Project Report
<u>Project: Cab Fare Prediction</u>

Aparajita Mallick Sarkar

Contents

1. Introduction

1.1. 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.

2.1. Data

The historical data from the pilot project is provided giving various aspects of the cab rides. There are 6 independent variables and one target variable, that is, 'fare_amount'. Total number of observations in 16067. The data set provided has missing values. Below is a sample data used to predict cab fare.

Table 1.1: Sample Data

fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
4.5	2009-06-15 17:26:21 UTC	-73.8443	40.72132	-73.8416	40.71228	1
16.9	2010-01-05 16:52:16 UTC	-74.016	40.7113	-73.9793	40.782	1
5.7	2011-08-18 00:35:00 UTC	-73.9827	40.76127	-73.9912	40.75056	2
7.7	2012-04-21 04:30:42 UTC	-73.9871	40.73314	-73.9916	40.75809	1
5.3	2010-03-09 07:51:00 UTC	-73.9681	40.76801	-73.9567	40.78376	1

The details of data attributes in the dataset are as follows:

- pickup_datetime – A timestamp value indicating when the cab ride started.
- pickup_longitude – A float value for longitude coordinate of where the cab ride started.
- pickup_latitude – A float value for latitude coordinate of where the cab ride started.
- dropoff_longitude – A float value for longitude coordinate of where the cab ride ended.
- dropoff_latitude – A float value for latitude coordinate of where the cab ride ended.

- passenger_count – An integer value indicating the number of passengers in the cabride.
- fare_amount- A float value indicating the fare of the cab ride. This is the target variable.

2. Methodology

2.1. Pre-Processing

Data Pre-processing is the step in which the data gets transformed, to bring it to such a state that the machine can easily understand it. In other words, the features of the data can then be easily interpreted by the algorithm. It includes exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis.

The following steps are taken under the pre-processing:

a. Initial data types of the variables are below:

• fare_amount	categorical
• pickup_datetime	categorical
• pickup_longitude	float
• pickup_latitude	float
• dropoff_longitude	float
• dropoff_latitude	float
• passenger_count	float

b. Converting the data type of 'fare_amount' to float.

c. Deriving the month, week of the day and the hour of pick up from 'pickup_datetime' as categorical variable with the names 'pickup_month', 'pickup_day' and 'pickup_hour' respectively. The value of 'pickup_month' varies from 1 to 12, where 1 denotes January and 12 denotes December. The value of 'pickup_hour' varies from 0 to 23, denoting 24 hours time format. The value of 'pickup_day' varies from 1 to 7 in R, where 1 denotes Monday and 7 denotes Sunday, whereas it varies from 0 to 6 in Python, where 0 denotes Monday and 6 denotes Sunday.

d. 'pickup_datetime' variable is dropped.

e. Distribution of all the categories of categorical data is visualised using histograms.

f. 'passenger_count' varies from 0 to 5345 and also contain float values. The 'passenger_count' can't have decimal values. Most of the cabs can seat maximum 6 passenger and 'passenger_count' cannot be zero. Thus the observations with 'passenger_count' from 1 to 6 are retained, and in the rest it is made as 'NA.

g. 'fare_amount' can't be zero or negative, in such observations the value is made 'NA'.

h. Location coordinates cannot be zero and latitude value cannot be more than 90, thus such observations are dropped.

Fig2.1: Visualization of categorical variables in R (for Raw Data)

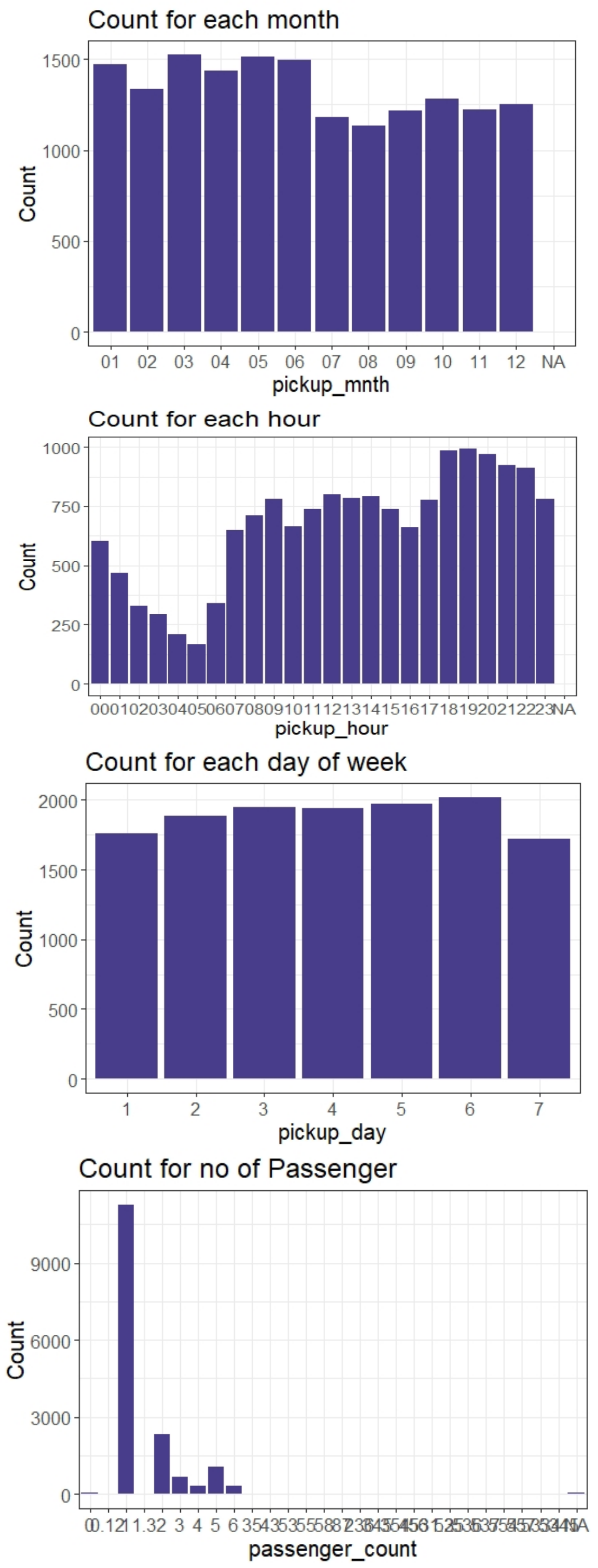
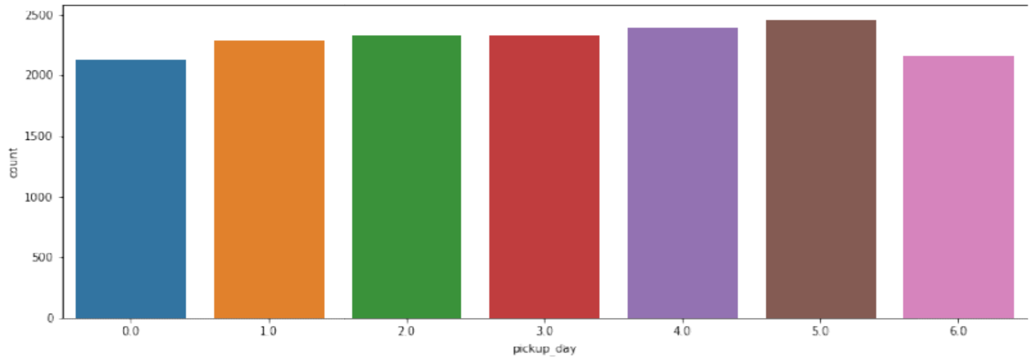
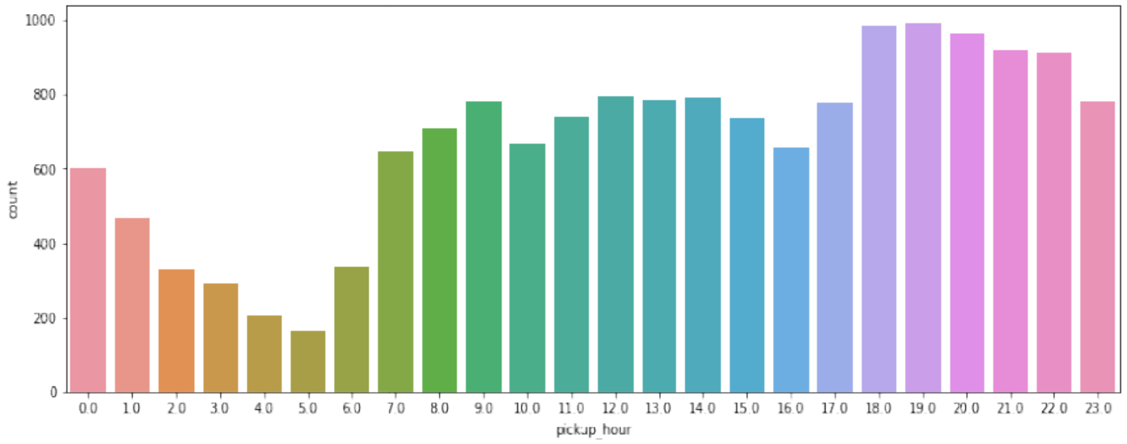
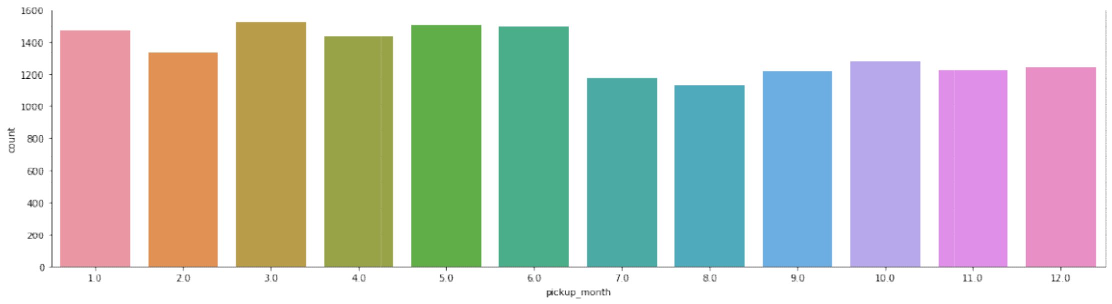
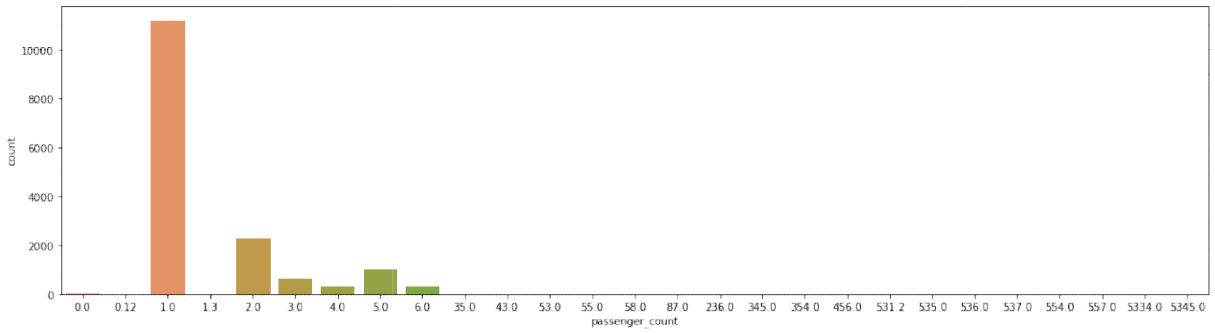


Fig2.2: Visualization of categorical variables in Python (for Raw Data)









2.1.1. Missing Value Analysis

Missing Value analysis is done to check for any missing values in the dataset. Missing values can be dealt in the following ways:

a. Dropping the variables in which the values are missing.
b. Dropping the observations where the values are missing.
c. In case the number of missing values in a variable is less than 30% of the total count of observation, the values can be imputed using central statistics, distance based methods like knn or prediction methods.
d. Table for the percentage of observation with missing values is shown for python and R. The difference in value in R and Python may have emerged during changing data types causing a value to become NA.
e. The paper "Multiple imputation: a primer." By Schafer JL, Stat Methods Med Res. 1999 Mar; 8(1):3-15 asserted that a missing rate of 5% or less is inconsequential. Thus the observations with the missing values are dropped.
f. After dropping the observations, there 15584 observations in R and 15583 in python.

Table 2.1: Missing value percentage calculated in R

	Columns	Missing_percentage
1	passenger_count	0.9784611
2	pickup_mnth	0.1651947
3	pickup_day	0.1651947
4	pickup_hour	0.1651947
5	fare_amount	0.1588411
6	pickup_longitude	0.1588411
7	pickup_latitude	0.1588411
8	dropoff_longitude	0.1588411
9	dropoff_latitude	0.1588411

Table2.2: Missing Value Percentage in Python

	Variables	Missing_percentage
0	passenger_count	0.825921
1	fare_amount	0.171537
2	pickup_month	0.006353

	Variables	Missing_percentage
3	pickup_day	0.006353
4	pickup_hour	0.006353
5	pickup_longitude	0.000000
6	pickup_latitude	0.000000
7	dropoff_longitude	0.000000
8	dropoff_latitude	0.00000

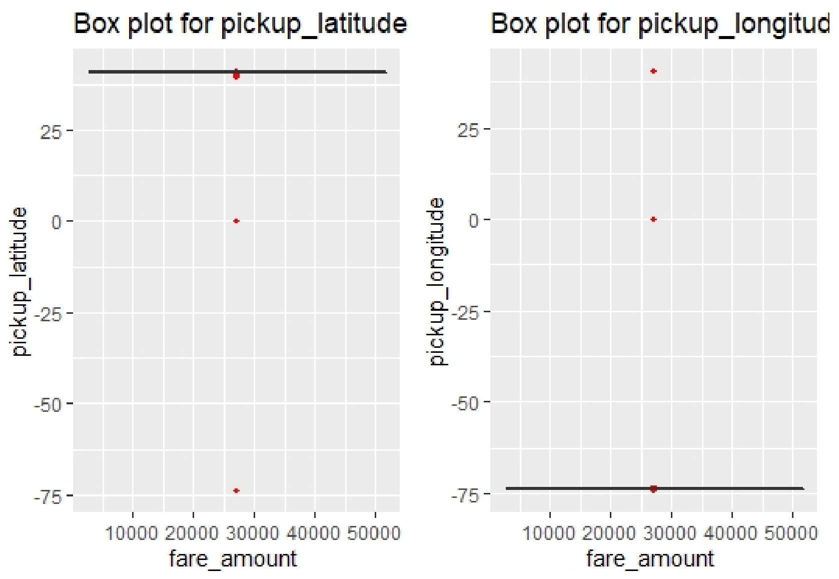
2.1.2. Outlier Analysis

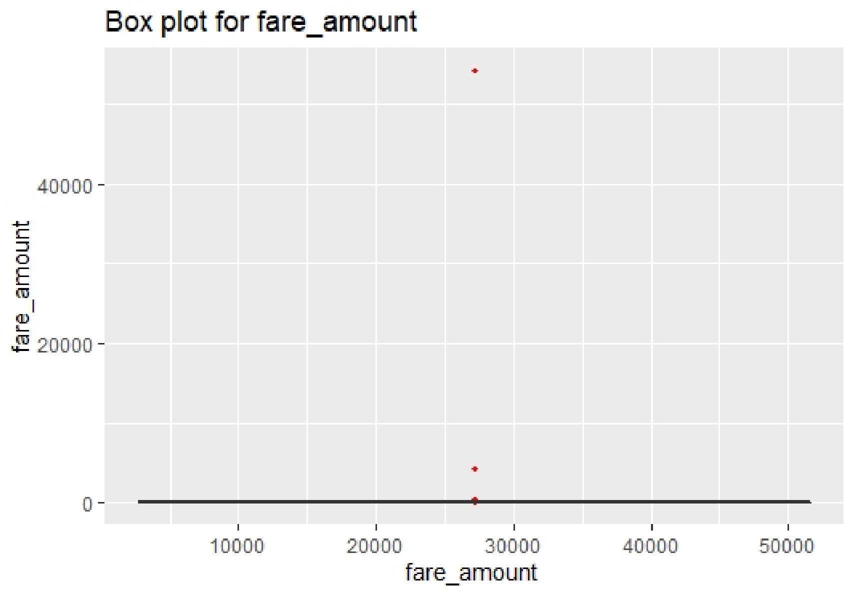
Outliers are observations which are inconsistent with rest of the data set and are defined only for continuous variables. In the given data set the following are the continuous independent variables:

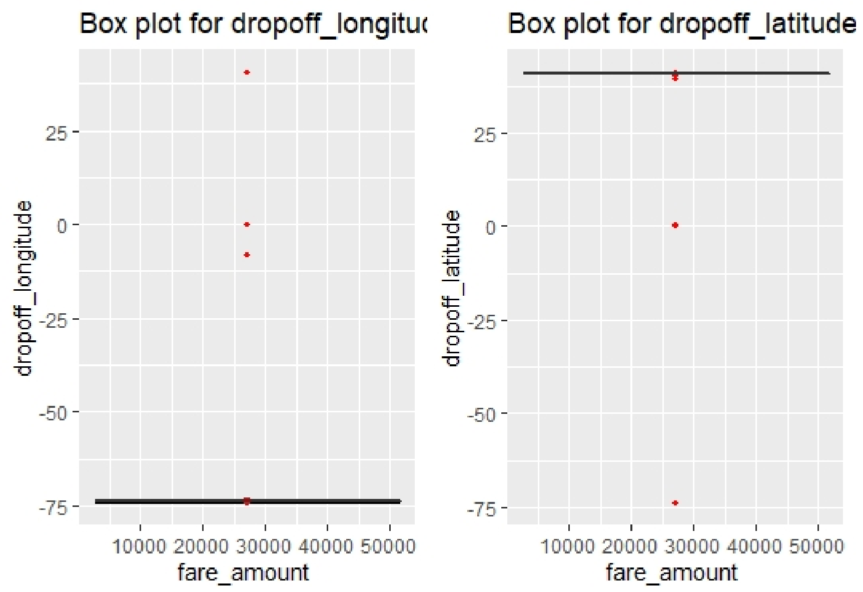
a. pickup_longitude
b. pickup_latitude
c. dropoff_longitude
d. dropoff_latitude
e. fare_amount

Boxplot is used to detect outliers.

Fig 2.3: Box Plots for Outlier Detection







As can be seen, outliers are present in all continuous variables. All the outlier values are detected and replaced with NA.

In the initial run, the NA values were imputed with the appropriate methods as given below.

a. Mean, Median and KNN imputation is tested to find the most appropriate method for imputation.
b. In R all the values are imputed using KNN Imputaion using the function knnImputation(), where k=3.
c. In python, Median is used for imputing ‘dropoff_longitude’, Mean is used for imputing 'pickup_longitude' and 'dropoff_latitude' and KNN Imputaion is used for imputing the values of ‘fare_amount’ and ‘pickup_latitude’ using KNNImputer() from sklearn library with n_neighbors=3.

The models developed on the imputed data were found to be inefficient. Thus it can be concluded that the outliers are result of noisy data, thus the observations with NA(outlier) values were deleted.

2.1.3. Feature Engineering

Feature engineering is the process of using domain knowledge to extract new features from given features. These features can be used to improve the performance of machine learning algorithms.

On a general account, the cab fare is dependent on the distance travelled. The pick and drop off coordinates for the cab are provided. The haversine formula is used to determine the distance between the points of pickup and dropoff.

The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. It is given by the following formula:

$$a = \sin^2(\frac{\Delta\varphi}{2}) + \cos\varphi1 \cdot \cos\varphi2 \cdot \sin^2(\frac{\Delta\lambda}{2})$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

ϕ is latitude in radian
λ is longitude in radian
R is earth’s radius (6,367 km)
D is the distance in Km

The observations with distance as 0, denotes that pickup and dropoff locations are same, thus these observations can be considered noisy data. These observations are dropped.

Fig 2.4: Distance Vs Fare Amount in R

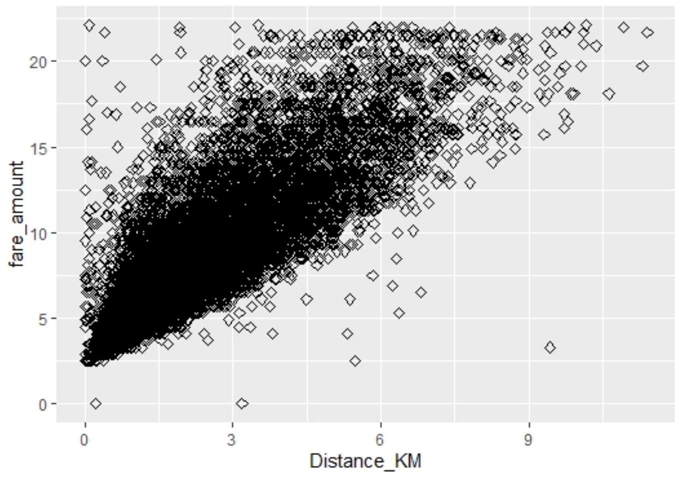
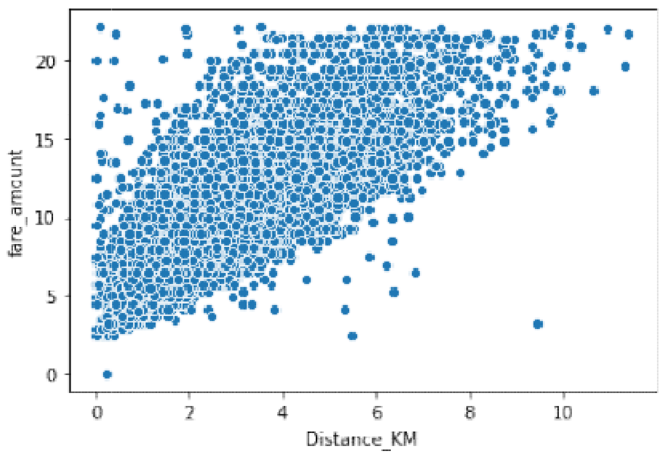


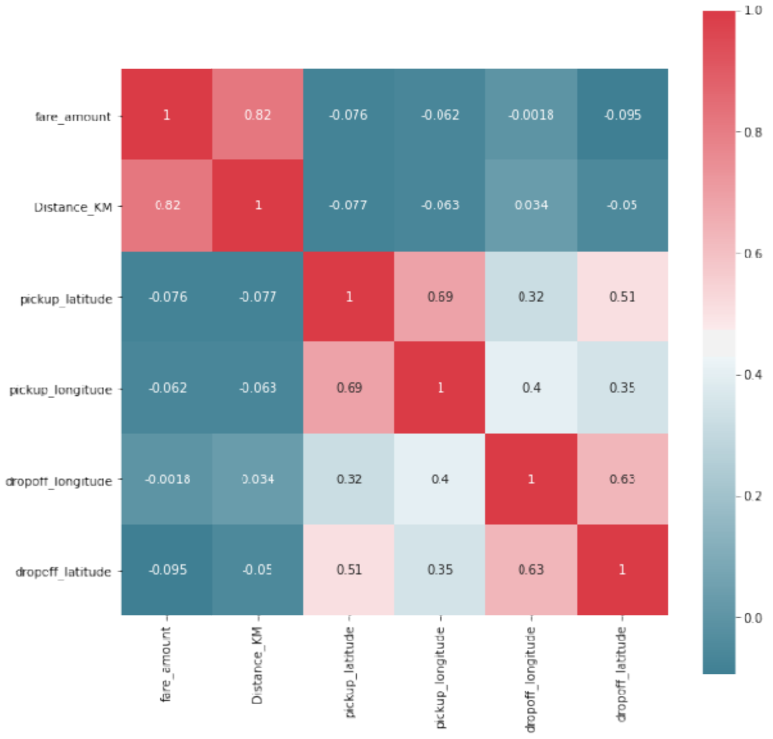
Fig 2.5: Distance Vs Fare Amount in Python



2.1.4. Correlation Analysis

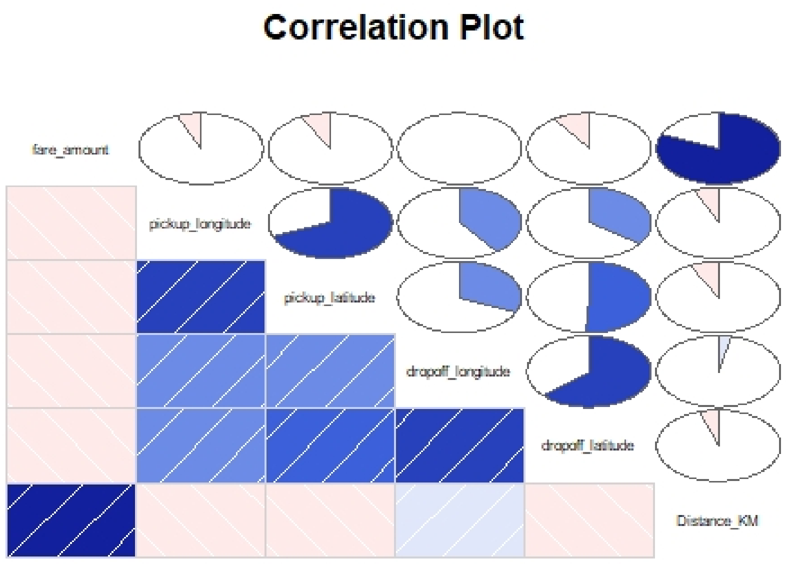
Feature selection includes extracting meaningful feature from the dataset. This includes removing the unimportant data thus reducing the complexity of the model. **Correlation Analysis** is done considering that the independent variables should have minimum correlation between them and the correlation between the target variable and the independent variables should be high. Any pair of independent variables showing high correlation leads to multicollinearity. Thus one of such variable should be removed. The correlation analysis shows no multicolliniarity amongst the independent variables. Thus all the continuous independent variables are retained

Fig 2.6: Correlation values for Continuous variables, calculated in python



	fare_amount	Distance_KM	pickup_latitude	pickup_longitude	dropoff_longitude	dropoff_latitude
fare_amount	1	0.82	-0.076	-0.062	-0.0018	-0.095
Distance_KM	0.82	1	-0.077	-0.063	0.034	-0.05
pickup_latitude	-0.076	-0.077	1	0.69	0.32	0.51
pickup_longitude	-0.062	-0.063	0.69	1	0.4	0.35
dropoff_longitude	-0.0018	0.034	0.32	0.4	1	0.63
dropoff_latitude	-0.095	-0.05	0.51	0.35	0.63	1

Fig. 2.7: Correlation Plot in R



2.2. Modelling

2.2.1. Model Selection

The problem is categorised under Regression Problem where a continuous target variable, 'fare_amount', is to be predicted using the historical data. Thus various regression models are built on the historical data. The motive of building of the model is to know how well the model predicts the new data rather than how well it fits the data it was trained on. Thus the historical data is divided into 'Train Data' on which the model is built and the 'Test Data' which is used to compare the actual value to that of the predicted values by the model, to estimate the performance of the model. The difference between the actual value and the predicted value is the 'error'. Low error is desirable.

Following steps are taken for each of the model:

- 80 % of the data is selected by simple random sampling as the 'Train Data' and the rest of the 20% as the 'Test Data'.
- Model is built on the Train Data.
- Prediction is made using the model for the 'Test Data'.
- The predictions are stored and further used to compare with the actual value to select the appropriate model.

Following regression models are built:

- Multiple Linear Regression
- Decision tree
- Random Forest (with 200 trees)
- Gradient Boosting

2.2.2. Multiple Linear Regression

Multiple Linear Regression is a statistical model used for regression problems, where the coefficients or weights are calculated for each of the independent variable. This coefficient explains the amount of information the independent variable carries to explain the target variable. The main idea is to identify a line that best fits the data. This algorithm is not very flexible.

It can be represented as below:

$$y = b_0 + b_1x_1 + b_2x_2 + \ldots\ldots + b_nx_n$$

$$b_0 \rightarrow \text{Intercept}$$

$$b_1, b_2, \ldots., b_n \rightarrow \text{Coefficients of independent variable}$$

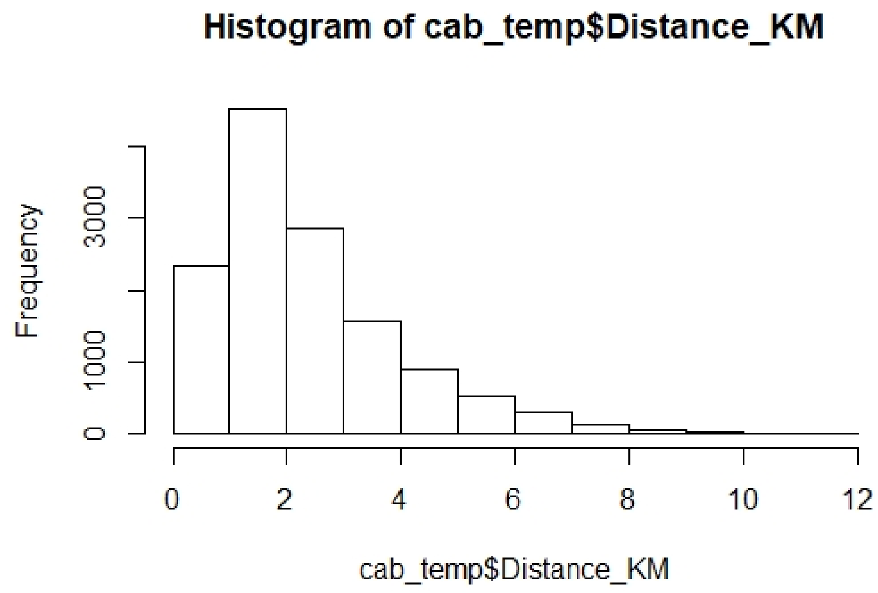
$$x_1, x_2, \ldots., x_n \rightarrow \text{Independent variables}$$

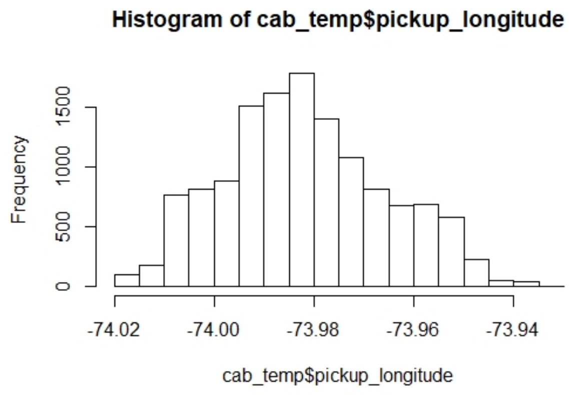
$$y \rightarrow \text{Target Variable}$$

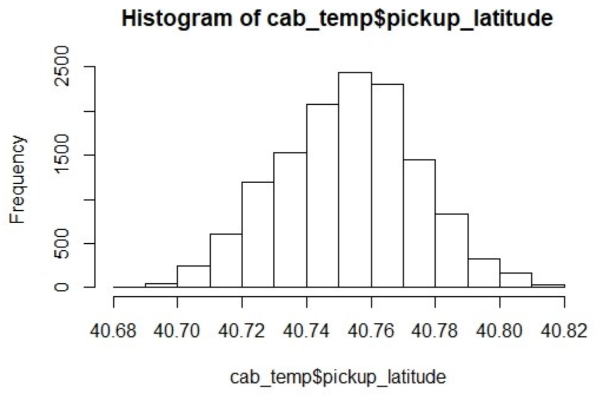
As linear regression model calculates weights for each independent variables, a variable having a high range can affect the values of the weights. Thus it is advised to perform feature scaling.

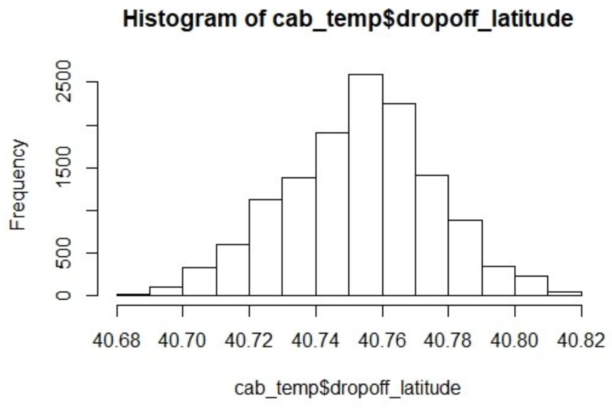
For normally distributed independent variables, standardisation is recommended else normalisation is used to scale the feature. Histograms are used to check the normality of the independent continuous variables.

Fig 2.8: Distribution of Independent Continuous variables in R




Histogram of cab_temp$pickup_longitude
Frequency
cab_temp$pickup_longitude


Histogram of cab_temp$pickup_latitude
Frequency
cab_temp$pickup_latitude


Histogram of cab_temp$dropoff_latitude
Frequency
cab_temp$dropoff_latitude

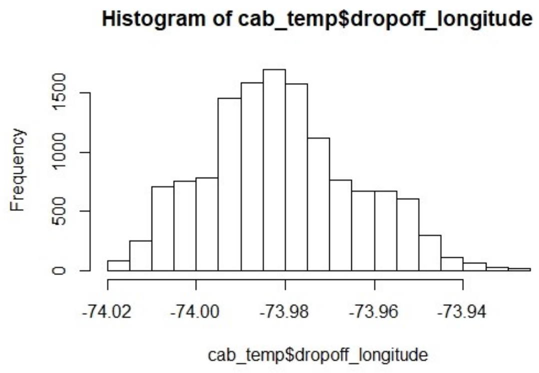
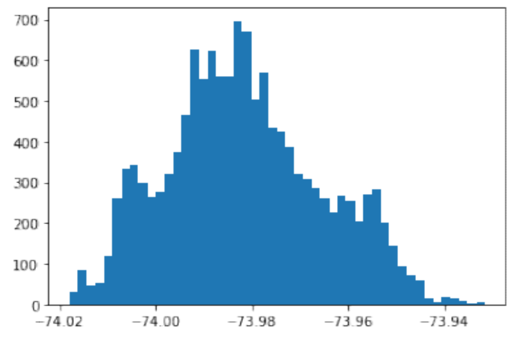
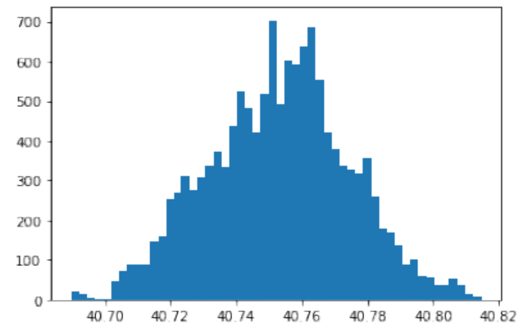

Histogram of cab_temp$dropoff_longitude
Frequency
cab_temp$dropoff_longitude

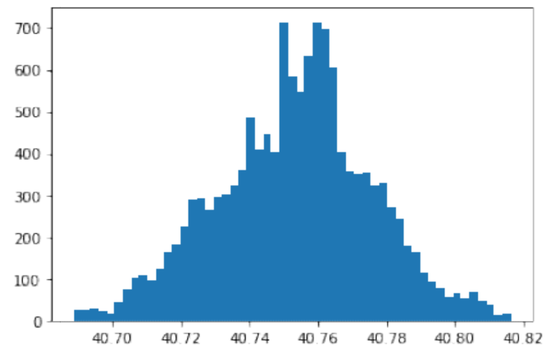
Fig 2.9: Distribution of Independent Continuous variables in Python



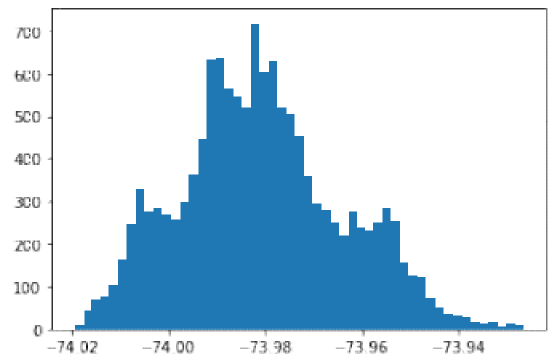
Distribution of pickup_longitude



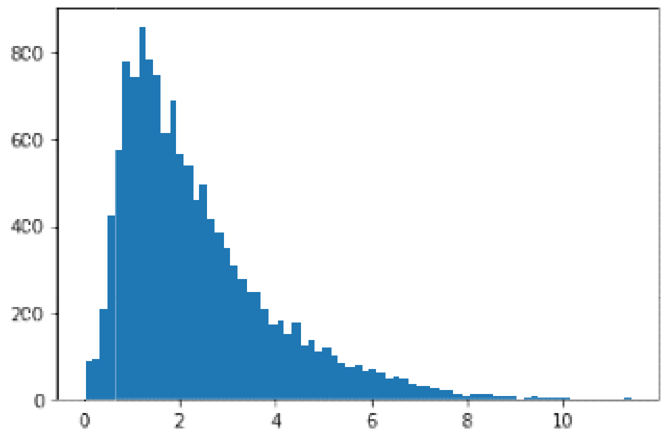
Distribution of pickup_latitude



Distribution of dropoff_latitude



Distribution of dropoff_longitude



Distribution of Distance

The continuous independent variables are not normally distributed, thus normalisation is used for feature scaling.

The normalised new value is given by the following:

$$Value_{new} = \frac{Value - minValue}{maxValue - minValue}$$

In python, dummy variables for each category of the categorical variables is created, on which we apply the linear regression model by using the LinearRegression() function from the sklearn library. This calculates the weight of each category of the categorical variables.

Creating dummy variables is not required in R, as the lm() function is used which itself calculates the weights of all the categories of the categorical variables separately.

In linear Regression**, R-squared** signifies the "percentage variation in target variable that is explained by independent variables". This statistic has a drawback, it increases with the number of predictors. Therefore, it becomes inconclusive in case when it is to be decided whether additional variable is adding to the predictability power of the regression. **Adj. R-squared** is the modified version of R-squared which is adjusted for the number of variables in the regression. It increases only when an additional variable adds to the explanatory power to the regression.

Linear Regression in R

R-squared: 0.6937
Adjusted R-squared: 0.6922

As can be seen by the Adjusted R-squared value, the model explains 69.22% of the variance of the target variable. It is expected that an optimum linear regression model explains at least 80% variance. Thus the model in not adequate

Linear Regression in Python

```
R-squared: 0.6934858599889403
Adjusted R-squared: 0.691293341104741
```

As can be seen by the Adjusted R-squared value, the model explains 69.12% of the variance of the target variable. It is expected that an optimum linear regression model explains at least 80% variance. Thus the model in not adequate

2.2.3. Decision Tree

A decision tree is a predictive model based on branching series of Boolean Tests and can be used for classification and regression problems. It has a flow chart type structure. To build the model, the CART (Classification and Regression Tree) algorithm is used. Here the nodes have a binary split.
The R implementation of CART is called as RPART(Recursive Partitioning And Regression Trees) available in the package with the same name. CART implementation for regression problem in python has been done using DecisionTreeRegressor from sklearn library.

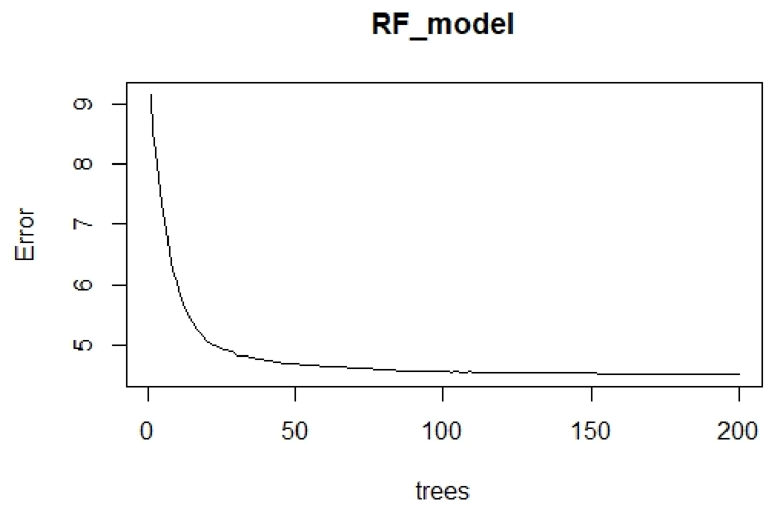
2.2.4. Random Forest

Random Forest is an ensemble technique which works on the bagging concept that includes multiple decision trees. Ensemble techniques use multiple learning algorithms to obtain better predictive performance than could be obtained from any one of the constituent learning algorithms alone. Random forests can be used for both regression and classification problems.

Bagging is a simple ensembling technique in which we build many independent predictors/models/learners and combine them using some model averaging techniques. (eg mode for categorical variable and mean for continuous variables)

While implementing Random Forest we can specify the number of trees to be created. Generally, the more the number of trees the better is the accuracy. If the number of trees to be created is not defined, decision trees are created and checked for error. The number of trees are created till the time the error no longer reduces. In the below plot it can be seen that error reduces no further after 200 trees. Thus we consider 200 trees in the Random Forest.

Fig 2.10: Plot of Random Forest in R



In R, randomForest() function from the library of the same name is used. In Python, RandomForestRegressor from the sklearn library is used.

2.2.5. Gradient Boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

In R, we use xgboost() function from the library of the same name to implement the algorithm. In Python we use XGBRegressor() from xgboost library. Both the functions work of numerical data, thus we convert the data to numeric before training the model on it.

3. Conclusion

3.1. Model Evaluation

Many measures of errors or error metrics are used for model evaluation. Below two of them are listed.

1. RMSE:
   Root Mean Square Error is defined as below:

$$RMSE = \sqrt{(\frac{1}{n})\sum_{t=1}^{n} (\mathrm{A_t} - \mathrm{F_t})^2}$$

$$\mathrm{A_t} \rightarrow \text{Actual Value}$$
$$\mathrm{F_t} \rightarrow \text{Predicted Value}$$
$$n \rightarrow \text{Number of observations}$$

2. MAPE:
   Mean Absolute Percentage Error (MAPE), which is defined as follows:

$$MAPE = \left(\frac{1}{\mathrm{n}}\right)\left|\frac{\sum_{t=1}^{n}(\mathrm{A_t} - \mathrm{F_t})}{\mathrm{A_t}}\right| * 100$$

$$\mathrm{A_t} \rightarrow \text{Actual Value}$$
$$\mathrm{F_t} \rightarrow \text{Predicted Value}$$
$$n \rightarrow \text{Number of observations}$$

Thus the lower RMSE and MAPE is desirable while selecting model.

3.2. Model Selection

Table 3.1: Calculated error metrics

	For R		For Python	
Model	RMSE	MAPE	RMSE	MAPE
Linear Regression	2.145203	18.9908	2.138040497135078	19.04744981494062
Decision Tree	2.297001	21.42963	2.3361752540093557	21.98300206836533
Random Forest	2.091435	19.07664	2.068920366401793	18.640213281346256
Gradient Boosting	2.021932	17.53534	2.058325509711939	17.936730188980263

It is seen in R as well as Python the Gradient Boosting model shows the least MAPE and RMSE. The Gradient Boosting model is selected.

The complete training data is used to train the model and then the test data is applied on the model and the output is stored in CSV file and submitted along with the project.

References

- Data Science Career Path Material, edWisor
- Dataquest: https://www.dataquest.io/
- Data to fish: https://datatofish.com/
- Stack Overflow: https://stackoverflow.com/
- Wikipedia: https://www.wikipedia.org/
- Seaborn: https://seaborn.pydata.org/
- Geeks for Geeks: https://www.geeksforgeeks.org/
- ML Review: https://mlreview.com/
- Statistical tools for high-throughput data analysis(STHDA): http://www.sthda.com/english/
- Towards Data Science: https://towardsdatascience.com/