Remote Summer Internship Program 2020
Machine Learning, Career Basic Program
Smartinternz, SmartBridge

Predicting Life Expectancy using Machine Learning

Internship Report
by
Vinny Chamoli

15/5/2020-15/6/2020

Contents

1. Introduction

Summer Internship Program by Smartbridge is an annual initiative taken up by them to teach and prepare students across the globe for industry experience. They believe that experiential learning and development in a professional like environment can only bridge the gap between students and industries opening ways for both of them to achieve better results. This initiative enables students to better their resume to embark upon a successful industrial journey.

They provide various roles for internship according to the possible aptitude of the students such as Artificial Intelligence, Machine Learning, Internet of Things etc.A project is assigned to students individually with access to the platform wherein the students code and deploy their model just like in industries. This not only introduces them to environments like IBM cloud but also helps them in understanding the industrial environment better.

Classes are organised according to the technologies to be taught and doubt sessions are taken up by the mentors to help students complete their projects too.
My role at the internship was of a Machine Learning Engineer and the project given to me was to generate a Regression based Machine Learning model.

2. Literature Survey

2.1 Existing Problem

[1]This paper was written to provide a cross-sectional model of life expectancy, using a comprehensive worldwide sample, which analyses the impact of country level variables on average life expectancy.The model presented here suggests robustly that proxies for technology, education, disposable income and healthcare all have a significant and positive effect on country variation in average life expectancy, at all income levels. A proxy for the health risks/epidemics factor is significantly negative. Such a model provides information to government, particularly in the developing world, since average life expectancy is predicted with high explanatory power by variables that can be influenced through public policy. Indeed, it is seen that quite low-cost policy interventions can have dramatic impacts on life expectancy, in addition to other benefits of those interventions.
In [2], a Bayesian hierarchical model for producing probabilistic forecasts of male period life expectancy at birth for all the countries of the world to 2100 has been proposed.To evaluate the method, a survey was conducted to do an out-of-sample cross-validation experiment, fitting the model to the data from 1950–1995 and using the estimated model to forecast for the subsequent 10 years. The 10-year predictions had a mean absolute error of about 1 year, about 40 percent less than the current UN methodology. The probabilistic forecasts were calibrated in the sense that, for example, the 80 percent prediction intervals contained the truth about 80 percent of the time. Illustration of the model was done with the results from Madagascar (a typical country with steadily improving life expectancy), Latvia (a country that has had a mortality crisis), and Japan (a leading country). Aggregated results for South Asia, a region with eight countries was also depicted.

2.2 Predicting Life Expectancy using Machine Learning - Proposed Solution

'Life Expectancy' refers to the number of years a person can be expected to live which is governed by a number of factors ranging from demographic to genetic to habits.

This project involved building a machine learning (ML) model to predict the life expectancy rate of a country given various features like year, GDP, education, alcohol intake of people in the country, expenditure on healthcare system and some specific disease related deaths that have happened.

The developed model works on the data set provided by the World Health Organization(WHO) to evaluate the life expectancy in different years of a country. The time frame offered in the data set is from the year 2000 to 2015. Regression based algorithm that has been used to predict the life expectancy for the data : - Random Forest Regression

The model that involves Python to code these regression techniques uses the Jupyter Notebook in IBM Watson Studio to import data and automate the ML model otherwise the IBM Watson Studio services are used to auto AI the experiment. A Node - RED flow is built to integrate the ML services or Auto AI.

3. Theoretical Analysis

Methodology used

work flow for this project can be divided into three sub-tasks.These include acquiring the data and understanding various features of the data, preprocessing the data set to align it with our requirements and remove any inconsistency, and finally analyzing the data using regression based prediction algorithm with the key performance index being accuracy of prediction.

Figure 3.1: Workflow of the project

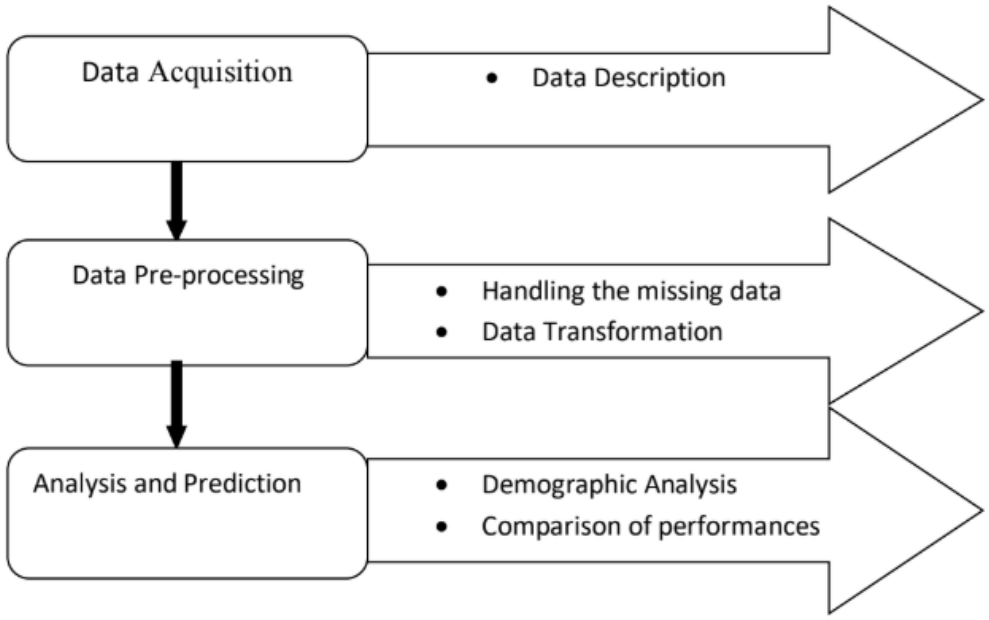


Fig 1. Work flow of the project

3.1 Machine Learning Model

3.1.1 Algorithm Formulated to solve the given Problem Statement

*A*lgorithm steps:
Step 1: Import the Data set
Step 2: Read and Understand the data
Step3: Explore the Data set
Step4: Decide the amount of data for training data and testing data
Step5: Give 70 percent data for training and remaining data for testing.
Step6: Assign train data set to the models
Step7: Choose the algorithm and create the model
Step8: Make predictions for test data set.
Step9: Calculate accuracy for the algorithm
Step 10: Apply the model for further predictions.

3.2 Setting Up the Environment - Software Designing

An IBM cloud account was set up to access various services to create and deploy the model.

The following services have been used in the project:

1. Watson Studio - This is where the notebook has been created in a project to write the regression code along with the data set.

2. Node Red - Node Red is the front end application that uses interconnecting nodes to interact with machine learning services of the cloud and the model to show predictions when inputs are given

4. Experimental Investigations

4.1 Data Acquisition

The Global Health Observatory (GHO) data repository under World Health Organization (WHO) keeps track of the health status as well as many other related factors for all countries The datasets are made available to public for the purpose of health data analysis. The dataset related to life expectancy, health factors for 193 countries has been collected from the same WHO data repository website and its corresponding economic data was collected from United Nation website. Among all categories of health-related factors only those critical factors were chosen which are more representative.

4.2 Model Requirements

4.2.1 Python

Python is a multi-paradigm, general purpose, high level programming language, which focuses on code readability. It has a large library, which provides tools for many tasks and has a wide support base. This project uses python 3.5.

4.2.2 Python Libraries

4.2.2.1 Pandas

Pandas is used for data manipulation and analysis through operations and data structures on numerical tables and time series.

4.2.2.2 Numpy

It adds support as well as contains high-level mathematical functions to operate on large multidimensional arrays and matrices.

4.2.2.3 Matplotlib

It is a plotting library that that enables 2d diagramming and begetting of bar charts, histograms and so forth.

4.2.2.4 Sci-kit learn

It is a free software machine learning library that features various regression, clustering and classification algorithms. It works in conjunction with numPy and python scientific library sciPy.

4.3 Data Preprocessing

Data preprocessing is an essential step in order to increase the accuracy of machine learning models. It involves handling inaccurate and missing data, noisy data in the form of outliers, and inconsistent data in the form of duplication and others.

4.4 Data Cleaning

Data was often not consistent; missing values or values out of range was common. The methods used for cleaning is to replace the missing or noisy values by forward filling them using mean of the feature.

4.5 Analysis and Prediction

Random Forest algorithm has been applied to the data set to train the model and increase the accuracy for prediction of the the life expectancy of any given country.

4.5.1 Data Transformation

4.5.1.1 Scaling

Scaling is required to standardize the independent feature in the dataset to a fixed range. Primarily, two types of feature scaling methods:

1. Min-max scaling (Normalization) (value - min)/(max - min) Sklearn provides a class called MinMaxScaler for this

2. Standardization (value - mean)/std Sklearn provides a class called StandardScaler for this

4.5.2 Evaluation Metrics

Evaluation metrics calculated are:

1. Mean Cross validation score

2. Score without cv

3. $R^2 score$

4.5.3 Random Forest Regression

4.5.3.1 Introduction

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The

individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

4.5.4 Flowchart

It works in four steps:

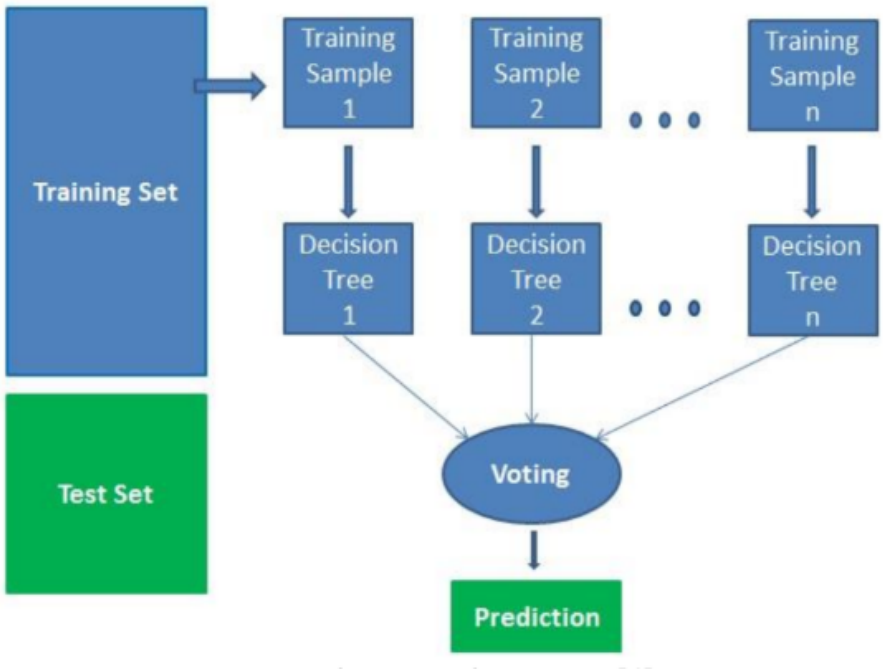
1) Select random samples from a given dataset.

2) Construct a decision tree for each sample and get a prediction result from each decision tree.

3) Perform a vote for each predicted result.

4) Select the prediction result with the most votes as the final prediction.

Figure 4.1: Workflow of the Random Forest Regression

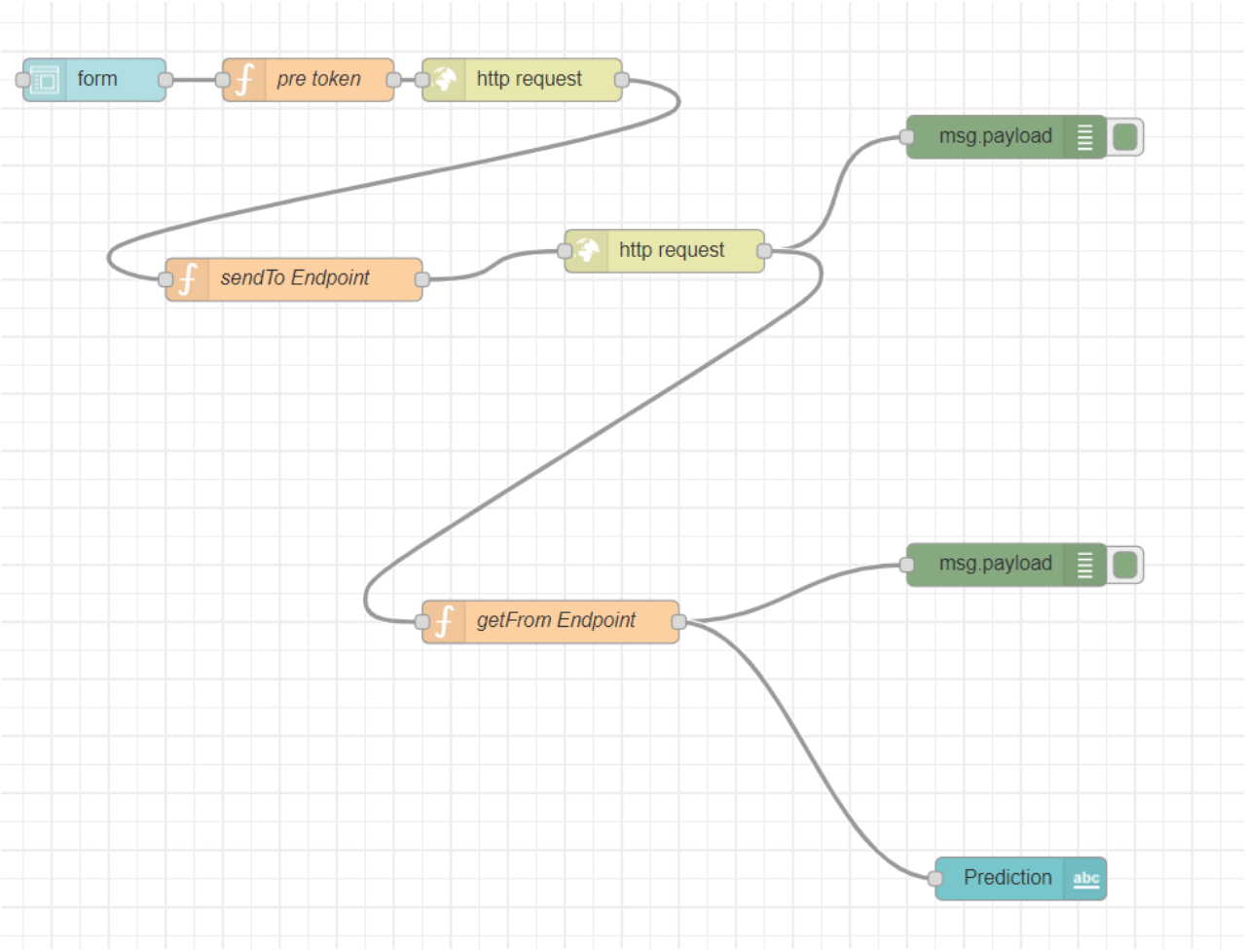


5. Results

5.0.1 Node Red Flow

A Node RED starter application was created to implement the front end of the project. In the starter application, nodes are dragged and dropped to create a flow to integrate the application with the machine learning model.

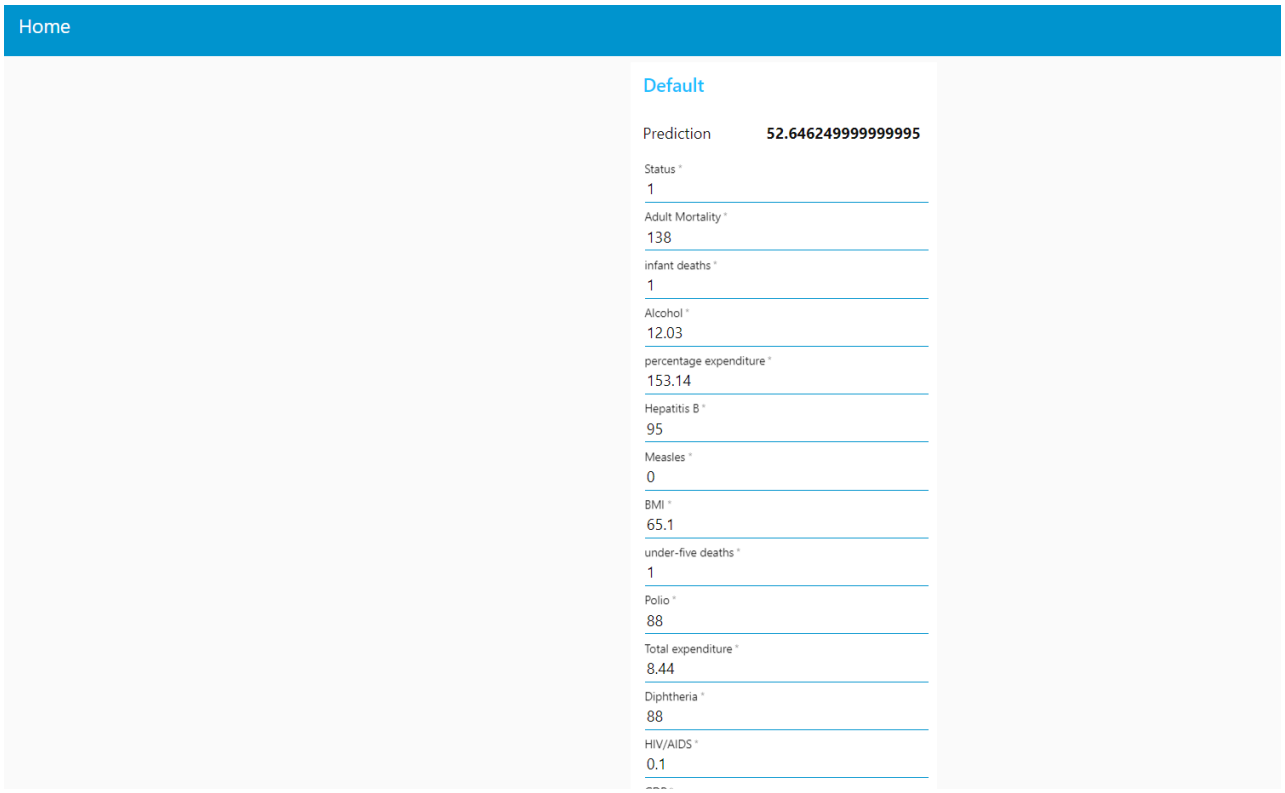
Figure 5.1: Node-RED flow



The following form appears after deployment of the app wherein the user can input values and life expectancy

prediction is displayed according to the inputs.

Figure 5.2: Form of the ML Model



6. Advantages and Disadvantages

Supervised Machine Learning helps in training the system through historical data making it learn from the past so that it can accurately analyze the future to a degree. A data set is divided into training and testing data set and a function is written to predict outputs based on the inputs of the data set.

6.1 Advantages

1. No human interference is required
2. Continuous Improvement in the modelling technique
3. Managing multi-dimensional large data set to train the model

6.2 Disadvantages

1. Acquiring data is time consuming and tedious.
2. Selection of the right algorithm to train the data for accurate results is another tedious process
3. High error susceptibility

7. Application

Prediction of life expectancy of any country according to various factors affecting it such as adult mortality, alcohol intake, GDP, Health sector facilities etc can help the government of that country to analyze the factors that can help in improving longevity of its citizens and how to as well as how much to invest in those factors.

A model that can facilitate this by managing large multi dimensional data and applying a function to accurately predict life expectancy can be deployed for government institutions. This will help them in not only analyzing the factors that influence the quality of life of their citizens but will also be able to bring about a significant change in economic factors like GDP.

8. Conclusion

In this project, I used machine learning algorithms to predict the life expectancy of a given country.I mentioned the steps to analyse the data set and how to handle missing attributes. These feature set were then given as an input to the model and a csv file was generated consisting of predicted life expectancy.
I found that Random forest regression fits the data set and gives the least error. Thus concluding that we can use random forest regression model to predict the life expectancy accurately to a certain degree.

Evaluation Metrics results:
1. R square on the test data of 93 percent.
2. MAE of 1.66
3. MSE of 6.05

9. Overview of Internship Experience

*D*uring my internship experience with Smart Internz, I was able to develop my Machine Learning skills. It was an enriching experience as I got to work in a professional like environment. The mentors were very helpful with the webinars they conducted on how to proceed with the project.Our doubts were solved on the slack channel regularly.

I particularly found the IBM cloud experience new and useful in improving my industrial skills. Although I found the Node RED service quite challenging, I found it to be valuable in developing my front end integration skills.

This internship has given me a clearer idea on how to proceed with improving my machine learning skills and I am grateful to Smart Internz platform for letting me be a part of this initiative.

Bibliography

[1] Audrey Hendricks and Philip E Graves. "Predicting life expectancy: A cross-country empirical analysis". In: *Available at SSRN 1477594* (2009).

[2] Adrian E Raftery et al. "Bayesian probabilistic projections of life expectancy for all countries". In: *Demography* 50.3 (2013), pp. 777–801.

10. Appendix

Source Code

```
cell 1:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


cell 2:
def __iter__(self): return 0
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0


LifeData = pd.read_csv(body)
LifeData.head()


cell 3:

LifeData.describe()


cell 4:
LifeData.columns


cell 5:
LifeData.info()


cell 6:
sns.set(rc={"figure.figsize": (8, 4)}); np.random.seed(0)
x = np.random.randn(100)
```

```
ax = sns.distplot(x)
plt.show()


cell 7:
sns.heatmap(LifeData.corr())


cell 8:
LifeData=LifeData.drop("Year",axis=1)
LifeData["status"] = pd.get_dummies(LifeData["Status"], drop_first = True)
print(LifeData["status"])


cell 9:
LifeData = LifeData.groupby('Country').mean()


cell 10:
LifeLabels = LifeData['Life expectancy ']
LifeFeatures = LifeData.drop('Life expectancy ', axis = 1)


cell 11:
LifeFeatures.isnull().head()
LifeFeatures.isnull().sum()
LifeLabels.isnull().sum()
LifeFeatures.fillna(value = LifeFeatures.mean(), inplace = True)
LifeLabels.fillna(value = LifeLabels.mean(), inplace = True)

from scipy import stats
stats.describe(LifeFeatures[1:])


cell 12:
from sklearn.preprocessing import MinMaxScaler
min_max_scaler = MinMaxScaler()
LifeFeatures = min_max_scaler.fit_transform(LifeFeatures)


cell 13:
LifeFeatures


cell 14:
from sklearn.model_selection import train_test_split
LifeFeatures_train, LifeFeatures_test, LifeLabels_train, LifeLabels_test = train_test_split(
        LifeFeatures, LifeLabels, train_size = 0.7, test_size = 0.3)
```

```
cell 15:
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
random_forest_model = RandomForestRegressor()
random_forest_fit = random_forest_model.fit(LifeFeatures_train, LifeLabels_train)

random_forest_score = cross_val_score(random_forest_fit, LifeFeatures_train, LifeLabels_trai
print("mean cross validation score: %.2f"
      % np.mean(random_forest_score))
print("score without cv: %.2f"
     % random_forest_fit.score(LifeFeatures_train, LifeLabels_train))
print("R^2 score on the test data %.2f"
     %r2_score(LifeLabels_test, random_forest_fit.predict(LifeFeatures_test)))


cell 16:
random_forest_model_predict = random_forest_model.predict(LifeFeatures_test)


cell 17:
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
scoring = make_scorer(r2_score)
grid_cv = GridSearchCV(RandomForestRegressor(),
              param_grid={'min_samples_split': range(2, 10)},
              scoring=scoring, cv=5, refit=True)
grid_cv.fit(LifeFeatures_train, LifeLabels_train)
grid_cv.best_params_

result = grid_cv.cv_results_
print("Best Parameters: " + str(grid_cv.best_params_))
result = grid_cv.cv_results_
print("R^2 score on training data: %.2f"  % grid_cv.best_estimator_.score(LifeFeatures_train
print("R^2 score: %.2f"
     % r2_score(LifeLabels_test, grid_cv.best_estimator_.predict(LifeFeatures_test)))
print("Mean squared error: %.2f"
     % mean_squared_error(LifeLabels_test, random_forest_model_predict))
print("Mean absolute error: %.2f"
     % mean_absolute_error(LifeLabels_test, random_forest_model_predict))

\newline
cell 18:
random_forest_model.score(LifeFeatures,LifeLabels)


cell 19:
```

```
y=random_forest_model.predict(LifeFeatures_test)

df=pd.DataFrame({'Actual':LifeLabels_test, 'Predicted':y})

df1=df.head(25)

print(df1)


cell 20:
df1.plot(kind='bar',figsize=(16,10))
plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')
plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
plt.show()


cell 21:
x=pd.DataFrame({'Status':[1],'Adult␣Mortality':[138], 'infant␣deaths':[1], 'Alcohol':[12.03]
        'percentage␣expenditure':[153.14], 'Hepatitis␣B':[95], 'Measles␣':[0], '␣BMI␣':[65.1]
        'under-five␣deaths␣':[1], 'Polio':[88], 'Total␣expenditure':[8.44], 'Diphtheria␣':[88
        '␣HIV/AIDS':[0.1], 'GDP':[7853.335], 'Population':[7223938], '␣thinness␣␣1-19␣years'
        '␣thinness␣5-9␣years':[1.9], 'Income␣composition␣of␣resources':[0.787], 'Schooling':[
prediction=random_forest_model.predict(x)
print(prediction)


cell 21:
scoring_endpoint = client.deployments.get_scoring_url(deployment)


cell 22:
scoring_endpoint
```