1/27/2020

DATA SCIENCE PROJECT

[CAB FARE PREDICTION]

By Hitesh Damani |

Contents

Introduction

Problem Statement

The objective of this project is to predict Cab Fare amount.

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

Attributes of Train Dataset:

- Fare_amount- object value, converted to float indicating the price charged for journey (Target variable)
- pickup_datetime - timestamp value indicating when the cab ride started.
- pickup_longitude - float for longitude coordinate of where the cab ride started.
- pickup_latitude - float for latitude coordinate of where the cab ride started.
- dropoff_longitude - float for longitude coordinate of where the cab ride ended.
- dropoff_latitude - float for latitude coordinate of where the cab ride ended.
- passenger_count - an integer indicating the number of passengers in the cab ride.

Attributes of Test Dataset:

- pickup_datetime - timestamp value indicating when the cab ride started.
- pickup_longitude - float for longitude coordinate of where the cab ride started.
- pickup_latitude - float for latitude coordinate of where the cab ride started.
- dropoff_longitude - float for longitude coordinate of where the cab ride ended.
- dropoff_latitude - float for latitude coordinate of where the cab ride ended.
- passenger_count - an integer indicating the number of passengers in the cab ride.

Methodology

Pre-Processing

Data pre-processing is the first stage of any type of project. In this stage we get the feel of the data. We do this by looking at plots of independent variables vs target variables. If the data is messy, we try to improve it by sorting deleting extra rows and columns. This stage is called as **Exploratory Data Analysis**. *This stage generally involves data cleaning, merging, sorting, looking for outlier analysis, looking for missing values in the data, imputing missing values if found by various methods such as mean, median, mode, KNN imputation, etc.*

Further we will look into what Pre-Processing steps do this project was involved in.

Getting feel of data via visualization:

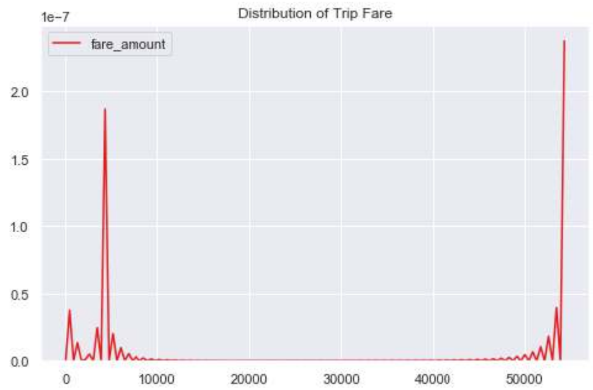
Some Histogram plots from seaborn library for each individual variable created using distplot() method.

Univariate Analysis.

1. Visualizing distribution of fare_amount

```
plt.figure(figsize=(8,5))
sns.kdeplot(train['fare_amount']).set_title("Distribution of Trip Fare")
```

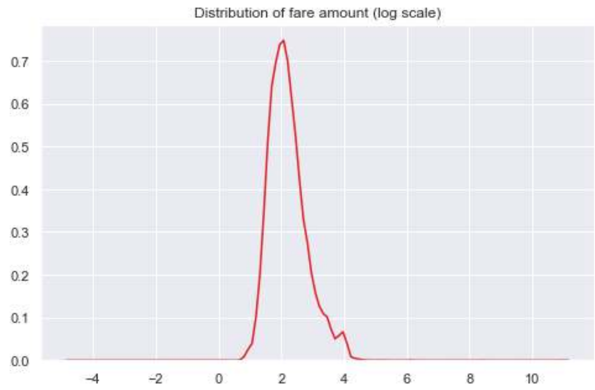
```
Text(0.5, 1.0, 'Distribution of Trip Fare')
```



Since we saw above that fare amount is highly skewed,let us take log transformation of the fare amount and plot the distribution

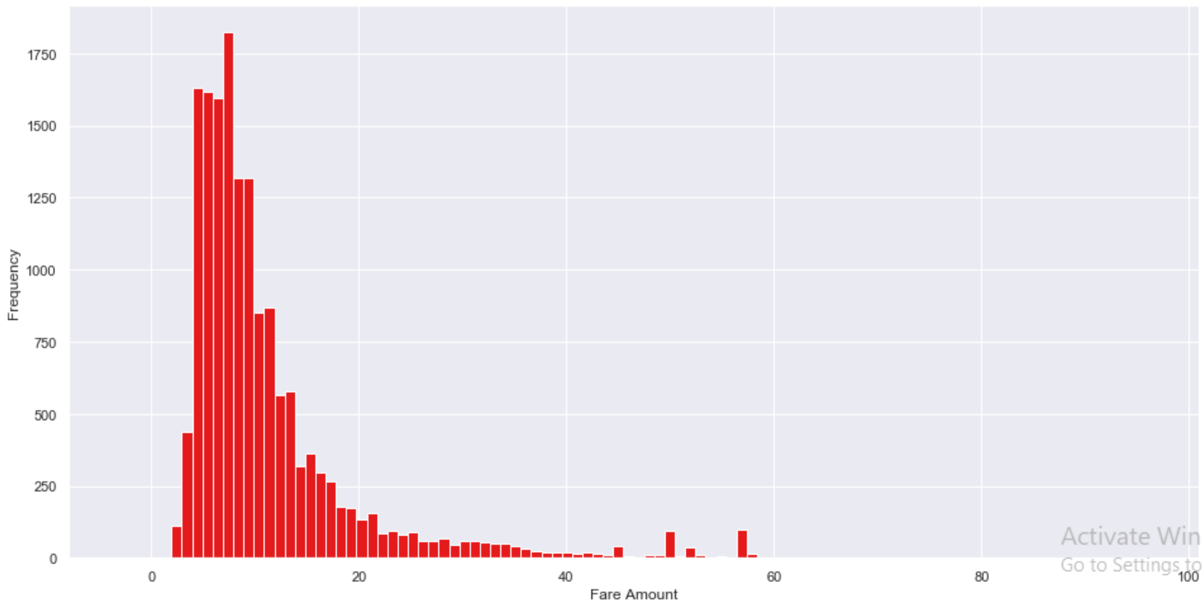
```
plt.figure(figsize=(8,5))
sns.kdeplot(np.log(train['fare_amount'].values)).set_title("Distribution of fare amount (log scale)")
```

```
Text(0.5, 1.0, 'Distribution of fare amount (log scale)')
```



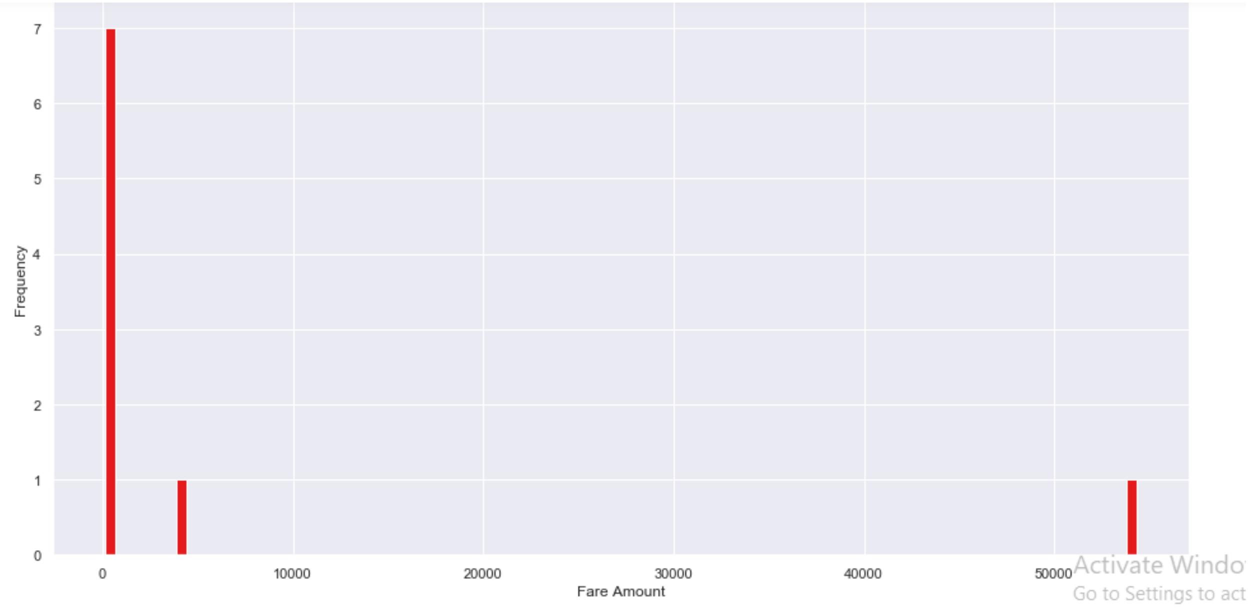
Most fares are between 2.7.Median fare is around 10

Distribution of fare_amount < 100



There are few points between 40 and 60 dollars which has slightly high frequency and that could be airport trips

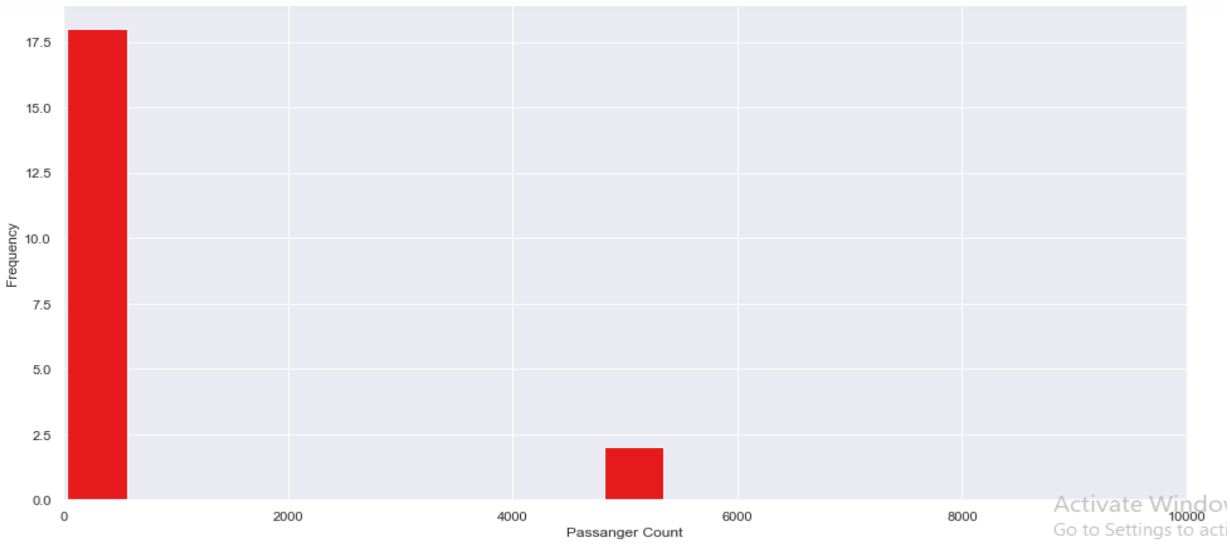
Distribution of fare_amount < 100



- We can see here that there are total 9 trips which are above 100 dollars
- Some of them might be outliers or few of them might be long distance trip from/to airport, we will see it in later section

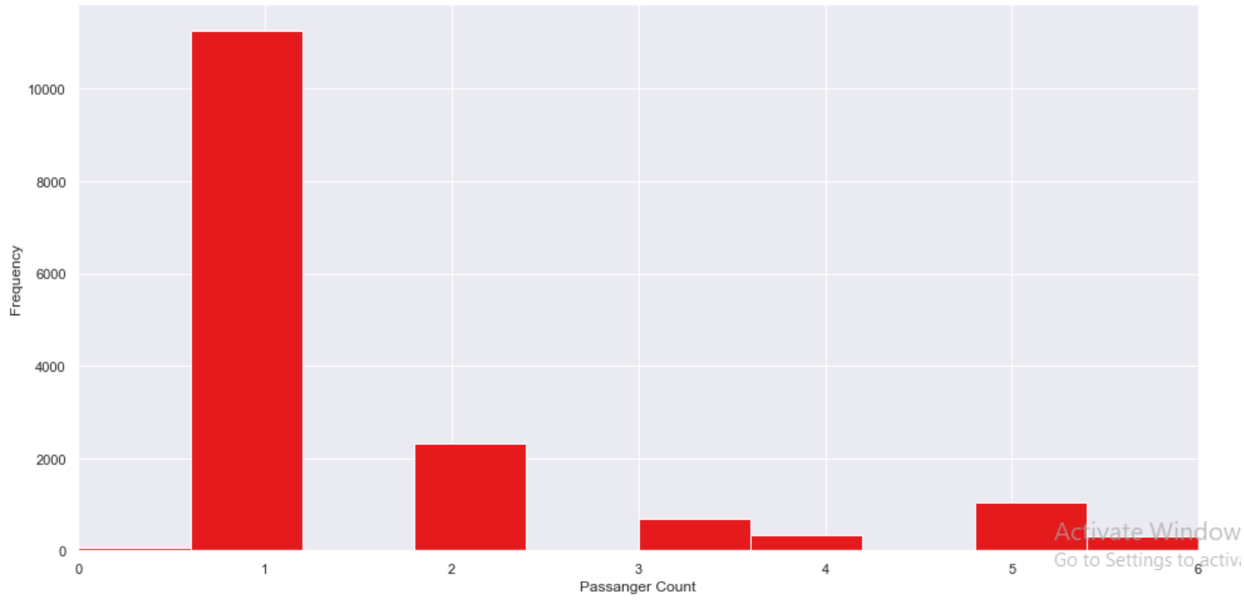
2. Visualization of Passenger_count variable

Paseenger_count >7



There are 20 values of passenger count >7 we will we will anayze this value in Outlier Analysis Section.

Passenger_count<7



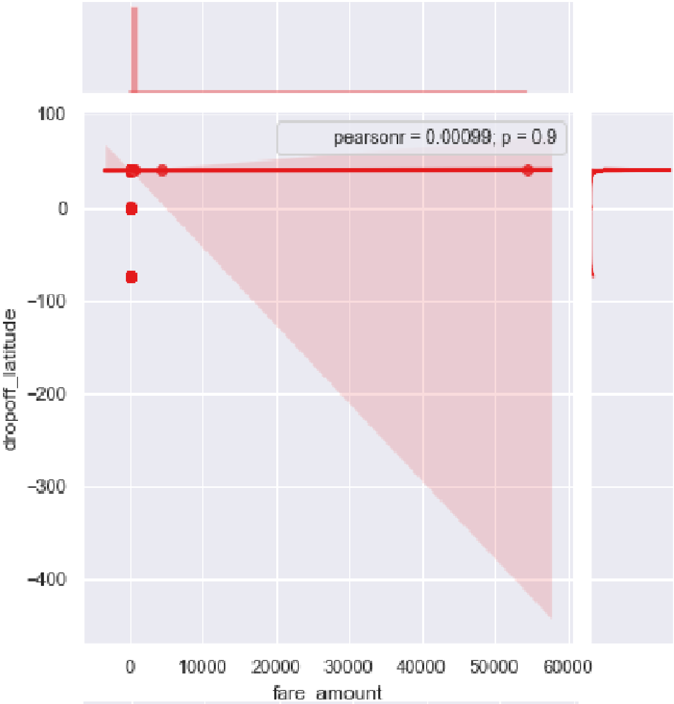
Most of the trips are taken by single passenger, we will try to see if there is any relation between passenger count and fare amount.

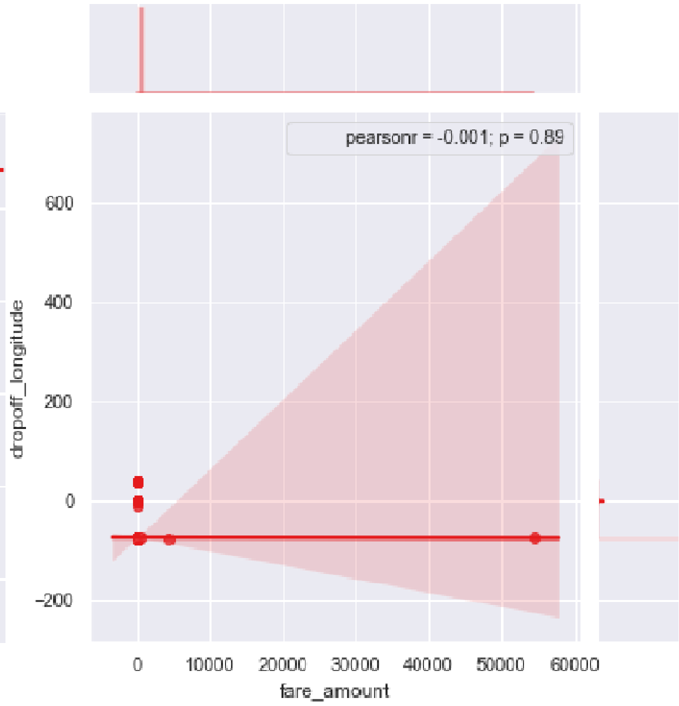
We have 57 such cases where passanger count is zero, there can be two possibility

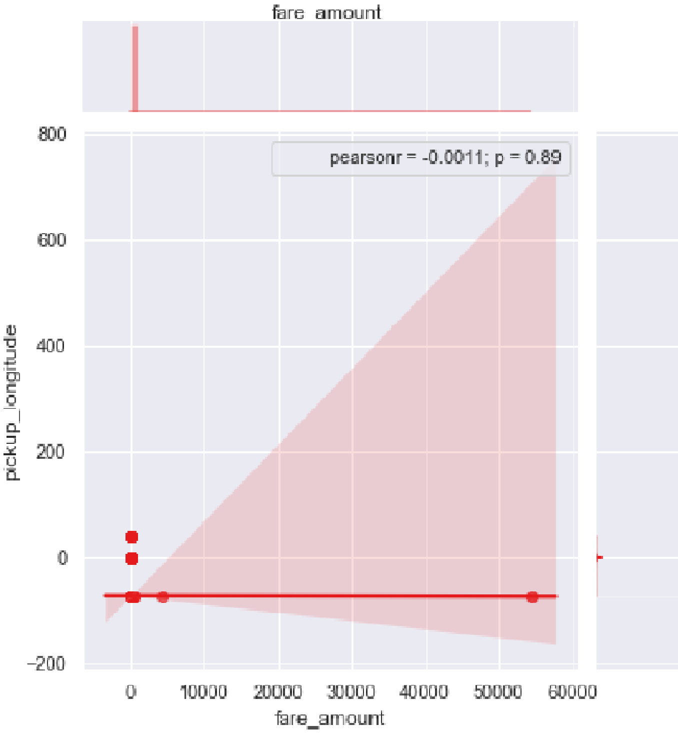
- Passanger count is incorrectly populated
- Taxi was not carrying any passanger, may be taxi was used for goods.

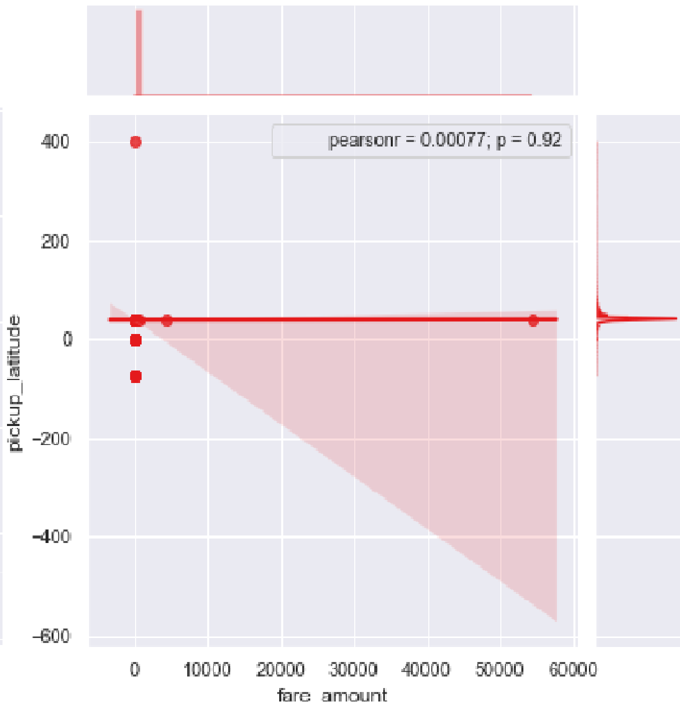
Bivariate Analysis.

- Here we have plotted Scatter plot with Regression line between 2 variables along with separate Bar plots of both variables.
- Also, we have annotated Pearson correlation coefficient and p value.
- Plotted only for numerical/continuous variables
- Target variable ‘fare_amount’ Vs each numerical variable.

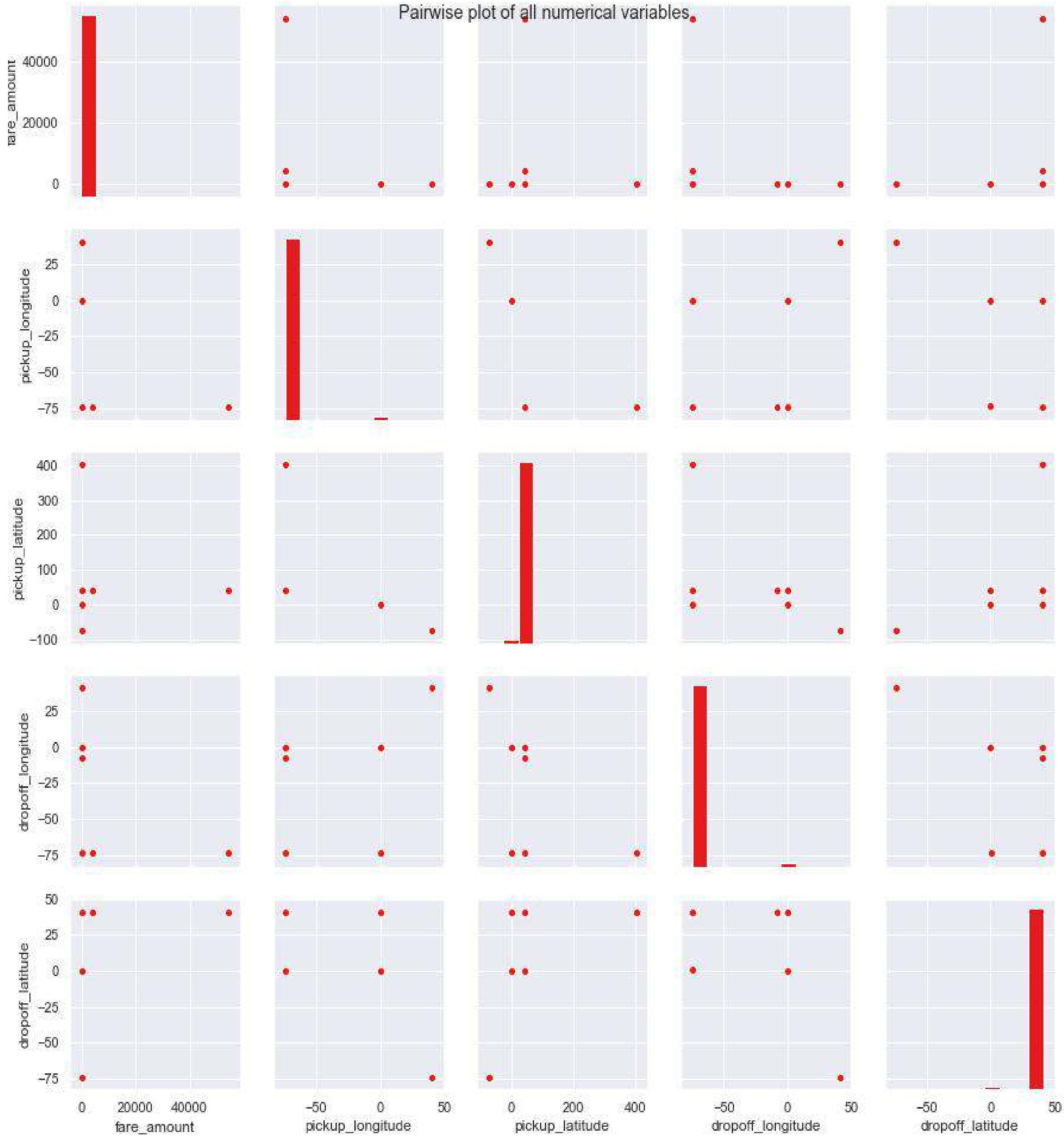

pearsonr = 0.00099; p = 0.9
dropoff_latitude
fare_amount
100
0
−100
−200
−300
−400
0
10000
20000
30000
40000
50000
60000


pearsonr = -0.001; p = 0.89
dropoff_longitude
fare_amount
600
400
200
0
−200
0
10000
20000
30000
40000
50000
60000


pearsonr = -0.0011; p = 0.89
pickup_longitude
fare_amount
800
600
400
200
0
−200
0
10000
20000
30000
40000
50000
60000


pearsonr = 0.00077; p = 0.92
pickup_latitude
fare_amount
400
200
0
−200
−400
−600
0
10000
20000
30000
40000
50000
60000

Pair-Plot to visualize the spread of numerical data



Analysing values which are not within desired range(outlier) depending upon basic understanding of dataset.

In this step we will analyse values in each variable which are not within desired range and we will consider them as outliers depending upon basic understanding of all the variables.

After analysing the variables we decided to impute them. So for that we had marked all the values which were outside the desired range as "NA".

Missing Values after analysing the outlier Values:-

```
train.isnull().sum()
```

```
fare_amount          30
pickup_datetime       0
pickup_longitude    315
pickup_latitude     316
dropoff_longitude   314
dropoff_latitude    312
passenger_count     133
dtype: int64
```

So after Exploratory Data Analysis we have observed that the missing values has been increased. Now we have to impute them with any of the statistical method

Missing Value Analysis

In this step we look for missing values in the dataset like empty row column cell which was left after removing special characters and punctuation marks. Some missing values are in form of "NA".Missing values left behind after outlier analysis; missing values can be in any form.

Unfortunately, in this dataset we have found some missing values. Therefore, we will do some missing value analysis. Before imputed we selected random row no-1000 and made it NA, so that we will compare original value with imputed value and choose best method which will impute value closer to actual value.

	index	0
0	fare_amount	30
1	pickup_datetime	0
2	pickup_longitude	315
3	pickup_latitude	316
4	dropoff_longitude	314
5	dropoff_latitude	312
6	passenger_count	133

	Variables	Missing_percentage
0	pickup_latitude	1.966764
1	pickup_longitude	1.960540
2	dropoff_longitude	1.954316
3	dropoff_latitude	1.941868
4	passenger_count	0.827784
5	fare_amount	0.186718
6	pickup_datetime	0.000000

We will impute values for all the variables except pickup_datetime.

We'd tried central statistical methods in Python and KNN method in R to impute missing values in the dataset:

1. For fare_amount:
   Actual value = 7.0
   Mean= 15.020
   Median= 8.5

   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable.

2. For pickup_longitude:
   Actual value = -73.99,
   Mean = -73.91,
   Median = -73.98,

   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable.

3. For pickup_latitude:
   Actual value =40.75 ,
   Mean = 40.69,
   Median =40.75,

   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable

4. For dropoff_longitude:
   Actual value =-73.98,
   Mean = -73.90,
   Median = -73.98,

   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable

5. For dropoff_latitude:
   Actual value = 40.75,
   Mean =40.68,
   Median = 40.75,

   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable

6. For passenger_count:
   Actual value = 1,
   Mean = 1.64,
   Median = 1,

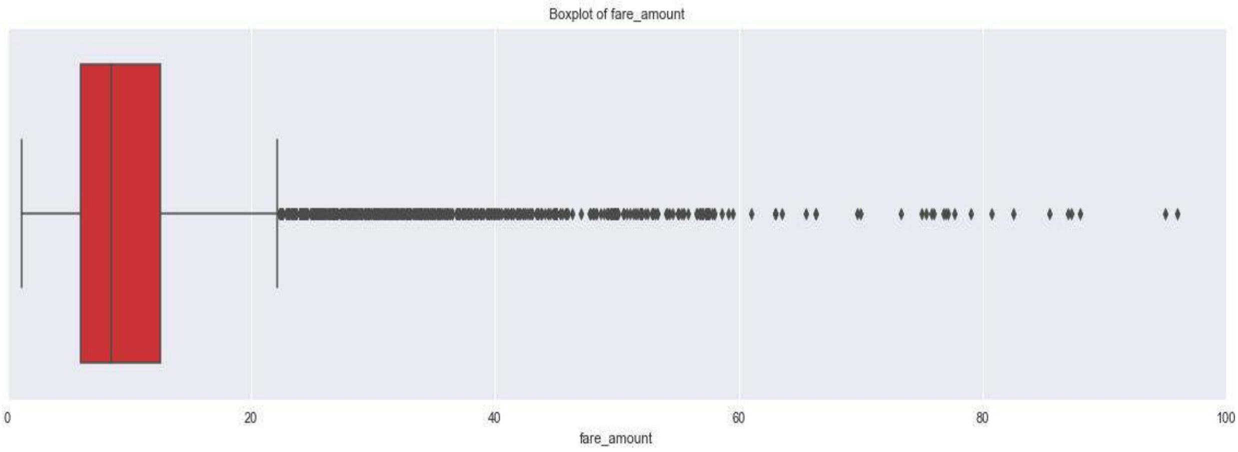
   We will Choose **median** method here because it imputes value closest to actual value also it maintains the Standard deiviation of the variable.

Outlier Analysis

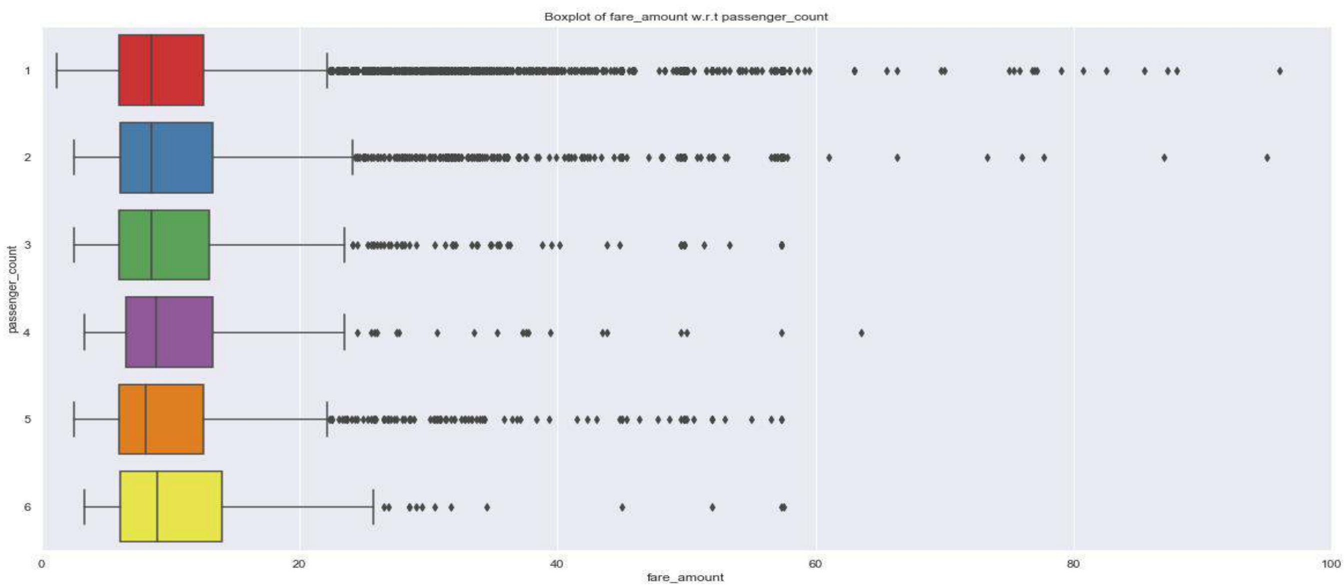
We look for outlier in the dataset by plotting Boxplots. There are outliers present in the data. we have removed these outliers. This is how we done,

- We replaced them with Nan values or we can say created missing values.
- Then we imputed those missing values with **median** method.
- We Will do Outlier Analysis only on Fare_amount just for now and we will do outlier analysis after feature engineering laitudes and longitudes.

Univariate Boxplots: Boxplots for target variable



Bivariate Boxplots: Boxplots for fare_amount Variables Vs passenger_count variable



From above Boxplots we see that 'fare_amount' have outliers in it:

'fare_amount' has 1359 outliers.

We successfully imputed these outliers with **median** method.

Feature Engineering

Feature Engineering is used to drive new features from existing features.

1. For 'pickup_datetime' variable:

We will use this timestamp variable to create new variables.

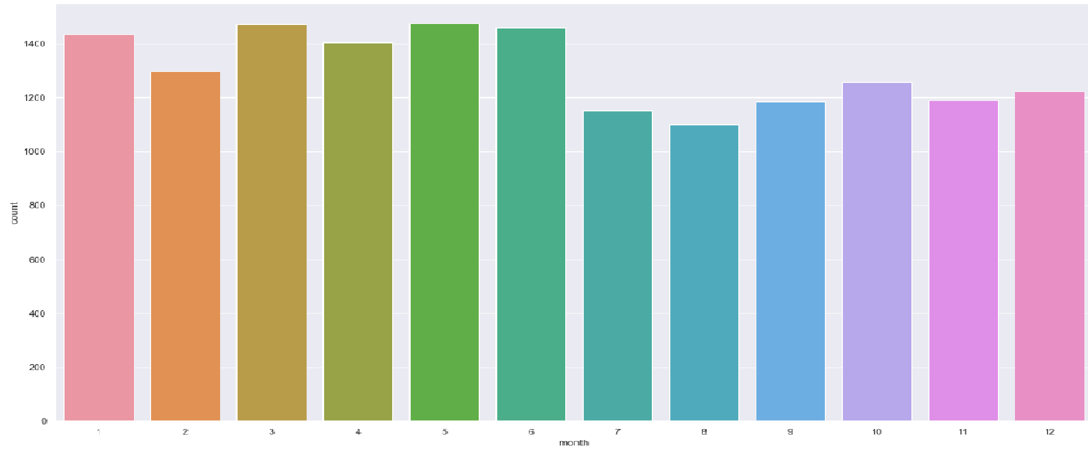
New features will be year, month, day_of_week, hour.

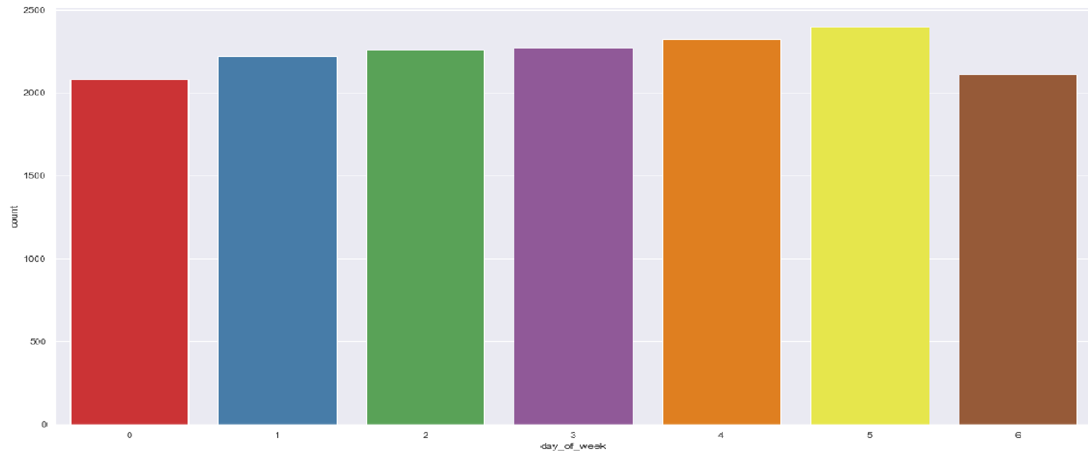
'year' will contain only years from pickup_datetime. For ex. 2009, 2010, 2011, etc.

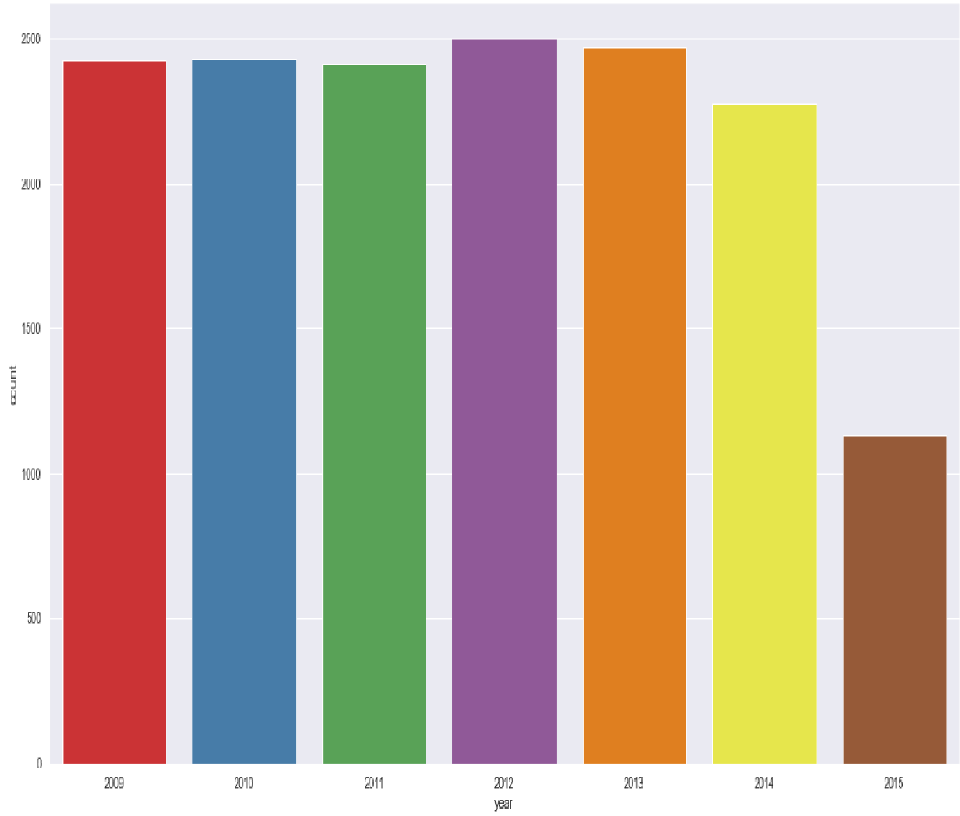
'month' will contain only months from pickup_datetime. For ex. 1 for January, 2 for February, etc.

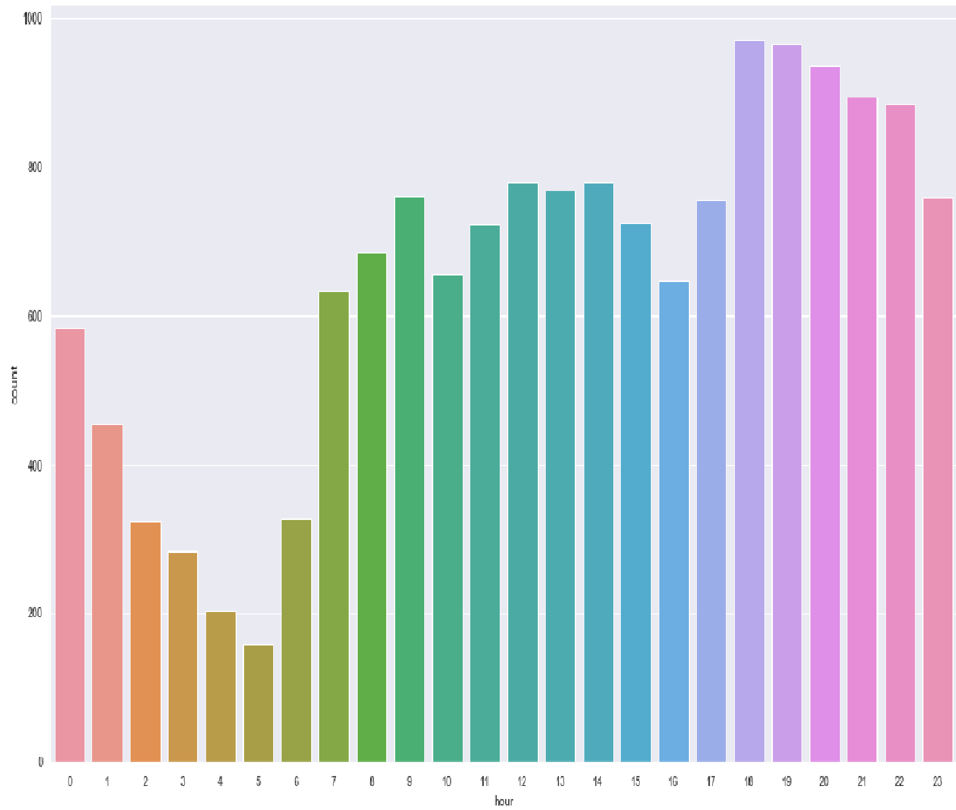
'day_of_week' will contain only week from pickup_datetime. For ex. 1 which is for Monday,2 for Tuesday,etc.

'hour' will contain only hours from pickup_datetime. For ex. 1, 2, 3, etc.






2500
2000
1500
1000
500
0
count
2009
2010
2011
2012
2013
2014
2015
year


1000
800
600
400
200
0
count
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
hour

As we have now these new variables we will categorize them to new variables like Session from hour column, seasons from month column, week:weekday/weekend from day_of_week variable.

So, session variable which will contain categories—morning, afternoon, evening, night_PM, night_AM.

Seasons variable will contain categories—spring, summer, fall, winter.

Week will contain categories—weekday, weekend.

We will one-hot-encode session, seasons, week variable.

2. For 'passenger_count' variable:

As passenger_count is a categorical variable we will one-hot-encode it.

3. For 'Latitudes' and 'Longitudes' variables:

As we have latitude and longitude data for pickup and dropoff, we will find the distance the cab travelled from pickup and dropoff location.

We will use both haversine and vincenty methods to calculate distance. For haversine, variable name will be 'great_circle' and for vincenty, new variable name will be 'geodesic'.

As Vincenty is more accurate than haversine. Also, vincenty is prefered for short distances.

Therefore, we will drop great_circle.

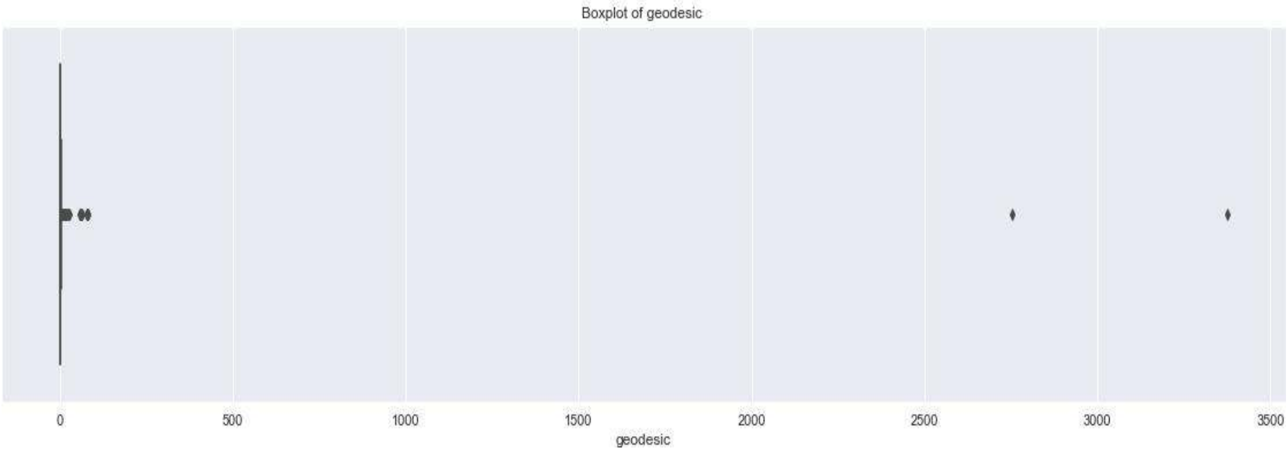
Columns in training data after feature engineering:

Index(['fare_amount', 'passenger_count_2', 'passenger_count_3','passenger_count_4','passenger_count_5', 'passenger_count_6','season_spring', 'season_summer', 'season_winter', 'week_weekend','session_evening', 'session_morning', 'session_night_AM','session_night_PM', 'year_2010', 'year_2011', 'year_2012', 'year_2013','year_2014', 'year_2015', 'geodesic'],dtype='object')

Columns in testing data after feature engineering:

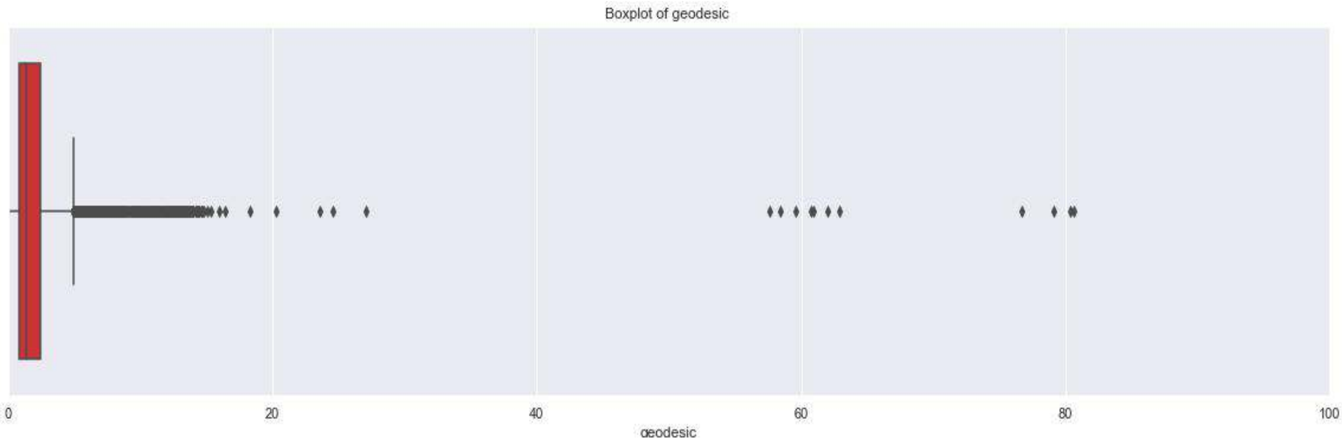
Index(['passenger_count_2', 'passenger_count_3', 'passenger_count_4','passenger_count_5', 'passenger_count_6', 'season_spring','season_summer', 'season_winter', 'week_weekend', 'session_evening','session_morning', 'session_night_AM', 'session_night_PM', 'year_2010','year_2011', 'year_2012', 'year_2013', 'year_2014', 'year_2015','geodesic'],dtype='object')

We will plot boxplot for our new variable 'geodesic':



We see that there are outliers in 'geodesic' and also a cab cannot go upto 3400 miles

Boxplot of 'geodesic' for range 0 to 100 miles.



We will treat these outliers like we previously did

Feature Selection

In this step we would allow only to pass relevant features to further steps. We remove irrelevant features from the dataset. We do this by some statistical techniques, like we look for features which will not be helpful in predicting the target variables. In this dataset we have to predict the fare_amount.

Further below are some types of test involved for feature selection:

1 Correlation analysis –:

This requires only numerical variables. Therefore, we will filter out only numerical variables and feed it to correlation analysis. We do this by plotting correlation plot for all numerical variables. There should be no correlation between independent variables but there should be high correlation between independent variable and dependent variable. So, we plot the correlation plot. we can see that in correlation plot faded colour like skin colour indicates that 2 variables are highly correlated with each other. As the colour fades correlation values increases.

From below correlation plot we see that:

- 'fare_amount' and 'geodesic' are very highly correlated with each other.
- As fare_amount is the target variable and ‘geodesic’ is independent variable we will keep ‘geodesic’ because it will help to explain variation in fare_amount.

```
Correlation_Matrix
```

	fare_amount	geodesic
fare_amount	1.000000	0.700095
geodesic	0.700095	1.000000

As we can see from above correlation matrix fare_amount and geodesic are highly positively correlated to each other.

2 Chi-Square test of independence – :

Unlike correlation analysis we will filter out only categorical variables and pass it to Chi-Square test. Chi-square test compares 2 categorical variables in a contingency table to see if they are related or not.

I. Assumption for chi-square test: Dependency between Independent variable and dependent variable should be high and there should be no dependency among independent variables.

II. Before proceeding to calculate chi-square statistic, we do the hypothesis testing:
Null hypothesis: 2 variables are independent.
Alternate hypothesis: 2 variables are not independent.

The interpretation of chi-square test:

I. For theorical or excel sheet purpose: If chi-square statistics is greater than critical value then reject the null hypothesis saying that 2 variables are dependent and if it’s less, then accept the null hypothesis saying that 2 variables are independent.

II. While programming: If p-value is less than 0.05 then we reject the null hypothesis saying that 2 variables are dependent and if p-value is greater than 0.05 then we accept the null hypothesis saying that 2 variables are independent.

Here we did the test between categorical independent variables pairwise.

- If p-value<0.05 then remove the variable
- If p-value>0.05 then keep the variable.

3 Analysis of Variance(Anova) Test –:

I. It is carried out to compare between each group in a categorical variable.
II. ANOVA only lets us know the means for different groups are same or not. It doesn't help us identify which mean is different.

Hypothesis testing:

- **Null Hypothesis**: mean of all categories in a variable are same.
- **Alternate Hypothesis**: mean of at least one category in a variable is different.
- If p-value is less than 0.05 then we reject the null hypothesis.
- And if p-value is greater than 0.05 then we accept the null hypothesis.

Below is the anova analysis table for each categorical variable:

	df	sum_sq	mean_sq	F	PR(>F)
C(passenger_count_2)	1.0	8.346344	8.346344	0.547145	4.594976e-01
C(passenger_count_3)	1.0	10.606757	10.606757	0.695326	4.043711e-01
C(passenger_count_4)	1.0	83.366252	83.366252	5.465075	1.941245e-02
C(passenger_count_5)	1.0	32.290446	32.290446	2.116800	1.457102e-01
C(passenger_count_6)	1.0	195.173348	195.173348	12.794589	3.486463e-04
C(TimeInterval_EarlyMorning)	1.0	977.727525	977.727525	64.094929	1.266323e-15
C(TimeInterval_Evening)	1.0	24.815971	24.815971	1.626811	2.021632e-01
C(TimeInterval_LateNight)	1.0	255.360275	255.360275	16.740143	4.307380e-05
C(TimeInterval_Morning)	1.0	78.788397	78.788397	5.164973	2.305998e-02
C(Seasons_Spring)	1.0	66.230823	66.230823	4.341762	3.720421e-02
C(Seasons_Summer)	1.0	20.533260	20.533260	1.346058	2.459857e-01
C(Seasons_Winter)	1.0	213.880378	213.880378	14.020928	1.814189e-04
C(WeekendWeekday_Weekend)	1.0	2.116058	2.116058	0.138718	7.095636e-01
C(year_2010)	1.0	893.668167	893.668167	58.584418	2.057955e-14
C(year_2011)	1.0	769.255188	769.255188	50.428525	1.287691e-12
C(year_2012)	1.0	308.746707	308.746707	20.239891	6.879311e-06
C(year_2013)	1.0	298.315342	298.315342	19.556063	9.833236e-06
C(year_2014)	1.0	938.869952	938.869952	61.547621	4.592403e-15

Looking at above table every variable has p value less than 0.05 so reject the null hypothesis.

4. Multicollinearity–

In regression, "multicollinearity" refers to predictors that are correlated with other predictors. Multicollinearity occurs when your model includes multiple factors that are correlated not just to your response variable, but also to each other.

I. Multicollinearity increases the standard errors of the coefficients.

II. Increased standard errors in turn means that coefficients for some independent variables may be found not to be significantly different from 0.

III. In other words, by overinflating the standard errors, multicollinearity makes some variables statistically insignificant when they should be significant.Without multicollinearity (and thus, with lower standard errors), those coefficients might be significant.

IV. VIF is always greater or equal to 1.

if VIF is 1 --- Not correlated to any of the variables.

if VIF is between 1-5 --- Moderately correlated.

if VIF is above 5 --- Highly correlated.

If there are multiple variables with VIF greater than 5, only remove the variable with the highest VIF.

V. And if the VIF goes above 10, you can assume that the regression coefficients are poorly estimated due to multicollinearity.

Below is the table for VIF analysis for each independent variable

	VIF	features
0	15.279042	Intercept
1	1.040013	passenger_count_2[T.1]
2	1.019106	passenger_count_3[T.1]
3	1.011659	passenger_count_4[T.1]
4	1.024421	passenger_count_5[T.1]
5	1.017196	passenger_count_6[T.1]
6	1.643048	Seasons_Spring[T.1]
7	1.553378	Seasons_Summer[T.1]
8	1.588540	Seasons_Winter[T.1]
9	1.050953	WeekendWeekday_Weekend[T.1]
10	1.530262	TimeInterval_Evening[T.1]
11	1.563400	TimeInterval_Morning[T.1]
12	1.366456	TimeInterval_EarlyMorning[T.1]
13	1.426068	TimeInterval_LateNight[T.1]
14	1.696844	year_2010[T.1]
15	1.698962	year_2011[T.1]
16	1.720523	year_2012[T.1]

We have checked for multicollinearity in our Dataset and all VIF values are below 5.

Feature Scaling

Data Scaling methods are used when we want our variables in data to scaled on common ground. It is performed only on continuous variables.

- Normalization: Normalization refer to the dividing of a vector by its length. normalization normalizes the data in the range of 0 to 1. It is generally used when we are planning to use distance method for our model development purpose such as KNN. Normalizing the data improves convergence of such algorithms. Normalisation of data scales the data to a very small interval, where outliers can be loosed.
- Standardization: Standardization refers to the subtraction of mean from individual point and then dividing by its SD. Z is negative when the raw score is below the mean and Z is positive when above mean. When the data is distributed normally you should go for standardization.

Linear Models assume that the data you are feeding are related in a linear fashion, or can be measured with a linear distance metric.

We will use sklearn's **Standard Scalar** method to scale down our variables.

Spliting Train and Validation Dataset

a) We have used sklearn's train_test_split() method to divide whole Dataset into train and validation datset.

b) 20% is in validation dataset and 80% is in training data.

c) 12852 observations in training and 3214 observations in validation dataset.

d) We will test the performance of model on validation datset.

e) The model which performs best will be chosen to perform on test dataset provided along with original train dataset.

f) X_train y_train--are train subset.

g) X_test y_test--are validation subset.

Hyperparameter Optimization

a. To find the optimal hyperparameter we have used sklearn.model_selection.GridSearchCV and sklearn.model_selection.RandomizedSearchCV

b. GridSearchCV tries all the parameters that we provide it and then returns the best suited parameter for data.

c. We gave parameter dictionary to GridSearchCV which contains keys which are parameter names and values are the values of parameters which we want to try for.

Multiple Linear Regression:

```
Tuned Decision reg Parameters: {'copy_X': True, 'fit_intercept': True}
Best score is 0.5070409739150635
```

Ridge Regression:

```
Tuned Decision ridge Parameters: {'alpha': 1.0, 'max_iter': 500, 'normalize':
False}
Best score is 0.5070426407281174
```

Lasso Regression:

```
Tuned Decision lasso Parameters: {'alpha': 0.0004498432668969444, 'max_iter': 500,
'normalize': False}
Best score is 0.5070593646184973
```

Decision Tree Regression:

```
Tuned Decision Tree Parameters: {'max_depth': 6, 'min_samples_split': 12}
Best score is 0.5585217838169321
```

Random Forest Regression:

```
Tuned Random Forest Parameters: {'n_estimators': 200, 'min_samples_split':
2, 'min_samples_leaf': 3, 'max_features': 'log2', 'max_depth': 16,
'bootstrap': True}
Best score is 0.557878668176511
```

Xgboost regression:

```
subsample= 0.7000000000000001, reg_alpha= 0.0005428675439323859,
n_estimators= 400, max_depth= 7, learning_rate= 0.1, colsample_bytree=
0.1, colsample_bynode= 0.9000000000000001, colsample_bylevel=
0.9000000000000001
```

Model Development

Our problem statement wants us to predict the fare_amount. This is a Regression problem. So, we are going to build regression models on training data and predict it on test data. In this project I have built models using 5 Regression Algorithms:

I. Linear Regression
II. Ridge Regression
III. Lasso Regression
IV. Decision Tree
V. Random Forest
VI. Xgboost Regression

We will evaluate performance on validation dataset which was generated using Sampling. We will deal with specific error metrics like:-

Regression metrics for our Models:

- R^2
- Adjusted R^2
- MAPE(Mean Absolute Percentage Error)
- MSE(Mean square Error)
- RMSE(Root Mean Square Error)
- RMSLE(Root Mean Squared Log Error)

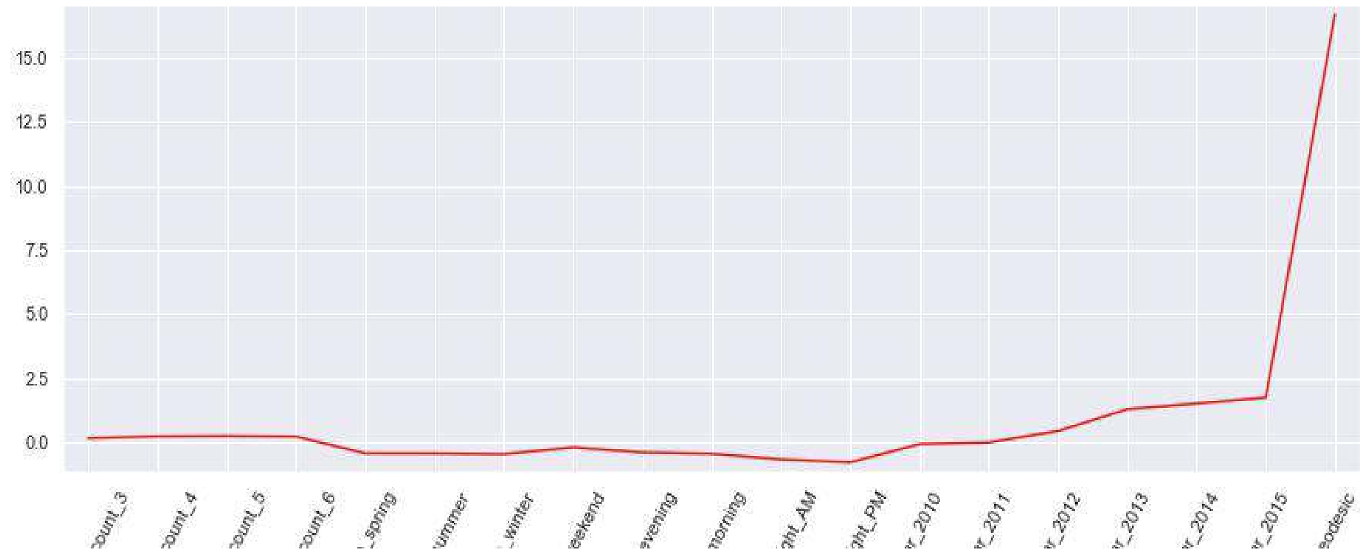
Model Performance

Here, we will evaluate the performance of different Regression models based on different Error Metrics

1. Multiple Linear Regression:-

Error Metrics	**R^2**	**Adj R^2**	**MAPE**	**MSE**	**RMSE**	**RMSLE**
Train	0.519	0.518	22.23	7.53	2.74	0.255
Validation	0.462	0.458	22.80	8.31	2.88	0.267

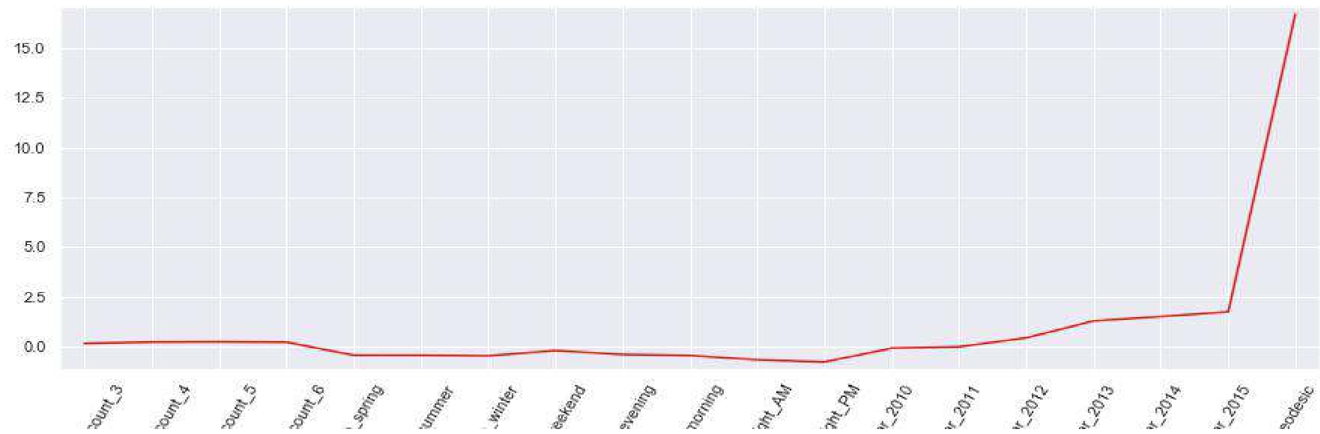
Line Plot for Coefficients of Multiple Linear regression:



2. Ridge Regression:-

Error Metrics	R^2	Adj R^2	MAPE	MSE	RMSE	RMSLE
Train	0.519	0.518	22.23	7.53	2.74	0.255
Validation	0.462	0.458	22.80	8.31	2.88	0.267

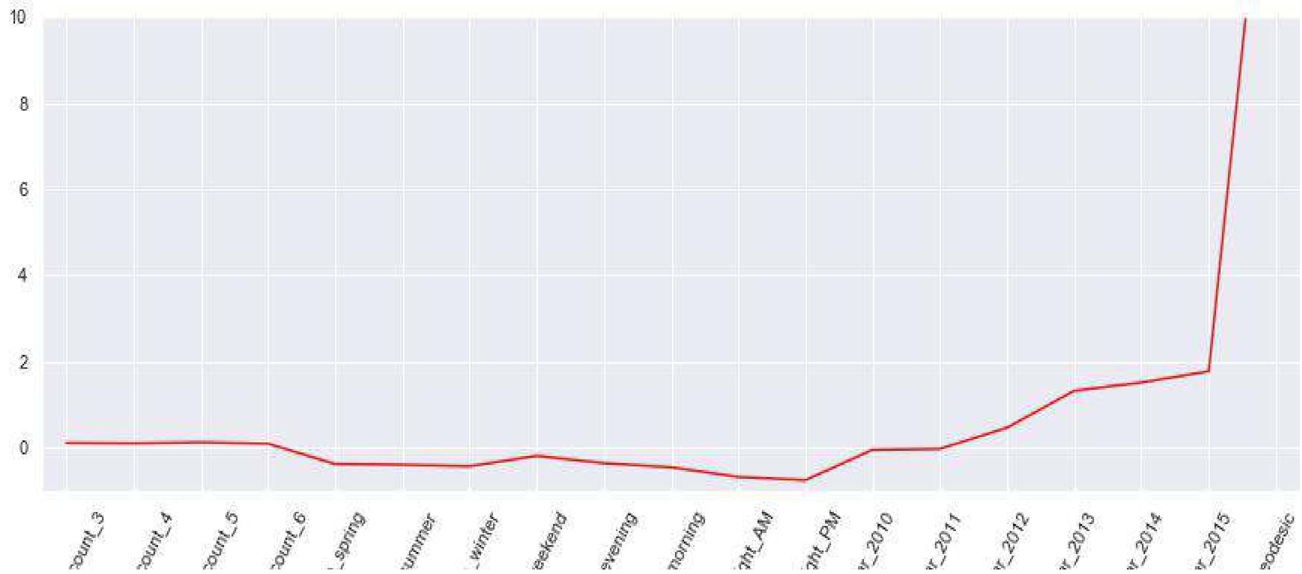
Line Plot for Coefficients of Ridge regression:



3. Lasso Regression:-

Error Metrics	R^2	Adj R^2	MAPE	MSE	RMSE	RMSLE
Train	0.519	0.518	22.23	7.53	2.74	0.255
Validation	0.462	0.458	22.80	8.31	2.88	0.267

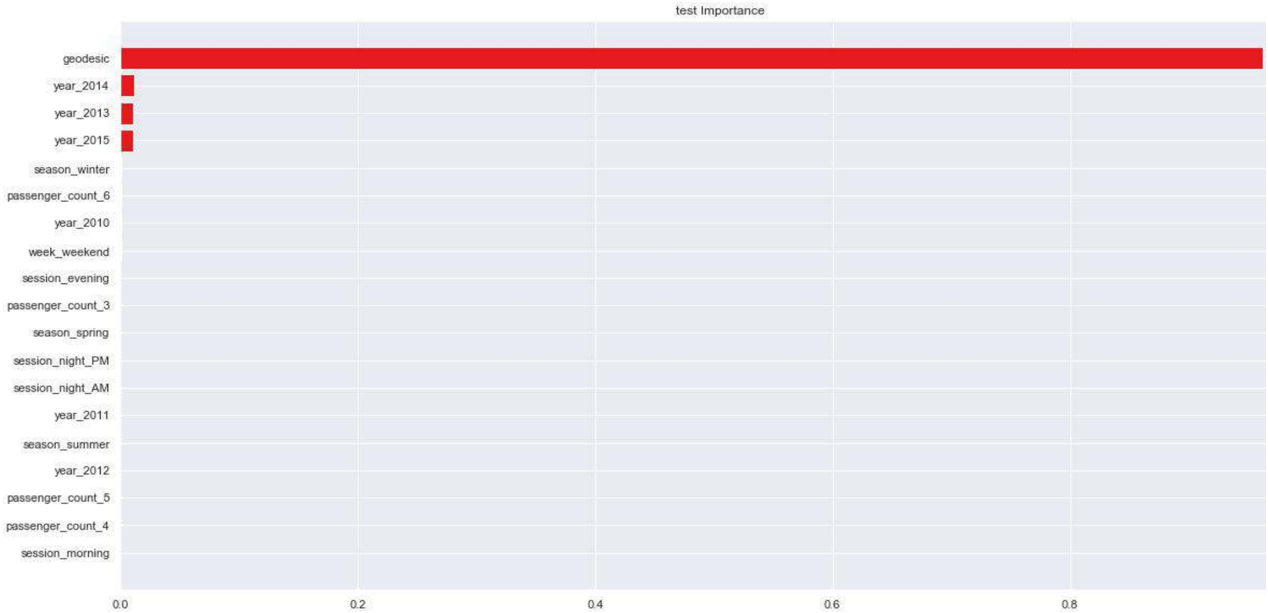
Line Plot for Coefficients of Lasso regression:



4. Decision Tree Regression:-

Error Metrics	R^2	Adj R^2	MAPE	MSE	RMSE	RMSLE
Train	0.585	0.584	20.83	6.50	2.54	0.23
Validation	0.528	0.525	21.84	7.28	2.69	0.24

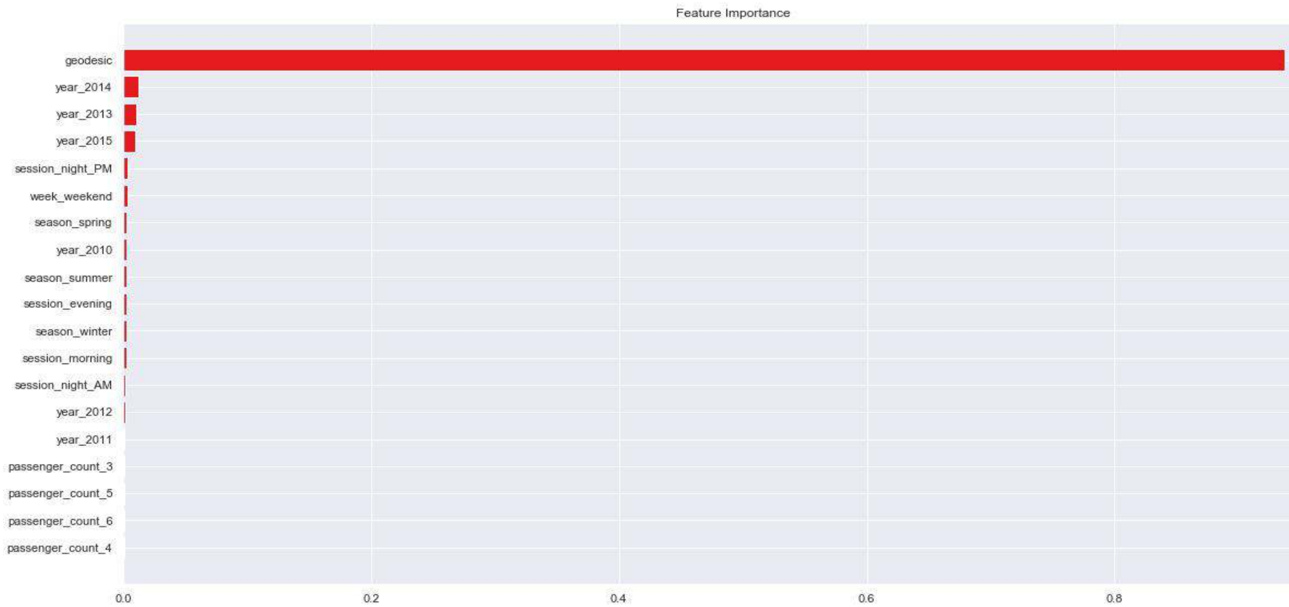
Line Plot for Coefficients of Decision Tree regression:



5. Random Forest Regression:-

Error Metrics	**R^2**	**Adj R^2**	**MAPE**	**MSE**	**RMSE**	**RMSLE**
Train	0.6598	0.6594	19.43	5.34	2.31	0.21
Validation	0.530	0.527	22.64	7.25	2.69	0.24

Line Plot for Coefficients of Random Forest Regression:




```
Cross- Validation Scores:[-6.67732762 -6.46167564 -6.55038413 -6.9730539
-6.972281  ]
Average 5-Fold CV Score: -6.7269444575011
```

Improving accuracy

- Improve Accuracy : - a) Algorithm Tuning b) Ensembles
- We have used **Xg boost** as an Ensemble Technique.

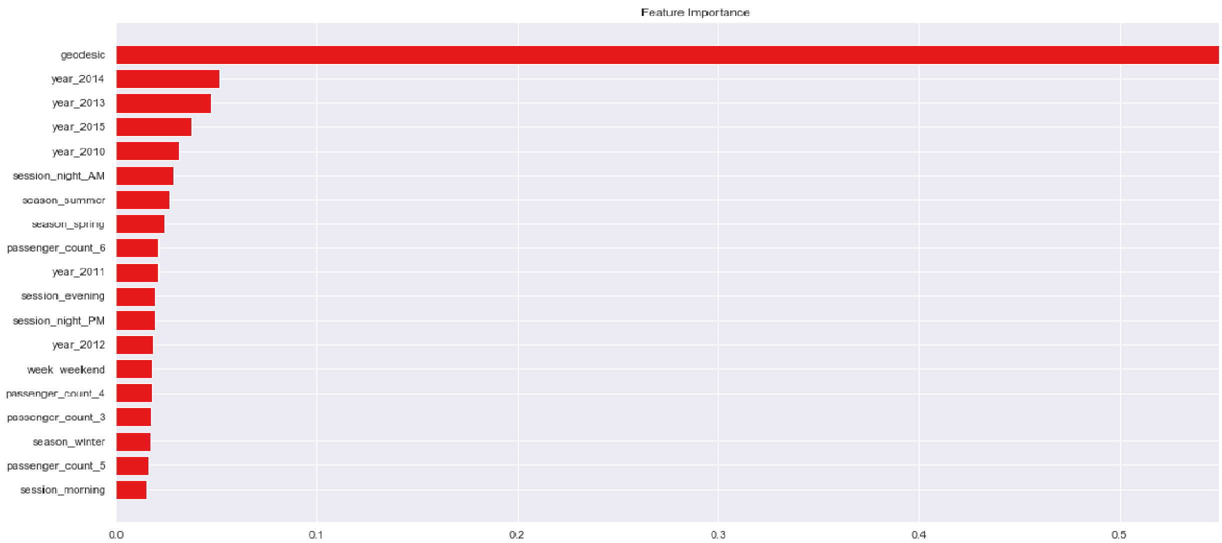
Xgboost hyperparameters tuned parameters:

Tuned Xgboost Parameters: {'subsample': 0.1, 'reg_alpha': 0.08685113737513521, 'n_estimators': 200, 'max_depth': 3, 'learning_rate': 0.05,'colsample_bytree': 0.7000000000000001, 'colsample_bynode': 0.7000000000000001,'colsample_bylevel': 0.9000000000000001}

Xg Boost Regression:-

Error Metrics	**R^2**	**Adj R^2**	**MAPE**	**MSE**	**RMSE**	**RMSLE**
Train	0.596	0.595	20.27	6.331	2.516	0.228
Validation	0.538	0.535	21.22	7.139	2.67	0.243

Bar Plot for Coefficients of Xg Boost Regression:-



Finalized model

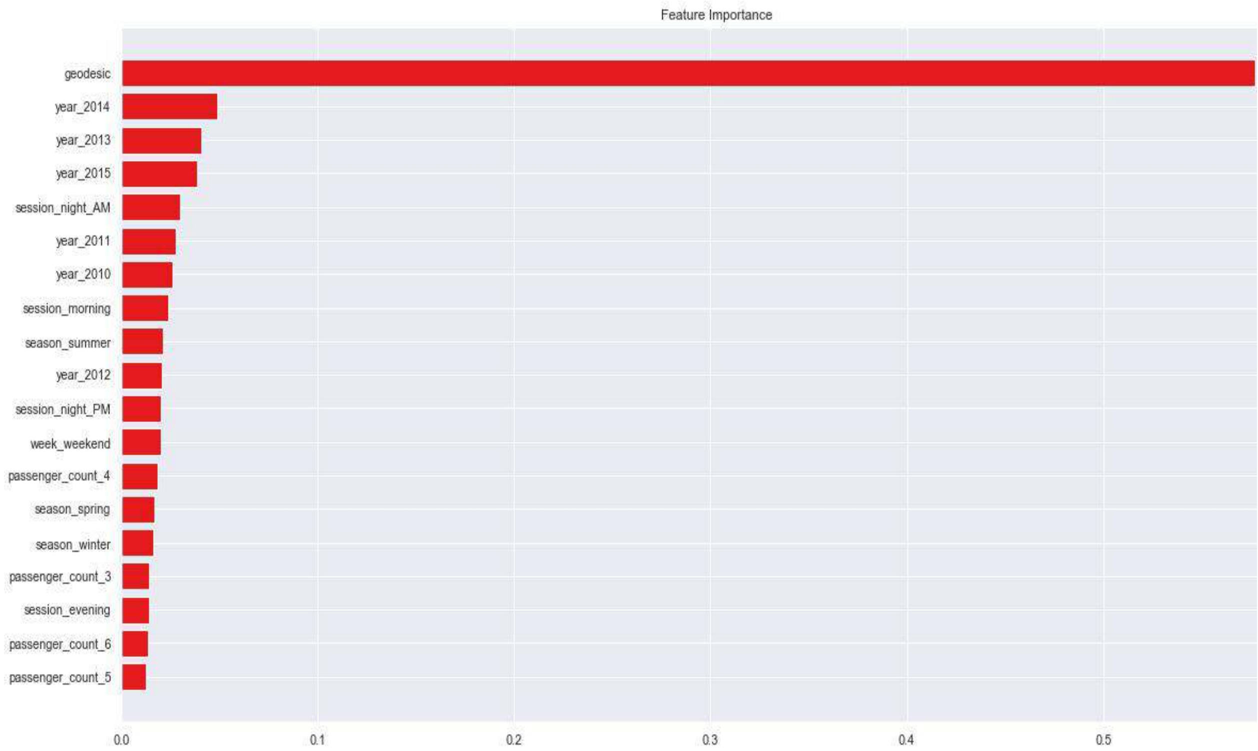
- Create standalone model on entire training dataset
- Save model for later use

We have trained a Xgboost model on entire training dataset and used that model to predict on test data.Also, we have saved model for later use.

```
<<<------------------- Training Data Score --------------------->

r square :{} 0.5544717216937789
Adjusted r square:0.5539163732633567
MAPE:22.338955482263938
MSE: 6.971021003568455
RMSE: 2.6402691157471914
RMSLE: 0.24305065499034056
```

Feature importance:



<u>Hence, we have predicted the fare_amount for test data with the help of Xg Boost Method.</u>