

Predicting Life Expectancy Using Machine Learning

INTRODUCTION

1.1 Overview

The project aims at prediction of life span rate of a given country. Here, we attempt to study the relationship between life expectancy and life span depending upon the ageing and death patterns of individuals. Monitoring the life span of an individual by considering various characteristics like topographical situations, Long-term illnesses including mental and physical both, birth year and many more statistical factors that has been studied using one of the different machine learning algorithms.

The life-expectancy predictions are done using Random Forest Regression machine learning algorithm and the deployment of the project is done using IBM Cloud.

The web interface for the project is deployed using Node RED application wherein the user may enter the required fields and get the average life expectancy value as an output.

1.2 Purpose

The term “life expectancy” refers to the number of years a person can expect to live. By definition, life expectancy is based on an estimate of the average age that members of a particular population group will be when they die. The purpose of this project is to predict average life expectancy of a country. Life expectancy is used in pricing and underwriting life insurance and insurance products like annuities, as well as in retirement and pension planning.

LITERATURE SURVEY

2.1 Existing Problem

- It is unable to give statistics regarding the deaths due to morbidity.
- The reasons for increased death rates remains unknown to many.
- Due to inappropriate reports on death analysis, the government is unable to provide the necessary medical and sanitation facilities to the citizens.

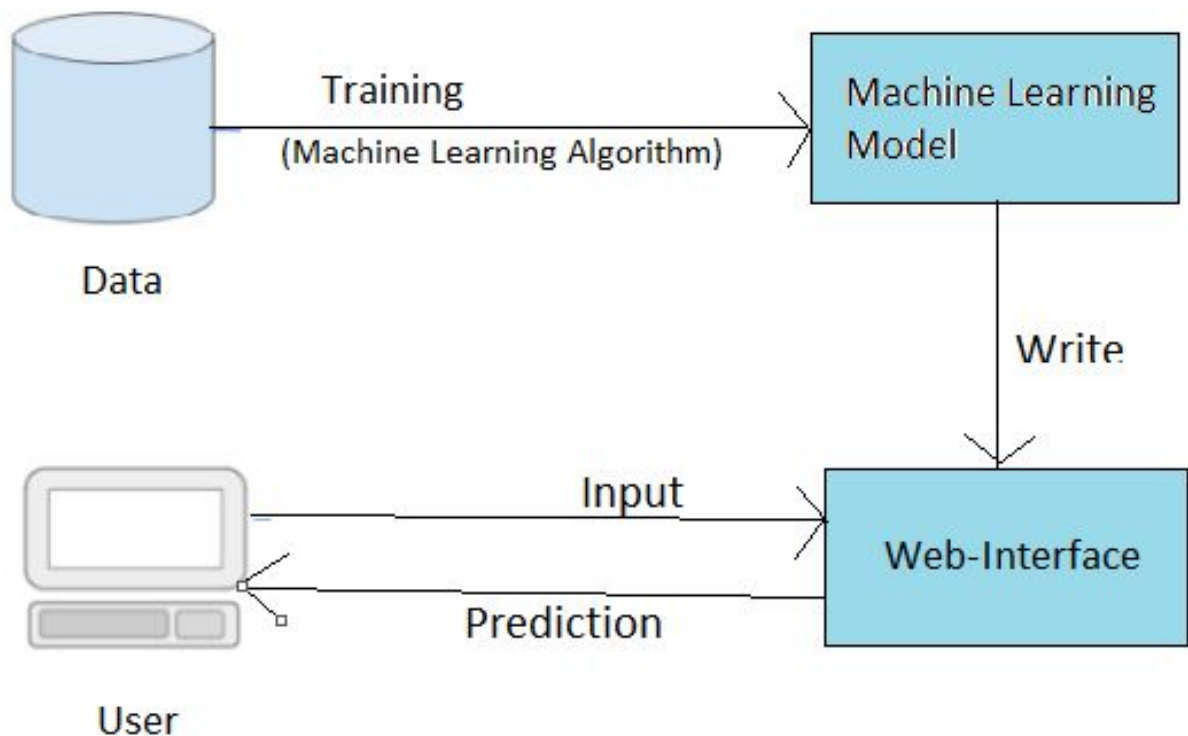
- People are not educated on the fatal consequences of a deadly disease.

2.2 Proposed Solution

- An efficient and easily accessible machine learning model has been proposed to analyze the relationship between dependent variable (life expectancy) and a number of other factors.
- These project predictions may prove to be beneficial to analyze the factors related to life expectancy.

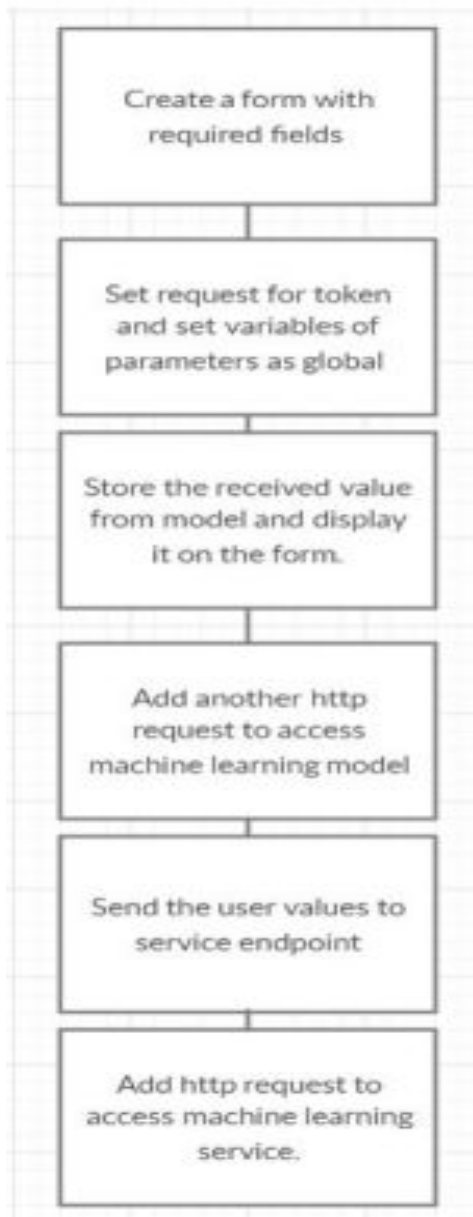
THEORETICAL ANALYSIS

3.1 Block Diagram



3.2 Software Designing

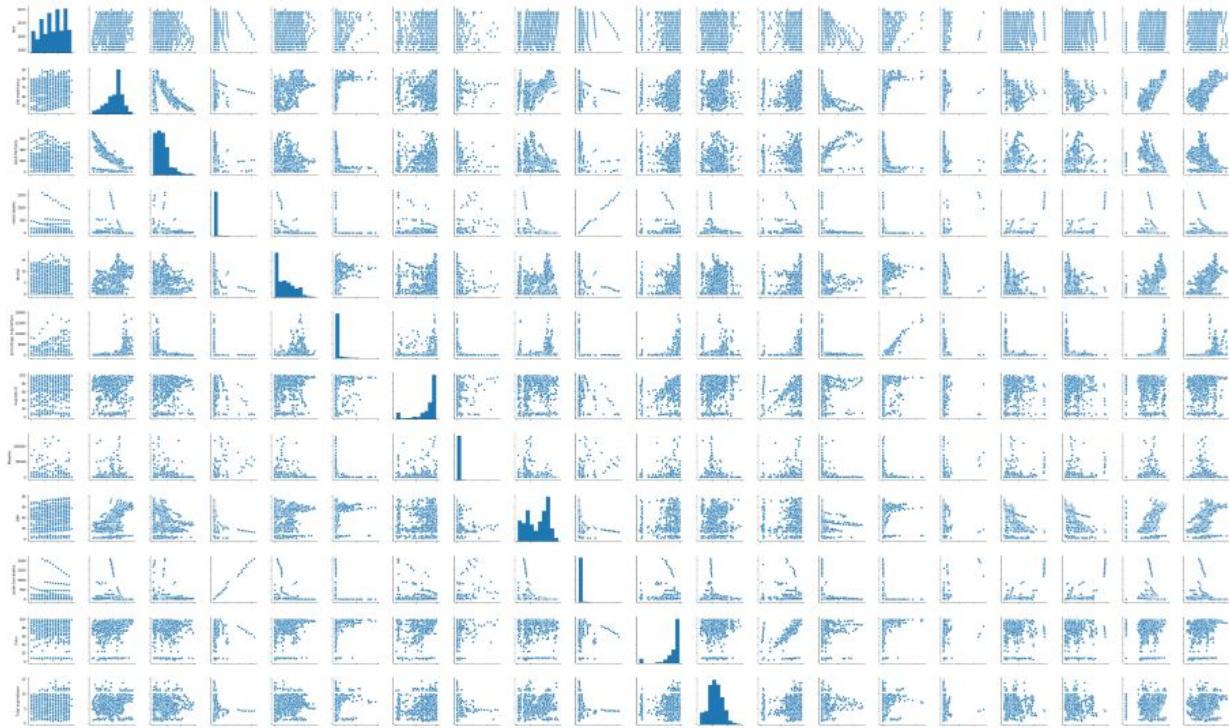
Node-Red App Designing :



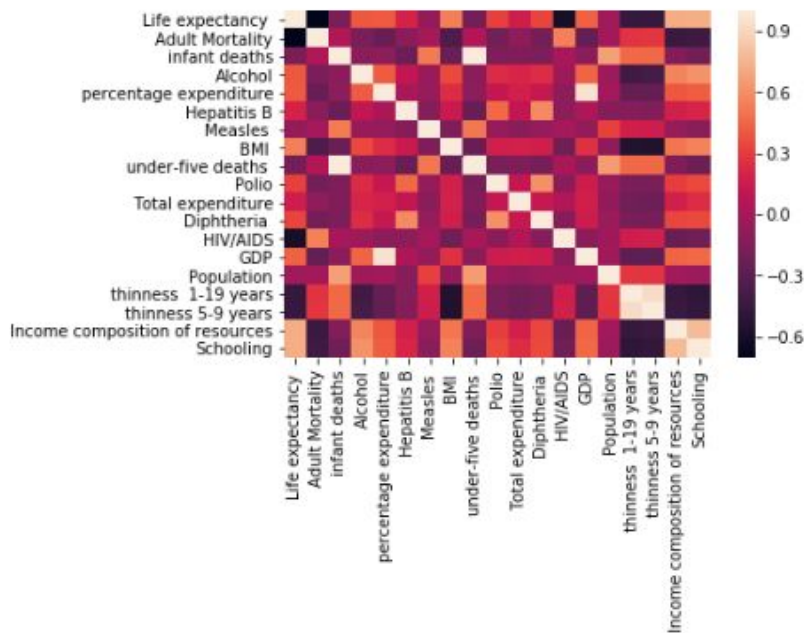
EXPERIMENTAL INVESTIGATIONS

Data of dataset is analyzed using different plots :

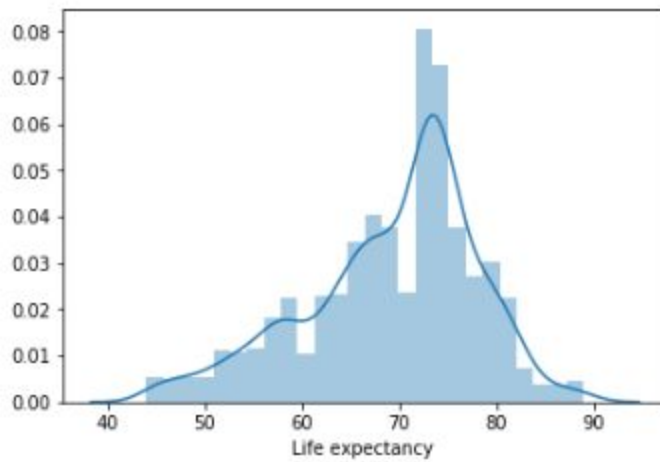
1. Pair Plot :



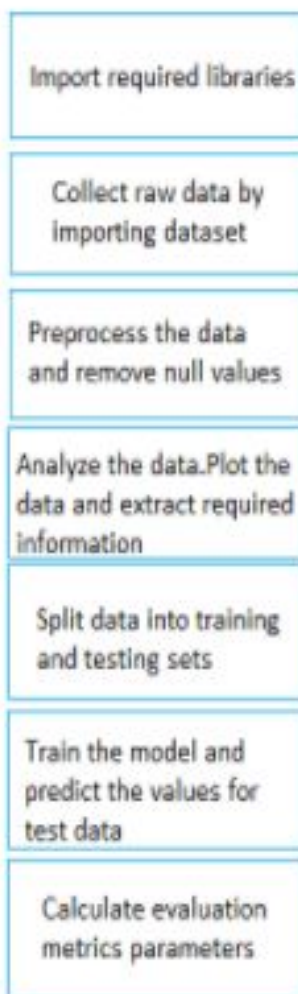
2.HeatMap :



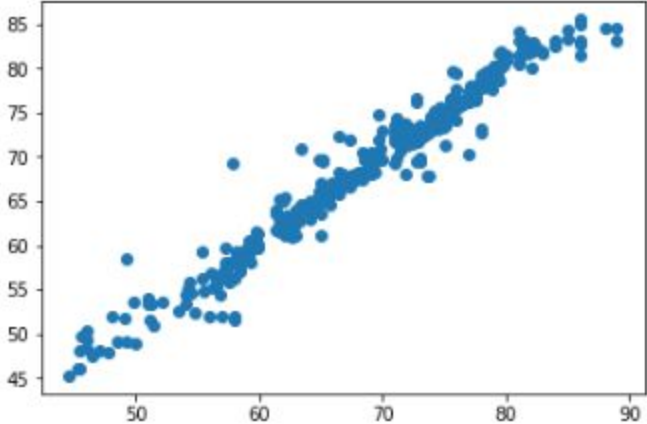
3.DistPlot :



FLOWCHART



RESULT

Parameters	Random Forest Regression
Accuracy	95.6%
Scatter Plot	
MAE	1.1583535108958865
MSE	3.4548060968523013
RMSE	1.8587108696223578

ADVANTAGES

- Appropriate statistics will be available to analyse the death rate of a particular area.
- The reasons of death will be examined and hence awareness will be promoted among citizens.
- Proper amount of medical and sanitation facilities would be available for everyone

DISADVANTAGES

- It requires internet connection.
- Application is unable to predict value for multiple sets of data at the same time.

APPLICATIONS

- Life expectancy plays an important role in building the financial world like insurance, pension planning, etc.
- It helps in development of healthcare and sanitation facilities.

CONCLUSION

Hence considering various parameters life expectancy value can be calculated easily by the user with the help of Node-Red app. Thus reducing the complexity and also can be used by a person with less technical knowledge.

FUTURE SCOPE

- Accuracy of the model can be improved further.
- The system can be modified in a way such that the user will be able to predict life expectancy values for multiple sets of data simultaneously.

BIBLIOGRAPHY

- <https://www.who.int/whosis/whostat2006DefinitionsAndMetadata.pdf>
- [https://www.thelancet.com/journals/lancet/article/PIIS0140-6736\(15\)60296-3](https://www.thelancet.com/journals/lancet/article/PIIS0140-6736(15)60296-3)
- <https://www.news-medical.net/health/What-is-Life-Expectancy.aspx>

APPENDIX

A.Source Code

```
# -*- coding: utf-8 -*-
```

""Life expectancy (1).ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1u6XIINVG8uJbYSDK3_BrMNF2BrVdXkIH

Code:

```
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_8d89fa22a78a4b04971d60158bbd2857 =
```

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

```
    ibm_api_key_id='-osT57k3a4iHMqEwVm3wHt86rKniaFuJtyUrcVC_9Elv',
```

```
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
```

```
    config=Config(signature_version='oauth'),
```

```
    endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.com')
```

```
body =
```

```
client_8d89fa22a78a4b04971d60158bbd2857.get_object(Bucket='nikita-donotdelete-pr-oraobiuwbf8mpt',Key='datasets_12603_17232_Life Expectancy Data  
(1).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_data_1 = pd.read_csv(body)
```



```
df_data_1.head()
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn import metrics
```

```
df_data_1.dropna(inplace=True)
```

```
df_data_1['Country']=df_data_1['Country'].astype('category')
df_data_1['Year']=df_data_1['Year'].astype('category')
df_data_1['Status']=df_data_1['Status'].astype('category')
df_data_1
```

```
df_data_1.info()
```

```
df_data_1.describe()
```

```
df_data_1.columns
```

```
sns.pairplot(df_data_1)
```

```
sns.distplot(df_data_1['Life expectancy '])
```

```
sns.heatmap(df_data_1.corr())
```

```
real_x=df_data_1[['Country', 'Year', 'Status', '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']]
real_y=df_data_1['Life expectancy ']
real_x

print(df_data_1['Country'].value_counts())

print(df_data_1['Status'].value_counts())

print(df_data_1['Year'].value_counts().count())

labelencoder=LabelEncoder()
real_x['Country']=labelencoder.fit_transform(real_x['Country'])
real_x['Year']=labelencoder.fit_transform(real_x['Year'])
real_x['Status']=labelencoder.fit_transform(real_x['Status'])
real_x

train_x,test_x,train_y,test_y=train_test_split(real_x,real_y,test_size=0.25,random_
state=0)
test_x

reg=RandomForestRegressor(n_estimators=100,random_state=0)

reg.fit(train_x,train_y)

predr_y=reg.predict(test_x)

plt.scatter(test_y,predr_y)

print('MAE:', metrics.mean_absolute_error(test_y, predr_y))

```

```
print('MSE:', metrics.mean_squared_error(test_y, predr_y))
print('RMSE:', np.sqrt(metrics.mean_squared_error(test_y, predr_y)))
```

```
p=reg.predict(np.array([0,15,0,263,62,0.01,71.27962362,65,1154,19.1,83,6,8.16,65
,0.1,584.25921,33736494,17.2,17.3,0.479,10.1
])).reshape(1, -1))
p
```

```
sns.distplot((predr_y),bins=50);
```

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

```
from watson_machine_learning_client import WatsonMachineLearningAPIClient
```

```
wml_credentials={
    "apikey": "r67lRK72BJFOmG3_5nyDOPghzZO1UayvC57FHWYnfdUX",
    "instance_id": "cb4e8990-e53e-4b67-a142-d79a27583302",
    "url": "https://us-south.ml.cloud.ibm.com"
}
```

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

```
model_props = {client.repository.ModelMetaNames.AUTHOR_NAME: "Nikita",
                client.repository.ModelMetaNames.AUTHOR_EMAIL:
                "nikitapatel0723@gmail.com",
                client.repository.ModelMetaNames.NAME: "Life expectancy"}
```

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

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

```
deployment = client.deployments.create(published_model_uid, name="Life  
expectancy")
```

```
scoring_endpoint =  
client.deployments.get_scoring_url(deployment)scoring_endpoint
```