**Cab Fare Prediction**

**Project**

**Content:**

1. **DATA:**

1.1 Problem Statement

1.2 Number of attributes

1. **Data Visualization**
2. **DATA PRE-PROCESSING**

3.1 Missing value analysis

3.2 Feature Engineering

3.3 Outlier analysis

3.4 Feature Scaling

1. **Model Selection**:

4.1 Linear regression

4.2 Decision Tree

4.3 Random Forest

4.4 R Code file

4.5 Python code file

1. **Deployment of Machine Learning models**
   1. Deployment of python code
   2. Deployment of R code
2. **Summary**
3. **DATA:**

**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.

**Problem type:** Above problem statement is a Regression problem.A **regression problem** is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear **regression**. It tries to fit data with the best hyper-plane which goes through the points.

Below are the few regression machine learning algorithms.

Linear regression, Decision tree, Random Forest etc.

**1.2 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.

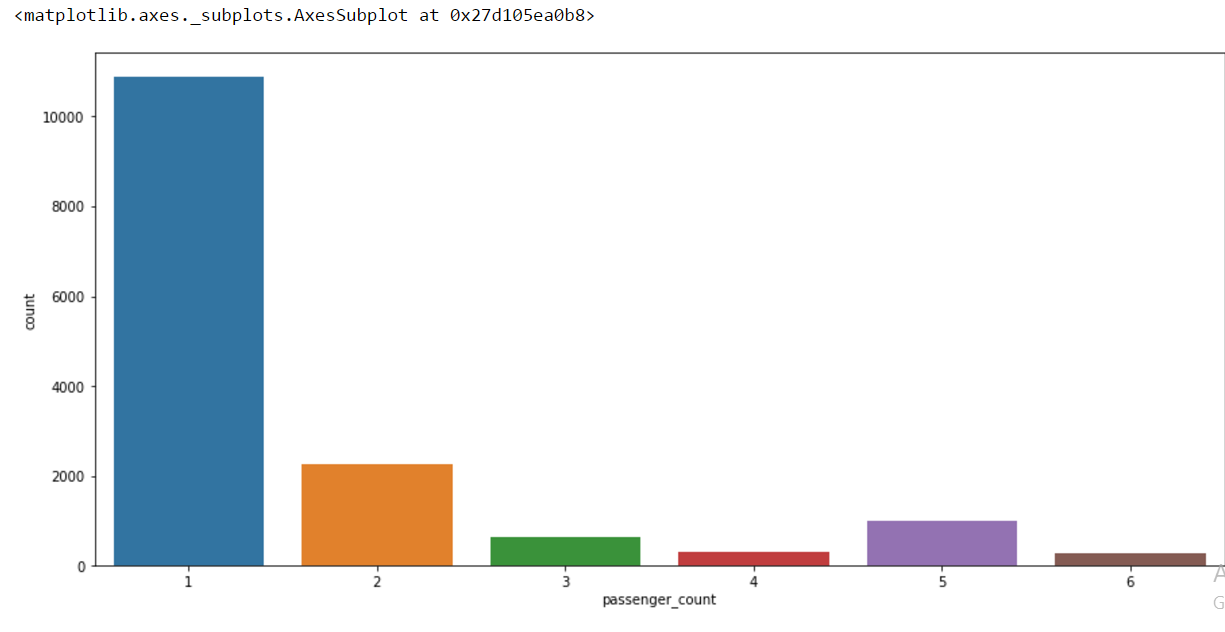
dropoff\_longitude - float for longitude coordinate of where the cab ride ended.

dropoff\_latitude - float for latitude coordinate of where the cab ride ended.

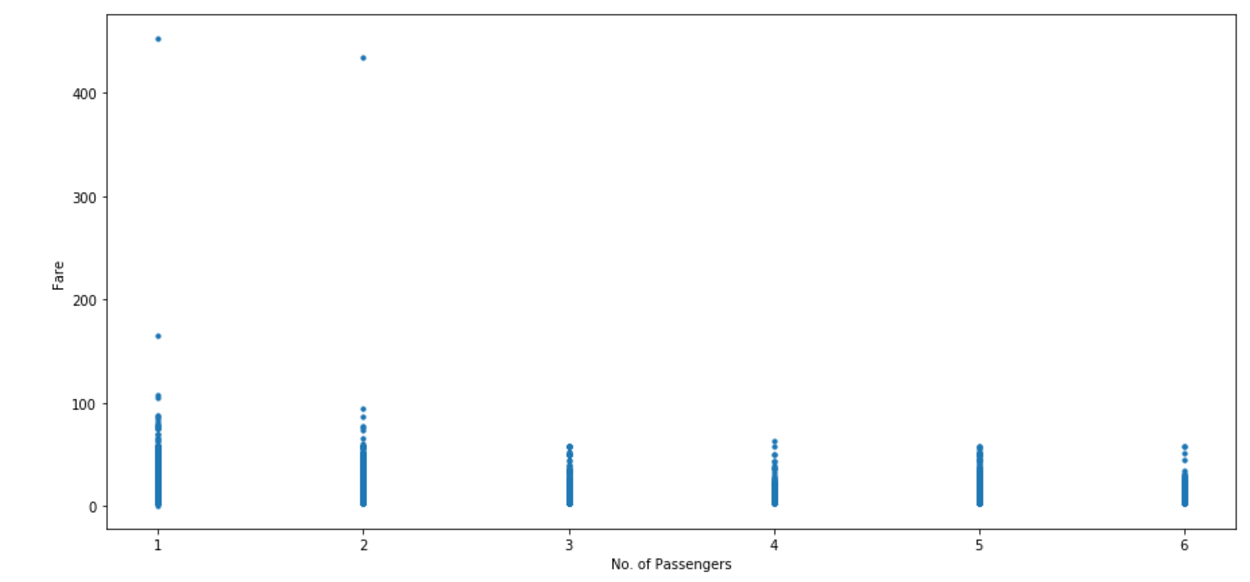
passenger\_count - an integer indicating the number of passengers in the cab ride.

1. **Data Visualization:**

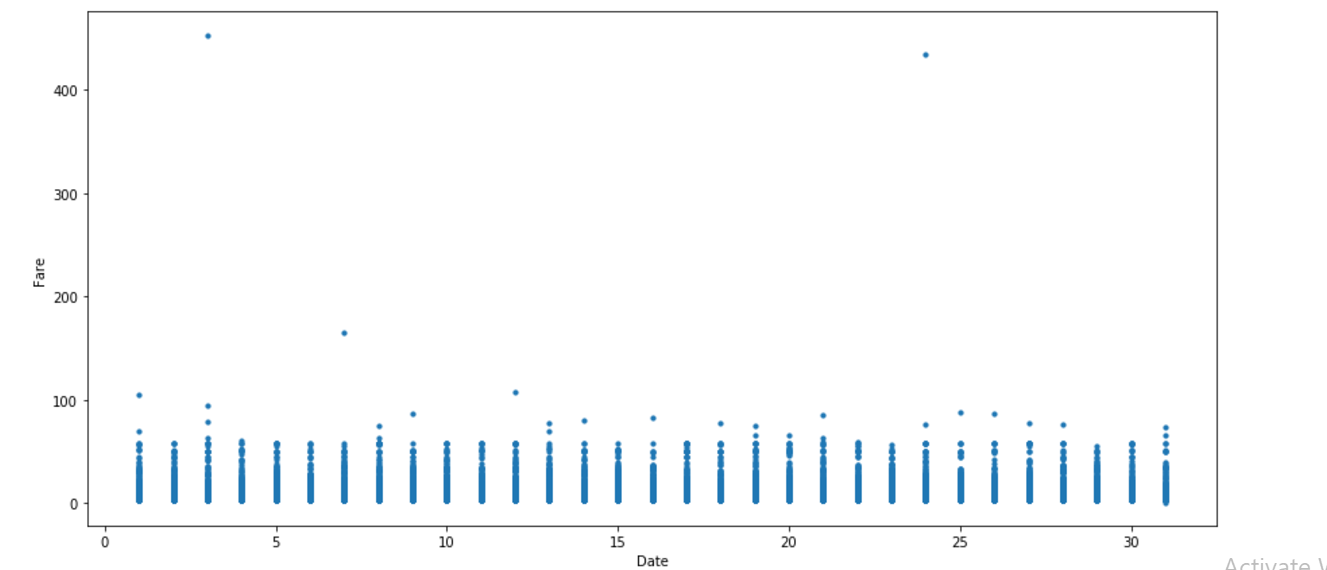
* Count Plot on passenger count



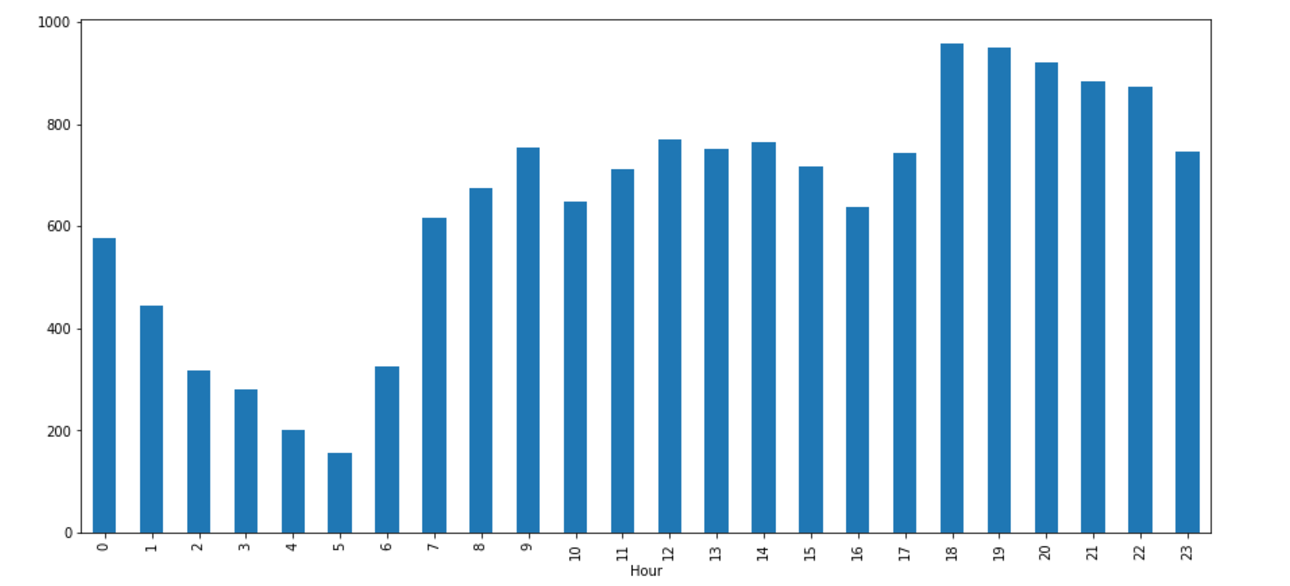
* Relationship between number of passengers and Fare(Scatter plot).



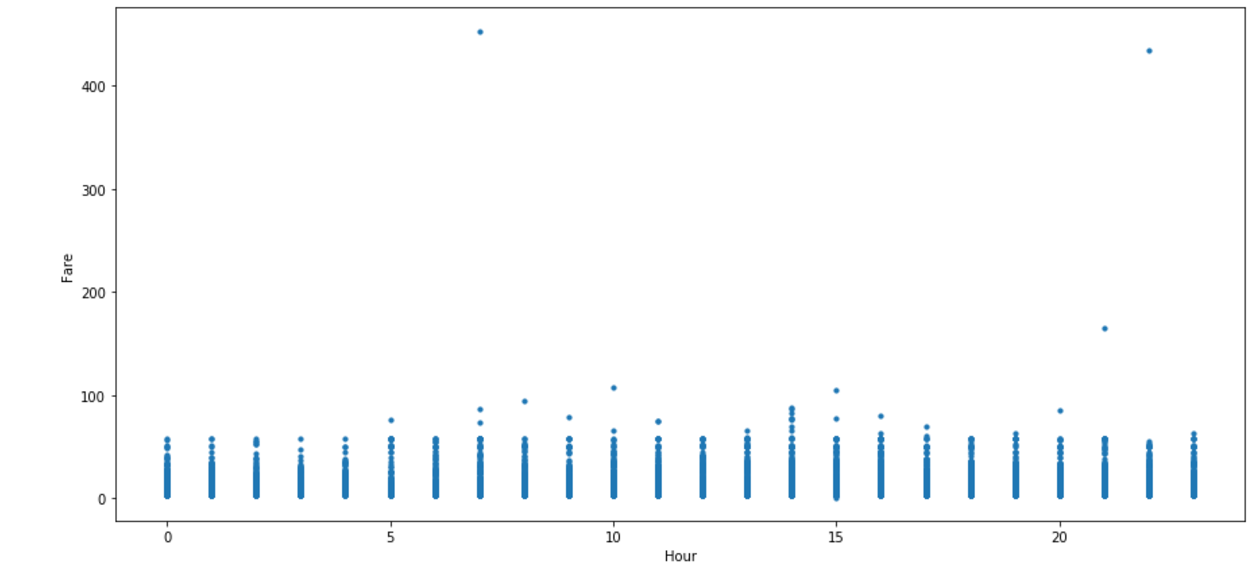
* Relationship between date and Fare (scatter plots).



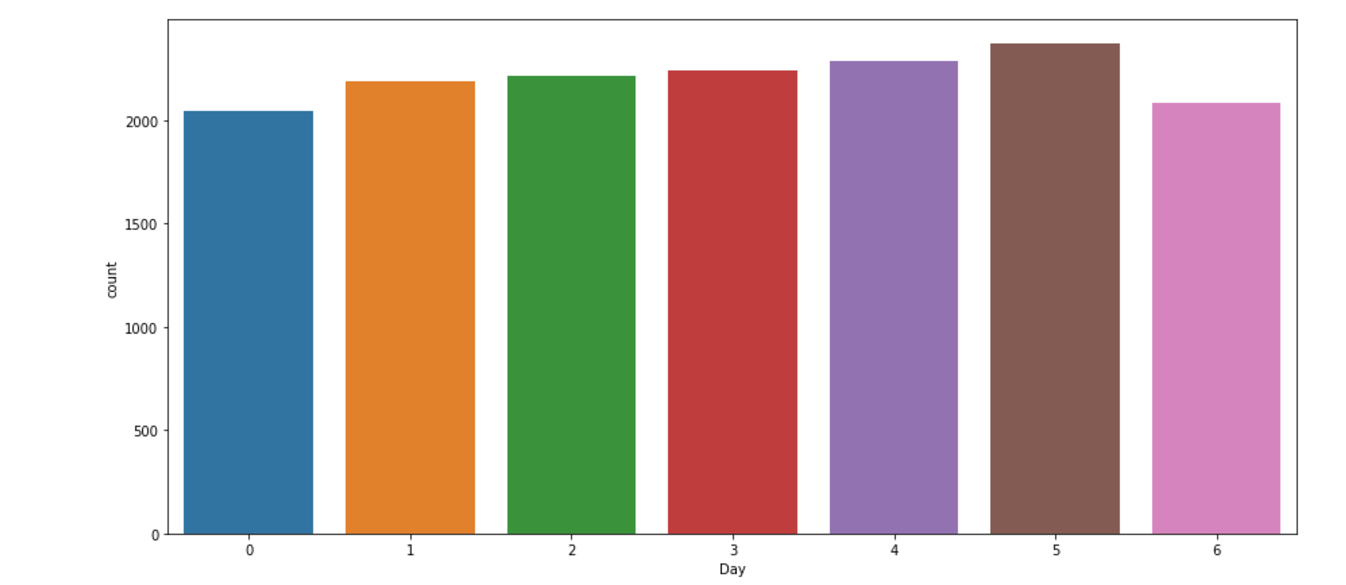
* Bar plot



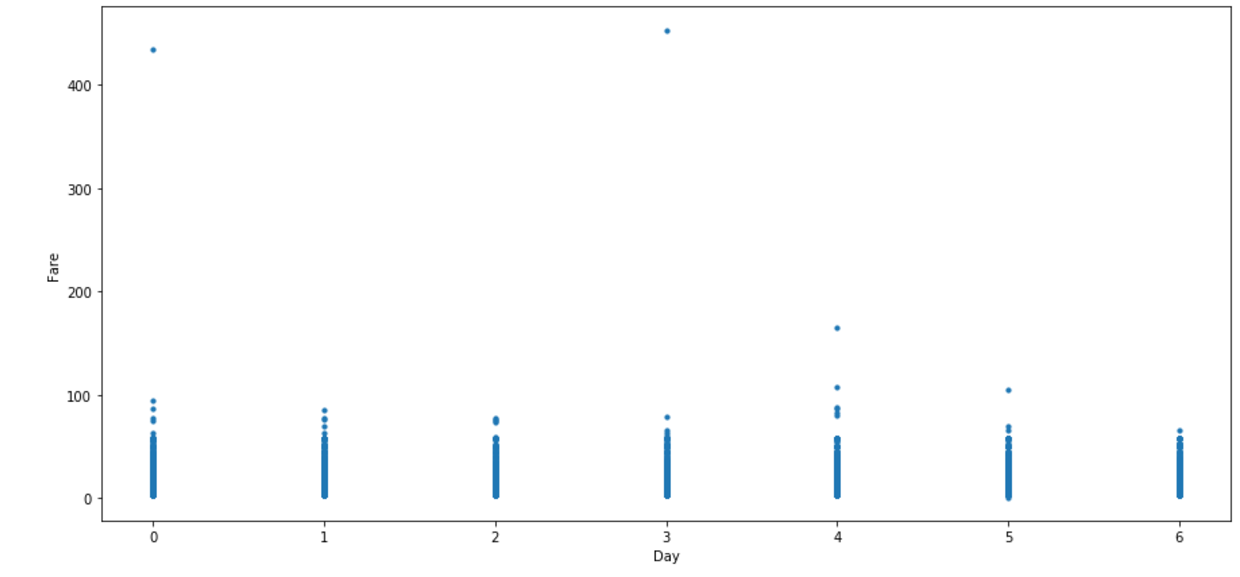
* Relationship between Time and Fare(scatter plot).



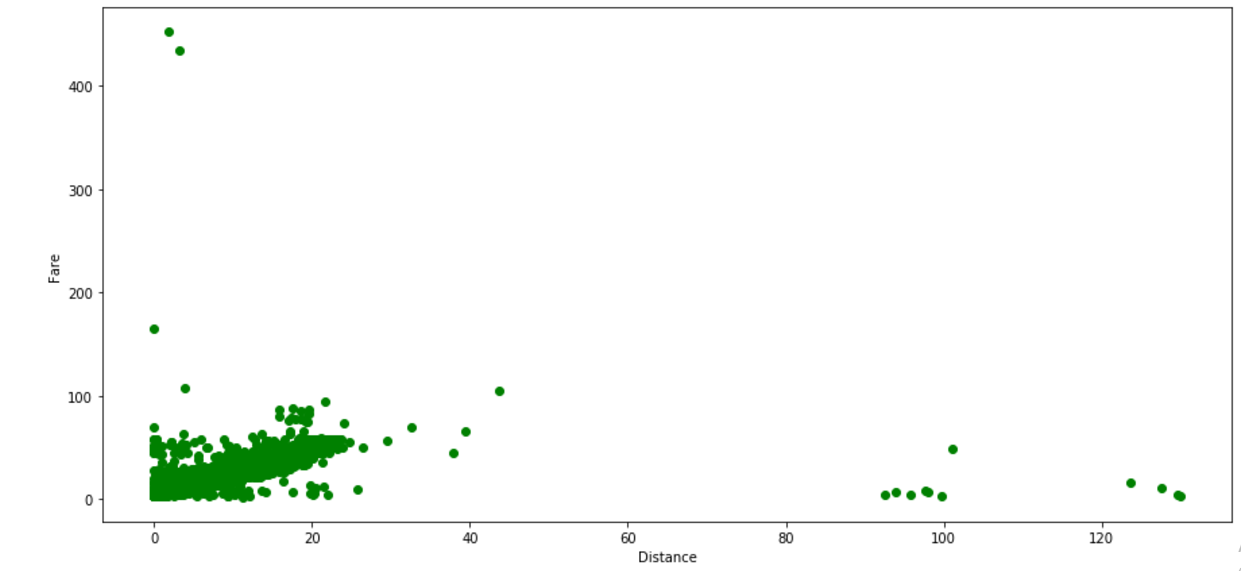
* Impact of Day on the number of cab ride (Bar plot).



* Relationships between day and Fare



* Relationship between distance and fare (Scatter plot).



Observations from the above graphs:

* Number of Passengers effects the fare
* Pickup date and time effects the fare
* Day of the week does effects the fare
* Distance effects the fare

1. **DATA PRE-PROCESSING :**

Below are the steps we have done in project as data pre-processing.

* Changing the data types of variables
* Removing values which are not within desired range (outlier) depending upon basic understanding of dataset.
* Fare amount has a negative value. A price amount cannot be -ve and also cannot be 0. So we will remove these fields.
* 20 observations of passenger\_count is consistenly from 7,8,9,10 passenger\_counts.
* Also we removed value/row having passenger\_count==0, because there is no meaning in zero passenger count.
* We have removed 20 observation which are above 6 value, because a cab cannot hold these number of passengers.
* Latitudes range from -90 to 90.Longitudes range from -180 to 180.Removing which does not satisfy these ranges.
* There's only one outlier which is in variable pickup\_latitude. So we removed it with nan.

**3.1 Missing value analysis:**

Data, or missing values, occur when no data value is stored for the variable in an observation. Missing data are a common occurrence and can have a significant effect on the conclusions that can be drawn from the data. Used KNN, Mode method to impute the missing value analysis in this project.

**3.2 Feature Engineering:**

**Feature engineering** is the process of using domain knowledge to extract **features** from raw data via data mining techniques. These **features** can be used to improve the performance of **machine learning** algorithms. **Feature engineering** can be considered as applied **machine learning** itself. We are done with Feature Engineering for timestamp variable for both given test and train data set.

* derive new features from pickup\_datetime variable and got new features year,month,day\_of\_week,hour
* Convert pickup\_datetime from factor to date time
* Calculate the distance travelled using longitude and latitude
* Used haversing formula to calculate the distance from longitude and latitude values. The **haversine formula** determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general **formula** in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles.

deg\_to\_rad = function(deg){

(deg \* pi) / 180

}

haversine = function(long1,lat1,long2,lat2){

phi1 = deg\_to\_rad(lat1)

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))

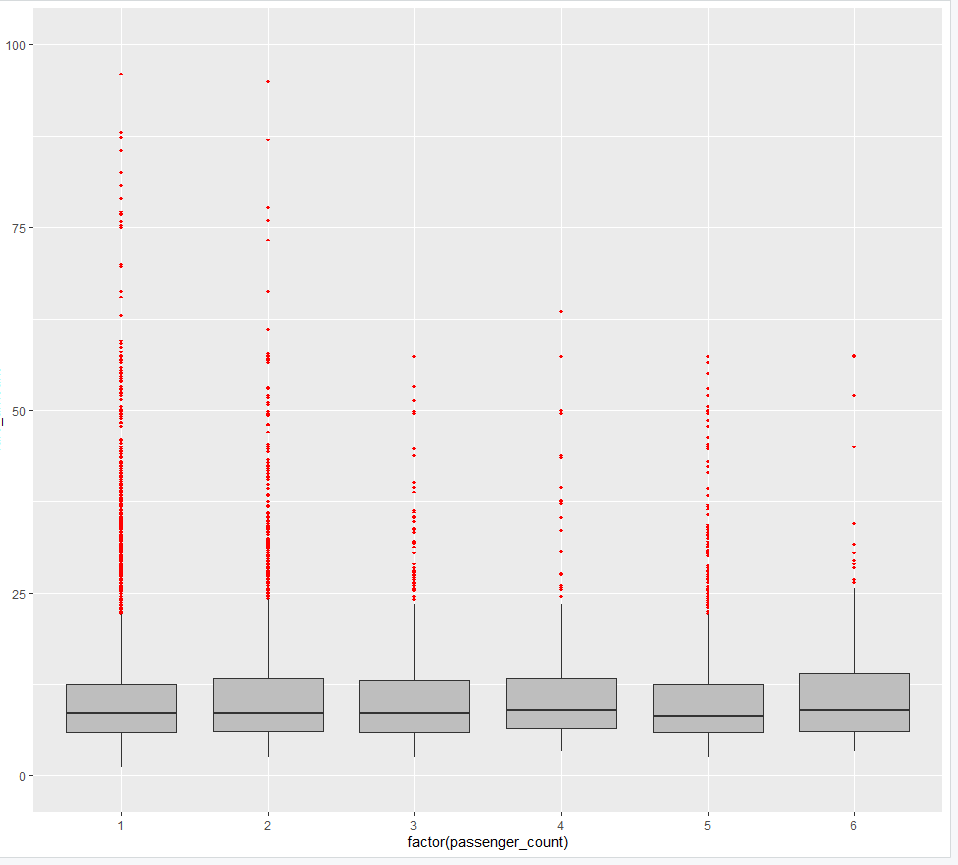
R = 6371e3

R \* c / 1000

}

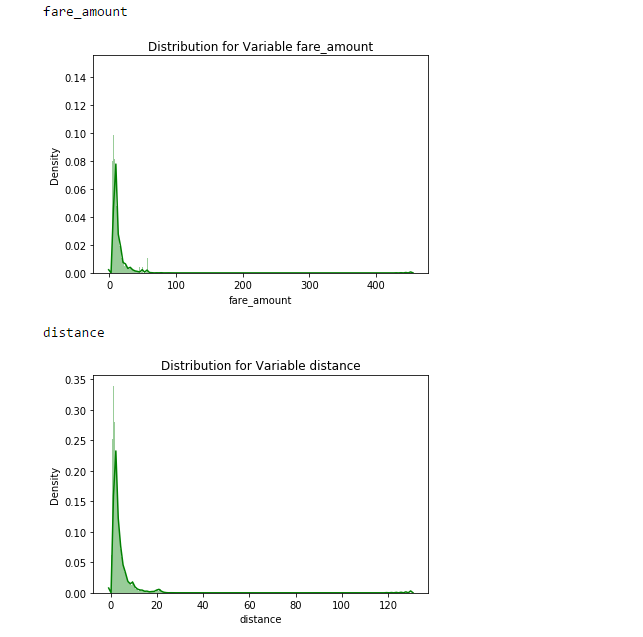
**3.3 Outlier analysis:**

An Outlier analysis is finding the outlier from the given dataset. An **outlier** is an observation that diverges from an overall pattern on a sample. **Outliers** can be of two kinds: univariate and multivariate. Univariate **outliers** can be found when looking at a distribution of values in a single feature space. Below graph is the representation of outliers.

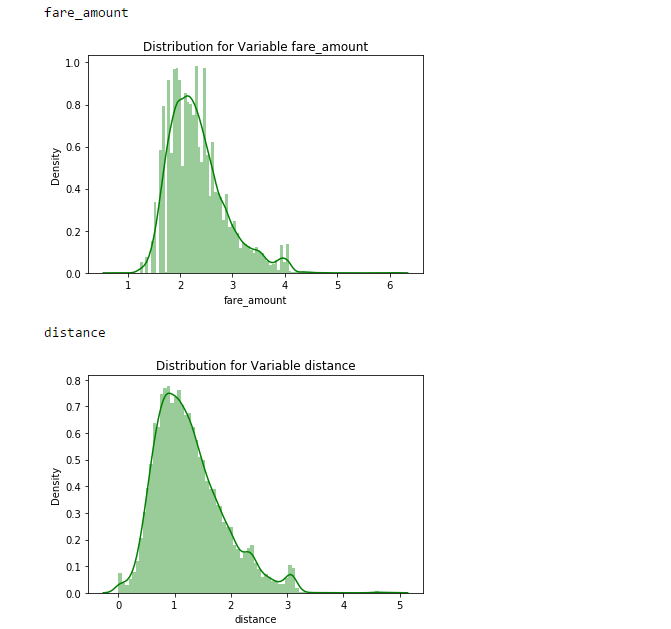


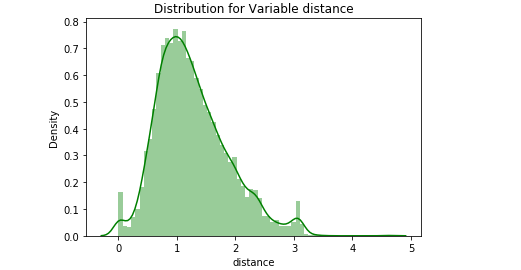
**3.4 Feature Scaling:**

Feature scaling is a method used to normalize the range of independent variables or features of data. Need to check the distribution of data. Below graph are the distribution of fare amount and distance.



* Since skewness of distance variable is high, apply log transform to reduce the skewness. Skewness refers to distortion or asymmetry in a symmetrical bell curve, or normal **distribution**, in a set of **data**. If the curve is shifted to the left or to the right, it is said to be **skewed**. Skewness can be quantified as a representation of the extent to which a given **distribution** varies from a normal **distribution.** Re-check the Normality to check data is uniformly distributed or not, after log transformation.





**Note:**

Test & train data set :

As we can see a bell shaped distribution. Hence our continuous variables are now normally distributed, we will not use any Feature Scaling technique. i.e, Normalization or Standardization for our test data

By seeing the above plots we can easily conclude that:

* Single travelling passengers are most frequent travellers.
* At the sametime we can also conclude that highest Fare are coming from single & double travelling passengers.
* Lowest cabs at 5 AM and highest at and around 7 PM i.e the office rush hours.
* From the above plot We can observe that the cabs taken at 7 am and 23 Pm are the costliest. Hence we can assume that cabs taken early in morning and late at night are costliest
* Observations : The day of the week does not seem to have much influence on the number of cabs ride. The highest fares seem to be on a Sunday, Monday and Thursday, and the low on Wednesday and Saturday. May be due to low demand of the cabs on Saturday the cab fare is low and high demand of cabs on Sunday and Monday shows the high fare prices

**Splitting train into train and validation subset:** 75% in training and 25% in Validation Datasets

1. **Model Selection**:

**4.1 Linear regression:**

Linear regression is a machine learning algorithm based on supervised learning. Linear regression is most predictive analysis statistical model.it can be used for finding missing value analysis and only for regression problems. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship and forecasting.

Different regression models differ based on the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

Linear regression performs the task to predict a dependent variable value(*y*) based on a given independent variable(*x*).So this regression technique finds out a linear relationship between *x* and *y*. hence, the name is linear regression.

Y=b0 +b1\**x*

Here b0 is intercept

b1 is coefficient of *x*

For a given set of value we need to calculate values for b0 & b1.

The model aims to predicted Y value such that the error difference between predicted value and true value is minimum.

Formula to calculate b1 by least square estimators

b1 = [ i*yi* - i \* i)]/[i2 – 1/n(i)2]

*xi*  is independent variable, *yi* is target variable.

b0 = ymean - b1*x*mean

Hypothesis for linear regression model, H0(null hypothesis) and H1(alternative hypothesis).

(Null hypothesis)H0 🡪 b1 = 0.

H1 🡪 b1 ≠ 0.

After building a model will get p-value (probability value), based on the p will consider the hypothesis. If the p-value is less than 0.05 then, we can reject null hypothesis. **Regression coefficients** represent the **mean** change in the response variable for one unit of change in the predictor variable while holding other predictors in the model constant. ... The key to understanding the **coefficients** is to think of them as slopes, and they're often called slope **coefficients.**

**Reason to select:**

The biggest **advantage of linear regression** models is linearity: It makes the estimation procedure simple and, most importantly, these **linear** equations have an easy to understand interpretation on a modular level (i.e. the weights).

**4.2 Decision Tree:**

DT is the one of the supervised machine learning algorithm. It builds **regression** or **classification** models in the form of a **tree** structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated **decision tree** is incrementally developed. DT are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning.

The XGBoost library implements the gradient boosting decision tree algorithm.

This algorithm goes by lots of different names such as gradient boosting, multiple additive regression trees, stochastic gradient boosting or gradient boosting machines. Boosting is an ensemble technique where new models are added to correct the errors made by existing models. Models are added sequentially until no further improvements can be made. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

**Reason to select:**

Different kinds of models have different advantages. The decision tree model is very good at handling tabular data with numerical features, or categorical features with fewer than hundreds of categories. Unlike linear models, decision trees are able to capture non-linear interaction between the features and the target.

This approach supports both regression and classification predictive modeling problems.

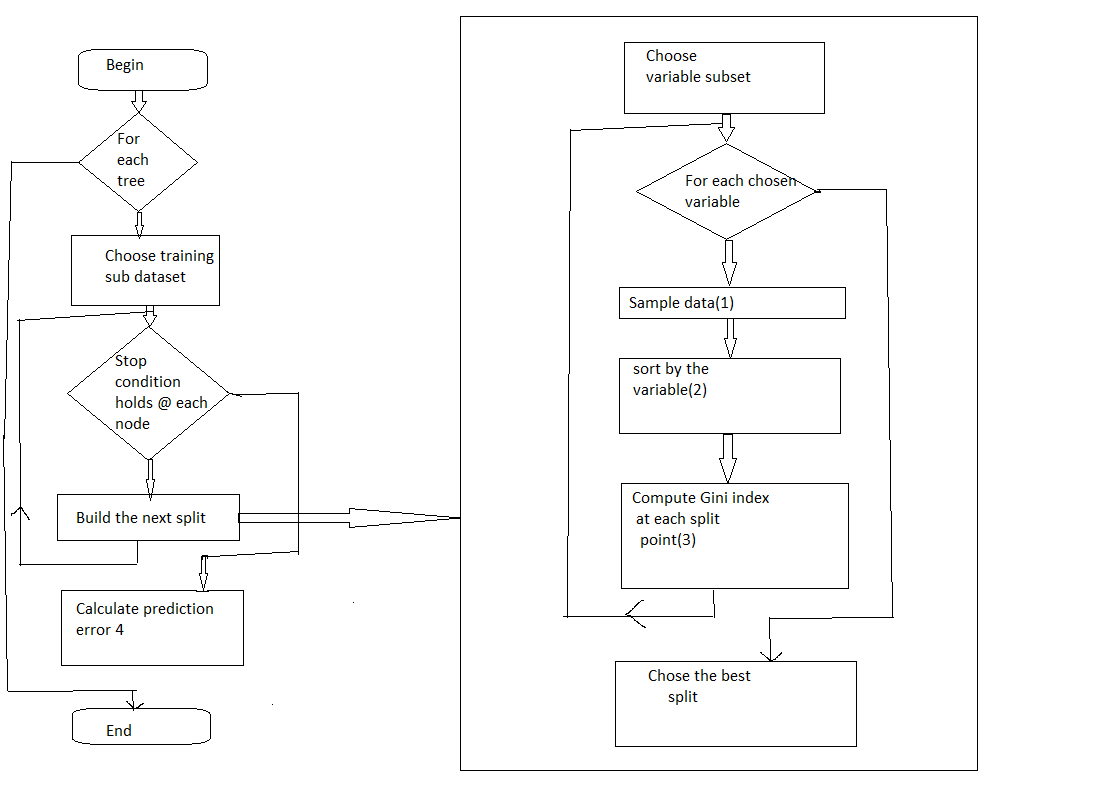
**4.3 Random Forest:**

**Random forest** is a supervised learning **algorithm** which is used for both classification as well as regression. Similarly, **random forest algorithm** creates **decision** trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting.

How it works: Number of decision tree depends on error rate- build trees until the error no longer decreases. Actually the next tree random forest is going to create is based on the error observed in the previous tree. If the new tree is observing the same error or you can say if the algorithm is capturing the same error consistently then it will stop creating the trees.

RF algorithm constructed using following steps:

1. Let the number of training cases be N and the number of variable in the classifier be M.
2. We are told the number of m of i/p variable to be used to determine the decision at a node of tree, m should be much less than M.
3. m = sqrt(M)
4. Choose a training set for this tree by choosing n times with replacement from all N available training cases. Use the rest of the cases to estimate the error of the tree, by predicting their classes.
5. For each node of the tree, randomly choose m variables on which to base the decision@ the node. Calculate the best split based on these m variables in the training set.
6. Each tree is fully grown and not pruned.
7. Number of decision tree depends on error rate-build trees until the error no longer decreases.
8. For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the training sample in terminal node it ends up in.



**Reason to select:**

Random forest runtimes are quite fast, and they are able to deal with unbalanced and missing data.

R2 values of above algorithms:

**R-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, or the coefficient of multiple determination for multiple regression.

Python:

|  |  |
| --- | --- |
| **ML Algorithm** | **R Square Value** |
| Linear regression | 0.7827 |
| Decision Tree | 0.7033 |
| Random Forest | 0.801 |

Based on R2value, selected RF as final test evaluation.

**MAPE**:

The mean absolute percentage error (**MAPE**) is a **statistical** measure of how accurate a forecast system is. It measures this accuracy as a percentage, and can be calculated as the average absolute percent error for each time period minus actual **values** divided by actual **values.**

R-Code:

|  |  |
| --- | --- |
| **ML Algorithm** | **MAPE** |
| Linear regression | 0.4154 |
| Decision Tree | 0.2241 |
| RandomForest | 0.2349 |
| Xgboost | 0.1861 |

Based on MAPE value, selected Xgboost as final test evaluation.

**4.4 R Code file:**

File name: Car\_prediction\_R.R

****

**4.5 Python code file:**

File name: Car\_prediction.ipynb

****

1. **Deployment of Machine Learning models:**

Deploying model is the key to making them useful. Different platforms and frame works available in market for model deployment. Mainly there are two methods of framework deployment one is ‘Online’, other ‘Offline’ method. Software deployment is all of the activities that make a software system available for use. Deployment types, Data mining tools (or cloud)

* RevoDeploy R
* Orange

Both are cloud based tool, here need to deploy our model into cloud and approach a client.

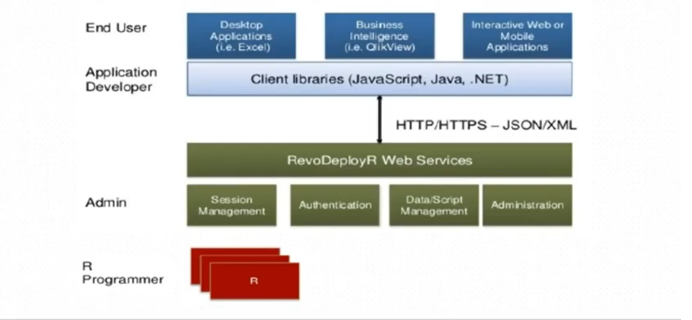
If data related to banking, insurance industry are confidential so they will prefer offline mode. Like e-commerce site data is not confidential, so they will be deployed in online mode. Our R, Python code should support programming languages (Java, C, VB). There will be library which support python to html, CSS. In few case our model should interact with database and SQL scripts (TSQL, PL-SQL).

PMML(Predictive model markup language) it’s also another way to deployment, PMML is a library which will supports R, Python.

Scheduler is also one way to deployment. Below is the offline method



Let us consider online method. RevoDeployR is one of the online deployment method. Below is the flow chart.



Web FrameWork using R. Rserve, Rshiny are interactive application, no web deployment skills required. It combines the computational power of R with the interactivity of the modern web.

**5.1 Deployment of python code:**

Following are the few libraries are resources which will be used,

Pickle, Flask. There is Django, Falcon, Hug and many more.

* **Pickle** is a python library to save (serialize) and load(de-serialize) python objects as files on disk.
* **Flask** is a python based web framework
* **Docker**: This is one of the most popular way of hosting scripts and deploying Machine Learning Models online. You can use Docker to containerize the code and host it as a micro service using different apps.
* **PEX**: [PEX](https://www.youtube.com/watch?v=NmpnGhRwsu0) is a clever tool being developed at Twitter that allows Python code to be shipped as executable zip files.
* **AWS**: Using AWS, you can create a free account and get started with hosting/deployment. Lot of resources availabel online.
* **PYSimple, Pyinstaller and Sparrow**: As mentioned in the answers above, we can use these packages also to do the task.
* **Flask App**: If you dont want to use Docker, using simple flask app, you will be able to host your script online. But there will be a lot of issues as it’s not containerized. Best to use Docker.
* **py2exe**: If you are looking to convert python file into windows executable.
* [**cx\_Freeze**](https://anthony-tuininga.github.io/cx_Freeze/): Similar to the py2exe, you can use this also.

Below are the environment setup for deploying the model using flask, Pickle and python anywhere.

* Pip install the flask, flask\_cors, jsonify and other python packages.

Steps need to be done as follows:

1. Train the model using jupyter notebook(python script).
2. Save the trained model as a pickle file(serialization)
3. Create a flask environment that will have an API endpoint which would encapsulate our trained model and enable it to receive inputs through GET requests over HTTP/HTTPS and then return the output after de-serializing the trained model.
4. Upload the flask script along with trained model on pythonanywhere.

**5.2 Deployment of R code:**

**DeployR is an** integrated technology for deploying R analytics inside web, desktop, mobile and dashboard application as well as backend system. A data scientist develops an R scripts using standard R tool and publishes that script to deploy server, where it become available for execution as analytics web service. Once published R scripts can be executed by any authorized application using DeployR.

Plumber an R package that convert existing R code to a web API by using a handful of special one line comments.

#install the plumber

Install.packages(“plumber”)

#import the plumber

Library(plumber)

R=plumber(“script.R”)

#where script.R is the location of file where R code is stored

#convert to web API

R$rum(port=8000) # local host server

**Instruction to run the code:**

1. Open the attached code files in respective platform (Rstudio/Jupyter notebook).
2. Setup the working directory.
3. Place the train\_cab.csv, test.csv file in working directory.
4. Run the code file by loading the data into ML model.
5. You will get the predictions as output.
6. **Summary:**

**Summarize the Understanding of how this project can help the business in**

**Achieving the strategic goals:**

This project is important, ensures what is being delivered, is right, and will deliver real value against the business opportunity. This project set the strategic goal to client and it helps to advance the business approach to customer. It helps the clients to manage business effectively, gain the fresh perspective of business and resolves problem quickly. It priorities your customers for efficient business. It identify price, will make a specific transaction in the future. Based on this solution clients can prepare the appropriate approaches to customer.