Project Report

Cab Fare Prediction

By Amol Pradhan

November 2019

INDEX

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.

1.2. Data Exploration

We have two datasets,

1. train_cab.csv
2. test.csv

In **train_cab csv** file we have 7 variables

1. **fare_amount** – cab ride money float variable
2. **pickup_datetime** – datetime stamp variable for start time of ride
3. **pickup_latitude** – float variable for latitude coordinates of where the cab ride started
4. **pickup_longitude** – float variable for longitude coordinates of where the cab ride started
5. **dropoff_latitude** – float variable for latitude coordinates of where the cab ride ended
6. **dropoff_longitude** – float variable for longitude coordinates of where the cab ride ended
7. **passenger_count** – integer variable indicating the number of passengers in the cab ride

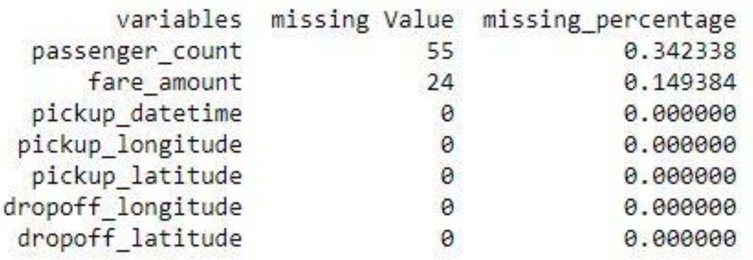
1.3. Defining problem Category

In given problem statement we have to predict fare amount for a cab ride. The fare amount is a continuous integer variable. So, for continuous value prediction we will use Regression problem approach. Our task is to build a Regression model which will predict fare amount based on some attributes gather during the ride.

2. Data Pre-processing

2.1. Missing Value Analysis

In given train dataset there are some missing values present. These missing values can affect the model as it will reduce the precision. These values can be introduced due to human error. So, I done missing value analysis on train dataset to get the amount of missing values. After summing the total no. of NA values present in each variable following is the result I get,



```
        variables  missing Value  missing_percentage
  passenger_count             55            0.342338
      fare_amount             24            0.149384
  pickup_datetime              0            0.000000
 pickup_longitude              0            0.000000
  pickup_latitude              0            0.000000
dropoff_longitude              0            0.000000
 dropoff_latitude              0            0.000000
```

Fig. 2.1. Missing values

From Above table it is seen that passenger_count and fare_amount variable has the missing values. The percentage of missing value is less than 30% we can impute those missing values. To impute those missing values, I used Mean method and median method. To check the accuracy of both methods I done missing value analysis on known data. I replace one target variable with NA and imputed missing value using mean and median method.

In python median method gives best result and in R mean method gives better accuracy.

2.2. Data Cleaning

Before processing further, we summaries the train data describe function. In the result of describe function it is seen that there are some abnormal values present in the dataset like minimum fare_amount is -3 and passenger count is 5345 which is not possible. So, by setting limit to dataset in a such way that it will not lose any important information and removes those anomalies.

2.2.1. removing Datetime stamp variable

In given train dataset we have one variable name 'pickup_datetime' which is a datetime stamp variable. this data is useful for model development because it done not give any meaningful information. So, we divide this data into Year, Month, Day, Weekday, Hour using datetime function present in python and R and remove 'pickup_datetime' variable.

The datatype of these derived variables is integer.

The weekday variables give output values in the range of 0 to 6,

0 = Monday

1 = Tuesday

2 = Wednesday

3 = Thursday

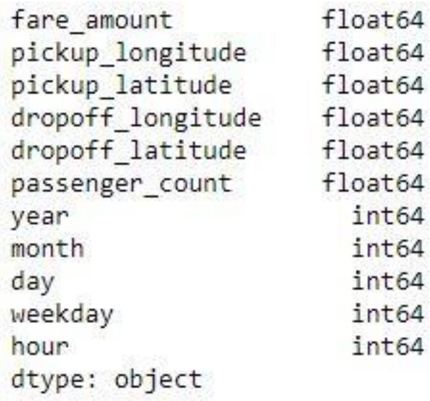
4 = Friday

5 = Saturday

6 = Sunday

2.3. Converting Datatypes

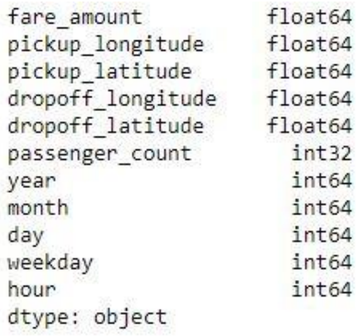
In data pre-processing we need to check datatypes of each variable. After checking all the datatypes, we get following table



```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count      float64
year                   int64
month                  int64
day                    int64
weekday                int64
hour                   int64
dtype: object
```

Fig. Datatypes

The passenger_count variable has datatype float. but, the values in the variable are integer. so, we change the datatype of ‘passenger’ count variable to integer.



```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int32
year                   int64
month                  int64
day                    int64
weekday                int64
hour                   int64
dtype: object
```

Fig. Datatypes

2.4. Outlier Analysis

An outlier is an observation point that is distant from other observation. In layman terms, we can say that an outlier is something which is separated from the crowd. Also, outlier analysis is very important because they affect the mean and median which in turn affects the error in any data set. When we plot the error, we might get big deviations if outlier is in the dataset.

To visualize outliers, we can use box plot method. So, after applying box plot on each variable present in the dataset we get following figures,

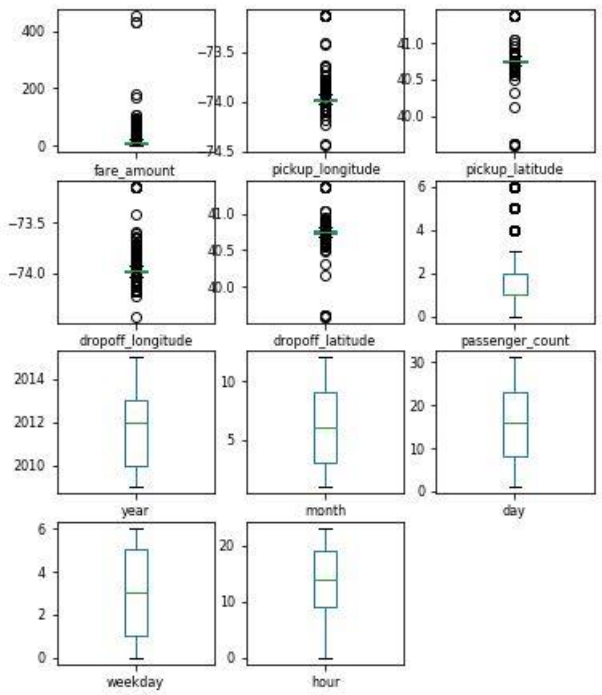


Fig. Box plot

From initial visualisation it is seen that only fare_amount, pickup_longitude, pickup_latitudde, dropoff_longitude, dropoff_latitude and passenger_count variable contains the outliers.

To remove outliers I used following formula,

min = q25 - (iqr*1.5)

max = q75 + (iqr*1.5)

where,

iqr = inter quartile range

q25 = 25th percentile of data

q75 = 75th percentile of data

We define the minimum and maximum range for data. We drop the values that are out of range.

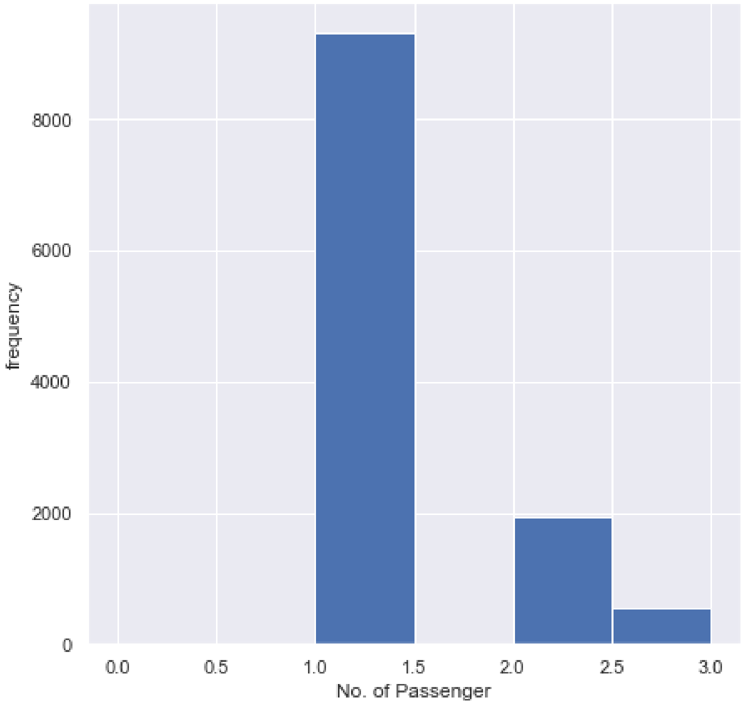
2.5. Data Visualization

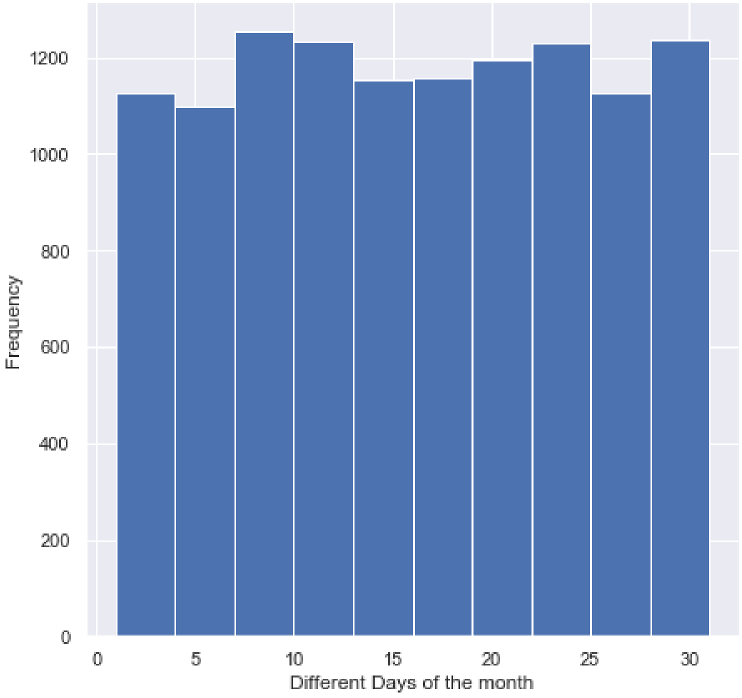
Data visualization helps us to get better insights of the data. By visualising data, we can identify areas that need to attention or improvement and also clarifies which factor influence fare of the cab and how the resource are used to determine it.

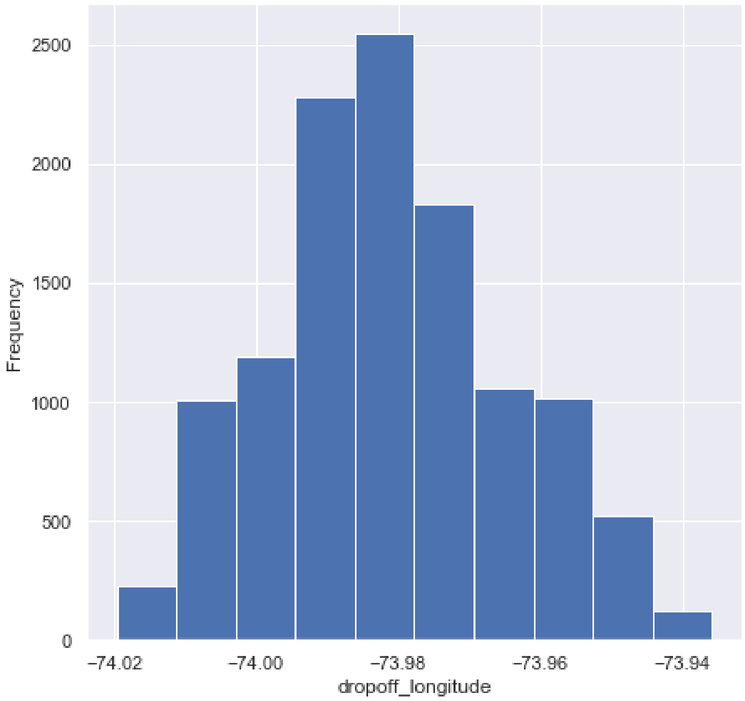
2.5.1. Univariate Analysis

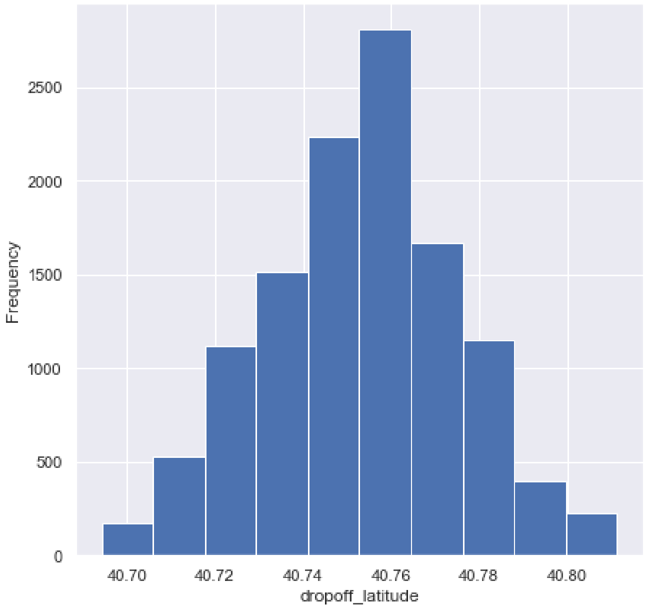
Univariate analysis is the simplest form of the data analysis where the data being analysed contains only one variable. Since it is a single variable it doesn’t deal with cause or relationships. The main purpose of univariate analysis is to describe the data and find patterns that exist within it. For Univariate analysis I used Histogram plot.

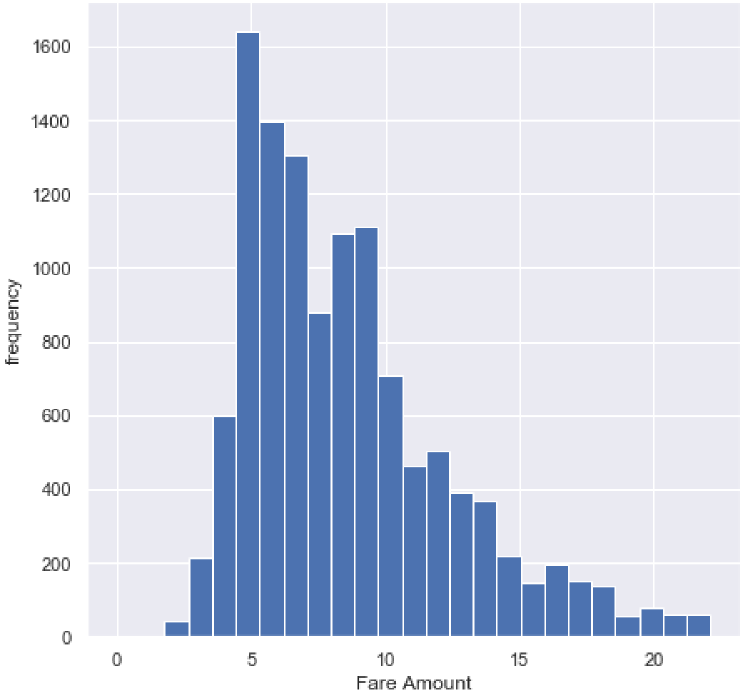
Histogram are constructed by binning the data and counting the number of observations in each bin. The objective of plotting histogram plot is usually to visualise the shape of the distribution. The number of bins need to be large enough to reveal interesting features and small enough to be too noisy.



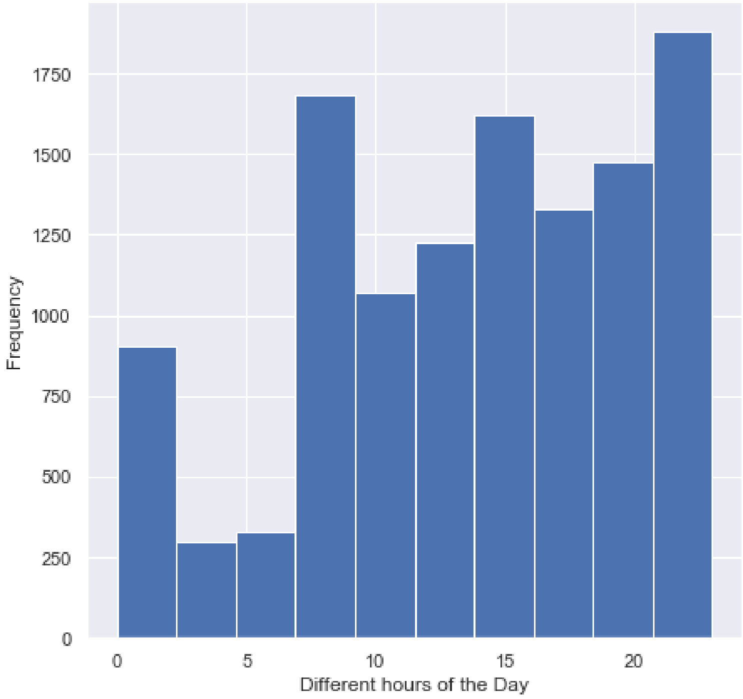



2500
2000
1500
1000
500
0
Frequency
−74.02
−74.00
−73.98
−73.96
−73.94
dropoff_longitude

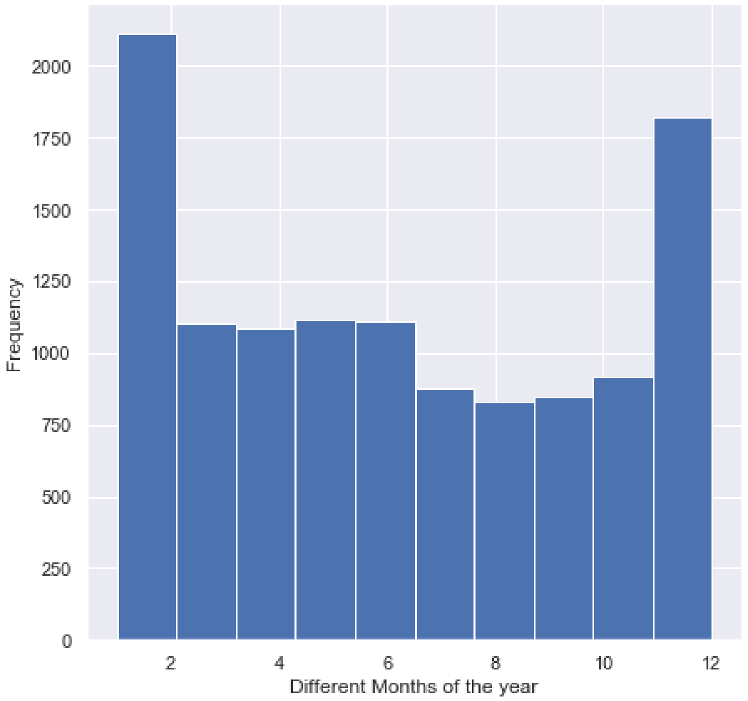

2500
2000
1500
1000
500
0
Frequency
40.70
40.72
40.74
40.76
40.78
40.80
dropoff_latitude

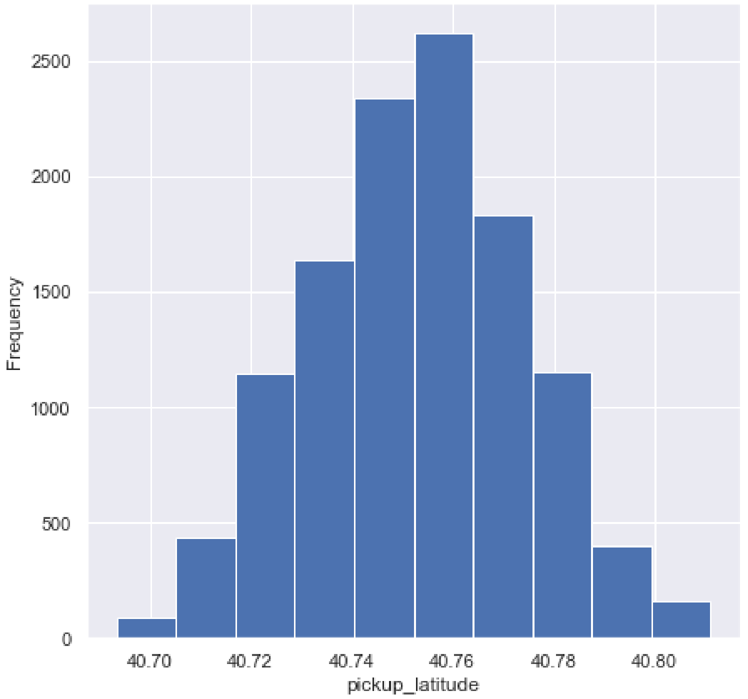


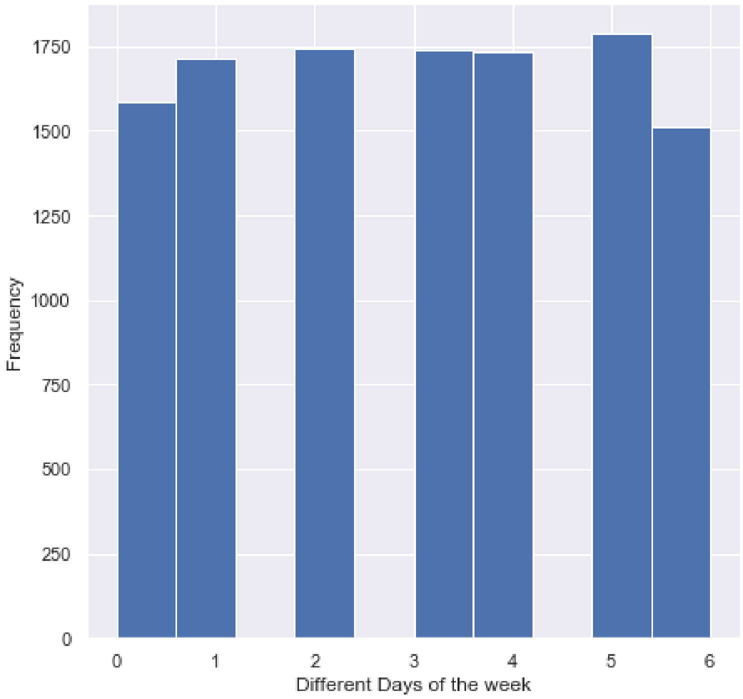
frequency
0
200
400
600
800
1000
1200
1400
1600
0
5
10
15
20
Fare Amount

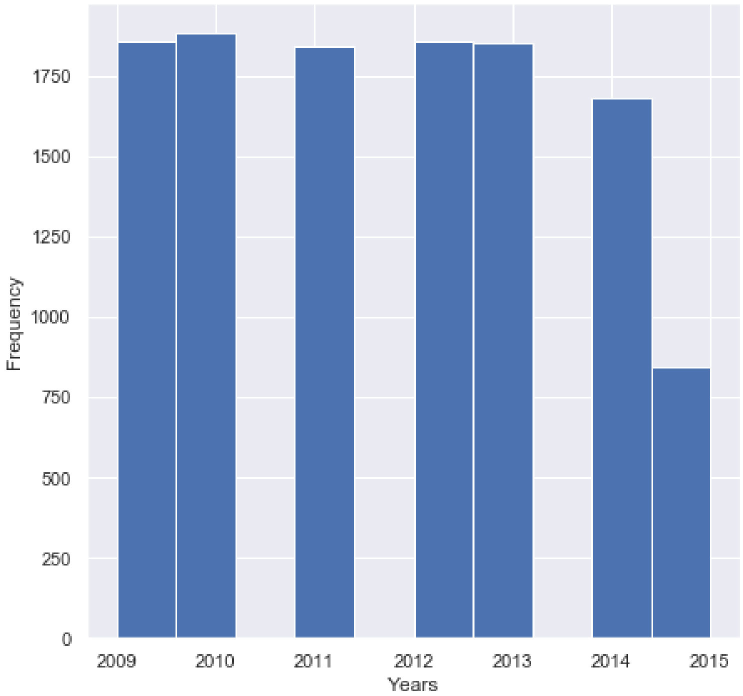


Frequency
0
250
500
750
1000
1250
1500
1750
0
5
10
15
20
Different hours of the Day


Frequency
0
250
500
750
1000
1250
1500
1750
2000
2
4
6
8
10
12
Different Months of the year


Frequency
0
500
1000
1500
2000
2500
40.70
40.72
40.74
40.76
40.78
40.80
pickup_latitude





2.5.2. Bivariate Analysis

In bivariate analysis we check the relationship between all variable with target variables. to know the behaviour of data. For bivariate analysis I used scatter plot.

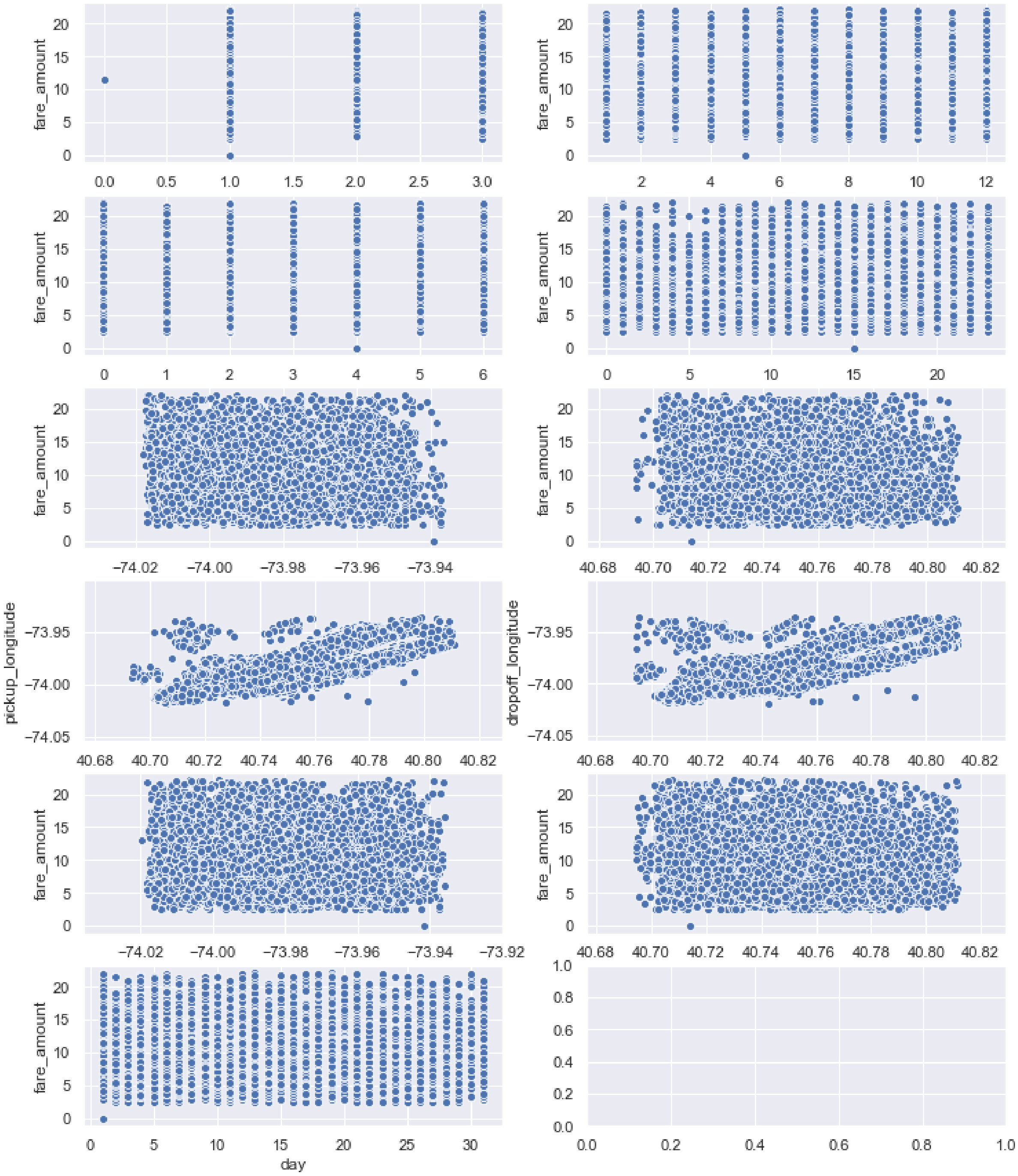


Fig. Scatter plot

2.6. Feature Selection

Feature selection is the process of selecting the useful feature for model training.

2.6.1. Correlation Analysis

If the values of one column to the other column are similar, then it is said to be collinear. Therefore, between predictor variables there should be less collinearity as compared to the collinearity among the predictors and variable.

The value for collinearity is between -1 to 1. So, any value close to -1 or 1 will result in high collinearity.

In our train data the correlation value is not greater than 0.4 and less than -0.1. Therefore, the dataset is free from collinearity. To plot correlation, I used Heatmap plot.

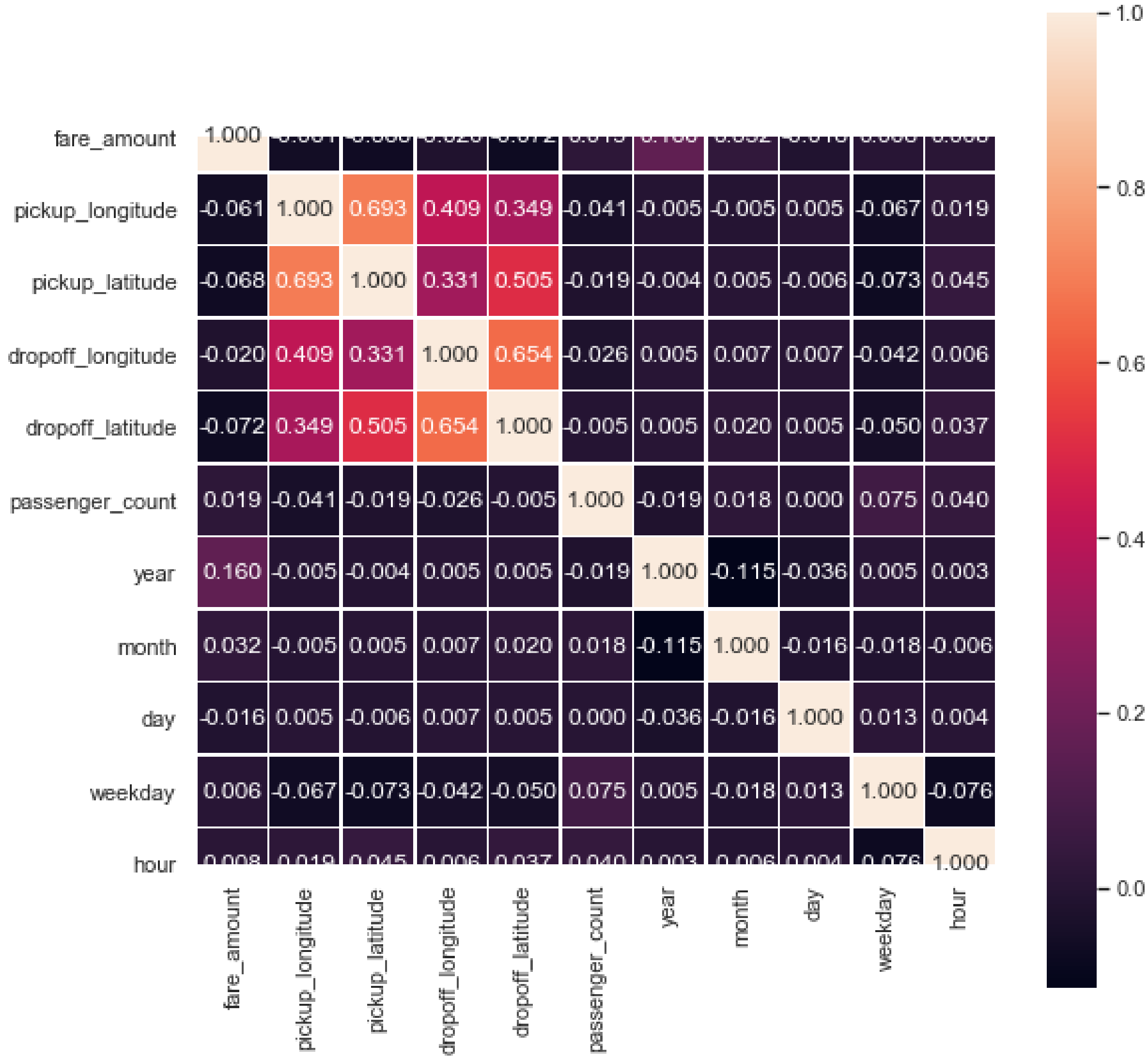


Fig. Correlation plot

2.7. Feature Scaling

Feature scaling is used to make the variables in dataset scale free for the ease of model development. If the scale between two variables is high, then there is chance that our model will be biased towards large scale variable.

For feature scaling there ae two methods mostly used,

1. Normalization
   a. Used when the data is normally distributed.
2. Standardization

a. Used when the data is not normally distributed.

After feature scaling the variable ranges from 0 to 1.

In data visualization it is seen that our data is normally distributed, So we will use Normalization method for feature scaling.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Where X is value to be scaled.

3. Model Development

In model development we have to create a model which will predict fare_amount based on the data given to models. This prediction is a continuous variable. For a continuous variable we can use various regression models. The model having less error rate and more accuracy will be our final model.

Models used for predictions are:

- Linear Regression
- KNN
- Decision Tree
- Random Forest

Before training our model, we will split our data into train and test data. Here I have taken 80% pf the total data as training data.

3.1. Performance Metrics

3.1.1. RMSE:

Root Mean Square Error (RMSE) is the standard deviation of the residuals. Residuals are the measure of how far from the regression line data points are. RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Also, since errors are squared before they are averaged, the RMSE gives relatively high weight to the large errors.

3.2. Modelling

For predictive model development we used multiple algorithm. The algorithm used are start from simple to complex.

3.2.1. Linear Regression

Linear Regression is most commonly used algorithm. Multiple linear Regression is a method used to model the linear relationship between a dependent variable (Target) and one or more independent variables. Multiple Linear Regression is based on ordinary least squares (OLS), the model is fit such that the sum of square of differences of observed and predicated values is minimized. The

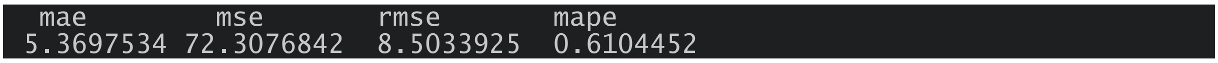
MLR model is based on the several assumptions (e.g. Errors are normally distributed with zero mean and constant variance)

After Training with Linear Regression model, I get following performance metrics

In Python:

```
MAE is: 2.973392366538636
MAPE is: 0.41842477657562493
MSE:  14.442326262657845
RMSE:  3.800306074865266
```

In R:



```
 mae        mse        rmse      mape
5.3697534 72.3076842 8.5033925 0.6104452
```

3.2.2. KNN

KNN stands for K Nearest Neighbour. KNN is the simplest algorithm that stores all available cases and predict the numerical target based on a similarity measure (e.g. distance function). KNN has been used in statistical estimation and pattern recognition as no parametric technique. A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbors. Another approach uses an inverse distance weighted average of the K nearest neighbors. KNN regression uses same distance function as KNN Classification.

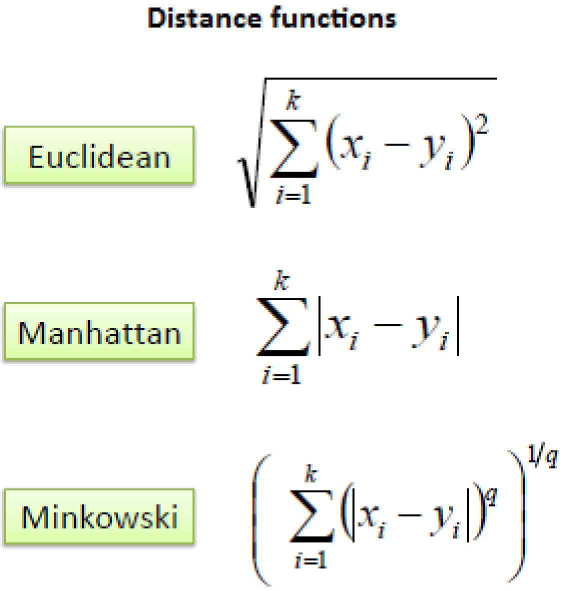

Distance functions
Euclidean
$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$
Manhattan
$\sum_{i=1}^{k}|x_i - y_i|$
Minkowski
$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Fig. Distance Function

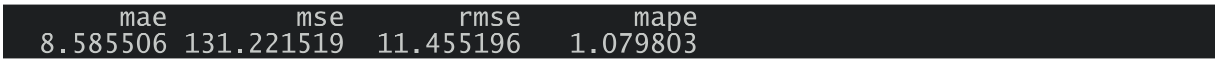
Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise. However, the compromise is that the distinct boundaries within the feature space are blurred. Cross validation is another way to retrospectively determine a good K value by using an independent dataset to validate your K value. The optimal K for most dataset is 10. That produces much better result than 1-KNN.

After Training KNN model, I get following performance metrics

In Python:

```
MAE is: 2.452239522175146
MAPE is: 0.3198996671035521
MSE:  10.642973863374007
RMSE:  3.262357102368471
```

In R:



```
       mae        mse       rmse       mape
  8.585506 131.221519  11.455196   1.079803
```

3.2.3. Decision Tree

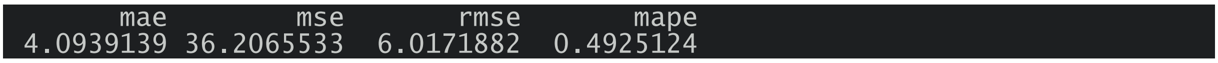
Decision tree builds regression models in the form of tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor are called root node. Decision trees can handle both categorical and numerical data.

After training Decision Tree, I get following performance metrics

In python:

```
MAE is: 2.279578587009114
MAPE is: 0.2959167425706746
MSE:  9.514700357563429
RMSE:  3.0845907925628366
```

In R:



```
       mae        mse       rmse       mape
 4.0939139 36.2065533  6.0171882  0.4925124
```

3.2.4. Random Forest

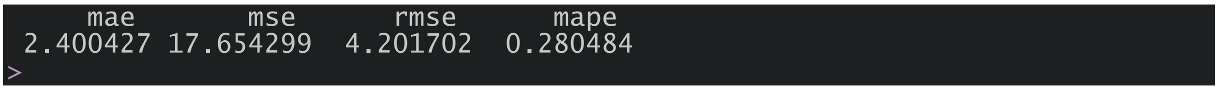
A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging. The basic behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

After training Random forest model, I get following performance metrics

In Python:

```
MAE is: 2.1851265754476654
MAPE is: 0.3141818897011347
MSE:  8.074809707030763
RMSE:  2.8416209647014434
```

In R



```
      mae         mse        rmse        mape
 2.400427 17.654299  4.201702  0.280484
>
```

3.3. Model Evaluation

Now we have 4 models for predicting the target variable, we need to decide which model to choose. There are several criteria that exist for evaluating and comparing models. We can compare the models using any of the following criteria.

1. Predictive Performance
2. Interpretability
3. Computational Efficiency

In our case, the Interpretability and Computation Efficiency, do not hold much significance. Therefore, we will use predictive performance as the criteria to compare and evaluate models. Predictive performance can be measured by comparing prediction of the model with real values of the target variables and calculating some average error measure.

3.4. Model Selection

From above error metrics it is seen that **random forest** has lowest RMSE value. So, I selected random forest for prediction.

4. Testing of Trained Model

The trained Random forest model can be used to test on new test data. I have one test data containing all the variable except fare_amount. So, I imported test data into environment. After doing some pre-processing on test data we predict the fare_amount data using trained model and save the result to local machine.