A

Project Report On

Predicting Life Expectancy using

Machine Learning

Internship under :

TheSMARTBRIDGE

NAME : Aashay Shah

EMAIL : ashu25093@gmail.com

PROJECT ID : SPS_PRO_215

INTERNSHIP TITLE : Predicting Life Expectancy using
Machine Learning - SB20922

CATEGORY : Machine Learning

Table Of Contents

1. **Introduction**

1.1 Overview

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

This project "Predicting Life Expectancy using Machine Learning" is an web application that predict the expected average life span of people of a given country based on various features. This project is built using IBM services(Watson studio, Node Red, Watson machine learning)

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.

- Project Requirements : Cloud for deploying projects
  Watson studio for building AI projects
- Functional Requirements : high performance, accurate result, efficiency, platform independent
- Technical Requirements : knowledge of python, IBM watson, IBM cloud
- Software Requirements : Watson studio, IBM cloud, Slack, Zoho
- Project Deliverables : Smartinternz Internship
- Project Team : Aashay Shah
- Project Duration : 28 days

1.2 Purpose

Life expectancy is the most important factor for decision making.Predicting the life expectancy will give the country an idea of the factors which can be improved to increase the lifespan of the people living, like by improving the health care facilities or immunization vaccines for infants.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.

2. Literature Survey

2.1 Existing Problem

- In our existing system we have identified that the factors used for predicting were just personal causes and not related to the surrounding, healthcare facilities, demographic, social, regional and economical factors of the country he resides. These country dependent factors can also be an important feature to predict the life expectancy of an individual. So we need more data to predict more accurately.

In our regular prediction system there are many problems exists such as :

- Health related disease
- Occupational or social class, area level deprivation, geographical area of residence (urban and rural), housing tenure
- Race-based inequalities.

Although there have been lot of studies undertaken in the past on factors affecting life expectancy considering demographic variables, income composition and mortality rates. It was found that effect of immunization and human development index was not taken into account in the past. Also, some of the past research was done considering multiple linear regression based on data set of one year for all the countries

2.1 Proposed Solution

To get better insights and predict the life expectancy more accurately, we need to consider some additional features such as country or surrounding dependent features. The previous factors were more human based but it is important to know the economical, regional, social and demographic factors like the GDP, population, education, immunizations, history of illness, health care facility, funds allocated by the government, schemes, medical expenditures like if it is very high then people will shy away to get regular medical checkups, and many more.

- For the above problem to get solved we have a dataset consist of various factors .In this system we have taken all the correlated features into consideration. So the target output variable i.e expected life span of the people depends upon variety of factors and not factors of particular fields.

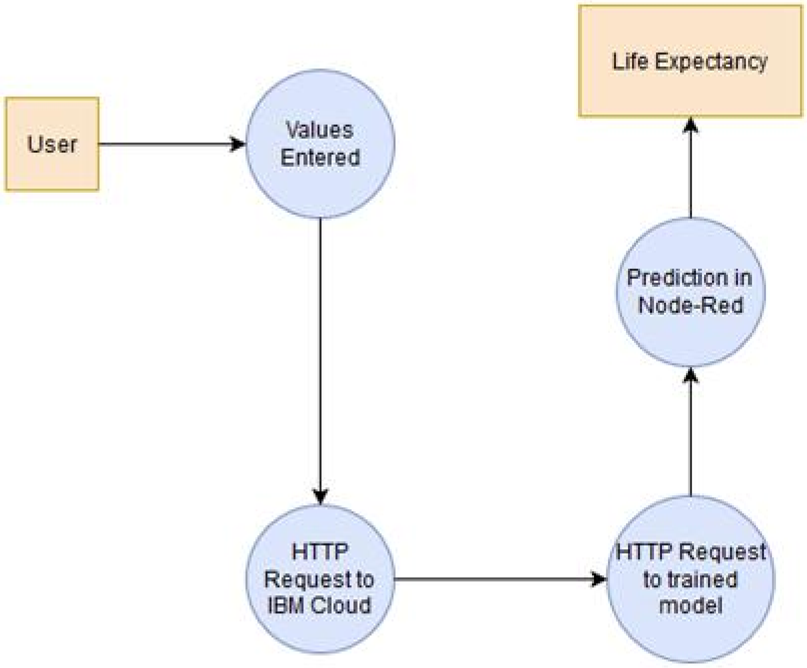
- The project uses immunization factors, mortality factors, economic factors, social factors and other health related factors to predict life expectancy of a country for a given year using a machine learning model.

- Important immunization like Hepatitis B, Polio and Diphtheria are also considered.

- The data-set related to life expectancy, health factors for 193 countries has been collected from 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. 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.

3. Theoritical Analysis

3.1 Block/Flow Diagram :



3.2 Hardware/Software designing :

1.Create necessary IBM Cloud services

2.Create Watson studio project

3.Configure Watson Studio

4.Create IBM Machine Learning instance

5.Create machine learning model in Jupyter notebook

6.Deploy the machine learning model

7.Create flow and configure node

8.Integrate node red with machine learning model

9.Deploy and run Node Red app.

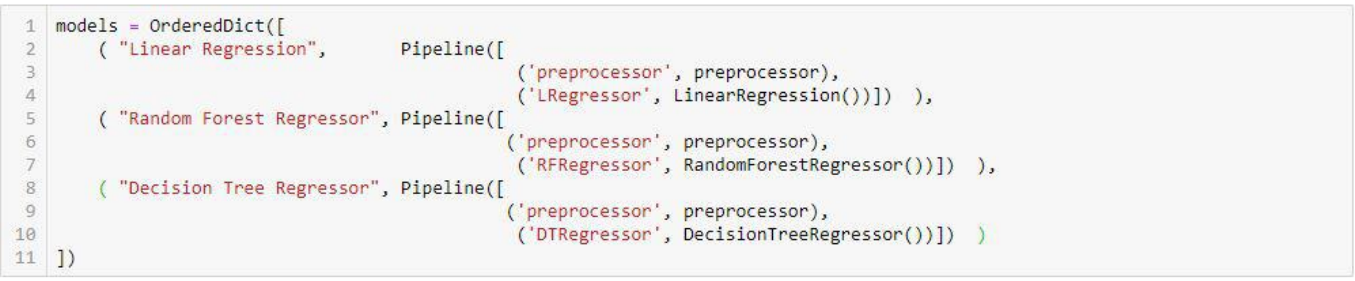
First of all, features are taken as an input by the user in the form fields. After submitting the HTTP request has been sent to the IBM cloud, which will send further to the deployment model using instance id. After successful verification of instance id, result will be predicted by our model and is sent to the Node-RED application. It will parse the output and displays it on the screen.

4. Experimental Investigations

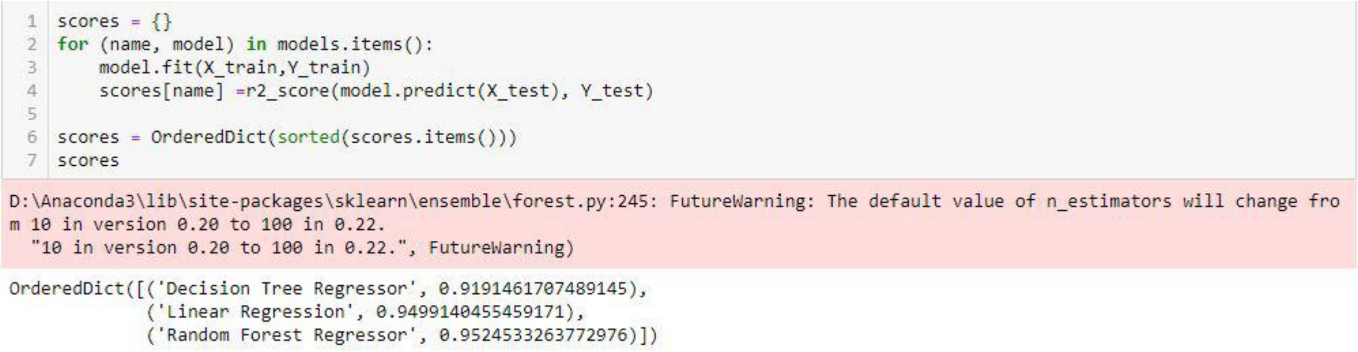
Following factors are taken into consideration for predicting the life expectancy of a country.

1. Country : For which country we want the prediction.
2. Status : Whether the county falls under Developing or Developed category.
3. Year : For which year we want the prediction.
4. Adult Mortality : Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population).
5. Infant deaths: Number of Infant Deaths per 1000 population.
6. Alcohol : Consumption in liters per capita.
7. Percentage Expenditure : Expenditure on health as a percentage of GDP per capita.
8. Measles : Measles - number of reported cases per 100 population.
9. BMI : Average Body Mass Index of entire population.
10. Under-five deaths :Number of under-five deaths per 1000 population,
11. Polio : Polio immunization coverage among 1 year old (%).
12. Diphtheria : Diphtheria tetanus toxoid and pertussis immunization coverage among 1 year old (%).
13. HIV/AIDS : Deaths per 1000 live births HIV/AIDS (0-4 years)
14. GDP : GDP per capita in USD.
15. Thinness 1-19 years : Prevalence of thinness among children and adolescents for Age 10 to 19 (%).
16. Income Composition of Resources : Human development index in terms of income composition of resources (index range 0 to 1).
17. Schooling : Number of years of schooling.

Finding the best suitable Algorithm :



```
models = OrderedDict([
    ( "Linear Regression",       Pipeline([
                                        ('preprocessor', preprocessor),
                                        ('LRegressor', LinearRegression())])  ),
    ( "Random Forest Regressor", Pipeline([
                                       ('preprocessor', preprocessor),
                                        ('RFRegressor', RandomForestRegressor())])  ),
    ( "Decision Tree Regressor", Pipeline([
                                       ('preprocessor', preprocessor),
                                        ('DTRegressor', DecisionTreeRegressor())])  )
])
```



```
scores = {}
for (name, model) in models.items():
    model.fit(X_train,Y_train)
    scores[name] =r2_score(model.predict(X_test), Y_test)

scores = OrderedDict(sorted(scores.items()))
scores
```

```
D:\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_estimators will change fro
m 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

```
OrderedDict([('Decision Tree Regressor', 0.9191461707489145),
             ('Linear Regression', 0.9499140455459171),
             ('Random Forest Regressor', 0.9524533263772976)])
```

Here Random Forest gives best performance.

Step by step Process :

Create IBM Cloud services

- Watson Studio
- Watson Machine Learning
- Node Red

- **Create Watson Studio service instance.**
  - ✓ Select Catalog found at the top right of the page.
  - ✓ Click on Watson from the menu on the left, which you can find under Platform services.
  - ✓ Select Watson Studio.
  - ✓ Enter the Service name or keep the default value and make sure to select the US South as the region/location and your desired organization, and space.
  - ✓ Select Lite for the Plan, which you can find under Pricing Plans and is already selected. Please note you are only allowed one instance of a Lite plan per service.

- ✓ Click on Create.
- ✓ You will be taken to the main page of the service. Click on Get Started.
- ✓ Create a New Project

- **Add WML service**
  - ✓ Click on the Settings in the project view, locate Associated services => Add Service => Watson.
  - ✓ You should also create a Access Token in the project setting. Click on New token, give it a name, then click Create.

- **Create Notebook**

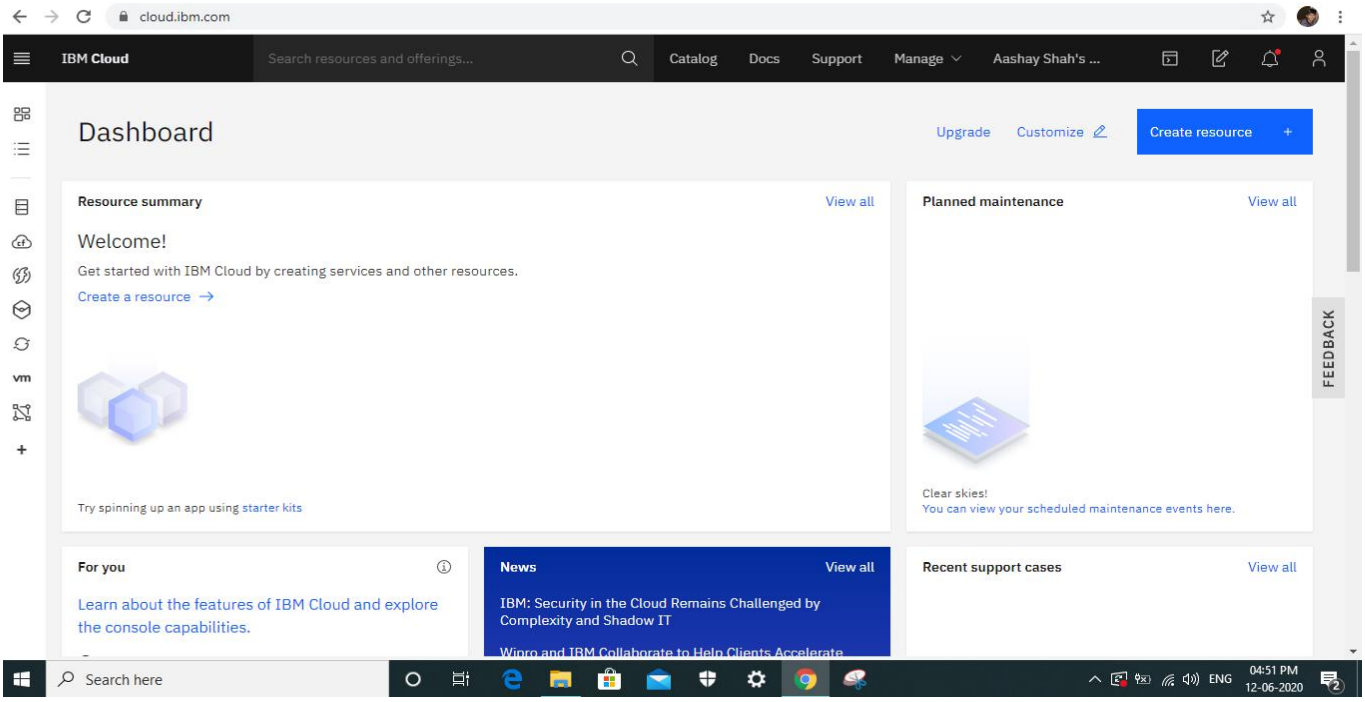
  Click Add to project => Notebook

  And create your Model here.

- **Deploy Model as a Service.**
- **Build Node-RED Flow To Integrate ML Services.**

Screenshots

IBM Cloud Dashboard :



Welcome!

Get started with IBM Cloud by creating services and other resources.

Create a resource →

Try spinning up an app using starter kits

Clear skies!

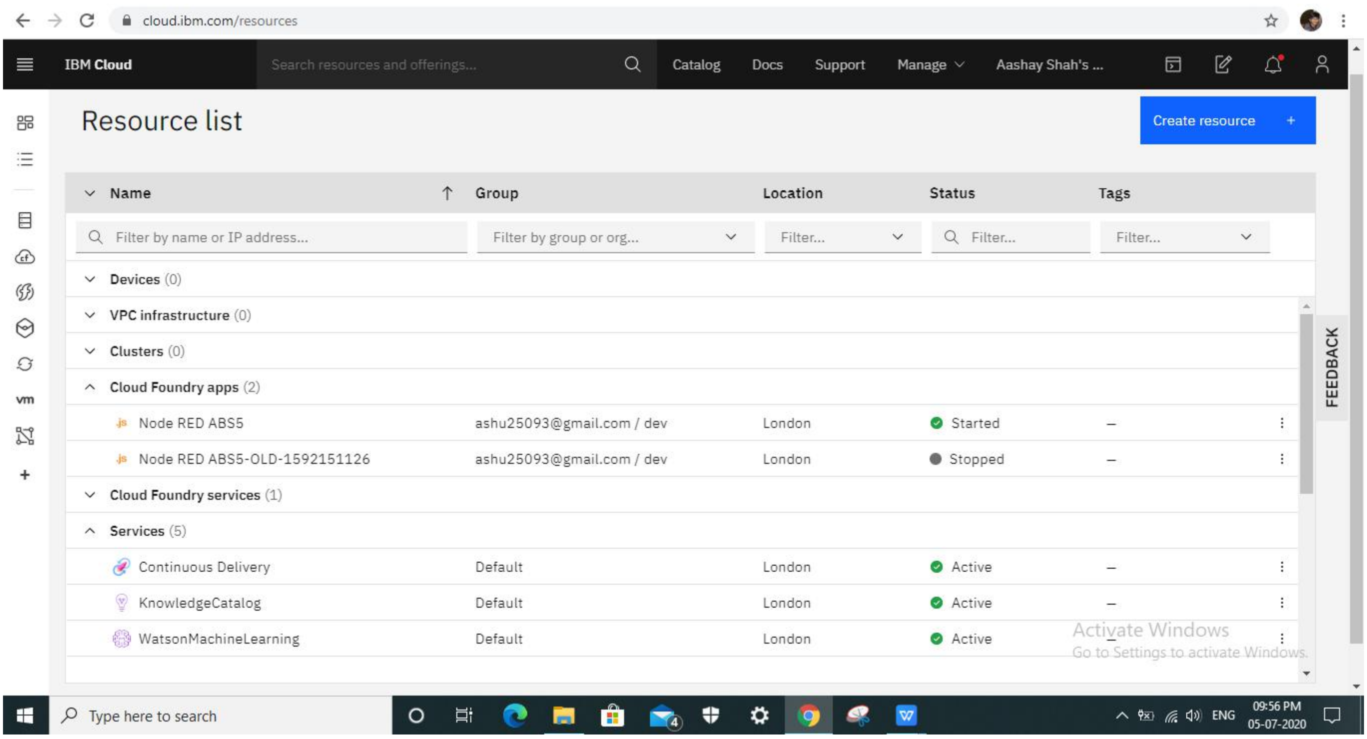
You can view your scheduled maintenance events here.

Learn about the features of IBM Cloud and explore the console capabilities.

IBM: Security in the Cloud Remains Challenged by Complexity and Shadow IT

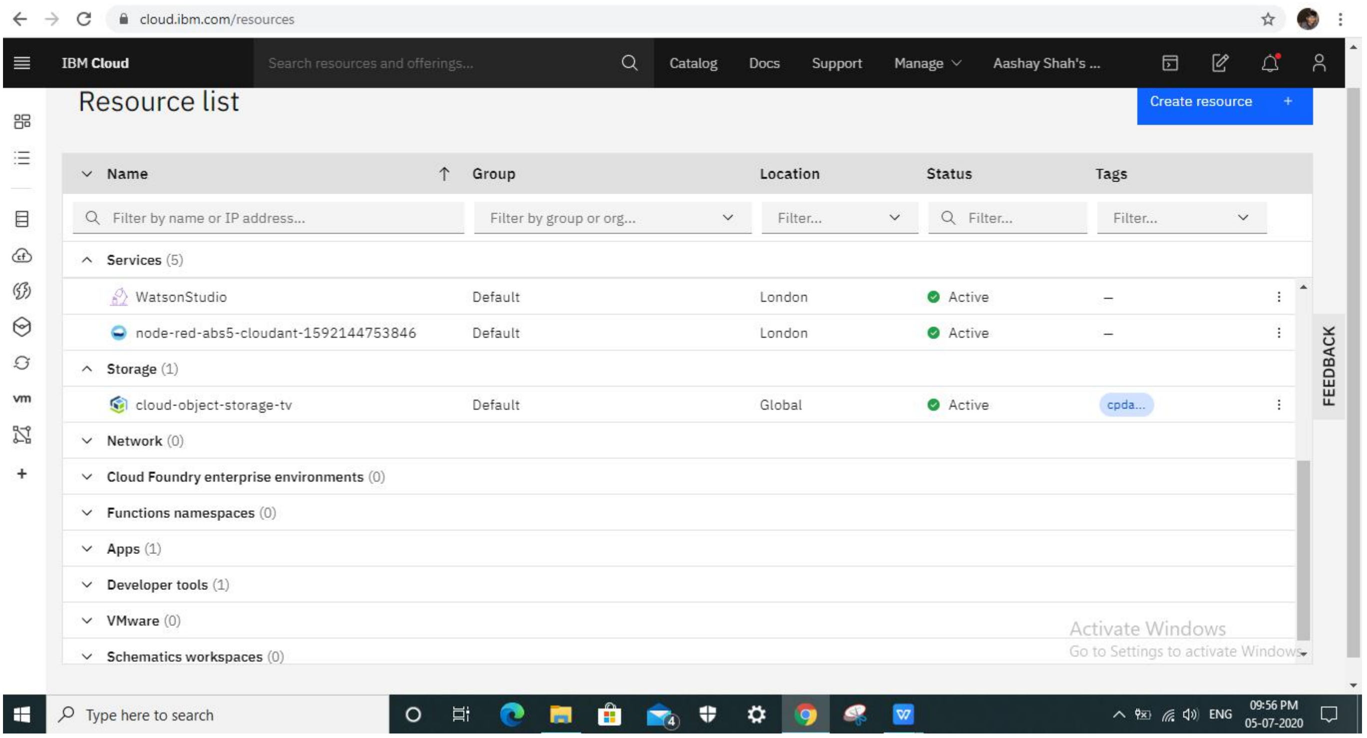
Wipro and IBM Collaborate to Help Clients Accelerate

Resource List :



Resource list

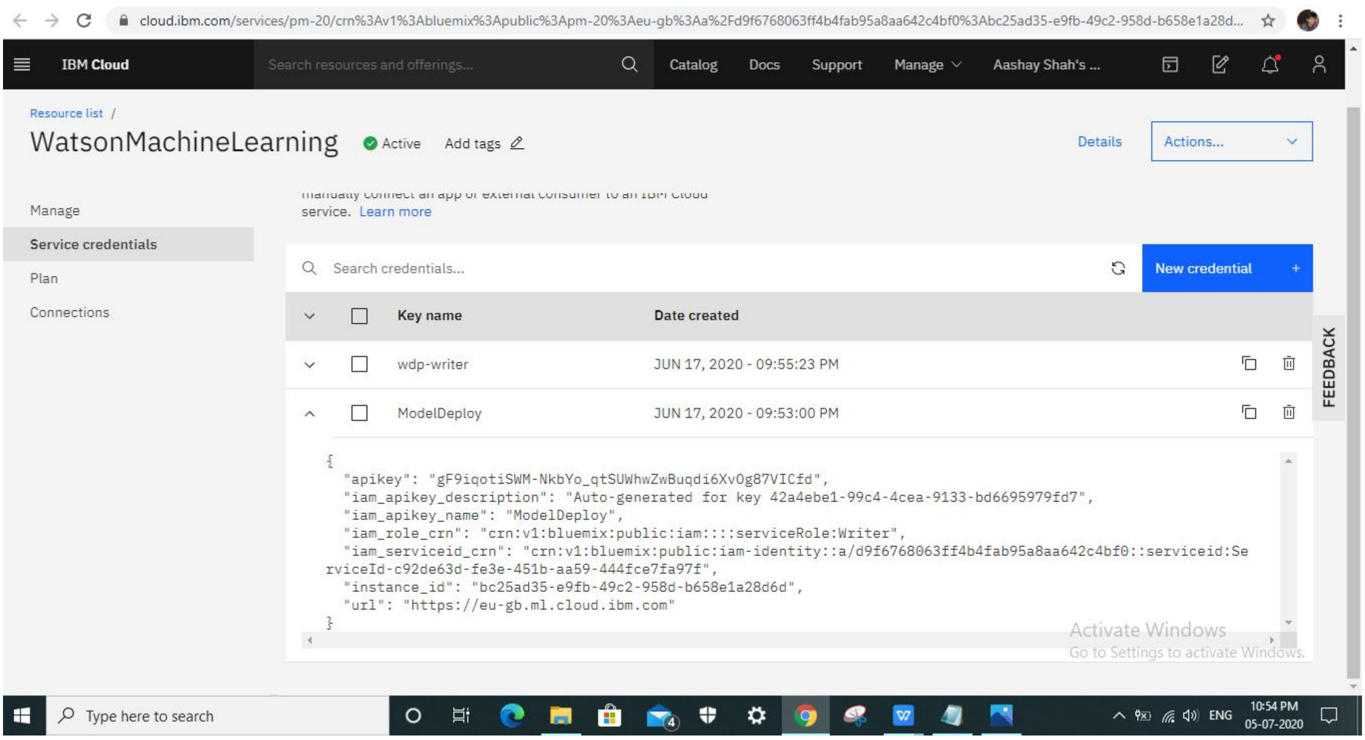
Name	Group	Location	Status	Tags
Filter by name or IP address...	Filter by group or org...	Filter...	Filter...	Filter...
Devices (0)				
VPC infrastructure (0)				
Clusters (0)				
Cloud Foundry apps (2)				
Node RED ABS5	ashu25093@gmail.com / dev	London	Started	–
Node RED ABS5-OLD-1592151126	ashu25093@gmail.com / dev	London	Stopped	–
Cloud Foundry services (1)				
Services (5)				
Continuous Delivery	Default	London	Active	–
KnowledgeCatalog	Default	London	Active	–
WatsonMachineLearning	Default	London	Active	



Resource list

Name	Group	Location	Status	Tags
Filter by name or IP address...	Filter by group or org...	Filter...	Filter...	Filter...
Services (5)				
WatsonStudio	Default	London	Active	–
node-red-abs5-cloudant-1592144753846	Default	London	Active	–
Storage (1)				
cloud-object-storage-tv	Default	Global	Active	cpda...
Network (0)				
Cloud Foundry enterprise environments (0)				
Functions namespaces (0)				
Apps (1)				
Developer tools (1)				
VMware (0)				
Schematics workspaces (0)				

Watson Machine Learning Service

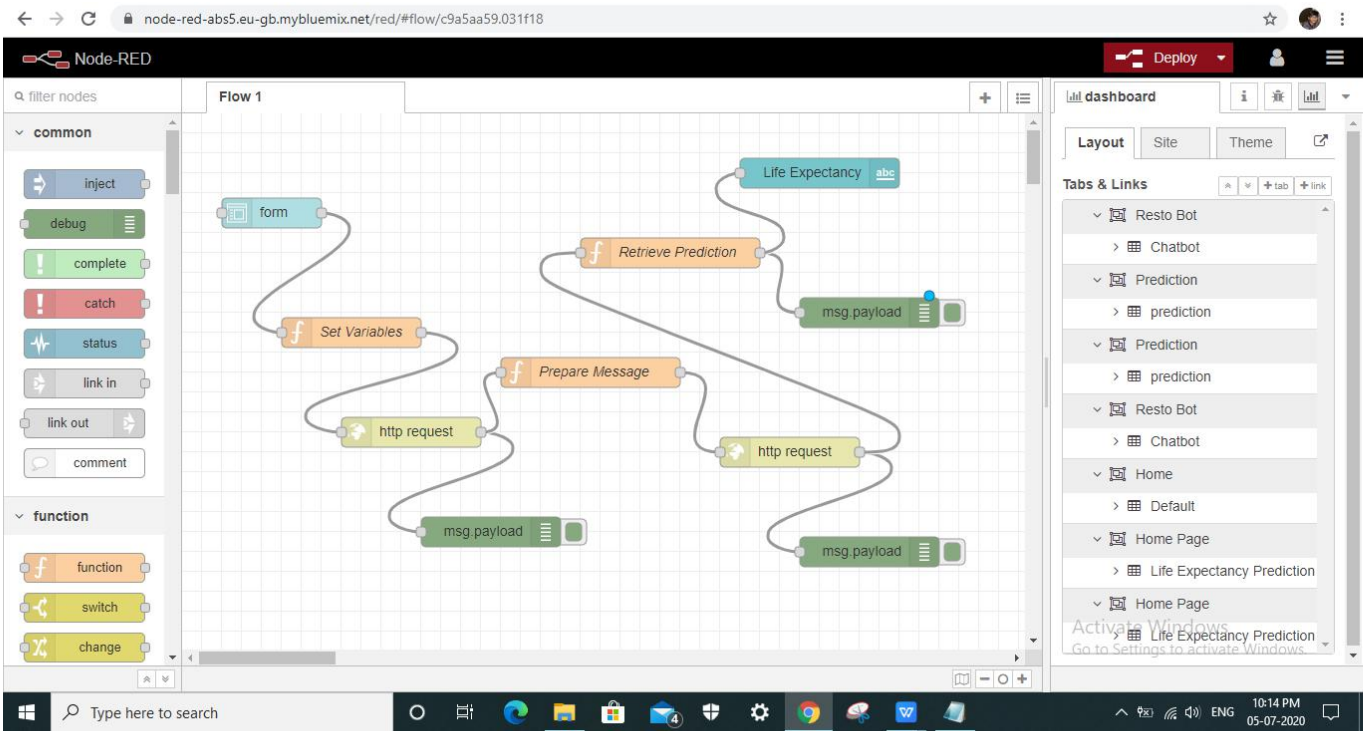


manually connect an app or external consumer to an IBM Cloud service. Learn more

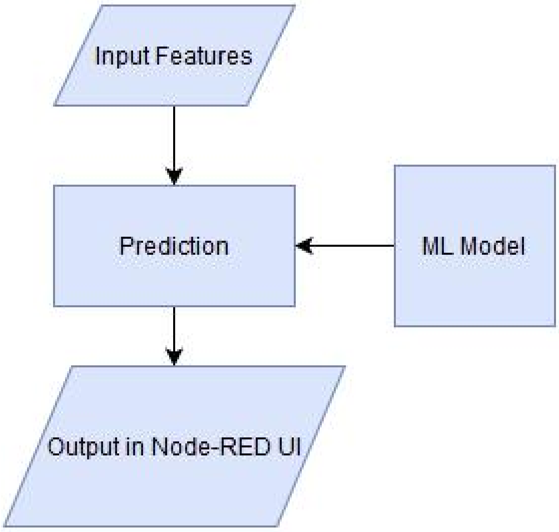
Key name	Date created
wdp-writer	JUN 17, 2020 - 09:55:23 PM
ModelDeploy	JUN 17, 2020 - 09:53:00 PM

```
{
  "apikey": "gF9iqotiSWM-NkbYo_qtSUWhwZwBuqdi6Xv0g87VICfd",
  "iam_apikey_description": "Auto-generated for key 42a4ebe1-99c4-4cea-9133-bd6695979fd7",
  "iam_apikey_name": "ModelDeploy",
  "iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Writer",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/d9f6768063ff4b4fab95a8aa642c4bf0::serviceid:Se
rviceId-c92de63d-fe3e-451b-aa59-444fce7fa97f",
  "instance_id": "bc25ad35-e9fb-49c2-958d-b658e1a28d6d",
  "url": "https://eu-gb.ml.cloud.ibm.com"
}
```

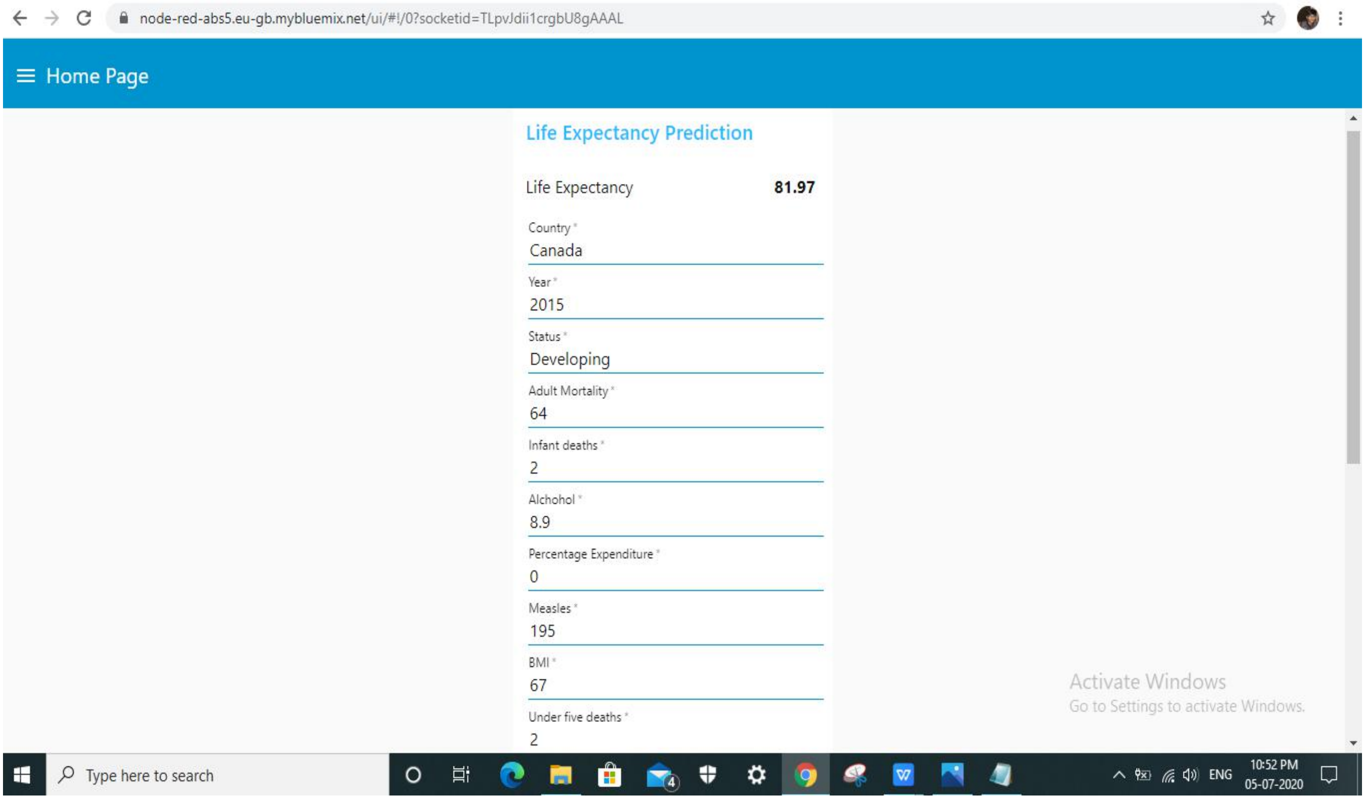
Node RED Flow :



5. **Flowchart**



6. Results


node-red-abs5.eu-gb.mybluemix.net/ui/#!/0?socketid=TLpvJdii1crgbU8gAAAL
Home Page
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
10:52 PM
05-07-2020

Life Expectancy Prediction

Life Expectancy **81.97**

Country *
Canada

Year *
2015

Status *
Developing

Adult Mortality *
64

Infant deaths *
2

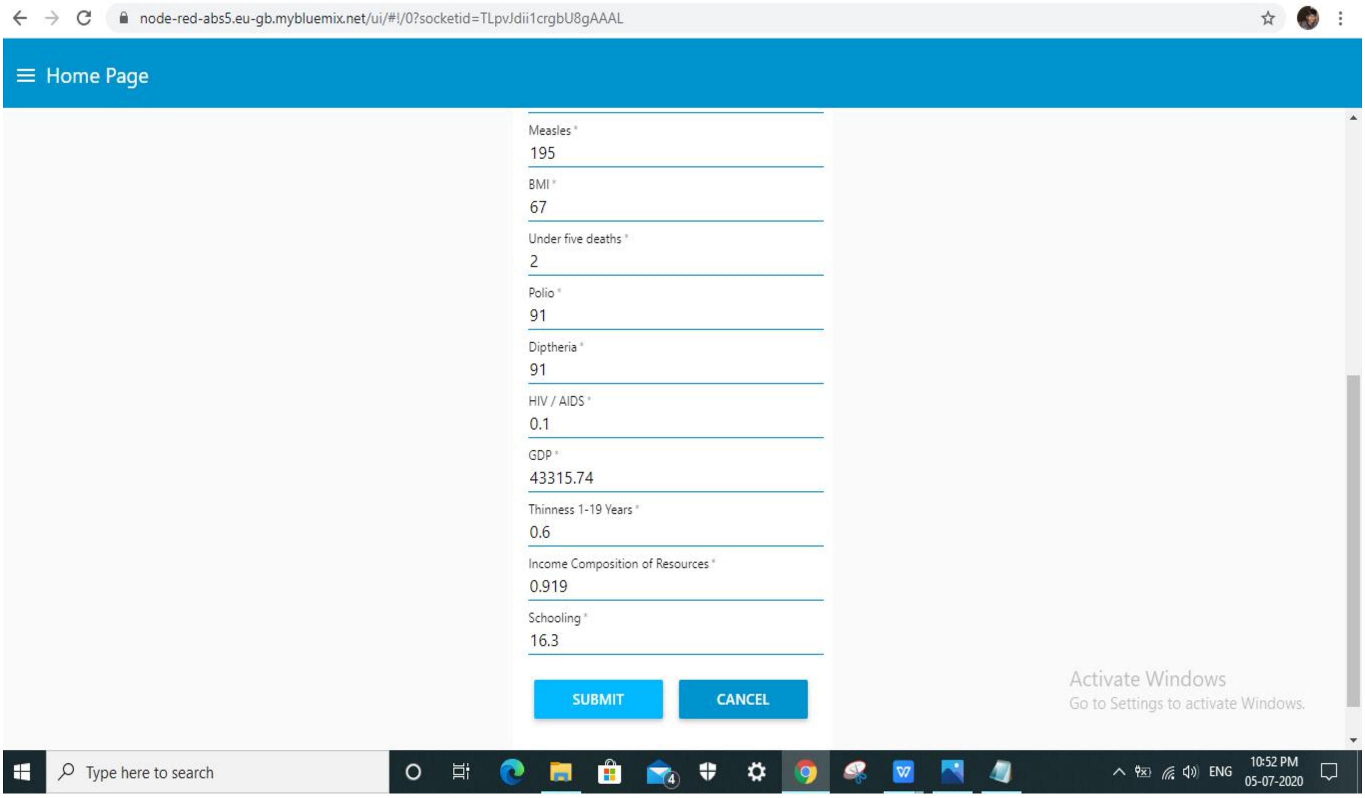
Alchohol *
8.9

Percentage Expenditure *
0

Measles *
195

BMI *
67

Under five deaths *
2


node-red-abs5.eu-gb.mybluemix.net/ui/#!/0?socketid=TLpvJdii1crgbU8gAAAL
Home Page
SUBMIT
CANCEL
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
10:52 PM
05-07-2020

Measles *
195

BMI *
67

Under five deaths *
2

Polio *
91

Diptheria *
91

HIV / AIDS *
0.1

GDP *
43315.74

Thinness 1-19 Years *
0.6

Income Composition of Resources *
0.919

Schooling *
16.3

7. Advantages and Disadvantages

Advantages :

1. The life expectancy predictor will give important insights and help people achieve good quality of life in future. The country can plan and improve various healthcare facilities.
2. The data-sets are made available to public for the purpose of health data analysis.
3. Random Forest is comparatively less impacted by noise.
4. Benefit the country's growth.
5. The application is easy and simple to use.
6. Can be used in any organization to analyze the data.
7. Advantages of using IBM Cloud : Easy to use and deploy, easy to connect with UI, takes care of large storage space.

Disadvantages :

1. Random Forest require much more time to train as compared to decision trees as it generates a lot of trees (instead of one tree in case of decision tree) and makes decision on the majority of votes.
2. User input is not saved in any database.
3. Can be only used by the people having the knowledge of data analysis.
4. Input should be in range only to predict accurate values.
5. As the model is deployed on cloud, so one requires good internet connection to use the application.

8. Applications

- ✧ To help government prepare life insurance policies for people. This will benefit the people.
- ✧ The project can be used as a basis to develop personalized health applications.
- ✧ This will help in suggesting a country which area should be given importance in order to efficiently improve the life expectancy of its population.
- ✧ To analyze all the factors and plan out measures to increase the life expectancy of the country.

9. Conclusion

By doing the above procedure and all we successfully created Life expectancy prediction system using IBM Watson studio, Watson machine learning and Node-RED service. The potential use of project is not limited to health care in practice, but could also be useful in other clinical applications such as clinical trials. The project makes a good use of machine learning in predicting life expectancy of a country that can help respective government in making policies that will serve for the benefit of the nation and entire humankind.

10. Future Scope

We could possibly collect more data by expanding the scope to cities instead of countries, and to explore other features (factors) affecting life expectancy. Also, we could split the data to male and female categories for such life expectancy regression analysis.

11. BIBILOGRAPHY

- A Systematic Literature Review of Studies Analyzing Inequalities in Health Expectancy among the Older Population (Benedetta Pongiglione, Bianca L. De Stavola,George B. Ploubidis)
- https://cloud.ibm.com/
- IBM Developer, "IBM Watson Studio: Create a project", 2019. [Online]. Available: https://www.youtube.com/watch?v=-CUi8GezG1I&list=PLzpeuWUENMK2PYtasCaKK4bZjaYzhW23L&index=2
- IBM Developer, "IBM Watson Studio: Jupyter notebook basics", 2019 [Online]. Available: https://www.youtube.com/watch?v=Jtej3Y6uUng
- IBM Cloud setup [Online]. Available: https://www.ibm.com/cloud/get-started .
- IBM Developer, "Node-RED starter tutorial" [Online]. Available: https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/ .
- "Node-RED labs on the use of the Watson Developer Cloud services - watson-developer-cloud/node-red-labs." [Online]. Available: https://github.com/watson-developer-cloud/node-red-labs .
- "Infuse AI into your applications with Watson AI to make more accurate predictions". [Online]. Available: https://www.ibm.com/watson/products-services .
- IBM Watson, "Intro to IBM Watson", 2018 [Online]. Available: https://www.youtube.com/watch?v=W3iPbFTAAds&feature=youtu.be .
- "Get an understanding of the principles of machine learning." [Online]. Available: https://developer.ibm.com/technologies/machine-learning/series/learning-path-machine-learning-for-developers/
- IBM Developer, "IBM Watson Machine Learning: Get Started in IBM Cloud", 2020 [Online]. Available: https://www.youtube.com/watch?v=NmdjtezQMSM .
- Watson Studio Workshop, "Chapter 4 Build and Deploy models in Jupyter Notebooks" [Online]. Available: https://bookdown.org/caoying4work/watsonstudio-workshop/jn.html .

- Kumar Rajarshi, “Life Expectancy (WHO) Statistical Analysis on factors influencing Life Expectancy”, 2018. [Online]. Available: https://www.kaggle.com/kumarajarshi/life-expectancy-who
- IBM Developer, “IBM Watson: Sign up for Watson Studio and Watson Knowledge Catalog”, 2019. [Online]. Available: https://www.youtube.com/watch?v=DBRGlAHdj48&list=PLzpeuWUENMK2PYtasCaKK4bZjaYzhW23L

Appendix

Source Code

Installing and Importing all the libraries :

```
pip install pandas-profiling[notabook,html]
import pandas as pd
import numpy as np
import types
import pandas as pd
from botocore.client import Config
import ibm_boto3
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from collections import OrderedDict
from sklearn.metrics import accuracy_score
from sklearn.metrics import r2_score,mean_squared_error
from watson_machine_learning_client import WatsonMachineLearningAPIClient
```

Loading the file :

```
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_252fb149ac854967a383d909584421f8 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='5fHfR8gNPvnMM6RTKWwCChTzCD13d2KXt9ZLKOzitMZr',
    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_252fb149ac854967a383d909584421f8.get_object(Bucket='lifeexpectancy2-donotdelete-
            pr-akjpyh8ahgdg3z',Key='Life_Expectancy1.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 )
dataset1 = pd.read_csv(body)
dataset1.head()
```

EDA

```
profile = ProfileReport(data, title='Pandas Profiling Report', explorative=True)
profile.to_notebook_iframe()
profile.to_file("Data Analysis.html")
```

Feature Selection

```
corrmat = dataset1.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))
#plot heat map
g=sns.heatmap(dataset1[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

Dropping Unnecessary features

```
dataset1.drop(['Hepatitis B','Total expenditure',' thinness 5-9 years','Population'],axis=1,inplace=True)
```

Handling Missing Values

GDP:-https://www.worldometers.info/gdp/gdp-per-capita/
BMI:-https://ourworldindata.org/obesity
HDI and Schooling:-http://hdr.undp.org/en/countries
Alcohols:-https://ourworldindata.org/alcohol-consumption
Thinness:-https://apps.who.int/gho/data/view.main.NCDBMIMINUS205-19Cv

OR can be handled by performing the following code for all columns.

```
dataset1['column_name'].fillna(dataset1.groupby(['Country'])['Life Expectancy'].transform('median'))
```

```
# After handling missing values for all columns we store it in a new csv file
dataset1.to_csv('Life_Expectancy.csv',index=False)
```

Loading new dataset after handling missing values.

```
body = client_252fb149ac854967a383d909584421f8.get_object
(Bucket='lifeexpectancy2-donotdelete-pr-akjpyh8ahgdg3z',Key='Life_Expectancy.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 )

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

print(dataset.columns)
Index(['Country', 'Year', 'Status', 'Life expectancy ', 'Adult Mortality', 'infant deaths', 'Alcohol',
'percentage   expenditure', 'Measles',' BMI ', 'under-five deaths ', 'Polio', 'Diphtheria', ' HIV/AIDS','GDP', '
thinness    1-19    years', 'Income composition of resources', 'Schooling'],dtype='object')

dataset.rename(columns = {'Life expectancy ' : 'Life Expectancy'}, inplace = True)

X = dataset.drop(['Life Expectancy'],axis=1)
Y = dataset['Life Expectancy']
```

Splitting Dataset for Training and Testing

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state =
0,shuffle=True)
```

Separating Numerical and Categorical Variables.

```
numeric_features = ['Year', 'Adult Mortality','infant deaths', 'Alcohol', 'percentage expenditure',
'Measles',
 ' BMI ', 'under-five deaths ', 'Polio', 'Diphtheria', ' HIV/AIDS', 'GDP', ' thinness    1-19 years',
'Income                    composition of resources','Schooling']
categorical_features = ['Country', 'Status']
```

Creating Pipeline

```
categorical_transformer = Pipeline(steps=[('onehot', OneHotEncoder(handle_unknown='ignore'))])
numeric_transformer = Pipeline(steps=[('imputer', SimpleImputer(strategy='median'))])
```

```
preprocessor = ColumnTransformer(transformers=[
            ('cat', categorical_transformer, categorical_features),
            ('num', numeric_transformer, numeric_features)])
```

Finding Best Algorithm

```
models = OrderedDict([
     ( "Linear Regression",          Pipeline([
                                          ('preprocessor', preprocessor),
                                          ('LRegressor', LinearRegression())])    ),
     ( "Decision Tree Regressor", Pipeline([
                                         ('preprocessor', preprocessor),
                                          ('DTRegressor',
DecisionTreeRegressor())])    ),
     ( "Random Forest Regressor", Pipeline([
                                         ('preprocessor', preprocessor),
                                          ('RFRegressor',
RandomForestRegressor())])    )
])

# finding accuracy for above three models and print them in ascending order of accuracy.
scores = {}
for (name, model) in models.items():
     model.fit(X_train,Y_train)
     scores[name] =r2_score(model.predict(X_test), Y_test)

scores = OrderedDict(sorted(scores.items()))
scores
#output : OrderedDict([('Decision Tree Regressor', 0.9220148178951159),
             ('Linear Regression', 0.9372148323411937),
             ('Random Forest Regressor', 0.9539041774661141)])
```

Random forest gives best performance amongst all algorithms, so I have used it for this project.

```
rf_model = Pipeline([
        ('preprocessor', preprocessor),
        ('RFRegressor', RandomForestRegressor())])

# Train our model
rf_model.fit(X_train,Y_train)
#output : Pipeline(memory=None,
        steps=[('preprocessor', ColumnTransformer(n_jobs=None, remainder='drop',
sparse_threshold=0.3,
              transformer_weights=None,
              transformers=[('cat', Pipeline(memory=None,
        steps=[('onehot', OneHotEncoder(categorical_features=None, categories=None,

        dtype=<class 'numpy.float64'>...ators=10, n_jobs=None,
                 oob_score=False, random_state=None, verbose=0, warm_start=False))])

X_test.shape
# Making prediction
pred = rf_model.predict(X_test)
print(Y_train)
pred
    # output : array([77.89, 67.77, 53.34, 57.54, 57.41, 68.11, 73.72, 84.48, 74.49,
                        53.64, 66.23, 81.6 , 75.24, 76.65, 64.65, 74.69, 59.02, 64.21,
                       73.32, 82.92, 74.74, 69.54, 72.74, 82.43, 69.81, 62.45, 82.22,
                    ..…………………………………………………………………….
                    75.63, 71.5 , 66.91, 73.7 , 81.65, 82.82, 80.    , 72.86, 73.84,
                    70.56, 74.07, 65.13, 52.98, 72.42, 68.14, 74.24, 82.83, 52.1,
                    72.74, 73.09, 82.1 ])
Y_test.shape
    (588,)
pred.shape
    (588,)
```

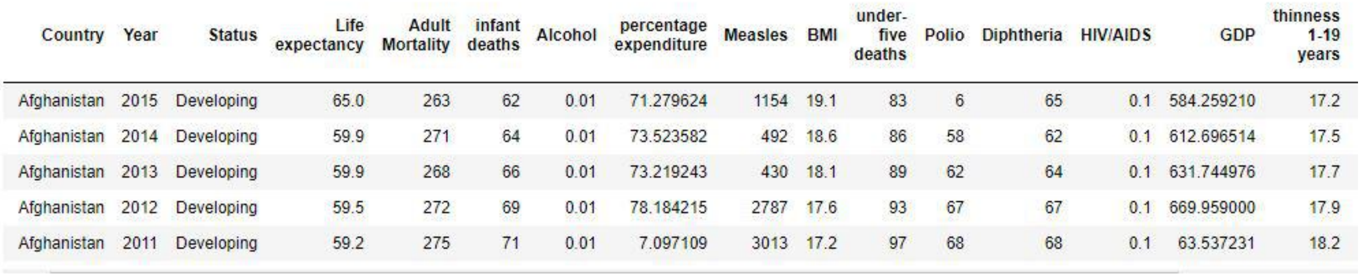
```
# Absolute errors between Actual values and the predicted values
errors = abs(pred - Y_test)
print(errors)
     #output : 867         0.59
               1780       1.17
               621        0.74
                 ...           …
               392        0.14
               27         0.09
               674        0.90

print('Average absolute error:', round(np.mean(errors), 2), 'degrees.')
#output : Average absolute error: 1.23 degrees.

mape = 100 * (errors / Y_test)
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')
#output    : Accuracy: 98.15 %.
```

Creating Watson Machine Learning Client

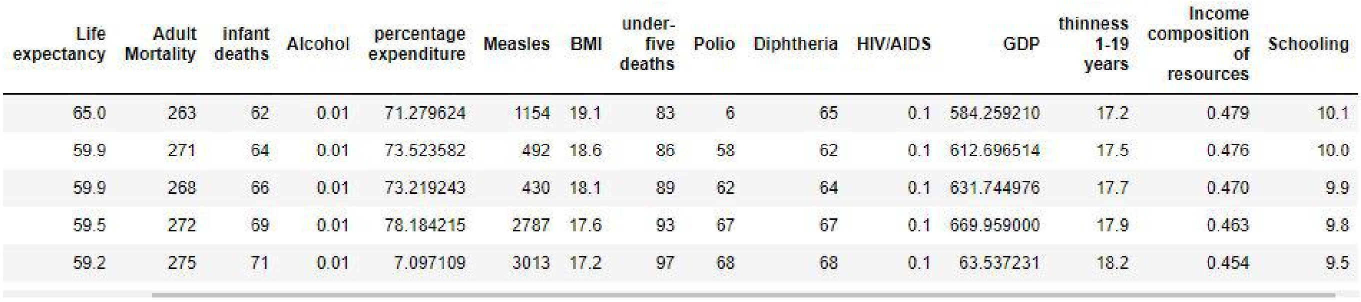
```
wml_credentials = {
   "apikey": "gHX4ABRU-VFFzwPyGUejEecXptY2IN0jnwv3eFMkIBNj",
   "iam_apikey_description": "Auto-generated for key 54a7f5a0-96d4-42a6-ad57-c7f9d378d895",
   "iam_apikey_name": "InternshipCredential",

"iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Manager",
"iam_serviceid_crn":"crn:v1:bluemix:public:iam-identity::a/d9f67680663ff4b4fab95a8aa642c4bf0::se
rviceid:  ServiceId-5e3caea5    -08af-4d5e-9d44-e6da7a56aee1",
"instance_id": "bc25ad35-e9fb-49c2-958d-b658e1a28d6d",
  "url": "https://eu-gb.ml.cloud.ibm.com"
}
client = WatsonMachineLearningAPIClient(wml_credentials)
print(client.version)
#output : 1.0.378
```

```
# Set meta-data to our model
model_props = {client.repository.ModelMetaNames.AUTHOR_NAME: "Aashay",
               client.repository.ModelMetaNames.AUTHOR_EMAIL:
                                                  "ashu25093@gmail.com",
               client.repository.ModelMetaNames.NAME: "Life_Expectancy"}

# Storing our model to the repository
model_artifact = client.repository.store_model(rf_model, meta_props=model_props)

# After storing the model,we will get model id which will be used for deploying the model
published_model_uid = client.repository.get_model_uid(model_artifact)
published_model_uid

# Creating deployment of our model
# After successfully deployment of our model, we will get scoring endpoint, to which our Node-RED
   app send the data.
deployment = client.deployments.create(published_model_uid, name="Life_Expectancy")
scoring_endpoint = client.deployments.get_scoring_url(deployment)
scoring_endpoint

   #output : Synchronous deployment creation for uid: 'b6ddf698-f63a-4a06-92d4-82b70728f34d'
                                                              started
           Successfully finished deployment creation,
              deployment_uid='1e912bc3-7fef-47ba-ba4d-8b175c44a0eb'
           Scoring endpoint : 'https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/
                      bc25ad35-e9fb-49c2-958d-b658e1a28d6d/deployments/
                      1e912bc3-7fef-47ba-ba4d-8b175c44a0eb/online'
```

Dataset



	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Measles	BMI	under-five deaths	Polio	Diphtheria	HIV/AIDS	GDP	thinness 1-19 years
	Afghanistan	2015	Developing	65.0	263	62	0.01	71.279624	1154	19.1	83	6	65	0.1	584.259210	17.2
	Afghanistan	2014	Developing	59.9	271	64	0.01	73.523582	492	18.6	86	58	62	0.1	612.696514	17.5
	Afghanistan	2013	Developing	59.9	268	66	0.01	73.219243	430	18.1	89	62	64	0.1	631.744976	17.7
	Afghanistan	2012	Developing	59.5	272	69	0.01	78.184215	2787	17.6	93	67	67	0.1	669.959000	17.9
	Afghanistan	2011	Developing	59.2	275	71	0.01	7.097109	3013	17.2	97	68	68	0.1	63.537231	18.2



Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Measles	BMI	under-five deaths	Polio	Diphtheria	HIV/AIDS	GDP	thinness 1-19 years	Income composition of resources	Schooling
65.0	263	62	0.01	71.279624	1154	19.1	83	6	65	0.1	584.259210	17.2	0.479	10.1
59.9	271	64	0.01	73.523582	492	18.6	86	58	62	0.1	612.696514	17.5	0.476	10.0
59.9	268	66	0.01	73.219243	430	18.1	89	62	64	0.1	631.744976	17.7	0.470	9.9
59.5	272	69	0.01	78.184215	2787	17.6	93	67	67	0.1	669.959000	17.9	0.463	9.8
59.2	275	71	0.01	7.097109	3013	17.2	97	68	68	0.1	63.537231	18.2	0.454	9.5

Node RED : Flows.json

[{"id":"2227635a.b24a4c","type":"tab","label":"Flow 1","disabled":false,"info":""},{"id":"eaadb788.6630b8","type":"function","z":"2227635a.b24a4c","name":"Set Variables","func":"global.set(\"ctry\",msg.payload.ctry)\nglobal.set(\"yr\",msg.payload.yr)\nglobal.set(\"sts\",msg.payload.sts)\nglobal.set(\"adult_mor\",msg.payload.adult_mor)\nglobal.set(\"inf_deaths\",msg.payload.inf_deaths)\nglobal.set(\"alcohol\",msg.payload.alcohol)\nglobal.set(\"per_expend\",msg.payload.per_expend)\nglobal.set(\"measles\",msg.payload.measles)\nglobal.set(\"bmi\",msg.payload.bmi)\nglobal.set(\"under_five_deaths\",msg.payload.under_five_deaths)\nglobal.set(\"polio\",msg.payload.polio)\nglobal.set(\"diptheria\",msg.payload.diptheria)\nglobal.set(\"hiv_aids\",msg.payload.hiv_aids)\nglobal.set(\"gdp\",msg.payload.gdp)\nglobal.set(\"thinness\",msg.payload.thinness)\nglobal.set(\"hdi\",msg.payload.hdi)\nglobal.set(\"edu_index\",msg.payload.edu_index)\nvar apikey=\"gHX4ABRU-VFFzwPyGUejEecXptY2IN0jnwv3eFMkIBNj\";\nmsg.headers={\"content-type\":\"application/x-www-form-urlencoded\"}\nmsg.payload={\"grant_type\":\"urn:ibm:params:oauth:grant-type:apikey\",\"apikey\":apikey}\nreturn msg;","outputs":1,"noerr":0,"x":170,"y":300,"wires":[["db683a27.e9acd8"]]},
{"id":"db683a27.e9acd8","type":"http request","z":"2227635a.b24a4c","name":"","method":"POST","ret":"obj","paytoqs":false,"url":"https://iam.cloud.Ibm.com/identity/token","tls":"","persist":false,"proxy":"","authType":"","x":350,"y":360,"wires":[["15db6b78.a7cb75","ca6409fc.75f6b8"]]},{"id":"688c698d.9ec098","type":"debug","z":"2227635a.b24a4c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":710,"y":180,"wires":[]},{"id":"ca6409fc.75f6b8","type":"function","z":"2227635a.b24a4c","name":"Prepare Message","func":"var ctry = global.get('ctry')\nvar yr = global.get('yr')\nvar sts = global.get('sts')\nvar adult_mor = global.get('adult_mor')\nvar inf_deaths = global.get('inf_deaths')\nvar alcohol = global.get('alcohol')\nvar per_expend = global.get('per_expend')\nvar measles = global.get('measles')\nvar bmi = global.get('bmi')\nvar under_five_deaths = global.get('under_five_deaths')\nvar polio = global.get('polio')\nvar diptheria = global.get('diptheria')\nvar hiv_aids = global.get('hiv_aids')\nvar gdp = global.get('gdp')\nvar thinness = global.get('thinness')\nvar hdi = global.get('hdi')\nvar edu_index = global.get('edu_index')\nvar token=msg.payload.access_token\nvar instance_id=\"bc25ad35-e9fb-49c2-958d-b658e1a28d6d\"\nmsg.headers={'Content-Type': 'application/json',\"Authorization\":\"Bearer \"+token,\"ML-Instance-ID\":instance_id}\nmsg.payload={\"fields\":[\"Country\", \"Year\", \"Status\", \"Adult Mortality\",\"infant deaths\", \"Alcohol\", \"percentage expenditure\", \"Measles\",\" BMI \", \"under-five deaths \", \"Polio\", \"Diphtheria\", \" HIV/AIDS\",\"GDP\", \" thinness 1-19 years\", \"Income composition of resources\",\"Schooling\"],\"values\":[[ctry,yr,sts,adult_mor,inf_deaths,alcohol,per_expend,measles,bmi,under_five_deaths,polio,diptheria,hiv_aids,gdp,thinness,hdi,edu_index]]}\nreturn msg;","outputs":1,"noerr":0,"x":450,"y":220,"wires":[["ad32b620.07db98"]]},{"id":"ad32b620.07db98","type":"httprequest","z":"2227635a.b24a4c","name":"","method":"POST","ret":"obj","paytoqs":false,"url":"https://eu-gb.ml.cloud.ibm.com/v3/wml_instances/bc25ad35-e9fb-49c2-958d-b658e1a28d6d/deployments/1e912bc3-7fef-47ba-ba4d-8b175c44a0eb/online","tls":"","persist":false,"proxy":"","authType":"","x":630,"y":280,"wires":[["4b8eaeee.ffb9e", "29707f81.77f6d"]]},{"id":"6e552812.98f3b8","type":"ui_form","z":"2227635a.b24a4c","name":"","label":"","group":"36d16e03.7fff62","order":0,"width":0,"height":0,"options":[{"label":"Country","value":"ctry","type":"text","required":true,"rows":null},{"label":"Year","value":"yr","type":"number","required":true,"rows":null},{"label":"Status

","value":"sts","type":"text","required":true,"rows":null},{"label":"Adult
Mortality","value":"adult_mor","type":"number","required":true,"rows":null},{"label":"Infant
deaths","value":"inf_deaths","type":"number","required":true,"rows":null},{"label":"Alchohol","value":"alcohol","t
ype":"number","required":true,"rows":null},{"label":"Percentage
Expenditure","value":"per_expend","type":"number","required":true,"rows":null},{"label":"Measles","value":"mea
sles","type":"number","required":true,"rows":null},{"label":"BMI","value":"bmi","type":"number","required":true,"
rows":null},{"label":"Under five
deaths","value":"under_five_deaths","type":"number","required":true,"rows":null},{"label":"Polio","value":"polio",
"type":"number","required":true,"rows":null},{"label":"Diptheria","value":"diptheria","type":"number","required":t
rue,"rows":null},{"label":"HIV /
AIDS","value":"hiv_aids","type":"number","required":true,"rows":null},{"label":"GDP","value":"gdp","type":"nu
mber","required":true,"rows":null},{"label":"Thinness 1-19
Years","value":"thinness","type":"number","required":true,"rows":null},{"label":"Income Composition of
Resources","value":"hdi","type":"number","required":true,"rows":null},{"label":"Schooling","value":"edu_index","
type":"number","required":true,"rows":null}],"formValue":{"ctry":"","yr":"","sts":"","adult_mor":"","inf_deaths":"",
"alcohol":"","per_expend":"","measles":"","bmi":"","under_five_deaths":"","polio":"","diptheria":"","hiv_aids":"","
gdp":"","thinness":"","hdi":"","edu_index":""},"payload":"","submit":"submit","cancel":"cancel","topic":"","x":70,"
y":180,"wires":[["eaadb788.6630b8"]]},{"id":"87759088.9d68c","type":"ui_text","z":"2227635a.b24a4c","group":"
36d16e03.7fff62","order":1,"width":0,"height":0,"name":"","label":"Life
Expectancy","format":"{{msg.payload}}","layout":"row-spread","x":480,"y":40,"wires":[]},{"id":"29707f81.77f6d
","type":"function","z":"2227635a.b24a4c","name":"Retrieve
Prediction","func":"msg.payload=msg.payload.values[0][0]\nreturn
msg;","outputs":1,"noerr":0,"x":570,"y":120,"wires":[["688c698d.9ec098","87759088.9d68c"]]},{"id":"15db6b78.a
7cb75","type":"debug","z":"2227635a.b24a4c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":fa
lse,"complete":"payload","targetType":"msg","x":430,"y":460,"wires":[]},{"id":"4b8eaeee.ffb9e","type":"debug","z
":"2227635a.b24a4c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload",
"targetType":"msg","x":710,"y":380,"wires":[]},{"id":"36d16e03.7fff62","type":"ui_group","z":"","name":"Life
Expectancy Prediction","tab":"b54ee5e.a5e6c18","order":1,"disp":true,"width":"6","collapse":false},
{"id":"b54ee5 e.a5e6c18","type":"ui_tab","z":"","name":"Home Page","icon":"dashboard"
,"disabled":false,"hidden":false}]

Links

Github link :

https://github.com/ashu25093/llSPS-INT-2914-Predicting-Life-Expectancy-using-Machine-Learning

Youtube Video links :

Project Demonstration : https://youtu.be/ndbWhsVILm8

Feedback : https://youtu.be/t-b18YCB_Go