|  |
| --- |
|  |
| Cab Fare Prediction Model |
| Data Science Project (Edwisor) |

|  |
| --- |
| Riddhesh Sajwan  February 26, 2020 |

Contents

[Introduction 2](#_Toc34873731)

[1. Problem Statement 2](#_Toc34873732)

[2. Data 2](#_Toc34873733)

[Methodology 4](#_Toc34873734)

[1. Pre-Processing 4](#_Toc34873735)

[a. Missing Data Analysis 4](#_Toc34873736)

[b. Outlier Analysis 6](#_Toc34873737)

[c. Feature Selection 10](#_Toc34873738)

[d. Encoding Categorical Data 10](#_Toc34873739)

[e. Rest of the Pre-Processing 10](#_Toc34873740)

[2. Data Visualization 11](#_Toc34873741)

[a. Probability Density Function with Histogram and Normal Fit 11](#_Toc34873742)

[b. Histogram and Mean of Predictors 12](#_Toc34873743)

[Modelling 15](#_Toc34873744)

[1. Training the Model 15](#_Toc34873745)

[Multiple Linear Regression 15](#_Toc34873746)

[Random Forest 16](#_Toc34873747)

[2. Conclusion 18](#_Toc34873748)

[1. Model Evaluation 18](#_Toc34873749)

[Prediction 19](#_Toc34873750)

[1. Model Testing 19](#_Toc34873751)

[Appendix 21](#_Toc34873752)

[1. R Code 21](#_Toc34873753)

[a. missingValueAnalysis.R 21](#_Toc34873754)

[b. outlierAnalysis.R 22](#_Toc34873755)

[c. dataVisualization.R 23](#_Toc34873756)

[d. mlr\_rf.R 23](#_Toc34873757)

[e. predict.R 24](#_Toc34873758)

[2. Instructions to Run the Code 24](#_Toc34873759)

[References 25](#_Toc34873760)

# Introduction

## 1. Problem Statement

We have to make a prediction model for a cab renting company. This prediction model should be able to predict the amount of fare on a particular ride given the source, destination and count of passengers. We are provided with a training dataset which has date of journey, source, destination and count of passengers. Also, we are given a test dataset with the same parameters to run our prediction model on for calculation of accuracy.

## 2. Data

Following is a sample of data provided to us to make our prediction model. We will preprocess the data provided to avoid any missing values or outliers.

##### Table 1.2.1 Training Data (Rows 1-13 and Columns 1-4)

|  |  |  |  |
| --- | --- | --- | --- |
| fare\_amount | pickup\_datetime | pickup\_longitude | pickup\_latitude |
| 4.5 | 2009-06-15 17:26:21 UTC | -73.844311 | 40.721319 |
| 16.9 | 2010-01-05 16:52:16 UTC | -74.016048 | 40.711303 |
| 5.7 | 2011-08-18 00:35:00 UTC | -73.982738 | 40.76127 |
| 7.7 | 2012-04-21 04:30:42 UTC | -73.98713 | 40.733143 |
| 5.3 | 2010-03-09 07:51:00 UTC | -73.968095 | 40.768008 |
| 12.1 | 2011-01-06 09:50:45 UTC | -74.000964 | 40.73163 |
| 7.5 | 2012-11-20 20:35:00 UTC | -73.980002 | 40.751662 |
| 16.5 | 2012-01-04 17:22:00 UTC | -73.9513 | 40.774138 |
|  | 2012-12-03 13:10:00 UTC | -74.0065 | 40.72671 |
| 8.9 | 2009-09-02 01:11:00 UTC | -73.9807 | 40.73387 |
| 5.3 | 2012-04-08 07:30:50 UTC | -73.9963 | 40.73714 |
| 5.5 | 2012-12-24 11:24:00 UTC | 0 | 0 |

##### Table 1.2.2 Training Data (Rows 1-13 and Columns 5-7)

|  |  |  |
| --- | --- | --- |
| dropoff\_longitude | dropoff\_latitude | passenger\_count |
| -73.84161 | 40.712278 | 1 |
| -73.979268 | 40.782004 | 1 |
| -73.991242 | 40.750562 | 2 |
| -73.991567 | 40.758092 | 1 |
| -73.956655 | 40.783762 | 1 |
| -73.972892 | 40.758233 | 1 |
| -73.973802 | 40.764842 | 1 |
| -73.990095 | 40.751048 | 1 |
| -73.9931 | 40.73163 | 1 |
| -73.9915 | 40.75814 | 2 |
| -73.9807 | 40.73356 | 1 |
| 0 | 0 | 3 |

As we can observe from the data that below are the variables that we will be using as predictors for our model:

##### Table 1.2.3 Predictors

|  |  |
| --- | --- |
| Sr. No. | Predictor |
| 1 | pickup\_longitude |
| 2 | pickup\_latitude |
| 3 | dropoff\_longitude |
| 4 | dropoff\_latitude |
| 5 | passenger\_count |

# Methodology

## 1. Pre-Processing

Data pre-processing is an important initial step in any data mining, data science or machine learning project. In simple terms, it is the process of converting raw data into a much readable, understandable format. The raw data usually received is always incomplete or has errors, hence data pre-processing is done to make sure that complete and correct data is sent to the deployment model to get a flawless and more accurate prediction result. The following steps take place in the process of pre-processing:

1. Missing Data Analysis
2. Outlier Analysis
3. Encoding Categorical Data
4. Standardization of Data
5. Dimensionality Reduction
6. Splitting of Data Set

### a. Missing Data Analysis

Missing Data is a common problem with databases. This situation usually arises due to technical glitches while collecting data or people refusing/forgetting to fill data completely while the time of data collection. Missing data analysis is a technique used to find out the presence of missing values in a dataset and treat this situation by either removing that observation or imputing values into that observation using various methods like mean, median, mode, KNN imputation etc.

Observing the above sample data, we can see that the dataset provided to us has missing values. The best way to figure out what to do with this data is by calculating the percentage of missing data in all the variables.

We will be going by the following two rules: -

1. If the percentage of missing data is less than 5% then we can eliminate the observations as their absence won’t affect the model much.
2. If the percentage of missing data is more than 30% then won’t consider imputation as it might change the entire outcome of the model.

We will be using R and python to perform missing value analysis.

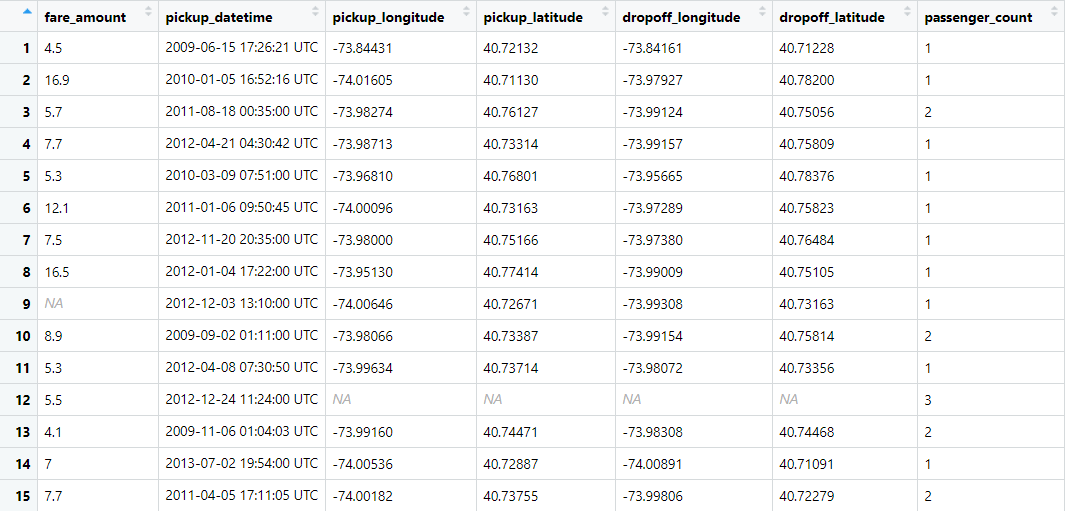
We have imported the csv file provided to us with the condition that all blank values, extra spaces and zeroes will be replaced by NA into a dataframe.

R code:

ds <- read.csv("C:/Users/riddh/OneDrive/Documents/Edwisor/Cab Fare Prediction/train\_cab.csv", sep = ",", header = T, na.strings = c(" ", "", "NA","0"))

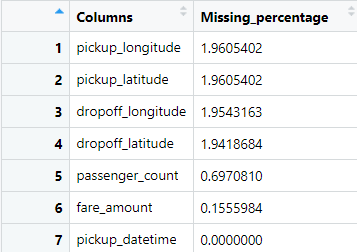
Dataframe Output:

##### Fig 2.1.a.1 Dataframe of Training Data



Now, we will calculate the missing value percentage of every column using R. We get the following dataframe:

##### Fig 2.1.a.2 Dataframe of Missing Value Percentage



As we can observe from the above figure, the missing data percentage of all the variables is less than 5% (rather 2%), so it is quite safe to assume that they don’t contribute as much in the outcome, hence it is only sensible to eliminate these observations.

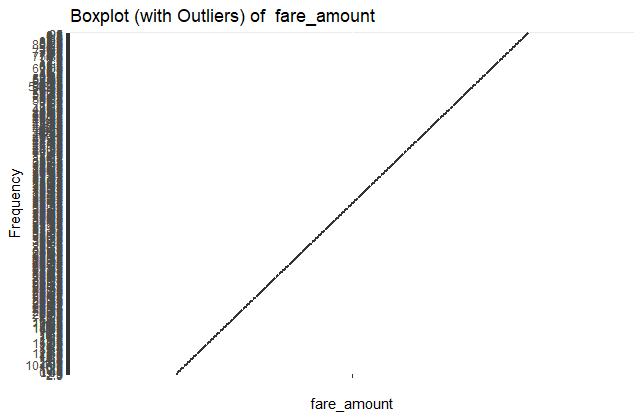
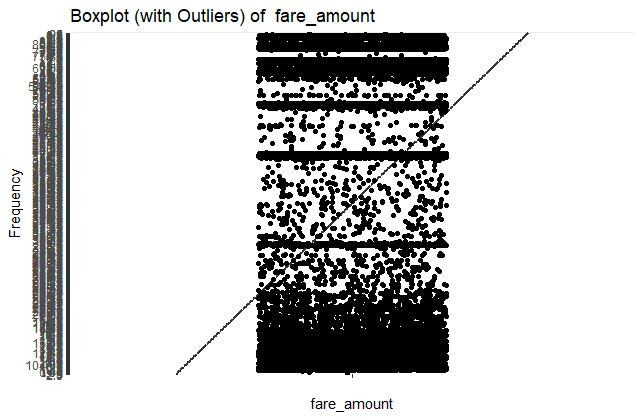
After performing the omission of these observations, we move onto the next step in pre-processing.

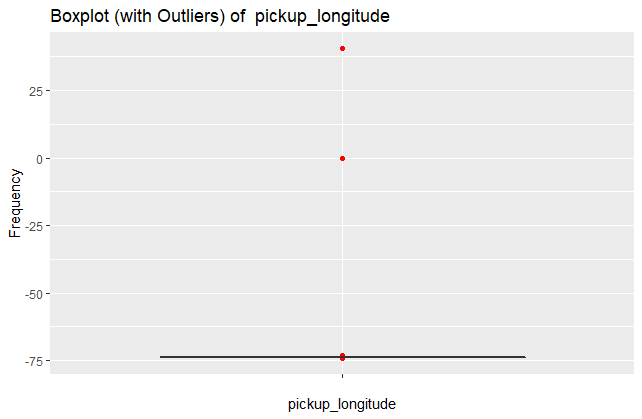
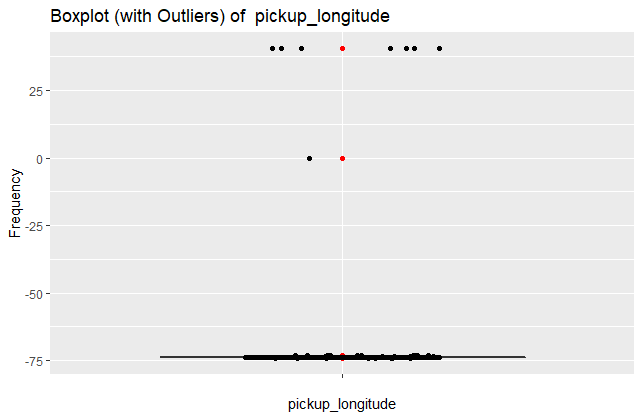
### b. Outlier Analysis

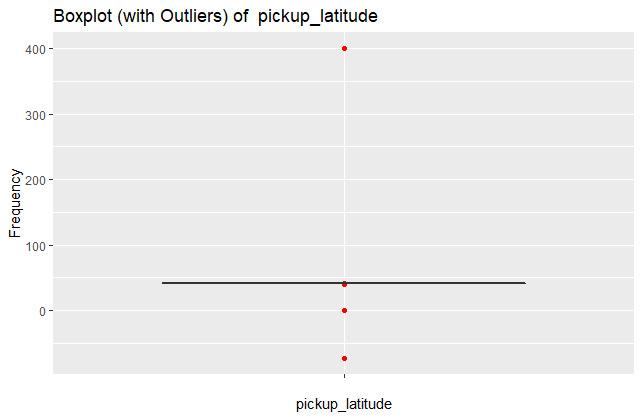
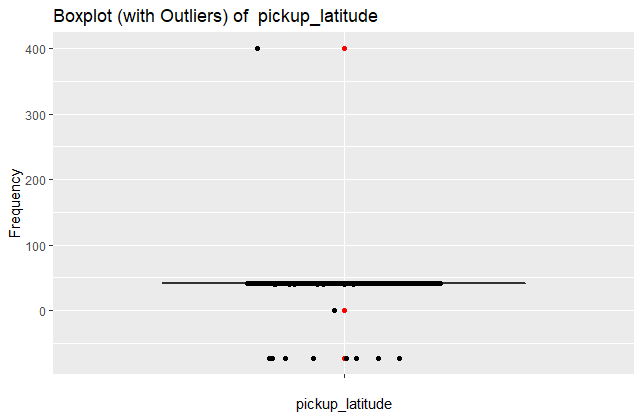
“Observation which deviates so much from other observations as to arouse suspicion it was generated by a different mechanism” — Hawkins(1980)

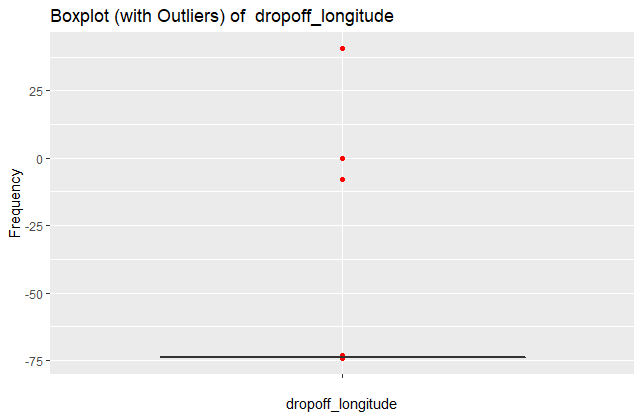
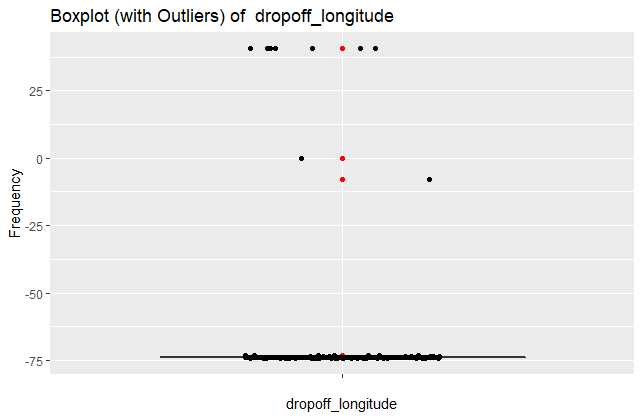
We have clearly observed from our visualizations that some predictors like *windspeed* had a few data points that weren’t as usual and hence they skewed the entire data outcome. We also proved the effects of outliers on the data using visualizations. The histograms and boxplots in the [Fig 2.1.b.1](#_Fig_2.1.b.1.1_Boxplot(with) precisely depict how the outliers affected the outcome.

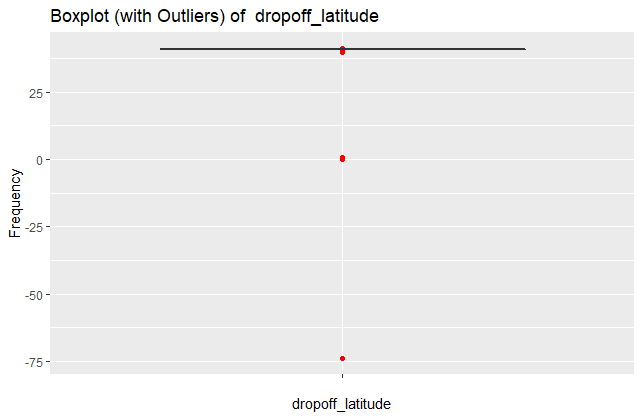
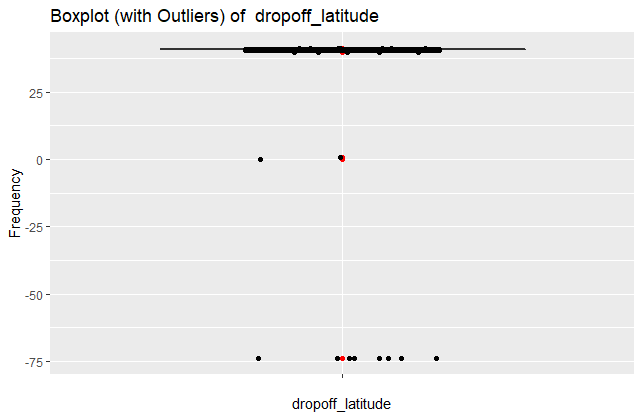
#### b.1 Boxplot with Outliers

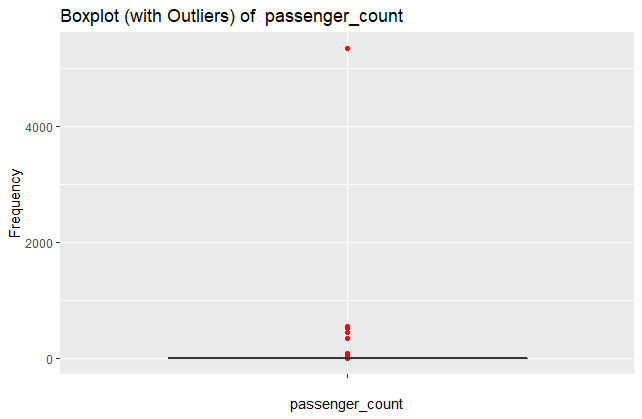
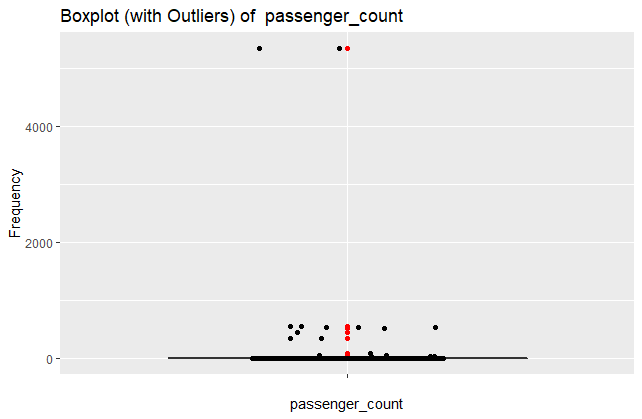










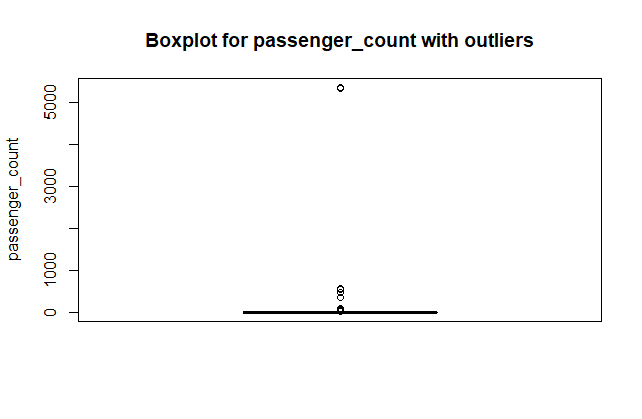
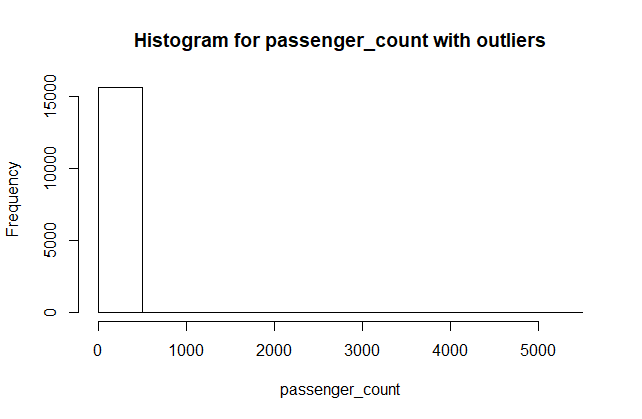


##### Fig 2.1.b.1 Boxplot (with outliers)

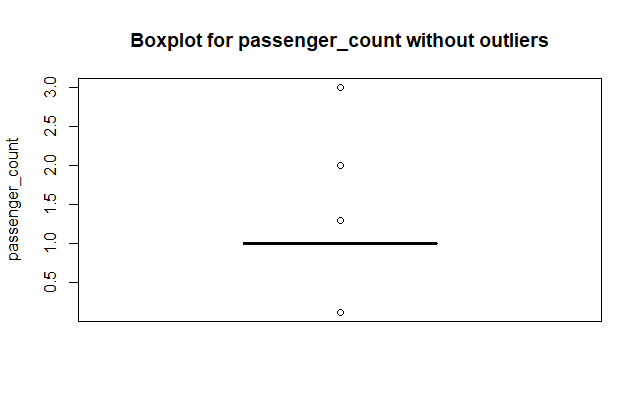
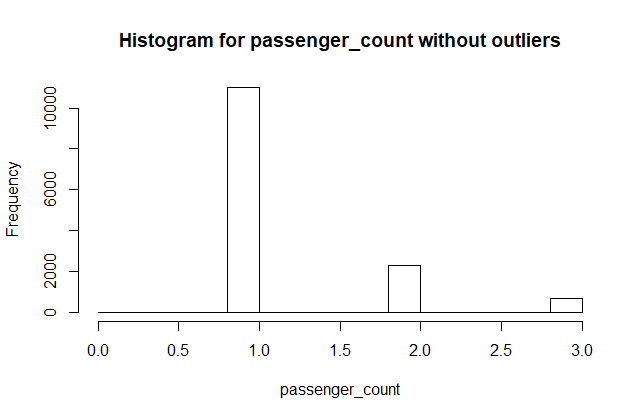
The red dots observed in the above diagrams are outliers and their presence can actually affect the graphs. In the above graphs, it is quite clear that the predictor *windspeed* has a lot of outliers and removing them can totally help the outcome.   
  
Hence, below [Fig 2.1.b.2.b](#_Fig_2.1.b.2.b_Boxplots) is a representation of boxplots after the removal of outliers from *passenger\_count* and refer [Fig 2.1.b.2.a](#_Fig_2.1.b.2.a_with) to see the effect of outliers on data, depicted by the use of boxplot and histogram.

#### b.2 Effect of Outliers

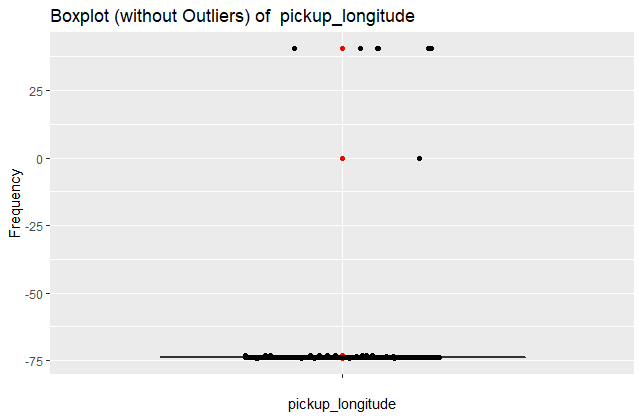
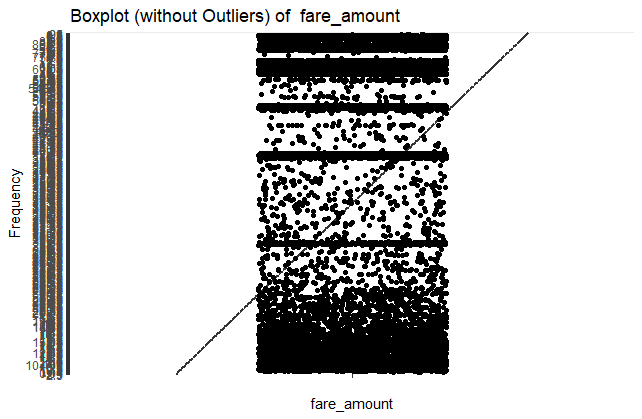
##### Fig 2.1.b.2.a with outliers

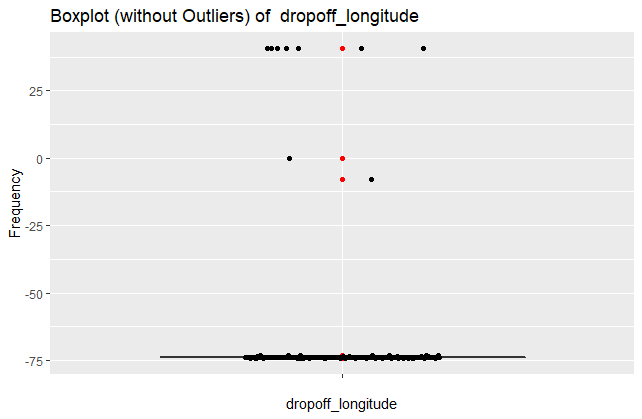
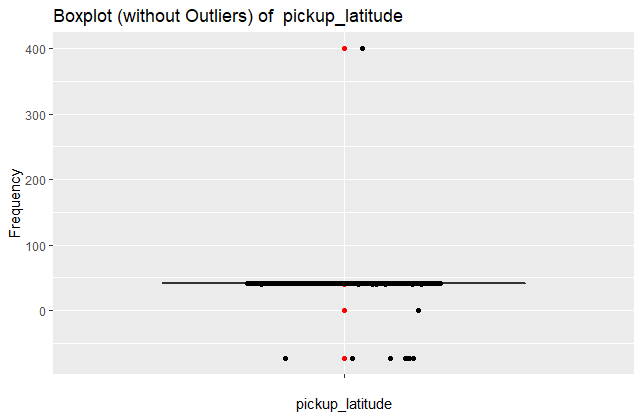


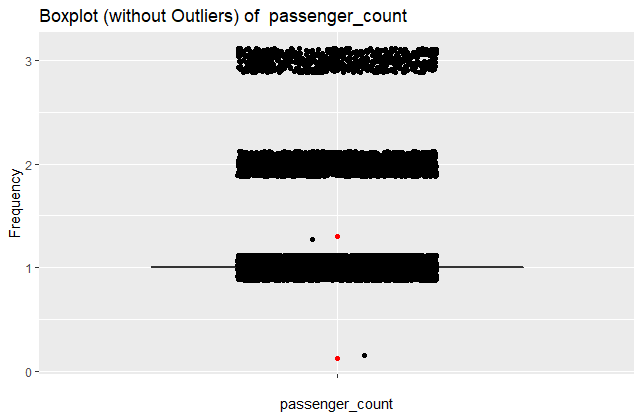
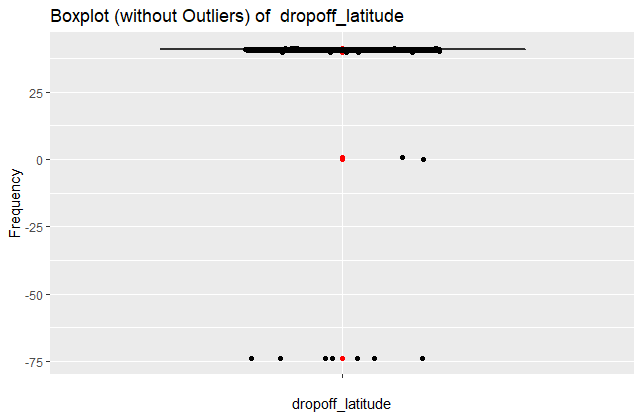
##### Fig 2.1.b.2.a without outliers



#### b.3 Boxplots After Removal of Outliers







##### Fig 2.1.b.2.b Boxplots without outliers

### c. Feature Selection

This is a technique used to make our dataset contain less but more contributing predictors. Given, we have a long list of predictors, it is quite possible that some of them are dependent on each other and hence do not contribute significantly in the final outcome. The process of discovering and eliminating such predictors is called feature selection. The features selected after this process become the part of our final dataset which is passed through our prediction model.

##### Table 2.1.c.1 Continuous Variables/Predictors

|  |  |
| --- | --- |
| Sr. No. | Predictor |
| 1 | pickup\_longitude |
| 2 | pickup\_latitude |
| 3 | dropoff\_longitude |
| 4 | dropoff\_latitude |
| 5 | passenger\_count |

As we can observe from the above data, the number of predictors are quite less and hence going for dimensionality reduction won’t be as beneficial. Also, in this case, where the number of predictors is already less and all have equal importance (pick-up and drop locations and passenger count are equally important for cab fare prediction) we should not be considering feature selection as it may lead to loss of data.

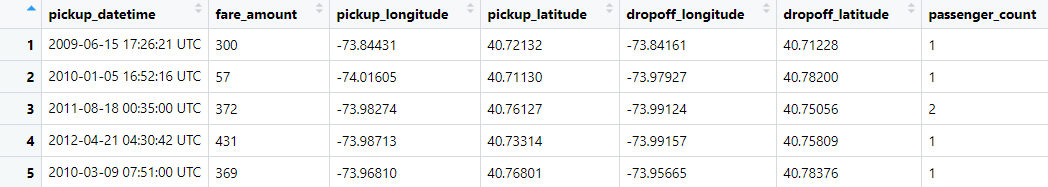
### d. Encoding Categorical Data

It is very important to encode categorical data into numbers, but as we can observe from the data, all our variables are already numeric. Hence, the question arises, are all of them continuous. So, we run the following line of code in R to convert any categorical data to continuous data.

for (i in 2:7){if(class(ds\_new[,i]) == "factor"){ds\_new[,i] <- as.numeric(ds\_new[,i])}}

We found out that *fare\_amount* was turned into continuous by this code.

##### Fig 2.1.d.1 Categorical to Continuous



### e. Rest of the Pre-Processing

As it is quite evident from the data that we don’t need the rest of the pre-processing techniques in our model. We rejected dimensionality reduction in the above section c and standardization doesn’t seem necessary at all for our data. Apart from that, the data provided to us was already divided into training and testing set, so splitting of data is also not required.

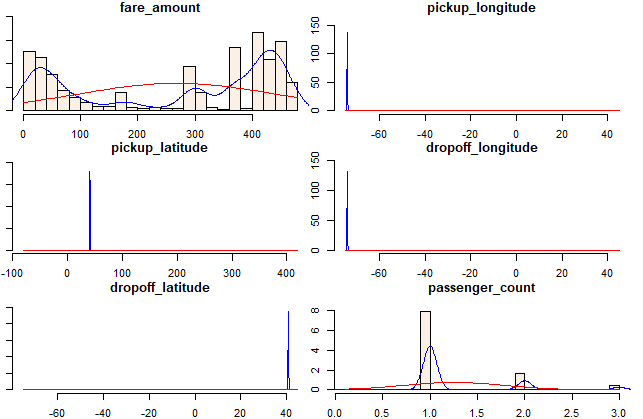
Let’s move onto the next section, where we perform data visualization on our data set.

## 2. Data Visualization

This is a very important aspect for any data science project as this step enables the engineer to observe the data carried by the predictors in a much more informative way. In data visualization we use various graphical representations to showcase the observations we have with us, which helps us to understand the future steps a bit more. We have the following kind of graphs in our report: (R code in Appendix)

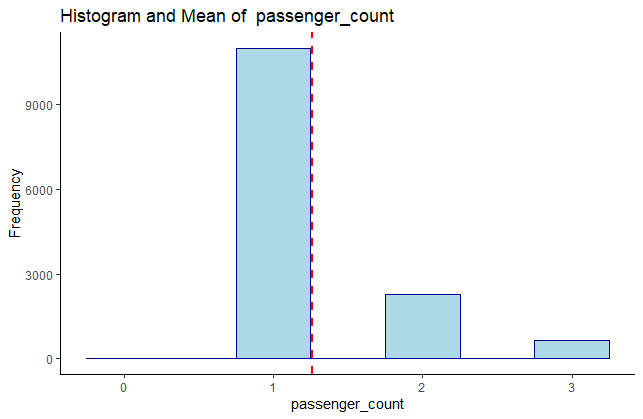
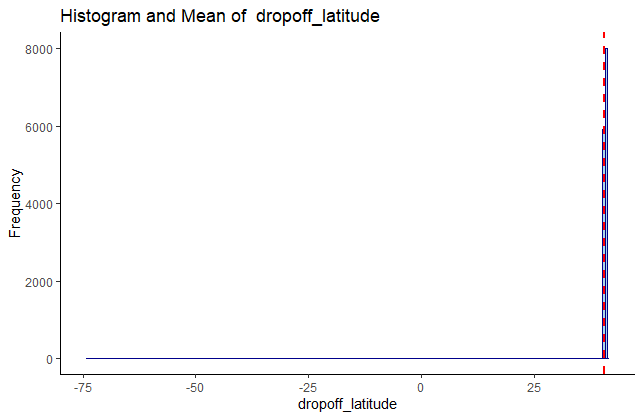
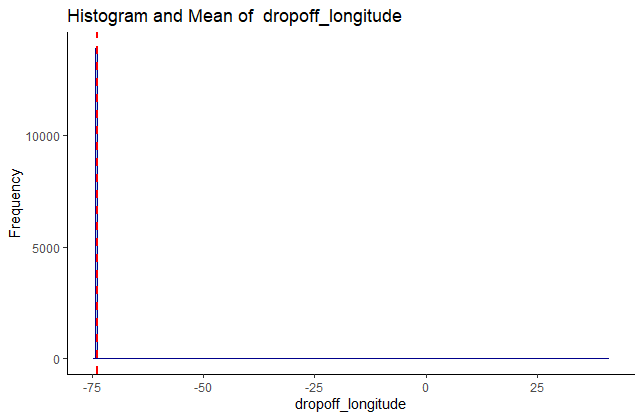
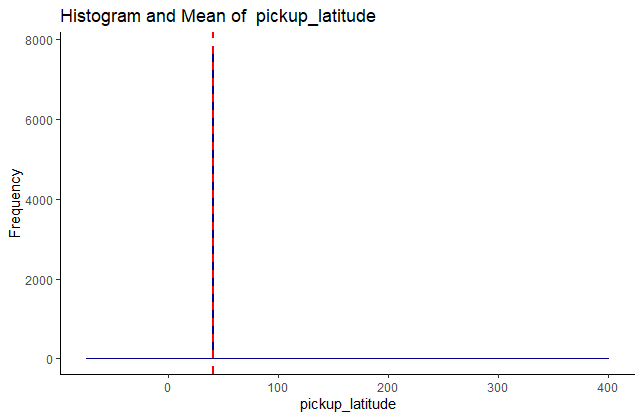
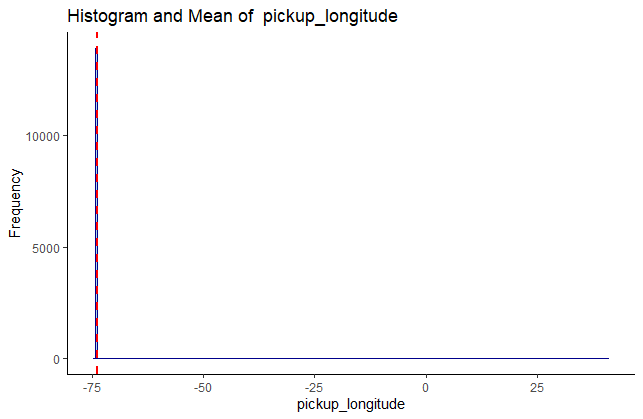
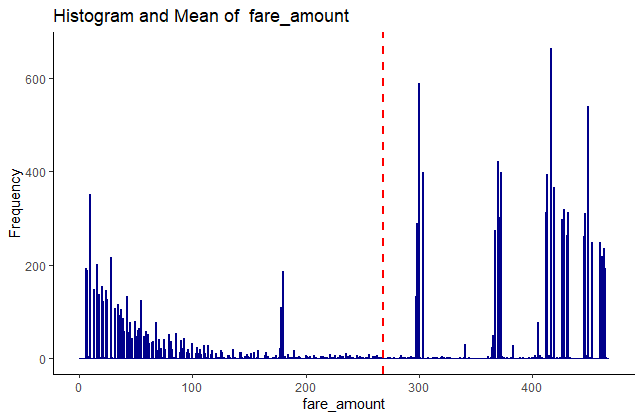
1. Probability Density Function with Histogram and normal fit. ([Fig 2.2.a.1](#_Fig_2.2.a.1_PDF))
2. Histogram and Mean of Predictors ([Fig 2.2.b.1](#_Fig_2.2.b.1_Histogram))
3. Boxplots for each Predictor (Before [Fig 2.1.b.1](#_Fig_2.1.b.1_Boxplot) and After [Fig 2.1.b.2.b](#_Fig_2.1.b.2.b_Boxplots) removal of outliers)

### a. Probability Density Function with Histogram and Normal Fit



##### Fig 2.2.a.1 PDF with Normal

### b. Histogram and Mean of Predictors



##### Fig 2.2.b.1 Histogram with Mean line

# Modelling

In the above analysis done by us using various methods, we have been able to identify that the variable *fare\_amount* is our target variable. Hence, we only need to predict that variable using different methods. Given, our target variable is continuous (converted into continuous from encoded categorical), we will be relying only on various models to make our predictions. Below are the models we have used in our project.

## 1. Training the Model

We will be starting with the most basic of prediction models and see how this goes and then jump onto complex models like random forest.

### Multiple Linear Regression

Considering that we have multiple predictors, it is safe and efficient to use multiple regression on our variables in order to get a prediction model out of them.

##### Output of MLR

lm(formula = fare\_amount ~ ., data = dsPredictors)

Residuals:

Min 1Q Median 3Q Max

-267.72 -208.63 71.86 157.28 282.71

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 164.2615 111.9022 1.468 0.142

pickup\_longitude -3.0052 4.1818 -0.719 0.472

pickup\_latitude -0.2432 0.4765 -0.510 0.610

dropoff\_longitude 1.3025 2.5994 0.501 0.616

dropoff\_latitude -0.2795 3.3195 -0.084 0.933

passenger\_count -0.2531 2.7222 -0.093 0.926

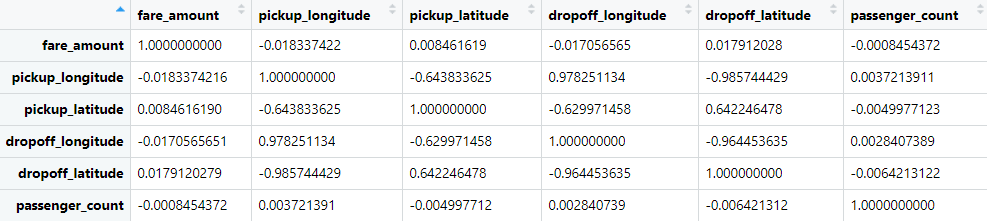
Residual standard error: 172.1 on 13946 degrees of freedom

Multiple R-squared: 0.0003745, Adjusted R-squared: 1.614e-05

F-statistic: 1.045 on 5 and 13946 DF, p-value: 0.3891

It is quite evident from the output that multiple linear regression has significantly failed. Multiple R-squared value suggests that we can explain about 0.03% of the data. Also, the p-value is greater than 0.05 which means that not even a single variable is dependent on the target. We should confirm this hypothesis from correlation matrix.

##### Fig 3.1.a.1 Correlation Matrix



If we observe the first row/column then, we can clearly conclude that *fare\_amount* (our target variable) is very poorly correlated with all our predictors and hence we cannot get a good prediction out of multiple linear regression model.

Hence, we will try with a much complex prediction model, i.e. Random Forests, in the next section.

### Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

In the coming sub-sections, you will see the output in R after running training data through a random forest model.

##### Output of Random Forest

Call:

randomForest(formula = fare\_amount ~ ., data = dsPredictors)

Type of random forest: regression

Number of trees: 500

No. of variables tried at each split: 1

Mean of squared residuals: 20575.19

% Var explained: 30.56

As we can clearly see, even random forest is giving a poor accuracy, only 30.56%. Although this result was with 500 trees (which is by default). Hence, we will try with number of trees 1000 and 1500 as well to see if we get better accuracy.

##### Output of Random Forest (with ntree=1000)

Call:

randomForest(formula = fare\_amount ~ ., data = dsPredictors, ntree = 1000)

Type of random forest: regression

Number of trees: 1000

No. of variables tried at each split: 1

Mean of squared residuals: 20518.71

% Var explained: 30.75

##### Output of Random Forest (with ntree=1500)

Call:

randomForest(formula = fare\_amount ~ ., data = dsPredictors, ntree = 1500)

Type of random forest: regression

Number of trees: 1500

No. of variables tried at each split: 1

Mean of squared residuals: 20585.57

% Var explained: 30.52

Clearly, increasing the number of trees has not changed anything, the model still cannot explain more than 30% of the data.

## 2. Conclusion

### 1. Model Evaluation

Now that we have already passed our training data through the prediction models, we will now determine which model should be actually used to predict our target variables. Selection of model can be done using various parameters, but in this project, we have decided to stick to *accuracy* of predictions as a measure to choose.

#### a. Root Mean Squared Error (RMSE)

To determine the accuracy of predictions, we will be using the RMSE (Root Mean Squared Error) method. Below is the formula used to calculate accuracy percentage, code of which was written in R:

##### Accuracy of our Models (R Output)

"Accuracy of Multiple Linear Regression is 35.732844553607"

"Accuracy of Random Forest is 30.8021895178643"

#### b. Model Selection

We can see from the above results that multiple linear regression is giving slightly better result. Hence, going forward to the next section, we will use multiple linear regression model to predict our target variable.

# Prediction

## 1. Model Testing

Now that we have already trained our models in the previous section, we will use this trained model (random forest, as concluded in the previous section) to predict our target variable (fare\_amount) in the test dataset provided to us.

##### Fig 4.1.a Test Dataset



We have run the above dataset against our random forest prediction model developed from the previous section to get [Fig 4.1.c](#_Fig_4.1.c_Final).

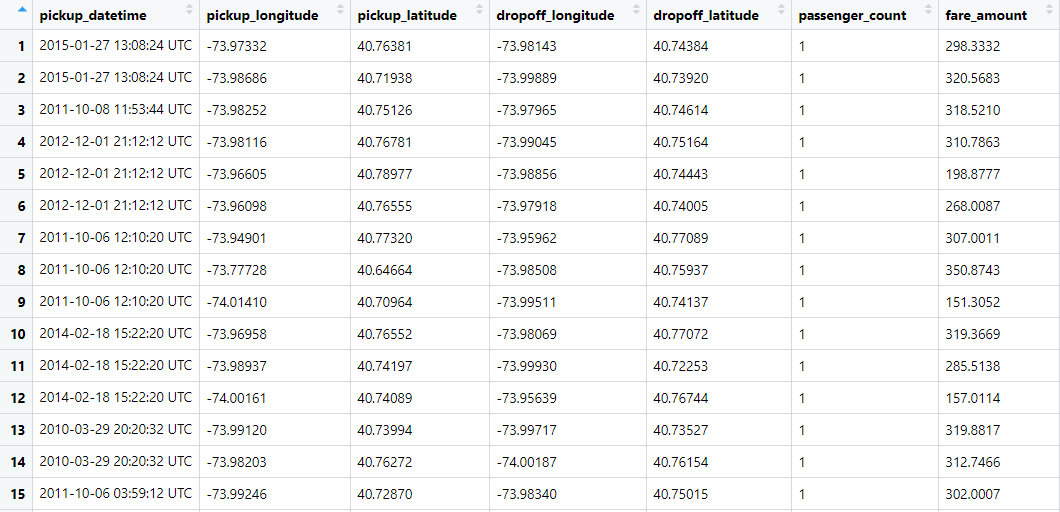
##### Fig 4.1.b R Code to predict fare\_amount

ds\_Test <- read.csv("C:/Users/riddh/OneDrive/Documents/Edwisor/Cab Fare Prediction/test.csv", sep = ",", header = T)

ds\_Final <- ds\_Test

ds\_Final$fare\_amount <- predict(mlrmodel, newdata=ds\_Final[,2:6])

##### Fig 4.1.c Final Dataset with Target Variable



# Appendix

## 1. R Code

### a. missingValueAnalysis.R

ds <- read.csv("C:/Users/riddh/OneDrive/Documents/Edwisor/Cab Fare Prediction/train\_cab.csv", sep = ",", header = T, na.strings = c(" ", "", "NA","0"))

ds\_missingVal = data.frame(apply(ds,2,function(x){sum(is.na(x))}))

ds\_missingVal$Columns = row.names(ds\_missingVal)

names(ds\_missingVal)[1] = "Missing\_percentage"

ds\_missingVal$Missing\_percentage = (ds\_missingVal$Missing\_percentage/nrow(ds)) \* 100

ds\_missingVal = ds\_missingVal[order(-ds\_missingVal$Missing\_percentage),]

row.names(ds\_missingVal) = NULL

ds\_missingVal = ds\_missingVal[,c(2,1)]

write.csv(ds\_missingVal, "Missing\_perc.csv", row.names = F)

ds\_new <- na.omit(ds)

ds\_new = ds\_new[,c(2,1,3,4,5,6,7)]

for (i in 2:7){

if(class(ds\_new[,i]) == "factor")

{

ds\_new[,i] <- as.numeric(ds\_new[,i])

}

}

write.csv(ds\_new, "new\_trainCab.csv", row.names = F)

### b. outlierAnalysis.R

library(psych)

library(ggplot2)

drawBoxP <- function(df){

for (i in 2:7){

title\_val <- paste("Boxplot (without Outliers) of ",colnames(df[i])) # paste("Boxplot without Outliers of ",colnames(df[i]))

box\_plot <- ggplot(df, aes(x='', y= df[,i])) +

geom\_boxplot(outlier.color = "red", outlier.shape = 19,

outlier.size = 1.5, outlier.stroke = 0.5) +

labs(title=title\_val,x=colnames(df[i]), y = "Frequency") +

geom\_jitter(shape=16, position=position\_jitter(0.2)) #with jitter

print(box\_plot)

}

}

#boxplot with outliers

drawBoxP(ds\_new)

#removing outliers from passenger\_count

unwanted\_outliers <- boxplot(ds\_new$passenger\_count, plot = FALSE)$out

dsOutRemoved <- ds\_new[-which(ds\_new$passenger\_count %in% unwanted\_outliers ),]

drawBoxP(dsOutRemoved)

#effect of outliers

#with outliers

boxplot(ds\_new$passenger\_count, main="Boxplot for passenger\_count with outliers",

ylab="passenger\_count")

hist(ds\_new$passenger\_count, main="Histogram for passenger\_count with outliers",

xlab="passenger\_count")

#without outliers

boxplot(dsOutRemoved$passenger\_count, main="Boxplot for passenger\_count without outliers",

ylab="passenger\_count")

hist(dsOutRemoved$passenger\_count, main="Histogram for passenger\_count without outliers",

xlab="passenger\_count")

### c. dataVisualization.R

library(psych)

library(ggplot2)

#Probability Density Funciton with histogram and normal fit

multi.hist(dsOutRemoved[,2:7], main = NULL, dcol = c("blue", "red"),

dlty = c("solid", "solid"), bcol = "linen")

#histogram with mean line

for (i in 2:7){

title\_val <- paste("Histogram and Mean of ",colnames(dsOutRemoved[i]))

hist\_plot<-ggplot(dsOutRemoved, aes(x=dsOutRemoved[,i])) +

geom\_histogram(binwidth=0.5,color="darkblue", fill="lightblue",

linetype="solid")+

labs(title=title\_val,x=colnames(dsOutRemoved[i]), y = "Frequency")+

theme\_classic() +

geom\_vline(aes(xintercept=mean(dsOutRemoved[,i])),

color="red", linetype="dashed", size=1)

print(hist\_plot)

}

### d. mlr\_rf.R

library(tidyverse)

library(randomForest)

library(caret)

dsPredictors <- dsOutRemoved[,2:7]

mlrmodel <- lm(fare\_amount ~., data = dsPredictors)

summary(mlrmodel)

rfmodel <- randomForest(fare\_amount ~ ., data = dsPredictors)

print(rfmodel)

trainIndex = createDataPartition(dsPredictors$fare\_amount, p = .80, list = FALSE)

dsTest <- dsPredictors[-trainIndex,1:6]

predictionsRF\_DT = predict(rfmodel, dsTest[,2:6])

predictionsMLR\_DT = predict(mlrmodel, dsTest[,2:6])

AccuracyMLR<- 100 - RMSE(predictionsMLR\_DT, dsTest$fare\_amount)\*100/mean(dsTest$fare\_amount)

print(paste("Accuracy of Multiple Linear Regression is",AccuracyMLR))

AccuracyRF <- 100 - RMSE(predictions\_DT, dsTest$fare\_amount)\*100/mean(dsTest$fare\_amount)

print(paste("Accuracy of Random Forest is",AccuracyRF))

### e. predict.R

ds\_Test <- read.csv("C:/Users/riddh/OneDrive/Documents/Edwisor/Cab Fare Prediction/test.csv", sep = ",", header = T)

ds\_Final <- ds\_Test

ds\_Final$fare\_amount <- predict(mlrmodel, ds\_Final[,2:6])

## 2. Instructions to Run the Code

To run the code, follow the below steps: -

* Please open the R Studio.
* Ensure that all the packages required in all the code snippets above are installed in the R Studio.
* Wherever ‘csv’ file location is mentioned, please change the location as per the presence of that file in the machine used to run the R script.
* The order in which these scripts must be run is pretty straightforward, when all the above steps are cross-checked, just the above scripts in alphabetical order, that is, see below -

a. missingValueAnalysis.R -> b. outlierAnalysis.R -> c. dataVisualization.R -> d. mlr\_rf.R -> e. predict.R

# References

[www.geeksforgeeks.org](http://www.geeksforgeeks.org)

[www.wikipedia.org](http://www.wikipedia.org)

[www.towardsdatascience.com](http://www.towardsdatascience.com)

[www.medium.com](http://www.medium.com)

[www.data-flair.training](http://www.data-flair.training)

[www.sthda.com](http://www.sthda.com)

[www.r-graph-gallery.com](http://www.r-graph-gallery.com)

[www.machinelearningmastery.com](http://www.machinelearningmastery.com)

<www.ucl.ac.uk>