PROJECT

CAB FARE PREDICTION

Submitted By:

DIBYA JYOTI MANDAL

TABLE OF CONTENTS

CHAPTER 1: PROBLEM STATEMENT

The project is about a cab company who has done its pilot project and now they are looking to predict the fare for their future transactional cases. As, nowadays there are number of cab companies like Uber, Ola, Meru Cabs etc. And these cab companies deliver services to lakhs of customers daily. Now it becomes really important to manage their data properly to come up with new business ideas to get best results. In this case, earn most revenues.

So, it becomes really important estimate the fare prices accurately.

CHAPTER 2: PROCEDURE

According to industry standards, the process of Data Analyzing mainly includes 6 main steps and this process is abbreviated as CRISP DM Process, which is Cross-Industry Process for Data Mining. And the Six main steps of CRISP DM Methodology for developing a model are:

1. Business understanding
2. Data understanding
3. Data preparation/Data Preprocessing
4. Modeling
5. Evaluation
6. Deployment

And in this project, all the above steps are followed to develop the model.

2.1 Business Understanding

It is important to understand the idea of business behind the data set. The given data set is asking us to predict fare amount. And it really becomes important for us to predict the fare amount accurately. Else, there might be great loss to the revenue of the firm. Thus, we have to concentrate on making the model most efficient.

2.2 Data Understanding

To get the best results, to get the most effective model it is really important to our data very well. Here, the given train data is a CSV file that consists 7 variables and 16067 Observation. A snapshot of the data provided.

fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
4.5	2009-06-15 17:26:21 UTC	-73.844311	40.721319	-73.84161	40.712278	1
16.9	2010-01-05 16:52:16 UTC	-74.016048	40.711303	-73.979268	40.782004	1
5.7	2011-08-18 00:35:00 UTC	-73.982738	40.76127	-73.991242	40.750562	2
7.7	2012-04-21 04:30:42 UTC	-73.98713	40.733143	-73.991567	40.758092	1
5.3	2010-03-09 07:51:00 UTC	-73.968095	40.768008	-73.956655	40.783762	1
12.1	2011-01-06 09:50:45 UTC	-74.000964	40.73163	-73.972892	40.758233	1

Table 2.1 : Train Data

The different variables of the data are:

- **fare_amount** : fare of the given cab ride.
- **pickup_datetime** : timestamp value explaining the time of ride start.
- **pickup_longitude** : a float value explaining longitude location of the ride start.
- **pickup_latitude** : a float value explaining latitude location of the ride start.
- **dropoff_longitude:** a float value explaining longitude location of the ride end**.**
- **dropoff_latitude** : a float value explaining latitude location of the ride end
- **passenger_count** : an integer indicating the number of passengers

Following table explains how the variables are categorized.

Independent Variables
pickup_datetime
pickup_longitude
pickup_latitude
dropoff_longitude
dropoff_latitude
passenger_count

Table 2.2 : Independent Variables

Dependent Variables
Fare_amount

Table 2.3 : Dependent/Target Variable

From the given train data it is understood that, we have to predict fare amount, and other variables will help me achieve that, here pickup_latitude/longitude, dropoff_latitude/longitude this data are signifying the location of pick up and drop off. It is explaining starting point and end point of the ride. So, these variables are crucial for us. Passenger_count is another variable, that explains about how many people or passenger boarded the ride, between the pickup and drop off locations. And pick up date time gives information about the time the passenger is picked up and ride has started. But unlike pick up and drop off locations has start and end details both in given data. The time data has only start details and no time value or time related information of end of ride. So, during pre-processing of data we will drop this variable. As it seems the information of time is incomplete.

Also, there is a separate test data given, in the format of CSV file containing 9514 observations and 6 variables. All of them are the Independent variables. An in these data at the end we have to predict the fare or the target variable. Following is a snap of the test data provided.

pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
2015-01-27 13:08:24 UTC	-73.97332	40.7638054	-73.9814301	40.7438355	1
2015-01-27 13:08:24 UTC	-73.9868622	40.7193832	-73.9988861	40.7392006	1
2011-10-08 11:53:44 UTC	-73.982524	40.75126	-73.979654	40.746139	1

Table 2.4 : Test Data

2.3 Data Preparation

The next step in the CRISP DM Process is, Data preprocessing. It is a data mining process that involves transformation of raw data into a format that helps us execute our model well. As, the data often we get are incomplete, inconsistent and also may contain many errors. Thus, Data preprocessing is a generic method to deal with such issues and get a data format that is easily understood by machine and that helps developing our model in best way. In this project also we have followed data pre-processing methods to rectify errors and issues in our data. And this is done by popular data preprocessing techniques, this are following below.

Note: I have removed the variable "Pick up date Time" as it is a timestamp value and it shows only the start time of pick up time , whereas there is no drop off time, so in this data set it seems, it will have no impact in the target variable, and also it lead to redundancy and model accuracy issues, so I preferred to drop it.

a) Missing Value Analysis

Missing value is availability of incomplete observations in the dataset. This is found because of reasons like, incomplete submission, wrong input, manual error etc.These Missing values affect the accuracy of model. So, it becomes important to check missing values in our given data.

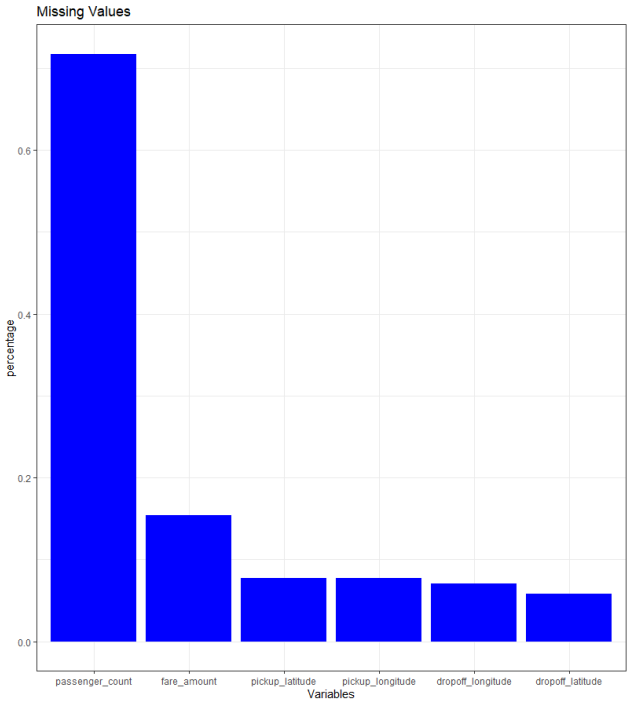
Missing Value Analysis in Given Data:

In the given dataset it is found that there are lot of values which are missing. It is found in the following types:

- 1. Blank spaces : Which are converted to NA and NaN in R and Python respectively for further operations
- 2. Zero Values : This is also converted to NA and Nan in R and python respectively prior further operations
- 3. Repeating Values : there are lots of repeating values in pickup_longitude, pickup_latitude, dropoff_longitude and dropoff_latitude. This will hamper our model, so such data is also removed to improve the performance.

Following the standards of percentage of missing values we now have to decide to accept a variable or drop it for further operations. Industry standards ask to follow following standards:

1. Missing value percentage < 30% : Accept the variable
2. Missing value percentage > 30 % : Drop the variable



Graph 2.1 : Missing Values

It is found from the above graph plot that the there is no variable exceeding the 30% range so we not need to exclude any of our variable.

ii) Impute the missing value:

After the identification of the missing values the next step is to impute the missing values. And this imputation is normally done by following methods.

1. Central Tendencies: by the help of Mean, Median or Mode
2. Distance based or Data mining method like KNN imputation
3. Prediction Based: It is based on Predictive Machine Learning Algorithm

To use the best method it is necessary for us to check, which method predicts values close to the original data. And this done by taking a subset of data, taking an example variable and noting down its original value and the replacing that value with NA and then applying available methods. And noting down every value from the above methods for the example variable we have taken, now we chose the method which gives most close value.

In this project, KNN imputation worked the best. So, I am using KNN method to impute missing Values.

	Variable	Missing Value count	Missing value Percentage
1	fare_amount	0	0
2	pickup_longitude	0	0
3	pickup_latitude	0	0
4	dropoff_longitude	0	0
5	dropoff_latitude	0	0
6	passenger_count	0	0

Table 2.5 : Missing values after Imputation with KNN

b) **Outlier Analysis**

Outlier is an abnormal observation that stands or deviates away from other observations. These happens because of manual error, poor quality of data and it is correct but exceptional data. But, It can cause an error in predicting the target variables. So we have to check for outliers in our data set and also remove or replace the outliers wherever required

Outliers in this project.

In this dataset, I have found some irregular data, those are considered as outliers. These are explained below.

i) Fare_Amount :

I have always seen fare of a cab ride as positive, I have never seen any cab driver, giving me money to take a ride in his cab. But in this dataset, there are many instances where fare amount is negative. Given below are such instances:

fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
-2.9	2010-03-09 23:37:10 UTC	-73.7895	40.6435	-73.7887	40.64195	1
-2.5	2015-03-22 05:14:27 UTC	-74	40.72063	-73.9998	40.72054	1
-3	2013-08-30 08:57:10 UTC	-73.9951	40.74076	-73.9959	40.74136	4

Table 2.6 : Fare Outliers

ii) Passenger_count:

I have always found a cab with 4 seats to maximum of 8 seats. But in this dataset I have found passenger count more than this, and in some cases a large number of values. This seems irregular data, or a manual error. Thus, these are outliers and needs to be removed. Few instances are following.

Fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
4.9	2010-07-12 09:44:33 UTC	-73.9832	40.73466	-73.9913	40.73892	456
6.1	2011-01-18 23:48:00 UTC	-74.0066	40.73893	-74.0108	40.71791	5334
8.5	2013-06-18 10:27:05 UTC	-73.9921	40.7642	-73.973	40.7627	535
8.1	2009-08-21 19:35:05 UTC	-73.9609	40.76156	-73.9763	40.74836	354

Table 2.7 : Passenger_count Outliers

ii) Location points:

When I checked the data it is found that most of the longitude points are within the 70 degree and most of the latitude points are within the 40 degree. This symbolizes all the data belongs to a specific location and a specific range. But I also found some data which consists location points too far from the average location point's range of 70 Degree Longitude and 40 Degree latitude. It seems these far point locations are irregular data. And I consider this as outlier. I have collected the maximum and minimum values of location point as a reference to identify the outliers.

Variable	Minimum Value	Maximum Value
pickup_longitude	-74.43823	40.76613
pickup_latitude	-74.00689	401.08333
dropoff_longitude	-74.22705	40.80244
dropoff_latitude	-74.00638	41.36614

Table 2.8: Range of Variables

Following are the instances, where the values exceeds far from average location points. And I consider this as Outliers.

Fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
15	40.72913	-74.0069	40.76337	-73.9616	1
52	40.73688	-74.0062	40.73689	-74.0064	6
15.5	40.76442	-73.9929	40.80244	-73.9507	1
6.5	40.74826	-73.9918	40.74037	-73.979	1
3.3	-73.9472	401.0833	-73.9514	40.77893	1

Table 2.9: Location Outliers

All this outliers mentioned above happened because of manual error, or interchange of data, or may be correct data but exceptional. But all these outliers can hamper our data model. So there is a requirement to eliminate or replace such outliers. And impute with proper methods to get better accuracy of the model. In this project, I used mode method to impute the outliers in passenger count and mean for location Points and fare amount.

c) **Feature Selection**

Sometimes it happens that, all the variables in our data may not be accurate enough to predict the target variable, in such cases we need to analyze our data, understand our data and select the dataset variables that can be most useful for our model. In such cases we follow feature selection. Feature selection helps by reducing time for computation of model and also reduces the complexity of the model.

After understanding the data, preprocessing and selecting specific features, there is a process to engineer new variables if required to improve the accuracy of the model.

In this project the data contains only the pick up and drop points in longitude and latitude. The fare_amount will mailnly depend on the distance covered between these two points. Thus, we have to create a new variable prior further processing the data. And in this project the variable I have created is Distance variable (dist), which is a numeric value and explains the distance covered between the pick up and drop of points. After researching I found a formula called The haversine formula, that determines the distance between two points on a sphere based on their given longitudes and latitudes. These formula calculates the shortest distance between two points in a sphere.

The function of haversine function is described, which helped me to engineer our new variable, Distance.

Used in Python :

```
# haversine function

def haversine(lat1, lon1, lat2, lon2, to_radians=True, earth_radius=6371):

    if to_radians:
        lat1, lon1, lat2, lon2 = np.radians([lat1, lon1, lat2, lon2])

    a = np.sin((lat2-lat1)/2.0)**2 + \
        np.cos(lat1) * np.cos(lat2) * np.sin((lon2-lon1)/2.0)**2

    return earth_radius * 2 * np.arcsin(np.sqrt(a))
```

Used In R :

```
#create new variable

library(geosphere)

train$dist= distHaversine(cbind(train$pickup_longitude, train$pickup_latitude), cbind(train$dropoff_longitude,train$dropoff_latitude))

#the output is in metres, Change it to kms

train$dist=as.numeric(train$dist)/1000
```

After excecuting the haversine function in our project, I got new variable distance and some instances of data are mentioned below.

Fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	Dist
4.5	-73.98172	40.72132	-73.98011	40.71228	1	1.0156499
16.9	-74.01605	40.7113	-73.97927	40.782	1	8.4595997
5.7	-73.98274	40.76127	-73.99124	40.75056	2	1.3910818
7.7	-73.98713	40.73314	-73.99157	40.75809	1	2.8024061
5.3	-73.9681	40.76801	-73.95665	40.78376	1	2.0013963

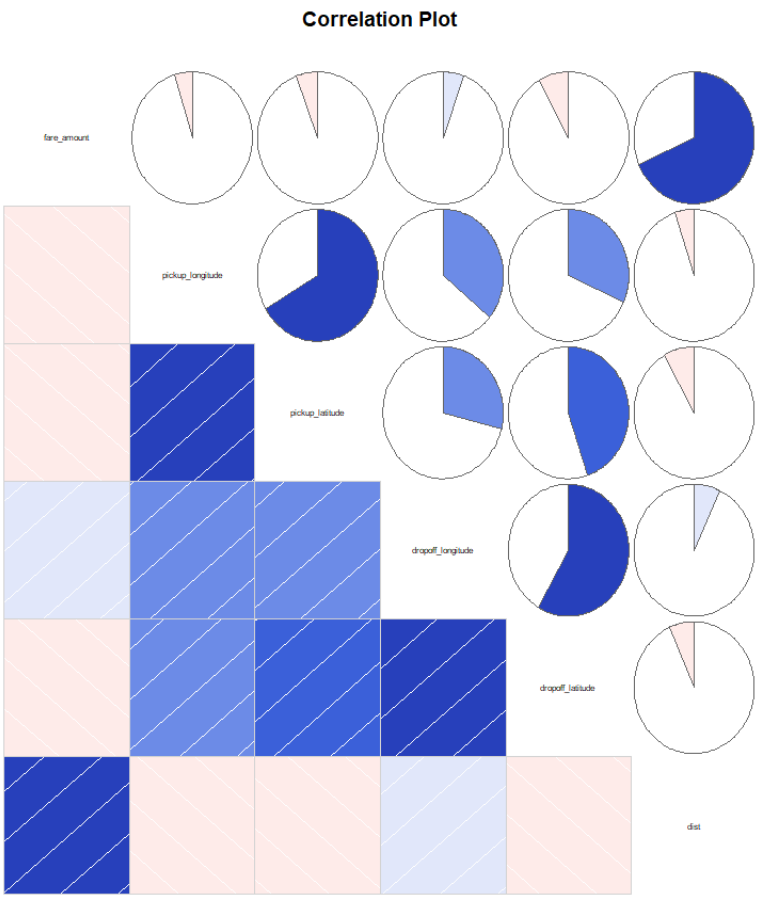
Table 2.10: Engineering new Variable Distance

a. Correlation Analysis:

In some cases it is asked that models require independent variables free from collinearity issues. This can be checked by correlation analysis for the categorical variables and continuous variables. Correlation analysis is a process that is defined to identify the level of relation between two variables.

In this project, our Predictor variable is continuous, so we will plot a correlation table that will predict the correlation strength between independent variables and the 'fare_amount' variable

Correlation Plot:



Graph 2.2 : Correlation Plot

From the above plot it is found that most of the variables are highly correlated with each other, like fare amount is highly correlated with distance variable. All the dark blue charts represents that variables are highly correlated. And as there is no dark red charts, which represents negative correlation, it can be summarized that our dataset has strong or highly positive correlation between the variables.

2.4 : Model Development

After all the above processes the next step is developing the model based on our prepared data.
In this project we got our target variable as “fare_amount”. The model has to predict a numeric value. Thus, it is identified that this is a Regression problem statement. And to develop a regression model, the various models that can be used are Decision trees, Random Forest, Linear Regression and KNN imputation.

i) Decision Tree

Decision Tree is a supervised learning predictive model that uses a set of binary rules to calculate the target value/dependent variable.
Decision trees are divided into three main parts this are :

- **Root Node** : performs the first split
- **Terminal Nodes** : that predict the outcome, these are also called leaf nodes
- **Branches** : arrows connecting nodes, showing the flow from root to other leaves.

In this project Decision tree is applied in both R and Python, details are described following.

a) Decision Tree in R

The Decision tree Method is used R with all the input variables except the pickup_datetime variable, which we have dropped in initial stages of data preparation.

```
> fit
n= 11820

node), split, n, deviance, yval
      * denotes terminal node

 1) root 11820 313310.300  9.535630
   2) dist< 2.726869 7767 104927.500  7.358837
     4) dist< 1.438237 3843  38609.150  6.036391 *
     5) dist>=1.438237 3924  53015.290  8.653986 *
   3) dist>=2.726869 4053 101051.000 13.707150
     6) dist< 4.493959 2507  44707.010 12.037380 *
     7) dist>=4.493959 1546  38019.590 16.414840
      14) dist< 7.015429 1272  28612.010 15.662300 *
      15) dist>=7.015429 274   5343.071 19.908390 *
> |
```

Plot : Decisiont Tree Fit

The above plot shows the rules of splitting of trees. The main root splits into 2 nodes having dist< 2.726869 and dist>=2.726869 as its conditions. Nodes further split, The line with * shows that it is the terminal node. These rules are then applied on the test data to predict values.

b) Decision Tree in Python

```
fit_DT
```

```
DecisionTreeRegressor(criterion='mse', max_depth=2, max_features=None,
           max_leaf_nodes=None, min_impurity_decrease=0.0,
           min_impurity_split=None, min_samples_leaf=1,
           min_samples_split=2, min_weight_fraction_leaf=0.0,
           presort=False, random_state=None, splitter='best')
```

Plot: Decision Tree Fit in Python

The above fit plot shows the criteria that is used in developing the decision tree in Python. To develop the model in python, I haven't provided any input argument of my choice, except the depth as 2, to visualize the tree better. All other arguments in the model are default, in developing the model. After this the fit_DT is used to predict in test data and the error rate and accuracy is calculated.

ii) Random Forest

The next model to be followed in this project is Random forest. It is a process where the machine follows an ensemble learning method for classification and regression that operates by developing a number of decision trees at training time and giving output as the class that is the mode of the classes of all the individual decision trees.

In this project Random Forest is applied in both R and Python, details are described following.

a) Random Forest in R

In a RandomForest model the importance contributed by individual variables can be seen using importance function, it is mentioned below.

```
> importance(RF_model, type = 1)
                    %IncMSE
pickup_longitude  25.2795121
pickup_latitude   19.2820283
dropoff_longitude 26.4679583
dropoff_latitude  20.9344432
passenger_count   -0.1305208
dist              63.6148850
```

Plot : importance of variables

The above RF Model shows that the variable contributing most for predicting the fare_amount is distance and the least important is passenger_count

b) Random Forest in Python

```
RF_model
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
           oob_score=False, random_state=None, verbose=0, warm_start=False)
```

Plot: Random Forest in Python

Like the Decision tree above are all the criteria values that are used to develop the Random Forest model in python.

iii) Linear Regression

The next method in the process is Linear regression. It is used to predict the value of variable *Y* based on one or more input predictor variables *X*. The goal of this method is to establish a linear relationship between the predictor variables and the response variable. Such that, we can use this formula to estimate the value of the response *Y*, when only the predictors (*X- Values*) are known.

In this project Linear Regression is applied in both R and Python, details are described following.

a) Linear regression in R

After running the model the details I got are as follows.

```
> summary(lm_model)

Call:
lm(formula = fare_amount ~ ., data = train1)

Residuals:
     Min       1Q   Median       3Q      Max
-19.6146  -1.9157  -0.9133   0.6722  23.7607

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       856.85299  326.17166   2.627 0.008625 **
pickup_longitude  -10.88446    2.95875  -3.679 0.000235 ***
pickup_latitude     8.28527    2.28406   3.627 0.000287 ***
dropoff_longitude  17.65568    2.58412   6.832 8.76e-12 ***
dropoff_latitude  -16.90918    2.03420  -8.312  < 2e-16 ***
passenger_count2    0.09186    0.10009   0.918 0.358751
passenger_count3    0.21761    0.17059   1.276 0.202104
passenger_count4   -0.09552    0.24794  -0.385 0.700062
passenger_count5   -0.09241    0.14128  -0.654 0.513088
passenger_count6    0.94765    0.24626   3.848 0.000120 ***
dist                2.01194    0.02013  99.923  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.739 on 11809 degrees of freedom
Multiple R-squared:  0.4732,    Adjusted R-squared:  0.4728
F-statistic:  1061 on 10 and 11809 DF,  p-value: < 2.2e-16
```

Plot: Summary Linear Regression Model

The above plot shows how the target variable fare_amount varies with change in each individual variable. Like, if there is one unit change in the pickup_longitude, the fare_amount decreases by 10 units (Approx), keeping all other variables constant.

The P-Value shows which values are significant in predicting the target variable. Here, we reject null hypothesis which is less then 0.05 and declare that the variable is significant for the model. F-Statistic explains about the quality of the model, and describes the relationship among predictor and target variables. The R squared and adjusted R squared values shows how much variance of the output variable is explained by the independent or input variables. Here the adjusted r square value is 47.28%, which indicated that only 47.28% of the variance of fare_amount is explained by the input variables. This explains the model is not upto the mark.

b) Linear Regression in Python

After this the model is developed following details are found.

Dep. Variable:	fare_amount	R-squared:	0.451
Model:	OLS	Adj. R-squared:	0.450
Method:	Least Squares	F-statistic:	971.0
Date:	Thu, 11 Jul 2019	Prob (F-statistic):	0.00
Time:	15:20:51	Log-Likelihood:	-32642.
No. Observations:	11836	AIC:	6.531e+04
Df Residuals:	11825	BIC:	6.539e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
pickup_longitude	-8.1153	3.056	-2.655	0.008	-14.106	-2.124
pickup_latitude	6.9690	2.345	2.972	0.003	2.372	11.566
dropoff_longitude	14.7490	2.685	5.492	0.000	9.485	20.013
dropoff_latitude	-16.4963	2.107	-7.828	0.000	-20.627	-12.366
dist	1.9859	0.021	95.766	0.000	1.945	2.027
passenger_count_1	883.5936	341.928	2.584	0.010	213.358	1553.830
passenger_count_2	883.7456	341.930	2.585	0.010	213.506	1553.985
passenger_count_3	883.8646	341.934	2.585	0.010	213.618	1554.111
passenger_count_4	883.8406	341.933	2.585	0.010	213.597	1554.085
passenger_count_5	883.5124	341.928	2.584	0.010	213.277	1553.748
passenger_count_6	884.8060	341.929	2.588	0.010	214.569	1555.043

Omnibus:	5963.342	Durbin-Watson:	1.993
Prob(Omnibus):	0.000	Jarque-Bera (JB):	45147.146
Skew:	2.313	Prob(JB):	0.00
Kurtosis:	11.375	Cond. No.	2.85e+06

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.85e+06. This might indicate that there are
strong multicollinearity or other numerical problems.

Here, F-Statistic explains about the quality of the model. AIC is Akkaine information criterion, if we have multiple models with same accuracy then we need to refer this to choose the best model. The table three values containing Omnibus and JB test are mostly required for timestamp data sets. Here, as we are not using any time values in our project we can ignore this table 3. T-statistic explain how much statistically significant the coefficient is. It is also used to calculate the P –Value. And if P-Value is less then 0.05 we reject null hypothesis and say that the variable is significant. Here, all the variables are less then 0.05 and are significant. The R squared and adjusted R squared values show how much variance of the output variable is explained by the independent or input variables. Here the adjusted r square value is only 45.01%, which explains that only 45% of the variance of fare_amount is explained by the input variables. This shows that the model is performing very poor. This may be because the relationship between the independent and dependent variable might be nonlinear.

iv) KNN imputation model

The next process to be followed is The KNN model. It finds the nearest neighbors and tries to predict target value. The method goes as, for the value of new point to be assigned, this value is assigned on the basis of how closely this point resembles the other points in the training set. The process of implementing KNN methodology is little easy in compare other models. After implementing the KNN model, the KNN function is imported from respective libraries in Python and R, package “Class” in R and “Scikit Learn” library in R. After that the model is Run, and the prediction fit is used to predict in test data. Finally the error and accuracy is calculated.

Model Summary:

Above mentioned Decision Tree, Random Forest, Linear Regression and KNN Method are the various models that can be developed for the given data. At first place, The Data is divided into train and test. Then the models are developed on the train data. After that the model is fit into it to test data to predict the target variable. After predicting the target variable in test data, the actual and predicted values of target variable are compare to get the error and accuracy. And looking over the error and accuracy rates, the best model for the data is identified and it is kept for future usage.

CHAPTER 3: EVALUATION OF THE MODEL

So, now we have developed few models for predicting the target variable, now the next step is to identify which one to choose for deployment. To decide these according to industry standards, we follow several criteria. Few among this are, calculating the error rate, and the accuracy. MAE and MAPE is used in our project. RMSE is not used because we are not working with Timestamp value.

3.1 Mean Absolute Error (MAE)

MAE or Mean Absolute Error, it is one of the error measures that is used to calculate the predictive performance of the model. In this project we will apply this measure to our models

In R, Define MAPE using function

```
MAPE = function(y, yhat){
mean(abs((y - yhat)/y)*100)
}
```

a) In R :

Method	Mape Error(in Percentage)
Decision Tree	27.75005
Random Forest	22.50844
Linear Regression	26.12016
KNN Imputation	33.7978

Table 3.1: Mape in R

b) In Python :

Method	Mape Error(in Percentage)
Decision Tree	28.7746632
Random Forest	25.1316683
Linear Regression	27.4313709
KNN Imputation	34.21410560

Table 3.2: Mape in Python

3.2 Accuracy

The second matrix to identify or compare for better model is Accuracy. It is the ratio of number of correct predictions to the total number of predictions made.

Accuracy= number of correct predictions / Total predictions made

It can also be calculated from MAE as
Accuracy = 1- MAPE

Method	Accuracy (in Percentage)
Decision Tree	71.23
Random Forest	74.87
Linear Regression	73.88
KNN Imputation	66.21

Table 3.3: Accuracy in R Models

Method	Accuracy (in Percentage)
Decision Tree	72.09
Random Forest	76.20
Linear Regression	72.56
KNN Imputation	65.80

Table 3.4: Accuracy in Python Models

3.3 Model Selection

After comparison of the error matrix, the next step we come to is Selection of the most effective model. From the values of Error and accuracy, it is found that all the models perform close to each other. In this case any model can best used for further processes, but Random forest gives better results compared to all other methods. So I will prefer Random Forest Model to be used for further processes.

CHAPTER 4: REFERENCES

Websites:

- www.edwisor.com : Videos from Mentor :
- https://rpubs.com/ : Coding Doubts :
- https://stackoverflow.com/questions/51488949/use-haversine-package-to-compare-all-distances-possibilities-of-a-csv-list-of-lo : Haversine Doubt
- https://www.r-bloggers.com/ : Miscellaneous Doubts in R
- https://gist.github.com/rochacbruno/2883505 : Calculate Haversine in python

Videos Channels:

- https://www.youtube.com/watch?v=7YfyIhhmwq4 : Distance development in Python
- https://www.youtube.com/user/marinstatlectures : R Coding
- https://www.youtube.com/watch?v=Uct_EbThV1E&list=PLZ7s-Z1aAtmIbaEj_PtUqkqdmI1k7libK&index=2&t=690s : Python Coding

THANK YOU