Part 1

Airline Passenger Satisfaction

Introduction:

Since India being the most populous country in the world, transportation plays an important role in the Indian economy. Among different modes of transportation aviation is a fast growing sector. In upcoming years the majority of the Indian population is going to travel by air as it is fast and efficient . Air travelers in India are looking for an airline which is affordable and with the most satisfying services. Even across the globe people expect the same. Not only price, services associated with air travel also play an important role in successful running of civilian aviation.

Objective:

- To construct an Exploratory Data Analysis for the data
- What factors lead to customer satisfaction in air travel ?
- How much do these factors influence the satisfaction level ?
- Predicting passenger satisfaction based on different factors.
- Building logistic regression and decision tree and comparing them.

Statistical tool used in analysis:

R is a programming language for statistical computing and graphics supported by the R Core Team and the R Foundation for Statistical Computing. Created by statisticians Ross Ihaka and Robert Gentleman, R is used among data miners, bioinformaticians and statisticians for data analysis and developing statistical software.[6] Users have created packages to augment the functions of the R language.According to user surveys and studies of scholarly literature databases, R is one of the most commonly used programming languages used in data mining.

Data collection:

Data obtained from an open sourced data repository.

Defining the variables in the dataset:

Gender	Gender of the passengers (Female, Male)
Customer Type	The customer type (Loyal customer, disloyal customer)
Age	The actual age of the passengers
Type of Travel	Purpose of the flight of the passengers (Personal Travel, Business Travel)
Class	Travel class in the plane of the passengers (Business, Eco, Eco Plus)
Flight distance	The flight distance of this journey
Inflight wifi service	Satisfaction level of the inflight wifi service
Departure/Arrival time convenient	Satisfaction level of Departure/Arrival time convenient
Ease of Online booking	Satisfaction level of online booking
Gate location	Satisfaction level of Gate location
Food and drink:	Satisfaction level of Food and drink
Online boarding	Satisfaction level of online boarding
Seat comfort	Satisfaction level of Seat comfort

Inflight entertainment	Satisfaction level of inflight entertainment
On-board service	Satisfaction level of On-board service
Leg room service	Satisfaction level of Leg room service
Baggage handling	Satisfaction level of baggage handling
Check-in service	Satisfaction level of Check-in service
Inflight service	Satisfaction level of inflight service
Cleanliness	Satisfaction level of Cleanliness
Departure Delay in Minutes	Minutes delayed when departure
Arrival Delay in Minutes:	Minutes delayed when Arrival
Satisfaction:	Airline satisfaction level (Satisfaction, dissatisfaction)

Methodology:

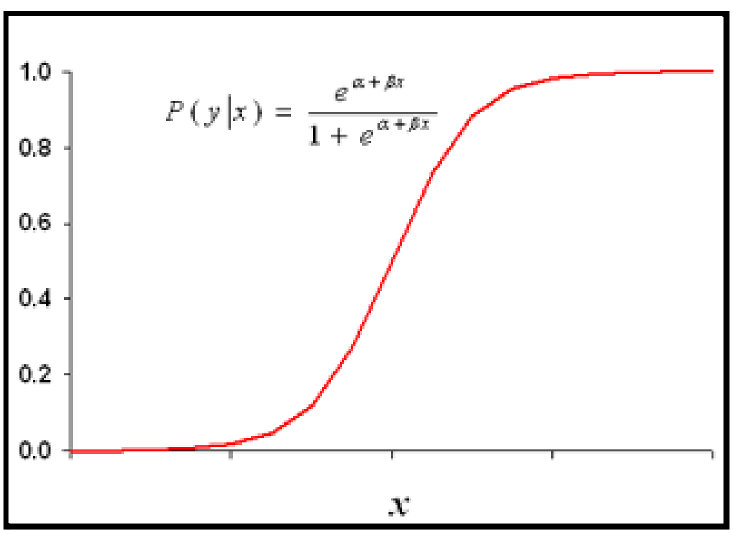
Predict the satisfaction of passengers using logistic regression and decision tree. And comparing the two models.

1)Logistic regression

Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set.

A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables. For example, a logistic regression could be used to predict whether an airline customer will be satisfied or not based on factors like gender, age, travel class, etc.

Logistic regression has become an important tool in the discipline of machine learning. It allows algorithms used in machine learning applications to classify incoming data based on historical data. As additional relevant data comes in, the algorithms get better at predicting classifications within data sets.



- **Sensitivity and Specificity?**

This is what a confusion matrix looks like:

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

From the confusion matrix, it can derive some important metrics.

<u>Sensitivity / True Positive Rate</u>

$$Sensitivity = \frac{TP}{TP + FN}$$

Sensitivity tells what proportion of the positive class got correctly classified

<u>False Negative Rate</u>

$$FNR = \frac{FN}{TP + FN}$$

False Negative Rate (FNR) tells us what proportion of the positive class got incorrectly classified by the classifier.

A higher TPR and a lower FNR is desirable since we want to correctly classify the positive class.

Specificity / True Negative Rate

$$Specificity = \frac{TN}{TN + FP}$$

Specificity tells us what proportion of the negative class got correctly classified.

False Positive Rate

$$FPR = \frac{FP}{TN + FP} = 1 - Specificity$$

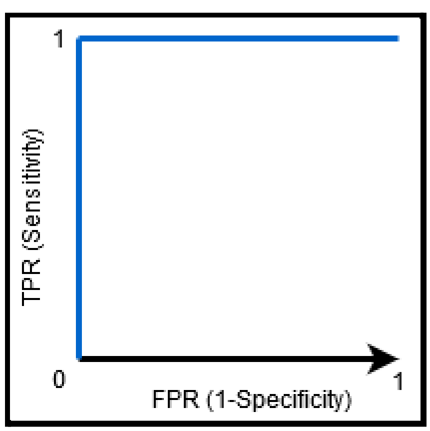
FPR tells us what proportion of the negative class got incorrectly classified by the classifier.

A higher TNR and a lower FPR is desirable since we want to correctly classify the negative class.

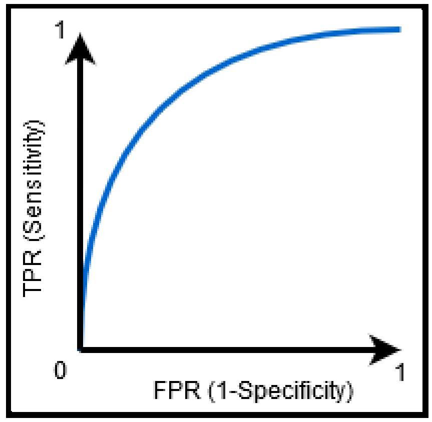
- **ROC curve and AUC**

The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.

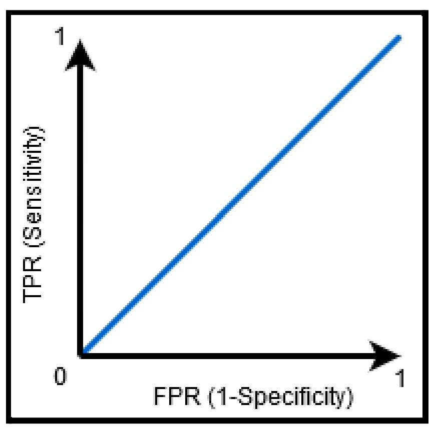
The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.



When AUC = 1, then the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly. If, however, the AUC had been 0, then the classifier would be predicting all Negatives as Positives, and all Positives as Negatives.



When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.



When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points. Meaning either the classifier is predicting random class or constant class for all the data points.

So, the higher the AUC value for a classifier, the better its ability to distinguish between positive and negative classes.

2) Decision Tree

- **Introduction on decision tree:**

Classification is a two-step process, learning step and prediction step, in machine learning. In the learning step, the model is developed based on given training data. In the prediction step, the model is used to predict the response for given data. Decision Tree is one of the easiest and popular classification algorithms to understand and interpret.

- **Decision Tree Algorithm**

The Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can be used to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data).

In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

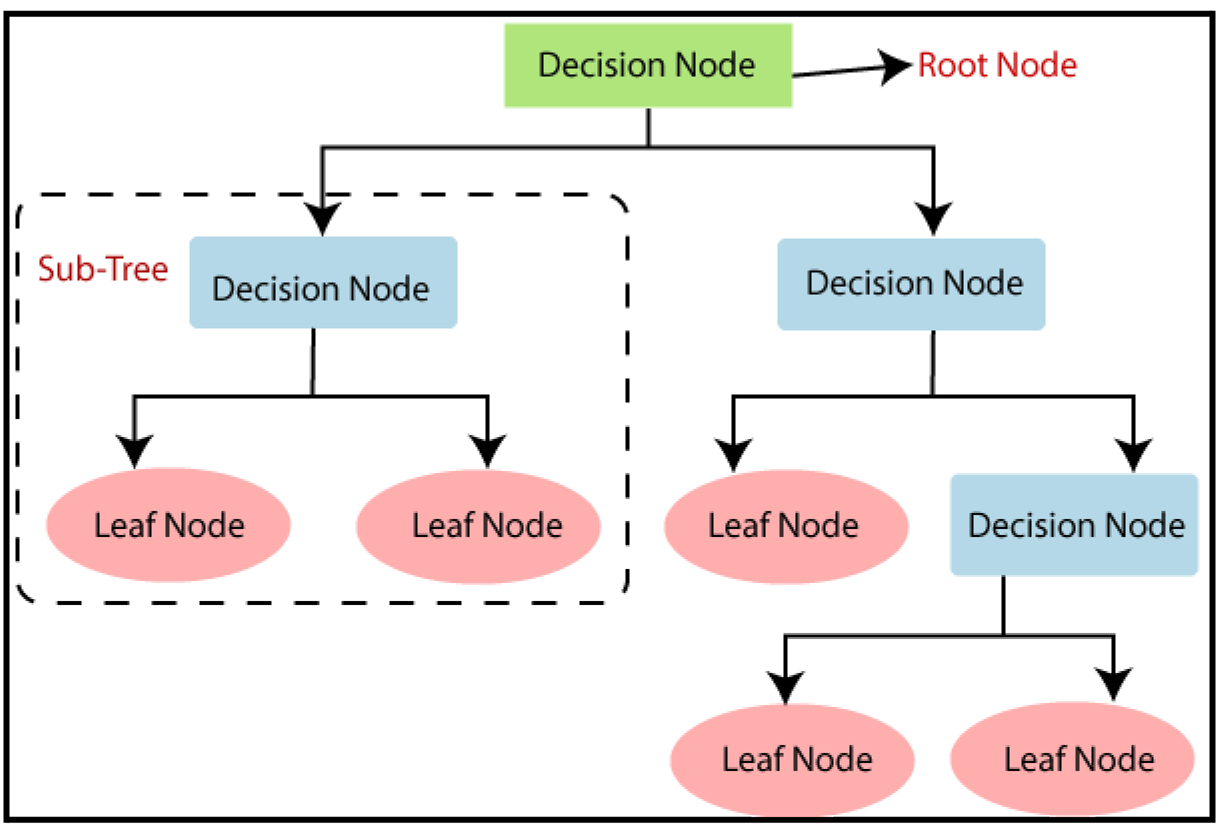
- **Types of Decision Trees**

Types of decision trees are based on the type of target variable we have. It can be of two types:

1. Categorical Variable Decision Tree: Decision Tree which has a categorical target variable then it is called a Categorical variable decision tree.
2. Continuous Variable Decision Tree: Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.

- **Important Terminology related to Decision Trees**

1. Root Node: It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. Splitting: It is a process of dividing a node into two or more sub-nodes.
3. Decision Node: When a sub-node splits into further sub-nodes, then it is called the decision node.
4. Leaf / Terminal Node: Nodes that do not split are called Leaf or Terminal nodes.
5. Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
6. Branch / Sub-Tree: A subsection of the entire tree is called branch or sub-tree.
7. Parent and Child Node: A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.



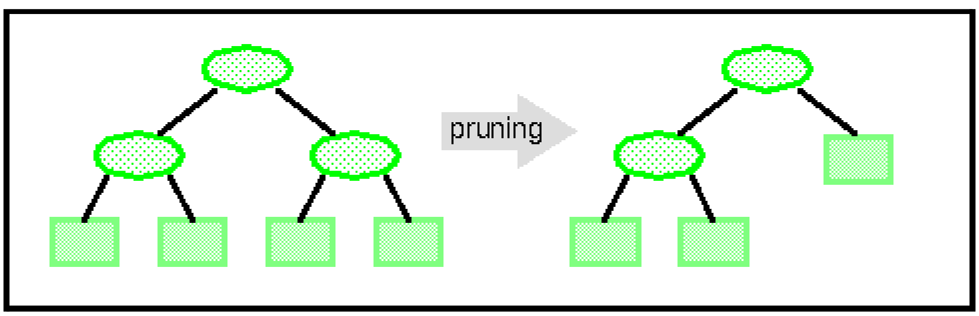
Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example.

Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new node.

- **Pruning Decision Trees:**

The splitting process results in fully grown trees until the stopping criteria are reached. But, the fully grown tree is likely to overfit the data, leading to poor accuracy on unseen data.

In pruning, you trim off the branches of the tree, i.e., remove the decision nodes starting from the leaf node such that the overall accuracy is not disturbed. This is done by segregating the actual training set into two sets: training data set and testing data set. Prepare the decision tree using the segregated training data set. Then continue trimming the tree accordingly to optimize the accuracy of the training data set.



- **Auc in decision tree:**

AUC means Area Under Curve ; you can calculate the area under various curves though. Common is the ROC curve which is about the tradeoff between true positives and false positives at different thresholds. This AUC value can be used as an evaluation metric, especially when there are imbalanced classes.
The AUC score is simply the area under the curve which can be calculated with Simpson's Rule. The bigger the AUC score the better our classifier is.

DATA ANALYSIS

IMPORTING DATASET INTO R STUDIO:

```
> data = read.csv('C:/Users/aaa/Desktop/bsc project/flight_survey.csv',
+                 sep = ",", header = T)
```

Columns in raw data

```
> colnames(data)
 [1] "X"                                 "id"
 [3] "Gender"                            "Customer.Type"
 [5] "Age"                               "Type.of.Travel"
 [7] "Class"                             "Flight.Distance"
 [9] "Inflight.wifi.service"             "Departure.Arrival.time.convenient"
[11] "Ease.of.Online.booking"            "Gate.location"
[13] "Food.and.drink"                    "Online.boarding"
[15] "Seat.comfort"                      "Inflight.entertainment"
[17] "On.board.service"                  "Leg.room.service"
[19] "Baggage.handling"                  "Checkin.service"
[21] "Inflight.service"                  "Cleanliness"
[23] "Departure.Delay.in.Minutes"        "Arrival.Delay.in.Minutes"
[25] "satisfaction"
```

DATA CLEANING:

Dropping columns ‘X’ and ‘id’ which has no use in the analysis

```
> data <- subset(data, select = -X)
> data <- subset(data, select = -id)
> |
```

For convenient column names are renames to lowercase

```
> colnames(data) <- c('gender', 'customer_type', 'age', 'travel_type', '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', 'satisfaction')
> |
```

Checking for null values and removing null values

```
> sum(is.na(data))
[1] 393
> data = na.omit(data)
> sum(is.na(data))
[1] 0
```

Rows and columns

```
> nrow(data)
[1] 129487
> ncol(data)
[1] 23
```

Data has 129487 observation and 23 variables after removing null values
Converting categorical variables content to lowercase:

```
> data$gender = tolower(data$gender)
> data$customer_type = tolower(data$customer_type)
> data$travel_type = tolower(data$travel_type)
> data$class = tolower(data$class)
```

Converting data types of categorical and integer variables to factor type and continuous variables to numeric type:

```
> for (i in colnames(data)){
+   if (i == "age"){next}
+   if (i == "flight_distance"){next}
+   if (i == "departure_delay_in_minutes"){next}
+   if (i == "arrival_delay_in_minutes"){next}
+   data[[i]] = as.factor(data[[i]])}
>
>
> data$age <- as.numeric(data$age)
> data$flight_distance <- as.numeric(data$flight_distance)
> data$departure_delay_in_minutes <- as.numeric(data$departure_delay_in_minutes)
> data$arrival_delay_in_minutes <- as.numeric(data$arrival_delay_in_minutes)
```

Data types of the each variables:

```
> str(data)
'data.frame':	129487 obs. of  23 variables:
 $ 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                              : num  13 25 26 25 61 26 47 52 41 20 ...
 $ travel_type                      : 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                  : num  460 235 1142 562 214 ...
 $ inflight_wifi_service            : Factor w/ 6 levels "0","1","2","3",..: 4 4 3 3 4 4 3 5 2 4 ...
 $ departure_arrival_time_convenient: Factor w/ 6 levels "0","1","2","3",..: 5 3 3 6 4 5 5 4 3 4 ...
 $ ease_of_online_booking           : Factor w/ 6 levels "0","1","2","3",..: 4 4 3 6 4 3 3 5 3 4 ...
 $ gate_location                    : Factor w/ 6 levels "0","1","2","3",..: 2 4 3 6 4 2 4 5 3 5 ...
 $ food_and_drink                   : Factor w/ 6 levels "0","1","2","3",..: 6 2 6 3 5 2 3 6 5 3 ...
 $ online_boarding                  : Factor w/ 6 levels "0","1","2","3",..: 4 4 6 3 6 3 3 6 4 4 ...
 $ seat_comfort                     : Factor w/ 6 levels "0","1","2","3",..: 6 2 6 3 6 2 3 6 4 4 ...
 $ inflight_entertainment           : Factor w/ 6 levels "0","1","2","3",..: 6 2 6 3 4 2 3 6 2 3 ...
 $ on_board_service                 : Factor w/ 6 levels "0","1","2","3",..: 5 2 5 3 4 4 4 6 2 3 ...
 $ leg_room_service                 : Factor w/ 6 levels "0","1","2","3",..: 4 6 4 6 5 5 4 6 3 4 ...
 $ baggage_handling                 : Factor w/ 5 levels "1","2","3","4",..: 4 3 4 3 4 4 4 5 1 4 ...
 $ checkin_service                  : Factor w/ 6 levels "0","1","2","3",..: 5 2 5 2 4 5 4 5 5 5 ...
 $ inflight_service                 : Factor w/ 6 levels "0","1","2","3",..: 6 5 5 5 4 5 6 6 2 4 ...
 $ cleanliness                      : Factor w/ 6 levels "0","1","2","3",..: 6 2 6 3 4 2 3 5 3 3 ...
 $ departure_delay_in_minutes       : num  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 "dissatisfied",..: 1 1 2 1 2 1 1 2 1 1 ...
 - attr(*, "na.action")= 'omit' Named int [1:393] 214 1125 1530 2005 2109 2486 2631 3622 4042 4491 ...
  ..- attr(*, "names")= chr [1:393] "214" "1125" "1530" "2005" ...
>
```

EXPLORATORY DATA ANALYSIS

Displaying first 5 rows of the data

```
> head(data)
  gender      customer_type age     travel_type    class flight_distance inflight_wifi_service departure_arrival_time_convenient ease_of_online_booking
1   male    loyal customer  13 personal travel eco plus             460                     3                                 4                      3
2   male disloyal customer  25 business travel business             235                     3                                 2                      3
3 female    loyal customer  26 business travel business            1142                     2                                 2                      2
4 female    loyal customer  25 business travel business             562                     2                                 5                      5
5   male    loyal customer  61 business travel business             214                     3                                 3                      3
6 female    loyal customer  26 personal travel      eco            1180                     3                                 4                      2
  gate_location food_and_drink online_boarding seat_comfort inflight_entertainment on_board_service leg_room_service baggage_handling checkin_service
1             1              5               3            5                      5                4                3                4               4
2             3              1               3            1                      1                1                5                3               1
3             2              5               5            5                      5                4                3                4               4
4             5              2               2            2                      2                2                5                3               1
5             3              4               5            5                      3                3                4                4               3
6             1              1               2            1                      1                3                4                4               4
  inflight_service cleanliness departure_delay_in_minutes arrival_delay_in_minutes satisfaction
1                5           5                         25                       18 dissatisfied
2                4           1                          1                        6 dissatisfied
3                4           5                          0                        0    satisfied
4                4           2                         11                        9 dissatisfied
5                3           3                          0                        0    satisfied
6                4           1                          0                        0 dissatisfied
```

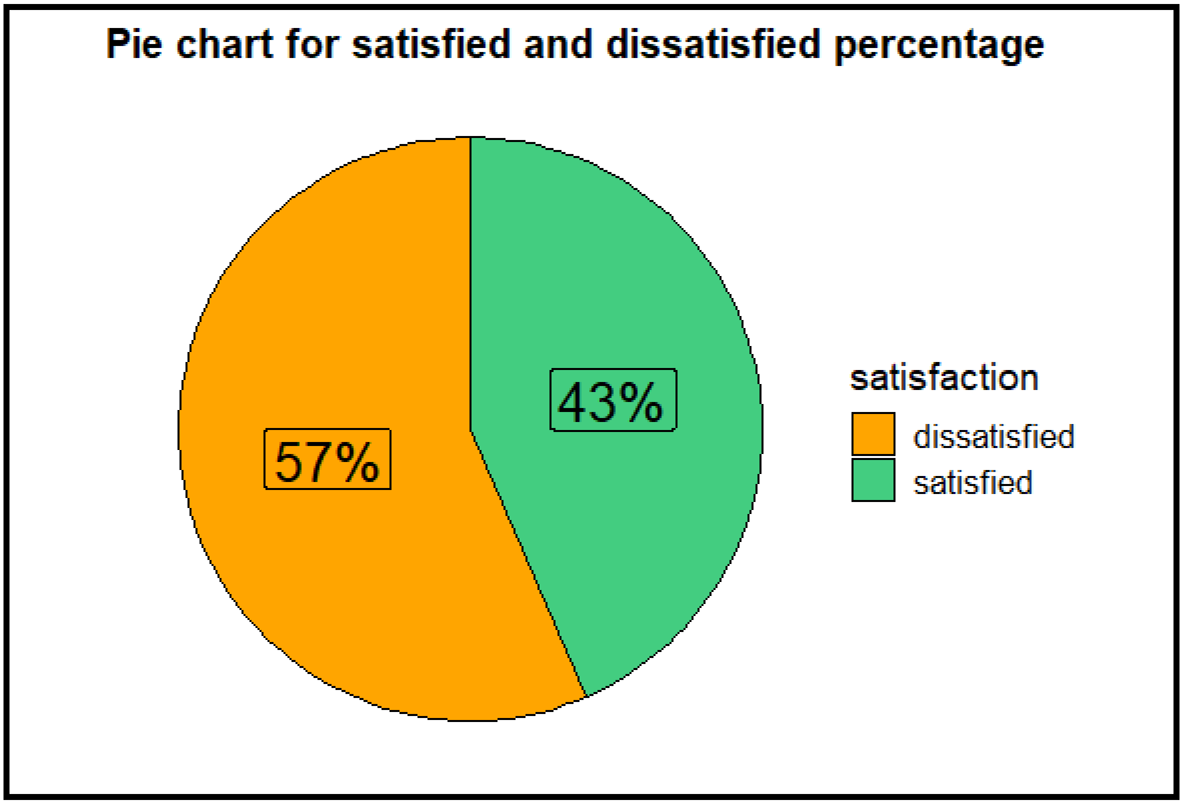
Summary of data:

```
> summary(data)
    gender                 customer_type         age                  travel_type          class        flight_distance
 female:65703   disloyal customer: 23714   Min.   : 7.00   business travel:89445   business:61990   Min.   :  31
 male  :63784   loyal customer   :105773   1st Qu.:27.00   personal travel:40042   eco     :58117   1st Qu.: 414
                                           Median :40.00                           eco plus: 9380   Median : 844
                                           Mean   :39.43                                            Mean   :1190
                                           3rd Qu.:51.00                                            3rd Qu.:1744
                                           Max.   :85.00                                            Max.   :4983
 inflight_wifi_service departure_arrival_time_convenient ease_of_online_booking gate_location food_and_drink online_boarding
 0: 3908               0: 6664                           0: 5666                0:     1      0:  130        0: 3071
 1:22250               1:19351                           1:21808                1:21926       1:16010        1:13216
 2:32236               2:21478                           2:29983                2:24219       2:27293        2:21866
 3:32087               3:22302                           3:30297                3:35611       3:27712        3:27040
 4:24702               4:31786                           4:24362                4:30376       4:30477        4:38353
 5:14304               5:27906                           5:17371                5:17354       5:27865        5:25941
 seat_comfort inflight_entertainment on_board_service leg_room_service baggage_handling checkin_service inflight_service cleanliness
 0:     1     0:   18                0:     5         0:  596          1: 9008          0:     1        0:     5         0:   14
 1:15059      1:15634                1:14738          1:12846          2:14316          1:16058         1: 8838          1:16680
 2:18462      2:21897                2:18290          2:24469          3:25771          2:16056         2:14252          2:20049
 3:23258      3:23805                3:28460          3:24982          4:46631          3:35343         3:25232          3:30552
 4:39651      4:36682                4:38587          4:35779          5:33761          4:36229         4:47198          4:33871
 5:33056      5:31451                5:29407          5:30815                           5:25800         5:33962          5:28321
 departure_delay_in_minutes arrival_delay_in_minutes       satisfaction
 Min.   :   0.00            Min.   :   0.00          dissatisfied:73225
 1st Qu.:   0.00            1st Qu.:   0.00          satisfied   :56262
 Median :   0.00            Median :   0.00
 Mean   :  14.64            Mean   :  15.09
 3rd Qu.:  12.00            3rd Qu.:  13.00
 Max.   :1592.00            Max.   :1584.00
>
```

DATA VISUALIZATION:

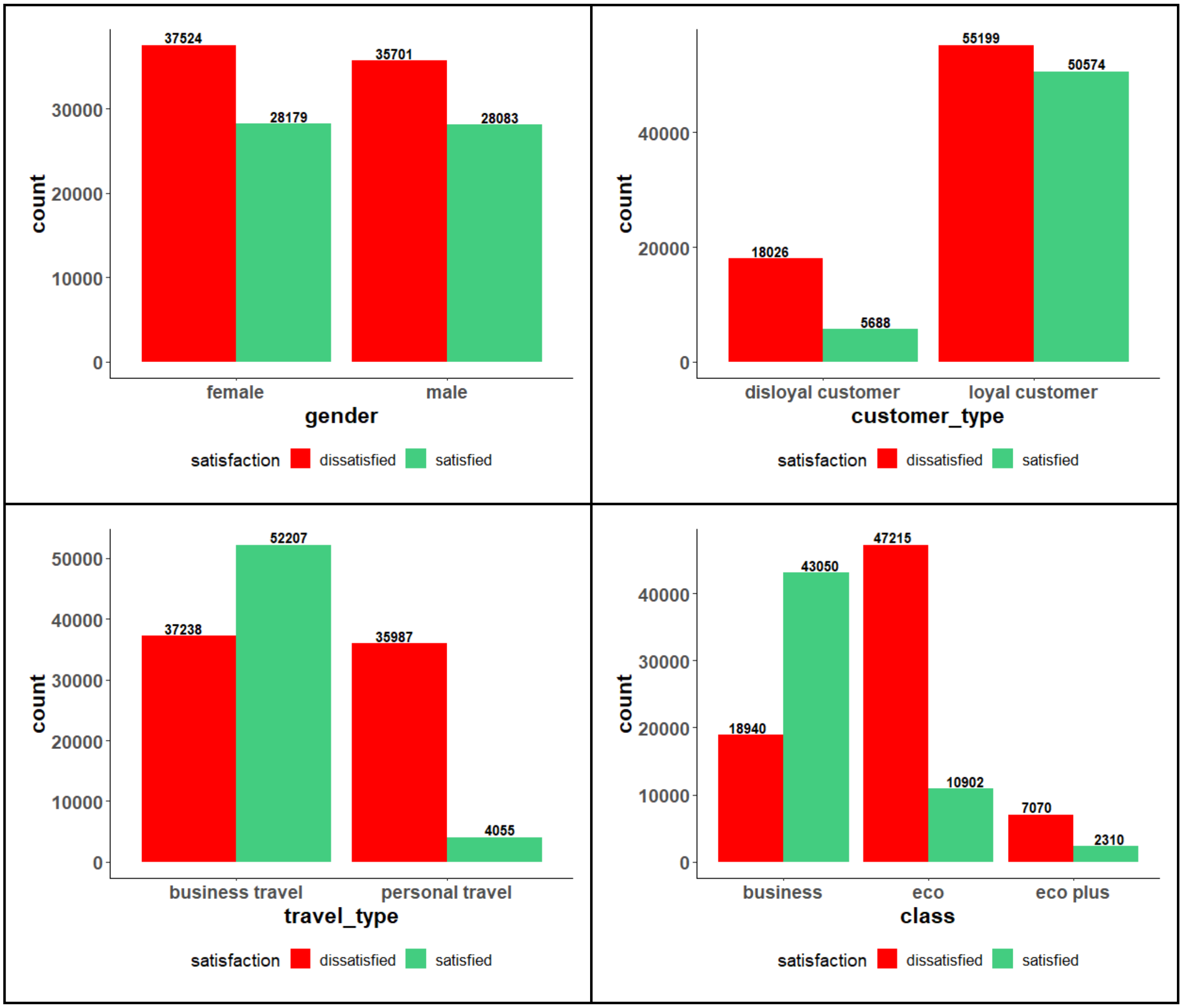
Pie chart:

```
> library(tidyverse)
> pie <- data %>%
+   group_by(satisfaction) %>%
+   count() %>%
+   ungroup() %>%
+   mutate(perc = `n` / sum(`n`)) %>%
+   arrange(perc) %>%
+   mutate(labels = scales::percent(perc))
>
>
> library(ggplot2)
> ggplot(pie, aes(x = "", y = perc, fill = satisfaction)) +
+   geom_col(color = "black") +
+   geom_label(aes(label = labels),
+              position = position_stack(vjust = 0.5),
+              show.legend = F, size = 7) +
+   guides(fill = guide_legend(title = "satisfaction")) +
+   scale_fill_manual(values=c('orange',"seagreen3")) +
+   coord_polar(theta = "y") +
+   theme_void()+
+   labs(title="Pie chart for satisfied and dissatisfied percentage")+
+   theme(plot.title = element_text(face = 'bold',size = 15),
+         legend.title = element_text(size = 14),
+         legend.text = element_text(size=13))
```



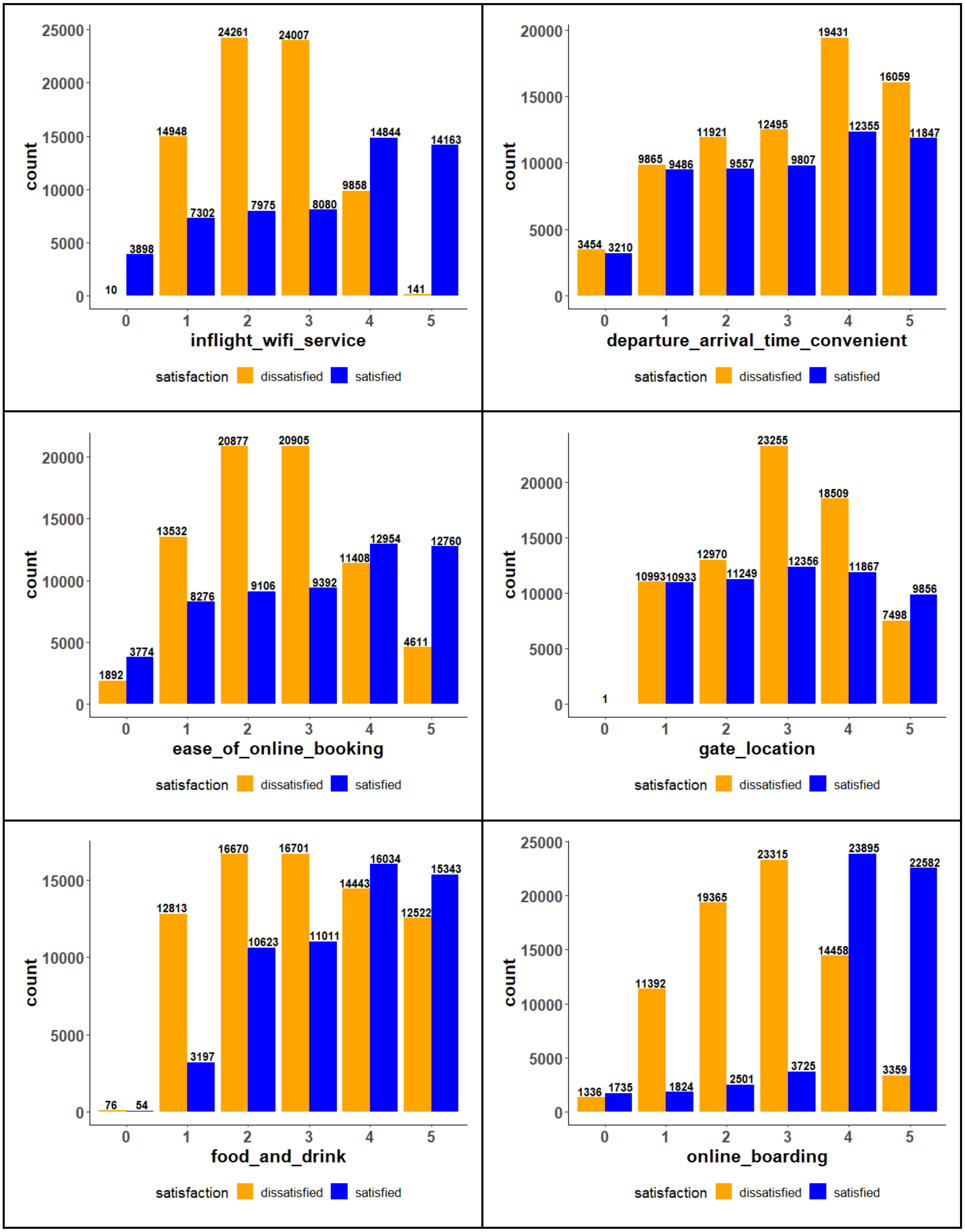
Bar plot for categorical variables:

```
> for (i in c('gender','customer_type','travel_type',
+              'class')){
+   c = factor(data[[i]])
+   print(ggplot(data, aes(x = c, fill = satisfaction))+
+           geom_bar(stat = 'count',position = 'dodge')+
+           scale_fill_manual(values=c("red1","seagreen3"))+
+           geom_text(stat = "count",aes(label = ..count..),
+                     fontface = "bold",vjust= -0.2,
+                     position = position_dodge(width = 1))+labs(x = i)+
+           theme_classic()+
+           theme(axis.text = element_text(face = "bold",size = 14),
+                 axis.title = element_text(face = "bold",size = 17),
+                 legend.title = element_text(size=14),
+                 legend.text = element_text(size=13),
+                 legend.position = "bottom"))}
>
```

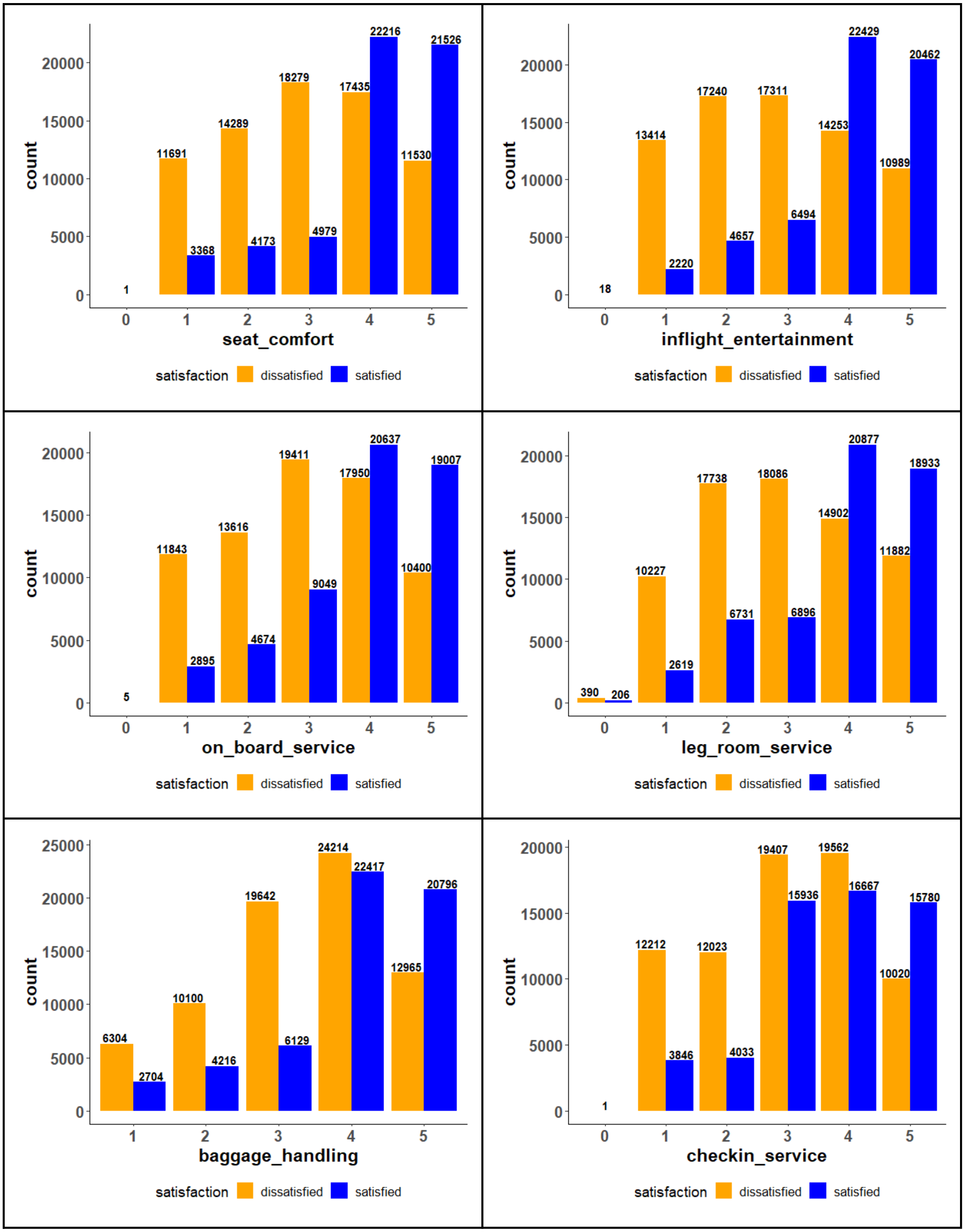


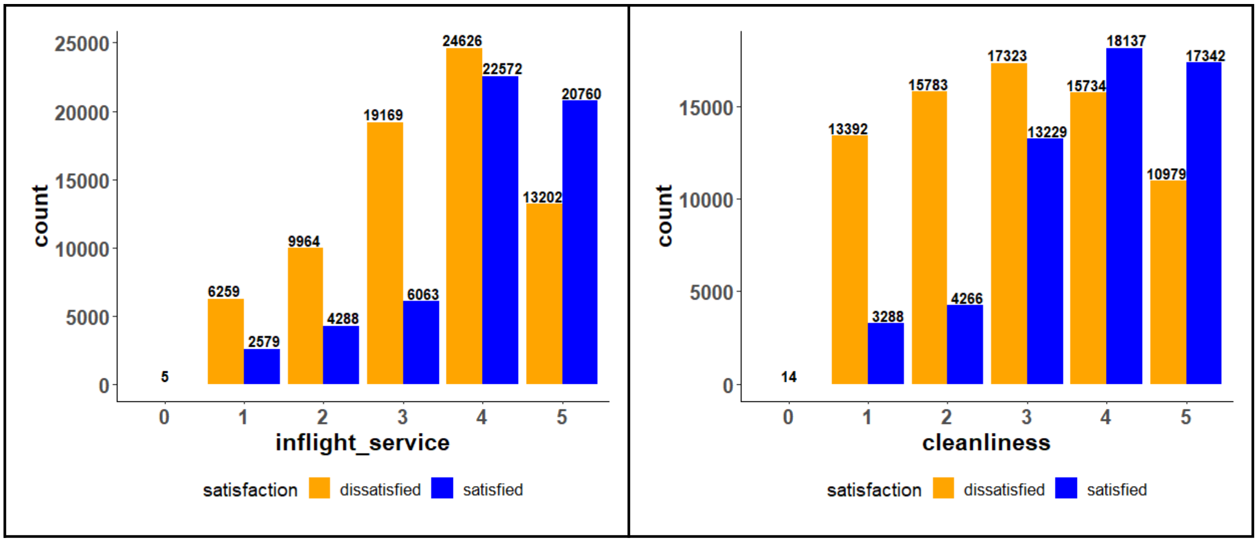
Barplot for rating given to airline services:

```
> for (i in colnames(select_if(data,is.factor))){
+   if (i == "gender"){next}
+   if (i == "travel_type"){next}
+   if (i == "class"){next}
+   if (i == "customer_type"){next}
+   if (i == "satisfaction"){next}
+   c = factor(data[[i]])
+   print(ggplot(data, aes(x = c, fill = satisfaction))+
+           geom_bar(stat = 'count',position = 'dodge')+
+           scale_fill_manual(values=c("#FFA500","blue"))+
+           geom_text(stat = "count",aes(label = ..count..),
+                     fontface = "bold",vjust= -0.2,
+                     position = position_dodge(width = 1))+labs(x = i)+
+           theme_classic()+
+           theme(axis.text = element_text(face = "bold",size = 15),
+                 axis.title = element_text(face = "bold",size = 17),
+                 legend.title = element_text(size=14),
+                 legend.text = element_text(size=13),
+                 legend.position = "bottom"))}
> |
```



count
inflight_wifi_service
0 1 2 3 4 5
10 3898 14948 7302 24261 7975 24007 8080 9858 14844 141 14163
satisfaction dissatisfied satisfied
count
departure_arrival_time_convenient
0 1 2 3 4 5
3454 3210 9865 9486 11921 9557 12495 9807 19431 12355 16059 11847
satisfaction dissatisfied satisfied
count
ease_of_online_booking
0 1 2 3 4 5
1892 3774 13532 8276 20877 9106 20905 9392 11408 12954 4611 12760
satisfaction dissatisfied satisfied
count
gate_location
0 1 2 3 4 5
1 10993 10933 12970 11249 23255 12356 18509 11867 7498 9856
satisfaction dissatisfied satisfied
count
food_and_drink
0 1 2 3 4 5
76 54 12813 3197 16670 10623 16701 11011 14443 16034 12522 15343
satisfaction dissatisfied satisfied
count
online_boarding
0 1 2 3 4 5
1336 1735 11392 1824 19365 2501 23315 3725 14458 23895 3359 22582
satisfaction dissatisfied satisfied


20000
15000
10000
5000
0
count
1
11691
3368
14289
4173
18279
4979
17435
22216
11530
21526
0 1 2 3 4 5
seat_comfort
satisfaction dissatisfied satisfied
20000
15000
10000
5000
0
count
18
13414
2220
17240
4657
17311
6494
14253
22429
10989
20462
0 1 2 3 4 5
inflight_entertainment
satisfaction dissatisfied satisfied
20000
15000
10000
5000
0
count
5
11843
2895
13616
4674
19411
9049
17950
20637
10400
19007
0 1 2 3 4 5
on_board_service
satisfaction dissatisfied satisfied
20000
15000
10000
5000
0
count
390
206
10227
2619
17738
6731
18086
6896
14902
20877
11882
18933
0 1 2 3 4 5
leg_room_service
satisfaction dissatisfied satisfied
25000
20000
15000
10000
5000
0
count
6304
2704
10100
4216
19642
6129
24214
22417
12965
20796
1 2 3 4 5
baggage_handling
satisfaction dissatisfied satisfied
20000
15000
10000
5000
0
count
1
12212
3846
12023
4033
19407
15936
19562
16667
10020
15780
0 1 2 3 4 5
checkin_service
satisfaction dissatisfied satisfied



Interpretation: For rating variables as rating increases from 0 to 5 no of satisfied in satisfaction also increases.

MODEL BUILDING:

1)Logistic Regression:

Splitting dataset into training and testing sets:

Training data 80%

Testing data 20%

```
> set.seed(1)
> sample = sample(2,nrow(data), replace = T, prob = c(0.8, 0.2))
> train <- data[sample==1, ]
> test  <- data[sample==2, ]
> dim(train)
[1] 103675     23
> dim(test)
[1] 25812    23
> 
```

Dependent variable : satisfaction (satisfied(1), dissatisfied (0))

Independent variables : customer type + age + travel type + class + departure delay in minutes + arrival delay in minutes.

Variables with ratings ie (0 to 5) such as 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, on-board service, baggage handling, check-in service, inflight service, cleanliness were not included in the model because dependent variable ‘satisfaction’ is a direct function of these variables.

And variables gender and flight distance were not included in the model because these variables were insignificant i.e. p value > 0.05.

Therefore, logistic regression model built with significant variables:

```
>
> #logistic regression:
> logistic <- glm(satisfaction ~ customer_type + age + travel_type +
+                   class + departure_delay_in_minutes +
+                   arrival_delay_in_minutes, data=train, family="binomial")
> summary(logistic)

Call:
glm(formula = satisfaction ~ customer_type + age + travel_type +
    class + departure_delay_in_minutes + arrival_delay_in_minutes,
    family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8626  -0.5848  -0.4243   0.6936   2.9587

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -0.4042885  0.0270679 -14.936  < 2e-16 ***
customer_typeloyal customer  1.7514423  0.0219717  79.713  < 2e-16 ***
age                         -0.0007899  0.0005861  -1.348    0.178
travel_typepersonal travel  -2.2920696  0.0235450 -97.348  < 2e-16 ***
classeco                    -1.2482947  0.0185171 -67.413  < 2e-16 ***
classeco plus               -1.3963494  0.0317733 -43.947  < 2e-16 ***
departure_delay_in_minutes   0.0048764  0.0008098   6.022 1.72e-09 ***
arrival_delay_in_minutes    -0.0094140  0.0008007 -11.758  < 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: 141955  on 103674  degrees of freedom
Residual deviance:  99804  on 103667  degrees of freedom
AIC: 99820

Number of Fisher Scoring iterations: 4
```

Exp of coefficients from the model

```
> print('Exp')
[1] "Exp"
> exp(-0.4042885195) #intercept
[1] 0.6674515
> exp(1.7514423082 ) #customer_type loyal customer
[1] 5.762909
> exp(-0.0007898689) #age
[1] 0.9992104
> exp(-2.2920695774) #travel_type personal travel
[1] 0.1010571
> exp(-1.2482946730) #class eco
[1] 0.2869938
> exp(-1.3963493577) #class eco plus
[1] 0.2474988
> exp(0.0048764152)  #departure delay in minutes
[1] 1.004888
> exp(-0.0094140084) #arrival delay in minutes
[1] 0.9906302
```

Interpretation form above summary:

- From the intercept it is clear the odds for satisfaction are generally low.
- Compared to disloyal customers, loyal customers have a very high feeling of satisfaction.
- As age increases by one unit, odds for satisfaction reduce slightly.
- Compared to business travel, the odds for satisfaction among personal travel is 90% lower. This may be because business travel usually occurs in the business class section where facilities and comfort are more compared to economy section.
- Compared to business class the odds for satisfaction among economy class passengers is 71% lower.
- Compared to business class the odds for satisfaction among economy plus class passengers is 75% lower.
- For one unit increase in departure delay in minutes odds for satisfaction is slightly higher, almost negligible. This could be because a few minutes delay in departure time will allow the passengers who arrive late to boarding gates to board the flight. But a delay of several hours will also occur due to harsh weather conditions such as thunderstorms,

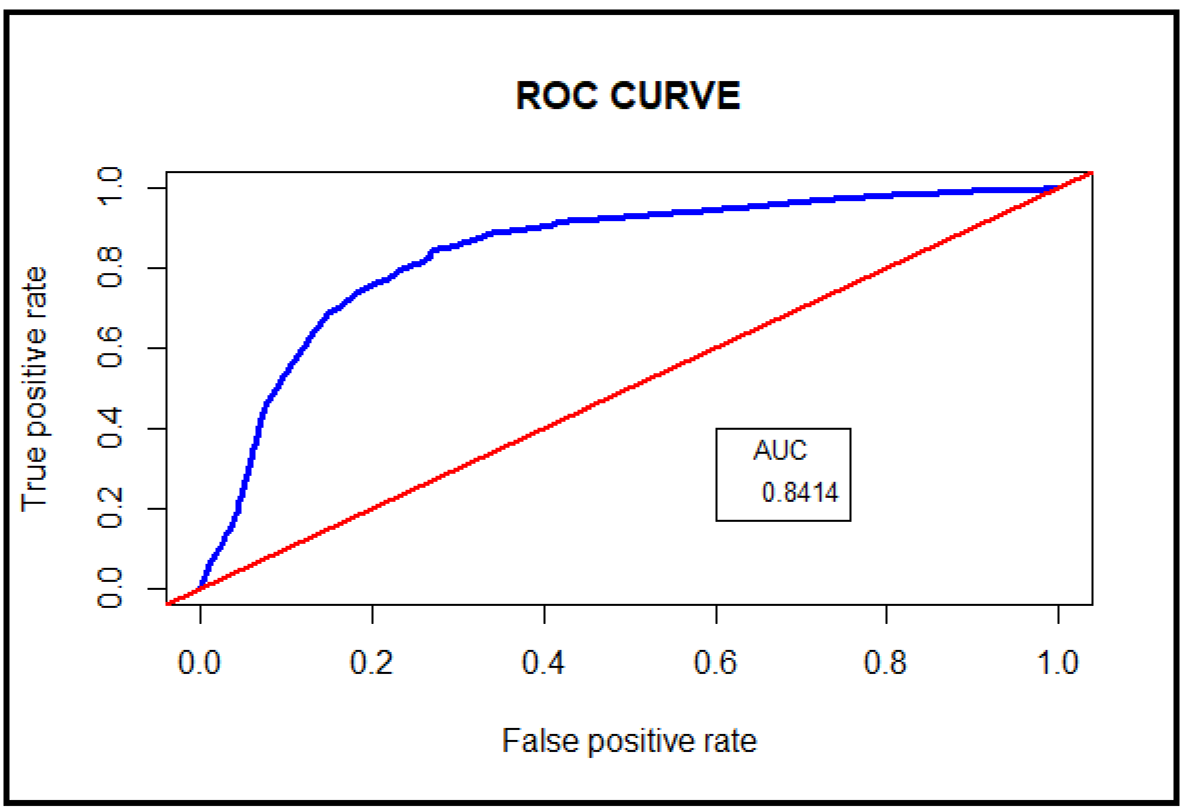
lightning, rain, fog, wind and snow. And these types of issues will be understood by the passengers as flying in bad weather can be dangerous, and no one wants to risk their life.

- For one unit increase in arrival delay in minutes odd for satisfaction is 1% lower. So passengers want to land at the exact arrival time and they can't accept any delay in it.

ROC curve and AUC:

```
> library('ROCR')
> pred = predict(logistic, test, type = 'response')
> rocr_pred = prediction(pred, test$satisfaction)
> rocr_perf = performance(rocr_pred, 'tpr', 'fpr')
> plot(rocr_perf, col = 'blue', main = 'ROC CURVE', lwd = 3 )
> abline(0,1,col = 'red', lwd = 2)
>
> auc = performance(rocr_pred,'auc')
> auc = unlist(slot(auc, 'y.values'))
> auc
[1] 0.841434
> legend(.6,.4,round(auc,4), cex = 0.8, title = 'AUC')
>
```

Area under the curve is found to be 0.8414



Finding optimum cut value:

True positive rate and true negative rate is computed for cut values (0 to1) and absolute difference between each TPR and TNR is calculated. Finally optimum cut value is chosen for which the corresponding absolute difference between each TPR and TNR is minimum.

```
> actual = ifelse(train$satisfaction == 'satisfied', 1, 0)
> predprob = predict(logistic, train, type = 'response')
>
> getTPRFPR=function(actual, predProb, cutoff){
+   if(cutoff==1){tpr=0; fpr=0}else{
+     if(cutoff==0){tpr=1; fpr=1}else{
+       predClass=ifelse(predProb<cutoff,0,1); tab=table(actual, predClass)
+       tpr=tab[2,2]/ ( tab[2,2]+tab[2,1] )
+       tnr=tab[1,1]/ ( tab[1,1]+tab[1,2] ); fpr=1-tnr
+     }
+   }
+   return(c(TPR=tpr, TNR=tnr))
+ }
> cutoff_values=seq(0.3,0.7,0.01)->TPR->TNR
> for(i in 1: length(TPR)){
+   metrics=getTPRFPR(train$satisfaction,logistic$fitted.values,cutoff_values[i])
+   TPR[i]=metrics[1]
+   TNR[i]=metrics[2]
+ }
> cut = data.frame((cbind(cutoff_values, TPR, TNR)))
>
> cut$abs_diff= abs(cut$TPR-cut$TNR)
```

```
> cut
   cutoff_values       TPR       TNR    abs_diff
1           0.30 0.8880311 0.6649147 0.223116384
2           0.31 0.8875430 0.6660068 0.221536158
3           0.32 0.8867887 0.6670137 0.219775034
4           0.33 0.8857016 0.6683447 0.217356899
5           0.34 0.8841708 0.6704437 0.213727140
6           0.35 0.8826179 0.6724744 0.210143456
7           0.36 0.8805768 0.6755631 0.205013676
8           0.37 0.8774043 0.6792662 0.198138115
9           0.38 0.8734332 0.6851365 0.188296648
10          0.39 0.8666223 0.6944369 0.172185436
11          0.40 0.8287743 0.7380034 0.090770852
12          0.41 0.8228730 0.7432082 0.079664798
13          0.42 0.8206545 0.7455973 0.075057195
14          0.43 0.8191237 0.7474573 0.071666345
15          0.44 0.8173710 0.7498805 0.067490502
16          0.45 0.8150860 0.7528498 0.062236138
17          0.46 0.8120466 0.7571331 0.054913483
18          0.47 0.8073655 0.7625768 0.044788710
19          0.48 0.7935219 0.7745392 0.018982659
20          0.49 0.7780366 0.7868089 0.008772268
21          0.50 0.7719800 0.7943174 0.022337373
22          0.51 0.7642374 0.8022867 0.038049307
23          0.52 0.7041819 0.8412287 0.137046750
24          0.53 0.6948197 0.8486007 0.153780938
25          0.54 0.6929784 0.8500512 0.157072825
26          0.55 0.6924903 0.8503413 0.157851003
27          0.56 0.6919578 0.8504437 0.158485838
28          0.57 0.6916473 0.8506143 0.158967080
29          0.58 0.6911814 0.8507167 0.159535359
30          0.59 0.6906933 0.8509044 0.160211148
31          0.60 0.6900055 0.8511604 0.161154863
32          0.61 0.6892956 0.8513652 0.162069569
33          0.62 0.6884526 0.8516212 0.163168581
34          0.63 0.6875208 0.8520819 0.164561113
35          0.64 0.6861897 0.8524744 0.166284719
36          0.65 0.6850361 0.8529693 0.167933232
37          0.66 0.6833943 0.8536348 0.170240470
38          0.67 0.6814199 0.8540956 0.172675707
39          0.68 0.6792235 0.8548976 0.175674095
40          0.69 0.6765835 0.8557167 0.179133252
41          0.70 0.6732113 0.8566382 0.183426911
> cut[cut$abs_diff == min(cut$abs_diff),]
   cutoff_values       TPR       TNR    abs_diff
20          0.49 0.7780366 0.7868089 0.008772268
> 
```

Optimum cut value is found to be = 0.49 (logistic regression)

Applying optimum cut value in testing dataset:

Confusion matrix is built using optimum cut value

```
> actual = test$satisfaction
> predicted <- predict(logistic, test, type = 'response')
> predicted <- ifelse(predicted > 0.49 , 'satisfied' , 'dissatisfied')
> tab = table(predicted, actual)
> tab
              actual
predicted      dissatisfied satisfied
  dissatisfied        11488      2569
  satisfied            3137      8618
>
```

$$\textbf{Sensitivity} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}}$$

$$\textbf{Specificity} = \frac{\textbf{True Negatives}}{\textbf{True Negatives} + \textbf{False Positives}}$$

- Sensitivity = 8618 / (8618+2569) = 0.77

From the logistic regression model using optimum cut value 0.49, sensitivity is found to be 0.77, so 77% of satisfied in satisfaction variable can be clearly classified.

- Specificity = 11488 / (11488+3137) = 0.785

From the logistic regression model using optimum cut value 0.49, specificity is found to be 0.785, so 78.5% of dissatisfied in satisfaction variable can be clearly classified.

2) Decision tree:

Splitting dataset into training and testing sets:

Training data 80%

Testing data 20%

```
> set.seed(1)
> sample = sample(2,nrow(data), replace = T, prob = c(0.8, 0.2))
> train1 <- data[sample==1, ]
> test1  <- data[sample==2, ]
> dim(train1)
[1] 103675     23
> dim(test1)
[1] 25812    23
> |
```

Dependent variable : satisfaction (satisfied, dissatisfied)

Independent variables : gender + customer type + age + travel type + class + flight distance + departure delay in minutes + arrival delay in minutes.

Variables with ratings ie (0 to 5) such as 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, on-board service, baggage handling, check-in service, inflight service, cleanliness were not included in the model because dependent variable 'satisfaction' is a direct function of these variables.

Therefore, decision tree ie (classification tree) is built,

```
> library(rpart)
> library(rpart.plot)
>
> ctree = rpart(satisfaction ~ gender + customer_type + age + travel_type +
+                class + flight_distance + departure_delay_in_minutes +
+                arrival_delay_in_minutes,train1, cp = 0)
> |
```

```
> ctree[["cptable"]]
             CP nsplit rel error    xerror        xstd
1  4.292180e-01      0 1.0000000 1.0000000 0.003541147
2  3.515252e-02      1 0.5707820 0.5707820 0.003085531
3  2.072102e-02      3 0.5004770 0.5004770 0.002947409
4  5.627658e-03      4 0.4797560 0.4797560 0.002902315
5  1.601775e-03      7 0.4628730 0.4628730 0.002863980
6  1.530782e-03     12 0.4548641 0.4561952 0.002848409
7  7.764836e-04     15 0.4488963 0.4506711 0.002835348
8  6.655574e-04     17 0.4473433 0.4502718 0.002834398
9  3.993344e-04     18 0.4466778 0.4496062 0.002832812
10 3.216861e-04     19 0.4462784 0.4492734 0.002832018
11 2.884082e-04     21 0.4456351 0.4486301 0.002830481
12 2.773156e-04     22 0.4453466 0.4484748 0.002830110
13 2.218525e-04     24 0.4447920 0.4480976 0.002829208
14 1.996672e-04     27 0.4441265 0.4478536 0.002828624
15 1.885746e-04     30 0.4435275 0.4481420 0.002829314
16 1.663894e-04     32 0.4431503 0.4478758 0.002828677
17 1.552967e-04     34 0.4428175 0.4482529 0.002829579
18 1.508597e-04     36 0.4425069 0.4484748 0.002830110
19 1.497504e-04     41 0.4417526 0.4484526 0.002830057
20 1.442041e-04     47 0.4405990 0.4483860 0.002829898
21 1.331115e-04     49 0.4403106 0.4484969 0.002830163
22 1.220189e-04     56 0.4393788 0.4487854 0.002830852
23 9.983361e-05     66 0.4379590 0.4485191 0.002830216
24 9.428730e-05     71 0.4374487 0.4480532 0.002829102
25 8.874099e-05     75 0.4370715 0.4480754 0.002829155
26 8.134590e-05     86 0.4360732 0.4491403 0.002831700
27 7.764836e-05     94 0.4353411 0.4490072 0.002831382
28 7.395082e-05    128 0.4320799 0.4489407 0.002831223
29 7.289438e-05    146 0.4304382 0.4499612 0.002833658
30 7.210205e-05    178 0.4269107 0.4499612 0.002833658
31 7.099279e-05    182 0.4266223 0.4499612 0.002833658
32 6.655574e-05    187 0.4262673 0.4500499 0.002833869
33 6.211869e-05    234 0.4229839 0.4519578 0.002838405
```

```
34 6.100943e-05    267 0.4200333 0.4535108 0.002842083
35 5.916066e-05    274 0.4195674 0.4535108 0.002842083
36 5.768164e-05    305 0.4173489 0.4539989 0.002843236
37 5.546312e-05    315 0.4167277 0.4543760 0.002844126
38 5.324459e-05    394 0.4116251 0.4552413 0.002846165
39 5.176558e-05    404 0.4110926 0.4558181 0.002847522
40 5.042102e-05    432 0.4096062 0.4560621 0.002848096
41 4.991681e-05    447 0.4087854 0.4560621 0.002848096
42 4.753981e-05    471 0.4071436 0.4575485 0.002851584
43 4.437049e-05    490 0.4060122 0.4575485 0.002851584
44 4.159734e-05    675 0.3968497 0.4587909 0.002854490
45 3.882418e-05    706 0.3952080 0.4612091 0.002860122
46 3.697541e-05    720 0.3944759 0.4616306 0.002861101
47 3.549639e-05    743 0.3935663 0.4634276 0.002865263
48 3.327787e-05    762 0.3928564 0.4639601 0.002866493
```

```
49 3.142910e-05    882 0.3882418 0.4661786 0.002871602
50 3.105935e-05    901 0.3874653 0.4665336 0.002872417
51 2.958033e-05    929 0.3859789 0.4667998 0.002873028
52 2.852389e-05   1021 0.3828286 0.4677759 0.002875264
53 2.773156e-05   1028 0.3826290 0.4686190 0.002877192
54 2.535457e-05   1045 0.3820965 0.4691736 0.002878458
55 2.218525e-05   1070 0.3813866 0.4759401 0.002893778
56 1.848771e-05   1431 0.3720022 0.4784914 0.002899494
57 1.774820e-05   1454 0.3714920 0.4829063 0.002909308
58 1.696519e-05   1460 0.3713810 0.4839268 0.002911563
59 1.663894e-05   1477 0.3710926 0.4842818 0.002912345
60 1.584660e-05   1536 0.3700277 0.4874764 0.002919364
61 1.552967e-05   1566 0.3693844 0.4875430 0.002919510
62 1.479016e-05   1580 0.3691403 0.4876983 0.002919850
63 1.386578e-05   1650 0.3678758 0.4889850 0.002922661
64 1.331115e-05   1661 0.3677205 0.4895840 0.002923967
65 1.267728e-05   1688 0.3673211 0.4898946 0.002924643
66 1.232514e-05   1790 0.3652357 0.4898946 0.002924643
67 1.109262e-05   1799 0.3651248 0.4955740 0.002936929
68 1.035312e-05   1942 0.3634831 0.4977038 0.002941496
69 9.507963e-06   1957 0.3633278 0.4982585 0.002942681
70 8.874099e-06   1977 0.3631059 0.4998558 0.002946088
71 8.319468e-06   2005 0.3628397 0.5008541 0.002948211
72 7.395082e-06   2013 0.3627732 0.5046478 0.002956233
73 6.655574e-06   2126 0.3617748 0.5058902 0.002958846
74 6.338642e-06   2145 0.3616195 0.5059567 0.002958986
75 5.546312e-06   2159 0.3615308 0.5091958 0.002965761
76 4.930055e-06   2255 0.3608209 0.5099057 0.002967239
77 4.753981e-06   2273 0.3607321 0.5114143 0.002970373
78 4.437049e-06   2292 0.3606212 0.5114809 0.002970511
79 4.033681e-06   2442 0.3598003 0.5116140 0.002970787
80 3.697541e-06   2453 0.3597560 0.5124126 0.002972441
81 3.169321e-06   2503 0.3595563 0.5134332 0.002974550
82 2.773156e-06   2510 0.3595341 0.5140765 0.002975877
83 2.465027e-06   2526 0.3594897 0.5145646 0.002976883
84 1.848771e-06   2544 0.3594454 0.5149196 0.002977613
85 1.386578e-06   2568 0.3594010 0.5152745 0.002978343
86 1.232514e-06   2584 0.3593788 0.5152745 0.002978343
87 0.000000e+00   2602 0.3593566 0.5152745 0.002978343
> cpvalues<-ctree$cptable[,1]
```

There are 87 cp values for each number of splits. Among these 87 cp values optimum cp value is needed.

For each cp value model is built on a training dataset and scored using testing dataset. Then optimum cp value is selected for which corresponding auc value of test data is maximum.

```
> auc_test = numeric(87)
> for (i in 1:87)
+   {
+     tr = rpart(satisfaction ~ gender + customer_type + age + travel_type +
+                class + flight_distance + departure_delay_in_minutes +
+                arrival_delay_in_minutes,train1, cp = cpvalues[i])
+   pred <- predict(tr, test1, type = 'prob')
+   test_dv <- ifelse(test1$satisfaction=="satisfied",1,0)
+   pred2 = cbind(pred[,2], test_dv)
+   names(pred2) <- c("prob","test_dv")
+   pred3 <- prediction(pred2[,1], pred2[,2])
+   pef_measure <- performance(pred3, "auc")
+   auc_test[i] <- pef_measure@y.values[[1]]
+ }
> cp_auc = data.frame(cpvalues,auc_test)
> cp_auc
       cpvalues  auc_test
1  4.292180e-01 0.5000000
2  3.515252e-02 0.7528170
3  2.072102e-02 0.7893817
4  5.627658e-03 0.7918908
5  1.601775e-03 0.8372072
6  1.530782e-03 0.8517650
7  7.764836e-04 0.8524871
8  6.655574e-04 0.8525902
9  3.993344e-04 0.8525975
10 3.216861e-04 0.8526435
11 2.884082e-04 0.8528766
12 2.773156e-04 0.8529457
13 2.218525e-04 0.8529478
14 1.996672e-04 0.8530151
15 1.885746e-04 0.8535673
16 1.663894e-04 0.8536148
17 1.552967e-04 0.8535689
18 1.508597e-04 0.8535357
19 1.497504e-04 0.8539048
20 1.442041e-04 0.8539105
21 1.331115e-04 0.8539721
22 1.220189e-04 0.8544407
23 9.983361e-05 0.8544271
24 9.428730e-05 0.8544160
25 8.874099e-05 0.8544330
26 8.134590e-05 0.8542264
27 7.764836e-05 0.8543042
28 7.395082e-05 0.8536195
29 7.289438e-05 0.8534302
30 7.210205e-05 0.8532140
31 7.099279e-05 0.8532283
32 6.655574e-05 0.8530457
```

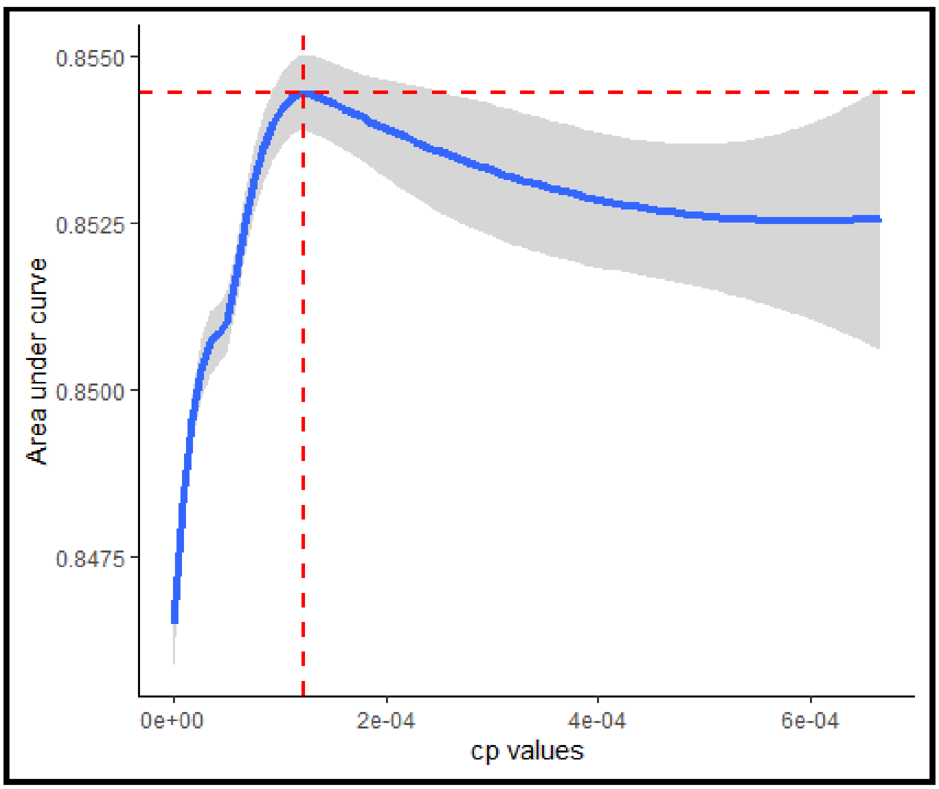
```
33 6.211869e-05 0.8527394
34 6.100943e-05 0.8522661
35 5.916066e-05 0.8522740
36 5.768164e-05 0.8517243
37 5.546312e-05 0.8515684
38 5.324459e-05 0.8507740
39 5.176558e-05 0.8507684
40 5.042102e-05 0.8503585
41 4.991681e-05 0.8500107
42 4.753981e-05 0.8496893
43 4.437049e-05 0.8517494
44 4.159734e-05 0.8506121
45 3.882418e-05 0.8501839
46 3.697541e-05 0.8500652
47 3.549639e-05 0.8500167
48 3.327787e-05 0.8497587
49 3.142910e-05 0.8487448
50 3.105935e-05 0.8487431
51 2.958033e-05 0.8484862
52 2.852389e-05 0.8515186
53 2.773156e-05 0.8514722
54 2.535457e-05 0.8513973
55 2.218525e-05 0.8514289
56 1.848771e-05 0.8519677
57 1.774820e-05 0.8516691
58 1.696519e-05 0.8516360
59 1.663894e-05 0.8511299
60 1.584660e-05 0.8504765
61 1.552967e-05 0.8505472
62 1.479016e-05 0.8505522
63 1.386578e-05 0.8494881
64 1.331115e-05 0.8494490
65 1.267728e-05 0.8495159
66 1.232514e-05 0.8491210
67 1.109262e-05 0.8491092
68 1.035312e-05 0.8471413

69 9.507963e-06 0.8475041
70 8.874099e-06 0.8476017
71 8.319468e-06 0.8470223
72 7.395082e-06 0.8469595
73 6.655574e-06 0.8460814
74 6.338642e-06 0.8459916
75 5.546312e-06 0.8461588
76 4.930055e-06 0.8489351
77 4.753981e-06 0.8486758
78 4.437049e-06 0.8487237
79 4.033681e-06 0.8472976
80 3.697541e-06 0.8473301
81 3.169321e-06 0.8468940
82 2.773156e-06 0.8468818
83 2.465027e-06 0.8468189
84 1.848771e-06 0.8469278
85 1.386578e-06 0.8469422
86 1.232514e-06 0.8467333
87 0.000000e+00 0.8468031
>
```

```
> cp_auc[which.max(cp_auc$auc_test),]
       cpvalues  auc_test
22 0.0001220189 0.8544407
```

Here among 87 cp values optimum cp value is 0.0001220189 for which corresponding auc test score is maximum i.e. 0.8544407.

Cp values vs auc test graph:

```
> library(ggplot2)
> cp_auc = data.frame(cpvalues[8:87], auc_test[8:87])
>
> ggplot(cp_auc, aes(x = cpvalues[8:87], y = auc_test[8:87] )) +
+   geom_smooth(cex = 1.5,method = 'loess') + theme_classic() +
+   geom_vline(xintercept = 0.0001220189, cex = 1, color = 'red', linetype="dashed")+
+   geom_hline(yintercept= 0.8544407, cex = 1, color = 'red',linetype="dashed")+
+   labs(x = 'cp values', y = 'Area under curve')
`geom_smooth()` using formula 'y ~ x'
>
```

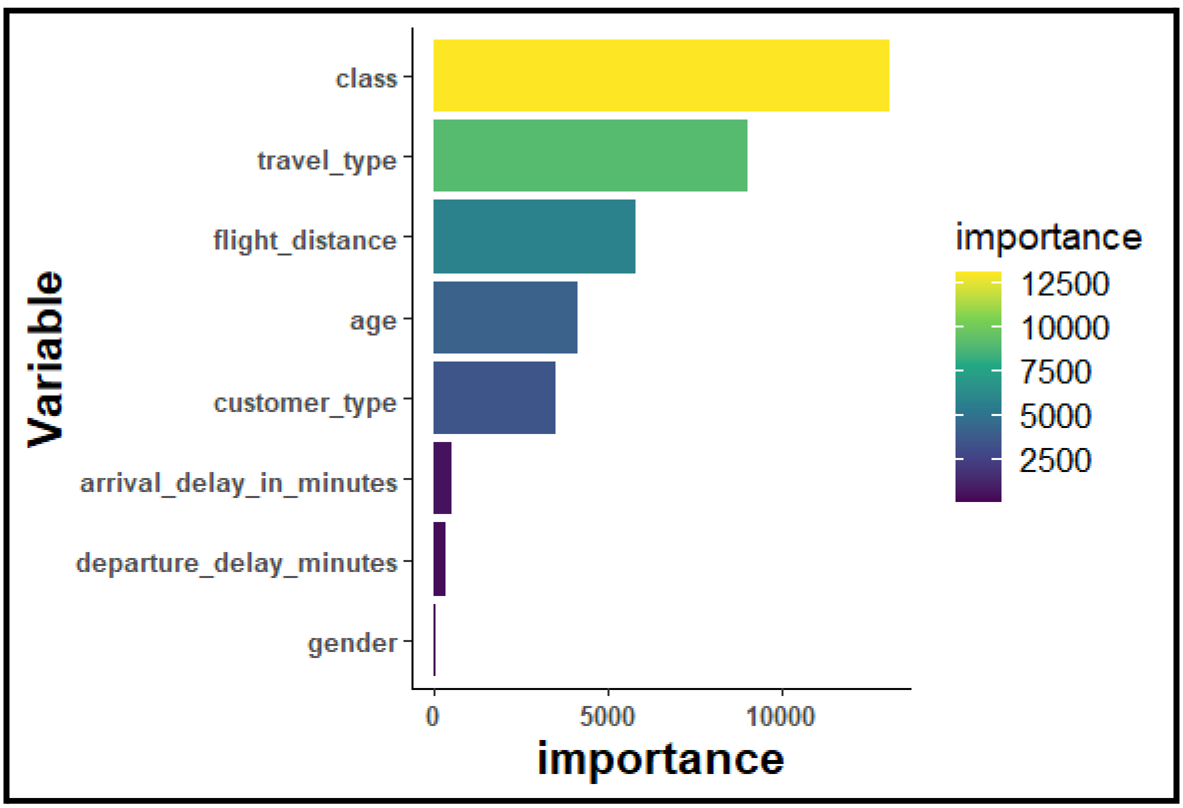


Based on the optimum cp value final tree is built:

```
> final_tree = rpart(satisfaction ~ gender + customer_type + age + travel_type +
+                     class + flight_distance + departure_delay_in_minutes +
+                     arrival_delay_in_minutes,train1, cp = 0.0001220189)
>
```

Variable importance chart in for tree with optimum cp value:

```
> variable = c('class', 'travel_type','flight_distance', 'age','customer_type',
+              'arrival_delay_in_minutes', 'departure_delay_minutes', 'gender')
> importance = c(13059.54573, 9022.73309, 5759.77193, 4088.26411, 3482.01173,
+                507.00337, 307.461, 14.23077)
> vi = data.frame(variable,importance)
> ggplot(vi, aes(x=reorder(variable,importance), y=importance, fill = importance))+
+   scale_fill_viridis_c()+
+   geom_col()+ coord_flip()+xlab('Variable')+ theme_classic()+
+   theme(axis.text = element_text(face = "bold",size = 10),
+         axis.title = element_text(face = "bold",size = 17),
+         legend.title = element_text(size=14),
+         legend.text = element_text(size=13))
```



The above clearly shows the importance of each variable in the decision tree.

Finding optimum cut value:

True positive rate and true negative rate is computed for cut values (0 to1) and absolute difference between each TPR and TNR is calculated. Finally optimum cut value is chosen for which the corresponding absolute difference between each TPR and TNR is minimum.

```
> actual = ifelse(train1$satisfaction == 'satisfied', 1, 0)
> predprob = predict(final_tree, train1, type = 'prob')[,2]
>
> getTPRFPR=function(actual, predProb, cutoff){
+   if(cutoff==1){tpr=0; fpr=0}else{
+     if(cutoff==0){tpr=1; fpr=1}else{
+       predClass=ifelse(predProb<cutoff,0,1); tab=table(actual, predClass)
+       tpr=tab[2,2]/ ( tab[2,2]+tab[2,1] )
+       tnr=tab[1,1]/ ( tab[1,1]+tab[1,2] ); fpr=1-tnr
+     }
+   }
+   return(c(TPR=tpr, TNR=tnr))
+ }
> cutoff_values=seq(0.3,0.7,0.01)->TPR->TNR
> for(i in 1: length(TPR)){
+   metrics=getTPRFPR(train1$satisfaction,predprob,cutoff_values[i])
+   TPR[i]=metrics[1]
+   TNR[i]=metrics[2]
+ }
> cut = data.frame((cbind(cutoff_values, TPR, TNR)))
>
> cut$abs_diff= abs(cut$TPR-cut$TNR)
>
```

```
> cut
   cutoff_values       TPR       TNR     abs_diff
1           0.30 0.8698613 0.7217577 0.1481036630
2           0.31 0.8698613 0.7217577 0.1481036630
3           0.32 0.8698613 0.7217577 0.1481036630
4           0.33 0.8659124 0.7280546 0.1378577608
5           0.34 0.8659124 0.7280546 0.1378577608
6           0.35 0.8659124 0.7280546 0.1378577608
7           0.36 0.8615419 0.7342491 0.1272927279
8           0.37 0.8071880 0.8068430 0.0003450166
9           0.38 0.8065890 0.8076109 0.0010219032
10          0.39 0.8058791 0.8084983 0.0026192031
11          0.40 0.8052801 0.8092150 0.0039349283
12          0.41 0.8039712 0.8106655 0.0066943698
13          0.42 0.8022185 0.8125597 0.0103412023
14          0.43 0.8022185 0.8125597 0.0103412023
15          0.44 0.7952080 0.8195563 0.0243483273
16          0.45 0.7952080 0.8195563 0.0243483273
17          0.46 0.7952080 0.8195563 0.0243483273
18          0.47 0.7952080 0.8195563 0.0243483273
19          0.48 0.7952080 0.8195563 0.0243483273
20          0.49 0.7952080 0.8195563 0.0243483273
```

```
21          0.50 0.7952080 0.8195563 0.0243483273
22          0.51 0.7952080 0.8195563 0.0243483273
23          0.52 0.7952080 0.8195563 0.0243483273
24          0.53 0.7855130 0.8261945 0.0406815054
25          0.54 0.7810094 0.8292321 0.0482226532
26          0.55 0.7790349 0.8304949 0.0514599388
27          0.56 0.7760399 0.8323720 0.0563320802
28          0.57 0.7733777 0.8339590 0.0605813405
29          0.58 0.7722463 0.8345904 0.0623441874
30          0.59 0.7282307 0.8586348 0.1304040857
31          0.60 0.7282307 0.8586348 0.1304040857
32          0.61 0.6837049 0.8807679 0.1970629819
33          0.62 0.6774709 0.8837372 0.2062663195
34          0.63 0.6736994 0.8854778 0.2117784258
35          0.64 0.6686633 0.8876792 0.2190158420
36          0.65 0.6686633 0.8876792 0.2190158420
37          0.66 0.6667332 0.8884471 0.2217138766
38          0.67 0.6667332 0.8884471 0.2217138766
39          0.68 0.6662230 0.8886348 0.2224118506
40          0.69 0.6451692 0.8962287 0.2510595064
41          0.70 0.6451692 0.8962287 0.2510595064
```

```
> cut[cut$abs_diff == min(cut$abs_diff),]
  cutoff_values      TPR      TNR     abs_diff
8          0.37 0.807188 0.806843 0.0003450166
```

Optimum cut value is found to be = 0.37 (decision tree)

Applying optimum cut value in testing dataset:

Confusion matrix is built using optimum cut value

```
> actual = test1$satisfaction
> predicted = predict(final_tree,test1,type='prob')[,2]
> predicted = ifelse(predicted>0.37,'satisfied', 'dissatisfied')
> tab = table(predicted, actual)
> tab
              actual
predicted      dissatisfied satisfied
  dissatisfied        11707      2221
  satisfied            2918      8966
```

- Sensitivity = 8966/ (8966+2221) = 0.801

From the decision tree model using optimum cut value 0.37, sensitivity is found to be 0.801, so 80.1% of satisfied in satisfaction variable can be clearly classified.

- Specificity =   11707/ (11707+2918) = 0.800

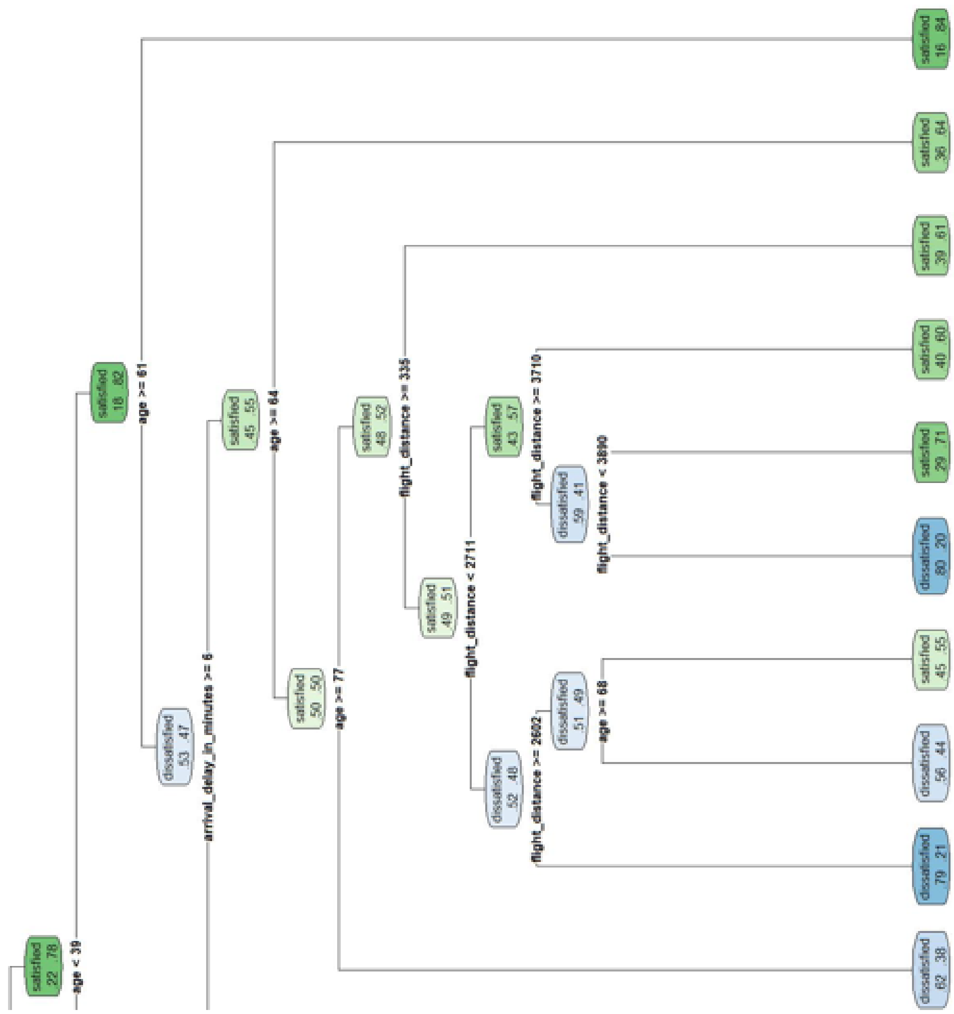
From the decision model using optimum cut value 0.37, specificity is found to be 0.800, so 80% of dissatisfied in satisfaction variable can be clearly classified.

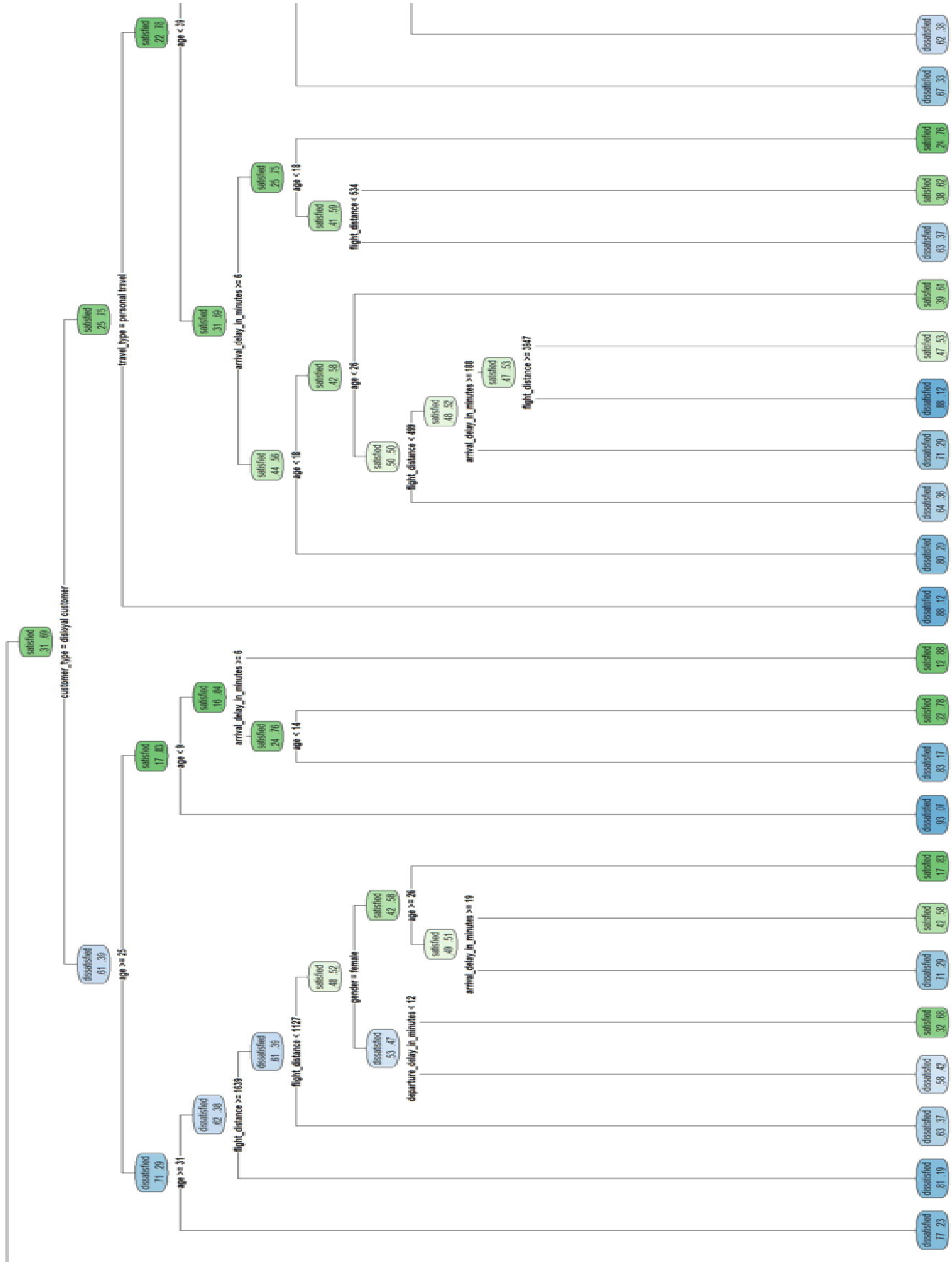
Comparison of two models ie (Logistic Regression and decision tree):

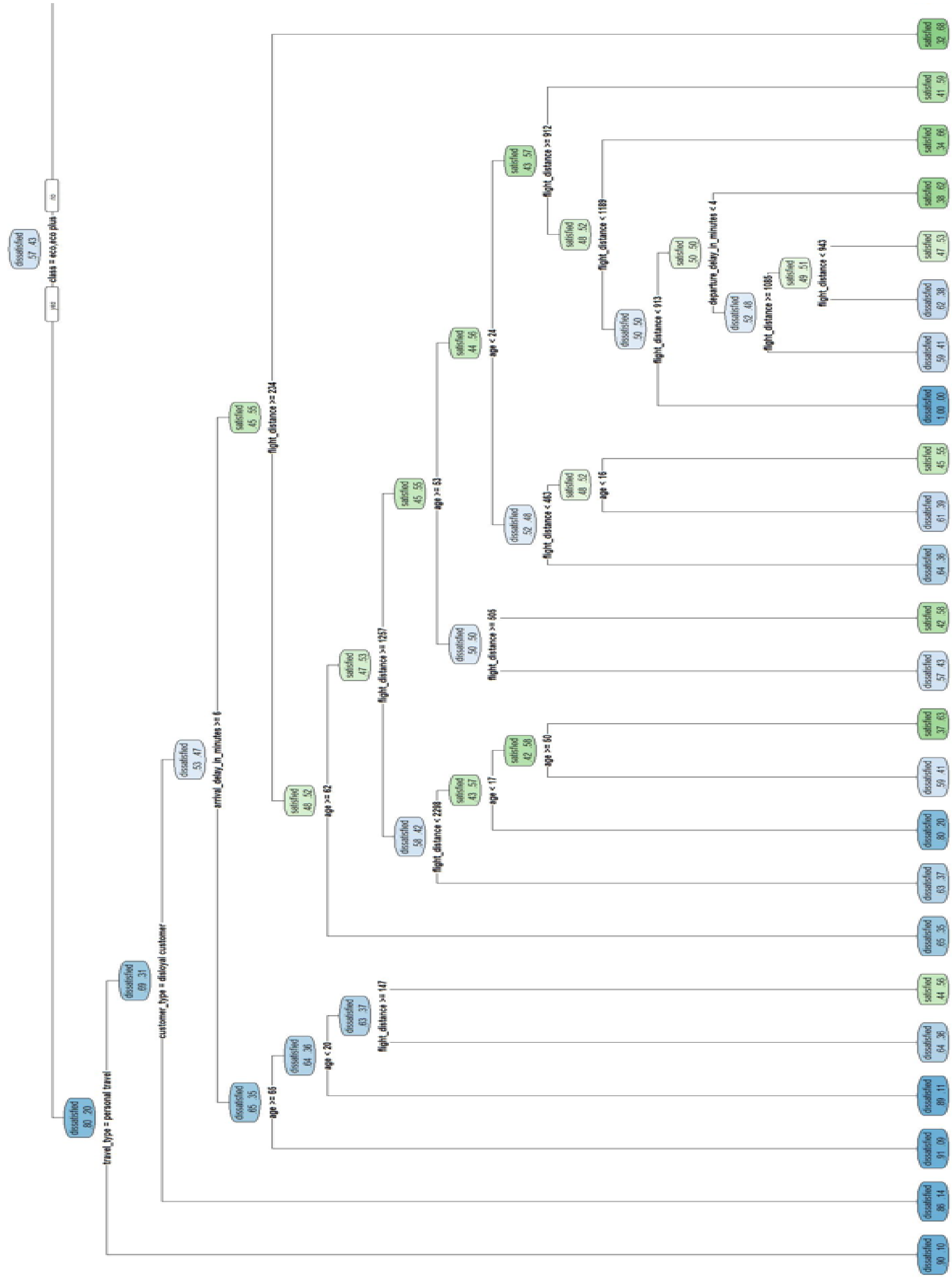
Auc for logistic regression (test dataset) = 0.8414
Auc for decision tree (test dataset) = 0.8544

So comparing the auc of two models it can be said that decision tree is a better model compared to logistic regression in predicting satisfaction of airline passengers.

Decision tree plot:

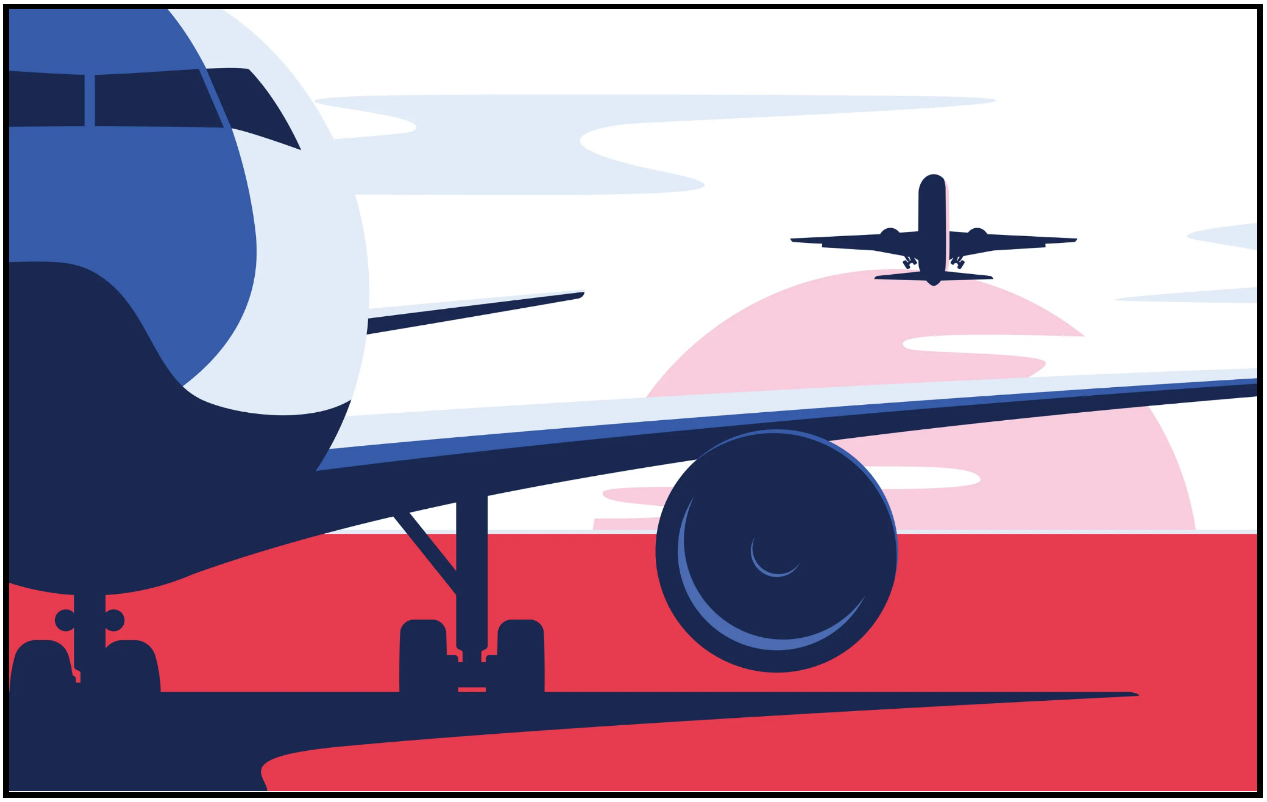



customer_type = disloyal customer
travel_type = personal travel
age >= 25
age < 39
arrival_delay_in_minutes >= 6
age < 18
age < 18
flight_distance < 534
age < 28
flight_distance < 499
arrival_delay_in_minutes >= 108
flight_distance >= 3947
age < 9
arrival_delay_in_minutes >= 6
age < 14
age >= 31
flight_distance >= 1639
flight_distance < 1127
gender = female
age >= 26
arrival_delay_in_minutes >= 19
departure_delay_in_minutes < 12


class = eco,eco plus
yes
no
travel_type = personal travel
customer_type = disloyal customer
arrival_delay_in_minutes >= 6
age >= 66
age < 20
flight_distance >= 147
flight_distance >= 234
age >= 62
flight_distance >= 1257
flight_distance < 2298
age < 17
age >= 60
age >= 53
flight_distance >= 506
age < 24
flight_distance < 463
age < 16
flight_distance >= 912
flight_distance < 1189
flight_distance < 913
departure_delay_in_minutes < 4
flight_distance >= 1086
flight_distance < 943

Part 2

Airline price prediction



Introduction:

Since India being the most populous country in the world, transportation plays an important role in the Indian economy. Among different modes of transportation aviation is a fast growing sector. In upcoming years the majority of the Indian population is going to travel by air as it is fast and efficient . Air travelers in India are looking for an airline which is affordable and with the most satisfying services. Even across the globe people expect the same. Price has a significant role in making a customer determine what airline one has to travel from source to destination.

Objective:

- To construct an Exploratory Data Analysis for the dataset.
- Predict price for per hour travel in an airline from source to destination city.
- Calculate MAPE for multiple regression model

Data collection:

Data obtained from an open sourced data repository.

Defining variable in the data:

Airline	The name of the airline company is stored in the airline column. It is a categorical feature having 6 different airlines
Flight	Flight stores information regarding the plane's flight code. It is a categorical feature.
Source City	City from which the flight takes off. It is a categorical feature having 6 unique cities
Departure Time	This is a derived categorical feature obtained by grouping time periods into bins. It stores information about the departure time and has 6 unique time labels.
Stops	A categorical feature with 3 distinct values that stores the number of stops between the source and destination cities.
Arrival Time	This is a derived categorical feature created by grouping time intervals into bins. It has six distinct time labels and keeps information about the arrival time.

Destination City	City where the flight will land. It is a categorical feature having 6 unique cities.
Class	A categorical feature that contains information on seat class; it has two distinct values: Business and Economy.
Duration	A continuous feature that displays the overall amount of time it takes to travel between cities in hours.
Days Left	This is a derived characteristic that is calculated by subtracting the trip date by the booking date.
Price	Target variable stores information of the ticket price.

Methodology:

1)Multiple Linear Regression:

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression is to model the linear relationship between the explanatory (independent) variables and response (dependent) variables. In essence, multiple regression is the extension of ordinary least-squares (OLS) regression because it involves more than one explanatory variable.

Formula and Calculation of Multiple Linear Regression

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon$$

where, for $i = n$ observations:

y_i = dependent variable

x_i = explanatory variables

β_0 = y-intercept (constant term)

β_p = slope coefficients for each explanatory variable

ϵ = the model's error term (also known as the residuals)

- **Mean Absolute Percentage Error (MAPE)**

MAPE is a statistical measure to define the accuracy of a machine learning algorithm on a particular dataset. MAPE can be considered as a loss function to define the error termed by the model evaluation. Using MAPE, we can estimate the accuracy in terms of the differences in the actual v/s estimated values.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

As seen above, in MAPE, we initially calculate the absolute difference between the Actual Value (A) and the Estimated/Forecast value (F). Further, we apply the mean function on the result to get the MAPE value.MAPE can also be expressed in terms of percentage. Lower the MAPE, better fit is the model.

Data analysis

IMPORTING DATASET INTO R STUDIO:

```
> df = read.csv('C:/Users/aaa/Desktop/bsc project/flightdata.csv',
+               header = T)
```

DATA CLEANING:

Initial columns in the data

```
> colnames(df)
 [1] "X"                "airline"          "flight"           "source_city"
 [5] "departure_time"   "stops"            "arrival_time"     "destination_city"
 [9] "class"            "duration"         "days_left"        "price"
> 
```

Removing columns 'X' and 'flight' which are not necessary for the analysis.

```
> df <- subset(df, select = -X)
> df = subset(df , select = -flight)
> 
```

Checking for null values

```
> sum(is.na(data))
[1] 0
```

Data has no null values

Dimension of the data

```
> nrow(df)
[1] 300153
> ncol(df)
[1] 10
```

Data has 300153 observations and 10 variables

Changing categorical variables data types to factor type

```
> df$airline = as.factor(df$airline)
> df$source_city = as.factor(df$source_city)
> df$departure_time = as.factor(df$departure_time)
> df$stops = as.factor(df$stops)
> df$arrival_time = as.factor(df$arrival_time)
> df$destination_city = as.factor(df$destination_city)
> df$class = as.factor(df$class)
> 
```

Changing integer variables data types to numeric type

```
> df$duration = as.numeric(df$duration)
> df$days_left = as.numeric(df$days_left)
> df$price = as.numeric(df$price)
> 
```

For convenient categorical variables are converted to lowercase

```
> for (i in names(df)){
+   if  (i == 'price'){next}
+   if ( i == 'duration'){next}
+   if (i == 'days_left'){next}
+   df[[i]] = tolower(df[[i]])}
```

Data types of each variable

```
> str(df)
'data.frame':   300153 obs. of  10 variables:
 $ airline         : Factor w/ 6 levels "air_india","airasia",..: 5 5 2 6 6 6 6 6 3 3 ...
 $ source_city     : Factor w/ 6 levels "bangalore","chennai",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ departure_time  : Factor w/ 6 levels "afternoon","early_morning",..: 3 2 2 5 5 5 5 1 2 1 ...
 $ stops           : Factor w/ 3 levels "one","two_or_more",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ arrival_time    : Factor w/ 6 levels "afternoon","early_morning",..: 6 5 2 1 5 1 5 3 5 3 ...
 $ destination_city: Factor w/ 6 levels "bangalore","chennai",..: 6 6 6 6 6 6 6 6 6 6 ...
 $ class           : Factor w/ 2 levels "business","economy": 2 2 2 2 2 2 2 2 2 2 ...
 $ duration        : num  2.17 2.33 2.17 2.25 2.33 2.33 2.08 2.17 2.17 2.25 ...
 $ days_left       : num  1 1 1 1 1 1 1 1 1 1 ...
 $ price           : num  5953 5953 5956 5955 5955 ...
> 
```

Creating a new variable called travel route by combining source and destination city.

```
> df$travel_route = paste(df$source_city, df$destination_city, sep="-")
> head(df['travel_route'])
  travel_route
1 delhi-mumbai
2 delhi-mumbai
3 delhi-mumbai
4 delhi-mumbai
5 delhi-mumbai
6 delhi-mumbai
```

EXPLORATORY DATA ANALYSIS

Displaying first 10 observation from the dataset

```
> head(df, 10)
    airline source_city departure_time stops  arrival_time destination_city   class duration days_left
1  spicejet       delhi        evening  zero         night           mumbai economy     2.17         1
2  spicejet       delhi  early_morning  zero       morning           mumbai economy     2.33         1
3   airasia       delhi  early_morning  zero early_morning           mumbai economy     2.17         1
4   vistara       delhi        morning  zero     afternoon           mumbai economy     2.25         1
5   vistara       delhi        morning  zero       morning           mumbai economy     2.33         1
6   vistara       delhi        morning  zero     afternoon           mumbai economy     2.33         1
7   vistara       delhi        morning  zero       morning           mumbai economy     2.08         1
8   vistara       delhi      afternoon  zero       evening           mumbai economy     2.17         1
9  go_first       delhi  early_morning  zero       morning           mumbai economy     2.17         1
10 go_first       delhi      afternoon  zero       evening           mumbai economy     2.25         1
   price travel_route
1   5953 delhi-mumbai
2   5953 delhi-mumbai
3   5956 delhi-mumbai
4   5955 delhi-mumbai
5   5955 delhi-mumbai
6   5955 delhi-mumbai
7   6060 delhi-mumbai
8   6060 delhi-mumbai
9   5954 delhi-mumbai
10  5954 delhi-mumbai
> 
```

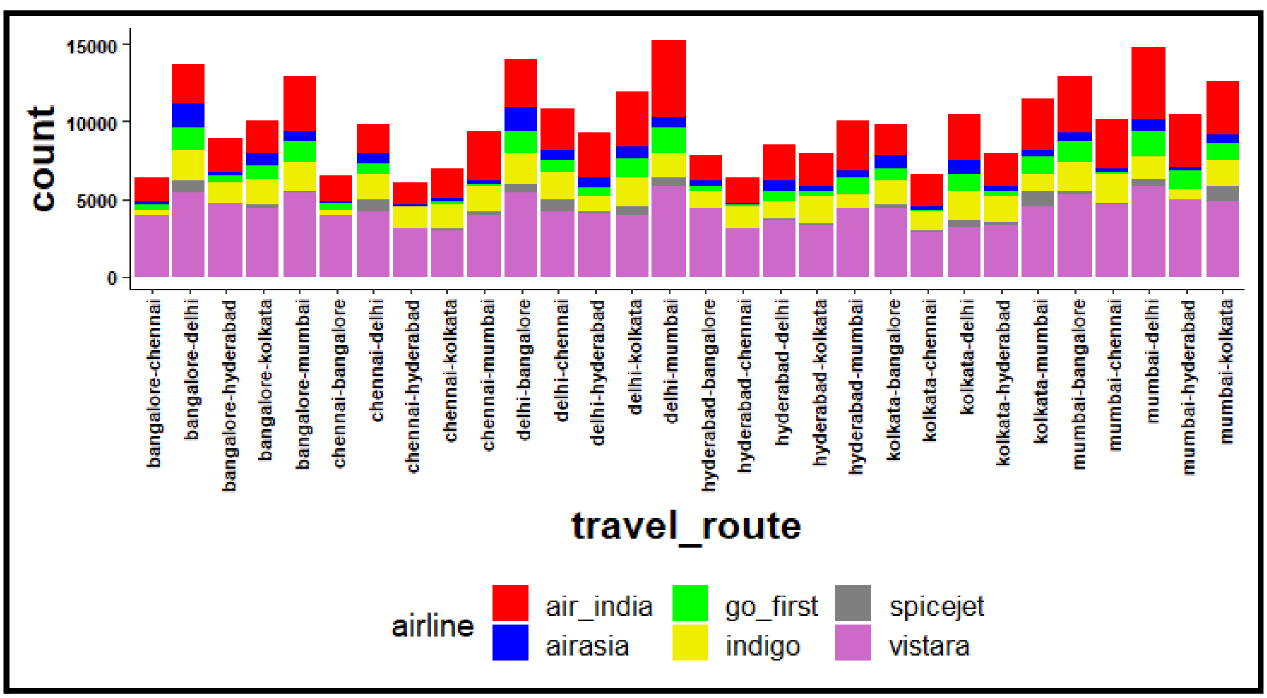
Summary of the data

```
> summary(df)
      airline          source_city           departure_time          stops       
 air_india: 80892   bangalore:52061   afternoon    :47794   one         :250863  
 airasia  : 16098   chennai  :38700   early_morning:66790   two_or_more: 13286  
 go_first : 23173   delhi    :61343   evening      :65102   zero        : 36004  
 indigo   : 43120   hyderabad:40806   late_night   : 1306                        
 spicejet :  9011   kolkata  :46347   morning      :71146                        
 vistara  :127859   mumbai   :60896   night        :48015                        
        arrival_time     destination_city       class           duration        days_left    
 afternoon    :38139   bangalore:51068   business: 93487   Min.   : 0.83   Min.   : 1  
 early_morning:15417   chennai  :40368   economy :206666   1st Qu.: 6.83   1st Qu.:15  
 evening      :78323   delhi    :57360                     Median :11.25   Median :26  
 late_night   :14001   hyderabad:42726                     Mean   :12.22   Mean   :26  
 morning      :62735   kolkata  :49534                     3rd Qu.:16.17   3rd Qu.:38  
 night        :91538   mumbai   :59097                     Max.   :49.83   Max.   :49  
     price         travel_route      
 Min.   :  1105   Length:300153     
 1st Qu.:  4783   Class :character  
 Median :  7425   Mode  :character  
 Mean   : 20890                     
 3rd Qu.: 42521                     
 Max.   :123071                     
```

DATA VISUALIZATION:

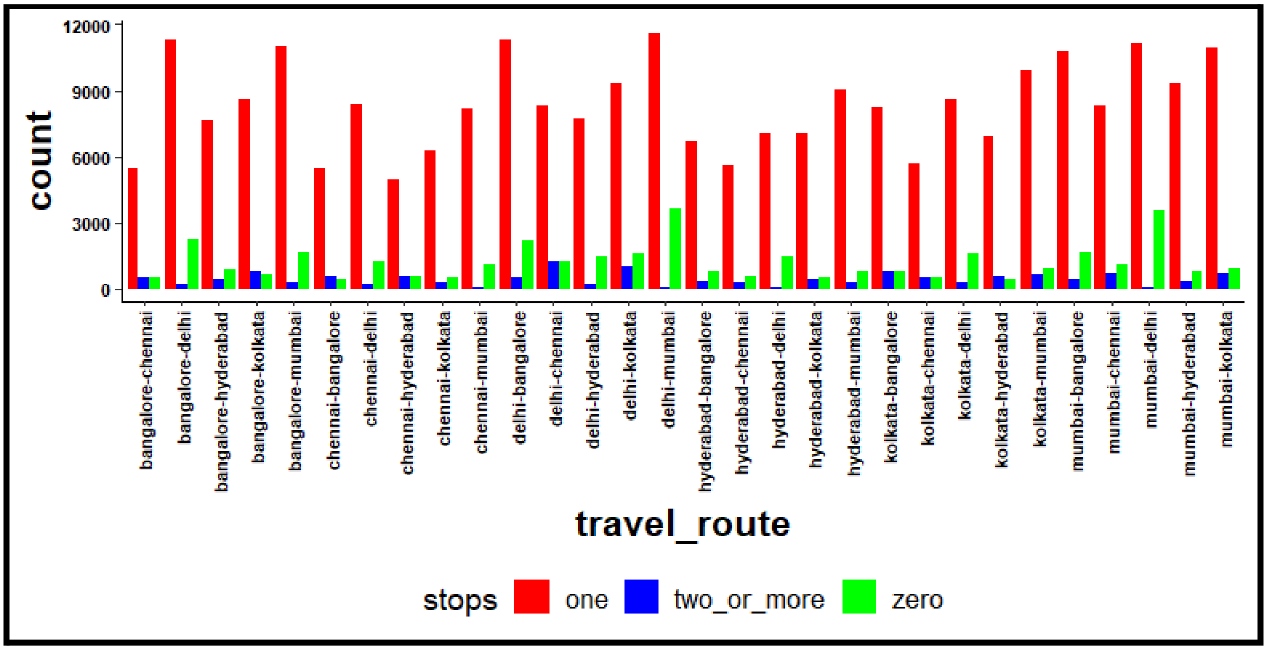
Number of airline services in each route

```
> ggplot(df, aes(x = travel_route, fill = airline))+geom_bar(stat = 'count')+
+   theme_classic()+scale_fill_manual(values=c("red1","blue",'green1', 'yellow2', 'grey50','orch
id3')) +
+   theme(axis.text = element_text(face = 'bold', size = 8, color ='black'),
+         axis.title = element_text(face = "bold",size = 17),
+         legend.title = element_text(size=14),
+         legend.text = element_text(size=13),
+         legend.position = "bottom")+
+   theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
> 
```



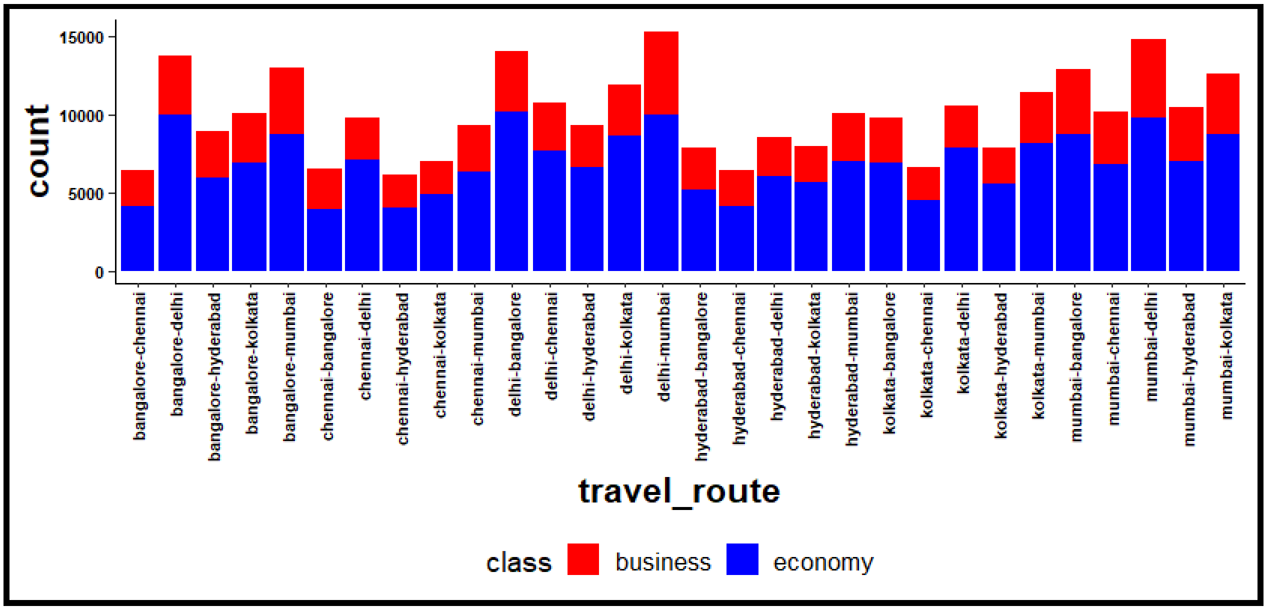
Number of stops a flight takes in each route:

```
> ggplot(df, aes(x = travel_route, fill = stops))+geom_bar(stat = 'count', position = 'dodge')+
+   theme_classic()+scale_fill_manual(values=c("red1","blue",'green1')) +
+   theme(axis.text = element_text(face = 'bold', size = 8, color ='black'),
+         axis.title = element_text(face = "bold",size = 17),
+         legend.title = element_text(size=14),
+         legend.text = element_text(size=13),
+         legend.position = "bottom")+
+   theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
> |
```



Business and economy class passengers traveled in each route

```
> ggplot(df, aes(x = travel_route, fill = class))+geom_bar(stat = 'count')+
+   theme_classic()+scale_fill_manual(values=c("red1","blue")) +
+   theme(axis.text = element_text(face = 'bold', size = 8, color ='black'),
+         axis.title = element_text(face = "bold",size = 17),
+         legend.title = element_text(size=14),
+         legend.text = element_text(size=13),
+         legend.position = "bottom")+
+   theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
> |
```

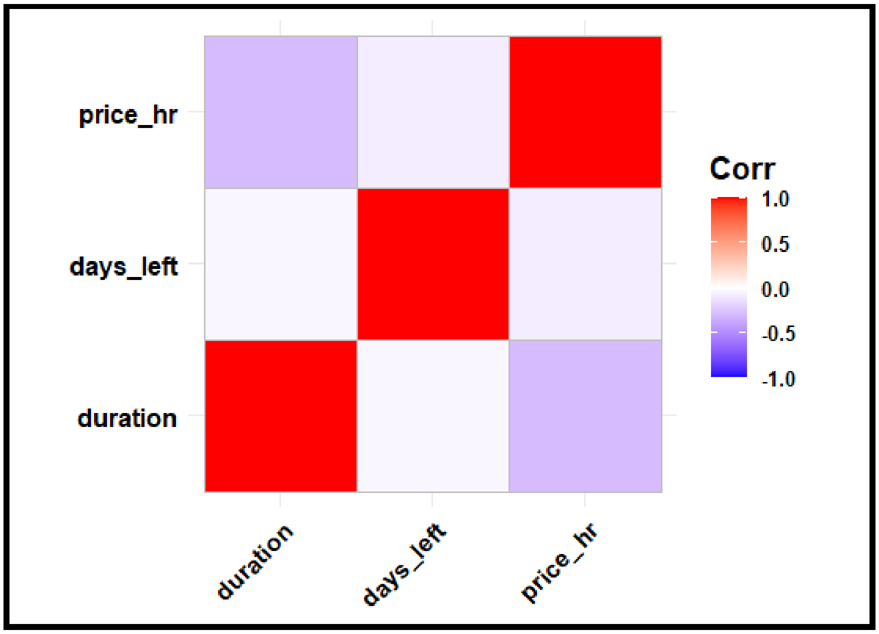


Converting price to price per hour:

```
> df$price_hr = df$price / df$duration
> df = subset(df , select = -price)
> head(df['price_hr'],10)
   price_hr
1  2743.318
2  2554.936
3  2744.700
4  2646.667
5  2555.794
6  2555.794
7  2913.462
8  2792.627
9  2743.779
10 2646.222
>
```

Correlation heatmap for numeric columns:

```
> library(ggcorrplot)
> c = df[c('duration', 'days_left', 'price_hr')]
> ggcorrplot::ggcorrplot(cor(c)) +
+   theme(legend.title=element_text(size=15, face = 'bold'),
+         legend.text=element_text(size=10, face = 'bold'),
+         axis.text = element_text(size=15, face= "bold",
+                                  color = 'black'))
>
```



To take log transformation price per hour column is change to log base e and then price per hour column is removed

```
> for (i in df['price_hr']){
+   df['price_hr_log'] = log(i)}
> df = subset(df , select = -price_hr)
```

```
> head(df['price_hr_log'],10)
   price_hr_log
1      7.916923
2      7.845782
3      7.917427
4      7.881056
5      7.846118
6      7.846118
7      7.977097
8      7.934738
9      7.917091
10     7.880888
>
```

Removing column 'travel_route' which has no further use in analysis.

```
>
> df = subset(df, select = -travel_route)
>
```

MODEL FITTING:

Splitting dataset into training and testing sets:

Training data 80%

Testing data 20%

```
> set.seed(1)
> sample = sample(2,nrow(df), replace = T, prob = c(0.8, 0.2))
> train2 <- df[sample==1, ]
> test2 <- df[sample==2, ]
> dim(train2)
[1] 240048     10
> dim(test2)
[1] 60105    10
> 
```

Multiple linear regression:

Dependent variable : price per hr

Independent variable : airline + source_city + departure timing + stop + arrival timing + departure timing + class + duration +days_left

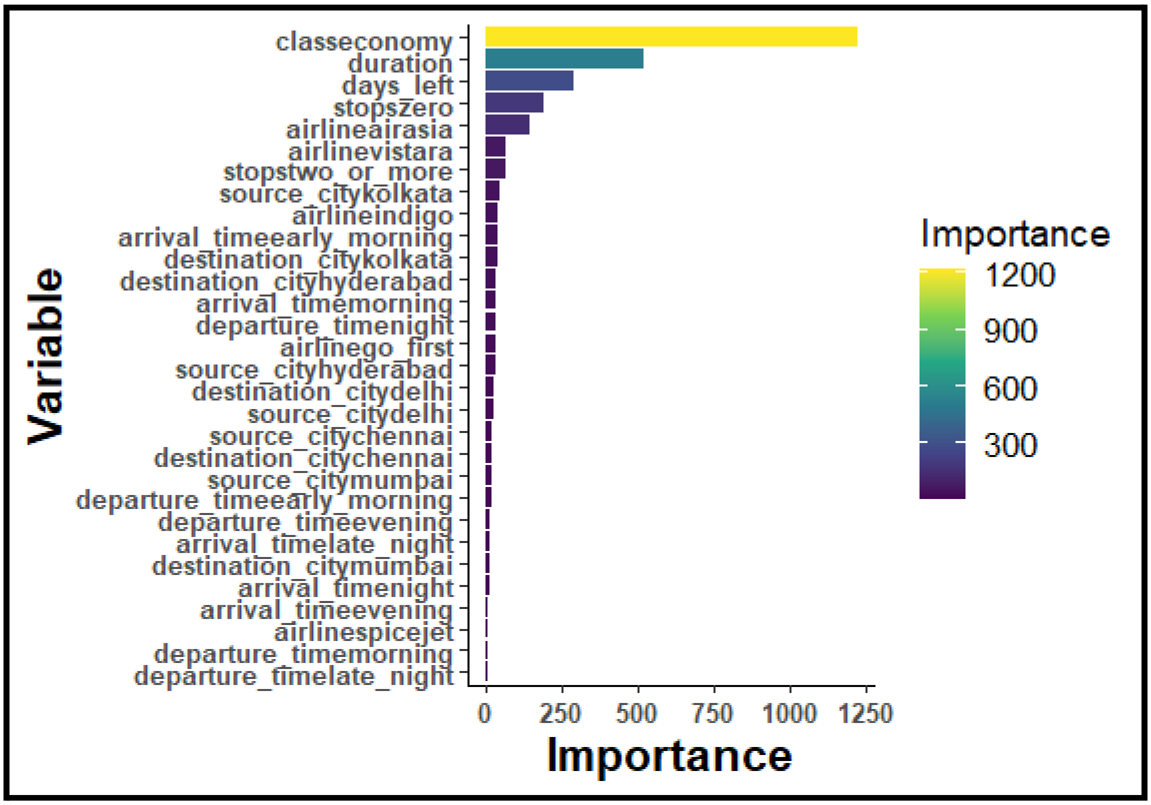
Log transformation is taken for price per hour ie dependent variable

```
> regm = lm(price_hr_log~.,data = train2)
> summary(regm)

Call:
lm(formula = price_hr_log ~ ., data = train2)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.57108 -0.22484 -0.01087  0.20790  2.03836 

Coefficients:
                              Estimate Std. Error   t value Pr(>|t|)    
(Intercept)                  9.655e+00  4.546e-03  2123.960  < 2e-16 ***
airlineairasia              -4.996e-01  3.443e-03  -145.078  < 2e-16 ***
airlinego_first             -9.628e-02  2.988e-03   -32.225  < 2e-16 ***
airlineindigo               -9.971e-02  2.591e-03   -38.489  < 2e-16 ***
airlinespicejet             -2.237e-02  4.218e-03    -5.303 1.14e-07 ***
airlinevistara               1.133e-01  1.705e-03    66.444  < 2e-16 ***
source_citychennai          -5.022e-02  2.537e-03   -19.795  < 2e-16 ***
source_citydelhi            -4.852e-02  2.301e-03   -21.085  < 2e-16 ***
source_cityhyderabad        -7.093e-02  2.514e-03   -28.213  < 2e-16 ***
source_citykolkata           1.059e-01  2.436e-03    43.481  < 2e-16 ***
source_citymumbai           -4.133e-02  2.290e-03   -18.048  < 2e-16 ***
departure_timeearly_morning -3.916e-02  2.269e-03   -17.259  < 2e-16 ***
departure_timeevening       -3.158e-02  2.304e-03   -13.710  < 2e-16 ***
departure_timelate_night    -2.620e-02  1.053e-02    -2.488 0.012834 *  
departure_timemorning       -8.516e-03  2.219e-03    -3.838 0.000124 ***
departure_timenight         -8.134e-02  2.502e-03   -32.515  < 2e-16 ***
stopstwo_or_more             2.114e-01  3.396e-03    62.249  < 2e-16 ***
stopszero                    4.686e-01  2.513e-03   186.459  < 2e-16 ***
arrival_timeearly_morning   -1.356e-01  3.626e-03   -37.396  < 2e-16 ***
arrival_timeevening          1.389e-02  2.346e-03     5.922 3.19e-09 ***
arrival_timelate_night      -4.385e-02  3.822e-03   -11.472  < 2e-16 ***
arrival_timemorning         -8.261e-02  2.467e-03   -33.486  < 2e-16 ***
arrival_timenight           -2.222e-02  2.299e-03    -9.667  < 2e-16 ***
destination_citychennai     -4.685e-02  2.516e-03   -18.624  < 2e-16 ***
destination_citydelhi       -5.039e-02  2.362e-03   -21.338  < 2e-16 ***
destination_cityhyderabad   -8.437e-02  2.492e-03   -33.856  < 2e-16 ***
destination_citykolkata      8.547e-02  2.407e-03    35.511  < 2e-16 ***
destination_citymumbai      -2.446e-02  2.321e-03   -10.535  < 2e-16 ***
classeconomy                -2.012e+00  1.650e-03 -1219.186  < 2e-16 ***
duration                    -6.667e-02  1.284e-04  -519.242  < 2e-16 ***
days_left                   -1.440e-02  4.992e-05  -288.453  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.331 on 240017 degrees of freedom
Multiple R-squared:  0.9136,	Adjusted R-squared:  0.9135 
F-statistic: 8.455e+04 on 30 and 240017 DF,  p-value: < 2.2e-16
```

All variables in the model are significant at 5% significance level.

Variable importance chart:

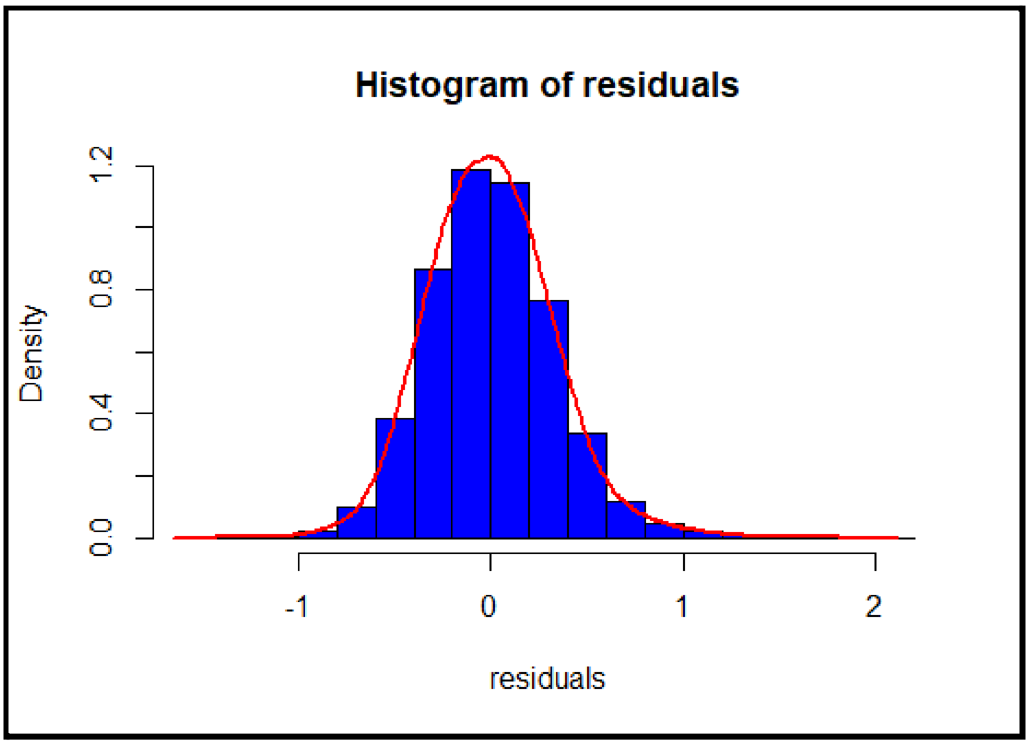
```
> library(caret)
> V = caret::varImp(regm)
> ggplot2::ggplot(V, aes(x=reorder(rownames(V),Overall), y=Overall, fill = Overall))+
+   scale_fill_viridis_c('Importance')+ylab('Importance')+
+   geom_col()+ coord_flip()+xlab('Variable')+ theme_classic()+
+   theme(axis.text = element_text(face = "bold",size = 10),
+         axis.title = element_text(face = "bold",size = 17),
+         legend.title = element_text(size=14),
+         legend.text = element_text(size=13))
> |
```



The above plot clearly shows the the importance of each variable in multiple regression model

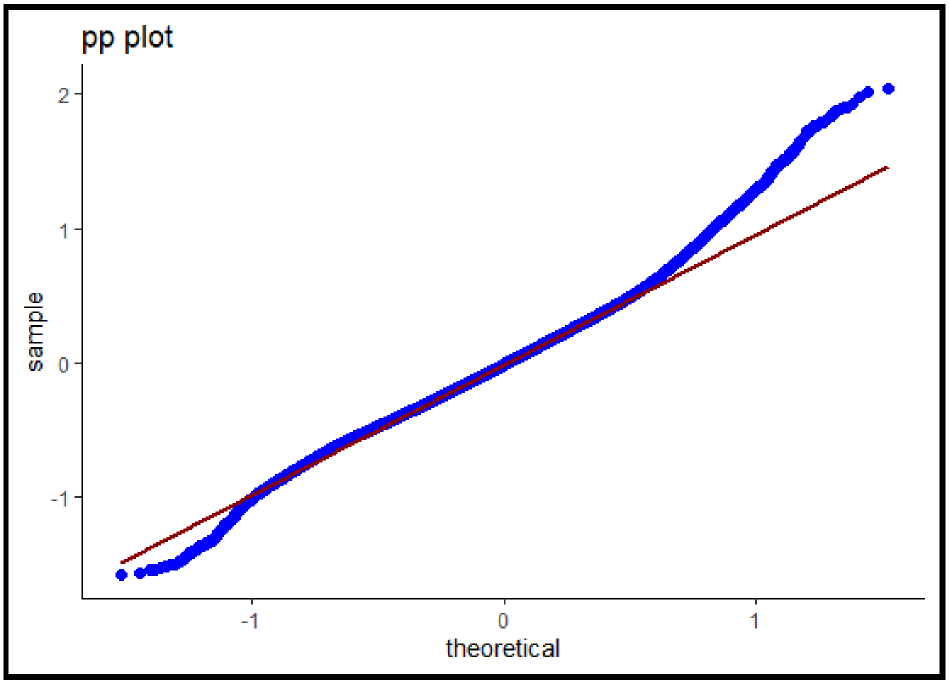
Residual plot:

```
> hist(regm$residuals,
+      col="blue",
+      border="black",
+      prob = TRUE,
+      xlab = "residuals",
+      main = "Histogram of residuals")
> lines(density(regm$residuals),
+       lwd = 2,
+       col = "red")
> 
```



PP plot for residuals:

```
> library(ggplot2)
> library(qqplotr)
> ggplot(mapping = aes(sample = regm$residuals)) +
+   stat_qq_point(size = 2,color = "blue")+theme_classic()+
+   stat_qq_line(color="darkred")+xlab('theoretical')+
+   ylab('sample')+ggtitle('pp plot')
> 
```



KS-test for residuals

```
> ks.test(regm$residuals, 'pnorm')

        One-sample Kolmogorov-Smirnov test

data:  regm$residuals
D = 0.25546, p-value < 2.2e-16
alternative hypothesis: two-sided
```

Since p value (2.2e-16)< 0.05. At 5% significant it can be said that residuals do not follow a normal distribution.

Mean absolute percentage error:

MAPE for training dataset:

```
> predm = predict(regm, train2)
> ap = data.frame(predm)
>
> for (i in train2['price_hr_log']){
+   ap['actual price per hr'] = exp(i)}
>
> for (i in ap['predm']){
+   ap['predicted price per hr'] = exp(i)}
>
> ap = subset(ap, select = -predm)
> print(head(ap,10))
   actual price per hr predicted price per hr
1             2743.318               2449.093
2             2554.936               2263.897
3             2744.700               1346.612
5             2555.794               2673.447
8             2792.627               3001.326
9             2743.779               2125.150
10            2646.222               2420.982
11            2646.222               2420.982
12            2555.365               2354.742
13            2744.240               2117.867
>
> mean(abs((ap[,1]-ap[,2])/ap[,1])) * 100
[1] 26.5587
```

MAPE for testing dataset:

```
> predm = predict(regm, test2)
> ap = data.frame(predm)
>
>
> for (i in test2['price_hr_log']){
+   ap['actual price per hr'] = exp(i)}
>
> for (i in ap['predm']){
+   ap['predicted price per hr'] = exp(i)}
>
> ap = subset(ap, select = -predm)
> print(head(ap,10))
   actual price per hr predicted price per hr
4            2646.6667              2919.2080
6            2555.7940              2903.6800
7            2913.4615              2718.3784
18           2744.2396              2339.9335
21            506.7234               773.4219
29           4403.4335              2423.1077
35            675.0000               593.6886
41           1104.5455               779.3238
52            583.8202               395.9279
61            687.6923               504.7739
>
> mean(abs((ap[,1]-ap[,2])/ap[,1])) * 100
[1] 26.73932
```

CONCLUSION:

For both training and testing datasets the mean absolute percentage error is approximately 26%. So that the expected average percentage error in the fitted multiple linear model is 26.6%.