EDWISOR.COM

REPORT

Project Name:

Cab Fare Prediction

Submitted By:

Rishabh Singh

ABSTRACT

The objective of this Case is that we are a cab rental start-up company and have successfully run the pilot project and now want to launch our cab service across the country. We have collected the historical data from our pilot project and now have a requirement to apply analytics for fare prediction. We need to design a system that predicts the fare amount for a cab ride in the city. The dataset of our project contains 7 variables each representing different attributes and properties. The total number of observations are 16000+ entries. This project is divided into four major stages.

The first part contains the Exploratory Data Analysis where we check the head of the dataset, how each variable is type defined and use describe method to check the max, min and other values.

The second part contains the Data preprocessing and data cleaning techniques to make it ready for various Machine Learning Algorithms.

The third part contains the training our model on various Machine Learning Algorithms and testing our model on them.

The last and the final part of the project contains the prediction of our sample test data by most appropriate algorithm.

List of Contents

INTRODUCTION

We are a cab rental start-up company and have successfully run the pilot project and now want to launch our cab service across the country. We have collected the historical data from our pilot project and now have a requirement to apply analytics for fare prediction. We need to design a system that predicts the fare amount for a cab ride in the city.

Number/ Nature of attributes:

- pickup_datetime – time stamp 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.

Out of the total of 7 variables of the dataset:

Categorical variables are: 0

Continuous variables are: 6

Index/String variable: 1

Exploratory Data Analysis

1) Set working directory: To import and export all data to a location
2) Load Data: training data as cabcount_train
3) Checking the shape of Data: 16067 rows, 7 columns
4) Checking the type of each variable in the dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16067 entries, 0 to 16066
Data columns (total 7 columns):
fare_amount          16043 non-null object
pickup_datetime      16067 non-null object
pickup_longitude     16067 non-null float64
pickup_latitude      16067 non-null float64
dropoff_longitude    16067 non-null float64
dropoff_latitude     16067 non-null float64
passenger_count      16012 non-null float64
dtypes: float64(5), object(2)
memory usage: 878.7+ KB
```

5) Check the head of the dataset to know the first five entries:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	4.5	2009-06-15 17:26:21 UTC	-73.844311	40.721319	-73.841610	40.712278	1.0
1	16.9	2010-01-05 16:52:16 UTC	-74.016048	40.711303	-73.979268	40.782004	1.0
2	5.7	2011-08-18 00:35:00 UTC	-73.982738	40.761270	-73.991242	40.750562	2.0
3	7.7	2012-04-21 04:30:42 UTC	-73.987130	40.733143	-73.991567	40.758092	1.0
4	5.3	2010-03-09 07:51:00 UTC	-73.968095	40.768008	-73.956655	40.783762	1.0

6) Apply describe () method to calculate the mean, variance, and standard deviation etc of the dataset:

	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	16067.000000	16067.000000	16067.000000	16067.000000	16012.000000
mean	-72.462787	39.914725	-72.462328	39.897906	2.625070
std	10.578384	6.826587	10.575062	6.187087	60.844122
min	-74.438233	-74.006893	-74.429332	-74.006377	0.000000
25%	-73.992156	40.734927	-73.991182	40.734651	1.000000
50%	-73.981698	40.752603	-73.980172	40.753567	1.000000
75%	-73.966838	40.767381	-73.963643	40.768013	2.000000
max	40.766125	401.083332	40.802437	41.366138	5345.000000

DATA Pre-processing

1) **Missing Value Analysis**: Missing value pertains to condition when one or more values are absent from one or more independent variables of the dataset. The reason for missing values could be many. Some commonly included reasons are Human error, refuse to answer during survey, optional box questionnaire etc. Missing values in a dataset may be imputed or that particular variable may be dropped depending upon the percentage of missing values present in the column.

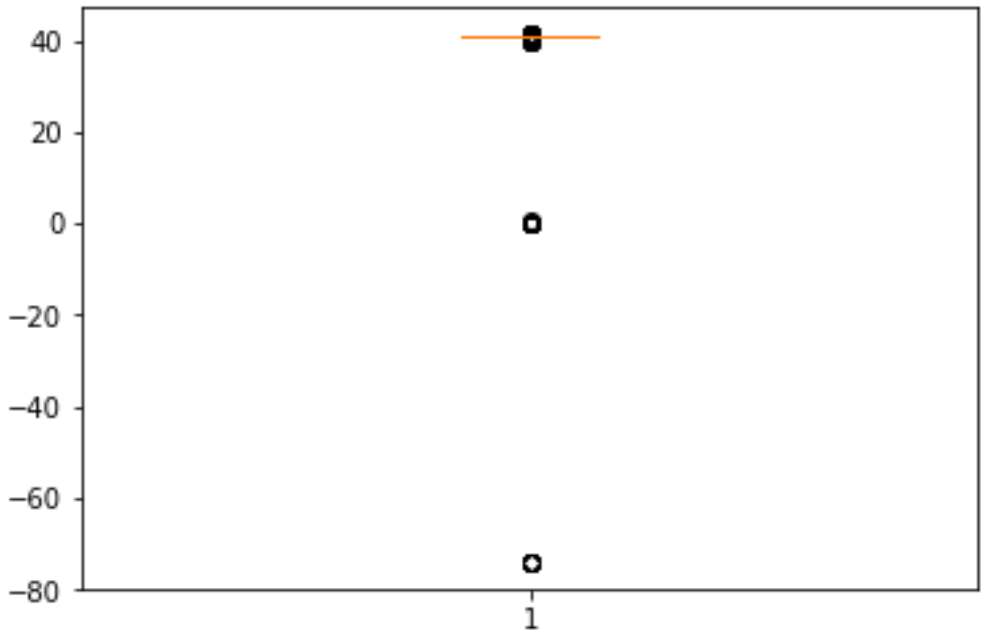
   There are various methods to impute missing values, some of which are:

   a) Mean
   b) Mode
   c) Median
   d) KNN imputation
   e) Prediction method

   Checking the missing value in the Dataset: Our Dataset contain missing values and so to gain highest accuracy in our model we impute them with mean.

2) **Outlier Analysis**: Observations inconsistent with the rest of the dataset is called an outlier. Causes of outliers are poor or contaminated data, low quality measurements, malfunctioning equipment, correct but exceptional data.

- Plot boxplot to visualize Outliers.
- Separate Numeric and Categorical variables.



Box plot for dropoff_latitude

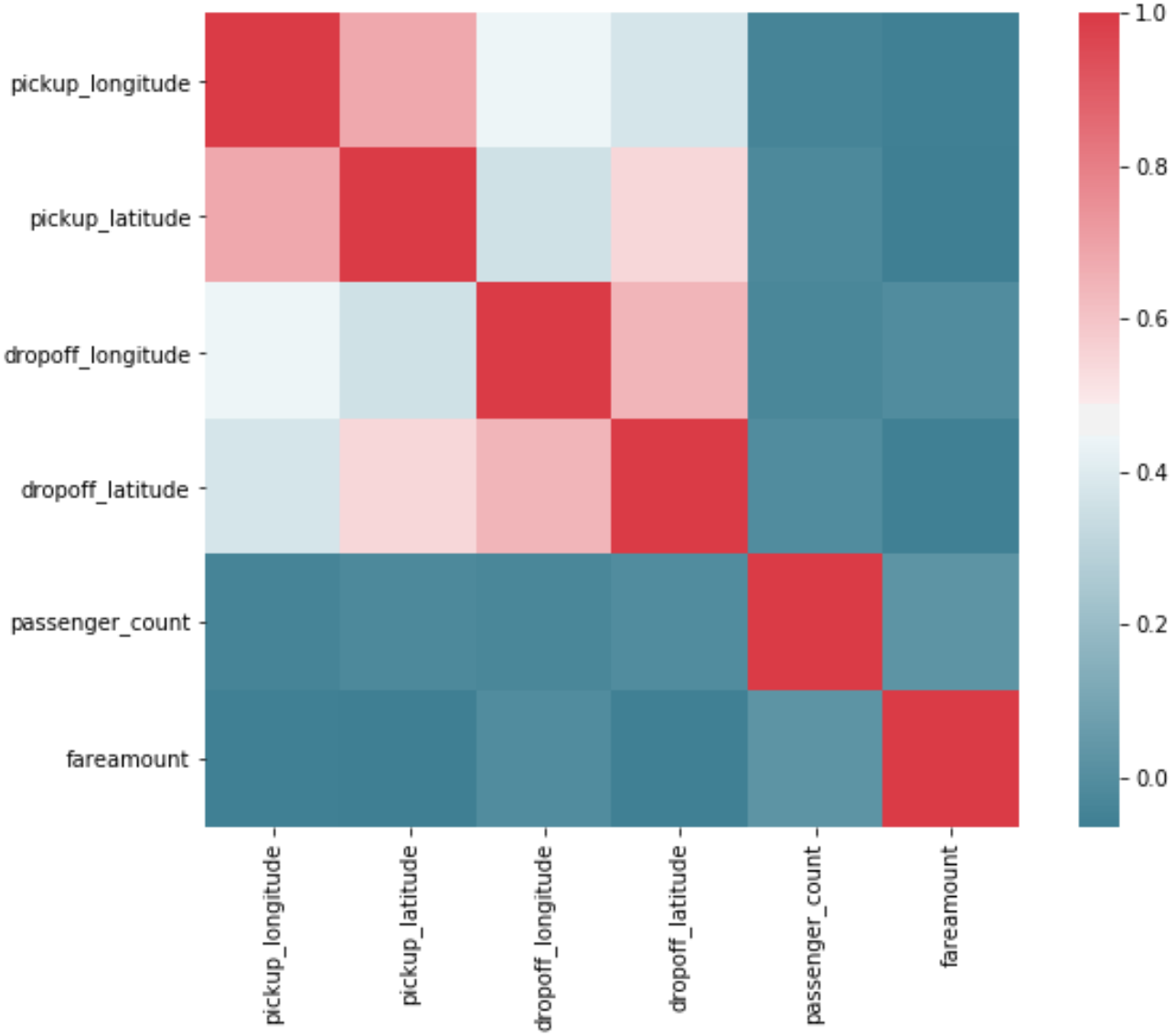
- Calculate outliers and inliers in each numeric variable.
- Calculate inter quartile range.
- Drop those observations which are beyond outliers and inliers.
- Checking again the shape of Dataset after removing outliers.

3) **Feature Selection**: Feature selection refers to selecting a subset of relevant features for use in model development. It refers to a subset of learning algorithms input variable upon which it should focus attention while ignoring the rest. It also reduces the dimensionality of the dataset.

The following set of steps are followed for feature selection:

- Generate Correlation Matrix.

- Plot heatmap using seaborn library to identify correlation between the various continuous variables:



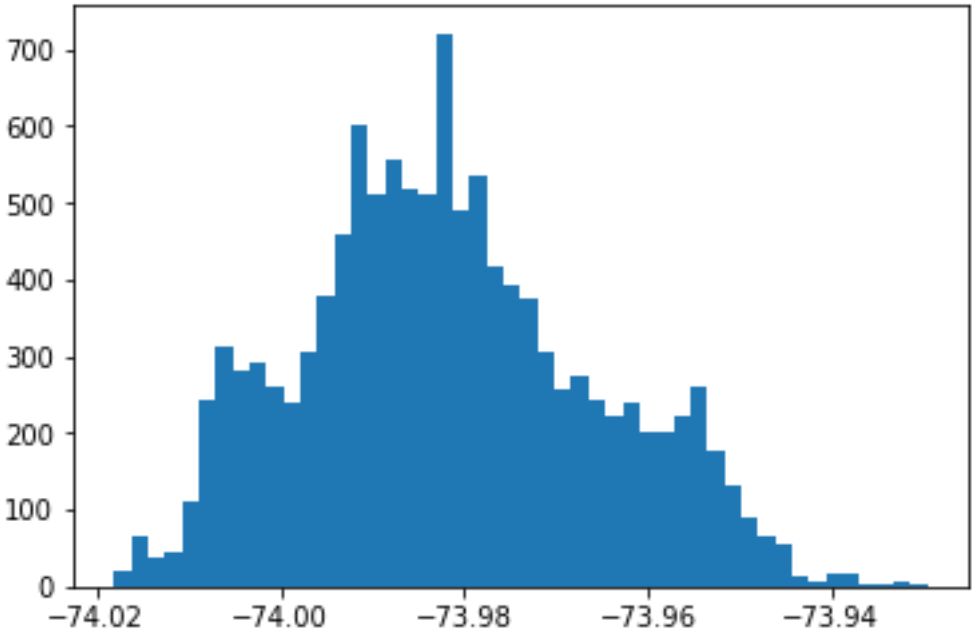
Heatmap for Feature selection

- As can be seen from the above correlation plot that there are no very highly correlated independent variables which may hamper the accuracy of our model.
- Since our Dataset contain categorical variable so there is need to do Chi square test for feature selection.

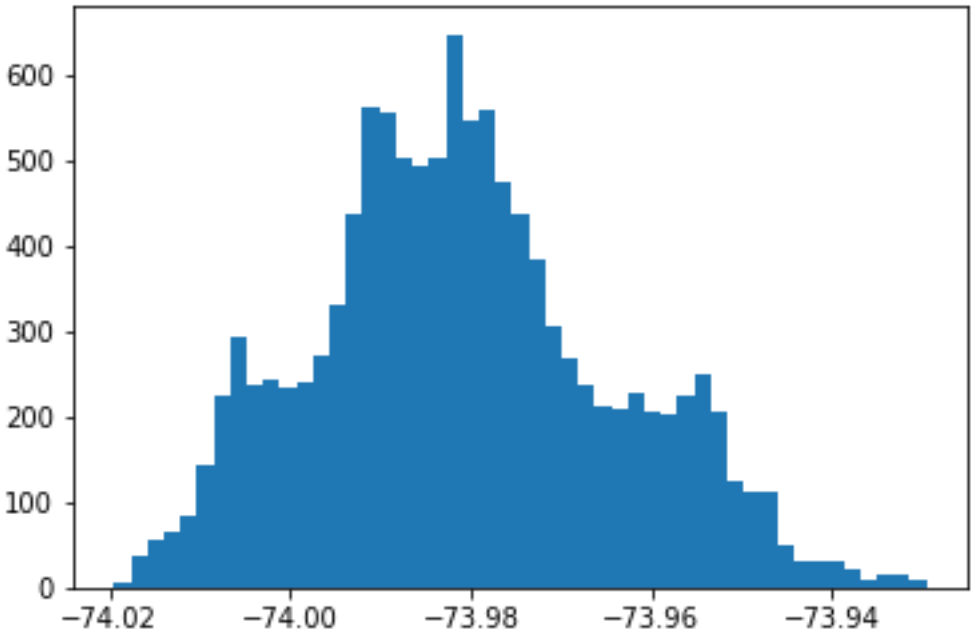
4) **Feature Scaling**: Feature scaling refers to the task of converting the various values of the column within the specified limits. Feature scaling is done to bring all the variable values within the limits so that no single variable has more influence over other variables while developing the model. It is the process of reducing the variations either within or between the variables. It brings all the variables into proper proportions with one another.

Feature scaling commonly uses two methods to compute.

- Normalization
- Standardisation



data distribution for pickup_longitude



distribution of data in dropoff_longitude

As can be seen from above histograms our variables are not exactly normally distributed so we have to adopt Normalization method of feature scaling.

MODEL DEVELOPMENT

Machine learning is, programming computers to optimize a performance criterion using example data or past experience. During the development of this model we applied various machine learning algorithms.

- Divide data into train and test for model development and to check accuracy and deduce which model is best.

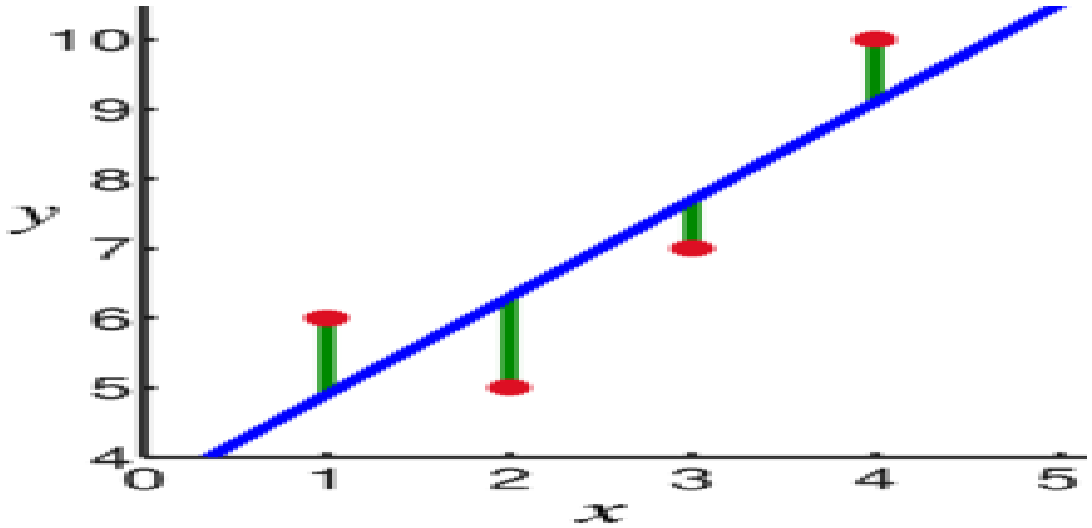
1) Linear Regression:

It is a Prediction Model. There are two types of Linear Regression

- Simple linear regression
- Multiple linear regression

It describes relationship among variables. The one simple case is where a dependent variable may be related to independent or explanatory variable.

- Equation expressing this relation is the line:
  $Y = b_0 + b_1x_1 + b_2x_2 + \ldots$
  Where b_0 is the intercept and x_1, x_2 are independent variables.
- For a given set of values we need to calculate values for b_0 and b_1.
- Look for a line which minimizes the sum of the residuals.



Linear Regression Line

- Import Libraries for Linear Regression.
- Build and train Linear Regression model using X_train and y_train.
- Predict model outcome using test data X_test.
- Regression Evaluation Metrics:

```
MAE: 0.14864871305584834
MAPE: 14.864871305584835
MSE: 0.03445213367764688
RMSE: 0.18561285967746652
```

2) **Decision Tree**:

Decision Tree is a predictive model based on a branching series of Boolean tests. It can be used for classification and regression both type of target variables. There are number of different types of decision trees that can be used in Machine learning algorithms. Decision tree is a rule. Each branch connects nodes with “and” and multiple branches are connected by “or”. They are extremely easy to understand by the business users. Decision trees build

some intuitions about your customer base. E.g. “are customers with different family sizes truly different?”

- Import Libraries.
- Model Development: Build and train model using X_train and y_train
- Predict model outcome using test data X_test
- Evaluation Metrics:

```
MAE: 0.14181602659490603
MAPE: 14.181602659490602
MSE: 0.03183425036933897
RMSE: 0.1784215524238565
```

3) **Random Forest**:

Random forest is an ensemble that consists of many decision trees. The term came from random decision forests that was first proposed by Tin Kam Ho of Bell Labs in 1995. This method combines Breiman's "bagging" idea and the random selection of features. It outputs the class that is the mode of the class's output by individual trees. It calculates results by evaluating mean for regression and mode for classification. This algorithm can be used for both classification and regression

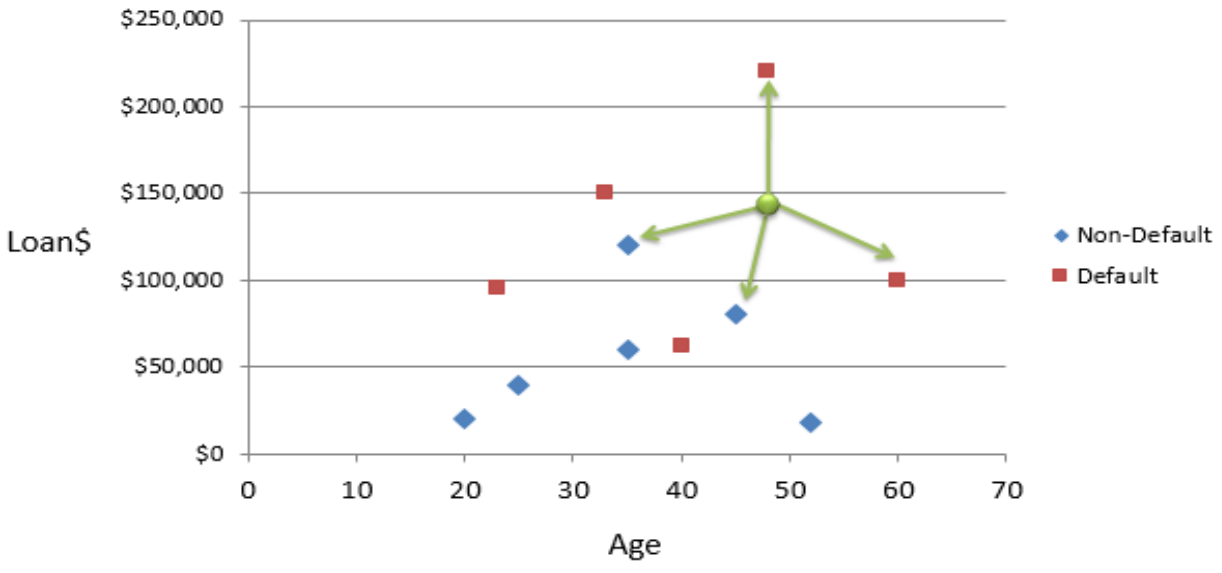
- Import Libraries
- Develop and train random forest model on train data using X_train and y_train
- Predict new test cases using X_test
- Evaluation Metrics:

```
MAE: 0.08484306361784265
MAPE: 8.484306361784265
MSE: 0.012912566667796574
RMSE: 0.11363347511977522
```

4) K Nearest Neighbour:

KNN stands for K-Nearest Neighbour. KNN is simple algorithm that stores all available cases and classifies new cases based on a similarity measure. It is a Supervised Machine Learning Algorithm. It can be used for both classification and regression. It is considered to be a Lazy Learning Algorithm.

Pick a number of neighbours you want to use for classification or regression. Choose a method to measure distances from test point. Keep a data set with records. For every new point, identify the number of nearest neighbours you picked using the method you choose. Let them vote if it is a classification or take a mean/median for regression



KNN Algorithm

- Import Libraries
- Develop and train KNN model on train data using X_train and y_train
- Predict new test cases using X_test
- Evaluation Metrics:

```
MAE: 0.08543941084437791
MAPE: 8.54394108443779
MSE: 0.013762374467597194
RMSE: 0.1173131470364562
```

SUMMARY OF MODELS

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. Based on MAPE we can deduce that Linear Regression is the best Algorithm to predict our target variable.

- DECISION TREE

  MAE for Decision tree = 0.1418
  MAPE for Decision tree = 14.18%
  MSE for Decision tree = 0.0318
  RMSE for Decision tree = 17842

- RANDOM FOREST

  MAE for Random Forest = 0.0848
  MAPE for Random Forest = 8.484%
  MSE for Random Forest = 0.01291
  RMSE for Random Forest = 0.1136

- LINEAR REGRESSION

  MAE for Linear Regression = 0.1486
  MAPE for Linear Regression = 14.864%
  MSE for Linear Regression = 0.03445
  RMSE for Linear Regression = 0.18561

- KNN K NEAREST NEIGHBOUR

  MAE for KNN Regression = 0.08543
  MAPE for KNN Regression = 8.543%
  MSE for KNN Regression = 0.01376
  RMSE for KNN Regression = 0.1173

PREDICTING TEST DATA

After forming the machine learning algorithm and tested it for test data we have fetch out from train dataset the most accurate method. Now we are ready to predict any external data given to us. Now since we are not given with any test data we create a sample dataset out of our given train data just to confirm the working of our model using the most appropriate algorithm we have formed in previous section. As we have deduced from MAPE values in our previous section that KNN gives most accurate model so we will use KNN to predict the target value.

Steps followed for predicting target variable in test data:

- Let us import the test dataset to predict our target variable
- Checking the shape and info of the test dataset
- Change the “passenger_count” variable from int to float type as it was in our train dataset so that it may not hamper our model in any way.
- Drop the "pickup_datetime” column from the Dataset as our model is not trained for it.
- Now we apply our model to KNN to predict our target variable “target_amount”.
- Our “target_amt” column gets added to as the last column of the dataset.
- Writing this new dataset with predicted variable as a CSV file.

The CSV file of the target variable is also included.

APPENDIX: Python Code

- import os
  import pandas as pd
  import numpy as np
  from fancyimpute import KNN
  import matplotlib.pyplot as plt
  from scipy.stats import chi2_contingency
  import seaborn as sns
  from random import randrange, uniform

- os.chdir("C:/Users/risha/Desktop/Cab_project/cab")

- os.getcwd()
- cabcount_train = pd.read_csv('train_cab.csv')
- cabcount_train.shape
- cabcount_train.info()
- cabcount_train.head(5)
- cabcount_train.describe()
- cabcount_train["fareamount"] =
  pd.to_numeric(cabcount_train["fare_amount"],
  downcast='float',errors="coerce")
- cabcount_train = cabcount_train.drop(["fare_amount","pickup_datetime"],
  axis=1)

- missing_val = pd.DataFrame(cabcount_train.isnull().sum())

- cabcount_train["passenger_count"] =
  cabcount_train["passenger_count"].fillna(cabcount_train["passenger_count"].
  mean())
- cabcount_train["fareamount"] =
  cabcount_train["fareamount"].fillna(cabcount_train["fareamount"].mean())
- cnames =
  ["pickup_longitude","pickup_latitude","dropoff_longitude","dropoff_latitude",
  "passenger_count","fareamount"]
- %matplotlib inline
- plt.boxplot(cabcount_train[‘dropoff_latitude'])
- plt.boxplot(cabcount_train['casual'])
- for i in cont_names:
  q75, q25 = np.percentile(cabcount_train.loc[:,i], [75 ,25])

```
iqr = q75 - q25
min = q25 - (iqr*1.5)
max = q75 + (iqr*1.5)
cabcount_train =  cabcount_train.drop(cabcount_train[cabcount_train.loc[:,i] < min].index)
cabcount_train = cabcount_train.drop(cabcount_train[cabcount_train.loc[:,i] > max].index)
```

- df_corr = cabcount_train.loc[:,cont_names]
  f, ax = plt.subplots(figsize=(10, 7))
  corr = df_corr.corr()
  sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool),
  cmap=sns.diverging_palette(220, 10, as_cmap=True),square=True, ax=ax)

- %matplotlib inline
- plt.hist(cabcount_train["dropoff_longitude"], bins='auto')
- for i in cnames:
  cabcount_train[i] = (cabcount_train[i] -
  np.min(cabcount_train[i]))/(np.max(cabcount_train[i]) -
  np.min(cabcount_train[i]))

- from sklearn import tree
  from sklearn.metrics import accuracy_score
  from sklearn.model_selection import train_test_split
  from sklearn import metrics
  from sklearn.model_selection import cross_val_score

- X = cabcount_train.values[:, 0:5]
  Y = cabcount_train.values[:,5]
  X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2)

- DT_model = tree.DecisionTreeRegressor(max_depth=2).fit(X_train, y_train)
- predictions_DT = DT_model.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, predictions_DT))
  MAPE = metrics.mean_absolute_error(y_test, predictions_DT)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, predictions_DT))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions_DT)))

- from sklearn.ensemble import RandomForestRegressor
- RF_model = RandomForestRegressor(n_estimators = 10).fit(X_train, y_train)

- RF_Predictions = RF_model.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, RF_Predictions))
  MAPE = metrics.mean_absolute_error(y_test, RF_Predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, RF_Predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, RF_Predictions)))

- from sklearn.linear_model import LinearRegression
- lm = LinearRegression()
  lm.fit(X_train,y_train)
- LR_predictions = lm.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, LR_predictions))
  MAPE = metrics.mean_absolute_error(y_test, LR_predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, LR_predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, LR_predictions)))

- from sklearn.neighbors import KNeighborsRegressor
- KNN_model = KNeighborsRegressor(n_neighbors = 3).fit(X_train, y_train)
- KNN_Predictions = KNN_model.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, KNN_Predictions))
  MAPE = metrics.mean_absolute_error(y_test, KNN_Predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, KNN_Predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, KNN_Predictions)))

- cab_test = pd.read_csv("test.csv")
- cab_test.shape
- cab_test.info()
- cab_test["passenger_count"] = cab_test.passenger_count.astype(float)
- cab_test.info()
- cont_names =
  ["pickup_longitude","pickup_latitude","dropoff_longitude","dropoff_latitude",
  "passenger_count"]
- # Nomalisation
  for i in cont_names:
  #print(i)
  cab_test[i] = (cab_test[i] - np.min(cab_test[i]))/(np.max(cab_test[i]) -
  np.min(cab_test[i]))

- cab_test = cab_test.drop(["pickup_datetime"], axis=1)
- cab_test['target_amt'] = KNN_model.predict(cab_test)
- cab_test.to_csv('cab_test_predict.csv',index=False)

Important Note

- INSTRUCTION TO RUN AND DEPLOY THE CODE IS WRITTEN WITH THE CODE ITSELF.
- ONLY THE FILE LOCATION SHOULD BE CHANGED IN OS.CHDIR FUNCTION ACCORDING TO THE USER FILE LOCATION.
- USER CAN SIMPLY USE SHIFT + ENTER TO GET ALL COMMAND RUN.
- FILES INCLUDED ARE:
    1. PYTHON CODE FILE
    2. PROJECT REPORT IN BOTH WORD AND PDF FORMATS
    3. R CODE FILE
    4. DATASET WITH TARGET VARIABLE IN .CSV FORMAT


Thank you