ARTIFICIAL INTELLIGENCE

CAPSTONE PROJECT.

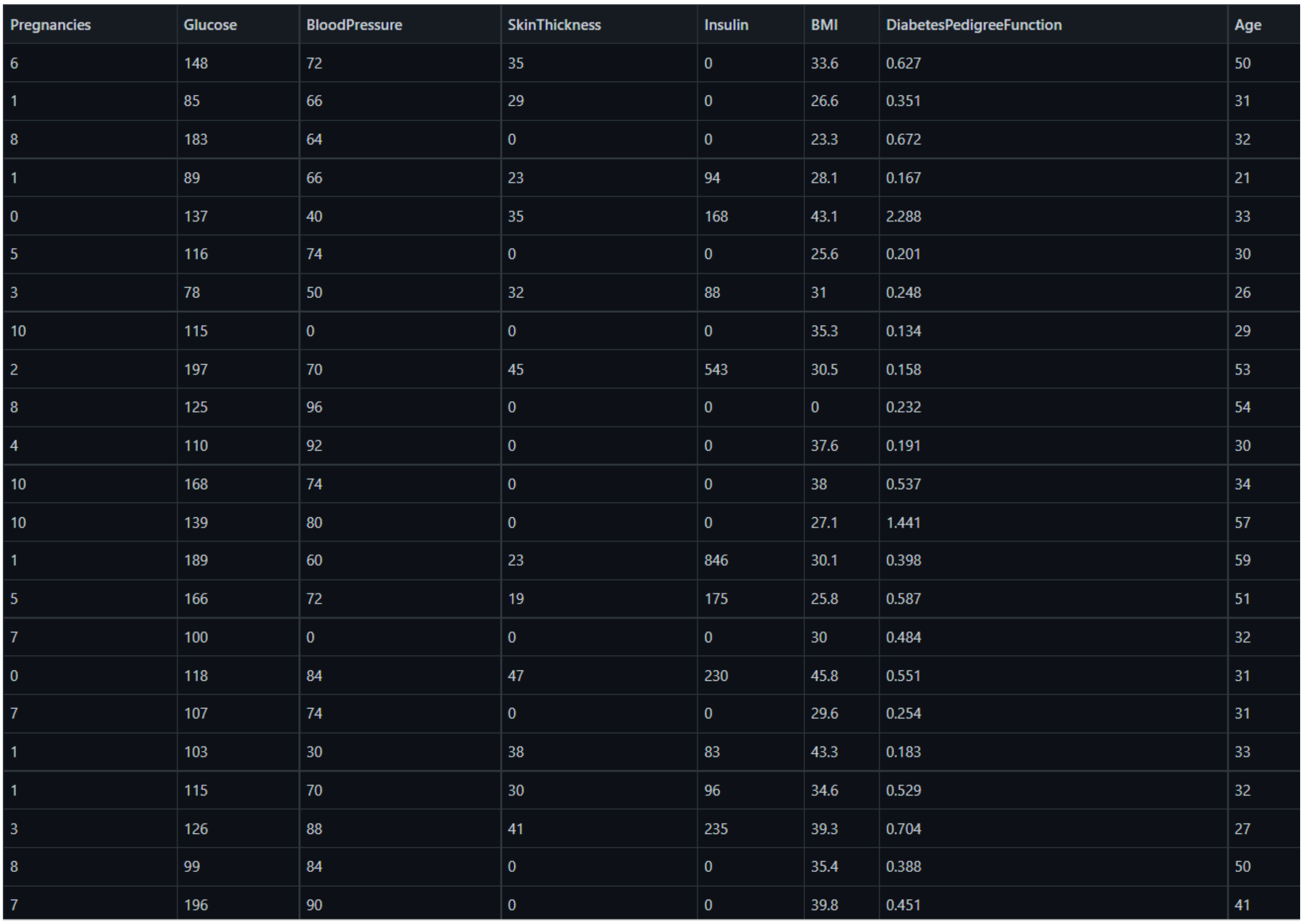
ANIRUDH SENGAR 12-B2

INTRODUCTION

- A Capstone Project is a multifaceted body of work that serves as a culminating academic and intellectual experience for students.
- The final project of an academic program, typically integrating all of the learning from the program is called the Capstone Project.
- A capstone project is a project where students must research a topic independently to find a deep understanding of the subject matter.
- It gives an opportunity for the student to integrate all their knowledge and demonstrate it through a comprehensive project.
- Every AI project lifecycle encompasses three main stages:
    - Stage I - Project planning and data collection
    - Stage II - Design and training of the Machine Learning model
    - Stage III - Deployment and maintenance
- We have taken up the problem of rising diabetes cases in America amongst women and made a program that will make the doctor’s job easier to determine whether the patient is diabetic or not.

DATASET

- We have taken the data of 768 women, out of which 258 tested positive for diabetes and 510 tested negative for diabetes.
- The dataset has been taken from a repository (GitHub).
- Source: https://raw.githubusercontent.com/npradaschnor/Pima-Indians-Diabetes-Dataset/master/diabetes.csv



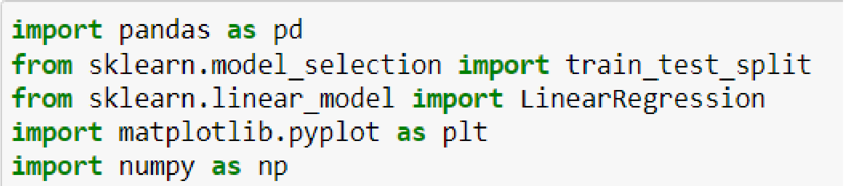
Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
6	148	72	35	0	33.6	0.627	50
1	85	66	29	0	26.6	0.351	31
8	183	64	0	0	23.3	0.672	32
1	89	66	23	94	28.1	0.167	21
0	137	40	35	168	43.1	2.288	33
5	116	74	0	0	25.6	0.201	30
3	78	50	32	88	31	0.248	26
10	115	0	0	0	35.3	0.134	29
2	197	70	45	543	30.5	0.158	53
8	125	96	0	0	0	0.232	54
4	110	92	0	0	37.6	0.191	30
10	168	74	0	0	38	0.537	34
10	139	80	0	0	27.1	1.441	57
1	189	60	23	846	30.1	0.398	59
5	166	72	19	175	25.8	0.587	51
7	100	0	0	0	30	0.484	32
0	118	84	47	230	45.8	0.551	31
7	107	74	0	0	29.6	0.254	31
1	103	30	38	83	43.3	0.183	33
1	115	70	30	96	34.6	0.529	32
3	126	88	41	235	39.3	0.704	27
8	99	84	0	0	35.4	0.388	50
7	196	90	0	0	39.8	0.451	41

STAGE I – PROBLEM SCOPING

- Problem Scoping: It involves identifying a problem and having a vision to solve it. It involves a series of steps to narrow down to a problem statement from a broad theme. It is basically selecting a problem which we want to solve using our AI Knowledge.
- Since the cases of diabetes in near Phoenix, Arizona, USA are increasing at a rapid rate we decided the take up this issue and make it easier for the doctors and the medical industry to determine quickly whether the patient has diabetes or not and treat them as soon as possible.
- The required data for this model are as follows -
    - Number of pregnancies
    - Glucose Levels
    - Blood Pressure Levels
    - Thickness of the skin
    - Insulin Levels
    - Body Mass Index (BMI)
    - Diabetes Pedigree Function (Family history)
    - Age
    - Diabetic or not

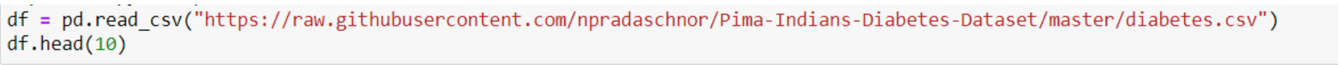
STAGE II-
BUILD/MODEL PHASE

- First, we import all the necessary modules for our AI model.

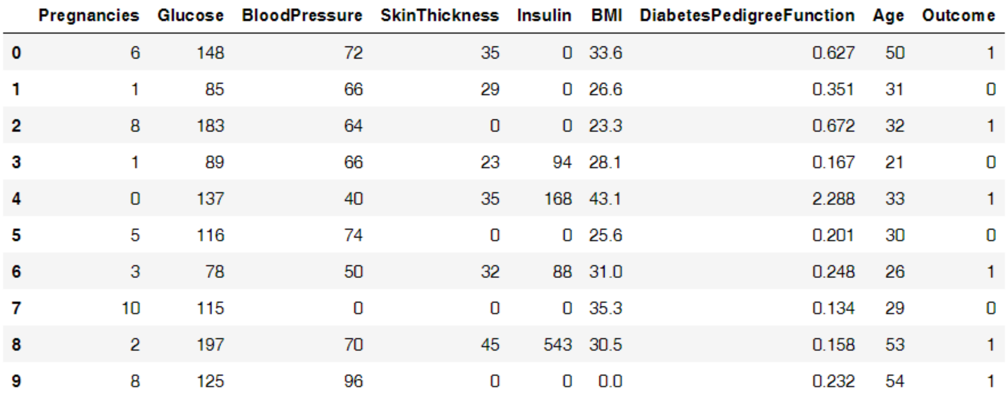


```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import numpy as np
```

- Second, we load the dataset in Jupyter Notebook and Display the first 10 elements using the .head() function.

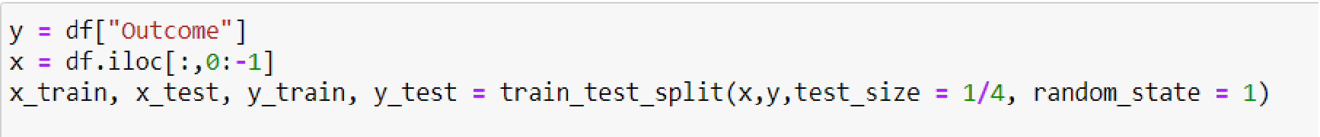


```
df = pd.read_csv("https://raw.githubusercontent.com/npradaschnor/Pima-Indians-Diabetes-Dataset/master/diabetes.csv")
df.head(10)
```



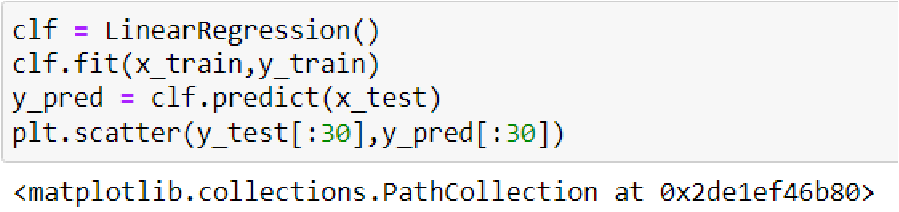
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

- Now we declare the Dependent(y) and Independent(x) variables and divide the data into train and test data using train_test_split.



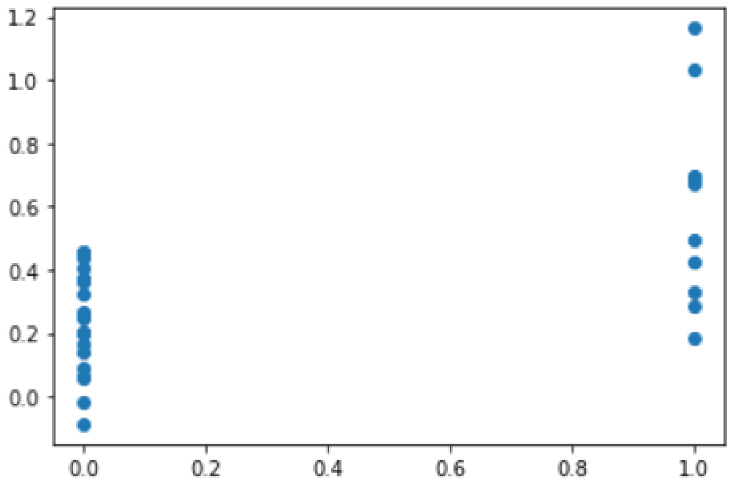
```
y = df["Outcome"]
x = df.iloc[:,0:-1]
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 1/4, random_state = 1)
```

- Now we use the Linear Regression module to predict the values of the test data after learning the training data. We can then plot the graph using matplotlib to show the spread of the data.



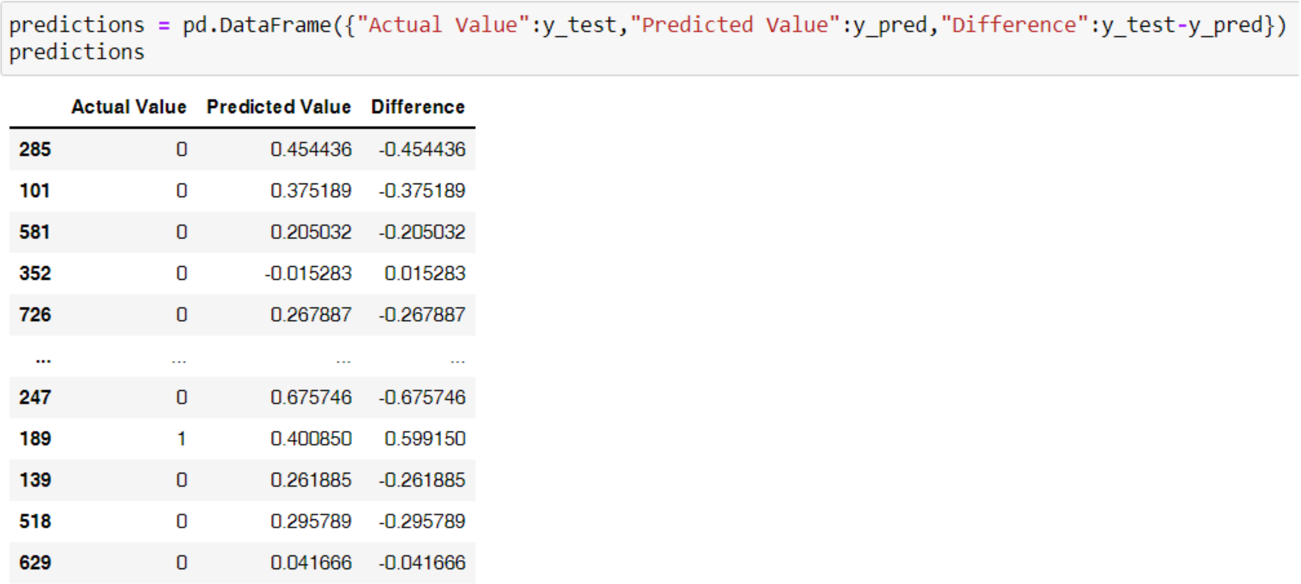
```
clf = LinearRegression()
clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)
plt.scatter(y_test[:30],y_pred[:30])
```

```
<matplotlib.collections.PathCollection at 0x2de1ef46b80>
```



Note: Here 0 means the patient is not diabetic and 1 means that the patient is diabetic.

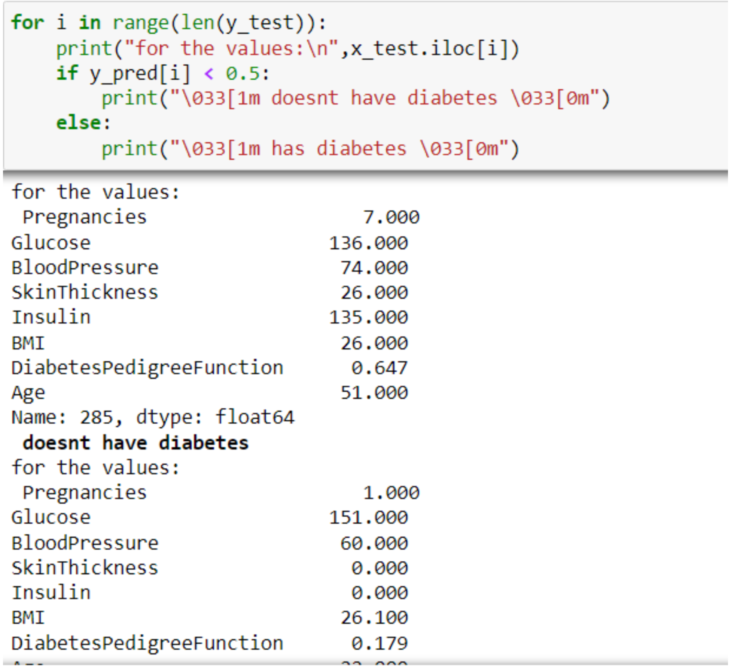
- We can check the difference of the predicted value and the actual value and display them as a dataframe.



```
predictions = pd.DataFrame({"Actual Value":y_test,"Predicted Value":y_pred,"Difference":y_test-y_pred})
predictions
```

	Actual Value	Predicted Value	Difference
285	0	0.454436	-0.454436
101	0	0.375189	-0.375189
581	0	0.205032	-0.205032
352	0	-0.015283	0.015283
726	0	0.267887	-0.267887
...	...	...	...
247	0	0.675746	-0.675746
189	1	0.400850	0.599150
139	0	0.261885	-0.261885
518	0	0.295789	-0.295789
629	0	0.041666	-0.041666

- We can display all the predictions of the test dataset and check whether the prediction is accurate or not.

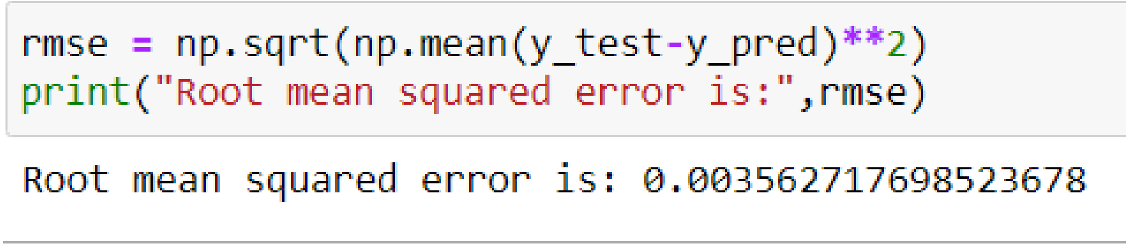


```
for i in range(len(y_test)):
    print("for the values:\n",x_test.iloc[i])
    if y_pred[i] < 0.5:
        print("\033[1m doesnt have diabetes \033[0m")
    else:
        print("\033[1m has diabetes \033[0m")
```

```
for the values:
 Pregnancies                  7.000
Glucose                     136.000
BloodPressure                74.000
SkinThickness                26.000
Insulin                     135.000
BMI                          26.000
DiabetesPedigreeFunction      0.647
Age                          51.000
Name: 285, dtype: float64
 doesnt have diabetes
for the values:
 Pregnancies                  1.000
Glucose                     151.000
BloodPressure                60.000
SkinThickness                 0.000
Insulin                       0.000
BMI                          26.100
DiabetesPedigreeFunction      0.179
```

- We can find the Root Mean squared error using the formula

$$RMSE=\sqrt{\frac{\sum_{i=1}^{N}(Predicted_i-Actual_i)^2}{N}}$$



```
rmse = np.sqrt(np.mean(y_test-y_pred)**2)
print("Root mean squared error is:",rmse)
```

```
Root mean squared error is: 0.003562717698523678
```

- We got an RMSE value of 0.0035 which means that the model can predict with high accuracy.

STAGE III - DEPLOYMENT AND MAINTENANCE

- The process of taking a trained ML model and making its predictions available to users or other systems is known as deployment.
- Model Deployment helps you showcase your work to the world and make better decisions with it.
- We can use this model in various hospitals in Phoenix Arizona and test the accuracy of the model and make improvements based on the feedback received.
- We can revert back to the build and model phase in order to make changes in the data of the model or the model itself.