MINI PROJECT REPORT

on

Predicting Life Expectancy

Submitted in partial fulfilment for the completion of

B.E., V Semester

INFORMATION TECHNOLOGY

By

MAHITHA KOTHAPALLY (160118737009)
SWETHA VALAKONDA (160118737020)

Under the guidance of

S. Rakesh,
Asst. Professor,
Dept. of IT, CBIT.

DEPARTMENT OF INFORMATION TECHNOLOGY
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)
(Affiliated to Osmania University; Accredited by NBA(AICTE) and NAAC(UGC), ISO Certified 9001:2015)
GANDIPET, HYDERABAD – 500 075
Website: www.cbit.ac.in

2020-2021

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)

DEPARTMENT OF INFORMATION TECHNOLOGY

(Affiliated to Osmania University)

GANDIPET, HYDERABAD – 500 075

CERTIFICATE

This is to certify that Mini Project-III entitled "**PREDICTING LIFE EXPECTANCY**" submitted to **CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY,** in partial fulfilment of the requirements for the completion of B.E.,V semester Information Technology, during the academic year 2020-2021, is a record of original work done by **MAHITHA KOTHAPALLY (160118737009), SWETHA VALAKONDA (160118737020),** during the period of study in the Department of IT,CBIT, HYDERABAD, under my supervision and guidance.

Guide
S. Rakesh,
Asst. Professor, Dept. of IT,
CBIT, Hyderabad.

Head of the Department
Dr. K. Radhika,
Professor, Dept. of IT,
CBIT, Hyderabad.

CONTENTS

Page No.

DECLARATION

This is to certify that the work reported in the present report titled “PREDICTING LIFE EXPECTANCY” is a record of work done by is in the department of information technology, Chaitanya Bharathi Institute of Technology, Hyderabad.

No part of the report is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred. The reported results are based on the project work done entirely by us and not copied from any other source.

Mahitha Kothapally (160118737009)

Swetha Valakonda(160118737020)

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to **S.Rakesh**, our project guide, for his invaluable guidance and constant support, along with his capable instruction and persistent encouragement.

We are grateful to our Head of Department, **Dr. K. Radhika**, for her steady support and the provision of every resource required for the completion of this project.

We would like to take this opportunity to thank our Principal, **Prof. GPS Varma**, as well as the management, for having designed an excellent learning atmosphere.

Our thanks to all members of the staff and our lab assistants for helping us to carry out the groundwork of this project.

We also take this opportunity to thank our parents for their support to complete the project.

ABSTRACT

As a result of the evolution of biotechnologies and related technologies such as the development of sophisticated medical equipment, humans are able to enjoy longer life expectancies than previously before. Predicting a human's life expectancy has been a long-term question to mankind. Many calculations and research have been done to create an equation despite it being impractical to simplify these variables into one equation.

Life expectancy refers to the number of years a person is expected to live based on the statistical average. Life expectancy varies by geographical area and by era. In mathematical terms, life expectancy refers to the expected number of years remaining for an individual at any given age. The life expectancy for a particular person or population group depends on several variables such as their lifestyle, access to healthcare, diet, economical status and the relevant mortality and morbidity data. However, as life expectancy is calculated based on averages, a person may live for many years more or less than expected. It can be developed using the Machine Learning model.

LIST OF FIGURES

1.INTRODUCTION

1.1 Motivation

Life expectancy refers to the number of years a person is expected to live based on the statistical average. Life expectancy varies by geographical area and by era. In mathematical terms, life expectancy refers to the expected number of years remaining for an individual at any given age. The life expectancy for a particular person or population group depends on several variables such as their lifestyle, access to healthcare, diet, economic status and the relevant mortality and morbidity data. However, as life expectancy is calculated based on averages, a person may live for many years more or less than expected.

1.2. Basic Definitions

1.2.1. Machine learning- Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

1.2.2. Random forest - The Random forest or Random Decision Forest is a supervised Machine learning algorithm used for classification, regression, and other tasks using decision trees. The Random forest classifier creates a set of decision trees from a randomly selected subset of the training set. It is basically a set of decision trees (DT) from a randomly selected subset of the training set and then It collects the votes from different decision trees to decide the final prediction.

1.2.3. Jupyter Notebook - JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

1.2.4. Ridge regression - Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors. It is hoped that the net effect will be to give estimates that are more reliable.

1.2.5. Linear regression - Linear regression is the next step up after correlation. It is used when we want to predict the value of a variable based on the value of another variable. The variable we want to predict is called the dependent variable (or sometimes, the outcome variable). The variable we are using to predict the other variable's value is called the independent variable (or sometimes, the predictor variable).

1.3. Problem Statement

Life expectancy is perhaps the most important measure of health. Life expectancy increases due to healthcare improvements like the introduction of vaccines, the development of drugs or positive behaviour changes like the reduction in smoking or drinking rates. Life expectancy increases with age as the individual survives the higher mortality rates associated with childhood.

2. EXISTING SYSTEM

As a result of the evolution of biotechnologies and related technologies such as the development of sophisticated medical equipment, humans are able to enjoy longer life expectancies than previously before. Predicting a human’s life expectancy has been a long-term question to mankind. Many calculations and research have been done to create an equation despite it being impractical to simplify these variables into one equation.

Currently there are various smart devices and applications such as smartphone apps and wearable devices that provide wellness and fitness tracking. Some apps provide health related data such as sleep monitoring, heart-rate measuring, and calorie expenditure collected and processed by the devices and servers in the cloud. However, no existing work provides the Personalized Life expectancy.

3. PROPOSED SYSTEM

3.1. Methodology

We are using a machine learning model to predict life expectancy. Machine learning techniques offer a feasible and promising approach to predicting life expectancy. The project tries to build a model based on the given dataset. Firstly, the dataset is preprocessed which means preparing the raw data and making it suitable for the machine learning model, then we convert the string values to integers, since the machine learning model does not accept object data types. Later we split the data into training, test sets and we train the data using some regression model. Random forest regression is used to predict the model and it gives accuracy about 96%.

3.2. Architecture of Proposed System

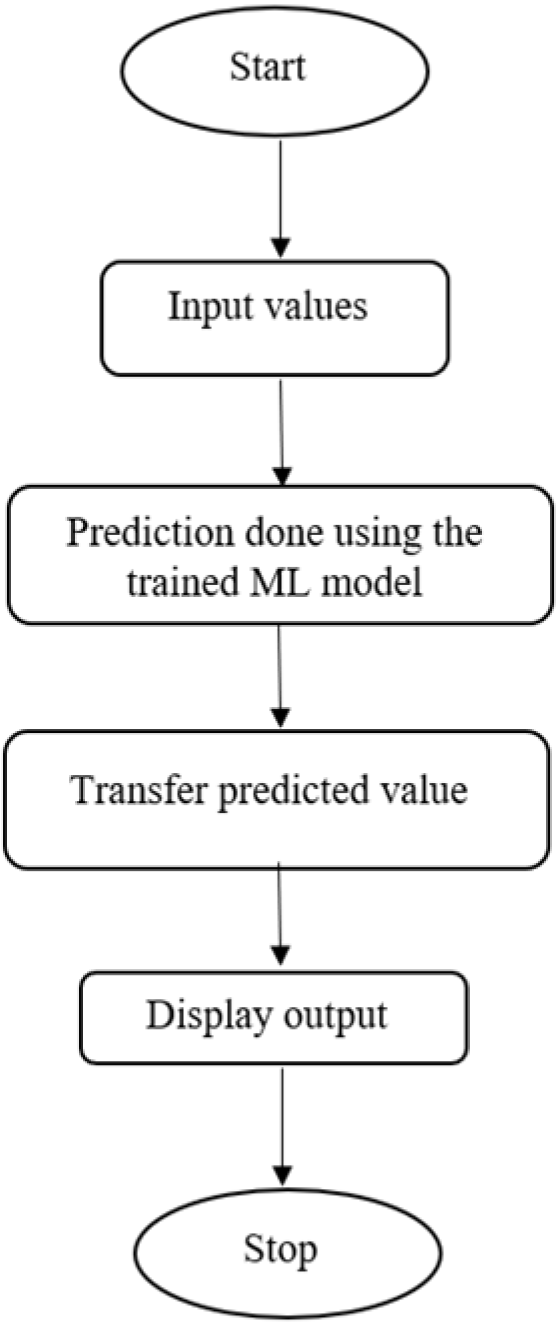


Fig.3.1 Flowchart of Predicting life expectancy model

4. SOFTWARE& HARDWARE REQUIREMENTS

Software:

Python IDE

Jupyter notebook

Hardware:

Processors: Intel Atom® processor or Intel® Core™ i3 processor.

Input device (mouse / trackpad) to select options

32 – 64 bit processor

Sufficient RAM to run the program

5. IMPLEMENTATION OF PROJECT

5.1. Implementation

Predicting life expectancy comes under supervised machine learning tasks. The project tries to build a model based on the given dataset. First, we need to import the necessary libraries and the dataset (LifeExpectancy.xlsx).

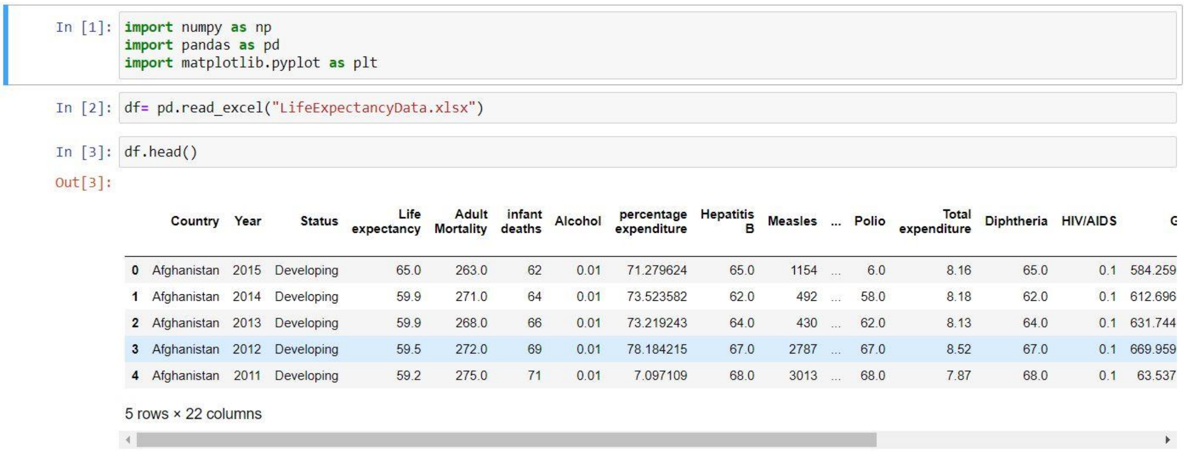
The raw data is not suitable for us to start building a model so some preprocessing will be done. We have some null values in the table, the isnull function has been used to find the number of nul values present in the dataset. We should remove these null values from the dataset. bfill method will fill the null values with the backward values. The Status of the country is turned into numerical with the LabelEncoder() function. And the country column is dropped as it is object data type.

Then the Life expectancy column is removed to form the y variable or the output, and the rest is stored as the x variable. Now we will split the data into a training part of 80% and a testing of 20%. After splitting the data, we will use a model to fit the data and find the score function which gives the accuracy of the model we are using. MSE(Mean Square Error), MAE(Mean Absolute Error) and RMSE(Root Mean Square Error) are also calculated for the model. Graph is plotted for the predicted values and real values using scatter() from matplot.

We used three regression models, they are Random Forest Regression, Ridge Regression, and Linear Regression. First, the model is fitted for the Random Forest regression which gives 96.8% accuracy. The value of MAE is 1.0796, MSE is 2.708 and RMSE is 1.645. Next, the model is fitted for the Ridge regression which gives 80% accuracy. The value of MAE is 3.000, MSE is 16.739 and RMSE is 4,091. Later, the model is fitted for the Linear regression which gives 81% accuracy. The value of MAE is 2.910, MSE is 15,732 and RMSE is 3.966.

After comparing the three regression models, Random Forest is the best model since it has more accuracy and less values of MSE, MAE and RMSE.

5.2. Results



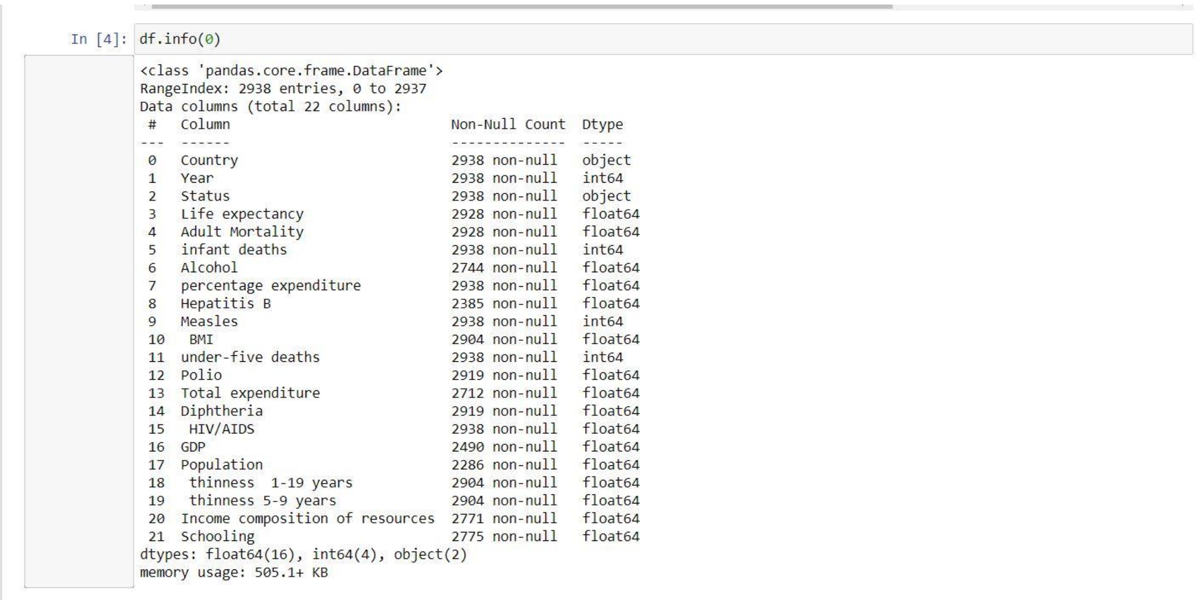
```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt

In [2]: df= pd.read_excel("LifeExpectancyData.xlsx")

In [3]: df.head()
Out[3]:
```

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	Polio	Total expenditure	Diphtheria	HIV/AIDS	G
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	6.0	8.16	65.0	0.1	584.259
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	58.0	8.18	62.0	0.1	612.696
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	62.0	8.13	64.0	0.1	631.744
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	67.0	8.52	67.0	0.1	669.959
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	68.0	7.87	68.0	0.1	63.537

5 rows × 22 columns

Fig.5.1 Import dataset

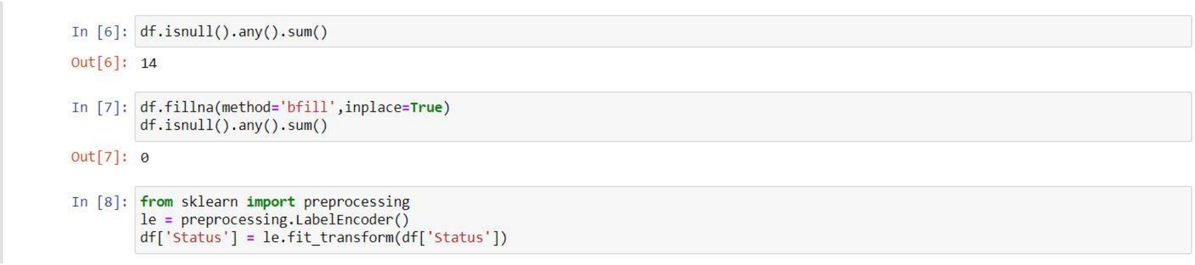
Numpy, pandas, matplotlib packages have been imported and the data set is read in the above figure. By writing the df.head() function it displays the first five rows in the dataset.



```
In [4]: df.info(0)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Country                          2938 non-null   object
 1   Year                             2938 non-null   int64
 2   Status                           2938 non-null   object
 3   Life expectancy                  2928 non-null   float64
 4   Adult Mortality                  2928 non-null   float64
 5   infant deaths                    2938 non-null   int64
 6   Alcohol                          2744 non-null   float64
 7   percentage expenditure           2938 non-null   float64
 8   Hepatitis B                      2385 non-null   float64
 9   Measles                          2938 non-null   int64
 10   BMI                             2904 non-null   float64
 11  under-five deaths                2938 non-null   int64
 12  Polio                            2919 non-null   float64
 13  Total expenditure                2712 non-null   float64
 14  Diphtheria                       2919 non-null   float64
 15   HIV/AIDS                        2938 non-null   float64
 16  GDP                              2490 non-null   float64
 17  Population                       2286 non-null   float64
 18   thinness  1-19 years            2904 non-null   float64
 19   thinness 5-9 years              2904 non-null   float64
 20  Income composition of resources  2771 non-null   float64
 21  Schooling                        2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
```

Fig.5.2 Description of dataset

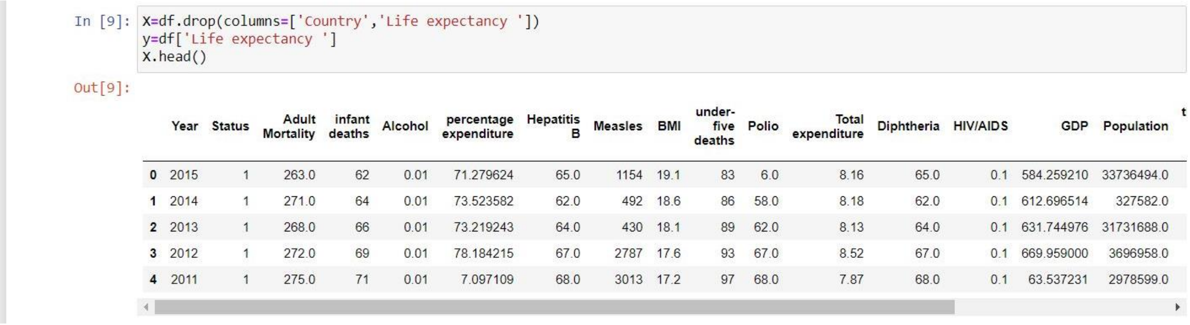
df.info(0) function gives the description table of the dataset



```
In [6]: df.isnull().any().sum()

Out[6]: 14

In [7]: df.fillna(method='bfill',inplace=True)
        df.isnull().any().sum()

Out[7]: 0

In [8]: from sklearn import preprocessing
        le = preprocessing.LabelEncoder()
        df['Status'] = le.fit_transform(df['Status'])
```

Fig.5.3 Data preprocessing

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. First, we need to deal with the null values which are present in the dataset. isnull() function will return the number of null values present in the dataset. Here, we have a total 14 null values present. We should remove these null values from the dataset. bfill method will fill the null values with the backward values. Machine learning models only take integers as inputs so we need to replace the status column with integers. For that, we used LabelEncoder() which replaces strings with integers.

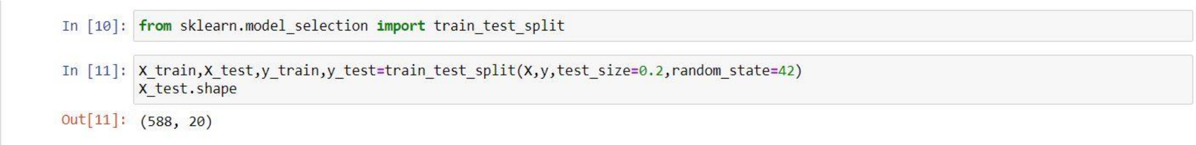


```
In [9]: X=df.drop(columns=['Country','Life expectancy '])
        y=df['Life expectancy ']
        X.head()

Out[9]:
```

	Year	Status	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio	Total expenditure	Diphtheria	HIV/AIDS	GDP	Population	t
0	2015	1	263.0	62	0.01	71.279624	65.0	1154	19.1	83	6.0	8.16	65.0	0.1	584.259210	33736494.0	
1	2014	1	271.0	64	0.01	73.523582	62.0	492	18.6	86	58.0	8.18	62.0	0.1	612.696514	327582.0	
2	2013	1	268.0	66	0.01	73.219243	64.0	430	18.1	89	62.0	8.13	64.0	0.1	631.744976	31731688.0	
3	2012	1	272.0	69	0.01	78.184215	67.0	2787	17.6	93	67.0	8.52	67.0	0.1	669.959000	3696958.0	
4	2011	1	275.0	71	0.01	7.097109	68.0	3013	17.2	97	68.0	7.87	68.0	0.1	63.537231	2978599.0	

Fig.5.4 Drop columns

The datatype of country column is the object and it is not taken by machine learning model, Since the machine learning model takes only integers as inputs. Life expectancy is taken at the output end therefore country and life expectancy columns are dropped from the dataset by drop() function. Life expectancy column is stored as output.

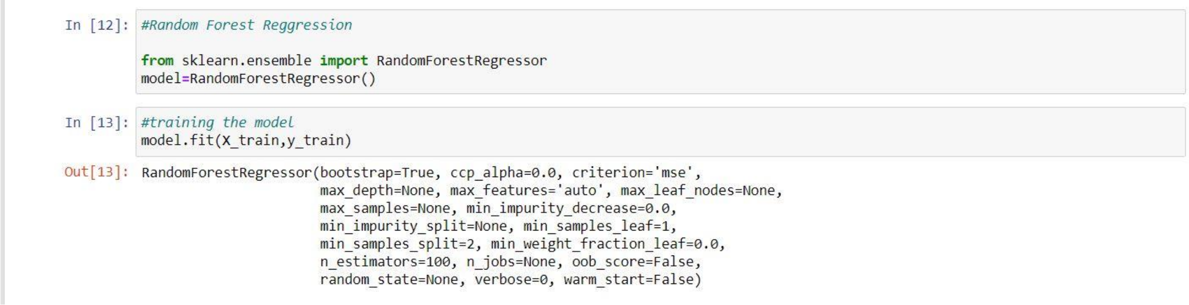


```
In [10]: from sklearn.model_selection import train_test_split

In [11]: X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
         X_test.shape

Out[11]: (588, 20)
```

Fig.5.5 Splitting of data

We need to split the data into training and test sets. Now we can use the train_test_split function in order to make the split. The test_size=0.2 inside the function indicates the percentage of the data that should be held over for testing.

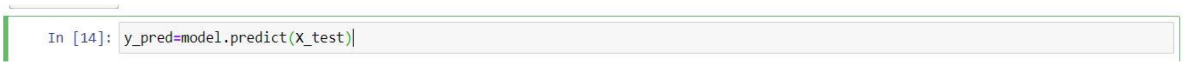
Random Forest Regression



```
In [12]: #Random Forest Reggression

         from sklearn.ensemble import RandomForestRegressor
         model=RandomForestRegressor()

In [13]: #training the model
         model.fit(X_train,y_train)

Out[13]: RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                               max_depth=None, max_features='auto', max_leaf_nodes=None,
                               max_samples=None, min_impurity_decrease=0.0,
                               min_impurity_split=None, min_samples_leaf=1,
                               min_samples_split=2, min_weight_fraction_leaf=0.0,
                               n_estimators=100, n_jobs=None, oob_score=False,
                               random_state=None, verbose=0, warm_start=False)
```

Fig.5.6 Random forest regression

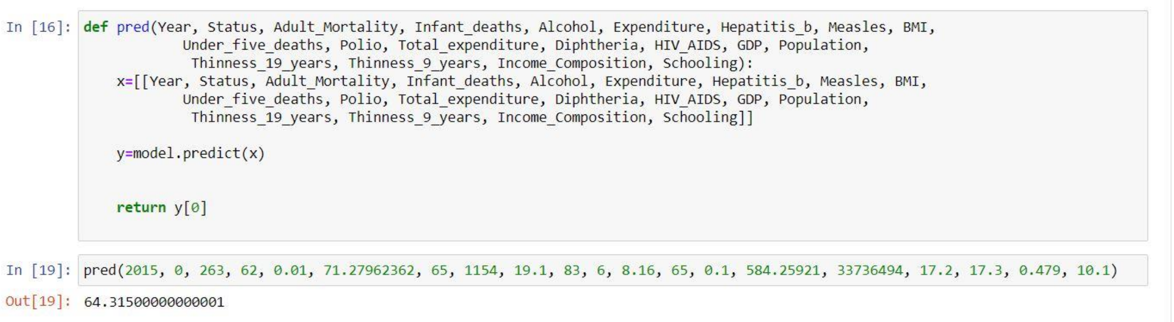
We used Random forest regression for our model. We’re fitting the model on the training data and trying to predict the test data.



```
In [14]: y_pred=model.predict(X_test)
```

Fig.5.7 Predict

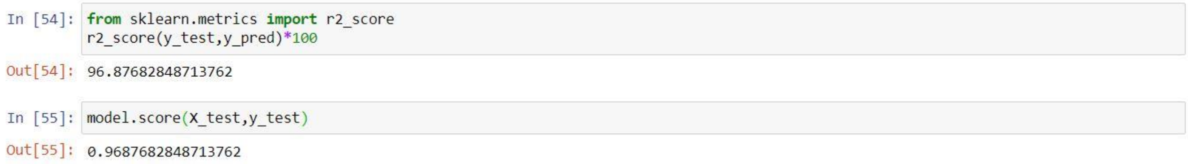
By using predict function we are predicting the life expectancy for the test dataset. The output of the function gives an array of predicted values.



```
In [16]: def pred(Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                     Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                      Thinness_19_years, Thinness_9_years, Income_Composition, Schooling):
             x=[[Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                     Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                      Thinness_19_years, Thinness_9_years, Income_Composition, Schooling]]

             y=model.predict(x)


             return y[0]

In [19]: pred(2015, 0, 263, 62, 0.01, 71.27962362, 65, 1154, 19.1, 83, 6, 8.16, 65, 0.1, 584.25921, 33736494, 17.2, 17.3, 0.479, 10.1)

Out[19]: 64.31500000000001
```

Fig.5.8 Predict function for Random Forest

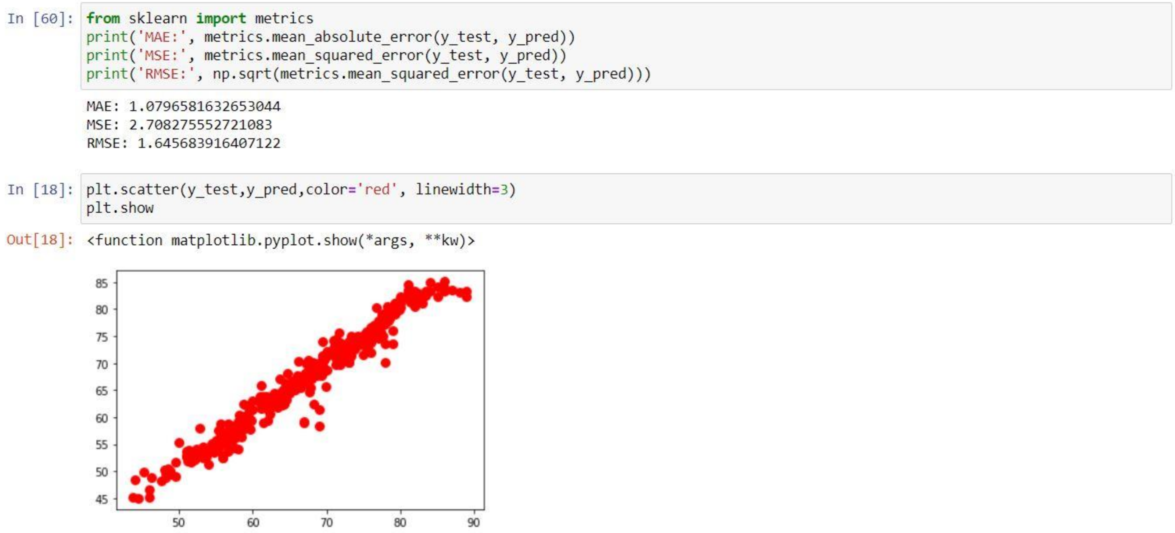
To predict life expectancy for given input pred() function is written. The output is the predicted life expectancy.



```
In [54]: from sklearn.metrics import r2_score
         r2_score(y_test,y_pred)*100

Out[54]: 96.87682848713762

In [55]: model.score(X_test,y_test)

Out[55]: 0.9687682848713762
```

Fig.5.9 Random forest Score function

r2_score() function is used to evaluate the performance of the regression model. In other words, it gives the efficiency of the predicted value. For Random Forest Regression, it is giving 96 percent accuracy.

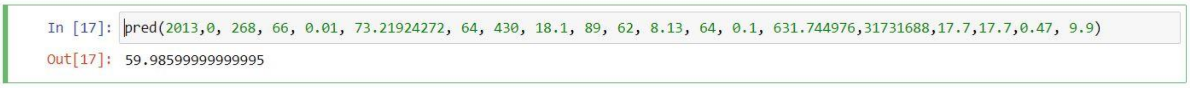


```
In [60]: from sklearn import metrics
         print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
         print('MSE:', metrics.mean_squared_error(y_test, y_pred))
         print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

         MAE: 1.0796581632653044
         MSE: 2.708275552721083
         RMSE: 1.645683916407122

In [18]: plt.scatter(y_test,y_pred,color='red', linewidth=3)
         plt.show

Out[18]: <function matplotlib.pyplot.show(*args, **kw)>
```

Fig.5.10 Random forest Metric functions

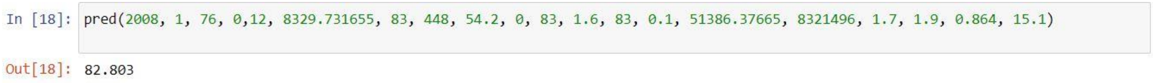
Mean Absolute Error (MAE) measures the absolute average distance between the real data and the predicted data, but it fails to punish large errors in prediction. Mean Square Error (MSE) measures the squared average distance between the real data and the predicted data. Root Mean Square Error (RMSE) is simply the root of MSE. Also, this metric solves the problem of squaring the units. As they are the distance between the real data and predicted data, lesser these values give the more accuracy.

A scatter plot is a diagram where each value in the data set is represented by a dot. The scatter() method in the matplotlib library is used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one affects the other. Here, we are plotting a graph for predicted values and real values where x-axis is for predicted values and y-axis for real values.



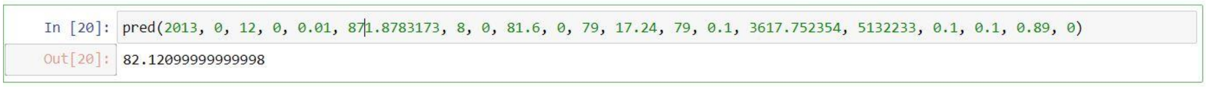
```
In [17]: pred(2013,0, 268, 66, 0.01, 73.21924272, 64, 430, 18.1, 89, 62, 8.13, 64, 0.1, 631.744976,31731688,17.7,17.7,0.47, 9.9)
Out[17]: 59.98599999999995
```

Fig.5.11 Test case 1



```
In [18]: pred(2008, 1, 76, 0,12, 8329.731655, 83, 448, 54.2, 0, 83, 1.6, 83, 0.1, 51386.37665, 8321496, 1.7, 1.9, 0.864, 15.1)

Out[18]: 82.803
```

Fig.5.12 Test case 2

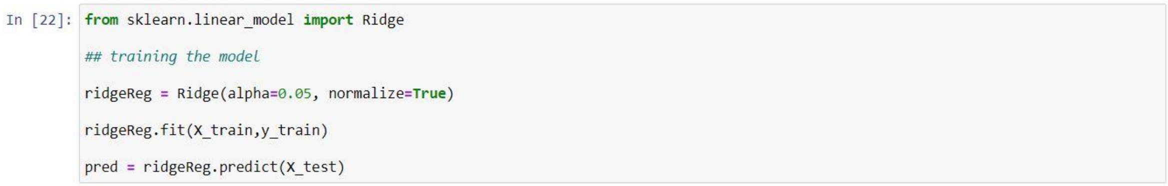


```
In [20]: pred(2013, 0, 12, 0, 0.01, 871.8783173, 8, 0, 81.6, 0, 79, 17.24, 79, 0.1, 3617.752354, 5132233, 0.1, 0.1, 0.89, 0)
Out[20]: 82.12099999999998
```

Fig.5.13 Test case 3

Above are some test cases for random forest regression, whose outputs are the predicted life expectancy. In the test case 1 for the given input values, we got 59.98 as the predicted life expectancy, for the test case 2 the predicted life expectancy is 82.803 for another set input values. Similarly, for the test case 3 the output is 82.12. All these outputs have around 96 percent accuracy.

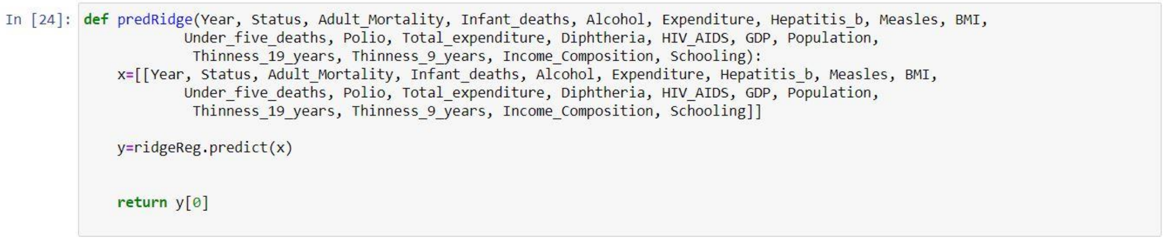
Ridge regression



```
In [22]: from sklearn.linear_model import Ridge

         ## training the model

         ridgeReg = Ridge(alpha=0.05, normalize=True)

         ridgeReg.fit(X_train,y_train)

         pred = ridgeReg.predict(X_test)
```

Fig.5.14 Ridge regression

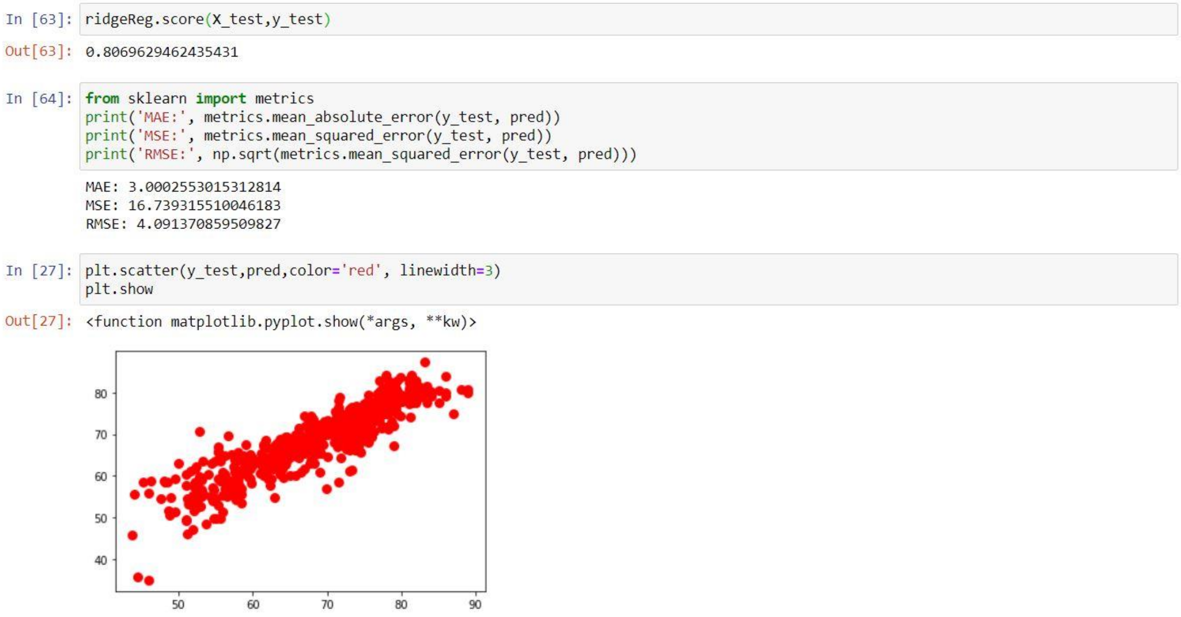
We used Ridge regression for our model. We’re fitting the model on the training data and trying to predict the test data. By using predict function we are predicting the life expectancy for the test dataset. The output of the function gives an array of predicted values.



```
In [24]: def predRidge(Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                      Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                       Thinness_19_years, Thinness_9_years, Income_Composition, Schooling):
             x=[[Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                      Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                       Thinness_19_years, Thinness_9_years, Income_Composition, Schooling]]

             y=ridgeReg.predict(x)


             return y[0]
```

Fig.5.15 Predict Function for Ridge regression

To predict life expectancy for given input predRidge() function is written. The output is the predicted life expectancy.



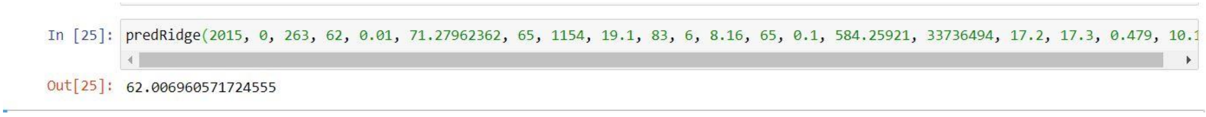
```
In [63]: ridgeReg.score(X_test,y_test)

Out[63]: 0.8069629462435431

In [64]: from sklearn import metrics
         print('MAE:', metrics.mean_absolute_error(y_test, pred))
         print('MSE:', metrics.mean_squared_error(y_test, pred))
         print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred)))

         MAE: 3.0002553015312814
         MSE: 16.739315510046183
         RMSE: 4.091370859509827

In [27]: plt.scatter(y_test,pred,color='red', linewidth=3)
         plt.show

Out[27]: <function matplotlib.pyplot.show(*args, **kw)>
```

Fig.5.16 Ridge regression score function

r2_score() function is used to evaluate the performance of the regression model. In other words, it gives the accuracy of the predicted value. For Ridge Regression, it is giving 80 percent accuracy.

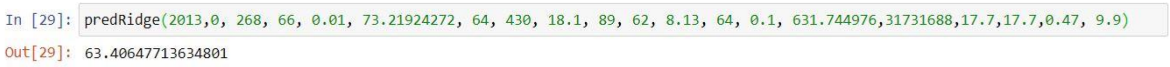
Mean Absolute Error (MAE) measures the absolute average distance between the real data and the predicted data, but it fails to punish large errors in prediction. Mean Square Error (MSE) measures the squared average distance between the real data and the predicted data. Root Mean Square Error (RMSE) is simply the root of MSE. Also, this metric solves the problem of squaring the units. As they are the distance between the real data and predicted data, lesser these values give the more accuracy.

A scatter plot is a diagram where each value in the data set is represented by a dot. The scatter() method in the matplotlib library is used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one affects the other. Here, we are plotting a graph for predicted values and real values where x-axis is for predicted values and y-axis for real values.



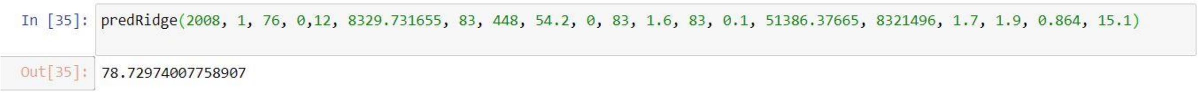
```
In [25]: predRidge(2015, 0, 263, 62, 0.01, 71.27962362, 65, 1154, 19.1, 83, 6, 8.16, 65, 0.1, 584.25921, 33736494, 17.2, 17.3, 0.479, 10.1
Out[25]: 62.006960571724555
```

Fig.5.17 Test case 1



```
In [29]: predRidge(2013,0, 268, 66, 0.01, 73.21924272, 64, 430, 18.1, 89, 62, 8.13, 64, 0.1, 631.744976,31731688,17.7,17.7,0.47, 9.9)
Out[29]: 63.40647713634801
```

Fig.5.18 Test case 2

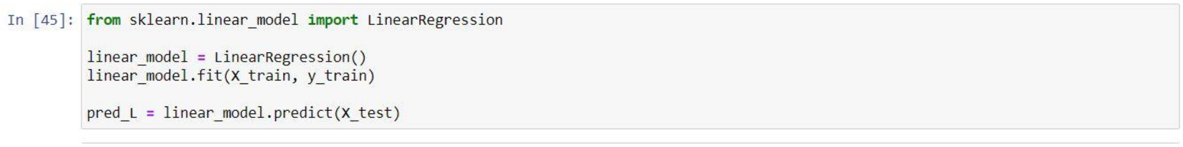


```
In [35]: predRidge(2008, 1, 76, 0,12, 8329.731655, 83, 448, 54.2, 0, 83, 1.6, 83, 0.1, 51386.37665, 8321496, 1.7, 1.9, 0.864, 15.1)

Out[35]: 78.72974007758907
```

Fig.5.19 Test case 3

Above are some test cases for ridge regression, whose outputs are the predicted life expectancy. For all the test cases the outputs have around 80 percent accuracy.

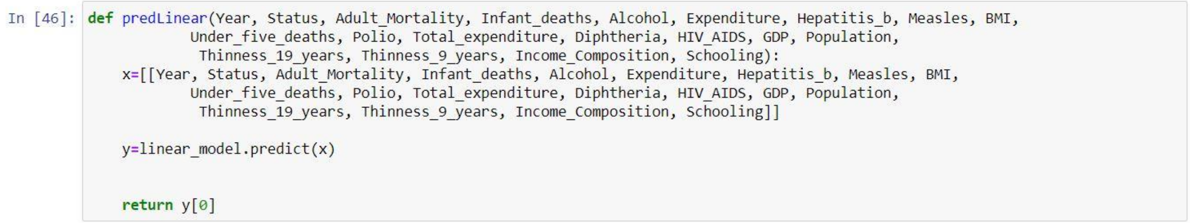
Linear Regression



```
In [45]: from sklearn.linear_model import LinearRegression

         linear_model = LinearRegression()
         linear_model.fit(X_train, y_train)

         pred_L = linear_model.predict(X_test)
```

Fig.5.20 Linear regression

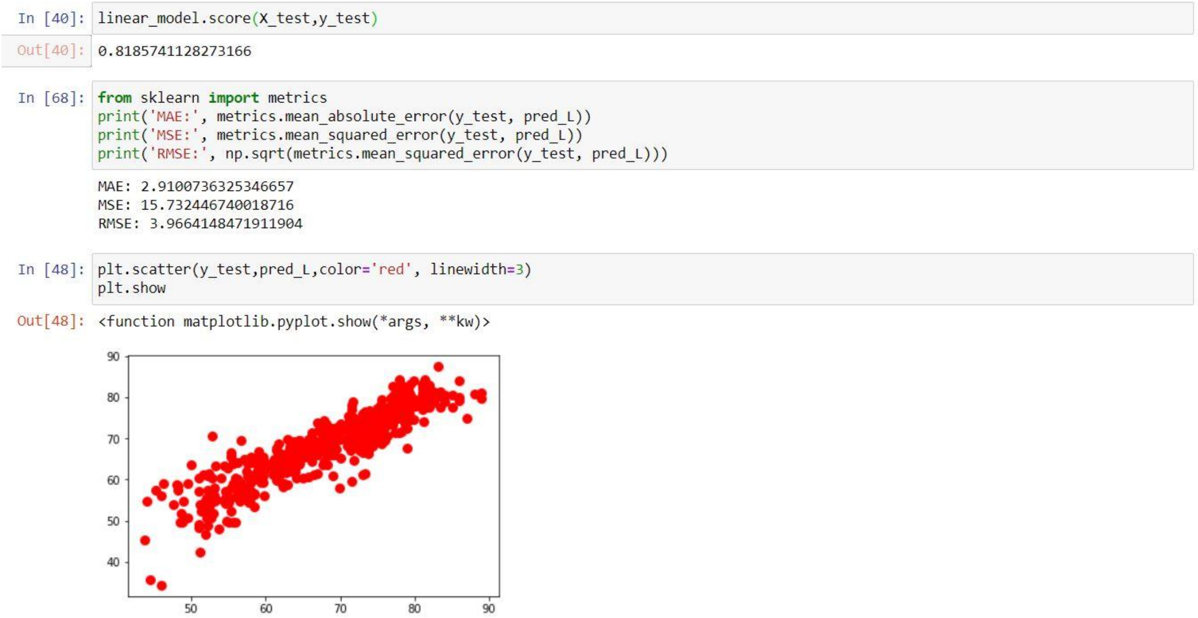
We used Linear regression for our model. We're fitting the model on the training data and trying to predict the test data. By using predict function we are predicting the life expectancy for the test dataset. The output of the function gives an array of predicted values.



```
In [46]: def predLinear(Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                     Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                      Thinness_19_years, Thinness_9_years, Income_Composition, Schooling):
             x=[[Year, Status, Adult_Mortality, Infant_deaths, Alcohol, Expenditure, Hepatitis_b, Measles, BMI,
                     Under_five_deaths, Polio, Total_expenditure, Diphtheria, HIV_AIDS, GDP, Population,
                      Thinness_19_years, Thinness_9_years, Income_Composition, Schooling]]

             y=linear_model.predict(x)


             return y[0]
```

Fig.5.21 Predict Function for Linear regression

To predict life expectancy for given input predLinear() function is written. The output is the predicted life expectancy.



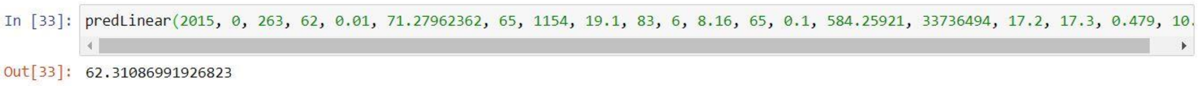
```
In [40]: linear_model.score(X_test,y_test)

Out[40]: 0.8185741128273166

In [68]: from sklearn import metrics
         print('MAE:', metrics.mean_absolute_error(y_test, pred_L))
         print('MSE:', metrics.mean_squared_error(y_test, pred_L))
         print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_L)))

         MAE: 2.9100736325346657
         MSE: 15.732446740018716
         RMSE: 3.9664148471911904

In [48]: plt.scatter(y_test,pred_L,color='red', linewidth=3)
         plt.show

Out[48]: <function matplotlib.pyplot.show(*args, **kw)>
```

Fig.5.22 Linear regression score function

r2_score() function is used to evaluate the performance of the regression model. In other words, it gives the accuracy of the predicted value. For Linear Regression, it is giving 81 percent accuracy.

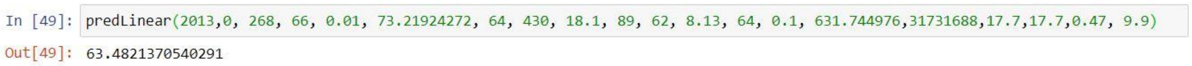
Mean Absolute Error (MAE) measures the absolute average distance between the real data and the predicted data, but it fails to punish large errors in prediction. Mean Square Error (MSE) measures the squared average distance between the real data and the predicted data. Root Mean Square Error (RMSE) is simply the root of MSE. Also, this metric solves the problem of squaring the units. As they are the distance between the real data and predicted data, lesser these values give the more accuracy.

A scatter plot is a diagram where each value in the data set is represented by a dot. The scatter() method in the matplotlib library is used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one affects the other. Here, we are plotting a graph for predicted values and real values where x-axis is for predicted values and y-axis for real values.



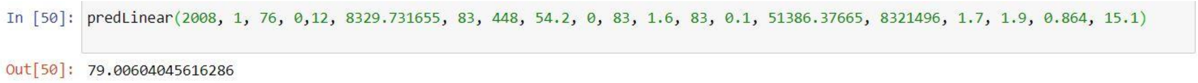
```
In [33]: predLinear(2015, 0, 263, 62, 0.01, 71.27962362, 65, 1154, 19.1, 83, 6, 8.16, 65, 0.1, 584.25921, 33736494, 17.2, 17.3, 0.479, 10.

Out[33]: 62.31086991926823
```

Fig.5.23 Test case 1



```
In [49]: predLinear(2013,0, 268, 66, 0.01, 73.21924272, 64, 430, 18.1, 89, 62, 8.13, 64, 0.1, 631.744976,31731688,17.7,17.7,0.47, 9.9)
Out[49]: 63.4821370540291
```

Fig.5.24 Test case 2



```
In [50]: predLinear(2008, 1, 76, 0,12, 8329.731655, 83, 448, 54.2, 0, 83, 1.6, 83, 0.1, 51386.37665, 8321496, 1.7, 1.9, 0.864, 15.1)

Out[50]: 79.00604045616286
```

Fig.5.25 Test case 3

Above are some test cases for linear regression, whose outputs are the predicted life expectancy. For all the test cases the outputs have around 81 percent accuracy.

6. ADVANTAGES AND DISADVANTAGES

Advantages:

Life expectancy can be estimated at any age, e.g., life expectancy at 65 years. Gives more weight to deaths at younger ages. Life expectancy has been used nationally to monitor health inequalities.

The application learns the patterns and trends hidden within the data without human intervention which makes predicting much simpler and easier. The more data is fed to the algorithm, the higher the accuracy of the algorithm is. It is also the key component in technologies for automation.

Disadvantages:

This model is developed using Machine Learning in which human involvement is very less and might cause some errors and if any error occurs it takes a lot of time for the developer to identify the root cause.

7. CONCLUSION & FUTURE SCOPE

Machine learning techniques offer a feasible and promising approach to predicting life expectancy. The research has potential for real-life applications, such as supporting timely recognition of the right moment to start Advance Care Planning. This breakthrough can widely impact health sectors and economic sectors by improving the resources, funds and services provided to the common people. It can also increase the ease of access to the individuals.

The scalability and flexibility of the application can also be improved with advancement in technology and availability of new and improved resources. Also, with the growth in Artificial Neural networks and Deep learning, one can integrate that with our existing application. With the help of Convolutional Neural networks and Computer vision, we can also try to take into account the physical health and appearance of a person. Mental health can also be taken into account while predicting life expectancy with the help of sentiment analysis systems as well.

BIBLIOGRAPHY

1. https://www.kaggle.com/kumarajarshi/life-expectancy-who
2. https://en.wikipedia.org/wiki/Life_expectancy
3. https://www.youtube.com/watch?v=LOCkV-mENq8&feature=youtu.be
4. https://www.w3schools.com/howto/howto_make_a_website.asp
5. https://www.datasciencesociety.net/using-machine-learning-to-explain-and-predict-the-life-expectancy-of-different-countries/