Project Name – Cab Fare Prediction

Submitted By:

PRAKHAR ACHARYA

Content:

SI.No.	Topic	Page No.
1	Problem Statement	2
2	Procedure	3
3	Business Understanding	4
4	Data Understanding	4
5	Data Preparation	6
6	Missing Value Analysis	6
7	Outlier Analysis	8
8	Feature Selection	9
9	Correlation Analysis	11
10	Model Development	14
11	Decision Tree	14
12	Random Forest	15
13	Linear Regression	17
14	KNN Imputation	19
15	Evaluation of the Model	20
16	Mean Absolute Error(MAE)	20
17	Accuracy	21
18	Model Selection	22
19	Reference	23

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.

It becomes really important to predict the fare prices accurately as nowadays there are many cab rental companies like Ola, Uber etc , which have a great customer base. It is very important to manage their data properly and come with ideas where they can earn more revenues.

Procedure

According to the Cross-Industry Process of Data Mining there are mainly 6 steps for Data Analyzing and in this project, will follow all the steps to develop the model.

Following are the list of steps:

1. Business Understanding
2. Data Understanding
3. Data Pre Processing
4. Modeling
5. Evaluation
6. Deployment

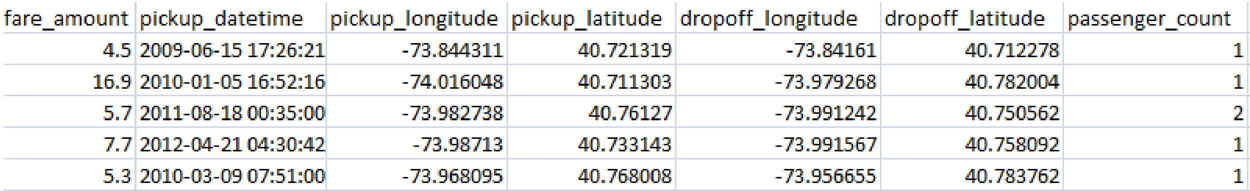
Business Understanding

In the given data set we are asked t predict the fare amount and it is really important to predict it accurately. If not, it can result in revenue losses for the firm. Hence, we have make model that is most efficient.

Data Understanding

In order to get good result we need to understand our data very well.

We have been given train data in csv format which contains 16067 observations and 7 variables.



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

Train Data

Following are the list of variables given in the data:

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 cabride.

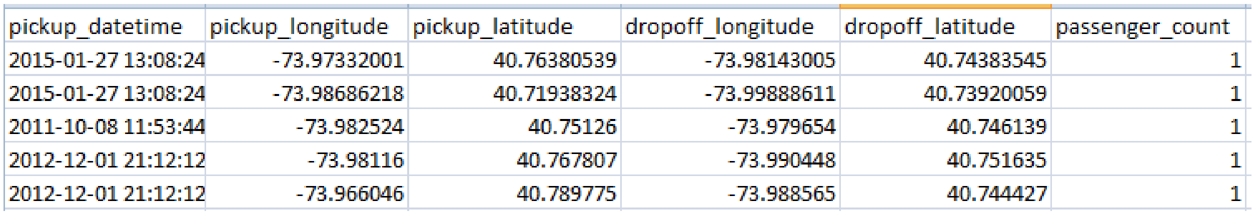
Independent Variables
pickup_datetime
pickup_longitude
pickup_latitude
dropoff_longitude
dropoff_latitude
passenger_count

Dependent Variable
fare_amount

Passenger_count variable explains the number of passengers that boarded the cab whereas pickup_longitude/latitude and dropoff_longitude/latitude gives their location. From the data it is clear that I have to predict the fare_amount with the help of other variables.

Pickup_Datetime gives the time when the passenger was picked and the ride started but it does nt give the end time , it looks incomplete whereas pickup_longitude/latitude and dropoff_longitude/latitude variables gives both the pick up and the drop location. Hence, in our pre processing method we will drop the pickup_datetime.

Along with the train data, I have also been given with the test data which contains 9514 observations and 6 variables in csv format. All the variables are independent and we have to predict the fare_amount from the given test data.



pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
2015-01-27 13:08:24	-73.97332001	40.76380539	-73.98143005	40.74383545	1
2015-01-27 13:08:24	-73.98686218	40.71938324	-73.99888611	40.73920059	1
2011-10-08 11:53:44	-73.982524	40.75126	-73.979654	40.746139	1
2012-12-01 21:12:12	-73.98116	40.767807	-73.990448	40.751635	1
2012-12-01 21:12:12	-73.966046	40.789775	-73.988565	40.744427	1

Test Data

Data Preparation:

The next step in the CRISP-DM process is Data Pre Processing. Data Preprocessing is that step in which the data gets transformed, or Encoded, to bring it to such a state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm. 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.

Pick up Data Time variable is removed from as it shows only the starting time and not the end time, which shows incomplete, so in the data , it happens that the variable won't have impact in the target variable and also it lead to redundancy and model accuracy issues. Hence, I dropped the variable.

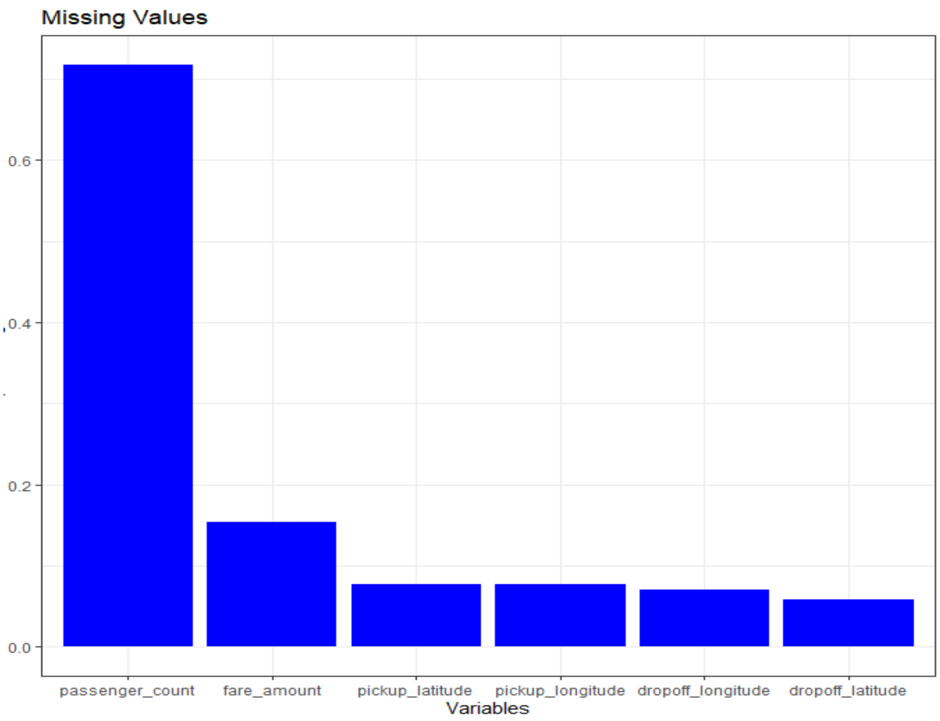
a. **Missing Value Analysis**:

Missing value is availability of incomplete observations in the data set. It mainly occurs because of incomplete submission, wrong input, manual error etc., which mainly affects the accuracy of the model. Hence it is important to check if missing values are present in the dataset.

Missing values that are present in the dataset
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

According to the standard of missing values percentage, we need to decide whether to accept or drop the variable.

a. If missing value percentage>30% then we can accept the variable
b. And, if the missing value percentage<30% then we need to drop the variable.



From the above graph plot we can observe that there is no variable exceeding 30% range, so we don’t need to exclude any variable.

Imputing the missing values:

After we have identified the missing values, we need to impute the missing values. Following are the methods we will follow to impute the missing values:

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

We will choose the imputation method in which the predicted value will be closest to the actual values. We will implement this by taking a subset of data, noting the value of a variable, replacing it with NA and implementing different missing value imputing methods. We will then finalize the method which will be closest to the original value of the variable we chose.

After following the above methods, KNN imputation was the best method as the value predicted is closest to the actual value.

	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

Missing Values after KNN Imputation

b.Outlier Analysis:

An outlier is an element of a data set that distinctly stands out from the rest of the data. This happens mainly 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.

In the dataset given, there are some irregular data. I have mentioned the observations below:

i. Fare_amount

On observing we see that some of the fare values are negative , which is not possible as it will be really odd that for a ride instead of the passenger giving money to driver , the driver gives money to the passenger.

ii. Passenger_count

In a cab we notice seats are with 4 or maximum 8 seats. But, in some of the observations we can observe that some of them have passenger count more than 8, which is not possible.Hence, we will remove these outliers from the dataset.

iii. Location Points

On observing I saw 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 consist 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	Maximum Value	Minimum Value
Pickup_longitude	-74.43823	40.76613
Pickup_latitude	-74.00689	40.08333
Dropoff_longitude	-74.22705	40.80244
Dropoff_latitude	-74.00638	41.36614

All the outliers mentioned 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:

Feature Selection is the process where we automatically or manually select those features which contribute most to our prediction variable or output in which you are interested in.

Having irrelevant features in our data can decrease the accuracy of the models and make your model learn based on irrelevant features.

After we have understood 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 mainly 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 pickup and drop of points.

After researching and searching on different sources, I found the formula called The **Haversine** formula, that determines the distance between two points on a sphere based on their given longitudes and latitudes. This formula calculates the shortest distance between two points in a sphere.

Used in Python:

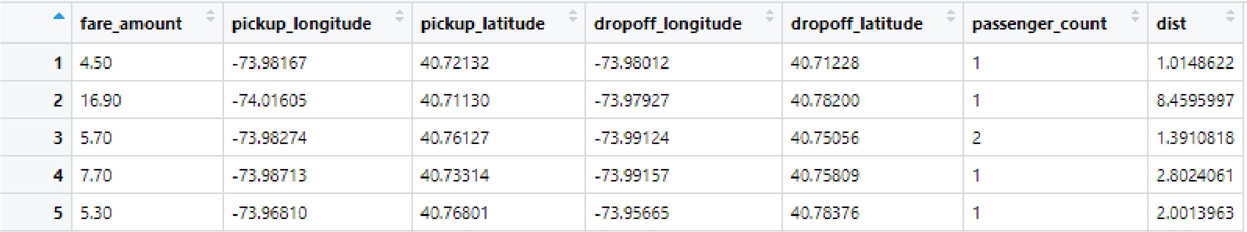
```
 # haversine function

 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))

 #since the output is in metres, we will change it to kms

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

After executing 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
1	4.50	-73.98167	40.72132	-73.98012	40.71228	1	1.0148622
2	16.90	-74.01605	40.71130	-73.97927	40.78200	1	8.4595997
3	5.70	-73.98274	40.76127	-73.99124	40.75056	2	1.3910818
4	7.70	-73.98713	40.73314	-73.99157	40.75809	1	2.8024061
5	5.30	-73.96810	40.76801	-73.95665	40.78376	1	2.0013963

New Variable Distance

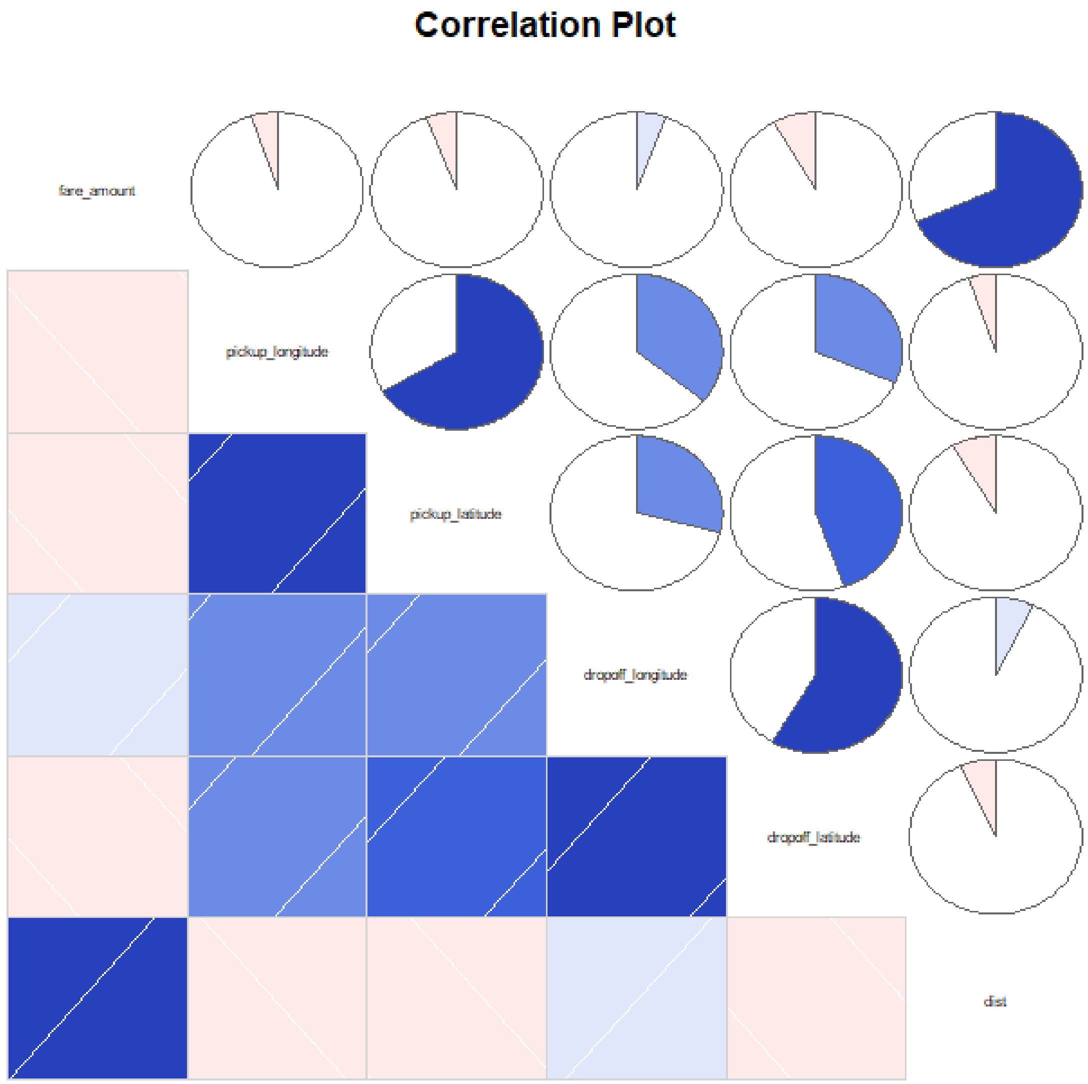
d. Correlation Analysis:

Correlation analysis is a statistical method used to evaluate the strength of relationship between two quantitative variables.

A high correlation means that two or more variables have a strong relationship with each other, while a weak correlation means that the variables are hardly related.

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

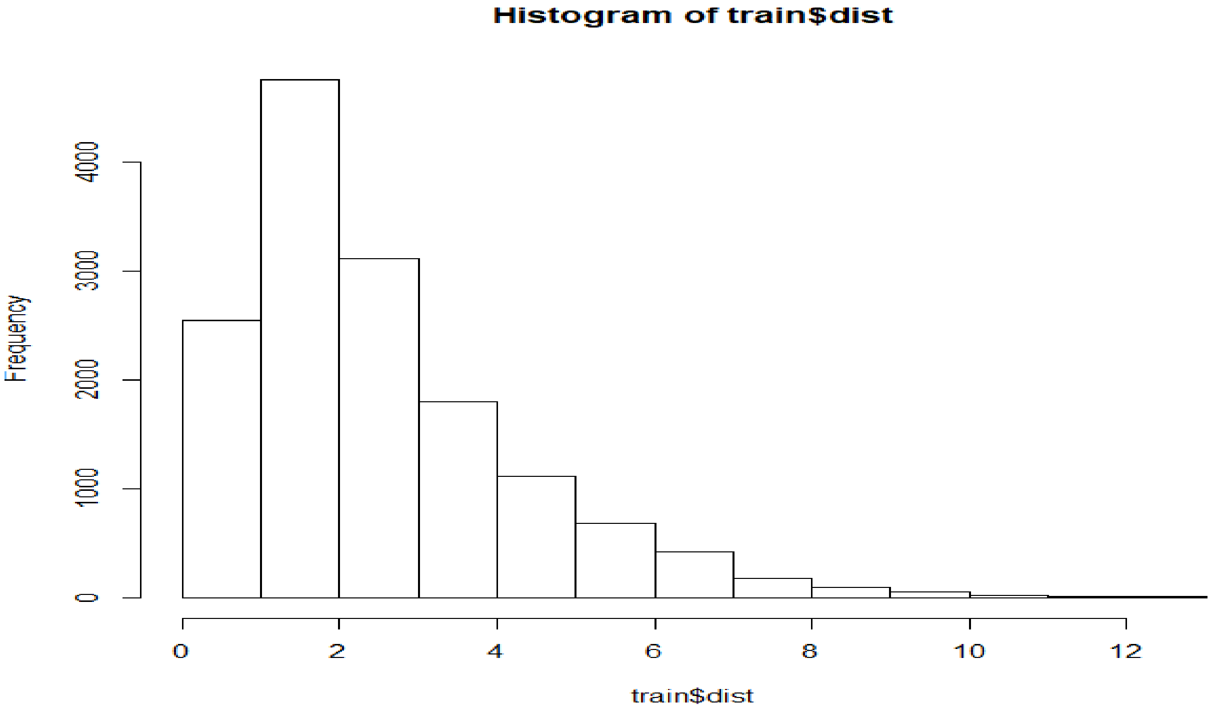


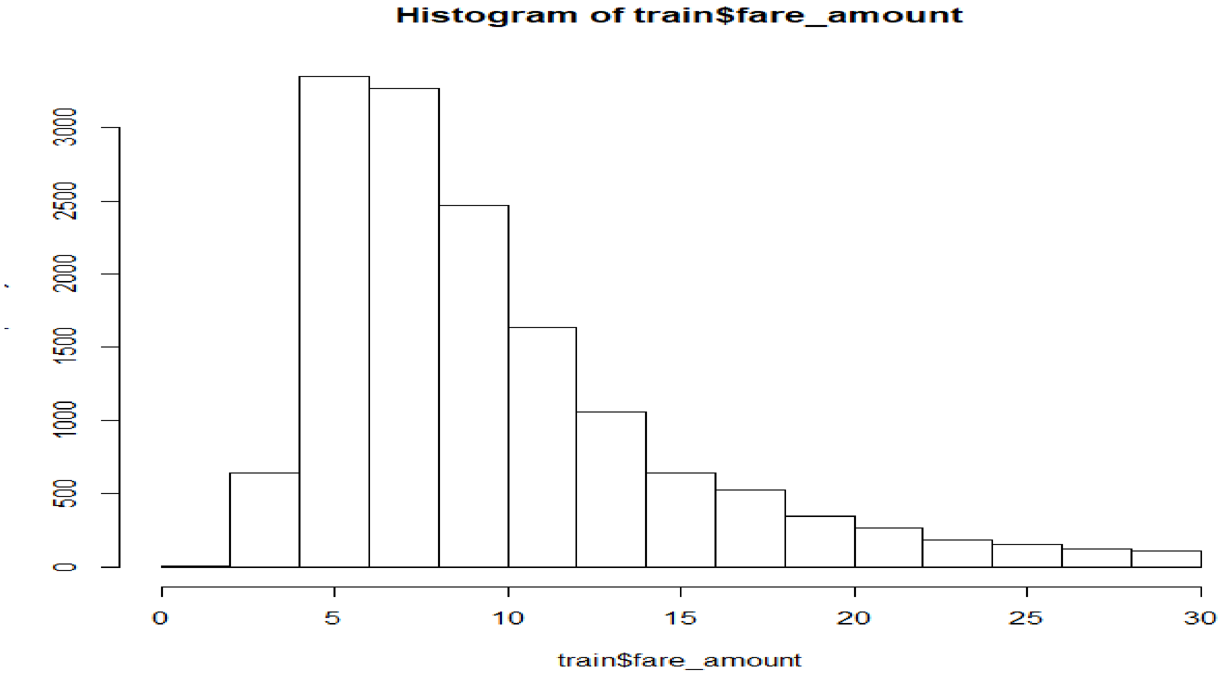
As we can observe from the above plot, that most of the variables are highly correlated with each other, like fare amount is highly correlated with distance variable.

All the dark blue charts represents that variables are highly correlated.

We can also observe that 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.

We can also observe the individual variables through below plots.





Model Development

As we have completed the above process, now we will proceed with the Model Development.

In this project we have 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.

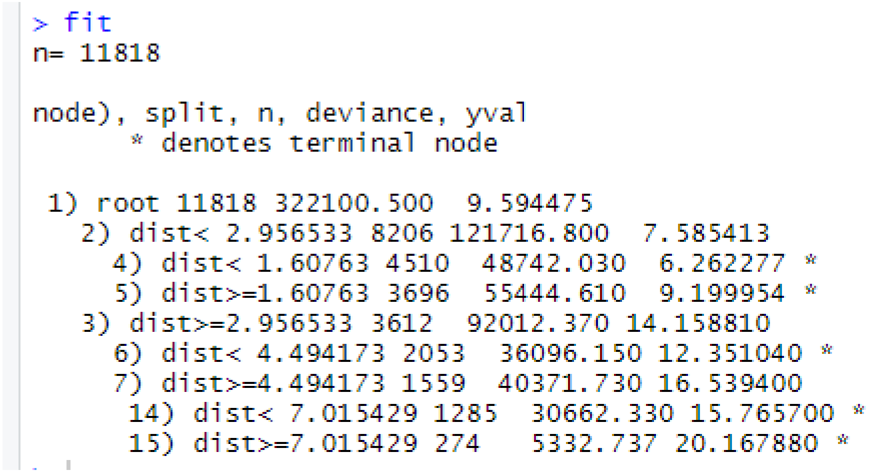
And to develop a regression model, the various models that can be used are **Decision trees, Random Forest, Linear Regression and KNN imputation**.

i. Decision Tree:

Decision tree is one of the predictive modeling approaches used in statistics , data mining and machine learning .Decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions.

Decision Tree in R

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



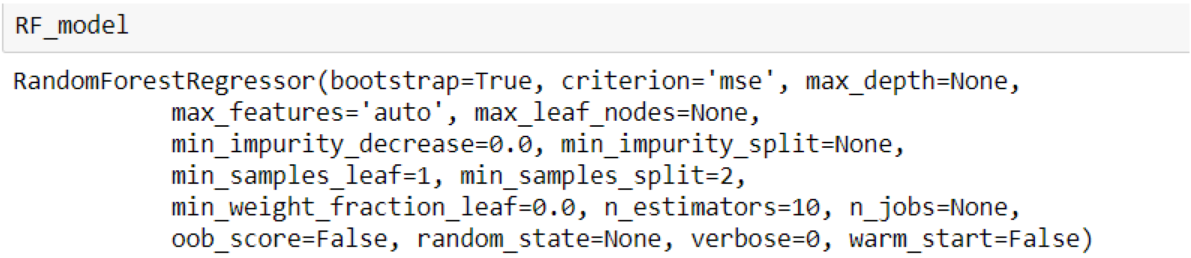
```
> fit
n= 11818

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

 1) root 11818 322100.500  9.594475
   2) dist< 2.956533 8206 121716.800  7.585413
     4) dist< 1.60763 4510  48742.030  6.262277 *
     5) dist>=1.60763 3696  55444.610  9.199954 *
   3) dist>=2.956533 3612  92012.370 14.158810
     6) dist< 4.494173 2053  36096.150 12.351040 *
     7) dist>=4.494173 1559  40371.730 16.539400
      14) dist< 7.015429 1285  30662.330 15.765700 *
      15) dist>=7.015429 274   5332.737 20.167880 *
```

The above plot shows the rules of splitting of trees. The main root splits into 2 nodes having

dist< 2.956533 and dist>=2.956533 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

Decision Tree 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)
```

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

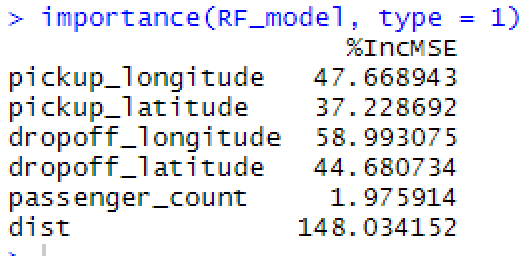
The next model that we will follow in this project is Random forest.

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.

Random Forest in R:

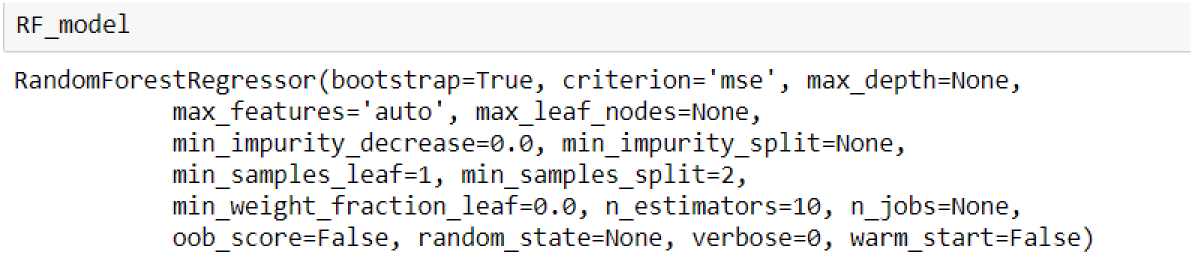
In the RandomForest model the importance contributed by individual variables can be seen using importance function, it is mentioned below.



```
> importance(RF_model, type = 1)
                     %IncMSE
pickup_longitude   47.668943
pickup_latitude    37.228692
dropoff_longitude  58.993075
dropoff_latitude   44.680734
passenger_count     1.975914
dist              148.034152
```

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

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)
```

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

iii. Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning.

It performs a regression task. Regression models a target prediction value based on independent variables.

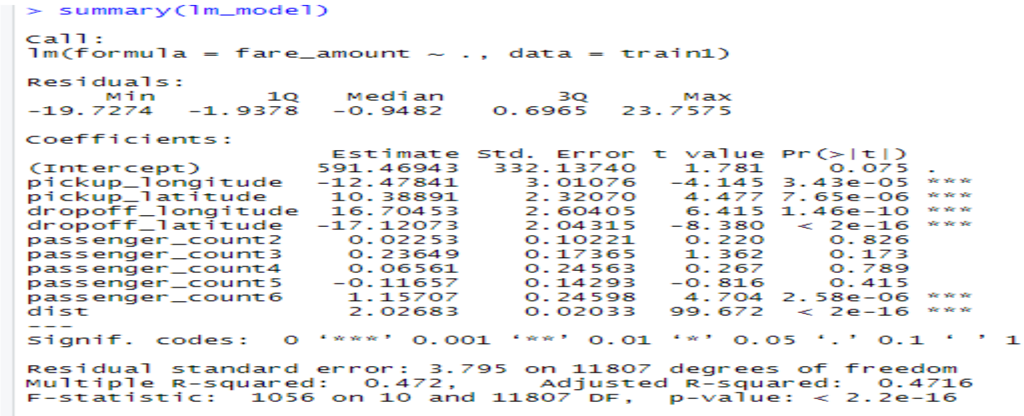
So, this regression technique finds out a linear relationship between :

x (input) and y(output).

In this project Linear Regression is applied in both R and Python,

Linear Regression in R:

On running the model the details that I got are mentioned below:



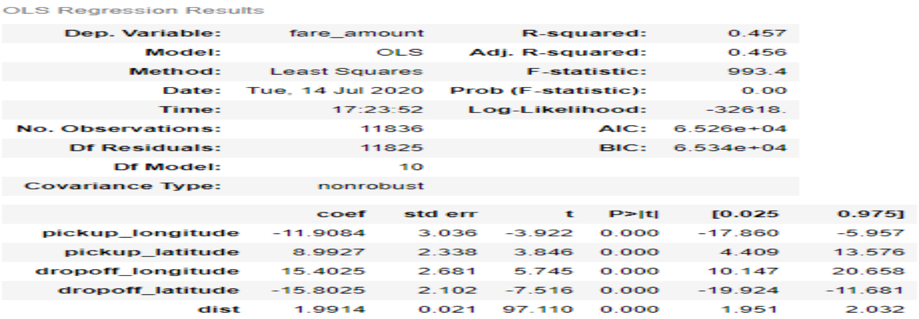
```
> summary(lm_model)

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

Residuals:
     Min       1Q   Median       3Q      Max 
-19.7274  -1.9378  -0.9482   0.6965  23.7575 

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)    
(Intercept)         591.46943  332.13740   1.781    0.075 .  
pickup_longitude    -12.47841    3.01076  -4.145 3.43e-05 ***
pickup_latitude      10.38891    2.32070   4.477 7.65e-06 ***
dropoff_longitude    16.70453    2.60405   6.415 1.46e-10 ***
dropoff_latitude    -17.12073    2.04315  -8.380  < 2e-16 ***
passenger_count2      0.02253    0.10221   0.220    0.826    
passenger_count3      0.23649    0.17365   1.362    0.173    
passenger_count4      0.06561    0.24563   0.267    0.789    
passenger_count5     -0.11657    0.14293  -0.816    0.415    
passenger_count6      1.15707    0.24598   4.704 2.58e-06 ***
dist                  2.02683    0.02033  99.672  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.795 on 11807 degrees of freedom
Multiple R-squared:  0.472,	Adjusted R-squared:  0.4716 
F-statistic:  1056 on 10 and 11807 DF,  p-value: < 2.2e-16
```

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 12 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 47.16%, which indicated that only 47.16% of the variance of fare_amount is explained by the input variables. This explains the model is not on the mark.

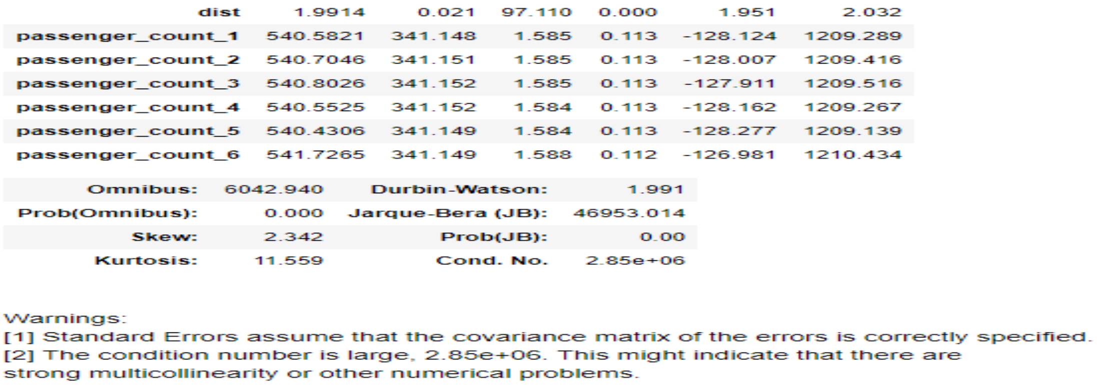
Linear Regression in Python:



OLS Regression Results

Dep. Variable:	fare_amount	R-squared:	0.457
Model:	OLS	Adj. R-squared:	0.456
Method:	Least Squares	F-statistic:	993.4
Date:	Tue, 14 Jul 2020	Prob (F-statistic):	0.00
Time:	17:23:52	Log-Likelihood:	-32618.
No. Observations:	11836	AIC:	6.526e+04
Df Residuals:	11825	BIC:	6.534e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
pickup_longitude	-11.9084	3.036	-3.922	0.000	-17.860	-5.957
pickup_latitude	8.9927	2.338	3.846	0.000	4.409	13.576
dropoff_longitude	15.4025	2.681	5.745	0.000	10.147	20.658
dropoff_latitude	-15.8025	2.102	-7.516	0.000	-19.924	-11.681
dist	1.9914	0.021	97.110	0.000	1.951	2.032



dist	1.9914	0.021	97.110	0.000	1.951	2.032
passenger_count_1	540.5821	341.148	1.585	0.113	-128.124	1209.289
passenger_count_2	540.7046	341.151	1.585	0.113	-128.007	1209.416
passenger_count_3	540.8026	341.152	1.585	0.113	-127.911	1209.516
passenger_count_4	540.5525	341.152	1.584	0.113	-128.162	1209.267
passenger_count_5	540.4306	341.149	1.584	0.113	-128.277	1209.139
passenger_count_6	541.7265	341.149	1.588	0.112	-126.981	1210.434

Omnibus:	6042.940	Durbin-Watson:	1.991
Prob(Omnibus):	0.000	Jarque-Bera (JB):	46953.014
Skew:	2.342	Prob(JB):	0.00
Kurtosis:	11.559	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 45.6%, 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 that we will follow 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 bit easy when compared to 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

The above performed methods of Decision Tree, Random Forest, Linear Regression and KNN Method are the various models that can be developed for the given data.

When we got the the Data it was 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.

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.

In order 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.

We have not used RMSE because we are not working with Timestamp value.

Mean Absolute Error(MAE)

Given any test data-set, Mean Absolute Error of your model refers to the mean of the absolute values of each prediction error on all instances of the test data-set.

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

MAPE = function(y, yhat){

  mean(abs((y - yhat)/y*100))

}
```

a. In R:

Method	MAPE error(%age)
Decision tree	48.62096
Random forest	38.53504
Linear regression	43.76524
KNN Imputation	43.35813

b. In Python:

Method	MAPE error(%age)
Decision tree	28.15971
Random forest	23.65777
Linear regression	44.94933
KNN Imputation	42.42999

Accuracy:

In order to identify or compare for better model the next step is observing 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)

We can also calculate Accuracy by using MAE value.

Accuracy = 1- MAPE

Accuracy in R:

Method	Accuracy(%age)
Decision tree	51.37
Random forest	61.46
Linear regression	56.23
KNN Imputation	56.64

Accuracy in Python:

Method	Accuracy(%age)
Decision tree	71.84
Random forest	76.34
Linear regression	55.05
KNN Imputation	57.57

Model Selection:

After we have completed the comparison of the error matrix, the next step that we need to follow is Selection of the most effective model.

On observing the values of Error and accuracy, we find that all the models perform close to each other.

In this case any model can best used for further processes, but on close observation we can conclude that Random forest gives better results compared to all other methods.

So I will prefer Random Forest Model to be used for further processes of prediction.

References:

Websites:

- www.edwisor.com
  : Videos from Mentor
- https://stackoverflow.com/questions/51488949/use-haversine-package-to-compare-all-distances-possibilities-of-a-csv-list-of-lo

  : Haversine Doubt

- https://gist.github.com/rochacbruno/2883505 : Calculate Haversine in Python

Video Channels:

- https://www.youtube.com/watch?v=7YfyIhhmwq4 – distance development in python
- Anaconda Inc – python coding
- CS Dojo – python coding
- Treehouse – R coding