**PREDICTING LIFE EXPECTANCY USING MACHINE LEARNING**

**A PROJECT REPORT**

submitted as a course project for

**MACHINE LEARNING INTERNSHIP**

**under SMARTBRIDGE**

Submitted by

Atharava Rakhe (atharvarakhe@gmail.com)

Under the Guidance of

V. Prashanth

M. Charan Reddy

L Lalita Gayatri

Contents: -

|  |  |  |
| --- | --- | --- |
| ***SL.NO*** | ***TITLE*** | ***PAGE NO.*** |
| 1 | **INTRODUCTION**   * 1. Overview   2. Purpose | 3 |
| 2 | **LITERATURE SURVEY**  2.1 Existing problem  2.2 Proposed solution | 5 |
| 3 | **THEORITICAL ANALYSIS**  3.1 Block diagram  3.2 Hardware / Software designing | 7 |
| 4 | **EXPERIMENTAL INVESTIGATIONS** | 14 |
| 5 | **FLOWCHART** | 21 |
| 6 | **RESULT** | 22 |
| 7 | **ADVANTAGES & DISADVANTAGES** | 24 |
| 8 | **APPLICATIONS** | 25 |
| 9 | **CONCLUSION** | 25 |
| 10 | **FUTURE SCOPE** | 26 |
| 11 | **BIBILOGRAPHY** | 26 |
| 12 | **APPENDIX**  (Source code) | 27 |

**1.INTRODUCTION**

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. A typical Regression **Machine Learning** project leverages historical data to predict insights into the future. This problem statement is aimed at predicting ***Life Expectancy rate*** of a country given various features.

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

*Where do we get data to predict the life expectancy?*

The Global Health Observatory (GHO) data repository under the World Health Organization (WHO) keeps track of the health status as well as many other related factors for all countries the data-sets are made available to the public for the purpose of health data analysis. The data-set related to life expectancy, health factors for 193 countries have been collected from the same 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

Dataset Link: - *https://www.kaggle.com/kumarajarshi/life-expectancy-who*

* 1. Overview: -

In mathematical terms, life expectancy refers to the expected number of years remaining for an individual at any given age.

The life expectancy for a particular person or population group depends on several variables such as their lifestyle, access to healthcare, diet, economic status and the relevant mortality and morbidity data. However, as life expectancy is calculated based on averages, a person may live for many years more or less than expected. Understanding potential trajectories in health and drivers of health is crucial to guiding long-term investments and policy implementation. Past work on forecasting has provided an incomplete landscape of future health scenarios, highlighting a need for a more robust modelling platform from which policy options and potential health trajectories can be assessed.

In order to predict life expectancy rate of a given country, we will be using Machine Learning algorithms to draw inferences from the given dataset and give an output. For better usability by the customer, we are also going to be creating a UI for the user to interact with using Node-Red.

* 1. Purpose: -

Good prognostication 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 in a country. 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. However, Physicians tend to overestimate life expectancy, and miss the window of opportunity to initiate Advance Care Planning

***Economic growth: -***

Predicting life expectancy would play a vital role in judging the growth and development of the economy.

Across countries, high life expectancy is associated with high income per capita. Increase in life expectancy also leads to an increase in the “manpower” of a country. The knowledge asset of a country increases with the number of individuals in a country.

***Population Growth: -***

Helps the government bodies take appropriate measures to control the population growth and also direct the utilization of the increase in human resources and skillset acquired by people over many years.

***Growth in social activities: -***

Based on the factors used to calculate life expectancy of an individual and the outcome, health care will be able to fund and provide better services to those with greater need.

Some of the Insurance Companies will be able to provide individualized services to people based on the life expectancy outcomes and factors.

**2. LITERATURE SURVEY**

2.1 Existing Problem: -

As a result of the evolution of biotechnologies and related technologies such as the development of sophisticated medical equipment, humans are able to enjoy longer life expectancies than previously before. Predicting a human’s life expectancy has been a long-term question to humankind. Many calculations and research have been done to create an equation despite it being impractical to simplify these variables into one equation.

Currently there are various smart devices and applications such as smartphone apps and wearable devices that provide wellness and fitness tracking. Some apps provide health related data such as sleep monitoring, heart rate measuring, and calorie expenditure collected and processed by the devices and servers in the cloud. However no existing works provide the Personalized Life expectancy.

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 timeframe from 2000 to 2015. The project relies on accuracy of data. 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 data-sets are made available to public for the purpose of health data analysis.

The data-set 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. 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. On initial visual inspection of the data showed some missing values. As the data-sets were from WHO, we found no evident errors. Missing data was handled by using Python software. The result indicated that most of the missing data was for population, Hepatitis B and GDP. The missing data were from less known countries like Vanuatu, Tonga, Togo, Cabo Verde etc. Finding all data for these countries was difficult and hence, it was decided that we exclude these countries from the final model data-set. The final merged file (final dataset) consists of 22 Columns and 2938 rows which meant 20 predicting variables. All predicting variables was then divided into several broad categories: ​Immunization related factors, Mortality factors, Economical factors and Social factors.

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. Two algorithms have been used:

1)*Decision Tree Regression*

2)*Random Forest Regression*

**3. THEORITICAL ANALYSIS**

3.1 Block Diagram: -

User Interface

Machine Learning Model

Taking Input and Display’s output

Exchange of input and predicted output

Store the model

Node-RED Application

Deploy model

Watson Studio

Notebook

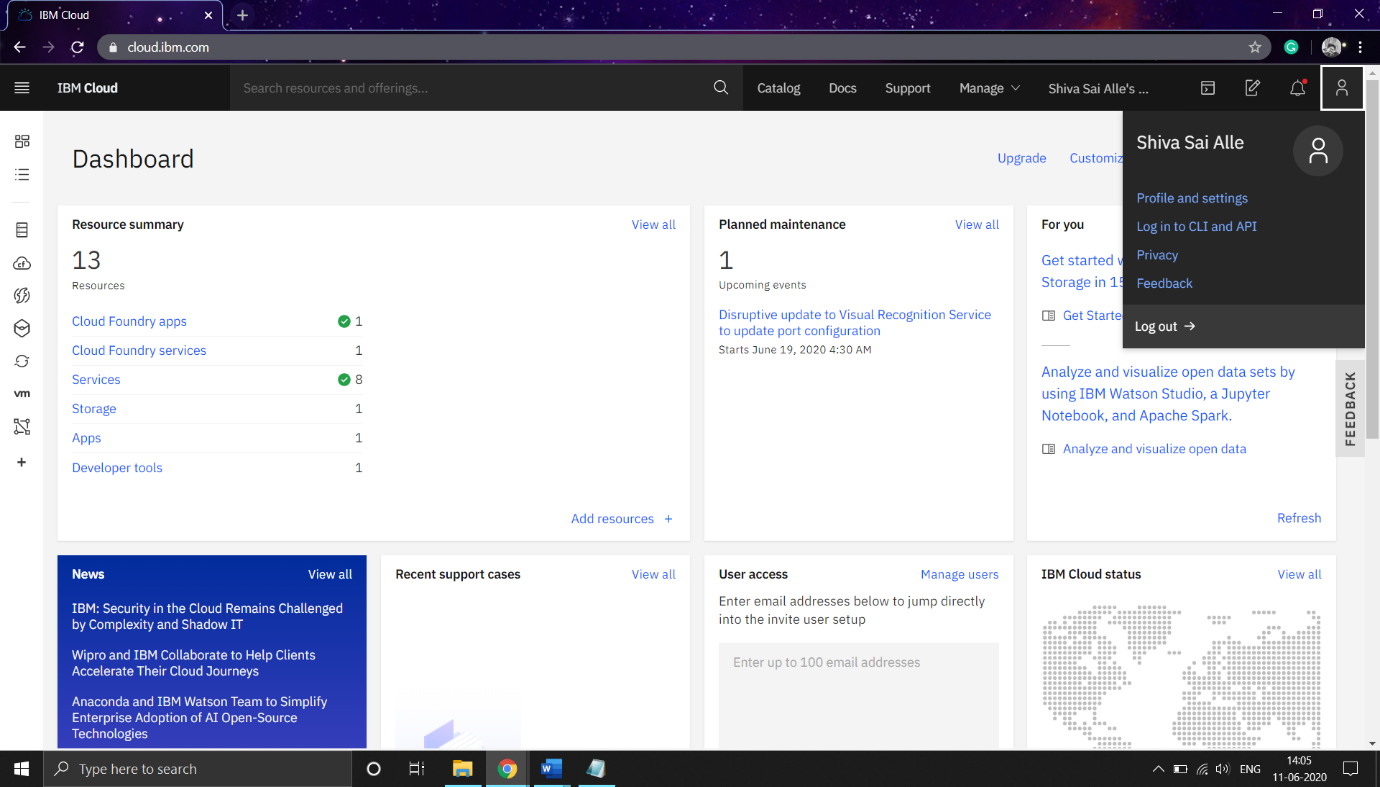
ML Web Service

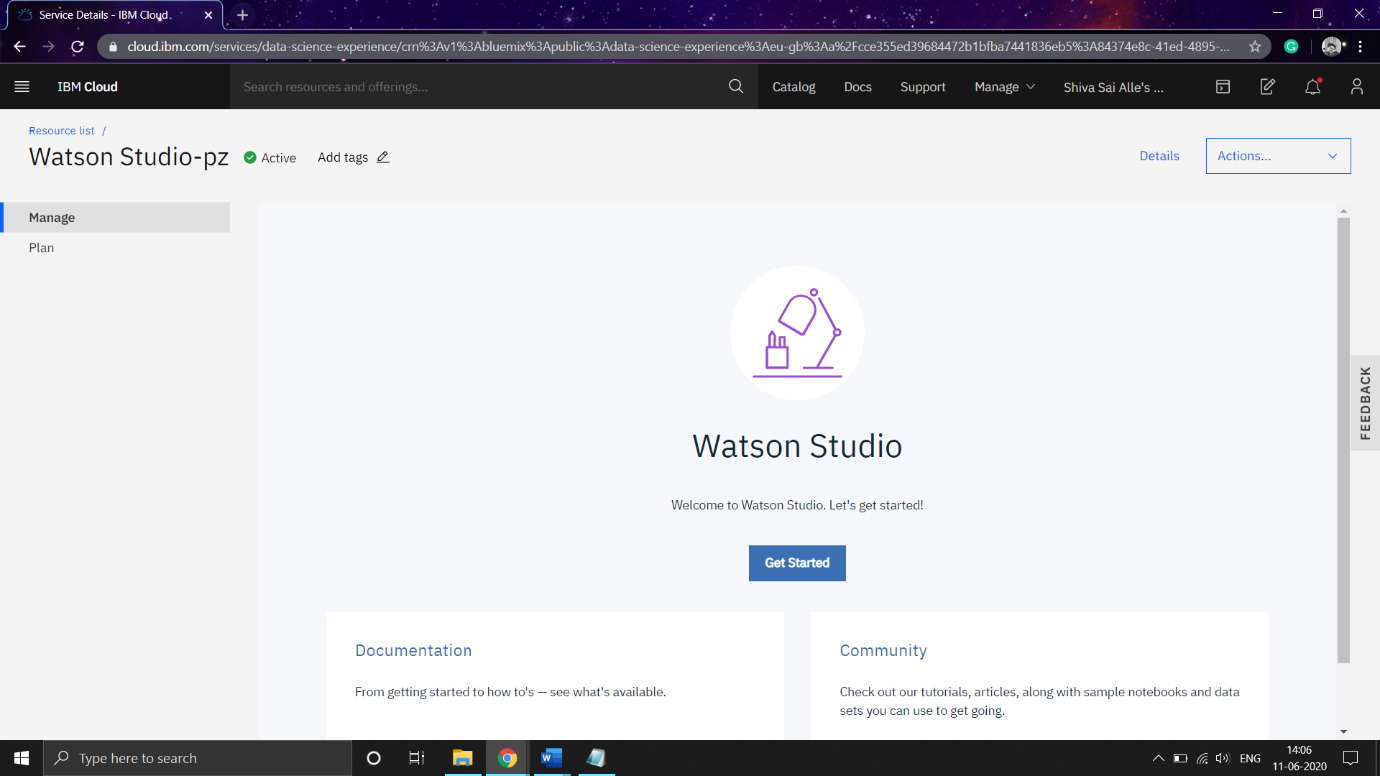
3.2 Hardware / Software designing: -

***Software Designing: -***

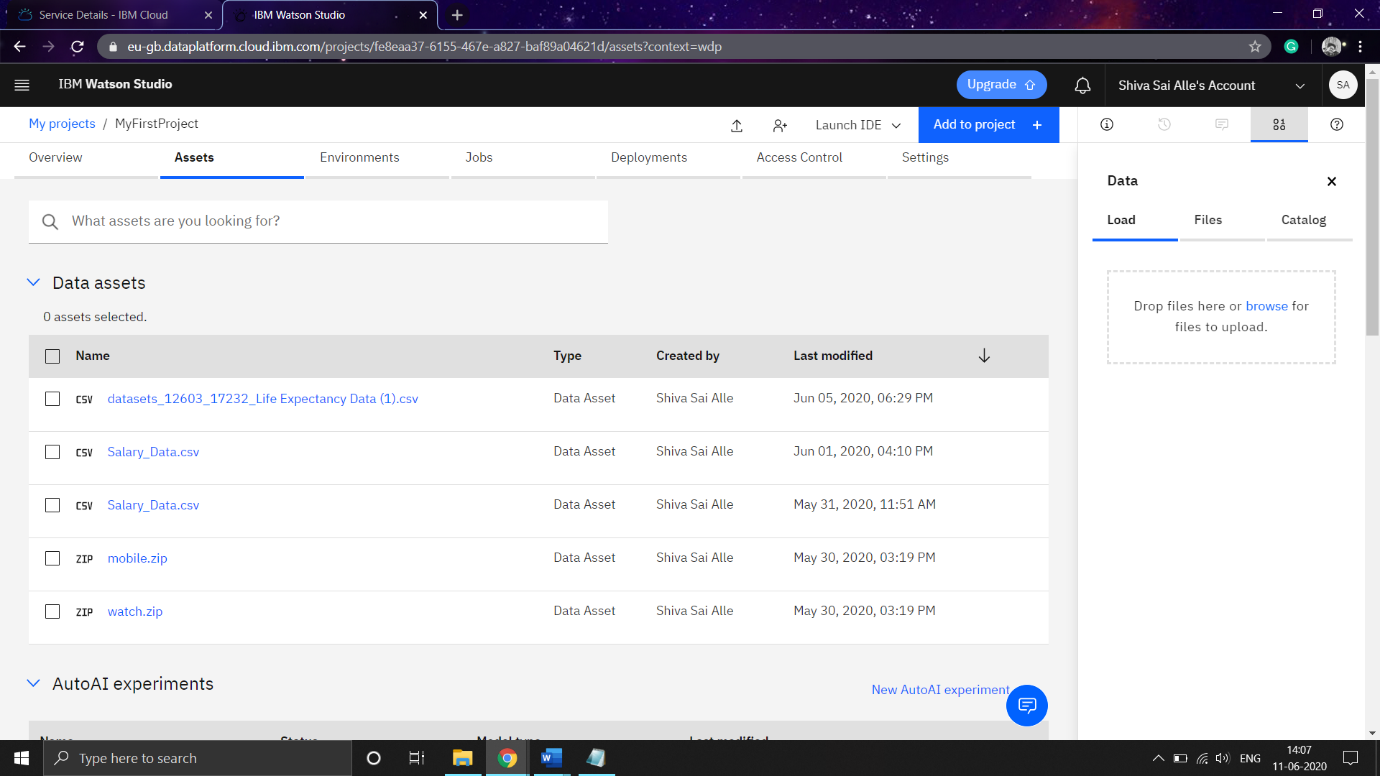
The regression model built in python is deployed on IBM cloud. The Node-RED application then sends HTTP request with all the required parameters to the trained model. The model then sends the HTTP response which is then parsed and displayed on the UI.

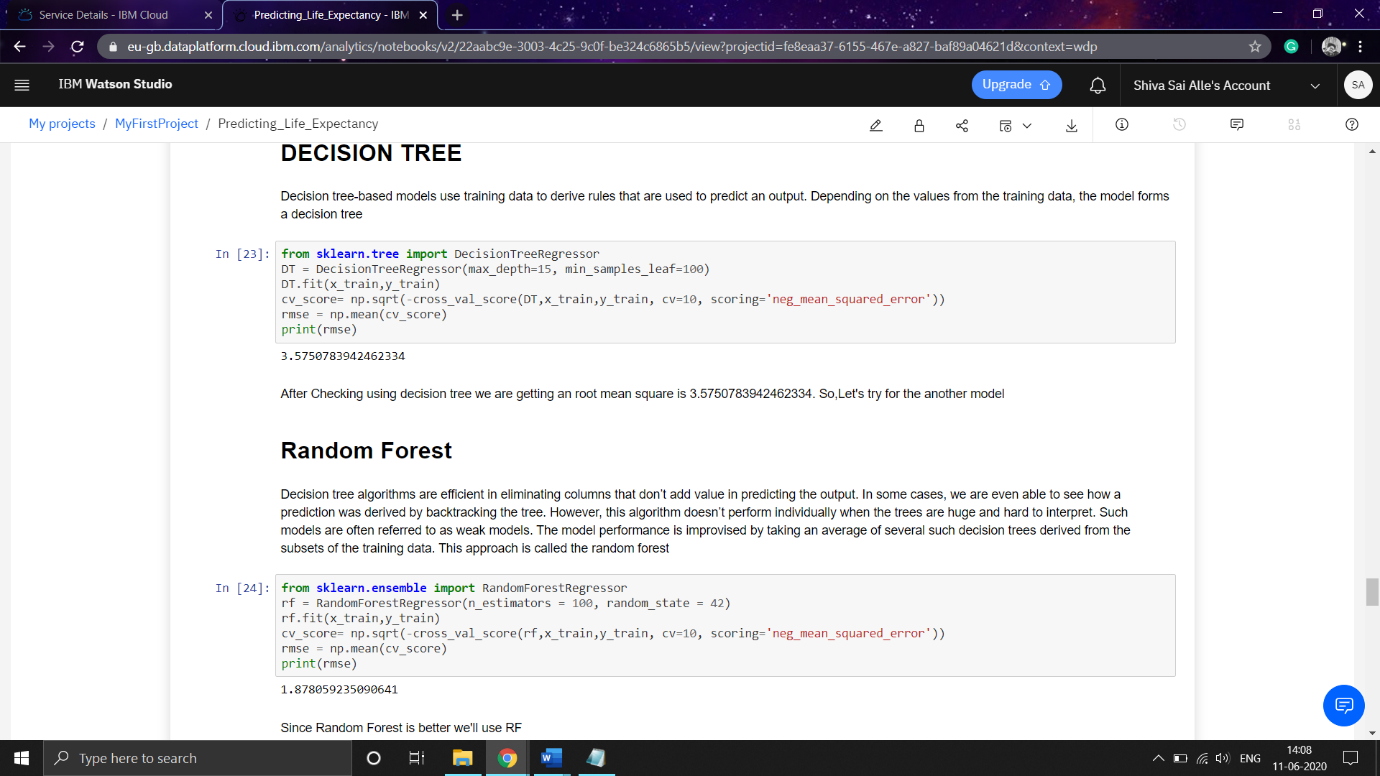
*Model Designing (Watson Studio): -*

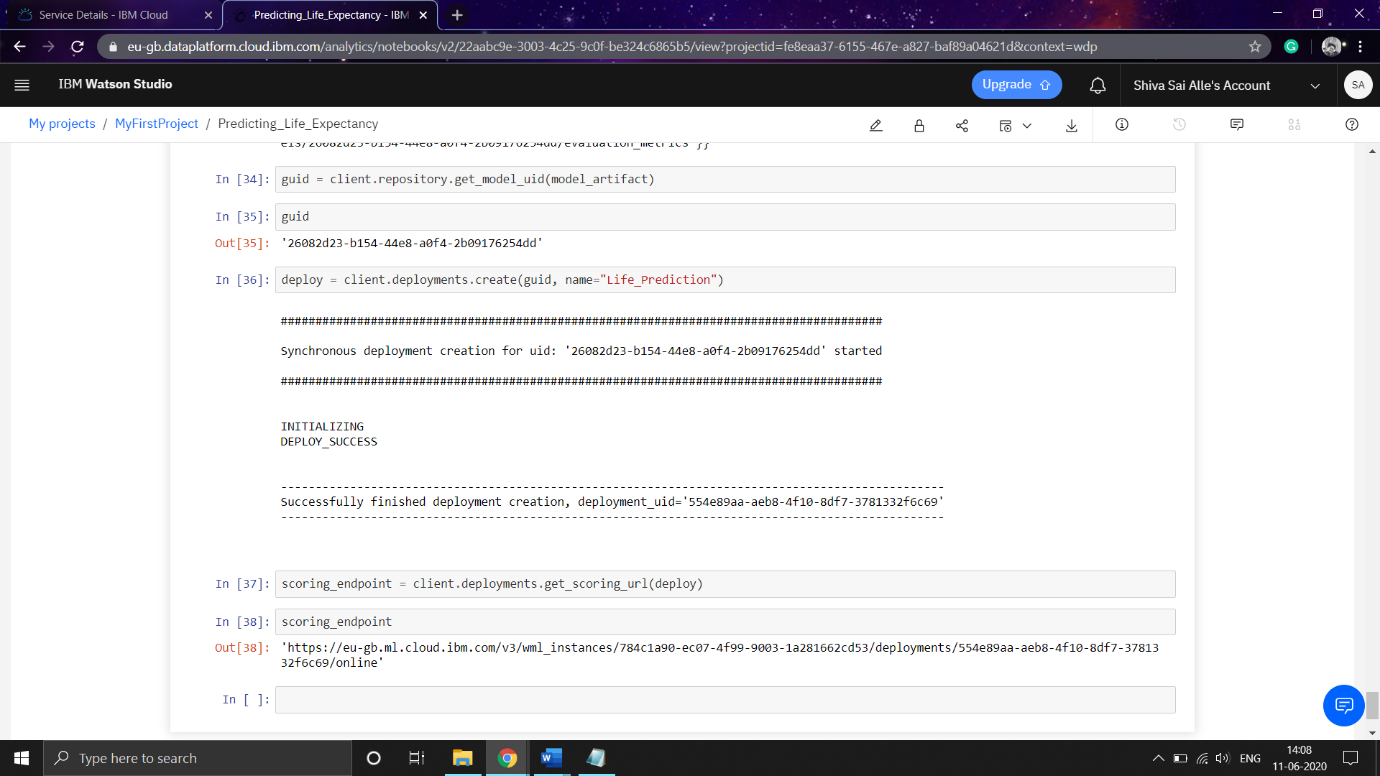
Step1: - Login to the IBM Cloud. This is how Dashboard look like.

Step2: -Go to Resource List, in services we can find the Watson studio.

(Predicting Life Expectancy using Python)

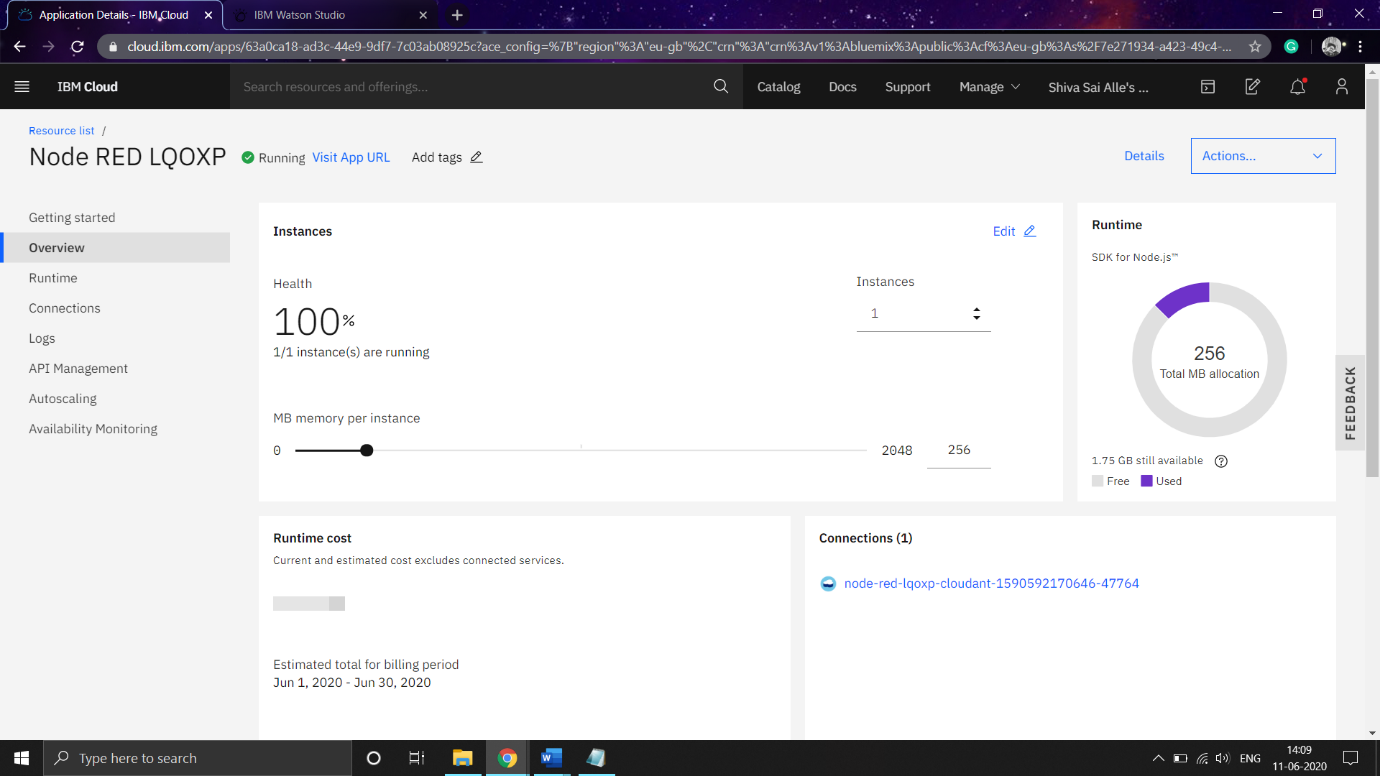
Step3: - New Project => Create an empty Project => Give project name => Click Create => Add to Project => Notebook



Scoring Endpoint: - For WML credentials, replace with your own credentials of the service. Services => Machine Learning Service => Service Credentials => Copy the credentials.

***User Interface Integration with ML Model (Node- Red) :***

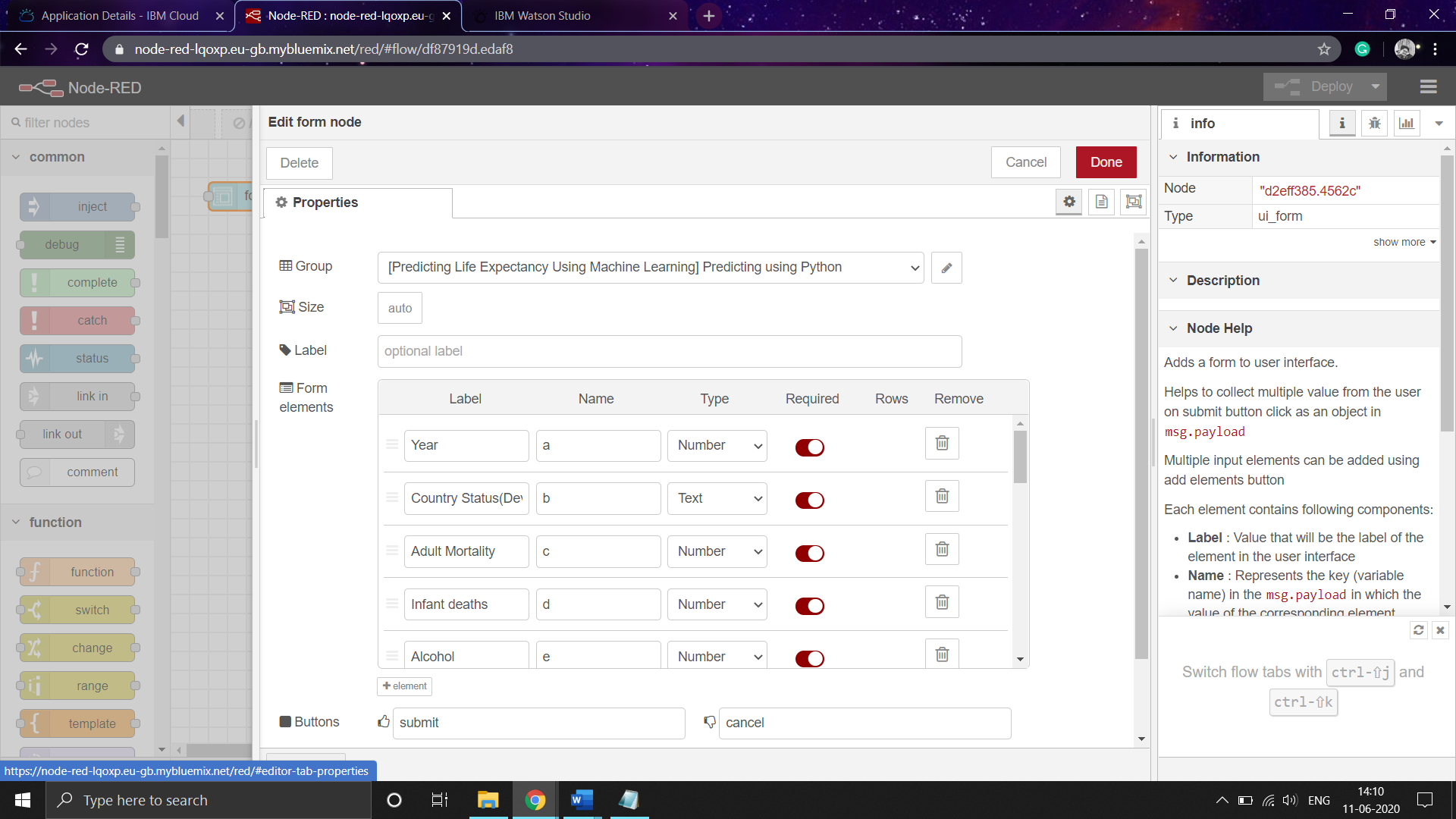
Resources => Cloud Foundry Apps => Node-RED => Visit App URL



***Nodes:***

1) Form Node: Edit => Add New UI Tab

2) Function Node: To obtain access to Machine Learning Services. Requires API Key

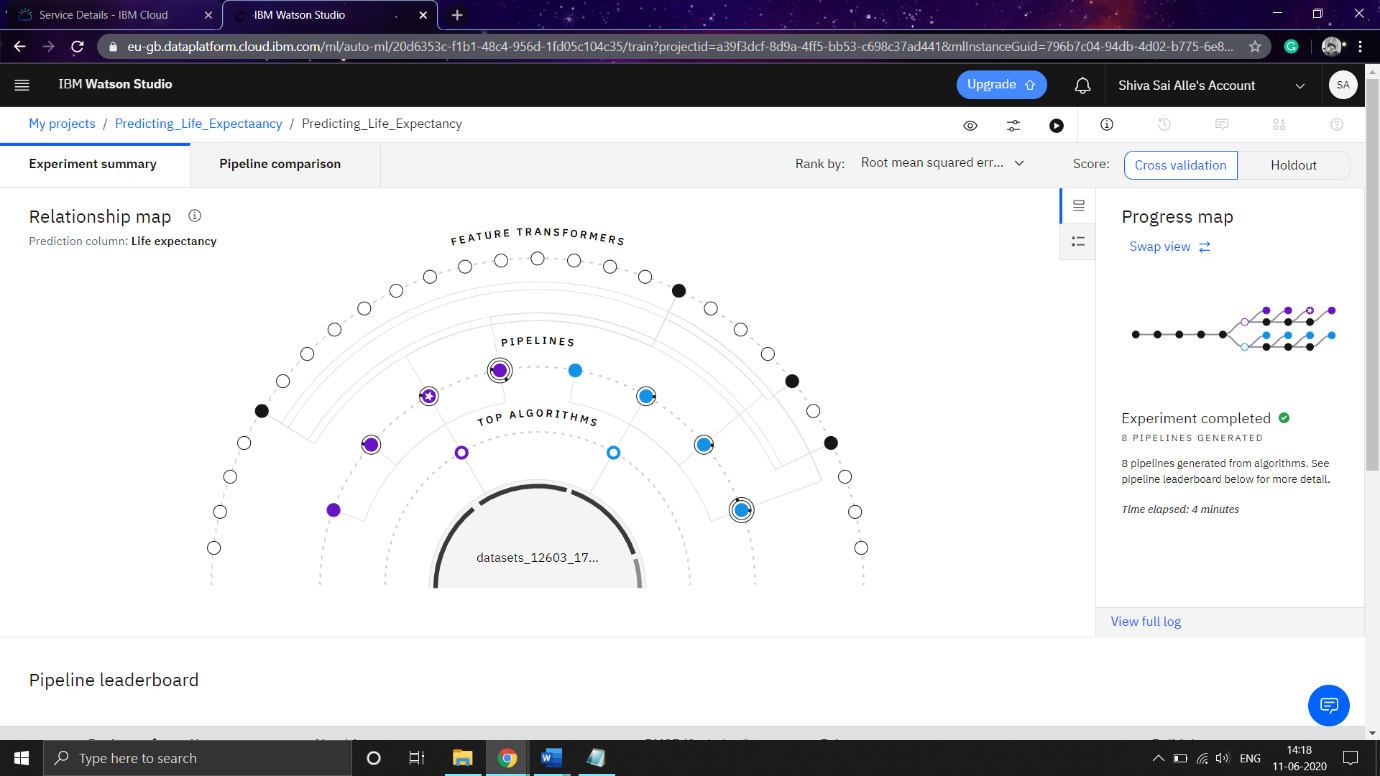
3) HTTP Request Node: POST method and returns a parsed JSON object. Gains access to Machine Learning services.

(Predicting Life Expectancy using Auto AI)

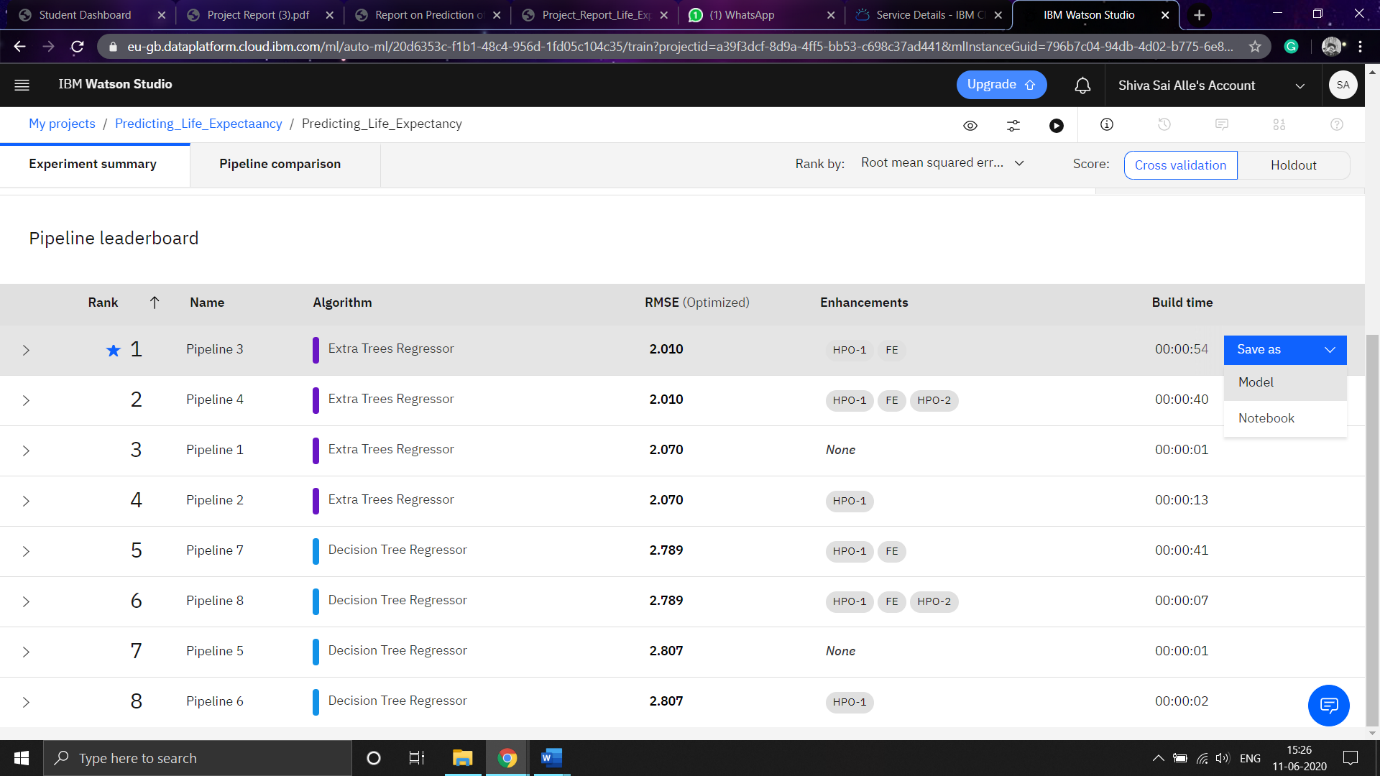
Follow the step 1 and step2 on above;

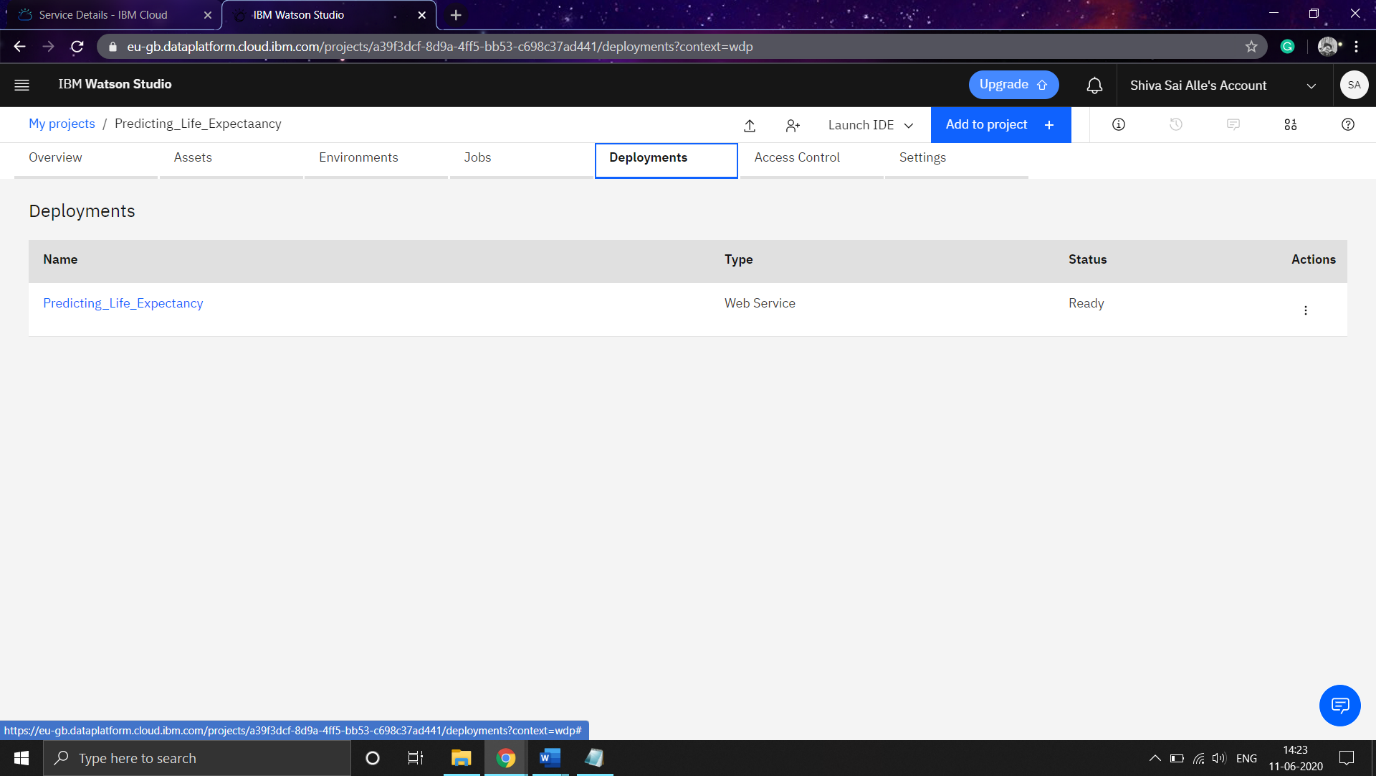
Step4: - New Project => Create an empty Project => Give project name => Click Create => Add to Project => Auto AI Experiment.

Step5: -Import the Dataset => Select Prediction Column =>Once Check Experiment Settings => Run Experiment.



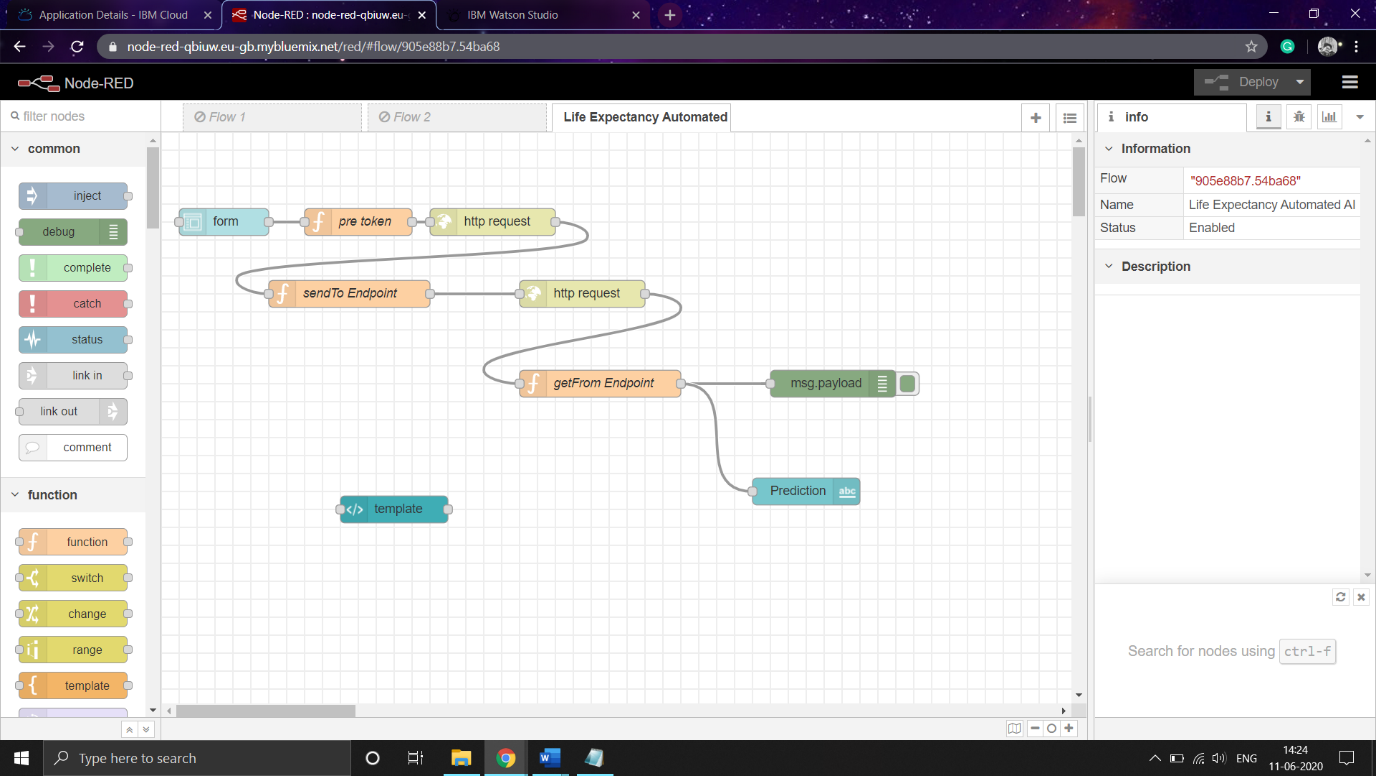
Step 6: -After Running the Experiment. Select the high rated Pipeline and save as model and after model saved click on ***view project*** .Go to Deployments =>Add Deployment => After DEPLOY\_SUCCESS => View Deployment.



After Deployment, You can Test the Model in Test Tab.

***User Interface Integration with AUTO AI(Node- Red) :***

Resources => Cloud Foundry Apps => Node-RED => Visit App URL



**4. EXPERIMENTAL INVESTIGATIONS**

Analysing the relations between various features can help us improve the performance of the model as well as decide which model would be more suitable.

4.1. Factors affecting Life Expectancy: -

After Importing the Dataset in Notebook, I analysed the Dataset like Variable Descriptions. Below are the factors (given in the dataset) which affect life expectancy of a country.

*1. Adult Mortality:* Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population)

*2. Infant Deaths:* Number of Infant Deaths per 1000 population

*3. Alcohol:* Alcohol, recorded per capita (15+) consumption (in litres of pure alcohol)

*4. Percentage Expenditure:* Expenditure on health as a percentage of Gross Domestic Product per capita (%)

*5. Hepatitis B:* Hepatitis B immunization coverage among 1-year-olds (%)

*6. Measles:* Measles - number of reported cases per 1000 population

*7. BMI:* Average Body Mass Index of the entire population

*8. Under-five deaths:* Number of under-five deaths per 1000 population

*9. Polio:* Polio (Pol3) immunization coverage among 1-year-olds (%)

*10. Total Expenditure:* General government expenditure on health as a percentage of total government expenditure (%)

*11. Diphtheria:* Diphtheria tetanus toxoid and pertussis (DTP3) immunization coverage among 1-year-olds (%)

*12. HIV/AIDS:* Deaths per 1 000 live births HIV/AIDS (0-4 years)

*13. GDP:* Gross Domestic Product per capita (in USD)

*14. Population:* Population of the country

*15*.*Thinness 5-9 years:* Prevalence of thinness among children for Age 5 to 9(%)

*16. Thinness 1-19 years:* Prevalence of thinness among children and adolescents for Age 10 to 19 (%)

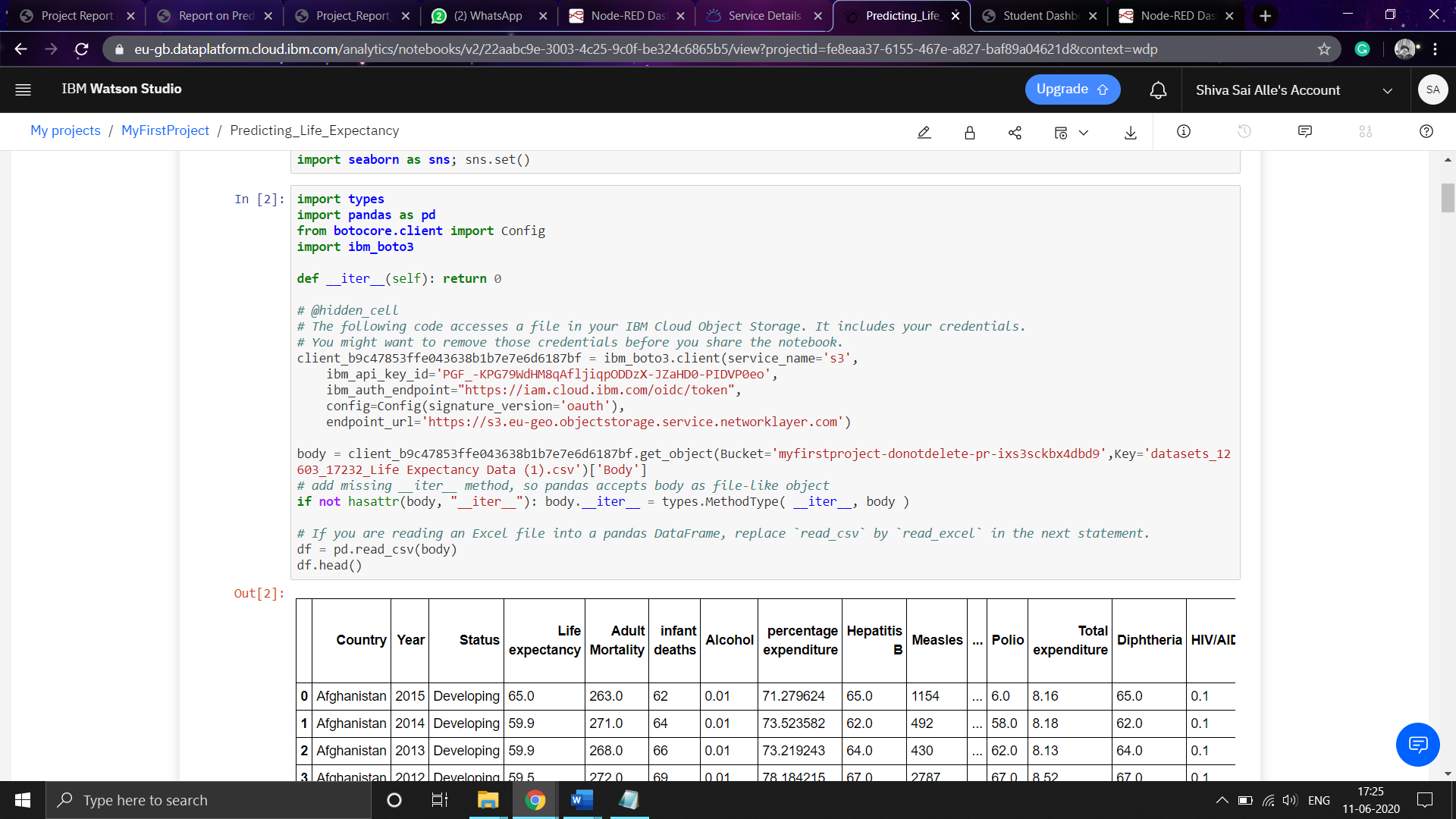
*17. Income composition of resources:* Human Development Index in terms of income composition of resources (index ranging from 0 to 1)

*18. Schooling:* Number of years of Schooling(years)

4.2 Import the Dataset to IBM Cloud: -

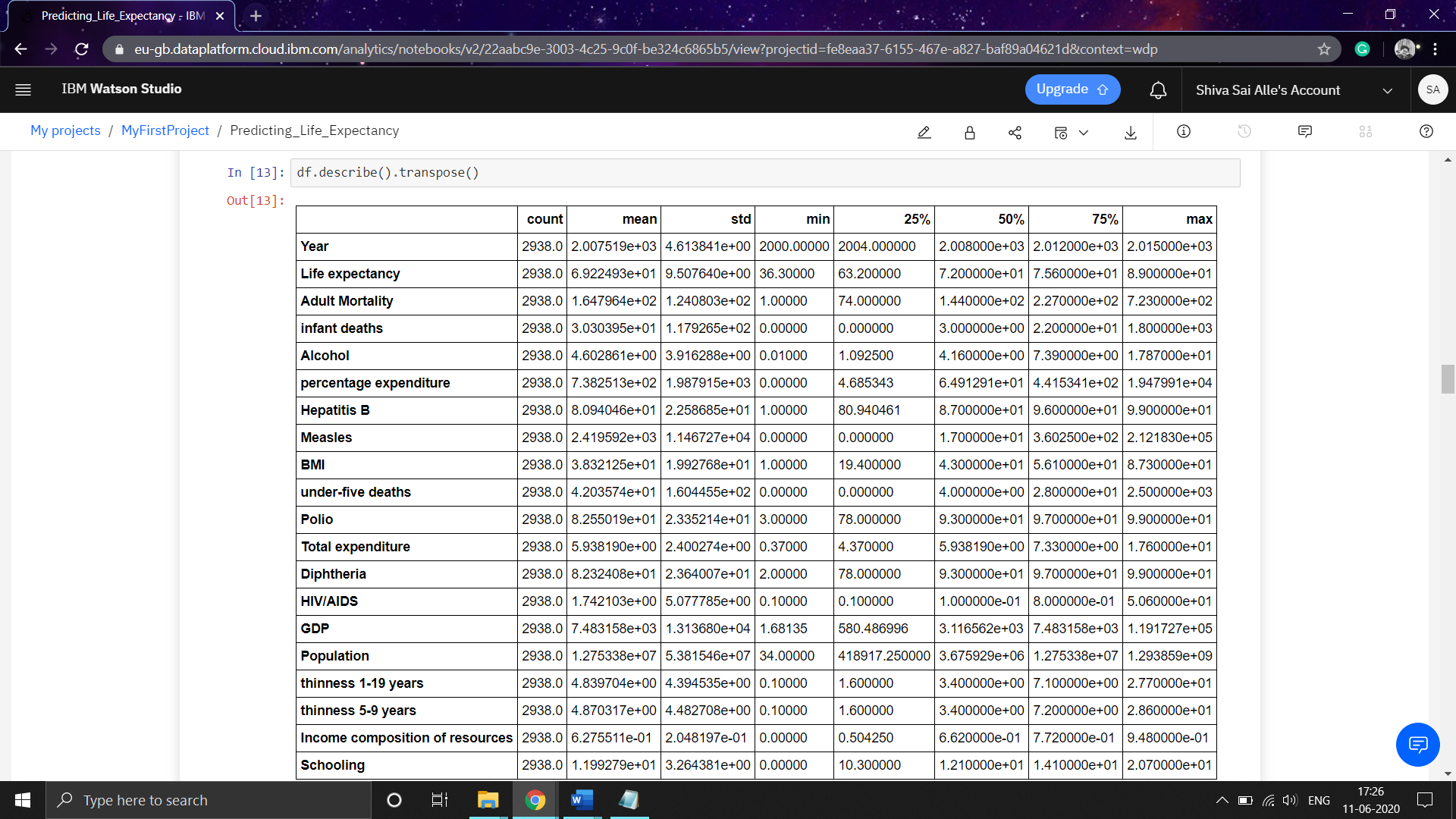
Importing the dataset in IBM cloud => Go to Find and Add Data => Adding Dataset as Pandas Data frame.

This is what I learnt.

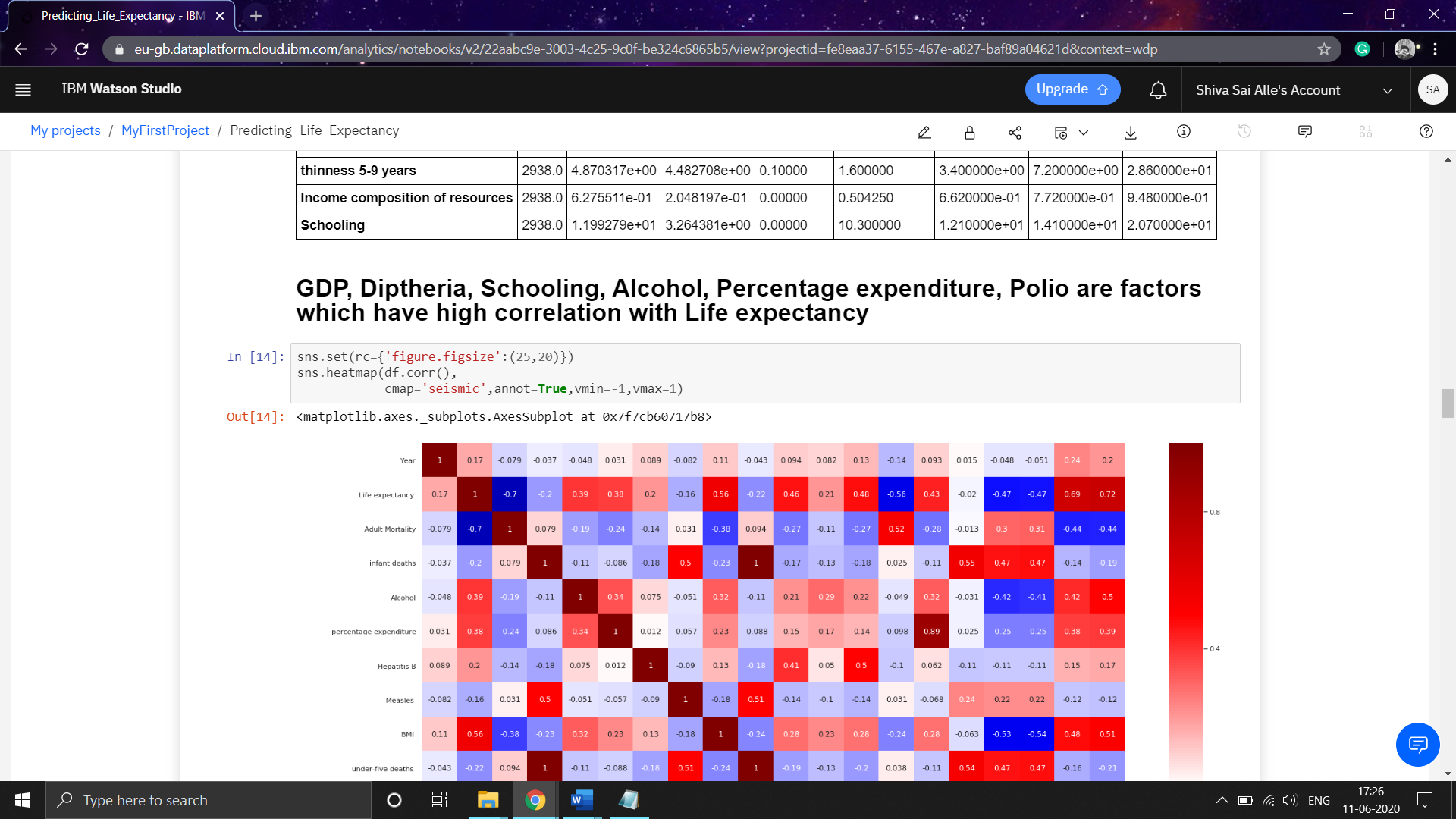
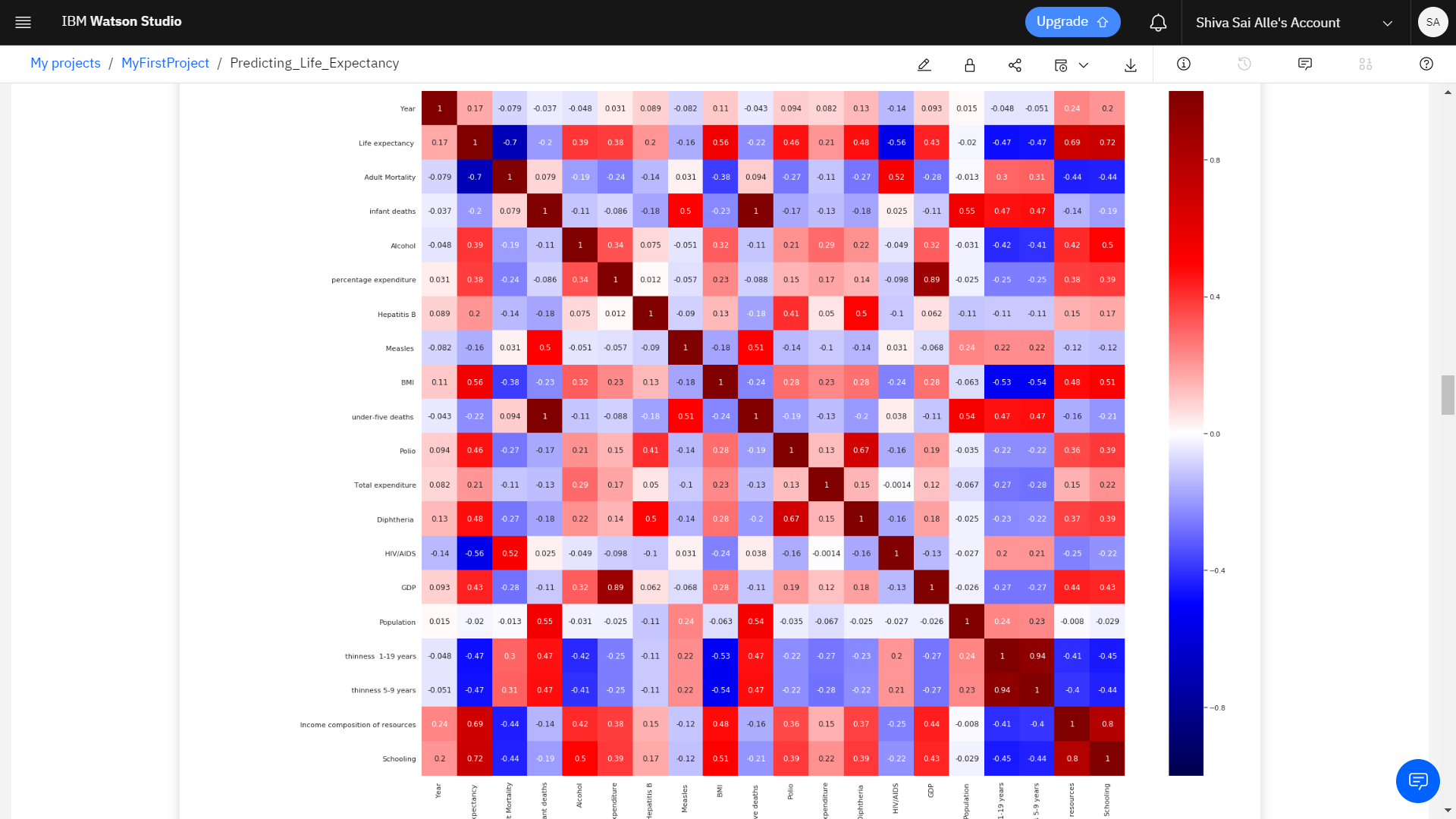


4.3 Describing the Data: -

By Describing the data, We Know the computational factors of each Column like average mean, Standard Deviation, Count, Maximum Values, Minimum Values and So…on.

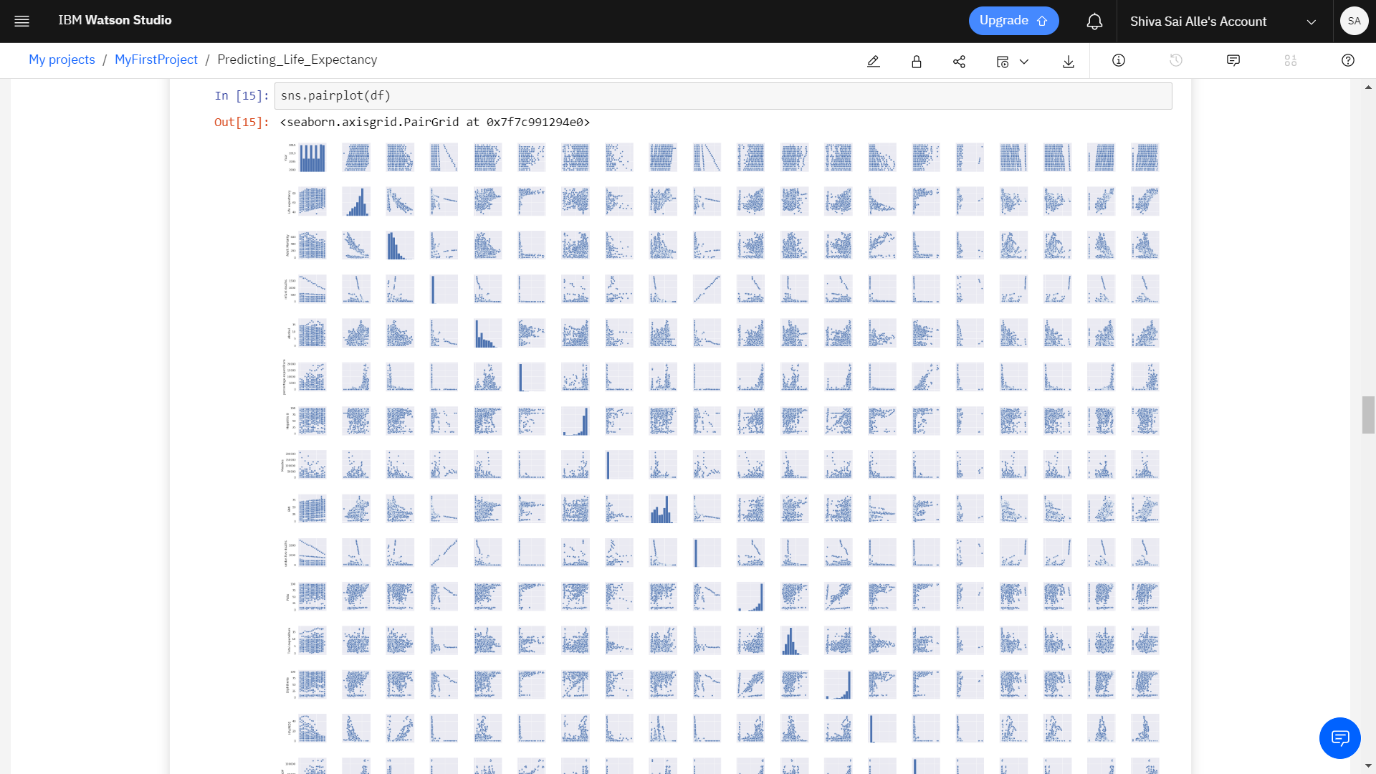


4.3 Correlation between factors and Life Expectancy:

A heatmap further showed the correlation between different columns  

The legend tells that the warmer colours show higher and positive correlation, while the colder low or negative. There is a very high correlation between thinness of 5-9-year-old and that of 1-19-year-old. Also, between population and infant deaths, under 5 deaths, another is between schooling and income composition of resources. On the other hand, Life expectancy and Adult Mortality are very highly negatively correlated.

4.4 Pairplot of the Columns:-



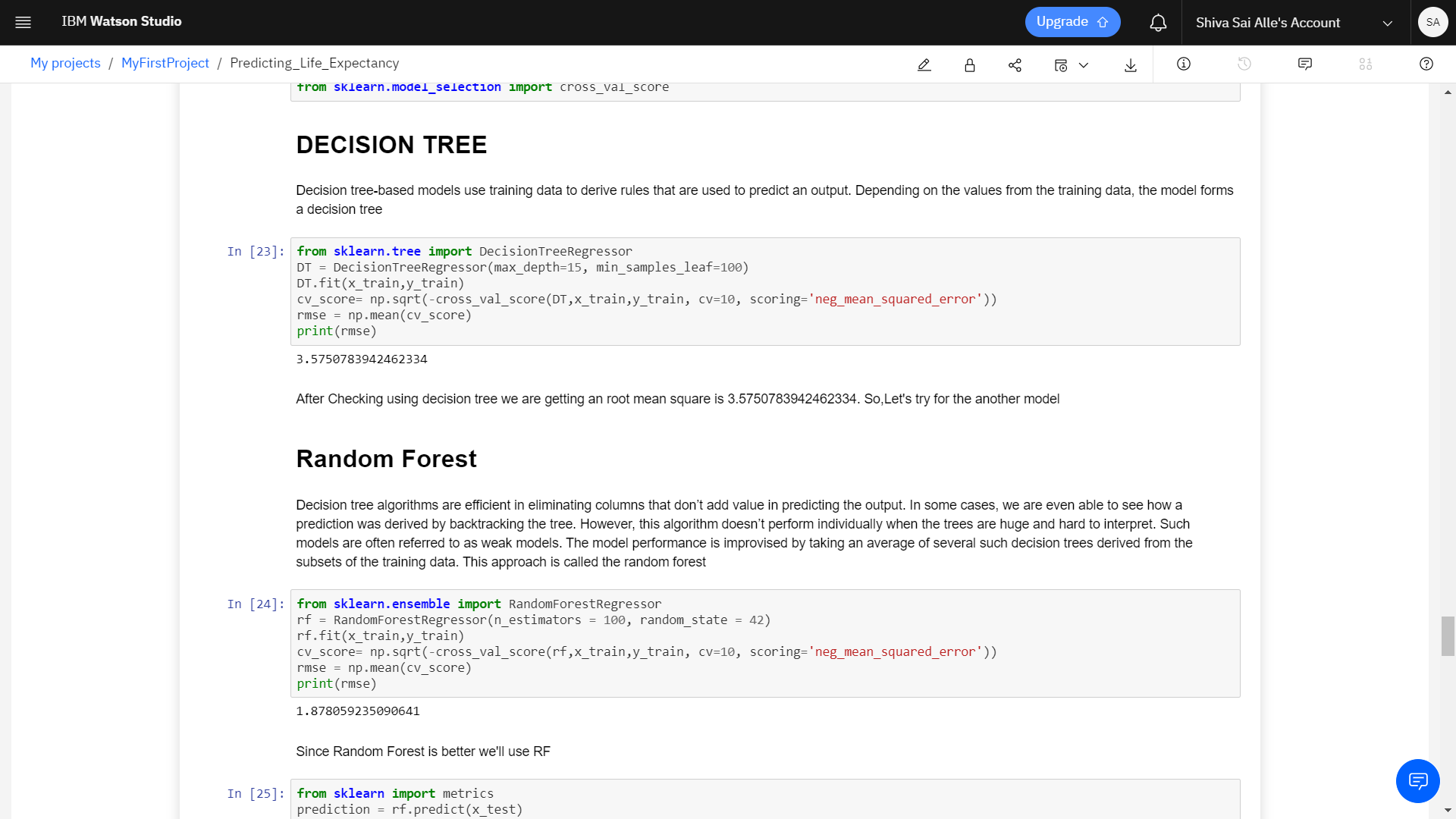
By seeing some of the graohs from the Pairplot.It seems to be a positive correlation between the Percentage of Healthcare Expenditure, Schooling, GDP and BMI and Life Expectancy, while there is a negative one between Adult 8 Mortality, AIDS and Life Expectancy, there does not seem to have any correlation between Alcohol, under 5 years – old deaths and Life Expectancy.

4.5 Implementing Regression Models

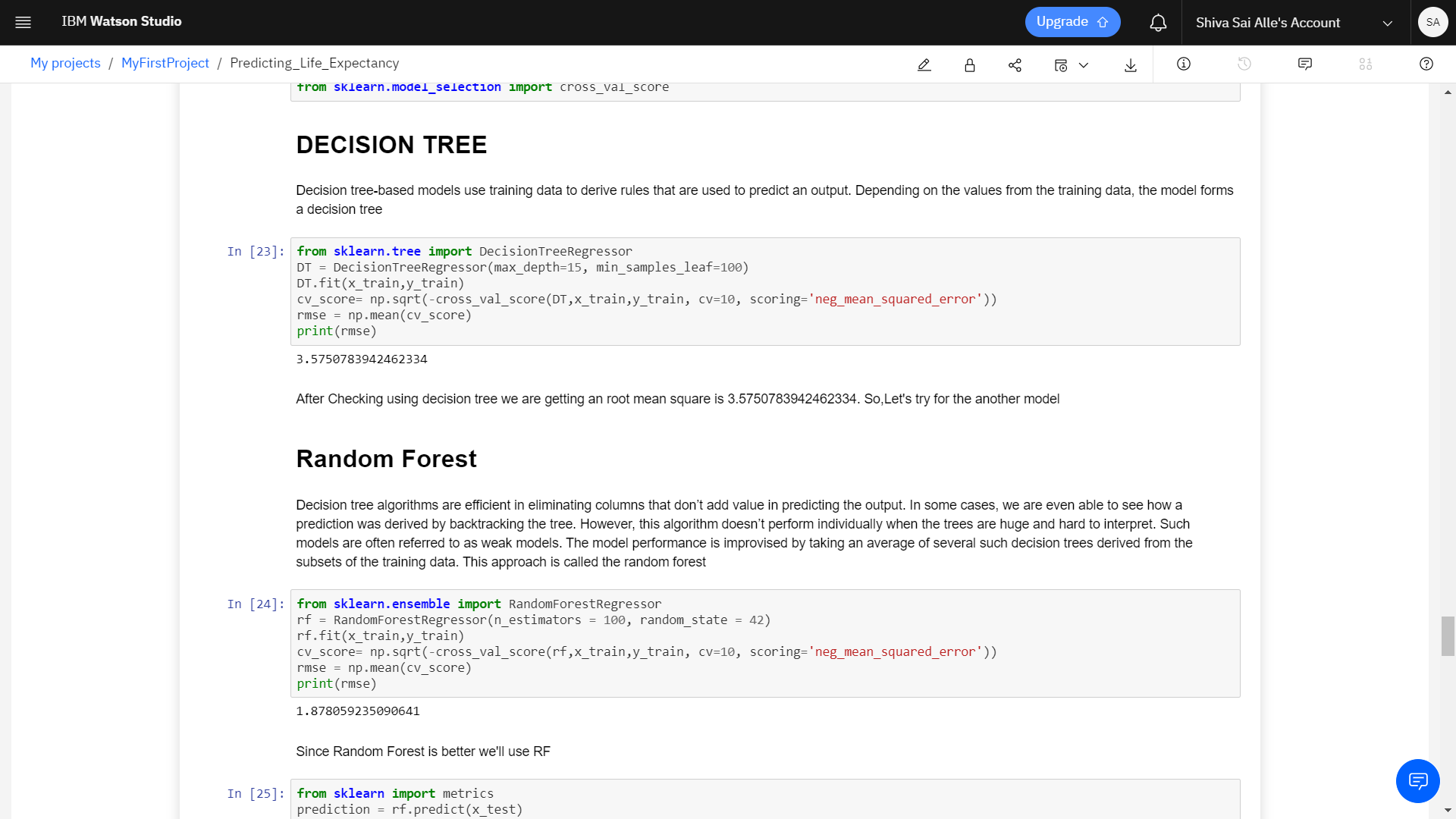
Two Regression Models were Applied :-

*1)Decision Trees: -*

A decision tree-based model builds a set of rules from the training data to be able to predict the outcome. For the sake of understanding, this algorithm is compared to trees formed through decisions. The model contains branches that represent the rules that lead to the path of the outcome, that is, the leaf. Each prediction path leads to a leaf that contains multiple values. The same principle is applied to classification-type problems as well. For regression-type problems, the final prediction is usually the average of all of the values contained in the leaf it falls under.

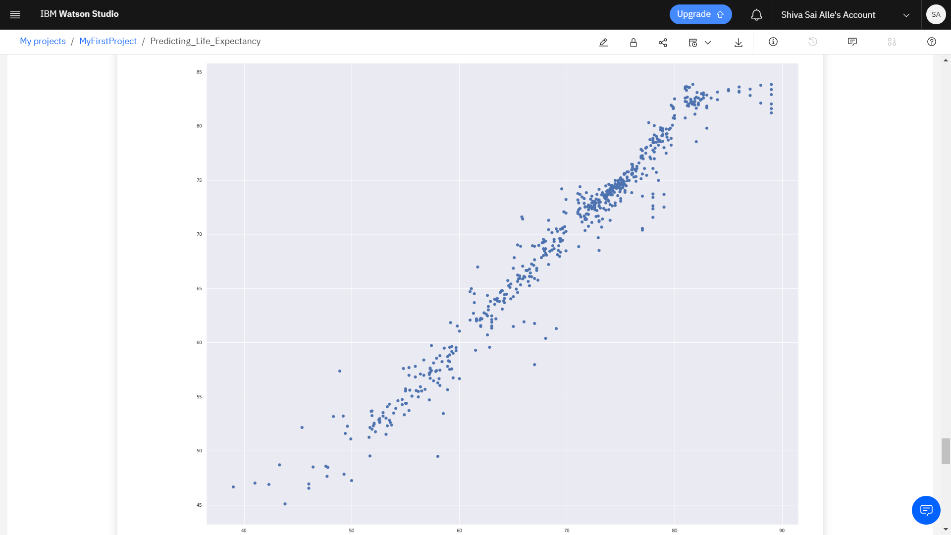


*2)Random Forest Trees: -*

Decision trees are generally considered weak models because their performance usually is not up to the expected mark when the data set is relatively large. However, when several decision trees are combined into a single model, they provide greater accuracy. Each decision tree within this random forest is built using a subset of the training data. The number of decision trees that make this random forest is an arbitrary number that can be tuned to see the changes in accuracy. When a value to be predicted is run through this resulting model, it is the average of the values acquired from each of these individual trees.

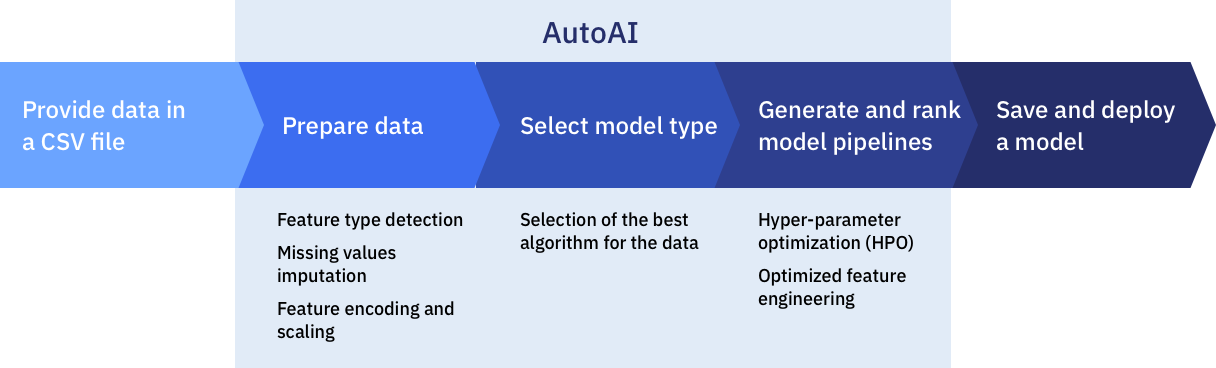
It is observable that the Mean Absolute Error in Random Forest Regression is lower than that in Decision Trees. So, for the final deployment, Random Forest Regression is used.

4.6 Prediction Plot: -



4.7 AUTO AI: -

Using Auto AI, we can build and deploy a machine learning model with sophisticated training features and no coding. The tool does most of the work for us.



**Data Pre-processing: -**

Most data sets contain different data formats and missing values, but standard machine learning algorithms work with numbers and no missing values. Auto AI applies various algorithms, or estimators, to analyse, clean, and prepare your raw data for machine learning. It automatically detects and categorizes features based on data type, such as categorical or numerical. Depending on the categorization, it uses hyper-parameter optimization to determine the best combination of strategies for missing value imputation, feature encoding, and feature scaling for your data.

**Automated Model Selection: -**

The next step is automated model selection that matches your data.  Auto AI uses a novel approach that enables testing and ranking candidate algorithms against small subsets of the data, gradually increasing the size of the subset for the most promising algorithms to arrive at the best match. This approach saves time without sacrificing performance.  It enables ranking a large number of candidate algorithms and selecting the best match for the data.

**Automated feature engineering: -**

Feature engineering attempts to transform the raw data into the combination of features that best represents the problem to achieve the most accurate prediction. Auto AI uses a unique approach that explores various feature construction choices in a structured, non-exhaustive manner, while progressively maximizing model accuracy using reinforcement learning. This results in an optimized sequence of transformations for the data that best match the algorithms of the model selection step.

**Hyperparameter optimization: -**

Finally, a hyper-parameter optimization step refines the best performing model pipelines. Auto AI uses a novel hyper-parameter optimization algorithm optimized for costly function evaluations such as model training and scoring that are typical in machine learning. This approach enables fast convergence to a good solution despite long evaluation times of each iteration.

**5. FLOWCHART**

User Gives Input in Form

It throws error in debug tab

Checks input valid or not

No

Yes

Displays the Output to User

Node-red (output passed to Node-RED)

Predicting the output

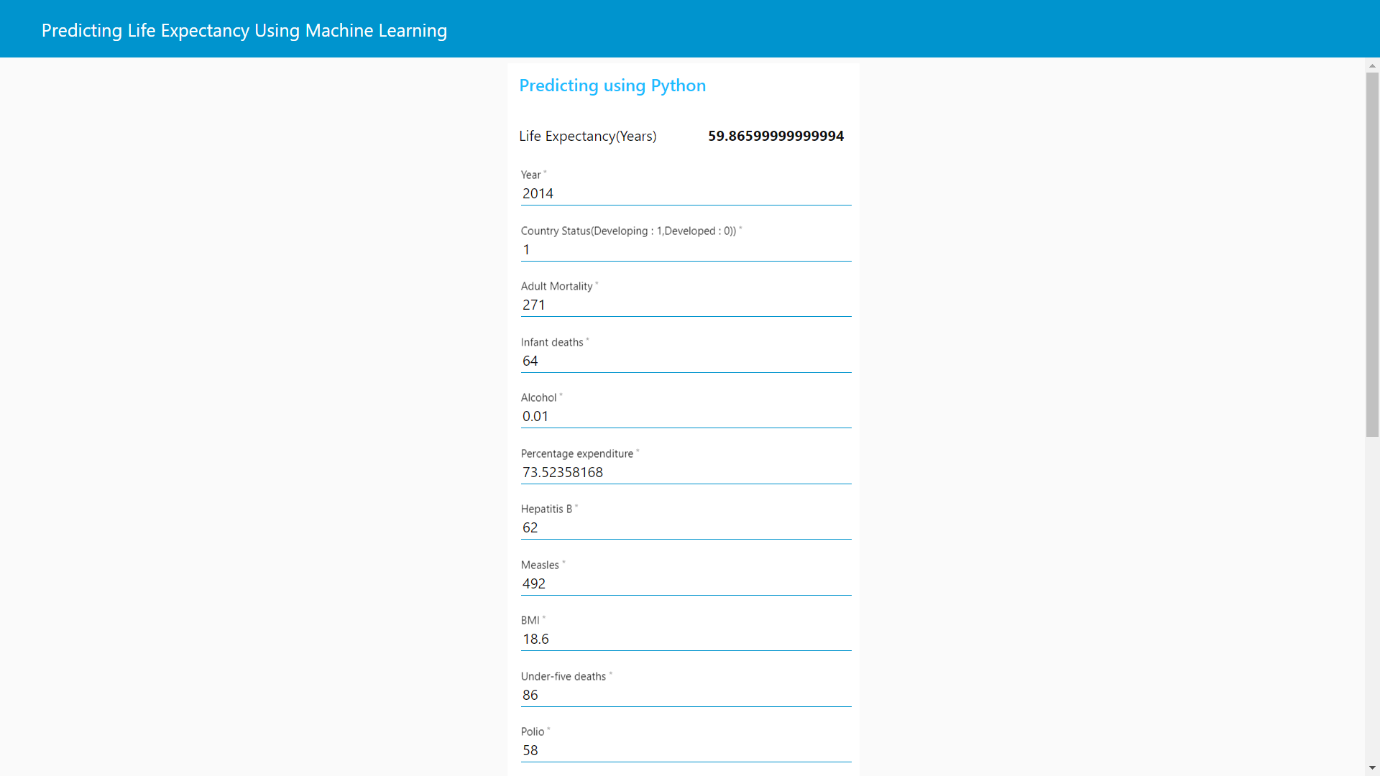
Inputs are passed to ML model

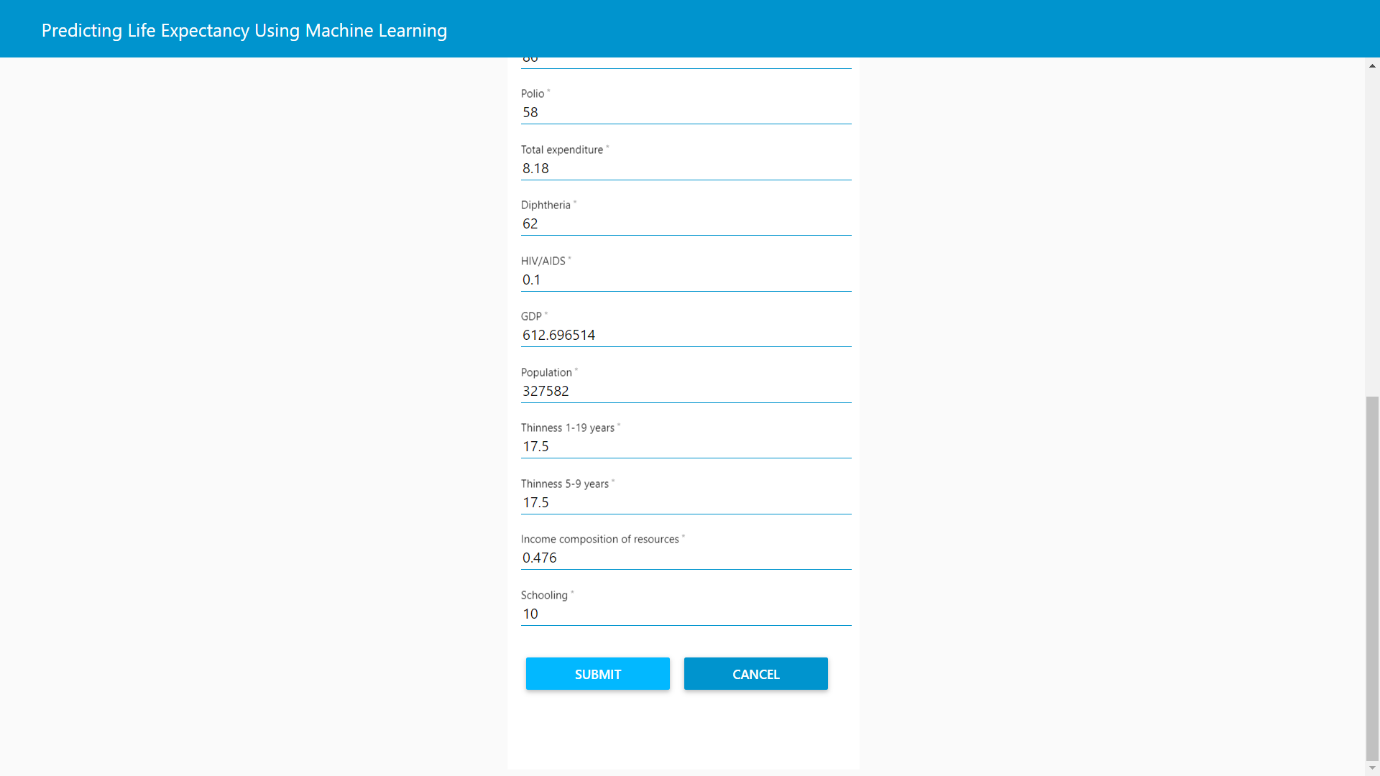
Node-RED sends request to ML web service

Node-RED

**6.RESULT**

6.1 Prediction Using Python: -





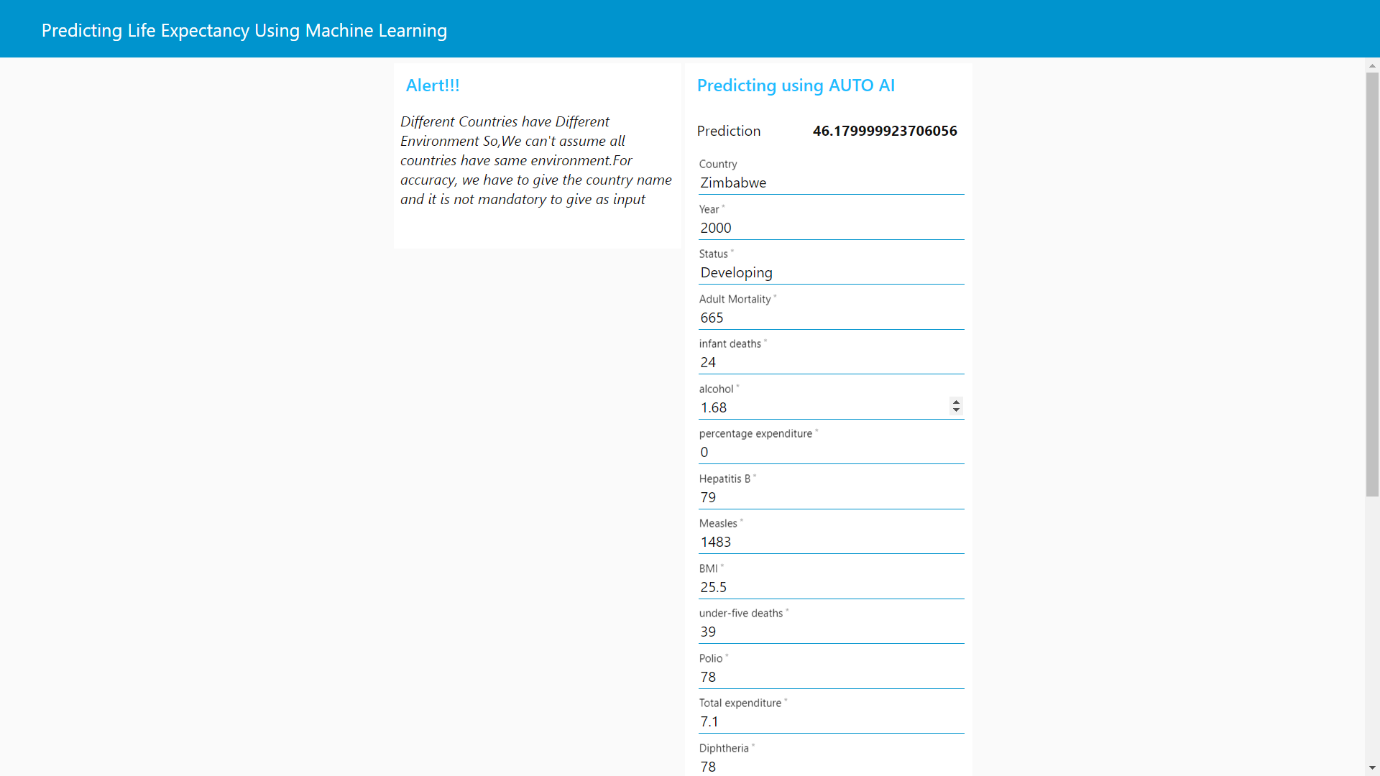
For Afghanistan: 2014

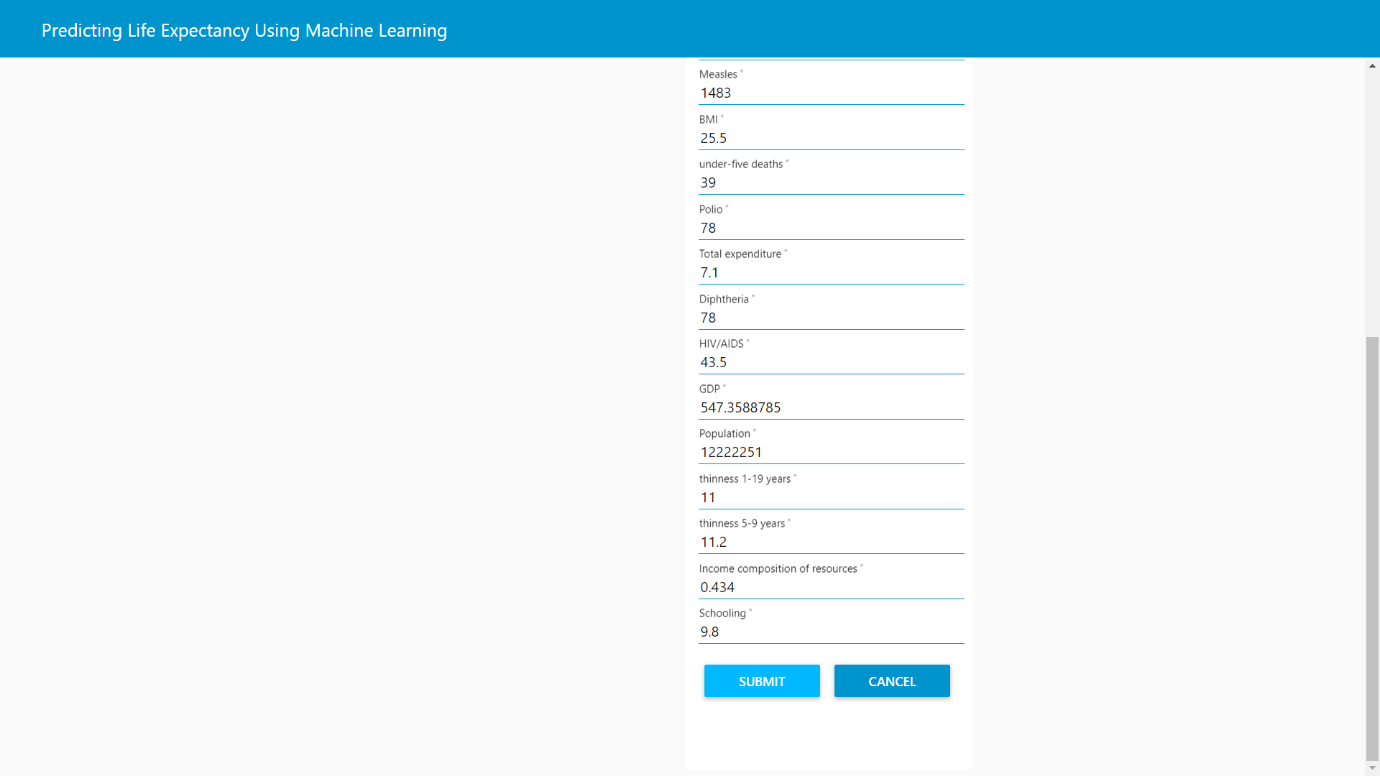
Actual Value: 59.9

Predicted value: 59.86

Error Percentage: (59.9-59.86)/59.9 = 0.06%

6.2 Predicting Using Auto AI: -





For Zimbabwe: 2000

Actual Value: 46

Predicted value: 46.17

Error Percentage: (46-46.17)/46 = 0.36%(negative)

**7. ADVANTAGES & DISADVANTAGES**

7.1 Advantages: -

* The machine learning algorithm used in this 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.
* Random Forest is usually robust to outliers and can handle them automatically. It is comparatively less impacted by noise data.
* The application learns the patterns and trends hidden within the data without human intervention which makes predicting much simpler and easier. The more data is fed to the algorithm, the higher the accuracy of the algorithm is. It is also the key component in technologies for automation.
* We can create a user interface easily with help of Node-RED and give the input to the model and predicts the Life expectancy.
* No risk of front-end HTML and CSS programming.

7.2 Disadvantages: -

* **Complexity:** Random Forest creates a lot of trees (unlike only one tree in case of decision tree) and combines their outputs. By default, it creates 100 trees in Python ski-learn library. To do so, this algorithm requires much more computational power and resources. On the other hand, decision tree is simple and does not require so much computational resources. Random Forest require much more time to train as compared to decision trees. But, Predicts the most accuracy value.
* Machine learning can also be very time-consuming. When the size of the data fed to the machine learning is very large, the computational cost and the time taken to train the model on the data increases drastically. This can increase the cost of resources required to implement the application on a large scale.
* The main disadvantage is that no one can predict the future. No one knows when someone will die, who will get cancer or not, who will recover and who won't. Statistics work in generalities. Humans, however, do not.
* It may happen that our model will not predict right when sudden influencing factors affect human life like Ex : COVID-19

**8. APPLICATIONS**

* Life expectancy predictions have the potential to be beneficial to individuals, health service providers and governments. For instance, they would make people more aware of their general health, and its improvement or deterioration over time. This may motivate them to make healthier lifestyle choices.
* They could also be used by insurance companies to provide individualised services, such as how some car insurance companies use black-box technology to reduce premiums for more cautious drivers.
* It could help the government bodies take appropriate measures to control the population growth and also direct the utilization of the increase in human resources and skillset acquired by people over many years. Across countries, high life expectancy is associated with high income per capita. Increase in life expectancy also leads to an increase in the “manpower” of a country. The knowledge asset of a country increases with the number of individuals in a country.
* Advance Care Planning.

**9.** **CONCLUSION**

Predicting lifespan of human beings can greatly alter our lives. Human behaviour and activities are so unpredictable, it may almost be impossible to correctly predict lifespan. However, with the help of Machine learning algorithms such as Regression models, we can get close to predicting a roundabout value.

This breakthrough can widely impact health sectors and economic sectors by improving the resources, funds and services provided to the common people. It can also increase the ease of access to the individuals.

With the help of Machine Learning algorithms, one can ease the process of automating the application and predicting the expectancy with an admirable accuracy. It also reduces the effort and time put into deploying the application and making it more accessible to the users.

User can interact with the system via a simple user interface which is in the form of a form with input spaces which the user needs to fill the inputs into the form.

**10. FUTURE SCOPE**

Planning Health Services: -

The government can plan health services better using the data and future predictions. Life expectancy plays a major role in development of a country, hence, using predictions and trends, the health infrastructure can be improved. A mobile application can be developed that uses personal health data (from Smart Watch and Health apps) and historical data of the country that user lives in and predict the expected life span of that user

Future Usage: -

For future use, one can integrate the life expectancy prediction with providing suggestions and medications to the individual using the application. This will help predict as well as increase the individual’s life expectancy.

The scalability and flexibility of the application can also be improved with advancement in technology and availability of new and improved resources. we can connect the model to the database to have the record of predictions. This will help us analyse the trends in the life span.

Pharmaceutical companies can check which diseases impact more people and therefore impact life expectancy and based on this manufacture medicine.

**11. BIBILOGRAPHY**

1.Product and Services: -

<https://www.ibm.com/watson/products-services>

2.Machine Learning Reference: -

<https://developer.ibm.com/technologies/machine-learning/series/learning-path-machine-learning-for-developers/>

<https://bookdown.org/caoying4work/watsonstudio-workshop/jn.html>

3.Auto AI: -

<https://developer.ibm.com/tutorials/watson-studio-auto-ai/>

<https://www.youtube.com/watch?v=IDKCmC1fCiU>

<https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/autoai-overview.html>

4.Data-set: -<https://www.kaggle.com/kumarajarshi/life-expectancy-who>

5.Smart Bridge(Bootcamp): - <https://www.youtube.com/channel/UCvB8PgOZdb2y7lgToPE-Dfw>

6.Additional Websites: -<https://towardsdatascience.com/what-really-drives-higher-life-expectancy-e1c1ec22f6e1>

**APPENDIX**

A. Source Code: -

1)Machine Learning Notebook:

Notebook.ipynb

<https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/d674e3a4-4396-4d1a-b235-6789211e58b3/view?access_token=af8589ba0a914b6c9c7b3d0a29ff12f38ff6f8f474a1362f46c3e46e7c52dc58>

3)Node-RED URL: -

[node-red-gfxho.eu-gb.mybluemix.net](https://node-red-gfxho.eu-gb.mybluemix.net/)

3)Auto-AI: -

Autoai.ipynb

In [ ]:

**try**:

**import** **autoai\_libs**

**except** **Exception** **as** e:

**import** **subprocess**

out = subprocess.check\_output('pip install autoai-libs'.split(' '))

**for** line **in** out.splitlines():

print(line)

**import** **autoai\_libs**

**import** **sklearn**

**try**:

**import** **xgboost**

**except**:

print('xgboost, if needed, will be installed and imported later')

**try**:

**import** **lightgbm**

**except**:

print('lightgbm, if needed, will be installed and imported later')

**from** **sklearn.cluster** **import** FeatureAgglomeration

**import** **numpy**

**from** **numpy** **import** inf, nan, dtype, mean

**from** **autoai\_libs.sklearn.custom\_scorers** **import** CustomScorers

**import** **sklearn.ensemble**

**from** **autoai\_libs.cognito.transforms.transform\_utils** **import** TExtras, FC

**from** **autoai\_libs.transformers.exportable** **import** \*

**from** **autoai\_libs.utils.exportable\_utils** **import** \*

**from** **sklearn.pipeline** **import** Pipeline

known\_values\_list=[]

In [ ]:

*# compose a decorator to assist pipeline instantiation via import of modules and installation of packages*

**def** decorator\_retries(func):

**def** install\_import\_retry(\*args, \*\*kwargs):

retries = 0

successful = **False**

failed\_retries = 0

**while** retries < 100 **and** failed\_retries < 10 **and** **not** successful:

retries += 1

failed\_retries += 1

**try**:

result = func(\*args, \*\*kwargs)

successful = **True**

**except** **Exception** **as** e:

estr = str(e)

**if** estr.startswith('name ') **and** estr.endswith(' is not defined'):

**try**:

**import** **importlib**

module\_name = estr.split("'")[1]

module = importlib.import\_module(module\_name)

globals().update({module\_name: module})

print('import successful for ' + module\_name)

failed\_retries -= 1

**except** **Exception** **as** import\_failure:

print('import of ' + module\_name + ' failed with: ' + str(import\_failure))

**import** **subprocess**

**if** module\_name == 'lightgbm':

**try**:

print('attempting pip install of ' + module\_name)

process = subprocess.Popen('pip install ' + module\_name, shell=**True**)

process.wait()

**except** **Exception** **as** E:

print(E)

**try**:

**import** **sys**

print('attempting conda install of ' + module\_name)

process = subprocess.Popen('conda install --yes --prefix **{sys.prefix}** -c powerai ' + module\_name, shell = **True**)

process.wait()

**except** **Exception** **as** lightgbm\_installation\_error:

print('lightgbm installation failed!' + lightgbm\_installation\_error)

**else**:

print('attempting pip install of ' + module\_name)

process = subprocess.Popen('pip install ' + module\_name, shell=**True**)

process.wait()

**try**:

print('re-attempting import of ' + module\_name)

module = importlib.import\_module(module\_name)

globals().update({module\_name: module})

print('import successful for ' + module\_name)

failed\_retries -= 1

**except** **Exception** **as** import\_or\_installation\_failure:

print('failure installing and/or importing ' + module\_name + ' error was: ' + str(

import\_or\_installation\_failure))

**raise** (**ModuleNotFoundError**('Missing package in environment for ' + module\_name +

'? Try import and/or pip install manually?'))

**elif** type(e) **is** **AttributeError**:

**if** 'module ' **in** estr **and** ' has no attribute ' **in** estr:

pieces = estr.split("'")

**if** len(pieces) == 5:

**try**:

**import** **importlib**

print('re-attempting import of ' + pieces[3] + ' from ' + pieces[1])

module = importlib.import\_module('.' + pieces[3], pieces[1])

failed\_retries -= 1

**except**:

print('failed attempt to import ' + pieces[3])

**raise** (e)

**else**:

**raise** (e)

**else**:

**raise** (e)

**if** successful:

print('Pipeline successfully instantiated')

**else**:

**raise** (**ModuleNotFoundError**(

'Remaining missing imports/packages in environment? Retry cell and/or try pip install manually?'))

**return** result

**return** install\_import\_retry

In [ ]:

*# Metadata used in retrieving data and computing metrics. Customize as necessary for your environment.*

*#data\_source='replace\_with\_path\_and\_csv\_filename'*

target\_label\_name = \_input\_metadata['target\_label\_name']

learning\_type = \_input\_metadata['learning\_type']

optimization\_metric = \_input\_metadata['optimization\_metric']

random\_state = \_input\_metadata['random\_state']

cv\_num\_folds = \_input\_metadata['cv\_num\_folds']

holdout\_fraction = \_input\_metadata['holdout\_fraction']

**if** 'data\_provenance' **in** \_input\_metadata:

data\_provenance = \_input\_metadata['data\_provenance']

**else**:

data\_provenance = **None**

**if** 'pos\_label' **in** \_input\_metadata **and** learning\_type == 'classification':

pos\_label = \_input\_metadata['pos\_label']

**else**:

pos\_label = **None**

In [ ]:

credentials\_0 = {

'ENDPOINT': 'https://s3.eu-geo.objectstorage.softlayer.net',

'IBM\_AUTH\_ENDPOINT': 'https://iam.bluemix.net/oidc/token/',

'APIKEY':’ 6EXQncDn644ci9EmQSIZI15xQntbIVSKtAuqDjq-jTzU'\_,

'BUCKET': 'predictinglifeexpectaancy-donotdelete-pr-zntkaclc92nifs',

'FILE': 'datasets\_12603\_17232\_Life Expectancy Data (1).csv',

'SERVICE\_NAME': 's3',

'ASSET\_ID': '1',

}

In [ ]:

**import** **pandas** **as** **pd**

csv\_encodings=['UTF-8','Latin-1'] *# supplement list of encodings as necessary for your data*

df = **None**

readable = **None** *# if automatic detection fails, you can supply a filename here*

*# First, obtain a readable object*

*# Cloud Object Storage data access*

*# Assumes COS credentials are in a dictionary named 'credentials\_0'*

credentials = df = globals().get('credentials\_0')

**if** readable **is** **None** **and** credentials **is** **not** **None** :

**try**:

**import** **types**

**import** **pandas** **as** **pd**

**import** **io**

**import** **os**

**except** **Exception** **as** import\_exception:

print('Error with importing packages - check if you installed them on your environment')

**try**:

**if** credentials['SERVICE\_NAME'] == 's3':

**try**:

**from** **botocore.client** **import** Config

**import** **ibm\_boto3**

**except** **Exception** **as** import\_exception:

print('Installing required packages!')

!pip install ibm-cos-sdk

print('accessing data via Cloud Object Storage')

**try**:

cos\_client = ibm\_boto3.resource(service\_name=credentials['SERVICE\_NAME'],

ibm\_api\_key\_id=credentials['APIKEY'],

ibm\_auth\_endpoint=credentials['IBM\_AUTH\_ENDPOINT'],

config=Config(signature\_version='oauth'),

endpoint\_url=credentials['ENDPOINT'])

**except** **Exception** **as** cos\_exception:

print('unable to create client for cloud object storage')

**try**:

cos\_client.meta.client.download\_file(Bucket=credentials['BUCKET'], Filename=credentials['FILE'], Key=credentials['FILE'])

**except** **Exception** **as** cos\_access\_exception:

print('unable to access data object in cloud object storage with credentials supplied')

**try**:

**for** encoding **in** csv\_encodings:

df = pd.read\_csv(credentials['FILE'], encoding = encoding, sep = **None**, engine = 'python')

os.remove(credentials['FILE'])

print('Data loaded from cloud object storage with encoding ' + encoding)

**break**

**except** **Exception** **as** cos\_object\_read\_exception:

print('unable to access data object from cos object with encoding ' + encoding)

**elif** credentials['SERVICE\_NAME'] == 'fs':

print('accessing data via File System')

**try**:

df = pd.read\_csv(credentials['FILE'], sep = **None**, engine = 'python')

**except** **Exception** **as** FS\_access\_exception:

print('unable to access data object in File System with path supplied')

**except** **Exception** **as** data\_access\_exception:

print('unable to access data object with credentials supplied')

*# IBM Cloud Pak for Data data access*

project\_filename = globals().get('project\_filename')

**if** readable **is** **None** **and** 'credentials\_0' **in** globals() **and** 'ASSET\_ID' **in** credentials\_0:

project\_filename = credentials\_0['ASSET\_ID']

**if** project\_filename != 'None' **and** project\_filename != '1':

print('attempting project\_lib access to ' + str(project\_filename))

**try**:

**from** **project\_lib** **import** Project

project = Project.access()

storage\_credentials = project.get\_storage\_metadata()

readable = project.get\_file(project\_filename)

**except** **Exception** **as** project\_exception:

print('unable to access data using the project\_lib interface and filename supplied')

*# Use data\_provenance as filename if other access mechanisms are unsuccessful*

**if** readable **is** **None** **and** type(data\_provenance) **is** str:

print('attempting to access local file using path and name ' + data\_provenance)

readable = data\_provenance

*# Second, use pd.read\_csv to read object, iterating over list of csv\_encodings until successful*

**if** readable **is** **not** **None**:

**for** encoding **in** csv\_encodings:

**try**:

df = pd.read\_csv(readable, encoding=encoding, sep = **None**, engine = 'python')

print('successfully loaded dataframe using encoding = ' + str(encoding))

**break**

**except** **Exception** **as** exception\_csv:

print('unable to read csv using encoding ' + str(encoding))

print('handled error was ' + str(exception\_csv))

**if** df **is** **None**:

print('unable to read file/object as a dataframe using supplied csv\_encodings ' + str(csv\_encodings))

print(f'Please use **\'**insert to code**\'** on data panel to load dataframe.')

**raise**(**ValueError**('unable to read file/object as a dataframe using supplied csv\_encodings ' + str(csv\_encodings)))

**if** isinstance(df,pd.DataFrame):

print('Data loaded succesfully')

In [ ]:

*# Drop rows whose target is not defined*

target = target\_label\_name *# your target name here*

**if** learning\_type == 'regression':

df[target] = pd.to\_numeric(df[target], errors='coerce')

df.dropna('rows', how='any', subset=[target], inplace=**True**)

In [ ]:

preprocessor\_index = -1

preprocessing\_steps = []

**for** i, step **in** enumerate(pipeline.steps):

preprocessing\_steps.append(step)

**if** step[0]=='preprocessor':

preprocessor\_index = i

**break**

*#if len(pipeline.steps) > preprocessor\_index+1 and pipeline.steps[preprocessor\_index + 1][0] == 'cognito':*

*#preprocessor\_index += 1*

*#preprocessing\_steps.append(pipeline.steps[preprocessor\_index])*

**if** preprocessor\_index >= 0:

preprocessing\_pipeline = Pipeline(memory=pipeline.memory, steps=preprocessing\_steps)

pipeline = Pipeline(steps=pipeline.steps[preprocessor\_index+1:])

In [ ]:

*# preprocessor should see all data for cross\_validate on the remaining steps to match autoai scores*

known\_values\_list.clear() *# known\_values\_list is filled in by the preprocessing\_pipeline if needed*

preprocessing\_pipeline.fit(df\_X.values, df\_y.values)

X\_prep = preprocessing\_pipeline.transform(df\_X.values)

In [ ]:

**if** learning\_type **is** **None**:

*# When the problem type is not available in the metadata, use the sklearn type\_of\_target to determine whether to stratify the holdout split*

*# Caution: This can mis-classify regression targets that can be expressed as integers as multiclass, in which case manually override the learning\_type*

**from** **sklearn.utils.multiclass** **import** type\_of\_target

**if** type\_of\_target(df\_y.values) **in** ['multiclass', 'binary']:

learning\_type = 'classification'

**else**:

learning\_type = 'regression'

print('learning\_type determined by type\_of\_target as:',learning\_type)

**else**:

print('learning\_type specified as:',learning\_type)

**from** **sklearn.model\_selection** **import** train\_test\_split

**if** learning\_type == 'classification':

X, X\_holdout, y, y\_holdout = train\_test\_split(X\_prep, df\_y.values, test\_size=holdout\_fraction, random\_state=random\_state, stratify=df\_y.values)

**else**:

X, X\_holdout, y, y\_holdout = train\_test\_split(X\_prep, df\_y.values, test\_size=holdout\_fraction, random\_state=random\_state)

In [ ]:

fe\_pipeline = **None**

**if** pipeline.steps[0][0] == 'cognito':

**try**:

fe\_pipeline = Pipeline(steps=[pipeline.steps[0]])

X = fe\_pipeline.fit\_transform(X, y)

X\_holdout = fe\_pipeline.transform(X\_holdout)

pipeline.steps = pipeline.steps[1:]

**except** **IndexError**:

**try**:

print('Trying to compose pipeline with some of cognito steps')

fe\_pipeline = Pipeline(steps = list([pipeline.steps[0][1].steps[0],pipeline.steps[0][1].steps[1]]))

X = fe\_pipeline.fit\_transform(X, y)

X\_holdout = fe\_pipeline.transform(X\_holdout)

pipeline.steps = pipeline.steps[1:]

**except** **IndexError**:

print('Composing pipeline without cognito steps!')

pipeline.steps = pipeline.steps[1:]

In [ ]:

**def** make\_pos\_label\_scorer(scorer, pos\_label):

kwargs = {'pos\_label':pos\_label}

**for** prop **in** ['needs\_proba', 'needs\_threshold']:

**if** prop+'=True' **in** scorer.\_factory\_args():

kwargs[prop] = **True**

**if** scorer.\_sign == -1:

kwargs['greater\_is\_better'] = **False**

**from** **sklearn.metrics** **import** make\_scorer

scorer=make\_scorer(scorer.\_score\_func, \*\*kwargs)

**return** scorer

In [ ]:

pipeline.fit(X,y)

y\_pred = pipeline.predict(X\_holdout)

In [ ]:

*# cross\_validate pipeline using training data*

**from** **sklearn.model\_selection** **import** cross\_validate

**from** **sklearn.model\_selection** **import** StratifiedKFold, KFold

**if** learning\_type == 'classification':

fold\_generator = StratifiedKFold(n\_splits=cv\_num\_folds, random\_state=random\_state)

**else**:

fold\_generator = KFold(n\_splits=cv\_num\_folds, random\_state=random\_state)

cv\_results = cross\_validate(pipeline, X, y, cv=fold\_generator, scoring={optimization\_metric:scorer}, return\_train\_score=**True**)

**import** **numpy** **as** **np**

np.mean(cv\_results['test\_' + optimization\_metric])

cv\_results

5)Node-RED Flow .Json File: -

Autoai.json

[{"id":"905e88b7.54ba68","type":"tab","label":"Life Expectancy Automated AI","disabled":false,"info":""},{"id":"9e7b99e9.68b9c8","type":"ui\_form","z":"905e88b7.54ba68","name":"","label":"","group":"b894d3e1.38628","order":0,"width":0,"height":0,"options":[{"label":"Country","value":"a","type":"text","required":false,"rows":null},{"label":"Year","value":"b","type":"number","required":true,"rows":null},{"label":"Status","value":"c","type":"text","required":true,"rows":null},{"label":"Adult Mortality","value":"d","type":"number","required":true,"rows":null},{"label":"infant deaths","value":"e","type":"number","required":true,"rows":null},{"label":"alcohol","value":"f","type":"number","required":true,"rows":null},{"label":"percentage expenditure","value":"g","type":"number","required":true,"rows":null},{"label":"Hepatitis B","value":"h","type":"number","required":true,"rows":null},{"label":"Measles","value":"i","type":"number","required":true,"rows":null},{"label":"BMI","value":"j","type":"number","required":true,"rows":null},{"label":"under-five deaths","value":"k","type":"number","required":true,"rows":null},{"label":"Polio","value":"l","type":"number","required":true,"rows":null},{"label":"Total expenditure","value":"m","type":"number","required":true,"rows":null},{"label":"Diphtheria","value":"n","type":"number","required":true,"rows":null},{"label":"HIV/AIDS","value":"o","type":"number","required":true,"rows":null},{"label":"GDP","value":"p","type":"number","required":true,"rows":null},{"label":"Population","value":"q","type":"number","required":true,"rows":null},{"label":"thinness 1-19 years","value":"r","type":"number","required":true,"rows":null},{"label":"thinness 5-9 years","value":"s","type":"number","required":true,"rows":null},{"label":"Income composition of resources","value":"t","type":"number","required":true,"rows":null},{"label":"Schooling","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":"submit","cancel":"cancel","topic":"","x":70,"y":100,"wires":[["ebb5154f.58e9d8"]]},{"id":"ebb5154f.58e9d8","type":"function","z":"905e88b7.54ba68","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=\"6EXQncDn644ci9EmQSIZI15xQntbIVSKtAuqDjq-jTzU\";\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":[["45bb2a67.ba0ec4"]]},{"id":"87f4dfe4.ecde5","type":"http request","z":"905e88b7.54ba68","name":"","method":"POST","ret":"obj","paytoqs":false,"url":"https://eu-gb.ml.cloud.ibm.com/v4/deployments/6ed2053d-2c4b-4194-a719-4bfb84df7873/predictions","tls":"","persist":false,"proxy":"","authType":"","x":470,"y":180,"wires":[["59b8ce5f.77cad"]]},{"id":"f124eb3a.d90a38","type":"debug","z":"905e88b7.54ba68","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":750,"y":280,"wires":[]},{"id":"59b8ce5f.77cad","type":"function","z":"905e88b7.54ba68","name":"getFrom Endpoint","func":"msg.payload=msg.payload.predictions[0].values[0][0];\nreturn msg;","outputs":1,"noerr":0,"x":490,"y":280,"wires":[["f124eb3a.d90a38","829d1793.701b78"]]},{"id":"e5f445f.e9f70b8","type":"function","z":"905e88b7.54ba68","name":"sendTo Endpoint","func":"//get token and make headers\nvar token=msg.payload.access\_token;\nvar instance\_id=\"796b7c04-94db-4d02-b775-6e8dca25dbf5\"\nmsg.headers={'Content-Type': 'application/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.get(\"b\");\nvar c = global.get(\"c\");\nvar d = global.get(\"d\");\nvar e = global.get(\"e\");\n\nvar f = global.get(\"f\");\nvar g = global.get(\"g\");\nvar h = global.get(\"h\");\nvar i = global.get(\"i\");\nvar j = global.get(\"j\");\nvar k = global.get(\"k\");\nvar l = global.get(\"l\");\nvar m = global.get(\"m\");\nvar n = global.get(\"n\");\nvar o = global.get(\"o\");\nvar p = global.get(\"p\");\nvar q = global.get(\"q\");\nvar r = global.get(\"r\");\nvar s = global.get(\"s\");\nvar t = global.get(\"t\");\nvar u = global.get(\"u\");\n\n//send the user values to service endpoint\nmsg.payload = \n{\"input\_data\":[{\"fields\":[\"Country\", \"Year\", \"Status\",\t\n \"Adult Mortality\",\t\"infant deaths\",\n \"Alcohol\",\"percentage expenditure\",\n \"Hepatitis B\", \"Measles\",\"BMI\",\n \"under-five deaths\", \"Polio\",\n \"Total expenditure\", \"Diphtheria\",\t \n \"HIV/AIDS\",\t\"GDP\",\n \"Population\",\"thinness 1-19 years\",\n \"thinness 5-9 years\",\n \"Income composition of resources\",\n \"Schooling\"],\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":[["87f4dfe4.ecde5"]]},{"id":"45bb2a67.ba0ec4","type":"http request","z":"905e88b7.54ba68","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":[["e5f445f.e9f70b8"]]},{"id":"829d1793.701b78","type":"ui\_text","z":"905e88b7.54ba68","group":"b894d3e1.38628","order":1,"width":0,"height":0,"name":"","label":"Prediction","format":"{{msg.payload}}","layout":"row-spread","x":720,"y":400,"wires":[]},{"id":"b1ff778b.4d7d08","type":"ui\_template","z":"905e88b7.54ba68","group":"2619f089.6ac26","name":"","order":0,"width":"6","height":"3","format":"<html>\n <body>\n <i>Different Countries have Different Environment So,We can't assume all countries have same environment.For accuracy, we have to give the country name and it is not mandatory to give as input</i>\n </body>\n</html>","storeOutMessages":true,"fwdInMessages":true,"resendOnRefresh":true,"templateScope":"local","x":260,"y":420,"wires":[[]]},{"id":"b894d3e1.38628","type":"ui\_group","z":"","name":"Predicting using AUTO AI","tab":"d26ce5f5.7ae8e8","order":2,"disp":true,"width":"6","collapse":false},{"id":"2619f089.6ac26","type":"ui\_group","z":"","name":"Alert!!!","tab":"d26ce5f5.7ae8e8","order":1,"disp":true,"width":"6","collapse":false},{"id":"d26ce5f5.7ae8e8","type":"ui\_tab","z":"","name":"Predicting Life Expectancy Using Machine Learning","icon":"dashboard","disabled":false,"hidden":false}]

***Overview of Internship Experience***

During 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.