Unsupervised Learning

Beaty

9/01/2021

Customer Segmentation

Data Understanding

1. Define the question:

- Perform clustering stating insights drawn from your analysis and visualizations.
- Upon implementation, provide comparisons between the approaches learned this week i.e. K-Means clustering vs Hierarchical clustering highlighting the strengths and limitations of each approach in the context of your analysi

2. Metric for success:

- Create models using K-means Modeling & Hierarchical clustering and compare them.
- Highlighting the strengths and limitations of each approach

3. Understanding the context:

Kira Plastinina is a Russian brand that is sold through a defunct chain of retail stores in Russia, Ukraine, Kazakhstan, Belarus, China, Philippines, and Armenia. The brand's Sales and Marketing team would like to understand their customer's behavior from data that they have collected over the past year. More specifically, they would like to learn the characteristics of customer groups. My findings should help inform the team in formulating the marketing and sales strategies of the brand.

4. Experimental design:

Steps to be undertaken during this study include: - Problem Definition - Loading the data & needed packages. - Exploring the dataset. - Cleaning the data. - Feature engineering. - Exploratory Data Analysis. - Clustering(K-Means & Hierarchical) - Challenging the solution

5. Data appropriateness:

This will be well checked & described in the data cleaning.

Loading required packages, since they were already instaled

```
#Loading the R packages in my notebook.
library("data.table")
library("ggplot2")
library(dplyr)

##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library("corrplot")

## corrplot 0.90 loaded
```

Loading the data

```
#Loading the datas
df = read.csv(url("http://bit.ly/EcommerceCustomersDataset"))

head(df)

##   Administrative Administrative_Duration Informational
Informational_Duration
## 1              0                       0             0
0
## 2              0                       0             0
0
## 3              0                      -1             0
-1
## 4              0                       0             0
0
## 5              0                       0             0
0
## 6              0                       0             0
0
##   ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
## 1              1                0.000000  0.20000000 0.2000000          0
## 2              2               64.000000  0.00000000 0.1000000          0
## 3              1               -1.000000  0.20000000 0.2000000          0
## 4              2                2.666667  0.05000000 0.1400000          0
## 5             10              627.500000  0.02000000 0.0500000          0
## 6             19              154.216667  0.01578947 0.0245614          0
##   SpecialDay Month OperatingSystems Browser Region TrafficType
## 1          0   Feb                1       1      1           1
## 2          0   Feb                2       2      1           2
## 3          0   Feb                4       1      9           3
## 4          0   Feb                3       2      2           4
## 5          0   Feb                3       3      1           4
## 6          0   Feb                2       2      1           3
```

```
##         VisitorType Weekend Revenue
## 1 Returning_Visitor   FALSE   FALSE
## 2 Returning_Visitor   FALSE   FALSE
## 3 Returning_Visitor   FALSE   FALSE
## 4 Returning_Visitor   FALSE   FALSE
## 5 Returning_Visitor    TRUE   FALSE
## 6 Returning_Visitor   FALSE   FALSE
```

Getting the dimensions of our dataset

```
dim(df)

## [1] 12330    18
```

Our dataset has 18 columns and 12,330 entries.

Getting the summary statistics of our data

```
#Printing out the basic statistics in ou dataset.
summary(df)

##  Administrative   Administrative_Duration Informational
##  Min.   : 0.000   Min.   :  -1.00         Min.   : 0.000
##  1st Qu.: 0.000   1st Qu.:   0.00         1st Qu.: 0.000
##  Median : 1.000   Median :   8.00         Median : 0.000
##  Mean   : 2.318   Mean   :  80.91         Mean   : 0.504
##  3rd Qu.: 4.000   3rd Qu.:  93.50         3rd Qu.: 0.000
##  Max.   :27.000   Max.   :3398.75         Max.   :24.000
##  NA's   :14       NA's   :14              NA's   :14
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :  -1.00        Min.   :  0.00   Min.   :   -1.0
##  1st Qu.:   0.00        1st Qu.:  7.00   1st Qu.:  185.0
##  Median :   0.00        Median : 18.00   Median :  599.8
##  Mean   :  34.51        Mean   : 31.76   Mean   : 1196.0
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1466.5
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##  NA's   :14             NA's   :14       NA's   :14
##   BounceRates         ExitRates         PageValues        SpecialDay
##  Min.   :0.000000   Min.   :0.00000   Min.   :  0.000   Min.   :0.00000
##  1st Qu.:0.000000   1st Qu.:0.01429   1st Qu.:  0.000   1st Qu.:0.00000
##  Median :0.003119   Median :0.02512   Median :  0.000   Median :0.00000
##  Mean   :0.022152   Mean   :0.04300   Mean   :  5.889   Mean   :0.06143
##  3rd Qu.:0.016684   3rd Qu.:0.05000   3rd Qu.:  0.000   3rd Qu.:0.00000
##  Max.   :0.200000   Max.   :0.20000   Max.   :361.764   Max.   :1.00000
##  NA's   :14         NA's   :14
##     Month           OperatingSystems    Browser           Region
##  Length:12330       Min.   :1.000    Min.   : 1.000   Min.   :1.000
##  Class :character   1st Qu.:2.000    1st Qu.: 2.000   1st Qu.:1.000
##  Mode  :character   Median :2.000    Median : 2.000   Median :3.000
##                     Mean   :2.124    Mean   : 2.357   Mean   :3.147
##                     3rd Qu.:3.000    3rd Qu.: 2.000   3rd Qu.:4.000
```

```
##                     Max.   :8.000    Max.   :13.000   Max.   :9.000
##
##   TrafficType    VisitorType         Weekend         Revenue
##  Min.   : 1.00   Length:12330       Mode :logical   Mode :logical
##  1st Qu.: 2.00   Class :character   FALSE:9462      FALSE:10422
##  Median : 2.00   Mode  :character   TRUE :2868      TRUE :1908
##  Mean   : 4.07
##  3rd Qu.: 4.00
##  Max.   :20.00
##
```

The above contains the rages of our columns, the quartile values(IQR can be calculated from those), aswell as the median and mean for each row.

We can also see that the Admistrative, AdministrativeDuration, Informational, InformationalDuration, ProductRelated Duration, BounceRates & ExitRates columns have nulls in their columns.

Data Cleaning

From the codes above, we saw quite some null values. We will now deal with them First, we will check to see how many nulls are there in the different columns

```
#Checking for missing values
colSums(is.na(df))
```

```
##          Administrative Administrative_Duration           Informational
##                      14                       14                      14
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                      14                       14                      14
##             BounceRates               ExitRates              PageValues
##                      14                       14                       0
##              SpecialDay                   Month        OperatingSystems
##                       0                       0                       0
##                 Browser                  Region             TrafficType
##                       0                       0                       0
##             VisitorType                 Weekend                 Revenue
##                       0                       0                       0
```

We will then use the MICE package to impute the missing values

Installing the MICE package

```
#Installing the package
#install.packages("mice", dependencies = TRUE)

# Loading the library
library("mice")
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind

#Using the MICE package to fill in the missing values
mice_mod <- mice(df[,
c("Administrative","Administrative_Duration","Informational","Informational_D
uration","ProductRelated","ProductRelated_Duration","BounceRates","ExitRates"
)],method='rf')

##
##  iter imp variable
##   1   1  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   1   2  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   1   3  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   1   4  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   1   5  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   2   1  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   2   2  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   2   3  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   2   4  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   2   5  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   3   1  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   3   2  Administrative  Administrative_Duration  Informational
```

```
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   3   3  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   3   4  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   3   5  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   4   1  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   4   2  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   4   3  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   4   4  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   4   5  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   5   1  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   5   2  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   5   3  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   5   4  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates
##   5   5  Administrative  Administrative_Duration  Informational
Informational_Duration  ProductRelated  ProductRelated_Duration  BounceRates
ExitRates

mice_complete <- complete(mice_mod)

#Transferring the missing values into the main dataset
df$Administrative <- mice_complete$Administrative
df$Administrative_Duration <- mice_complete$Administrative_Duration
df$Informational <- mice_complete$Informational
df$Informational_Duration <- mice_complete$Informational_Duration
df$ProductRelated <- mice_complete$ProductRelated
```

```
df$ProductRelated_Duration <- mice_complete$ProductRelated_Duration
df$BounceRates <- mice_complete$BounceRates
df$ExitRates <- mice_complete$ExitRates
```

We will now check if all the nulls have been imputed successfully

```
#Checking for missing values in our dataset
colSums(is.na(df))
```

```
##          Administrative Administrative_Duration           Informational
##                       0                       0                       0
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                       0                       0                       0
##             BounceRates               ExitRates              PageValues
##                       0                       0                       0
##              SpecialDay                   Month        OperatingSystems
##                       0                       0                       0
##                 Browser                  Region             TrafficType
##                       0                       0                       0
##             VisitorType                 Weekend                 Revenue
##                       0                       0                       0
```

There are no nulls anymore

Checking for duplicates

```
length(which(duplicated(df)))
```

```
## [1] 117
```

We will remove the duplicates

```
df = unique(df)
```

Checking if the duplicates have been removed

```
anyDuplicated(df)
```

```
## [1] 0
```

There are no duplicates anymore in our data

Checking for outliers

```
#Selecting only the numeric columns in our dataset
num_cols <- unlist(lapply(df, is.numeric))
num_cols
```

```
##          Administrative Administrative_Duration           Informational
##                    TRUE                    TRUE                    TRUE
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                    TRUE                    TRUE                    TRUE
##             BounceRates               ExitRates              PageValues
```

```
##                     TRUE                      TRUE                      TRUE
##              SpecialDay                     Month           OperatingSystems
##                     TRUE                     FALSE                      TRUE
##                  Browser                    Region               TrafficType
##                     TRUE                      TRUE                      TRUE
##              VisitorType                   Weekend                   Revenue
##                    FALSE                     FALSE                     FALSE

#Subsetting the numeric columns
data_num <- df[ , num_cols]
```

We will use the melt function to view the outliers in our dataset First, we will install the reshape package

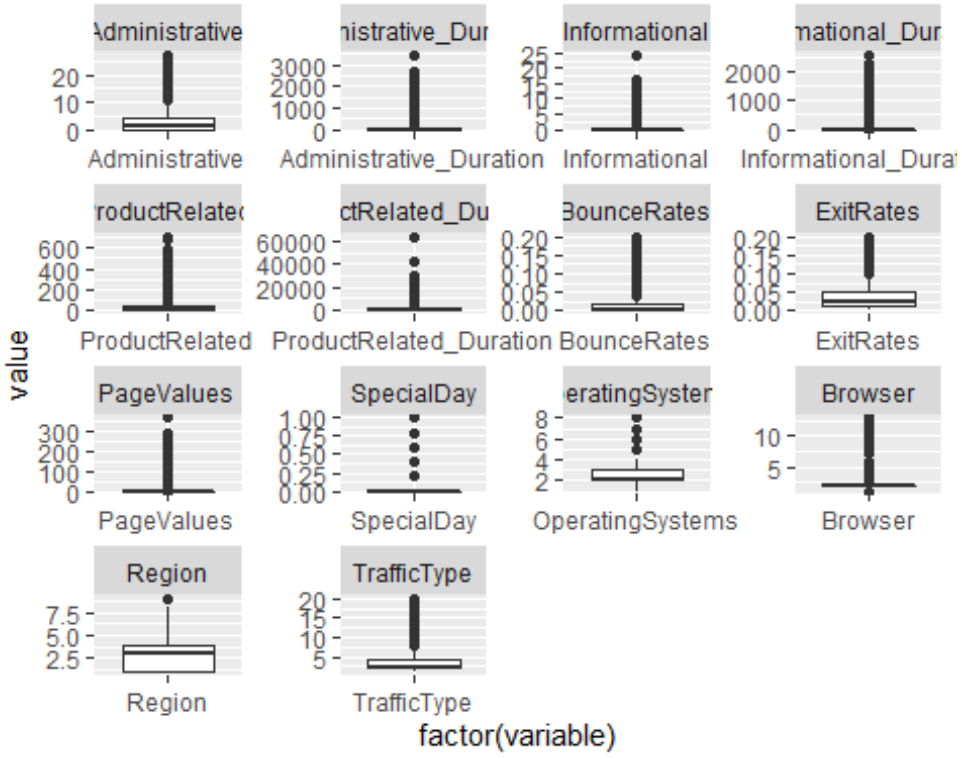
```
#Installing the package
#install.packages("reshape",dependencies = TRUE)
```

Loading the library

```
library("reshape")

##
## Attaching package: 'reshape'

## The following object is masked from 'package:dplyr':
##
##     rename

## The following object is masked from 'package:data.table':
##
##     melt
```

We will melt the data to view the outliers

```
#Melting the data
meltData <- melt(data_num)

## Using  as id variables

#Plotting the boxplots to view the outliers
p <- ggplot(meltData, aes(factor(variable), value))
p + geom_boxplot() + facet_wrap(~variable, scale="free")
```



The columns with outliers are Administrative Duration, Informational, Informational Duration, ProductRelated, Product Related Duration & Page Values columns. We will not be dropping the outliers for now.

Exploratory Data Analysis

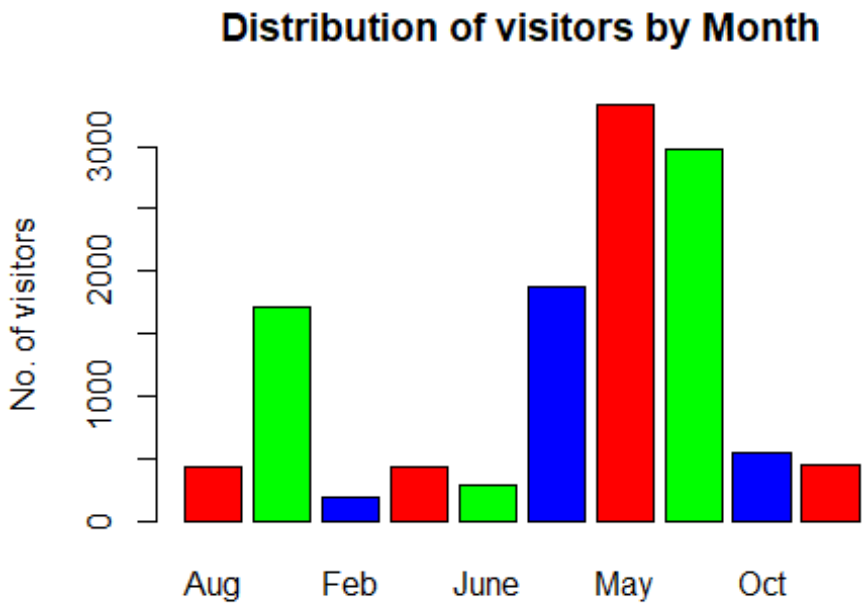
a) Univariate Measures of dispersion & spread

```
summary(df)

##  Administrative   Administrative_Duration Informational
##  Min.   : 0.000   Min.   :  -1.00         Min.   : 0.0000
##  1st Qu.: 0.000   1st Qu.:   0.00         1st Qu.: 0.0000
##  Median : 1.000   Median :   9.00         Median : 0.0000
##  Mean   : 2.339   Mean   :  81.66         Mean   : 0.5088
##  3rd Qu.: 4.000   3rd Qu.:  94.70         3rd Qu.: 0.0000
##  Max.   :27.000   Max.   :3398.75         Max.   :24.0000
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :  -1.00        Min.   :  0.00   Min.   :   -1.0
##  1st Qu.:   0.00        1st Qu.:  8.00   1st Qu.:  192.9
##  Median :   0.00        Median : 18.00   Median :  608.9
##  Mean   :  34.82        Mean   : 32.04   Mean   : 1207.2
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1477.4
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##   BounceRates         ExitRates         PageValues        SpecialDay
##  Min.   :0.000000   Min.   :0.00000   Min.   :  0.000   Min.   :0.0000
##  1st Qu.:0.000000   1st Qu.:0.01422   1st Qu.:  0.000   1st Qu.:0.0000
##  Median :0.002899   Median :0.02500   Median :  0.000   Median :0.0000
```

```
##  Mean   :0.020446   Mean   :0.04149   Mean   :  5.946   Mean   :0.0619
##  3rd Qu.:0.016667   3rd Qu.:0.04848   3rd Qu.:  0.000   3rd Qu.:0.0000
##  Max.   :0.200000   Max.   :0.20000   Max.   :361.764   Max.   :1.0000
##     Month           OperatingSystems    Browser           Region
##  Length:12213       Min.   :1.000    Min.   : 1.000   Min.   :1.000
##  Class :character   1st Qu.:2.000    1st Qu.: 2.000   1st Qu.:1.000
##  Mode  :character   Median :2.000    Median : 2.000   Median :3.000
##                     Mean   :2.124    Mean   : 2.358   Mean   :3.153
##                     3rd Qu.:3.000    3rd Qu.: 2.000   3rd Qu.:4.000
##                     Max.   :8.000    Max.   :13.000   Max.   :9.000
##   TrafficType     VisitorType         Weekend          Revenue
##  Min.   : 1.000   Length:12213       Mode :logical   Mode :logical
##  1st Qu.: 2.000   Class :character   FALSE:9354      FALSE:10305
##  Median : 2.000   Mode  :character   TRUE :2859      TRUE :1908
##  Mean   : 4.074
##  3rd Qu.: 4.000
##  Max.   :20.000
```

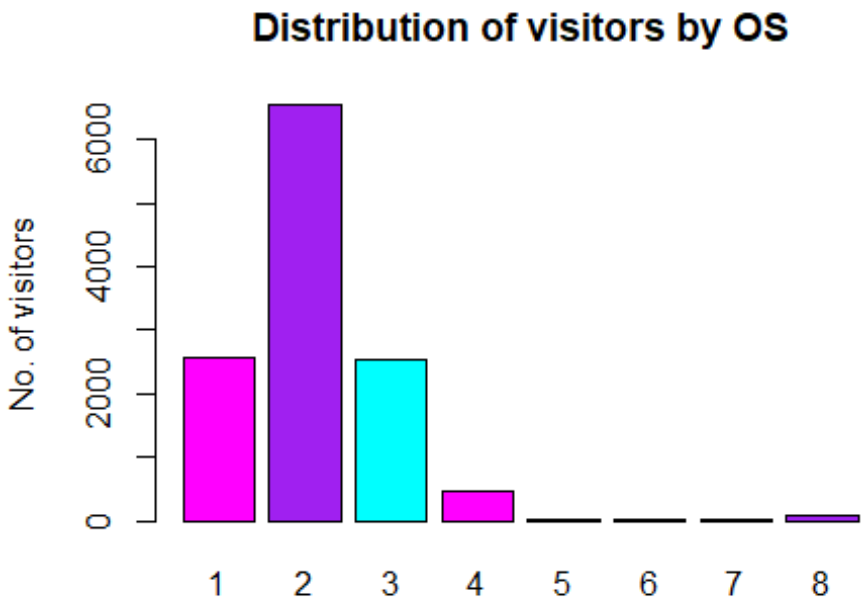
From the above, we can see the measures of dispersion for the columns. It also shows the ranges for each, as well as the IQR.

```
# A bar plot showing the distribution of visitors by Month
barplot(table(df$Month),  col =  c("red" , "green", "blue"), ylab = "No. of visitors", main = "Distribution of visitors by Month")
```



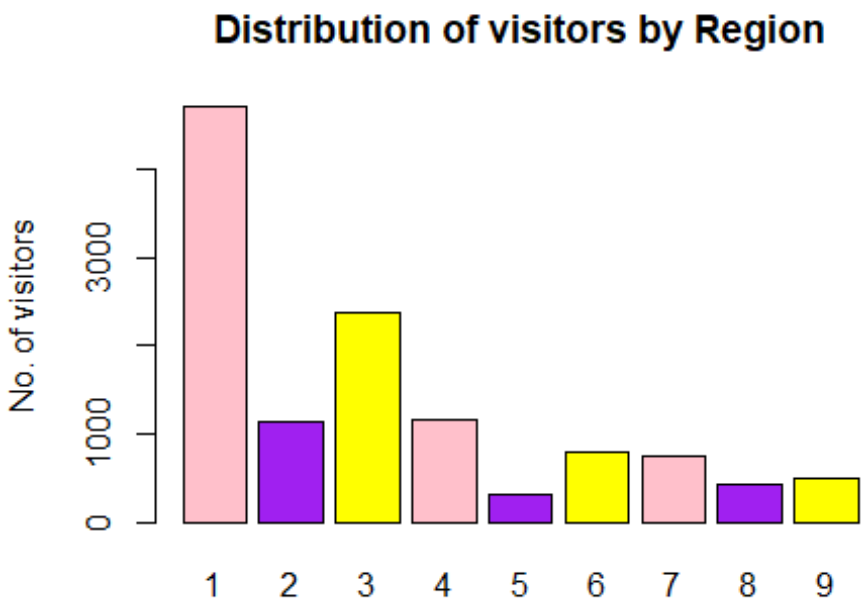
From the plot, we can see that May had the highest number of visitors on the site, followed by November then March & December. February had the least number of visitors.

```
# A bar plot showing the distribution of visitors by the operating system
barplot(table(df$OperatingSystems),  col =  c("magenta" , "purple", "cyan"),
ylab = "No. of visitors", main = "Distribution of visitors by OS")
```



Most of the users used the Operating system denoted by 2.

```
# A bar plot showing the distribution of visitors by the region
barplot(table(df$Region),  col =  c("pink" , "purple", "yellow"), ylab = "No.
of visitors", main = "Distribution of visitors by Region")
```



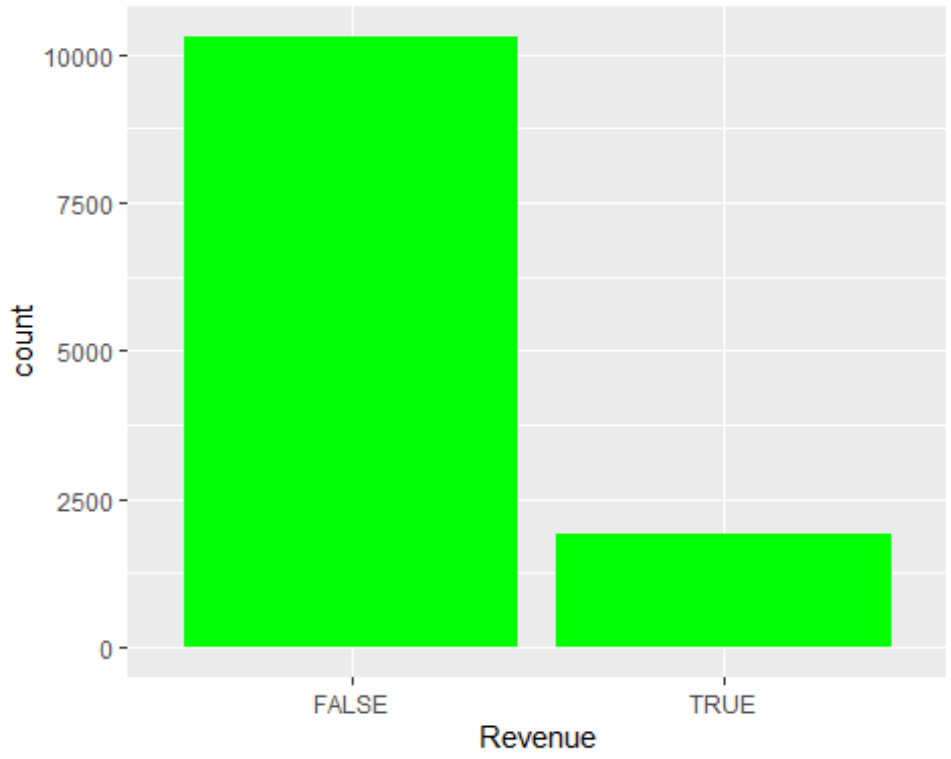
Most of the visitors were from the region denoted by 1, then followed by region 3, then region 4. Region 5 had the least number of visitors.

```
#Plotting the results
#z <- ggplot(month, aes(x = `Month`, y = n, group = 1))

#z + geom_line(aes(fill = `Month`))
```

November had the highest amount of revenue on the site, which may be attributed to Black Friday & Cyber Monday period sales. It is followed by May, though there is quite a huge difference in revenue between the 2 months. December comes a close third, which may be explained by people's spending habits during the Christmas season.

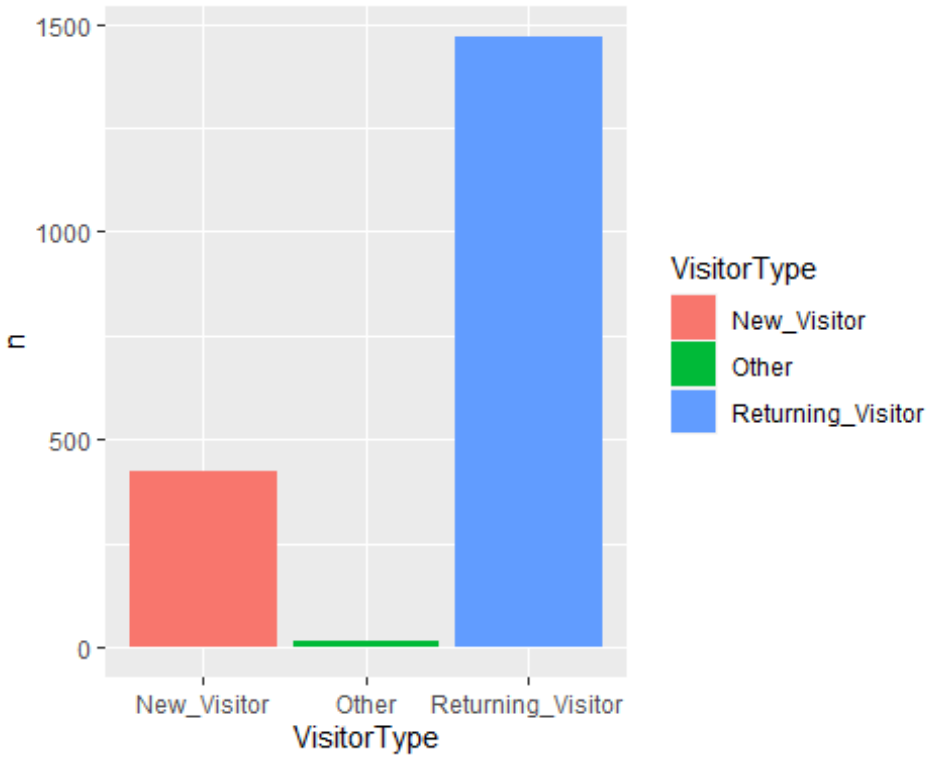
b) Bivariate

```
#Plotting the distribution of clients who brought in revenues.
ggplot(df, aes(Revenue)) +
  geom_bar(fill = "green")
```



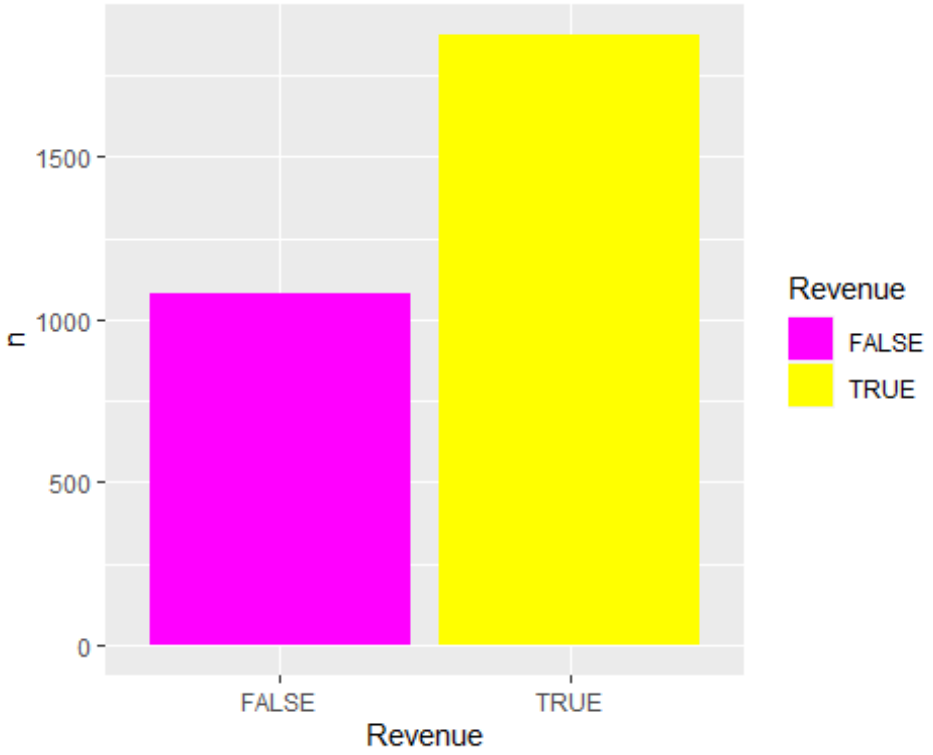
We can see that a huge number of clients didn't bring in revenue to the site.

```
#Grouping the month with the total number of persons who had revenue
month <- df %>%
  group_by(Month) %>%
  summarise(n=sum(Revenue, na.rm=TRUE)) %>%
  arrange(desc(n))%>%
  head(10)

#Grouping the visitor type by the revenues
visitor <- df %>%
  group_by(VisitorType) %>%
  summarise(n=sum(Revenue, na.rm=TRUE)) %>%
  arrange(desc(n))%>%
  head(10)

#Plotting the results
a <- ggplot(visitor, aes(x = `VisitorType`, y = n))

a + geom_col(aes(fill = `VisitorType`))
```



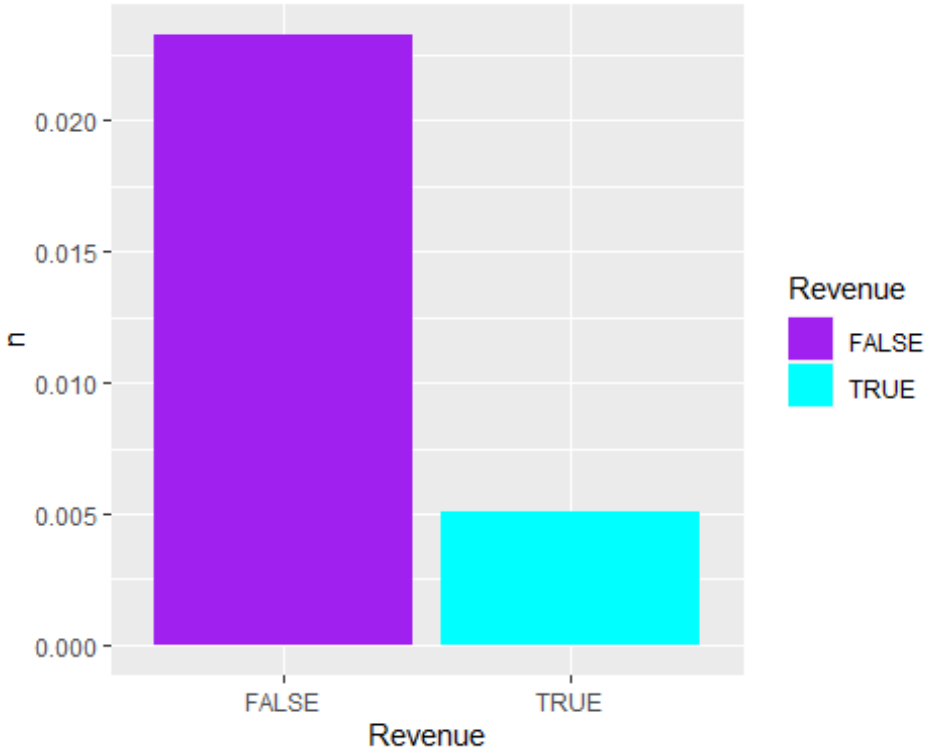
From the plots, we can see that it is more likely for a returning visitor to purchase as compared to a ew one or other.

```
#Grouping the mean number of product related duration by whether one brought
in revenue or not.
product_related <- df %>%
  group_by(Revenue) %>%
  summarise(n=mean(ProductRelated_Duration, na.rm=TRUE)) %>%
  arrange(desc(n))%>%
  head(10)

#Plotting the results
c <- ggplot(product_related, aes(x = `Revenue`, y = n))

c + geom_col(aes(fill = `Revenue`))+
  scale_fill_manual(values = c('magenta', 'yellow'))
```



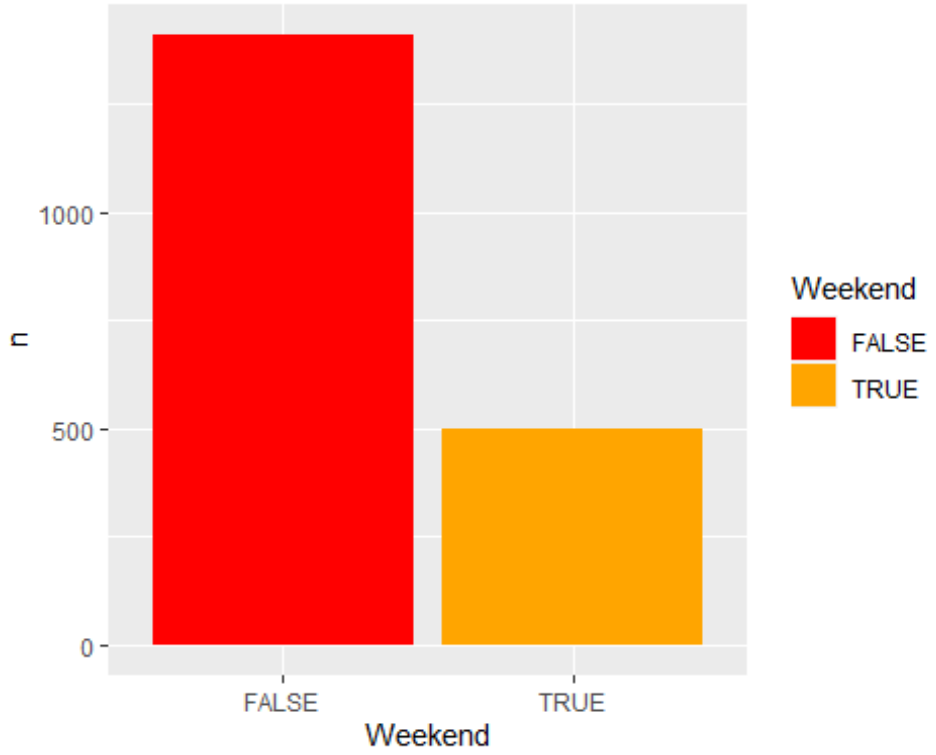
We can see that the longer one spent on the product page, the more likely they are to bring in revenue.

```
#Grouping the mean bounce rate by the earning of revenue from an individual
bounce_rate <- df %>%
  group_by(Revenue) %>%
  summarise(n=mean(BounceRates, na.rm=TRUE)) %>%
  arrange(desc(n))%>%
  head(10)

#Plotting
c <- ggplot(bounce_rate, aes(x = `Revenue`, y = n))

c + geom_col(aes(fill = `Revenue`)) +
  scale_fill_manual(values = c('purple', 'cyan'))
```



People who do not bring in revenue have a higher mean bounce rate as compared to those that brought in revenue.

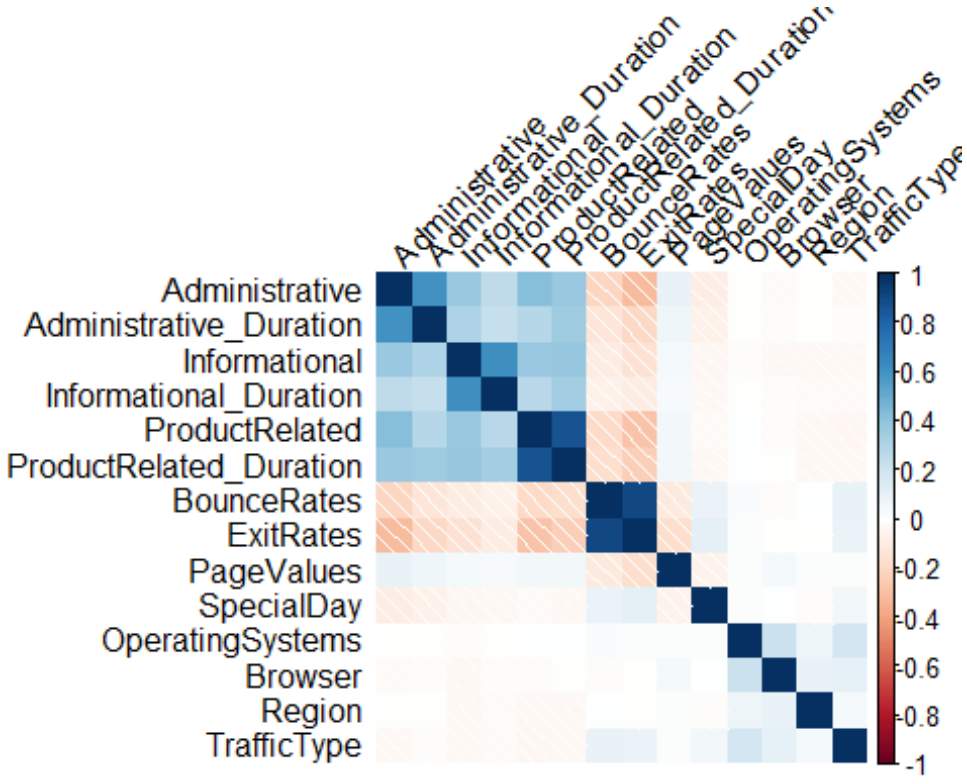
```
#Grouping the weekends by the number of persons who brought in Revenue
weekend <- df %>%
  group_by(Weekend) %>%
  summarise(n=sum(Revenue, na.rm=TRUE))

#Viewing the results.
c <- ggplot(weekend, aes(x = `Weekend`, y = n))

c + geom_col(aes(fill = `Weekend`)) +
  scale_fill_manual(values = c('red', 'orange'))
```



Weekdays have more revenues as compared to the weekends. However, this may be due to the fact that there are more weekdays than weekends.

Correlation

```
#Calculating the correlation between columns

correlations = cor(data_num)

# Creating a correlogram to plot our correlation for better presentation
corrplot(correlations, method="shade", tl.col="black", tl.srt=45)
```



We can see high positive correlations between different types of pages and the duration spent on each e.g. Administrative & Administrative duration. There is a high positive correlation between the bounce rates and the exit rates. Thus

There are low positive correlations between the different types of pages e.g. Administrative & Informational. Thus

There is a low negative correlation between the different types of pages and the bounce rates & exit rates. Thus as one increases, the other decreases.

c) Multivariate

```
#Factoring categorical variables in our dataset.
df$VisitorType <- as.integer(as.factor(df$VisitorType))
df$Month <- as.integer(as.factor(df$Month))
df$Weekend <- as.integer(as.factor(df$Weekend))

#Using the principal component analysis to check for component variance.
df.pca <- prcomp(df[,c(1:17)], center = TRUE, scale. = TRUE)
summary(df.pca)

## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
PC7
## Standard deviation     1.8420 1.3446 1.17614 1.08675 1.03804 1.01206
0.98906
## Proportion of Variance 0.1996 0.1064 0.08137 0.06947 0.06338 0.06025
0.05754
```

```
## Cumulative Proportion  0.1996 0.3059 0.38730 0.45678 0.52016 0.58041
0.63795
##                            PC8    PC9    PC10    PC11    PC12    PC13
PC14
## Standard deviation     0.97711 0.9661 0.93509 0.91871 0.89896 0.87064
0.64971
## Proportion of Variance 0.05616 0.0549 0.05143 0.04965 0.04754 0.04459
0.02483
## Cumulative Proportion  0.69412 0.7490 0.80045 0.85010 0.89764 0.94223
0.96706
##                          PC15   PC16    PC17
## Standard deviation     0.5933 0.3522 0.28986
## Proportion of Variance 0.0207 0.0073 0.00494
## Cumulative Proportion  0.9878 0.9951 1.00000
```

From the analysis, we can see that 12 components account for close to 90% variance in the data at 89.8%. 5 components in our dataset have a 52% explanation of variance. So we could actually use a few variable in our analysis and achive the desired results.

Clustering

1. K-Means

```
#Confirming that there are no nulls
sum(is.na(df))

## [1] 0

#Separating the response variables and the class variable.
df.new<- df[, c(1:17)]
df.class<- df[, "Revenue"]
head(df.new)

##   Administrative Administrative_Duration Informational
Informational_Duration
## 1              0                       0             0
0
## 2              0                       0             0
0
## 3              0                      -1             0
-1
## 4              0                       0             0
0
## 5              0                       0             0
0
## 6              0                       0             0
0
##   ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
## 1              1                0.000000  0.20000000 0.2000000          0
## 2              2               64.000000  0.00000000 0.1000000          0
## 3              1               -1.000000  0.20000000 0.2000000          0
## 4              2                2.666667  0.05000000 0.1400000          0
```

```
## 5             10              627.500000  0.02000000 0.0500000          0
## 6             19              154.216667  0.01578947 0.0245614          0
##   SpecialDay Month OperatingSystems Browser Region TrafficType VisitorType
## 1          0     3                 1       1      1           1           3
## 2          0     3                 2       2      1           2           3
## 3          0     3                 4       1      9           3           3
## 4          0     3                 3       2      2           4           3
## 5          0     3                 3       3      1           4           3
## 6          0     3                 2       2      1           3           3
##   Weekend
## 1       1
## 2       1
## 3       1
## 4       1
## 5       2
## 6       1

#Normalizing our continuous variables.
normalize <- function(x){
  return ((x-min(x)) / (max(x)-min(x)))
}
df.new$Administrative_Duration<- normalize(df.new$Administrative_Duration)
df.new$ProductRelated<- normalize(df.new$ProductRelated)
df.new$ProductRelated_Duration<- normalize(df.new$ProductRelated_Duration)
df.new$BounceRates<- normalize(df.new$BounceRates)
df.new$ExitRates<- normalize(df.new$ExitRates)
head(df.new)

##   Administrative Administrative_Duration Informational
Informational_Duration
## 1              0           0.0002941393             0
0
## 2              0           0.0002941393             0
0
## 3              0           0.0000000000             0
-1
## 4              0           0.0002941393             0
0
## 5              0           0.0002941393             0
0
## 6              0           0.0002941393             0
0
##   ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
## 1    0.001418440            1.563122e-05  1.00000000  1.000000          0
## 2    0.002836879            1.016029e-03  0.00000000  0.500000          0
## 3    0.001418440            0.000000e+00  1.00000000  1.000000          0
## 4    0.002836879            5.731448e-05  0.25000000  0.700000          0
## 5    0.014184397            9.824223e-03  0.10000000  0.250000          0
## 6    0.026950355            2.426226e-03  0.07894737  0.122807          0
##   SpecialDay Month OperatingSystems Browser Region TrafficType VisitorType
```

```
## 1          0     3                 1       1      1            1           3
## 2          0     3                 2       2      1            2           3
## 3          0     3                 4       1      9            3           3
## 4          0     3                 3       2      2            4           3
## 5          0     3                 3       3      1            4           3
## 6          0     3                 2       2      1            3           3
##   Weekend
## 1       1
## 2       1
## 3       1
## 4       1
## 5       2
## 6       1
```

```
#Defining the number of clusters in our dataset
result<- kmeans(df.new,2)

#Previewing the no. of records in each cluster
result$size
```

```
## [1] 11957   256
```

```
#Checking how our labels have been placed.
table(result$cluster, df.class)
```

```
##    df.class
##     FALSE  TRUE
##   1 10120  1837
##   2   185    71
```

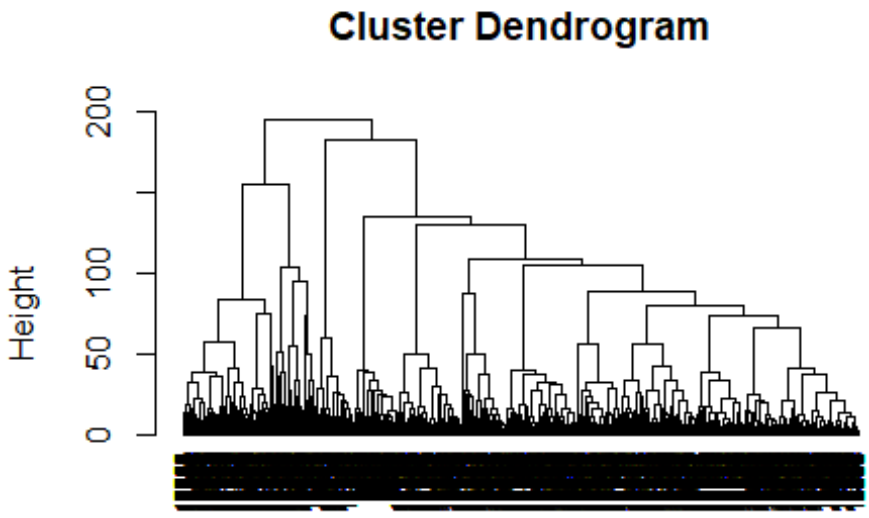
We can see that first cluster correctly classified 10,229 values and classified 1835 values incorrectly. The second cluster classified 73 values correctly and classified 193 values incorrectly.

2. Hierarchical Clustering

```
#Scaling the data
df.h <- scale(df)
head(df.h)
```

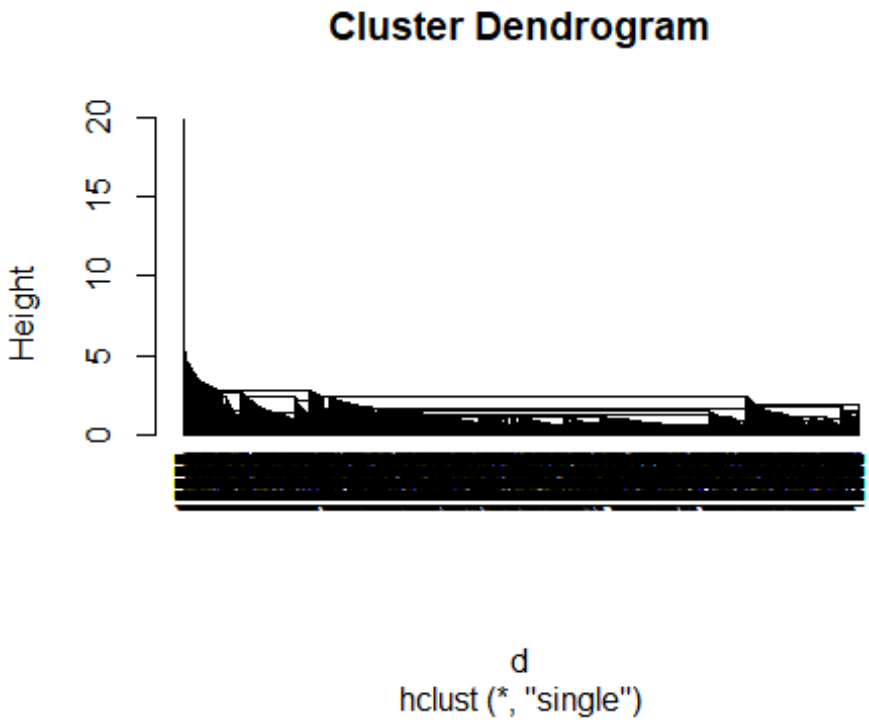
```
##   Administrative Administrative_Duration Informational
Informational_Duration
## 1     -0.7023784              -0.4601187    -0.3985772            -
0.2462710
## 2     -0.7023784              -0.4601187    -0.3985772            -
0.2462710
## 3     -0.7023784              -0.4657536    -0.3985772            -
0.2533428
## 4     -0.7023784              -0.4601187    -0.3985772            -
0.2462710
## 5     -0.7023784              -0.4601187    -0.3985772            -
0.2462710
```

```
## 6     -0.7023784              -0.4601187    -0.3985772             -
0.2462710
##   ProductRelated ProductRelated_Duration  BounceRates  ExitRates
PageValues
## 1     -0.6962374              -0.6288736  3.954082814  3.4272351 -
0.3188341
## 2     -0.6738062              -0.5955341 -0.450250118  1.2650129 -
0.3188341
## 3     -0.6962374              -0.6293945  3.954082814  3.4272351 -
0.3188341
## 4     -0.6738062              -0.6274845  0.650833115  2.1299018 -
0.3188341
## 5     -0.4943562              -0.3019901 -0.009816825  0.1839018 -
0.3188341
## 6     -0.2924750              -0.5485375 -0.102539616 -0.3661372 -
0.3188341
##   SpecialDay     Month OperatingSystems    Browser     Region TrafficType
## 1 -0.3101155 -1.334661       -1.2398307 -0.7940299 -0.8962493 -0.76528498
## 2 -0.3101155 -1.334661       -0.1369864 -0.2091745 -0.8962493 -0.51630590
## 3 -0.3101155 -1.334661        2.0687022 -0.7940299  2.4345662 -0.26732683
## 4 -0.3101155 -1.334661        0.9658579 -0.2091745 -0.4798973 -0.01834776
## 5 -0.3101155 -1.334661        0.9658579  0.3756809 -0.8962493 -0.01834776
## 6 -0.3101155 -1.334661       -0.1369864 -0.2091745 -0.8962493 -0.26732683
##   VisitorType    Weekend    Revenue
## 1   0.4094967 -0.5528287 -0.4302763
## 2   0.4094967 -0.5528287 -0.4302763
## 3   0.4094967 -0.5528287 -0.4302763
## 4   0.4094967 -0.5528287 -0.4302763
## 5   0.4094967  1.8087303 -0.4302763
## 6   0.4094967 -0.5528287 -0.4302763

#Using the dist
d = dist(df.h, method = "euclidean")

#Using the ward method in our hierarchical clustering
res.hc <- hclust(d, method = "ward.D2" )

#Plotting the dendogram of our hierarchical clustering
plot(res.hc, cex = 0.6, hang = -1)
```



I am unable to draw insights from thedendogram

```
#Using the single'smethod to get our dendogram
res.sc <- hclust(d, method = "single" )

#Plotting the results
plot(res.sc, cex = 0.6, hang = -1)
```



Still not able to draw insights from the dendogram

Hierachical Clustering Method did not perform as well, which might have been caused by the high number of columns. We could have reduced them using the Principal Component Analysis.