CAB FARE PREDICTION

PROJECT SUBMITED BY VISHAKHA GAIKWAD

Table of content

CHAPTER 1

1.1 PROBLEM STATEMENT

You are a cab rental start-up company. You have successfully run the pilot project and now want to launch your cab service across the country. You have collected the historical data from your pilot project and now have a requirement to apply analytics for fare prediction. You need to design a system that predicts the fare amount for a cab ride in the city.

So, by understanding the problem, need we have to develop system which predict accurate value of the fare amount. Our problem statement comes under the category of forecasting/prediction statement category.

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

2.1 Business Understanding

Understanding business of client is important to define the objective of the problem statement. In this case the company want to predict the fare amount, to generate the good amount of revenue from it.

2.2 Data understanding

Data understanding is very crucial part for the further procedure. Here, we have two sets of data in the form of CSV file.

train data = consist of 7 variables and 16067 Observation.

test data = consist of 6 variable and 9514 Observations.

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

from the above we have come to know the categories of the variable

there are 5 independent variable such as :-

pickup_datetime, pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, passenger_count.

and one dependent variable is :-**fare_amount**

From the given train data, it is come to know that, we have to predict fare amount, and other variables will help were going to help to achieve that, here pickup_latitude/longitude, dropoff_latitude/longitude this data are signifying the location of ride starting and ride ending. So, these variables are important in the process. Passenger_count is another variable, that explains about how many people or passenger are in 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.

So we considering pick up date time variable as incomplete source of data we take this variable as redundant I have removed it from my given data.

2.3 Data Preparation

Data preparation is the act of manipulating raw data into a form that can readily and accurately be analyzed for business purposes. 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.

The data preparation generally done by exploring the data, the process also known by by Exploratory Data Analysis: - exploratory data analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics, which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing and handling missing values and making transformations of variables as needed.

So here we have explored our data by the following EDA technique. by checking structure of data here I found that the **fare amount** variable which in categorical form so I convert into a numeric as data required. I have also convert some observation into Nan, or NA form because the variable **pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude have same value.**

Also convert all '0' to NA from all variable because counting 0 from variable list not give us important information.

2.4 Data Preprocessing

Data preprocessing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. In this process we clean our data with process such as missing value and outlier analysis.

The following process were applied on data to improve data quality for further process.

2.4.1 Missing Value Analysis: -

In the given data their value missing which is found in following type

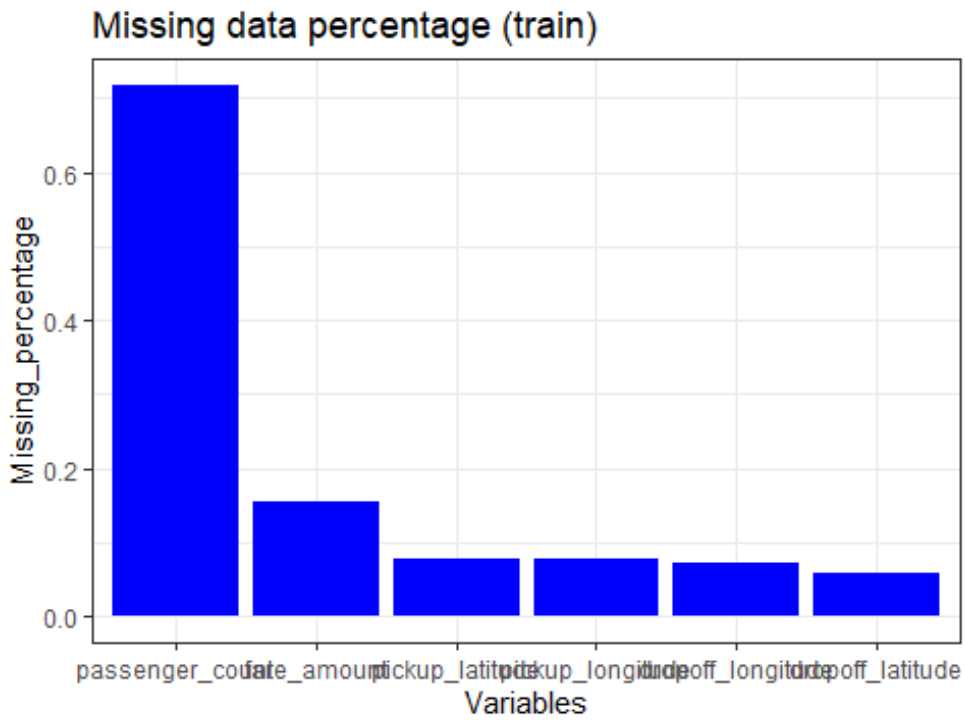
1. Blank spaces: Which are converted to NA and Nan in R and Python respectively for further operations
2. Zero Values: This is also converted to NA and Nan in R and python respectively prior further operations
3. Repeating Values: there are lots of repeating values in pickup_longitude, pickup_latitude,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

Given data has missing value which is summaries in table with variable respectively.

Sr.no	Variables	count	Missing_percentage
1	passenger_count	112	0.717581
2	fare_amount	24	0.153767
3	pickup_longitude	12	0.076884
4	pickup_latitude	12	0.076884
5	dropoff_longitude	10	0.070477
6	dropoff_latitude	9	0.057663

Table 1: - Missing percentage

Missing value with graphical presentation as below



Graph 1: Missing Values

Above mentioned 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

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

2.4.1.1 Imputing 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.

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.

Given below showing the result of data

#experiment to choose exact method

actual value = 17.5

mean = 15.15801

#median = 8.5

#in = 15.90051

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

2.4.2 Outlier Analysis

Outlier is an abnormal observation that stands or deviates away from other observations. 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.

In the given data set I have done outlier analysis I found that data is some value are extremely far away from actual value this are explain below according to variable.

Fare_Amount:

Under the fare amount variable some observation has -ve value which not convenient because **Fare_Amount** can't be negative so I convert it into NA

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: - Fare_Amount -ve ranges

passenger_count

Here in this data the range passenger is more than 6, Generally sitting capacity of cab is 6-seater so I convert all other ranges of passenger to NA which is more than 6-seater.

```
> unique(train$passenger_count)
 [1]    1    2    3    6    5    4  456 5334  535  354   55  554   53   35  3
45 5345  536   43   58
[20]  537   87  531  557
```

pickup_longitude, pickup_latitude, dropoff_longitude and dropoff_latitude

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. so convert them into NA.

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 3:- pickup_longitude, pickup_latitude, dropoff_longitude and dropoff_latitude above ranges

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.

2.4.3 Feature Engineering

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

In this project the data contains only the pick up and drop points in longitude and latitude. The fare_amount will main 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.by using haversine formula have calculate the distance in python and r respectively

In r:

Create function to radiant and use to other variable haversine

```
deg_to_rad = function(deg){
  (deg * pi) / 180
}
haversine = function(long1,lat1,long2,lat2){
  #long1rad = deg_to_rad(long1)
  phi1 = deg_to_rad(lat1)
  #long2rad = deg_to_rad(long2)
  phi2 = deg_to_rad(lat2)
  delphi = deg_to_rad(lat2 - lat1)
  dellamda = deg_to_rad(long2 - long1)
  a = sin(delphi/2) * sin(delphi/2) + cos(phi1) * cos(phi2) *
    sin(dellamda/2) * sin(dellamda/2)
 c = 2 * atan2(sqrt(a),sqrt(1-a))
  R = 6371 # radius of earth  (R * c) }
```

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))
```

and after using function I got value of distance which is tabled below

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	dist
1	4.500000	-73.84431	40.72132	-73.84161	40.71228	1	1.0307639
2	16.900000	-74.01605	40.71130	-73.97927	40.78200	1	8.4501336
3	5.700000	-73.98274	40.76127	-73.99124	40.75056	2	1.3895252
4	7.700000	-73.98713	40.73314	-73.99157	40.75809	1	2.7992702
5	5.300000	-73.96810	40.76801	-73.95665	40.78376	1	1.9991568

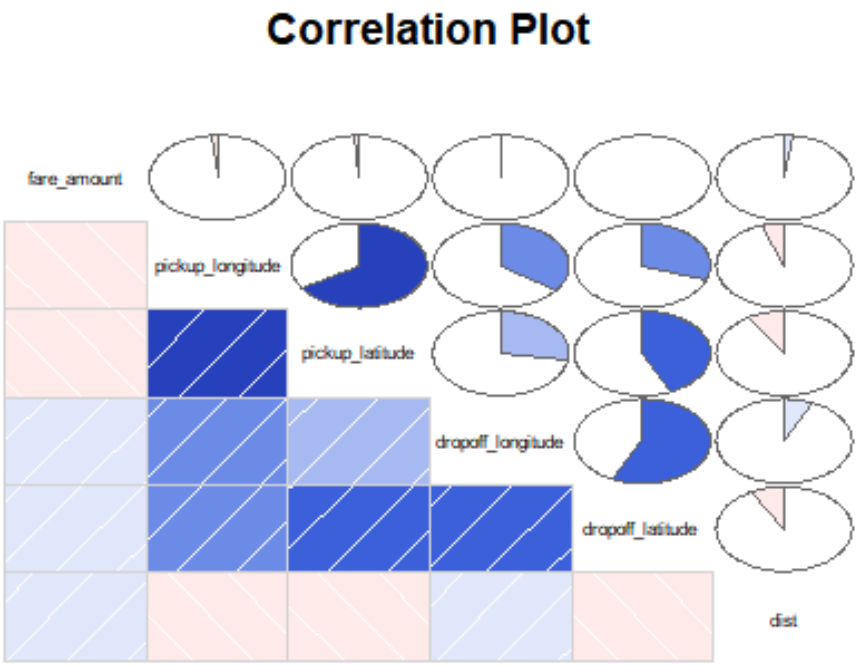
Table 4 Engineering new Variable Distance

2.4.4 Checking For Collinearity And Multicollinearity

Collinearity

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.



Graph2 -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.

Multicollinearity

Multicollinearity is a state of very high intercorrelations or inter-associations the among independent variables. It is therefore a type of disturbance in the data and if present in the data the statistical inferences made about the data may not be reliable.

Problem caused by multicollinearity

It is caused by the inclusion of a variable which is computed from other variables in the data set.

Multicollinearity can also result from the repetition of the same kind of variable

```
          Variables      VIF
1  pickup_longitude 1.926663
2   pickup_latitude 2.059613
3 dropoff_longitude 1.700380
4  dropoff_latitude 1.800112
5              dist 1.044998
>
> vifcor(train[,-c(1,6)], th = 0.9)
No variable from the 5 input variables has collinearity problem.

The linear correlation coefficients ranges between:
min correlation ( dist ~ dropoff_latitude ):  -0.04986098
max correlation ( pickup_latitude ~ pickup_longitude ):  0.6572771

---------- VIFs of the remained variables --------
          Variables      VIF
1  pickup_longitude 1.888913
2   pickup_latitude 2.026291
3 dropoff_longitude 1.660275
4  dropoff_latitude 1.771107
5              dist 1.031828
```

2.5 Model Development Step

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.

2.5.1 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.

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.

```
n= 11839

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

 1) root 11839 312023.100  9.529000
   2) dist< 2.972071 8309 121172.400  7.588610
     4) dist< 1.682712 4790  53237.180  6.366011 *
     5) dist>=1.682712 3519  51029.500  9.252791 *
   3) dist>=2.972071 3530  85928.530 14.096340
     6) dist< 4.476965 2006  34766.140 12.385390 *
     7) dist>=4.476965 1524  37560.650 16.348410
      14) dist< 7.025449 1262  28387.130 15.613460 *
      15) dist>=7.025449 262   5208.306 19.888550 *
```

Plot :Decision tree in r

The above plot shows the rules of splitting of trees. The main root splits into 2 nodes having dist< 2.972071 8309 and dist>=2.972071 8309 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

```
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 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.

2.5.2 Random Forrest

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.

Random forrest in r

```
Call:
 randomForest(formula = fare_amount ~ ., data = train1, importance = TRUE,
ntree = 100)
               Type of random forest: regression
                     Number of trees: 100
No. of variables tried at each split: 2

          Mean of squared residuals: 9.406779
                    % Var explained: 64.31
```

Plot: Random Forrest in r

Here in this model we have provide the n of tree and node split into =2

```
importance(RF_model, type = 1)
                   %IncMSE
pickup_longitude  23.256222
pickup_latitude   19.482203
dropoff_longitude 31.995182
dropoff_latitude  20.997627
passenger_count    1.764448
dist              64.562459
```

Plot: importance Random Forrest in r

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

Random Forrest In Python

```
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 Forrest In Python

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

2.5.3 Linear regression model

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.

Linear Regression In r

```
lm_model = lm(fare_amount ~. , data = train1)
> summary(lm_model)

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

Residuals:
     Min       1Q   Median       3Q      Max
-19.6573  -1.9199  -0.9360   0.6795  23.7470

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       493.03208  330.16674   1.493 0.135390
pickup_longitude  -10.72520    2.99645  -3.579 0.000346 ***
pickup_latitude     8.68713    2.31122   3.759 0.000172 ***
dropoff_longitude  14.17704    2.59784   5.457 4.93e-08 ***
dropoff_latitude  -14.40895    2.04693  -7.039 2.04e-12 ***
passenger_count2    0.12433    0.10024   1.240 0.214891
passenger_count3    0.13870    0.17155   0.809 0.418800
passenger_count4    0.24395    0.24533   0.994 0.320059
passenger_count5   -0.02057    0.14115  -0.146 0.884138
passenger_count6    0.80058    0.25466   3.144 0.001672 **
dist                2.00992    0.02036  98.716  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.753 on 11828 degrees of freedom
Multiple R-squared:  0.4661,   Adjusted R-squared:  0.4656
F-statistic:  1033 on 10 and 11828 DF,  p-value: < 2.2e-16
```

Plot : linear regression in r

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 46.61%, which indicated that only 46.56% of the variance of fare_amount is explained by the input variables. This explains the model is not upto the mark.

Linear Regression Model In Python

Out[70]:

OLS Regression Results

Dep. Variable:	fare_amount	**R-squared:**	0.446
Model:	OLS	**Adj. R-squared:**	0.446
Method:	Least Squares	**F-statistic:**	953.7
Date:	Sat, 21 Dec 2019	**Prob (F-statistic):**	0.00
Time:	15:35:12	**Log-Likelihood:**	-32690.
No. Observations:	11836	**AIC:**	6.540e+04
Df Residuals:	11825	**BIC:**	6.548e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	**std errt**	**P>|t|**	**[0.025**	**0.975]**
pickup_longitude	-13.0647	3.055	-4.277	0.000	-19.053 -7.076
pickup_latitude	9.5563	2.352	4.062	0.000	4.945 14.167
dropoff_longitude	15.9457	2.700	5.905	0.000	10.653 21.239
dropoff_latitude	-16.6564	2.113	-7.881	0.000	-20.799 -12.514
Distance	1.9740	0.021	94.976	0.000	1.933 2.015
passenger_count_1	507.1022	337.695	1.502	0.133	-154.836 1169.040
passenger_count_2	507.1442	337.698	1.502	0.133	-154.799 1169.087
passenger_count_3	507.4153	337.701	1.503	0.133	-154.534 1169.364
passenger_count_4	507.2745	337.698	1.502	0.133	-154.669 1169.218
passenger_count_5	507.0954	337.695	1.502	0.133	-154.843 1169.034
passenger_count_6	508.3031	337.695	1.505	0.132	-153.635 1170.241

Omnibus:	6068.909	**Durbin-Watson:**	2.016
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	46692.662
Skew:	2.360	**Prob(JB):**	0.00
Kurtosis:	11.509	**Cond. No.**	2.81e+06

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 44.6%, which explains that only 44.66% 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.

2.5.4 KNN Imputation

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

3.1 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)
}
```

In r :

Method of Model	Error Rate (MAPE in percentage)
1.Decision tree	28.03431
2.Random forrest	22.11141
3 Linear Regression	26.15438
4. KNN Imputation	34.37037

In Python :

Method of Model	Error Rate (MAPE in percentage)
1.Decision tree	29.92841367123023
2.Random forrest	25.141362147213382
3 Linear Regression	26.814478088303616
4. KNN Imputation	33.96770745236633

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

Accuracy Rate = Accuracy*100

In r :

Method of Model	Accuracy Rate(MAPE in percentage)
1**.**Decision tree	71.96569
2.Random forrest	77.88859
3 Linear Regression	73.84562
4. KNN Imputation	65.62963

In Python :

Method of Model	Accuracy Rate(MAPE in percentage)
1**.**Decision tree	70.07158632876977
2.Random forrest	74.85863785278661
3 Linear Regression	73.18552191169638
4. KNN Imputation	66.03229254763367

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.

References

https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15

https://blog.exploratory.io/a-practical-guide-of-exploratory-data-analysis-with-linear-regression-part-1-9f3a182d7a9

https://video.edwisor.com/video/

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

https://www.youtube.com/watch?v=E5RjzSK0fvY