top 20%", "Bottom 80%")  
customers$pareto <- factor(customers$pareto, levels=c("Top 20%", "Bottom 80%"), ordered=TRUE)  
levels(customers$pareto)

## [1] "Top 20%" "Bottom 80%"

## Scatter Plots

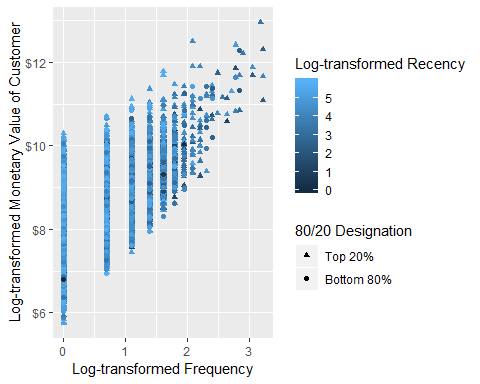
#### Raw Values

scatter.1 <- ggplot(customers, aes(x = frequency, y = monetary))  
scatter.1 <- scatter.1 + geom\_point(aes(colour = recency, shape = pareto))  
scatter.1 <- scatter.1 + scale\_shape\_manual(name = "80/20 Designation", values=c(17, 16))  
scatter.1 <- scatter.1 + scale\_colour\_gradient(name="Recency\n(Days since Last Booking)")  
scatter.1 <- scatter.1 + scale\_y\_continuous(label=dollar)  
scatter.1 <- scatter.1 + xlab("Frequency (Number of Bookings)")  
scatter.1 <- scatter.1 + ylab("Monetary Value of Customer (Annual Sales)")  
scatter.1



#### Log-transformed Values

scatter.1 <- ggplot(customers, aes(x = log(frequency), y = log(monetary)))  
scatter.1 <- scatter.1 + geom\_point(aes(colour = log(recency), shape = pareto))  
scatter.1 <- scatter.1 + scale\_shape\_manual(name = "80/20 Designation", values=c(17, 16))  
scatter.1 <- scatter.1 + scale\_colour\_gradient(name="Log-transformed Recency")  
scatter.1 <- scatter.1 + scale\_y\_continuous(label=dollar)  
scatter.1 <- scatter.1 + xlab("Log-transformed Frequency")  
scatter.1 <- scatter.1 + ylab("Log-transformed Monetary Value of Customer")  
scatter.1

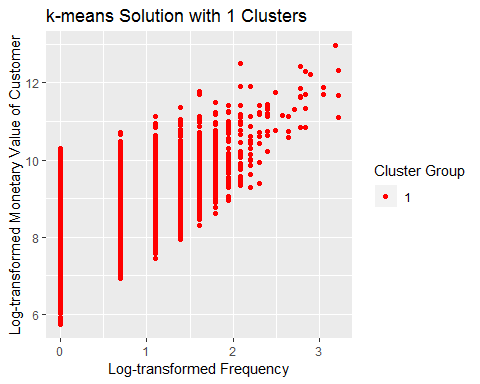


## Modeling

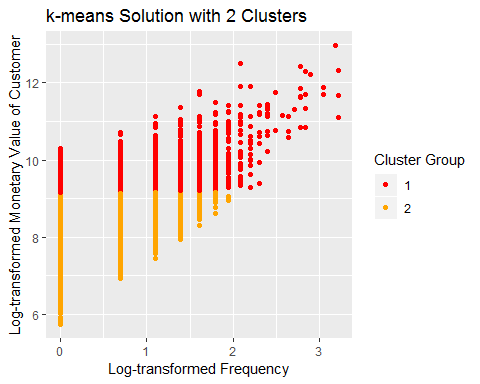
#### Test number of clusters

preprocessed <- customers[c(2:4)]  
j <- 10 # maximum number of clusters  
  
# Initiate model dataframe  
models <- data.frame(k=integer(),  
 tot.withinss=numeric(),  
 betweenss=numeric(),  
 totss=numeric(),  
 rsquared=numeric())  
  
# Add cluster membership to customers dataset  
for (k in 1:j ) {  
   
 print(k)  
   
 # Run kmeans  
 output <- kmeans(preprocessed, centers = k, nstart = 20)  
   
 # Add cluster membership to customers dataset  
 var.name <- paste("cluster", k, sep="\_")  
 customers[,(var.name)] <- output$cluster  
 customers[,(var.name)] <- factor(customers[,(var.name)], levels = c(1:k))  
   
 # Graph clusters  
 cluster\_graph <- ggplot(customers, aes(x = log(frequency), y = log(monetary)))  
 cluster\_graph <- cluster\_graph + geom\_point(aes(colour = customers[,(var.name)]))  
 colors <- c('red','orange','green3','deepskyblue','blue','darkorchid4','violet','pink1','tan3','black')  
 cluster\_graph <- cluster\_graph + scale\_colour\_manual(name = "Cluster Group", values=colors)  
 cluster\_graph <- cluster\_graph + xlab("Log-transformed Frequency")  
 cluster\_graph <- cluster\_graph + ylab("Log-transformed Monetary Value of Customer")  
 title <- paste("k-means Solution with", k, sep=" ")  
 title <- paste(title, "Clusters", sep=" ")  
 cluster\_graph <- cluster\_graph + ggtitle(title)  
 print(cluster\_graph)  
   
 # Cluster centers in original metrics  
 print(title)  
 cluster\_centers <- ddply(customers, .(customers[,(var.name)]), summarize,  
 monetary=round(median(monetary),2), frequency=round(median(frequency),1),  
 recency=round(median(recency), 0))  
 names(cluster\_centers)[names(cluster\_centers)=="customers[, (var.name)]"] <- "Cluster"  
 print(cluster\_centers)  
 cat("\n")  
   
 # Collect model information  
 models[k,("k")] <- k  
 models[k,("tot.withinss")] <- output$tot.withinss   
 models[k,("betweenss")] <- output$betweenss  
 models[k,("totss")] <- output$totss   
 models[k,("rsquared")] <- round(output$betweenss/output$totss, 3)   
 assign("models", models, envir = .GlobalEnv)  
 remove(output, var.name, cluster\_graph, cluster\_centers, title, colors)  
}

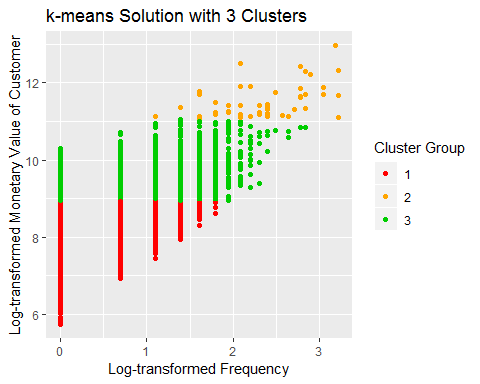
## [1] 1



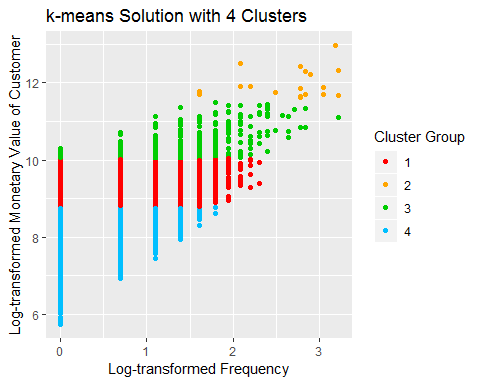
## [1] "k-means Solution with 1 Clusters"  
## Cluster monetary frequency recency  
## 1 1 3502.98 1 158  
##   
## [1] 2



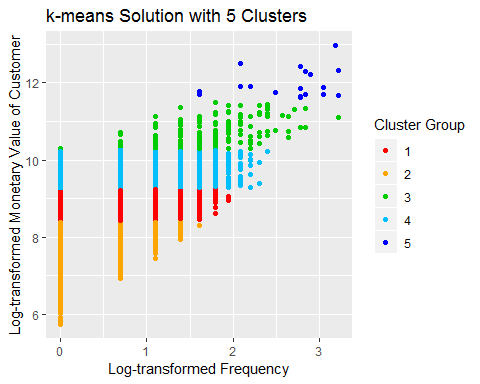
## [1] "k-means Solution with 2 Clusters"  
## Cluster monetary frequency recency  
## 1 1 12392.48 3 119  
## 2 2 3276.98 1 164  
##   
## [1] 3



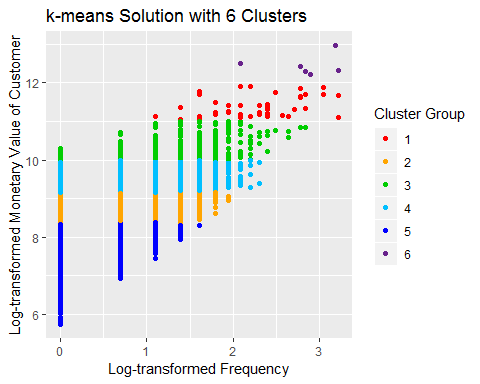
## [1] "k-means Solution with 3 Clusters"  
## Cluster monetary frequency recency  
## 1 1 3121.59 1 167  
## 2 2 87261.93 10 60  
## 3 3 10089.68 3 121  
##   
## [1] 4



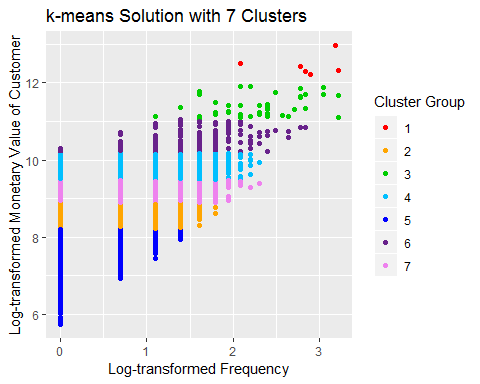
## [1] "k-means Solution with 4 Clusters"  
## Cluster monetary frequency recency  
## 1 1 8634.10 2 122  
## 2 2 142114.32 16 60  
## 3 3 30841.91 4 92  
## 4 4 2917.07 1 171  
##   
## [1] 5



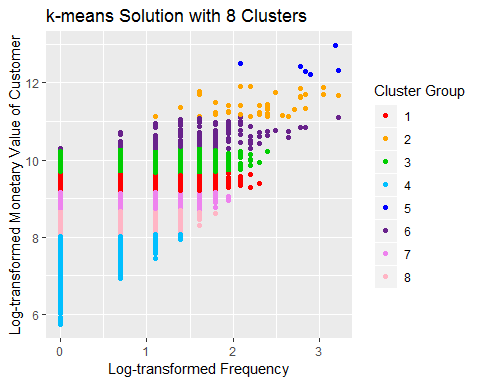
## [1] "k-means Solution with 5 Clusters"  
## Cluster monetary frequency recency  
## 1 1 6137.99 2 142  
## 2 2 2453.98 1 178  
## 3 3 38273.84 5 92  
## 4 4 12931.55 3 118  
## 5 5 142114.32 16 60  
##   
## [1] 6



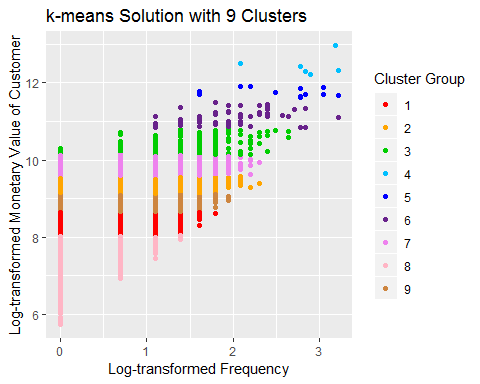
## [1] "k-means Solution with 6 Clusters"  
## Cluster monetary frequency recency  
## 1 1 81232.30 9.5 63  
## 2 2 5843.51 2.0 142  
## 3 3 28220.45 4.0 100  
## 4 4 11868.87 3.0 121  
## 5 5 2420.40 1.0 179  
## 6 6 234831.18 17.5 30  
##   
## [1] 7



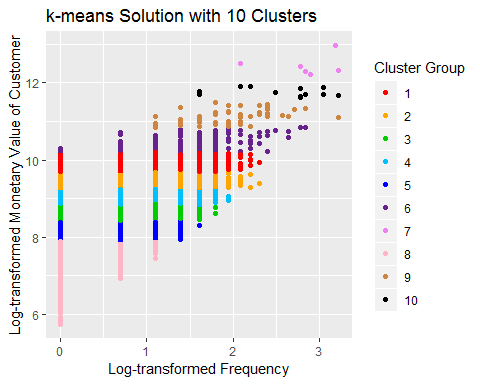
## [1] "k-means Solution with 7 Clusters"  
## Cluster monetary frequency recency  
## 1 1 234831.18 17.5 30  
## 2 2 5081.29 2.0 150  
## 3 3 81572.21 10.0 66  
## 4 4 15667.49 3.0 116  
## 5 5 2320.75 1.0 181  
## 6 6 34051.75 4.0 108  
## 7 7 8898.02 2.0 121  
##   
## [1] 8



## [1] "k-means Solution with 8 Clusters"  
## Cluster monetary frequency recency  
## 1 1 11490.82 3.0 121  
## 2 2 84928.00 10.0 63  
## 3 3 18533.49 3.0 101  
## 4 4 2176.37 1.0 182  
## 5 5 234831.18 17.5 30  
## 6 6 36191.01 4.0 106  
## 7 7 7309.24 2.0 128  
## 8 8 4431.70 2.0 153  
##   
## [1] 9



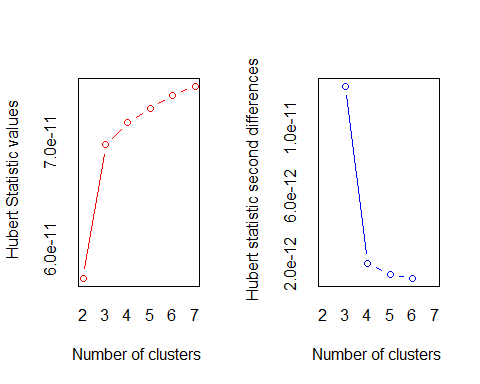
## [1] "k-means Solution with 9 Clusters"  
## Cluster monetary frequency recency  
## 1 1 4279.77 2.0 155  
## 2 2 10872.55 3.0 121  
## 3 3 31032.26 4.0 102  
## 4 4 234831.18 17.5 30  
## 5 5 122884.97 16.0 72  
## 6 6 66076.84 7.0 61  
## 7 7 17037.38 3.0 111  
## 8 8 2129.94 1.0 183  
## 9 9 7066.82 2.0 131  
##   
## [1] 10



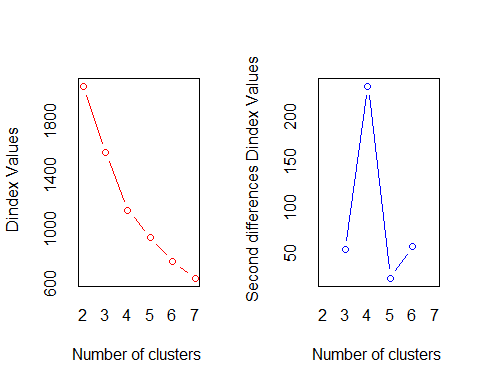
## [1] "k-means Solution with 10 Clusters"  
## Cluster monetary frequency recency  
## 1 1 18618.24 3.0 100  
## 2 2 12254.72 3.0 121  
## 3 3 5411.39 2.0 150  
## 4 4 8189.38 2.0 122  
## 5 5 3481.92 1.0 165  
## 6 6 33396.71 4.0 107  
## 7 7 234831.18 17.5 30  
## 8 8 1862.77 1.0 183  
## 9 9 67292.57 7.0 72  
## 10 10 122884.97 16.0 72

# Use NBClust to determine optimal number of clusters   
library(NbClust)  
set.seed(1)  
nc <- NbClust(preprocessed[sample(nrow(preprocessed), 1000),], min.nc=2, max.nc=7, method="kmeans")

## [1] "Frey index : No clustering structure in this data set"



## \*\*\* : The Hubert index is a graphical method of determining the number of clusters.  
## In the plot of Hubert index, we seek a significant knee that corresponds to a   
## significant increase of the value of the measure i.e the significant peak in Hubert  
## index second differences plot.   
##



## \*\*\* : The D index is a graphical method of determining the number of clusters.   
## In the plot of D index, we seek a significant knee (the significant peak in Dindex  
## second differences plot) that corresponds to a significant increase of the value of  
## the measure.   
##   
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
## \* Among all indices:   
## \* 5 proposed 2 as the best number of clusters   
## \* 9 proposed 3 as the best number of clusters   
## \* 2 proposed 4 as the best number of clusters   
## \* 2 proposed 5 as the best number of clusters   
## \* 3 proposed 6 as the best number of clusters   
## \* 2 proposed 7 as the best number of clusters   
##   
## \*\*\*\*\* Conclusion \*\*\*\*\*   
##   
## \* According to the majority rule, the best number of clusters is 3   
##   
##   
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

barplot(table(nc$Best.n[1,]),  
 xlab="Number of Clusters", ylab="Number of Criteria",  
 main="Number of Clusters Chosen by Criteria",  
 cex.axis = .8,  
 cex.names = .8)

