CAB FARE PREDICTION

Project Report

Gaurav Parmar

Index

Introduction

1.1 Problem Statement

You are a cab service 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

Understanding of data is the very first and important step in the process of finding solution of any business problem. Here in our case our company has provided a data set with following features, we need to go through each and every variable of it to understand and for better functioning.
Size of Dataset Provided: - 16067 rows, 7 Columns (including dependent variable)
Missing Values: Yes
Outliers Presented: Yes

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

Methodology

➢ Pre-Processing

To build a predictive model, we look and manipulate the data before we start modelling which includes multiple preprocessing steps such as exploring the data, cleaning the data as well as visualizing the data through graph and plots, all these steps is combined under one term i.e. **Exploratory Data Analysis**, which includes following steps:

- Data exploration and Cleaning
- Missing values treatment
- Outlier Analysis
- Feature Selection
- Features Scaling
- Skewness and Log transformation
- Visualization

➢ Modelling

Once all the Pre-Processing steps has been done on our data set, we will now further move to our next step which is modelling. Choice of models depends upon the problem statement and data set. As per our problem statement and dataset, we will try some models on our preprocessed data and post comparing the output results we will select the best suitable model for our problem. As per our data set following models need to be tested:

- Linear regression
- Decision Tree
- Random forest,
- Gradient Boosting

We have also used hyper parameter tunings to check the parameters on which our model runs best. We have used Random Search CV and Grid Search CV

➢ Model Selection

The final step of our methodology will be the selection of the model based on the different output and results shown by different models. We have multiple parameters which we will study further in our report to test whether the model is suitable for our problem statement or not.

Pre-Processing

3.1 Data exploration and Cleaning (Missing Values and Outliers)

The very first step which comes with any data science project is data exploration and cleaning which includes following points as per this project:

a. As we know we have some negative values in fare amount so we have to remove those values.

b. Passenger count would be max 6 if it is a SUV vehicle not more than that. We have to remove the rows having passengers counts more than 6 and less than 1.

c. There are some outlier figures in the fare (values > 400) so we need to remove those.

d. Latitudes range from -90 to 90. Longitudes range from -180 to 180. We need to remove the rows if any latitude and longitude lies beyond the ranges.

e. Null values in fare and distance were imputed using formula-

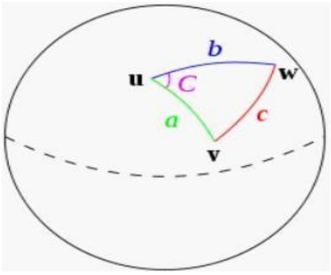
distance = (fare_amount - 2.5)/1.56

3.2 Creating some new variables from the given variables.

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

- Year
- Month
- Date
- Day of Week
- Hour

Also, we tried to find out the distance using the haversine formula which says:
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.



So, our new extracted variables are:

- fare_amount
- pickup_datetime
- pickup_longitude
- pickup_latitude
- dropoff_longitude
- dropoff_latitude
- passenger_count
- year
- month
- date
- day_of_week
- hour
- haversine_distance

3.3 Selection of variables

Since we have extracted values from the important columns, we will drop the redundant variables:

- pickup_datetime
- pickup_longitude
- pickup_latitude
- dropoff_longitude
- dropoff_latitude

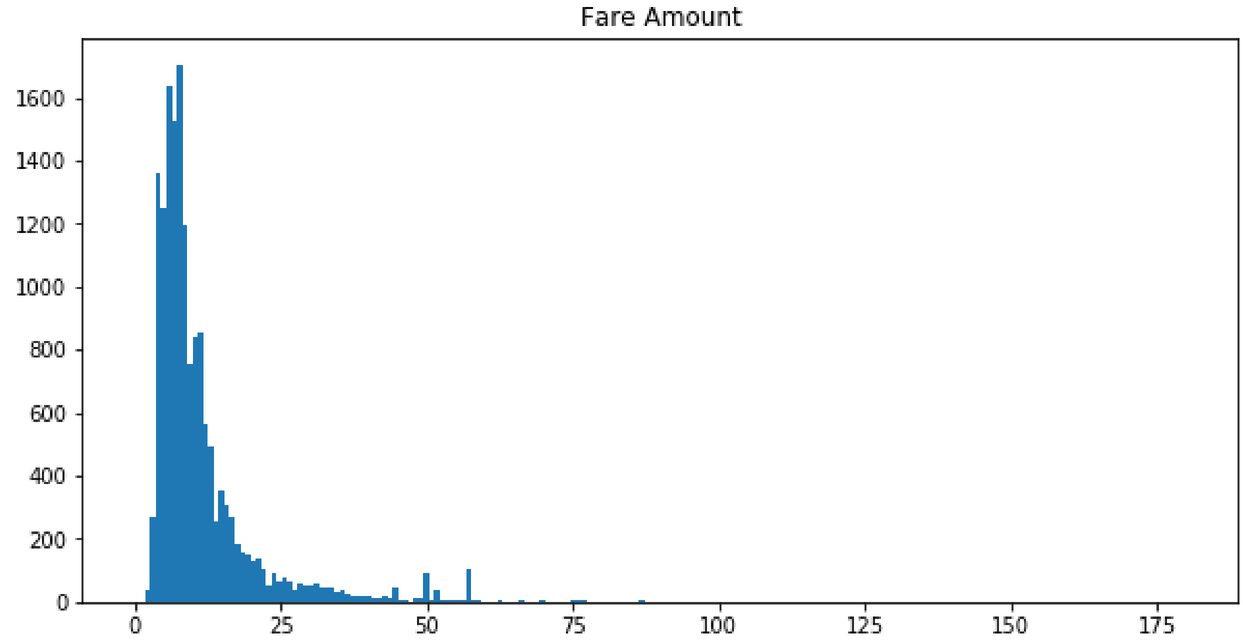
Now only following variables we will use for further steps:

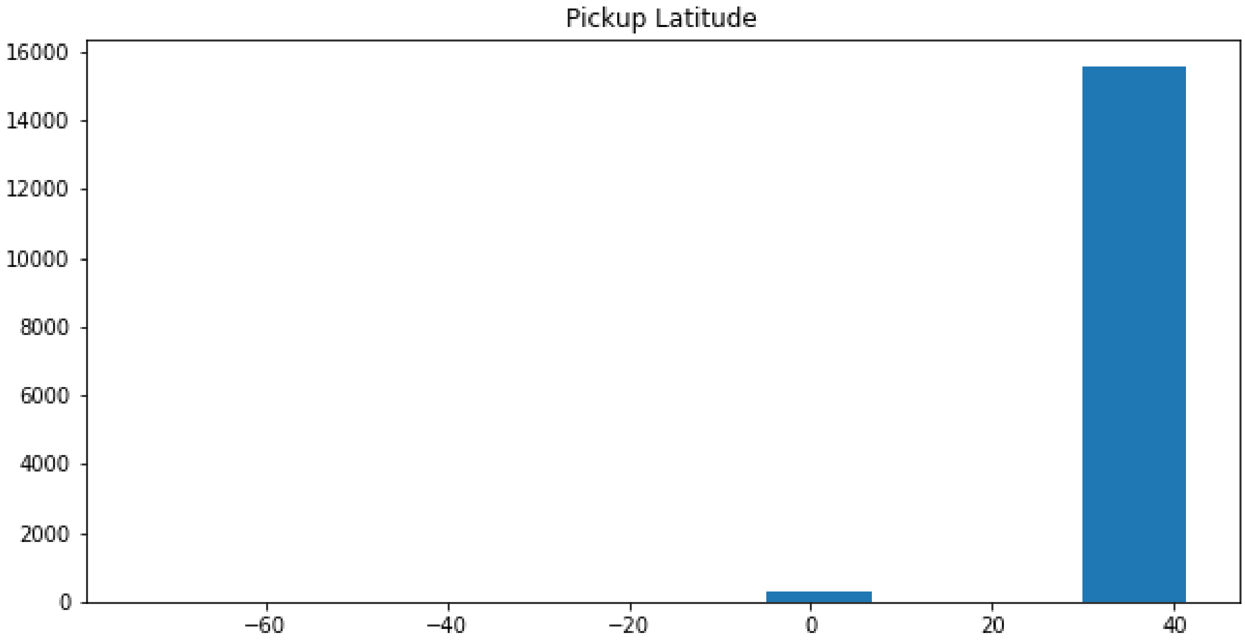
Variable Names	Variable Data Types
fare_amount	float64
passenger_count	int64
year	object
month	object
date	object
day_of_week	object
hour	object
haversine_distance	float64

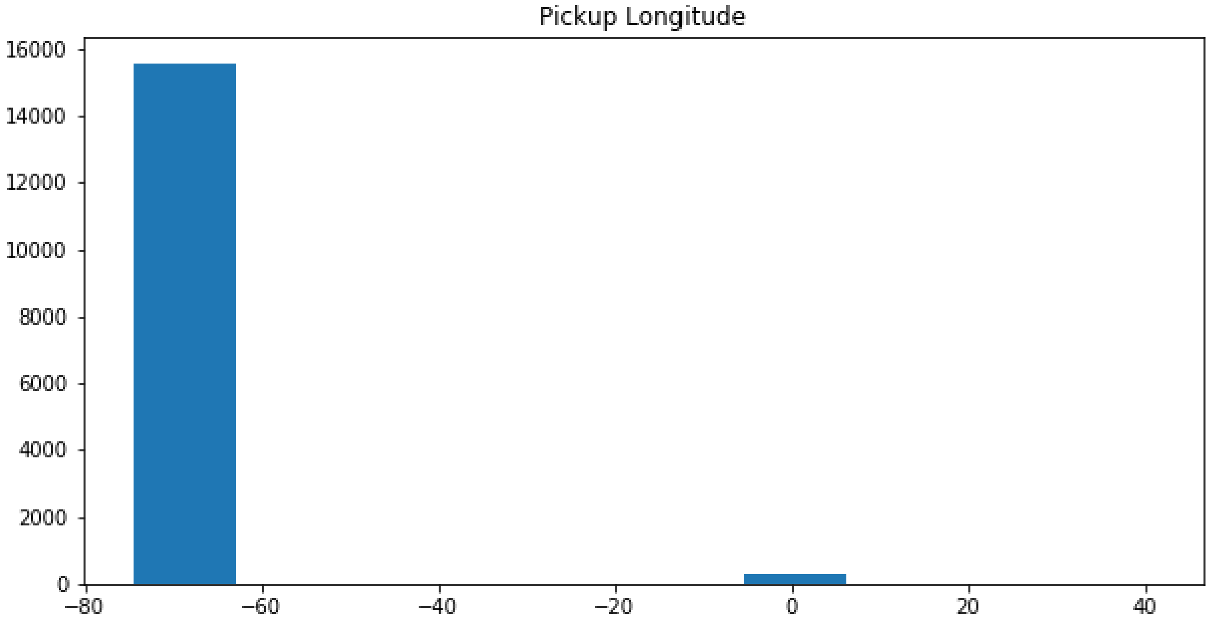
3.4 Data Visualization

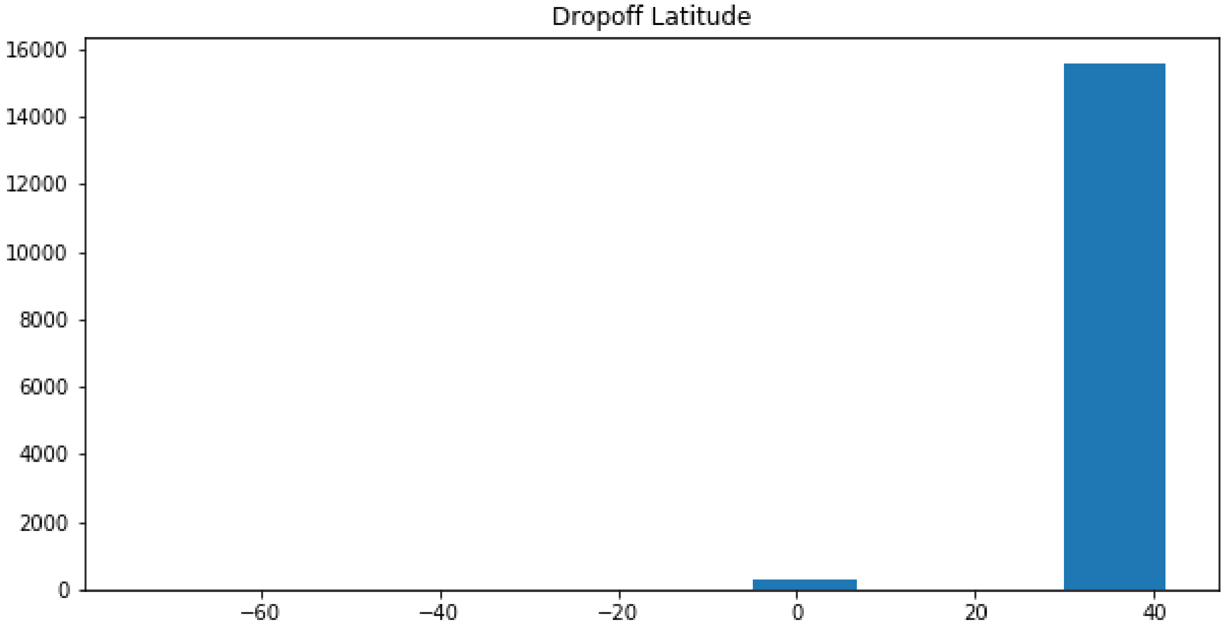
3.4.1 Univariate Analysis

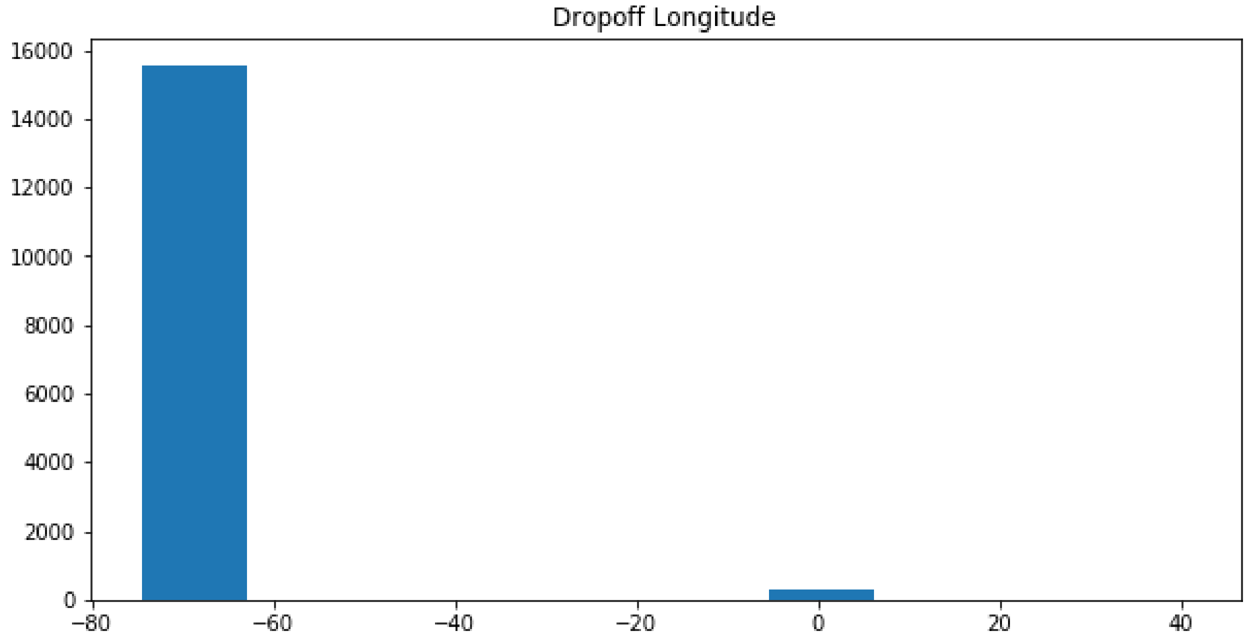
In this phase we do visual analysis of a single variable. We check for maximum range of data and its skewness.



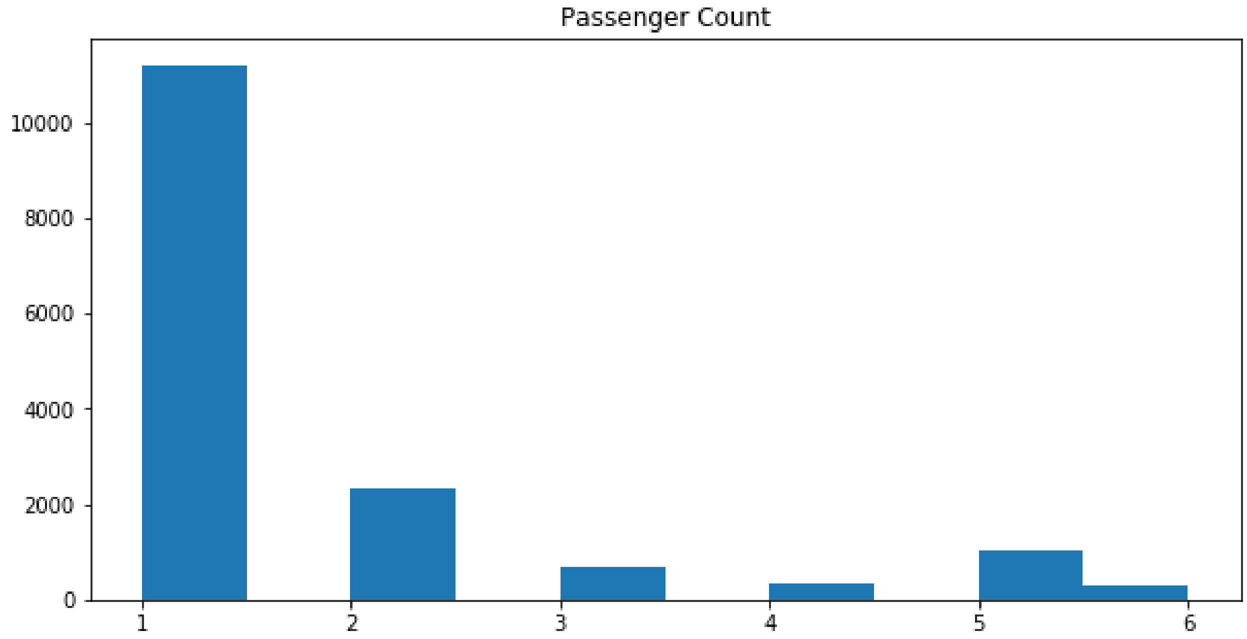



Pickup Longitude

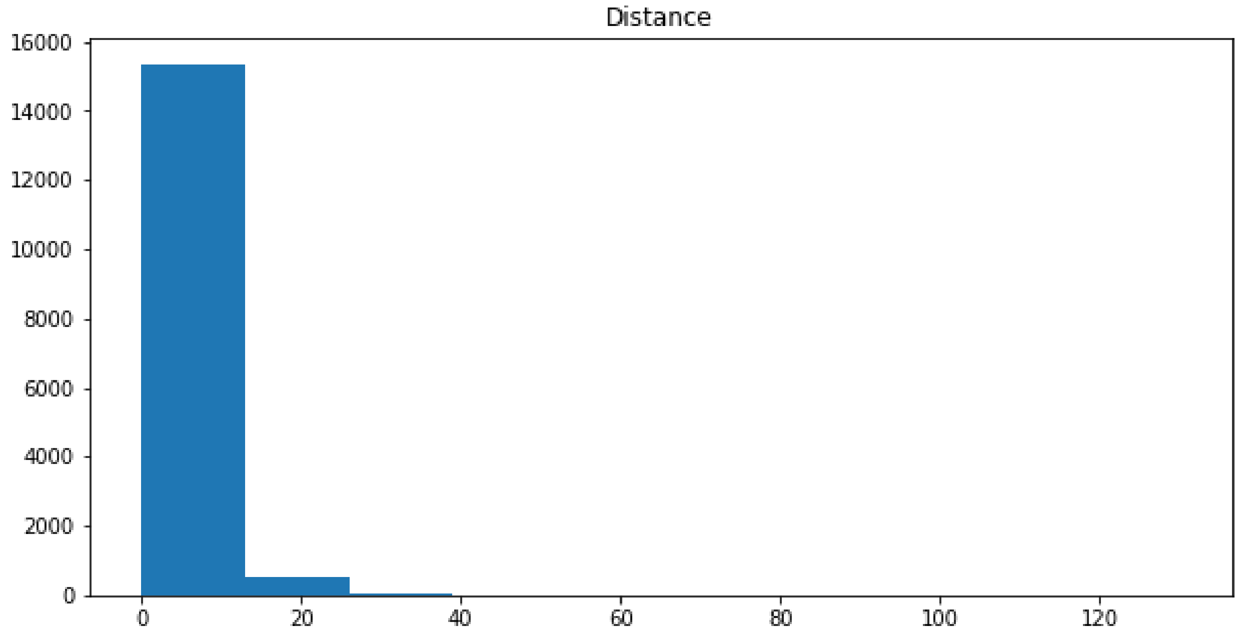

Dropoff Latitude



Dropoff Longitude

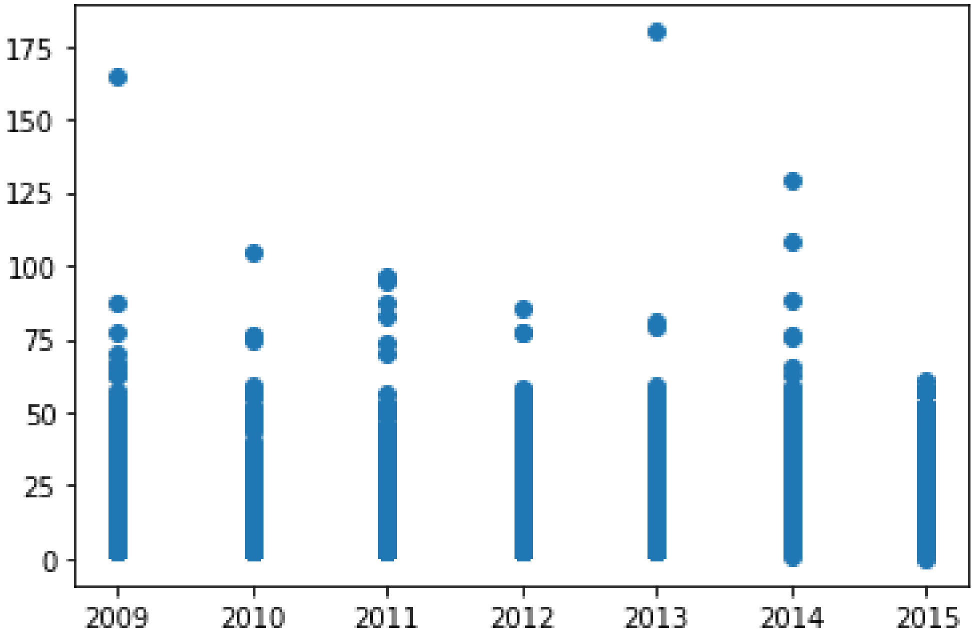


Passenger Count



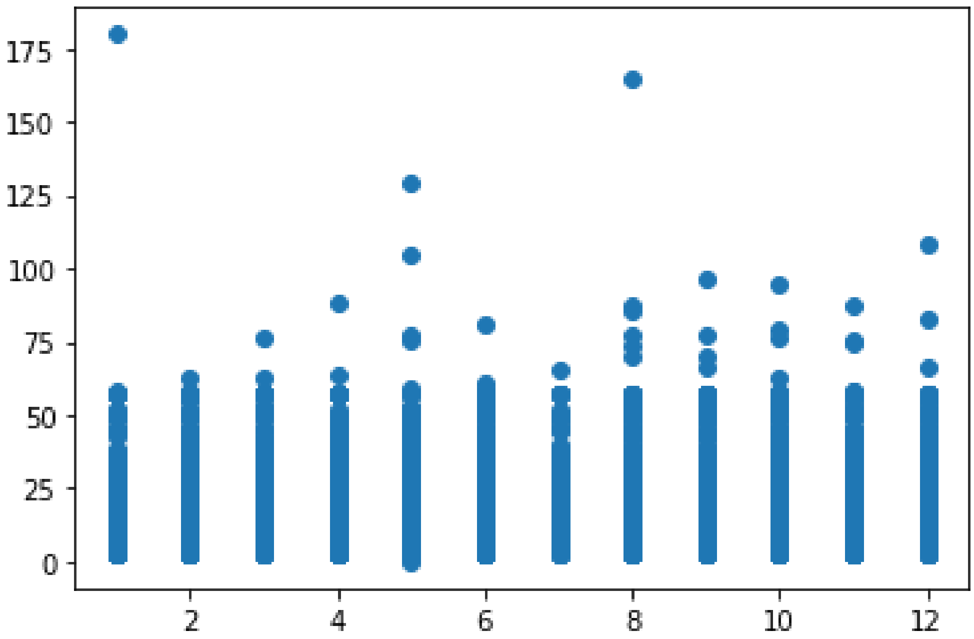
3.4.2 Bivariate Analysis

In this phase, we will be visualizing two variables (one dependent and one independent) and check if there exists any correlation between them.



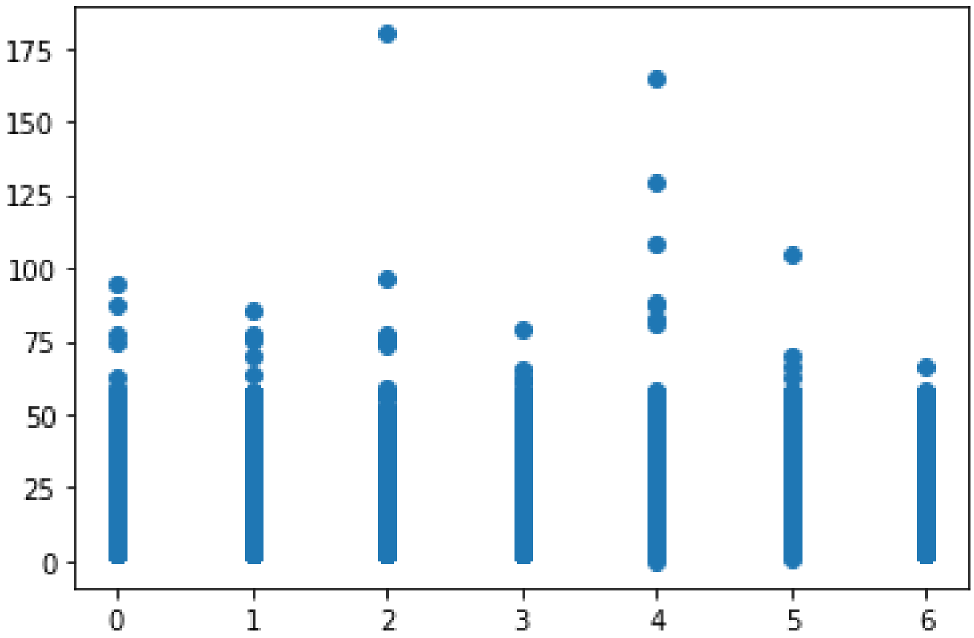
Year on X-axis and Fare Amount on Y-axis

(Based on the scatterplot, in year 2009, 2013, 2014 there were rides which got high fare_amount and very low in year 2015)



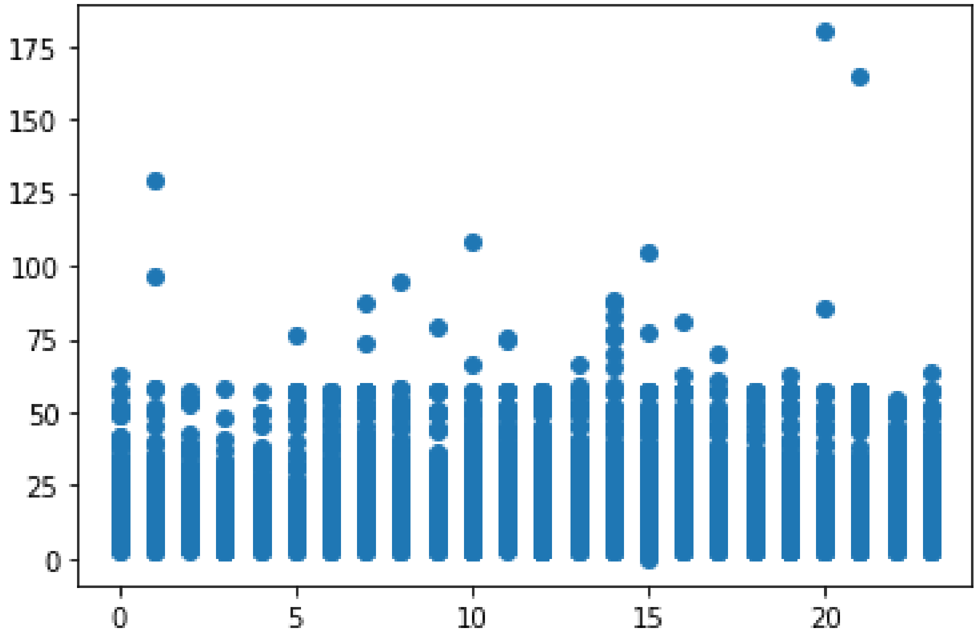
Month on X-axis and Fare Amount on Y-axis

(Based on the scatterplot, we can see Jan month fare amount is very high and low in July)



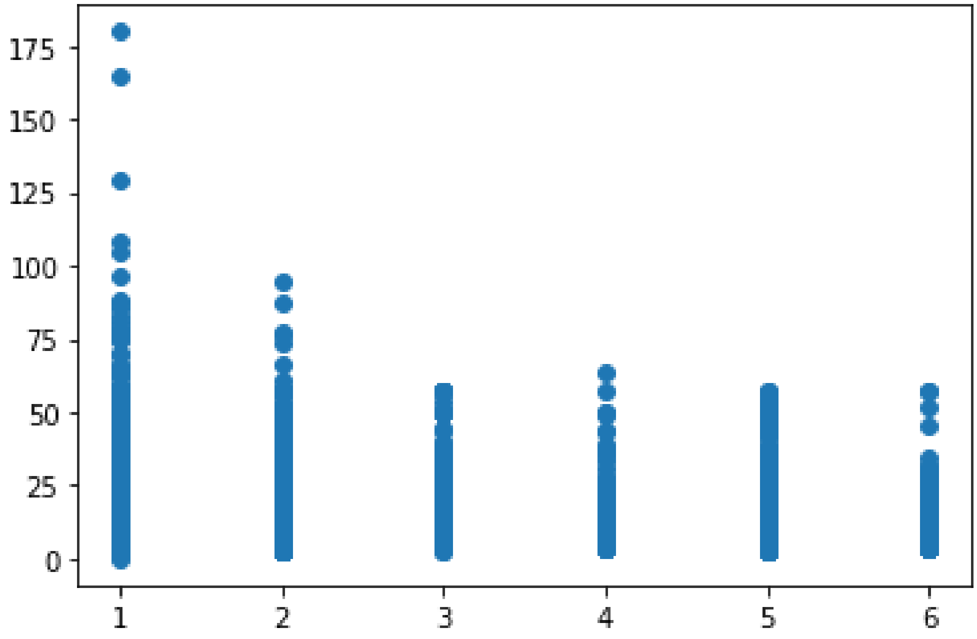
Day of week on X-axis and Fare Amount on Y-axis

(Based on the scatterplot, we can see that Tuesday and Thursday rides have high fare_amount)



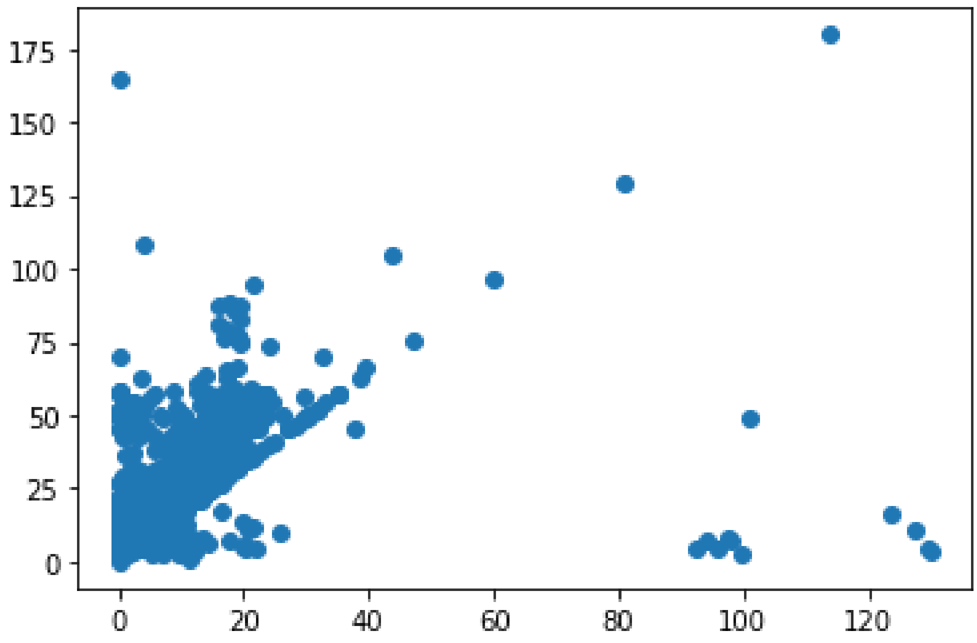
Hour on X-axis and Fare Amount on Y-axis

(Rides taken during 8 pm to 9 pm gives highest fare_amount)



Passenger count on X-axis and Fare on Y-axis

(Based on the above graph, maximum number of travelling has been done by single passengers)



Distance on X-axis and Fare on Y-axis

(we can see as the distance increases fare amount also increases)

3.5 Feature Selection:

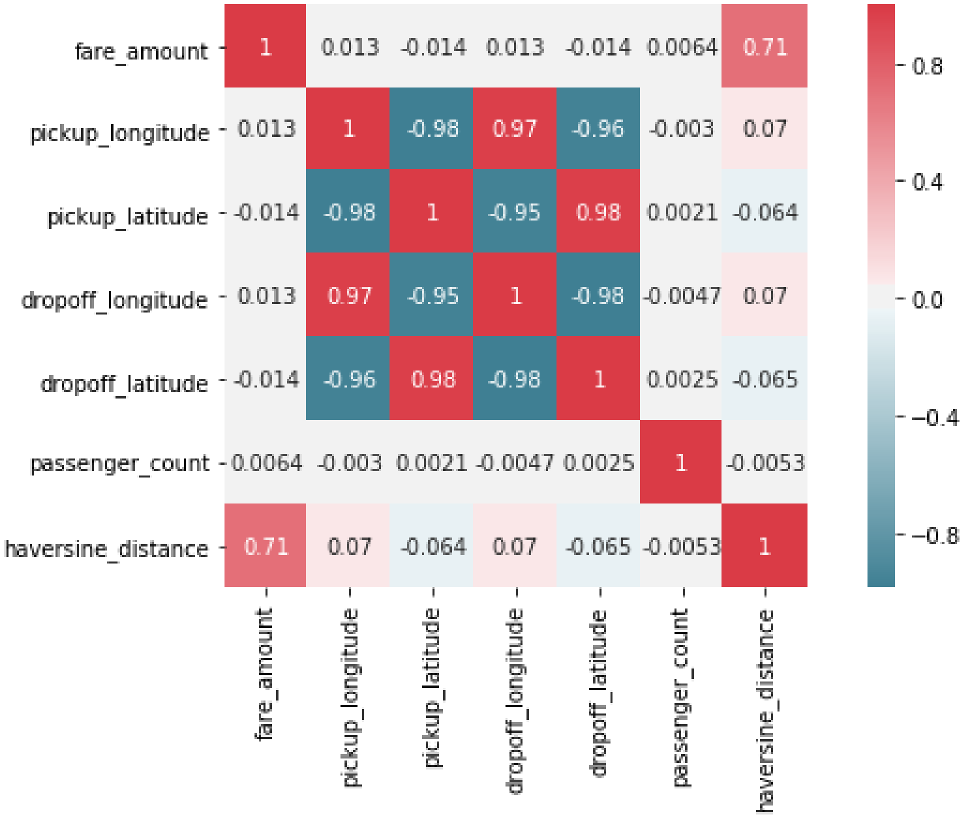
Continuous variables - 'fare_amount', 'pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude', 'passenger_count', 'haversine_distance'

Categorical Variables - ' 'year', 'month', 'date', 'day_of_week', 'hour'

Correlation Analysis –

We perform correlation analysis, to check which independent variable has higher impact on dependent variable. Only those variables which have an effect on target variable will be chosen.

For this, we plot a correlation matrix from seaborn library. As we can see distance has the highest impact on fare amount (target variable), we will be keeping it. Also, passenger count also affects fare amount. So, passenger count variable will be kept as well and others will be dropped.



Anova Test –

Anova Test is performed between categorical independent variables & fare_amount (continuous target variable).

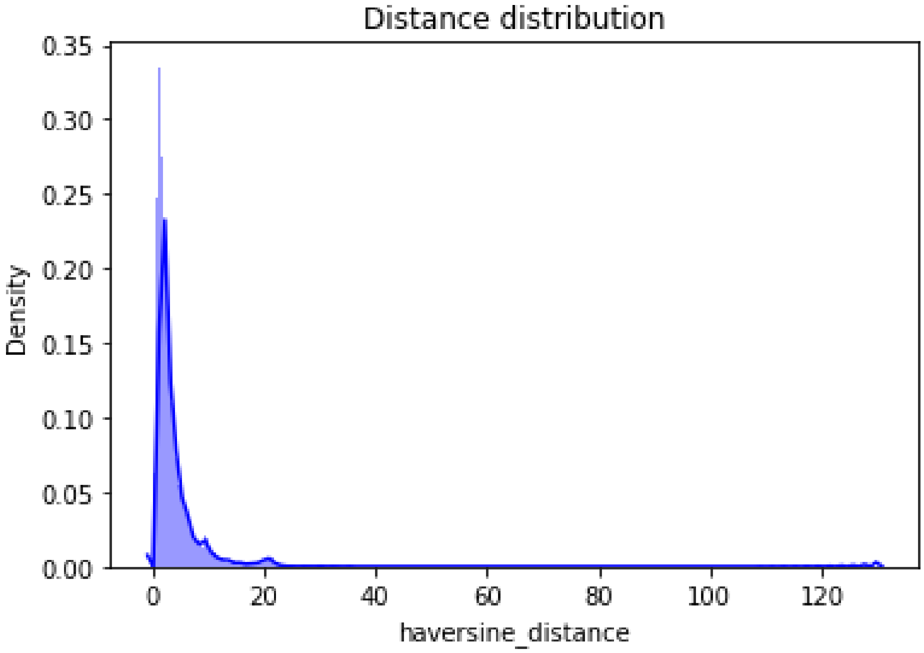
```
              sum_sq        df          F        PR(>F)
year       2.154747e+04   6.0      38.958425  2.762441e-47
Residual   1.462551e+06   15866.0      NaN         NaN
              sum_sq        df          F        PR(>F)
month      4.150925e+03   11.0      4.044236    0.000006
Residual   1.479947e+06   15861.0      NaN        NaN
              sum_sq        df         F     PR(>F)
date       1.822500e+03     30.0   0.649273   0.929301
Residual   1.482276e+06  15842.0      NaN        NaN
              sum_sq        df         F     PR(>F)
day_of_week  6.995622e+02   6.0   1.247052   0.278602
Residual     1.483399e+06   15866.0   NaN        NaN
              sum_sq        df         F         PR(>F)
hour       8.883680e+03     23.0   4.149652   9.308214e-11
Residual   1.475215e+06  15849.0      NaN            NaN
```

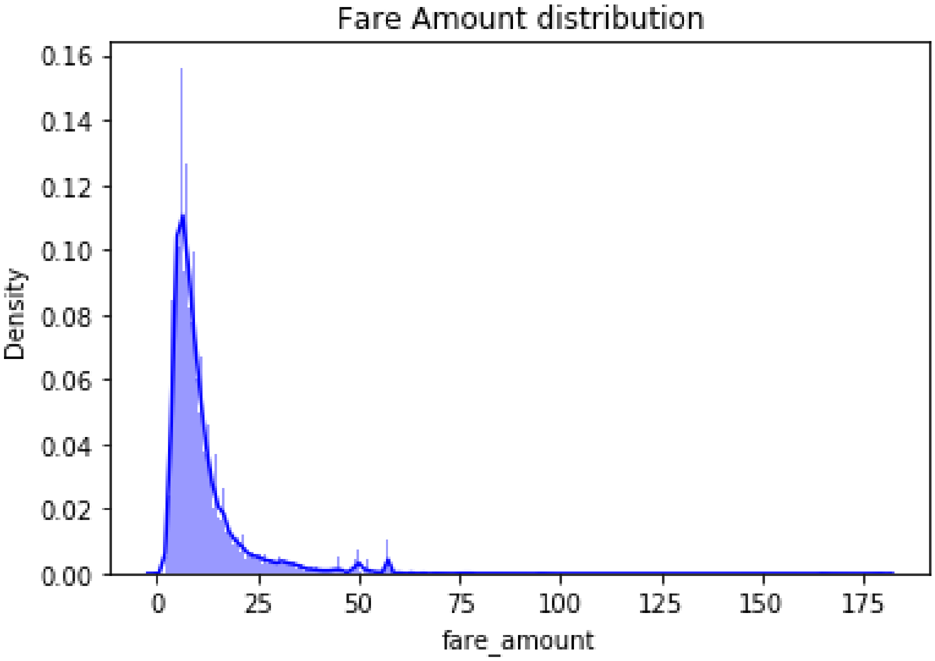
From the anova result, we can observe date, day_of_week has p value > 0.05, so delete these variables and should not be considered in model.

3.6 Feature Scaling

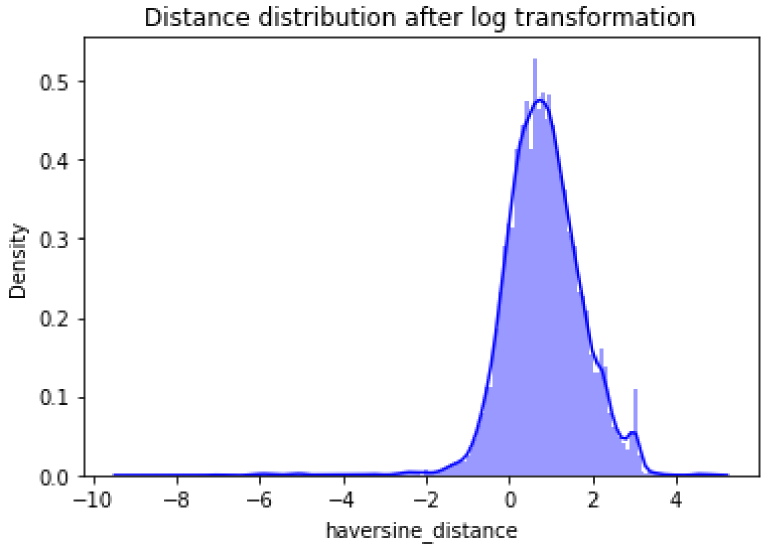
Skewness is asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right. Skewness can be quantified to define the extent to which a distribution differs from a normal distribution. Here we tried to show the skewness of our variables and we find that our target variable absenteeism in hours having is one sided skewed so by using **log transform** technique we tried to reduce the skewness of the same.

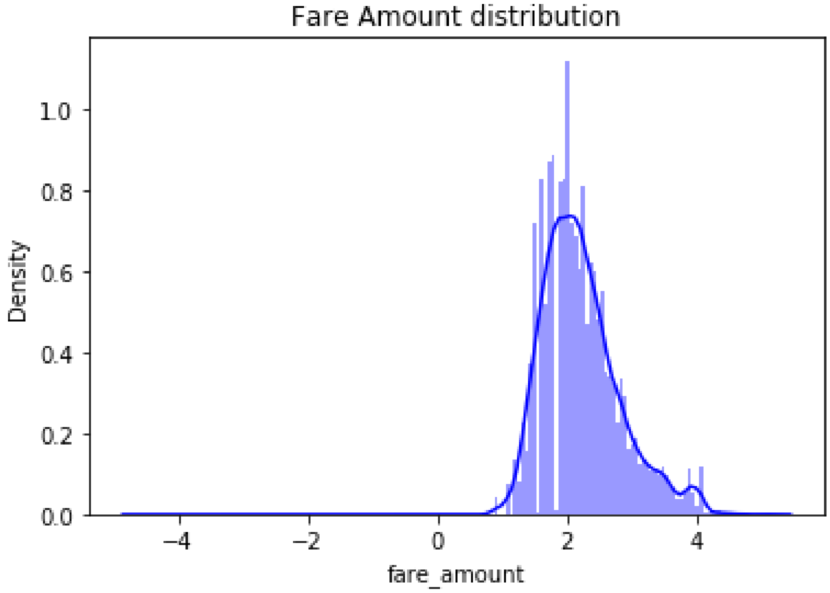
Below mentioned graphs shows the probability distribution plot to check distribution before log transformation:





Below mentioned graphs shows the probability distribution plot to check distribution after log transformation:





As our continuous variables appears to be normally distributed so we don’t need to use feature scaling techniques like normalization and standardization for the same.

Chapter 4

Modelling

After a thorough preprocessing, we will use some regression models on our processed data to predict the target variable. Following are the models which we have built –

- Linear Regression
- Decision Tree
- Random Forest
- Gradient Boosting

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 80% of the data as our train data.

4.1 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).

4.2 Decision Tree

A tree has many analogies in real life, and turns out that it has influenced a wide area of machine learning, covering both classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions.

4.3 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other task, that operate 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. Random decision forests correct for decision trees' habit of overfitting to their training set.

To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

4.4 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 like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

4.5 Hyper Parameters Tunings for optimizing the results

Model hyperparameters are set by the data scientist ahead of training and control implementation aspects of the model. The weights learned during training of a linear regression model are parameters while the number of trees in a random forest is a model hyperparameter because this is set by the data scientist. Hyperparameters can be thought of as model settings. These settings need to be tuned for each problem because the best model hyperparameters for one particular dataset will not be the best across all datasets. The process of hyperparameter tuning (also called hyperparameter optimization) means finding the combination of hyperparameter values for a machine learning model that performs the best - as measured on a validation dataset - for a problem.
Here we have used two hyper parameters tuning techniques

- Random Search CV
- Grid Search CV

1. **Random Search CV**: This algorithm set up a grid of hyperparameter values and select random combinations to train the model and score. The number of search iterations is set based on time/resources.
2. **Grid Search CV**: This algorithm set up a grid of hyperparameter values and for each combination, train a model and score on the validation data. In this approach, every single combination of hyperparameters values is tried which can be very inefficient.

Chapter 5

Conclusion

5.1 Model Evaluation

The main concept of looking at what is called residuals or difference between our predictions f(x[I,]) and actual outcomes y[i].

In general, most data scientists use two methods to evaluate the performance of the model:

I. **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.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

II. **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.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

III. We have shown both train and test data results, the main reason behind showing both the results is to check whether our data is overfitted or not.

Below table shows the model results before applying hyper tuning:

Model Name	RMSE	R Squared
Linear Regression	0.39	0.57
Decision Tree	0.29	0.76
Random Forest	0.30	0.75
Gradient Boosting	0.28	0.78

Below table shows results post using hyper parameter tuning techniques:

Model Name	Parameter	RMSE	R Squared
Random Search CV	Random Forest	0.31	0.74
	Gradient Boosting	0.33	0.7
Grid Search CV	Random Forest	0.29	0.76
	Gradient Boosting	0.29	0.75

Above table shows the results after tuning the parameters of our two best suited models i.e. Random Forest and Gradient Boosting.

For tuning the parameters, we have used Random Search CV and Grid Search CV under which we have given the range of n_estimators, depth and CV folds.

5.2 Model Selection

On the basis RMSE and R Squared results a good model should have least RMSE and max R Squared value. So, from above tables we can see:

- From the observation of all RMSE Value and R-Squared Value we have concluded that,
- Both the models- Gradient Boosting Default and Random Forest perform comparatively well while comparing their RMSE and R-Squared value.
- After this, I chose Random Forest CV and Grid Search CV to apply cross validation technique and see changes brought about by that.
- After applying tunings Random forest model shows best results compared to gradient boosting.
- So finally, we can say that Random forest model is the best method to make prediction for this project with highest explained variance of the target variables and lowest error chances with parameter tuning technique Grid Search CV.

Finally, I used this method to predict the target variable for the test data file shared in the problem statement. Results that I found are attached with my submissions.