INTERNSHIP REPORT

PREDICTING LIFE EXPECTANCY USING MACHINE LEARNING - SB52080

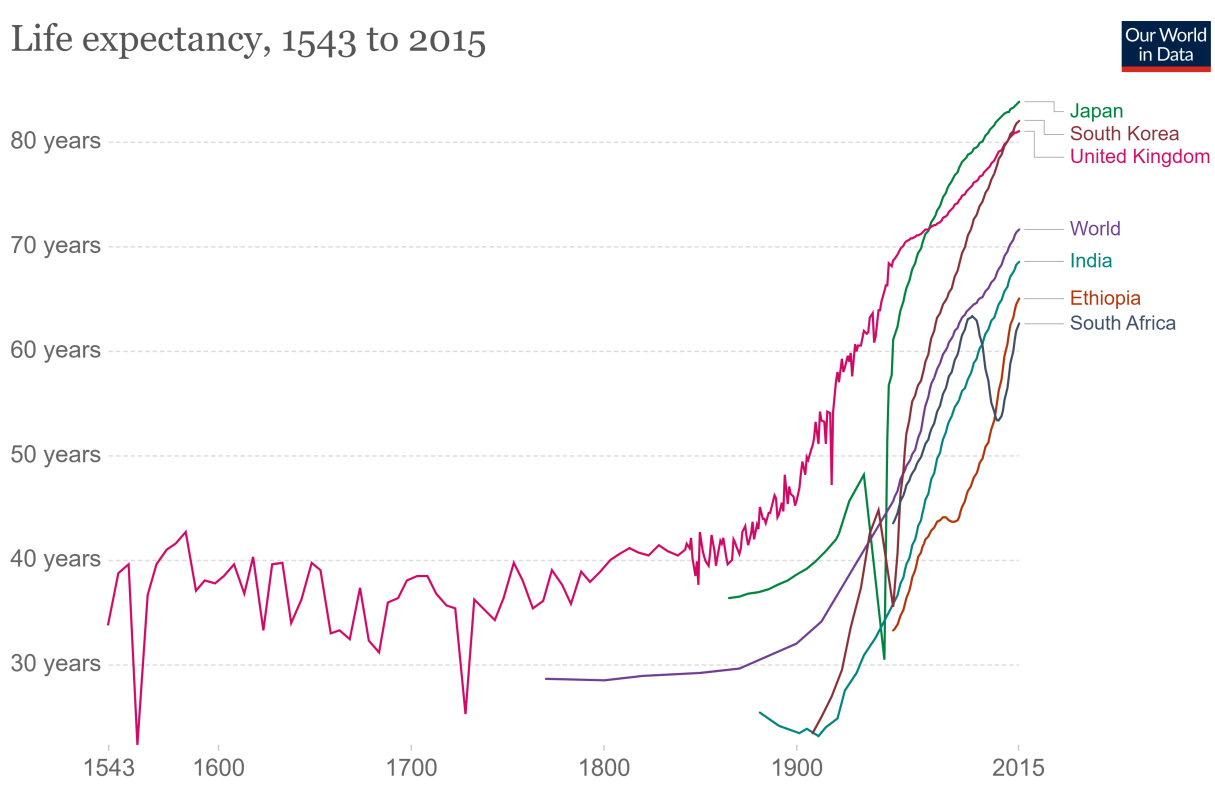
Project ID	**:**	**SPS_PRO_215**
Category	**:**	**Machine Learning**
Internship under	**:**	**SMARTBRIDGE**
Submitted by	**:**	**Swaminathan M**

1. INTRODUCTION

1.1 Overview

Life expectancy is the key metric for assessing population health. Broader than the narrow metric of the infant and child mortality, which focus solely at mortality at a young age, life expectancy captures the mortality along the entire life course. It tells us the average age of death in a population.

Estimates suggest that in a pre-modern, poor world, life expectancy was around 30 years in all regions of the world.



Life expectancy has increased rapidly since the Age of Enlightenment. In the early 19th century, life expectancy started to increase in the early industrialized countries while it stayed low in the rest of the world. This led to a very high inequality in how health was distributed across the world.

Good health in the rich countries and persistently bad health in those countries that remained poor. Over the last decades this global inequality decreased. No country in the world has a lower life expectancy than the countries with the highest life expectancy in 1800. Many countries that not long ago were suffering from bad health are catching up rapidly.

Since 1900 the global average life expectancy has more than doubled and is now above 70 years. The inequality of life expectancy is still very large across and within countries. in 2019 the country with the lowest life expectancy is the Central African Republic with 53 years, in Japan life expectancy is 30 years longer.

1.2 Purpose

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

2. LITERATURE SURVEY

2.1 Existing problem

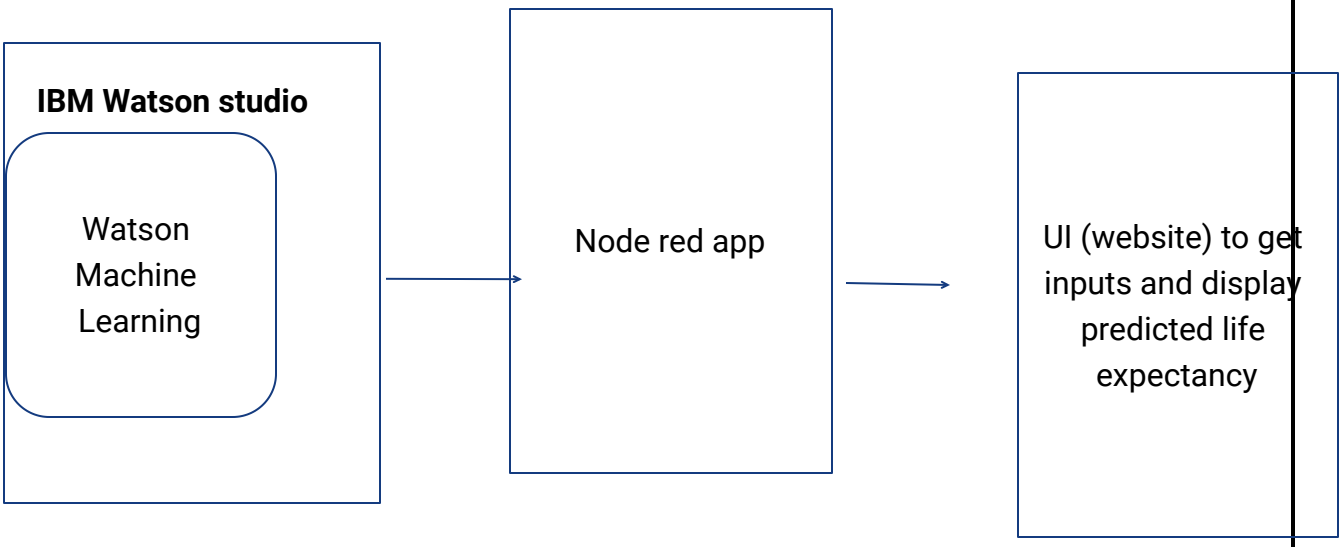
The existing problem is to predict the life expectancy of such a huge population based on the various factors. Predicting life expectancy helps in the development of various fields. For example, life expectancy plays a major role in helping life insurance companies. It also plays a vital role in several other industries. So, this life expectancy needs to predicted with more accuracy to gain more insights. Starting from manual methods, statisticians tried various techniques in predicting life expectancy. But still they are working for a much higher accuracy.

2.2 Proposed solution

To predict life expectancy, this project proposes a machine learning model to visualise the data from different perspectives with the help of various tools and to predict the life expectancy with a high accuracy. The proposed solution works on a dataset which consists of historical data on various factors and the life expectancy. It splits the data into train and test datasets and train a machine learning model with a regression algorithm with the train dataset. The trained model is tested with the test dataset to determine the accuracy of the model. Thus, the proposed solution will ease the work of predicting life expectancy with Machine Learning algorithms.

3. THEORETICAL ANALYSIS

3.1 Block diagram



3.2 Watson Machine Learning model

3.2.1 Watson studio

IBM Watson Studio, part of IBM Watson, can increase productivity by giving a single environment to work with the best of open source and IBM software, to build and deploy an AI solution. IBM Watson Studio accelerates the machine and deep learning workflows required to infuse AI into business to drive innovation. It provides a suite of tools for data scientists, application developers and subject matter experts to collaboratively and easily work with data and use that data to build, train and deploy models at scale.

3.2.2 Watson Machine Learning

IBM Watson Machine Learning is an IBM Cloud service that's available through IBM Watson Studio.

Machine learning is everywhere – influencing nearly everything. Uber is the world's largest taxi company, yet owns no vehicles. Facebook, the world's most popular media owner, creates no content. Alibaba, the most valuable retailer, has no inventory.

And Airbnb, the world's largest accommodation provider, owns no real estate. But what one hasn't explicitly heard is that all of these companies are machine learning companies at their very core. Companies like Netflix use machine learning to recommend movies for us to watch. Navigation apps like Waze use machine learning to help optimize our driving experience.

Using IBM Watson Machine Learning, one can build analytical models and neural networks, trained with own data, that one can deploy for use in applications.

Watson Machine Learning provides a full range of tools and services so one can build, train, and deploy Machine Learning models.

3.2.3 Building own ML model

A machine learning service is created with object cloud storage and it is connected with IBM watson. A new project is created within which a new notebook is created to develop an ML model. For the given problem statement, dataset is fetched from Kaggle, a subsidiary of Google Inc. And the dataset is split into train and test datasets for the ML model. Since, the expected result is life expectancy, which is a continuous value, regression algorithms are used. With Python and scikit learn library, an ML model is trained and it is tested with the help of test dataset. Its accuracy is determined with the predicted output and actual results of test dataset. With the help of Watson Machine Learning API Client, the API credentials are used to create a scoring endpoint, which is then linked with the Node-RED.

3.2.4 AutoAI experiment

AutoAI experiment is a part of IBM Watson, which automates the AI lifecycle management. With the help of AutoAI, the dataset is loaded, the required parameters are set, ratio of splitting the data for training and testing is set and algorithms are chosen. AutoAI reads the data, pre-process it and choose the top n algorithms, where n is user-defined. n pipelines are created and models are created and improvised with the help of automated hyper parameter optimisation and feature engineering. The models are listed below with their evaluation metrics. The best model is saved and it is deployed. Under the implementation tab, it automatically creates the scoring endpoint and it can also be tested under test tab.

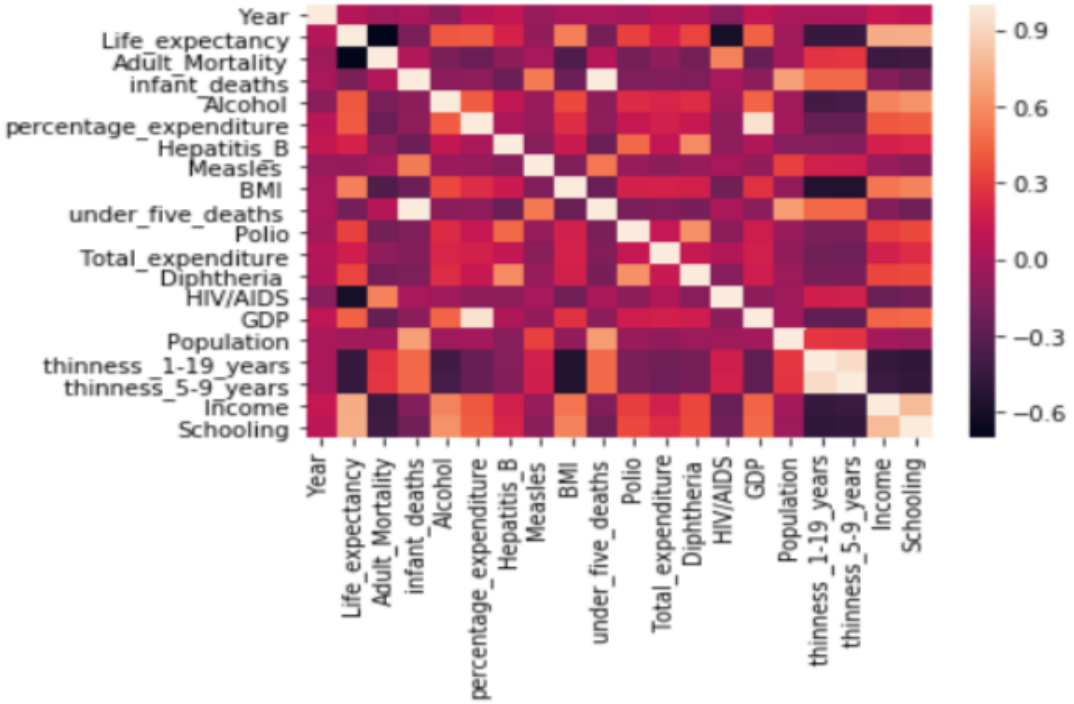
3.2.5 Node-RED flow

A Node-RED app is created, through which the scoring endpoint of the machine learning model is to be linked. Node-RED dashboard is installed to the node palette, because Node-RED makes the process of creating an end UI much easier. A Node-RED flow is created, within which a form is created to get the data and a label is created to show the predicted life expectancy. Two tabs were created, one for ML model created using notebook and the other by AutoAI. And the inputs are declared as variables and the values of these variables are passed to the ML model with the help of scoring endpoint.

4. EXPERIMENTAL INVESTIGATION

4.1 Data visualisation

After loading and splitting the dataset, data visualisation is an important process, which helps to choose the best parameters from the dataset.



4.2 Random Forest Regressor

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging. Bagging, in the Random Forest method, involves training each decision tree on a different data sample where sampling is done with replacement.

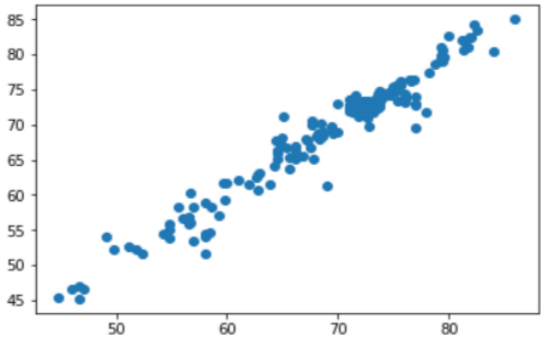
The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. The proposed solution uses Random forest regressor for training the model.

4.3 Evaluation metrics

R-squared is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted.

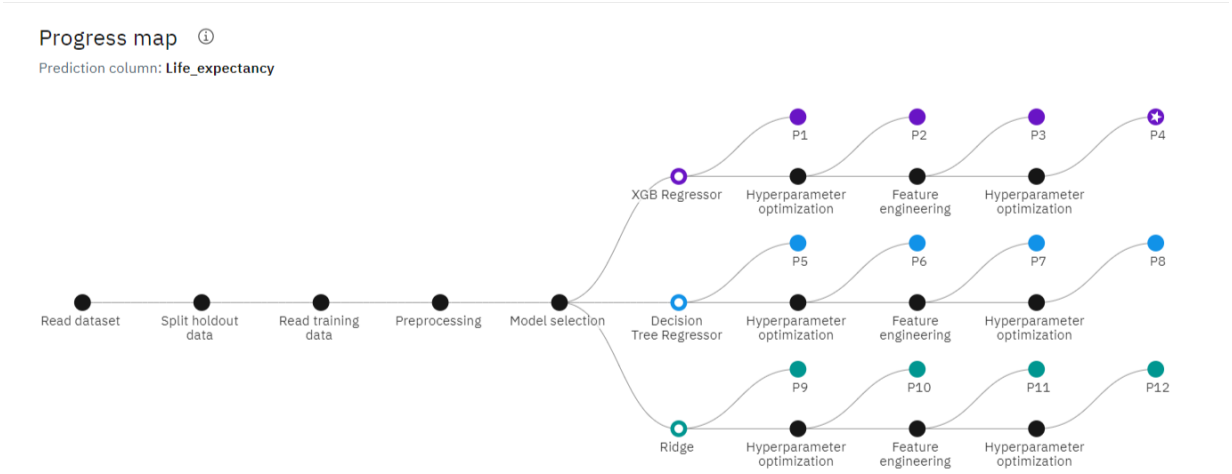
For the proposed solution, R- squared value for the trained random forest regressor is **0.95**.

```
In [17]: from sklearn.metrics import r2_score
         print("Mean absolute error: %.2f" % np.mean(np.absolute(test_labels - predictions)))
         print("Residual sum of squares (MSE): %.2f" % np.mean((test_labels - predictions) ** 2))
         print("R2-score: %.2f" % r2_score(test_labels ,predictions) )

         Mean absolute error: 9.63
         Residual sum of squares (MSE): 153.81
         R2-score: 0.95

In [18]: plt.scatter(test_labels,predictions)

Out[18]: <matplotlib.collections.PathCollection at 0x7f3d34221160>
```



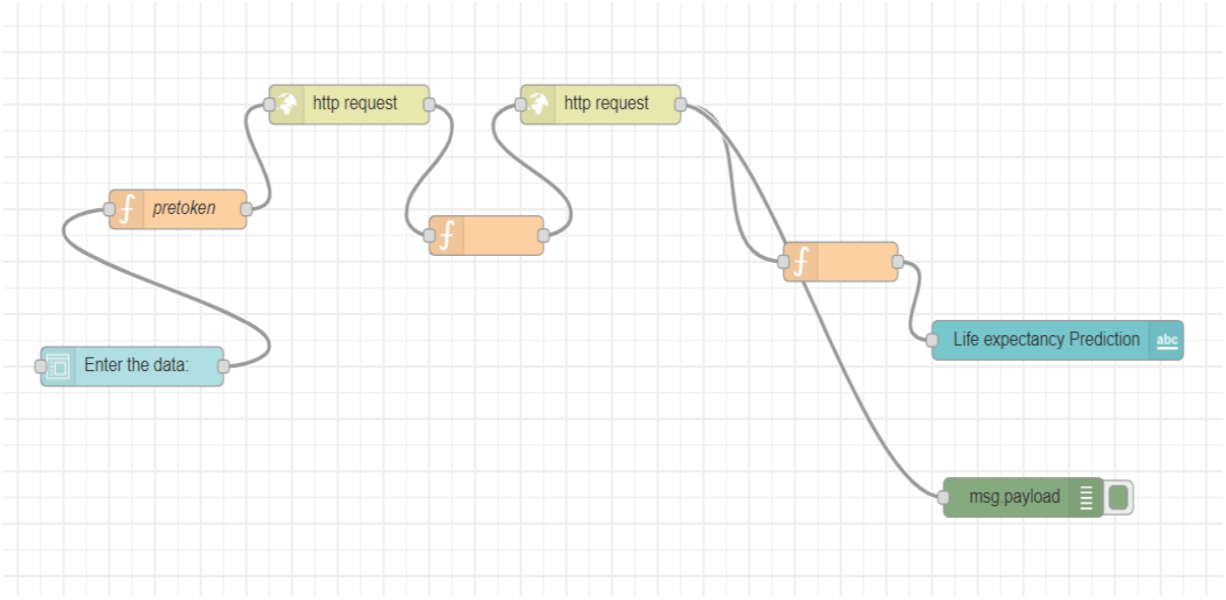
4.4 AutoAI experiment

After loading the dataset, the columns(parameters) are chosen for the AutoAI experiment and the split holdout data ratio is set. For the given dataset, 3 pipelines are chosen and accordingly AutoAI chose the top 3 algorithms and list the models based on the evaluation metrics.



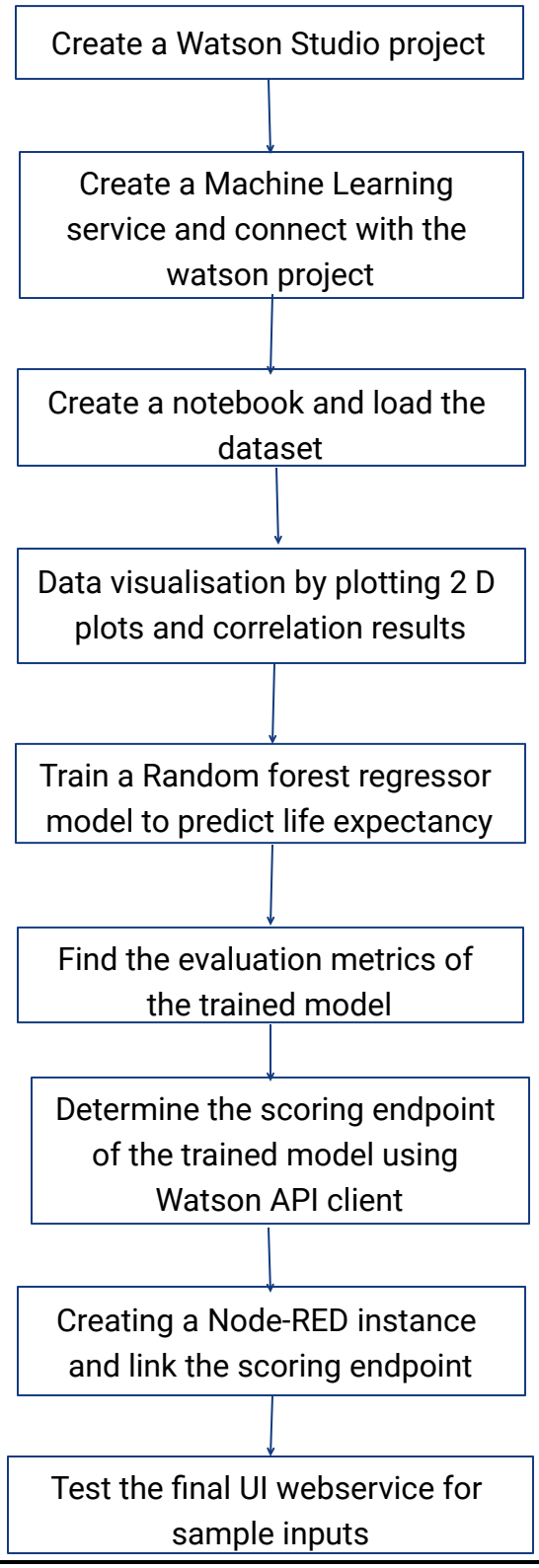
4.5 Node-RED flow

Node-RED is a programming tool for wiring together hardware devices, APIs and online services. Primarily, it is a visual tool designed for the Internet of Things, but it can also be used for other applications to very quickly assemble flows of various services. With the help of Node-RED and the scoring endpoint of the trained models, Watson ML is linked with a webservice to predict life expectancy.

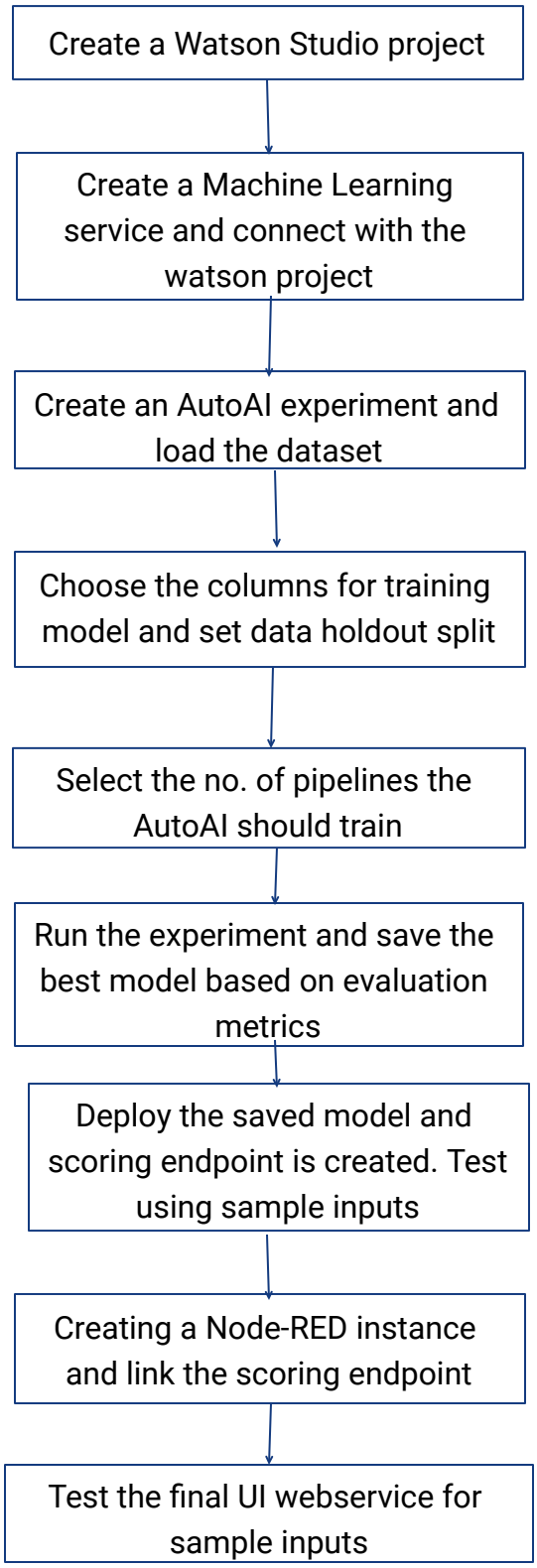


5. FLOW CHART

5.1 Flow chart for ML model with Python

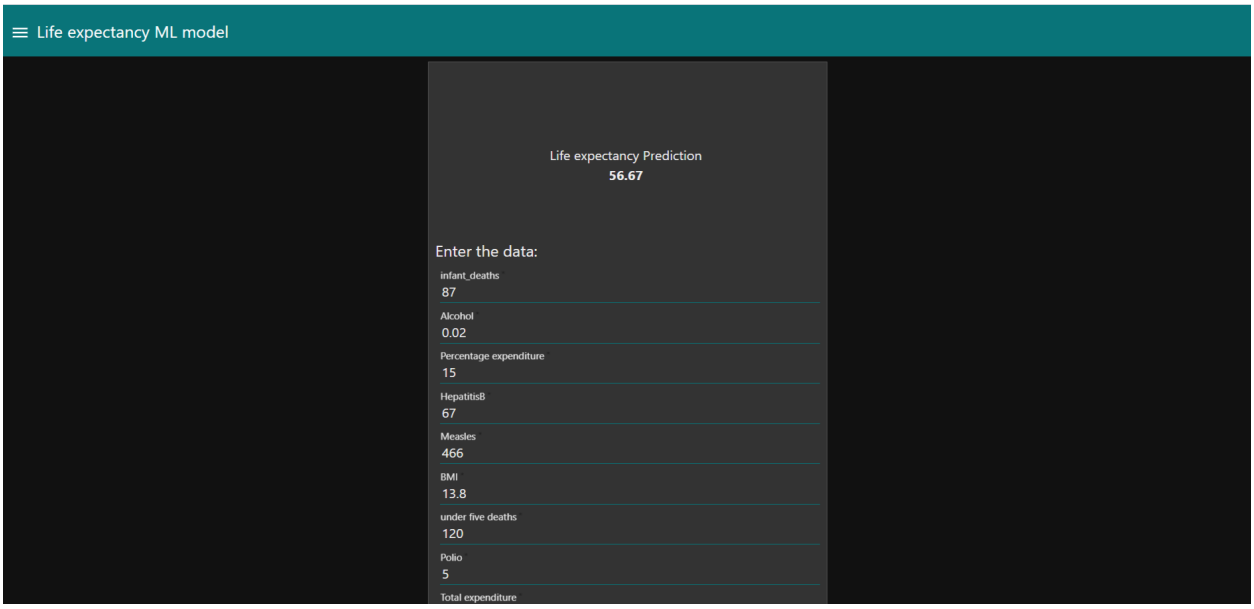


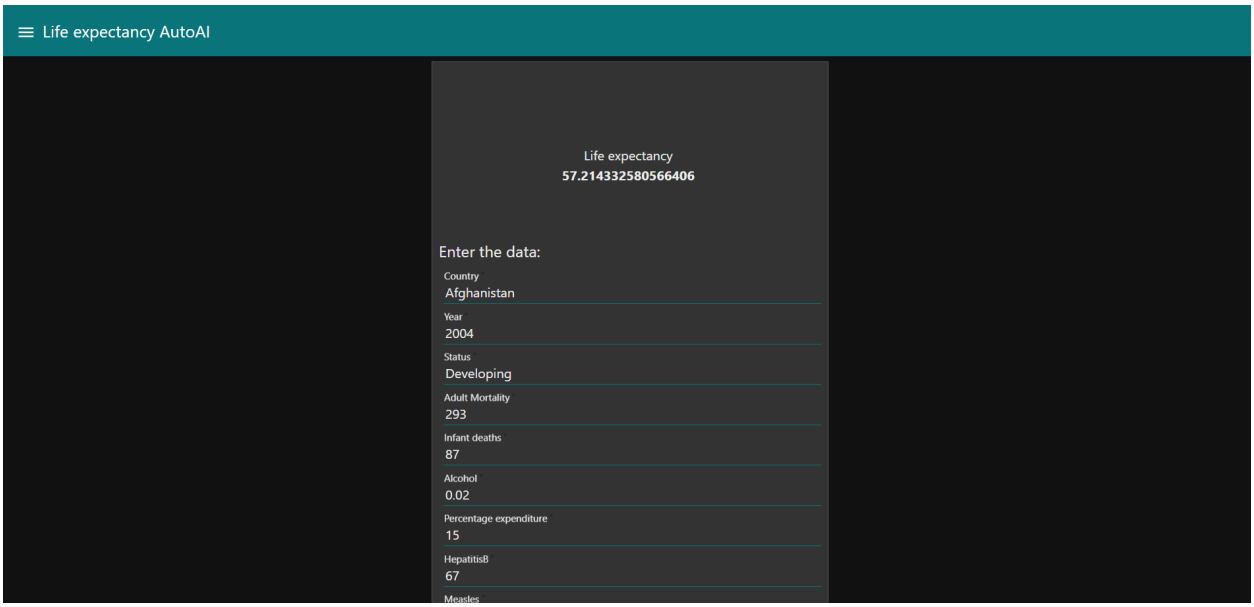
5.2 Flow chart for AutoAI experiment



6. RESULT

The final deliverable is a web service which takes inputs from the user and displays the predicted life expectancy. It consists of two tabs - one by ML model trained using Python notebook and the other by AutoAI experiment.





7. ADVANTAGES & DISADVANTAGES

Advantages

The proposed solution analyses the historical data of various countries and their life expectancies and a high accuracy model is trained which predicts life expectancy. The factors are chosen in such a way that it has a maximum impact on life expectancy and others are avoided.

Disadvantages

Though the proposed solution predicts the life expectancy with its maximum possible accuracy, the predictions have a little high difference at times than actual values. And the ML model has to be trained with much more factors impacting life expectancy to improve the accuracy.

8. APPLICATIONS

Predicting life expectancy is an important thing which has a huge impact on various fields. Surveys show that a country's GDP changes due to change in life expectancy. It also helps insurance companies a lot. Predicting life expectancy helps countries in framing better health policies.

9. CONCLUSION

Thus, this project provides an efficient Machine Learning model to predict life expectancy with high accuracy based on the input data using IBM Watson studio and a final web user interface is created with the help of connecting the trained model with Node-RED.

10. FUTURE SCOPE

With a fast changing life style, a lot of factors has to be taken into consideration and analysis. With varying trends and factors, Machine Learning models has to be tuned better for a better efficiency of predicting life expectancy.

11. BIBLIOGRAPHY

1. https://developer.ibm.com/clouddataservices/docs/ibm-watson-machine-learning/
2. https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-overview.html
3. https://www.bbc.com/news/health-23411975
4. https://ourworldindata.org/life-expectancy
5. https://medium.com/datadriveninvestor/random-forest-regression-9871bc9a25eb

APPENDIX

A. Source code

Python notebook code

```
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import numpy as np
%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_ff852431dd6b466db35b162efd0e1ec7 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='8okGPMWk_YnLEq8VNWg00vEJYjEo06hOYBJkGGiHHAgO',
    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_ff852431dd6b466db35b162efd0e1ec7.get_object(Bucket='lifeexpectancypredictionwithpytho-
donotdelete-pr-zalnsxfnzezjn5',Key='datasets_Life Expectancy Data.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

# If you are reading an Excel file into a pandas DataFrame, replace `read_csv` by `read_excel` in the
next statement.
df_data_0 = pd.read_csv(body)
df_data_0.head()
labels=df_data_0[['Life_expectancy']]
df_data_0=df_data_0.drop('Country',axis=1)
df_data_0=df_data_0.drop('Status',axis=1)
df_data_0=df_data_0.drop('Year',axis=1)
```

```
df_data_0=df_data_0.drop('Adult_Mortality',axis=1)
df_data_0=df_data_0.drop('Life_expectancy',axis=1)
from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
train_features, test_features, train_labels, test_labels = train_test_split(df_data_0, labels, test_size =
0.10, random_state = 10)
from sklearn.ensemble import RandomForestRegressor
# Instantiate model with 1000 decision trees
rf = RandomForestRegressor(n_estimators = 10, random_state = 10)
# Train the model on training data
rf.fit(train_features, train_labels);
predictions = rf.predict(test_features)
test_labels = np.asanyarray(test_labels[['Life_expectancy']])
from sklearn.metrics import r2_score
print("Mean absolute error: %.2f" % np.mean(np.absolute(test_labels - predictions)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_labels - predictions) ** 2))
print("R2-score: %.2f" % r2_score(test_labels ,predictions) )
plt.scatter(test_labels,predictions)
!pip install watson_machine_learning_client
from watson_machine_learning_client import WatsonMachineLearningAPIClient
wml_credentials={
  "apikey": "meIN7M1xd7J3RgfKyqVGdwzNIXI-U85RdQEg51gDjTId",
  "instance_id": "000a2024-2090-4c67-a493-e634fa599251",
  "url": "https://eu-gb.ml.cloud.ibm.com"
}
client=WatsonMachineLearningAPIClient(wml_credentials)
model_props={client.repository.ModelMetaNames.AUTHOR_NAME: "Swami",
        client.repository.ModelMetaNames.AUTHOR_EMAIL: "swaminathan.m0908@pec.edu",
        client.repository.ModelMetaNames.NAME: "Life expectancy"}
model_artifact=client.repository.store_model(model=rf,meta_props=model_props)
published_model_uid=client.repository.get_model_uid=(model_artifact)
deployment=client.deployments.create('a89ab479-2b6d-424b-94d3-3afec813d3ac',name="Life
expectancy")
scoring_endpoint=client.deployments.get_scoring_url(deployment)
```

Github link for node red flow and dataset

https://github.com/SmartPracticeschool/llSPS-INT-2181-Predicting-Life-Expectancy-using-Machine-Learning