Taxi Fare Prediction

Author: Mayank Juneja

INDEX

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.

The objective is to predict the taxi fare amount for the new test case, by anlaysing the given historical data.

1.2 Data Understanding

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

2. DATA PRE-PROCESSING

When we required to build a predictive model, we require to 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 shed which is **Exploratory Data Analysis**, which includes following steps:

- Data Exploration And Cleaning
- Missing Value Analysis
- Outlier Analysis
- Feature Engineering
- Feature Selection
- Feature Scaling

Missing Value Analysis

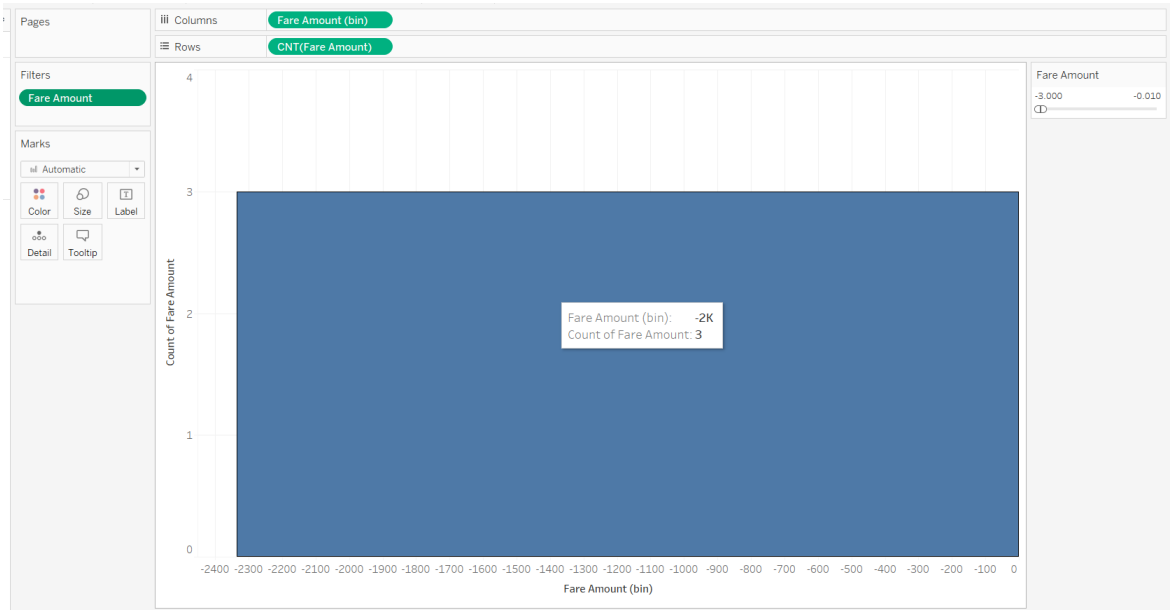
There are few missing values in train dataset.
Nearly 81 missing values, but these missing values are omitted, as they are in very low composition.

```
fare_amount          25
pickup_datetime       1
pickup_longitude      0
pickup_latitude       0
dropoff_longitude     0
dropoff_latitude      0
passenger_count      55
```

Data Exploration And Cleaning

Exploring the data, which means looking at all the features and knowing their characters. According to our problem statement, we are analysing the data of cab rental startup, whose features are comprised of fare amount, passenger count and geometrical coordinates like pickup latitude,longitude and drop off longitude and latitude.
So keeping in mind that these variables play an important role in the analysis, we keep checking the minimum and maximum values in this features.
There were a lot of incorrect information in this feature, like observation showed up passenger count as 0 and 55 , 5000...and few showed decimal values 1.3.

And fare amount ranged from 1 to 400+ and its normal, few showed high as 4000 and 5000...etc , which is not possible as seen in the visualizations below which show many irrelevant values in the variable.



Latitudes range from -90 to 90.Longitudes range from -180 to 180. So Removing those features, which does not satisfy these ranges. Many showed 0 .

So these residual information must be removed from data set, this is known as data cleaning.

Many observations nearly 400 to 600 has this wrong information, so considering this as OUTLIERS in the data and setting them to NA.

Deriving the new features **year, day, date, month, hour** by timestamp variable pickup_datetime.

Count of Data to be cleaned

```
fare_amount           3
pickup_longitude      0
pickup_latitude       1
dropoff_longitude     0
dropoff_latitude      0
passenger_count      21
pickup_year           0
pickup_month          0
pickup_day            0
pickup_weekday        0
pickup_hour           0
```

We set the above values to NA and then perform outlier analysis to check whether to impute them or remove them.

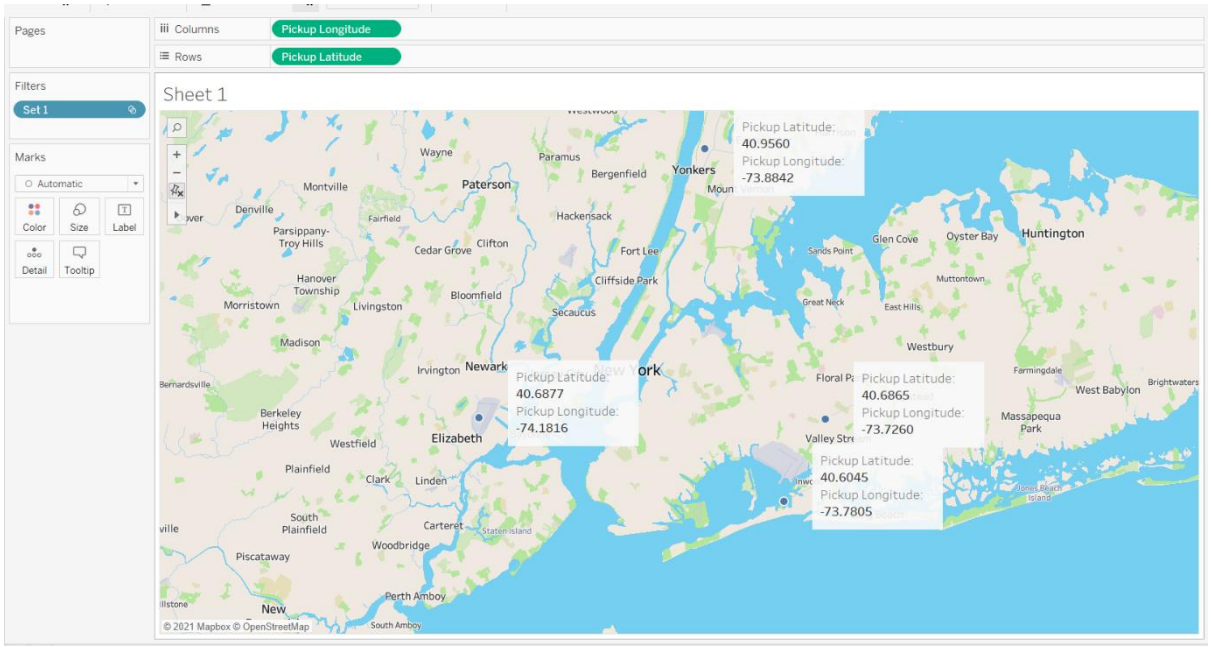
Outlier Analysis

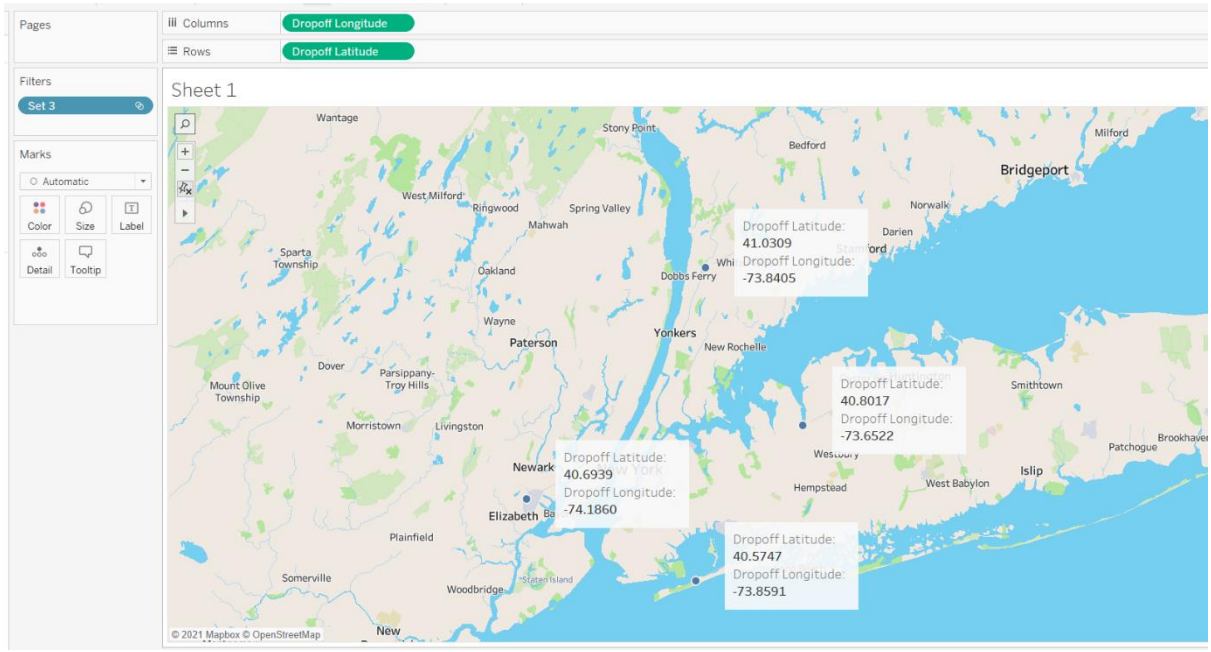
For Pickup and Dropoff Co-ordinates

After visualizing the data points for pickup and dropoff latitudes and longitudes, we found that most of our points lie between the quadrilateral formed by the 4 set of points from dropoff latitude and longitude data. So these points set the range for our co-ordinates and the points which are not in the range are outliers and set to NA to impute them later.

Count of outliers in Co-ordinate Data

```
pickup_longitude     341
pickup_latitude      345
dropoff_longitude    345
dropoff_latitude     341
```





For Fare Amount

We can see that the values above 73.3 can be considered as outliers but we set the value to be 100. So the values beyond 100 will be set to NA.

Now the NA count is:

```
fare_amount         13
passenger_count     21
pickup_year          0
pickup_month         0
pickup_day           0
pickup_weekday       0
pickup_hour          0
```

The above NA's after outlier analysis can be removed as they are very less in number and can account for error if kept.

Feature Engineering

Distance

We create a new variable named Distance from pickup and dropoff coordinates.

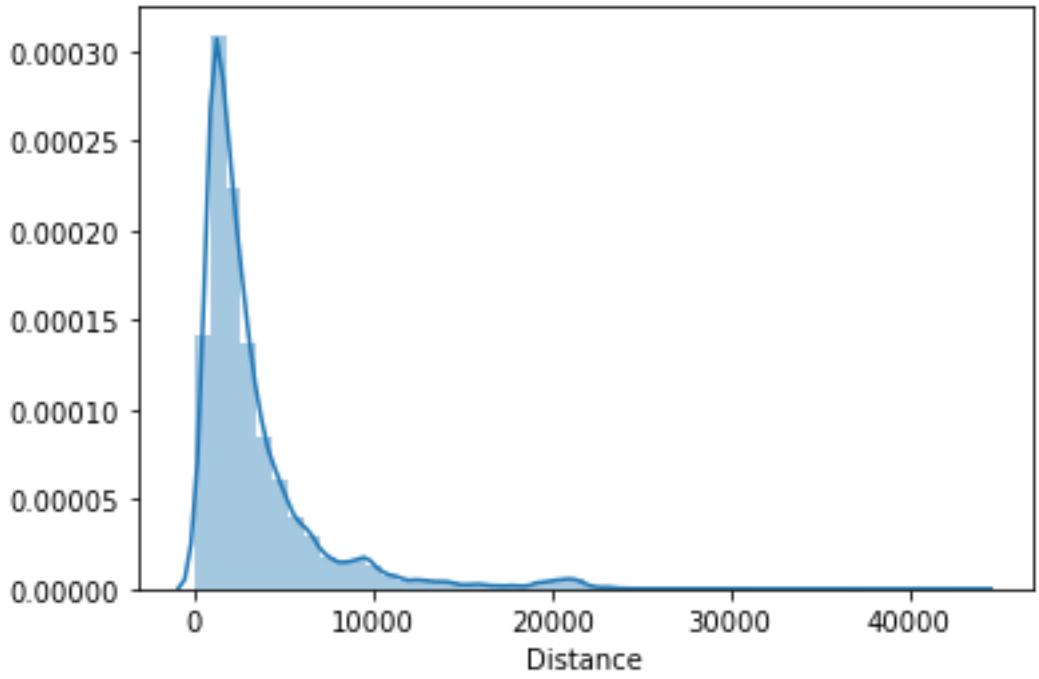
The **haversine formula** determines the great-circle distance between two points on a sphere given their longitudes and latitudes.

Count of NA's in Distance variable is 380 as we set some points from the outlier analysis of co-ordinates to NA.

```
fare_amount           13
pickup_datetime        0
pickup_longitude     341
pickup_latitude      345
dropoff_longitude    345
dropoff_latitude     341
passenger_count       21
pickup_year            0
pickup_month           0
pickup_day             0
pickup_weekday         0
pickup_hour            0
Distance             380
```

We see from the distribution that it is heavily right skewed and also has values equal to zero as seen in the table below. So we will set these values to NA and impute them with outliers.

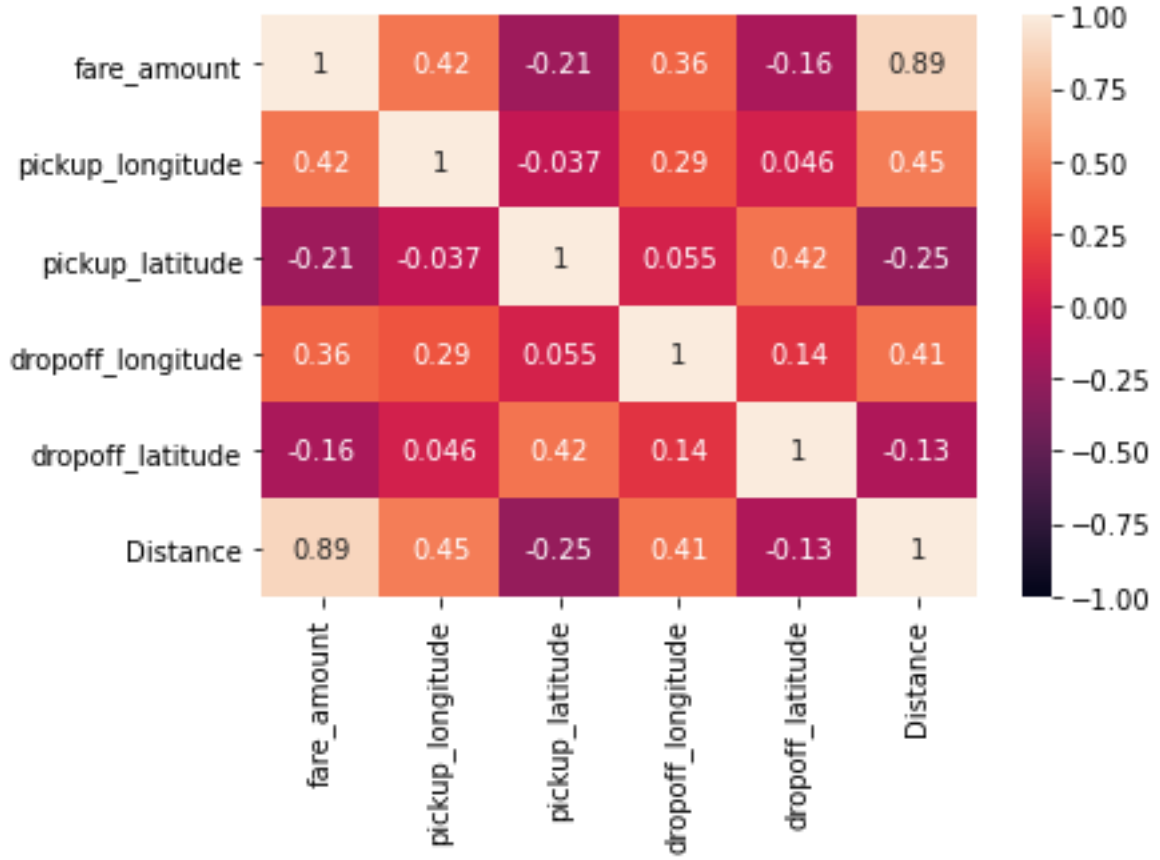
```
count    15606.000000
mean      3319.599052
std       3573.167902
min          0.000000
25%       1258.823151
50%       2170.876936
75%       3889.098211
max      43697.398167
Name: Distance, dtype: float64
```



Imputing NA's in Distance variable

About 3% of values in Distance variable are NA so it will be better to impute them. We used ***KNN Imputation*** for imputing those NA's.

Correlation Analysis



We see only distance variable is highly correlated to fare_amount which is our target variable but no significant other correlations were found.

Also we can remove our redundant co-ordinate variables from data as there is no correlation between them and with the dependent variable.

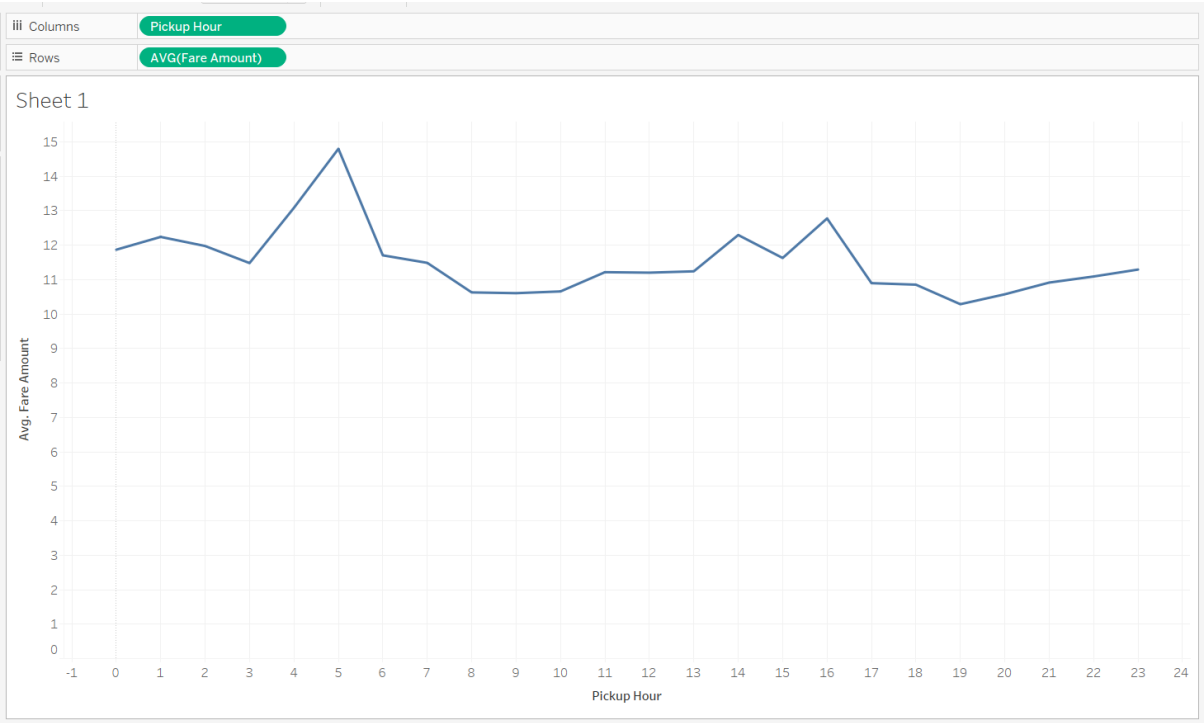
ANOVA for categorical variables

We used one-way anova for categorical variables and for variables whose p-value is less than 0.05 we drop them as was in the case of pickup_day and pickup_weekday. From table we can see which variables to keep and which to drop from the Boolean value of result column.

	Variable_Name	p_value	Result
1	passenger_count	4.479842e-05	TRUE
2	pickup_year	3.828928e-50	TRUE
3	pickup_month	4.000798e-02	TRUE
4	pickup_day	8.727961e-01	FALSE
5	pickup_wday	1.441116e-01	FALSE
6	pickup_hour	2.165490e-12	TRUE

Pickup_Session

This new variable was derived from the bi-variate analysis of fare_amount and pickup_hour.

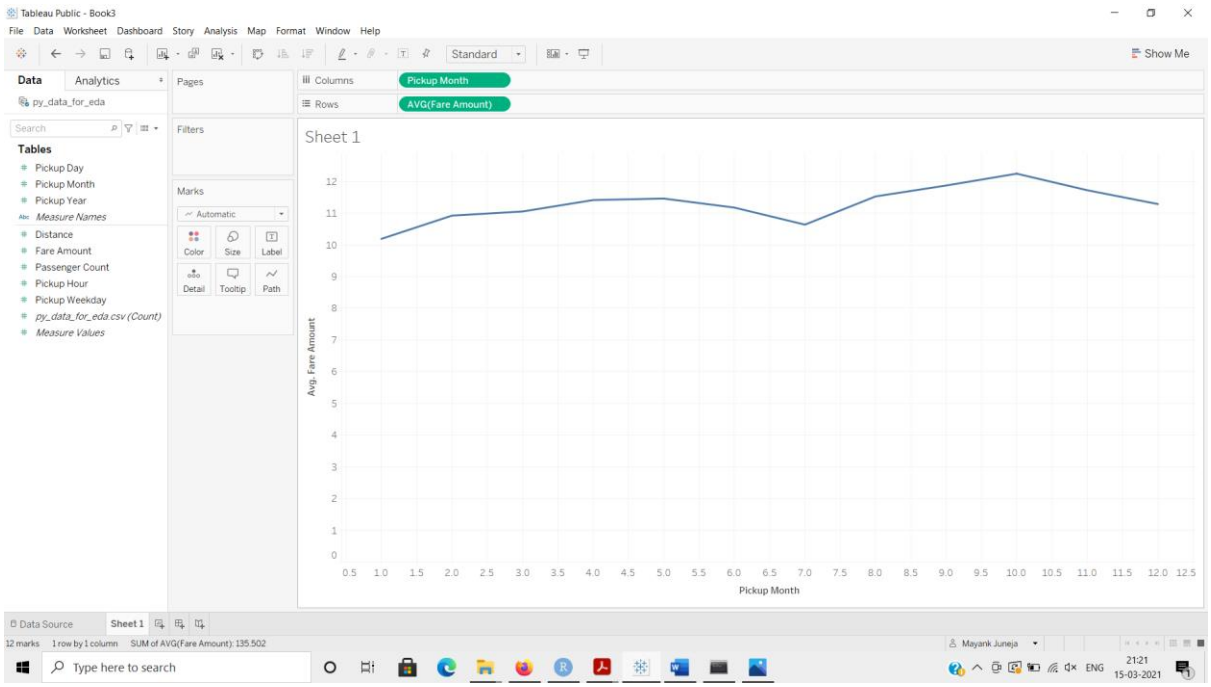


Based on average fare_amount for different pickup_hours we create new variable named ‘pickup_session’ having categories labelled as Morning, Midnight, Afternoon, Evening, Night for different intervals for pickup_hour.

Pickup_Season:

This variable is derived from bi-variate analysis of fare_amount and pickup_month.

Based on average fare_amount for different pickup_months we create new variable named ‘pickup_season’ having categories labelled as Summer, Spring, Autumn, Winter for different intervals for pickup_months.



Feature Scaling

We used MinMax Scaling method for distance variable and used One-Hot Encoding for categorical variables which involves creating dummy variables for each class of categorical variable.

MODELING

The next step is to differentiate the train data into 2 parts i.e. train and valid. The splitting of train data into 2 parts is very important factor to verify the model performance and to understand the problem of over-fitting and under-fitting. Over-fitting is the term where training error is low and testing error is high and under-fitting is the term where both training and testing error is high. Those are the common problem of complex model. In this analysis, since we are predicting fare amount which is the numeric variable. So, we come to know that, our problem statement is predicting type. So, what we can do is we will apply supervise machine learning algorithms to predict our target variable. As we know our target variable is continuous in nature so, here we will build regression model. **Root Mean Square Error** (RMSE) to measures how much **error** there is between two data sets. In other words, it compares a predicted value and an observed or known value. The RMSE is directly interpretable in terms of measurement units, and so is a better measure of goodness of fit. So, in our case any model we build should have lower value of an **RMSE** and higher value of variance i.e**. R square.** Other metrics which are used for evaluation of model are **Mean Absolute Error(MAE)** and **Mean Absolute Percentage Error(MAPE).**

LINEAR REGRESSION

Linear Regression is one of the statistical methods of prediction. It is used to find a linear relationship between the target and one or more predictors. It means the target variables should be continuous in nature. The main idea is to identify a line that best fits the data. To build any model we have some assumptions to put on data and model. This algorithm is not very flexible, and has a very high bias. Below we calculated RMSE values using linear regression.

	Python	R
RMSE Train	4.22	4.678
RMSE Valid(test)	3.72	4.881

RANDOM FOREST

Random forest is the collection of multiple decision trees. In Random Forest, output is the average prediction by each of these trees. For this method to work,

the baseline models must have a lower bias. The idea behind Random Forest is to build n number of trees to have more accuracy in dataset. Random Forest uses bagging method for predictions. It can handle large no of independent variables without variable deletion and it will give the estimates that what variables are important. The RMSE values are shown below.

	Python	R
RMSE Train	1.64	3.304
RMSE Valid(test)	3.96	4.833

Hyperparameter tuning for Random Forest Model

Here we have used Random Search CV for hyper parameters tuning.

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.

After tuning the model we get following results:

	Python	R
RMSE Train	3.92	3.075
RMSE Valid(test)	3.91	4.842

EXTREME GRADIENT BOOSTING

Gradient boosting is currently one of the most popular machine learning techniques for efficient modeling of tabular datasets of all sizes. It is very fast, takes quite less RAM to run, and focuses on the accuracy of the result. It is a technique which applicable for regression and classification type of problems. In this method, multiple weak learners are ensemble to create strong learners. Gradient boosting uses all data to train each learner. But instances that were misclassified by the previous learners are given more weight, so that subsequent learners can give more focus to them during training. Here we select XGBoost for model development.

	Python	R
RMSE Train	2.44	2.98
RMSE Valid(test)	4.04	4.154

Hyperparameter tuning for Extreme Gradient Boosting

Using Randomized Search CV to find the best parameters

	Python	R
RMSE Train	3.76	1.694
RMSE Valid(test)	3.7	2.225

CONCLUSION

Model Evaluation:

Above model help us to calculate the **Root Mean Square Error (RMSE)** and **R-Squared** Values. RMSE is the standard deviation of the prediction errors. Residuals are a 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. RMSE is an absolute measure of fit. R-squared is a relative measure of fit. R-squared is basically explains the degree to which input variable explain the variation of the output. In simple words R-squared tells how much variance of dependent variable explained by the independent variable. It is a measure if goodness of fit in regression line. Value of R-squared is between 0-1, where 0 means independent variable unable to explain the target variable and 1 means target variable is completely explained by the independent variable. So, Lower values of RMSE and higher value of R-Squared Value indicate better fit of model.

Model Selection

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

- From the observation of all RMSE Value we have concluded that,
- Gradient Boosting perform comparatively well while comparing their RMSE.
- So finally, we can say that Extreme Gradient Boosting(XGBoost) 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.