CAB FARE PREDICTION USING PYTHON

Ayush Kumar Jain

CONTENT

1. PROBLEM STATEMENT

The objective of this project is to predict Cab Fare amount. 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.

ABOUT DATA:

TRAIN DATA: Train data is Historical data for which we know the fare_amount. Train data carries total of 16067 observation with attributes pickup_datetime, pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude, passenger_count and fare_amount.

TEST DATA: Test data is the new data for which we have to predict fare_amount value by building model using train data set. Test data carries total 9914 observation with attributes pickup_datetime, pickup_latiutde, pickup_longitude, dropoff_latitude, dropoff_longitude, passenger_count.

About Attributes:

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 value indicating the number of passengers in the cab ride.
fare_amount – float for fare of the trip. This attribute is to be predicted for the test case

2. DATA PRE-PROCESSING

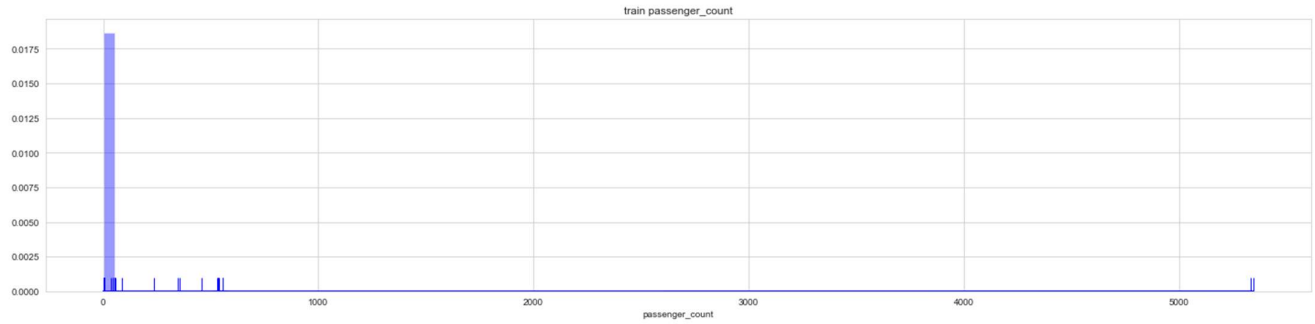
In this step we try to understand the giving data by plotting some visualizations and by using some functions of Python. If the data is messy, we will clean it by removing observation or by treating them as missing values, this step is known as Exploratory Data Analysis.

After cleaning the insensible observations, we will go for missing value. After dealing with missing values we will plots some visualization to understand our data more clearly.

2.1 Dealing with Insensible data or observation:

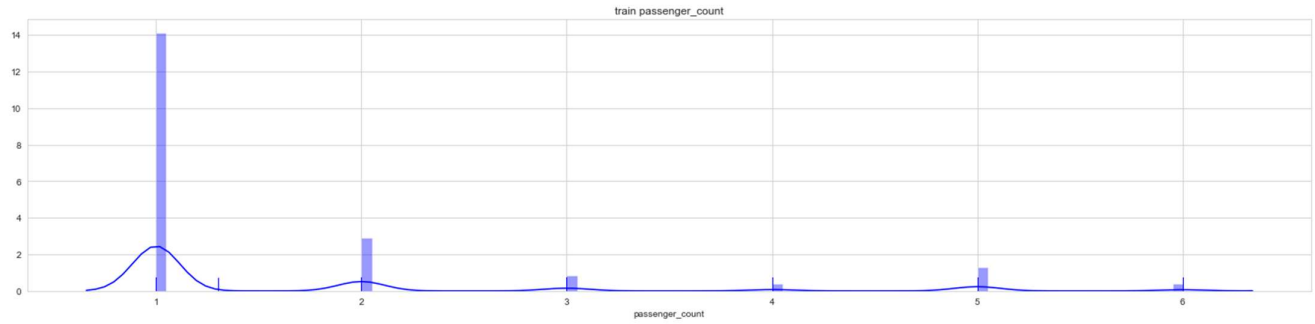
1) Passenger_count:

The passenger_count variable values should be a positive integer. As we can see from our test data that maximum number of passengers is 6. That mean any value which is non integer, negative and more than 6, does not make any sense. So, to understand the passenger_count variable lets observe the its distribution plot as shown below.



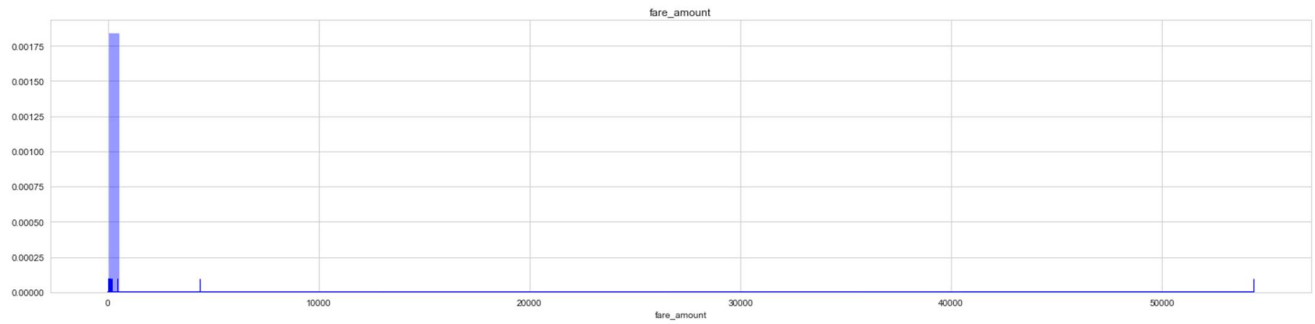
As from above plot we can see that the passenger_count in train data set contains some negative values, non-integer value and also some values greater than 6.

On using some function in python, we found out total 78 observations with negative values and values greater than 6. Now what we can do that we can replace these observations as NA than treat them as missing values or we can remove these observations. We go for both methods one by one than select the method for which our model gives better results. So, after doing this lets again take a look at the new plot



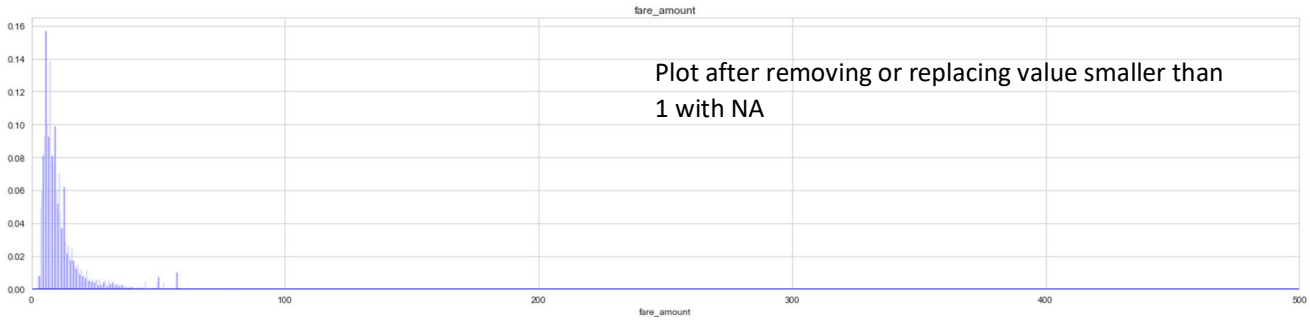
There are some non-integer values between 1 to 6 so we round up the values of passenger_count Also, after doing this step we will convert passenger_count data type to a categorical type data type that is “object” in python.

2) Fare_amount:



Now, fare_amount is a numeric value that indicates the fare for the trip. This value cannot be a negative value. So, let's plot some visualization to see how fare_amount is distributed

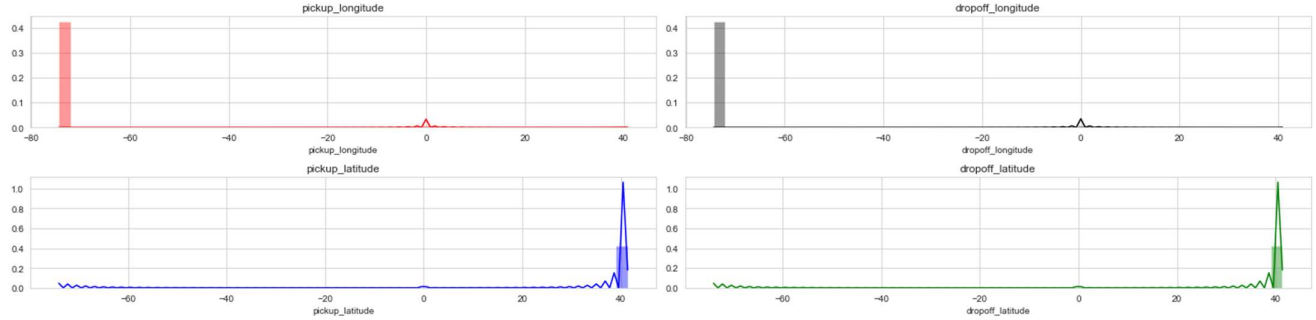
As we can see from above plot fare amount have some negative values, also fare_amount have some abnormally high values. Most of the value for fare_amount is less than 60 and some which are greater than 60 can be possible depending on the other data. On using some functions in Python, we find out there are only 4 values with fare_amount greater than 200(taken as maximum value for fare_amount) and 5 value which are less than 1. Now what we can do that we can replace these observations as NA than treat them as missing values or we can remove these observations. We go for both methods one by one than select the method for which our model gives better results.



Plot after removing or replacing value smaller than 1 with NA

3) Latitude and Longitude Parameters:

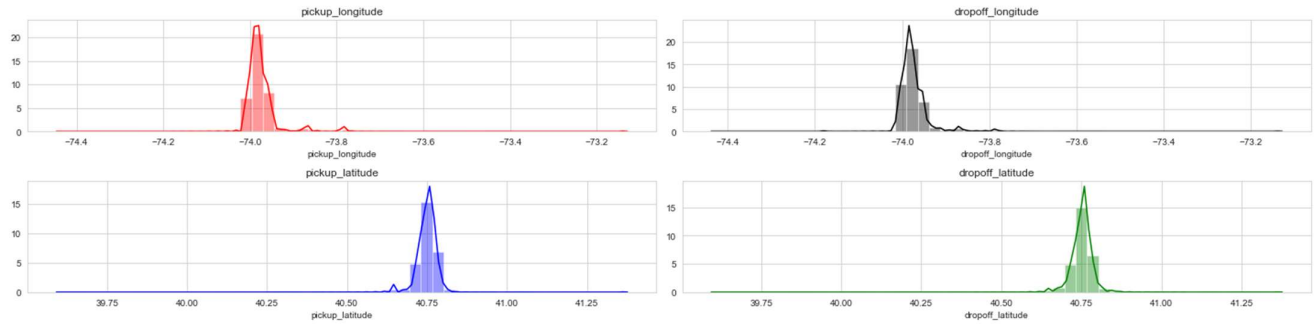
In our data we have pickup_latitude and pickup_longitude which tell us about our pickup location. And also, we have dropoff_latitude and dropoff_longitude which tells us about our drop-off location. Now from our general understanding of latitude and longitude we know that latitude can be between -90 and 90 and longitude can be between -180 and 180. Values out of these values are insensible values, so let's see the summary of these variable using function df["variable"].describe()
From using the function, we found out that the values are lies under the sensible range for all except pickup_latitude as its maximum value is 401 which is not making any sense. So, let's find out how many values are greater than 90.
On using python, we find out data have only one such value.
Now let's plot some visualizations to see distribution of these parameters



we can see from above most of latitude values lies near to 40 and longitude values lies near to -72. From latlong.net we find out the given data is for New York city. So, by taking range for latitude as (39,42) and longitude range as (-72 to -75) we found out

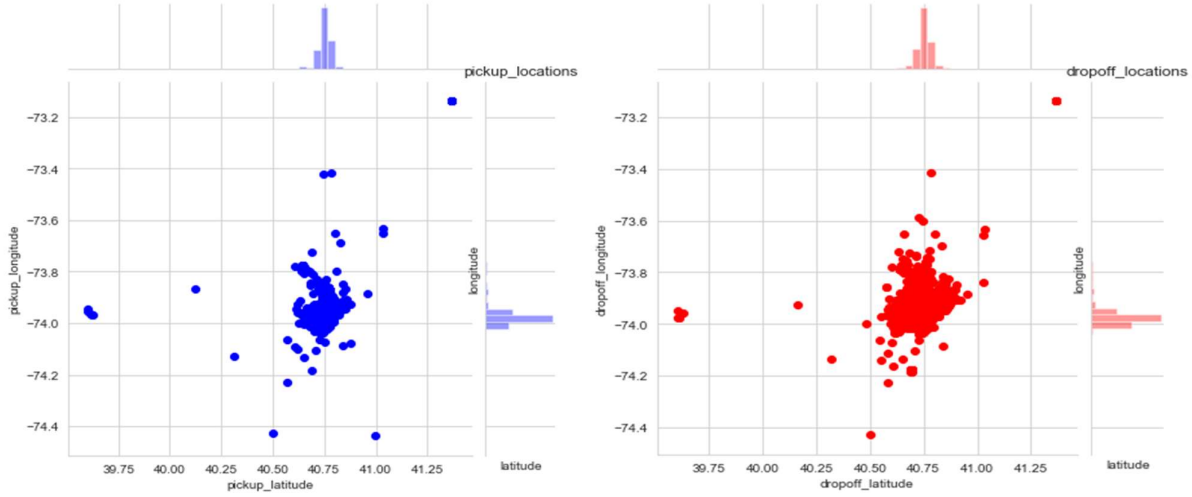
total 337(if we have replaced insensible values for passenger_count and fare_amount as NA, it is 333 when we remove these observations) point lies away from New York. So, with these 338 what we can do is we can drop these or treat them as missing value. We do these methods one by one and select the one with better results.

After removing all these observations, we again see the distribution from below plots



As we can see all the values of latitude and longitudes are in range, distribution for both longitudes are left skewed and for both latitudes it is right skewed.

Let's also observe the distribution of pickup location and drop-off location by plotting a scattered plot.



As we can see most of the values are concentrated between latitude value (40.5,41) and longitude values (-74.3, -73.7).

4) Pickup_datetime:

This data contains Date and time at which the trip stars, we can use this attribute to create new attributes lies year, month, day, hour to find out how our fare_amount is varying with time and date. We converted this in date-time data type.

2.2 Missing value Analysis:

As the name indicate we have analysed the values which are missing in our data.

Let's find out number of missing values in every column by using Python.

Number of missing values when we remove all insensible as discussed above:

Variable	Number of missing values	Missing percentage
pickup_latitude	0	0
pickup_longitude	0	0
dropoff_latitude	0	0
dropoff_longitude	0	0
Passenger_count	55	0.3515050
Fare_amount	22	0.14060203
Pickup_datetime	0	0.000000

What we can do is we can drop these observations or we can impute them using different missing value imputation methods.

Number of missing values when we impute insensible data as NA.

Variable	Number of missing values	Missing percentage
pickup_latitude	325	2.0227796
pickup_longitude	324	2.0165557
dropoff_latitude	324	2.0165557
dropoff_longitude	322	2.0041078
Passenger_count	133	0.8277837
Fare_amount	34	0.2116139
Pickup_datetime	0.00	0.000000

We will impute these missing values using different missing value imputation methods.

<u>Missing value imputation:</u>

1) passenger_count:
   As we discussed above passenger count can only take integer value between 1 to 6, so it is a categorical type or factor type variable. Hence for imputing missing values for passenger_count have 2 methods: a) we can impute missing value with mode and b) using KNN.
   To find out which method is better we create a missing value in passenger_count at some random observation say 1000, than we impute the value using both methods, than we compare new value with actual value.
   #Actual value for passenger_count at location 1000 is 1
   #Mode for passenger_count is 1
   #When we impute this value using KNN we also get 1 at this observation.
   Now, we cannot use the mode method it will bias our data towards passenger_count value 1. Therefore, we use KNN.
2) fare_amount:

As we see the fare_amount is numeric type data, we can impute missing value for fare_amount with its mean or with its median or we can impute it using KNN method. To find out which method is better we create a missing value in fare_amount at some random observation say 1000, than we impute the value using both methods, than we compare new value with actual value.
Actual Value of fare_amount at observation 1000= 7,
Mean of fare_amount= 11.31
Median of fare_amount = 8.5
When we impute using KNN=7.46
As we can see KNN giving better results, hence we use KNN method for imputing missing values for fare_amount.

3) Latitude and longitude variable:
As we see these data is numeric type data, we can impute missing value with its mean or with its median or we can impute it using KNN method. To find out which method is better we create a missing value at some random observation say 1000, than we impute the value using both methods, than we compare new value with actual value.

	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude
Actual value	-73.9954	40.7597	-73.9876	40.7512
mean	-73.9748	40.7509	-73.9738	40.7514
median	-73.9820	40.7533	-73.9806	40.7542
KNN	-73.9945	40.7465	-73.9882	40.7451

we use Mean and median for imputing dropoff_latitude and pickup_latitude, and KNN for all other variable.

After imputing all insensible data as missing values and missing value we are here with 16067 observations and in other case when we have dropped all insensible data and missing value, we end here with 15569 observations.

NOTE: All plots below are for case when we dropped insensible data and missing values.

2.3 Feature Engineering:

In this step we create new feature with the help of existing feature

1) trip_distance:

As we know our pickup latitudes and longitude & drop-off latitude and longitude, we can find the distance between these two points by using Haversine method which is describe below (this is the distance between two point on the surface of sphere):

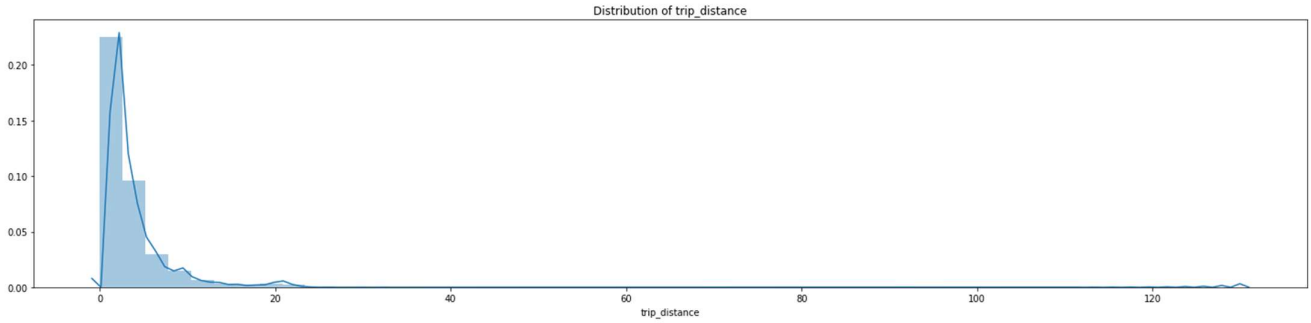
The word "Haversine" comes from the function: haversine(θ) = $\sin^2(\theta/2)$

The haversine formula is a very accurate way of computing distances between two points on the surface of a sphere using the latitude and longitude of the two points. The

following equation where ϕ is latitude, λ is longitude, R is earth's radius (mean radius = 6,378km) is how we translate the above formula to include latitude and longitude coordinates. Note that angles need to be in radians to pass to trig functions:

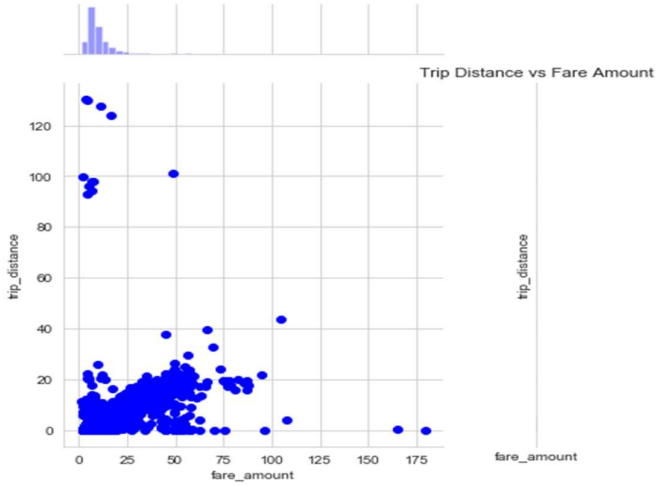
$a = \sin^2(\phi B - \phi A/2) + \cos \phi A * \cos \phi B * \sin^2(\lambda B - \lambda A/2)$
$c = 2 * \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$
$d = R \cdot c$

where A, B are two points and all latitude and longitudes points are in radian. So, after creating this new feature let's take a look at its distribution:



As we can see the distribution is left skewed, also for some observation trip distance is 0. This may be because of a round trip. It is also possible that it is because of cab booking cancelation and fare_amount value at that time is cancelation penalty. Also, on observing trip_distance in test data, we found out it also have 155 observations for which trip_distance is 0. So, we have to considered these observations in train data.

Let's see how fare_amount is varying with trip_distance



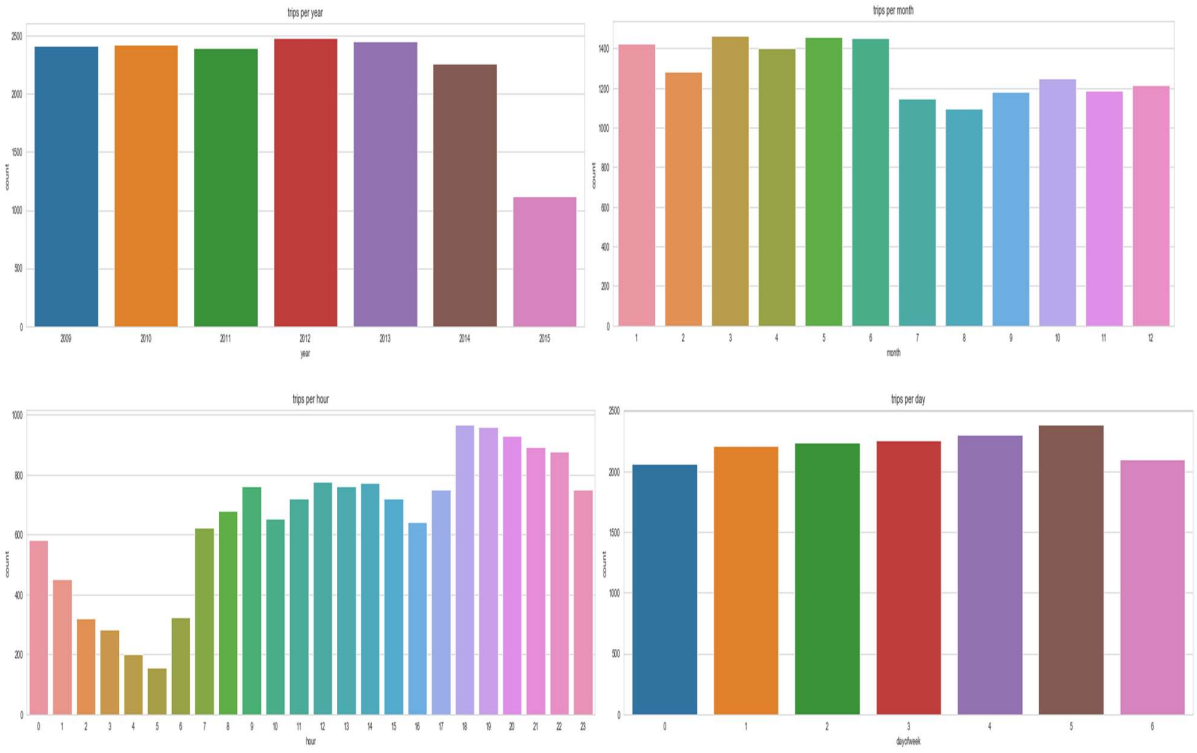
So, as we can see the plot between fare_amount and trip_distance almost following linear pattern that when trip_distance is increasing fare_amount is also increasing.

2) Year, month, dayofweek and hour:

As we see in our day to day life, a taxi fare can vary because of climate/season, can be different on weekdays and weekends, also be different on same day at peak traffic hours or normal traffic hours. So, in this case to find more about these type of variation in

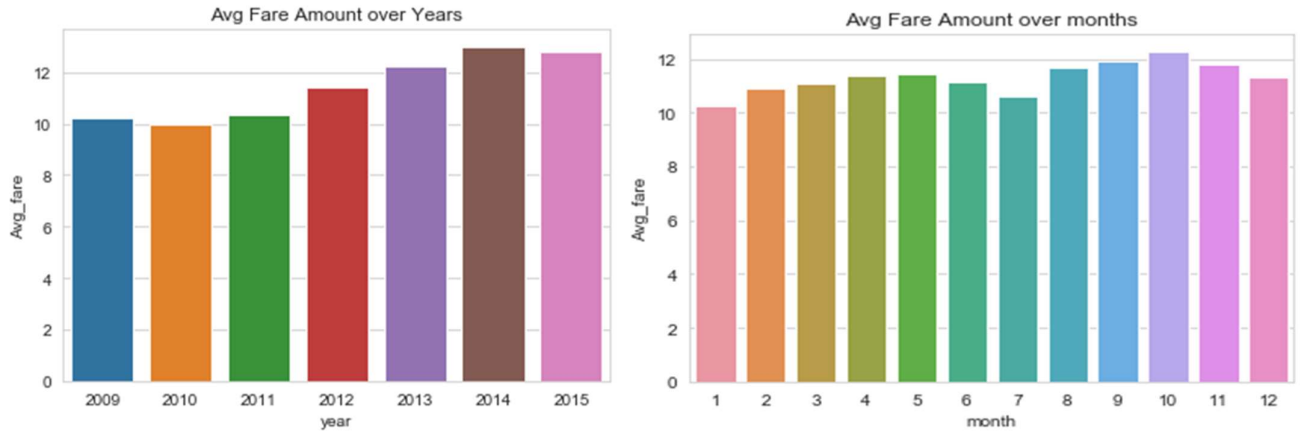
fare_amount let's create new features as Trip year as year, Trip month as month, Trip day of week as dayofweek and pickup time as hour using pickup_datetime. Now on checking for missing value in these features we found out they have each one missing value for same observation, we will drop this value.

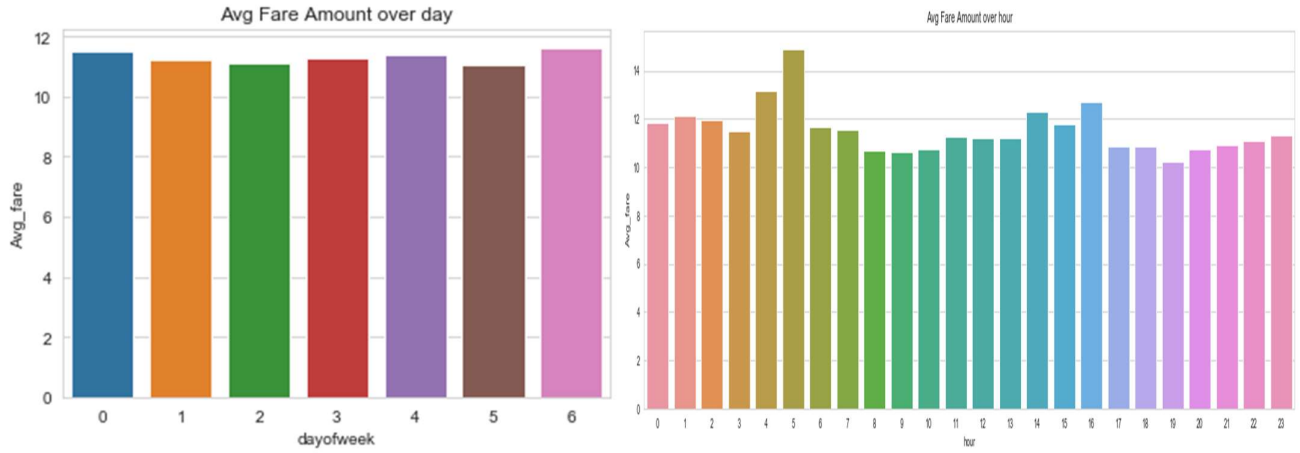
After creating let's see how number of trips varying with these features:
As we can see from plots that we have less data for 2015 year, also we can see that from month of January (01) to June (06) there are more trips as compare to other month.



Also, from 12:00 AM to 06:00 AM number of trips are very less and from 6:00 PM to 9:00 PM trips are more. Similarly, we can also observe that number of trips on are almost same on every day except for Sunday (0) and Saturday (6).

Let's see how average fare_amount varying with these features

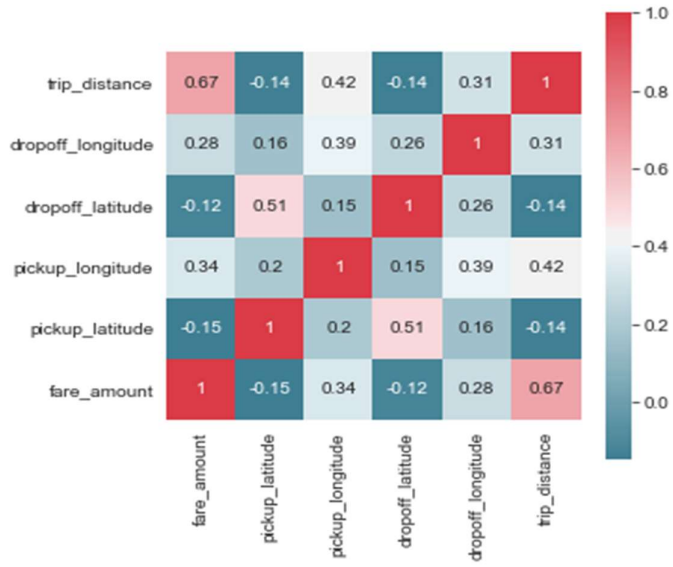




2.4 Feature selection:

In this step we select the valid feature that is going to drive our model. So, for doing so we first plot correlation plot for Numeric variables, then we do chi-square test between out categorical variable, then ANOVA test to find out effect of categorical variable on our numeric target variable.

Correlation plot between numeric variables:



No independent variable is highly correlated to other.

Chi-Square test between categorical variables:

Chi-square test compares 2 categorical variables in a contingency table to see if they are related or not. Assumption for chi-square test: Dependency between Independent variable and dependent variable should be high and there should be no dependency among independent variables.

Null Hypothesis: two variables is independent.
Alternative hypothesis: two variables are not independent.
If p-value from chi-square test comes lower than 0.05 we accept the null hypothesis, otherwise we reject the null hypothesis by saying they are not independent.

From chi-square test we found out (passenger_count and month), (year and hour), (year and dayofweek), (month and hour) and (month and dayofweek) have p-value greater than 0.05. we will only drop those variables which have very low impact on our target variable. So, to find out let's do ANOVA test.

ANOVA Test for independent categorical variable and numeric target variable:

This test helps us to find out the mean of target variable is different for different values of categorical variable or not.

Hypothesis:

Null hypothesis: mean for target variable is same for all values of categorical variable.
Alternative hypothesis: mean for target variable is different for different values of categorical variable.

If p-value from ANOVA test is higher than 0.05 we accept null hypothesis, otherwise reject it. From analysing the results from ANOVA test, shown below, we found out dayofweek does not affect our fare_amount mean **so we will drop this feature (starts in the end indicate importance of variable).**

	Df	sum_sq	mean_sq	F	PR(>F)
C(passenger_count)	5.0	2.099163e+03	419.832699	4.648301	3.055335e-04
C(year)	6.0	2.095049e+04	3491.748352	38.659920	6.697679e-47
C(dayofweek)	6.0	6.181555e+02	103.025914	1.140682	3.355667e-01
C(month)	11.0	5.892329e+03	535.666276	5.930787	1.022714e-09
C(hour)	23.0	8.239487e+03	358.238581	3.966344	4.789158e-10
Residual	15516.0	1.401399e+06	90.319597	NaN	NaN

2.5 Feature Scaling:

In this step we convert the value of independent numeric variables in such a way that they are comparable. For doing so we have two methods: Normalization and standardization.

We can standardization only for those variables for which the distribution is normal distribution. In our case all 5 variables doesn't have normal distribution, so we do feature scaling for all 5 variables i.e. pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude and trip distance using normalization method.

New scaled/Normalized value= $\frac{\text{(original value – minimum value)}}{\text{(maximum value – minimum value)}}$

So, after doing all these pre-processing steps we ended here with total 16066 observations and 10 variables (when we replace insensible data as NA and impute them) or with 15569 observations and 10 variables (when we drop every insensible values and missing values) in our train data. Now let's move towards model building process.

3. DIVIDING DATA:

In this step we divide our data into two parts with 80% data as training set and 20 % data as validation set.

From training set we train our model for predicting the value of target variable and from validation data set we check how accurate our model is.

We divided our train data as x_train ,y_train(training set) and x_test ,y_test(validation set).

4. MODEL DEVELOPMENT:

In this step we develop some models using liner regression, decision tree and random forest algorithms. Then we compare our different models using some Error matrix, than we select the best fitted model and use it to predict our fare_amount values for test data.

4.1 Liner Regression:

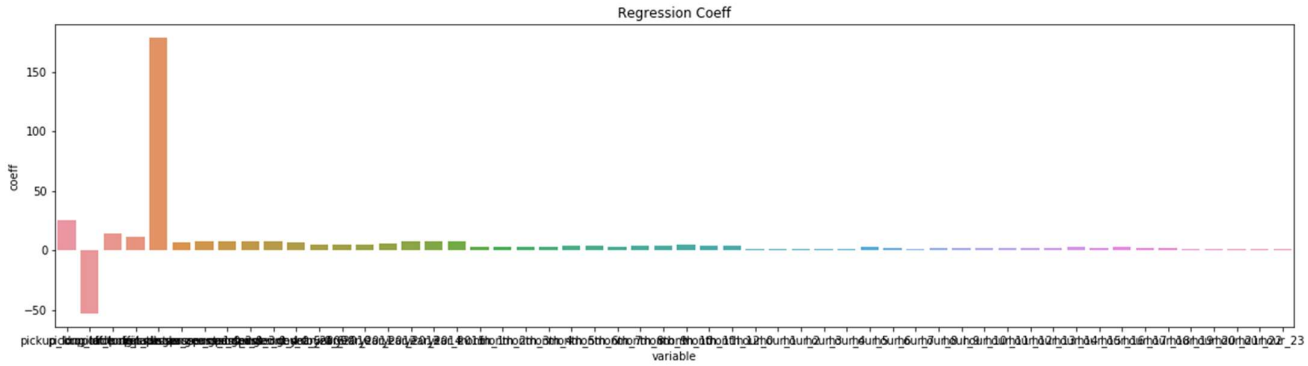
Before building a model we first create dummy for each value in all categorical variables because as we know linear regression work on an equation, so for e.g.: for value of year 2012 and 2015 instead of taking them as category it takes it as numeric values.

Multicollinearity check– In regression, "multicollinearity" refers to predictors that are correlated with other predictors. Multicollinearity occurs when your model includes multiple factors that are correlated not just to your response variable, but also to each other. For checking we find out VIF (Variance inflation factor) that is measure of multicollinearity in set of multiple regression variables. VIF is always greater or equal to 1.
if VIF is 1, Not correlated to any of the variables.
if VIF is between 1-5, Moderately correlated.
if VIF is above 5, Highly correlated.
If there are multiple variables with VIF greater than 5, only remove the variable with the highest VIF.
We have checked for multicollinearity in our Dataset and all VIF values are below 5.
Model summary:

Regression Results

Dep. Variable:	fare_amount	**R-squared:**	0.515
Model:	OLS	**Adj. R-squared:**	0.513
Method:	Least Squares	**F-statistic:**	263.1
Date:	Wed, 08 Jan 2020	**Prob (F-statistic):**	0.00
Time:	12:24:34	**Log-Likelihood:**	-41302.
No. Observations:	12454	**AIC:**	8.271e+04
Df Residuals:	12403	**BIC:**	8.309e+04
Df Model:	50		
Covariance Type:	nonrobust		

Let's see histogram for coefficients of all the variables



trip_distance have highest coefficient.

Error matrix:

When we Dropped all insensible data		When we Replace it with NA and imputed with missing values	
MAPE	:35.72	MAPE	:37.75
MSE	:61.44	MSE	:56.62
RMSE	:7.83	RMSE	:7.52

4.2 Decision Tree:

Error matrix for model on Decision tree

When we Dropped all insensible data		When we Replace it with NA and imputed with missing values	
r square	:0.693	r square	:0.613
Adjusted r square	:0.687	Adjusted r square	:0.606
MAPE	:27.92	MAPE	:28.79
MSE	:24.11	MSE	:38.10
RMSE	:4.91	RMSE	:6.17

4.3 Random Forest:

Error matrix for model on Random forest

When we Dropped all insensible data		When we Replace it with NA and imputed with missing values	
r square	:0.807	r square	:0.712
Adjusted r square	:0.807	Adjusted r square	:0.711

MAPE	:22.21	MAPE	:22.66
MSE	:15.12	MSE	:28.32
RMSE	:3.88	RMSE	:5.32

4.4 KNN: Error matrix for model on Random forest

When we Dropped all insensible data		When we Replace it with NA and imputed with missing values	
r square	:0.0513	r square	:0.0130
Adjusted r square	:0.0485	Adjusted r square	:0.0102
MAPE	:47.908	MAPE	:49.682
MSE	:90.651	MSE	:97.214
RMSE	:9.521	RMSE	:9.859

5. MODEL SELECTION AND PREDICTION

As this is a regression model the R-square value plays important role here.

R-square value is indication of how much variance of target variable is explained by our independent variable. For a good model this R-square should be equal or greater than 0.80.

In case of Linear regression, the value of R-square we get is only 0.515 and value of mean absolute percentage error is from 35 to 37 and Root mean square error is from 7.5 to 7.8

In case of Decision tree, the value of R-square we get is 0.69 and value of mean absolute percentage error is from 27.9 to 28.79 and Root mean square error is from 4.91 to 6.17

In case of Random forest, the value of R-square we get is only 0.80 and value of mean absolute percentage error is from 22.21 to 22.66 and Root mean square error is from 5.32 to 3.88.

From above, we can see that from all the model we build we get maximum value for R-square i.e. 0.80 from Random forest model which we build on data that we get when we drop all the insensible data and missing values imputation method. Also, we can see that from this model we get minimum value for RMSE i.e. 3.88.

Hence, we select the random forest model that we built after dropping insensible data and missing value to predict our test cases.