A

Project Report
On

Predicting Life Expectancy using Machine Learning

By

Rohit Tiwari

Email : rtiwari432@gmail.com

Web Page(User Interface) Link

https://node-red-mvldi.eu-gb.mybluemix.net/ui/#!/0?socketid=5vG-jW5aL_NYFmVJAAA0

1. INTRODUCTION

Life expectancy is the average number of years a person in a population could expect to live after age x. It is the life table parameter most commonly used to compare the survival experience of populations. The age most often selected to make comparisons is 0.0 (i.e., birth), although, for many substantive and policy analyses, other ages such as 65+ and 85+ are more relevant and may be used (e.g., for determining person-years of Medicare and Social Security benefit entitlement). To calculate life expectancy at age x (ex), age-specific mortality and population counts are needed to determine the age-specific mortality rates (i.e., the qx) and survival probabilities (lx) used in life table computations. Life expectancy is determined by multiplying the sequence of the probabilities of survival at each age to determine the proportion of a population expected to survive to age x. The number of persons expected to be alive in each single year of age category after age x is summed to determine the total number of years left to be lived after the index age (Lx). The total number of person-years to be lived after age x divided by the expected number of survivors to that age yields the life expectancy at age x.

1.1 Overview

In this project, we have to create a new model based on the data provided to evaluate the life expectancy.
The data offers a timeframe from 2015 to 2022. The output algorithms have been used to test if they can maintain their accuracy in predicting the life expectancy for data they haven't been trained. Following algorithms have been used:
Linear Regression
Ridge Regression
Lasso Regression
Elastic Net Regression
Linear Regression with Polynomic features
Decision Tree Regression
Random Forest Regression

World Health Organization (WHO) keeps track of the health status as well as many other related factors for all countries. The data-sets are made available to public for the purpose of health data analysis. The data-set related to life expectancy, health factors for 193 countries has been collected from the same WHO data repository website and its corresponding economic data was collected from United Nation website.

1.2 Purpose

The purpose of the project is to design a model for predicting Life Expectancy rate of a country given various features 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.

2. LITERATURE SURVEY

2.1 Existing Problem

To Predict the Life expectency of a country based on various factor such as GDP, BMI, HIV/AIDS, Year, Alcohol intake and etc.

2.2 Proposed solution

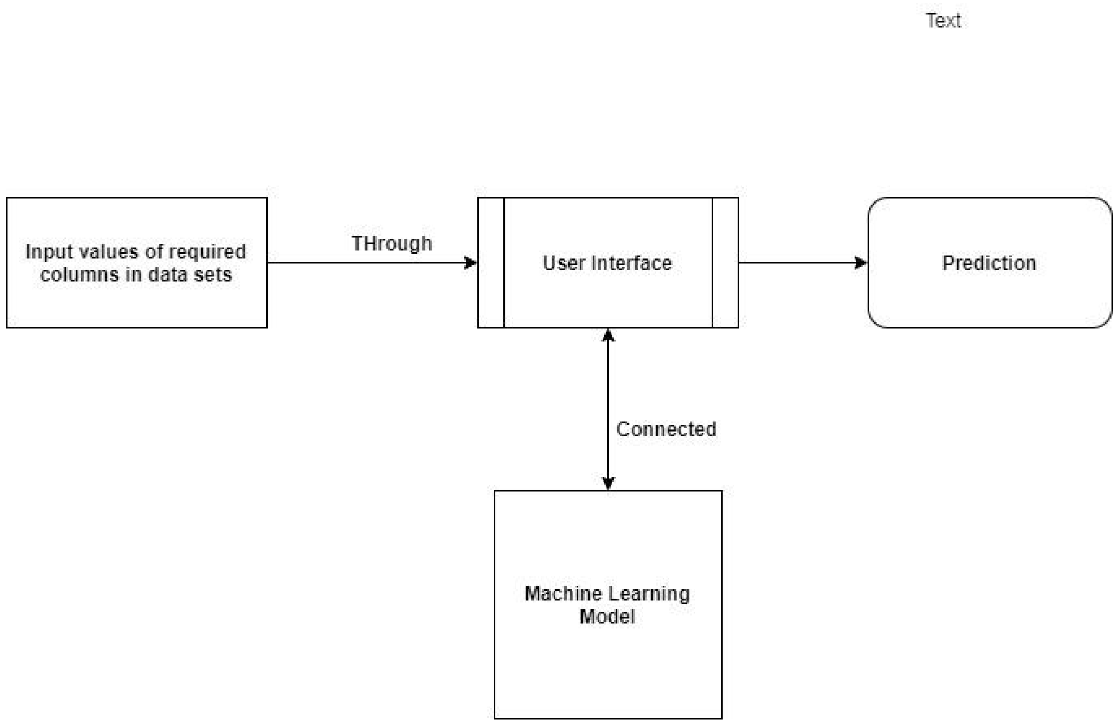
We are using Machine Learning Algorithm called Random forest Regressor to solve this problem.
We are using data set provided by WHO.

Steps

- Import Data set
- Fill the values of columns if they are empty.
- Convert all String values in to integer or float.
- Drop the unneccesary columns.
- Import and train the model.
- Cheack accuracy of model.
- Predict the life Expectency of the Country.

3. <u>THEORETICAL ANALYSIS</u>

3.1 Block Diagram



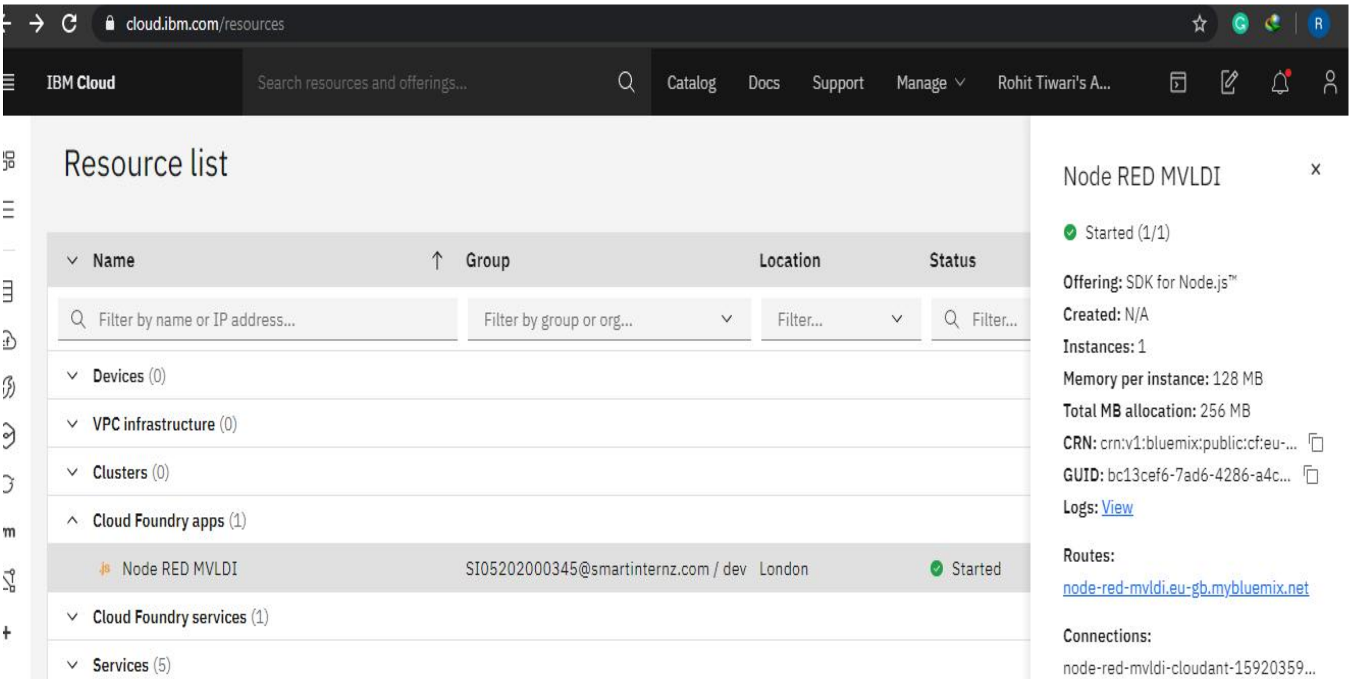
3.2 Hardware / Software designing

8 GB RAM

Python, IBM Cloud, IBM Watson,ML, WATSON Studio,Node-Red.

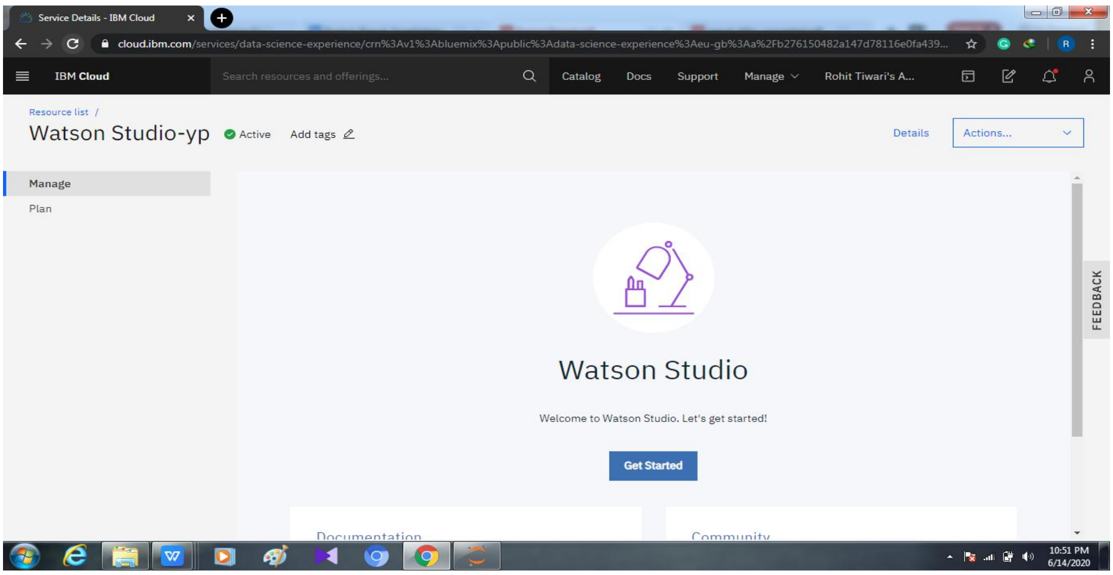
4. EXPERIMENTAL INVESTIGATIONS

A) IBM Cloud Resource List

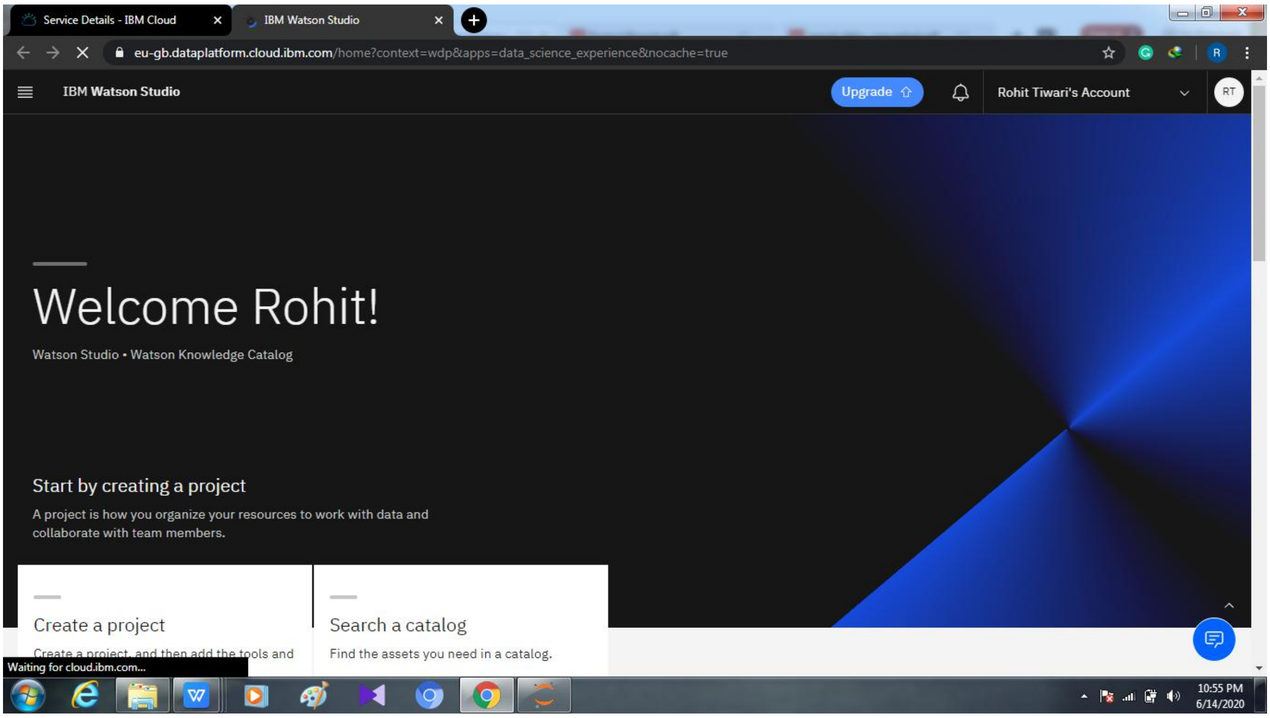


B) IBM Watson Studio

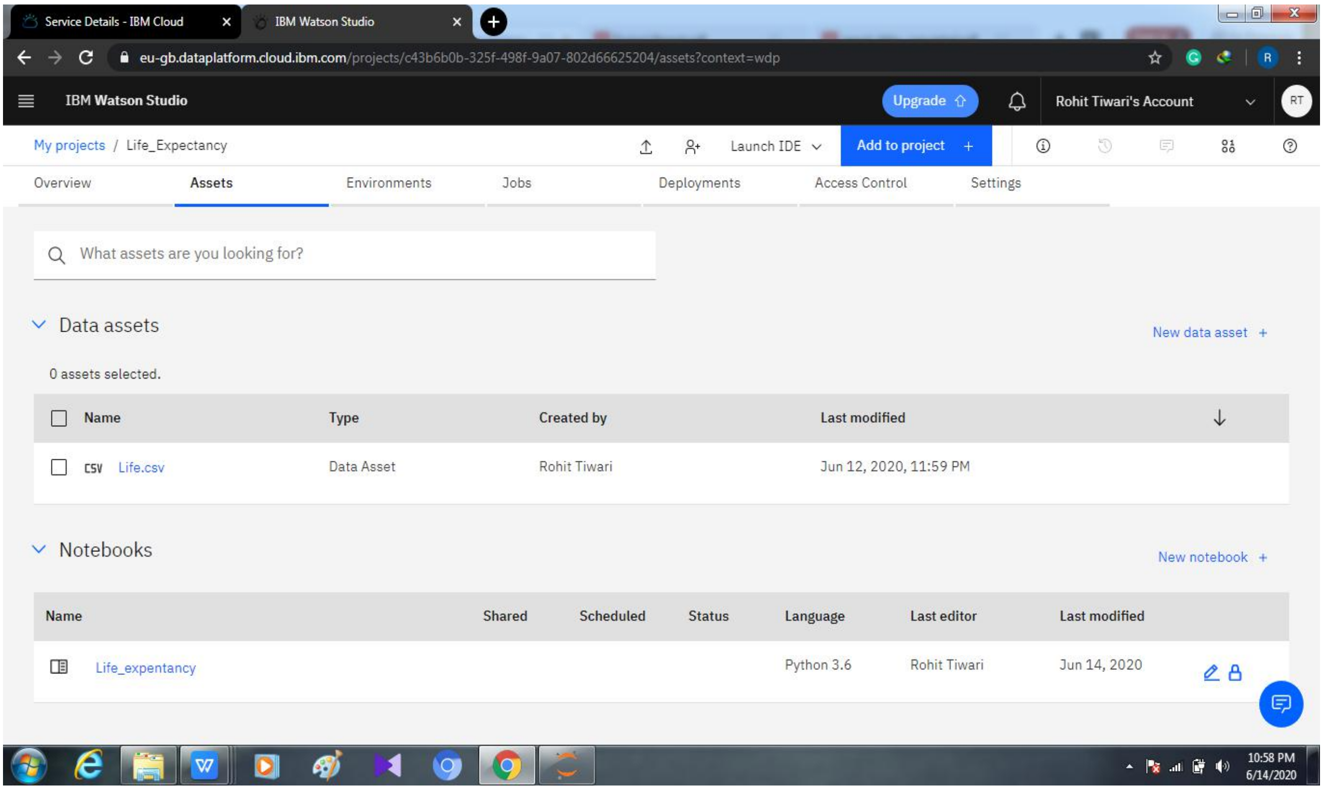
i)



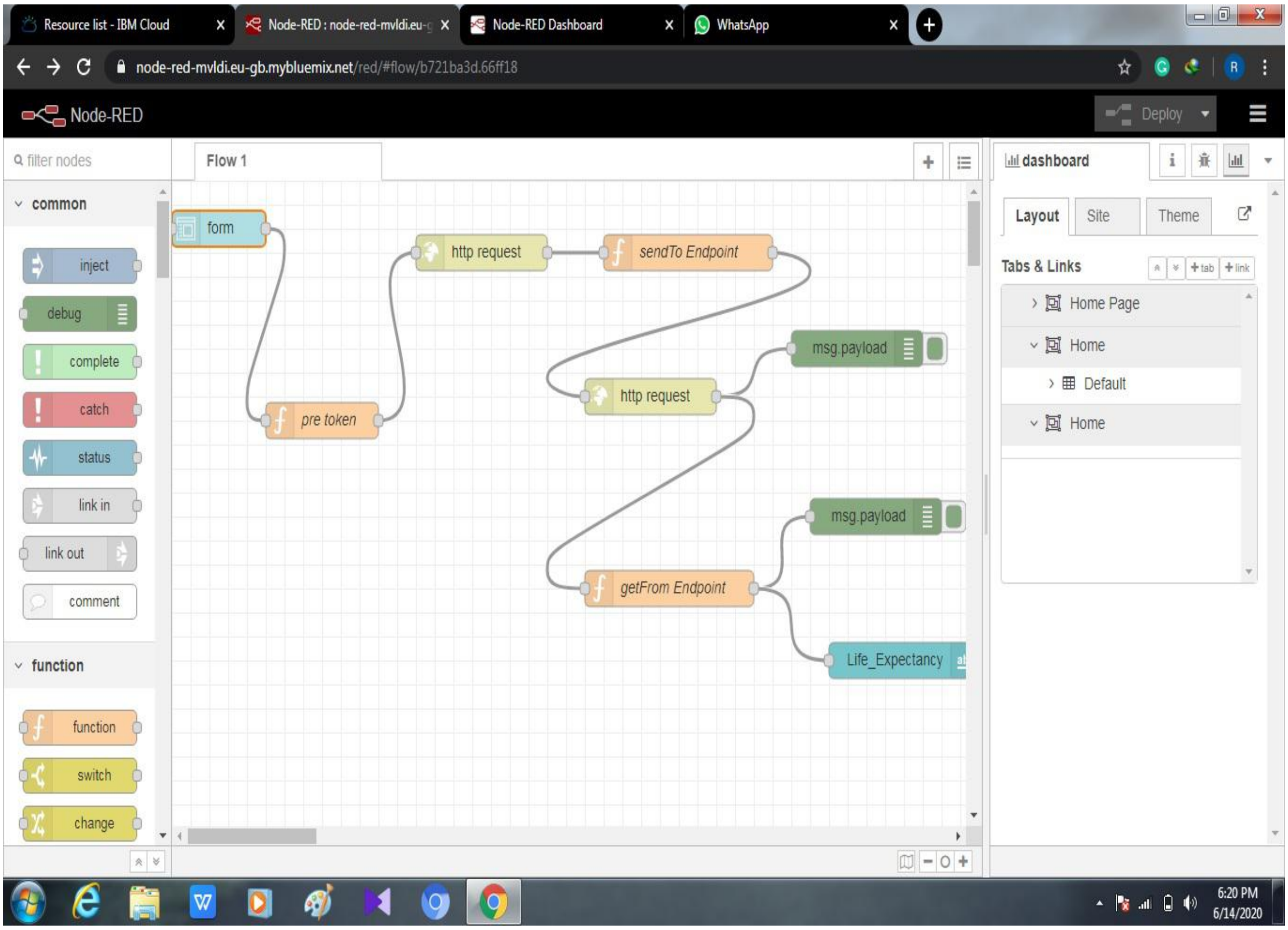
ii)



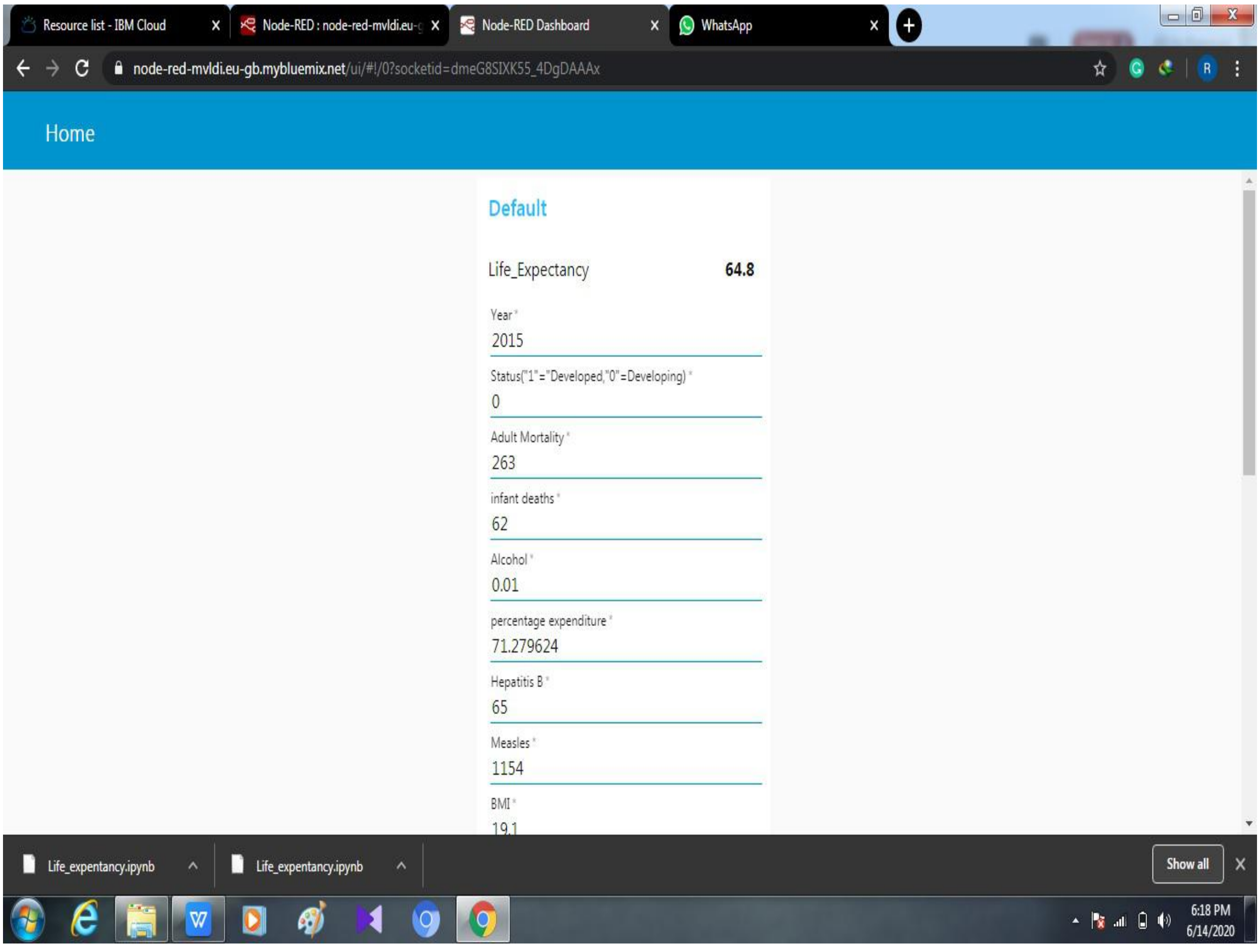
C) IBM Cloud Project Details



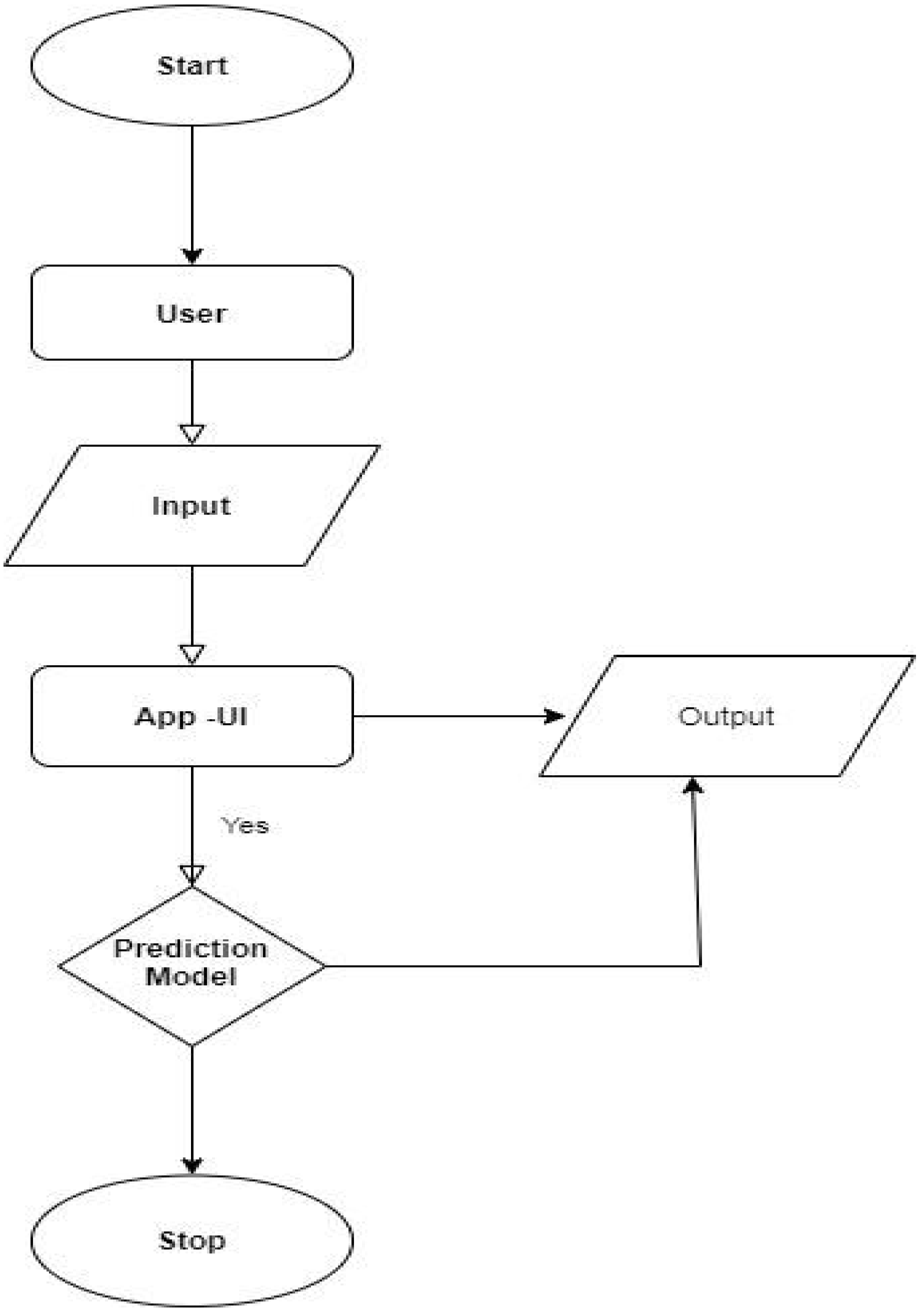
D) **Node-Red Flow**



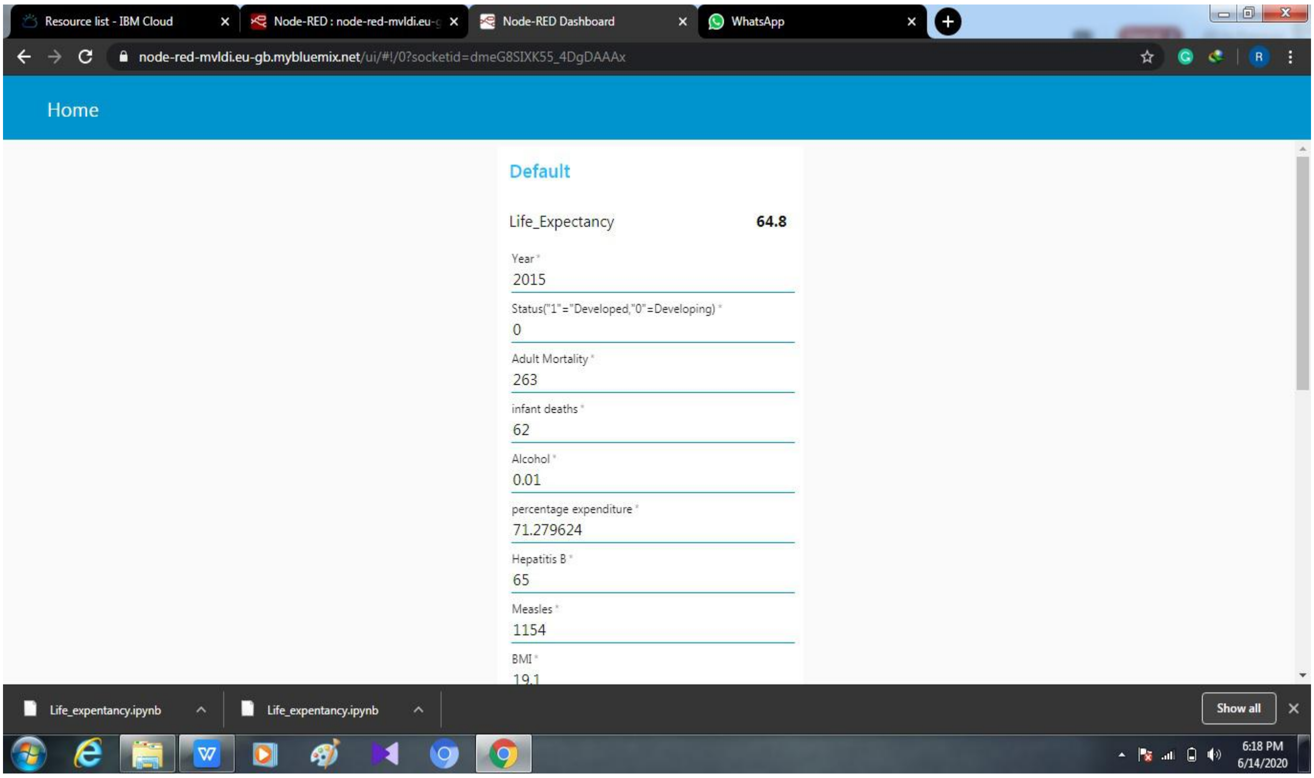
E) Life Expectancy Prediction UI

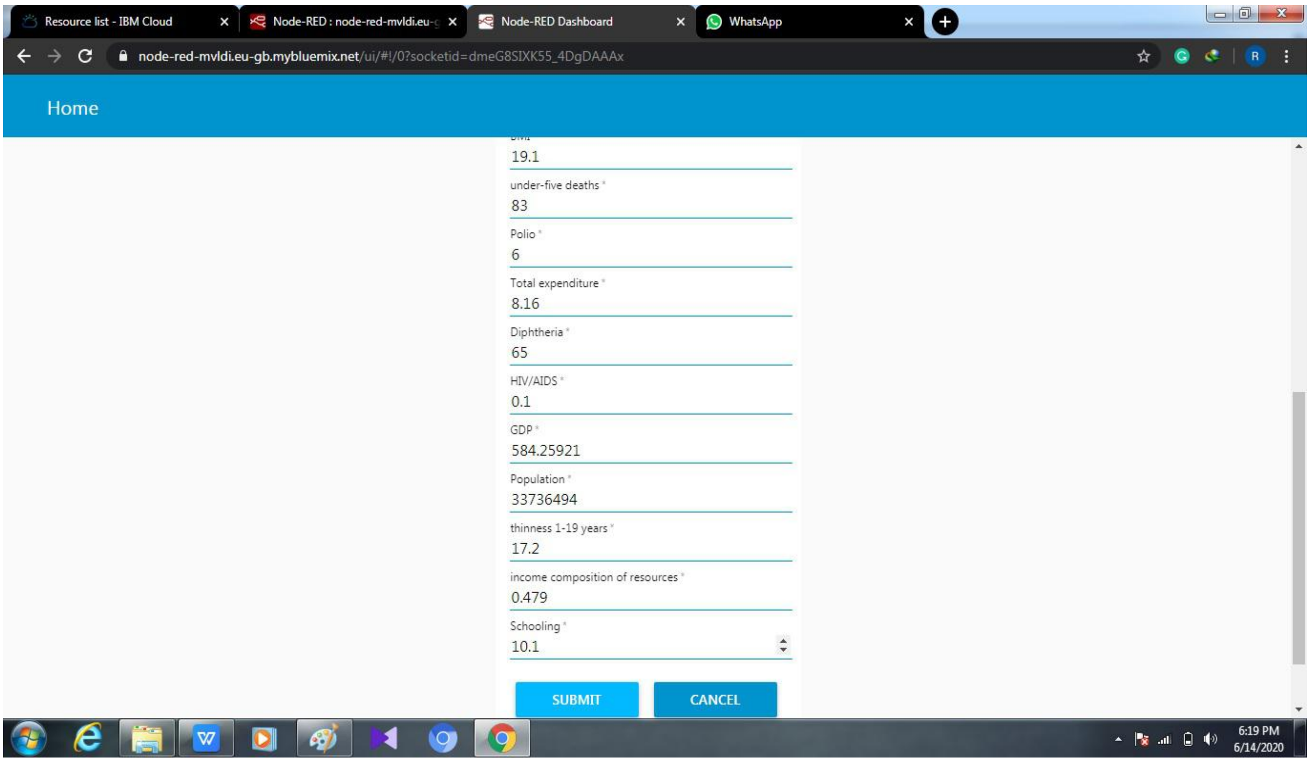


5. FLOWCHART



6. RESULT


Resource list - IBM Cloud
Node-RED : node-red-mvldi.eu-g
Node-RED Dashboard
WhatsApp
node-red-mvldi.eu-gb.mybluemix.net/ui/#!/0?socketid=dmeG8SIXK55_4DgDAAAx
Home
Default
Life_Expectancy 64.8
Year
2015
Status("1"="Developed,"0"=Developing)
0
Adult Mortality
263
infant deaths
62
Alcohol
0.01
percentage expenditure
71.279624
Hepatitis B
65
Measles
1154
BMI
Life_expentancy.ipynb
Life_expentancy.ipynb
Show all
6:18 PM
6/14/2020


Resource list - IBM Cloud
Node-RED : node-red-mvldi.eu-g
Node-RED Dashboard
WhatsApp
node-red-mvldi.eu-gb.mybluemix.net/ui/#!/0?socketid=dmeG8SIXK55_4DgDAAAx
Home
19.1
under-five deaths
83
Polio
6
Total expenditure
8.16
Diphtheria
65
HIV/AIDS
0.1
GDP
584.25921
Population
33736494
thinness 1-19 years
17.2
income composition of resources
0.479
Schooling
10.1
SUBMIT
CANCEL
6:19 PM
6/14/2020

7. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

a) Health Inequalities: Life expectancy has been used nationally to monitor health inequalities of a country.
b) Reduced Costs: This is a simple webpage and can be accessed by any citizen of a country to calculate life expectancy of their country and doesnot required any kind of payment neither for designing nor for using.
c) User Friendly Interface: 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.

DISADVANTAGES:

a) Wrong Prediction: As it depends completely on user, so if user provides some wrong values then it will predict wrong value.
b) Average Prediction: The model predicts average or approximate value with 97.07% accuracy but not accurate value.

8. APPLICATION

a) It can be used to monitor health inequalities of a country.
b) It can be used to develop statistics for country development process.
c) It can be used to analyse the factors for high life expectancy.
d) It is user friendly and can be used by anyone.

9. CONCLUSION

This user interface will be useful for the user to predict life expectancy value of their own country or any other country based on some required details such as GDP, BMI, Year, Alcohol Intake, Total expenditure and etc.

10. FUTURE SCOPE

Future Scope of the Model can be:

a) Feature Reduction

It requires much more data about 21 columns to be known prior for predicting life expectancy which can be again difficult for a normal user to gather such datas so I have decided to do some kind of feature reduction or replacement of some features as individuals or groups to make it more user friendly.

b) Attractive UI

It is a simple webpage only asking inputs and predict output. In future I have decided to make it more user friendly by providing some useful information about the country in the webpage itself so that user does not need to do any kind of prior research for the values.

c) Integrating with services such as speech recognition

11. BIBLIOGRAPHY

- https://cloud.ibm.com/docs/overview?topic=overview-whatis-platform
- https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application
- https://nodered.org
- https://www.youtube.com/embed/r7E1TJ1HtM0
- https://www.kaggle.com/kumarajarshi/life-expectancy-who
- https://bookdown.org/caoying4work/watsonstudio-workshop/jn.html

APPENDIX

A. Source code

Importing Neccesary packages

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
import missingno
```

Importing Data set

In [2]:
```
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 incl
udes your credentials.
# You might want to remove those credentials before you share the notebook.
client_c54edf641c5244a980cb66d9b7dabff7 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='gxTpRMtR5VbP9zPeeMLTaieydKR9gKUnNo5hd0xNCPQi',
    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_c54edf641c5244a980cb66d9b7dabff7.get_object(Bucket='lifeexpectan
cy-donotdelete-pr-thiqdsvdtqyg3y',Key='Life.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 )

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

Out[2]:

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0

5 rows × 22 columns

In [3]:
```
df.columns
```

Out[3]:
```
Index(['Country', 'Year', 'Status', 'Life expectancy ', 'Adult Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles ', ' BMI ', 'under-five deaths ', 'Polio', 'Total expenditur
e',
       'Diphtheria ', ' HIV/AIDS', 'GDP', 'Population',
       ' thinness  1-19 years', ' thinness 5-9 years',
       'Income composition of resources', 'Schooling'],
      dtype='object')
```

Renaming some column name

In [4]:
```
df.rename(columns = {'Life expectancy ': 'Life_expectancy',' thinness 5-9 year
s':'thinness_5-9_years'}, inplace=True)
df.columns
```

Out[4]:
```
Index(['Country', 'Year', 'Status', 'Life_expectancy', 'Adult Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles ', ' BMI ', 'under-five deaths ', 'Polio', 'Total expenditur
e',
       'Diphtheria ', ' HIV/AIDS', 'GDP', 'Population',
       ' thinness  1-19 years', 'thinness_5-9_years',
       'Income composition of resources', 'Schooling'],
      dtype='object')
```

In [5]:
```
df.head()
```

Out[5]:

	Country	Year	Status	Life_expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepat
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	6
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	6
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	6
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	6
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	6

5 rows × 22 columns

In [6]:
```
df.describe()
```

Out[6]:

	Year	Life_expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hep
count	2938.000000	2928.000000	2928.000000	2938.000000	2744.000000	2938.000000	2385
mean	2007.518720	69.224932	164.796448	30.303948	4.602861	738.251295	80
std	4.613841	9.523867	124.292079	117.926501	4.052413	1987.914858	25
min	2000.000000	36.300000	1.000000	0.000000	0.010000	0.000000	1
25%	2004.000000	63.100000	74.000000	0.000000	0.877500	4.685343	77
50%	2008.000000	72.100000	144.000000	3.000000	3.755000	64.912906	92
75%	2012.000000	75.700000	228.000000	22.000000	7.702500	441.534144	97
max	2015.000000	89.000000	723.000000	1800.000000	17.870000	19479.911610	99

checking data types of all columns

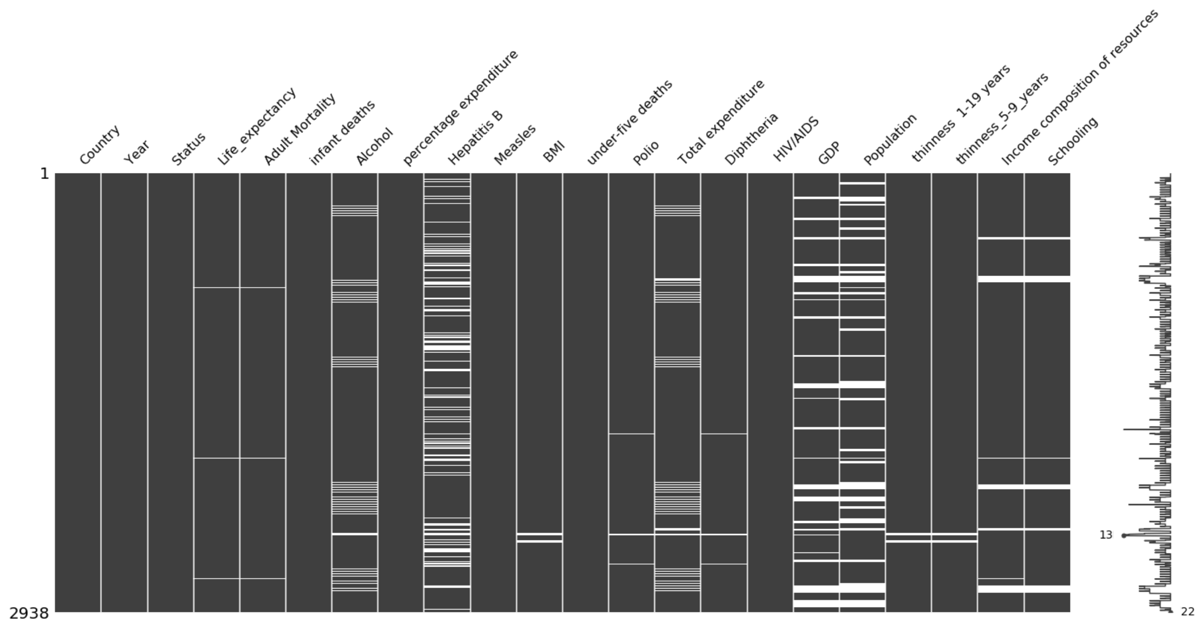
In [7]:
```
df.info()
```

```
<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
```

Showing empty values in columns

In [8]:
```
missingno.matrix(df)
```

Out[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fac72c1a128>
```



In [9]:
```
df.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
```

Filling the values in columns

In [10]:
```
for col in df.columns:
    df[col]=df[col].fillna(method="bfill")
df.shape
```

Out[10]:
```
(2938, 22)
```

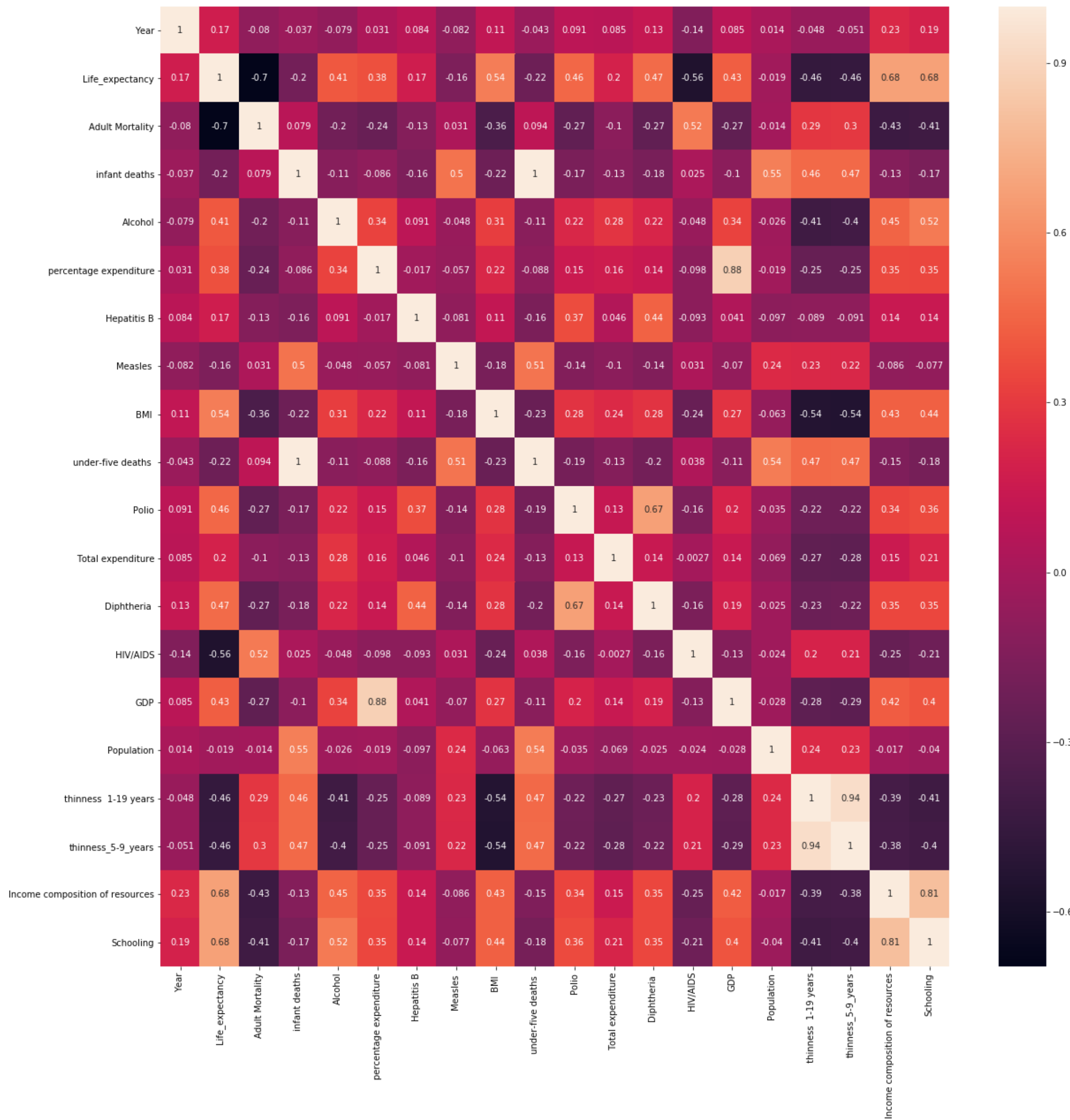
In [11]:
```
df.isnull().sum()
```

Out[11]:
```
Country                            0
Year                               0
Status                             0
Life_expectancy                    0
Adult Mortality                    0
infant deaths                      0
Alcohol                            0
percentage expenditure             0
Hepatitis B                        0
Measles                            0
 BMI                               0
under-five deaths                  0
Polio                              0
Total expenditure                  0
Diphtheria                         0
 HIV/AIDS                          0
GDP                                0
Population                         0
 thinness  1-19 years              0
thinness_5-9_years                 0
Income composition of resources    0
Schooling                          0
dtype: int64
```

Feature Selection

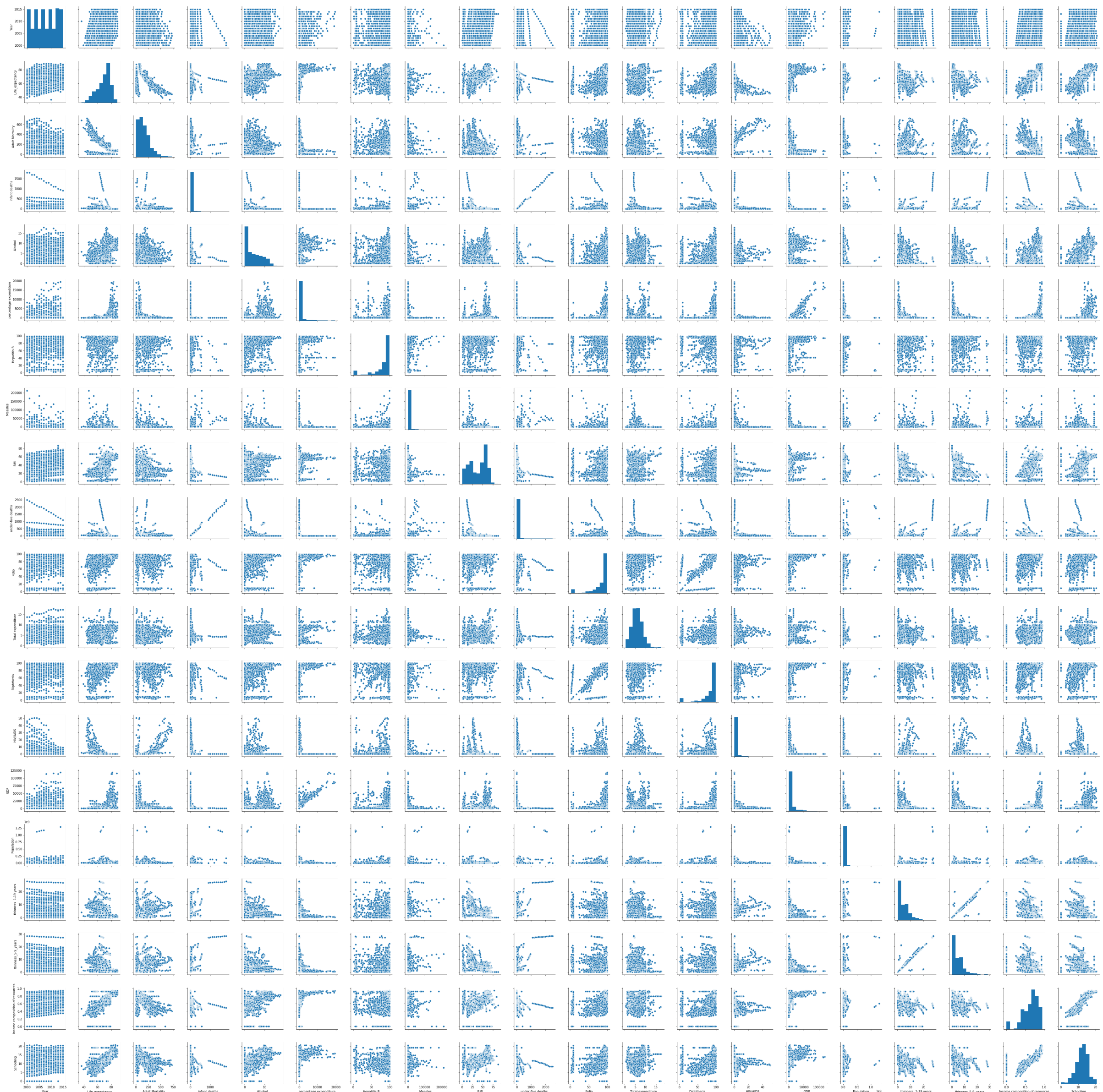
In [12]:
```
plt.figure(figsize=(20,20))
sn.heatmap(df.corr(),annot=True)
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7fac729fdeb8>



Visualization of all columns

In [13]:
```
sn.pairplot(df)
plt.show()
```



Droping some columns

In [14]:
```
df = df.drop(['Country','thinness_5-9_years'],axis = 1)
df.head()
```

Out[14]:

	Year	Status	Life_expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles
0	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154
1	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492
2	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430
3	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787
4	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013

Converting string values to integer(Binary-0,1)

In [16]:
```
df['Status'] = (df['Status'] == 'Developed').astype(int)
df['Status'].head()
```

Out[16]:
```
0    0
1    0
2    0
3    0
4    0
Name: Status, dtype: int64
```

In [18]:
```
x=df.drop(['Life_expectancy'],axis=1)
y=df.Life_expectancy
```

In [19]:
```
x.head()
```

Out[19]:

	Year	Status	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	BMI	under-five deaths	Polio
0	2015	0	263.0	62	0.01	71.279624	65.0	1154	19.1	83	6.0
1	2014	0	271.0	64	0.01	73.523582	62.0	492	18.6	86	58.0
2	2013	0	268.0	66	0.01	73.219243	64.0	430	18.1	89	62.0
3	2012	0	272.0	69	0.01	78.184215	67.0	2787	17.6	93	67.0
4	2011	0	275.0	71	0.01	7.097109	68.0	3013	17.2	97	68.0

Importing and Training of Model

In [20]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.4)
```

In [21]:
```
from sklearn.ensemble import RandomForestRegressor
model=RandomForestRegressor()
model.fit(x_train,y_train)
```

```
/opt/conda/envs/Python36/lib/python3.6/site-packages/sklearn/ensemble/forest.
py:246: FutureWarning: The default value of n_estimators will change from 10
in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[21]:
```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
           max_features='auto', max_leaf_nodes=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=10, n_jobs=None,
           oob_score=False, random_state=None, verbose=0, warm_start=False)
```

Testing for inputs

In [22]:
```
x_test.shape
```

Out[22]:
```
(1176, 19)
```

In [52]:
```
y_pred=model.predict(x_test)
print(y_pred)
```

```
[69.98 68.29 84.78 ... 77.18 71.21 71.98]
```

Accuracy of Model

In [24]:
```
model.score(x_train,y_train)
```

Out[24]:
```
0.990564037740675
```

In [25]:
```
model.score(x_test,y_test)
```

Out[25]:
```
0.9486506073424641
```

Predecting Result

In [53]:
```
Result = model.predict([[2015,0,263.0,62,0.01,71.279624,65.0,1154,19.1,83,6.0,
8.16,65.0,0.1,584.259210,33736494.0,17.2,0.479,10.1]])
print("Life Expectancy of this Country is : ",Result)
```

```
Life Expectancy of this Country is :  [64.8]
```

Code for Generating Scoring Endpoint Url

In [27]:
```
from watson_machine_learning_client import WatsonMachineLearningAPIClient
```

```
2020-06-14 09:52:45,012 - watson_machine_learning_client.metanames - WARNING
- 'AUTHOR_EMAIL' meta prop is deprecated. It will be ignored.
2020-06-14 09:52:48,683 - watson_machine_learning_client.wml_client_error - W
ARNING - Deployment creation failed. Error: 402. {"trace":"19fq370oimcz","err
ors":[{"code":"deployments_plan_limit_reached","message":"Current plan 'lite'
only allows 5 deployments"}]}
```

In [28]:
```
wml_credentials={
  "apikey": "oA2xpCtFiQPRpgoeYUGpr_fD6ck_tdl7a1w-vD7MNj9D",
  "instance_id": "ee299d7d-5851-45a4-bf9e-87063cce7323",
  "url": "https://eu-gb.ml.cloud.ibm.com",
  "username": "18b33122-f157-4bb2-bb23-7aca666aef8e"
}
```

In [29]:
```
client = WatsonMachineLearningAPIClient( wml_credentials )
```

In [41]:
```
model_props = {client.repository.ModelMetaNames.AUTHOR_NAME: "rohit",
               client.repository.ModelMetaNames.AUTHOR_EMAIL: "rtiwari432@gmai
l.com",
               client.repository.ModelMetaNames.NAME: "Life_Expectancy"}
```

In [42]:
```
model_artifact = client.repository.store_model(model,meta_props = model_props)
```

In [43]:
```
published_model_uid = client.repository.get_model_uid(model_artifact)
```

In [44]:
```
published_model_uid
```

Out[44]:
```
'e77eb785-a419-483a-b320-99cd17885abb'
```

In [45]:
```
deployment = client.deployments.create(published_model_uid,name = "Life_expectancy")
```

```
######################################################################################

Synchronous deployment creation for uid: 'e77eb785-a419-483a-b320-99cd17885abb' started

######################################################################################


INITIALIZING
DEPLOY_SUCCESS


---------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='5fc4b71a-21c4-4d22-ac07-82bd9d71178c'
---------------------------------------------------------------------------------------------
```

In [47]:
```
scoring_endpoint = client.deployments.get_scoring_url(deployment)
```

In [48]:
```
scoring_endpoint
```

Out[48]:
```
'https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/ee299d7d-5851-45a4-bf9e-87063cce7323/deployments/5fc4b71a-21c4-4d22-ac07-82bd9d71178c/online'
```

In [ ]: