PROJECT REPORT

Name: Uddeshya Gupta

E-mail: uddeshya.gupta@gmail.com

Project Name: Predicting Life Expectancy Using Machine Learning

Date: 25th May, 2020

Project scope: The scope of this project is " **Predicting Life Expectancy Using Machine Learning**" in this project we are given task to predict the life expectancy, life expectancy is the average time period for which the subject lives.

Project schedule:

1. Understanding what to do in the above given Project
2. Identify and get familiar to the tools needed to complete this project
3. Writing codes
4. Collecting Data sets
5. The time duration **23.5 days**

Deliverables:

1. Predicting Life Expectancy Using Machine Learning.
2. Making a user interface too as front-end work and writing code as backend work to make the user to interact and calculate the Life Expectancy.

Setting The Development Environment:

1. Creating GitHub account
2. Creating Slack account
3. Signing-up for cloud services
    i. Node-Red for front end
    ii. Watson Studio for coding
    iii. Machine Learning services

1. INTRODUCTION

1.1 Overview:

This project is based on predicting the life expectancy of a person. It is the statistical average of the number of years a person is expected to live. Factors affecting life expectancy are Country, Mental and Physical Illness, lifestyle, diet, health care services, financial condition, BMI, alcohol consumption, Diseases etc.

Here in this project our motive is to find life expectancy of a person after providing details such as the country he is living in is developed or is developing, BMI of the person, Disease history, Income, Population of that country, Expenditure etc. So here I have used Machine learning and Artificial Intelligence to predict the life expectancy. The data used in training of the model was the data by WHO taken from Kaggle. There were almost 22 columns stating different factor affecting Life expectancy and 2939 rows comprising data of different persons from different countries. Based on the results we got on Watson Studio some factors which were not affecting the Life expectancy much were removed and then scoring end point was obtained after running full code. This scoring end point is the URL that helps us to send payload data to a model or function development for analysis (such as to classify the data or to make prediction).

After obtaining the end point the next step is to work on Node red which is the platform, we can use for developing our front-end page that will have a form asking you your details such as year of birth, adult mortality, infant deaths, BMI etc. rest we'll discuss in details later-on.

Requirements: IBM Cloud, GitHub, Slack, IBM Watson, Node-Red

1.2 **Purpose:**

The Purpose of this project is to build a model that will predict the Life Expectancy of a person after giving the details of the BMI, Expenditure, Disease history etc.

2. Literature Survey

2.1 Existing Problem:

Life expectancy is one of the most important factors in end-of-life decision making. Good prophecy for example helps to determine 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. Physicians, however, tend to overestimate life expectancy, and miss the window of opportunity to initiate Advance Care Planning. This research tests the potential of using machine learning and Artificial Intelligence techniques for predicting life expectancy from electronic medical records.

Accurate prognosis of life expectancy is essential for general practitioners (GPs) to decide when to introduce the topic of ACP to the patient, and it is a key determinant in end-of-life decisions. Increasing the accuracy of prognoses has the potential to benefit patients in various ways by enabling more consistent ACP, earlier and better anticipation on palliative needs, and preventing excessive treatment. This study focuses on automatic life expectancy prediction based on medical records.

Once we have estimates of the fraction of people dying across age intervals, it is simple to calculate a 'life table' showing the evolving probabilities of survival and the corresponding life expectancies by age. Period life expectancy figures can be obtained from 'period life tables' (i.e. life tables that rely on age-specific mortality rates observed from deaths among individuals of different age groups at a fixed point in time). And similarly, cohort life expectancy figures can be obtained from 'cohort life tables' (i.e. life tables that rely on age-specific mortality rates observed from tracking and forecasting the death and survival of a group of people as they become older).

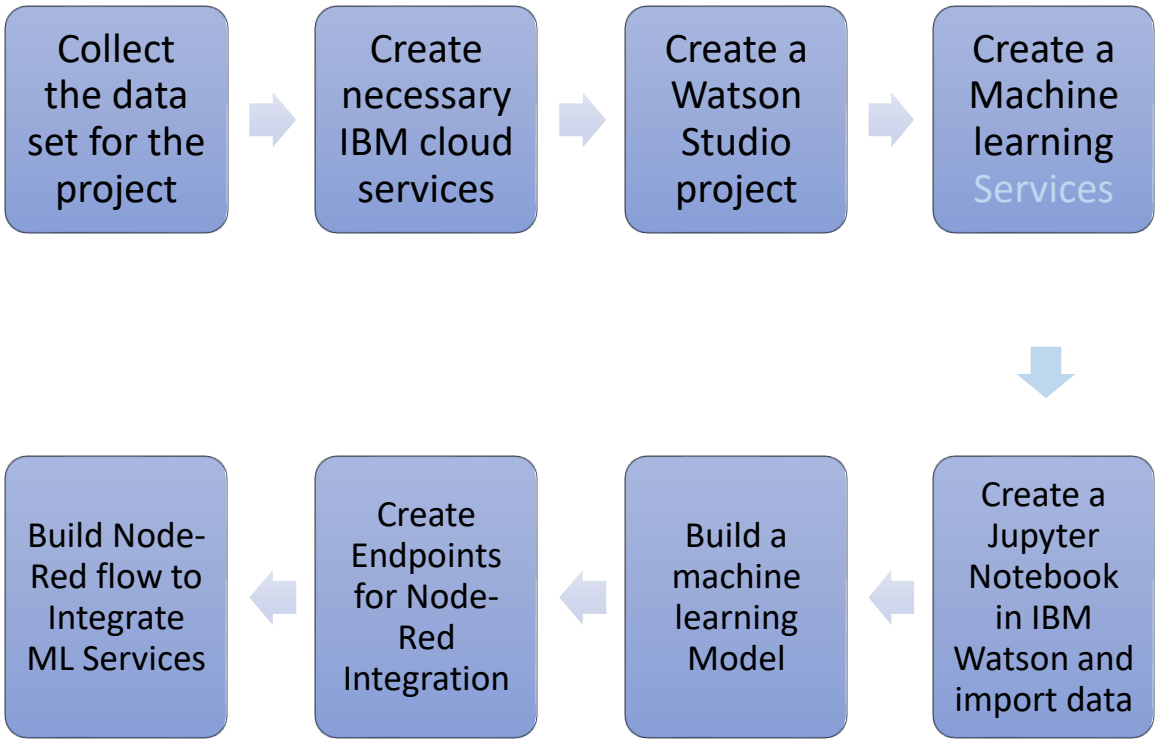
2.2 Proposed Solution:

The 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 data of different persons their Physical health, Mental health etc of the time frame of 2000 to 2015. The data was taken from the website: https://www.kaggle.com/kumarajarshi/life-expectancy-who/data. 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.

So, using machine learning technique we suppose to predict the value of Life Expectancy based on some common attributes like 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 etc. Anyone can find this data and get the Life Expectancy value based on the their Country and Year.

3. Theoretical Survey

3.1 Block diagram



Block Diagram for Predicting Life Expectancy with Python

3.2 Hardware/Software Designing:

- **GitHub**

  GitHub is the largest community of developers in the world with millions of people sharing their projects, ideas for benefiting many people in a very unique way. Any person living in any corner of this world can access this

platform for his/her benefit. They can share their problem, their ideas, solution to some problems. In simple words it is basically a platform in which anyone can come and share their problems and solutions. It is easy to manage. A team working on same project can easily monitor the progress and can easily access their work anywhere.

- **Slack**

  It is a messaging tool which is intended to contact your internal team esily. As it gives you a platform through which we can communicate to our team members easily under one roof. It is not as hectic as sending mails and reading them. It directly comes as message to you from the group created having your team members. It is great if you are having a team more than 2 members. Searching of messages become easy, fast medium, searching old messages.

- **IBM Cloud**

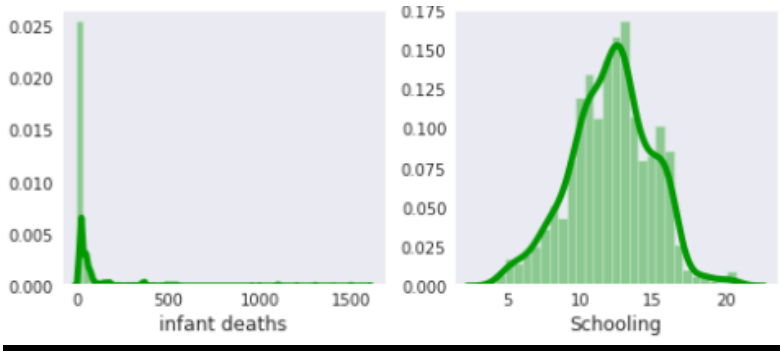
  It is the platform that enables us to use its various features such as Watson Studio that provides a platform where we can write our python code for and observe our results in the form of heat maps, graphs and tables. In this project we used it and got our scoring end point. It is the URL that helps us to send payload data to a model or function development for analysis (such as to classify the data or to make prediction).

- **Node Red**

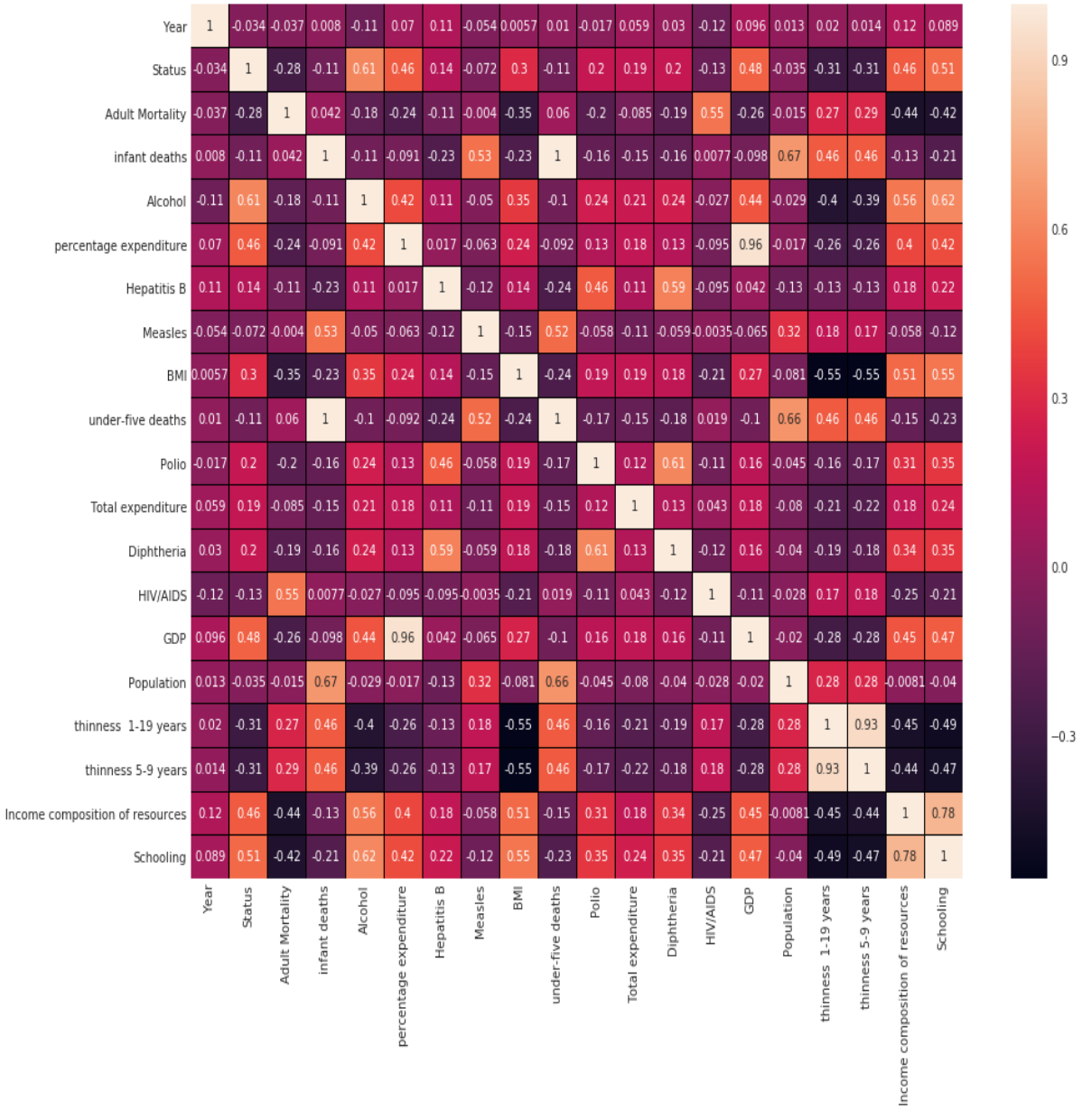
  Node-Red helps us to create a front-end window on which we can get the data from the user such as his Year, BMI, Alcohol intake etc. and it will then connect to the code written on Watson Studio via the scoring end point created after running the python code.

4. Experimental Investigations

The graphs of various Factors affecting the prediction of life expectancy is shown in figure given below:



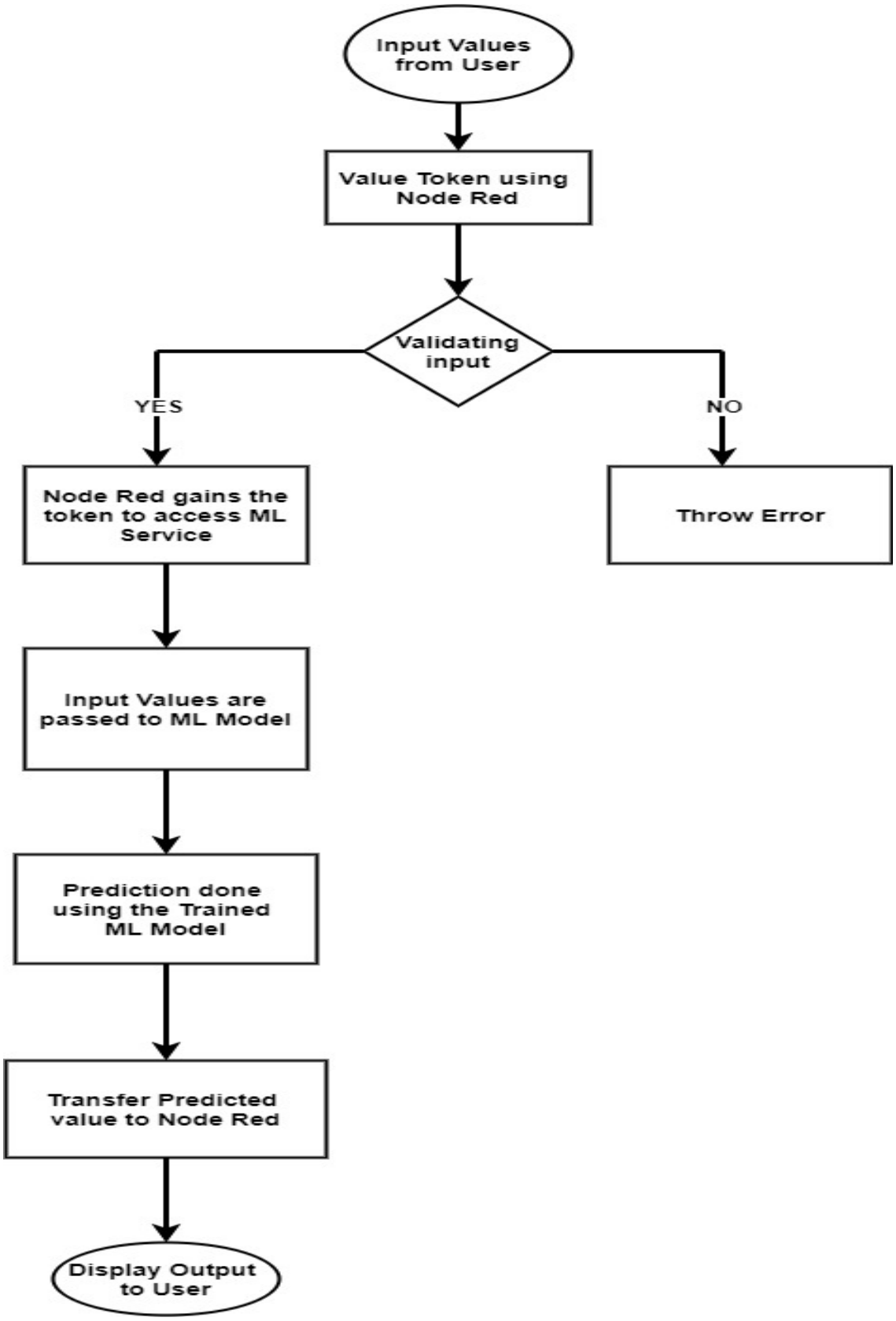
Curves of life expectancy v/s different factors



Heat map of different factors

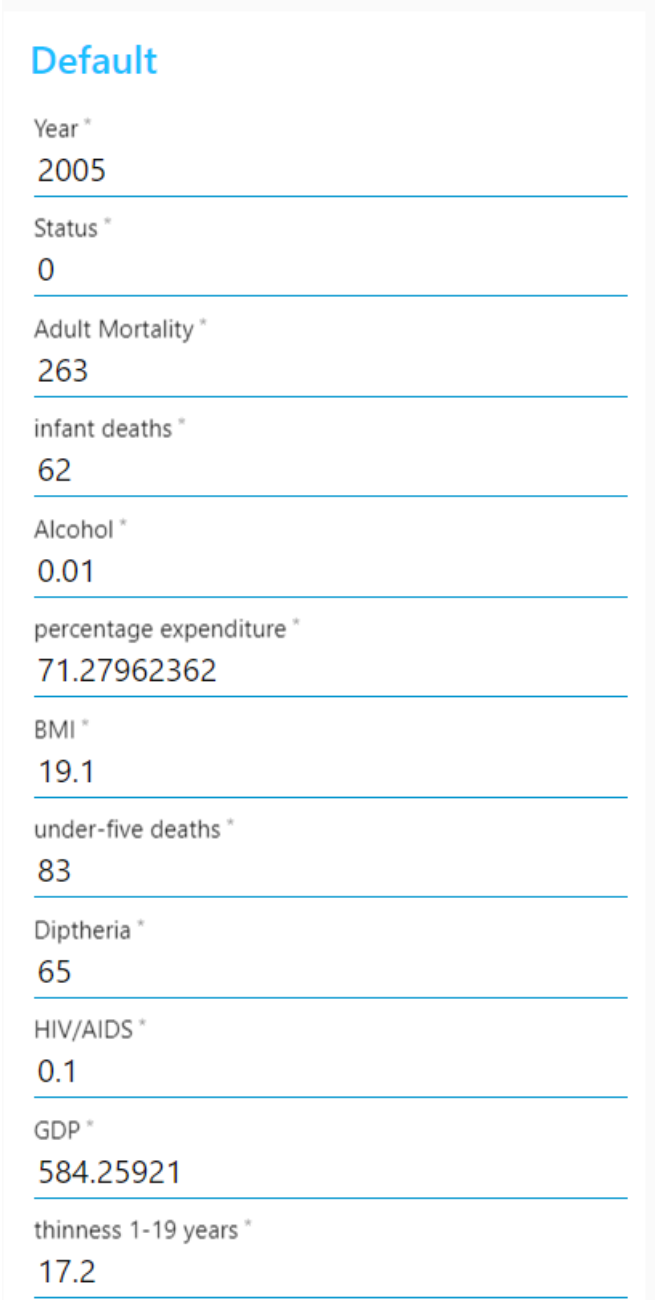
Shown above is the heat map of the various factors affecting other various factors some of them have positive values some of them have negative but the thing we have to keep in mind that we can't neglect the factors having negative value because it will have the adverse effect which will affect the life expectancy. After some observations I decided not to include 6 factors that are not affecting life expectancy much and will reduce the calculations and make our model less complex.

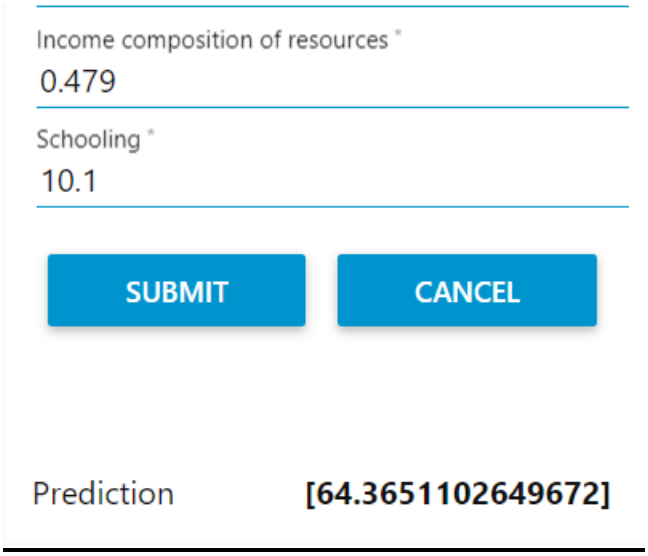
5.Flow Chart


Input Values from User
Value Token using Node Red
Validating input
YES
NO
Node Red gains the token to access ML Service
Throw Error
Input Values are passed to ML Model
Prediction done using the Trained ML Model
Transfer Predicted value to Node Red
Display Output to User

6. Result

After filling all the necessary details asked in the UI form, we got the prediction of the life expectancy. The accuracy of our model was 94.41%





Screenshot of the prediction of life expectancy obtained

7. Advantages and Disadvantages

7.1 Advantages:

- Easily identifies trends and patterns
- Wide Applications
- Handling multi-dimensional and multi-variety data
- No human intervention needed (automation)
- Continuous Improvement

7.2 Disadvantages:

- High error-susceptibility
- Needs a lot of time to implement
- Interpreting the results accurately
- Data set collection is a complex task

8. Applications

1. The form created is easy to understand and is easy to fill by anyone.
2. It can be used for monitoring health conditions in a particular country
3. It can be used to get the data about the factor affecting Life expectancy the most in order to work in the direction of obtaining high life expectancy
4. It can be used to develop statistics for country development process

9. Conclusions

This user interface enables any user to predict the life expectancy value of anyone on the basis of the details asked in the form. Our research shows that machine learning and natural language processing techniques offer a feasible and promising approach to predicting life expectancy. The research has potential for real-life applications, such as supporting timely recognition of the right moment to start Advance Care Planning.

10. Future Scope

1. Increase model accuracy
2. Gives suggestion on how to increase Life Expectancy
3. Mental health data was missing from the WHO data set which also plays the important role in affecting life expectancy
4. The scalability and flexibility of the application can be improved.

11. Bibliography

- https://smartinternz.com/
- https://app.slack.com/
- https://github.com/
- https://cloud.ibm.com/
- https://node-red-kxdba.mybluemix.net/
- https://node-red-kxdba.mybluemix.net/ui/#!/0?socketid=HzpHVbmG7CYZn_klAAAN
- https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/1ca640db-342f-4a25-a665-d3217fdf81c3?projectid=ba3a3a9e-4415-4221-8d90-486fc848d7e7&context=wdp

Apendix

Source code

```
import pandas as pd

import numpy as np

import matplotlib. pyplot as plt

import seaborn as sns

plt.style.use('fivethirtyeight')

sns.set_style(style='dark')

pd.options.display.max_rows = None

pd.options.display.max_columns = None

%matplotlib inline

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_af74b09b9a4a4324a213f40c677ad593 = ibm_boto3.client(service_name='s3',
```

```
    ibm_api_key_id='lLZrrGcEzrSA7hKsLhCbeWy2ERezNMnwv5pKk2fkwK4f',
    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_af74b09b9a4a4324a213f40c677ad593.get_object(Bucket='lifeexpectancy-donotdelete-pr-ykz2okt8ldv7k6',Key='Life.csv')['Body']
# add missing __iter__ method, so pandas accept body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )
df_data_1 = pd.read_csv(body)
df_data_1.head()
df.describe()
df.columns = list(map(str.strip, df.columns.tolist()))
df.isnull().sum()
df.dropna(axis=0, inplace=True)
df.shape
country = df.Country
df.drop(['Country'], axis=1, inplace=True)
df.columns.tolist()
df.Status.value_counts()
val = {'Developing':0, 'Developed':1}
df.Status = df.Status.map(val)
df.dtypes
y_train = pd.DataFrame(df['Life expectancy'])
X_train = df.drop(['Life expectancy'], axis=1)
fig, axes = plt.subplots(4,5, figsize=(19,15))
```

```
for index, column in enumerate(X_train.columns.tolist()):
    sns.distplot(a = X_train[column],
                 color= '#009900' ,
                ax = axes[index%4][index%5])
cor= df.corr()
cor['Life expectancy'].sort_values(ascending=False)
sns.jointplot(x='Schooling', y='Life expectancy',
              data=df, alpha=0.2,
              color= '#ff4d4d', size=7)


sns.jointplot(x='Adult Mortality', y='Life expectancy',
              data=df, alpha=0.2,
              color= '#ff4d4d', size=7)
fig, axes = plt.subplots(2,2, figsize=(17,15))
size_set = ((-100, 1300), (-200, 22000000))
for index, column in enumerate(['Measles', 'Population']):
    sns.boxplot(y=column, data = X_train,
                color = '#e60073', ax = axes[index][0])
    sns.boxplot(y=column, data=X_train,
                color = '#e60073', ax= axes[index][1])
    axes[index][1].set_ylim(size_set[index])


axes[0][0].set_title('Box plot for Measles')
axes[0][1].set_title('Zooming Box plot for Measles to see the quartiles and IQR clearly')
```

```
axes[1][0].set_title('Box plot for Population')

axes[1][1].set_title('Zooming Box plot for Population to see the quartiles and IQR clearly')

plt.show()

X_train.describe()[['Measles','Population']]

from pandas.plotting import scatter_matrix

scatter_matrix(X_train[['Schooling','Income composition of resources','BMI','thinness  1-19 years','HIV/AIDS','Adult Mortality']],

               figsize=(20,20))

plt.plot()

corr = X_train.corr()

fig, ax = plt.subplots(figsize=(14, 12))

sns.heatmap(corr, linewidth=0.01, linecolor='black', annot=True, ax=ax)

def get_corr(data, threshold):

    cor_set = set()

    for i in range(data.shape[0]):

        for j in range(i):

            if abs(data.iloc[i, j]) >= threshold:

                cor_set.add(data.columns[j])

    return cor_set

corr_group = get_corr(corr, 0.95)

corr_group

X_train.shape

X_train.dtypes

from statsmodels.api import OLS

ols = OLS(endog=y_train, exog=X_train).fit()
```

```
ols.summary()

X_train.drop(['thinness 5-9 years','Population','Total expenditure','Measles','Hepatitiace=True)

pd.options.display.max_rows = None

pd.options.display.max_columns = None

ols = OLS(endog=y_train, exog=X_train).fit()

ols.summary()


from sklearn.model_selection import train_test_split

#from sklearn.preprocessing import StandardScaler

#std = StandardScaler()

#X_train = std.fit_transform(X_train)

X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.15, random_state=4)

X_train.shape, X_test.shape

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

linear = LinearRegression()

linear.fit(X_train, y_train)

y_pred = linear.predict(X_test)

#r2_score

score = r2_score(y_test, y_pred)

print(f'r2 score: {score}')

#mse

error = mean_squared_error(y_test, y_pred)

print("mean squared error: {}".format(error))
```

```
from sklearn.ensemble import RandomForestRegressor

rf_regressor = RandomForestRegressor(n_estimators = 300, random_state = 0)

rf_regressor.fit(X_train,y_train)

print(rf_regressor.score(X_test, y_test))

y_pred = rf_regressor.predict(X_test)

#r2_score

score = r2_score(y_test, y_pred)

print(f'r2 score: {score}')

#mse

error = mean_squared_error(y_test, y_pred)

print("mean squared error: {}".format(error))

#mae

error = mean_absolute_error(y_test, y_pred)

print(f'mean absolute error: {error}')

#mae

error = mean_absolute_error(y_test, y_pred)

print(f'mean absolute error: {error}')

!pip install watson-machine-learning-client

from watson_machine_learning_client import WatsonMachineLearningAPIClient

client = WatsonMachineLearningAPIClient( wml_credentials )

model_props = {client.repository.ModelMetaNames.AUTHOR_NAME: "Uddeshya Gupta",

        client.repository.ModelMetaNames.AUTHOR_EMAIL:
"uddeshya.gupta@gmail.com",

        client.repository.ModelMetaNames.NAME: "Life Expectancy Prediction"}

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

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

published_model_uid

client.deployments.list()

deployment = client.deployments.create(published_model_uid, name="LifeExpectancyPrediction")

scoring_endpoint = client.deployments.get_scoring_url(deployment)

scoring_endpoint
```