**CAB RENTAL PROJECT**

**Introduction**:

Today’s world we know the vast increase in rental cabs in big cities.We can see how radiant increase in customers for the usage of cabs every minute.We can book online cabs in minutes of time and reach to required destination very fast.Even though everyone have their own means of transport if his automobile is in trouble he opts to find for a cab to save time while running to office, schools,meetings,functions etc.Lot of young people who don’t have own means of transport opt for rental cabs for one or the other issue.Even in case of emergencies what if our own transport is not working or one of our relatives took the cab to out of station? If we have to rush to hospital and need of a transport, Cab is required. So, there is a huge scope in Cab rental company. As it is a startup company we have fix our machines to predict the cost for the usage of our company cabs in a productive manner.Fare is the main aspect of any cab company.

We need a software which can predict the cost of usage of cabs by our customers as per the distance it travelled.To start the company first we need to predict the cost as per the distance cab going to travel .Machine learning algorithms plays a vital role to estimate the cost.In this project we are going to see how machine learning algorithms are useful to predict the cost.From linear regression analysis to random forest algorithm we will see which is a great algorithm to predict the right amount in future by the usage of past data.

**Data Set Intro:**

We have a past data which helps us to predict the cost for future cases. We have a split of data where one is train data with fare cost included in it. We have a test data on which we have to predict the cost based on some common variables as distance, pickup location and drop location etc..So lets have a clear understanding of given data.

In train data we have 7 variables and 16067 observations.Our purpose of this project is to analyse which machine learning algorithm fits best for this data to predict the fare\_amount variable in the given data set.

Train data explains the cost of fare how it was calculated in past by previous companies by using attributes like pickup time, pick up location, drop location etc..

The variables ‘pick up longitude’ and ‘pick up latitude’ explains the pick up location on earth from where passenger was picked up.In the same way ‘drop longitude’ and ‘drop latitude’ explains where passenger is dropped off.

**Exploratory Data Analysis** :

So, this is a main part of our project which clearly very necessary for our right predictions and less error rate and to choose the right algorithm. Data pre - processing is a technique where we analyse the data most probably we try to understand the data which could be most important part of any data scientist.This is the place where we pre-process the data for the model .In this process we see how data is significantly explaining the dependent variable to have a better understanding to which model we choose to predict the dependent variable.

When we see the data which has been given, we could see there are variables like pickup longitude , pick up latitude, drop off latitude and drop off longitude. Which is the most important part of this data as it explains the distance of the cab it travelled. Wherever we use any means of transport any fare prediction will be based on how much distance we travelled using the transport. So do you think these variables clearly explain how much distance it is travelled?

Before proceeding to do further calculations lets see whether we have any missing values in the data.

By using ‘summary’ (is.na(dataset)) function in R and ‘isnull().sum()’ function in python we could see there are missing values in variables ‘fare\_amount’ and ‘passenger count’ as 24 and 55 observations.

Hence in R I went ahead and imputed missing values by using KNN imputation from library called ‘DMwR’ by choosing k = 1 as I checked k =1 fits the right accurate value as observations in the variables doesnt have much deviation.

*library(DMwR)*

*CabCompany = knnImputation(CabCompany,k=1).*

In python I went and imputed the values by ‘mean’ which I felt it was a best fit by some basic calculations.

*CabCompany['passenger\_count'] = CabCompany['passenger\_count'].fillna(CabCompany['passenger\_count'].mean())*

*CabCompany['fare\_amount'] = CabCompany['fare\_amount'].fillna(CabCompany['fare\_amount'].mean())*

Now its a time for calculating the distance :

Its better we calculate the distance from the given values and tabulate into a new column distance (in km).

So, to do that we have a ***Haversine*** formula which calculates the distance by the given values.

HAVERSINE FORMULA :

|  |  |
| --- | --- |
|  | a = sin²(Δφ/2) + cos φ1 ⋅ cos φ2 ⋅ sin²(Δλ/2) |
| c = 2 ⋅ atan2( √a, √(1−a) ) |
| d = R ⋅ c |

So, where: φ is latitude, λ is longitude

Δφ = (lat2-lat1)

Δλ = (lon2-lon1) , in radians

Which gives the distance in kilometers of how much the cab travelled.So, this gives a clear picture of how data and makes easy to predict the required values.

Code used to calculate distance in R:

First of all I have collaborated all required values into data set called distance and then calculated the distance by below code which is clearly shown in R file .

Using cbind() function created a dataset to calculate distance:

*distance* = *cbind(CabCompany['pickup\_longitude'],CabCompany['pickup\_latitude'],CabCompany['dropoff\_longitude'],*

*CabCompany['dropoff\_latitude'])*

Converted the values to radians from degrees using library called (NISTUNITS):

*distance$pickup\_longitude=NISTdegTOradian(distance$pickup\_longitude)*

*distance$pickup\_latitude=NISTdegTOradian(distance$pickup\_latitude)*

*distance$dropoff\_longitude=NISTdegTOradian(distance$dropoff\_longitude)*

*distance$dropoff\_latitude=NISTdegTOradian(distance$dropoff\_latitude)*

Calculated the required difference :

Δφ = (lat2-lat1)

Δλ = (lon2-lon1) , in radians

*a=(sin(distance$latitudediff/2)\*sin(distance$latitudediff/2))+*

*cos(distance$pickup\_latitude)\*cos(distance$dropoff\_latitude)\**

*sin(distance$longitudediff/2)\*sin(distance$longitudediff/2)*

*c = 2\*atan2(sqrt(a),sqrt(1-a))*

Created a variable called distance\_covered :

*distance$distance\_covered = 6371\*c*

Code used to calculate distance in PYTHON:

Its the same way how I did it in R. Created a data set for distance and follow the below code.

In python also created a data set called distance by using below code:.

*Distance = CabCompany[['pickup\_longitude','pickup\_latitude','dropoff\_longitude','dropoff\_latitude']]*

Converted the values to radians from degrees using numpy(np) library and built - in function (np.deg2rad):

*Distance['pickup\_longitude']=(np.deg2rad(Distance['pickup\_longitude']))*

*Distance['pickup\_latitude']=(np.deg2rad(Distance['pickup\_latitude']))*

*Distance['dropoff\_longitude']=(np.deg2rad(Distance['dropoff\_longitude']))*

*Distance['dropoff\_latitude']=(np.deg2rad(Distance['dropoff\_latitude']))*

Calculated the required difference :

Δφ = (lat2-lat1)

Δλ = (lon2-lon1) , in radians

*a = (np.sin(Distance['latitudediff']/2)\*np.sin(Distance['latitudediff']/2))+np.cos(Distance['pickup\_latitude'])\*np.cos(Distance['dropoff\_latitude'])\*np.sin(Distance['longitudediff']/2)\*np.sin(Distance['longitudediff']/2)*

*c = 2\*np.arctan2(np.sqrt(a),np.sqrt(1-a))*

Created a variable called distance\_covered :

*Distance['distance\_covered'] = 6371\*c*

Then added this variable to our main dataset (CabCompany).

Main part of the data pre - processing is completed as we got the values for distance for which we have to basically predict the fare amount according to the distance cab is travelled.

So we have got the distance,now I have removed all other variables which were not necessary.

Before removing the variables I have extracted the Year from ‘pick\_up\_datetime’ variable to see whether there is any linear relationship between amount and year by using ‘as.Date’ function which is an inbuilt function in R which can be seen by the below code:

*Dateconvert = as.Date(strptime(CabCompany$pickup\_datetime, "%Y-%m-%d %H:%M:%S",tz='UTC'))*

*CabCompany$Date = Dateconvert*

*year = as.numeric(format(CabCompany$Date,'%Y'))*

*CabCompany$year = year*

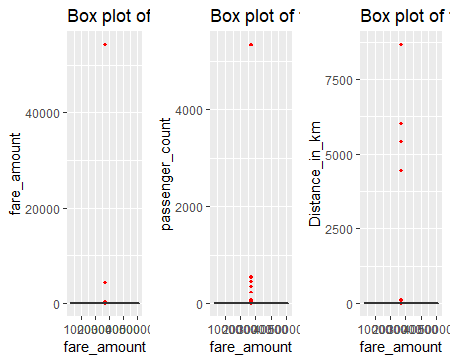
So we have got only 4 variables in the data set after doing basic calculations to have a clear picture on the data. The 4 variables are:

‘fare\_amount’,’passenger\_count’,Distance\_in\_km’,’Year’.In which one is dependent variable(fare\_amount) and rest are independent variables.

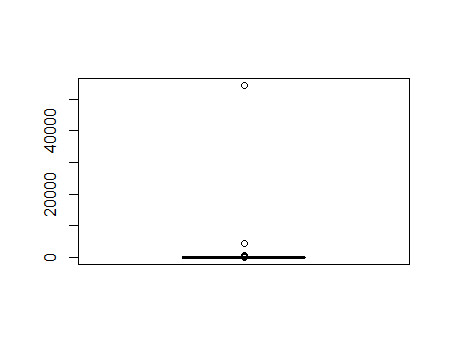
**VISUALIZATIONS**:

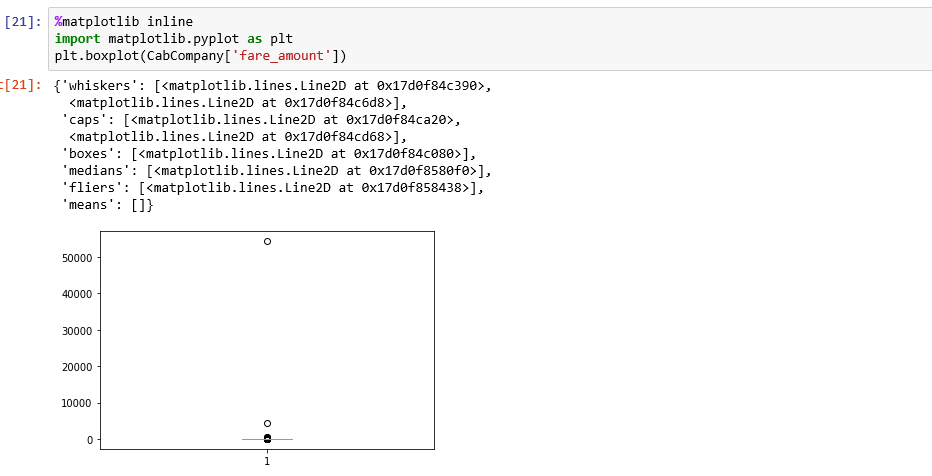
**OUTLIERS:**

Now its the time to check for outliers as we have dimension-ed the data and chose required variables.So is there any outliers in the data which might impact model performance or even model development . I have plotted the box plot and we could see below that box plots are not visible clearly but there are outliers in the data (which are red dots) and are very high in values. As observations are pretty enough to predict the values ,I have removed all outliers by using from the data which lead to dimension reduction in the data. Below is the graph of outliers .



*We can even simply type boxplot(Cabcompany$fare\_amount) and visulaize the outliers as shown below in R and python.*

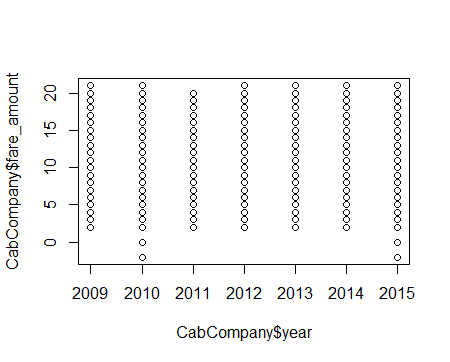
**



So from above graphs we confirm that outliers are present in the data which can affect the model development hence I removed the outliers and after removing the outliers.Its a time for visualization of the present data.

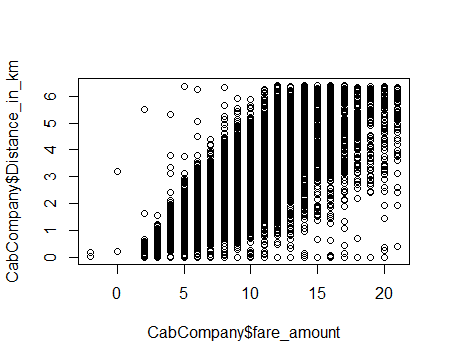
Do you think is there any linear-relation ship between year and fare amount?Don’t you think as increase in year fare prices would have changed?.So lets plot a graph and see how the data is distributed.

*plot(CabCompany$year , CabCompany$fare\_amount )*



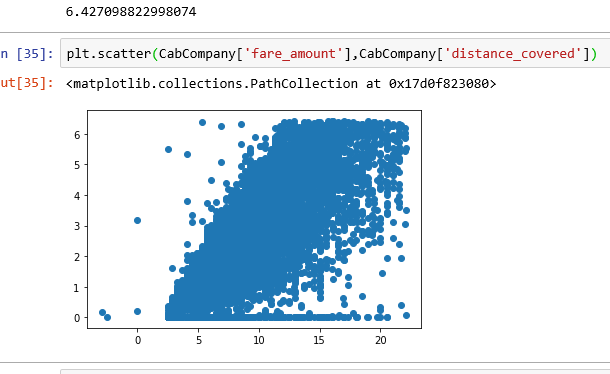
We were wrong, there is no linear relationship between fare amount and year but by the above graph it confirms that there is no fare change but the amount is equally distributed in every year.

So is there any variable which is having linear relation like fare\_amount and distance.Lets see by using plot function again.



Isn't look like linear relation ship.Do we really have to apply LINEAR REGRESSION on this data?.

Lets have a look on scatter plot of same data from python:



Do we really need to implement Linear regression ?.Lets apply linear regression and see how data is distributed and confirm whether it is a fit model for this data for future predictions.

> vifcor(CabCompany[,-1], th = 0.9)No variable from the 3 input variables has collinearity problem.

**MULTIPLE LINEAR REGRESSION**:

Linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables).Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a \*X + b

In R:

We know for linear regression we use lm function .

Syntax:

lm(predictor variable ~ required variables , data= dataset)

R code:

*Linear\_regression = lm(fare\_amount ~. ,data = CabCompany)*

*summary(Linear\_regression)*

Call:

lm(formula = fare\_amount ~ ., data = CabCompany)

Residuals:

Min 1Q Median 3Q Max

-11.7685 -1.4423 -0.4817 0.8798 17.0027

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -771.81175 22.21659 -34.740 < 2e-16 \*\*\*

passenger\_count 0.11913 0.03828 3.112 0.00186 \*\*

Distance\_in\_km 1.94237 0.01428 136.028 < 2e-16 \*\*\*

year 0.38547 0.01104 34.905 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.303 on 12486 degrees of freedom

Multiple R-squared: 0.614, Adjusted R-squared: 0.6139

F-statistic: 6622 on 3 and 12486 DF, p-value: < 2.2e-16

*So, we got R-squared value as 0.614 which is fine.*

1. squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination

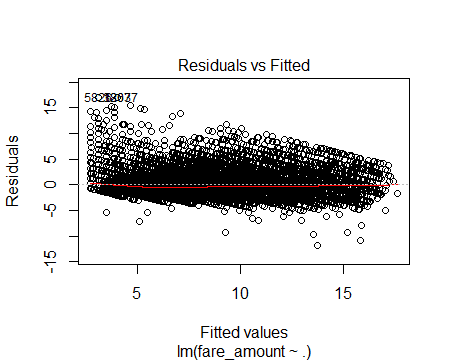
The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model

We could even see how passenger\_count, Distance\_in\_km , year variables are related and explaining the predictor variable by p value and t value.

But intercept value is -771 which is negatively related to the regression line.

Residual error is also 2.303

Lets have visuals and have a better understanding on residuals and how data is distributed to regression line.



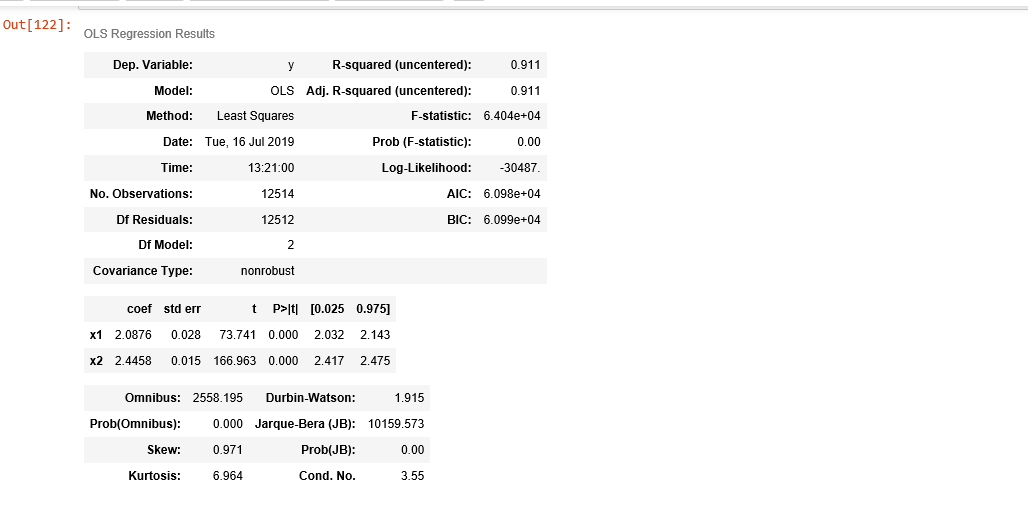
Python code :

By using statsmodel module we have predicted linear regression in python.

*import statsmodels.api as sm*

*Linearmodel = sm.OLS(Y, X).fit()*

*Linearmodel.summary()*

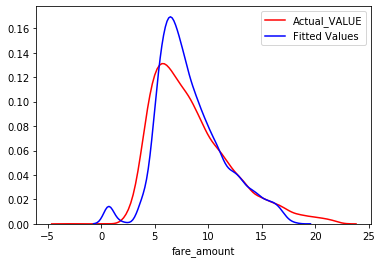


So, here in python we got R-squared value as 0.911 which is good that means that 90% of data is explained by the regression line.

So,I have predicted the values by using the above model

*predictions = Linearmodel.predict(X)*

and plotted a graph to see how well the dependent variable is predicted by the above python model and could see as shown below.



Well, its a pretty good prediction by Multiple Linear regression on given data.as could see a less difference .But until unless we see a value we cannot be able to identify how well the model is accurate.

We have error metrics for regression models to have how correctly model has predicted the data .We can calculate by squaring the difference between actual values and predicted values sum.

So I have got values for error metrics in R and python as shown below:

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.648497 | 1.7534 |
| RMSE | 2.302404 | 1.32417 |
| MSE | 5.301062 | 6.52608 |
| MAPE | Inf | Inf |

So above values are calculated from built in functions .

In R code is : by using regr.eval function

*regr.eval(CabCompany$fare\_amount,prediction, stats = c('mae','rmse','mape','mse'))*

In Python : by using metrics function from sklearn module.

*from sklearn import metrics*

*metrics.mean\_squared\_error(Y,predictions)*

*metrics.mean\_absolute\_error(Y,predictions)*

*np.sqrt(metrics.mean\_absolute\_error(Y,predictions))*

**LOGISTIC REGRESSION:**

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

I have implemented the logistic regression in R by the following code

With the use of glm function in R.

*Logistic\_regression = glm(fare\_amount~ . ,data = CabCompany)*

*summary(Logistic\_regression)*

Call:

glm(formula = fare\_amount ~ ., data = CabCompany)

Deviance Residuals:

Min 1Q Median 3Q Max

-11.7685 -1.4423 -0.4817 0.8798 17.0027

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -771.81175 22.21659 -34.740 < 2e-16 \*\*\*

passenger\_count 0.11913 0.03828 3.112 0.00186 \*\*

Distance\_in\_km 1.94237 0.01428 136.028 < 2e-16 \*\*\*

year 0.38547 0.01104 34.905 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 5.302761)

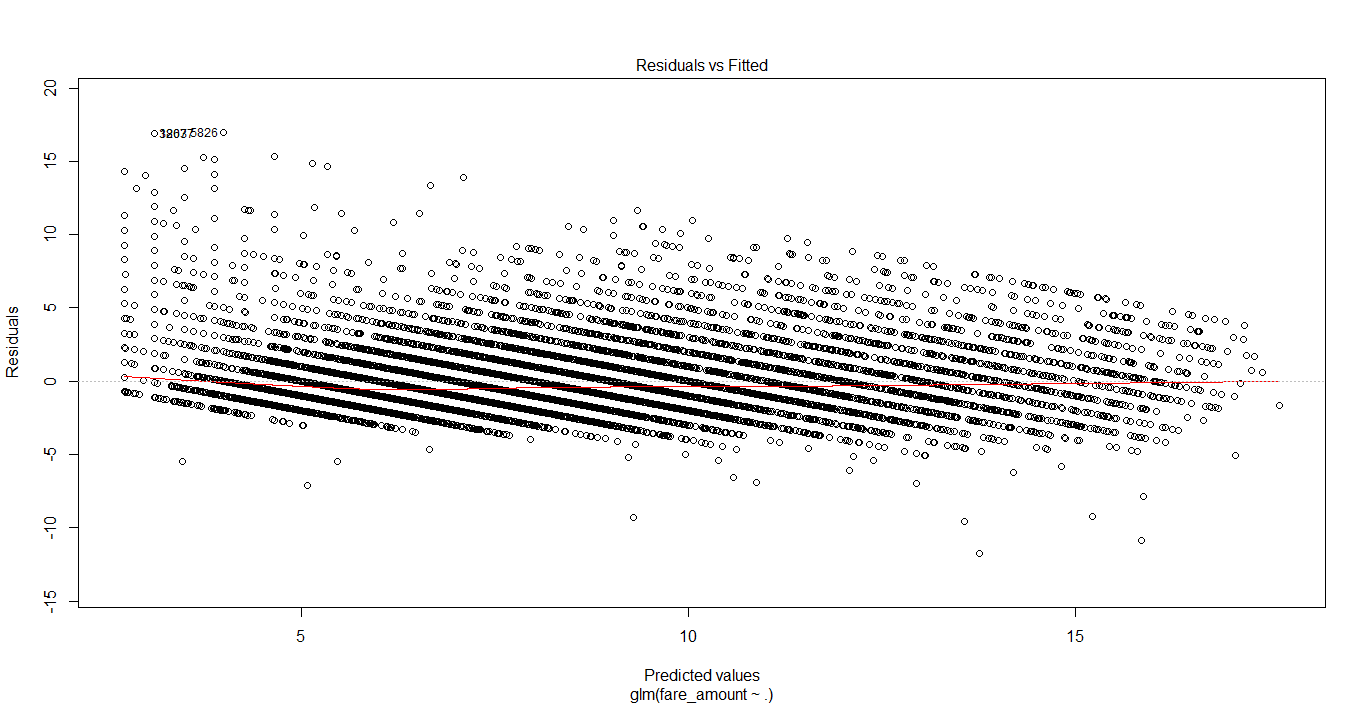
Null deviance: 171548 on 12489 degrees of freedom

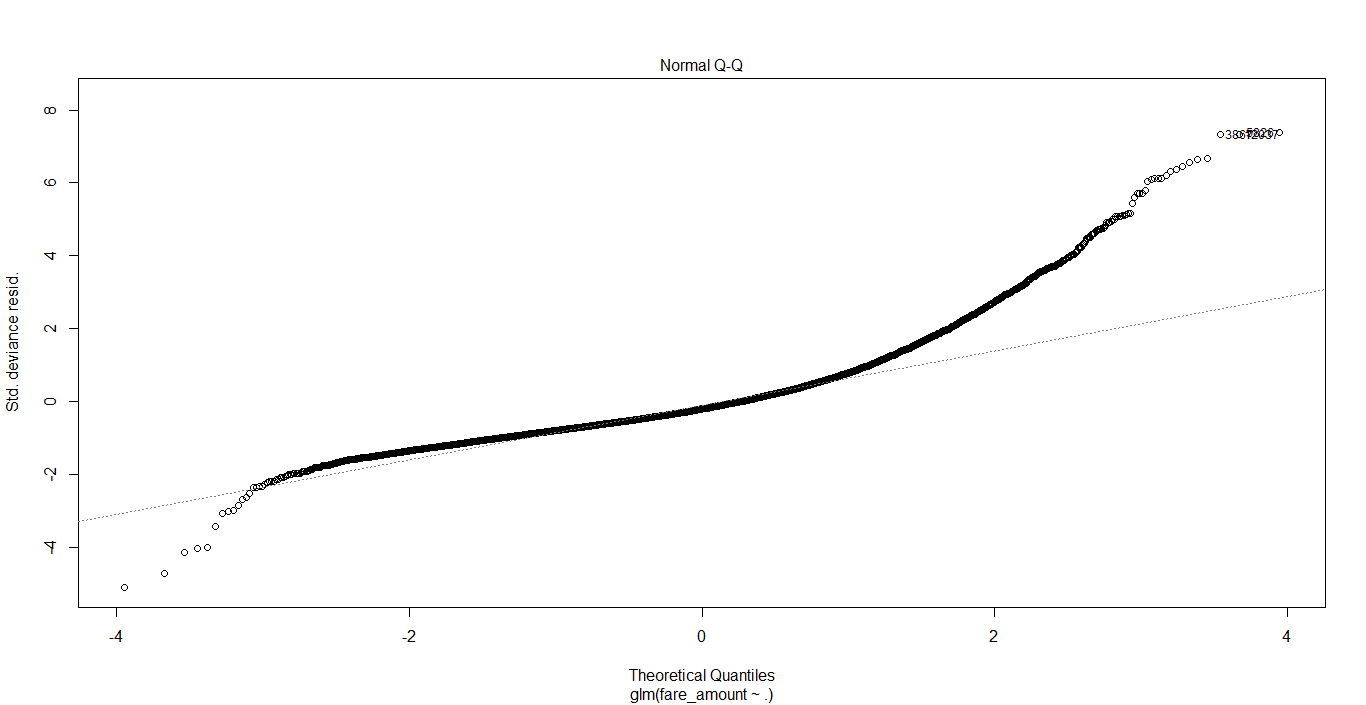
Residual deviance: 66210 on 12486 degrees of freedom

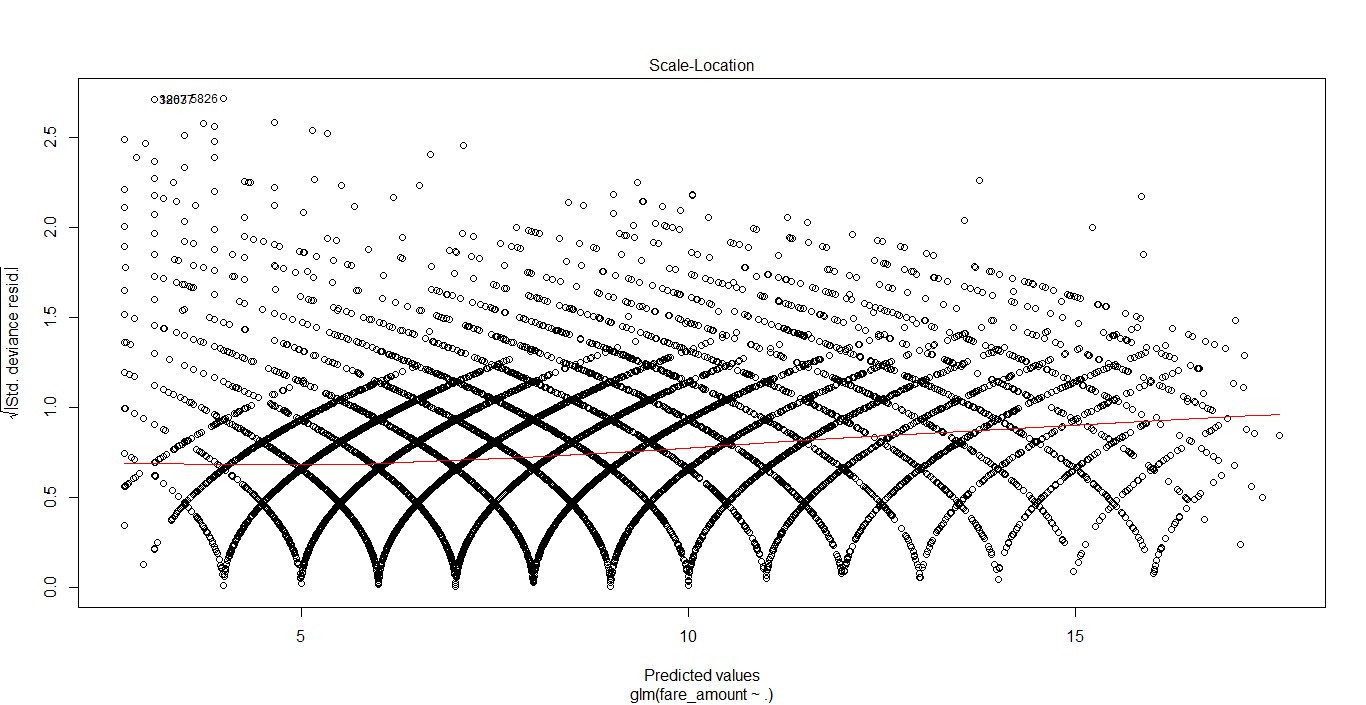
AIC: 56287

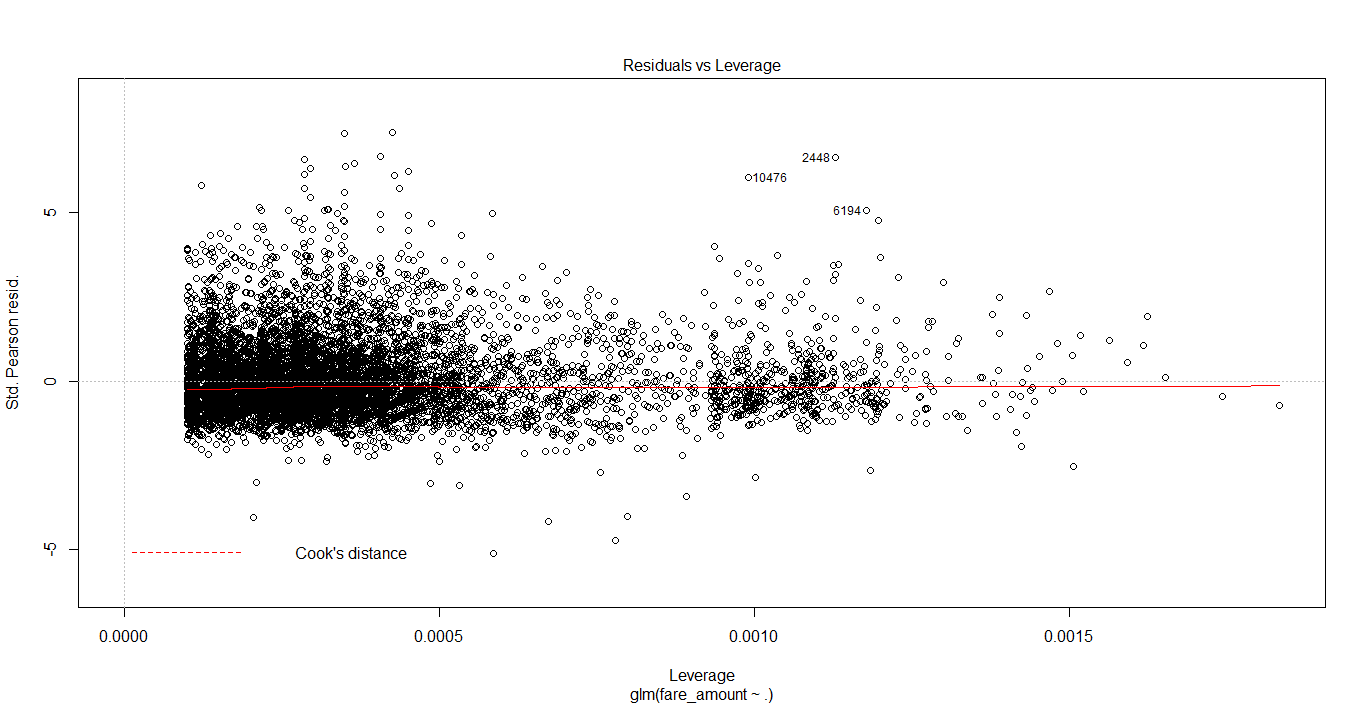
Number of Fisher Scoring iterations: 2

**Visuals:**









Is it quite similar as linear regression model.Yes,it is. Because logistic regression is a generalized version of linear regression.But here main observed value is AIC which should be less for consideration of model.So there will be no difference in the error metric calculations as observed below.

|  |  |
| --- | --- |
|  | R - programming |
| MAE | 1.648497 |
| RMSE | 2.302404 |
| MSE | 5.301062 |
| MAPE | Inf |

As we will get same values didn’t run logistic regression in python.

**DECISION TREE:**

**Now** its time for decision tree analysis.

Decision tree is a rule. Each branch connects nodes with “and” and multiple branches are connected by “or”

So we saw how linear regression was useful to predict the target variable.Now lets see the decision tree model and see whether it will be helpful for model evaluation for the right fit with less errors?

R code:

In R we use ‘Rpart’ library for model evaluation and we select method as ANOVA as it is a regression analysis.Below is the code:

*library(rpart)*

*fit = rpart(fare\_amount ~ ., data = CabCompany, method = "anova")*

*fit*

n= 12490

node), split, n, deviance, yval

\* denotes terminal node

1) root 12490 171548.000 8.084868

2) Distance\_in\_km< 2.533013 8256 49400.980 6.334908

4) Distance\_in\_km< 1.450251 4610 23164.500 5.404555 \*

5) Distance\_in\_km>=1.450251 3646 17201.040 7.511245

10) year< 2012.5 2269 7757.862 6.903482 \*

11) year>=2012.5 1377 7224.028 8.512709 \*

3) Distance\_in\_km>=2.533013 4234 47564.470 11.497170

6) Distance\_in\_km< 4.053708 2634 19685.780 10.171220

12) year< 2012.5 1655 8865.360 9.383686 \*

13) year>=2012.5 979 8058.744 11.502550 \*

7) Distance\_in\_km>=4.053708 1600 15624.160 13.680000

14) year< 2012.5 1005 8008.378 12.691540 \*

15) year>=2012.5 595 4975.287 15.349580 \*

Great! R did a great work for us by evaluation the rules and developing the model.

Next we predict the values by using predict function as below.

*prediction = predict(fit,CabCompany[,-1])*

*Python code:*

In python we will use DecisionTreeRegressor function from sklearn module as mentioned below.

*from sklearn.tree import DecisionTreeRegressor*

*fit\_DT = DecisionTreeRegressor(max\_depth=2).fit(X,Y)*

*predictions\_DT = fit\_DT.predict(X)*

*Here I selected max\_depth = 2 so that tree will not grow wide and less nodes for the simplicity purpose as we will be implementing random forest too.*

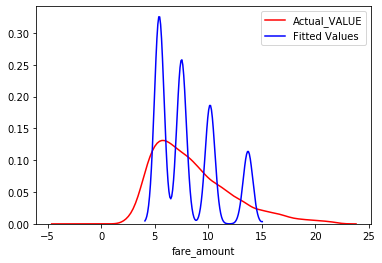
Its time for error metric calculations to know exactly how our model predicted the values.

I have plotted a distribution plot to see how the values are predicted compared to actual values.

*import seaborn as sns*

*ax1 = sns.distplot(CabCompany['fare\_amount'] , hist = False ,color = 'r',label = 'Actual\_VALUE')*

*sns.distplot(predictions\_DT, hist = False ,color ='b' ,label = 'Fitted Values',ax=ax1)*



*Error Table:*

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.76922 | 1.83688 |
| RMSE | 2.334243 | 1.35531 |
| MSE | 5.448692 | 5.9996 |
| MAPE | Inf | Inf |

We could see in R RMSE value is 2.33 and in python it is 1.355.

Is this value is required for analysis?

Lets implement our final model which is Random Forest.

**RANDOM FOREST:**

Now its time for random forest model to be built.

Random forest is an ensemble that consists of many decision trees.As we could see decision tree model has predicted quite different values lets see how random forest will be helpful.

In R:

We use randomforest library and we select ntree value to implement how many trees for the right prediction.In R I selected 52 trees as it was giving less error values and right predictions.Here is the code.

*library(randomForest)*

*RF\_model = randomForest(fare\_amount ~ ., CabCompany, importance = TRUE, ntree = 52 )*

*RF\_model*

Call:

randomForest(formula = fare\_amount ~ ., data = CabCompany, importance = TRUE, ntree = 52)

Type of random forest: regression

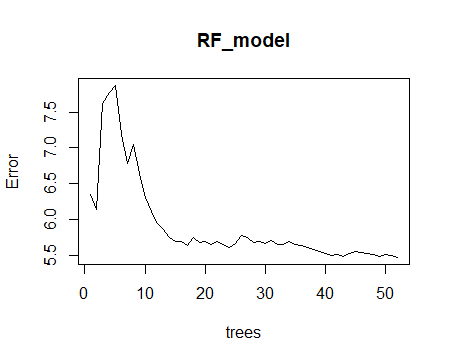
Number of trees: 52

No. of variables tried at each split: 1

Mean of squared residuals: 5.474052

% Var explained: 60.14

We can plot the graph and see how error reduced as we increase number of trees.



By using inTrees library we can execute the rules and see:

exec = extractRules(treeList, CabCompany[-1])

1768 rules (length<=6) were extracted from the first 52 trees.

We can see some extracted rules as below.

exec[1:2,][1] "X[,1]<=1.15 & X[,1]<=0.5 & X[,2]<=2.5246406035706 & X[,3]<=2012.5 & X[,3]<=2011.5"

1. "X[,1]<=1.15 & X[,1]>0.5 & X[,2]<=2.5246406035706 & X[,3]<=2012.5 & X[,3]<=2011.5 & X[,3]<=2009.5"

We can predict the values for random forest by predict function.

Now lets go to python.

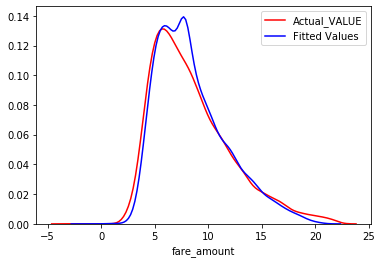
In Python by using RandomForestRegressor function from sklearn module we can implement the model for random forest as shown below.

*from sklearn.ensemble import RandomForestRegressor*

*RF\_model = RandomForestRegressor(n\_estimators = 20).fit(X,Y)*

*RF\_Predictions = RF\_model.predict(X)*

*So I have plotted a distribution plot for Random forest actual and predicted values.*



Great! Don’t you see the difference of predicted values of how accurate they are.

Error Table:

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.667116 | 0.78073 |
| RMSE | 2.260885 | 0.88359 |
| MSE | 5.111603 | 1.43884 |
| MAPE | Inf | Inf |

Even you could observe how drastically there is a reduction in error rate when we developed random forest for the given data in both visualization and error metrics.

So we have developed the models and I choose Random Forest as the best model of fit as it has very low error rate in both R and python programming.

**ERROR CALCULATIONS:**

We want to know how well the model predicts new data, not how well it fits the data it was trained with.Key component of most measures is difference between actual y and predicted y (“error”).

The *quality* of a regression model is how well its predictions match up against actual values, but how do we actually evaluate quality? Luckily, smart statisticians have developed **error metrics** to judge the quality of a model and enable us to compare regressions against other regressions with different parameters. These metrics are short and useful summaries of the quality of our data.

# **Regression Metrics**

* Mean Squared Error (MSE): basically measures average squared error of our predictions. For each point, it calculates square difference between the predictions and the target and then average those values.The higher this value, the worse the model is.
* Root Mean Squared Error (RMSE) : is just the square root of MSE. The square root is introduced to make scale of the errors to be the same as the scale of targets.First, they are similar in terms of their minimizers, every minimizer of MSE is also a minimizer for RMSE and vice versa since the square root is an non-decreasing function. For example, if we have two sets of predictions, A and B, and say MSE of A is greater than MSE of B, then we can be sure that RMSE of A is greater RMSE of B.And it also works in the opposite direction.
* Mean Absolute Error (MAE) : In MAE the error is calculated as an average of absolute differences between the target values and the predictions. The MAE is a linear score which means that **all the individual differences are weighted equally** in the average. For example, the difference between 10 and 0 will be twice the difference between 5 and 0. However, same is not true for RMSE.
* R Squared (R²) : The coefficient of determination, or R² (sometimes read as R-two), is another metric we may use to evaluate a model and it is closely related to MSE, but has the advantage of being **scale-free** — it doesn’t matter if the output values are very large or very small, **the R² is always going to be between -∞ and 1.**When R² is negative it means that the model is worse than predicting the mean.
* Adjusted R Squared (R²) :R² shows how well terms (data points) fit a curve or line. Adjusted R2 also indicates how well terms fit a curve or line, but adjusts for the number of terms in a model. If you add more and more **useless** variables to a model, adjusted R squared will decrease. If you add more **useful** variables, adjusted R squared will increase.
* Mean Absolute Percentage Error(MAPE) :The relative error preference can also be expressed with Mean Absolute Percentage Error, MAPE.For each object, the absolute error is divided by the target value, giving relative error. MAPE can also be thought as weighted versions of MAE.

We observed the above explained values in each and every model.We can compare the values for all the models.

**Linear regression:**

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.648497 | 1.7534 |
| RMSE | 2.302404 | 1.32417 |
| MSE | 5.301062 | 6.52608 |
| MAPE | Inf | Inf |

**Decision Tree:**

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.76922 | 1.83688 |
| RMSE | 2.334243 | 1.35531 |
| MSE | 5.448692 | 5.9996 |
| MAPE | Inf | Inf |

**Random Forest:**

|  |  |  |
| --- | --- | --- |
|  | R - programming | Python |
| MAE | 1.667116 | 0.78073 |
| RMSE | 2.260885 | 0.88359 |
| MSE | 5.111603 | 1.43884 |
| MAPE | Inf | Inf |

So as its a regression model which we are calculating we know MAE is the average value of difference of actual and predicted values. So I’ll choose MAE as my error metric and when I compare the values for all models and it clearly shows Random Forest is the best fit model for my prediction.Hence I proceed with Random Forest model for this data set.

**CONCLUSION**:

*Google’s self-driving cars and robots get a lot of press, but the company’s real future is in machine learning, the technology that enables computers to get smarter and more personal.*

*– Eric Schmidt (Google Chairman)*

Regression is the process of identifying patterns and calculating the

predictions of continuous outcomes. The system has to understand the numbers, their values, grouping (for example, heights and widths), etc.as you can see, different types of machine learning algorithms are solving different kinds of problems the combination of different algorithms makes a powerful capable of handling a wide variety of tasks and extracting valuable insights out of all sorts of information.But in this project Random Forest algorithm played a vital role and helped in predicting the values in a best way which is an outstanding job from machine learning algorithms.So thats all we have ended up with the prediction values so we can deploy the model in any application and use it for our future test cases. Rather we sit and calculate a machine learning algorithm will do it easily and help to start our new Cab Rental Company.

Lets start our new Cab Company with a built model for our fare prediction .

Thank you.