|  |
| --- |
| EDWISOR |
| FARE AMOUNT PREDICTION REPORT USING PYTHON |
| PYTHON REPORT |

|  |
| --- |
| BY: RAJAT AGRAWAL  16-Jul-19 |

**CONTENT**

1. LIBRARY INSTALLATION
2. DATA EXPLORATION
3. DATA PREPROCESSING

* *MISSING VALUE ANALYSIS*
* *OUTLIER ANALYSIS*
* *DATA FEATURING*
* *DATA SCALING*

1. DATA MODELLING

* *LINEAR REGRESSION*
* *DECISION TREE*
* *RANDOM FORREST*
* *K-NEAREST NEIGHBOURS*

1. CONCLUSION
2. PREDICTION OF OF TEST DATA
3. SAVED FILES DURING PROCESS
4. IMPORTANT NOTE

**ABSTRACT**

THE DATA IS GIVEN TO US HAS CONTINUOUS TARGET VARIABLE WHICH IS FARE AMOUNT SO ALL THE ALGORITHM USED ARE REGRESSIVE IN NATURE. THE ALGORITHM USED FOR THIS MODEL ARE LINEAR REGERSSION, DECISION TREE, RANDOM FOREST, KNN.

THIS REPORT GIVE US AN IDEA ABOUT HOW TO PREDICT A DATA FROM A GIVEN DATA WHICH IS INCONSISTENT IN NATURE. SO FIRST WE DO DATA EXPLORATORY WORK WHICH WILL AS NAME SUGGEST EXPLORE THE DATA . IN EXPLORING THE DATA WE WILL EXPLORE ALL THE COMPONENT OF DATASET.

AFTER THAT WE WILL DO PREPROCESSING WHICH MAKE THE DATA IN THE FORMAT THAT MODELLING METHOD CAN BE USED ON IT AND THEY GIVE THE CORRECT PREDICTION.

WHEN PREPROCESSING IS DONE OUR DATA IS READY TO FORM A MODEL ON IT SO THE BASIS OF CHARACTER OF TARGET VARIABLE VARIOUS METHOD OF MODELLING CAN BE USED.ONCE THE MODEL IS READY ON BASIS OF ERROR BEST MODEL CAN BE DECIDED .

NOW WE HAVE DECIDED OUR BEST MODEL AFTER THAT ON THE BASIS OF TEST DATASET WE CAN PREDICT OUR TARGET VARIABLE AFTER USING BEST METHOD OF PREDICTION .

**ABOUT**

A question has given a two dataset first is a train dataset and second is a test dataset . this train dataset contain 7 columns and 16067 rows so on the basis of this dataset we have to predict a model and using that model predict a fare amount of test data which contain all the same variable as train data contain except for the fare amount variable.

The data contain some null value also . as follow

Fare amount has 25

Passenger count 55

Pickup datetime 1

**DATA EXPLORATION**

We have form a python file on jupyter notebook in which various data exploration techniques have been used.

In first step we import various libraries which will use during the course of project. These libraries contain various inbuilt function. Various libraries are as follows:

Import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import os

from fancyimpute import KNN

from random import randrange, uniform

from scipy.stats import chi2\_contingency

%matplotlib inline

After importing libraries we will upload train dataset . start analyzing the datset . we can use describe fuction on dataset which will gives median at 50%, 75%, 25% , mean , number of counts etc.

We will use info function to get the data type of various variable.

During analyzing we have seen a various observation:

1. Datset contain null value which we can see by using a function .isnull().sum().
2. It contain date and time together in one column and also contain UTC string with every date and time observation . This UTC string has no use so first we split that utc string from date and time and than remove it.
3. After that form a separate columns contain date and time independently. And at last remove that column contain both of them together.
4. Now we will convert a data type into required form like fare amount data type will convert into numeric from object type.
5. We can use heat map to see no of missing values

Fare data type is as follow:

<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.

**DATA PREPROCESSING**

***MISSING VALUE ANALYSIS:***

Dataset contain the null value as follow:

1. fare\_amount 25
2. pickup\_datetime 0
3. pickup\_longitude 0
4. pickup\_latitude 0
5. dropoff\_longitude 0
6. dropoff\_latitude 0
7. passenger\_count 55
8. datetime 1
9. date 1
10. time 1

dtype: int64

now based on the datatype we can used various method to imputation

As fare\_amoumt is numeric in nature and has 25 missing value we can use three method to impute and used the method which will most closer value to the true value. To check the best method we will fill make the known value null and then based on all the three method found out value on that null location after finding all the three value. The value which comes out closet to true value will be used.

So three method used for imputation for continuous value are mean, median, KNN imputation.

For categorical variable method used for imputation are mode and KNN imputation.

So to check correct method will will remove the following value

fare["fare\_amount"].loc[1] =np.nan

fare["passenger\_count"].loc[27] = np.nan

fare["date"].loc[100]=np.nan

fare["time"].loc[50]=np.nan

and after using various method we get the following values:

# for fare amount true value is 16.9

#mean value=15.014

#median value=8.5

#knn value =7.9130

# SO HERE WE CHOOSE MEAN FOR FARE AMOUNT

# for passenger count value is 3

#mean value =2.625

#median value=1

#mode value=1

#knn value =2.44

# HERE WE CHOOSE mode VALUE FOR PASSENGER COUNT BECAUSE PASSENGER COUNT CAN NOT COMES IN DECIMAL

# for date =(2014, 11, 12)

#mode value=2011-06-13

#knn=2014-11-12

#for time(9, 9, 21)

#mode value=19:43:00

#knn=(9, 9, 25)

# so for fare amount nearest value is from mean and for passenger count nearest value is from knn

# DATE AND TIME HAS ONLY ONE MISSING VALUE WE CAN IGNORE IT OR IMPUTE BY ANY METHOD.

We will form a new dataset using a missing value and saved it as miising\_percent.csv.

To get the KNN value for categorical variable we have to convert the categorical variables into codes as knn only take numerical value after conversion knn imputation is used . after imputation convert them from codes to original value.

After knn imputation save data the data without any missing value as “fair\_train” .

***OUTLIER ANALYSIS***

Outliers are values which are inconsistent in nature which make data inappropriate . Which simply check that outlier using the box plot and the values locate at the extreme of the box plot are outlier we can also used the loop to get the outlier.

So once we get outlier we can remove them from data or we can put null value on the place of it and the impute null value by the various method we have learnt in previous section.

Outlier can be shown mathematically as:

Here q75: it is value of a variable at 75 percentile value

q25:it is value of a variable at 25 percentile value

iqr = q75 - q25

min = q25 - (iqr\*1.5)

max = q75 + (iqr\*1.5)

outlier is a observations can comes in data due to incorrect experiment which when used to form the model can effect the models predictability criteria. Sometimes they are correctly filled due to some exceptions in dataset but in that case also they does not give the correct trend.

Loop used for outliers is as follow:

for i in cnames:

print(i)

q75, q25 = np.percentile(fare.loc[:,i], [75 ,25])

iqr = q75 - q25

min = q25 - (iqr\*1.5)

max = q75 + (iqr\*1.5)

print(min)

print(max)

fare = fare.drop(fare[fare.loc[:,i] < min].index)

fare = fare.drop(fare[fare.loc[:,i] > max].index)

here we have saved the outliers value as null and saved them as separate dataset of missing value named as

“missing outlier perc.csv”. we have also saved the finel dataset after removing the outlier as “fare outlier analysis.csv”. this dataset is free from outlier and missing values.

***Feature selection***

It is method in which we remove the continuous variable which are dependent on each other so that no two variables denoted same variation . this method used the concept of collinearity to find the dependency between the independent variable . most of the time we used heat maps to find there collinearity. If collinearity between two variables is equal to 1 or -1 we remove one variable from them.

For categorical variable we used chi-square test to find dependency between target and independent variable. So if the p value due to chi-square test between target variable and continuous variable is less than .005 it shows that two variables are dependent than we take that variable for modeling but those variables whose value comes out to be greater than .05 will remove from dataset.

In heat map the variables which shows perfectly red are dependent and those which are perfectly blue are independent.

So in python script we can see that no two variables in heat map shows perfectly dark colour so all them can be assume to be independent.

Heat map script for dataset is as follow:

#Set the width and height of the plot

f, ax = plt.subplots(figsize=(7, 5))

#Generate correlation matrix

corr = df\_corr.corr()

#Plot using seaborn library

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=True) ,square=True, ax=ax) .

We can see that after using chi square script p value of both the categorical variable date and time is greater than .05 we will remove them both.

Before using chi square we have to convert data type of independent variable into “object” type as it only take categorical variable

Chi square scrip is as follow:

for i in catnames\_select:

print(i)

chi2, p, dof, ex = chi2\_contingency(pd.crosstab(fare['fare\_amount'], fare[i]))

print(p)

p values are comes as:

date :0.6749450185820663

Time : 1.0

*Feature scaling*

It used when modeling method used for prediction require distance between two variable as algorithm. Variations between two variable plays an important role so make them equally variate we will use standardization and normalization for each continuous variable.

Standardization is prefer when distribution of target variable is normal in nature otherwise normalization is used:

Formulae:

Standardization:

http://ci.columbia.edu/ci/premba_test/c0331/images/s6/5836240103.gif

Here µ is mean

Normalisation :

https://www.statisticshowto.datasciencecentral.com/wp-content/uploads/2015/11/normalize-data.png

We will used feature scaling according to the algorithm we used:

For example: if we used linear regression or KNN we used normalization as our data is skew symmetric but when we used random forest or decision tree there is no requirement of feature scaling.

Loop used in this script is:

#NORMALISATION LOOP

for i in knnnames:

print(i)

fare\_knn[i] = (fare\_knn[i] - fare\_knn[i].min())/(fare\_knn[i].max() - fare\_knn[i].min())

**PREDICTING OUR MODEL**

Model is start to build when all the work on data is completed and we have get the data in neat and clean format without any missing value, without any outlier , no dependency of independent variable with each other. Data is in standardize form if require for algorithm.

THE DATA IS GIVEN TO US HAS CONTINUOUS TARGET VARIABLE WHICH IS FARE AMOUNT SO ALL THE ALGORITHM USED ARE REGRESSIVE IN NATURE. THE ALGORITHM USED FOR THIS MODEL ARE **LINEAR REGERSSION, DECISION TREE, RANDOM FOREST, KNN.**

All the machine learning algorithm used following steps to

Get the prediction:

1. first we remove zero fare\_amount value from dataset as it will give mape value infinite. We will use mape value to compare the data as it is easy to understand.
2. we copied the original data so that we can used various respective transformation on that copied data
3. In all the algorithm before starting anything first we import their respective library .
4. Then we import the libraries to split the data into training and test set . we usually take 40% of data as test set. command used for this purpose is

“From sklearn.model\_selection import train\_test\_split”

1. Now we set the train data into algorithm using respective command as algorithm.
2. Than we predict the data using .predict function and putting the test independent variable.
3. After all this work we import error metric or sometime used simple loop to *MAPE*. Library used for finding error between true test target value and predicted value which we have found using test values is “from sklearn import metrics”. Various methods of error are as follow:

* 'MAE:', metrics.mean\_absolute\_error(y\_test, prediction\_dt)
* 'MSE:', metrics.mean\_squared\_error(y\_test, prediction\_dt)
* 'RMSE:', np.sqrt(metrics.mean\_squared\_error(y\_test, prediction\_dt))

After all that work we can plot a graph scattering graph between predicted and test y value it should form a graph of points forming 45 degree with x axis

We will also form a distribution curve using (y\_test-predicted value). This distribution should have max point at zero . should be normally distributed about zero.

***Linear regression:***

This method used min least square distance method to form a prediction.

Library used: from sklearn.linear\_model import LinearRegression

import statsmodels.api as sm: this library gives summary of whole the model with their coefficient and p value at one dataset at a time which is easy to understand.

Coefficient p value

pickup\_longitude -12.558990 0.00

dropoff\_longitude 17.092550 0.00

dropoff\_latitude -22.231644 0.00

**here we can see that fare\_amount is depends on pickup\_longitude , dropoff\_longitude, dropoff\_latitude as only its p value is less than .5**

After forming the model we will predict the values from test data used error metrics on that predicted value so errr we get are as follow:

**MAE: 2.98113587130155**

**MSE: 14.195154659577986**

**RMSE: 3.7676457715101064**

**MAPE:** **41.39160318975558%**

***Decision tree:***

Decision tree form a tree after splitting at every point using mse

Library used:

from sklearn.tree import DecisionTreeRegressor

criterion="mae", max\_depth=9

after using various criterion and depth we get different error as follow:

#for max\_depth =9

#form= mse split error is 29.69%

**#for mae split error is 28.32**

#for friedman\_mse error is 29.64

# for max\_depth = max ,error for mae splitte is 31.67

So we used mae split with max depth equal to 9

Different errors are:

**MAE: 2.4257908348218398**

**MSE: 11.257192803915558**

**RMSE: 3.3551740348178005**

**MAPE:** **28.191118154166446%**

***RANDOM FORREST***

Random forest used bagging and boosting for model development

Library used:

from sklearn.ensemble import RandomForestRegressor

n\_estimator used: 500

y\_train must be change into integer as random forest can’t take float value.

Different error are as follow:

**MAE: 1.5802160316722333**

**MSE: 5.223044938490796**

**RMSE: 2.2853982012968324**

**MAPE:** **18.340839699434994%**

for different value of estimator erroe are as follow:

for max depth of 9

#for 100 , 25.816

#50, 26.11

#10, 26.37

#100 for max depth 18.59

#50 18.97

#**500 , 18.35**

***K-NEAREST NEIGHBOUR METHOD****:*

This method used Euclidean distance method to predict the value for categorical variable used mode of K nearest neighbor and for continuous variable use mode of K nearest neighbor.

We form a graph between the different K values and error to get most appropriate value of K it should form elbow curve least point of elbow curve gives us K value.

But for the dataset curve form is inverse of elbow curve.

Library used:

from sklearn.neighbors import KNeighborsRegressor

after that we will used normalization loop as knn require distance between two observations.

Error form as follow:

**MAE: 0.09427756931887206**

**MSE: 0.017612481788999164**

**RMSE: 0.13271202578892075**

**MAPE:** **24.807500465160505%**

**These error are comes very less as we have done a normalization on our dataset. So we can see that *MAPE* Isthe best method to compare thedifferent algorithm. It is easy to understand for any layman person also.**

**Without normalization** error are as follow:

**MAE: 2.0750292941435204**

**MSE: 8.553579672701744**

**RMSE: 2.9246503505037564**

**MAPE:** **24.512647860057548%**

After draw a curve between error rate and different k value we get to know that most appropriate value for k is 1

**CONCLUSION**

**SUMMARY:** after forming all the algorithm least MAPE due to LINEAR REGRESSION: **41.39160318975558%**

DECISION TREE **: 28.191118154166446%**

RANDOM FOREST : **18.340839699434994%**

KNN : **24.512647860057548%**

***FROM ALL THESE ERROR WE CAN FIGURE IT OUT THAT MOST APPROPRIATE MACHINE LEARNING ALGORITHM USED TO PREDICT THE DATA IS RANDOM FOREST AS IT GIVES LEAST MAPE ERROR***

***LR>DT>RF>KNN ERROR VALUES***

***WE CAN USED ANY ERROR METRICS TO CHOOSE OUR MACHINE LEARNING ALGORITHM, ERROR SHOW SAME TREND AS LR>DT>RF>KNN ERROR VALUES.***

**PREDICTION OF GIVEN TEST DATA**

The test data given to us contain all variables of train data except fare amount variable. Our task is to predict the fare amount value using the most appropriate algorithm we have formed in previous section.

As we have developed in our previous section that Random Forest gives least error so we will use random forest to predict the fare amount.

1. So to do that first we upload dataset into python
2. Now in this dataset remove pickup\_datetime column as it is not readable by our random forest model
3. Now form the different dataset of name pickup\_datetime so that after prediction we can join this dataset into main dataset.
4. So now convert remaining dataset into flaot datatype so that random forest model can read it.
5. Now use the command rfr.predict(fare\_test) to predict from given dataset . rfr is a function we have form earlier from random\_forrest\_regressor() function
6. This predict array will save as predict\_test. Form the dataframe of the array using panda of name df
7. After all this work join this df and pickup\_datetime column with remaining set.
8. Save this predicted dataframe as "test prediction result.csv" into hard disk
9. Now we have get a dataset having a seven number of column of which one is a predicted target variable name “fare\_amount”. Which gives us desired result.

**SAVED FILES DURING PROCESS :**

* **Miising\_percentage.csv**: missing percentage of initial dataset given
* **fare\_train.csv**:train set form after imputing missing values without categorical variable
* **missing outlier perc.csv:**dataset contain number of outliers present in dataset given after missing value imputation
* **fare outlier analysis.csv:**dataset form after removing all the outliers from train dataset
* **test prediction result.csv**: gives the finel result of test data with predictions

**IMPORTANT NOTE**

INSTRUCTION TO RUN AND DEPLOY THE PYTHON CODE IS WRITTEN WITH THE CODE ITSELF.

ONLY THE FILE LOCATION SHOULD BE CHANGED IN OS.CHDIR FUNCTION ACCORDING TO THE USER FILE LOCATION.

ALL THE SAVED FILE DURING THE PROCESS ARE GIVEN WITH CODE.

USER CAN SIMPLY USE SHIFT + ENTER TO GET ALL COMMAND RUN.

