INSTITUTE FOR ADVANCED
COMPUTING AND
SOFTWARE
DEVELOPMENT
AKURDI, PUNE

Documentation On

"Regression Analysis on Medicare Spends !"

e-DBDA Sept 2020

Submitted By:

Group No: 07

Dhanashree Prakash Deore 1511
Minal Arvindrao Yawale 1557

Mr. Prashant Karhale
Centre Coordinator

Mr. Akshay Tilekar
Project Guide

Contents

Figure List

Acknowledgement

First and Foremost, we thank to almighty God for giving us the support, strength, positive spirit and talent to do this project.

“The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible”. “This work is the result of inspiration, motivation, knowledge, interest, support, guidance, cooperation and efforts by many people at different levels. We are indebted to all of them”. We would also like to take this opportunity to acknowledge the valuable contributions made by our family members by supporting and motivating us in every walk of life. We are thankful to Mr. Prashant Karhale, Centre Co-ordinator IACSD CDAC, Akurdi Pune for providing the opportunity, infrastructure and facilities for entire work. We would like to express our great appreciation to our project Guide Mr. Akshay Tilekar, Internal Project Guides Mr. Rahul Pund, and Mr. Manish Bendale for their valuable and constructive suggestions during the planning and development of this project. We also thank all staff members from the IACSD who in some way or other Helped us in completion of this project. We cannot conclude our acknowledgement without expressing our thanks to our friends who helped us directly or indirectly during the course of this project. Feedback for improving the contents of the report would be more than welcome.

1. Introduction

1.1 PROBLEM STATEMENT:

Regression Analysis on Medicare Spends!

1.2 Abstract

Medical expenses are one of the major recurring expenses in a human life. It's a common knowledge that one life style and various physical parameters dictates diseases or ailments one can have and these ailments dictates medical expenses. According to our dataset prospective payments system based on diagnosis related groups (DRGs) by many third-party payers represents an important shift in hospital payment policy. In this study, we aim to find a correlation between average medical expenses and different factors, and compare them. Then we use the prominent attributes as predictors to predict medical expenses by creating regression models and comparing them. Regression analysis refers to the method of studying the relationship between independent variable and dependent variable. All regression model that corresponds to the practical situation is proposed in the project, which is to set up regression model based on practical problem and then to implement the following with the help of the latest and most popular Python3.8. The features of pure object-oriented, platform independence and concise and elegant language. So, we will call the corresponding library function to predict the spends of Medicare.

Keywords: Regression Machine Learning algorithms, Python, Medicare expenses.

1.3 Aims & Objectives

The goal of this project is to use the trained models to predict Medicare charges. The training of the models will be done using a bunch of different Regressor ML models and after the training is done the Regressor ML models will be compared based on their R2 score and RMSE error and the best model will be selected which will then be used to make predictions.

1.4 Initial functional requirement will be:

- Understanding the data to get the proper Insights.
- Selecting the algorithm meeting requirement.
- Testing the algorithm with test sets.
- Analyse the results and if necessary, do the changes in algorithm.

1.5 Major observation from the data:

- Our Dataset "healthcare_census.csv" is of USA (country). It contains Medicare details.
- Target variable of our dataset is Average Medicare Payments. In which all data in numeric form.
- DRG- According to our dataset prospective payments system based on diagnosis related groups (DRGs) by many third-party payers represents an important shift in hospital payment policy.
- It contains columns as Insured and Non-insured males and females of age group 18-25.
- It has Hospital, Medicare provider, state details**.**

2. Overall Description

2.1 Workflow of Project:

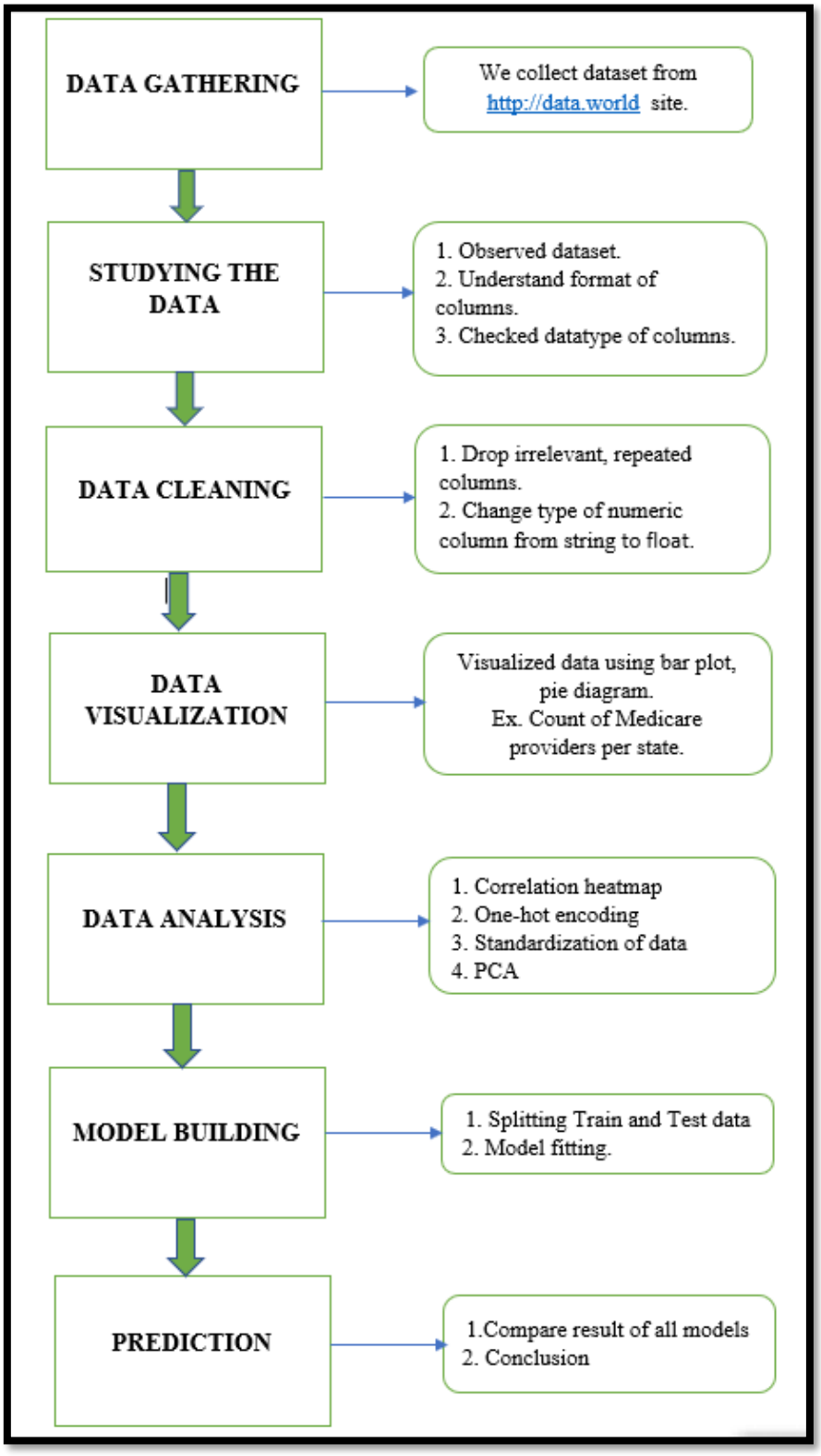

Fig. 1 Work Flow of Project

2.2 Gathering Data:

Data Gathering is the first step. In this step, we need to identify the different data sources, as data can be collected from various sources such as files, database, internet, or mobile devices. It is one of the most important step. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

2.3 Data exploration:

It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data. A better understanding of data leads to an effective outcome.

2.4 Data Cleaning:

The data can have many irrelevant columns, many columns have numeric data as string type. Hence, we need to convert it into float type and renamed the columns name in our dataset.

2.5 Data Visualization:

In this we done data analysis using different types of plot and pie diagram. With the help this we find top Medicare provider name, state. Count of insured males and females, State wise total Medicare payments.

2.6 Data Analysis:

This step is to build a machine learning model to analyse the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the supervised machine learning techniques such as Regression, then build the model using prepared data, and evaluate the model.

3. Data Visualization

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

Following are some plots we used to extract some useful information

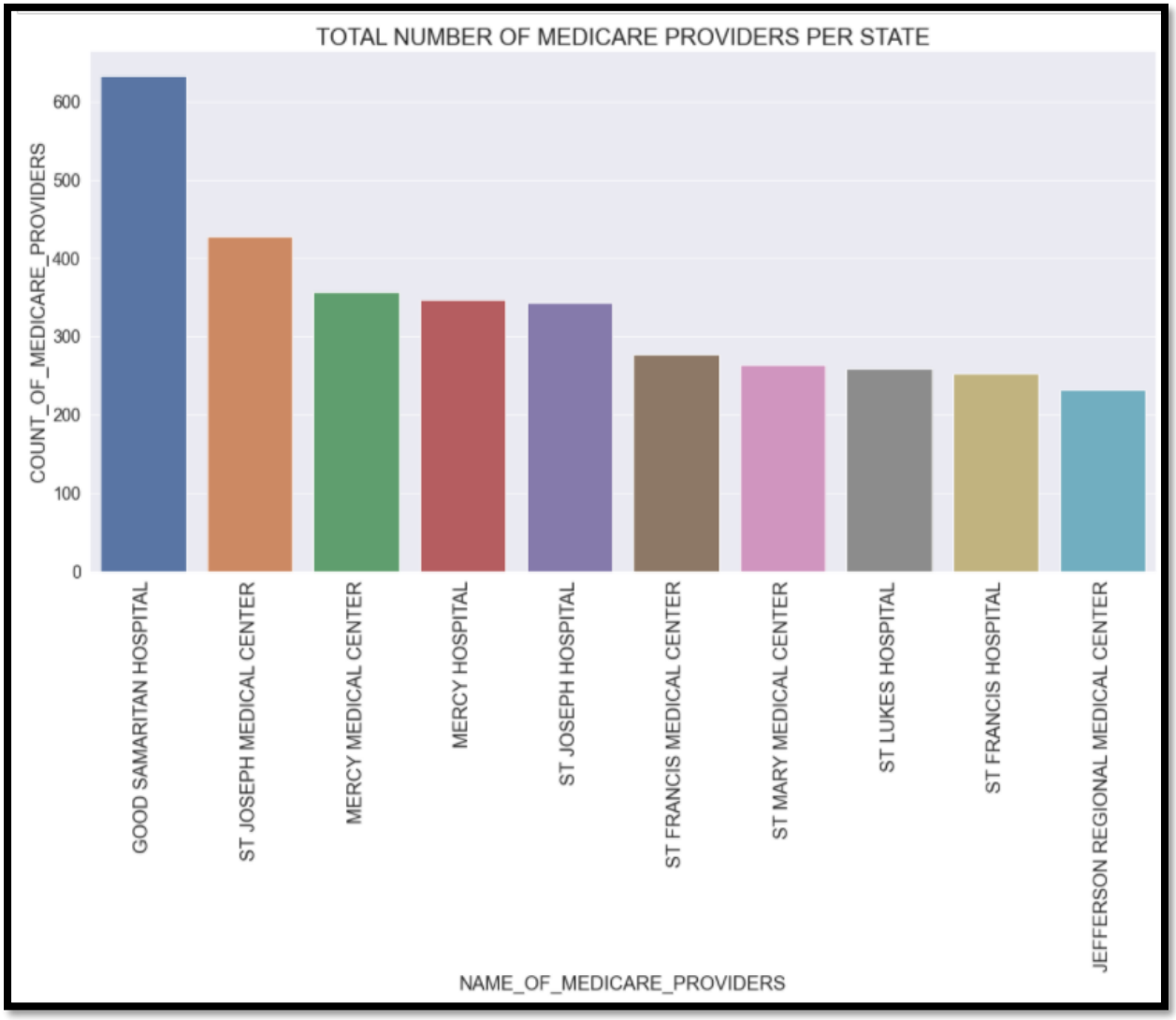

Fig.2. Bar chart shows Total no of Medicare providers per state

In this we see that GOOD SAMARITAN HOSPITAL and ST JOSEPH MEDICAL CENTER both are the top Medicare providers in state. Total number of Medicare provided by GOOD S AMARITAN HOSPITAL is 633 and ST JOSEPH MEDICAL CENTER is 427.

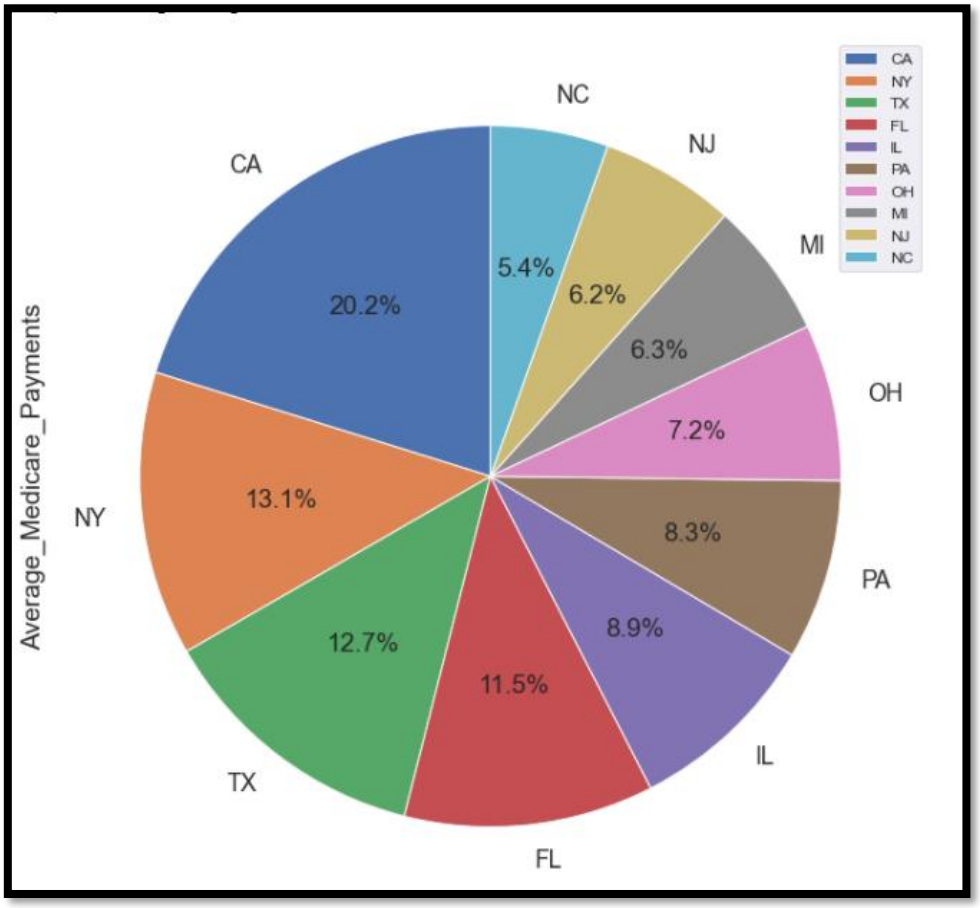

Fig.3 Pie chart of Top Medicare providers with their Average Medicare payments.

The pie chart shows that, Top 10 states with their Average Medicare Payments. Total Medicare payment of state:

1. California (CA): 20.2%
2. New York (NY): 13.1%.

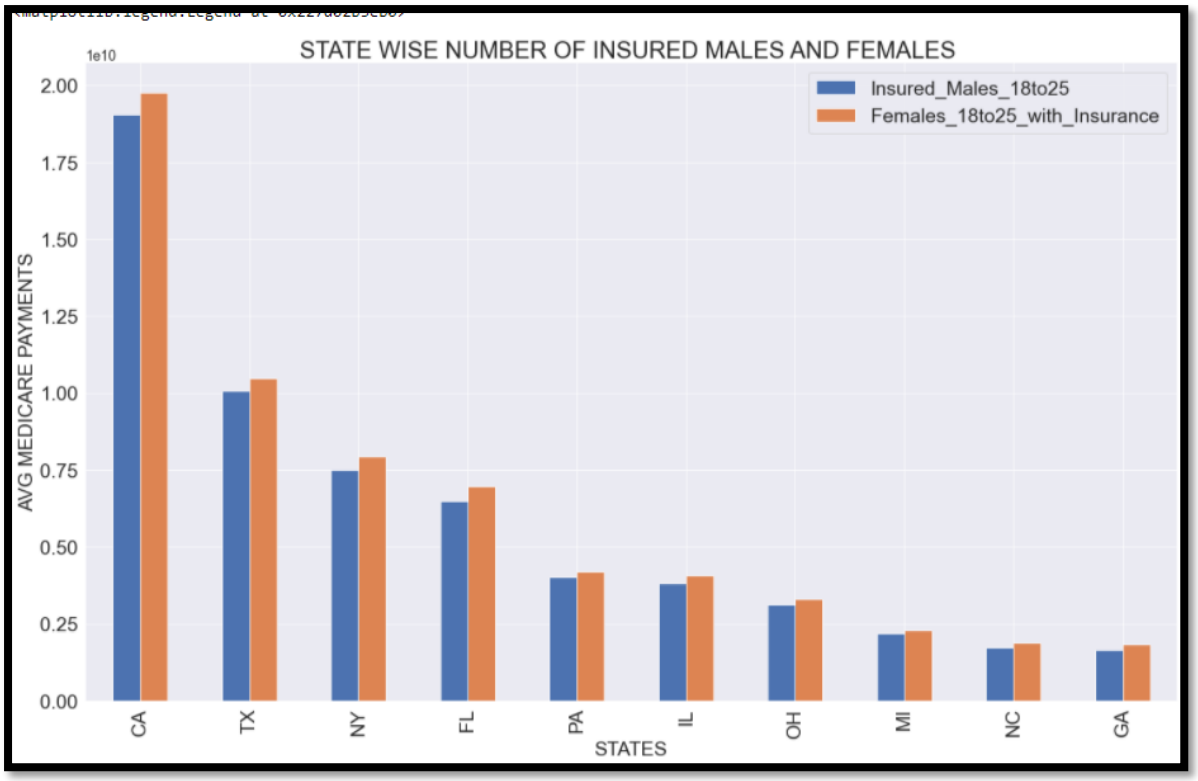

Fig.4 Bar graph of states wise number of Insured Males and females

The bar graph shows, California (CA) state is having highest number of insured males and females than other states, but in that all states have number of insured females are more than males.

4. Data Analysis

Now the next step is pre-processing of data for its analysis. In this, we find Correlations between all independent variables in our dataset.

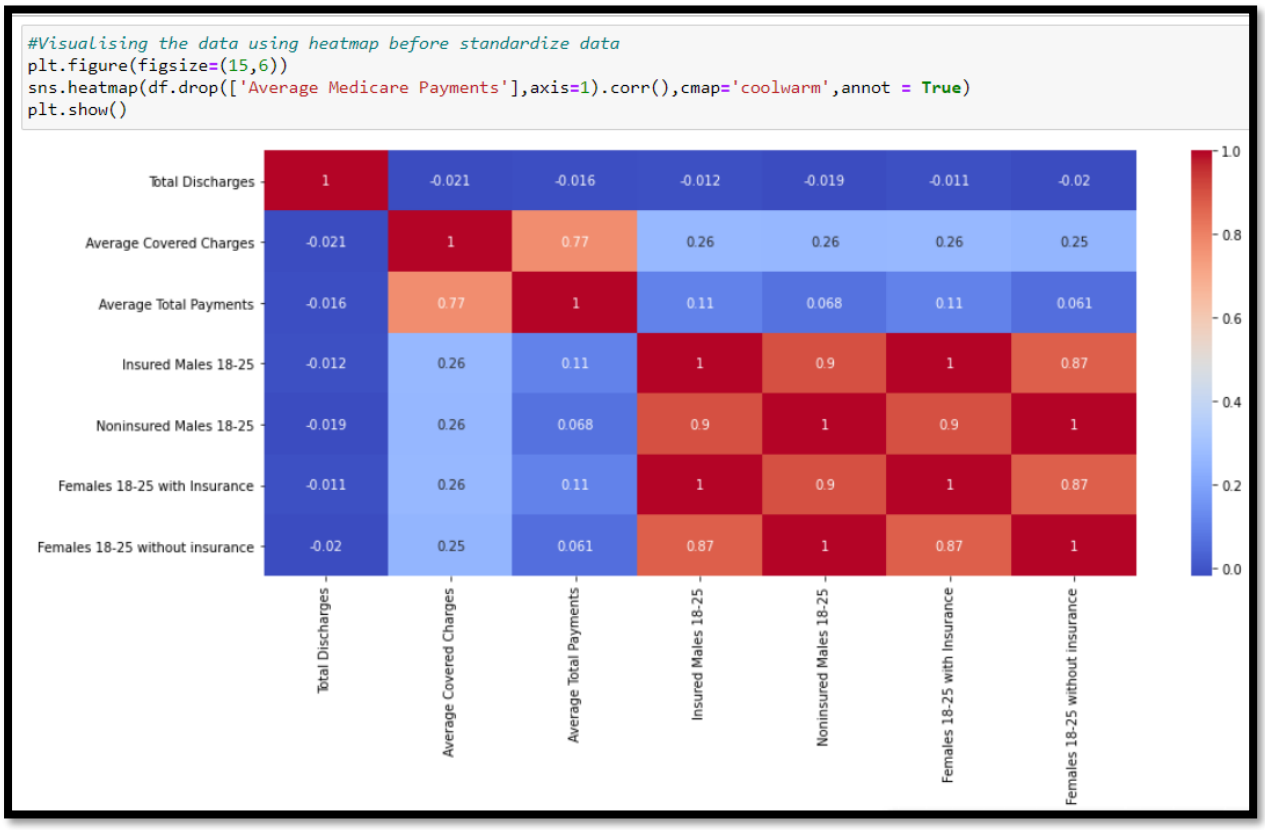

Fig.5. Correlation between Independent Variables

After that we applied One Hot Encoding, standardization, Principal Component Analysis (PCA). Explain as below

4.1 One Hot Encoding:

It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

4.2 Standardization:

Many machine learning algorithms perform better when numerical input variables are scaled to a standard range. Standardization scales each input variable separately by subtracting the mean (called centring) and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one.

4.3 Principal Component Analysis (PCA):

An important machine learning method for dimensionality reduction is called Principal Component Analysis. It is a method that uses simple matrix operations from linear algebra and statistics to calculate a projection of the original data into the same number or fewer dimensions.

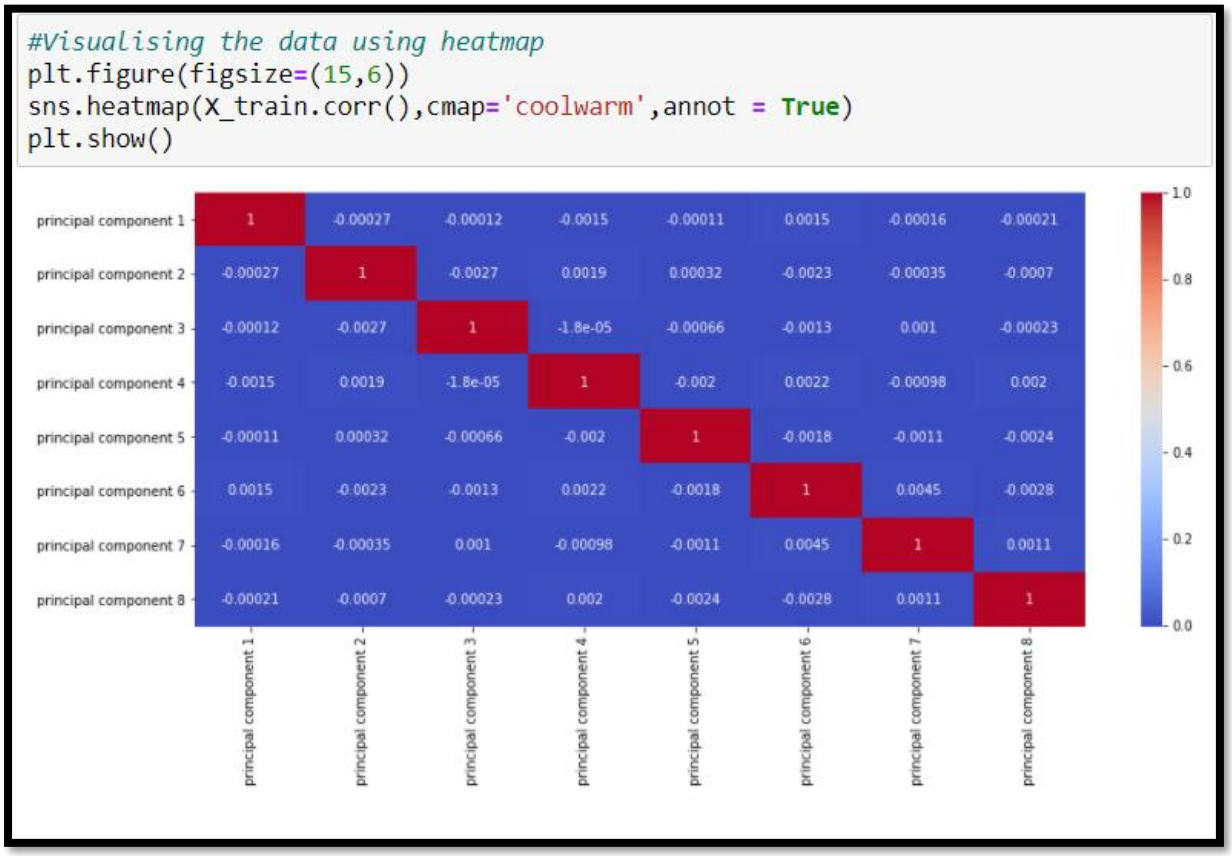

Fig.6 Correlation heatmap

Above heatmap generated after completed one hot encoding, standardization, PCA.

5. Modeling

5.1. Train/Test split:

One important aspect of all machine learning models is to determine their accuracy. Now, in order to determine their accuracy, one can train the model using the given dataset and then predict the response values for the same dataset using that model and hence, find the accuracy of the model. A better option is to split our data into parts: first one for training our machine learning model, and second one for testing our model.

Split the dataset into two pieces: a training set and a testing set.

5.2 Train Dataset:

- Used to fit the machine learning model.
- We train our model to improve its performance for better outcome of the problem.
- We use datasets to train the model using various machine learning algorithms.
- Training a model is required so that it can understand the various patterns, rules, and, features.

5.3 Test Dataset:

- Used to evaluate the fit machine learning model.
- Once our machine learning model has been trained on a given dataset, then we test the model.
- In this step, we check for the accuracy of our model by providing a test dataset to it.
- Testing the model determines the RMS error of the model as per the requirement of project or problem.
- The R-squared score is a good indicator of how well data set is fitting the model.

5.4 Prediction error plot:

A prediction error plot shows the actual targets from the dataset against the predicted values generated by our model. This allows us to see how much variance is in the model. Data

scientists can diagnose regression models using this plot by comparing against the 45 degree line, where the prediction exactly matches the model.

- bestfitbool, default: True Draw a linear best fit line to estimate the correlation between the predicted and measured value of the target variable.
- identitybool, default: True Draw the 45 degree identity line, y=x in order to better show the relationship or pattern of the residuals.
- Returns :R2 score in float value.
- Import:

  !pip install yellowbrick

  !pip install -U yellowbrick

6. Regression Models

6.1 Linear Regression:

It is used to estimate real values based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation

Y= a *X + b.

In this equation:

- Y – Dependent Variable
- a – Slope
- X – Independent variable
- b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Linear Regression is mainly of two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent variable. And, Multiple Linear Regression (as the name suggests) is characterized by multiple (more than 1) independent variables. While finding the best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.

```
#from sklearn.linear_model import LinearRegression

lr = LinearRegression()

# fit the model using the training data and training targets
lr.fit(X_train, y_train)

y_pred = lr.predict(X_test)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_test, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_test, y_pred))
print("R-squared score : ", r2_score(y_test, y_pred))

#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-lr.score(X_train, y_train))*(len(y_train)-1)/(len(y_train)-X_train.shape[1]-1))
```

```
Root Mean squared error:  3183.048041288705
Mean Absolute error:  2121.0685736021182
R-squared score :  0.8113142048629126
adjusted R-sqr :  0.8154269350093037
```

Fig.7. linear regression model

```
# Instantiate the linear model and visualizer
visualizer = prediction_error(lr, X_train, y_train, X_test, y_test)
```

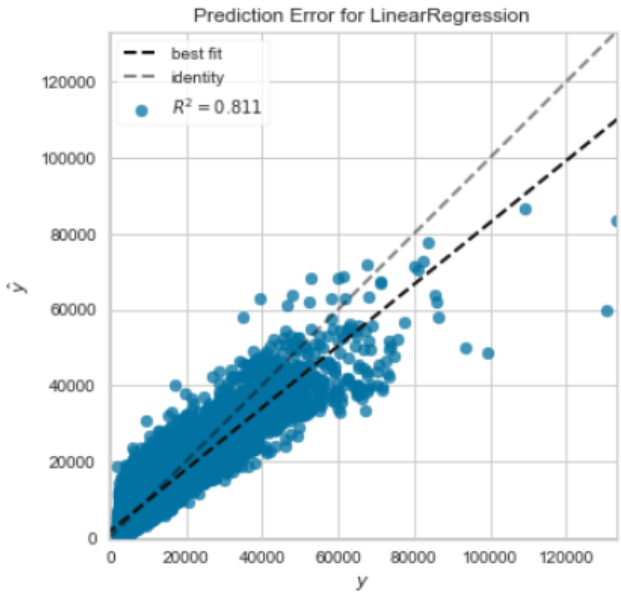

Fig.8. Prediction error for Linear Regression

R2 score of linear regression on test set: 0.811 and RMSE is 3183.04

Above graph shows prediction error. It shows, best fit line and identity line. In This plot identity line is bit far from best fit line. Hence Linear regression model is not so good.

6.2 Decision Tree Regressor:

A decision tree falls under supervised Machine Learning Algorithms in Python and comes of use for both classification and regression- although mostly for classification. This model takes an instance, traverses the tree, and compares important features with a determined conditional statement. Whether it descends to the left child branch or the right depends on the result. Usually, more important features are closer to the root.

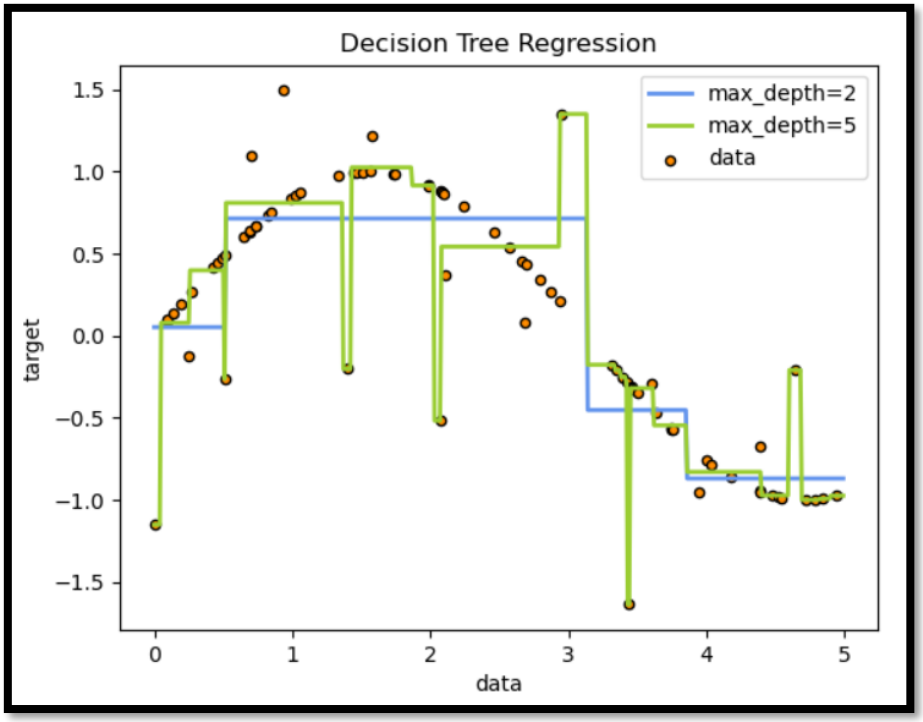

Fig. 9. Decision Tree regression.

Decision Tree, a Machine Learning algorithm in Python can work on both categorical and continuous dependent variables.

```
#from sklearn.tree import DecisionTreeRegressor
DT = DecisionTreeRegressor(random_state=7)

# fit the model using the training data and training targets
DT.fit(X_train,y_train)

y_pred = DT.predict(X_test)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_test, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_test, y_pred))
print("R-squared score : ", r2_score(y_test, y_pred))

#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-DT.score(X_train, y_train))*(len(y_train)-1)/(len(y_train)-X_train.shape[1]-1))
```

```
Root Mean squared error:  2060.0311058767093
Mean Absolute error:  1214.2631486099756
R-squared score :  0.9209684725293646
adjusted R-sqr :  1.0
```

Fig.10. Decision Tree Regression Model

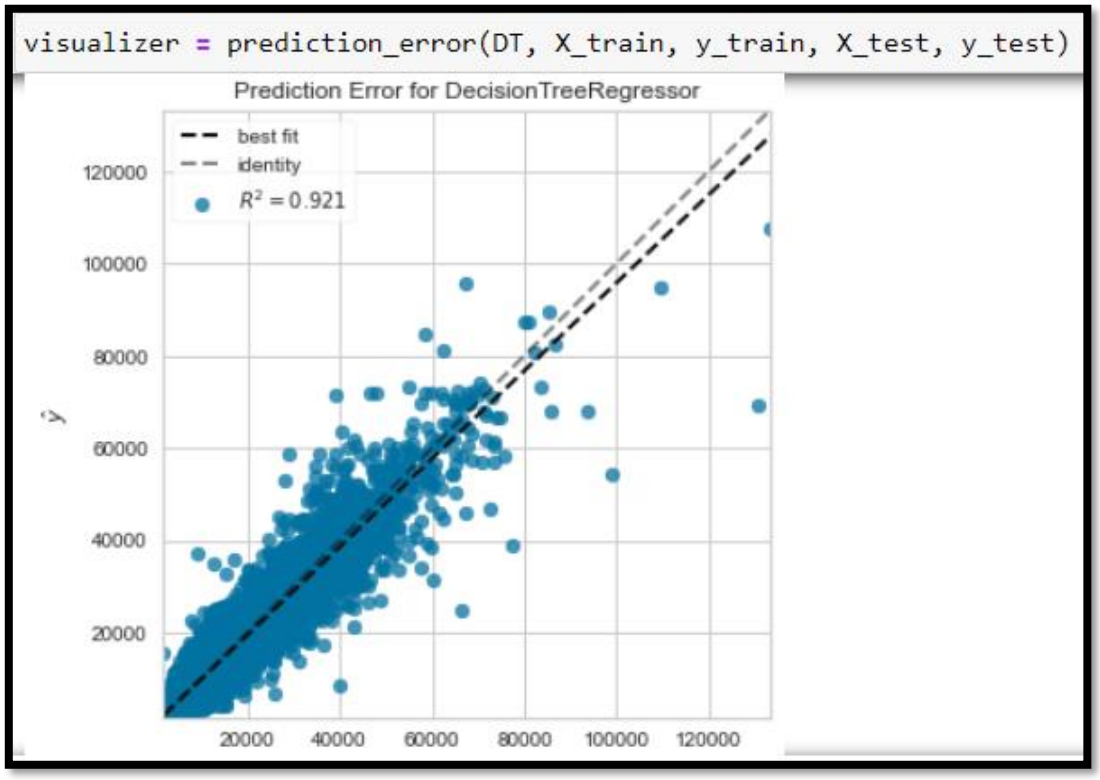

Fig. 11. Prediction error graph

R2 score of Decision tree regression on test set: 0.921 and RMSE is 2060.03

In This plot identity line is closer to best fit line. Hence Decision Tree regression model is good.

6.3 Random Forest Regressor:

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging**.** The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

The final prediction of the random forest is simply the average of the different predictions of all the different decision trees.

```
#from sklearn.ensemble import RandomForestRegressor
RF = RandomForestRegressor(random_state=7)

# fit the model using the training data and training targets
RF.fit(X_train, y_train)

y_pred = RF.predict(X_test)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_test, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_test, y_pred))
print("R-squared score : ", r2_score(y_test, y_pred))


#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-RF.score(X_train, y_train))*(len(y_train)-1)/(len(y_train)-X_train.shape[1]-1))
```

```
Root Mean squared error:  1480.4943756788427
Mean Absolute error:  865.8560469726084
R-squared score :  0.9591806257101969
adjusted R-sqr :  0.9945976779246304
```

Fig.12. Random Forest Regression Model

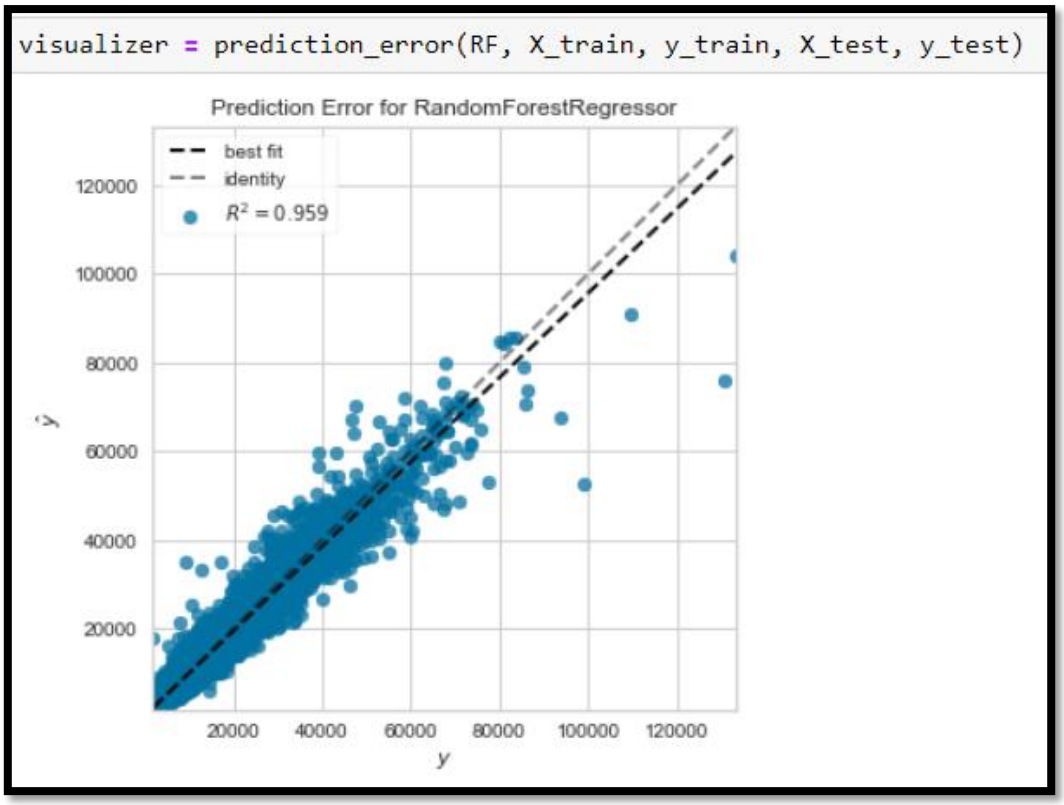

Fig.13 Prediction error for Random Forest regression

R2 score of Random forest regression is much more stable on test set: 0.959 and RMSE is 1480.49

In This plot identity line is very close to best fit line. Hence Random Forest regression model is good.

6.4 XGBOOST Regressor:

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient**,** flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.

```
#from xgboost import XGBRegressor

XG = XGBRegressor()
# Add silent=True to avoid printing out updates with each cycle
# fit the model using the training data and training targets
XG.fit(X_train, y_train, verbose=False)

y_pred = XG.predict(X_test)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_test, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_test, y_pred))
print("R-squared score : ", r2_score(y_test, y_pred))


#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-XG.score(X_train, y_train))*(len(y_train)-1)/(len(y_train)-X_train.shape[1]-1))
```

```
Root Mean squared error:  1558.9495722237725
Mean Absolute error:  975.1065794436189
R-squared score :  0.954739749442598
adjusted R-sqr :  0.9699903616820074
```

Fig.14. XGBoost Regression Model

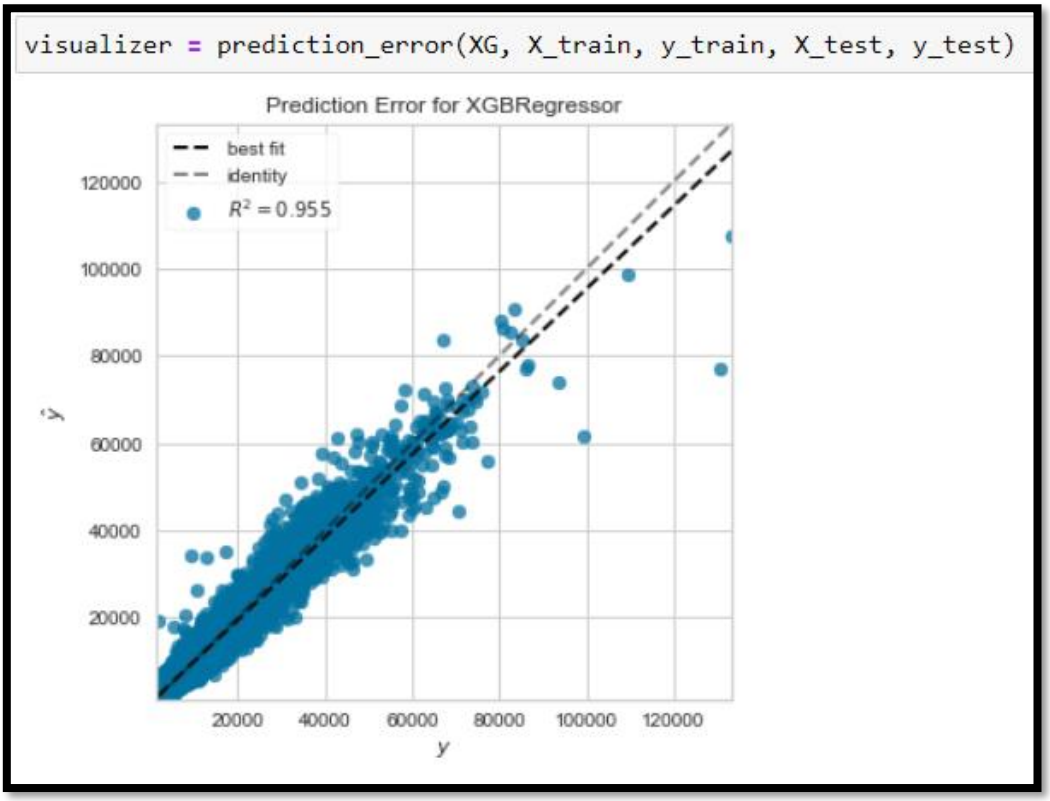

Fig.15.prediction error for XGboost regression model

R2 score of XGBoost regression on test set: 0.955 and RMSE is 1558.94

In This plot identity line is very close to best fit line. Hence XGBoost regression model is Best.

6.5 KNeighbors Regressor:

This is a Python Machine Learning algorithm for classification and regression- mostly for classification. This is a supervised learning algorithm that considers different centroids and uses a usually Euclidean function to compare distance. Then, it analyzes the results and classifies each point to the group to optimize it to place with all closest points to it. It classifies new cases using a majority vote of k of its neighbours. The case it assigns to a class is the one most common among its K nearest neighbours.

```
#from sklearn.neighbors import KNeighborsRegressor

# instantiate the model and set the number of neighbors to consider to 3
reg = KNeighborsRegressor(n_neighbors=3)

# fit the model using the training data and training targets
reg.fit(X_train, y_train)

y_pred = reg.predict(X_test)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_test, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_test, y_pred))
print("R-squared score : ", r2_score(y_test, y_pred))

#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-reg.score(X_train, y_train))*(len(y_train)-1)/(len(y_train)-X_train.shape[1]-1))
```

```
Root Mean squared error:  1602.3481240692524
Mean Absolute error:  929.1611441128373
R-squared score :  0.9521847344536817
adjusted R-sqr :  0.9786858164740054
```

Fig.16. KNeighbors Regression model

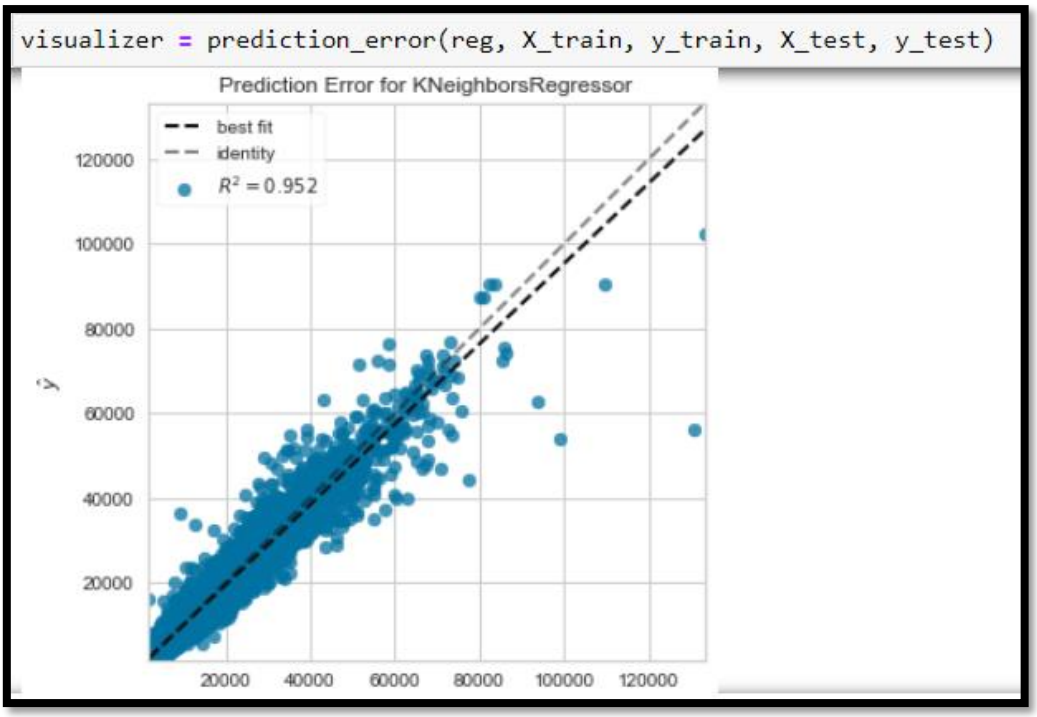

Fig.17. Prediction error for KNeighbors Regression

R2 score of KNeighbors regression on test set: 0.952 and RMSE is 1602.34

In This plot identity line is very close to best fit line. Hence KNeighbors regression model is Best.

7. Prediction

7.1. R2 score and RMSE Table:

Regression Model	R2	Adjusted R2	RMSE
Linear Regression	**0.81**	**0.81**	**3183.04**
Decision Tree Regressor	**1.0**	**0.92**	**2060.03**
Random Forest Regressor	**0.95**	**0.99**	**1480.49**
XGBoost Regressor	**0.95**	**0.96**	**1558.94**
KNeighbors Regressor	**0.95**	**0.97**	**1602.34**

Fig.18. Table of analysis

All applied regression model gives R2 score and Root Mean Square Error. From that we conclude, Random forest regression model is best fit model because, The R2 score is much more stable i.e. 0.95 and the RMSE is also less i.e. 1480.49 than others.

7.2. Model Testing:

We are testing our model (Random Forest Regressor) on test dataset.

```
#from sklearn.ensemble import RandomForestRegressor
RF = RandomForestRegressor(random_state=7)

# fit the model using the training data and training targets
RF.fit(X_test, y_test)

y_pred = RF.predict(X_train)

# calculate R2 score,RMSE,MAE
print("Root Mean squared error: ", mean_squared_error(y_train, y_pred) ** 0.5)
print("Mean Absolute error: ", mean_absolute_error(y_train, y_pred))
print("R-squared score : ", r2_score(y_train, y_pred))

#display adjusted R-squared
print("adjusted R-sqr : ", 1 - (1-RF.score(X_test, y_test))*(len(y_test)-1)/(len(y_test)-X_test.shape[1]-1))
```

```
Root Mean squared error:  1545.8565460278548
Mean Absolute error:  932.1511586920149
R-squared score :  0.9551764004139979
adjusted R-sqr :  0.993334534648326
```

Fig.19. Random Forest Regression Model for Test dataset.

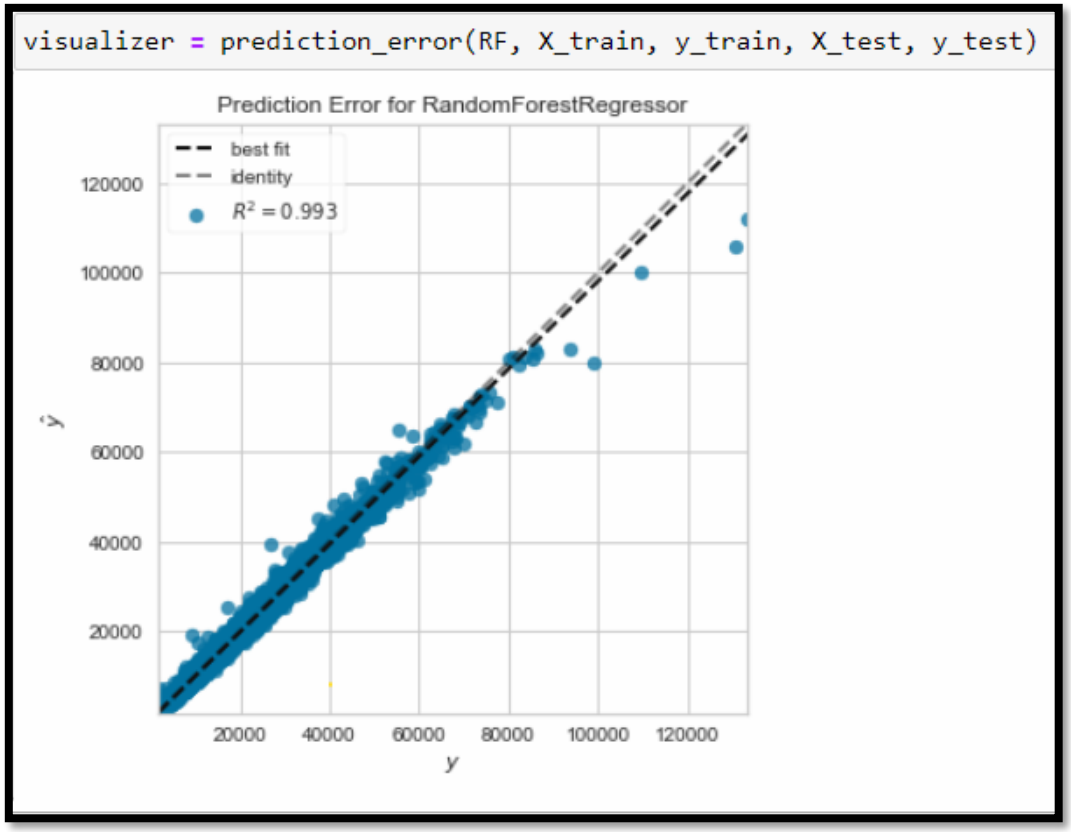

Fig.20. Prediction Error for Random Forest Regression

Random forest regression model on Test dataset, The R2 score is much more stable i.e. 0.95 and the RMSE is also less i.e. 1545.85 .It is giving similar output as Train Dataset.

In the Prediction error graph, we can see identity line is much more closer to best fit line.

7.3 Conclusion:

1. Linear Regression model:

R2 score: 0.81

RMSE: 3183.04

2.Decision Tree Regression:

R2 score: 1

RMSE: 2060.03

3.Random Forest Regression:

R2 score: 0.95

RMSE: 1480.49

4.XGboost Regression:

R2 score:0.95

RMSE: 1558.94

5. KNeighbors Regressor:

R2 score:0.95

RMSE: 1602.34

Above result shows RMSE and R2 score of all applied regression models. From that we can see most of models giving R2 score value above 0.95, which is stable. but its RMSE value is high. Hence, we conclude that, Random Forest regression is best fit to our dataset.

8. Future Scope

- For this project we used historical data. In future we can use this model on real time dataset to predict Medicare spends.
- Data set have no restrictions because new data gets added yearly into it.
- In some countries Medicare Insurance is compulsory.
- This model helps the people to plan their financial investments better way. Because it predicts appropriate cost of Medicare spend.

9. Requirements Specification

9.1 Hardware Requirement:

- 500 GB hard drive (minimum required)
- 8 GB RAM (minimum required)
- PCx64-bit CPU

9.2 Software Requirement:

- Windows/Mac/Linux
- Python-3.8
- Libraries:

    1. Pandas : 1.2.3
    2. numpy : 1.19.5
    3. matplotlib: 3.3.4
    4. Scikit-learn 0.24.1
    5. seaborn 0.11.1
    6. scipy 1.6.1
    7. yellowbrick 1.3.post1

10.References

1. Linear regression model for predicting medical expenses based on insurance data, December 2019, [DOI: 10.13140/RG.2.2.32478.38722].

2. Regression Analysis of Health Insurance Cost Affecting Factors, 2013 International Conference on Advances in Social Science, Humanities, and Management (ASSHM 2013)

3. The research of regression model in machine learning field, MATEC Web of Conferences 176, 01033 (2018) https://doi.org/10.1051/matecconf/201817601033 IFID 2018

4. Health Management based on History of Personalized Physiological Data using Linear Regression Analysis, 2018, GLOBAL MEDICAL ENGINEERING PHYSICS EXCHANGES/PAN AMERICAN HEALTH CARE EXCHANGES (GMEPE / PAHCE)

5. Improving Healthcare Services of Community Clinics using Machine Learning Techniques, 2018 2nd Int. Conf. on Innovations in Science, Engineering and Technology (ICISET) 27-28 October 2018, Chittagong, Bangladesh