REPORT

LIFE EXPECTANCY PREDICTING MODEL

DIXITA SHUKLA

1. INTRODUCTION

1.1 OVERVIEW

Life expectancy is one of the most important factors in end-of-life decision making.

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 data originates from here: https://www.kaggle.com/kumarajarshi/life-expectancy-who/data 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.

1.2 PURPOSE

The purpose is to predict Life Expectancy by looking at the positive and negatively correlated factors to improve the Life Quality.

It serves as an example for countries to assess to improve life expectancy for their citizens.

When you are deciding when to start receiving retirement benefits, one important factor to take into consideration is how long you might live.

2. LITERATURE SURVEY

2.1 Existing Problem

As we all know, Life expectancy is one of the most important factors in end-of-life decision making.

So, using the certain factors like Schooling, GDP, Adult Mortality Rate, Child Date, etc. life expectancy is predicted. All the factors are negatively or positively correlated.

When you are deciding when to start receiving retirement benefits, one important factor to take into consideration is how long you might live.

- *A man turning age 65 on April 1, 2020 can expect to live, on average, until age 84.0.*

- *A woman turning age 65 on April 1, 2020 can expect to live, on average, until age 86.5.*

2.2 Proposed Solution

Using this model, life expectancy of a person can be predicted by taking some input features from the user.

Life Expectancy depends on the following features-

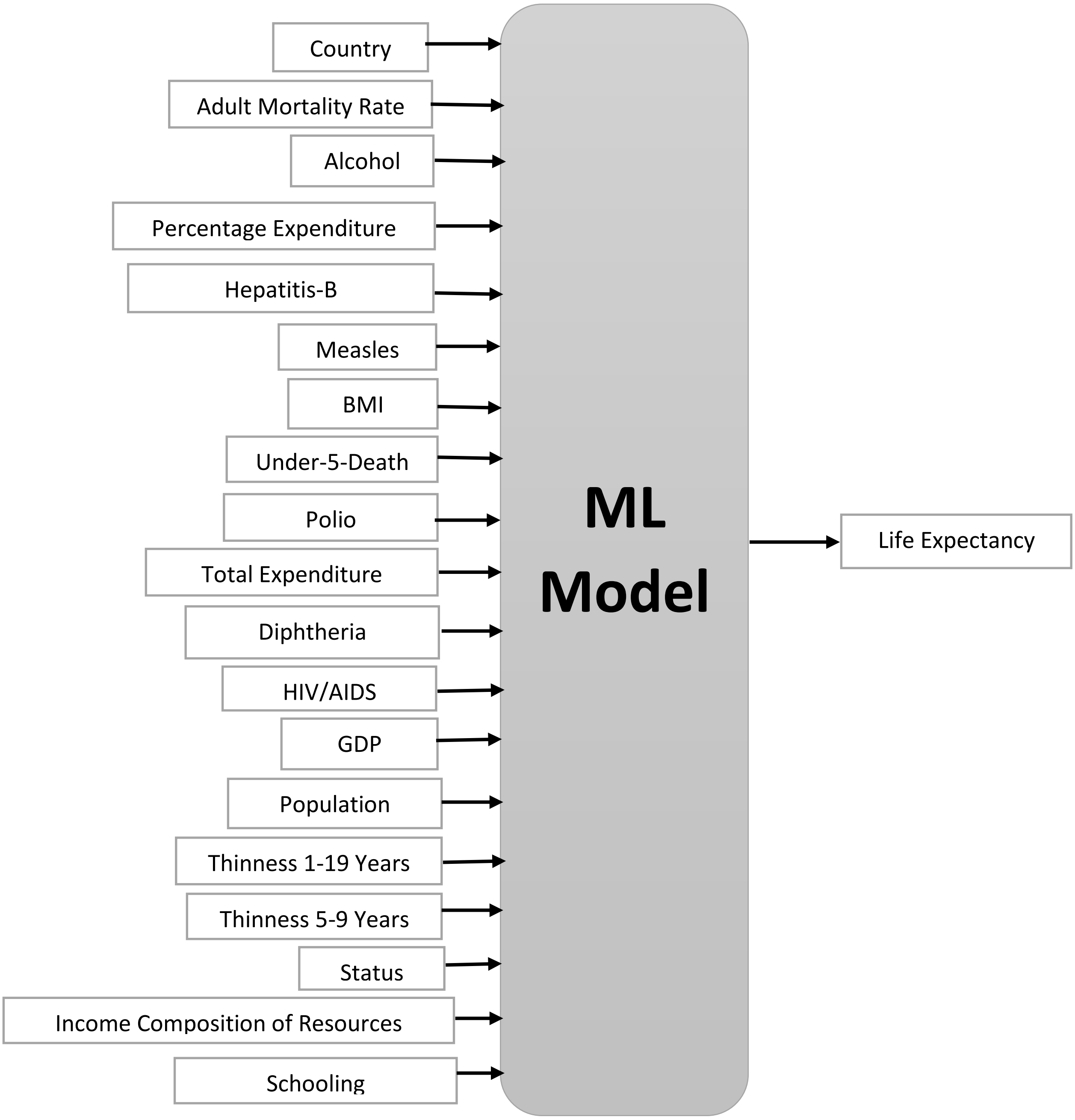
- *Country*
- *Status*
- *Life Expectancy*
- *Adult Mortality*
- *Alcohol*
- *percentage expenditure*
- *Hepatitis B*
- *Measles*
- *BMI*
- *under-five deaths*
- *Polio*
- *Total expenditure*
- *Diphtheria*
- *HIV/AIDS*
- *GDP*
- *Population*
- *thinness 1-19 years*
- *thinness 5-9 years*
- *Income composition of resources*
- *Schooling*

By taking the information regarding the above factors, model will predict the life expectancy.

Want to know your life expectancy? You can use our simple Life Expectancy Predicting Model to get a rough estimate of how long you (or your spouse) may live. Knowing this information can help you make a more informed choice regarding when to collect Social Security retirement benefits.

3. Theoretical Analysis

3.1 Block Diagram



3.2 Software Designing

IBM cloud *computing is a set of cloud computing services for business offered by the information technology company IBM.*

It provides many services like Node-Red, Watson Studio, etc for storing and processing data.

IBM **Cloud**

Node Red *is used for creating the User Interface (UI) application. Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.*

Node-RED

Watson Studio *helps data scientists and analysts prepare data and build models at scale across any cloud. With its open, flexible multicloud architecture,* **Watson Studio** *provides capabilities that empower businesses to simplify enterprise data science and AI: Automate AI lifecycle management with AutoAI.*

4. Experimental Investigation

Data was collected from "https://www.kaggle.com/kumarajarshi/life-expectancy-who/data" and then pre-processed so that it is understood by the Machine Learning Algorithms Properly.

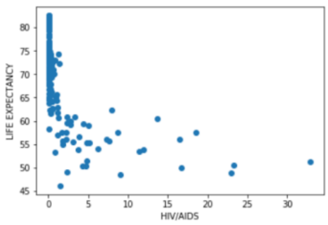
In [3]:
```
data.head()
```
Out[3]:

	Country	Year	Status	Life expectancy	Adult Mortality	infant deaths	Alcohol	percentage expenditure	Hepatitis B	Measles	...	Polio	Total expenditure	Diphtheria	HIV/AIDS	G
0	Afghanistan	2015	Developing	65.0	263.0	62	0.01	71.279624	65.0	1154	...	6.0	8.16	65.0	0.1	584.259
1	Afghanistan	2014	Developing	59.9	271.0	64	0.01	73.523582	62.0	492	...	58.0	8.18	62.0	0.1	612.696
2	Afghanistan	2013	Developing	59.9	268.0	66	0.01	73.219243	64.0	430	...	62.0	8.13	64.0	0.1	631.744
3	Afghanistan	2012	Developing	59.5	272.0	69	0.01	78.184215	67.0	2787	...	67.0	8.52	67.0	0.1	669.959
4	Afghanistan	2011	Developing	59.2	275.0	71	0.01	7.097109	68.0	3013	...	68.0	7.87	68.0	0.1	63.537

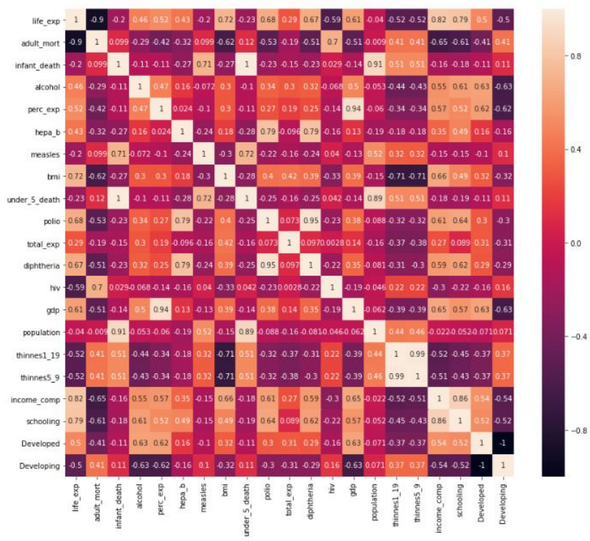
5 rows × 22 columns

Exploratory Data Analysis was done, in order to visualise the dataset, and to check the correlation of different parameters on the 'Life Expectancy'.

In [11]:
```
plt.scatter(data['hiv'],data['life_exp'])
plt.xlabel("HIV/AIDS")
plt.ylabel("LIFE EXPECTANCY")
```
Out[11]: Text(0, 0.5, 'LIFE EXPECTANCY')

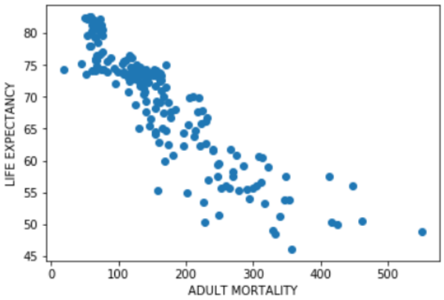


In [19]:
```
plt.figure(figsize=(14,12))
sns.heatmap(data.corr(),annot=True)
```
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x1e2fa1a9048>



In [16]:
```
plt.scatter(data['adult_mort'],data['life_exp'])
plt.xlabel('ADULT MORTALITY')
plt.ylabel('LIFE EXPECTANCY')
```

Out[16]: Text(0, 0.5, 'LIFE EXPECTANCY')



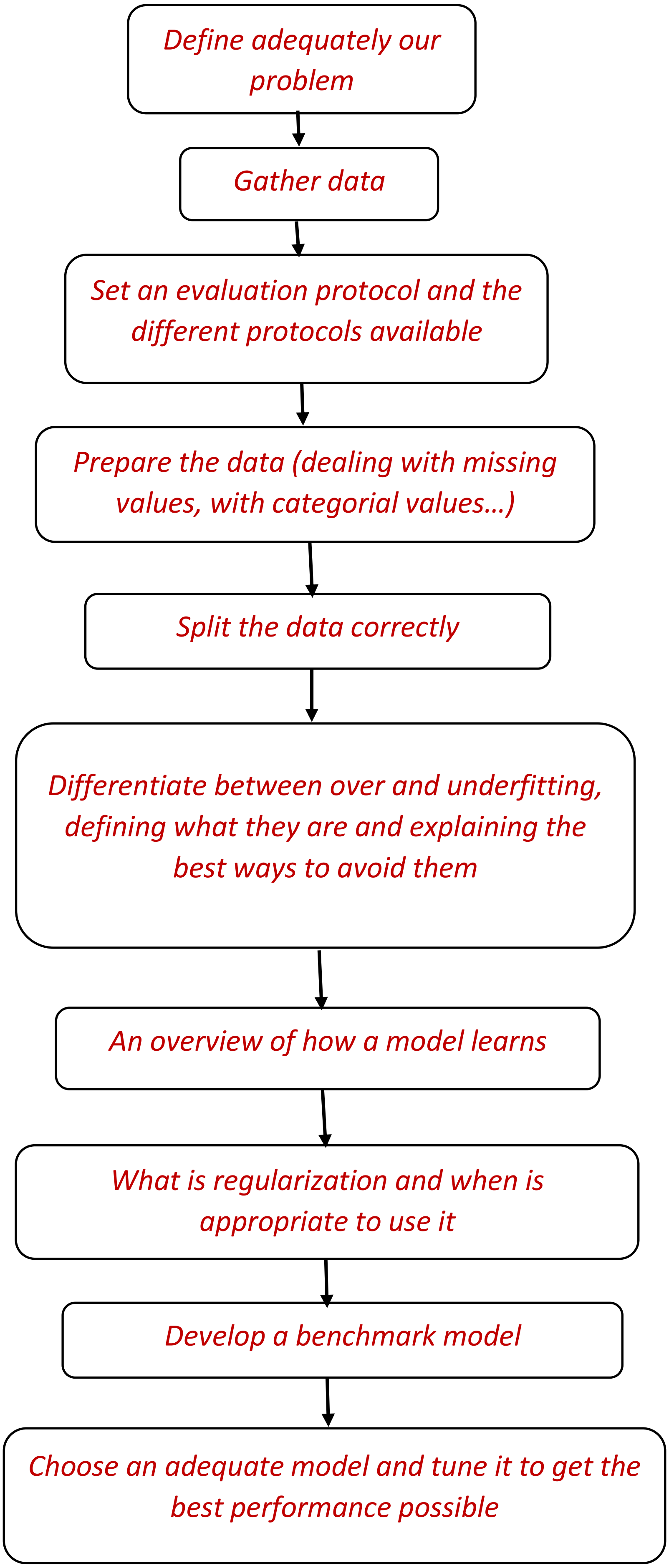
Then, different Regression Algorithms were applied and then accuracy is checked for each, so as to find the best fitted algorithm.

Fine Tuning was done, in order to find the best parameters so that we get the best possible accuracy.

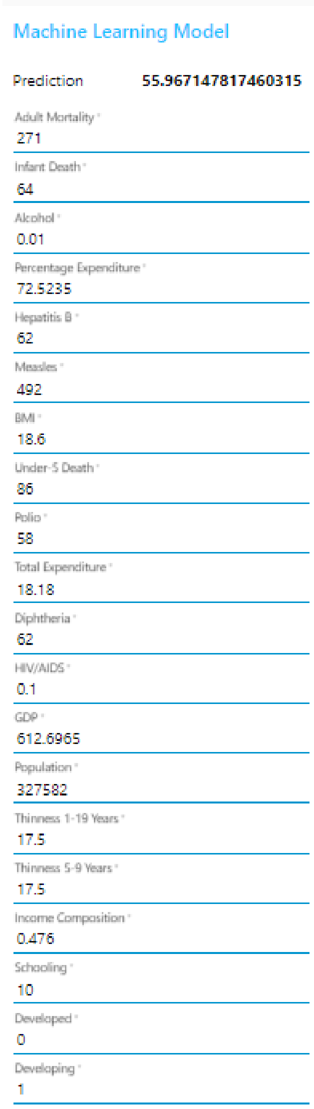
The output algorithms have been used to test if they can maintain their accuracy in predicting the life expectancy for data they haven't been trained. Four algorithms have been used:

- *Linear Regression*
- *Ridge Regression*
- *Lasso Regression*
- *ElasticNet Regression*
- *Linear Regression with Polynomic Features*
- *Decision Tree Regression*
- *Random Forest Regression*

5. Flowchart


Define adequately our problem
Gather data
Set an evaluation protocol and the different protocols available
Prepare the data (dealing with missing values, with categorial values…)
Split the data correctly
Differentiate between over and underfitting, defining what they are and explaining the best ways to avoid them
An overview of how a model learns
What is regularization and when is appropriate to use it
Develop a benchmark model
Choose an adequate model and tune it to get the best performance possible

6. RESULT



By taking the inputs from the user of the certain parameters Life Expectancy has been predicted successfully.

7. Advantages and Disadvantages

Advantages

- Life Expectancy can be predicted depending on certain parameters with great accuracy.
- Parameters which are increasing and decreasing the Life Expectancy can be known.

- *Knowing this information can help you make a more informed choice regarding when to collect Social Security retirement benefits.*

Disadvantages

- *Though, the accuracy of the model is very high. Still there is some chance that the does not give the exact Life Expectancy.*
- *It may create some tension when people got to know their wrong Life Expectancy age.*

8. Application

- *You can use our simple Life Expectancy Predicting Model to get a rough estimate of how long you (or your spouse) may live. Knowing this information can help you make a more informed choice regarding when to collect Social Security retirement benefits.*
- *When you are deciding when to start receiving retirement benefits, one important factor to take into consideration is how long you might live.*
- *Government can improve certain features on which the Life Expectancy depends, so the average life expectancy can be increased.*
- *Policy makers can benefit their customers by suggesting them the appropriate policies for them.*

9. Conclusion

After comparing all the algorithms we can conclude the Lasso and the Elastic Net Regression offer which are the same:

1. *Best Parameters: {'alpha': 0, 'max_iter': 10}*
2. *R square on the test data of 92%*
3. *MAE of 1.83*
4. *MSE of 6.05*

10. Future Scope

- *Look at class within a particular country and see if these same factors are same in determining life expectancy for an individual*

- *Use twitter API to incorporate NLP analysis for a country to see how it relates to Life Expectancy.*
- *Increase the dataset size with continuing UN and Global Data to incorporate new added features like population, GDP, etc in order to test and clarify country groupings.*

11. Bibliography

- https://www.kaggle.com/kumarajarshi/life-expectancy-who/data
- *Introduction to Machine Learning with Python by Andreas C. Müller & Sarah Guido.*
- *SmartInternz Webinars*
- *Mentors*

APPENDIX

Source Code

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from nose.tools import *
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso
from sklearn.linear_model import ElasticNet
from sklearn.metrics import make_scorer
from scipy import stats
import seaborn as sns
who.to_csv()
who.columns=['country','year','status','life_exp','adult_mort','infant_death','alcohol',
'perc_exp','hepa_b','measles','bmi','under_5_death','polio','total_exp','diphtheria','hi
v','gdp','population','thinnes1_19','thinnes5_9','income_comp','schooling']
who
who=who.drop('year',axis=1)
```

```
status=pd.get_dummies(who.status)
who=pd.concat([who,status],axis=1)
who=who.drop(['status'],axis=1)
who.rename(columns={'Developing':0,'Developed':1})
who=who.groupby('country').mean()
who.head()
who.columns
plt.scatter(who['hiv'],who['life_exp'])
plt.xlabel("HIV/AIDS")
plt.ylabel("LIFE EXPECTANCY")
plt.scatter(who['gdp'],who['life_exp'])
plt.xlabel('GDP')
plt.ylabel('LIFE EXPECTANCY')
plt.scatter(who['bmi'],who['life_exp'])
plt.xlabel('BMI')
plt.ylabel('LIFE EXPECTANCY')
plt.scatter(who['alcohol'],who['life_exp'])
plt.xlabel('ALCOHOL')
plt.ylabel('LIFE EXPECTANCY')
plt.scatter(who['adult_mort'],who['life_exp'])
plt.xlabel('ADULT MORTALITY')
plt.ylabel('LIFE EXPECTANCY')
plt.scatter(who['schooling'],who['life_exp'])
plt.xlabel('SCHOOLING')
plt.ylabel('LIFE EXPECTANCY')
plt.scatter(who['perc_exp'],who['life_exp'])
plt.xlabel('PERCENTAGE EXPENDITURE')
plt.ylabel('LIFE EXPECTANCY')
plt.figure(figsize=(14,12))
sns.heatmap(who.corr(),annot=True)
X=who.drop('life_exp',axis=1)
y=who['life_exp']
X.isnull().sum()
y.isnull().sum()
X.fillna(value=X.mean(),inplace=True)
y.fillna(value=y.mean(),inplace=True)
X.isnull().sum()
y.isnull().sum()
stats.describe(X[1:])
sc=MinMaxScaler()
X=sc.fit_transform(X)
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.7,test_size=0.3)
lin_reg=LinearRegression()
lin_reg.fit(X_train,y_train)
print('R_square score on the training data: ',lin_reg.score(X_train,y_train))
print("Coefficients: ",lin_reg.coef_)
print("Mean Squared Error: ",mean_squared_error(y_test,lin_reg_pred))
```

```
print("Absolute Squared Error: ",mean_absolute_error(y_test,lin_reg_pred))
print("R_square Score: ",r2_score(y_test,lin_reg_pred))
scoring=make_scorer(r2_score)
grid_cv=GridSearchCV(Ridge(),param_grid={'alpha':range(0,10),'max_iter':[10,100,10
00]},scoring=scoring,cv=5,refit=True)
grid_cv.fit(X_train,y_train)
print("Best Parameters: "+str(grid_cv.best_params_))
result=grid_cv.cv_results_
print('R_square score on the training data: ',grid_cv.score(X_train,y_train))
print("R_square Score: ",r2_score(y_test,grid_cv.best_estimator_.predict(X_test)))
print("Mean Squared Error: ",mean_squared_error(y_test,lin_reg_pred))
print("Absolute Squared Error: ",mean_absolute_error(y_test,lin_reg_pred))
scoring=make_scorer(r2_score)
grid_cv1=GridSearchCV(Lasso(),param_grid={'alpha':range(0,10),'max_iter':[10,100,1
000]},scoring=scoring,cv=5,refit=True)
grid_cv1.fit(X_train,y_train)
print("Best Parameters: "+str(grid_cv1.best_params_))
result=grid_cv1.cv_results_
print('R_square score on the training data: ',grid_cv1.score(X_train,y_train))
print("R_square Score: ",r2_score(y_test,grid_cv1.best_estimator_.predict(X_test)))
print("Mean Squared Error: ",mean_squared_error(y_test,lin_reg_pred))
print("Absolute Squared Error: ",mean_absolute_error(y_test,lin_reg_pred))
scoring=make_scorer(r2_score)
grid_cv=GridSearchCV(ElasticNet(),param_grid={'alpha':range(0,10),'max_iter':[10,10
0,1000],'l1_ratio':[0.1,0.4,0.8]},scoring=scoring,cv=5,refit=True)
grid_cv.fit(X_train,y_train)
print("Best Parameters: "+str(grid_cv.best_params_))
result=grid_cv.cv_results_
print('R_square score on the training data: ',grid_cv.score(X_train,y_train))
print("R_square Score: ",r2_score(y_test,grid_cv.best_estimator_.predict(X_test)))
print("Mean Squared Error: ",mean_squared_error(y_test,lin_reg_pred))
print("Absolute Squared Error: ",mean_absolute_error(y_test,lin_reg_pred))
quad_reg=PolynomialFeatures(2,interaction_only=True)
quad_reg.fit(X_train)
X_train_quad = quad_reg.transform(X_train)
X_test_quad=quad_reg.transform(X_test)
poly_reg=LinearRegression()
poly_reg.fit(X_train_quad,y_train)
score_poly=poly_reg.score(X_train_quad,y_train)
print("Accuracy: ",score_poly)
poly_reg_predict=poly_reg.predict(X_test_quad)
print("Mean Squared Error: ",mean_squared_error(y_test,poly_reg_predict))
print("Mean Absolute Error: ",mean_absolute_error(y_test,poly_reg_predict))
print("R_Squared Score: ",r2_score(y_test,poly_reg_predict))
dt=DecisionTreeRegressor()
dt_fit=dt.fit(X_train,y_train)
dt_score=cross_val_score(dt_fit,X_train,y_train,cv=5)
```

```
print("Mean Cross Validation Score: ",np.mean(dt_score))
print("Score without CV: ",dt_fit.score(X_train,y_train))
print("R_square Score on the test Data: ",r2_score(y_test,dt_fit.predict(X_test)))
dt_predict=dt.predict(X_test)
scoring=make_scorer(r2_score)
grid_cv=GridSearchCV(DecisionTreeRegressor(),param_grid={'min_samples_split':ran
ge(2,10)},scoring=scoring,cv=5,refit=True)
grid_cv.fit(X_train,y_train)
print("Best Parameters: ",str(grid_cv.best_params_))
result=grid_cv.cv_results_
print("R_squared Score on Training Data:
",grid_cv.best_estimator_.score(X_train,y_train))
print("R_square Score: ",r2_score(y_test,grid_cv.best_estimator_.predict(X_test)))
print("Mean Squared Error: ",mean_squared_error(y_test,dt_predict))
print("Mean Absolute Error: ",mean_absolute_error(y_test,dt_predict))
rf=RandomForestRegressor()
rf_fit=rf.fit(X_train,y_train)
rf_score=cross_val_score(rf_fit,X_train,y_train,cv=5)
print("Mean Cross Validation: ",np.mean(rf_score))
print("Score without CV: ",rf_fit.score(X_train,y_train))
print("R_squared on the test data: ",r2_score(y_test,rf_fit.predict(X_test)))
rf_predict=rf.predict(X_test)
scoring=make_scorer(r2_score)
grid_cv=GridSearchCV(RandomForestRegressor(),param_grid={'min_samples_split':r
ange(2,10)},scoring=scoring,cv=5,refit=True)
grid_cv.fit(X_train,y_train)
result=grid_cv.cv_results_
print("Best Parameters: ",str(grid_cv.best_params_))
print("R_squared Score on Training Data:
",grid_cv.best_estimator_.score(X_train,y_train))
print("R_square Score: ",r2_score(y_test,grid_cv.best_estimator_.predict(X_test)))
print("Mean Squared Error: ",mean_squared_error(y_test,rf_predict))
print("Mean Absolute Error: ",mean_absolute_error(y_test,rf_predict))
!pip install watson-machine-learning-client
from watson_machine_learning_client import WatsonMachineLearningAPIClient
client=WatsonMachineLearningAPIClient(wml_credentials)
model_props={ client.repository.ModelMetaNames.AUTHOR_NAME: "DIXITA",
        client.repository.ModelMetaNames.AUTHOR_EMAIL:
"dixitashukla25@gmail.com",
        client.repository.ModelMetaNames.NAME: "Life Expectancy Prediction"}
model_artifact=client.repository.store_model(grid_cv,meta_props=model_props)
published_model_uid = client.repository.get_model_uid(model_artifact)
published_model_uid
deployment = client.deployments.create(published_model_uid, name="Life
Expectancy Prediction")
scoring_endpoint = client.deployments.get_scoring_url(deployment)
scoring_endpoint
```