Project

Smit Patil

1. Data Acquisition

```
#Reading both the airline survey data
airline.data_1 <- read.csv("passenger_survey_data_1.csv", stringsAsFactors = T)
airline.data_2 <- read.csv("passenger_survey_data_2.csv", stringsAsFactors = T)

#Binding both the airline survey data into a single data frame
airline.data <- rbind(airline.data_1, airline.data_2)

#Looking at the head and the structure of the airline data
head(airline.data)
```

```
##   X     id Gender     Customer.Type Age  Type.of.Travel    Class
## 1 0  70172   Male    Loyal Customer  13 Personal Travel Eco Plus
## 2 1   5047   Male disloyal Customer  25 Business travel Business
## 3 2 110028 Female    Loyal Customer  26 Business travel Business
## 4 3  24026 Female    Loyal Customer  25 Business travel Business
## 5 4 119299   Male    Loyal Customer  61 Business travel Business
## 6 5 111157 Female    Loyal Customer  26 Personal Travel      Eco
##   Flight.Distance Inflight.wifi.service Departure.Arrival.time.convenient
## 1             460                     3                                 4
## 2             235                     3                                 2
## 3            1142                     2                                 2
## 4             562                     2                                 5
## 5             214                     3                                 3
## 6            1180                     3                                 4
##   Ease.of.Online.booking Gate.location Food.and.drink Online.boarding
## 1                      3             1              5               3
## 2                      3             3              1               3
## 3                      2             2              5               5
## 4                      5             5              2               2
## 5                      3             3              4               5
## 6                      2             1              1               2
##   Seat.comfort Inflight.entertainment On.board.service Leg.room.service
## 1            5                      5                4                3
## 2            1                      1                1                5
## 3            5                      5                4                3
## 4            2                      2                2                5
## 5            5                      3                3                4
## 6            1                      1                3                4
##   Baggage.handling Checkin.service Inflight.service Cleanliness
## 1                4               4                5           5
## 2                3               1                4           1
## 3                4               4                4           5
```

```
## 4                3              1                4           2
## 5                4              3                3           3
## 6                4              4                4           1
##   Departure.Delay.in.Minutes Arrival.Delay.in.Minutes            satisfaction
## 1                         25                       18 neutral or dissatisfied
## 2                          1                        6 neutral or dissatisfied
## 3                          0                        0               satisfied
## 4                         11                        9 neutral or dissatisfied
## 5                          0                        0               satisfied
## 6                          0                        0 neutral or dissatisfied
```

```
str(airline.data)
```

```
## 'data.frame':    129880 obs. of  25 variables:
##  $ X                               : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ id                              : int  70172 5047 110028 24026 119299 111157 82113 96462 79485 65
##  $ Gender                          : Factor w/ 2 levels "Female","Male": 2 2 1 1 2 1 2 1 1 2 ...
##  $ Customer.Type                   : Factor w/ 2 levels "disloyal Customer",..: 2 1 2 2 2 2 2 2 2 1
##  $ Age                             : int  13 25 26 25 61 26 47 52 41 20 ...
##  $ Type.of.Travel                  : Factor w/ 2 levels "Business travel",..: 2 1 1 1 1 2 2 1 1 1 .
##  $ Class                           : Factor w/ 3 levels "Business","Eco",..: 3 1 1 1 1 2 2 1 1 2 ..
##  $ Flight.Distance                 : int  460 235 1142 562 214 1180 1276 2035 853 1061 ...
##  $ Inflight.wifi.service           : int  3 3 2 2 3 3 2 4 1 3 ...
##  $ Departure.Arrival.time.convenient: int  4 2 2 5 3 4 4 3 2 3 ...
##  $ Ease.of.Online.booking          : int  3 3 2 5 3 2 2 4 2 3 ...
##  $ Gate.location                   : int  1 3 2 5 3 1 3 4 2 4 ...
##  $ Food.and.drink                  : int  5 1 5 2 4 1 2 5 4 2 ...
##  $ Online.boarding                 : int  3 3 5 2 5 2 2 5 3 3 ...
##  $ Seat.comfort                    : int  5 1 5 2 5 1 2 5 3 3 ...
##  $ Inflight.entertainment          : int  5 1 5 2 3 1 2 5 1 2 ...
##  $ On.board.service                : int  4 1 4 2 3 3 3 5 1 2 ...
##  $ Leg.room.service                : int  3 5 3 5 4 4 3 5 2 3 ...
##  $ Baggage.handling                : int  4 3 4 3 4 4 4 5 1 4 ...
##  $ Checkin.service                 : int  4 1 4 1 3 4 3 4 4 4 ...
##  $ Inflight.service                : int  5 4 4 4 3 4 5 5 1 3 ...
##  $ Cleanliness                     : int  5 1 5 2 3 1 2 4 2 2 ...
##  $ Departure.Delay.in.Minutes      : int  25 1 0 11 0 0 9 4 0 0 ...
##  $ Arrival.Delay.in.Minutes        : num  18 6 0 9 0 0 23 0 0 0 ...
##  $ satisfaction                    : Factor w/ 2 levels "neutral or dissatisfied",..: 1 1 2 1 2 1 1
```

```
#Removing column 1: x and column 2: id, as they are redundant for this application
airline.data <- airline.data[,-(1:2)]

#Printing the summary of the airline data
summary(airline.data)
```

```
##     Gender                Customer.Type         Age
##  Female:65899   disloyal Customer: 23780   Min.   : 7.00
##  Male  :63981   Loyal Customer   :106100   1st Qu.:27.00
##                                            Median :40.00
##                                            Mean   :39.43
##                                            3rd Qu.:51.00
##                                            Max.   :85.00
```

```
##
##          Type.of.Travel       Class       Flight.Distance Inflight.wifi.service
##  Business travel:89693   Business:62160   Min.   :  31    Min.   :0.000
##  Personal Travel:40187   Eco     :58309   1st Qu.: 414    1st Qu.:2.000
##                          Eco Plus: 9411   Median : 844    Median :3.000
##                                           Mean   :1190    Mean   :2.729
##                                           3rd Qu.:1744    3rd Qu.:4.000
##                                           Max.   :4983    Max.   :5.000
##
##  Departure.Arrival.time.convenient Ease.of.Online.booking Gate.location
##  Min.   :0.000                     Min.   :0.000          Min.   :0.000
##  1st Qu.:2.000                     1st Qu.:2.000          1st Qu.:2.000
##  Median :3.000                     Median :3.000          Median :3.000
##  Mean   :3.058                     Mean   :2.757          Mean   :2.977
##  3rd Qu.:4.000                     3rd Qu.:4.000          3rd Qu.:4.000
##  Max.   :5.000                     Max.   :5.000          Max.   :5.000
##
##  Food.and.drink  Online.boarding  Seat.comfort   Inflight.entertainment
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000
##  Median :3.000   Median :3.000   Median :4.000   Median :4.000
##  Mean   :3.205   Mean   :3.253   Mean   :3.441   Mean   :3.358
##  3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:5.000   3rd Qu.:4.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
##
##  On.board.service Leg.room.service Baggage.handling Checkin.service
##  Min.   :0.000    Min.   :0.000    Min.   :1.000    Min.   :0.000
##  1st Qu.:2.000    1st Qu.:2.000    1st Qu.:3.000    1st Qu.:3.000
##  Median :4.000    Median :4.000    Median :4.000    Median :3.000
##  Mean   :3.383    Mean   :3.351    Mean   :3.632    Mean   :3.306
##  3rd Qu.:4.000    3rd Qu.:4.000    3rd Qu.:5.000    3rd Qu.:4.000
##  Max.   :5.000    Max.   :5.000    Max.   :5.000    Max.   :5.000
##
##  Inflight.service  Cleanliness    Departure.Delay.in.Minutes
##  Min.   :0.000    Min.   :0.000   Min.   :   0.00
##  1st Qu.:3.000    1st Qu.:2.000   1st Qu.:   0.00
##  Median :4.000    Median :3.000   Median :   0.00
##  Mean   :3.642    Mean   :3.286   Mean   :  14.71
##  3rd Qu.:5.000    3rd Qu.:4.000   3rd Qu.:  12.00
##  Max.   :5.000    Max.   :5.000   Max.   :1592.00
##
##  Arrival.Delay.in.Minutes                  satisfaction
##  Min.   :   0.00          neutral or dissatisfied:73452
##  1st Qu.:   0.00          satisfied              :56428
##  Median :   0.00
##  Mean   :  15.09
##  3rd Qu.:  13.00
##  Max.   :1584.00
##  NA's   :393
```

2. Data Exploration

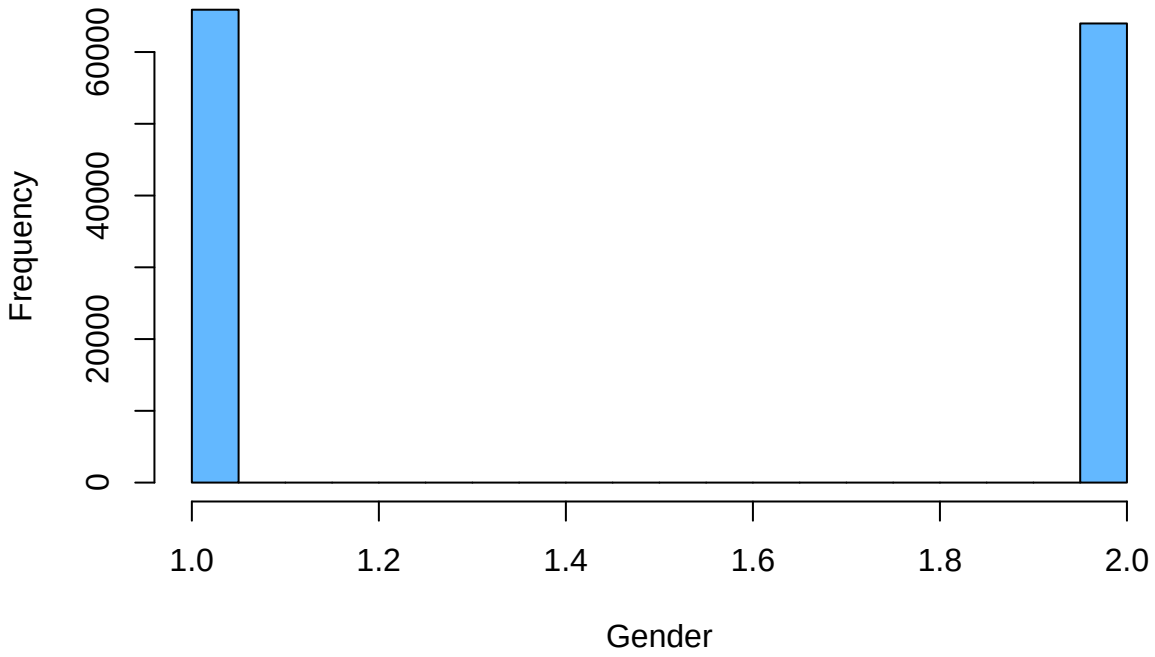
```r
#Importing libraries to calculate outliers and plot the correlation graph
library(ggplot2)
library(hrbrthemes)
library(viridis)
library(corrplot)
library(GGally)

#Looking for any NA values present in the dataset
anyNA(airline.data)
```

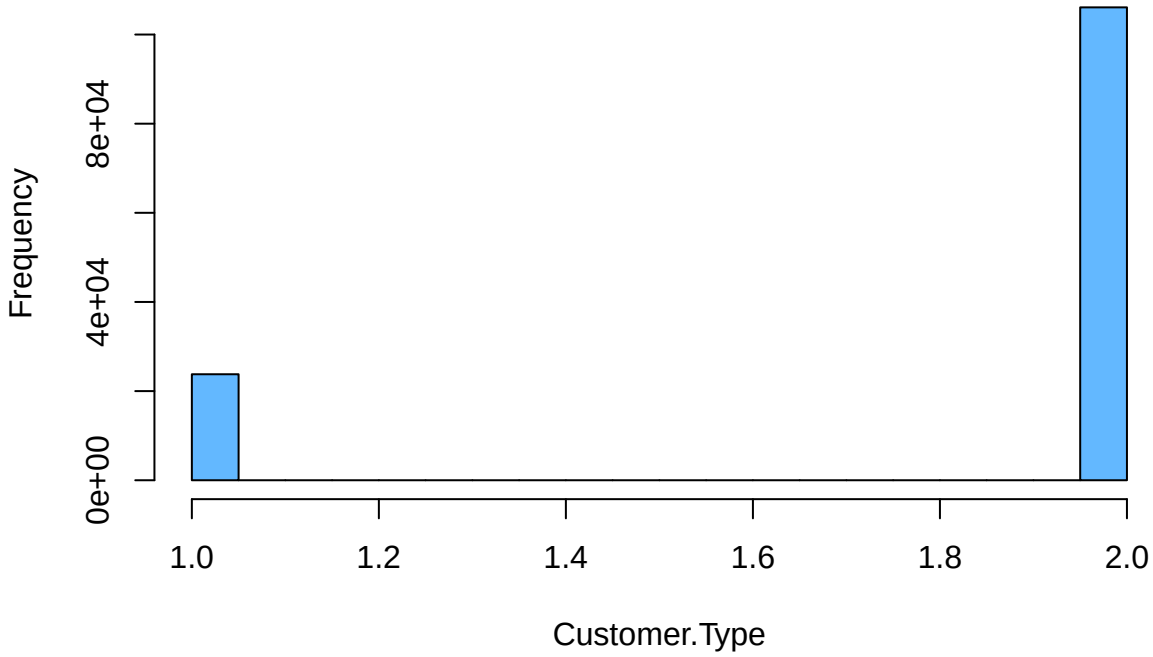
```
## [1] TRUE
```

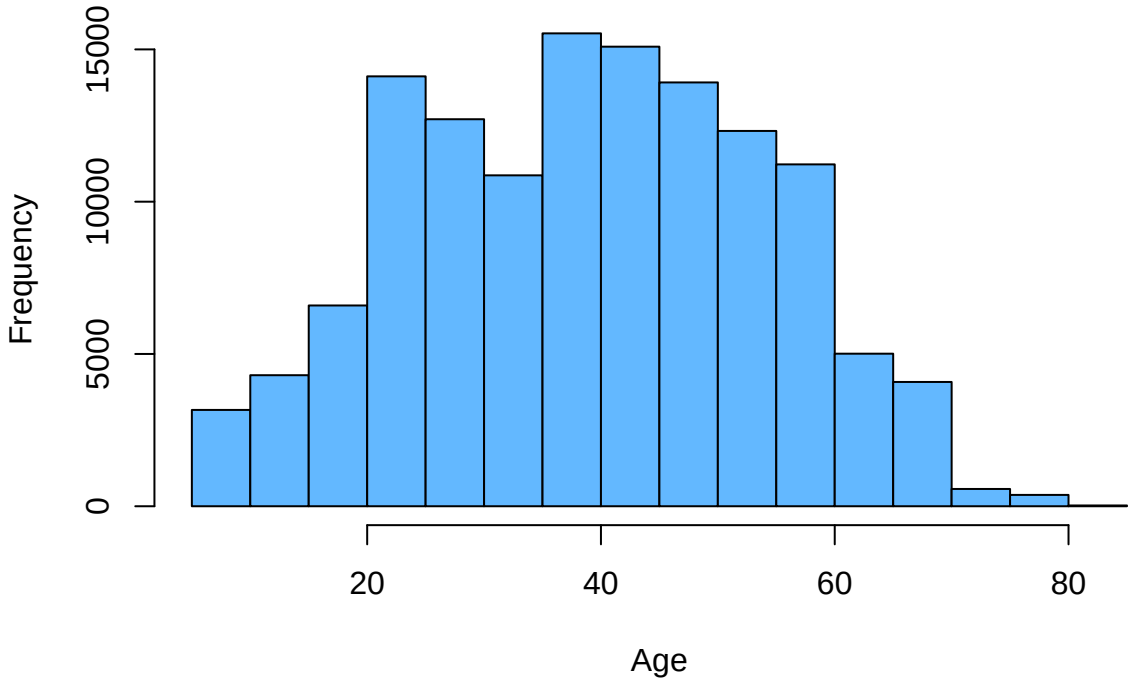
```r
#Finding the column names which have NA values present in them
colnames(airline.data)[colSums(is.na(airline.data)) > 0]
```

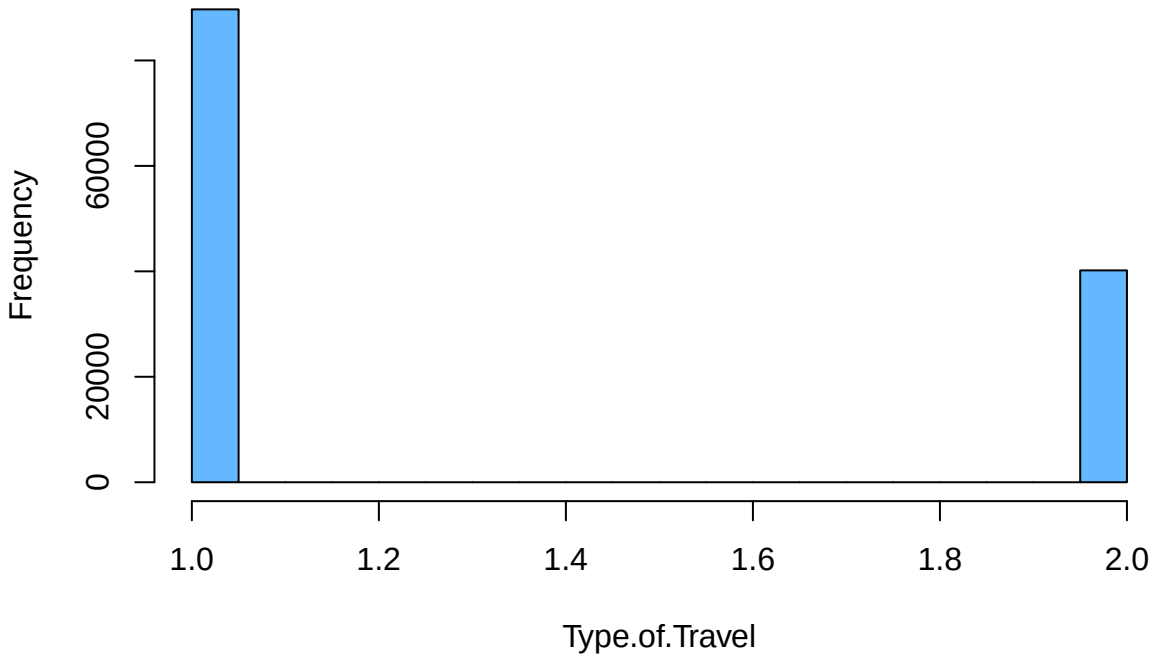
```
## [1] "Arrival.Delay.in.Minutes"
```

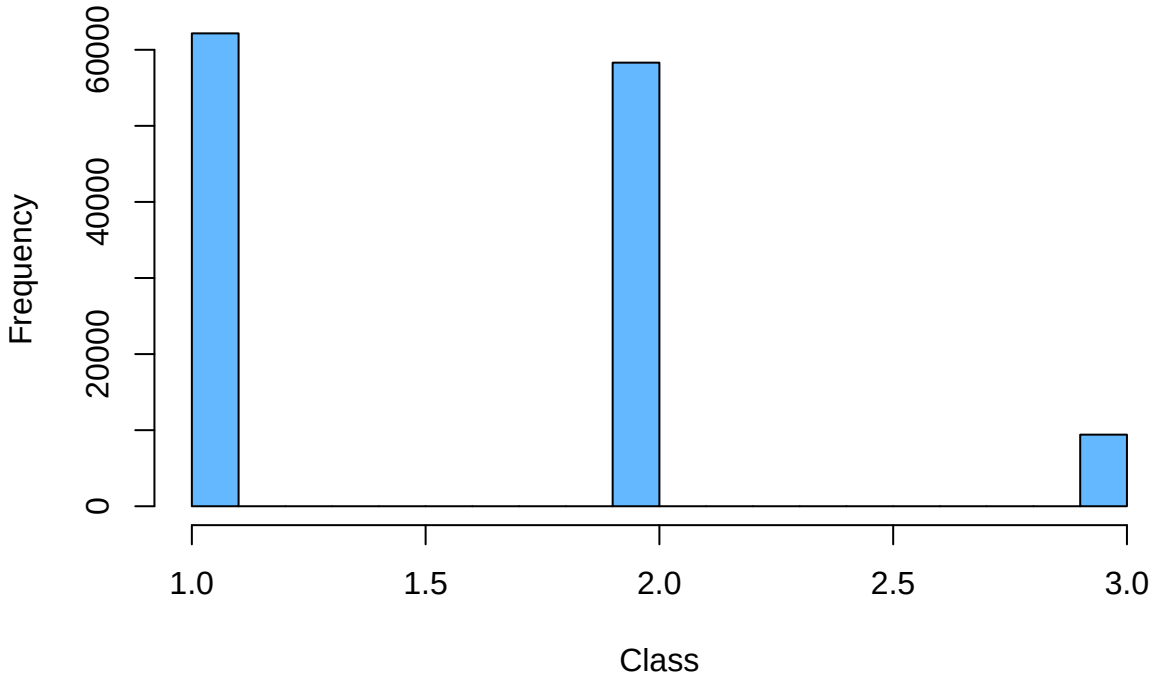
```r
#Using for loop to print histograms of airline data
for(i in 1:ncol(airline.data))
  {
    sprintf("Histogram for: ", colnames(airline.data[i]))
    hist((as.numeric(airline.data[,i])), xlab = colnames(airline.data[i]), col = 'steelblue1', main = NU
 }
```

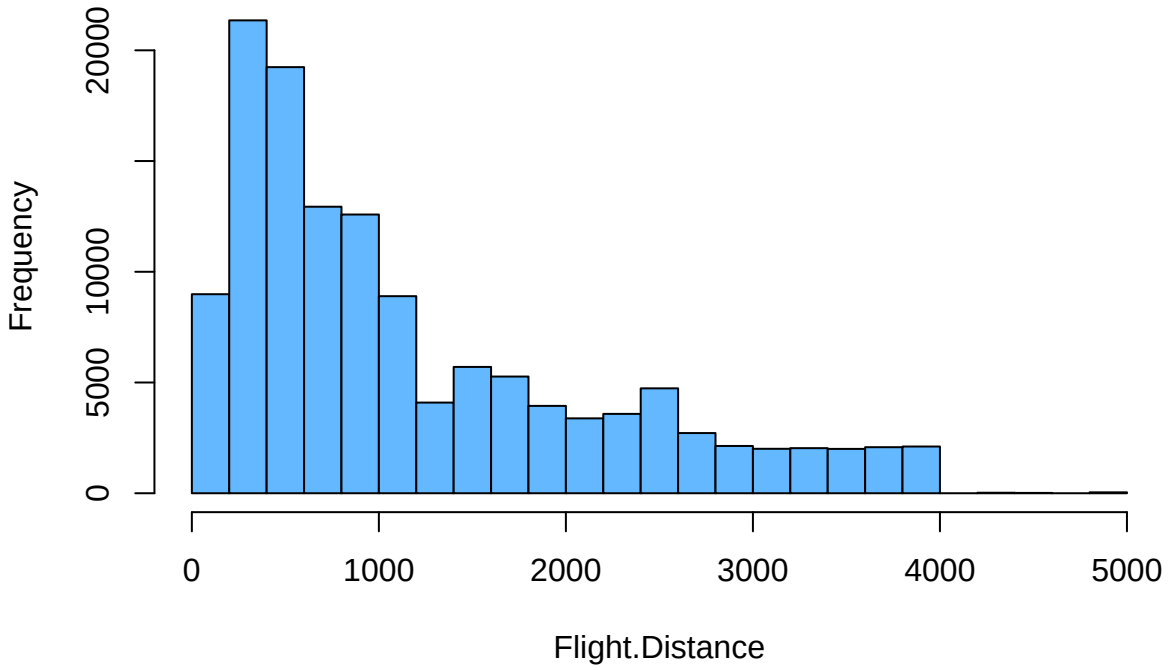


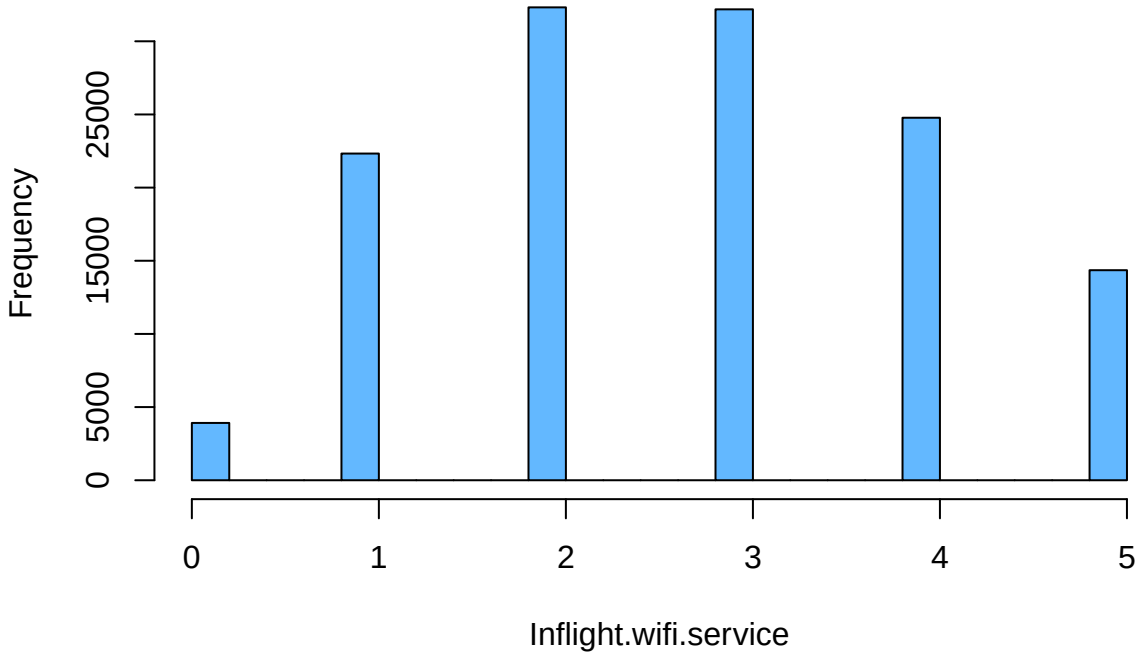

Frequency
0e+00
4e+04
8e+04
1.0
1.2
1.4
1.6
1.8
2.0
Customer.Type

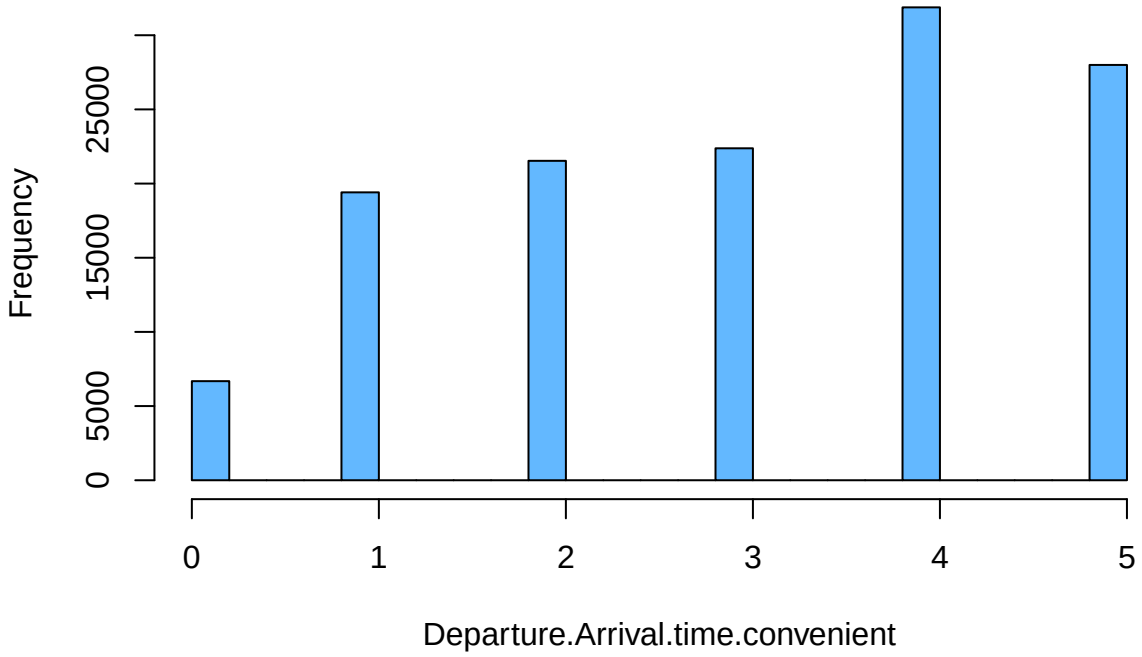

Frequency
0
5000
10000
15000
20
40
60
80
Age

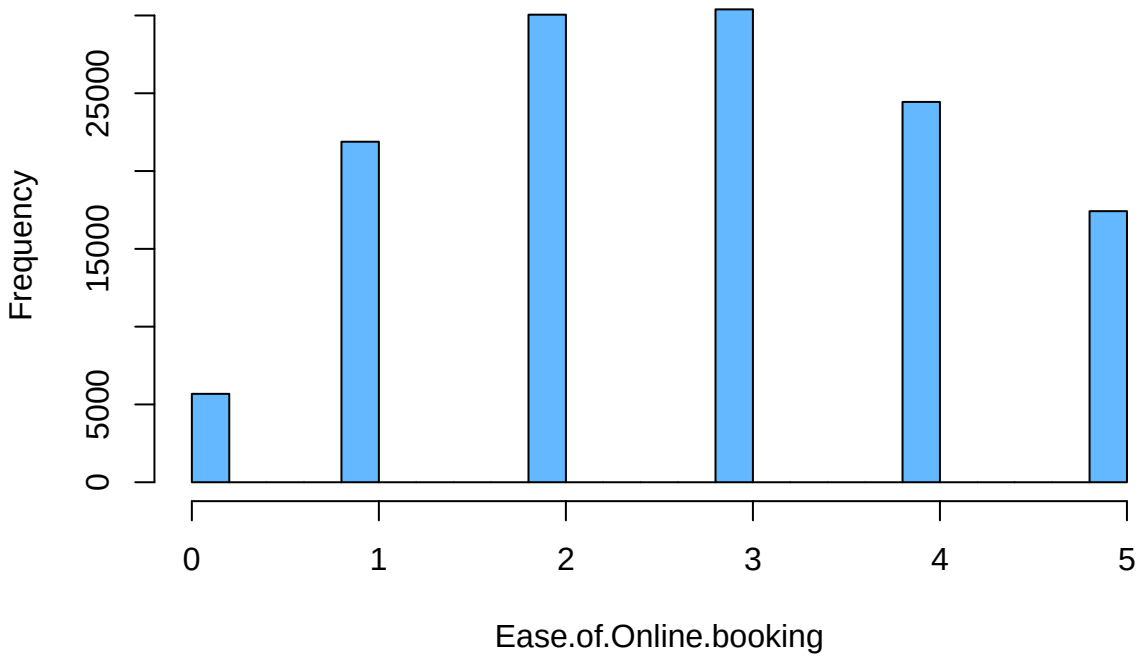

Frequency
0
20000
60000
1.0
1.2
1.4
1.6
1.8
2.0
Type.of.Travel

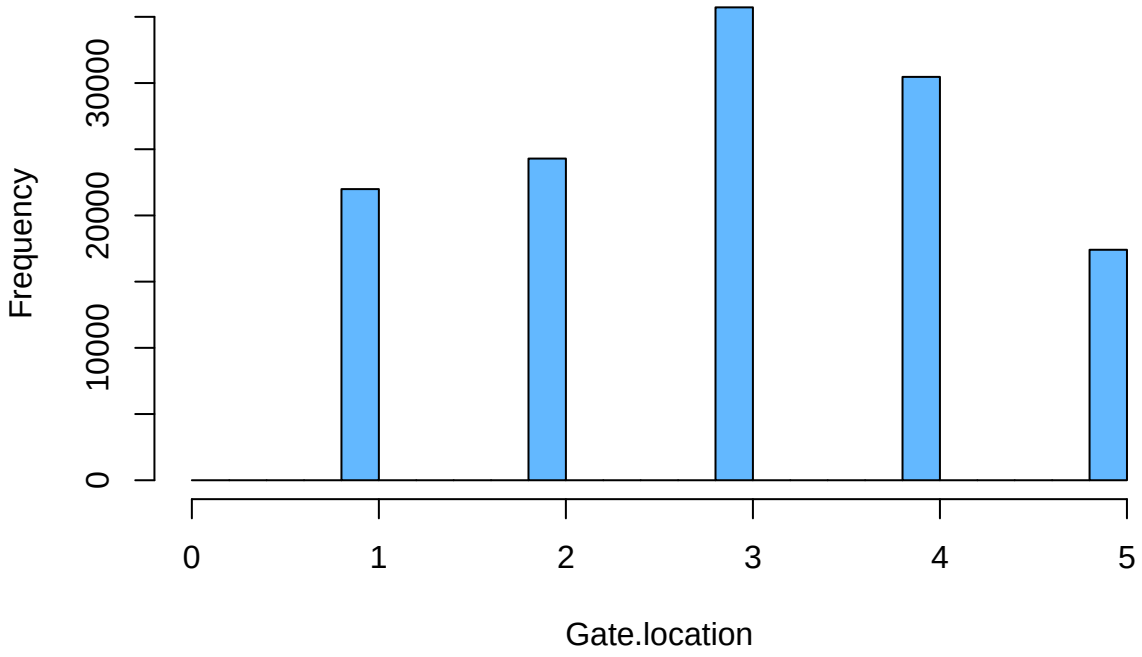

Frequency
0
20000
40000
60000
1.0
1.5
2.0
2.5
3.0
Class

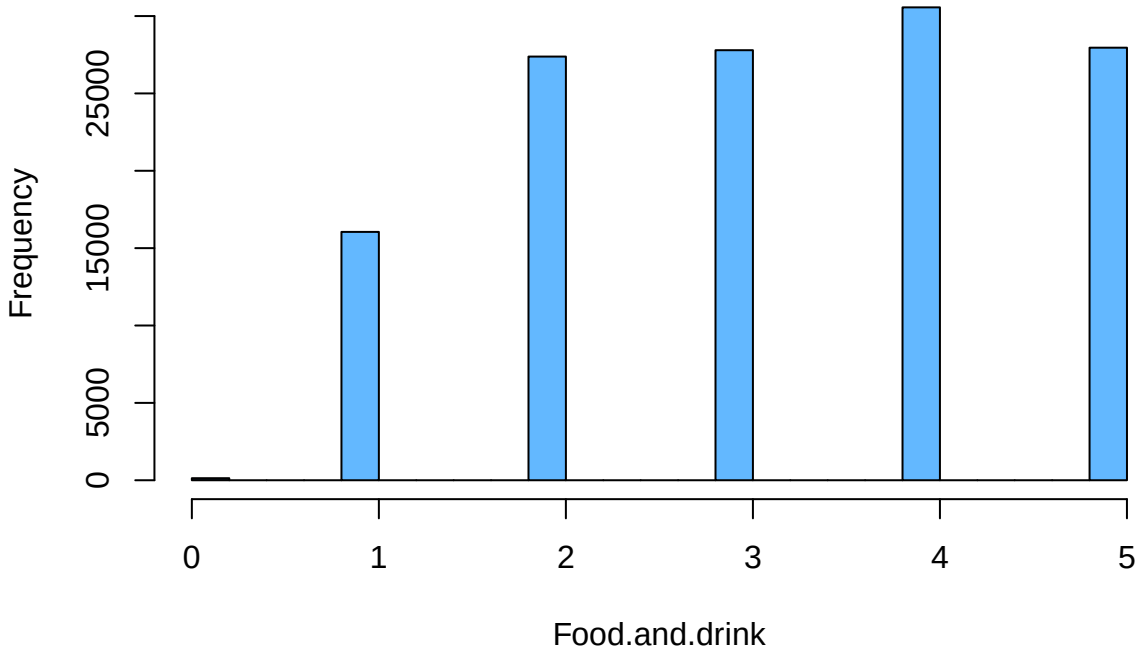

Frequency
0
5000
10000
20000
0
1000
2000
3000
4000
5000
Flight.Distance

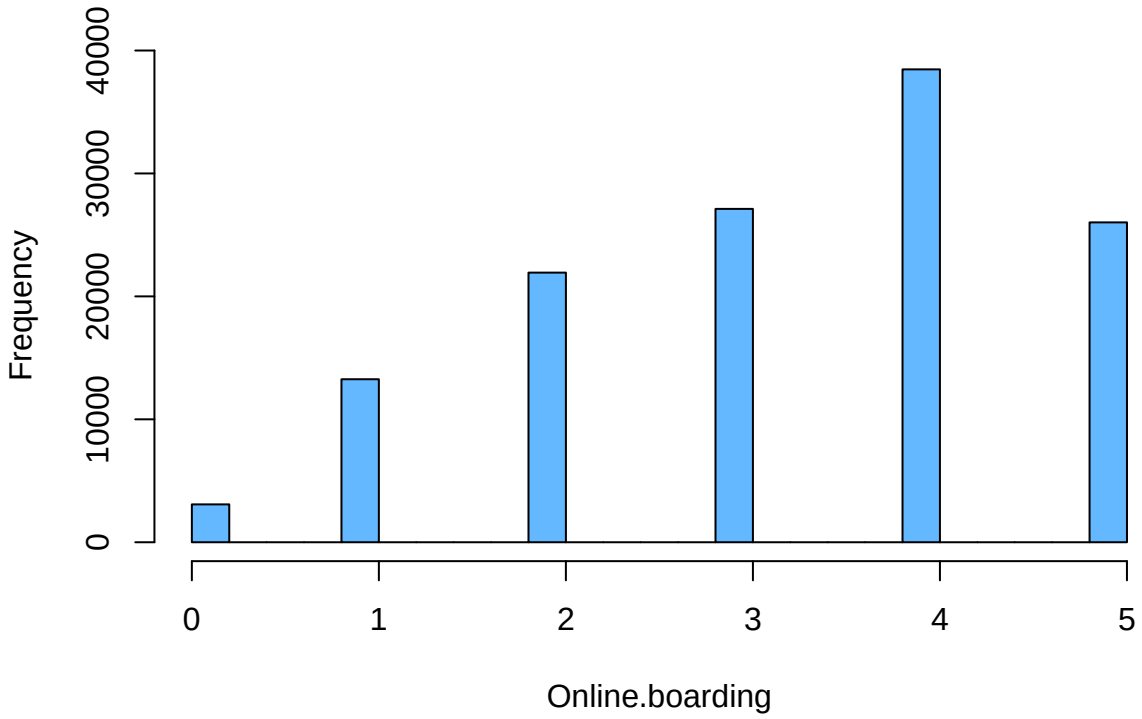

Frequency
0
5000
15000
25000
0
1
2
3
4
5
Inflight.wifi.service

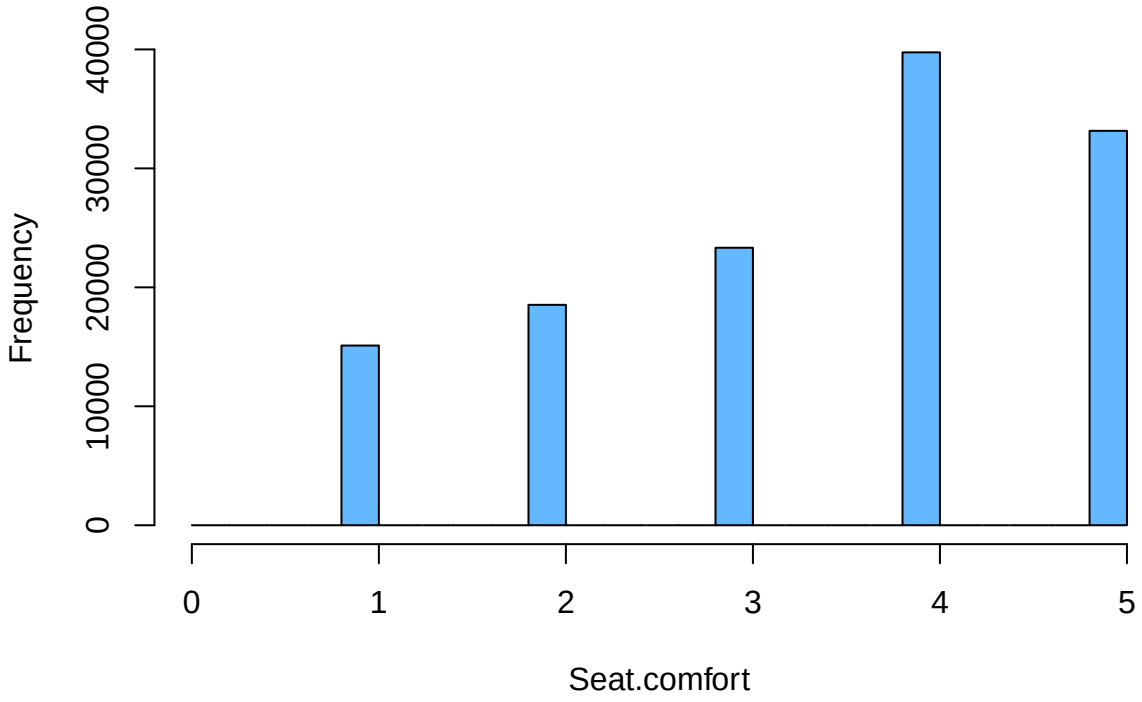

Frequency
0
5000
15000
25000
0
1
2
3
4
5
Departure.Arrival.time.convenient

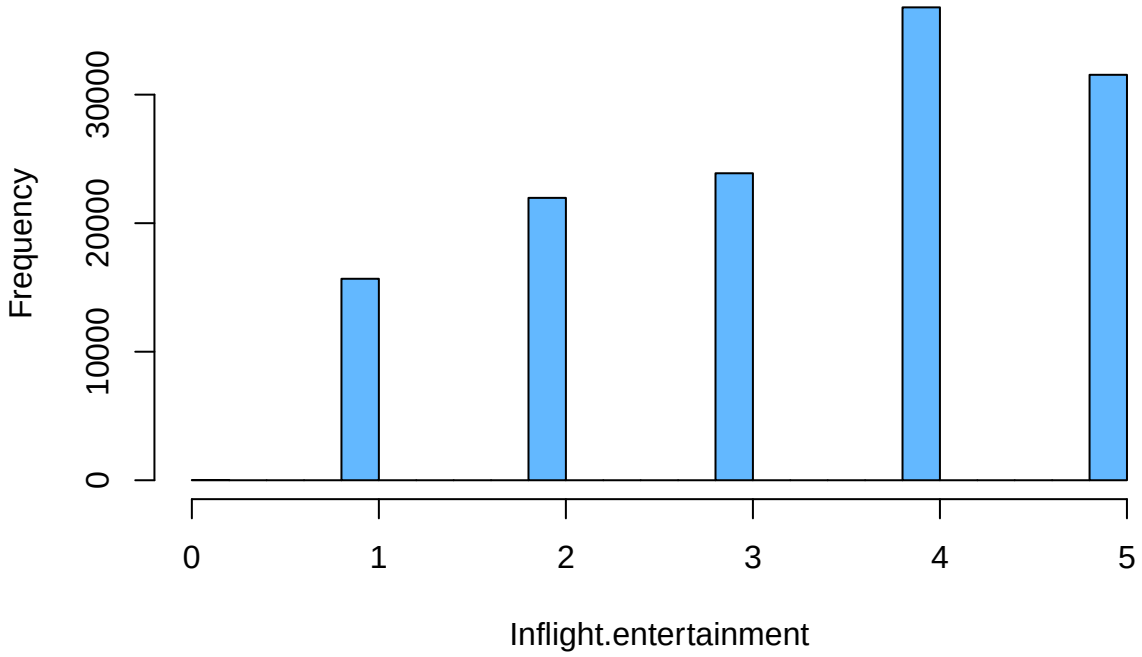

Frequency
0
5000
15000
25000
0
1
2
3
4
5
Ease.of.Online.booking

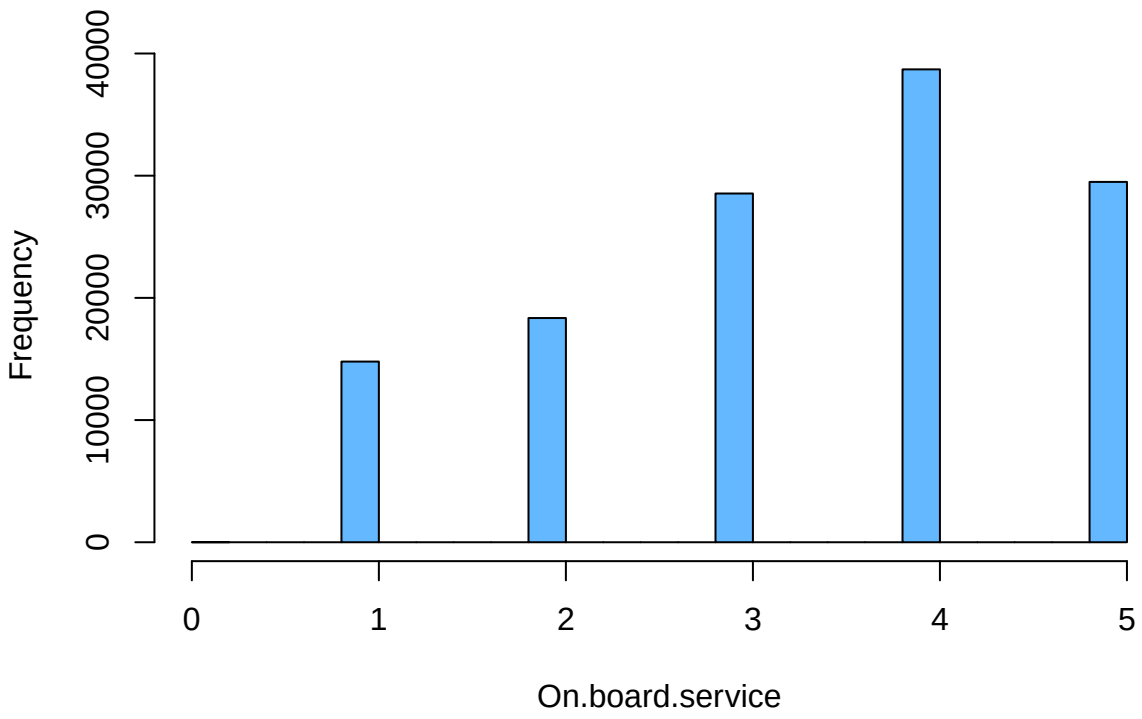

Frequency
0
10000
20000
30000
0
1
2
3
4
5
Gate.location

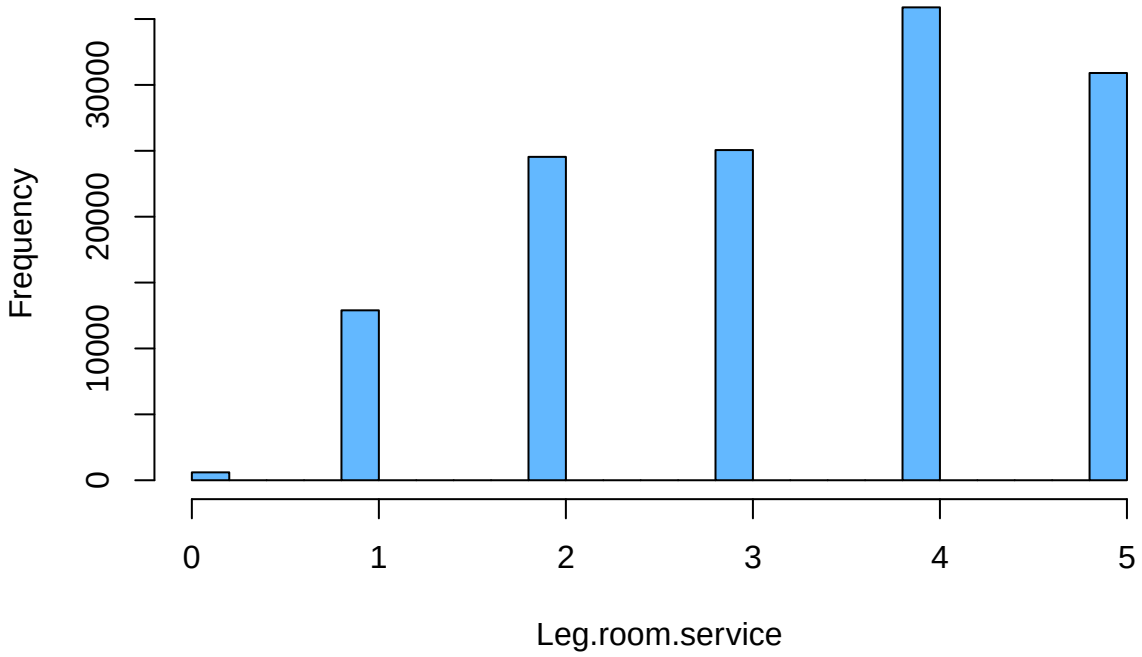

Frequency
0
5000
15000
25000
0
1
2
3
4
5
Food.and.drink

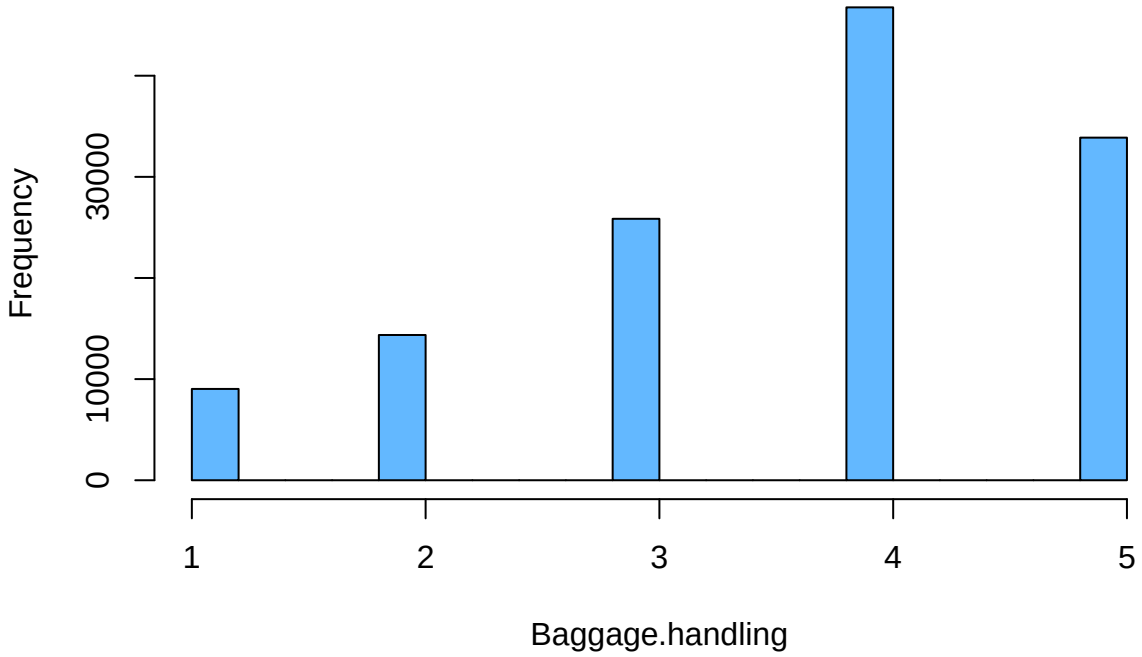

Frequency
0
10000
20000
30000
40000
0
1
2
3
4
5
Online.boarding

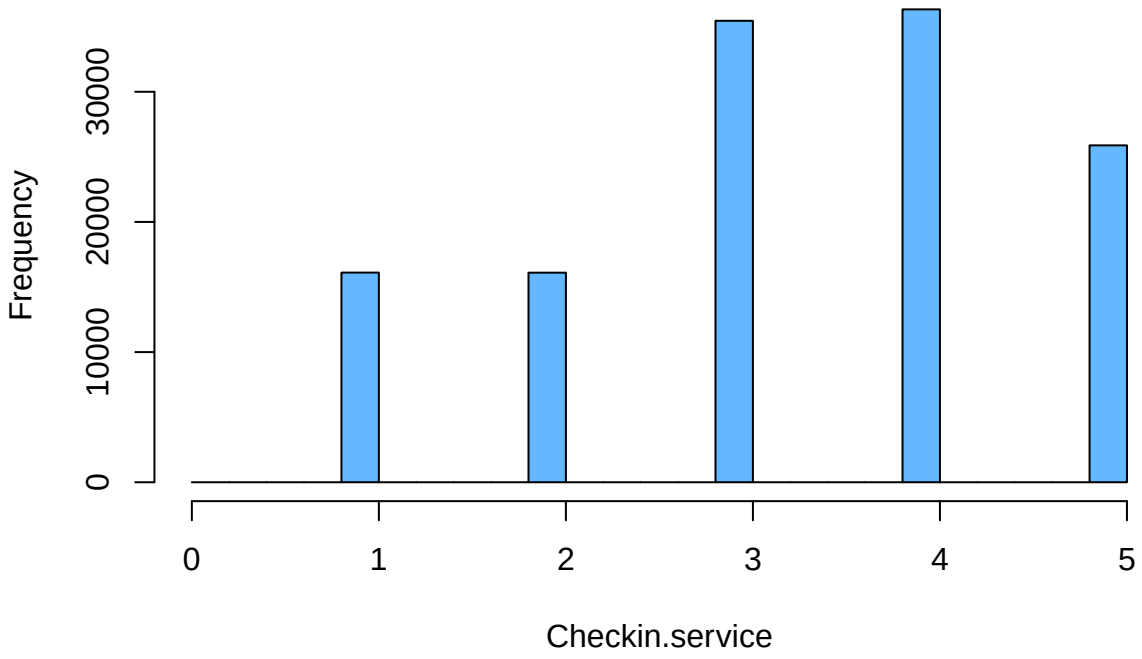

Frequency
0
10000
20000
30000
40000
0
1
2
3
4
5
Seat.comfort

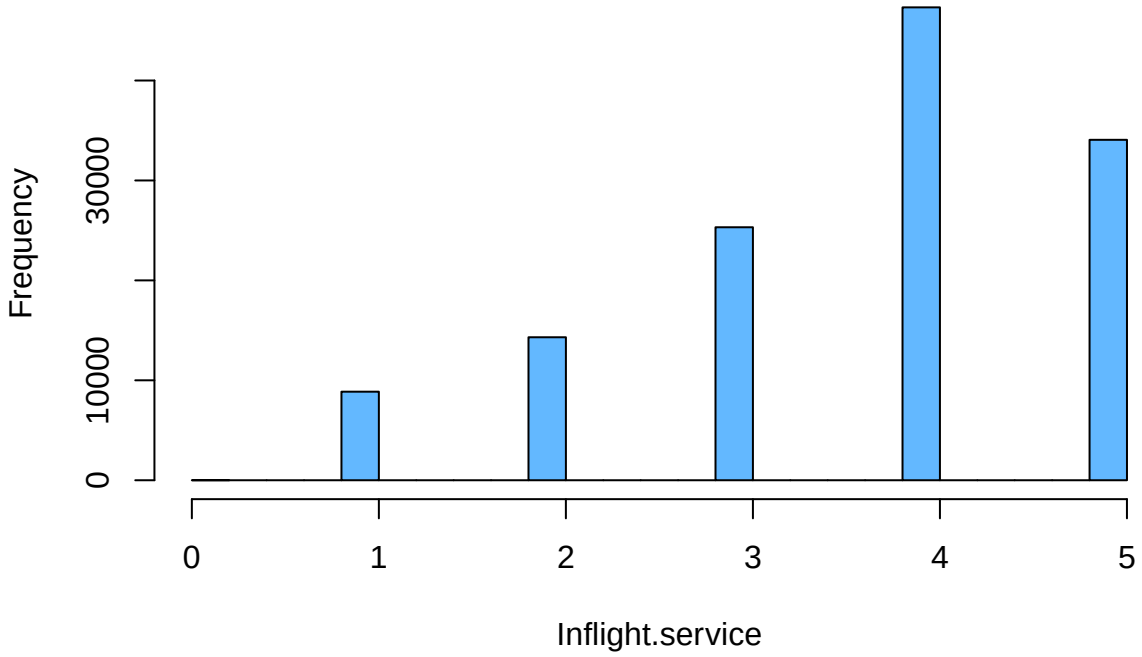

Frequency
0
10000
20000
30000
0
1
2
3
4
5
Inflight.entertainment

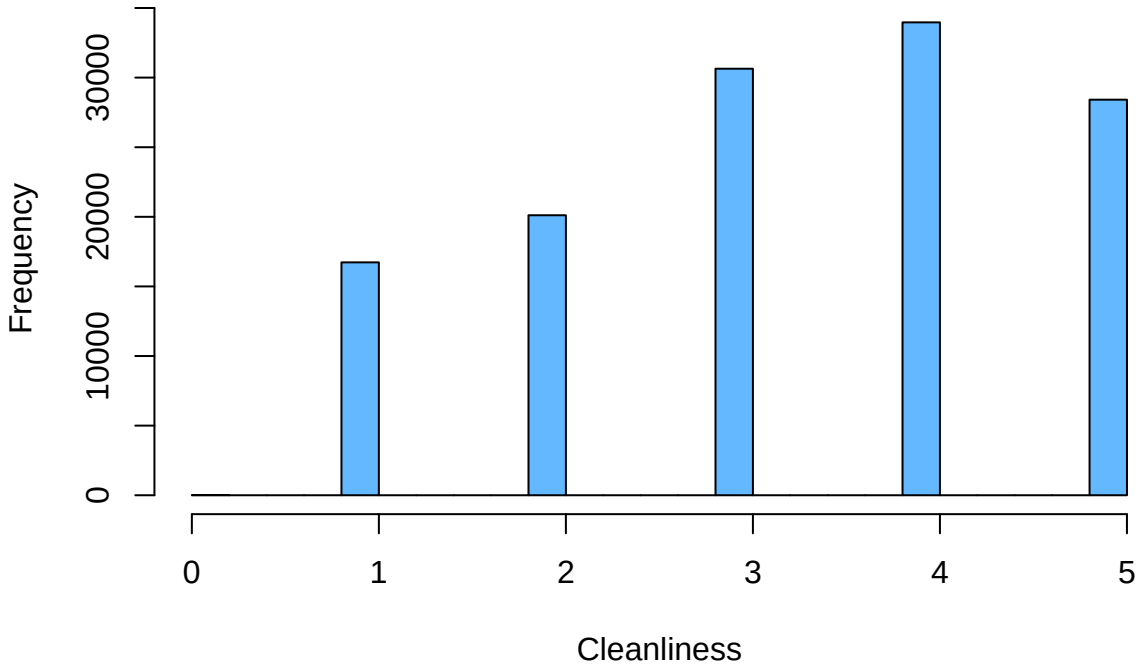

Frequency
0
10000
20000
30000
40000
0
1
2
3
4
5
On.board.service

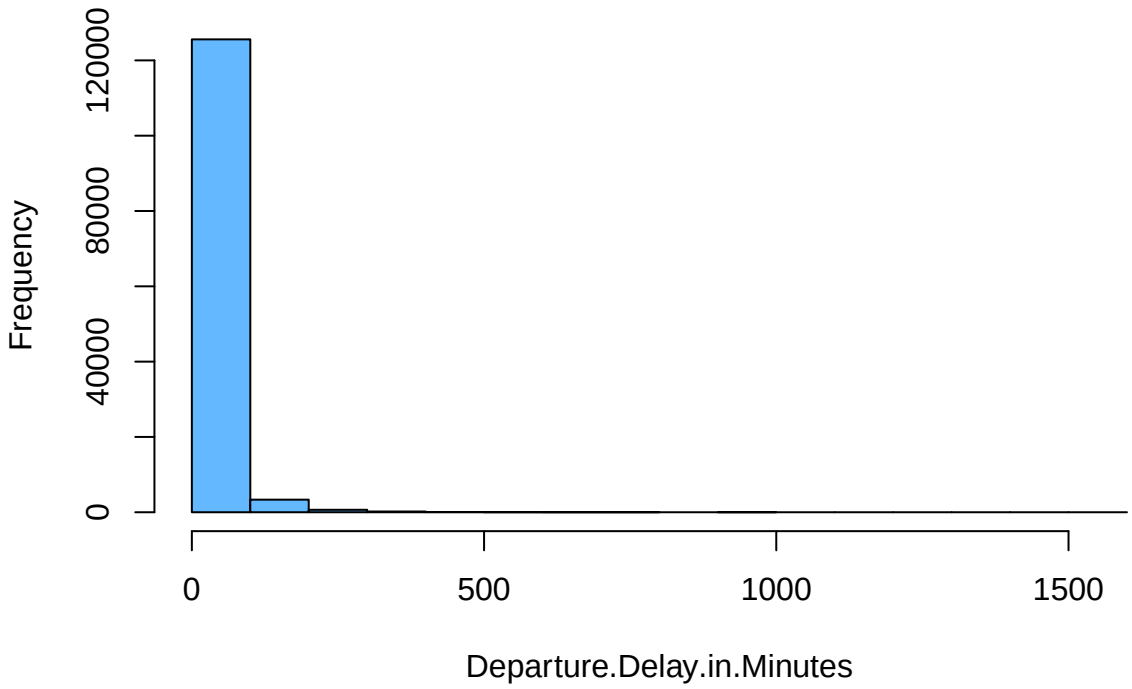

Frequency
0
10000
20000
30000
0
1
2
3
4
5
Leg.room.service

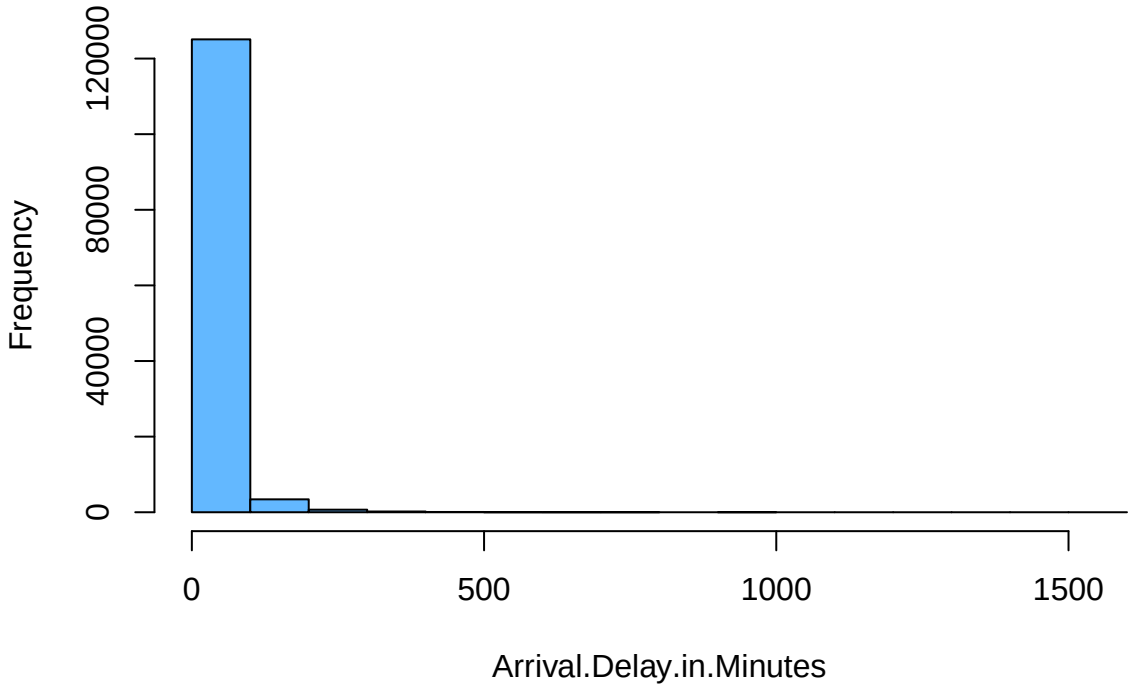

Frequency
0
10000
30000
1
2
3
4
5
Baggage.handling

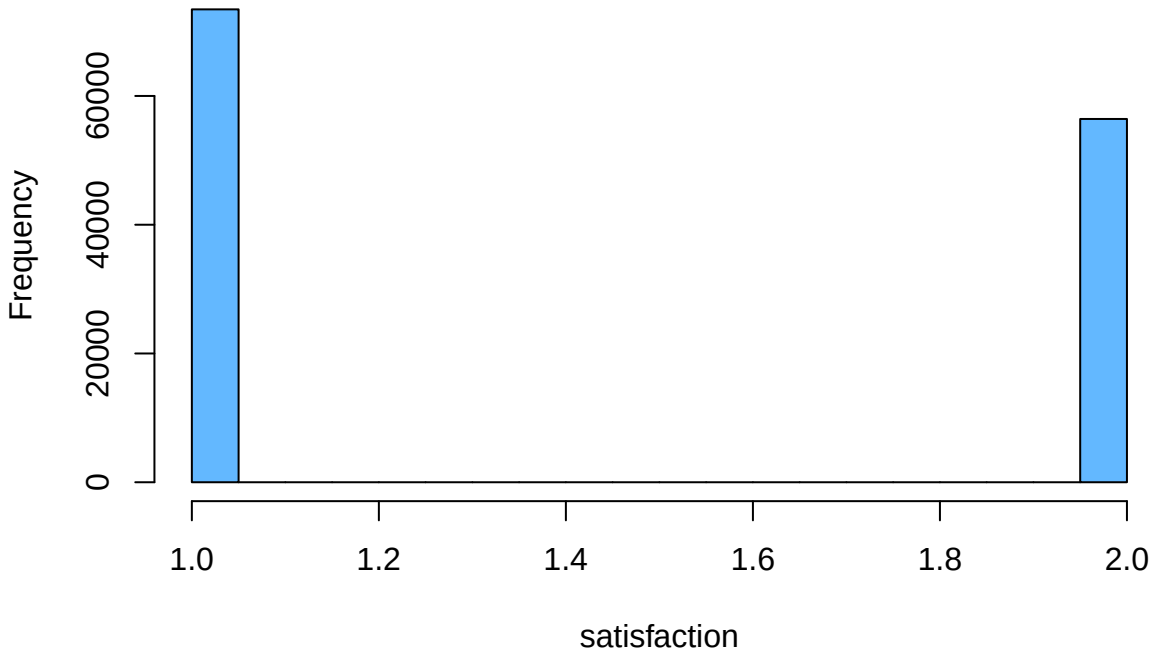

Frequency
0
10000
20000
30000
0
1
2
3
4
5
Checkin.service


Frequency
0
10000
30000
0
1
2
3
4
5
Inflight.service

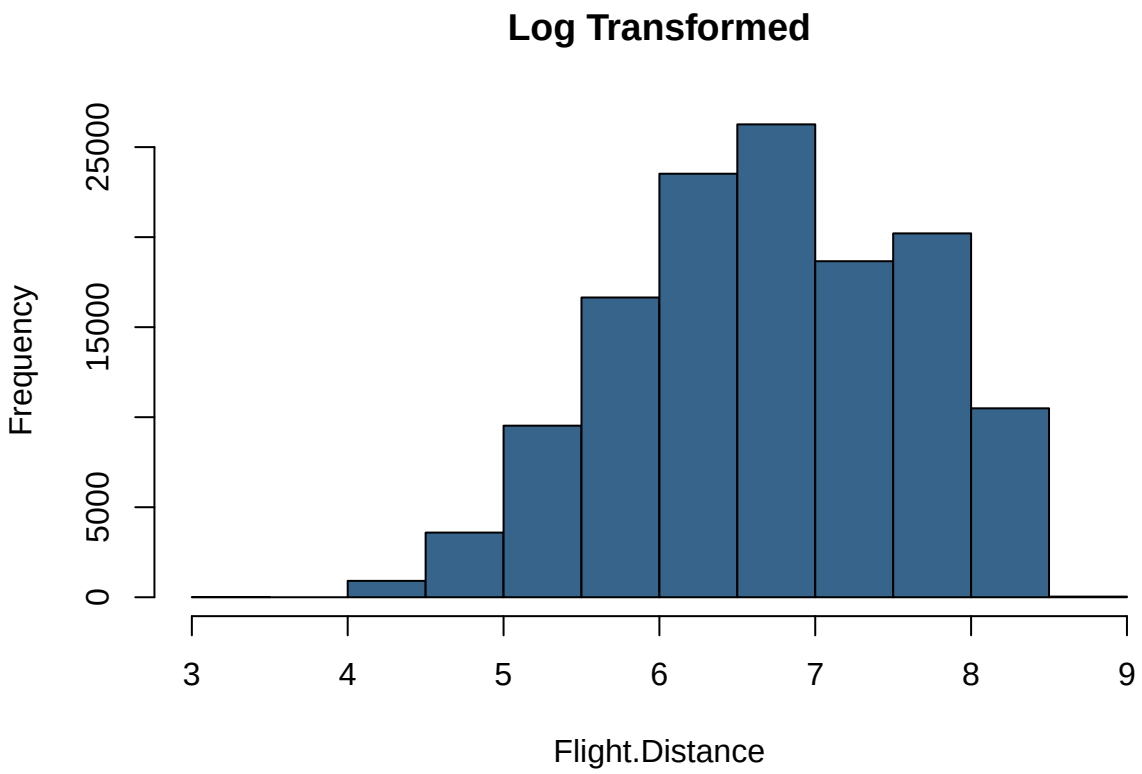

Frequency
0
10000
20000
30000
0
1
2
3
4
5
Cleanliness

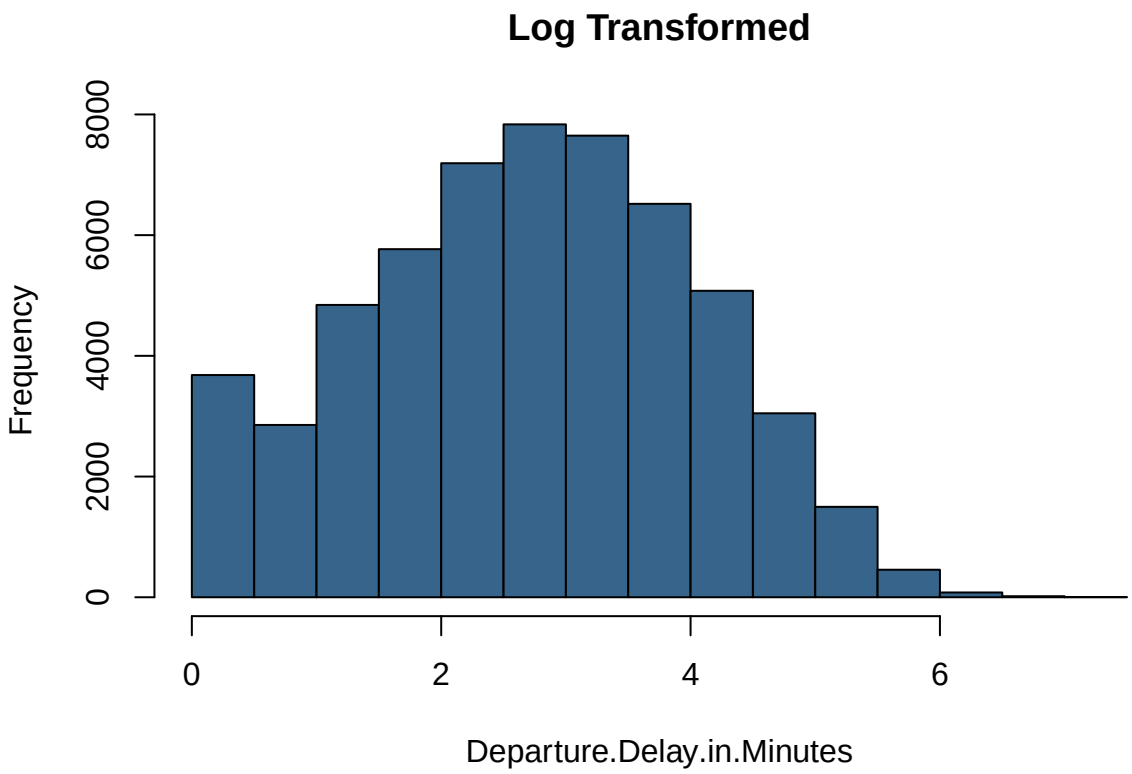

Frequency
0
40000
80000
120000
0
500
1000
1500
Departure.Delay.in.Minutes

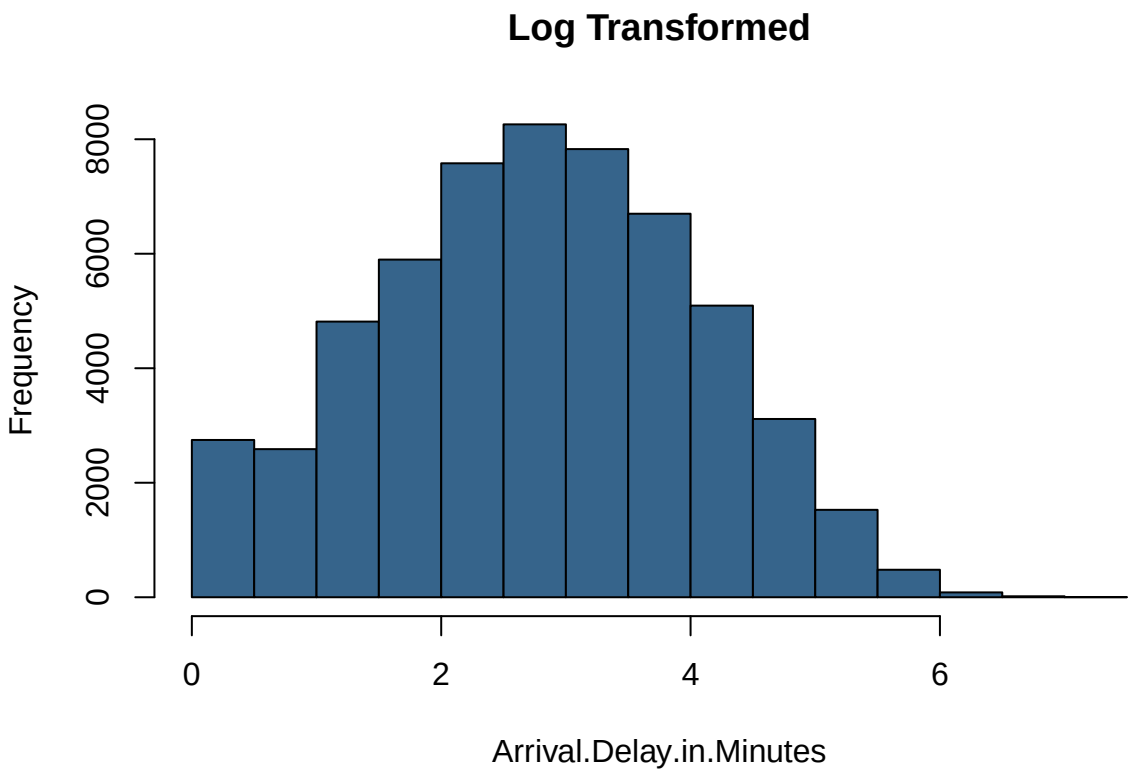

Frequency
0
40000
80000
120000
0
500
1000
1500
Arrival.Delay.in.Minutes



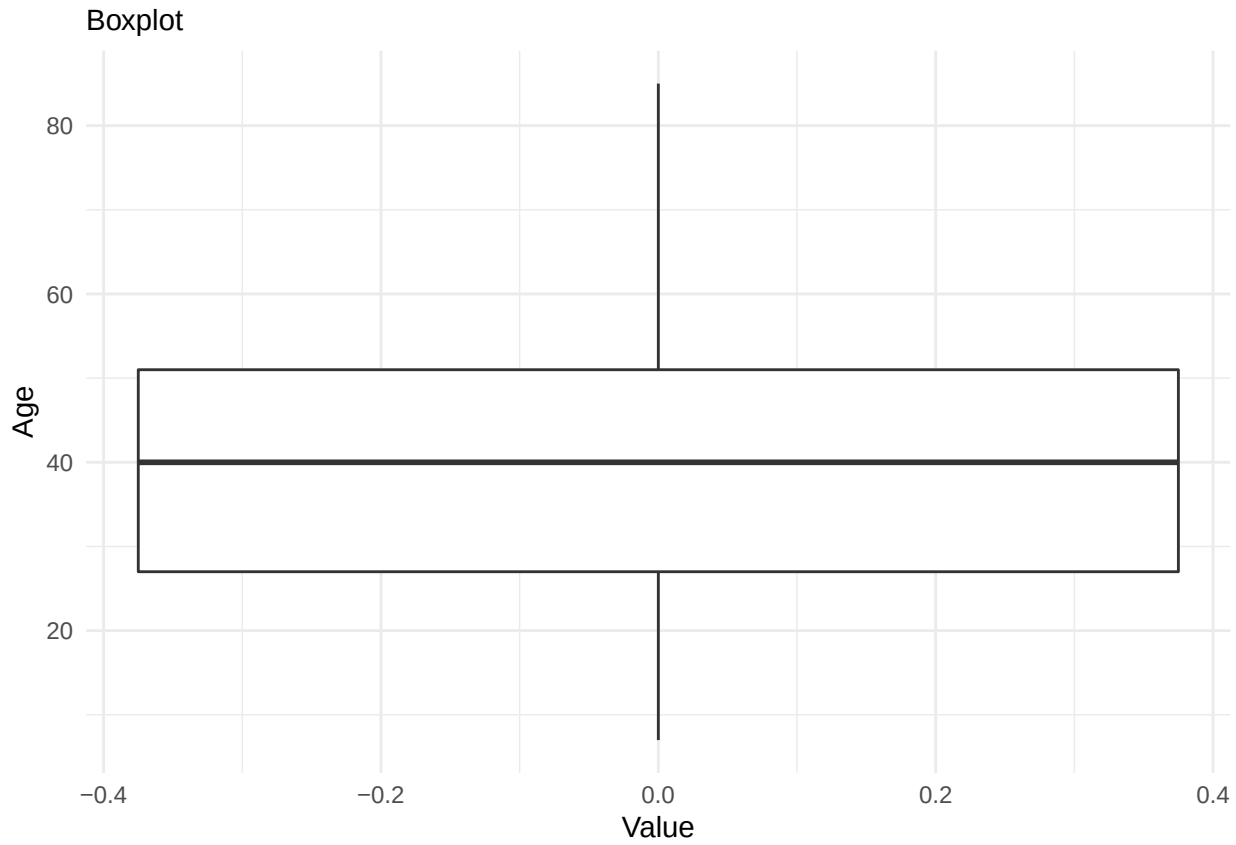

```
#Plotting log transformed histogram for "Flight Distance", "Departure Delay", and "Arrival Delay"
#as they are Rigt Skewed
for (i in c("Flight.Distance", "Departure.Delay.in.Minutes", "Arrival.Delay.in.Minutes"))
  {
    hist(log(airline.data[,i]), col = "steelblue4", main = "Log Transformed", xlab = i)
  }
```


Log Transformed
Frequency
0
5000
15000
25000
3
4
5
6
7
8
9
Flight.Distance


Log Transformed
Frequency
0
2000
4000
6000
8000
0
2
4
6
Departure.Delay.in.Minutes



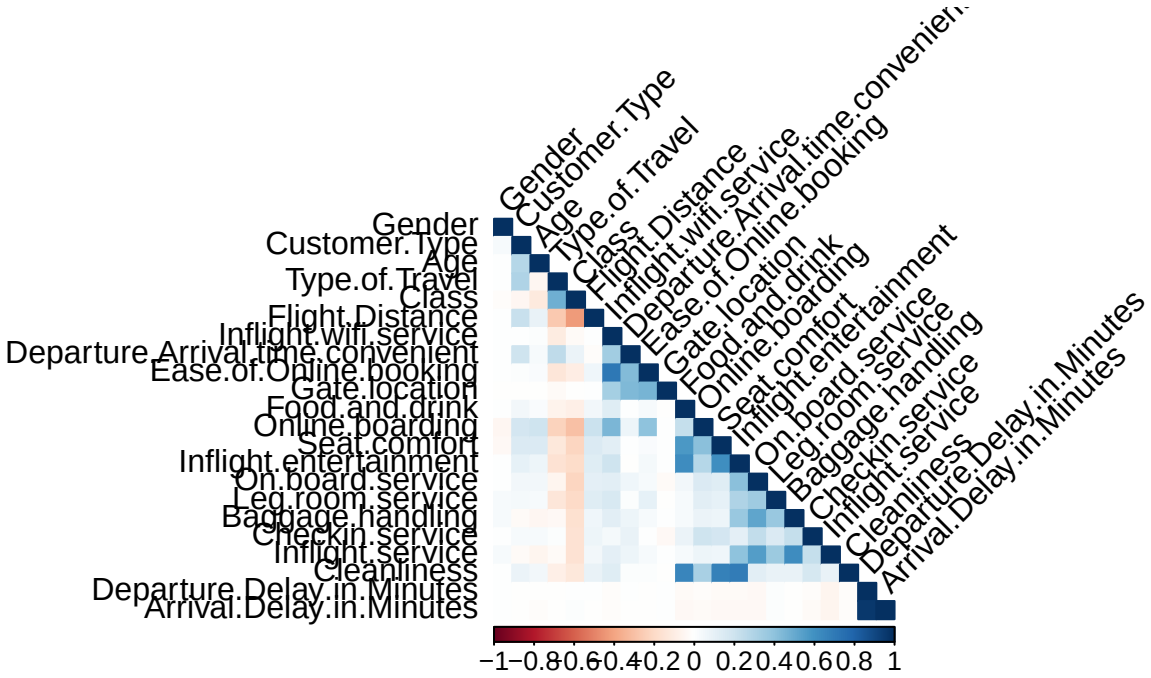

```
#Plotting a boxplot for the "Age" column in the airline data
airline.data %>%
  ggplot( aes(x=NULL, y=Age)) +
    geom_boxplot() +
    theme_minimal() +
    theme(
      legend.position="none",
      plot.title = element_text(size=11)
    ) +
    ggtitle("Boxplot") +
    xlab("Value")
```



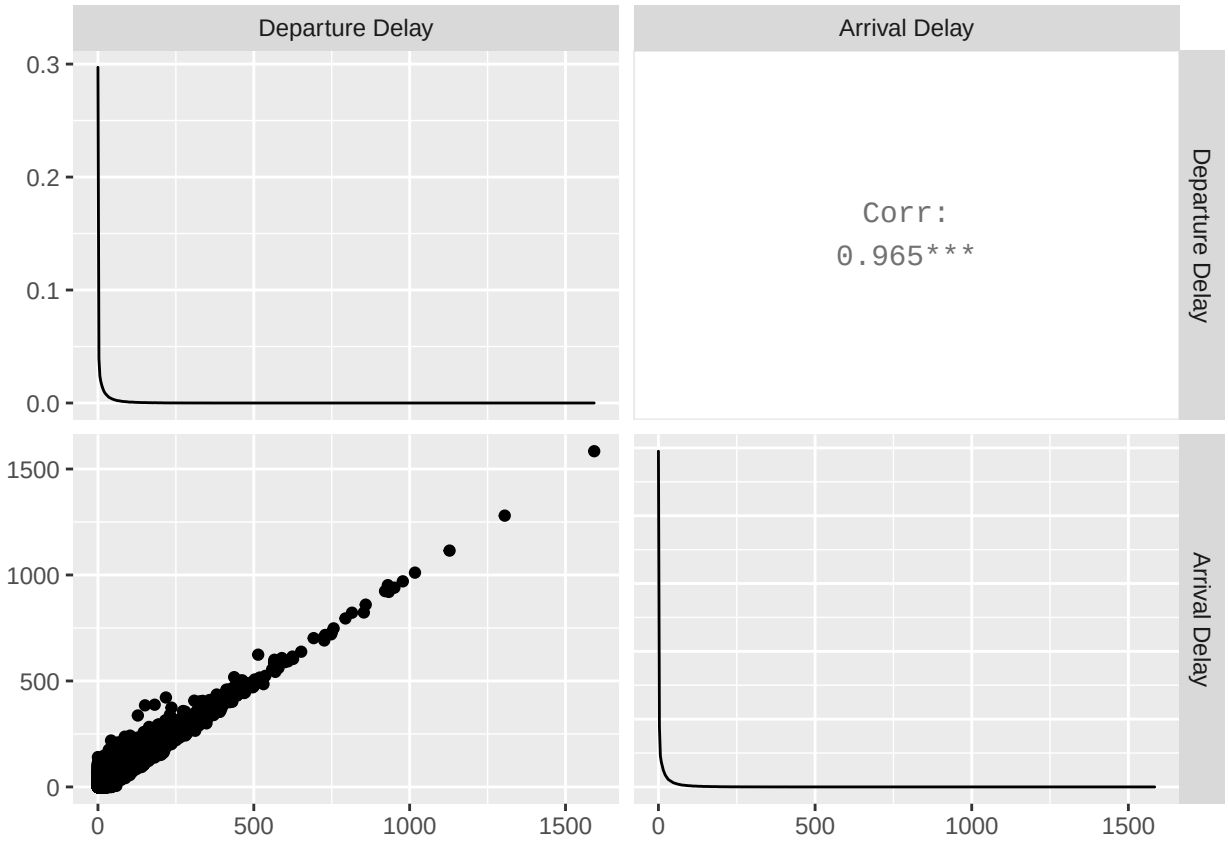

```
#Creating a data frame to calculate outliers for continious variables
airline.outliers <- airline.data[,c("Flight.Distance", "Departure.Delay.in.Minutes", "Arrival.Delay.in.M

#Looking for outliers in the continious variable columns
for (i in c("Flight.Distance", "Departure.Delay.in.Minutes", "Arrival.Delay.in.Minutes"))
  {
    mean_data <- mean(airline.outliers[,i], na.rm = T)
    sd_data <- sd(airline.outliers[,i], na.rm = T)
    zscore <- abs((airline.outliers[,i] - mean_data)/sd_data)
    airline.outliers[,i] <- zscore
  }

#Setting non outlier values to NA to retrive the outliers
airline.outliers[airline.outliers<3] <- NA
outliers <- airline.outliers[rowSums(is.na(airline.outliers)) != ncol(airline.outliers), ]

#Viewing the outliers
head(outliers)
```

```
##     Flight.Distance Departure.Delay.in.Minutes Arrival.Delay.in.Minutes
## 169              NA                   3.868714                 4.261175
## 179              NA                   3.317115                       NA
## 279              NA                   5.103245                 4.573142
## 283              NA                   3.527248                 3.195289
## 341              NA                   4.236446                 3.871217
## 405              NA                   4.840579                 4.417158
```

```
#Converting factor columns in airline data to numeric
for (i in 1:ncol(airline.data)-1)
  {
    if(is.factor(airline.data[,i]))
      {
        airline.data[,i] <- as.numeric(airline.data[,i])
      }
  }

#Plotting correlation plot for the airline data
correlation <- cor(na.omit(airline.data[,-23]))
corrplot(correlation, method = "color", type = "lower", col.text = "black", number.cex = .7, tl.col = "
```




```
#Subsetting the highly correlated variables into a new data frame
selected_features <- airline.data[,c("Departure.Delay.in.Minutes", "Arrival.Delay.in.Minutes")]

#Plotting correlation graph for highly correlated variables
ggpairs(selected_features, columnLabels = c("Departure Delay", "Arrival Delay"))
```



3. Data Cleaning and Shaping

```
#Importing Libraries to perform data cleaning
library(tidyverse)
library(arules)
library(stats)
library(factoextra)

#Creating a copy of airline data for data cleaning purpose
passenger.data <- airline.data

#Removing outliers from the passenger data
#passenger.data <- passenger.data[-as.numeric(row.names(outliers)), ]

#Replacing NA values in "Arrival Delay" column with mean value of the column
passenger.data$Arrival.Delay.in.Minutes <- passenger.data$Arrival.Delay.in.Minutes %>% replace_na(mean(

#Descretizing the "Age" column into 6 distinct bins
#passenger.data$Age <- discretize(passenger.data$Age, breaks = 6)

#Log transforming right skewed data
for (i in c("Flight.Distance", "Departure.Delay.in.Minutes", "Arrival.Delay.in.Minutes"))
  {
    passenger.data[,i] <- log(passenger.data[, i] + 1)
  }
```

```r
#Converting factor columns to numeric
for (i in 1:ncol(passenger.data)-1)
  {
    if(is.factor(passenger.data[,i]))
      {
        passenger.data[,i] <- as.numeric(passenger.data[,i])
      }
  }

#Performing PCA on the dataset
passenger.pca <- prcomp(passenger.data[,-23], center = T, scale = T)

#Printing PC's and summary of the PC's
print(passenger.pca)
```

```
## Standard deviations (1, .., p=22):
##  [1] 2.0133100 1.5432739 1.4857337 1.3479814 1.3162064 1.2168386 1.0085930
##  [8] 0.9843468 0.9721945 0.9144760 0.8318623 0.7444522 0.6933854 0.6796112
## [15] 0.6507950 0.6061685 0.5970549 0.5599340 0.5411605 0.4965564 0.4392490
## [22] 0.4213208
##
## Rotation (n x k) = (22 x 22):
##                                           PC1           PC2           PC3
## Gender                             -0.00509366  0.0004363322 -0.0466975425
## Customer.Type                      -0.08171187 -0.0632956812  0.1146809580
## Age                                -0.08106576 -0.0228322111  0.1204682714
## Type.of.Travel                      0.13544742 -0.0540024833 -0.0072314609
## Class                               0.21352576 -0.0682887601  0.0590173555
## Flight.Distance                    -0.13934453  0.0184692863  0.0006305524
## Inflight.wifi.service              -0.21375345 -0.4509356605  0.0135715691
## Departure.Arrival.time.convenient -0.07491922 -0.4370583914 -0.0344987408
## Ease.of.Online.booking             -0.15679456 -0.5306703928  0.0087442049
## Gate.location                      -0.05479961 -0.4308777972  0.0069449081
## Food.and.drink                     -0.28207931  0.1298631134  0.3428784826
## Online.boarding                    -0.29348195 -0.1586153483  0.1600952717
## Seat.comfort                       -0.33564863  0.1270390209  0.3230601075
## Inflight.entertainment             -0.40252108  0.1593822004  0.0778499889
## On.board.service                   -0.26631800  0.0788191862 -0.3667721336
## Leg.room.service                   -0.22096520  0.0270538171 -0.2912810808
## Baggage.handling                   -0.24335515  0.0705765613 -0.4269483316
## Checkin.service                    -0.17494119  0.0492279349 -0.1291731503
## Inflight.service                   -0.24566725  0.0767344592 -0.4360309959
## Cleanliness                        -0.33221687  0.1481649939  0.3177363716
## Departure.Delay.in.Minutes          0.03688835 -0.0077924588  0.0288003359
## Arrival.Delay.in.Minutes            0.04936194 -0.0103070546  0.0353509369
##                                           PC4          PC5           PC6
## Gender                              0.004677193 -0.03240874  0.0432118890
## Customer.Type                       0.071723512 -0.16983027  0.6366913199
## Age                                -0.025404328  0.07479249  0.4727002672
## Type.of.Travel                      0.268246220 -0.51799305  0.2505527373
## Class                               0.227512964 -0.43007598 -0.1359766087
## Flight.Distance                    -0.183425031  0.28589309  0.3104045143
```

```
## Inflight.wifi.service             -0.002062427  0.03071987 -0.1659540818
## Departure.Arrival.time.convenient  0.104372330 -0.25013780  0.1300206424
## Ease.of.Online.booking            -0.050395168  0.10381897 -0.1050163534
## Gate.location                      0.002280070 -0.04523330 -0.1283995959
## Food.and.drink                     0.072807399 -0.15821146 -0.1886347375
## Online.boarding                   -0.073741900  0.19089595  0.1367901545
## Seat.comfort                       0.024021594 -0.06706259  0.0034293428
## Inflight.entertainment             0.046637552 -0.16615648 -0.1338557996
## On.board.service                   0.014348029 -0.08642394  0.0698905662
## Leg.room.service                  -0.061695152  0.01969308  0.0318475051
## Baggage.handling                   0.013408426 -0.13249842 -0.0442439105
## Checkin.service                    0.023859507 -0.05248665  0.1431585446
## Inflight.service                   0.041775863 -0.13386496 -0.0432236091
## Cleanliness                        0.056860533 -0.15376930 -0.1312536011
## Departure.Delay.in.Minutes        -0.634012719 -0.30765030 -0.0004631339
## Arrival.Delay.in.Minutes          -0.631015273 -0.30854620 -0.0079035968
##                                          PC7          PC8          PC9
## Gender                            -0.850785641  0.360530718 -0.36024566
## Customer.Type                     -0.073657956 -0.043712363  0.15055144
## Age                                0.065613670  0.416464162  0.24822043
## Type.of.Travel                     0.057857504 -0.047849334 -0.07651679
## Class                              0.096259596  0.204799013  0.04347198
## Flight.Distance                   -0.194975507 -0.492184655  0.02074454
## Inflight.wifi.service              0.089377850  0.259423286 -0.01806788
## Departure.Arrival.time.convenient -0.070016063 -0.280345796 -0.05054387
## Ease.of.Online.booking             0.021698401  0.098779468 -0.01758938
## Gate.location                     -0.202464223 -0.338290699  0.15103093
## Food.and.drink                    -0.096269927 -0.079281089  0.02783636
## Online.boarding                    0.256870248  0.287756137 -0.19575932
## Seat.comfort                       0.020588115 -0.047184416 -0.03930447
## Inflight.entertainment            -0.084216223  0.006858595  0.21556882
## On.board.service                   0.064817391  0.044996652  0.09298723
## Leg.room.service                  -0.043177191  0.154479357  0.20184331
## Baggage.handling                  -0.001446129 -0.009211209  0.04424888
## Checkin.service                    0.262736561 -0.118875867 -0.77934032
## Inflight.service                  -0.007881858 -0.015592774  0.05708972
## Cleanliness                       -0.048532855 -0.071045797 -0.04667622
## Departure.Delay.in.Minutes         0.023790945  0.020633047 -0.01922263
## Arrival.Delay.in.Minutes           0.024553450  0.030175654 -0.01152086
##                                         PC10          PC11         PC12
## Gender                             0.014715568 -0.0535563081  0.004572225
## Customer.Type                      0.236979322  0.0151939557 -0.376101901
## Age                               -0.570702664 -0.0345448787  0.365224903
## Type.of.Travel                     0.206559858 -0.0004647577  0.025705098
## Class                              0.150135981  0.0728763259  0.543988370
## Flight.Distance                    0.338354528 -0.0403707973  0.603063474
## Inflight.wifi.service              0.232471402 -0.0376002446  0.124523442
## Departure.Arrival.time.convenient -0.158765045  0.0028283421 -0.084998867
## Ease.of.Online.booking             0.126899558 -0.0160249799  0.035981346
## Gate.location                     -0.432381870  0.0878213354 -0.033925043
## Food.and.drink                     0.008550948  0.0153147006  0.062376121
## Online.boarding                    0.245936951 -0.1741499245 -0.110864951
## Seat.comfort                      -0.085824105  0.0435398770 -0.068164495
## Inflight.entertainment             0.003022536 -0.0503059020  0.037632990
```

```
## On.board.service                   -0.059099811 -0.2463811569 -0.007506114
## Leg.room.service                    0.137291999  0.8596399412 -0.026012750
## Baggage.handling                   -0.004200358 -0.1994493199  0.022809276
## Checkin.service                    -0.254775930  0.2290791975  0.119382912
## Inflight.service                   -0.010280363 -0.2213075996  0.033232962
## Cleanliness                        -0.023955092  0.0542840618  0.023739461
## Departure.Delay.in.Minutes          0.002627644 -0.0175876902  0.004740658
## Arrival.Delay.in.Minutes           -0.005875832 -0.0145355527 -0.015831630
##                                            PC13         PC14          PC15
## Gender                             -0.032473141  0.027465991 -0.0703977727
## Customer.Type                       0.096641568  0.257990464  0.2764518558
## Age                                 0.068134480 -0.150579569  0.0741814927
## Type.of.Travel                      0.049565498 -0.008802976 -0.0536949902
## Class                               0.003552333  0.237935799 -0.1484415787
## Flight.Distance                    -0.011320182  0.004726722 -0.0546213419
## Inflight.wifi.service              -0.014344642  0.029451895  0.2153284402
## Departure.Arrival.time.convenient -0.219361452 -0.611425194 -0.2487477883
## Ease.of.Online.booking             -0.014493756 -0.041660961  0.1617410310
## Gate.location                       0.171181226  0.550883744 -0.0472677900
## Food.and.drink                     -0.073962505 -0.200249434  0.5367601514
## Online.boarding                     0.111395662  0.094591886 -0.3323704637
## Seat.comfort                        0.070599256  0.089063010 -0.5002765932
## Inflight.entertainment             -0.031071516  0.058500177  0.1043098327
## On.board.service                   -0.748444444  0.262340198 -0.0613553509
## Leg.room.service                   -0.033370733 -0.061544285 -0.0935272438
## Baggage.handling                    0.497786493 -0.142091332 -0.0333773909
## Checkin.service                     0.005002805  0.137210244  0.2374026829
## Inflight.service                    0.267284138 -0.029766881  0.0401021130
## Cleanliness                         0.011783464 -0.021409497 -0.1309499476
## Departure.Delay.in.Minutes         -0.002518302  0.001498464 -0.0009812466
## Arrival.Delay.in.Minutes           -0.003679742  0.002651690  0.0098053918
##                                            PC16         PC17          PC18
## Gender                             -0.003767793  0.008231234  0.028980958
## Customer.Type                       0.004916295 -0.370614897 -0.045579689
## Age                                 0.003654256  0.112954203 -0.043026479
## Type.of.Travel                     -0.042837172  0.698486860 -0.091570390
## Class                               0.018788867 -0.427407087  0.198593299
## Flight.Distance                    -0.008204425  0.055385556 -0.003287224
## Inflight.wifi.service              -0.042930002  0.021339718 -0.376169910
## Departure.Arrival.time.convenient   0.049527783 -0.268987358  0.107480174
## Ease.of.Online.booking              0.012730352  0.084431861 -0.127372181
## Gate.location                      -0.031072356  0.138443241  0.196394271
## Food.and.drink                     -0.056362847  0.140986374  0.524346616
## Online.boarding                     0.037429027  0.076994609  0.522132301
## Seat.comfort                        0.019410779  0.022631016 -0.152824534
## Inflight.entertainment              0.053242696 -0.128330854 -0.179390230
## On.board.service                   -0.168310276  0.063166032  0.060320388
## Leg.room.service                    0.006485698  0.080218663  0.115863012
## Baggage.handling                   -0.634066005 -0.091681368  0.043124214
## Checkin.service                     0.003020801 -0.113860811 -0.064620789
## Inflight.service                    0.744073021  0.027611491  0.046487890
## Cleanliness                        -0.011679387 -0.016810105 -0.333420603
## Departure.Delay.in.Minutes          0.014736855  0.033606764  0.004025054
## Arrival.Delay.in.Minutes            0.007914278 -0.018156075 -0.001801345
```

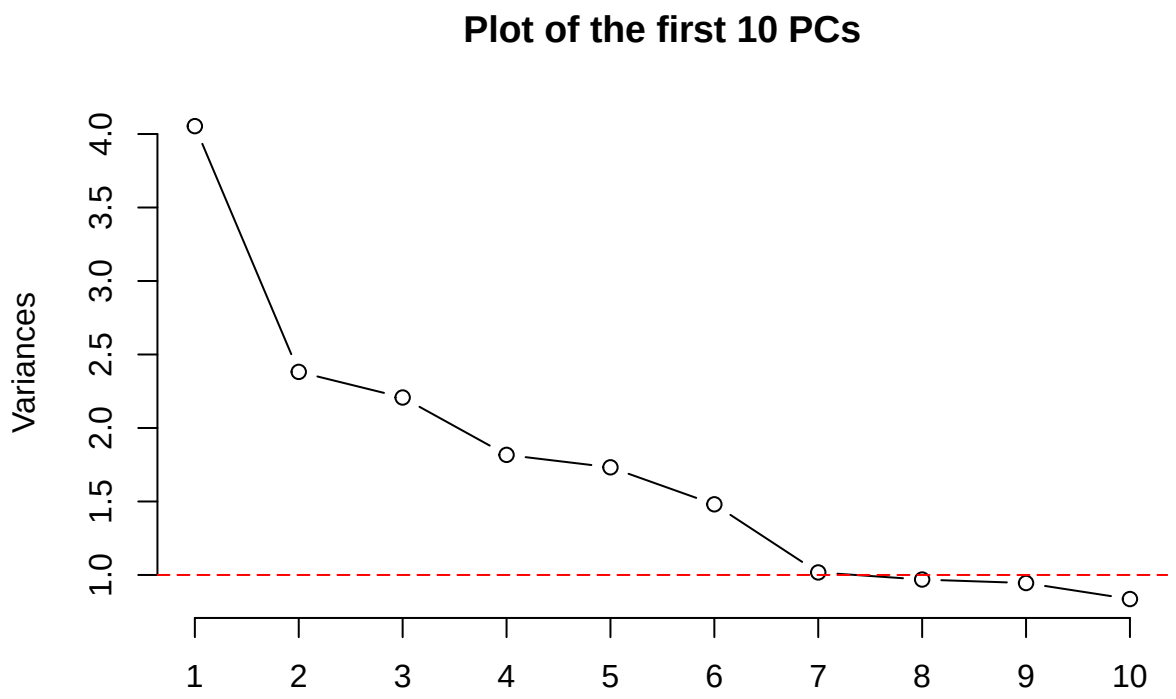
```
##                                          PC19         PC20          PC21
## Gender                                0.009949743 -0.006569183  0.0030397167
## Customer.Type                         0.032136179  0.037093110 -0.0269382178
## Age                                  -0.031042392  0.008997613  0.0021586985
## Type.of.Travel                       -0.027368500 -0.003678981  0.0362757545
## Class                                 0.023343020  0.098270881 -0.0161978293
## Flight.Distance                       0.001844719 -0.035291253  0.0212462579
## Inflight.wifi.service                 0.136606919 -0.583521378 -0.0020518622
## Departure.Arrival.time.convenient    -0.043271818 -0.129763012  0.0007505723
## Ease.of.Online.booking                0.090575677  0.760992592 -0.0085133554
## Gate.location                        -0.101742468 -0.120837234  0.0053904324
## Food.and.drink                        0.180447567 -0.025049876 -0.0092017924
## Online.boarding                      -0.267583210 -0.093937269  0.0219213009
## Seat.comfort                          0.662431531  0.062362944 -0.0058083931
## Inflight.entertainment               -0.089140886 -0.035684412  0.0552374699
## On.board.service                      0.015846572  0.052927969 -0.0075779599
## Leg.room.service                     -0.027398338 -0.011958633  0.0022746936
## Baggage.handling                      0.028062750  0.052802764 -0.0079428832
## Checkin.service                       0.026188012  0.009288480  0.0087866158
## Inflight.service                      0.036611692  0.016493589 -0.0050658068
## Cleanliness                          -0.634630494  0.124315165 -0.0306885671
## Departure.Delay.in.Minutes            0.002552365 -0.017411963 -0.7037014753
## Arrival.Delay.in.Minutes              0.004035359  0.008621684  0.7050589898
##                                            PC22
## Gender                                1.773404e-02
## Customer.Type                        -1.185002e-01
## Age                                  -2.392927e-02
## Type.of.Travel                        1.233539e-01
## Class                                -5.074643e-02
## Flight.Distance                       6.358569e-03
## Inflight.wifi.service                -1.590941e-01
## Departure.Arrival.time.convenient     6.642952e-02
## Ease.of.Online.booking                4.671630e-02
## Gate.location                        -3.128488e-05
## Food.and.drink                       -1.860406e-01
## Online.boarding                       1.399749e-01
## Seat.comfort                         -1.160528e-01
## Inflight.entertainment                7.888120e-01
## On.board.service                     -1.575839e-01
## Leg.room.service                     -3.106677e-02
## Baggage.handling                     -8.010846e-02
## Checkin.service                       1.052810e-01
## Inflight.service                     -1.902894e-01
## Cleanliness                          -4.005820e-01
## Departure.Delay.in.Minutes            5.050027e-02
## Arrival.Delay.in.Minutes             -5.394737e-02
```

```
summary(passenger.pca)
```

```
## Importance of components:
##                            PC1    PC2    PC3     PC4     PC5    PC6     PC7
## Standard deviation     2.0133 1.5433 1.4857 1.34798 1.31621 1.2168 1.00859
## Proportion of Variance 0.1842 0.1083 0.1003 0.08259 0.07875 0.0673 0.04624
## Cumulative Proportion  0.1842 0.2925 0.3928 0.47544 0.55418 0.6215 0.66772
```

```
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.98435 0.97219 0.91448 0.83186 0.74445 0.69339 0.67961
## Proportion of Variance 0.04404 0.04296 0.03801 0.03145 0.02519 0.02185 0.02099
## Cumulative Proportion  0.71177 0.75473 0.79274 0.82419 0.84939 0.87124 0.89223
##                           PC15   PC16   PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.65079 0.6062 0.5971 0.55993 0.54116 0.49656 0.43925
## Proportion of Variance 0.01925 0.0167 0.0162 0.01425 0.01331 0.01121 0.00877
## Cumulative Proportion  0.91149 0.9282 0.9444 0.95864 0.97195 0.98316 0.99193
##                           PC22
## Standard deviation     0.42132
## Proportion of Variance 0.00807
## Cumulative Proportion  1.00000
```

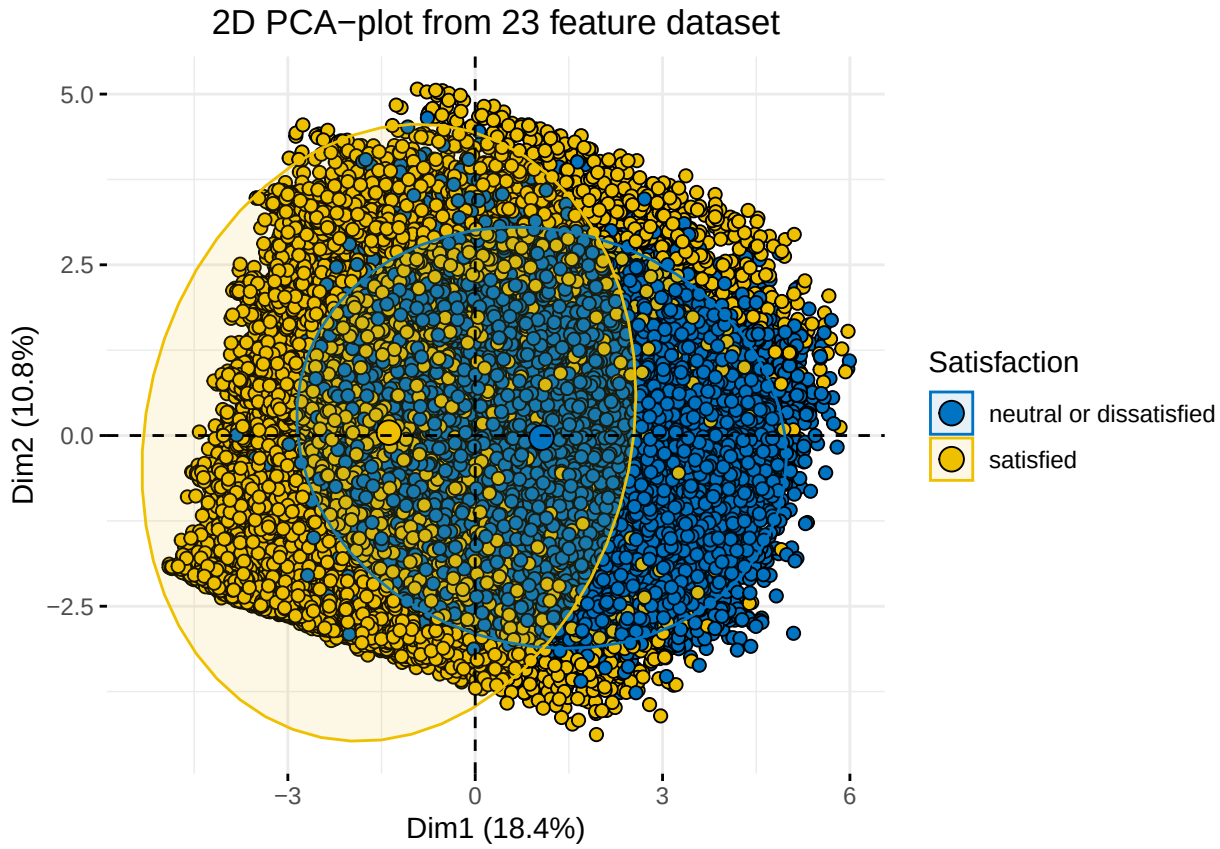
```
#Plotting variance plot for the first 10 PC's
screeplot(passenger.pca, type = "l", main = "Plot of the first 10 PCs") +
abline(h = 1, col="red", lty=5)
```




```
## integer(0)
```

```
#Plotting scatter plot for the satisfaction levels from the calculated values of PC's
fviz_pca_ind(passenger.pca, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = passenger.data$satisfaction,
             col.ind = "black",
             palette = "jco",
```

```
            addEllipses = TRUE,
            label = "var",
            col.var = "black",
            repel = TRUE,
            legend.title = "Satisfaction") +
  ggtitle("2D PCA-plot from 23 feature dataset") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
#From the above summary and graphs of the PCs, we see a low level of variance in the features, and redu
#dimension will not significantly increase the efficiency of the models.
#Hence I will not go ahead and implement PCA for this dataset.

#Creating a function for normalization
normalize <- function(x)
  {
    return ((x-min(x))/(max(x)-min(x)))
  }

#Normalized passenger data
passenger.data <- data.frame(lapply(passenger.data[,-23], normalize), passenger.data$satisfaction)
colnames(passenger.data)[23] <- "satisfaction"

str(passenger.data)
```

```
## 'data.frame':    129880 obs. of  23 variables:
```

```
##  $ Gender                           : num  1 1 0 0 1 0 1 0 0 1 ...
##  $ Customer.Type                    : num  1 0 1 1 1 1 1 1 1 0 ...
##  $ Age                              : num  0.0769 0.2308 0.2436 0.2308 0.6923 ...
##  $ Type.of.Travel                   : num  1 0 0 0 0 1 1 0 0 0 ...
##  $ Class                            : num  1 0 0 0 0 0.5 0.5 0 0 0.5 ...
##  $ Flight.Distance                  : num  0.528 0.396 0.708 0.568 0.377 ...
##  $ Inflight.wifi.service            : num  0.6 0.6 0.4 0.4 0.6 0.6 0.4 0.8 0.2 0.6 ...
##  $ Departure.Arrival.time.convenient: num  0.8 0.4 0.4 1 0.6 0.8 0.8 0.6 0.4 0.6 ...
##  $ Ease.of.Online.booking           : num  0.6 0.6 0.4 1 0.6 0.4 0.4 0.8 0.4 0.6 ...
##  $ Gate.location                    : num  0.2 0.6 0.4 1 0.6 0.2 0.6 0.8 0.4 0.8 ...
##  $ Food.and.drink                   : num  1 0.2 1 0.4 0.8 0.2 0.4 1 0.8 0.4 ...
##  $ Online.boarding                  : num  0.6 0.6 1 0.4 1 0.4 0.4 1 0.6 0.6 ...
##  $ Seat.comfort                     : num  1 0.2 1 0.4 1 0.2 0.4 1 0.6 0.6 ...
##  $ Inflight.entertainment           : num  1 0.2 1 0.4 0.6 0.2 0.4 1 0.2 0.4 ...
##  $ On.board.service                 : num  0.8 0.2 0.8 0.4 0.6 0.6 0.6 1 0.2 0.4 ...
##  $ Leg.room.service                 : num  0.6 1 0.6 1 0.8 0.8 0.6 1 0.4 0.6 ...
##  $ Baggage.handling                 : num  0.75 0.5 0.75 0.5 0.75 0.75 0.75 1 0 0.75 ...
##  $ Checkin.service                  : num  0.8 0.2 0.8 0.2 0.6 0.8 0.6 0.8 0.8 0.8 ...
##  $ Inflight.service                 : num  1 0.8 0.8 0.8 0.6 0.8 1 1 0.2 0.6 ...
##  $ Cleanliness                      : num  1 0.2 1 0.4 0.6 0.2 0.4 0.8 0.4 0.4 ...
##  $ Departure.Delay.in.Minutes       : num  0.442 0.094 0 0.337 0 ...
##  $ Arrival.Delay.in.Minutes         : num  0.4 0.264 0 0.312 0 ...
##  $ satisfaction                     : Factor w/ 2 levels "neutral or dissatisfied",..: 1 1 2 1 2 1 1
```

4. Model Construction and Evaluation

```r
#Importing libraries
library(caret)
library(e1071)
library(neuralnet)
library(class)
library(rpart)
library(C50)

#Creating a random sample using createDataPartition function
set.seed(1)
sample <- createDataPartition(passenger.data$satisfaction, p = 0.75, list = FALSE)

#Creating a training and testing dataset based on the sampled value
train.data <- passenger.data[sample,]
test.data <- passenger.data[-sample,]

##############################
#### Logistic Regression ####
##############################


##### Logistic Regression Model Implementation #####

#Creating Logistic Regression model
LR.model <- glm(satisfaction~., data = train.data, family = "binomial")

#Summary of Logistic regression model
summary(LR.model)
```

```
##
## Call:
## glm(formula = satisfaction ~ ., family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9362  -0.4847  -0.1664   0.3848   4.0852
##
## Coefficients:
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       -7.67374    0.08523 -90.039  < 2e-16 ***
## Gender                             0.07120    0.02021   3.524 0.000425 ***
## Customer.Type                      2.14168    0.03037  70.524  < 2e-16 ***
## Age                               -0.65801    0.05781 -11.383  < 2e-16 ***
## Type.of.Travel                    -2.92873    0.03134 -93.451  < 2e-16 ***
## Class                             -1.04595    0.03914 -26.724  < 2e-16 ***
## Flight.Distance                   -0.17691    0.06154  -2.875 0.004043 **
## Inflight.wifi.service              1.93021    0.05940  32.498  < 2e-16 ***
## Departure.Arrival.time.convenient -0.63973    0.04239 -15.093  < 2e-16 ***
## Ease.of.Online.booking            -0.68205    0.05875 -11.610  < 2e-16 ***
## Gate.location                      0.05408    0.04747   1.139 0.254589
## Food.and.drink                    -0.08886    0.05543  -1.603 0.108868
## Online.boarding                    3.08230    0.05293  58.232  < 2e-16 ***
## Seat.comfort                       0.33440    0.05820   5.746 9.13e-09 ***
## Inflight.entertainment             0.26651    0.07403   3.600 0.000318 ***
## On.board.service                   1.56667    0.05302  29.549  < 2e-16 ***
## Leg.room.service                   1.22265    0.04430  27.600  < 2e-16 ***
## Baggage.handling                   0.53375    0.04742  11.257  < 2e-16 ***
## Checkin.service                    1.65386    0.04427  37.356  < 2e-16 ***
## Inflight.service                   0.66712    0.06234  10.701  < 2e-16 ***
## Cleanliness                        1.11987    0.06281  17.829  < 2e-16 ***
## Departure.Delay.in.Minutes         0.17654    0.07812   2.260 0.023829 *
## Arrival.Delay.in.Minutes          -1.59325    0.07786 -20.462  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 133361  on 97409  degrees of freedom
## Residual deviance:  64407  on 97387  degrees of freedom
## AIC: 64453
##
## Number of Fisher Scoring iterations: 6
```

```r
#Predicting the values for test data
LR.predict_prob <- predict(LR.model, test.data, type = "response")
LR.predict <- as.factor(ifelse(LR.predict_prob < 0.5, "neutral or dissatisfied", "satisfied"))

#Producing Confusion Matrix for the classification
LR.CM <- confusionMatrix(LR.predict, test.data$satisfaction)
LR.CM
```

```
## Confusion Matrix and Statistics
##
```

```
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   16649      2396
##   satisfied                                  1714     11711
##
##                Accuracy : 0.8734
##                  95% CI : (0.8698, 0.877)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.741
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9067
##             Specificity : 0.8302
##          Pos Pred Value : 0.8742
##          Neg Pred Value : 0.8723
##              Prevalence : 0.5655
##          Detection Rate : 0.5128
##    Detection Prevalence : 0.5865
##       Balanced Accuracy : 0.8684
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
##### Logistic Regression Model Improvement #####

#Stepwise backward elimination using AIC
step(LR.model, direction =  "backward")
```

```
## Start:  AIC=64452.69
## satisfaction ~ Gender + Customer.Type + Age + Type.of.Travel +
##     Class + Flight.Distance + Inflight.wifi.service + Departure.Arrival.time.convenient +
##     Ease.of.Online.booking + Gate.location + Food.and.drink +
##     Online.boarding + Seat.comfort + Inflight.entertainment +
##     On.board.service + Leg.room.service + Baggage.handling +
##     Checkin.service + Inflight.service + Cleanliness + Departure.Delay.in.Minutes +
##     Arrival.Delay.in.Minutes
##
##                                      Df Deviance   AIC
## - Gate.location                       1    64408 64452
## <none>                                     64407 64453
## - Food.and.drink                      1    64409 64453
## - Departure.Delay.in.Minutes          1    64412 64456
## - Flight.Distance                     1    64415 64459
## - Gender                              1    64419 64463
## - Inflight.entertainment              1    64420 64464
## - Seat.comfort                        1    64440 64484
## - Inflight.service                    1    64522 64566
## - Baggage.handling                    1    64534 64578
## - Age                                 1    64537 64581
## - Ease.of.Online.booking              1    64542 64586
## - Departure.Arrival.time.convenient   1    64634 64678
```

```
## - Cleanliness                         1    64726 64770
## - Arrival.Delay.in.Minutes            1    64832 64876
## - Class                               1    65133 65177
## - Leg.room.service                    1    65173 65217
## - On.board.service                    1    65298 65342
## - Inflight.wifi.service               1    65499 65543
## - Checkin.service                     1    65851 65895
## - Online.boarding                     1    68016 68060
## - Customer.Type                       1    69968 70012
## - Type.of.Travel                      1    75088 75132
##
## Step:  AIC=64451.99
## satisfaction ~ Gender + Customer.Type + Age + Type.of.Travel +
##     Class + Flight.Distance + Inflight.wifi.service + Departure.Arrival.time.convenient +
##     Ease.of.Online.booking + Food.and.drink + Online.boarding +
##     Seat.comfort + Inflight.entertainment + On.board.service +
##     Leg.room.service + Baggage.handling + Checkin.service + Inflight.service +
##     Cleanliness + Departure.Delay.in.Minutes + Arrival.Delay.in.Minutes
##
##                                    Df Deviance   AIC
## <none>                                  64408 64452
## - Food.and.drink                    1    64411 64453
## - Departure.Delay.in.Minutes        1    64413 64455
## - Flight.Distance                   1    64416 64458
## - Gender                            1    64420 64462
## - Inflight.entertainment            1    64421 64463
## - Seat.comfort                      1    64442 64484
## - Inflight.service                  1    64523 64565
## - Baggage.handling                  1    64535 64577
## - Age                               1    64538 64580
## - Ease.of.Online.booking            1    64549 64591
## - Departure.Arrival.time.convenient 1    64648 64690
## - Cleanliness                       1    64728 64770
## - Arrival.Delay.in.Minutes          1    64833 64875
## - Class                             1    65134 65176
## - Leg.room.service                  1    65173 65215
## - On.board.service                  1    65298 65340
## - Inflight.wifi.service             1    65499 65541
## - Checkin.service                   1    65851 65893
## - Online.boarding                   1    68138 68180
## - Customer.Type                     1    69968 70010
## - Type.of.Travel                    1    75179 75221


##
## Call:  glm(formula = satisfaction ~ Gender + Customer.Type + Age + Type.of.Travel +
##     Class + Flight.Distance + Inflight.wifi.service + Departure.Arrival.time.convenient +
##     Ease.of.Online.booking + Food.and.drink + Online.boarding +
##     Seat.comfort + Inflight.entertainment + On.board.service +
##     Leg.room.service + Baggage.handling + Checkin.service + Inflight.service +
##     Cleanliness + Departure.Delay.in.Minutes + Arrival.Delay.in.Minutes,
##     family = "binomial", data = train.data)
##
## Coefficients:
##                       (Intercept)                            Gender
```

```
##                          -7.65195                            0.07097
##                     Customer.Type                                Age
##                           2.14136                           -0.65697
##                    Type.of.Travel                              Class
##                          -2.93209                           -1.04596
##                   Flight.Distance             Inflight.wifi.service
##                          -0.17873                            1.92882
## Departure.Arrival.time.convenient            Ease.of.Online.booking
##                          -0.62467                           -0.66170
##                    Food.and.drink                   Online.boarding
##                          -0.09038                            3.07087
##                     Seat.comfort            Inflight.entertainment
##                           0.33796                            0.26845
##                  On.board.service                  Leg.room.service
##                           1.56537                            1.22029
##                  Baggage.handling                   Checkin.service
##                           0.53294                            1.65324
##                  Inflight.service                       Cleanliness
##                           0.66538                            1.12107
##        Departure.Delay.in.Minutes          Arrival.Delay.in.Minutes
##                           0.17727                           -1.59322
##
## Degrees of Freedom: 97409 Total (i.e. Null);  97388 Residual
## Null Deviance:       133400
## Residual Deviance: 64410     AIC: 64450
```

```
#Creating a new model after backwards elimination
LR.model_new <- glm(formula = satisfaction ~ Gender + Customer.Type + Age + Type.of.Travel + Class +
                      Flight.Distance + Inflight.wifi.service + Departure.Arrival.time.convenient +
                      Ease.of.Online.booking + Food.and.drink + Online.boarding + Seat.comfort +
                      Inflight.entertainment + On.board.service + Leg.room.service + Baggage.handling +
                      Checkin.service + Inflight.service + Cleanliness + Departure.Delay.in.Minutes +
                      Arrival.Delay.in.Minutes, family = "binomial", data = train.data)

#Summary of the new logistic regression model
summary(LR.model_new)
```

```
##
## Call:
## glm(formula = satisfaction ~ Gender + Customer.Type + Age + Type.of.Travel +
##     Class + Flight.Distance + Inflight.wifi.service + Departure.Arrival.time.convenient +
##     Ease.of.Online.booking + Food.and.drink + Online.boarding +
##     Seat.comfort + Inflight.entertainment + On.board.service +
##     Leg.room.service + Baggage.handling + Checkin.service + Inflight.service +
##     Cleanliness + Departure.Delay.in.Minutes + Arrival.Delay.in.Minutes,
##     family = "binomial", data = train.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9375  -0.4847  -0.1661   0.3852   4.0853
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                      -7.65195    0.08301 -92.176  < 2e-16 ***
```

```
## Gender                                0.07097    0.02020   3.513 0.000444 ***
## Customer.Type                         2.14136    0.03037  70.507  < 2e-16 ***
## Age                                  -0.65697    0.05780 -11.366  < 2e-16 ***
## Type.of.Travel                       -2.93209    0.03121 -93.936  < 2e-16 ***
## Class                                -1.04596    0.03914 -26.726  < 2e-16 ***
## Flight.Distance                      -0.17873    0.06151  -2.906 0.003666 **
## Inflight.wifi.service                 1.92882    0.05940  32.472  < 2e-16 ***
## Departure.Arrival.time.convenient    -0.62467    0.04029 -15.505  < 2e-16 ***
## Ease.of.Online.booking               -0.66170    0.05599 -11.818  < 2e-16 ***
## Food.and.drink                       -0.09038    0.05541  -1.631 0.102821
## Online.boarding                       3.07087    0.05197  59.093  < 2e-16 ***
## Seat.comfort                          0.33796    0.05811   5.816 6.03e-09 ***
## Inflight.entertainment                0.26845    0.07400   3.628 0.000286 ***
## On.board.service                      1.56537    0.05301  29.529  < 2e-16 ***
## Leg.room.service                      1.22029    0.04426  27.574  < 2e-16 ***
## Baggage.handling                      0.53294    0.04740  11.242  < 2e-16 ***
## Checkin.service                       1.65324    0.04427  37.344  < 2e-16 ***
## Inflight.service                      0.66538    0.06232  10.678  < 2e-16 ***
## Cleanliness                           1.12107    0.06280  17.852  < 2e-16 ***
## Departure.Delay.in.Minutes            0.17727    0.07812   2.269 0.023254 *
## Arrival.Delay.in.Minutes             -1.59322    0.07787 -20.461  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 133361  on 97409  degrees of freedom
## Residual deviance:  64408  on 97388  degrees of freedom
## AIC: 64452
##
## Number of Fisher Scoring iterations: 6
```

```
#Predicting the values after backward elimination
LR.predict_prob_new <- predict(LR.model_new, test.data, type = "response")
LR.predict_new <- as.factor(ifelse(LR.predict_prob_new < 0.5, "neutral or dissatisfied", "satisfied"))

#Producing Confusion Matrix for the classification of new model
LR.CM_new <- confusionMatrix(LR.predict_new, test.data$satisfaction)
LR.CM_new
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   16647      2396
##   satisfied                                  1716     11711
##
##                Accuracy : 0.8734
##                  95% CI : (0.8697, 0.877)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7408
##
```

```
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9066
##             Specificity : 0.8302
##          Pos Pred Value : 0.8742
##          Neg Pred Value : 0.8722
##              Prevalence : 0.5655
##          Detection Rate : 0.5127
##    Detection Prevalence : 0.5865
##       Balanced Accuracy : 0.8684
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
#We see that there is no improvement in the accuracy of the model after backward elimination

##############################
#### k-Nearest Neighbors  ####
##############################


##### k-Nearest Neighbors model Implementation #####

#Creating train labels and test labels for kNN implementation
train.label <- train.data[,23]
test.label <- test.data[,23]

Sys.time()
```

```
## [1] "2020-08-05 00:57:19 EDT"
```

```
#Creating kNN model
kNN.predict <- knn(train = train.data[,-23], test = test.data[,-23], cl = train.label, k = 10)

#Producing Confusion matrix for the classification
kNN.CM <- confusionMatrix(kNN.predict, test.label)
kNN.CM
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17761      1773
##   satisfied                                   602     12334
##
##                Accuracy : 0.9269
##                  95% CI : (0.924, 0.9297)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8497
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
##             Sensitivity : 0.9672
##             Specificity : 0.8743
##          Pos Pred Value : 0.9092
##          Neg Pred Value : 0.9535
##              Prevalence : 0.5655
##          Detection Rate : 0.5470
##    Detection Prevalence : 0.6016
##       Balanced Accuracy : 0.9208
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
Sys.time()
```

```
## [1] "2020-08-05 00:58:42 EDT"
```

```
##### k-Nearest Neighbors model Tuning #####

Sys.time()
```

```
## [1] "2020-08-05 00:58:42 EDT"
```

```
#Creating kNN model
kNN.predict_new <- knn(train = train.data[,-23], test = test.data[,-23], cl = train.label, k = 5)

#Producing Confusion Matrix for the classification
kNN.CM_new <- confusionMatrix(kNN.predict_new, test.label)
kNN.CM_new
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17699      1697
##   satisfied                                   664     12410
##
##                Accuracy : 0.9273
##                  95% CI : (0.9244, 0.9301)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8508
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9638
##             Specificity : 0.8797
##          Pos Pred Value : 0.9125
##          Neg Pred Value : 0.9492
##              Prevalence : 0.5655
##          Detection Rate : 0.5451
```

```
##    Detection Prevalence : 0.5974
##       Balanced Accuracy : 0.9218
##
##        'Positive' Class : neutral or dissatisfied
##
```

```r
Sys.time()
```

```
## [1] "2020-08-05 01:00:00 EDT"
```

```r
#Decreasing the number of neighbors from 10 to 5 increases the accuracy of the model from 92.69% to 92.
#The gain in the accuracy in minuscule but the time required calculate for k = 5 is also less than k =
#I have also tried to implement the model with k = 311 which is the "sqrt(nrow(train.data))" on RStudio
#but the accuracy came out to be much more lesser that k = 5 or k = 10.

##############################
####### Decision Tree #######
##############################


##### Decision Tree Model Implementation #####

#Creating a Decision Tree model using rpart
rpart.model <- rpart(satisfaction ~ ., data = train.data, method = "class")

#Predicting the probabilities for the target variable from the test data
rpart.predict_prob <- predict(rpart.model, test.data)

#Replacing the probability values to the values of the target variable
rpart.predict <- as.factor(ifelse(rpart.predict_prob[,2] < 0.5, "neutral or dissatisfied", "satisfied"))

#Producing Confusion Matrix for the classification using rpart
rpart.CM <- confusionMatrix(rpart.predict, test.data$satisfaction)
rpart.CM
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   15867      1254
##   satisfied                                  2496     12853
##
##                Accuracy : 0.8845
##                  95% CI : (0.881, 0.888)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7674
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8641
##             Specificity : 0.9111
```

```
##          Pos Pred Value : 0.9268
##          Neg Pred Value : 0.8374
##              Prevalence : 0.5655
##          Detection Rate : 0.4887
##    Detection Prevalence : 0.5273
##       Balanced Accuracy : 0.8876
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
##### Decision Tree Model Improvement #####

#Creating a Decesion Tree model using C5.0
c50.model <- C5.0(x = train.data[,-23], y = train.data$satisfaction)

#Predicting the target variable of the test data based on the above model
c50.predict <- predict(c50.model, test.data[,-23])

#Producing Confusion Matrix for the classification using rpart
c50.CM <- confusionMatrix(c50.predict, test.data$satisfaction)
c50.CM
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17895       900
##   satisfied                                   468     13207
##
##                Accuracy : 0.9579
##                  95% CI : (0.9556, 0.96)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.914
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9745
##             Specificity : 0.9362
##          Pos Pred Value : 0.9521
##          Neg Pred Value : 0.9658
##              Prevalence : 0.5655
##          Detection Rate : 0.5511
##    Detection Prevalence : 0.5788
##       Balanced Accuracy : 0.9554
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
#We see that changing the Decision Tree model from rpart to C5.0 increases the accuracy from 88.45% to

################################
```

```r
### Support Vector Machine ###
##############################


##### SVM Model Implementation #####

Sys.time()
```

```
## [1] "2020-08-05 01:00:13 EDT"
```

```r
#Creating SVM model
SVM.model <- svm(satisfaction ~ ., data = train.data, probability = TRUE)

Sys.time()
```

```
## [1] "2020-08-05 01:45:17 EDT"
```

```r
#Predicting the values for the test data
SVM.predict <- predict(SVM.model, test.data, probability = TRUE)

#Printing the confusionMatrix for the SVM model
SVM.CM <- confusionMatrix(SVM.predict, test.data$satisfaction)
SVM.CM
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17738       874
##   satisfied                                   625     13233
##
##                Accuracy : 0.9538
##                  95% CI : (0.9515, 0.9561)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9059
##
##  Mcnemar's Test P-Value : 1.499e-10
##
##             Sensitivity : 0.9660
##             Specificity : 0.9380
##          Pos Pred Value : 0.9530
##          Neg Pred Value : 0.9549
##              Prevalence : 0.5655
##          Detection Rate : 0.5463
##    Detection Prevalence : 0.5732
##       Balanced Accuracy : 0.9520
##
##        'Positive' Class : neutral or dissatisfied
##
```

```r
Sys.time()
```

```
## [1] "2020-08-05 01:46:08 EDT"
```

```r
##### SVM Model Improvement #####

Sys.time()
```

```
## [1] "2020-08-05 01:46:08 EDT"
```

```r
#Creating SVM model
SVM.model_new <- svm(satisfaction ~ ., data = train.data, kernel = "linear", probability = TRUE)

Sys.time()
```

```
## [1] "2020-08-05 05:53:59 EDT"
```

```r
#Predicting the values for the test data
SVM.predict_new <- predict(SVM.model_new, test.data, probability = TRUE)

#Printing the confusionMatrix for the SVM model
SVM.CM_new <- confusionMatrix(SVM.predict_new, test.data$satisfaction)
SVM.CM_new
```

```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   16627      2343
##   satisfied                                  1736     11764
##
##                Accuracy : 0.8744
##                  95% CI : (0.8707, 0.878)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7431
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9055
##             Specificity : 0.8339
##          Pos Pred Value : 0.8765
##          Neg Pred Value : 0.8714
##              Prevalence : 0.5655
##          Detection Rate : 0.5121
##    Detection Prevalence : 0.5842
##       Balanced Accuracy : 0.8697
##
##        'Positive' Class : neutral or dissatisfied
##
```

```r
Sys.time()
```

```
## [1] "2020-08-05 05:55:13 EDT"
```

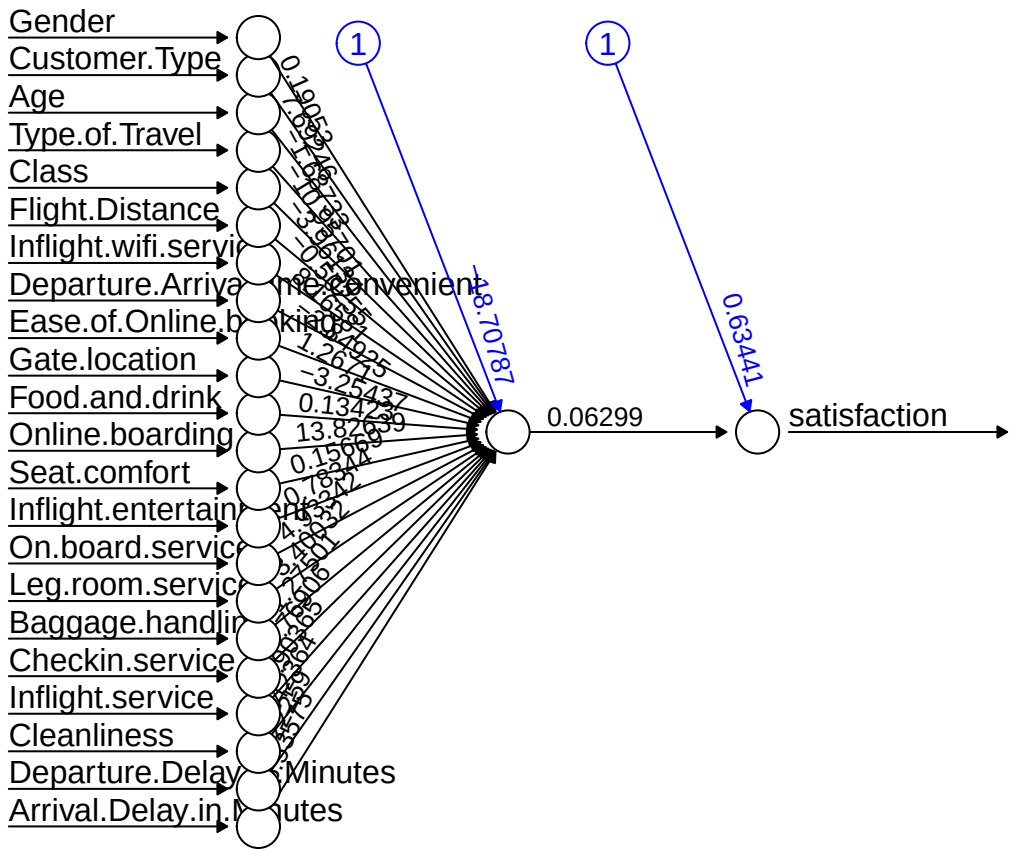
```r
#To increase the accuracy of the SVM model, I changed the kernel from "Radial"(default) to "Linear"
#The new kernal instead of increase the accuracy of the model decreased the accuracy
#Hence I will be considering the original SVM model for ensembler model construction

##############################
###### Neural Network  ######
##############################


##### Neural Network Model Implementation #####

#Creating training and testing datasets for Neural Netwrok(NN)
NN.train <- train.data
NN.test <- test.data

#Setting y as an integer
NN.train$satisfaction <- as.integer(NN.train$satisfaction)
NN.test$satisfaction <- as.integer(NN.test$satisfaction)

#Creating function softplus for smoothing
softplus <- function(x) log(1+exp(x))

Sys.time()
```

```
## [1] "2020-08-05 05:55:13 EDT"
```

```r
#Creating Neural Network model
NN.model <- neuralnet(satisfaction ~ ., NN.train, hidden = 1, threshold = 1, rep = 1, linear.output = F

Sys.time()
```

```
## [1] "2020-08-05 06:06:05 EDT"
```

```r
#plotting the neural network graph
plot(NN.model, rep="best")
```




```
Sys.time()
```

```
## [1] "2020-08-05 06:06:05 EDT"
```

```
#Computing the values for the test data
NN.predict <- compute(NN.model, NN.test[,-23])

#Calculating correlation between the actual and predicted values
NN.cor <- cor(NN.predict$net.result, NN.test$satisfaction)
sprintf("The correlation between actual values and predicted values by Neural Network model is: %s", NN
```

```
## [1] "The correlation between actual values and predicted values by Neural Network model is: 0.773744
```

```
#Denormalizing the predicted values of the Neural Network
NN.prob <- as.data.frame(NN.predict$net.result*(max(NN.train$satisfaction) - min(NN.train$satisfaction)

#Predicting the values for the Neural Network
NN.pred <- as.factor(ifelse(NN.prob < (min(NN.prob) + (max(NN.prob) - min(NN.prob))/2), "neutral or dis

#Printing the confusionMatrix for the Neural Network
NN.CM <- confusionMatrix(NN.pred, test.data[,23])
NN.CM
```

```
## Confusion Matrix and Statistics
```

```
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   18102      5191
##   satisfied                                   261      8916
##
##                Accuracy : 0.8321
##                  95% CI : (0.828, 0.8361)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6439
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9858
##             Specificity : 0.6320
##          Pos Pred Value : 0.7771
##          Neg Pred Value : 0.9716
##              Prevalence : 0.5655
##          Detection Rate : 0.5575
##    Detection Prevalence : 0.7174
##       Balanced Accuracy : 0.8089
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
Sys.time()
```

```
## [1] "2020-08-05 06:06:05 EDT"
```

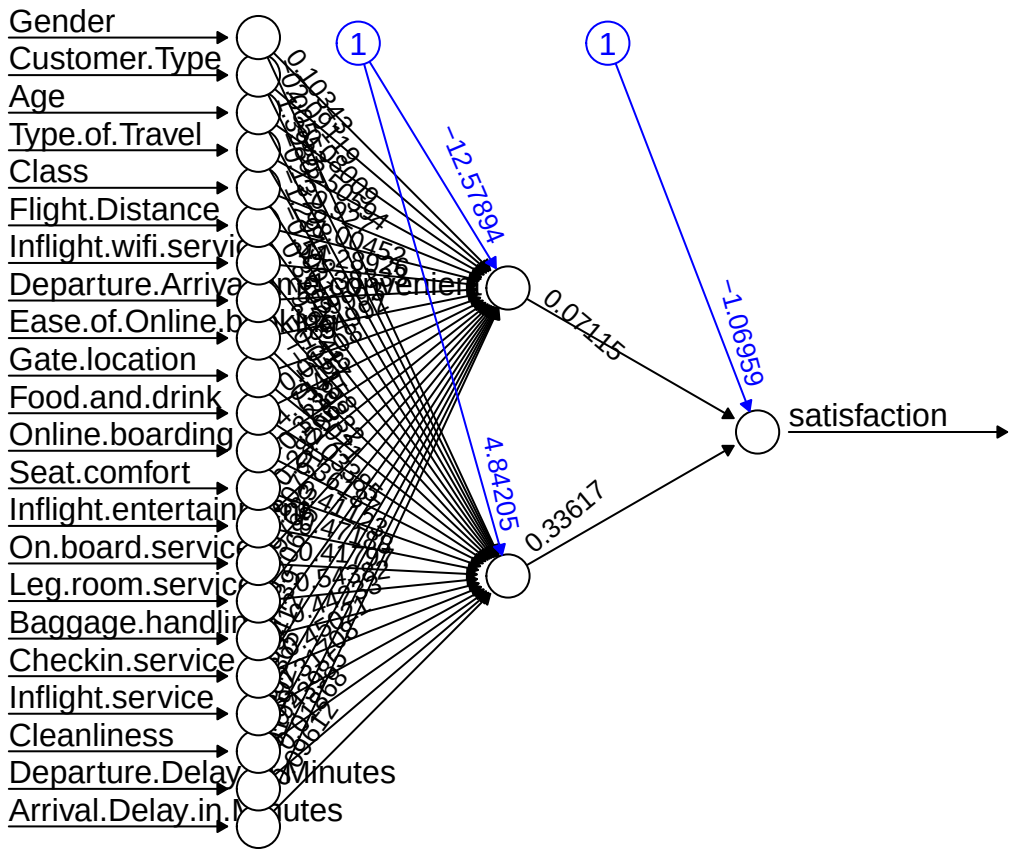
```
##### Neural Network Model Tuning #####

Sys.time()
```

```
## [1] "2020-08-05 06:06:05 EDT"
```

```
#Creating Neural Network model
NN.model_new <- neuralnet(satisfaction ~ ., NN.train, hidden = 2, threshold = 1, rep = 1, linear.output

Sys.time()
```

```
## [1] "2020-08-05 06:44:33 EDT"
```

```
#plotting the neural network graph
plot(NN.model_new, rep="best")
```




```
Sys.time()
```

```
## [1] "2020-08-05 06:44:33 EDT"
```

```
#Computing the values for the test data
NN.predict_new <- compute(NN.model_new, NN.test[,-23])

#Calculating correlation between the actual and predicted values
NN.cor_new <- cor(NN.predict_new$net.result, NN.test$satisfaction)
sprintf("The correlation between actual values and predicted values by the new Neural Network model is:
```

```
## [1] "The correlation between actual values and predicted values by the new Neural Network model is: (
```

```
#Denormalizing the predicted values of the new Neural Network model
NN.prob_new <- as.data.frame(NN.predict_new$net.result*(max(NN.train$satisfaction) - min(NN.train$satisf

#Predicting the values for the new Neural Network model
NN.pred_new <- as.factor(ifelse(NN.prob_new < (min(NN.prob_new) + (max(NN.prob_new) - min(NN.prob_new))/

#Printing the confusionMatrix for the Neural Network
NN.CM_new <- confusionMatrix(NN.pred_new, test.data[,23])
NN.CM_new
```

```
## Confusion Matrix and Statistics
```

```
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17450      1671
##   satisfied                                   913     12436
##
##                Accuracy : 0.9204
##                  95% CI : (0.9174, 0.9233)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.837
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9503
##             Specificity : 0.8815
##          Pos Pred Value : 0.9126
##          Neg Pred Value : 0.9316
##              Prevalence : 0.5655
##          Detection Rate : 0.5374
##    Detection Prevalence : 0.5889
##       Balanced Accuracy : 0.9159
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
Sys.time()
```

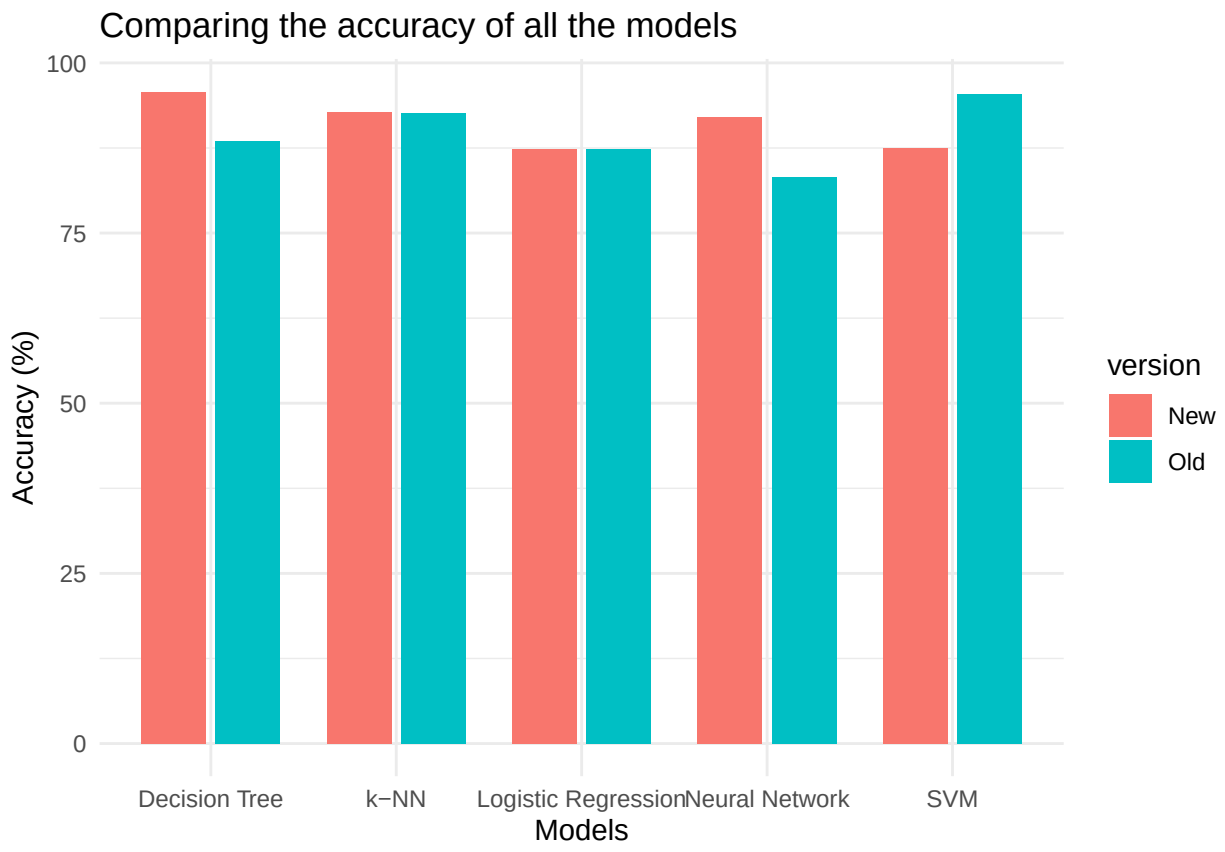
```
## [1] "2020-08-05 06:44:33 EDT"
```

```
#Increasing the number of hidden layers did not significantly increase the accuracy of the Neural Netwo
#I also tried to increase the setpmax and decrease the threshold but the dataset is too large both for
#computer and RStudio Cloud and returns error.



##### Comparision of models #####

#Creating a data frame to compare all models along with their optimized versions
model <- c(rep("Logistic Regression", 2), rep("k-NN", 2), rep("Decision Tree", 2),
           rep("SVM", 2), rep("Neural Network", 2))
version <- rep(c("Old", "New"), 5)
accuracy <- c(LR.CM$overall[1]*100, LR.CM_new$overall[1]*100, kNN.CM$overall[1]*100,
              kNN.CM_new$overall[1]*100, rpart.CM$overall[1]*100, c50.CM$overall[1]*100,
              SVM.CM$overall[1]*100, SVM.CM_new$overall[1]*100, NN.CM$overall[1]*100,
              NN.CM_new$overall[1]*100)

#Printing the accuracy for all the models
models.compare <- data.frame(model, version, accuracy)
models.compare
```

```
##                  model version accuracy
```

```
## 1  Logistic Regression     Old 87.34216
## 2  Logistic Regression     New 87.33600
## 3                  k-NN     Old 92.68556
## 4                  k-NN     New 92.72867
## 5         Decision Tree     Old 88.45088
## 6         Decision Tree     New 95.78688
## 7                   SVM     Old 95.38343
## 8                   SVM     New 87.43763
## 9        Neural Network     Old 83.20912
## 10       Neural Network     New 92.04188
```

```
#Plotting a grouped bar plot for the accuracy of all the models grouped by their versions
ggplot(models.compare, aes(fill=version, y=accuracy, x=model)) +
  geom_bar(stat="identity", width=0.7, position=position_dodge(width=0.8)) +
  theme_minimal() + ggtitle("Comparing the accuracy of all the models") + xlab("Models") + ylab("Accura
```




```
#After comparing the the accuracy of all the models we see that c5.0 is having the highest accuracy amo
#SVM with "kernal = radial" also provides us with a high accuracy


###############################
####### Ensemble Model #######
###############################

##### Construction of an Ensemble Model #####
```

```r
#Creating an ensemble model function
ensemble.model <- function(data)
  {
    LR.prediction <- as.factor(ifelse((predict(LR.model, data, type = "response")) < 0.5,
                                      "neutral or dissatisfied", "satisfied"))
    kNN.prediction <- knn(train = train.data[,-23], test = data[,-23], cl = train.label, k = 5)
    c50.prediction <- predict(c50.model, data[,-23])
    SVM.prediction <- predict(SVM.model, data, probability = TRUE)
    NN.data <- data
    NN.data$satisfaction <- as.integer(NN.data$satisfaction)
    NN.pred_prob <- compute(NN.model_new, NN.data[,-23])
    NN.prediction <- as.data.frame(NN.pred_prob$net.result*(max(NN.data$satisfaction) -
                                   min(NN.data$satisfaction)) + min(NN.data$satisfaction))
    NN.prediction <- as.factor(ifelse(NN.prediction < (min(NN.prediction) + (max(NN.prediction) -
                                      min(NN.prediction))/2), "neutral or dissatisfied", "satisfied"))
    ensemble.predict <-  as.factor(ifelse(LR.prediction == 'satisfied' & kNN.prediction == 'satisfied' &
                                          c50.prediction == 'satisfied', 'satisfied',
                                   ifelse(LR.prediction == 'satisfied' & kNN.prediction == 'satisfied' &
                                          SVM.prediction == 'satisfied', 'satisfied',
                                   ifelse(LR.prediction == 'satisfied' & kNN.prediction == 'satisfied' &
                                          NN.prediction == 'satisfied', 'satisfied',
                                   ifelse(LR.prediction == 'satisfied' & c50.prediction == 'satisfied' &
                                          SVM.prediction == 'satisfied', 'satisfied',
                                   ifelse(LR.prediction == 'satisfied' & c50.prediction == 'satisfied' &
                                          NN.prediction == 'satisfied', 'satisfied',
                                   ifelse(LR.prediction == 'satisfied' & SVM.prediction == 'satisfied' &
                                          NN.prediction == 'satisfied', 'satisfied',
                                   ifelse(kNN.prediction == 'satisfied' & c50.prediction == 'satisfied'
                                          SVM.prediction == 'satisfied', 'satisfied',
                                   ifelse(kNN.prediction == 'satisfied' & SVM.prediction == 'satisfied'
                                          NN.prediction == 'satisfied', 'satisfied',
                                   ifelse(kNN.prediction == 'satisfied' & SVM.prediction == 'satisfied'
                                          NN.prediction == 'satisfied', 'satisfied',
                                   ifelse(c50.prediction == 'satisfied' & SVM.prediction == 'satisfied'
                                          NN.prediction == 'satisfied', 'satisfied',
                                         'neutral or dissatisfied'))))))))))))
    return(ensemble.predict)
  }


#Predicting the values for the test data
ensemble.predict <- ensemble.model(test.data)

#Printing the confusionMatrix for the SVM model
ensemble.CM <- confusionMatrix(ensemble.predict, test.data[,23])
ensemble.CM
```

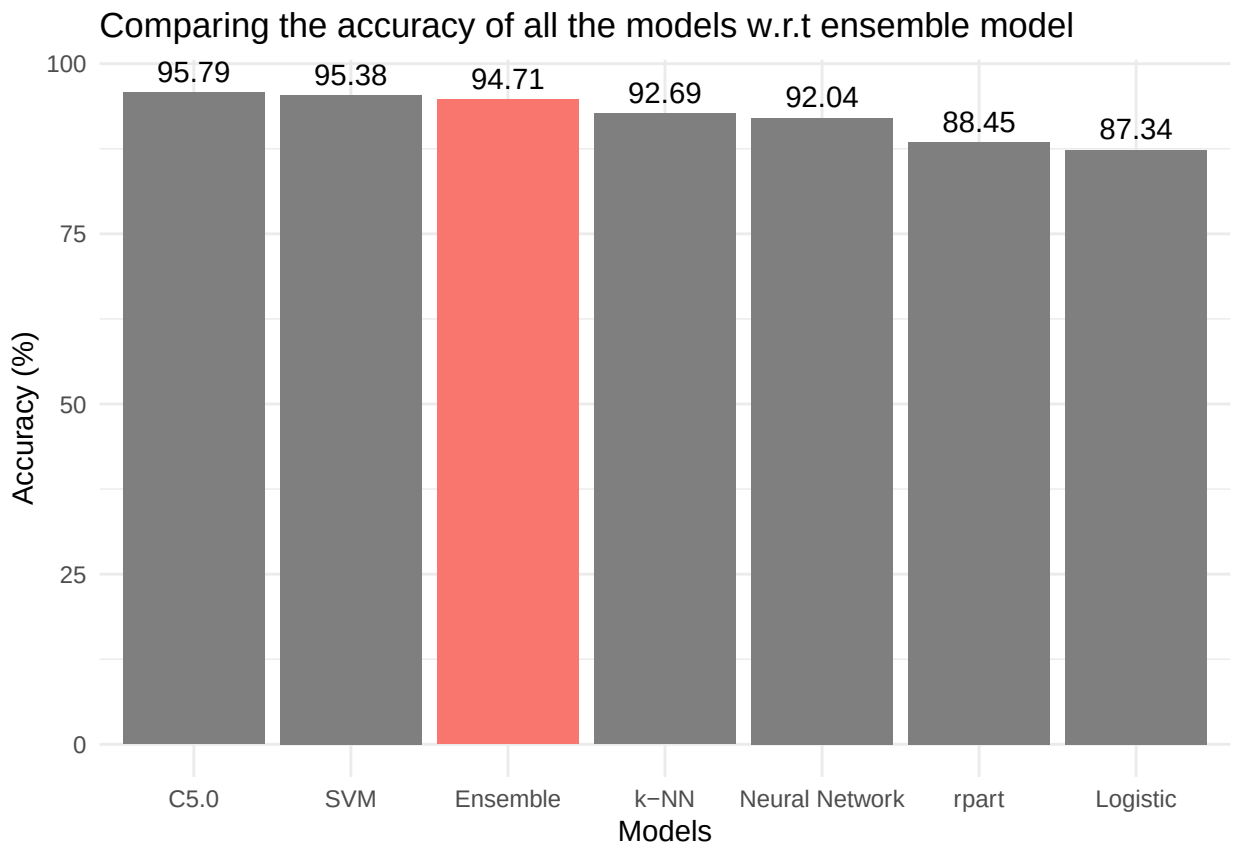
```
## Confusion Matrix and Statistics
##
##                          Reference
## Prediction                neutral or dissatisfied satisfied
##   neutral or dissatisfied                   17765      1119
##   satisfied                                   598     12988
```

```
##
##                Accuracy : 0.9471
##                  95% CI : (0.9446, 0.9495)
##     No Information Rate : 0.5655
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8919
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9674
##             Specificity : 0.9207
##          Pos Pred Value : 0.9407
##          Neg Pred Value : 0.9560
##              Prevalence : 0.5655
##          Detection Rate : 0.5471
##    Detection Prevalence : 0.5816
##       Balanced Accuracy : 0.9441
##
##        'Positive' Class : neutral or dissatisfied
##
```

```
#After creating the ensemble model for all the 5 models we get and accuracy of 93.99%


##### Comparision of models with the Ensemble model #####

#Creating a data frame to compare all models along with the ensemble model
model.names <- c("Logistic", "k-NN", "rpart", "C5.0", "SVM", "Neural Network", "Ensemble")
accuracy.all <- c(round(LR.CM_new$overall[1]*100,2), round(kNN.CM$overall[1]*100, 2), round(rpart.CM$ov
                  round(c50.CM$overall[1]*100, 2), round(SVM.CM$overall[1]*100, 2), round(NN.CM_new$ove
                  round(ensemble.CM$overall[1]*100, 2))

#Printing the accuracy for all the models along with the ensemble model
models.all <- data.frame(model.names, accuracy.all)
models.all
```

```
##      model.names accuracy.all
## 1       Logistic        87.34
## 2           k-NN        92.69
## 3          rpart        88.45
## 4           C5.0        95.79
## 5            SVM        95.38
## 6 Neural Network        92.04
## 7       Ensemble        94.71
```

```
area.color <- c(NA, NA, NA, NA, NA, NA, "withcolor")
#Plotting a grouped bar plot for the accuracy of all the models grouped by their versions
ggplot(models.all, aes(x= reorder(model.names, -accuracy.all), model.names, y=accuracy.all, fill = area
  geom_text(aes(label= accuracy.all), vjust= -0.5) +
  geom_bar(stat="identity") + theme_minimal() + theme(legend.position="none") +
  ggtitle("Comparing the accuracy of all the models w.r.t ensemble model") + xlab("Models") + ylab("Accu
```




```
#A comprehensive comaprison of all the models and their standing with respect to the ensemble model.
#Overall the top 2 models are Decision tree with C5.0 and SVM with Radial Kernal
```