REPORT

PREDICTING LIFE EXPECTANCY USING MACHINE LEARNING

DEEKSHYA DASH

1. INTRODUCTION:

1.1. OVERVIEW

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 in a dataset.

A Supervised Machine learning Regression algorithm with maximum accuracy which is trained and tested on the dataset works as the base model.The Project requires in depth knowledge of IBM Services. The User Interface is built on Node-Red which is an IBM Application and the backend uses Machine Learning Algorithm which is a typical Regression Model.

Software Requirements: IBM Cloud ,IBM Watson Studio,Node-red .

Project delivers a user interface which works on machine learning to predict the life expectancy by taking an input dataset consisting of all the various attributes that affect the model and observations to train it and provide accurate results.

1.2. PURPOSE

The purpose is to provide 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:

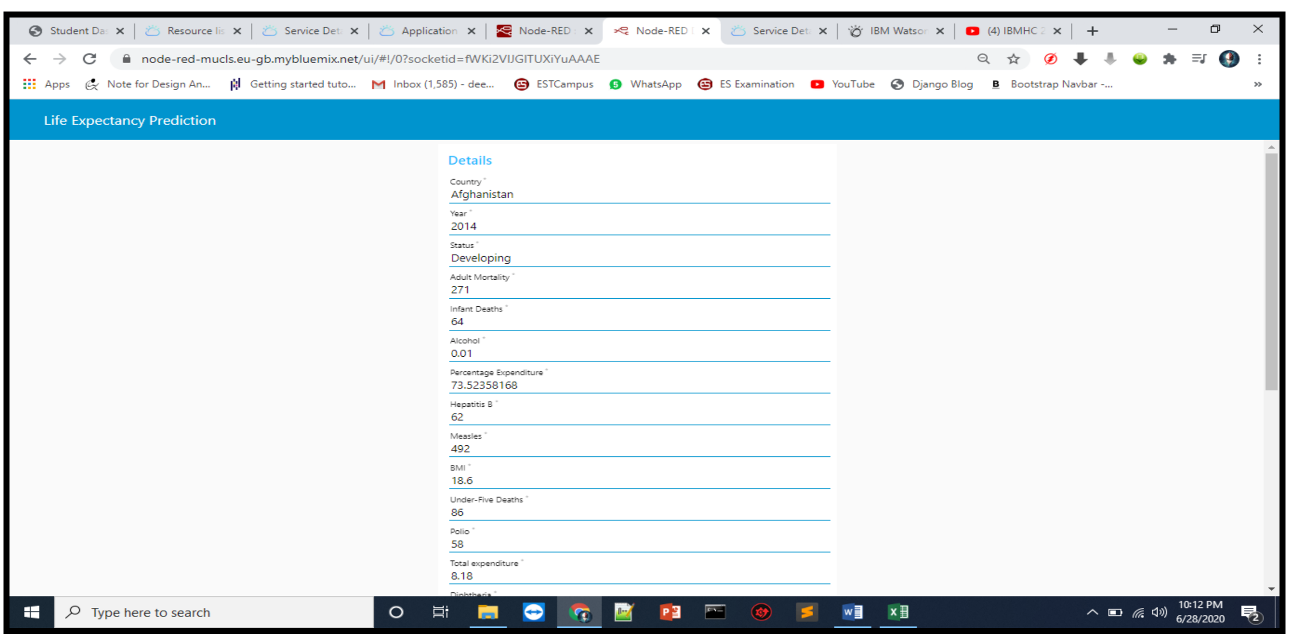
This problem statement is aimed at predicting Life Expectancy rate of a country given various features that can help determine it in the best way possible.

2.2. PROPOSED SOLUTION:

The solution encourages the creation of a regression model which, by taking various features from the dataset, into consideration gives the best possible algorithm to predict the Life Expectancy accurately.

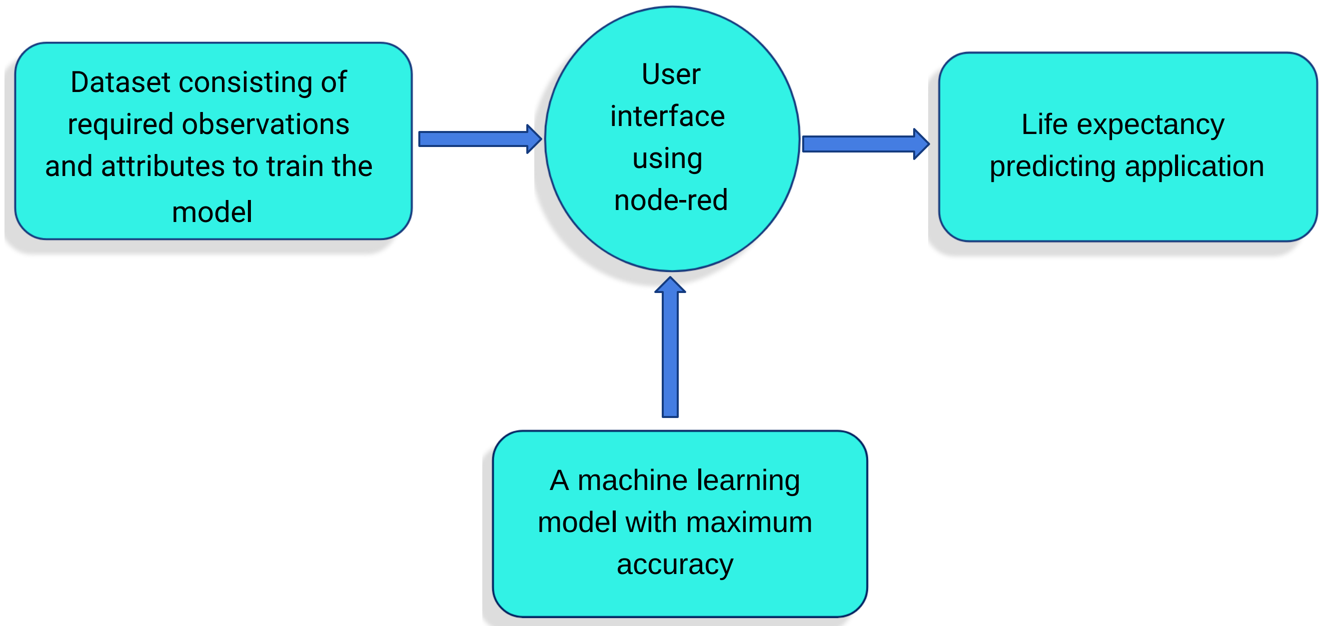
The solution is predicted by forming models using both python and Auto AI, made with the help of various Watson Services available on the IBM Cloud platform. After proper training, testing and deployment of the models, a User Interface is made on node-red which takes in the input , uses the proposed model in the backend and provides the output there itself.

THE UI



3. THEORETICAL ANALYSIS:

3.1. BLOCK DIAGRAM:



3.2. HARDWARE/SOFTWARE DESIGNING:

Steps required for software designing:

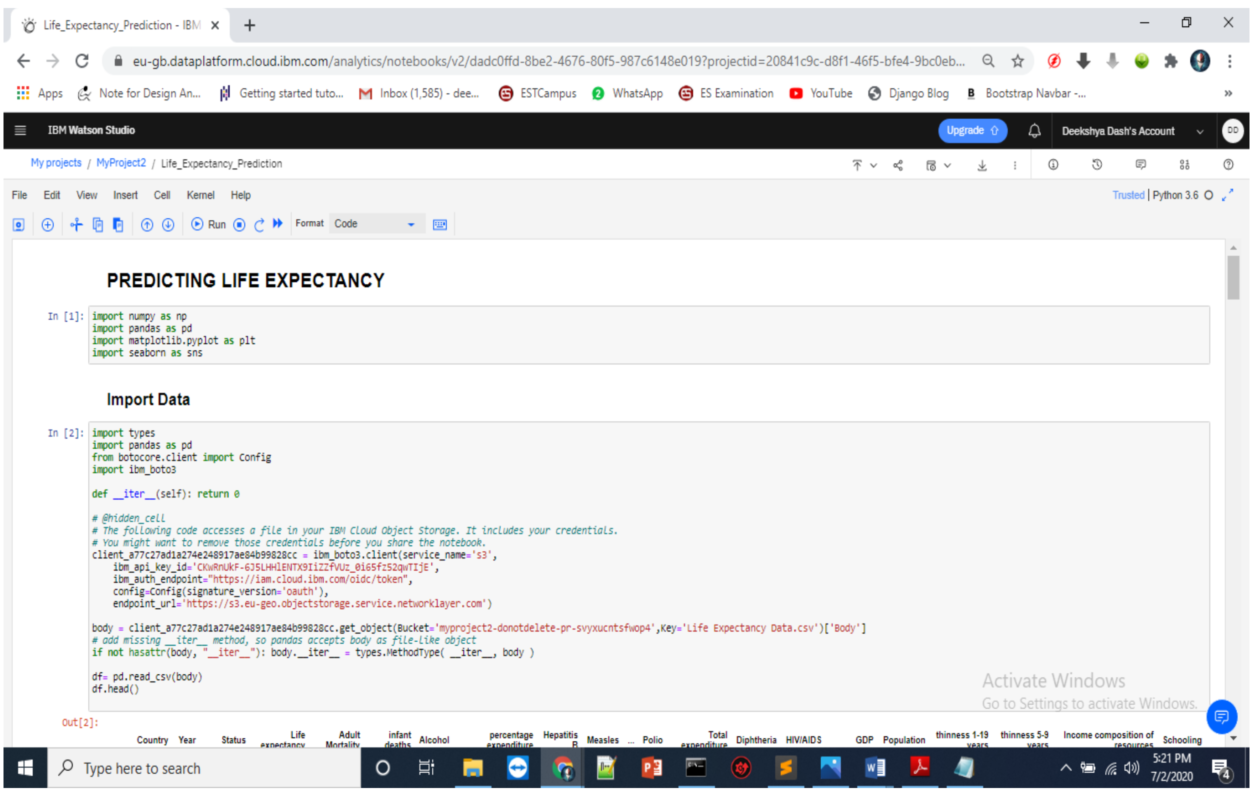
- Exploration of Cloud Platform
- Exploration of Watson Services
- Building a Machine Learning model on Jupyter
- Expermenting on Auto AI
- User Interface on Node-RED

4. EXPERIMENTAL INVESTIGATIONS:

The experiment comprised of two methods,

- **with Python :**
    - The Dataset was downloaded from Kaggle site.
    - A Linear Regression model helped in finding the R^2 and RMSE values that determined Life Expectancy.
    - The model was trained and tested before deployment.
    - The respective values of R^2 and RMSE resulted to be 0.8190807877191667 and 4.111825564792904 accurately.

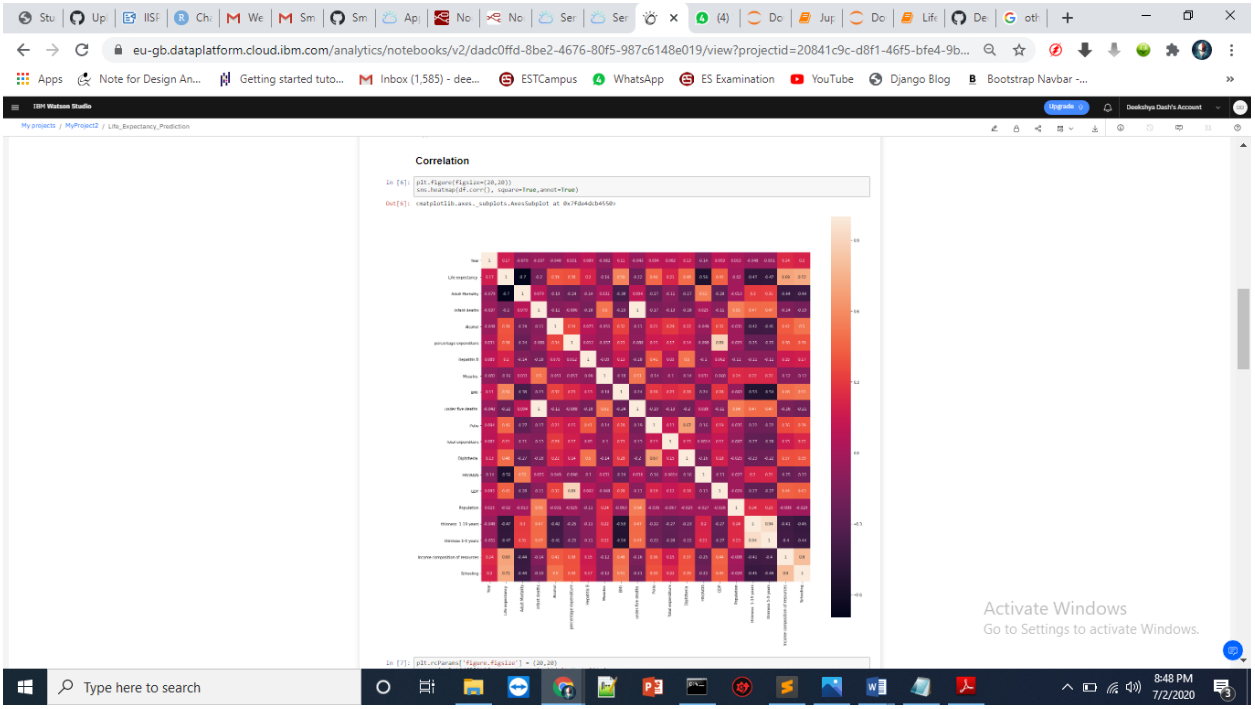
Notebook:



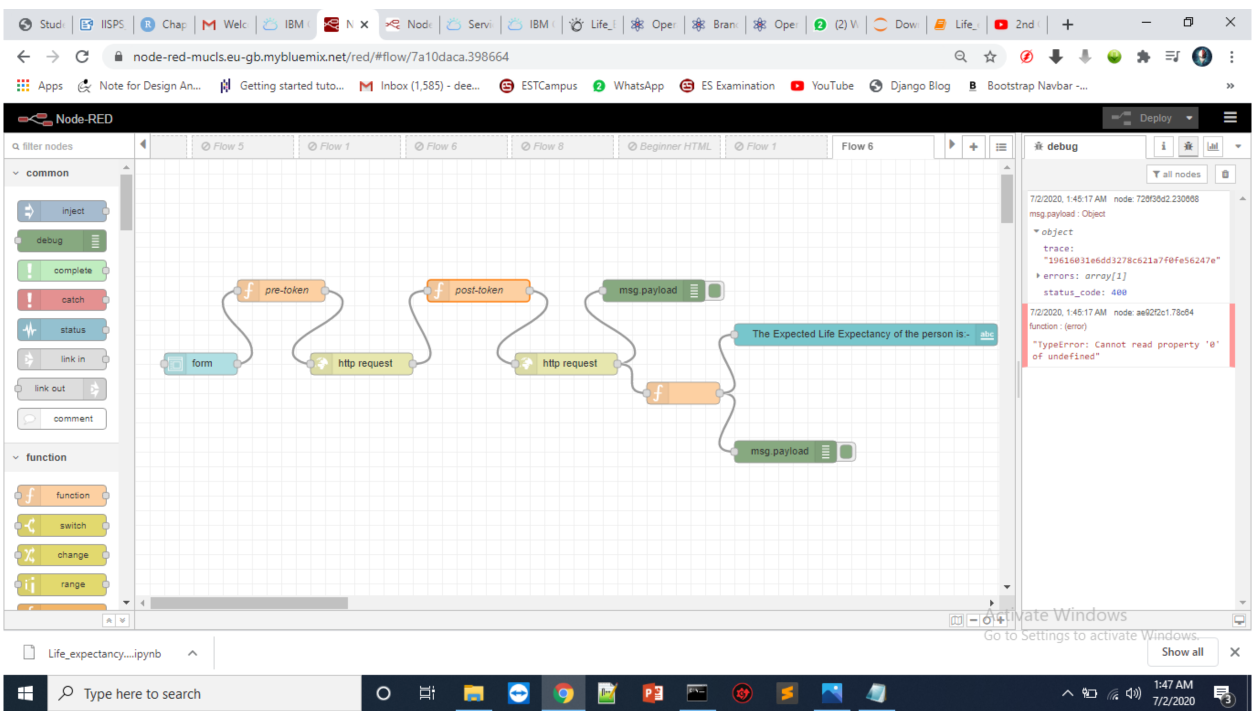
the link:

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/dadc0ffd-8be2-4676-80f5-987c6148e019/view?access_token=d74a48e8eddb164d6ec2a8a2a62477e731bcf283ef43c427f3bc2ab9a5a43d15

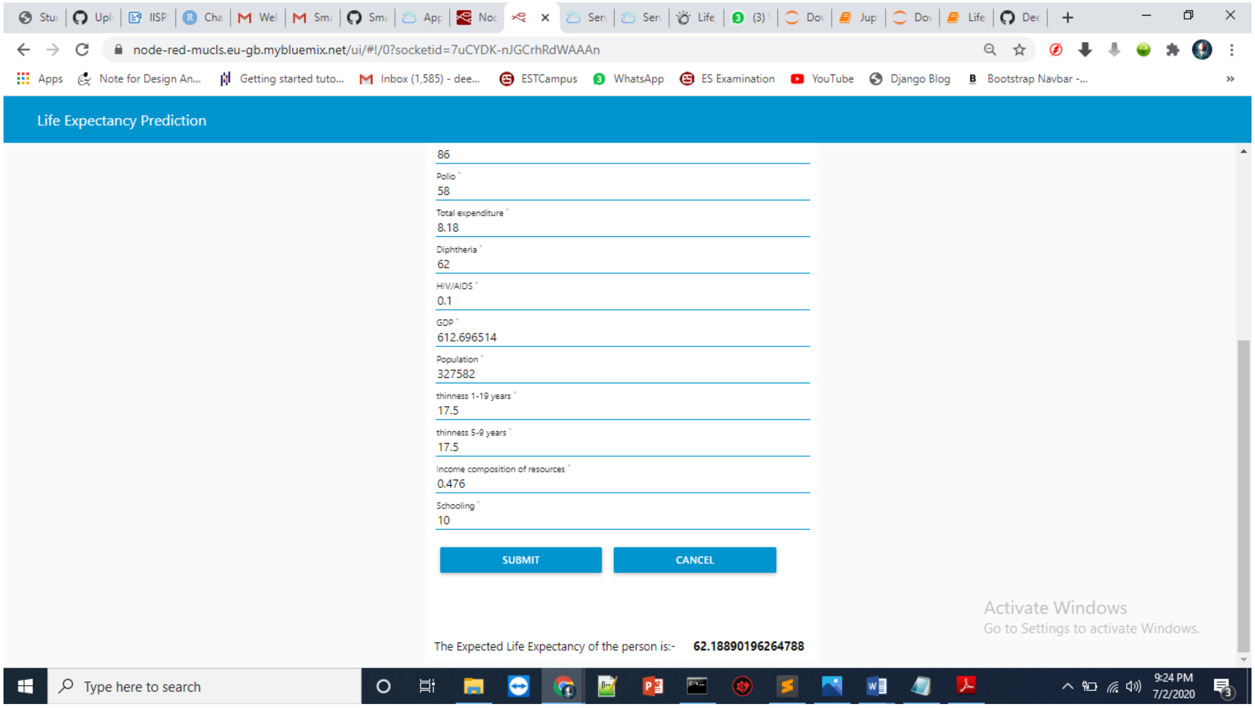
Correlation:



Node-RED flow:

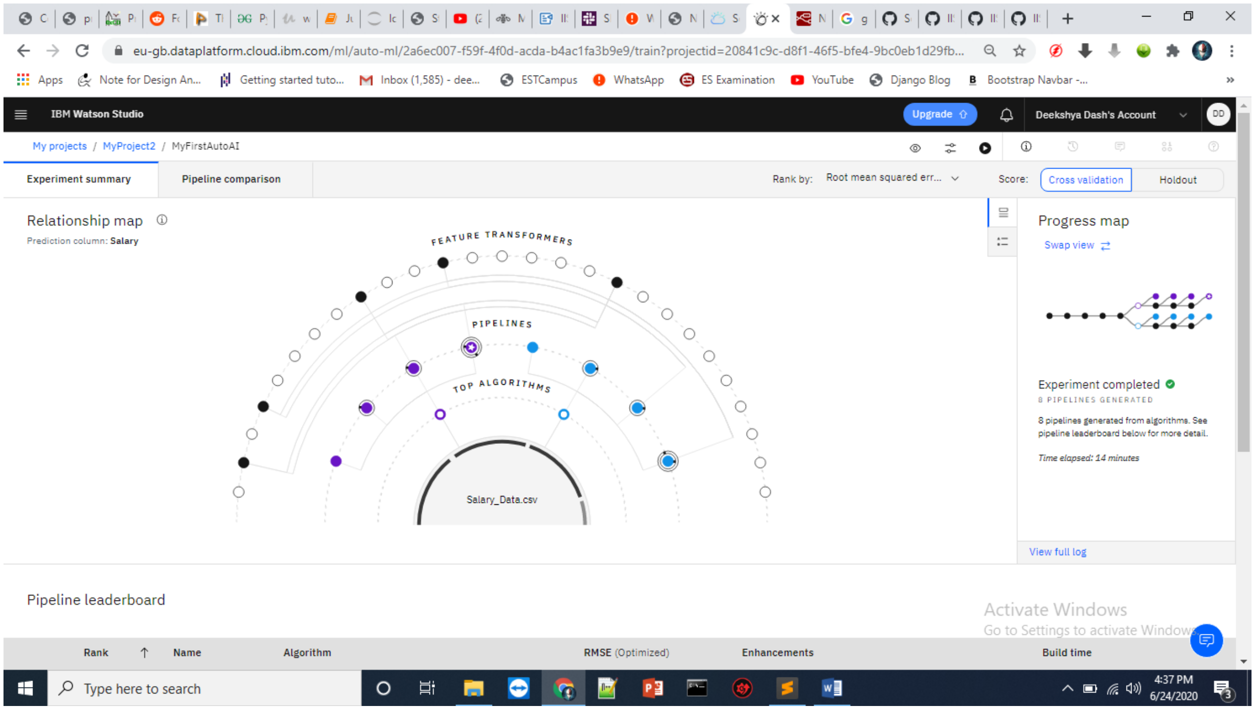


User-Interface:

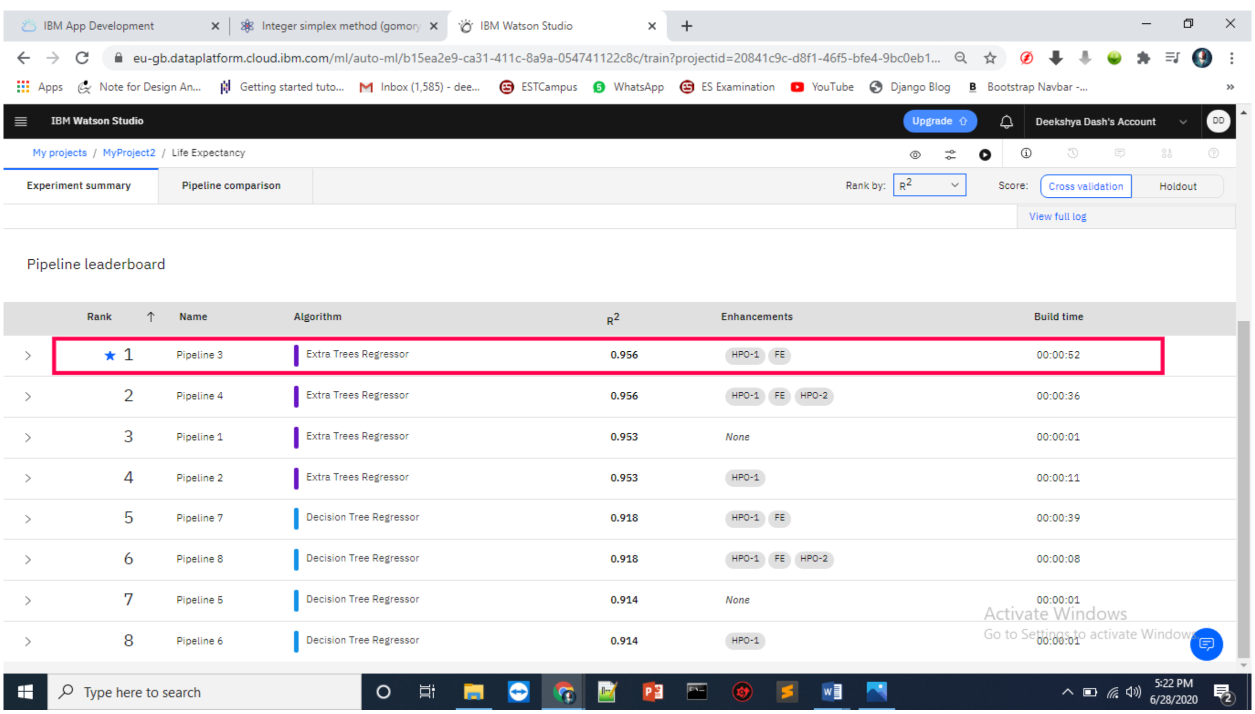


- **without Python:**
  - The downloaded dataset is uploaded on the Watson Studio
  - The Auto AI experiment chooses the best algorithm to pedict the column that we want.
  - The best model is saved after running the experiment and deployed.
  - The Extra Tree Regressor best explained the model with R^2 of 0.95 which means 95% of variance.

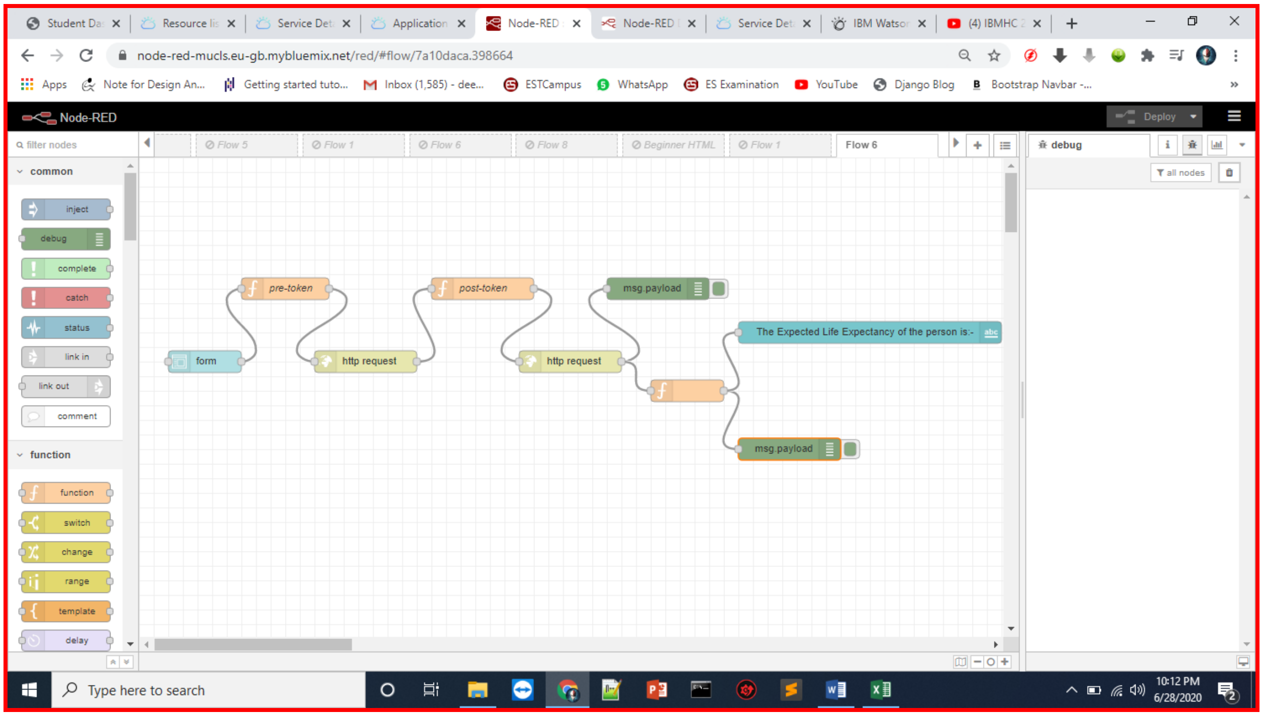
Auto AI Experiment:



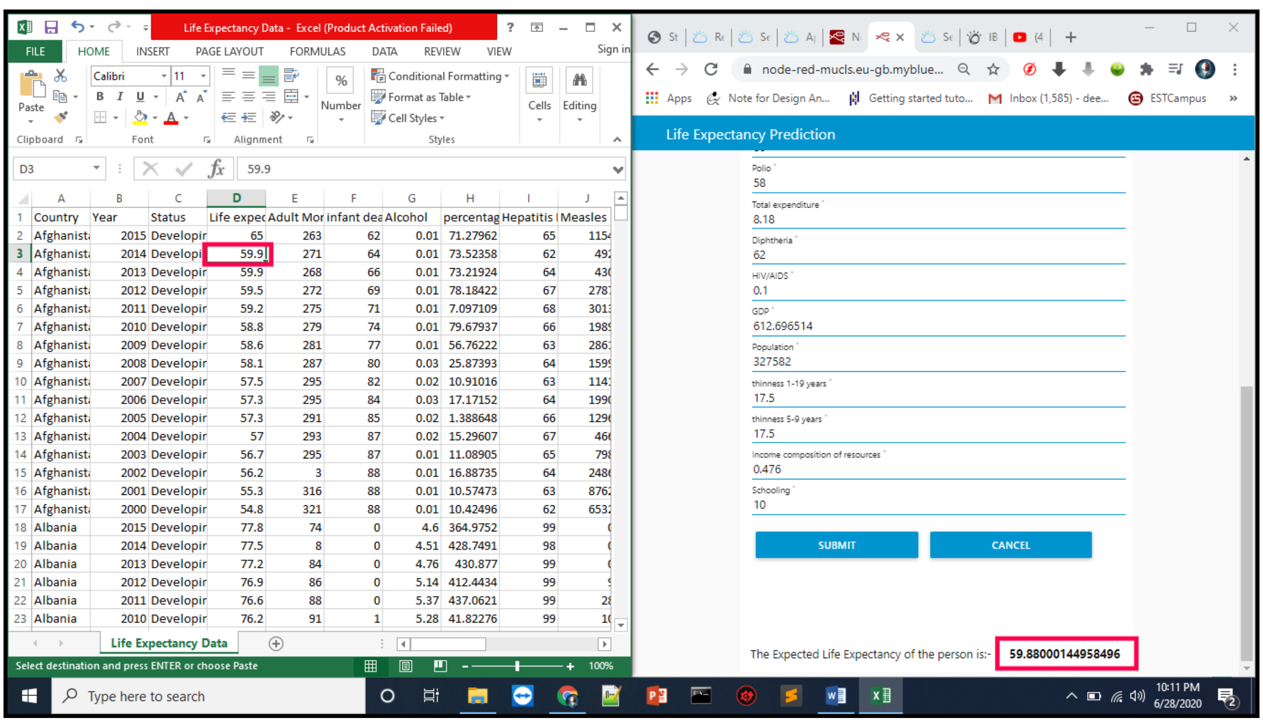
Best chosen model:



Node-red flow to integrate Auto AI:

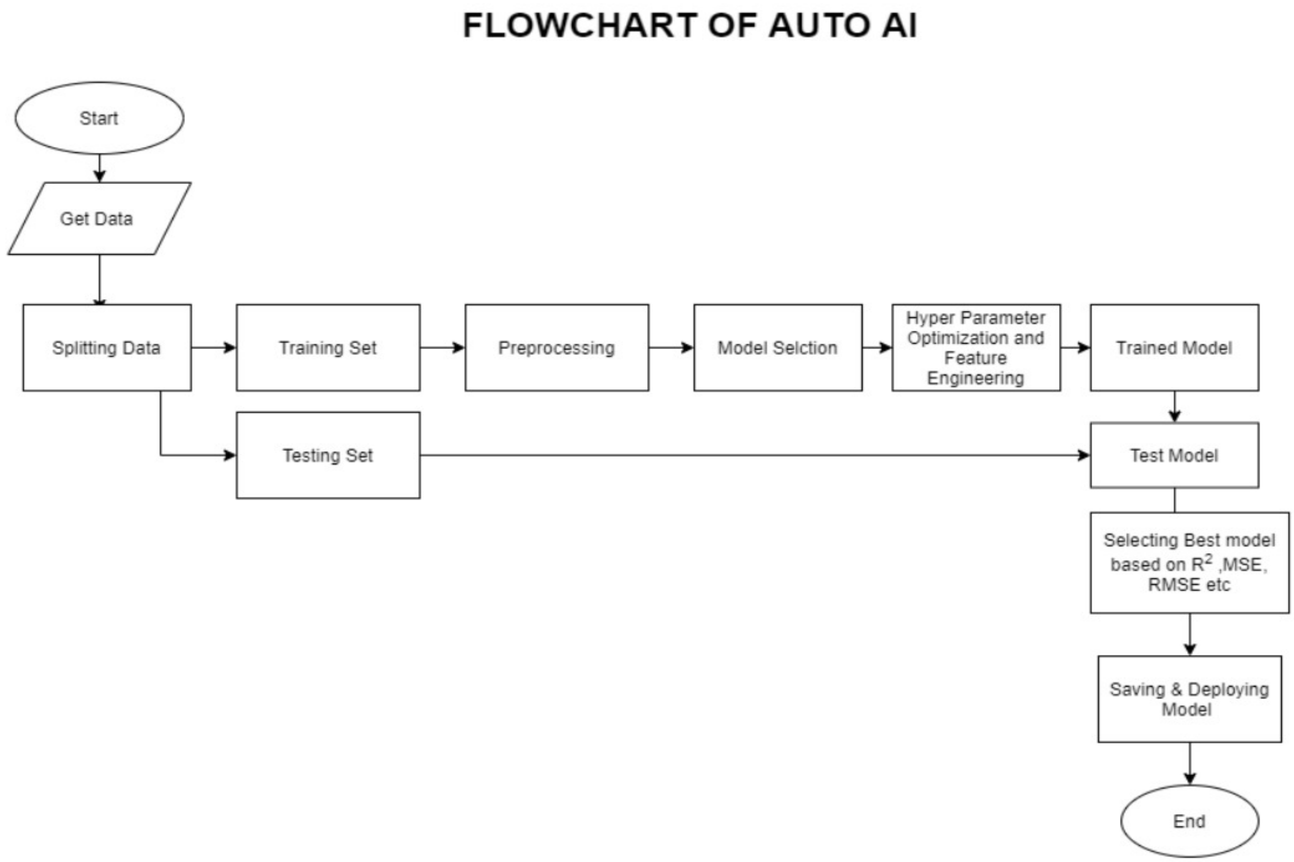


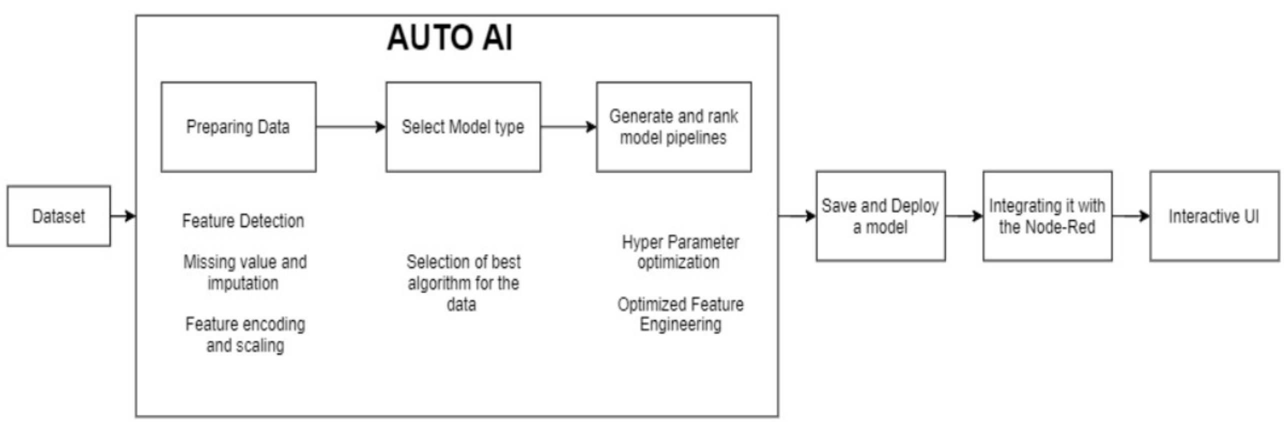
User-Interface:



Comparision between given value of dataset and predicted life expectancy value shown in the UI.

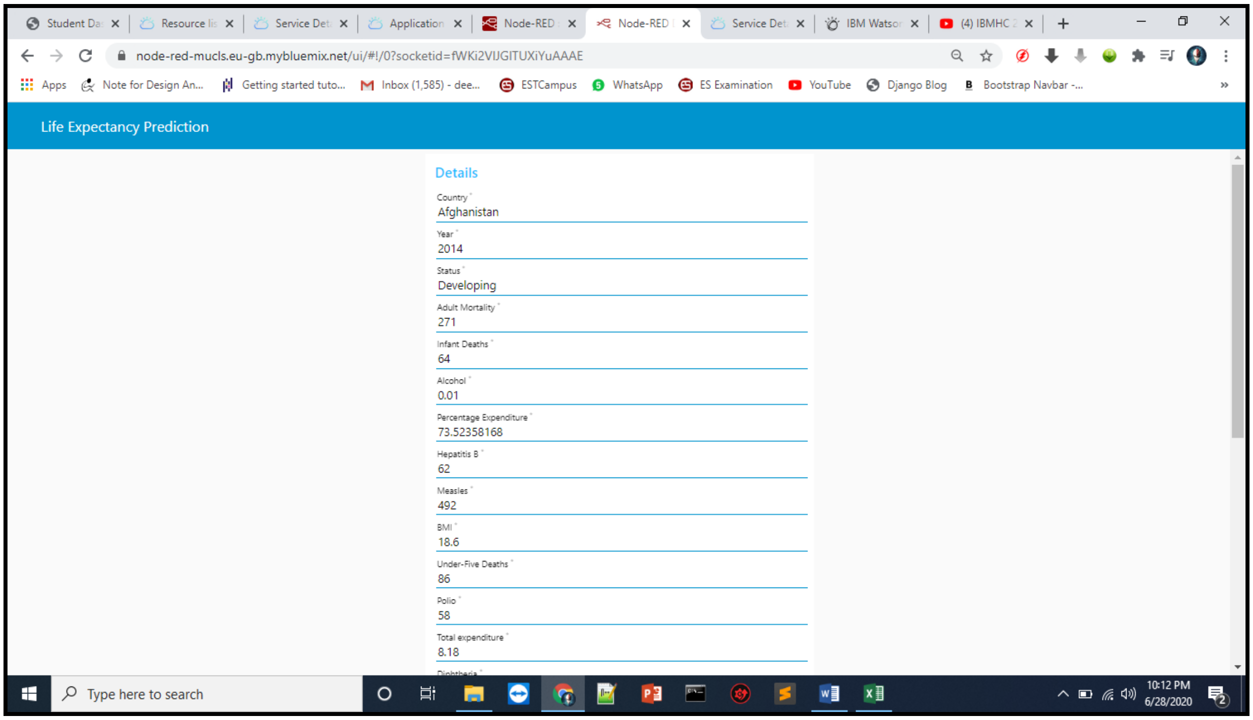
5. FLOWCHART:

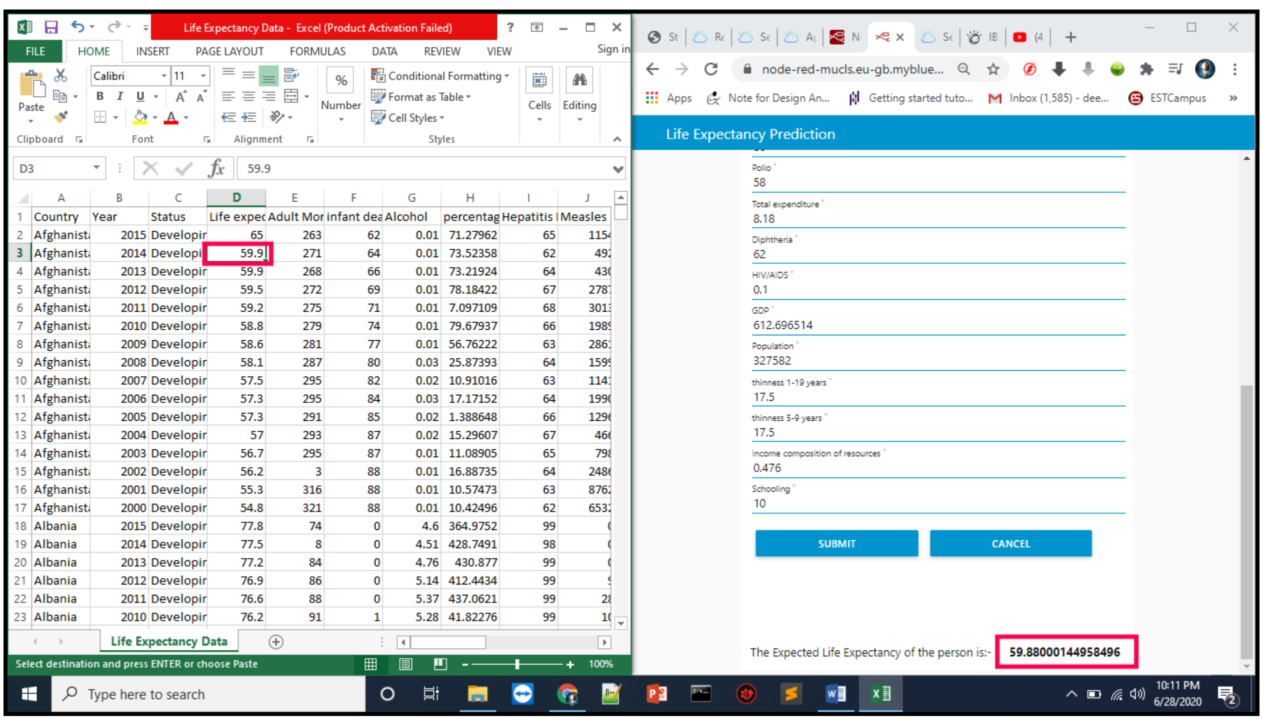

FLOWCHART OF AUTO AI
Start
Get Data
Splitting Data
Training Set
Preprocessing
Model Selction
Hyper Parameter Optimization and Feature Engineering
Trained Model
Testing Set
Test Model
Selecting Best model based on R2 ,MSE, RMSE etc
Saving & Deploying Model
End


AUTO AI
Dataset
Preparing Data
Select Model type
Generate and rank model pipelines
Feature Detection
Missing value and imputation
Feature encoding and scaling
Selection of best algorithm for the data
Hyper Parameter optimization
Optimized Feature Engineering
Save and Deploy a model
Integrating it with the Node-Red
Interactive UI

6. RESULT:

Data given to the User Interface gives the predicted Life Expectancy as result.





7. ADVANTAGES AND DISADVANTAGES:

ADVANTAGES:

- **Monitor Health Inequalities**: Life expectancy has been used nationally to monitor health inequalities of a country.
- **Reduced Costs:** This is a simple webpage and can be accessed by any citizen of a country to calculate life expectancy of their country and doesnot required any kind of payment neither for designing nor for using.
- **User Friendly Interface:** This interface requires no background knowledge of how to use it. It’s a simple interface and only ask for required values and predict the output.

DISADVANTAGES:

- **Wrong Prediction:** As it depends completely on user, so if user provides some wrong values then it will predict wrong value.
- **Average Prediction:** The model predicts average or approximate value with 95% accuracy but not accurate value.

8. APPLICATIONS:

a) It can be used to monitor health inequalities of a country. Used in health industries.
b) It can be used to develop statistics for country development process.
c) It can be used to analyse the factors for high life expectancy.
d) It is user friendly and can be used by anyone.

9. CONCLUSION:

This user interface will be useful for the user to predict life expectancy value of their own country or any other country based on some required details such as GDP, BMI, Year, Alcohol Intake, Total expenditure and etc.

10. FUTURE SCOPE:

Future Scope of the Model can be:

a) **Advanced Features:**

The UI asks for various data which can be difficult for a normal user to gather so I have decided to do some kind of feature modification which can help them to understand each feature and provide data for the same which may appear more user friendly.

b) **More Interactive UI:**

It is a simple webpage only asking inputs and predict output. In future I have decided to make it more user friendly by providing some useful information about the country in the webpage itself so that user does not need to do any kind of prior research for the values.

c) Integrating with services such as speech recognition.

11. BIBLIOGRAPHY:

https://www.youtube.com/watch?v=LOCkV-mENq8&feature=youtu.be
https://github.com/
https://slack.com/intl/en-in/
https://cloud.ibm.com/login
https://www.ibm.com/cloud/get-started
https://github.com/watson-developer-cloud/node-red-labs
https://www.youtube.com/watch?v=NmdjtezQMSM
https://developer.ibm.com/tutorials/watson-studio-auto-ai/
https://www.kaggle.com/kumarajarshi/life-expectancy-who
https://www.youtube.com/watch?v=-CUi8GezG1I&list=PLzpeuWUENMK2PYtasCaKK4bZjaYzhW23L&index=2
https://bookdown.org/caoying4work/watsonstudio-workshop/jn.html#deploy-model-as-web-service
https://bookdown.org/caoying4work/watsonstudio-workshop/auto.html#add-asset-as-auto-ai
https://www.youtube.com/watch?v=Tv_5DHwlYdE&list=PLjlJJFiCdXMInlWHEsgsnY3P55kGdSDh_&index=8&t=0s

12. APPENDIX:

Source code:

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

df= pd.read_csv(body)
df.head()
df.isnull().sum()

#FILL NULL VALUES TO AVOID TRAIN AND TEST ERROR
df=df.fillna(df.mean())
df.isnull().sum()

plt.figure(figsize=(20,20))
sns.heatmap(df.corr(), square=True,annot=True)

plt.rcParams['figure.figsize'] = (20,20)
sns.pairplot(df[['Life expectancy ', 'Adult Mortality',
       'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B',
       'Measles ']])

# deleting the non numeric values
df = df.drop(['Country','Year','Status'], axis=1)
df.head()

# labels(y) and data(X_all)
y = df['Life expectancy '].values
X_all = df.drop(['Life expectancy '], axis=1).values
```

```
# splitting the data to train and test parts
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_all, y, test_size=0.3,
random_state=42)
from sklearn.linear_model import LinearRegression
# create the model
model = LinearRegression()
# fitting the model to the train data
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
# accuracy
from sklearn.metrics import mean_squared_error
r2 = model.score(X_test, y_test)
rmse = np.sqrt(mean_squared_error(y_pred, y_test))
r2
rmse
from watson_machine_learning_client import
WatsonMachineLearningAPIClient
client = WatsonMachineLearningAPIClient(wml_credentials)
metadata = {
    client.repository.ModelMetaNames.AUTHOR_NAME : '-----------',
    client.repository.ModelMetaNames.AUTHOR_EMAIL : '-----------',
    client.repository.ModelMetaNames.NAME : '------------'
}
stored_data = client.repository.store_model(model, meta_props=metadata)
stored_data
guid = client.repository.get_model_uid(stored_data)
guid
deploy = client.deployments.create(guid)
scoring_endpoints = client.deployments.get_scoring_url(deploy)
scoring_endpoints
```