Project Report

Predicting Life Expectancy Using Machine Learning

Name: Shreyansh Shukla

Email : shreyanshshuklashukla@gmail.com

Project ID: SPS_PRO_215

Internship Title : Predicting Life Expectancy using Machine Learning - SB40253

Internship Under : Smartbridge Educational Services PVT LTD.

GitHub: https://github.com/SmartPracticeschool/llSPS-INT-2654-Predicting-Life-Expectancy-using-Machine-Learning

Note Book Link:

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/a05f232a-94ef-4148-bd9d-43352c744005/view?access_token=a0a63ac754f32f0206ca675f7e3ff4c85372e0b02c704bd7e87a32a9b893c356

Node Red Link: https://node-red-shreyansh.eu-gb.mybluemix.net/ui/#!/0?socketid=aLKSXaMfQ3ZNzp8hAAAD

1 INTRODUCTION

1.1 Overview

1.2 Purpose

2 LITERATURE SURVEY

2.1 Existing problem

2.2 Proposed solution

3 THEORITICAL ANALYSIS

3.1 Block diagram

3.2 Hardware / Software designing

4 EXPERIMENTAL INVESTIGATIONS

5 FLOWCHART

6 RESULT

7 ADVANTAGES & DISADVANTAGES

8 APPLICATIONS

9 CONCLUSION

10 FUTURE SCOPE

11 BIBILOGRAPHY

APPENDIX

A. Source code

PREDICTING LIFE EXPECTANCY USING MACHINE LEARNING

1. INTRODUCTION:

1.1 Overview:

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.

The life of a human depends on various factors such as Regional variations, Economic circumstances, Sex Differences, Mental Illnesses, Physical illnesses, Education, Year of their birth and other demographic factors.

The end product will be a webpage where you need to give all the required inputs and then submit it. Afterwards it will predict the life expectancy value based on your regression technique.

The dataset used for the prediction contains data from year 2000 to 2015 for 193 countries. It contains more than 2500 entries and around 22 columns with various features such as Population, Alcohol Consumption, Infant Mortality Rate etc., which aids the prediction of the model.

1.2 Purpose:

Life expectancy is a statistical measure of the average time a human being is expected to live, Life expectancy depends on various factors: Regional variations, Economic Circumstances, Sex Differences, Mental Illnesses, Physical Illnesses, Education, Year of their birth and other demographic factors.

If life expectancy is longer in a certain country, it speaks about the conditions of the place. It tells information on the health factors as well as the quality of life. If the conditions in a country and in its economy are good, obviously the life expectancy would be more and greater number of people would like to live in the same country. Life expectancy is the most important factor for decision making

By predicting life expectancy and having good prognostication can help in making valuable decision like the course of treatment and helps to anticipate the procurement of health care services and facilities.

2. LITERATURE SURVEY:

2.1 Existing Problem:

A typical Regression Machine Learning project leverages historical data to predict insights into the future. This problem statement is aimed at predicting Life Expectancy rate of a country given various features.

Life expectancy is a statistical measure of the average time a human being is expected to live, Life expectancy depends on various factors: Regional variations, Economic Circumstances, Sex Differences, Mental Illnesses, Physical Illnesses, Education, Year of their birth and other demographic factors. This problem statement provides a way to predict average life expectancy of people living in a country when various factors such as year, GDP, education, alcohol intake of people in the country, expenditure on healthcare system and some specific disease related deaths that happened in the country are given.

Predicting Life Expectancy has been a long-term question to humankind. Many calculations and Research have been done to create an equation despite it being impractical to simplify these variables into one equation.

2.2. Proposed Solution:

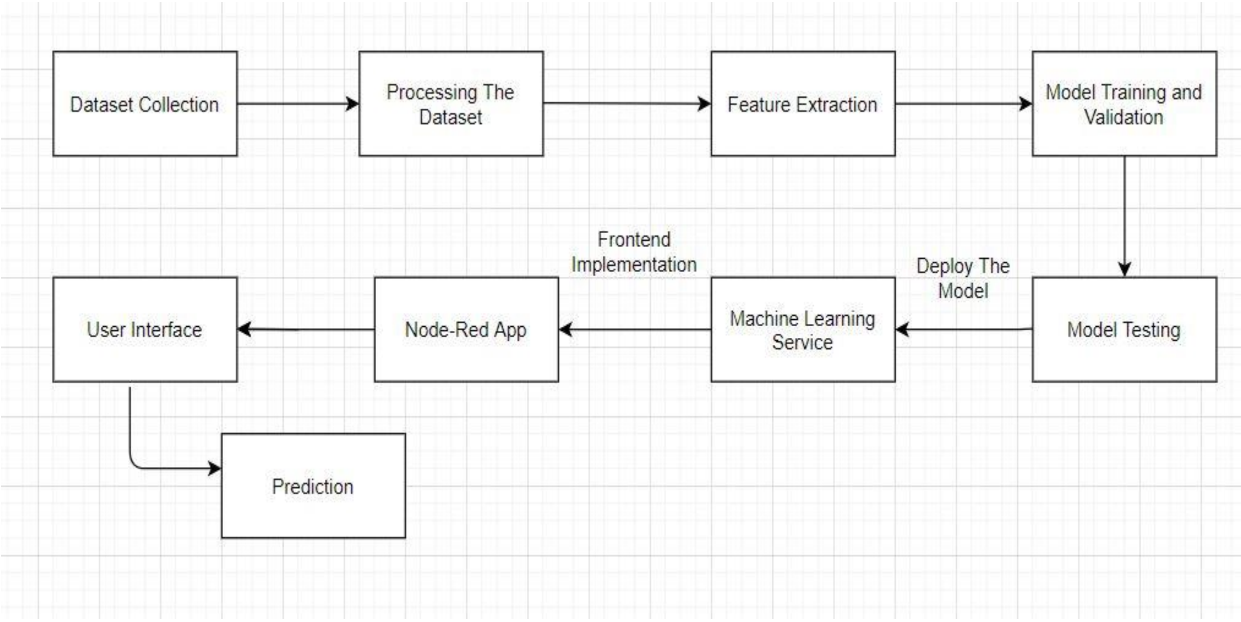
The Output and insights generated from the model will help to get better clinical understanding and more insights.

The proposed solution involves the use of Machine Learning Algorithms (Regression Algorithms). Here we propose a method for forecasting Life Expectancy of an individual from a country considering certain factors such as Status of the country, Adult Mortality Rate, Infant deaths, Alcohol, Hepatitis B, Measles, BMI, Polio, Total Expenditure, Diphtheria, HIV/AIDS, GDP of a country, Population, Income Composition of Resources, Schooling status of the country.

To access the trained model, we will use Node-Red App from IBM Cloud.

3. Theoretical Analysis:

3.1 Block Diagram:



3.2 Hardware / Software designing:

1. Collecting the Dataset
2. Creating Necessary IBM Cloud Service
3. Creating and Configuring Watson Studio
4. Create Machine Learning Service
5. Adding Jupyter Notebook
6. Build ML model and create Scoring Endpoint for Node-Red Integration
7. Build Node-Red Flow and integrate ML services and deploy.

4. EXPERIMENTAL INVESTIGATIONS:

This Project aims to predict Life Expectancy of a human in any Country. Whole Project is based on the dataset accuracy. Thus, the data set has been taken from WHO, which was provided publicly.

The 21 factors which are taken into account for predicting the life expectancy of a country are as follows:

1. Country

2. Year

3. Status: Developed or Developing status of the country.

4. Adult mortality: Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population).

5. Infant deaths: Number of Infant Deaths per 1000 population.

6. Alcohol: Alcohol, recorded per capita (15+) consumption.

7. Percentage Expenditure: Expenditure on health as a percentage of Gross Domestic Product per capita (%).

8. Hepatitis B: Immunization coverage among 1-year-olds (%).

9. Measles: Number of reported cases per 1000 population.

10. BMI: Average Body Mass Index of entire population.

11. Under-five deaths: Number of under-five deaths per 1000 population.

12. Polio: Immunization coverage among 1-year-olds (%).

13. Total expenditure: General government expenditure on health as a percentage of total government expenditure (%).

14. Diphtheria: Diphtheria tetanus toxoid and pertussis (DTP3) immunization coverage among 1-year olds (%).

15. HIV/AIDS: Deaths per 1 000 live births HIV/AIDS (0-4 years).

16. GDP: Gross Domestic Product per capita (in USD).

17. Population: Population of the country.

18. Thinness 10-19 years: Prevalence of thinness among children and adolescents for Age 10 to 19 (%).

19. Thinness 5-9 years: Prevalence of thinness among children for Age 5 to 9 (%).

20. Income composition of resources: Human Development Index in terms of income composition of resources (index ranging from 0 to 1).

21. Schooling: Number of years of schooling

Algorithm Used :- <u>Random Forest Regression</u>

Analysing the Features:

In [5]:
```
data.describe()
```

Out[5]:

	Year	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths
count	2938.000000	2928.000000	2928.000000	2938.000000	2744.000000	2938.000000	2385.000000	2938.000000	2904.000000	2938.000000
mean	2007.518720	69.224932	164.796448	30.303948	4.602861	738.251295	80.940461	2419.592240	38.321247	42.035739
std	4.613841	9.523867	124.292079	117.926501	4.052413	1987.914858	25.070016	11467.272489	20.044034	160.445548
min	2000.000000	36.300000	1.000000	0.000000	0.010000	0.000000	1.000000	0.000000	1.000000	0.000000
25%	2004.000000	63.100000	74.000000	0.000000	0.877500	4.685343	77.000000	0.000000	19.300000	0.000000
50%	2008.000000	72.100000	144.000000	3.000000	3.755000	64.912906	92.000000	17.000000	43.500000	4.000000
75%	2012.000000	75.700000	228.000000	22.000000	7.702500	441.534144	97.000000	360.250000	56.200000	28.000000
max	2015.000000	89.000000	723.000000	1800.000000	17.870000	19479.911610	99.000000	212183.000000	87.300000	2500.000000

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
Country                            2938 non-null object
Year                               2938 non-null int64
Status                             2938 non-null object
Life expectancy                    2928 non-null float64
Adult Mortality                    2928 non-null float64
infant deaths                      2938 non-null int64
Alcohol                            2744 non-null float64
percentage expenditure             2938 non-null float64
Hepatitis B                        2385 non-null float64
Measles                            2938 non-null int64
 BMI                               2904 non-null float64
under-five deaths                  2938 non-null int64
Polio                              2919 non-null float64
Total expenditure                  2712 non-null float64
Diphtheria                         2919 non-null float64
 HIV/AIDS                          2938 non-null float64
GDP                                2490 non-null float64
Population                         2286 non-null float64
 thinness  1-19 years              2904 non-null float64
 thinness 5-9 years                2904 non-null float64
Income composition of resources    2771 non-null float64
Schooling                          2775 non-null float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.0+ KB
```

In [7]:
```
data.size
```

Out[7]: 64636

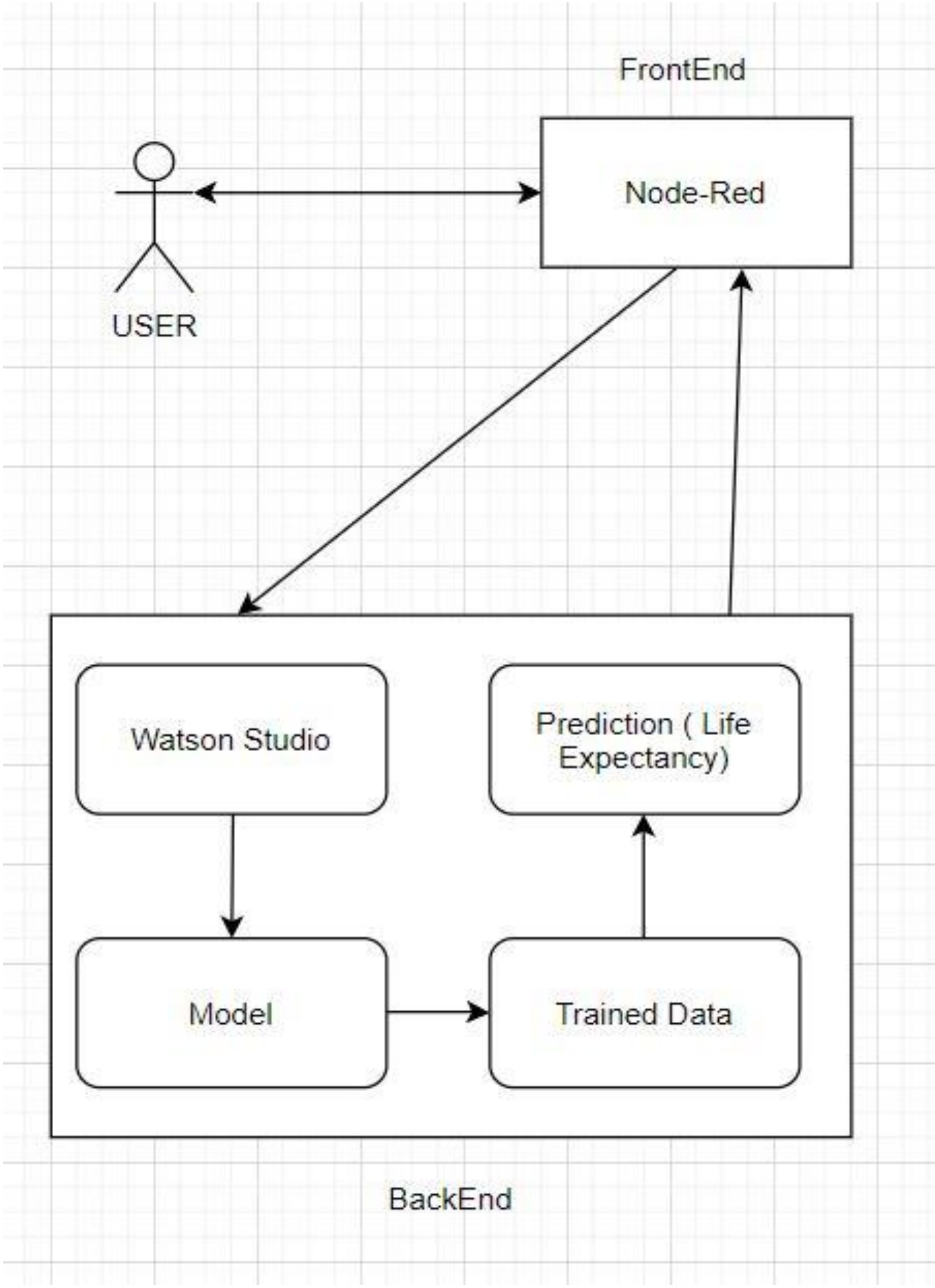
In [8]:
```
data.columns
```

Out[8]:
```
Index(['Country', 'Year', 'Status', 'Life expectancy ', 'Adult Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles ', ' BMI ', 'under-five deaths ', 'Polio', 'Total expenditure',
       'Diphtheria ', ' HIV/AIDS', 'GDP', 'Population',
       ' thinness  1-19 years', ' thinness 5-9 years',
       'Income composition of resources', 'Schooling'],
      dtype='object')
```

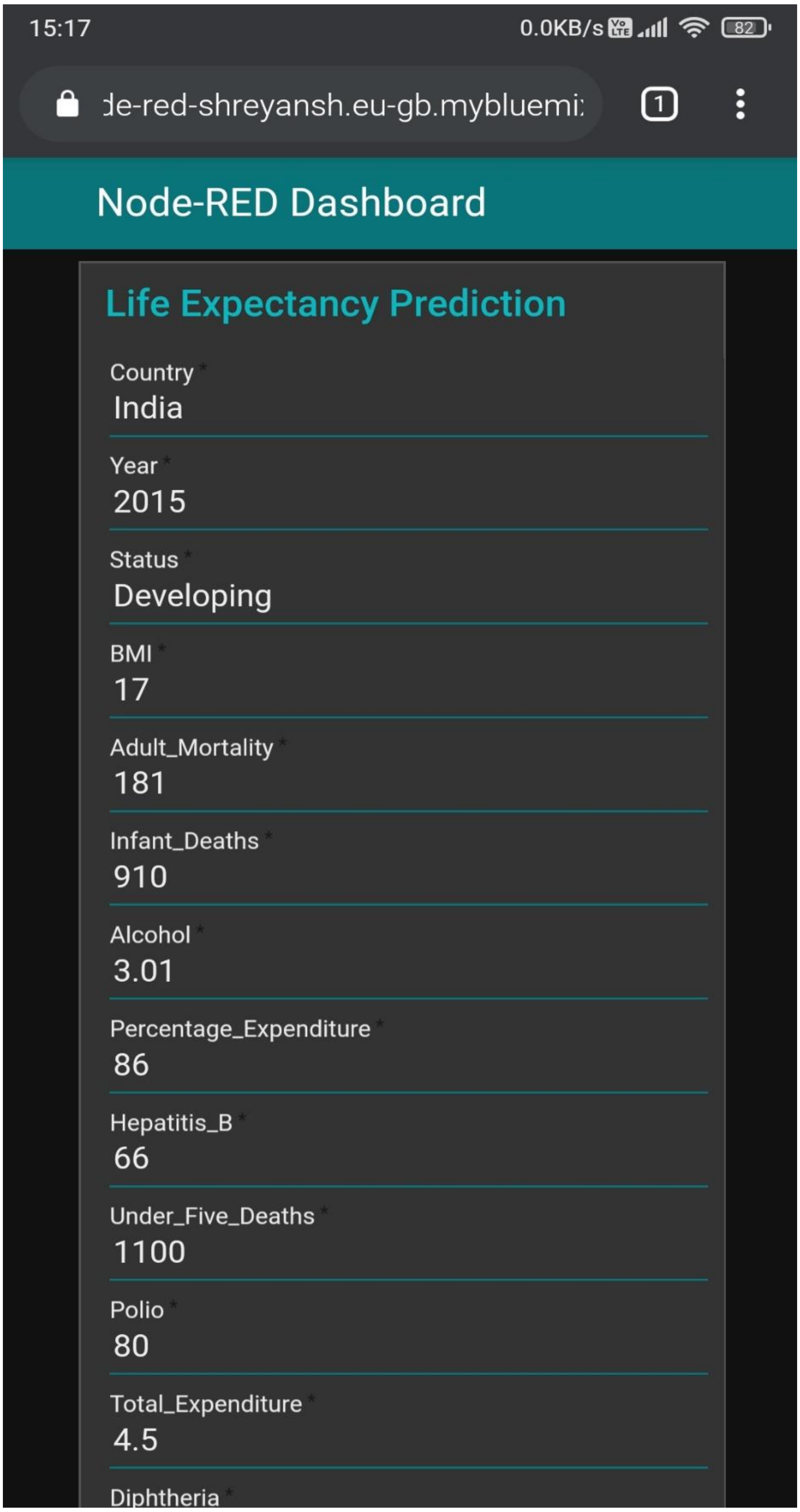
In [9]:
```
data.isnull().sum()
```

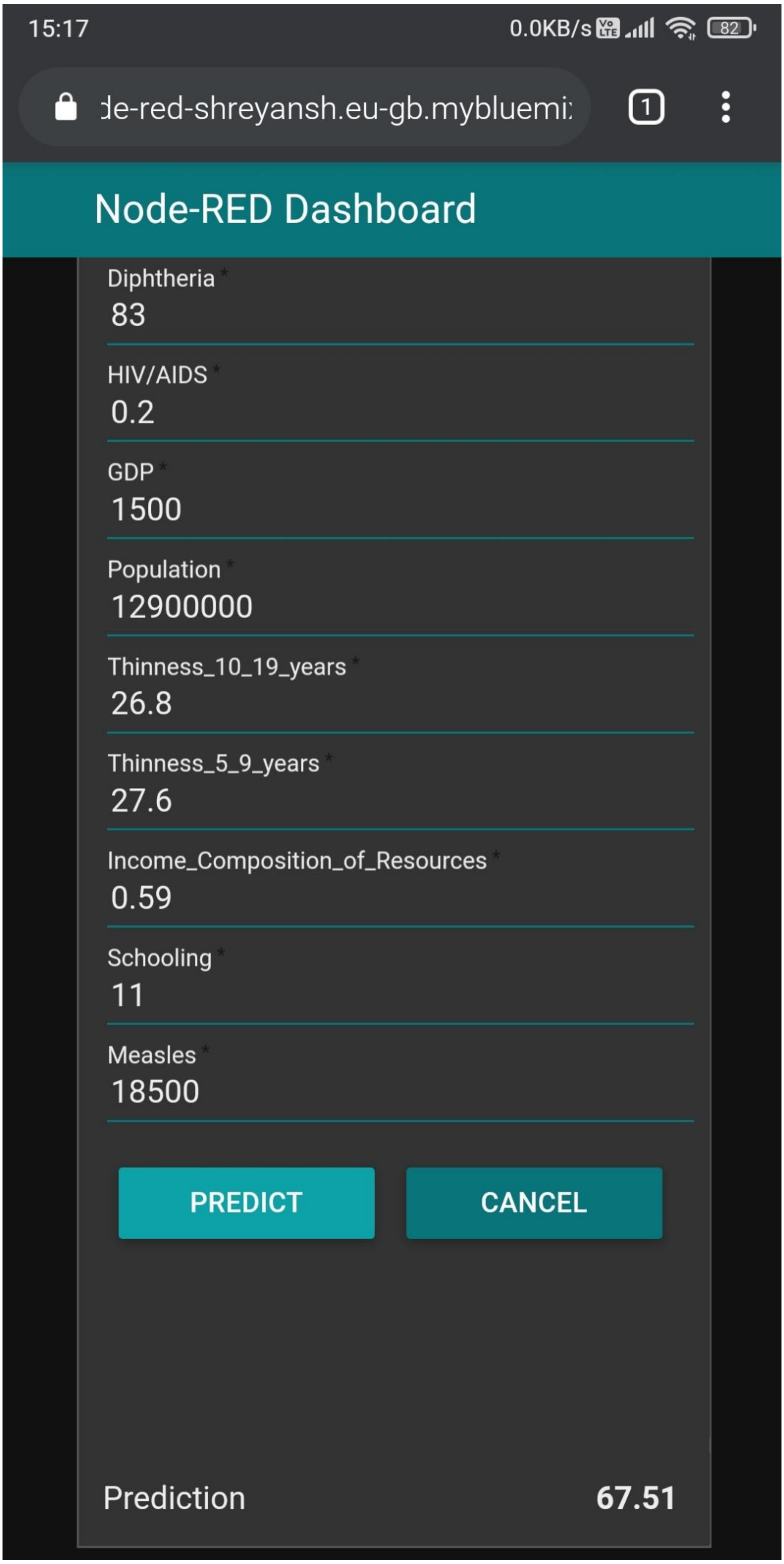
Out[9]:
```
Country                              0
Year                                 0
Status                               0
Life expectancy                     10
Adult Mortality                     10
infant deaths                        0
Alcohol                            194
percentage expenditure               0
Hepatitis B                        553
Measles                              0
 BMI                                34
under-five deaths                    0
Polio                               19
Total expenditure                  226
Diphtheria                          19
 HIV/AIDS                            0
GDP                                448
Population                         652
 thinness  1-19 years               34
 thinness 5-9 years                 34
Income composition of resources    167
Schooling                          163
dtype: int64
```

5. Flowchart:


FrontEnd
Node-Red
USER
Watson Studio
Prediction (Life Expectancy)
Model
Trained Data
BackEnd

6.Result:


15:17
0.0KB/s
de-red-shreyansh.eu-gb.mybluemi
Node-RED Dashboard
Life Expectancy Prediction
Country
India
Year
2015
Status
Developing
BMI
17
Adult_Mortality
181
Infant_Deaths
910
Alcohol
3.01
Percentage_Expenditure
86
Hepatitis_B
66
Under_Five_Deaths
1100
Polio
80
Total_Expenditure
4.5
Diphtheria


15:17
0.0KB/s
82
de-red-shreyansh.eu-gb.mybluemi
1
Node-RED Dashboard
Diphtheria *
83
HIV/AIDS *
0.2
GDP *
1500
Population *
12900000
Thinness_10_19_years *
26.8
Thinness_5_9_years *
27.6
Income_Composition_of_Resources *
0.59
Schooling *
11
Measles *
18500
PREDICT
CANCEL
Prediction
67.51

7. Advantages and Disadvantages:

Advantages :

- Our Application's Performance Efficiency and performance Improves over time because we deployed it in ML model and ML algorithms tend to improve over time.
- Easy for user to interact with the model via the UI: This interface requires no background knowledge of how to use it. It's a simple interface and only ask for required values and predict the output.
- Easy to build and deploy: Node-Red eases out the task of creating a front end. It connects well with our ML service through scoring endpoints.
- Doesn't require much storage space.

Disadvantages:

- Node-Red doesn't give much flexibility to design own templates, although it's a great service.
- If the data set is very big in size, computational cost would increase by a lot and training the model will incur more cost.
- Wrong Prediction: As it depends completely on user, so if user provides some wrong values then it will predict wrong value.
- Requires Internet Connection.

8. Applications:

- To analyse all the factors and plan out measures to increase the life expectancy of the country
- To help government prepare life insurance policies for people. This will benefit the people.
- To analyze country’s growth statistics in future years.
- It can be used to monitor health inequalities of a country.
- This will help in suggesting a country which area should be given importance in order to efficiently improve the life expectancy of its population.

9. Conclusion:

- In this Project, we developed a Machine Learning Model to predict Life Expectancy of humans in a country.
- Predicting Life Expectancy can lead to the development of the country. It can widely impact Health Sectors, Public Sectors and Economic Sectors by improving the resources, funds and services provided to people.

10. Future Scope:

- With increase in Data Set, more insightful prediction can be made.
- Other factors such as sentiment analysis and mental health can be added to predict life expectancy.
- Happiness index is also one such feature which can be proved vital in determining life expectancy

11. Bibliography :

- Project Planning and Kick-off:
  https://www.youtube.com/watch?v=LOCkV-mENq8&feature=youtu.be
  https://www.allbusinesstemplates.com/download/?filecode=2KBA4&lang=en&iuid=9f9faa69-9fab-40ee-8457-ea0e5df8c8de
- Node-Red starter tutorial:
  https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/
- Introductory workshop for Watson Studio Cloud:
  https://bookdown.org/caoying4work/watsonstudio-workshop/jn.html
- AutoAI References:
  https://developer.ibm.com/tutorials/watson-studio-auto-ai/
  https://www.youtube.com/watch?v=IDKCmC1fCiU
- Dataset : From Kaggle
  https://www.kaggle.com/kumarajarshi/life-expectancy-who
- Creating and Importing dataset in Jupyter Notebook:
  https://www.youtube.com/watch?v=Jtej3Y6uUng

Appendix

A) Source Code:

1) Data Set

Link : https://www.kaggle.com/kumarajarshi/life-expectancy-who

A1 Country

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Country	Year	Status	Life expec	Adult Mor	infant dea	Alcohol	percentag	Hepatitis E	Measles	BMI	under-five	Polio	Total expe	Diphtheria	HIV/AIDS	GDP	Populatior	thinness :	thinness 5	Income co	Schooling
2	Afghanista	2015	Developin	65	263	62	0.01	71.27962	65	1154	19.1	83	6	8.16	65	0.1	584.2592	33736494	17.2	17.3	0.479	10.1
3	Afghanista	2014	Developin	59.9	271	64	0.01	73.52358	62	492	18.6	86	58	8.18	62	0.1	612.6965	327582	17.5	17.5	0.476	10
4	Afghanista	2013	Developin	59.9	268	66	0.01	73.21924	64	430	18.1	89	62	8.13	64	0.1	631.745	31731688	17.7	17.7	0.47	9.9
5	Afghanista	2012	Developin	59.5	272	69	0.01	78.18422	67	2787	17.6	93	67	8.52	67	0.1	669.959	3696958	17.9	18	0.463	9.8
6	Afghanista	2011	Developin	59.2	275	71	0.01	7.097109	68	3013	17.2	97	68	7.87	68	0.1	63.53723	2978599	18.2	18.2	0.454	9.5
7	Afghanista	2010	Developin	58.8	279	74	0.01	79.67937	66	1989	16.7	102	66	9.2	66	0.1	553.3289	2883167	18.4	18.4	0.448	9.2
8	Afghanista	2009	Developin	58.6	281	77	0.01	56.76222	63	2861	16.2	106	63	9.42	63	0.1	445.8933	284331	18.6	18.7	0.434	8.9
9	Afghanista	2008	Developin	58.1	287	80	0.03	25.87393	64	1599	15.7	110	64	8.33	64	0.1	373.3611	2729431	18.8	18.9	0.433	8.7
10	Afghanista	2007	Developin	57.5	295	82	0.02	10.91016	63	1141	15.2	113	63	6.73	63	0.1	369.8358	26616792	19	19.1	0.415	8.4
11	Afghanista	2006	Developin	57.3	295	84	0.03	17.17152	64	1990	14.7	116	58	7.43	58	0.1	272.5638	2589345	19.2	19.3	0.405	8.1
12	Afghanista	2005	Developin	57.3	291	85	0.02	1.388648	66	1296	14.2	118	58	8.7	58	0.1	25.29413	257798	19.3	19.5	0.396	7.9
13	Afghanista	2004	Developin	57	293	87	0.02	15.29607	67	466	13.8	120	5	8.79	5	0.1	219.1414	24118979	19.5	19.7	0.381	6.8
14	Afghanista	2003	Developin	56.7	295	87	0.01	11.08905	65	798	13.4	122	41	8.82	41	0.1	198.7285	2364851	19.7	19.9	0.373	6.5
15	Afghanista	2002	Developin	56.2	3	88	0.01	16.88735	64	2486	13	122	36	7.76	36	0.1	187.846	21979923	19.9	2.2	0.341	6.2
16	Afghanista	2001	Developin	55.3	316	88	0.01	10.57473	63	8762	12.6	122	35	7.8	33	0.1	117.497	2966463	2.1	2.4	0.34	5.9
17	Afghanista	2000	Developin	54.8	321	88	0.01	10.42496	62	6532	12.2	122	24	8.2	24	0.1	114.56	293756	2.3	2.5	0.338	5.5
18	Albania	2015	Developin	77.8	74	0	4.6	364.9752	99	0	58	0	99	6	99	0.1	3954.228	28873	1.2	1.3	0.762	14.2
19	Albania	2014	Developin	77.5	8	0	4.51	428.7491	98	0	57.2	1	98	5.88	98	0.1	4575.764	288914	1.2	1.3	0.761	14.2
20	Albania	2013	Developin	77.2	84	0	4.76	430.877	99	0	56.5	1	99	5.66	99	0.1	4414.723	289592	1.3	1.4	0.759	14.2
21	Albania	2012	Developin	76.9	86	0	5.14	412.4434	99	9	55.8	1	99	5.59	99	0.1	4247.614	2941	1.3	1.4	0.752	14.2
22	Albania	2011	Developin	76.6	88	0	5.37	437.0621	99	28	55.1	1	99	5.71	99	0.1	4437.179	295195	1.4	1.5	0.738	13.3
23	Albania	2010	Developin	76.2	91	1	5.28	41.82276	99	10	54.3	1	99	5.34	99	0.1	494.3588	291321	1.4	1.5	0.725	12.5
24	Albania	2009	Developin	76.1	91	1	5.79	348.056	98	0	53.5	1	98	5.79	98	0.1	4114.137	2927519	1.5	1.6	0.721	12.2
25	Albania	2008	Developin	75.3	1	1	5.61	36.62207	99	0	52.6	1	99	5.87	99	0.1	437.5396	2947314	1.6	1.6	0.713	12
26	Albania	2007	Developin	75.9	9	1	5.58	32.24655	98	22	51.7	1	99	6.1	98	0.1	363.1369	29717	1.6	1.7	0.703	11.6
27	Albania	2006	Developin	74.2	99	1	5.31	3.302154	98	68	5.8	1	97	5.86	97	0.1	35.1293	2992547	1.7	1.8	0.696	11.4
28	Albania	2005	Developin	73.5	15	1	5.16	26.99312	98	6	49.9	1	97	6.12	98	0.1	279.1429	311487	1.8	1.8	0.685	10.8
29	Albania	2004	Developin	73	17	1	4.54	221.8428	99	7	48.9	1	98	6.38	97	0.1	2416.588	326939	1.8	1.9	0.681	10.9

Life Expectancy Data

2) Life Expectancy Notebook Code:

Analysing the dataset

Importing required libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.compose import ColumnTransformer, make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.impute import SimpleImputer
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.metrics import accuracy_score
from collections import OrderedDict
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import r2_score,mean_squared_error
```

Reading the dataset in IBM Watson Studio

```
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_d6a042f58dc44bfcac01d1a01afd0d38 =
ibm_boto3.client(service_name='s3',
    ibm_api_key_id='3okOdlerylavk0PK0-Xd-r5HYVMO2iegQ87elISoTdbn',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.eu-
geo.objectstorage.service.networklayer.com')

body =
client_d6a042f58dc44bfcac01d1a01afd0d38.get_object(Bucket='lifeexpectancy-
donotdelete-pr-6bb2hoexroj1xl',Key='Life_Expectancy_Data.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(
__iter__, body )

data = pd.read_csv(body)
data.head()
```

```
data.head()
data.shape
data.describe()
data.info()
data.size
data.columns
data.isnull().sum()
```

Handling Missing Value

```
country_list = data.Country.unique()
len(country_list)
country_list = data.Country.unique()
fill_list = ['Country', 'Year', 'Status', 'Life expectancy ', 'Adult
Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles ', ' BMI ', 'under-five deaths ', 'Polio', 'Total
expenditure',
       'Diphtheria ', ' HIV/AIDS', 'GDP', 'Population',
       ' thinness  1-19 years', ' thinness 5-9 years',
```

```
      'Income composition of resources', 'Schooling']
```

Filling missing value according to country column using interpolate()

```
for country in country_list:
    data.loc[data['Country'] == country,fill_list] =
data.loc[data['Country'] == country,fill_list].interpolate()
data.dropna(inplace=True)
data.shape
data.isna().sum()
```

Corelation matrix

```
corrMatrix = data.corr()
corrMatrix.style.background_gradient(cmap='plasma', low=.5,
high=0).highlight_null('red')
```

Renaming the columns as it contains trailing spaces

```
data.rename(columns={" BMI ":"BMI",'Life expectancy ':'Life expectancy',
                  "under-five deaths ":"under-five deaths","Measles
":"Measles","Diphtheria ":"Diphtheria",
                  ' HIV/AIDS':"HIV/AIDS",
                  " thinness  1-19 years":"thinness 10-19 years"," thinness
5-9 years":"thinness 5-9 years"},inplace=True)
```

Removing outliers

Taking numeric features , (country,year, status columns are excluded)

```
col_dict = {'Life expectancy':1 , 'Adult Mortality':2 ,
        'Alcohol':3 , 'percentage expenditure': 4, 'Hepatitis B': 5,
       'Measles' : 6, 'BMI': 7, 'under-five deaths' : 8, 'Polio' : 9,
'Total expenditure' :10,
       'Diphtheria':11, 'HIV/AIDS':12, 'GDP':13, 'Population' :14,
       'thinness 10-19 years' :15, 'thinness 5-9 years' :16,
       'Income composition of resources' : 17, 'Schooling' :18, 'infant
deaths':19}
```

Showing outliers using box plot

```
import matplotlib.pyplot as plt
plt.figure(figsize=(20,30))

for variable,i in col_dict.items():
                     plt.subplot(5,4,i)
                     plt.boxplot(data[variable],whis=1.5)
                     plt.title(variable)

plt.show()
```

BMI has no outliers

```
import numpy as np

for variable in col_dict.keys():
    q75, q25 = np.percentile(data[variable], [75 ,25])
    iqr = q75 - q25
    min_val = q25 - (iqr*1.5)
    max_val = q75 + (iqr*1.5)
    print("Number of outliers and percentage of it in {} : {} and
{}".format(variable,
```

```
len((np.where((data[variable] > max_val) | (data[variable] <
min_val))[0])),

len((np.where((data[variable] > max_val) | (data[variable] <
min_val))[0]))*100/1987))
```

```
from scipy.stats.mstats import winsorize
winsorized_Life_Expectancy = winsorize(data['Life expectancy'],(0.01,0))
winsorized_Adult_Mortality = winsorize(data['Adult Mortality'],(0,0.03))
winsorized_Infant_Deaths = winsorize(data['infant deaths'],(0,0.10))
winsorized_Alcohol = winsorize(data['Alcohol'],(0,0.01))
winsorized_Percentage_Exp = winsorize(data['percentage
expenditure'],(0,0.12))
winsorized_HepatitisB = winsorize(data['Hepatitis B'],(0.11,0))
winsorized_Measles = winsorize(data['Measles'],(0,0.19))
winsorized_Under_Five_Deaths = winsorize(data['under-five
deaths'],(0,0.12))
winsorized_Polio = winsorize(data['Polio'],(0.09,0))
winsorized_Tot_Exp = winsorize(data['Total expenditure'],(0,0.01))
winsorized_Diphtheria = winsorize(data['Diphtheria'],(0.10,0))
winsorized_HIV = winsorize(data['HIV/AIDS'],(0,0.16))
winsorized_GDP = winsorize(data['GDP'],(0,0.13))
winsorized_Population = winsorize(data['Population'],(0,0.14))
winsorized_thinness_10_19_years = winsorize(data['thinness 10-19
years'],(0,0.04))
winsorized_thinness_5_9_years = winsorize(data['thinness 5-9
years'],(0,0.04))
winsorized_Income_Comp_Of_Resources = winsorize(data['Income composition of
resources'],(0.05,0))
winsorized_Schooling = winsorize(data['Schooling'],(0.02,0.01))
```

```
winsorized_list =
[winsorized_Life_Expectancy,winsorized_Adult_Mortality,winsorized_Alcohol,w
insorized_Measles,winsorized_Infant_Deaths,

winsorized_Percentage_Exp,winsorized_HepatitisB,winsorized_Under_Five_Death
s,winsorized_Polio,winsorized_Tot_Exp,winsorized_Diphtheria,

winsorized_HIV,winsorized_GDP,winsorized_Population,winsorized_thinness_10_
19_years,winsorized_thinness_5_9_years,
            winsorized_Income_Comp_Of_Resources,winsorized_Schooling]

for variable in winsorized_list:
    q75, q25 = np.percentile(variable, [75 ,25])
    iqr = q75 - q25

    min_val = q25 - (iqr*1.5)
    max_val = q75 + (iqr*1.5)

    print("Number of outliers after winsorization in  : {}
".format(len(np.where((variable > max_val) | (variable < min_val))[0])))
```

Adding 18 new columns having no outliers to the dataframe

```
data['winsorized_Life_Expectancy'] = winsorized_Life_Expectancy
data['winsorized_Adult_Mortality'] = winsorized_Adult_Mortality
data['winsorized_Infant_Deaths'] = winsorized_Infant_Deaths
data['winsorized_Alcohol'] = winsorized_Alcohol
data['winsorized_Percentage_Exp'] = winsorized_Percentage_Exp
data['winsorized_HepatitisB'] = winsorized_HepatitisB
data['winsorized_Under_Five_Deaths'] = winsorized_Under_Five_Deaths
data['winsorized_Polio'] = winsorized_Polio
data['winsorized_Tot_Exp'] = winsorized_Tot_Exp
data['winsorized_Diphtheria'] = winsorized_Diphtheria
data['winsorized_HIV'] = winsorized_HIV
data['winsorized_GDP'] = winsorized_GDP
data['winsorized_Population'] = winsorized_Population
data['winsorized_thinness_10_19_years'] = winsorized_thinness_10_19_years
data['winsorized_thinness_5_9_years'] = winsorized_thinness_5_9_years
data['winsorized_Income_Comp_Of_Resources'] =
winsorized_Income_Comp_Of_Resources
data['winsorized_Schooling'] = winsorized_Schooling
data['winsorized_Measles'] = winsorized_Measles
```

```
data.shape #More 18 columns are added
```

Exploratory Data Analysis (EDA)

```
data.columns
sns.distplot(data['Life expectancy'],kde=True)
disease_cols=data[['Life expectancy','Alcohol','Hepatitis
B','Measles','BMI','Polio','Diphtheria','HIV/AIDS','Adult Mortality',
                   'infant deaths','under-five deaths','thinness 10-19
years','thinness 5-9 years','Schooling',
                   'percentage expenditure','Total
expenditure','GDP','Population','Income composition of resources']]
disease_cols.corr()
sns.pairplot(disease_cols,diag_kind='kde')
```

Hence all the features are significant to predict the target variable

```
col = ['Life expectancy','winsorized_Life_Expectancy','Adult
Mortality','winsorized_Adult_Mortality','infant deaths',

'winsorized_Infant_Deaths','Alcohol','winsorized_Alcohol','percentage
expenditure','winsorized_Percentage_Exp','Hepatitis B',
         'winsorized_HepatitisB','under-five
deaths','winsorized_Under_Five_Deaths','Polio','winsorized_Polio','Total
expenditure',

'winsorized_Tot_Exp','Diphtheria','winsorized_Diphtheria','HIV/AIDS','winso
rized_HIV','GDP','winsorized_GDP',
         'Population','winsorized_Population','thinness 10-19
years','winsorized_thinness_10_19_years','thinness 5-9 years',
         'winsorized_thinness_5_9_years','Income composition of
resources','winsorized_Income_Comp_Of_Resources',

'Schooling','winsorized_Schooling','Measles','winsorized_Measles','GDP','wi
nsorized_GDP']
```

```
plt.figure(figsize=(15,75))

for i in range(len(col)):
    plt.subplot(19,2,i+1)
    plt.hist(data[col[i]])
    plt.title(col[i])

plt.show()
```

```
data.describe(include= 'O')

plt.figure(figsize=(6,6))
plt.bar(data.groupby('Status')['Status'].count().index,data.groupby('Status
')['winsorized_Life_Expectancy'].mean())
plt.ylabel("Avg Life_Expectancy")
plt.title("Life_Expectancy w.r.t Status")
plt.show()

le_country =
data.groupby('Country')['winsorized_Life_Expectancy'].mean().sort_values(as
cending=True)
le_country.plot(kind='bar', figsize=(50,15), fontsize=25)
plt.title("Life_Expectancy w.r.t Country",fontsize=40)
plt.xlabel("Country",fontsize=35)
plt.ylabel("Avg Life_Expectancy",fontsize=35)
plt.show()

plt.figure(figsize=(7,5))
plt.bar(data.groupby('Year')['Year'].count().index,data.groupby('Year')['wi
nsorized_Life_Expectancy'].mean())
plt.xlabel("Year",fontsize=12)
plt.ylabel("Avg Life_Expectancy",fontsize=12)
plt.title("Life_Expectancy w.r.t Year")
plt.show()

cor_matrix=data.corr()
print(cor_matrix['winsorized_Life_Expectancy'].sort_values(ascending=False)
)

round(data[['Status','winsorized_Life_Expectancy']].groupby(['Status']).mea
n(),2)
```

Since 'status' is a categorical feature, we have to find the correlation with Life expectancy

```
import scipy.stats as stats
stats.ttest_ind(data.loc[data['Status']=='Developed','winsorized_Life_Expec
tancy'],data.loc[data['Status']=='Developing','winsorized_Life_Expectancy']
)
data.columns
```

Now our data has no null values and no outliers

Creating a new dataframe with refined data

```
new_data=pd.DataFrame(data=data,columns=['Country', 'Year', 'Status',
         'BMI', 'winsorized_Adult_Mortality',
        'winsorized_Infant_Deaths', 'winsorized_Alcohol',
        'winsorized_Percentage_Exp', 'winsorized_HepatitisB',
        'winsorized_Under_Five_Deaths', 'winsorized_Polio',
        'winsorized_Tot_Exp', 'winsorized_Diphtheria', 'winsorized_HIV',
        'winsorized_GDP', 'winsorized_Population',
        'winsorized_thinness_10_19_years', 'winsorized_thinness_5_9_years',
        'winsorized_Income_Comp_Of_Resources', 'winsorized_Schooling',
        'winsorized_Measles',
        'winsorized_Life_Expectancy'])
```

```
new_data.shape
new_data.head()
```

```
new_data.rename(columns={
              'winsorized_Adult_Mortality':'Adult_Mortality',
        'winsorized_Infant_Deaths' :'Infant_Deaths',
        'winsorized_Alcohol':'Alcohol',
        'winsorized_Percentage_Exp':'Percentage_Expenditure',
        'winsorized_HepatitisB':'Hepatitis_B',
        'winsorized_Under_Five_Deaths':'Under_Five_Deaths',
        'winsorized_Polio':'Polio',
        'winsorized_Tot_Exp':'Total_Expenditure',
        'winsorized_Diphtheria':'Diphtheria',
        'winsorized_HIV':'HIV/AIDS',
        'winsorized_GDP':'GDP',
        'winsorized_Population':'Population',
        'winsorized_thinness_10_19_years':'Thinness_10_19_years',
        'winsorized_thinness_5_9_years':'Thinness_5_9_years',

'winsorized_Income_Comp_Of_Resources':'Income_Composition_of_Resources',
        'winsorized_Schooling':'Schooling',
        'winsorized_Measles':'Measles',
        'winsorized_Life_Expectancy':'Life_Expectancy' } ,inplace=True)
```

```
new_data.head()
new_data.columns
```

Separating the input features and label

```
X = new_data.drop('Life_Expectancy', axis=1)
Y = pd.DataFrame(data=new_data,columns=['Life_Expectancy'])
X.head()
Y.head()
```

Splitting the data into train set and test set

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2,
random_state = 42)
```

Creating a pipeline

```
numeric_features = ['Year', 'BMI',
       'Adult_Mortality', 'Infant_Deaths', 'Alcohol',
'Percentage_Expenditure',
       'Hepatitis_B', 'Under_Five_Deaths', 'Polio', 'Total_Expenditure',
       'Diphtheria', 'HIV/AIDS', 'GDP', 'Population',
'Thinness_10_19_years',
       'Thinness_5_9_years', 'Income_Composition_of_Resources',
'Schooling',
       'Measles']
categorical_features = ['Country', 'Status']

from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder

categorical_transformer = Pipeline(steps=[
    ('onehot', OneHotEncoder(handle_unknown='ignore')),
])

from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler

numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median'))

])

from sklearn.compose import ColumnTransformer

preprocessor = ColumnTransformer(
    transformers=[
        ('cat', categorical_transformer, categorical_features),
        ('num', numeric_transformer, numeric_features)
    ]
)
```

Random forest regression

```
RFRegressor = Pipeline([
     ('preprocessor', preprocessor),
     ('RFRegressor', RandomForestRegressor())
])

RFRegressor.fit(X_train,Y_train)

predict= RFRegressor.predict(X_test)

r2_score(predict, Y_test)
```

Deploying model

```
!pip install watson-machine-learning-client

from watson_machine_learning_client import WatsonMachineLearningAPIClient

wml_credentials={
  "apikey": "ein0dLtA3GvhDOX6w0xbdM6A8niBiwsWcjvgP5nhlhCm",
  "instance_id": "bfcef6f2-d531-42d8-9977-4d790a2a145c",
  "url": "https://eu-gb.ml.cloud.ibm.com"
}

client = WatsonMachineLearningAPIClient( wml_credentials )

model_props = {client.repository.ModelMetaNames.AUTHOR_NAME:
"ShreyanshShukla",
               client.repository.ModelMetaNames.AUTHOR_EMAIL:
"shreyanshshuklashukla@gmail.com",
               client.repository.ModelMetaNames.NAME:
"Life_Expectancy_Prediction_ML_SmartInternz"}

model_artifact =client.repository.store_model(RFRegressor,
meta_props=model_props)

published_model_uid = client.repository.get_model_uid(model_artifact)
published_model_uid

deployment = client.deployments.create(published_model_uid,
name="Life_Expectancy_Prediction_ML_SmartInternz")
scoring_endpoint = client.deployments.get_scoring_url(deployment)
scoring_endpoint
```

3)Node Red Flow:


```
[
    {
        "id": "254b5c28.c52584",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "6adecb51.2758a4",
        "type": "ui_base",
        "theme": {
            "name": "theme-dark",
            "lightTheme": {
                "default": "#0094CE",
                "baseColor": "#0094CE",
                "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
                "edited": true,
                "reset": false
            },
            "darkTheme": {
                "default": "#097479",
                "baseColor": "#097479",
                "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
                "edited": true,
                "reset": false
            },
            "customTheme": {
                "name": "Untitled Theme 1",
                "default": "#4B7930",
                "baseColor": "#4B7930",
                "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
            },
            "themeState": {
                "base-color": {
                    "default": "#097479",
                    "value": "#097479",
                    "edited": false
                },
                "page-titlebar-backgroundColor": {
                    "value": "#097479",
                    "edited": false
                },
                "page-backgroundColor": {
                    "value": "#111111",
                    "edited": false
                },
                "page-sidebar-backgroundColor": {
                    "value": "#000000",
                    "edited": false
                },
                "group-textColor": {
                    "value": "#0eb8c0",
```

```
                    "edited": false
                },
                "group-borderColor": {
                    "value": "#555555",
                    "edited": false
                },
                "group-backgroundColor": {
                    "value": "#333333",
                    "edited": false
                },
                "widget-textColor": {
                    "value": "#eeeeee",
                    "edited": false
                },
                "widget-backgroundColor": {
                    "value": "#097479",
                    "edited": false
                },
                "widget-borderColor": {
                    "value": "#333333",
                    "edited": false
                },
                "base-font": {
                    "value": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
                }
            },
            "angularTheme": {
                "primary": "indigo",
                "accents": "blue",
                "warn": "red",
                "background": "grey"
            }
        },
        "site": {
            "name": "Node-RED Dashboard",
            "hideToolbar": "false",
            "allowSwipe": "true",
            "lockMenu": "true",
            "allowTempTheme": "true",
            "dateFormat": "DD/MM/YYYY",
            "sizes": {
                "sx": 48,
                "sy": 48,
                "gx": 6,
                "gy": 6,
                "cx": 6,
                "cy": 6,
                "px": 0,
                "py": 0
            }
        }
    },
    {
        "id": "66b76ac3.3b1104",
        "type": "ui_tab",
        "z": "",
        "name": "Home Page",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
```

```
    },
    {
        "id": "5fd975a1.c7c9cc",
        "type": "ui_group",
        "z": "",
        "name": "Life Expectancy Prediction",
        "tab": "a3d165b2.e31168",
        "order": 1,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "a3d165b2.e31168",
        "type": "ui_tab",
        "z": "",
        "name": "Home Page",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    },
    {
        "id": "310aed9a.d3fc52",
        "type": "ui_form",
        "z": "254b5c28.c52584",
        "name": "",
        "label": "",
        "group": "5fd975a1.c7c9cc",
        "order": 1,
        "width": 0,
        "height": 0,
        "options": [
            {
                "label": "Country",
                "value": "a",
                "type": "text",
                "required": true,
                "rows": null
            },
            {
                "label": "Year",
                "value": "b",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Status",
                "value": "c",
                "type": "text",
                "required": true,
                "rows": null
            },
            {
                "label": "BMI",
                "value": "d",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
```

```
            "label": "Adult_Mortality",
            "value": "e",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Infant_Deaths",
            "value": "f",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Alcohol",
            "value": "g",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Percentage_Expenditure",
            "value": "h",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Hepatitis_B",
            "value": "i",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Under_Five_Deaths",
            "value": "j",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Polio",
            "value": "k",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Total_Expenditure",
            "value": "l",
            "type": "number",
            "required": true,
            "rows": null
        },
        {
            "label": "Diphtheria",
            "value": "m",
            "type": "number",
            "required": true,
            "rows": null
```

```
            },
            {
                "label": "HIV/AIDS",
                "value": "n",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "GDP",
                "value": "o",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Population",
                "value": "p",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Thinness_10_19_years",
                "value": "q",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Thinness_5_9_years",
                "value": "r",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Income_Composition_of_Resources",
                "value": "s",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Schooling",
                "value": "t",
                "type": "number",
                "required": true,
                "rows": null
            },
            {
                "label": "Measles",
                "value": "u",
                "type": "number",
                "required": true,
                "rows": null
            }
        ],
        "formValue": {
            "a": "",
            "b": "",
```

```
            "c": "",
            "d": "",
            "e": "",
            "f": "",
            "g": "",
            "h": "",
            "i": "",
            "j": "",
            "k": "",
            "l": "",
            "m": "",
            "n": "",
            "o": "",
            "p": "",
            "q": "",
            "r": "",
            "s": "",
            "t": "",
            "u": ""
        },
        "payload": "",
        "submit": "Predict",
        "cancel": "Cancel",
        "topic": "",
        "x": 154.00001525878906,
        "y": 501.00000762939453,
        "wires": [
            [
                "ac3c3c7e.23be2"
            ]
        ]
    },
    {
        "id": "ac3c3c7e.23be2",
        "type": "function",
        "z": "254b5c28.c52584",
        "name": "pre token",
        "func": "//make user given values as global
variables\nglobal.set(\"a\",msg.payload.a);\nglobal.set(\"b\",msg.payload.b
);\nglobal.set(\"c\",msg.payload.c);\nglobal.set(\"d\",msg.payload.d);\nglo
bal.set(\"e\",msg.payload.e);\nglobal.set(\"f\",msg.payload.f);\nglobal.set
(\"g\",msg.payload.g);\nglobal.set(\"h\",msg.payload.h);\nglobal.set(\"i\",
msg.payload.i);\nglobal.set(\"j\",msg.payload.j);\nglobal.set(\"k\",msg.pay
load.k);\nglobal.set(\"l\",msg.payload.l);\nglobal.set(\"m\",msg.payload.m)
;\nglobal.set(\"n\",msg.payload.n);\nglobal.set(\"o\",msg.payload.o);\nglob
al.set(\"p\",msg.payload.p);\nglobal.set(\"q\",msg.payload.q);\nglobal.set(
\"r\",msg.payload.r);\nglobal.set(\"s\",msg.payload.s);\nglobal.set(\"t\",m
sg.payload.t);\nglobal.set(\"u\",msg.payload.u);\n\n//following are
required to receive a token\nvar
apikey=\"ein0dLtA3GvhDOX6w0xbdM6A8niBiwsWcjvgP5nhlhCm\";\nmsg.headers={\"co
ntent-type\":\"application/x-www-form-
urlencoded\"};\nmsg.payload={\"grant_type\":\"urn:ibm:params:oauth:grant-
type:apikey\",\"apikey\":apikey};\nreturn msg;\n",
        "outputs": 1,
        "noerr": 0,
        "x": 498.00000762939453,
        "y": 505.00000762939453,
        "wires": [
            [
                "6192a6bd.bcf478"
            ]
```

```
        ]
    },
    {
        "id": "48a0f3ab.e438fc",
        "type": "http request",
        "z": "254b5c28.c52584",
        "name": "",
        "method": "POST",
        "ret": "obj",
        "paytoqs": false,
        "url": "https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/bfcef6f2-
d531-42d8-9977-4d790a2a145c/deployments/3752ce82-a940-421e-8131-
89e2e4e6fed9/online",
        "tls": "",
        "persist": false,
        "proxy": "",
        "authType": "basic",
        "x": 254.00003051757812,
        "y": 87.00000762939453,
        "wires": [
            [
                "23140d8d.7b2452"
            ]
        ]
    },
    {
        "id": "23140d8d.7b2452",
        "type": "function",
        "z": "254b5c28.c52584",
        "name": "getFrom Endpoint",
        "func": "msg.payload=msg.payload.values[0][0];\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 538.0000076293945,
        "y": 86.00000762939453,
        "wires": [
            [
                "4eca0999.2c82e8"
            ]
        ]
    },
    {
        "id": "e3654838.2fabd8",
        "type": "function",
        "z": "254b5c28.c52584",
        "name": "sendTo Endpoint",
        "func": "//get token and make headers\nvar
token=msg.payload.access_token;\nvar instance_id=\"bfcef6f2-d531-42d8-9977-
4d790a2a145c\"\nmsg.headers={'Content-Type':
'application/json',\"Authorization\":\"Bearer \"+token,\"ML-Instance-
ID\":instance_id}\n\n//get variables that are set earlier\nvar a =
global.get(\"a\");\nvar b = global.get(\"b\");\nvar c =
global.get(\"c\");\nvar d = global.get(\"d\");\nvar e =
global.get(\"e\");\nvar f = global.get(\"f\");\nvar g =
global.get(\"g\");\nvar h = global.get(\"h\");\nvar i =
global.get(\"i\");\nvar j = global.get(\"j\");\nvar k =
global.get(\"k\");\nvar l = global.get(\"l\");\nvar m =
global.get(\"m\");\nvar n = global.get(\"n\");\nvar o =
global.get(\"o\");\nvar p = global.get(\"p\");\nvar q =
global.get(\"q\");\nvar r = global.get(\"r\");\nvar s =
global.get(\"s\");\nvar t = global.get(\"t\");\nvar u =
```

```
global.get(\"u\");\n\n//send the user values to service
endpoint\nmsg.payload = \n{\"fields\":[\"Country\", \"Year\", \"Status\",
\n\"BMI\", \"Adult_Mortality\", \"Infant_Deaths\", \"Alcohol\",
\"Percentage_Expenditure\", \"Hepatitis_B\", \"Under_Five_Deaths\",
\"Polio\", \"Total_Expenditure\", \"Diphtheria\", \"HIV/AIDS\",
\"GDP\",\"Population\", \"Thinness_10_19_years\", \"Thinness_5_9_years\",\n
\"Income_Composition_of_Resources\", \"Schooling\",
\"Measles\"],\n\"values\":[[a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u]]};\n
\nreturn msg;\n",
        "outputs": 1,
        "noerr": 0,
        "x": 536.0000076293945,
        "y": 303.0000057220459,
        "wires": [
            [
                "48a0f3ab.e438fc"
            ]
        ]
    },
    {
        "id": "6192a6bd.bcf478",
        "type": "http request",
        "z": "254b5c28.c52584",
        "name": "",
        "method": "POST",
        "ret": "obj",
        "paytoqs": false,
        "url": "https://iam.cloud.ibm.com/identity/token",
        "tls": "",
        "persist": false,
        "proxy": "",
        "authType": "basic",
        "x": 799.0000076293945,
        "y": 506.00000762939453,
        "wires": [
            [
                "e3654838.2fabd8"
            ]
        ]
    },
    {
        "id": "4eca0999.2c82e8",
        "type": "ui_text",
        "z": "254b5c28.c52584",
        "group": "5fd975a1.c7c9cc",
        "order": 2,
        "width": 0,
        "height": 0,
        "name": "",
        "label": "Prediction",
        "format": "{{msg.payload}}",
        "layout": "row-spread",
        "x": 814.0000610351562,
        "y": 87.00005626678467,
        "wires": []
    }
]
```