Cab Fare Amount Prediction Project

Table of contents

[Problem statement 2](#_Toc35270341)

[Methodology followed 2](#_Toc35270342)

[Data exploration / understanding the data 2](#_Toc35270343)

[Data cleaning and transformation 4](#_Toc35270344)

[Handling of the missing values and outliers 4](#_Toc35270345)

[Exploratory Data Analysis and correlation between variables 7](#_Toc35270346)

[Creating dummy variables and splitting into training and test Set. 10](#_Toc35270347)

[Model Selection 10](#_Toc35270348)

[Performance Evaluation 12](#_Toc35270349)

[Using the best model to predict fare\_amount for test data 13](#_Toc35270350)

# Problem statement

We have been given data for a cab rental start-up company.

Based on the pilot project we wish to launch this cab services facility. The historical data provided from the pilot project contains data regarding the pickup and drop locations (in co-ordinates), pickup time and passenger count.

We now need to apply analytics to predict fare amount i.e we need to design a system that predicts the fare amount for a cab ride in the city based on these variables.

Therefore fare\_amount is the dependent variable whereas other variables are independent variables for this project.

There are 2 files provided

1. Train\_cab – This is the pilot project dataset which has the fare amount column.
2. Test – This is data without the fare\_amount, we need to predict the fare\_amount based on this data(i.e. each line item wise)

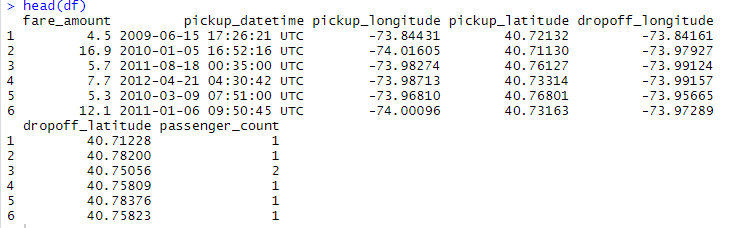
# Methodology followed

Below are the steps used to proceed in this project, each step is described along with the result as we proceed.

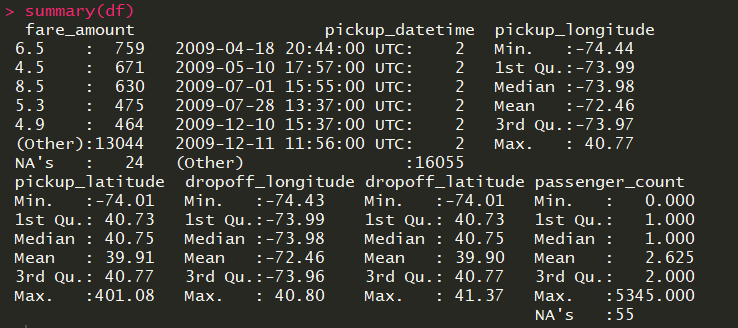
1. Data exploration / understanding the data
2. Data cleaning and transformation
3. Handling of the missing values and outliers
4. EDA , understanding the correlation between variables and feature/variable selection
5. Creating train and test set from within the train file.
6. Model fitting and testing different models
7. Model selection based on the performance evaluation of the test set
8. Using the best model to predict fare\_amount for test data ( i.e. test file provided which does not have the fare\_amount).

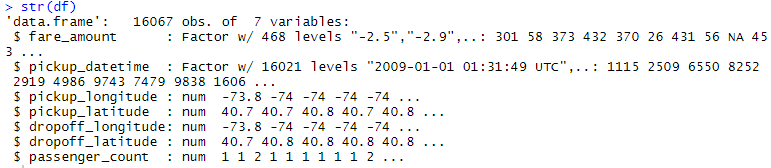
# Data exploration / understanding the data

First we check the data provided i.e. train data set as below:



We check the summary and data types of variables in the data:





Some inferences we draw are:

1. There are empty / missing values in the variables fare\_amount and passenger count.
2. The data must be having some error values e.g. – passenger count has maximum value in the column as 5345 which does not make any sense.
3. The variable fare\_amount is taken as factor data type whereas it should be number.
4. Logically we know that pickup coordinates will not be much useful to us in predicting the fare, but we can use these columns to get the distance of the cab ride.
5. There are negative values in the dependent fare\_amount variable. This must be error as fare cannot be of negative values
6. Pickupdate time – R has taken this variable as factor, we will need to convert this into date and source out the day, week, month, year variables from this to be useful in the prediction.

Based on this, we proceed to the next part - Data cleaning and transformation

# Data cleaning and transformation

As we have seen that we need to transform the data types for some columns as following: -

* Fare\_amount has been taken as factor type as seen above needs to be converted into numeric data type.

# Creating variables Date(Day), Year, Month, Weekday, Time from the variable pickup datetime)

* df$pickup\_datetime <- gsub('\\ UTC','', df$pickup\_datetime)

#Splitting Date and time

df$Date <- as.Date(df$pickup\_datetime)

df$Year <- substr(as.character(df$Date),1,4)

df$Month <- substr(as.character(df$Date),6,7)

df$Weekday <- weekdays(as.POSIXct(df$Date), abbreviate = F)

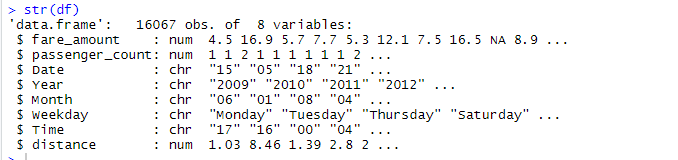
df$Date <- substr(as.character(df$Date),9,10)

df$Time <- substr(as.factor(df$pickup\_datetime),12,13)

df$fare\_amount <- as.numeric(as.character(df$fare\_amount))

* Using the 4 variables – pickup\_longitude, pickup\_latitude, dropoff\_longitude, dropoff\_latitude we can create a new variable distance (in Km) the code for this is available in the coding files.
* Once we have the new variable of distance, we can then remove these 4 variables as we can use the distance variable for further analysis.

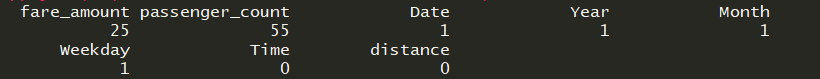
Below is the data now:

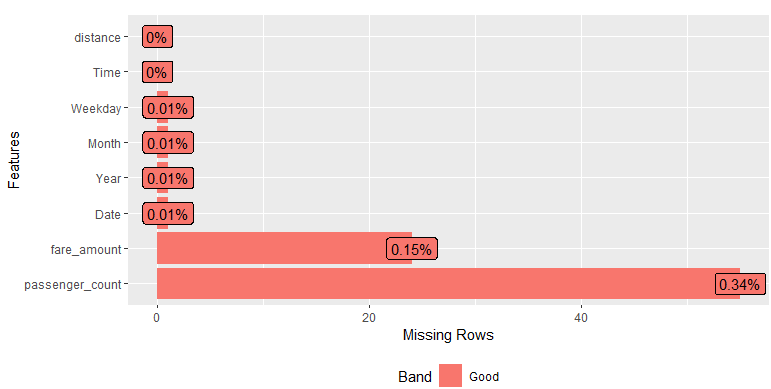


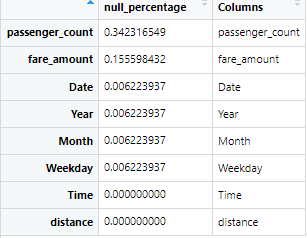
# Handling of the missing values and outliers

As we have seen earlier there are missing values in the data. Below we can see that most of the missing values are in variables fare\_amount and passenger\_count but the count is very small and insignificant.

* Missing values



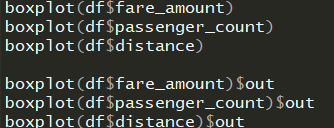




* From above we see that null values are very less in our data set i.e. less than 1%.
* Usually we do not remove the null value rows to avoid loss of data, but as the NA value columns are significantly small and dependent variable itself has missing values we can delete the rows having missing values.

Outlier detection and Treatment:

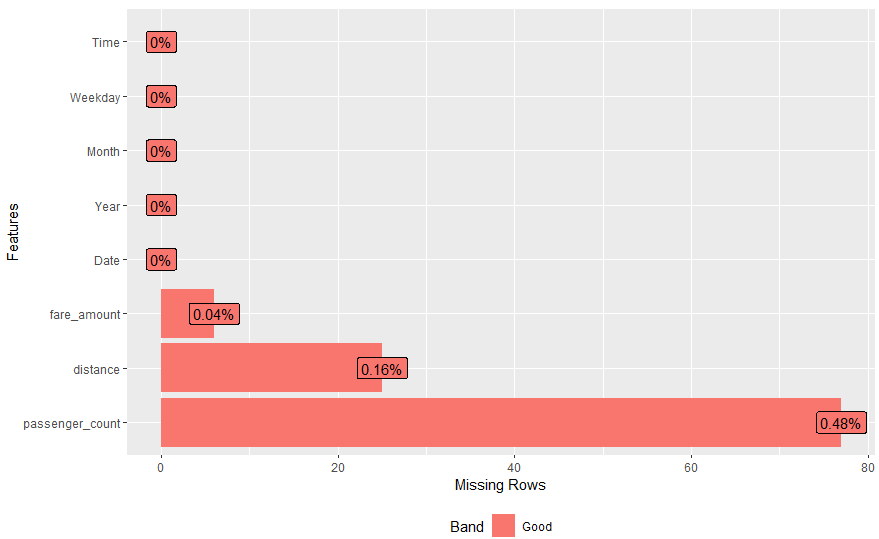
Using the boxplot outlier detection method as below:



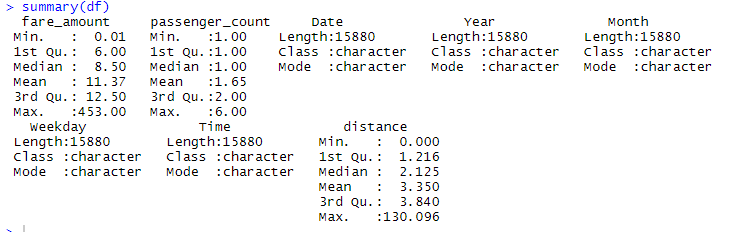
* We get that variables distance and fare\_amount has multiple outliers. **BUT** it is a good practise of consulting with the domain experts before treating those values as outliers and replacing them.
* Therefore based on self-assumptions we deduce following outliers for the variables :

1. Fare\_amount – Anything below 0 will be lower outlier and above 500 will be treated as higher outlier and replaced with NA values.
2. Passenger\_count – This being project regarding cab fare prediction we assume that passenger count above 6 is a higher outlier and replaced with NA values whereas lower outlier limit will be 1 passenger and min. 1 passenger will be required.
3. Distance – We take anything above 200 as outlier and replaced with NA values.

Now again we have NA values in the data but again they only constitute to less than 1 % of the data so we rather remove those rows having missing values.



# After we remove the outliers from the data variables we can see that the data makes much more sense:

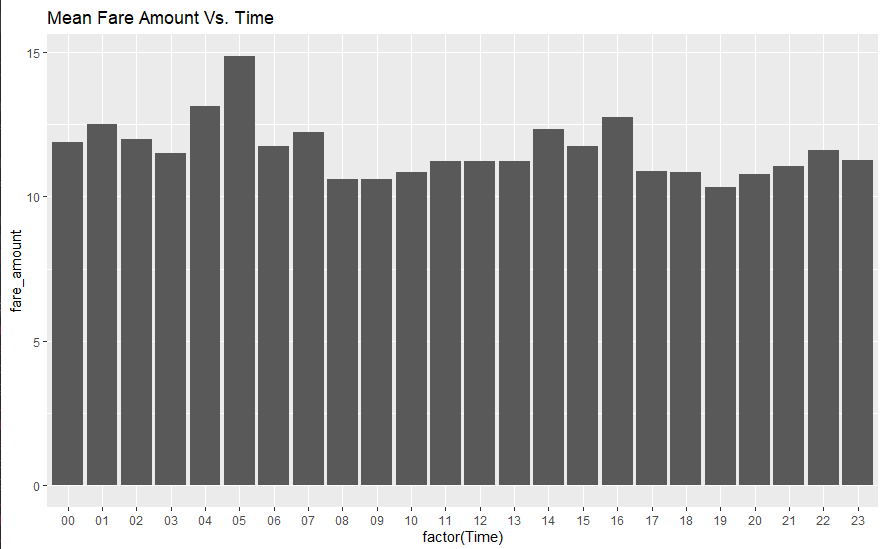


# Exploratory Data Analysis and correlation between variables

**Exploratory Data Analysis** (**EDA**) is the process of analysing and visualizing the data to get a better understanding of the data and glean insight from it.

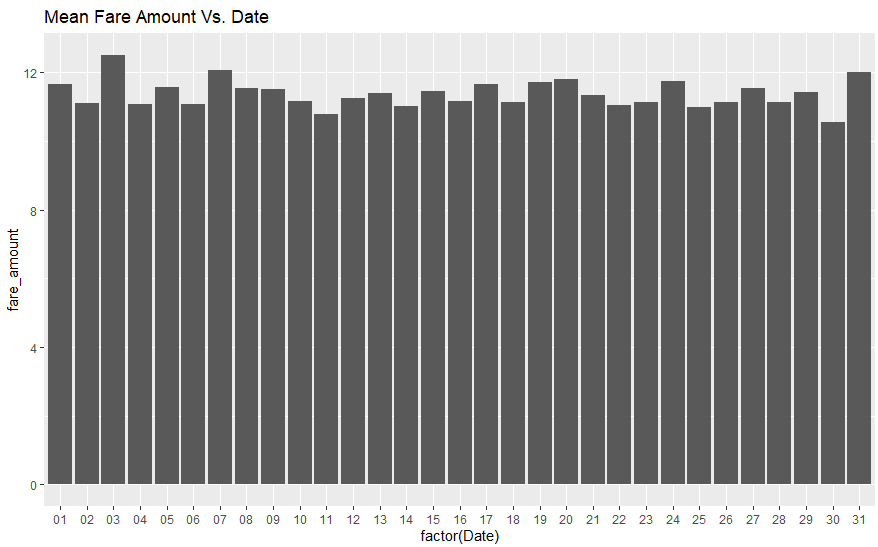
We use the bar graphs to plot relation between the dependent variable(fare\_amount) and the independent variables. Please note as we are analysing to predict for fare\_amount per ride, we plot the mean of fare\_amount with the independent variables and not count or sum as count and sum would not make sense:

1. Fare\_amount vs Time :



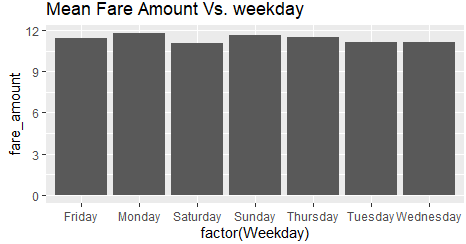
As we can see from above, Mean fare amount is usually higher in Hrs – 04,05,14,16,01 therefore we may include this variable for further analysis as of now.

1. Fare\_amount vs Date(day of month)



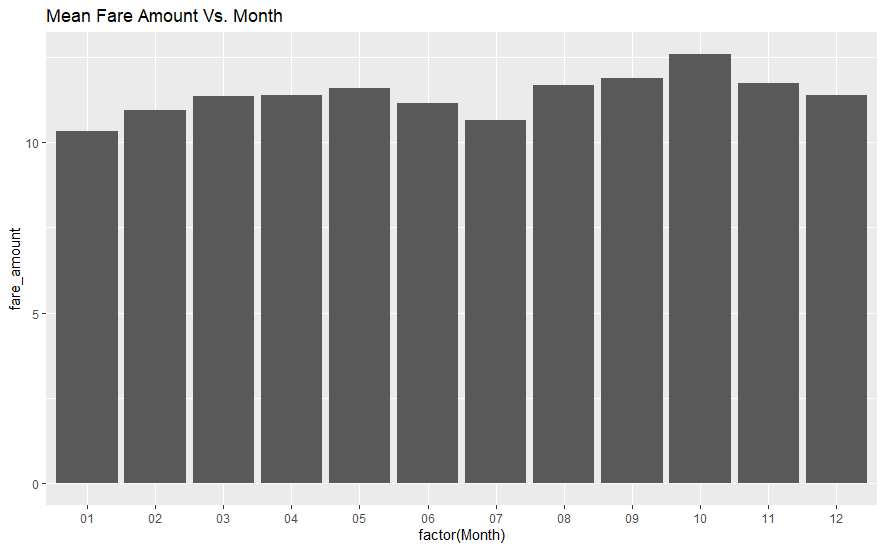
From above we can see that mean fare amount does not change with the date(day of month) as much and we may exclude this variable from further analysis.

1. Fare\_amount vs weekday :



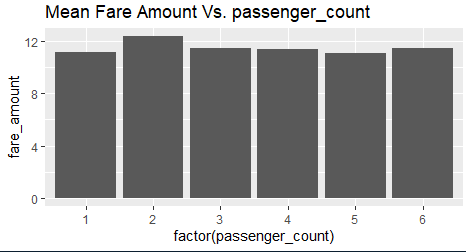
From above we can conclude that mean fare amount also does not change much with respect to the weekday variable and may be excluded from further analysis.

1. Fare\_amount vs Months :



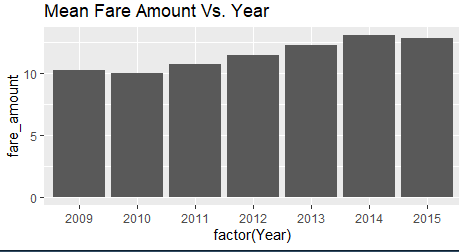
From above diagram and logically we can conclude that mean fare amount also does not change much with respect to the Month variable much and may be excluded from further analysis.

1. Fare\_amount vs passenger count:



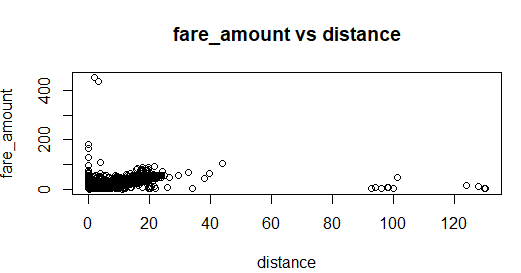
From above we can conclude that mean fare amount also does not change much with respect to the passenger\_count variable and may be excluded from further analysis.

1. Fare\_amount vs Year :



From above we can say that mean fare\_amount can be seen to gradually increase with each year and therefore we shall keep and use this variable(Year) in further analysis.

1. Fare\_amount vs distance:



Correlation between fare\_amount and distance:



From above we can conclude that mean fare amount is correlated with distance variable and should be used in further analysis.

Based on the above we conclude that significant variables for fare\_amount calculation are – distance , Year and Time and we shall exclude variables – Weekday, Month, Date(day of month) and passenger count from further analysis as they do not seem to affect the mean fare\_amount i.e dependent variable much.

# Creating dummy variables and splitting into training and test Set.

At this stage we convert the resulting dataset (df) into training and test set slitting in ratio of 75% : 25% in train and test set respectively.

set.seed(101)

sample = sample.split(df1$fare\_amount, SplitRatio = .75)

train = subset(df1, sample == TRUE)

test = subset(df1, sample == FALSE)

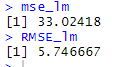
# Model Selection

The dependent variable is a continuous variable and hence we will use regression models to find the best fit and predict the results based on that.

We try the following regression models on the training data and:

1. Multiple Linear Regression

lrmodel <- lm(fare\_amount ~ ., data = train)

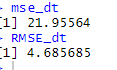


1. Decision Tree Regression

fit = rpart(fare\_amount ~ ., data = train,

control=rpart.control(minsplit=25))

parameter tuned = minsplit : It is used for Minimum number of observations for a node to be considered for a split. We have found the optimum value for this to be as 25 as post this the result do not seem to be changing much.

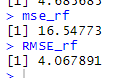


1. Random Forest Regression

regressor = randomForest(x = train[,-1], y = train$fare\_amount, ntree = 1000, importance = TRUE)

parameter tuned = ntree – Value started testing from 100 , 200, 500, 800 to 1000

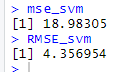
1000 seem to be optimum enough as the result did not change much post that.



1. Support Vector Machine Regression

regressor = svm(formula = fare\_amount ~ ., data = train, kernel = 'radial', cost = 10)

Parameters tuned: Kernel selected is radial



1. Gradient Boosting

gbm.fit <- gbm(

formula = fare\_amount ~ .,

distribution = 'gaussian',

data = train,

n.trees = 10000,

interaction.depth = 1,

shrinkage = 0.01,

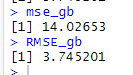
cv.folds = 5,

n.cores = NULL, # will use all cores by default

verbose = FALSE

)

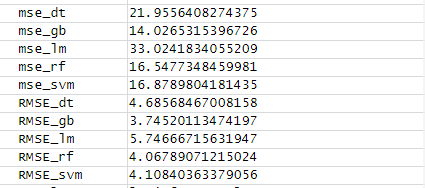
Parameters tuned n.trees = 10000, shrinkage = 0.01, distribution = ‘gaussian’



# Performance Evaluation

We have used 2 performance evaluation metrics to verify the resulting predictions:

1. MSE: 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.
2. RMSE: 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.
3. Below are the MSE and RMSE of each of the 5 models



As we can see from the above results that the best performing models seems to be the gradient boosting regression

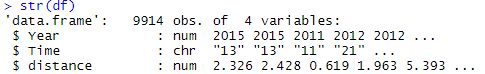
MSE\_gb = 14.02

RSME\_gb = 3.745  
Based on above testing we understand that Gradient boosting regression is the best fit for the data, It has given the lowest MSE and RMSE among all the 5 regression models.

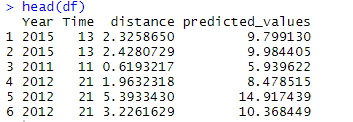
# Using the best model to predict fare\_amount for test data

We already have gradient boosted trained model which we find to be the best model. We use this model to then predict the values for the data in Test File provided for the project.

To predict the values, we have to first convert the location points into distance and source out variable Year and Time from the pickup\_DateTime and correct the data types of the columns as below:



df$predicted\_values <- predict.gbm(gbm.fit, df, n.trees = 1000)



**Conclusion**

As seen above the column predicted\_values is the resulting predicted values.

The results have been exported into file named Result.csv attached.