2021

Project Report

CAB FARE PREDICTION

SREEJITH SUKUMARAN

Contents

INTRODUCTION:

In this project, we were interested to see what factors can predict the cab fare and built a model for the same.

Problem Statement -

You are a cab rental start-up company. You have successfully run the pilot project and now want to launch your cab service across the country. You have collected the historical data from your pilot project and now have a requirement to apply analytics for fare prediction. You need to design a system that predicts the fare amount for a cab ride in the city.

Data Set:

1) train_cab.csv
   We will load this data as df.
2) test.csv
   We will load this data as df1

Exploratory data analysis

Exploratory Data Analysis refers to the critical process of performing initial investigations on data to discover patterns, spot anomalies, test hypotheses, and check assumptions with the help of summary statistics and graphical representations.

The train_cab.csv data set consists of the following features: fare_amount, pickup_datetime, pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, passenger_count. Among this fare_amount is the dependent variable and the remaining data variables are independent.

Following are the findings:

Data description:

Size of train_cab: - 16067 rows, 7 Columns (including dependent variable) Below mentioned is a list of all the variable names with their meanings:

Variables	Description
fare_amount	Fare amount
pickup_datetime	Cab pickup date with time
pickup_longitude	Pickup location longitude
pickup_latitude	Pickup location latitude

dropoff_longitude	Drop location longitude
dropoff_latitude	Drop location latitude
passenger_count	Number of passengers sitting in the cab

Size of test: - 16067 rows, 6 Columns. Below mentioned is a list of all the variable names with their meanings:

Variables	Description
pickup_datetime	Cab pickup date with time
pickup_longitude	Pickup location longitude
pickup_latitude	Pickup location latitude
dropoff_longitude	Drop location longitude
dropoff_latitude	Drop location latitude
passenger_count	Number of passengers sitting in the cab

1. Data type errors

a. fare_amount: The data type was character/object instead of being a numeric variable.
b. pickup_datetime: The data type was character/object instead of being a date-time variable.
c. passenger_count: The data type was numeric but was float instead of an integer.

2. Missing values

a. passenger_count: there were 55 missing values in this variable in train data.
b. fare_amount: 25
c. pickup_datetime:1

3. Outliers

a. Negative fare amount
b. Observations with fare_amount > 500 and equals to 0
c. pickup_latitude out of the range
d. Unique values of passenger_count indicate outliers.
   1 2 3 NA 6 5 4 236 456 5334 0 535 354 554 53 35 345 5345 536 43 58 537 87 531 557

4. Other observations

a. There were negative values in fare amount which need to be removed also there were outliers which had to be removed

b. The Maximum passenger count could be 6 in case the vehicle is an SUV also minimum should be 1. In the test data, this range is correctly mentioned.
c. Latitudes range from -90 to 90. Longitudes range from -180 to 180.

Data Cleaning

1. fare_amount:

a. Converted the column to a numeric datatype.
b. Removed negative values.
c. Removed outliers and the resulting minimum and maximum values are 1.14 and 453 respectively.
d. Removed missing values.

2. pickup_datetime:

e. Converted data type to DateTime format.
f. Removed missing values.

3. passenger_count:

a. Imputed missing value with the mode of the passenger count which was 1 in this scenario in R. We could also remove the data as there are only negligible missing values. In python, we removed it.

4. pickup_longitude

a. Removed rows with 0 as the pickup_longitude.

5. pickup_latitude

a. Outlier detected with value about 400. Latitude ranges from -90 to 90. Removed values that did not fit in the following range, viz; -90 to 90.
b. Removed rows with 0 as the pickup_latitude.

6. dropoff_longitude

a. Removed rows with 0 as the dropoff_longitude.

7. dropoff_latitude

a. Removed rows with 0 as the dropoff_latitude.

Feature Engineering

Derived columns:

Here in our data set our variable name pickup_datetime contains the date and time for pickup. So we tried to extract some important variables from pickup_datetime:

- Year
- Month
- Date

- Day of Week
- Hour

We can derive distance from the latitude and longitude data using the Haversine formula.

The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles. The formula is:

$$d = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

Therefore, the additional columns are:

- year
- month
- date
- day of Week
- hour
- distance

The derived variables are analyzed and processed by cleaning the missing values and outliers.

Feature Selection

As we have derived the useful variables, we will drop the redundant variables with the same information. we have split the pickup_datetime to the required data and we can remove it from the data frame. Also, we have derived distance from Latitudes and longitudes. Hence, we will be dropping the following variables:

- pickup_datetime
- pickup_longitude
- pickup_latitude
- dropoff_longitude
- dropoff_latitude

Therefore, the head of the train(df in R and python) will look like:

fare_amount	passenger_count	year	month	day	day_of_week	hour	distance
4.5	1	2009	6	15	1	17	1.030787
16.9	1	2010	1	5	2	16	8.450326
5.7	2	2011	8	18	4	0	1.389557

7.7	1	2012	4	21	6	4	2.799334
5.3	1	2010	3	9	2	7	1.999202
12.1	1	2011	1	6	4	9	3.787325

Feature Scaling

Machine learning algorithms like linear regression, logistic regression, neural network, etc. that use gradient descent as an optimization technique require data to be scaled.

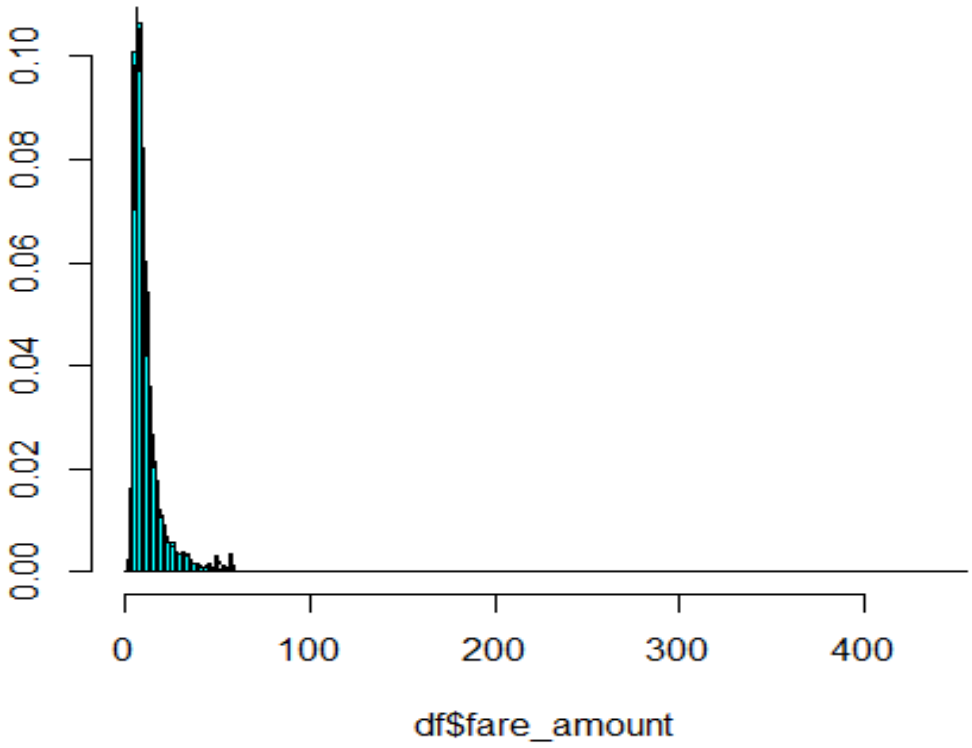
Distance algorithms like KNN, K-means, and SVM are most affected by the range of features. This is because behind the scenes they are using distances between data points to determine their similarity.

Tree-based algorithms, on the other hand, are fairly insensitive to the scale of the features.

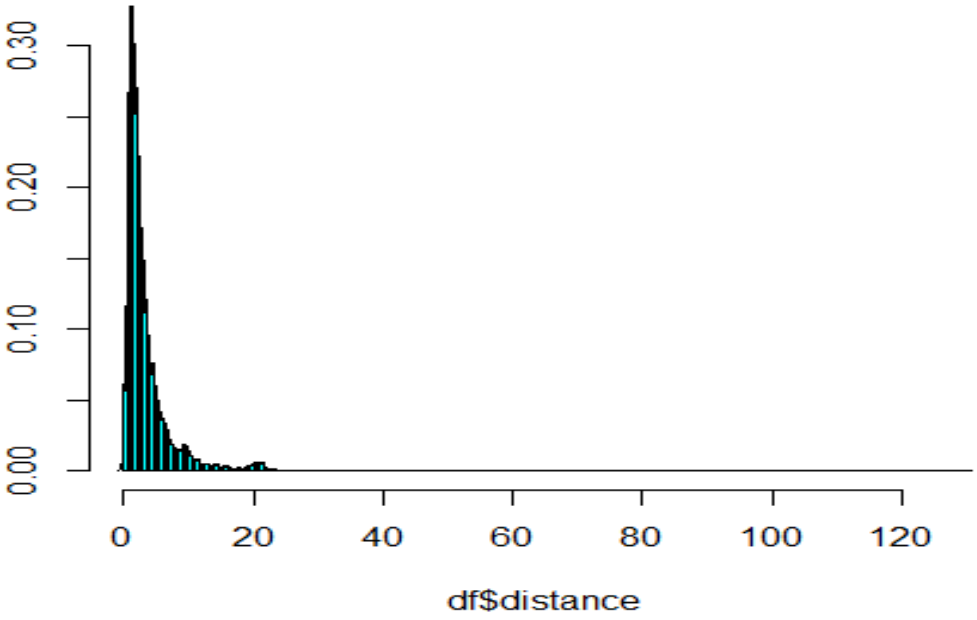
We have a regression problem and we will start from linear regression to test the most suitable model. Therefore, its important that we avoid skewness in our data.

We have to check the skewness of fare_amount and distance as they are continuous and the rest of the variables can be treated as categorical variables.

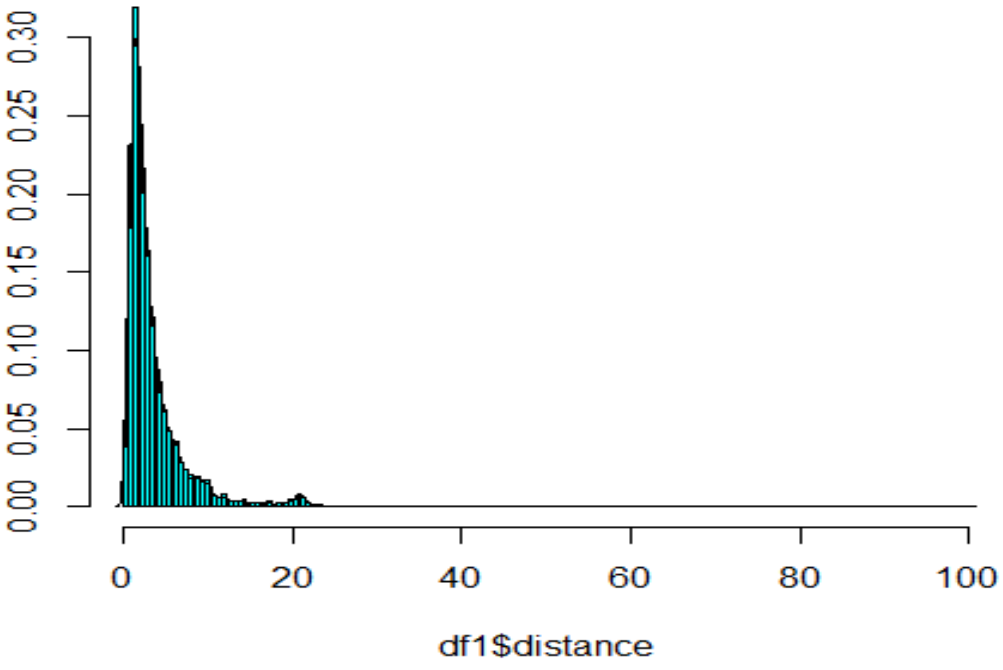
The distribution of fare_amount in df, before log transformation is as shown below:



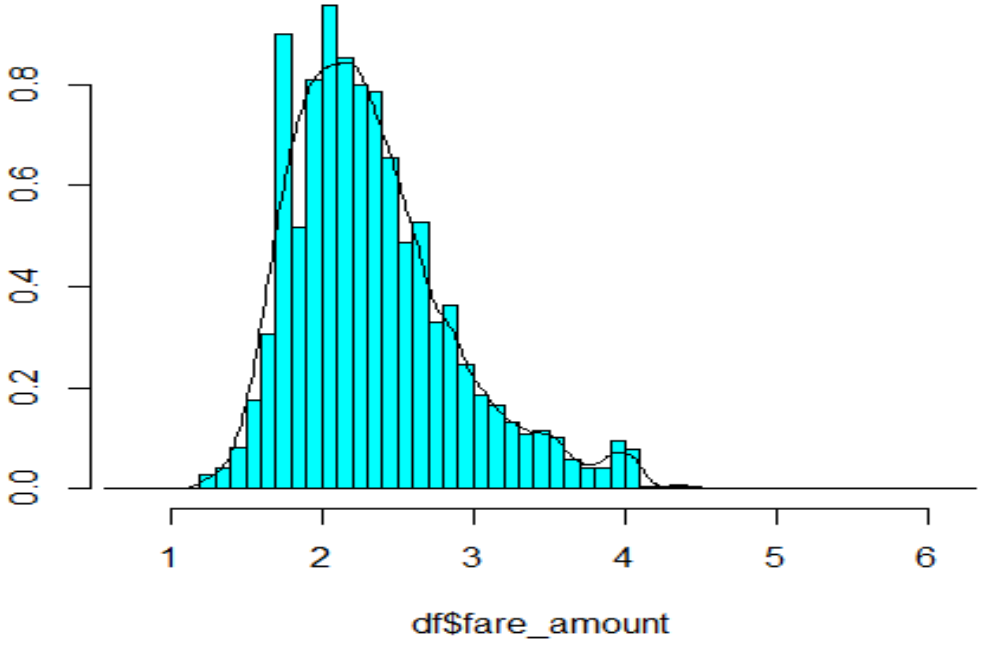
The distribution of distance in df, before log transformation is as shown below:



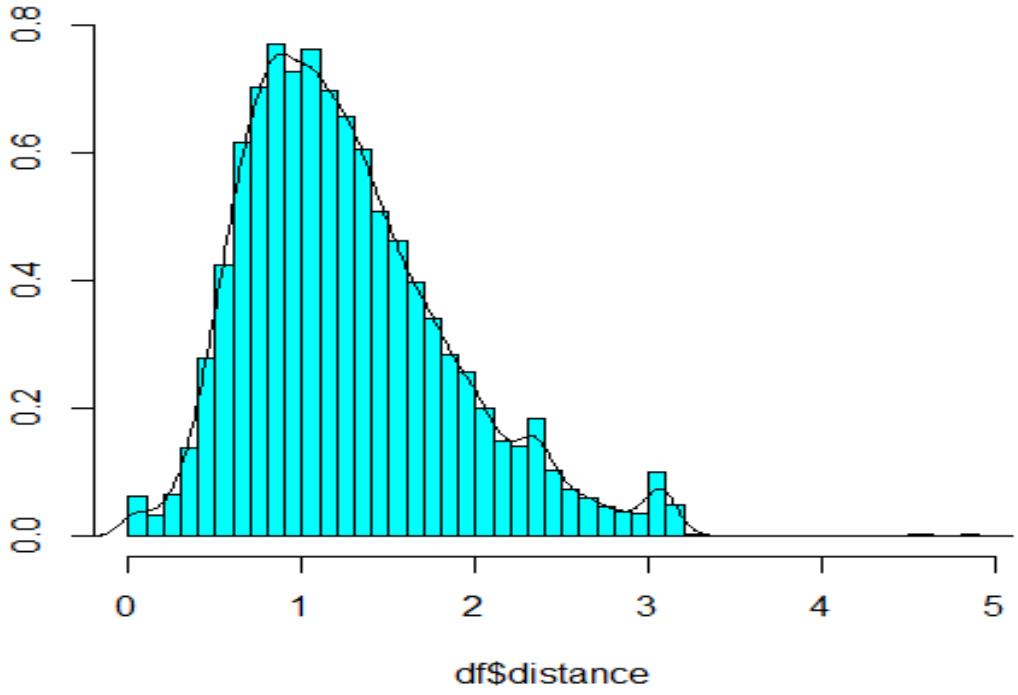
The distribution of distance in df1, before log transformation is as shown below:



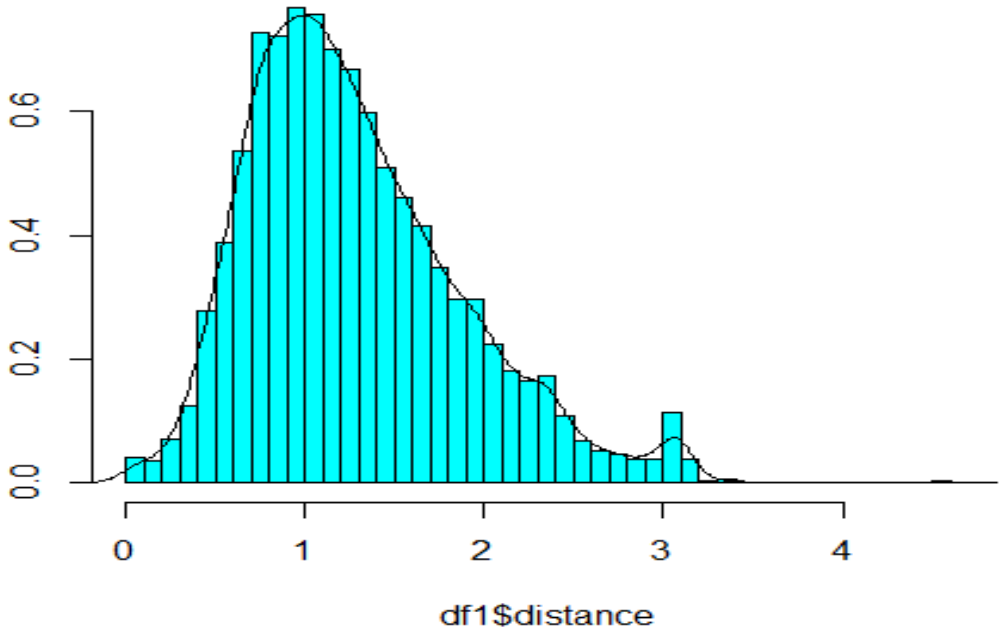
The distribution of fare_amount in df, after log transformation is as shown below:



The distribution of distance in df, after log transformation is as shown below:



The distribution of distance in df1, after log transformation is as shown below:



Now, the continuous variables appears to be normally distributed so we don't need to use feature scaling techniques like normalization and standardization for further scaling.

Model Building

We select the model based on the target variable or feature. Here, the target variable is continuous, and hence, this is a regression problem which requires a regression model, to predict continuous outcomes.

Before running any model, we will split our data into two parts which is train and test data. Here in our case we have taken 75% of the data as our train data.

Linear Regression

The first approach we tried was Multiple Linear Regression, Multiple linear regression is the most common form of linear regression analysis. Multiple regression is an extension of simple linear regression. It is used as a predictive analysis, when we want to predict the value of a variable based on the value of two or more other variables. The variable we want to predict is called the dependent variable (or sometimes, the outcome, target or criterion variable.

Below is a screenshot of the model we build and its output:

```
# Lets apply ML algorithms and select the best fit
##LINEAR REGRESSION

#Model development
LR_model<-lm(fare_amount~.,data=df_train)
summary(LR_model)
#predict test data by LR model
pred_test=predict(LR_model,df_test)

print(postResample(pred=pred_test,obs = df_test$fare_amount))
# RMSE        Rsquared         MAE
#0.2636713  0.7651328     0.09206735
library(DMwR)

regr.eval(df_test[,1],pred_test)

#Calculate MAPE

library(MLmetrics)

LR_mape=MAPE(df_test[,1],pred_test)
print(LR_mape)

#MAPE= 0.07417985
#Error rate=7.4%
#Accuracy=92.6%
```

Decision Tree

The second approach tried was of decision tree, Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. Following is a snapshot of the code used:

```
# decision tree regressor
library(rpart)
DT=rpart(fare_amount~.,data=df_train,method="anova")
DT_test=predict(DT,df_test)
summary(DT)

print(postResample(pred=DT_test,obs = df_test$fare_amount))
#   RMSE      Rsquared         MAE
#0.2697584   0.7541586    0.1859662

#Calculate MAPE
DT_mape=MAPE(df_test[,1],DT_test)
print(DT_mape)

#MAPE= 0.08165256
#Error rate=8.2%
#Accuracy=91.8%
```

Model Evaluation

The key metrices used to compare and evaluate the model are as follows,

1. RMSE (Root Mean Square Error): is a frequently used measure of the difference between values predicted by a model and the values actually observed from the environment that is being modelled.

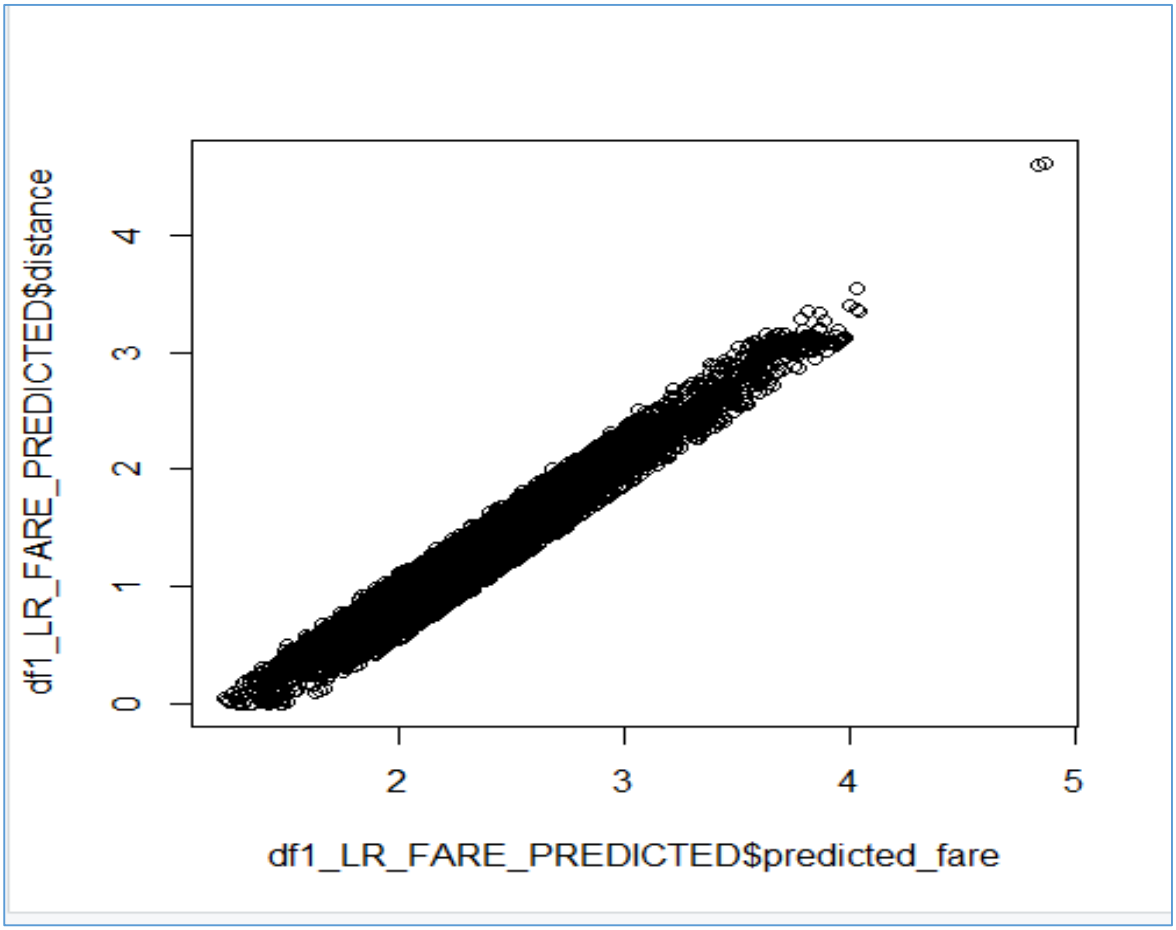
$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(X_{obs,i} - X_{mo\ del,i})^2}{n}}$$

2. R Squared(R^2): is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. In other words, we can say it explains as to how much of the variance of the target variable is explained.

SL	Model Type	MAPE	RSquared	Error Rate	Accuracy
1.	Multiple Linear Regression	0.07	0.765	7.4	92.6
2.	Decision Tree	0.08	0.754	8.2	91.8

By analysing the results obtained from both the approaches the best fit for the current problem statement is Linerar Regression as the Accuracy, RSquare score are better and also the error rate is comparatively less as compared to the Decision Tree approach.

Graphical representation distance vs predicted fare on the test dataset



Code snippet used for prediction

```
#PREDICT FARE OF TEST DATA
predicted_fare_DT=predict(DT,df1_final)
df1_DT_FARE_PREDICTED= cbind(df1,predicted_fare_DT)
summary(df1_DT_FARE_PREDICTED)
```

END OF REPORT

References

https://learning.edwisor.com/

https://www.analyticsvidhya.com/

https://towardsdatascience.com/

https://stackoverflow.com/

https://datatofish.com/

https://en.wikipedia.org/wiki/Wikipedia