SMARTBRIDGE

Let's Bridge the gap

Internship Report

Predicting Life Expectancy Using Machine Learning

Submitted By:-

Yash Bhatt
bhattyash031@gmail.com

Video Link

Webpage Link

https://node-red-vflwa.eu-gb.mybluemix.net/ui/#!/0?socketid=cldbzQDayv3jIr8CAAA3

Github Link

https://github.com/SmartPracticeschool/llSPS-INT-2147-Predicting-Life-Expectancy-using-Machine-Learning

Internship Title:- Predicting Life Expectancy using Machine Learning - SB39978

Category:- Machine Learning.

Table Of Contents

1. INTRODUCTION

1.1. Overview 2

1.2. Purpose 3

2. LITERATURE SURVEY

2.1 Existing Problem 4

2.2 Proposed solution 4

3. THEORITICAL ANALYSIS

2.3 Block Diagram 5

2.4 Hardware/Software design 5

4. EXPERIMENTAL INVESTIGATION 6

SCREENSHOTS 8

5. FLOWCHART 11

6. RESULT 12

7. ADVANTAGES & DISADVANTAGES 13

8. APPLICATION 15

9. CONCLUSION 15

10. FUTURE SCOPE 15

8. BIBILIOGRAPHY 16

APPENDIX

A. SOURCE CODE 17

1. Introduction

1.1 Overview

This Project consists of a machine learning model which predicts life expectancy based on various factors.This project is developed using IBM Machine Learning and deployed on IBM Cloud and is integrated with Node Red Application which provides Web Interface for the project.

Life expectancy is a statistical measure of the average time a human being is expected to live. Life expectancy 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.

This project tries to create a model based on data provided by the World Health Organization (WHO) to evaluate the life expectancy for different countries in years. The data offers a timeframe from 2000 to 2015. The data originates from here https://www.kaggle.com/kumarajarshi/life-expectancy-who/data.

This project 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.

- Project Requirements: IBM Cloud, IBM Watson Studio,Jupyter Notebook, Node-RED.
- Functional Requirements: IBM cloud, Machine Learning Libraries, Algorithms(Regression).
- Technical Requirements: WATSON Machine Learning, Python.
- Software Requirements: Python, Watson Studio, Node-Red
- Project Team: Yash Bhatt

1.2 Purpose and Scope

Life expectancy is the most significant aspect for decision making. Good projection for example helps to decide the course of treatment and helps to anticipate the procurement of health care services and facilities, or more broadly: facilitates Advance Care Planning. Advance Care Planning improves the quality of the final phase of life by stimulating doctors to explore the preferences for end-of-life care with their patients, and people close to the patients.

This project can be used in **hospitals** and the **doctors** can use it to predict the life expectancy of a patient with the underlying disease or a new born baby. It can be used by **government** to predict the life expactancy of the economic backward people due to poverty.With the help of this project it will be easy for **governments of the countries** with less life expectancies to improve their medical and healthcare services.

This Project will give overall prediction about the life expectancy of people living in various countries who have various diseases like Diptheria, HIV, Hepatites, Polio, Measles and also people taking alcohol based on Body Mass Index(BMI), GDP, Population, Mortality Rate of a particular country.

2. Literature Survey

SrNO.	Name	Author	Year	Key Findings
[1]	Forecasting life expectancy, years of life lost, and all-cause and cause-specific mortality for 250 causes of death: reference and alternative scenarios for 2016–40 for 195 countries and territories.	Kyle J Foreman, Neal Marquez, Andrew Dolgert.	October 16, 2018	Globally, most independent drivers of health were forecast to improve by 2040, but 36 were forecast to worsen. As shown by the better health scenarios, greater progress might be possible, yet for some drivers such as high body-mass index (BMI), their toll will rise in the absence of intervention.
[2]	Predicting Life Expectancy: A Cross-Country Empirical Analysis	Audrey Hendricks Philip E. Graves	September 23, 2009	Two different models were used to explain the variation in average life expectancy among the sample countries. Model one, the parsimonious model, is composed of the variables that proved to be of primary importance in predicting life expectancy: 1) LEi 0 1 2 3 4 5 6 7 i = b + b leduc + b lh2o + b ldrug + b lmeds + b lGDP + b aids + b aids + e.

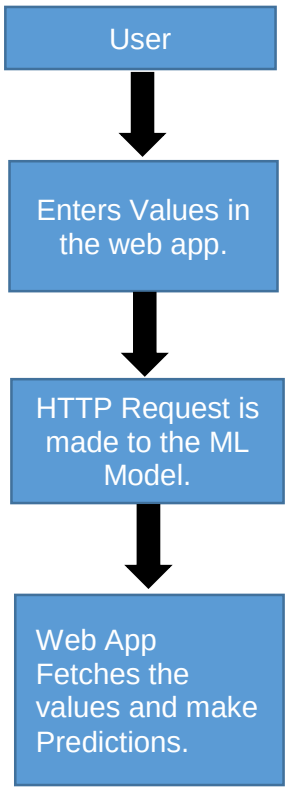
[3]	Predicting Human Lifespan Limits	**Byung Mook Weon**	August 2009	The life-extension strategies such as aggressive anti-aging therapies may allow more people to reach the limit of the natural human lifespan and thus the period of disease or senescence will be compressed against the natural barrier at the end of life, as expected [19,20]. The lifespan limit estimation may support current aging theories that presume the existence of the biological limit to human lifespan [21−23].
[4]	Healthy life expectancy, mortality, and age prevalence of morbidity	Timothy Riffe Alyson van Raalte Maarten Bijlsma	May 2017	The Sullivan method is the most commonly used method to partition life expectancy into estimates of the average life years lived in a state of good health (HLE) or disability (DLE). Its popularity owes to its minimal data requirements. Only current age-specific disability prevalence rates are needed in addition to a life table.

2.2 Proposed Solution

- For the above problem to get solved we have a dataset consist of various factors .In this system we have taken all the correlated features into consideration. So the target output variable i.e expected life span of the people depends upon variety of factors and not factors of particular fields.
- Important immunization like Hepatitis B, Polio and Diphtheria are also considered.
- The data-set related to life expectancy, health factors for 193 countries has been collected from WHO data repository website and its corresponding economic data was collected from the United Nations website. Among all categories of health-related factors only those critical factors were chosen which are more representative. It has been observed that in the past 15 years, there has been a huge development in health sector resulting in improvement of human mortality rates especially in the developing nations in comparison to the past 30 years. Therefore, in this project we have considered data from year 2000-2015 for 193 countries for further analysis. The individual data files have been merged together into a single data-set.
- The project uses immunization factors, mortality factors, economic factors, social factors and other health related factors to predict life expectancy of a country for a given year using a machine learning model.
- Since the observations in this dataset are based on different countries, it will be easier for a country to determine the predicting factor which is contributing to lower value of life expectancy. This will help in suggesting a country, which area should be given importance in order to efficiently improve the life expectancy of its population.

3. Theoritical Analysis

3.1 Block Diagram



3.2 Hardware / Software Designing

- Create necessary IBM Cloud services
- Create Watson studio project
- Configure Watson Studio
- Create IBM Machine Learning instance
- Create machine learning model in Jupyter notebook
- Deploy the machine learning model
- Create flow and configure node
- Integrate node red with machine learning model
- Deploy and run Node Red app.

Input is taken from the user using a “Form” element in Node-Red. Then, an HTTP request is made to the IBM cloud that further makes an HTTP request to the deployed model using model’s instance id. After verification of id, the model sends an HTTP response which is finally parsed by the Node-Red application and the result is displayed on the user screen.

4. Experimental Investigations

Following factors are taken into account for predicting the life expectancy of a country.

1. Country
2. Status: Developed or Developing status of the country.
3. Year
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: Hepatitis B =immunization coverage among 1-year-olds (%).
9. Measles: 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: Polio (Pol3) 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.

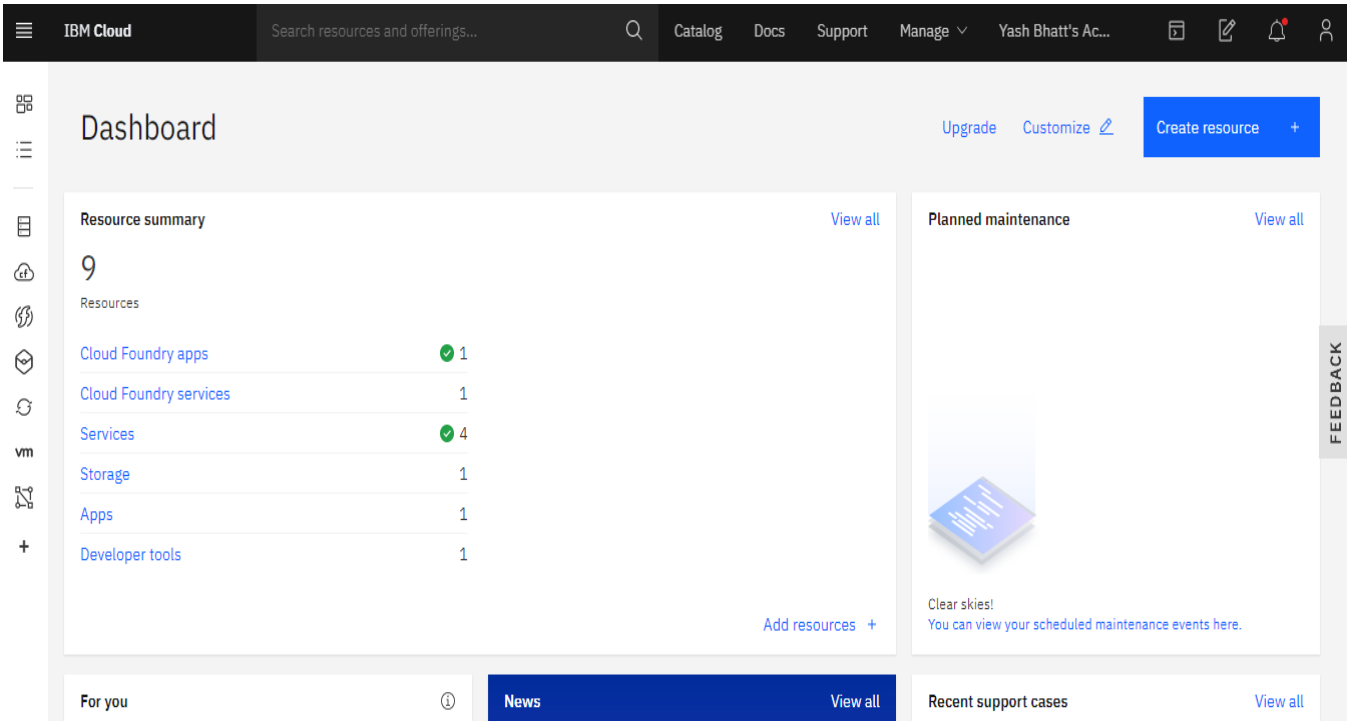
Finding The Best Algorithm:-

Random Forest Regression Provides Best Accuracy Of – 95.25%

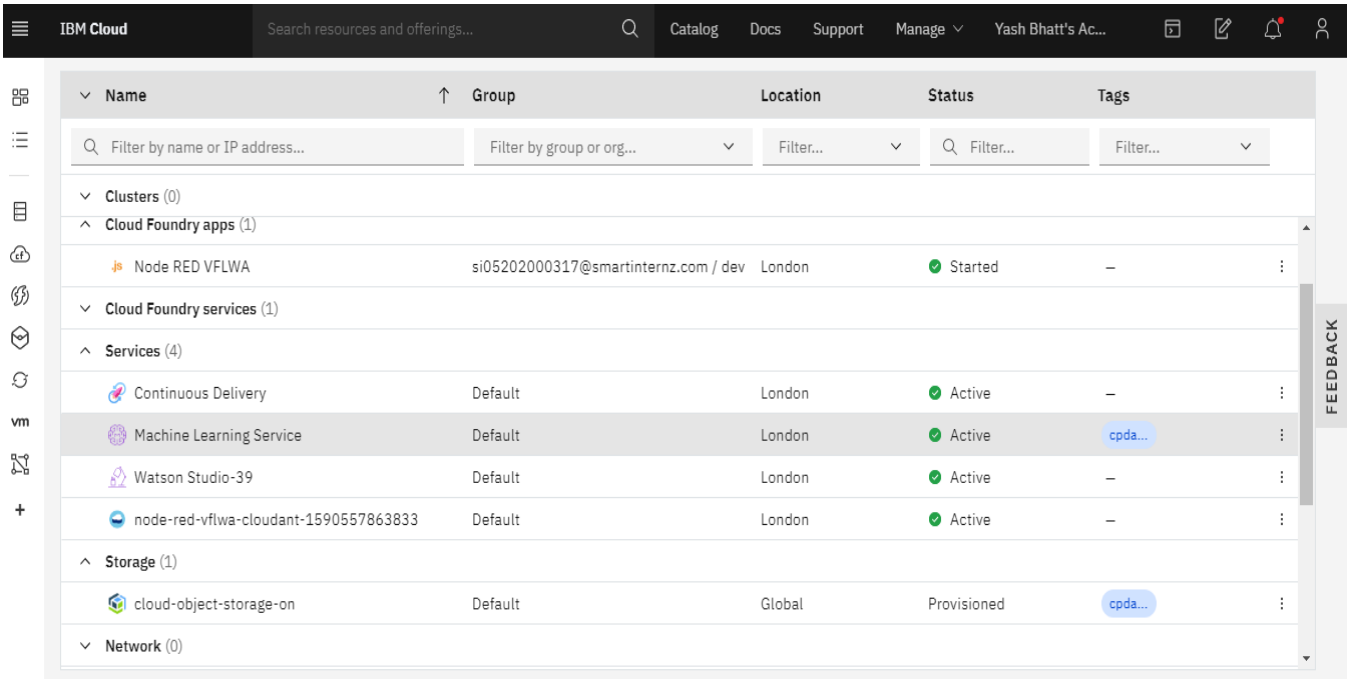
```
OrderedDict([('Decision Tree Regressor', 0.9189785430104337),
             ('Linear Regression', 0.8188082000368992),
             ('Random Forest Regressor', 0.9525555281025279)])
```

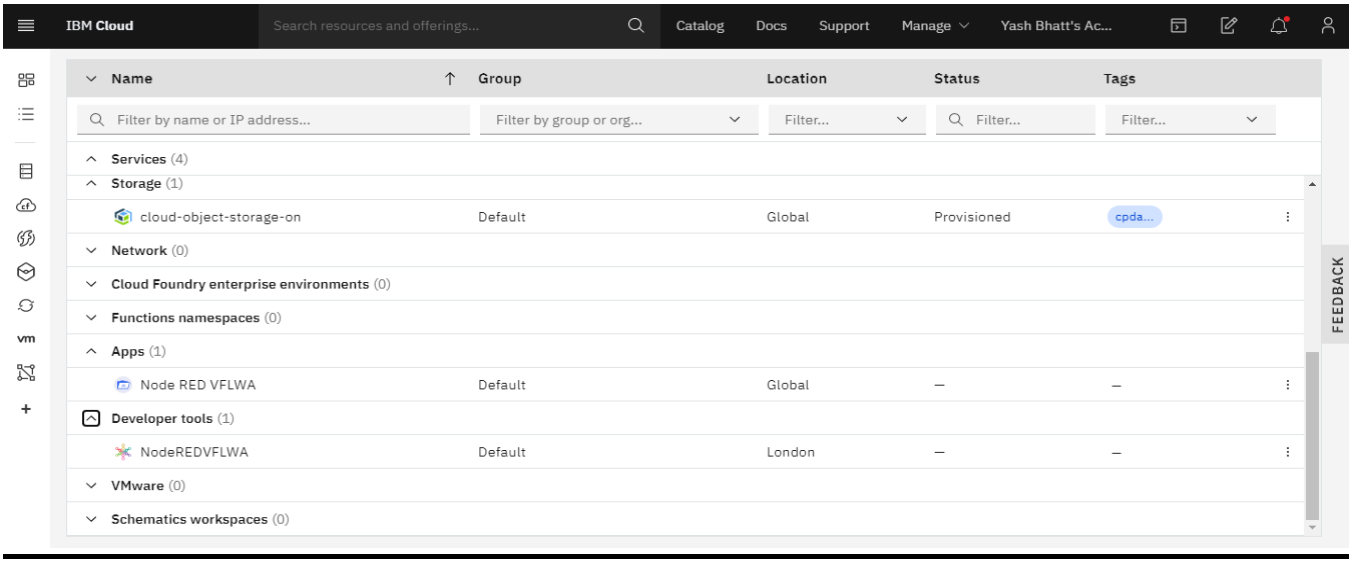
SCREENSHOTS

IBM CLOUD DASHBOARD:

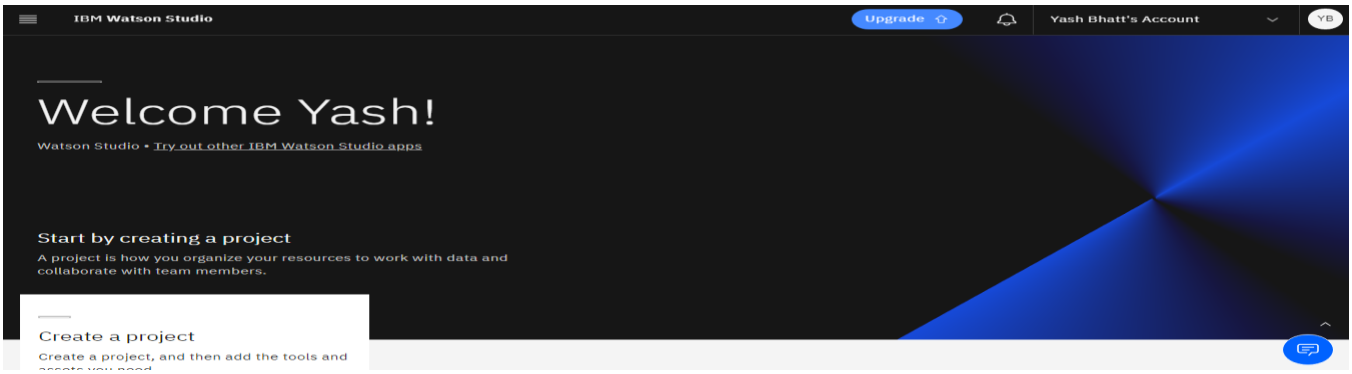


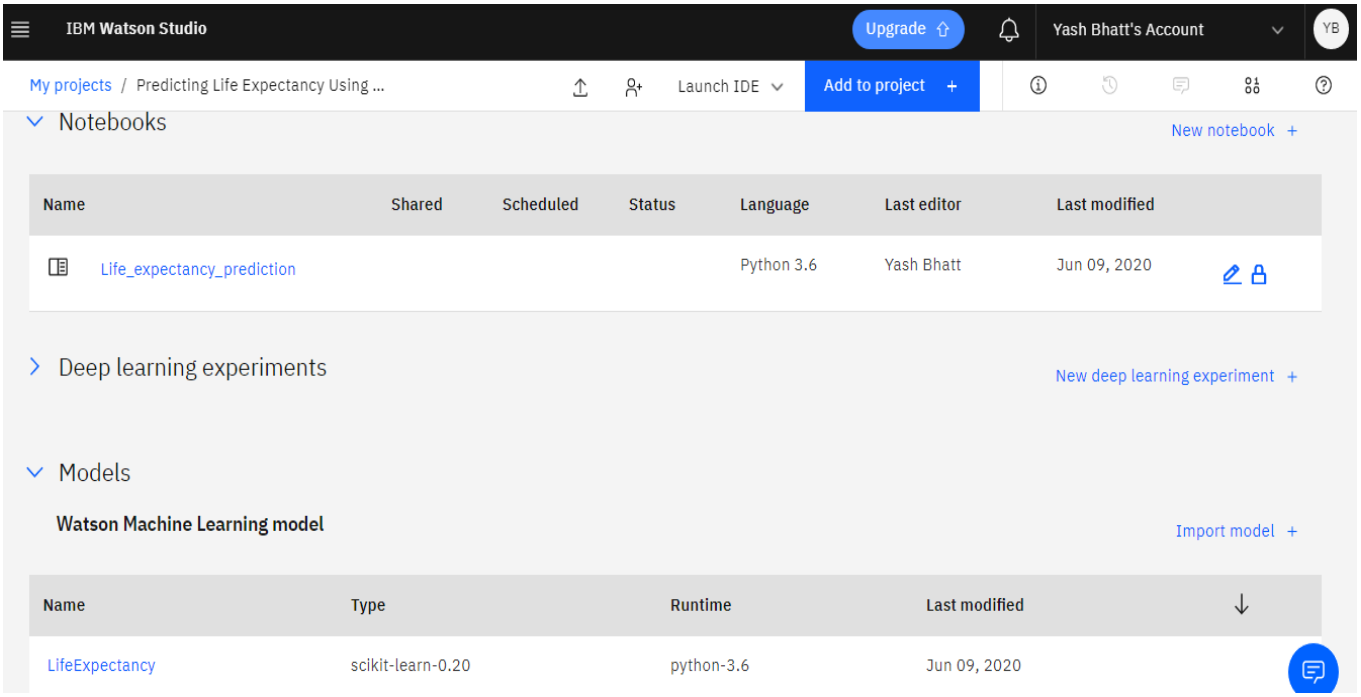
Resource List:



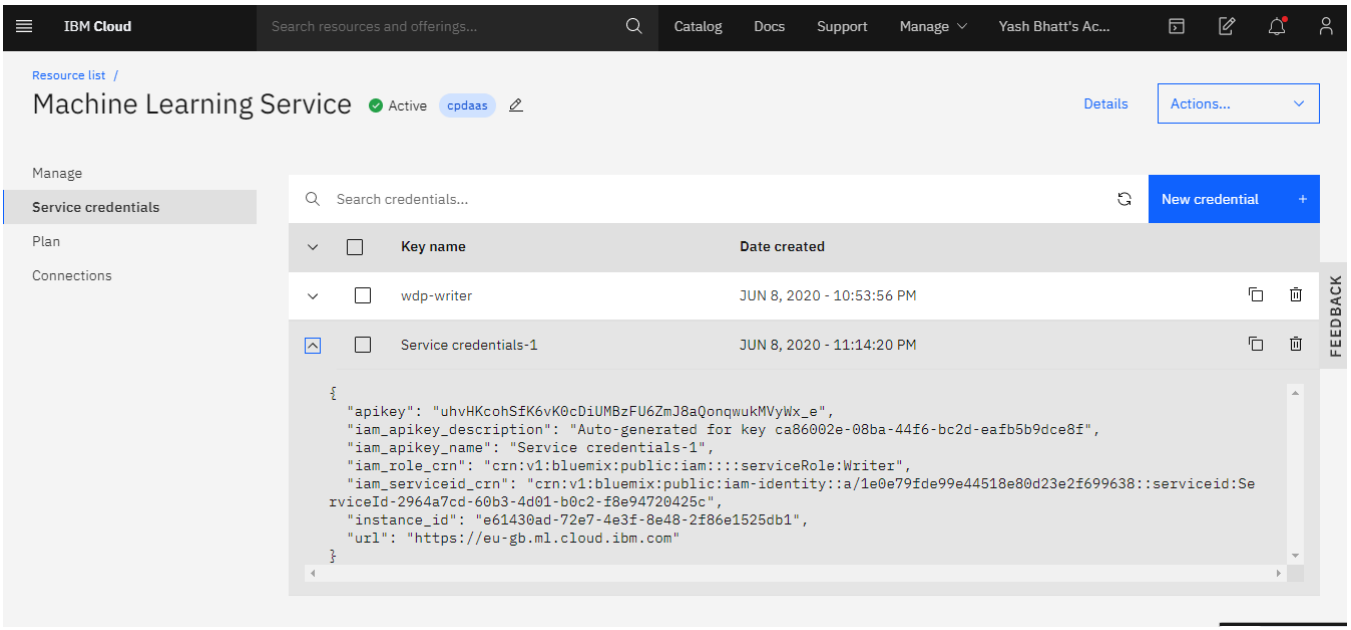


WATSON STUDIO :

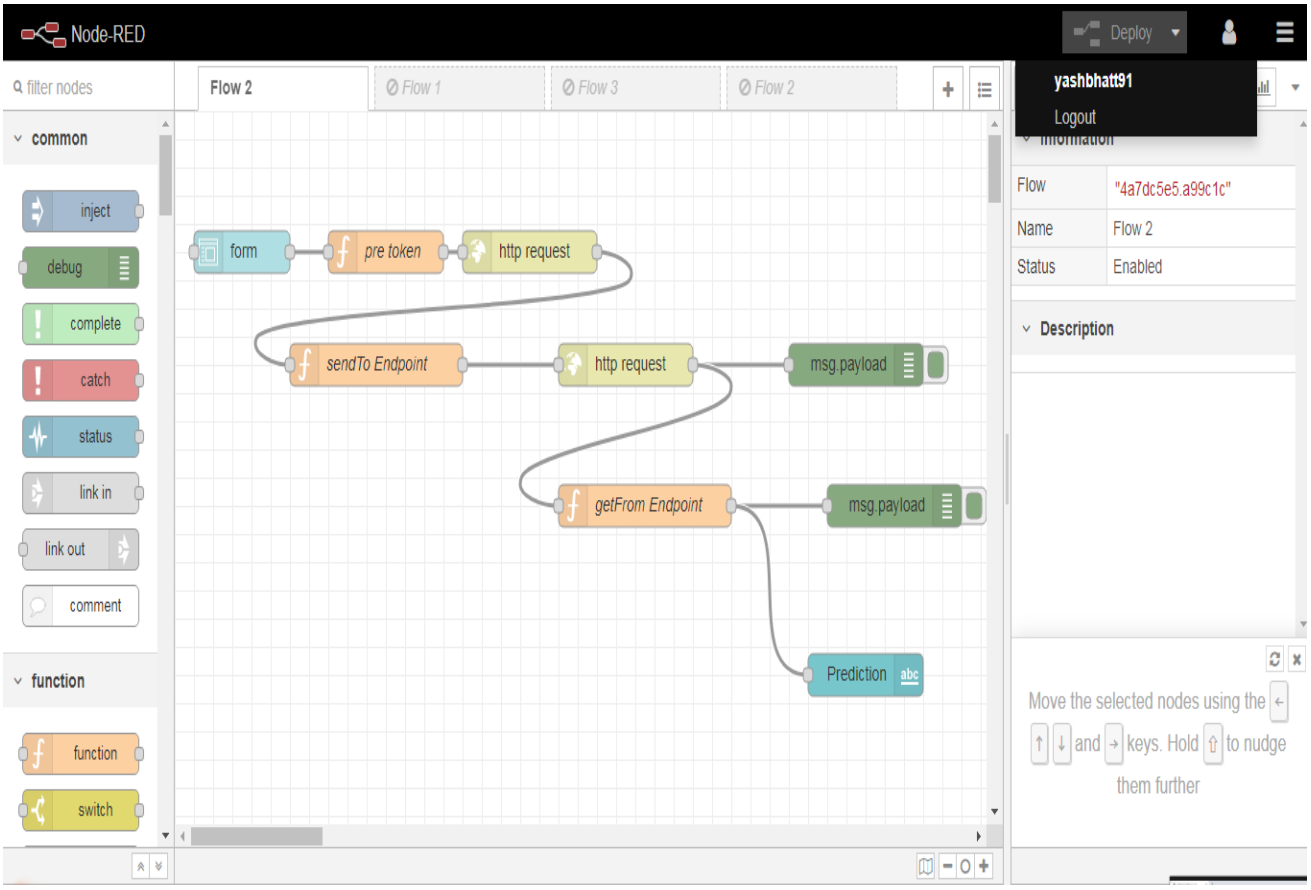




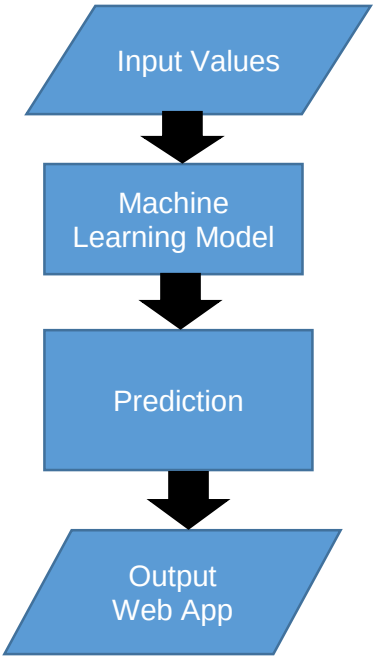
WATSON MACHINE LEARNING SERVICE:



NODE RED FLOW:

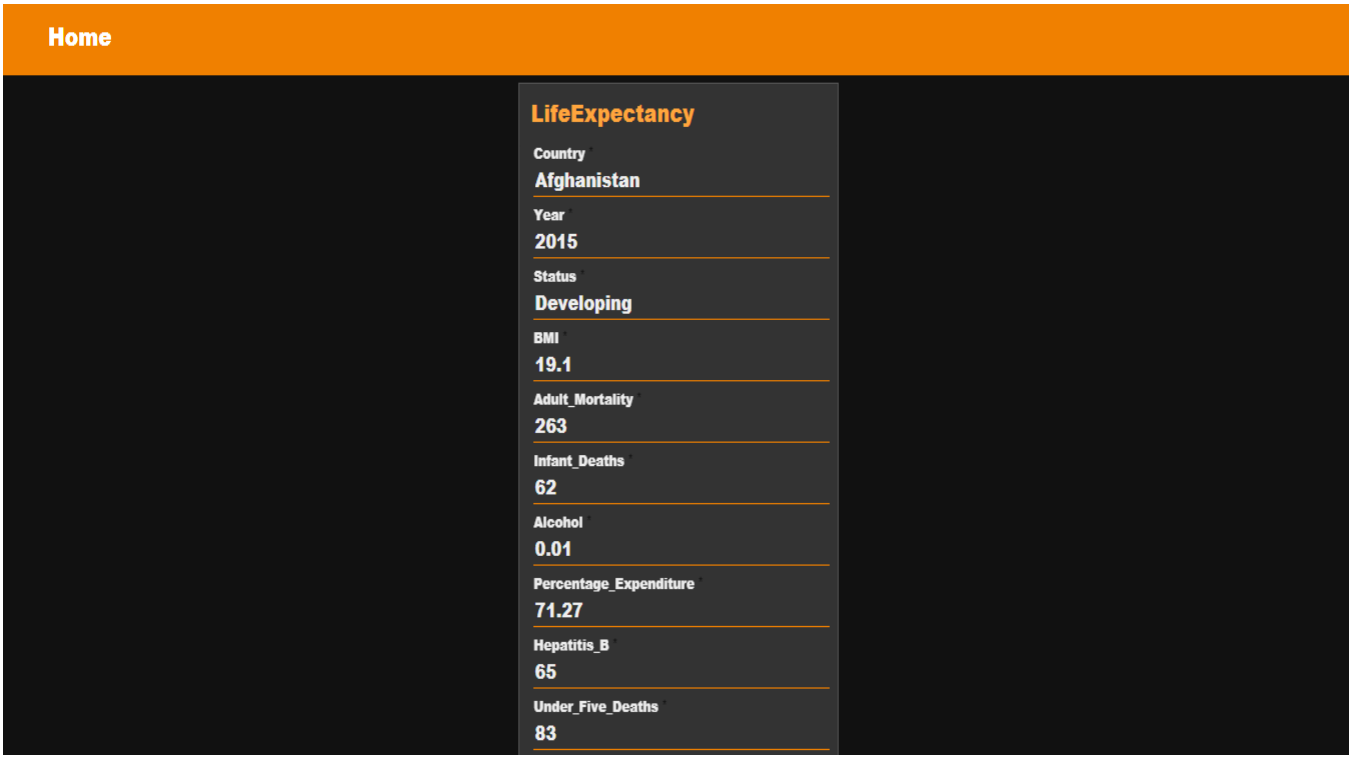


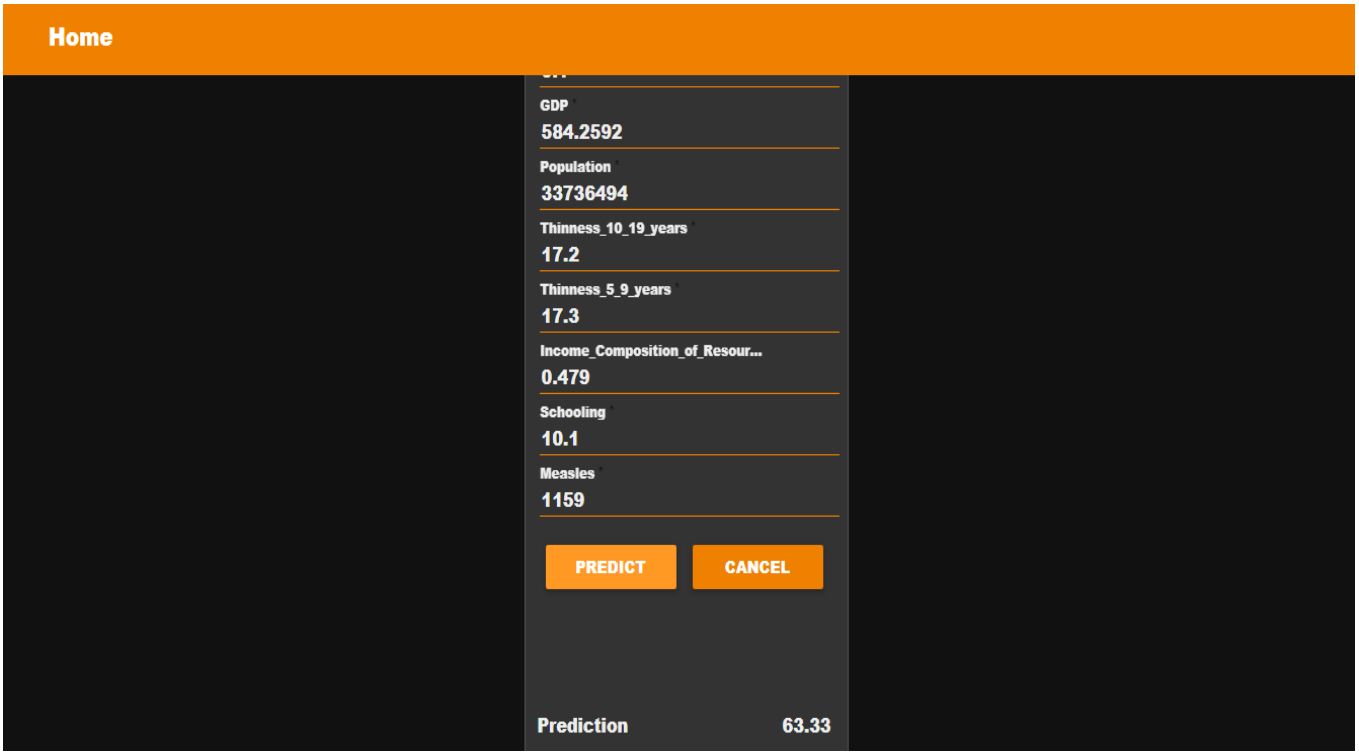
5. FLOWCHART


Input Values
Machine Learning Model
Prediction
Output Web App

6. Results

https://node-red-vflwa.eu-gb.mybluemix.net/ui/#!/0?socketid=5fvkCu_GSCrbHMbUAAAH





7. Advantages And Disadvantages

Advantages:-

- Can be used by any organization to predict.
- The observations in the dataset used are based on different countries, it will be easier for a country to determine the predicting factor which is contributing to lower value of life expectancy. This will help in suggesting a country, which area should be given importance in order to efficiently improve the life expectancy of its population.
- Some of the past research was done considering multiple linear regression based on data set of one year for all the countries. But the dataset used for training the model contained data of past 15 years to give a fairly better prediction.
- The application is easy and simple to use.
- The machine learning algorithm used in the project is Random Forest regression which is based on the bagging algorithm and uses Ensemble Learning technique. It creates as many trees on the subset of the data and combines the output of all the trees. In this way it reduces over fitting problem in decision trees and also reduces the variance and therefore improves the accuracy.

Disadvantages:-

- Can be only used by the people having the knowledge of data analysis.
- As the model is deployed on cloud, so one requires good internet connection to use the application.
- The Node-Red application needs to make HTTP request to IBM cloud and then another HTTP request to the model before providing the prediction. That makes the app a bit slow.

8. Applications

- This will help in suggesting a country which area should be given importance in order to efficiently improve the life expectancy of its population.
- It will be easier for a country to determine the predicting factor which is contributing to lower value of life expectancy and can be used in various organization to improve the quality of service.
- The project can be used as a basis to develop personalized health applications.
- The governments can plan and develop their health infrastructures by keeping the most correlated factors in mind.
- The project can help governments to keep track of their country's health status so they can plan for the future accordingly.

9. Conclusion

By doing this project I have successfully created Life expectancy prediction system using IBM Watson studio, Watson machine learning and Node-RED service. Gained knowledge about IBM Cloud, IBM Watson with its services, Node-Red and how to integrate Node-Red with Machine Learning Model. The project makes a good use of machine learning in predicting life expectancy of a country that can help respective government in making policies that will serve for the benefit of the nation and entire humankind.

Future Scope

- Increase the dataset size with continuing UN and Global Data to incorporate new added features like population, GDP, environmental, and etc in order to test and clarify country groupings.
- Mental Health versus Life Expectancy.
- Currently, the project is just a web application. It can be developed to support other platforms like Android, IOS and Windows Mobile.
- Other regression models can also be used for prediction and later the best among them should be chosen.

11 Bibliography

[1] https://cloud.ibm.com/

[2] https://www.ibm.com/cloud/get-started

[3] https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/

[4] https://www.ibm.com/watson/products-services

[5] https://www.kaggle.com/kumarajarshi/life-expectancy-who

[6] https://www.researchgate.net/publication/45868855_Predicting_Human_Lifespan_Limits

[7] https://www.thelancet.com/action/showPdf?pii=S0140-6736%2818%2931694-5

[8] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1477594

APPENDIX

A. Source Code

Life_expectancy_prediction .ipynb

```
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
```

```
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 includ
es your credentials.
# You might want to remove those credentials before you share the notebook.
client_f99f16128b5444479f4682d87bd72ed6 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='avxbV9iz7TSzUjGLhf1kdiVZDhNniWWWsxG0XBb0-lot',
    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_f99f16128b5444479f4682d87bd72ed6.get_object(Bucket='predictinglife
expectancyusingmach-donotdelete-pr-vbeb1hzb5mrjux',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__, bo
dy )

# If you are reading an Excel file into a pandas DataFrame, replace `read_csv` b
y `read_excel` in the next statement.
data = pd.read_csv(body)
data.head()

# If you are reading an Excel file into a pandas DataFrame, replace `read_csv` b
y `read_excel` in the next statement.
```

```
data.head()
```

```
data.shape #(2938, 22)
```

Out[38]:

```
(2938, 22)
```

In [39]:

```
data.describe()
```

```
data.info()
```

```
data.isnull().sum()
```

Handling Missing value

In [42]:

```
country_list = data.Country.unique()
len(country_list)
```

Out[42]:

```
193
```

In [43]:

```
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()

In [44]:

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

In [45]:

```
data.shape  #(1987, 22) size reduced
```

Out[45]:

```
(1987, 22)
```

In [46]:

```
data.isna().sum()
```

Corelation matrix

In [47]:

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

Renaming the columns as it contains trailing spaces

In [48]:

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

Removing outliers

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

In [49]:

```
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 expendi
ture' :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

In [50]:

```
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

In [51]:

```
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(variabl
e,
                                                                             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))
```

```
Number of outliers and percentage of it in Life expectancy : 4 and 0.201308505284348
27
Number of outliers and percentage of it in Adult Mortality : 58 and 2.91897332662304
98
Number of outliers and percentage of it in Alcohol : 3 and 0.1509813789632612
Number of outliers and percentage of it in percentage expenditure : 232 and 11.67589
3306492199
Number of outliers and percentage of it in Hepatitis B : 216 and 10.870659285354806
Number of outliers and percentage of it in Measles : 361 and 18.16809260191243
Number of outliers and percentage of it in BMI : 0 and 0.0
Number of outliers and percentage of it in under-five deaths : 227 and 11.4242576748
86764
Number of outliers and percentage of it in Polio : 159 and 8.002013085052843
Number of outliers and percentage of it in Total expenditure : 13 and 0.654252642174
1318
Number of outliers and percentage of it in Diphtheria : 195 and 9.813789632611979
Number of outliers and percentage of it in HIV/AIDS : 309 and 15.551082033215904
Number of outliers and percentage of it in GDP : 244 and 12.279818822345245
Number of outliers and percentage of it in Population : 260 and 13.085052843482638
Number of outliers and percentage of it in thinness 10-19 years : 70 and 3.522898842
4760947
Number of outliers and percentage of it in thinness 5-9 years : 75 and 3.77453447408
153
Number of outliers and percentage of it in Income composition of resources : 91 and
4.579768495218923
Number of outliers and percentage of it in Schooling : 32 and 1.6104680422747861
```

```
Number of outliers and percentage of it in infant deaths : 198 and 9.96477101157524
```
18 columns having outliers

as BMI has no outliers

In [52]:

```
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 resource
s'],(0.05,0))
winsorized_Schooling = winsorize(data['Schooling'],(0.02,0.01))
```

In [53]:

```
winsorized_list = [winsorized_Life_Expectancy,winsorized_Adult_Mortality,winsorized_
Alcohol,winsorized_Measles,winsorized_Infant_Deaths,
            winsorized_Percentage_Exp,winsorized_HepatitisB,winsorized_Under_Five_De
aths,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((va
riable > max_val) | (variable < min_val))[0])))
```

```
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
```

```
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
Number of outliers after winsorization in  : 0
```

Adding 18 new columns having no outliers to the dataframe

In [54]:

```
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
```

In [55]:

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

Out[55]:

```
(1987, 40)
```

EDA

In [69]:

```
data.columns
```

Out[69]:

```
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 10-19 years',
```

```
       'thinness 5-9 years', 'Income composition of resources', 'Schooling',
       'winsorized_Life_Expectancy', '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'],
      dtype='object')
```

In [70]:

```
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','thinn
ess 5-9 years','Schooling',
                   'percentage expenditure','Total expenditure','GDP','Population','
Income composition of resources']]
```

In [74]:

```
disease_cols.corr()
```

```
sns.pairplot(disease_cols,diag_kind='kde')
```

Hence all the features are significant to predict the target variable

In [76]:

```
col = ['Life expectancy','winsorized_Life_Expectancy','Adult Mortality','winsorized_
Adult_Mortality','infant deaths',
        'winsorized_Infant_Deaths','Alcohol','winsorized_Alcohol','percentage expen
diture','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_thi
nness_10_19_years','thinness 5-9 years',
        'winsorized_thinness_5_9_years','Income composition of resources','winsoriz
ed_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') #include specifies the list of datatype to be incluyded
.here is Object
```

Out[77]:

	Country	Status
count	1987	1987
unique	133	2
top	Albania	Developing
freq	16	1702

In [78]:

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

```
country_data = data.groupby('Country')['winsorized_Life_Expectancy'].mean().sort
_values(ascending=True)
country_data.plot(kind='bar' ,figsize=(50,15),fontsize=30,color='g')
plt.title("Life_Expectancy w.r.t Country",fontsize=30)
plt.xlabel("Country",fontsize=30)
plt.ylabel("Avg Life_Expectancy")
plt.show()
```

```
plt.figure(figsize=(7,5))
```

```
plt.bar(data.groupby('Year')['Year'].count().index,data.groupby('Year')['winsorized_Life_Expectancy'].mean())
plt.xlabel("Year",fontsize=12)
plt.ylabel("Avg Life_Expectancy",fontsize=12)
plt.show()
```

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

```
winsorized_Life_Expectancy             1.000000
Life expectancy                        0.999564
winsorized_Income_Comp_Of_Resources    0.823222
winsorized_Schooling                   0.762333
Schooling                              0.746461
Income composition of resources        0.728270
BMI                                    0.601434
winsorized_Percentage_Exp              0.557117
winsorized_GDP                         0.551722
winsorized_Diphtheria                  0.535687
winsorized_Polio                       0.516167
GDP                                    0.445022
Diphtheria                             0.443489
Polio                                  0.415868
percentage expenditure                 0.415074
winsorized_Alcohol                     0.396073
Alcohol                                0.394380
winsorized_HepatitisB                  0.268086
Hepatitis B                            0.250284
winsorized_Tot_Exp                     0.208400
Total expenditure                      0.203675
Year                                   0.170919
Population                            -0.011106
winsorized_Population                 -0.040138
Measles                               -0.138959
infant deaths                         -0.161837
under-five deaths                     -0.188370
winsorized_Measles                    -0.313013
thinness 5-9 years                    -0.452230
thinness 10-19 years                  -0.460880
winsorized_thinness_5_9_years         -0.498791
winsorized_thinness_10_19_years       -0.507870
winsorized_Infant_Deaths              -0.516494
winsorized_Under_Five_Deaths          -0.552035
HIV/AIDS                              -0.577316
winsorized_Adult_Mortality            -0.654255
Adult Mortality                       -0.659030
winsorized_HIV                        -0.793696
```

```
Name: winsorized_Life_Expectancy, dtype: float64
```

In [82]:

```
import seaborn as sns
from pandas.plotting import scatter_matrix
attributes= ['winsorized_Life_Expectancy','winsorized_Income_Comp_Of_Resources','winsorized_Schooling'
,'winsorized_Diphtheria','winsorized_Polio','winsorized_Adult_Mortality','winsorized_Alcohol','winsorized_Measles','winsorized_Infant_Deaths',
            'winsorized_Percentage_Exp','winsorized_HepatitisB','winsorized_Under_Five_Deaths','winsorized_Tot_Exp',
            'winsorized_HIV','winsorized_GDP','winsorized_Population','winsorized_thinness_10_19_years','winsorized_thinness_5_9_years']
cormat=data[attributes].corr()
plt.figure(figsize=(15,15))
sns.heatmap(cormat, square=True, annot=True, linewidths=.5)
plt.show()
```

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

Out[83]:

	winsorized_Life_Expectancy
Status	
Developed	78.83
Developing	66.19

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

In [84]:

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

Out[84]:

```
Ttest_indResult(statistic=23.02232052151534, pvalue=3.793119218299665e-104)
```

In [85]:

```
data.columns
```

Out[85]:

```
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 10-19 years',
```

```
       'thinness 5-9 years', 'Income composition of resources', 'Schooling',
       'winsorized_Life_Expectancy', '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'],
      dtype='object')
```

Now our data has no null values and no outliers

Creating a new dataframe with refined data

In [86]:

```
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'])
```

In [87]:

```
new_data.shape
```

Out[87]:

```
(1987, 22)
```

In [88]:

```
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)
```

In [90]:

```
new_data.head()
```

```
new_data.columns
```

Out[91]:

```
Index(['Country', 'Year', 'Status', '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',
       'Life_Expectancy'],
      dtype='object')
```

Separating the input features and label

In [92]:

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

In [93]:

```
X.head()
```

```
Y.head()
```

Out[94]:

	Life_Expectancy
0	65.0
1	59.9
2	59.9
3	59.5

	Life_Expectancy
4	59.2

Spilitting the data into train set and test set

In [95]:

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

In [96]:

```
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']
```

In [97]:

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

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

In [98]:

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

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

])
```

In [99]:

```
from sklearn.compose import ColumnTransformer

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

In [100]:

```
models = OrderedDict([
    ( "Linear Regression",       Pipeline([
                                          ('preprocessor', preprocessor),
                                          ('LRegressor', LinearRegression())])  ),
```

```
    ( "Decision Tree Regressor", Pipeline([
                                          ('preprocessor', preprocessor),
                                           ('DTRegressor', DecisionTreeRegressor())
])  ),
    ( "Random Forest Regressor", Pipeline([
                                          ('preprocessor', preprocessor),
                                           ('RFRegressor', RandomForestRegressor())
])  ),

])
```

In [101]:

```
scores = {}
for (name, model) in models.items():
  model.fit(X_train,Y_train)
  scores[name] =r2_score(model.predict(X_test), Y_test)

scores = OrderedDict(sorted(scores.items()))
scores
```

```
/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)
/opt/conda/envs/Python36/lib/python3.6/site-packages/sklearn/pipeline.py:267: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expected. Please c
hange the shape of y to (n_samples,), for example using ravel().
  self._final_estimator.fit(Xt, y, **fit_params)
```

Out[101]:

```
OrderedDict([('Decision Tree Regressor', 0.9189785430104337),
             ('Linear Regression', 0.8188082000368992),
             ('Random Forest Regressor', 0.9525555281025279)])
```

Hence Random forest regression is the most suitable algorithm for this dataset

Random forest regression

In [102]:

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

In [103]:

```
RFRegressor.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)
/opt/conda/envs/Python36/lib/python3.6/site-packages/sklearn/pipeline.py:267: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expected. Please c
hange the shape of y to (n_samples,), for example using ravel().
```

```
  self._final_estimator.fit(Xt, y, **fit_params)
```

Out[103]:

```
Pipeline(memory=None,
     steps=[('preprocessor', ColumnTransformer(n_jobs=None, remainder='drop', sparse
_threshold=0.3,
         transformer_weights=None,
         transformers=[('cat', Pipeline(memory=None,
     steps=[('onehot', OneHotEncoder(categorical_features=None, categories=None,
       dtype=<class 'numpy.float64'>...ators=10, n_jobs=None,
           oob_score=False, random_state=None, verbose=0, warm_start=False))])
```

In [104]:

```
predict= RFRegressor.predict(X_test)
```

In [105]:

```
r2_score(predict, Y_test)
```

Out[105]:

```
0.9515647138222864
```

In [106]:

```
!pip install watson-machine-learning-client
Requirement already satisfied: watson-machine-learning-client in /opt/conda/envs/Pyt
hon36/lib/python3.6/site-packages (1.0.376)
Requirement already satisfied: tqdm in /opt/conda/envs/Python36/lib/python3.6/site-p
ackages (from watson-machine-learning-client) (4.31.1)
Requirement already satisfied: certifi in /opt/conda/envs/Python36/lib/python3.6/sit
e-packages (from watson-machine-learning-client) (2020.4.5.1)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python36/lib/python3.6
/site-packages (from watson-machine-learning-client) (2.4.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python36/lib/python3.6/si
te-packages (from watson-machine-learning-client) (0.8.2)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python36/lib/python3.6/sit
e-packages (from watson-machine-learning-client) (1.24.1)
Requirement already satisfied: lomond in /opt/conda/envs/Python36/lib/python3.6/site
-packages (from watson-machine-learning-client) (0.3.3)
Requirement already satisfied: requests in /opt/conda/envs/Python36/lib/python3.6/si
te-packages (from watson-machine-learning-client) (2.21.0)
Requirement already satisfied: pandas in /opt/conda/envs/Python36/lib/python3.6/site
-packages (from watson-machine-learning-client) (0.24.1)
Requirement already satisfied: ibm-cos-sdk-core==2.*,>=2.0.0 in /opt/conda/envs/Pyth
on36/lib/python3.6/site-packages (from ibm-cos-sdk->watson-machine-learning-client)
(2.4.3)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.*,>=2.0.0 in /opt/conda/env
s/Python36/lib/python3.6/site-packages (from ibm-cos-sdk->watson-machine-learning-cl
ient) (2.4.3)
Requirement already satisfied: six>=1.10.0 in /opt/conda/envs/Python36/lib/python3.6
/site-packages (from lomond->watson-machine-learning-client) (1.12.0)
Requirement already satisfied: idna<2.9,>=2.5 in /opt/conda/envs/Python36/lib/python
3.6/site-packages (from requests->watson-machine-learning-client) (2.8)
```

```
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /opt/conda/envs/Python36/lib
/python3.6/site-packages (from requests->watson-machine-learning-client) (3.0.4)
Requirement already satisfied: pytz>=2011k in /opt/conda/envs/Python36/lib/python3.6
/site-packages (from pandas->watson-machine-learning-client) (2018.9)
Requirement already satisfied: python-dateutil>=2.5.0 in /opt/conda/envs/Python36/li
b/python3.6/site-packages (from pandas->watson-machine-learning-client) (2.7.5)
Requirement already satisfied: numpy>=1.12.0 in /opt/conda/envs/Python36/lib/python3
.6/site-packages (from pandas->watson-machine-learning-client) (1.15.4)
Requirement already satisfied: docutils>=0.10 in /opt/conda/envs/Python36/lib/python
3.6/site-packages (from ibm-cos-sdk-core==2.*,>=2.0.0->ibm-cos-sdk->watson-machine-l
earning-client) (0.14)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python36/li
b/python3.6/site-packages (from ibm-cos-sdk-core==2.*,>=2.0.0->ibm-cos-sdk->watson-m
achine-learning-client) (0.9.3)
```

In [107]:

```
from watson_machine_learning_client import WatsonMachineLearningAPIClient
```

```
2020-06-09 12:31:14,496 - watson_machine_learning_client.metanames - WARNING - 'AUTH
OR_EMAIL' meta prop is deprecated. It will be ignored.
```

In [108]:

```
wml_credentials={
  "apikey": "uhvHKcohSfK6vK0cDiUMBzFU6ZmJ8aQonqwukMVyWx_e",
  "instance_id": "e61430ad-72e7-4e3f-8e48-2f86e1525db1",
  "url": "https://eu-gb.ml.cloud.ibm.com"
}
```

In [109]:

```
client = WatsonMachineLearningAPIClient( wml_credentials )
```

In [110]:

```
model_props={client.repository.ModelMetaNames.AUTHOR_NAME: "Yash",
             client.repository.ModelMetaNames.AUTHOR_EMAIL: "bhattyash031@gmail.com"
,
             client.repository.ModelMetaNames.NAME: "LifeExpectancy"
            }
```

In [111]:

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

In [112]:

```
published_model_uid = client.repository.get_model_uid(model_artifact)
published_model_uid
```

Out[112]:

```
'f0b3d849-f121-42e1-8db4-1f4f94d4ed91'
```

In [113]:

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

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

```
Synchronous deployment creation for uid: 'f0b3d849-f121-42e1-8db4-1f4f94d4ed91' star
ted

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


INITIALIZING
DEPLOY_SUCCESS


------------------------------------------------------------------------------------
------------
Successfully finished deployment creation, deployment_uid='4d4889f8-081a-4a9e-b335-a
479440c673b'
------------------------------------------------------------------------------------
------------
```

Out[113]:

```
'https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/e61430ad-72e7-4e3f-8e48-2f86e1525db
1/deployments/4d4889f8-081a-4a9e-b335-a479440c673b/online'
```

NODE-RED Flow

Flows. Json

```
[{"id":"4a7dc5e5.a99c1c","type":"tab","label":"Flow 2","disabled":false,"info":""},{
"id":"36faa82a.927578","type":"tab","label":"Flow 1","disabled":true,"info":""},{"id
":"fb4a5c75.6b5d4","type":"tab","label":"Flow 3","disabled":true,"info":""},{"id":"3
c3b0d86.cc9742","type":"tab","label":"Flow 2","disabled":true,"info":""},{"id":"475e
ee25.213f9","type":"ui_base","theme":{"name":"theme-dark","lightTheme":{"default":"#
0094CE","baseColor":"#0d0d0d","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,
Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","edited":true,"reset"
:false},"darkTheme":{"default":"#097479","baseColor":"#f08000","baseFont":"Arial Bla
ck,Arial Black,Gadget,sans-serif","edited":true,"reset":false},"customTheme":{"name"
:"Untitled Theme 1","default":"#4B7930","baseColor":"#4B7930","baseFont":"-apple-sys
tem,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,s
ans-serif"},"themeState":{"base-color":{"default":"#097479","value":"#f08000","edite
d":true},"page-titlebar-backgroundColor":{"value":"#f08000","edited":false},"page-ba
ckgroundColor":{"value":"#111111","edited":false},"page-sidebar-backgroundColor":{"v
alue":"#ffffff","edited":false},"group-textColor":{"value":"#ffa53d","edited":false}
,"group-borderColor":{"value":"#555555","edited":false},"group-backgroundColor":{"va
lue":"#333333","edited":false},"widget-textColor":{"value":"#eeeeee","edited":false}
,"widget-backgroundColor":{"value":"#f08000","edited":false},"widget-borderColor":{"
value":"#333333","edited":false},"base-font":{"value":"Arial Black,Arial Black,Gadge
t,sans-serif"}},"angularTheme":{"primary":"indigo","accents":"blue","warn":"red","ba
ckground":"grey"}},"site":{"name":"Node-RED Dashboard","hideToolbar":"false","allowS
wipe":"false","lockMenu":"false","allowTempTheme":"true","dateFormat":"DD/MM/YYYY","
sizes":{"sx":48,"sy":48,"gx":6,"gy":6,"cx":6,"cy":6,"px":0,"py":0}}},{"id":"10555ae3
```

.fe6145","type":"ui_group","z":"","name":"LifeExpectancy","tab":"d0b94953.e50eb8","order":1,"disp":true,"width":"6","collapse":false},{"id":"d0b94953.e50eb8","type":"ui_tab","z":"","name":"Home","icon":"dashboard","disabled":false,"hidden":false},{"id":"b9705f88.560f1","type":"ui_group","z":"","name":"Life Expectancy Prediction","tab":"c6fbea6c.251768","order":1,"disp":true,"width":"6","collapse":false},{"id":"c6fbea6c.251768","type":"ui_tab","z":"","name":"Home Page","icon":"dashboard","disabled":false,"hidden":false},{"id":"2ac0c4ab.6bdcac","type":"ui_form","z":"4a7dc5e5.a99c1c","name":"","label":"","group":"10555ae3.fe6145","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":70,"y":100,"wires":[["8821b2c8.adb28"]]},{"id":"8821b2c8.adb28","type":"function","z":"4a7dc5e5.a99c1c","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);\nglobal.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.payload.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);\nglobal.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\",msg.payload.t);\nglobal.set(\"u\",msg.payload.u);\n\n//following are required to receive a token\nvar apikey=\"uhvHKcohSfK6vK0cDiUMBzFU6ZmJ8aQonqwukMVyWx_e\";\nmsg.headers={\"content-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":220,"y":100,"wires":[["2e1e365a.a6ab9a"]]},{"id":"40c739c5.0b76a8","type":"http request","z":"4a7dc5e5.a99c1c","name":"","method":"POST","ret":"obj","paytoqs":false,"url":"https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/e61430ad-72e7-4e3f-8e48-2f86e1525db1/deployments/4d4889f8-081a-4a9e-b335-a479440c673b/online","tls":"","persist":false,"proxy":"","authType":"basic","x":470,"y":180,"wires":[["85629e59.167e1","d8f31dd3.f744c"]]},{"id":"71dc2773.03b408","type":"debug","z":"4a7dc5e5.a99c1c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":750,"y":280,"wires":[]},{"id":"d8f31dd3.f744c","type":"function","z":"4a7dc5e5.a99c1c","name":"getFrom Endpoint","func":"msg.payload=msg.payload.values[0][0];\nreturn msg;","outputs":1,"noerr":0,"x":490,"y":280,"wires":[["71dc2773.03b408","716554.f3237aac"]]},{"id":"85629e59.167e1","type":"debug","z":"4a7dc5e5.a99c1c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":710,"y":180,"wires":[]},{"id":"8796cdb4.872ad","type":"function","z":"4a7dc5e5.a99c1c","name":"sendTo Endpoint","func":"//get token and make headers\nvar token=msg.payload.access_token;\nvar inst

ance_id=\"e61430ad-72e7-4e3f-8e48-2f86e1525db1\"\nmsg.headers={'Content-Type': 'appl
ication/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.ge
t(\"b\");\nvar c = global.get(\"c\");\nvar d = global.get(\"d\");\nvar e = global.ge
t(\"e\");\nvar f = global.get(\"f\");\nvar g = global.get(\"g\");\nvar h = global.ge
t(\"h\");\nvar i = global.get(\"i\");\nvar j = global.get(\"j\");\nvar k = global.ge
t(\"k\");\nvar l = global.get(\"l\");\nvar m = global.get(\"m\");\nvar n = global.ge
t(\"n\");\nvar o = global.get(\"o\");\nvar p = global.get(\"p\");\nvar q = global.ge
t(\"q\");\nvar r = global.get(\"r\");\nvar s = global.get(\"s\");\nvar t = global.ge
t(\"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
":210,"y":180,"wires":[["40c739c5.0b76a8"]]},{"id":"2e1e365a.a6ab9a","type":"http re
quest","z":"4a7dc5e5.a99c1c","name":"","method":"POST","ret":"obj","paytoqs":false,"
url":"https://iam.cloud.ibm.com/identity/token","tls":"","persist":false,"proxy":"",
"authType":"basic","x":370,"y":100,"wires":[["8796cdb4.872ad"]]},{"id":"716554.f3237
aac","type":"ui_text","z":"4a7dc5e5.a99c1c","group":"10555ae3.fe6145","order":2,"wid
th":0,"height":0,"name":"","label":"Prediction","format":"{{msg.payload}}","layout":
"row-spread","x":720,"y":400,"wires":[]}]

THANKYOU