CAB FARE PREDICTION

Nikhila K.G

nikhila.guruprasad@gmail.com

CONTENTS

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.

2. DATA SETS

1) train_cab.zip
2) test.zip

3. NUMBER OF 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 indicating the number of passengers in the cab ride.

4. BASIC ANALYTICS

- Install / import the required packages.
- Load train and test datasets in R / Python environment.
- To have a quick look into both the datasets.

5. DATA PRE-PROCESSING

This stage generally involves:

- Exploratory Data Analysis
- Missing value Analysis
- Outlier Analysis
- Feature engineering
- Feature selection
- Feature scaling

5.1 EXPLORATORY DATA ANALYSIS

➢ **fare_amount**

- Change from factor to numeric
- Remove all 0 and negative values: (5 values)

➢ **passenger_count**

- It shouldn’t exceed more than 6 passengers: (20 values)
- It shouldn’t be less than 1 passenger: (58 values)

➢ **pickup_latitude:** (latitude range = +90 to -90)

- pickup_latitude above 90: (1 value)
- pickup_latitude below -90: (0 value)
- pickup_latitude equal to 0: (311 values)

➢ **dropoff_latitude:**

- dropoff_latitude above 90: (0 value)
- dropoff_latitude below -90: (0 value)
- dropoff_latitude equal to 0: (9 values)

- ➢ **pickup_longitude:** (longitude range = +180 to -180)

- pickup_longitude above 180: (0 value)
- pickup_longitude below -180: (0 value)
- pickup_longitude equal to 0: (0 value)

- ➢ **dropoff_longitude**

- dropoff_longitude above 180: (0 value)
- dropoff_longitude below -180: (0 value)
- dropoff_longitude equal to 0: (2 values)

5.2 MISSING VALUE ANALYSIS

Missing value is any data value missing in given dataset which may be due to human error, refuse to answer while surveying or optional box in questionnaire.

We need to understand why each value is missing because:

- ➢ In some cases, more than 30% of the data are missing. In such cases we delete/ignore observation or variable where we do not intend to impute a value.

- ➢ In other cases, we consider to impute whose missing value is less than 30%.

Imputing the missing values can be done by:

- ➢ Central statistical method: by finding mean, median for the variable.
  In case if the variable is categorical in nature: mode

- ➢ Distance method: also called KNN (k nearest neighbour)
  We find out the nearest neighbour based on existing attributes by Euclidean distance.

Selection of best method:

- creating a small subset of data with complete observation.
- Deleting some values manually.
- Use mean, median, KNN or mode(categorical) methods to fill.
- See where they are filling with nearest value to the actual value.
- Choose the best method.

Steps:

- Creating dataframe with sum of missing values:
  fare_amount: (22 values)
  passenger_count: (55 values)

- Converting row index into column.

- Rename the column as **missing_percentage.**

- Calculating in terms of percentage
  fare_amount: (0.14%)
  passenger_count: (0.35%)

- Arranging in descending order.

- Rearranging the columns

- Imputation

  train[1887, 7]: (passenger_count)
  actual value = 1
  mean = 1.65
  KNN = 1.63 (R Language)

Conclusion: We will choose the KNN method with k = 789 in R

We will choose the mean method in Python

We will not use Mode method because whole variable will be more biased towards 1 passenger_count also passenger_count has maximum value equals to 1

train [6879, 1]: (fare_amount)
actual value = 6.5
mean = 15.11
median = 8.5
KNN = 7.73 (R language)

Conclusion: We will Choose KNN method here because it imputes value closest to actual value with k = 789.

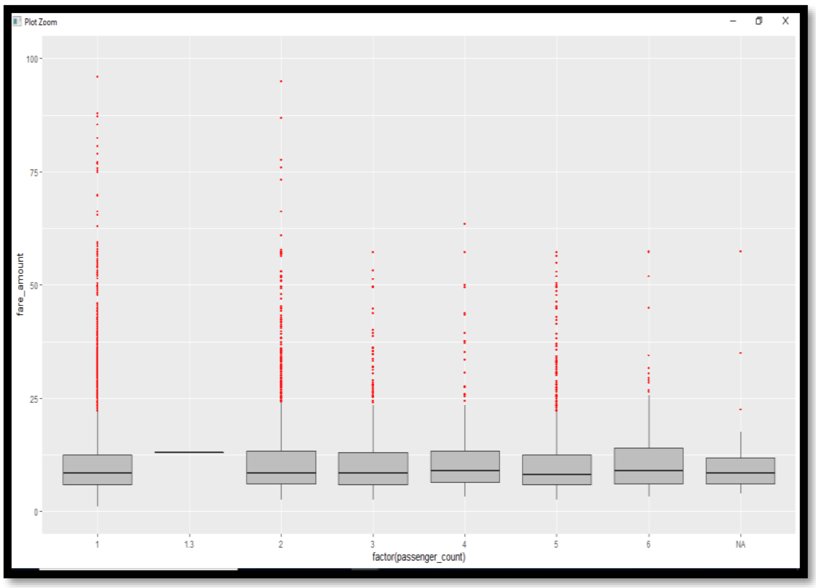
We will choose the mean method in Python

5.3 OUTLIER ANALYSIS

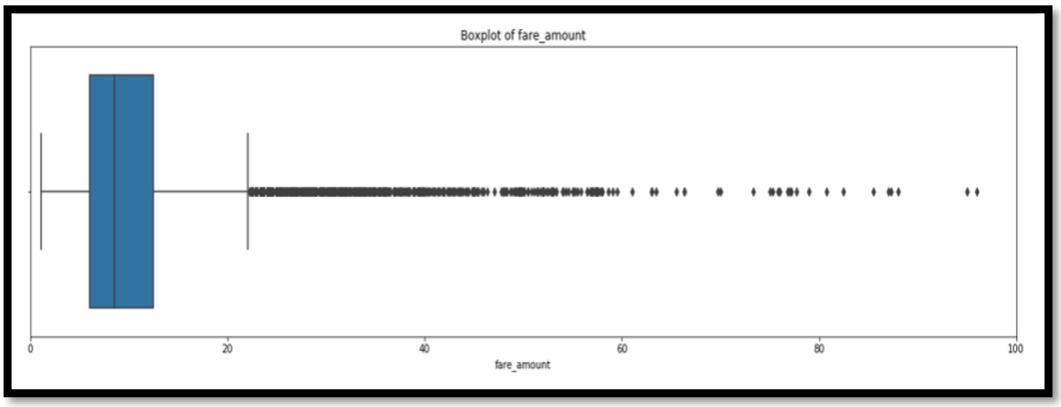
Outliers are the observations inconsistent with rest of dataset. This can be detected by Box plot.
A Box Plot is a convenient way of visually displaying the data distribution through their quartiles. It is a graphical representation of statistical data based on the minimum, first quartile, median, third quartile, and maximum.

fare_amount
Steps:

- We look for outlier in the dataset by plotting Boxplots.



R Language: boxplot for fare_amount



Python Language: boxplot for fare_amount

- Replace all outliers with NA.
- sum of NA's: (1361)
- KNN imputation with k value = 3 (R language)
- Median imputation(Python Language)

5.4 FEATURE ENGINNERING

Feature engineering efforts mainly have two goals:

- Preparing the proper input dataset, compatible with the machine learning algorithm requirements.
- Improving the performance of machine learning models.

- **pickup_datetime** variable from both test and train dataset

We will use this timestamp variable to create new variables.

- **pickup_date**: To obtain only date from **pickup_datetime**
- **pickup_weekday/ day_of_week**: will contain only week from pickup_date.
  For ex. 1 which is for Monday,2 for Tuesday, etc. (1 to 7)
- **pickup_mnth/month:** will contain only months from pickup_date. For ex. 1 for January, 2 for February, etc. (1 to 12)

- **pickup_yr/year:** will contain only years from pickup_date. For ex. 2009, 2010, 2011, etc. (2009 to 2015)

- **pickup_hour/hour:** will contain only hours from pickup_datetime. For ex. 1, 2, 3, etc. (0 to 23)

- Removing **pickup_datetime** and **pickup_date** variable from train and test dataset.

- Check for missing values(na): (4 values or 1 row)

- Omit the na values

- To Calculate the distance travelled using longitude and latitude

  - To calculate the distance travelled using longitude and latitude using from geopy library (Python Language)

  - To convert degree into rad: 1**Deg** × π/180

  - Distance uses the 'haversine' formula to calculate the great-circle distance between two points – that is, the shortest distance over the earth's surface (R language)

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

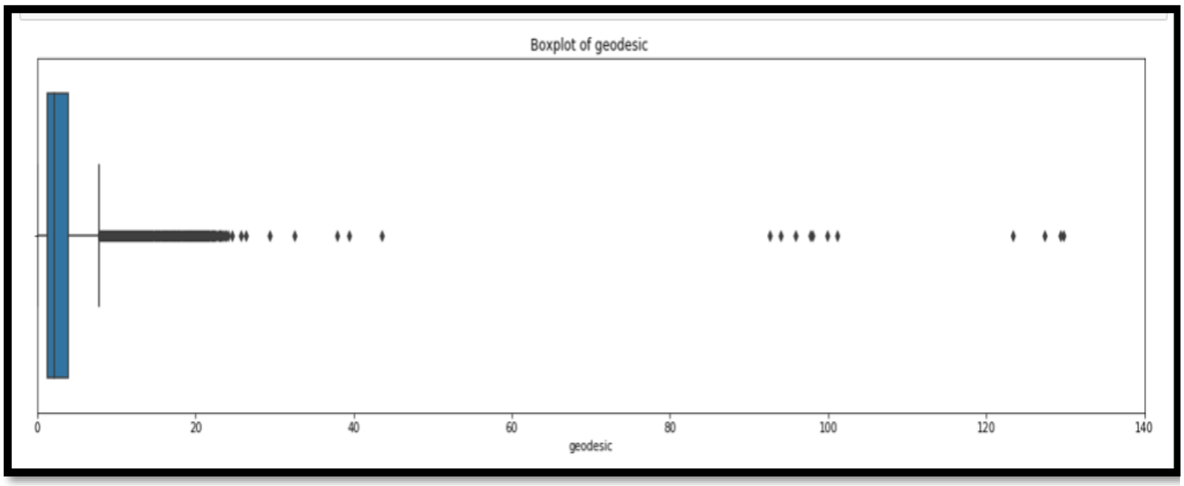
$$d = R \cdot c$$

φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km) R is in meters

- Creating a new variable called **dist** to store all the continuous values from **pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude** from both test and train datasets

- Removing the variables **pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude.**

- Checking for outliers for **dist/ geodesic** variable. (1348 values)

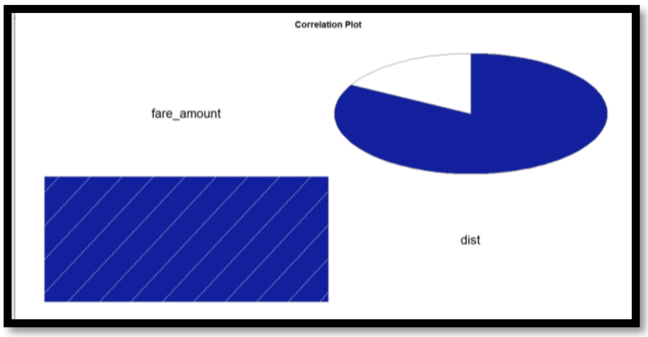


- Imputing with KNN with k=3
- Imputing with using median

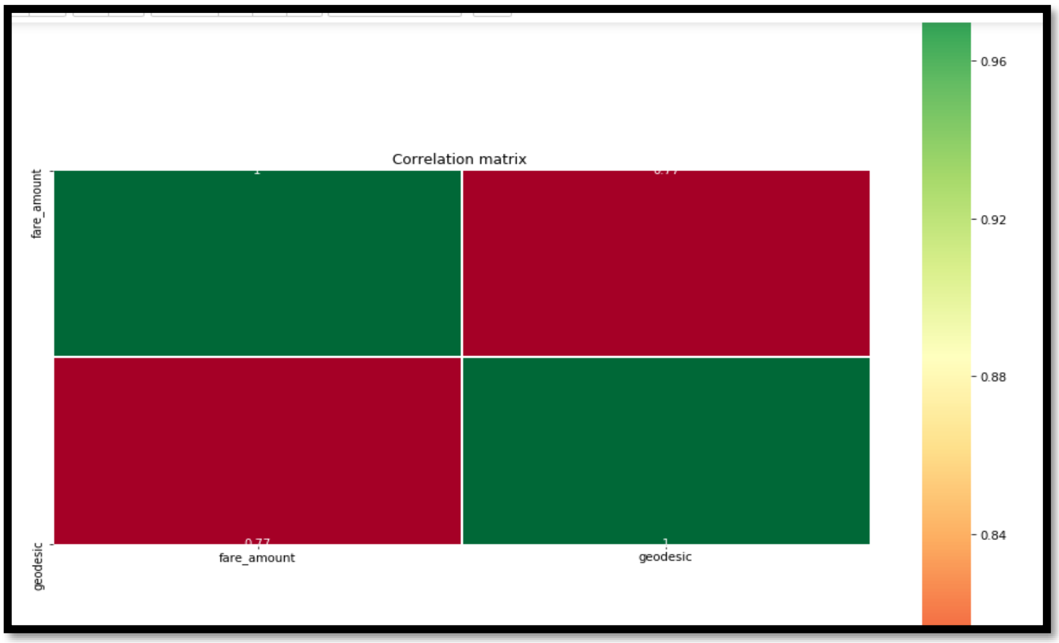
5.5 FEATURE SELECTION

In this step we would allow only to pass relevant features to further steps. We remove irrelevant features from the dataset. We do this by some statistical techniques, like we look for features which will not be helpful in predicting the target variables. (fare_amount)

Correlation analysis – This requires only continuous variables. Therefore, we will filter out only continuous variables and feed it to correlation analysis. We do this by plotting correlation plot for all numerical variables. There should be no correlation between independent variables but there should be high correlation between independent variable and dependent variable.



R language



Python Language

2 **Chi-Square test of independence** – Unlike correlation analysis we will filter out only categorical variables and pass it to Chi-Square test. 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.

Before proceeding to calculate chi-square statistic, we do the hypothesis testing:

- Null hypothesis: 2 variables are independent.
- Alternate hypothesis: 2 variables are not independent.

I. For theoretical purpose: If chi-square statistics is greater than critical value then reject the null hypothesis saying that 2 variables are dependent and if it's less, then accept the null hypothesis saying that 2 variables are independent.

II. While programming: If p-value is less than 0.05 then we reject the null hypothesis saying that 2 variables are dependent and if p-value is greater than 0.05 then we accept the null hypothesis saying that 2 variables are independent.

   - If p-value<0.05 then remove the variable.
   - If p-value>0.05 then keep the variable.

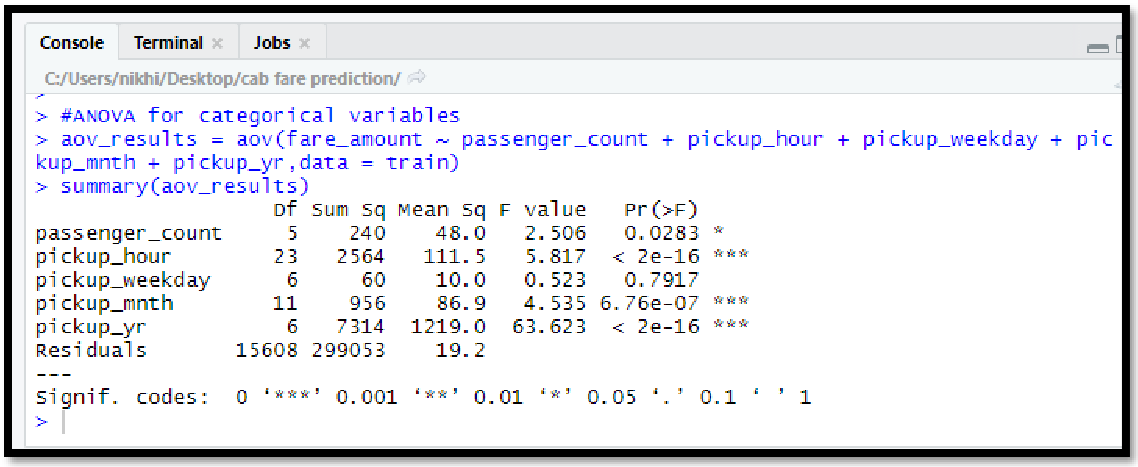
we cannot do chi-square test because fare_amount(target) is not categorical

3 **Analysis of Variance(Anova) Test** – It is carried out to compare between each group in a categorical variable.
ANOVA only lets us know the means for different groups are same or not. It doesn’t help us identify which mean is different.

Hypothesis testing:

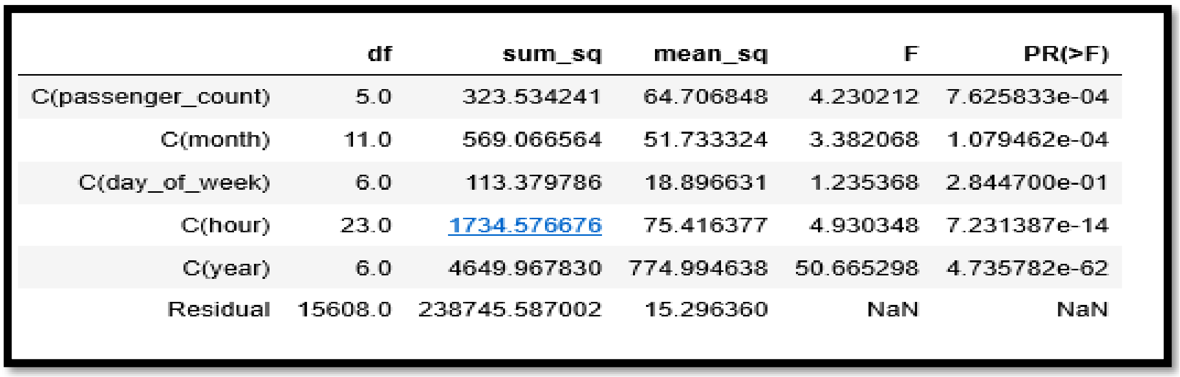
- **Null Hypothesis**: mean of all categories in a variable are same.
- **Alternate Hypothesis**: mean of at least one category in a variable is different.

- If p-value is less than 0.05 then we reject the null hypothesis.
- And if p-value is greater than 0.05 then we accept the null hypothesis.



```
> #ANOVA for categorical variables
> aov_results = aov(fare_amount ~ passenger_count + pickup_hour + pickup_weekday + pic
kup_mnth + pickup_yr,data = train)
> summary(aov_results)
                   Df Sum Sq Mean Sq F value   Pr(>F)    
passenger_count     5    240    48.0   2.506   0.0283 *  
pickup_hour        23   2564   111.5   5.817  < 2e-16 ***
pickup_weekday      6     60    10.0   0.523   0.7917    
pickup_mnth        11    956    86.9   4.535 6.76e-07 ***
pickup_yr           6   7314  1219.0  63.623  < 2e-16 ***
Residuals       15608 299053    19.2                     
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> 
```

➢ pickup_weekday has p value greater than 0.05 therefore we remove from train and test dataset (R language)



	df	sum_sq	mean_sq	F	PR(>F)
C(passenger_count)	5.0	323.534241	64.706848	4.230212	7.625833e-04
C(month)	11.0	569.066564	51.733324	3.382068	1.079462e-04
C(day_of_week)	6.0	113.379786	18.896631	1.235368	2.844700e-01
C(hour)	23.0	1734.576676	75.416377	4.930348	7.231387e-14
C(year)	6.0	4649.967830	774.994638	50.665298	4.735782e-62
Residual	15608.0	238745.587002	15.296360	NaN	NaN

Python Language

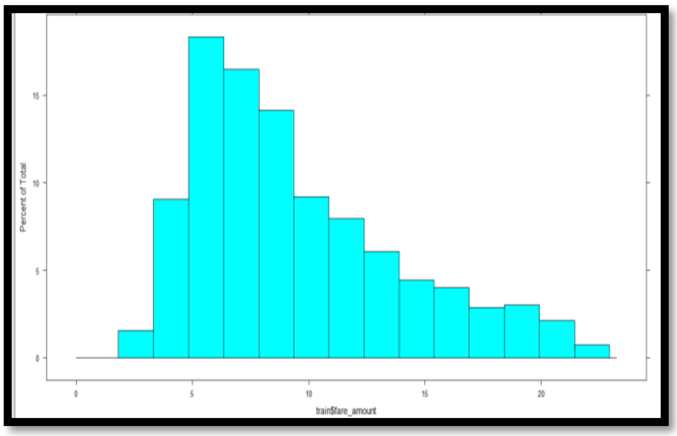
Every variable has p-value less than 0.05 therefore we reject the null hypothesis.

5.6 FEATURE SCALING

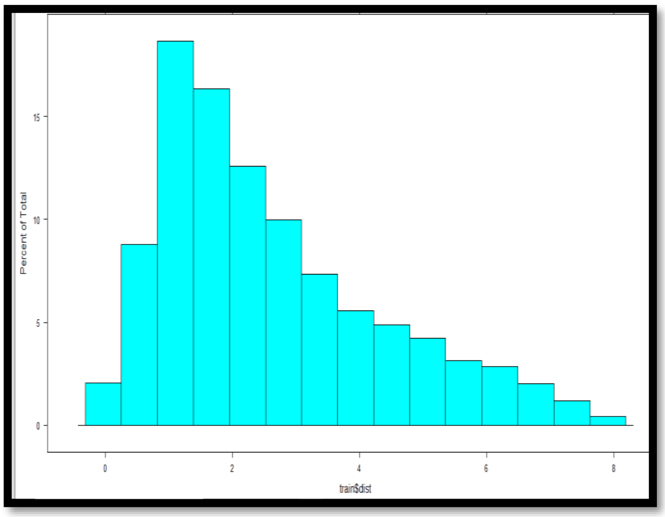
Data Scaling methods are used when we want our variables in data to scaled on common ground. It is performed only on continuous variables.

Normalization: normalization normalizes the data in the range of 0 to 1. It is generally used when we are planning to use distance method for our model development purpose such as KNN.

Standardization: Standardization refers to the subtraction of mean from individual point and then dividing by its SD. When the data is distributed normally you should go for standardization.



Histogram for fare_amount



Histogram for dist

Therefore both the variables are right skewed

$$\text{New value} = \frac{\text{value} - \text{min value}}{\text{Max value} - \text{min value}}$$

6. MACHINE LEARNING ALGORITHMS

- We have divided whole **train** Dataset into **train_data** for application of model and **test_data** for predictions (for testing performance of model)
- 25% is in test_data and 75% is in train_data.
- 11746 observations in train_data and 3914 observations in test_data.
- The model which performs best will be chosen to perform on test dataset provided along with original train dataset.

This is a regression problem as our target variable is fare_amount which is continuous is nature.

6.1 LINEAR REGRESSION MODEL

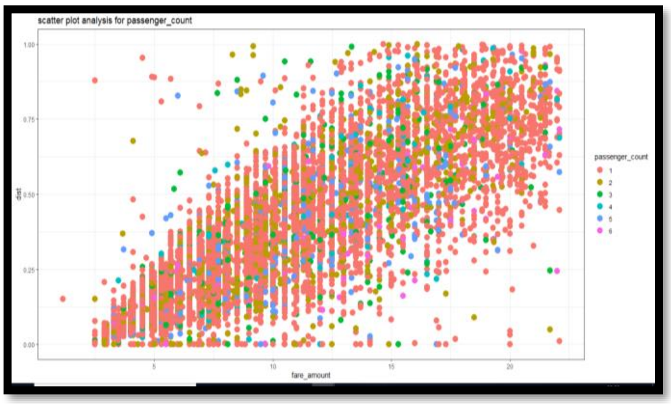
Multiple linear regression model is a statistical model used for regression.
Parameters which help in evaluation are:

- R- square
- Adjusted R –square
- RMSE

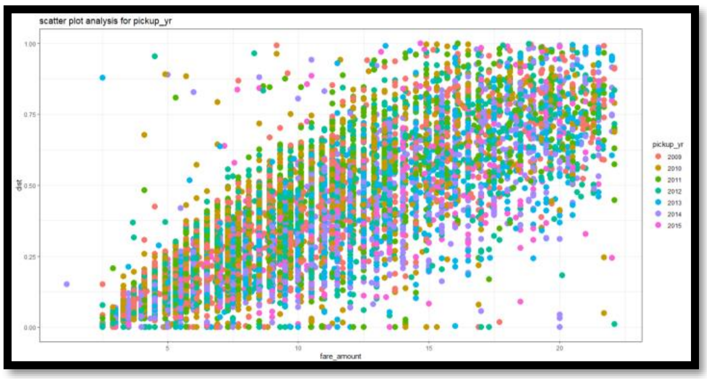
Key assumptions:

- Linear relationship (positive or negative) between dependent and independent variable

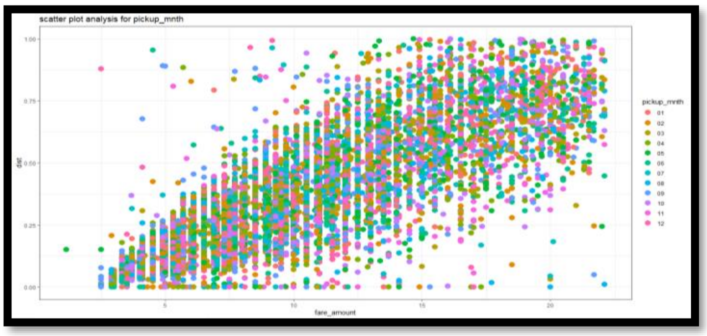
- passenger_count



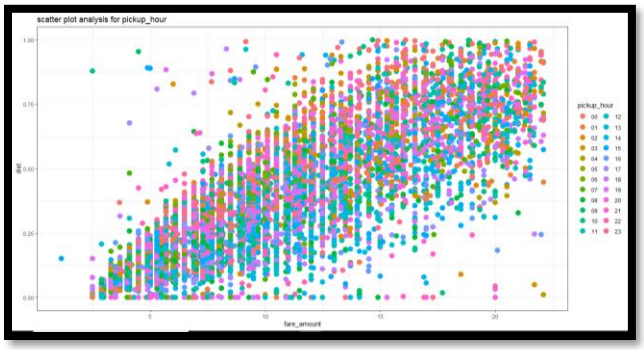
- pickup_yr



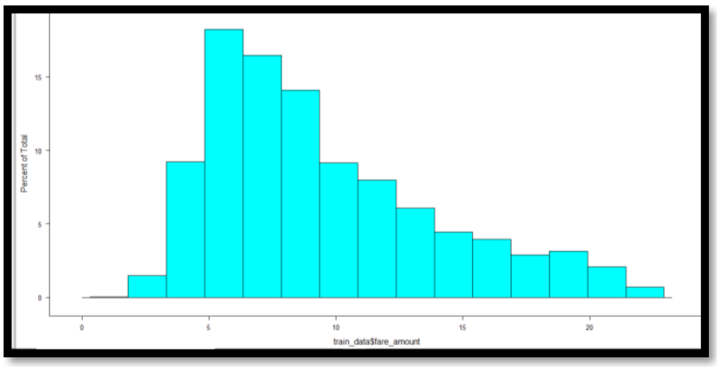
- pickup_mnth



- pickup_hour

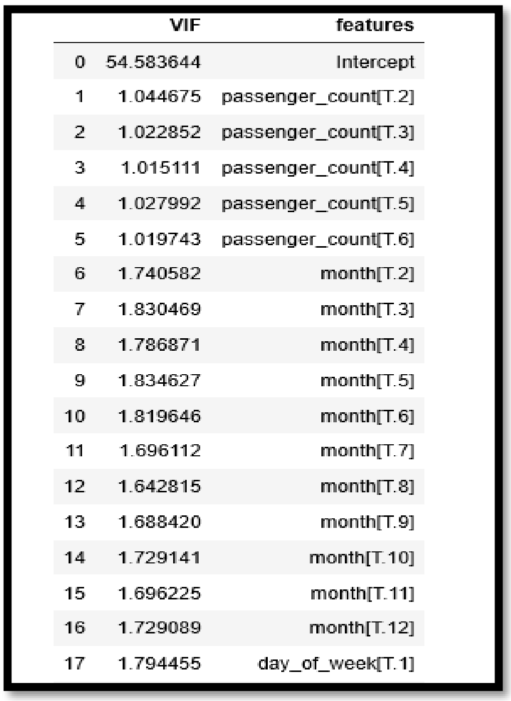
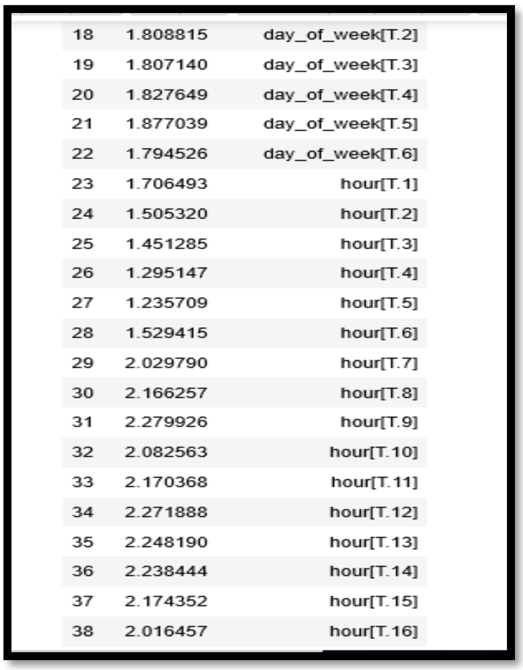


- target variable should be normally distributed



It is slightly right skewed

- check for multi collinearity for all variables by VIF test

	VIF	features
0	54.583644	Intercept
1	1.044675	passenger_count[T.2]
2	1.022852	passenger_count[T.3]
3	1.015111	passenger_count[T.4]
4	1.027992	passenger_count[T.5]
5	1.019743	passenger_count[T.6]
6	1.740582	month[T.2]
7	1.830469	month[T.3]
8	1.786871	month[T.4]
9	1.834627	month[T.5]
10	1.819646	month[T.6]
11	1.696112	month[T.7]
12	1.642815	month[T.8]
13	1.688420	month[T.9]
14	1.729141	month[T.10]
15	1.696225	month[T.11]
16	1.729089	month[T.12]
17	1.794455	day_of_week[T.1]
18	1.808815	day_of_week[T.2]
19	1.807140	day_of_week[T.3]
20	1.827649	day_of_week[T.4]
21	1.877039	day_of_week[T.5]
22	1.794526	day_of_week[T.6]
23	1.706493	hour[T.1]
24	1.505320	hour[T.2]
25	1.451285	hour[T.3]
26	1.295147	hour[T.4]
27	1.235709	hour[T.5]
28	1.529415	hour[T.6]
29	2.029790	hour[T.7]
30	2.166257	hour[T.8]
31	2.279926	hour[T.9]
32	2.082563	hour[T.10]
33	2.170368	hour[T.11]
34	2.271888	hour[T.12]
35	2.248190	hour[T.13]
36	2.238444	hour[T.14]
37	2.174352	hour[T.15]
38	2.016457	hour[T.16]

Results:

Training Data Score

- r square: 0.7343038486646876
- Adjusted r square:0.7338732962659352
- MAPE:18.736942495849092
- MSE: 5.284481967778569
- **RMSE: 2.2988001147943615**
- RMSLE: 0.2166043577109918

Test Data Score

- r square 0.7419452241961578
- Adjusted r square:0.7406864204117489
- MAPE:18.96299602213701
- MSE: 5.291113370017995
- **RMSE: 2.30024202422658**
- RMSLE: 0.2154998534679604

6.2 DECISION TREE MODEL

Results:

Training Data Score

- r square 0.7471531379429529
- Adjusted r square:0.7467434074202166
- MAPE:18.54709262686637
- MSE: 5.028919976577308
- **RMSE: 2.2425253569530286**
- RMSLE: 0.2087118879388543

Test Data Score

- r square 0.7408782468914072
- Adjusted r square:0.739614238339658
- MAPE:19.072219163205943
- MSE: 5.312990500038495
- **RMSE: 2.304992516265182**
- RMSLE: 0.2123390923855204

6.3 RANDOM FOREST

Results:

Training Data Score

- r square 0.7893248033533271
- Adjusted r square:0.7889834107105734
- MAPE:16.95972203709288
- MSE: 4.190159594493088
- **RMSE: 2.0469879321806195**
- RMSLE: 0.19142097538465247

Test Data Score

- r square 0.7542305881615392
- Adjusted r square:0.7530317129818394
- MAPE:18.565011130411502
- MSE: 5.039216255034242
- **RMSE: 2.2448198714004297**
- RMSLE: 0.20710042575083681