CAB FARE

KAPIL BHATT

20 May 2019

Index

Chapter 1

Introduction

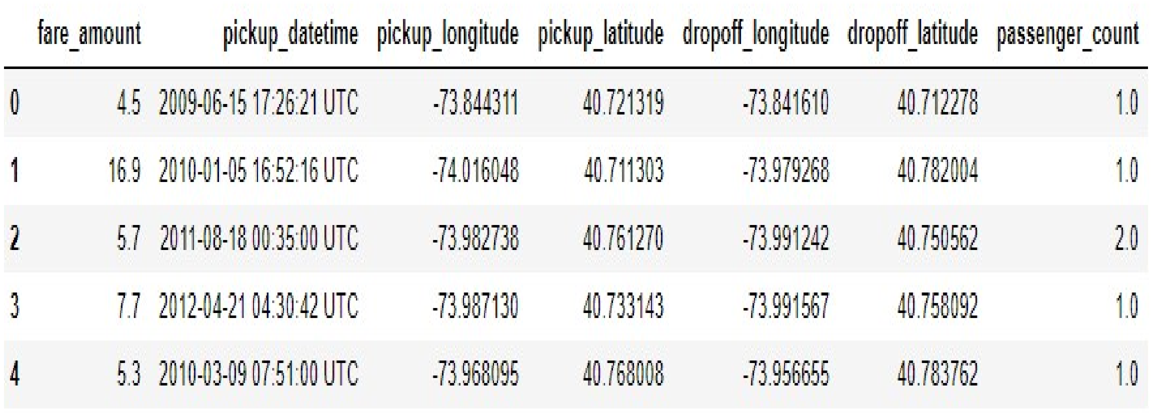
1.1 Problem Statement

The objective of this project is to forecast the fare of cabs on a given time based on the various time and number of passengers. Predicting the fare will help the passengers to get an estimate of their ride.

1.2 Data

Our task here is to build the regression model in order to predict the fare of cabs in a given time based on the predictors we had. Given below is the sample of data that we are using to predict the cab fare.

Table1.1 CAB FARE samples column 1 to 7.



	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	4.5	2009-06-15 17:26:21 UTC	-73.844311	40.721319	-73.841610	40.712278	1.0
1	16.9	2010-01-05 16:52:16 UTC	-74.016048	40.711303	-73.979268	40.782004	1.0
2	5.7	2011-08-18 00:35:00 UTC	-73.982738	40.761270	-73.991242	40.750562	2.0
3	7.7	2012-04-21 04:30:42 UTC	-73.987130	40.733143	-73.991567	40.758092	1.0
4	5.3	2010-03-09 07:51:00 UTC	-73.968095	40.768008	-73.956655	40.783762	1.0

As you can see in the table below we have the following 13 variables (two of them are cut out because there sum is in count). We have to use these variables to predict the cab fare and do other analysis:

S.no	**Predictor**
1	fare_amount
2	Pickup_datetime
3	Pickup_longitude
4	Pickup_latitude
5	Dropoff_longitude
6	Dropoff_latitude
7	Passenger_count

Table 1.2 : Number of Predictor variables

Chapter 2

Methodology

Preprocessing

Any predictive modeling requires to do exploratory analysis of data, make it clean and suitable to create various machine learning models. This step contains cleaning of data, creating visualization like graphs and plots to gain more insights.

2.1 Missing Value Analysis

In this analysis we found out the missing values in our data set and then how to deal with it. This dataset contains around 70-80 missing values of different variables which is quite less compare to whole data set. So we have remove them. We can also impute them according to the problem statement and number of missing values.

Also in the dataset the variables passenger_count and fare_amount have some unusual values. For example maximum number of passengers can be 6, but there are cases where count is greater than 6 or zero so we remove them. Same in case of fare amount as there are some cases where fare amount is 0.

2.2 Outlier Analysis

Outliers are the values that are distant from the main observations. They are detected by boxplots.

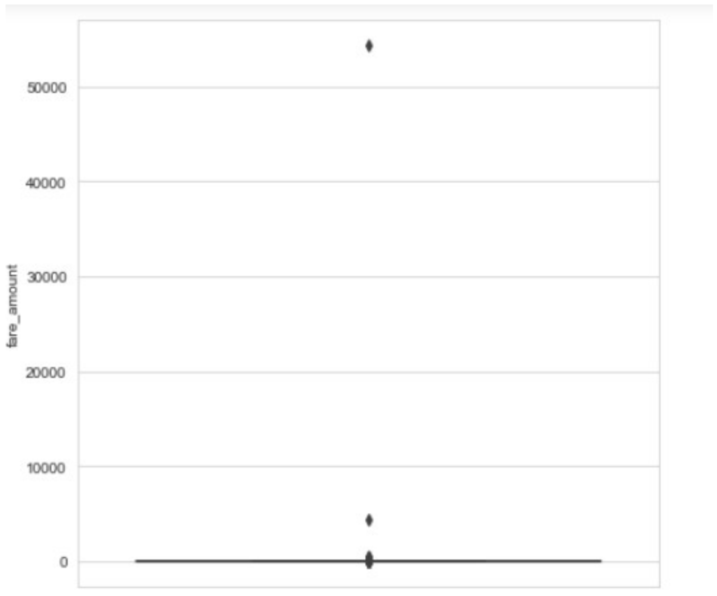


Fig 2.1 Box plot of fare_amount

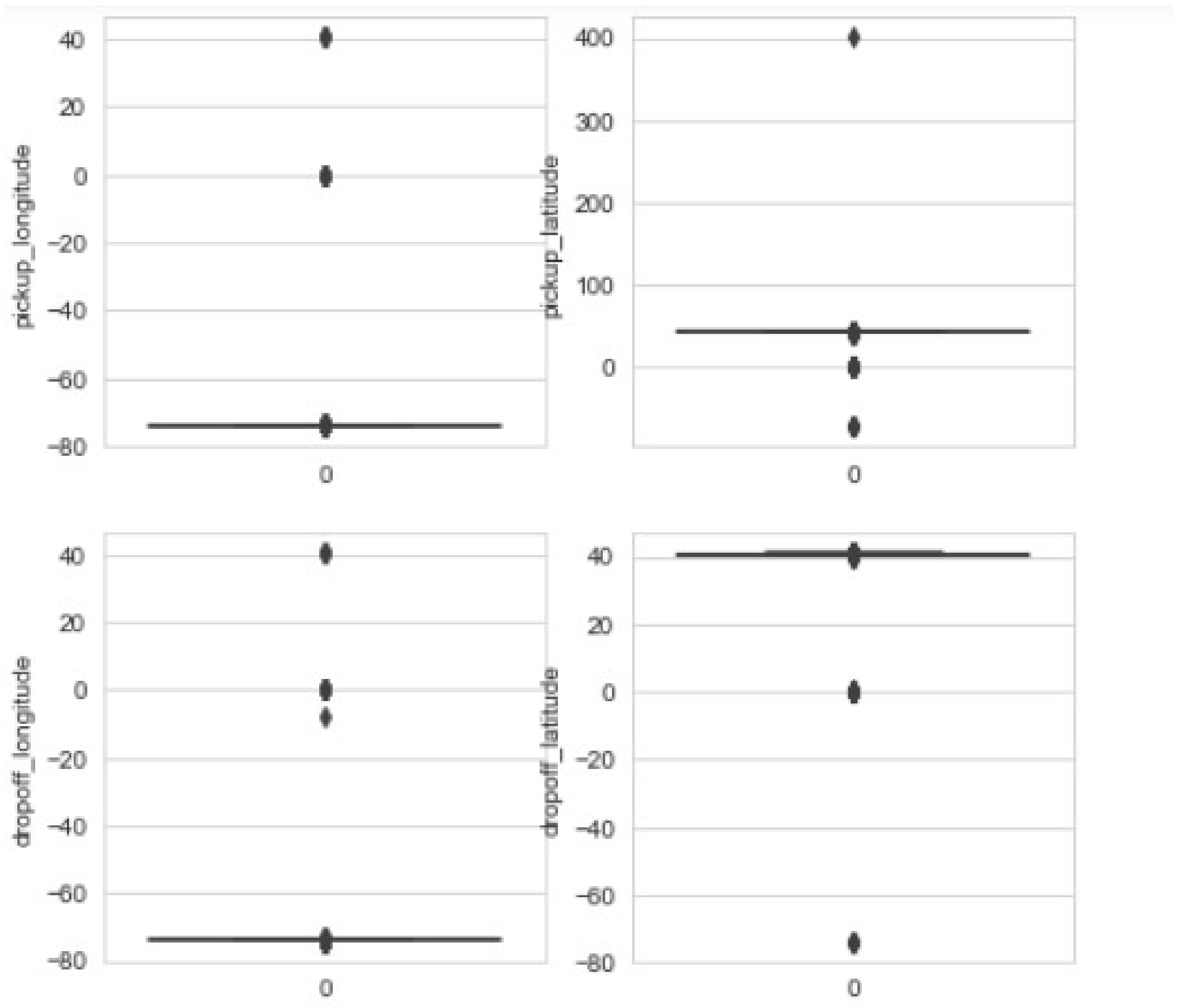


Fig 2.2 Box plot of Continous Variables

As you can see here there are many outliers in each plot. They can be removed by creating a function which calculates the interquartile range and minimum and maximum value of the variable. The values which lie beyond the minimum and maximum value is removed as they are outliers.

2.3 Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. Feature engineering is fundamental to the application of machine learning.

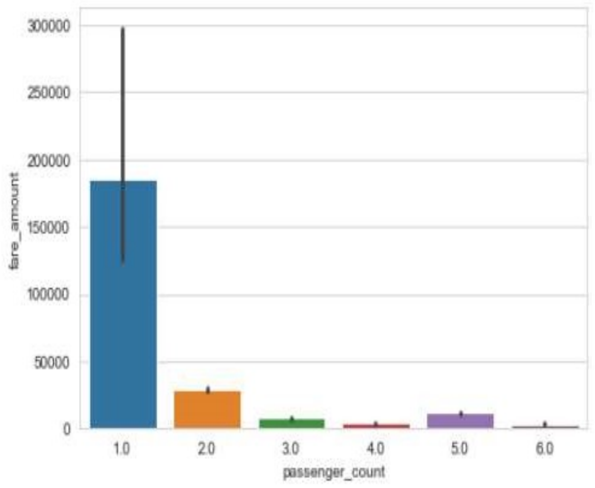
Here in this data we created some feature. From pickup_datetime we created

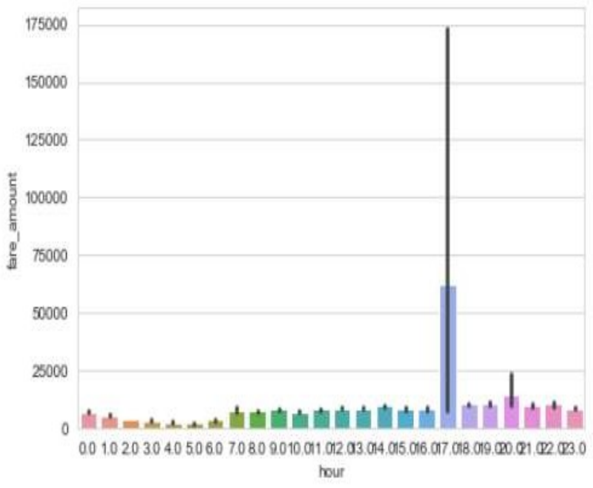
- Hour
- Day of week
- month
- year
- weekend or not
- peak hours

From the longitude and latitude we find the distance of a cab ride using haversine formula of calculating distance using latitude and longitude.

2.4 Bivariate Analysis (Distribution of Categorical variables with Fare)

This analysis is done by plotting bar graphs of categorical variables with respect to the numerical variable (Fare Amount).





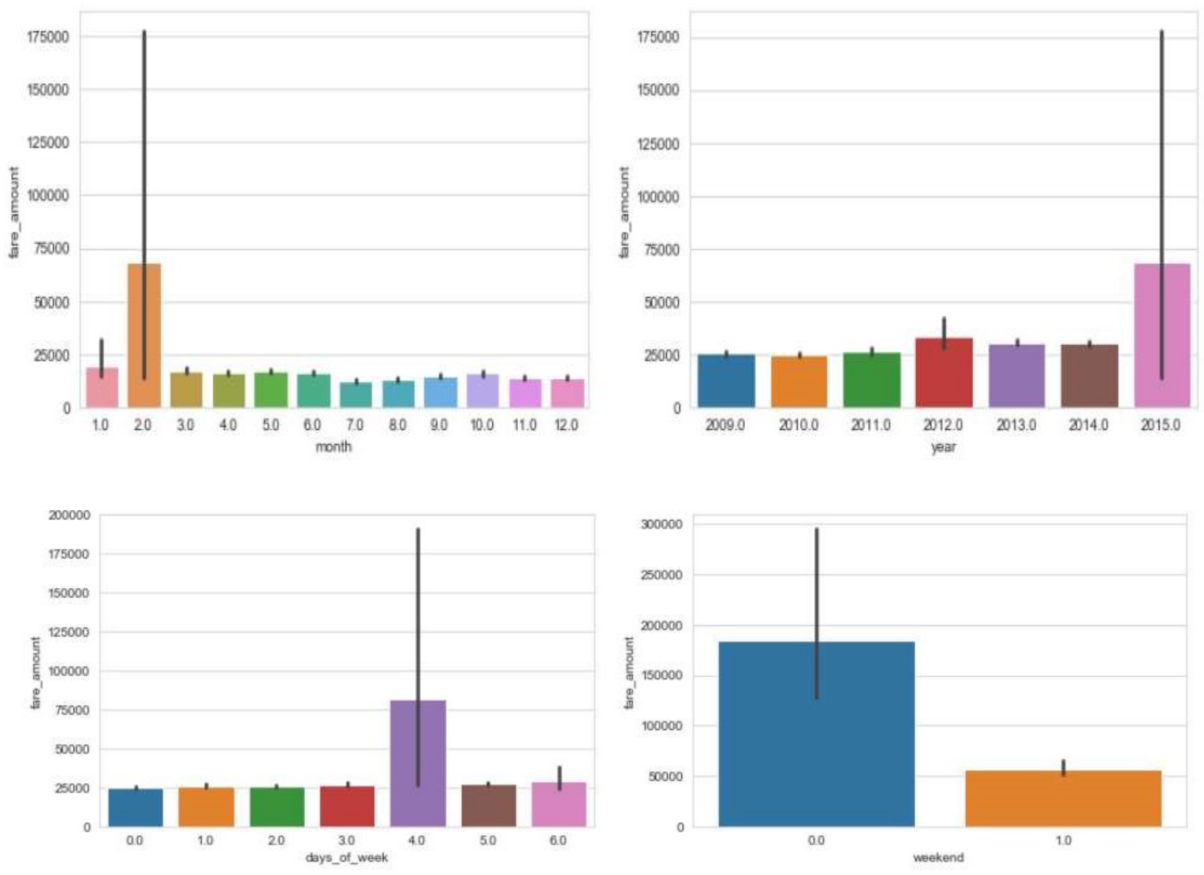
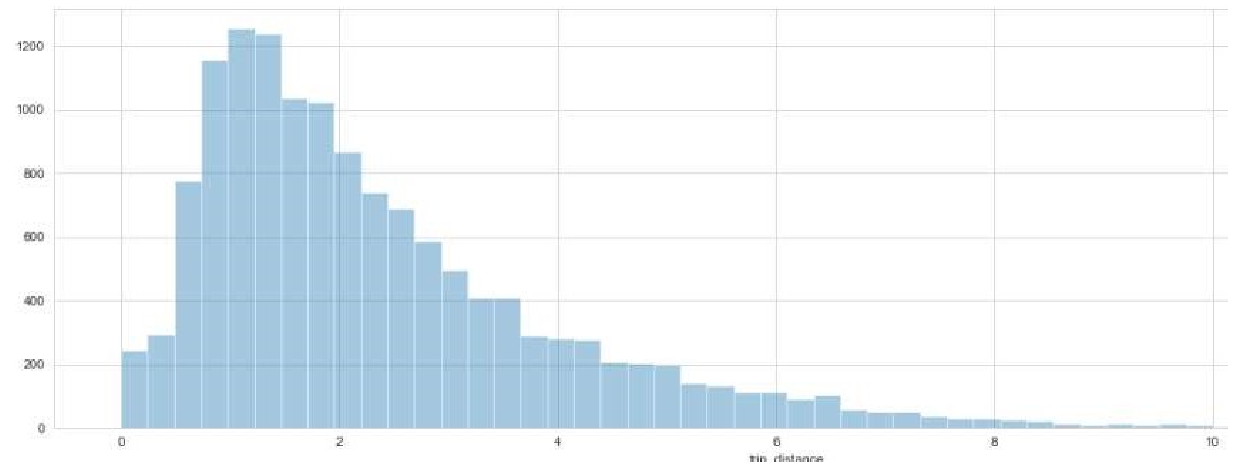


Fig 2.3 Bar Plot of categorical variables with respect to the fare_amount

The bar charts here tells about how the different categories is creating effect on the fare of the cabs. Let's take a look at first bar graph of passenger_count vs. fare amount as you can see that Single passenger contributes to more fare than rides with multiple passengers. Also from Year bar graph we get to know that fare is increasing in 2015 compare to rest of the years. The bar graph of hours here shows that the maximum fare amount comes at 5 pm which is the peak time as most of the offices closes at that time.

2.5 Histograms and Scatterplots

As you can see below the fare_amount is almost following normal distribution while the trip distance is skewed.This is maybe due to presence of data in extreme ends or outliers.



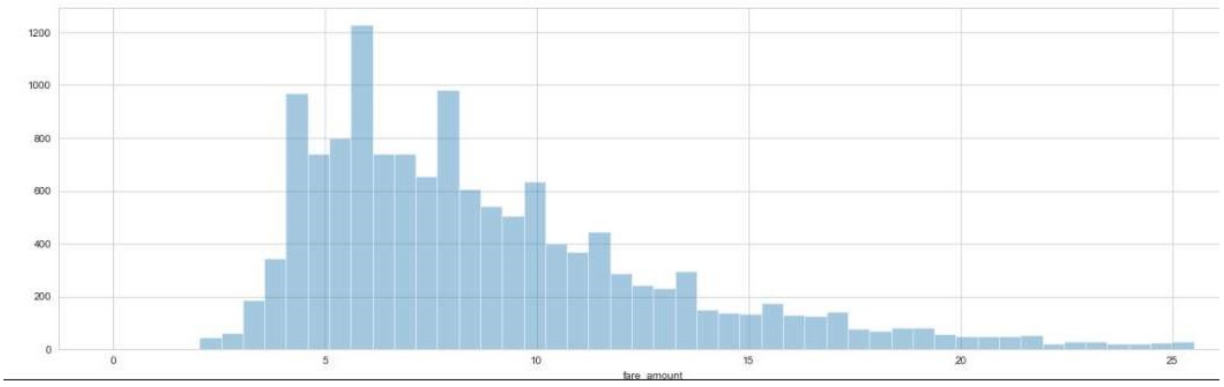


Fig 2.4 Histogram of trip_distance and fare_amount

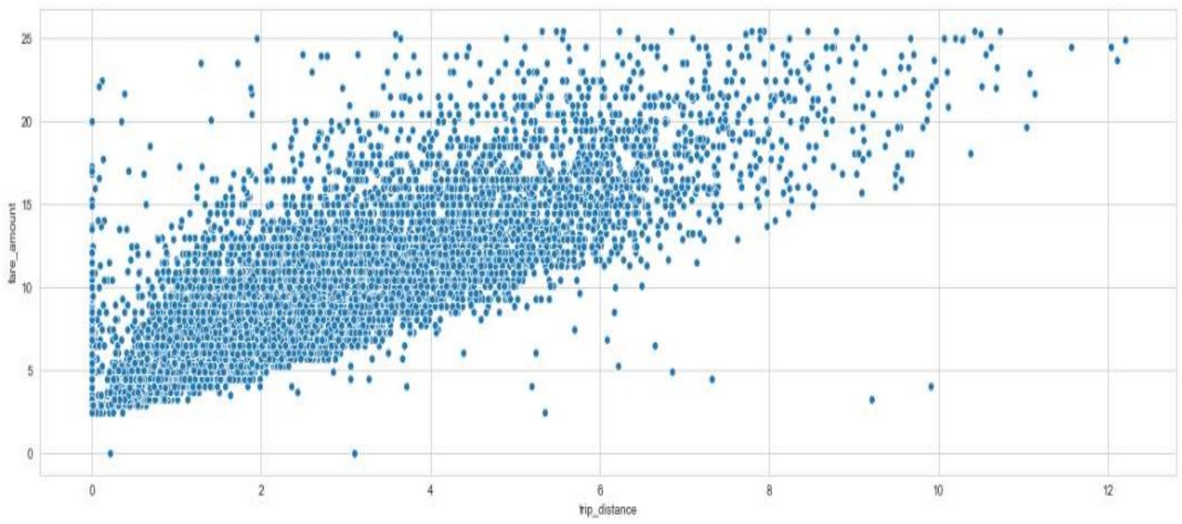


Fig 2.5 Scatter plot between fare amount and trip distance

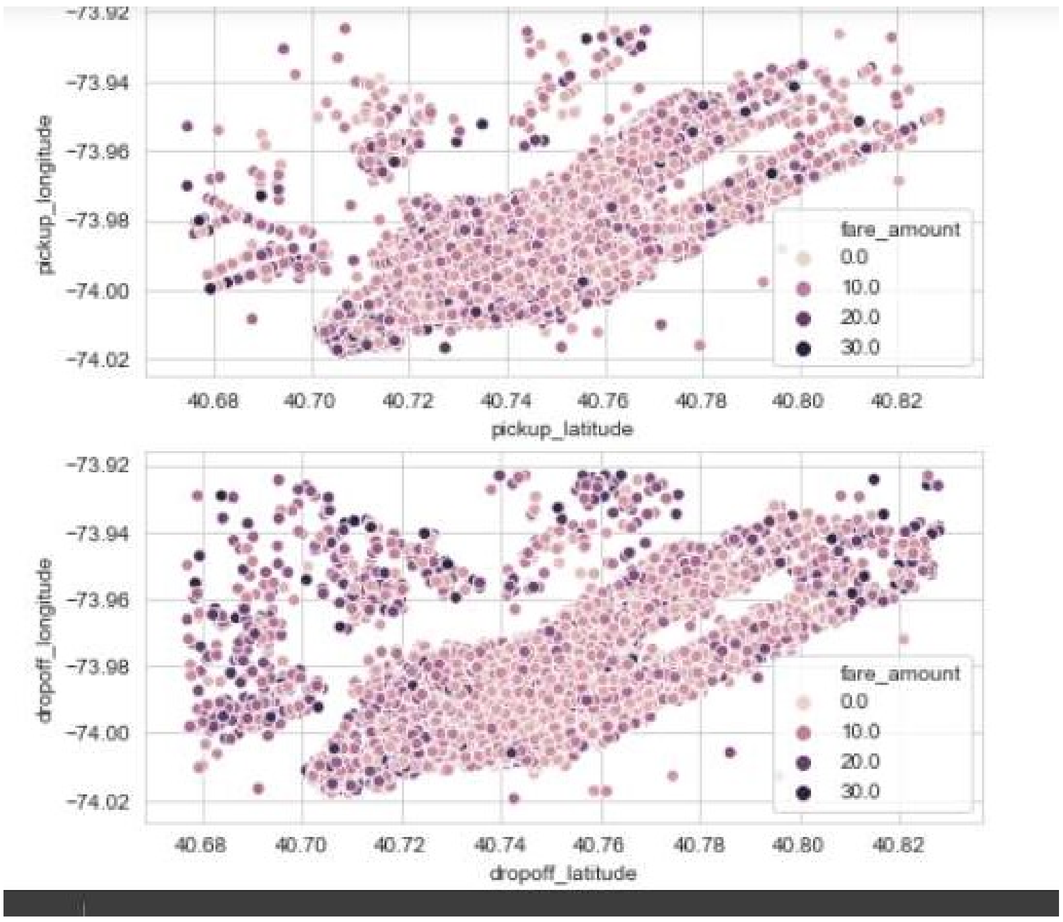


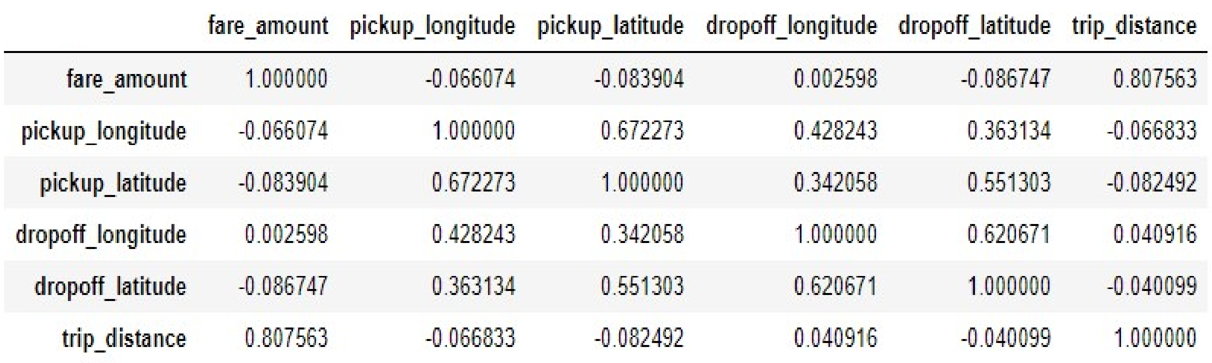
Fig 2.6 Scatter plot of longitudes and Latitudes with respect to the fare_Amount

Scatter plot of fare amount and trip distance is showing a highly positive relationship between them and this is further confirmed below in correlation matrix.

2.6 Feature Selection:

Feature selection is the most important step of data preprocessing. It is the process where you manually or automatically select the features which have contributed most to your prediction variable. It also reduces over fitting, increases accuracy and save some time to train the model because the data points became few after that.

In this data we uses the correlation scores to check the relationship between independent continuous variables and then remove those variables which have strong relationships among them. we use the correlation matrix, we can also use the correlation plot. For categorical variables we uses the ANOVA (Analysis of Variance) test in order to check their significance.



	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	trip_distance
fare_amount	1.000000	-0.066074	-0.083904	0.002598	-0.086747	0.807563
pickup_longitude	-0.066074	1.000000	0.672273	0.428243	0.363134	-0.066833
pickup_latitude	-0.083904	0.672273	1.000000	0.342058	0.551303	-0.082492
dropoff_longitude	0.002598	0.428243	0.342058	1.000000	0.620671	0.040916
dropoff_latitude	-0.086747	0.363134	0.551303	0.620671	1.000000	-0.040099
trip_distance	0.807563	-0.066833	-0.082492	0.040916	-0.040099	1.000000

There is a strong relationship between pickup_latitude and pickup_longitude and also dropoff_longitude and latitude because of their correlation score is too high(multicolinearity). so they both are contributing same thing to the dependent variable. Therefore we can remove pickup_longitude , dropoff_longitude from the dataset.

ANOVA

Analysis of variance (ANOVA) is a collection of statistical models and their associated estimation procedures (such as the "variation" among and between groups) used to analyze the differences among group means in a sample.

ANOVA test is used for checking the significance between a numeric and a categorical variable. Here in this test we perform the ANOVA test and find out that day of week and weekend is not making significant contribution.

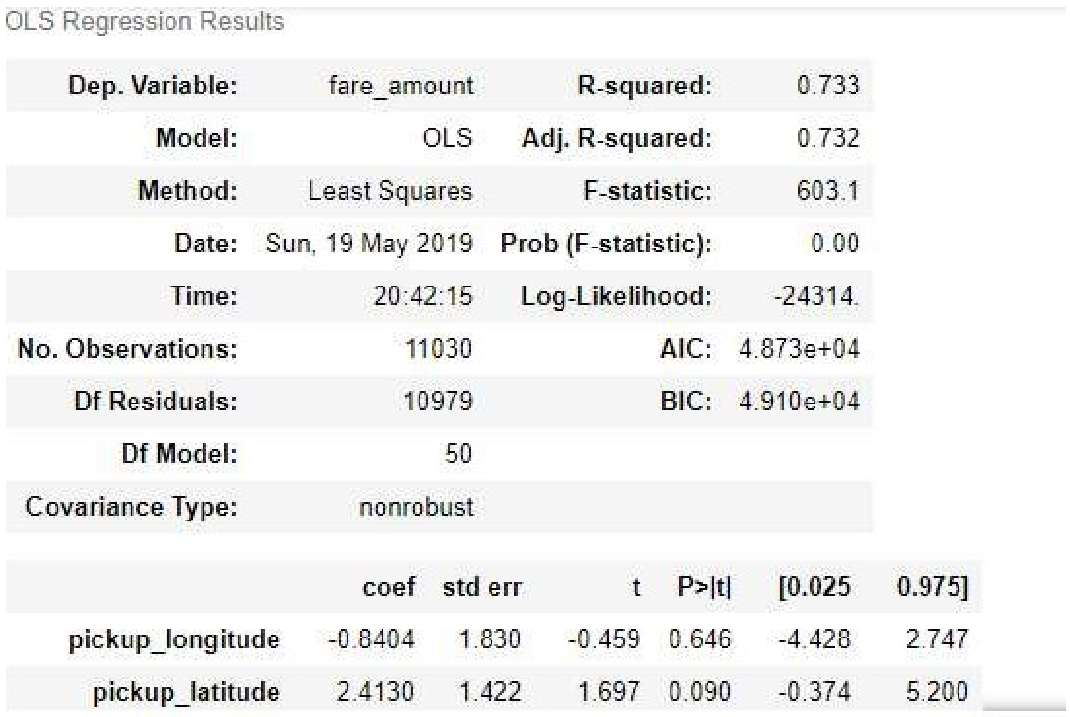
Chapter 3

Modeling

3.1 Model Selection

In the problem we need to predict the fare of cabs which is a continuous variable. So it falls under the forecasting problems. The models we use here are of regressions such as Linear Regression, Decision Tree Regressor and RandomForest. There are many error metrics are available but in this model we will compare the models in terms of R square and mean average error (MAE).

3.2 Multiple Linear Regression



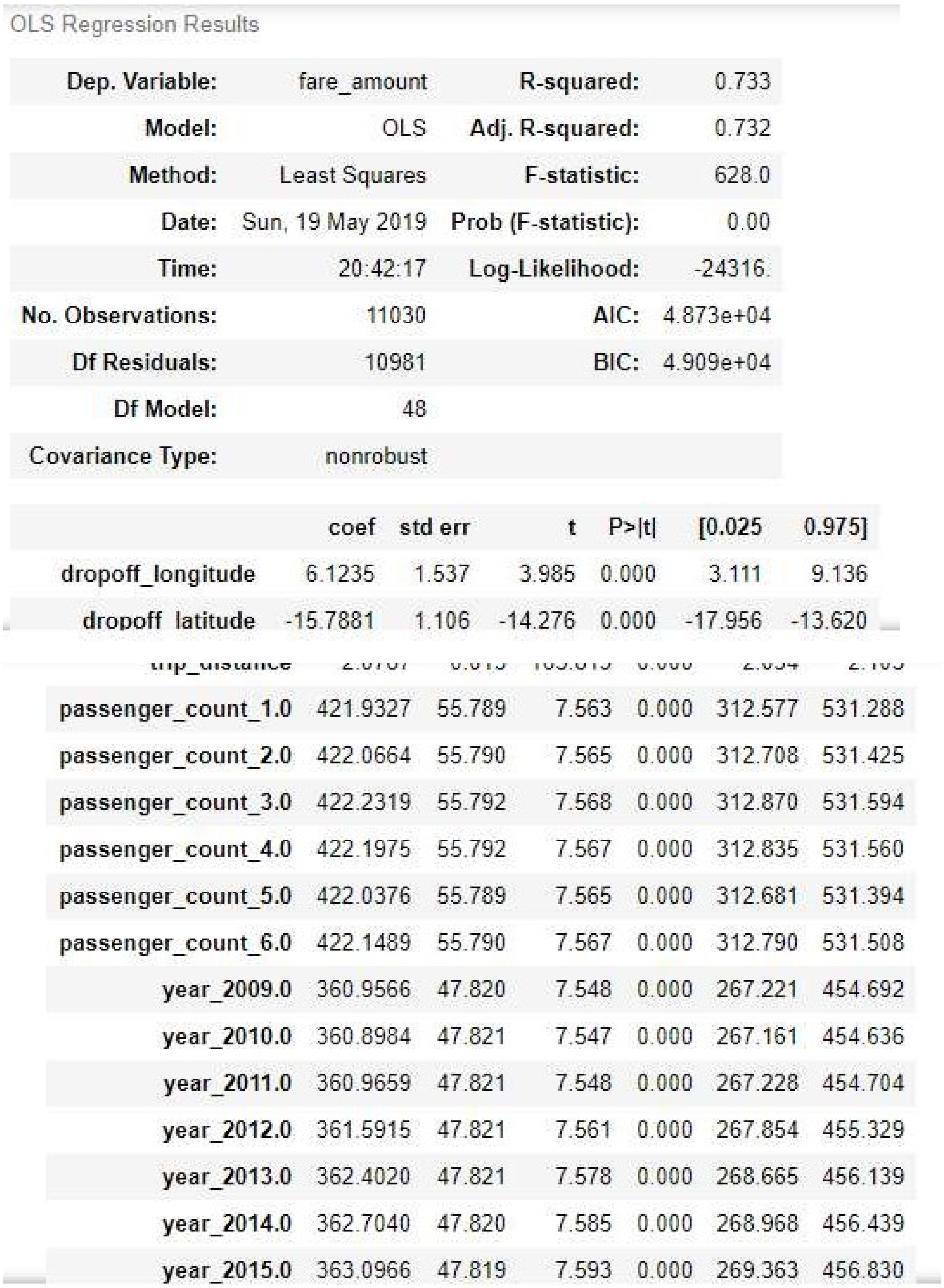
OLS Regression Results

Dep. Variable:	fare_amount	**R-squared:**	0.733
Model:	OLS	**Adj. R-squared:**	0.732
Method:	Least Squares	**F-statistic:**	603.1
Date:	Sun, 19 May 2019	**Prob (F-statistic):**	0.00
Time:	20:42:15	**Log-Likelihood:**	-24314.
No. Observations:	11030	**AIC:**	4.873e+04
Df Residuals:	10979	**BIC:**	4.910e+04
Df Model:	50		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
pickup_longitude	-0.8404	1.830	-0.459	0.646	-4.428	2.747
pickup_latitude	2.4130	1.422	1.697	0.090	-0.374	5.200

FIG 3.1 Multiple Regression Results before improving.

Multiple linear regression tells the relationship between one continuous variables to two or more continuous or categorical independent variables. In the above model R2 Square is 0.733 means we can explain 73.5% of the variance in fare amount from input variables and by looking the p value we can say that this model explained the data well.We can further improve this by discarding the variables which are less significant and their p value is high i.e pickup_latitude and dropoff_latitude.



OLS Regression Results

Dep. Variable:	fare_amount	R-squared:	0.733
Model:	OLS	Adj. R-squared:	0.732
Method:	Least Squares	F-statistic:	628.0
Date:	Sun, 19 May 2019	Prob (F-statistic):	0.00
Time:	20:42:17	Log-Likelihood:	-24316.
No. Observations:	11030	AIC:	4.873e+04
Df Residuals:	10981	BIC:	4.909e+04
Df Model:	48		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
dropoff_longitude	6.1235	1.537	3.985	0.000	3.111	9.136
dropoff_latitude	-15.7881	1.106	-14.276	0.000	-17.956	-13.620
trip_distance	2.0787	0.013	165.613	0.000	2.054	2.103
passenger_count_1.0	421.9327	55.789	7.563	0.000	312.577	531.288
passenger_count_2.0	422.0664	55.790	7.565	0.000	312.708	531.425
passenger_count_3.0	422.2319	55.792	7.568	0.000	312.870	531.594
passenger_count_4.0	422.1975	55.792	7.567	0.000	312.835	531.560
passenger_count_5.0	422.0376	55.789	7.565	0.000	312.681	531.394
passenger_count_6.0	422.1489	55.790	7.567	0.000	312.790	531.508
year_2009.0	360.9566	47.820	7.548	0.000	267.221	454.692
year_2010.0	360.8984	47.821	7.547	0.000	267.161	454.636
year_2011.0	360.9659	47.821	7.548	0.000	267.228	454.704
year_2012.0	361.5915	47.821	7.561	0.000	267.854	455.329
year_2013.0	362.4020	47.821	7.578	0.000	268.665	456.139
year_2014.0	362.7040	47.820	7.585	0.000	268.968	456.439
year_2015.0	363.0966	47.819	7.593	0.000	269.363	456.830

Fig 3.2 Linear Regression after optimizantion

In Above after removing the variables we find out that our f1 score has improved and thus making it more significant than previous model.

3.2 Decision Trees

Decision tree is used for both classification and regression problems. It uses a tree like model where each branch connected with "and" and multiple branches are connected with "or". There are two methods of splitting the tree one is by Information gain and other is by Gini Index for classification problems. For regression splitting of tree is done by analysis of variance or ANOVA. Here the R2_SQUARED value comes out to be 0.699 which means it explained 69.9% of the data from input variables which is better than Linear Regression.

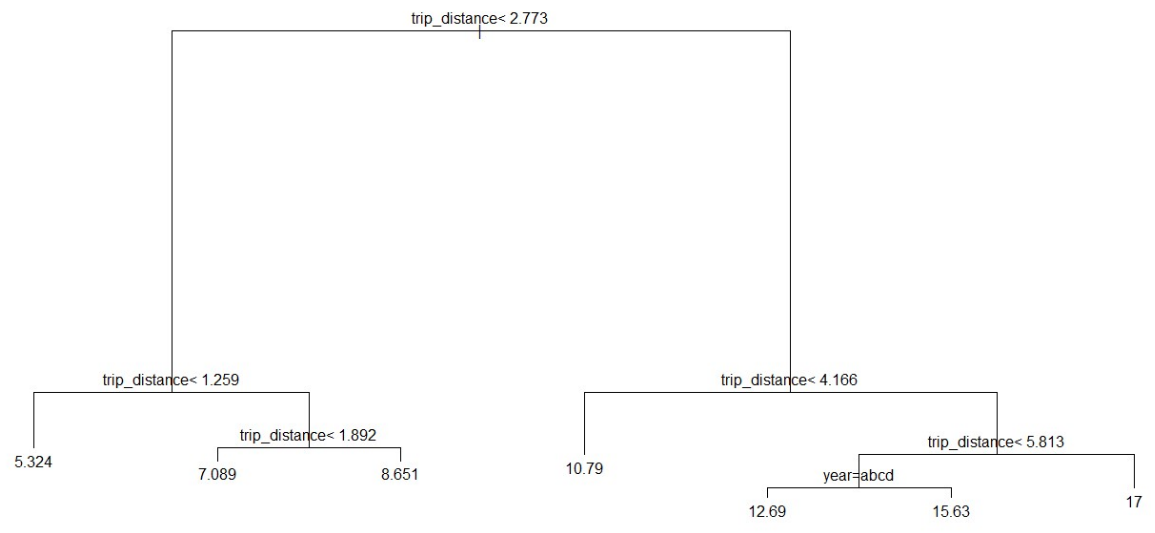


Fig 3.3 Plot of the Decision Tree model.

3.3 Random Forest

Random forest use number of decision trees and then calculate the result based on mean and mode. In classification it uses mode of class output by individual trees while in regression it uses the mean to get the results. It is the combination of weak methods and it feed error from one decision tree to another to get the better results. In predictions a sample is iterated all the trees and the mean vote of all the trees is considered. In our model the number of decision tree we use to create random forest model is 200 and we get R square value of 0.730 which is 73% of the data is explained by the input variables.

Chapter 4

Conclusion

4.1 Model Evaluation

Now that we have a few models for predicting the target variable, we need to decide which one 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 of cab fare the last two don't hold much significance. So we stick to the predictive performance or evaluation of our models using Root mean Squared error and R Squared Value.

Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure.

Root Mean Squared Error (RMSE)

RMSE is one of the error measures used to calculate the predictive performance of the model. We will apply this measure to our models that we have build. It is present in the metrics library of sklearn.

R SQUARED VALUE :

R Squared value is the variance within the dependent variable explained by the independent variables. It is also used as a metrics to determine the significance of the model. Adjusted R squared value is the adjusted R Squared based on the number of variables or independent variables.

Other Metrics include p-value, f-score, AIC, log likelihood.

So based on the above metrics following are the results:

- Linear Regression : RMSE - 1.99 R SQUARED = 73.33
- Decision Tree : RMSE - 2.29% -> ACCURACY = 69.9
- Random Forest : RMSE - 2.29% -> ACCURACY = 75.33

Based on the above results we can conclude that Random Forest is the better model If we consider R Squared value for our Analysis. If we consider RMSE then linear regression is the better model to choose. So we can tradeoff between two in this problem.

CHAPTER 5

APPENDIX

5.1 SOME R GRAPHS AND PREDICTIONS

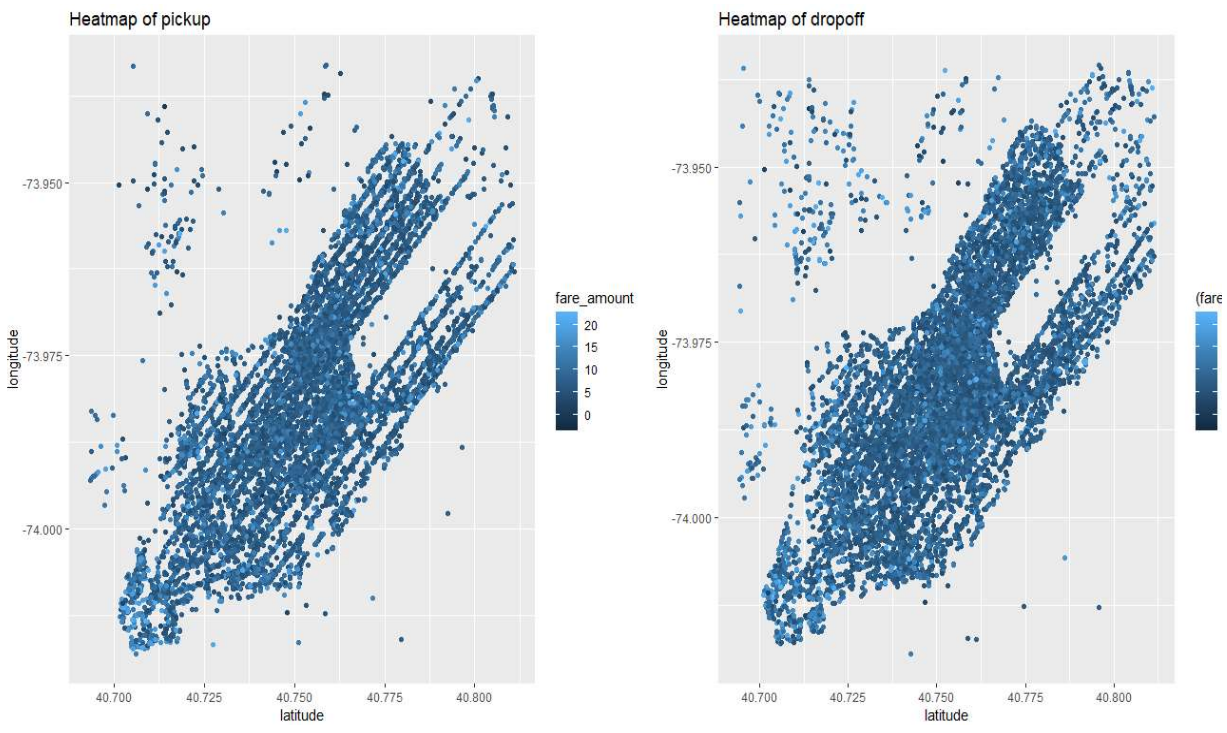
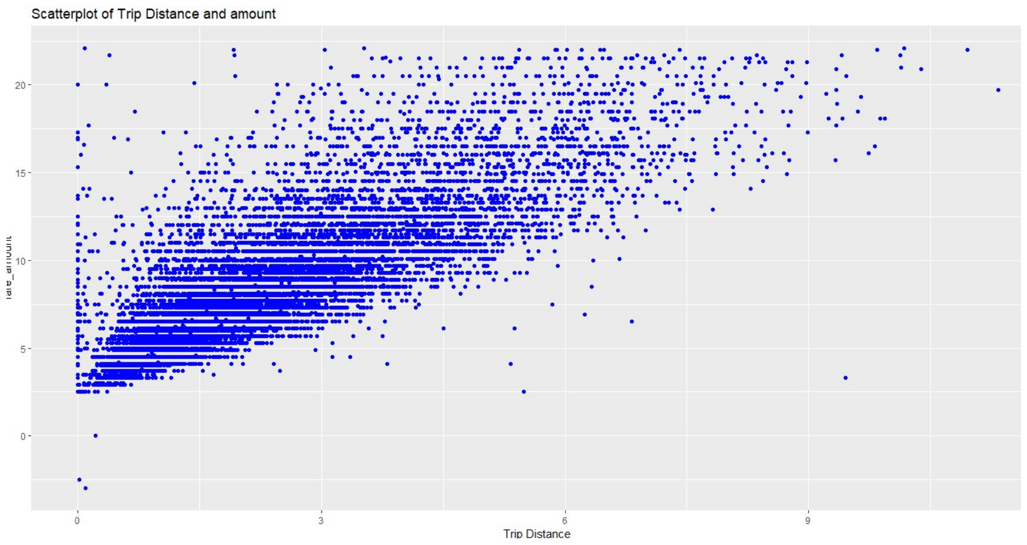


FIG 5.1 Heatmap of longitude and lattitudes with respect to fare amount. In below scatterplot of trip distance and fare amount



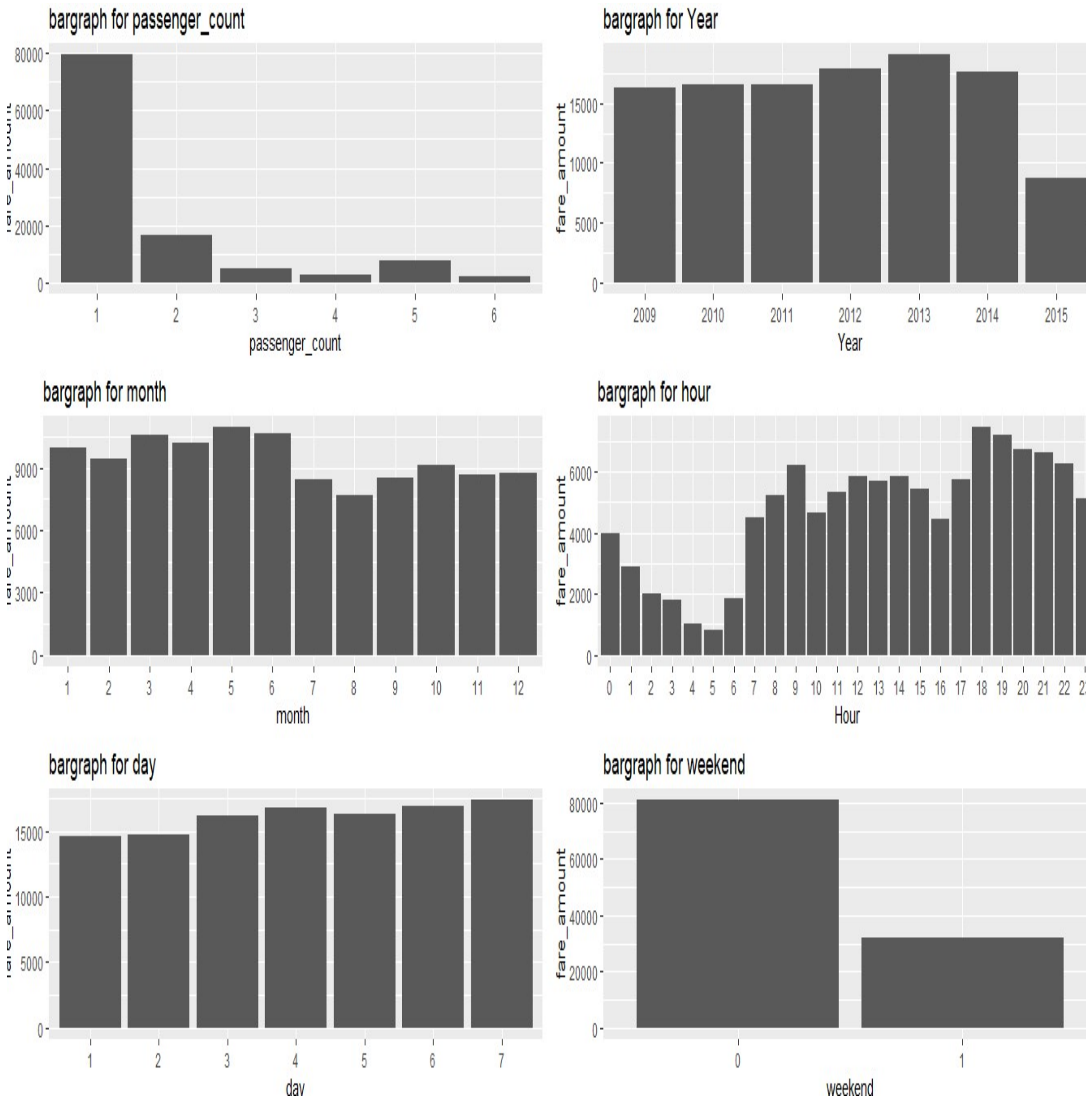


FIG 5.2 Bargraph of different categorical Variables in R

R PREDICTIONS:

The model which I got best in R is Random forest which have R SQUARED value of around 0.95. and RMSE error is also less. The model and metrics in R as follows:

- Linear Regression: R SQUARED- 0.7249, RMSE - 1.99
- Decision Tree : R SQUARED- 0.689, RMSE - 2.12
- Random Forest R SQUARED- 0.95, RMSE - 0.966

5.2 Test Predictions and batch files

The test data which came along with the actual data, I have processed it and predicted with the best model in R and Python. For python it is Linear Regression and for R its Random Forest.

I have included three batch files

- cabfarer.bat -FOR R CODE. Just run it and it will generate a ROut file with outputs
- runpy.bat - For python code. Using runipy it will generate a new notebook with outputs
- runpython.bat - For python code. using nbconvert, it will also generate a new notebook

Put all the files and data together with batch files and run them. If there is an error it's because of the uninstalled packages and old versions of packages.

CHAPTER 6

PYTHON CODE

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

cab_fare = pd.read_csv("train_cab.csv")

warnings.filterwarnings("ignore")

cab_fare.head()

cab_fare.shape

cab_fare['fare_amount'] = cab_fare['fare_amount'].replace('-','',regex = True).astype('float64')

cab_fare.dtypes

cab_fare.describe()

cab_fare['passenger_count'].value_counts()

cab_fare.dtypes

cab_fare['pickup_datetime'] = pd.to_datetime(cab_fare['pickup_datetime'], errors = 'coerce',
utc=True)

cab_fare.dtypes

cab_fare['year'] = cab_fare['pickup_datetime'].dt.year

cab_fare['month'] = cab_fare['pickup_datetime'].dt.month

cab_fare['days_of_week'] = cab_fare['pickup_datetime'].dt.dayofweek

cab_fare['hour'] = cab_fare['pickup_datetime'].dt.hour

cab_fare['weekend'] = cab_fare['days_of_week'].map({0:0,1:0,2:0,3:0,4:0,5:1,6:1})

a = [9,10,11,12,17,18,19,20]

cab_fare['peak_hour'] = cab_fare['hour'].apply(lambda x: 1 if x in a else 0)

cab_fare['peak_hour'].value_counts()
```

```
cab_fare.dtypes

cab_fare.isnull().sum()

cab_fare.dropna(axis = 0, how = 'any',inplace = True)

#cab_fare.drop(cab_fare[cab_fare['passenger_count'] > 6],axis=0,inplace = True)

cab_fare.isnull().sum()

cab_fare = cab_fare.loc[~((cab_fare['passenger_count']==0)|(cab_fare['passenger_count']>6) |
(cab_fare['passenger_count']==0.12) | (cab_fare['passenger_count']==1.30))]

cab_fare.shape

for i in range (-6,0):

    print (cab_fare.columns[i])

    print (cab_fare.iloc[:,i].value_counts())

for i in range (-7,0):

   cab_fare.iloc[:,i] = cab_fare.iloc[:,i].astype('category')

cab_fare.iloc[1016]

cab_fare.dtypes

#univariate analysis

cab_fare.iloc[1013,:]

sns.set_style("whitegrid")

plt.figure(figsize = (15,15))

plt.subplot(3,2,1)

sns.barplot("passenger_count","fare_amount",estimator = sum, data=cab_fare)

plt.subplot(3,2,2)

sns.barplot("hour","fare_amount",estimator = sum, data=cab_fare)

plt.subplot(3,2,3)

sns.barplot("month","fare_amount",estimator = sum, data=cab_fare)

plt.subplot(3,2,4)

sns.barplot("year","fare_amount",estimator = sum, data=cab_fare)
```

```
plt.subplot(3,2,5)

sns.barplot("days_of_week","fare_amount",estimator = sum, data=cab_fare)

plt.subplot(3,2,6)

sns.barplot("weekend","fare_amount",estimator = sum, data=cab_fare)

plt.figure(figsize = (15,15))

sns.barplot("peak_hour","fare_amount",estimator = sum, data=cab_fare)

plt.figure(figsize = (7,7))

sns.boxplot(cab_fare.iloc[:,0],orient ='v')

plt.figure(figsize = (7,7))

plt.subplot(2,2,1)

sns.boxplot(data = cab_fare['pickup_longitude'],orient ='v',)

plt.ylabel('pickup_longitude')

plt.subplot(2,2,2)

sns.boxplot(data = cab_fare['pickup_latitude'],orient ='v')

plt.ylabel('pickup_latitude')

plt.subplot(2,2,3)

sns.boxplot(data = cab_fare['dropoff_longitude'],orient ='v')

plt.ylabel('dropoff_longitude')

plt.subplot(2,2,4)

sns.boxplot(data = cab_fare['dropoff_latitude'],orient ='v')

plt.ylabel('dropoff_latitude')

var=["fare_amount","pickup_latitude","pickup_longitude","dropoff_latitude","dropoff_longitu
de"]

for i in var:

   q_25,q_75 = np.percentile(cab_fare[i],[25,75])

   iqr = q_75 -q_25

   max_out = q_75 + (iqr*2)
```

```
    min_out = q_25 - (iqr*2)
    cab_fare = cab_fare.drop(cab_fare[cab_fare[i]>max_out].index)
    cab_fare = cab_fare.drop(cab_fare[cab_fare[i]<min_out].index)
cab_fare.shape
plt.figure(figsize = (7,7))
plt.subplot(2,1,1)
sns.scatterplot("pickup_latitude","pickup_longitude",hue = "fare_amount",data= cab_fare)
plt.subplot(2,1,2)
sns.scatterplot("dropoff_latitude","dropoff_longitude",hue = "fare_amount",data= cab_fare)
def trip_distance(lat1, lat2, lon1,lon2):
    p = 0.017453292519943295 # Pi/180
    a = 0.5 - np.cos((lat2 - lat1) * p)/2 + np.cos(lat1 * p) * np.cos(lat2 * p) * (1 - np.cos((lon2 -
lon1) * p)) / 2
    return 0.6213712 * 12742 * np.arcsin(np.sqrt(a))
cab_fare['trip_distance']=cab_fare.apply(lambda
row:trip_distance(row['pickup_latitude'],row['dropoff_latitude'],row['pickup_longitude'],row['d
ropoff_longitude']),axis=1)
cab_fare['trip_distance'] = cab_fare['trip_distance']*1.57 #conversion to kms
plt.figure(figsize = (20,20))
plt.subplot(3,1,1)
sns.distplot(cab_fare['trip_distance'],kde = False)
plt.subplot(3,1,2)
sns.distplot(cab_fare['fare_amount'],kde = False)
plt.subplot(3,1,3)
sns.scatterplot('trip_distance','fare_amount',data = cab_fare)
from scipy.stats import shapiro
print (shapiro(cab_fare['fare_amount']))
print (shapiro(cab_fare['trip_distance']))
```

```
cab_fare.corr()

cab_fare.head(1).T

import statsmodels.formula.api as smf

anova1 = smf.ols(formula='fare_amount~ passenger_count',data = cab_fare).fit()

print (anova1.f_pvalue)

anova2 = smf.ols(formula='fare_amount~ year',data = cab_fare).fit()

print (anova2.f_pvalue)

anova3 = smf.ols(formula='fare_amount~ month',data = cab_fare).fit()

print (anova3.f_pvalue)

anova4 = smf.ols(formula='fare_amount~ days_of_week',data = cab_fare).fit()

print (anova4.f_pvalue)

anova5 = smf.ols(formula='fare_amount~ hour',data = cab_fare).fit()

print (anova5.f_pvalue)

anova6 = smf.ols(formula='fare_amount~ weekend',data = cab_fare).fit()

print (anova6.f_pvalue)

anova7 = smf.ols(formula='fare_amount~ peak_hour',data = cab_fare).fit()

print (anova7.f_pvalue)

cab_fare.drop(['pickup_datetime','weekend','days_of_week','peak_hour'],axis =1 ,inplace =
True)

from sklearn.model_selection import train_test_split

import statsmodels.api as sm

df = cab_fare.copy()

cat_names = ["passenger_count","year","month","hour"]

for i in cat_names:

    temp = pd.get_dummies(df[i],prefix = i)

    df = df.join(temp)

 df.drop(["passenger_count","year","month","hour"],axis = 1, inplace =True)
```

```
train,test = train_test_split(df,test_size = 0.2,random_state = 0)
print (train.shape)
print (test.shape)
model_lm = sm.OLS(train.iloc[:,0].astype('float'),train.iloc[:,1:57].astype('float')).fit()
model_lm.summary()
from sklearn.metrics import mean_squared_error
predict_lm = model_lm.predict(test.iloc[:,1:57])
np.sqrt(mean_squared_error(test.iloc[:,0],predict_lm))
from sklearn.metrics import r2_score
r2_score(test.iloc[:,0],predict_lm)
train.drop(['pickup_longitude','pickup_latitude'],axis = 1,inplace = True)
test.drop(['pickup_longitude','pickup_latitude'],axis = 1,inplace = True)
model_lm = sm.OLS(train.iloc[:,0].astype('float'),train.iloc[:,1:53].astype('float')).fit()
model_lm.summary()
predict_lm = model_lm.predict(test.iloc[:,1:53])
np.sqrt(mean_squared_error(test.iloc[:,0],predict_lm))
r2_score(test.iloc[:,0],predict_lm)
from sklearn import tree
model_dtr = tree.DecisionTreeRegressor(max_depth = 5).fit(train.iloc[:,1:53],train.iloc[:,0])
predict_dtr = model_dtr.predict(test.iloc[:,1:53])
r2_score(test.iloc[:,0],predict_dtr)
np.sqrt(mean_squared_error(test.iloc[:,0],predict_dtr))
from sklearn.ensemble import RandomForestRegressor
model_rfr = RandomForestRegressor(n_estimators = 200).fit(train.iloc[:,1:53],train.iloc[:,0])
predict_rfr = model_rfr.predict(test.iloc[:,1:53])
r2_score(test.iloc[:,0],predict_rfr)
np.sqrt(mean_squared_error(test.iloc[:,0],predict_rfr))
```

```
test_fare = pd.read_csv('test.csv')

test_fare.head()

test_fare['trip_distance']=test_fare.apply(lambda
row:trip_distance(row['pickup_latitude'],row['dropoff_latitude'],row['pickup_longitude'],row['d
ropoff_longitude']),axis=1)

test_fare['trip_distance'] = test_fare['trip_distance']*1.57

test_fare['pickup_datetime'] = pd.to_datetime(test_fare['pickup_datetime'], errors = 'coerce',
utc=True)

test_fare['year'] = test_fare['pickup_datetime'].dt.year

test_fare['month'] = test_fare['pickup_datetime'].dt.month

test_fare['hour'] = test_fare['pickup_datetime'].dt.hour

cat_names = ["passenger_count","year","month","hour"]

for i in cat_names:

   temp = pd.get_dummies(test_fare[i],prefix = i)

   test_fare = test_fare.join(temp)

test_fare =
test_fare.drop(['pickup_datetime','pickup_longitude','pickup_latitude','passenger_count','year',
'month','hour'],axis = 1)

test_fare.head()

predict_test = model_lm.predict(test_fare)

predict_test.head()
```