~MDA 620 Project Report~

Diabetes Prediction

Group Members-

Pritika Khurana

Lexi DeLorimiere

Shivani Chandramohan

TABLE OF CONTENTS

1. **BACKGROUND**
2. **PROBLEM SCENARIO**
3. **GOALS**
4. **DATASET**
5. **DATA DICTIONARY**
    - **Glucose Tolerance Test**
    - **Blood Pressure**
    - **Body Mass Index**
    - **Triceps Skinfolds**
    - **Causes of Diabetes**
6. **PROPOSED METHODS**
    - **DATA COLLECTION**
    - **DATA PRE-PROCESSING**
    - **SPLITTING OF DATA**
7. **DATA EXPLORATION**
8. **DATA VISUALIZATION**
9. **DATA MANIPULATION**
    - **SCALING AND NORMALIZATION**
10. **MODEL BUILDING AND ANALYSIS**
    - **SUPPORT VECTOR MACHINE**
    - **GOALS OF SVM ALGORITHM**
    - **TYPES OF SVM**
    - **CONFUSION MATRIX**
11. **CONCLUSION**

BACKGROUND

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The goal of the dataset is to accurately predict whether or not a patient has diabetes, based on certain diagnostic measurements in the dataset. Several limitations were placed on the selection of these instances from a larger database. Importantly, all patients within this dataset are females at least 21 years old of Pima Indian heritage.

The dataset consists of varying medical predictor variables and the one target variable being outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, etc.

PROBLEM SCENARIO

All around the world there are numerous infections, one being diabetes which is the deadliest sickness a person could have. Diabetes is a metabolic issue that causes blood sugar by producing a great degree of insulin in the body. Diabetes can even cause further damage to the body with illnesses such as coronary failure, visual deficiency, kidney ailments, nerve harm, and more.

In this project, we aim to accurately predict whether a person has diabetes or not based on various factors provided to the model we train. It is important that an individual monitors their blood sugar level to manage their diabetes and prevent further complications.

GOALS

1. Predict with precision the patient being diabetic or not, with the help of a machine learning model.
2. Early detection and supervision of ailment

DATASET

The dataset is originally from the Pima Indians Diabetes Database available on Kaggle. It consists of several medical analyst variables and one target variable.The objective of the dataset is to predict whether the patient has diabetes or not. The dataset consists of several independent variables and one dependent variable,(i.e., the outcome). Independent variables include the number of pregnancies the patient has had, BMI,insulin level, age, etc.

DATA DICTIONARY

The data set consists of 2000 data points with 9 features each.

Serial No.	**Attribute Names**	**Description**
1	Pregnancies	Number of pregnancies
2	Glucose	Plasma glucose concentration
3	Blood Pressure	Diastolic blood pressure
4	Skin Thickness	Triceps skin fold thickness (mm)
5	Insulin	2-h serum insulin
6	BMI	Body mass index
7	Diabetes pedigree function	Diabetes pedigree function
8	Outcome	Class variable (0 or 1)
9	Age	Age (years)

Listed below is the general information on the variables:

Glucose Tolerance Test

It is a blood test that involves taking multiple blood samples over roughly 2 hours. It is the primary test used to diagnose diabetes. The results can be classified as normal, impaired, or abnormal.

- **Normal Results for Diabetes ->** Two-hour glucose level less than 140 mg/dL
- **Impaired Results for Diabetes ->** Two-hour glucose level 140 to 200 mg/dL
- **Abnormal (Diagnostic) Results for Diabetes ->** Two-hour glucose level greater than 200 mg/dL

Blood Pressure

The bottom number(diastolic) is the pressure in the arteries when the heart rests between beats. This is the time when the heart fills with blood and gets oxygen. A normal diastolic blood pressure is lower than 80. A reading of 90 or higher would be considered as high blood pressure.

- **Normal**: Systolic below 120 and diastolic below 80
- **Elevated**: Systolic 120–129 and diastolic under 80
- **Hypertension stage 1**: Systolic 130–139 and diastolic 80–89
- **Hypertension stage 2**: Systolic 140-plus and diastolic 90 or more
- **Hypertensive crisis**: Systolic higher than 180 and diastolic above 120.

Body Mass Index

The standard weight status categories associated with BMI ranges for adults are shown in the following table.

- Below 18.5 -> **Underweight**
- 18.5 – 24.9 -> **Normal or Healthy Weight**
- 25.0 – 29.9 -> **Overweight**
- 30.0 and Above -> **Obese**

Triceps Skinfolds

For adult women, the standard normal values for triceps skinfolds are 18.0mm

Causes of Diabetes

Diabetes is mainly caused because of genetic factors. It is caused by at least two mutant genes in chromosome 6. There are two types of diabetes: 1 and 2.

Type 1: Diabetes occurs when cells fail to produce insulin in sufficient amounts. There is no research that proves the causes of type 1 diabetes and also no known methods of prevention currently.

Type 2: Diabetes occurs when cells produce a low quantity of insulin. This is the most common type of diabetes that is affecting 90% of the population.

PROPOSED METHODS

Dataset collection

It includes data collection and understanding the data to study the hidden patterns and trends which helps to predict the results. Dataset carries 1405 rows i.e., total number of data and 10 columns i.e., total number of features. Features include Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI,DiabetesPedigreeFunction, Age.

Data Pre-processing

This phase of the model handles inconsistent data in order to get more accurate and precise results like in this dataset Id are inconsistent so we dropped the feature.This dataset doesn’t contain missing values. The data was scaled using StandardScaler. Since there were a smaller number of features and important for prediction so no feature selection was done.

Splitting of Data

After data cleaning and preprocessing, the dataset becomes ready to train and test. In the train/split method, we split the dataset randomly into the training and testing set. For training, we took 80% of the samples and for testing we took 20% of the samples.

DATA EXPLORATION

In Python, we used various libraries to construct a dataframe. We created a table to visualize the findings. There are a total of 768 rows and 9 columns to display all of the variables.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
import seaborn as sns
```

	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

Data types of each variable:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
```

No null values of each variable were found:

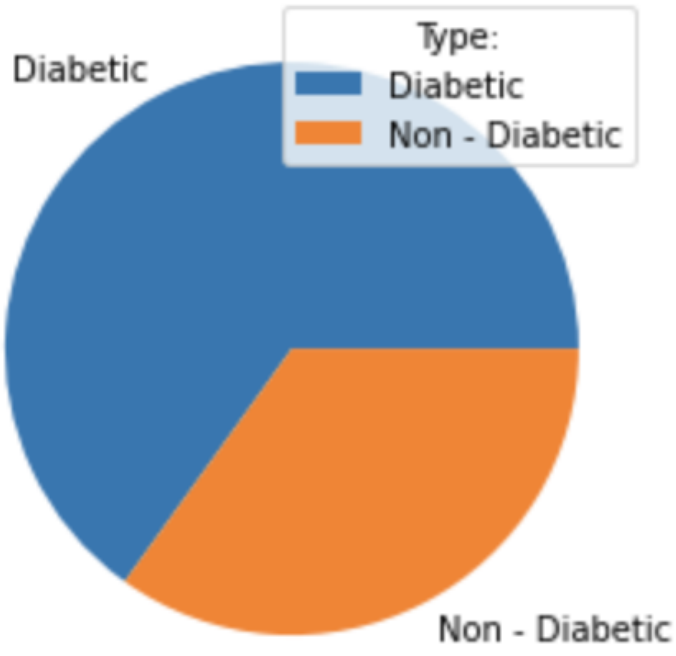
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

The statistical measures of the data:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

DATA VISUALIZATION

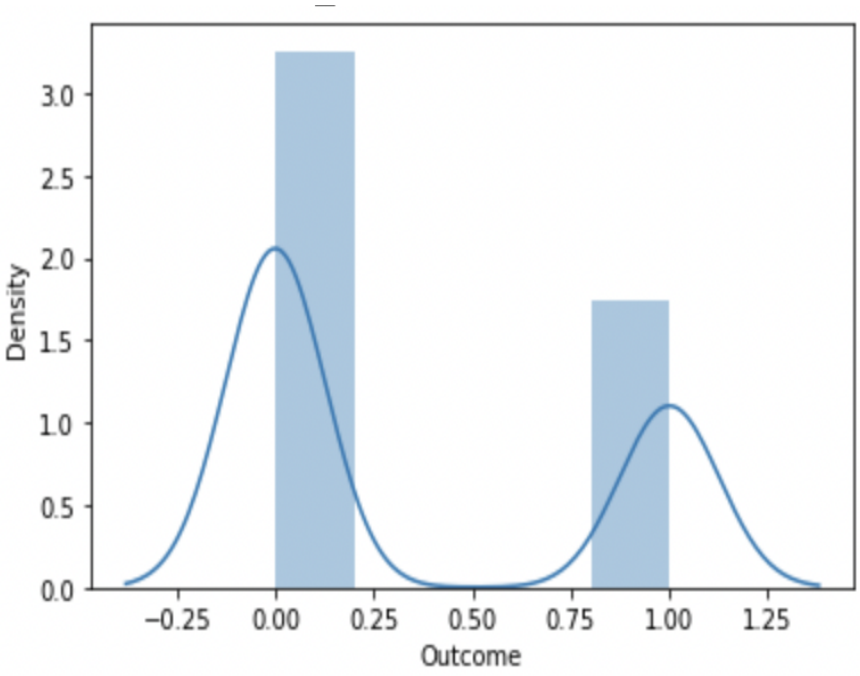
We created a pie and bar chart to visualize the outcomes of patients who are diabetic or not. From the 768 patients, 500 were found to be diabetic and the remaining 268 were not.



The displot refers to the distribution plot.It takes an input as an array and plots a curve corresponding to the distribution of points in the array.

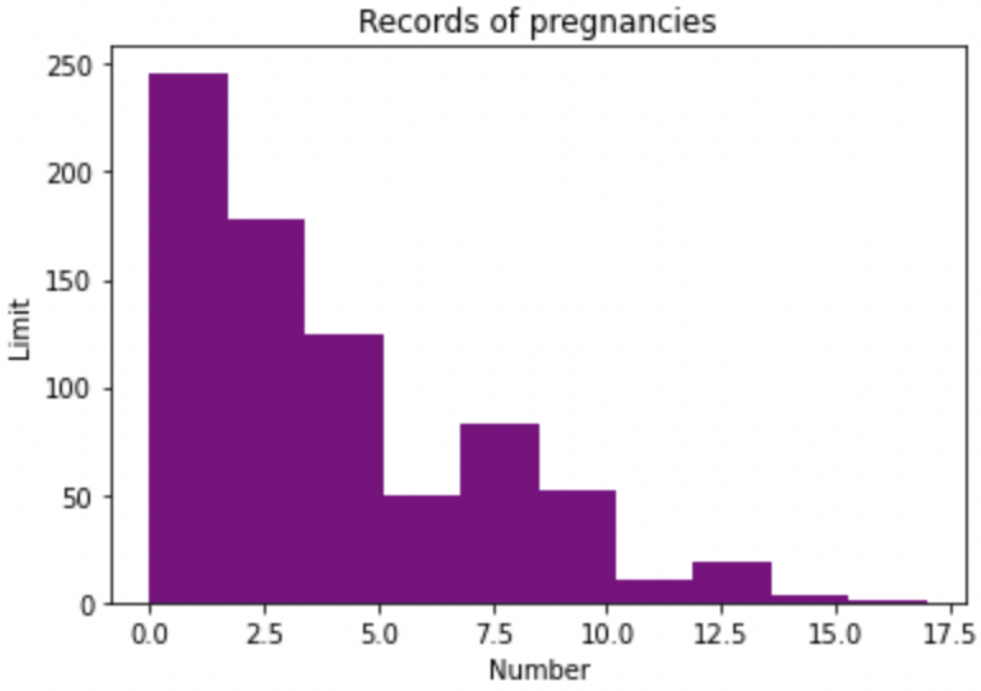
This plot displays two different plots for the same variable.

```
sns.distplot(df['Outcome'])
```



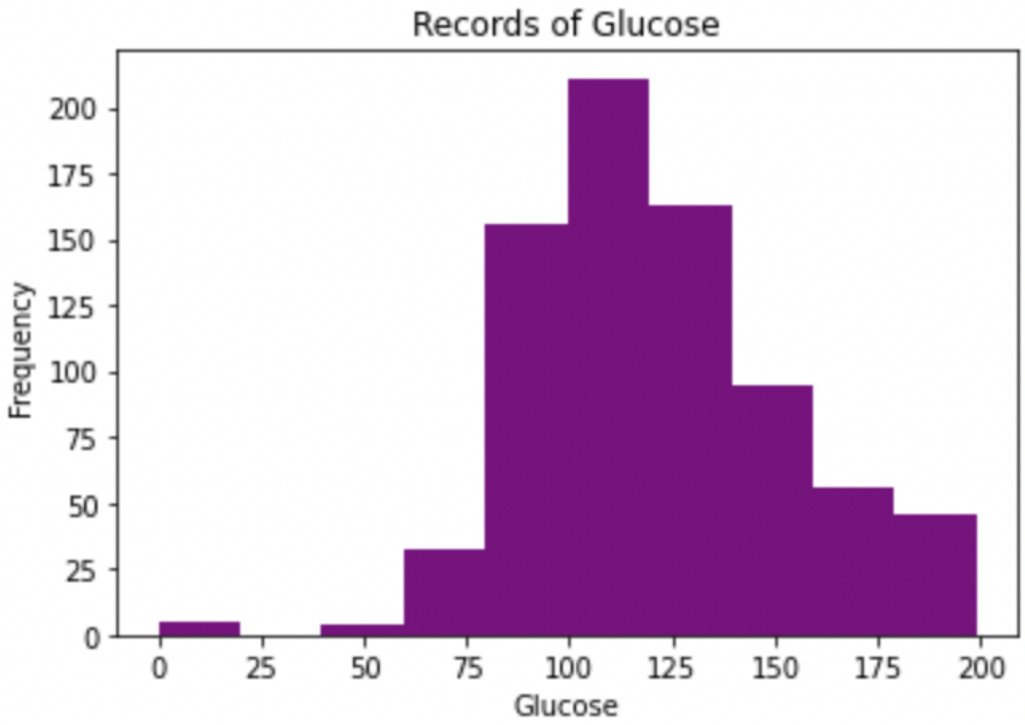
The below histogram we see that women who get pregnant more have a higher chance of testing positive for diabetes.

```
plt.hist(df['Pregnancies'],color='purple')
plt.title('Records of pregnancies')
plt.xlabel('Number')
plt.ylabel('Limit')
```



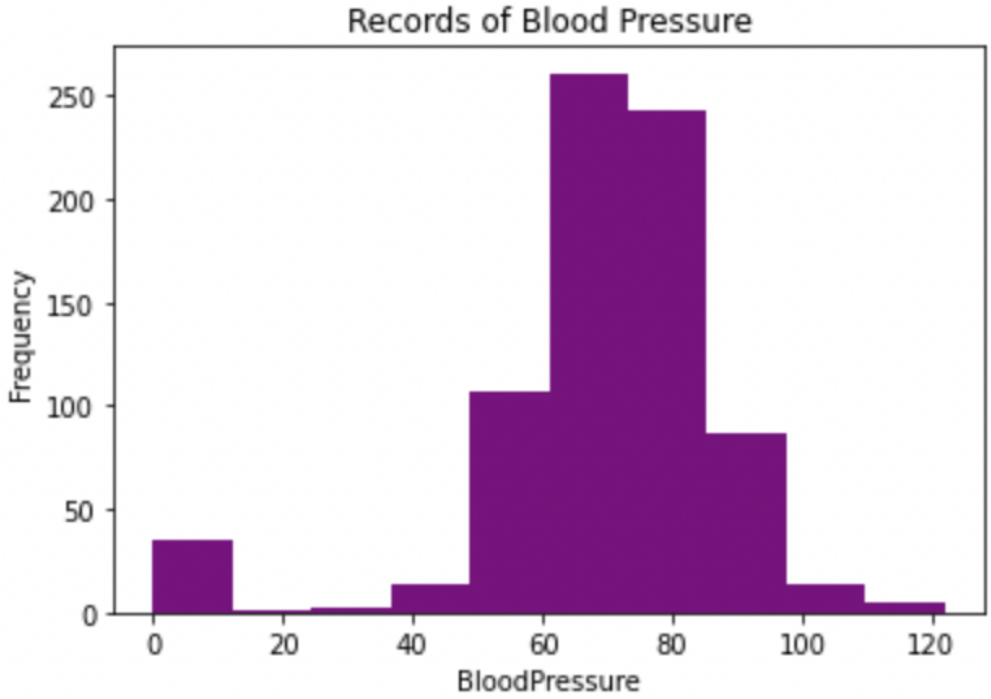
The plot below represents the average Glucose levels for everyone in the surveyed dataset.

```
plt.hist(df['Glucose'],color='purple')
plt.xlabel("Glucose")
plt.ylabel("Frequency")
plt.title('Records of Glucose')
```



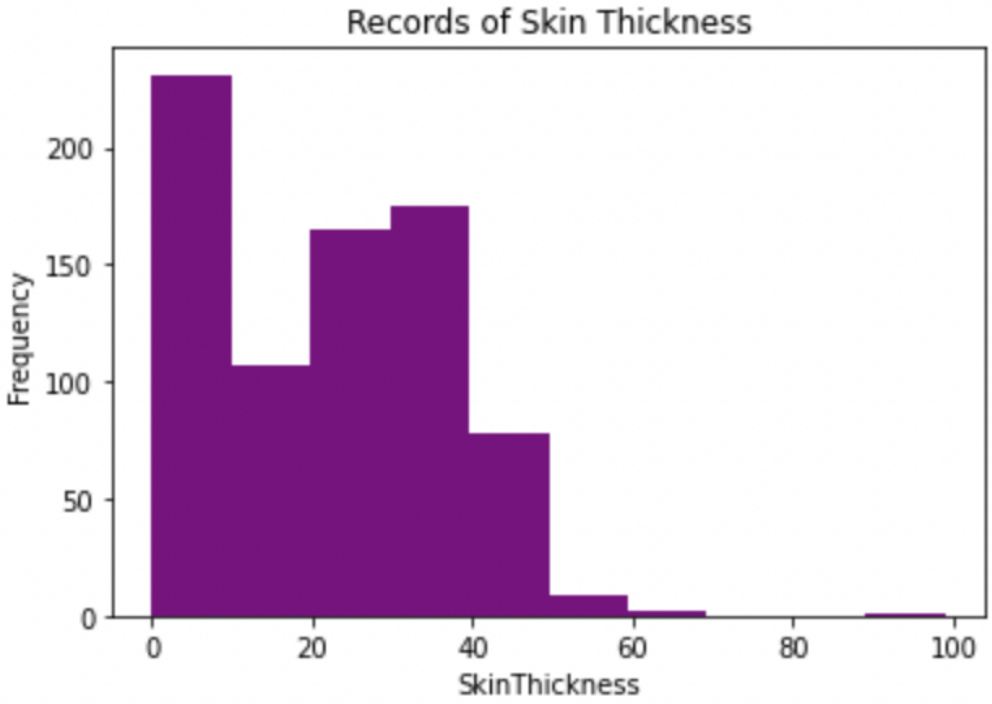
The below graph shows that the people who have low blood pressure have a high frequency of getting diabetes.

```
plt.hist(df['BloodPressure'],color='purple')
plt.xlabel("BloodPressure")
plt.ylabel("Frequency")
plt.title('Records of Blood Pressure')
```



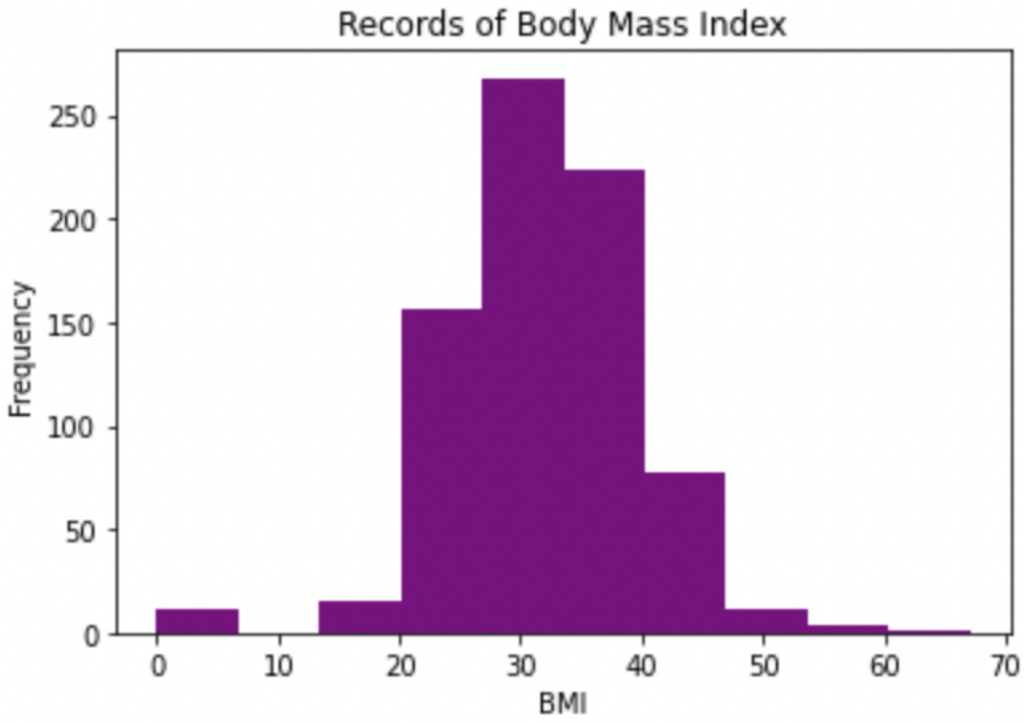
The graph tells us that most people with diabetes have slightly thicker skin than the people that don't have diabetes.

```
plt.hist(df['SkinThickness'],color='purple')
plt.xlabel("SkinThickness")
plt.ylabel("Frequency")
plt.title('Records of Skin Thickness')
```



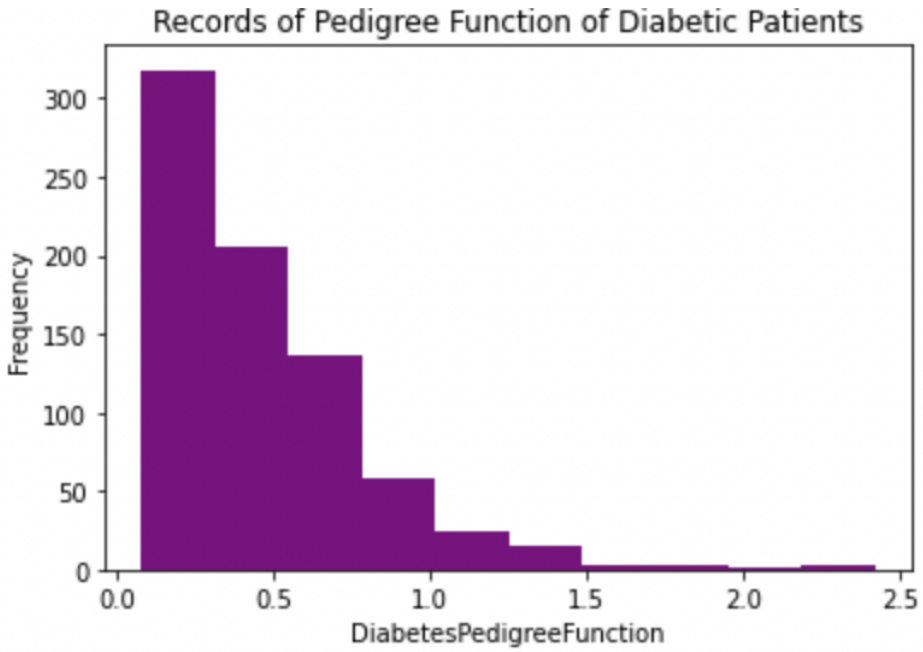
The below histogram shows that people who lie between the BMI of 20 - 40 are having diabetes.

```
plt.hist(df['BMI'],color='purple')
plt.xlabel("BMI")
plt.ylabel("Frequency")
plt.title('Records of Body Mass Index')
```



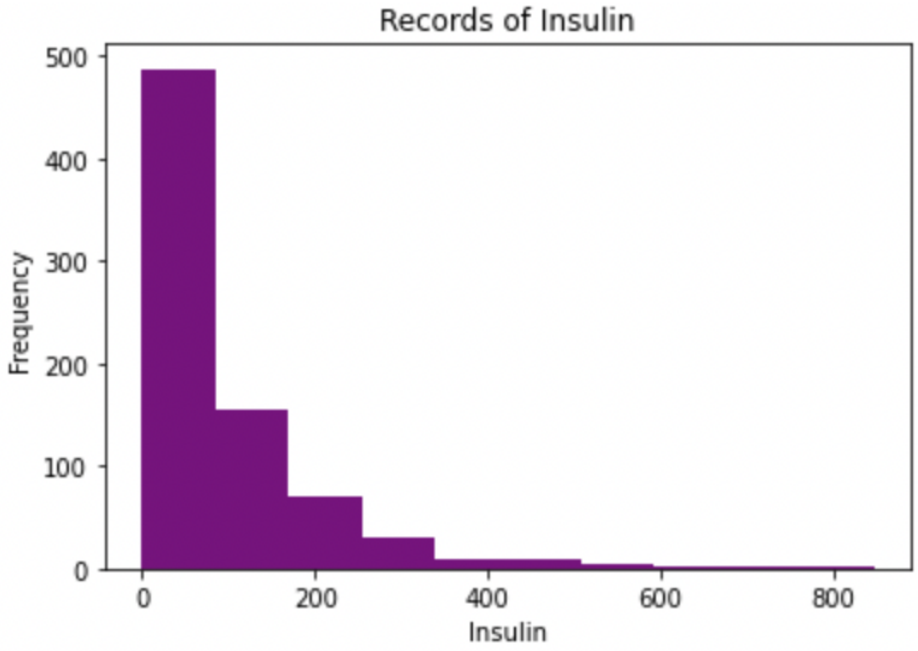
The below graph shows the type of diabetes.

```
plt.hist(df['DiabetesPedigreeFunction'],color='purple')
plt.xlabel("DiabetesPedigreeFunction")
plt.ylabel("Frequency")
plt.title('Records of Pedigree Function of Diabetic Patients')
```



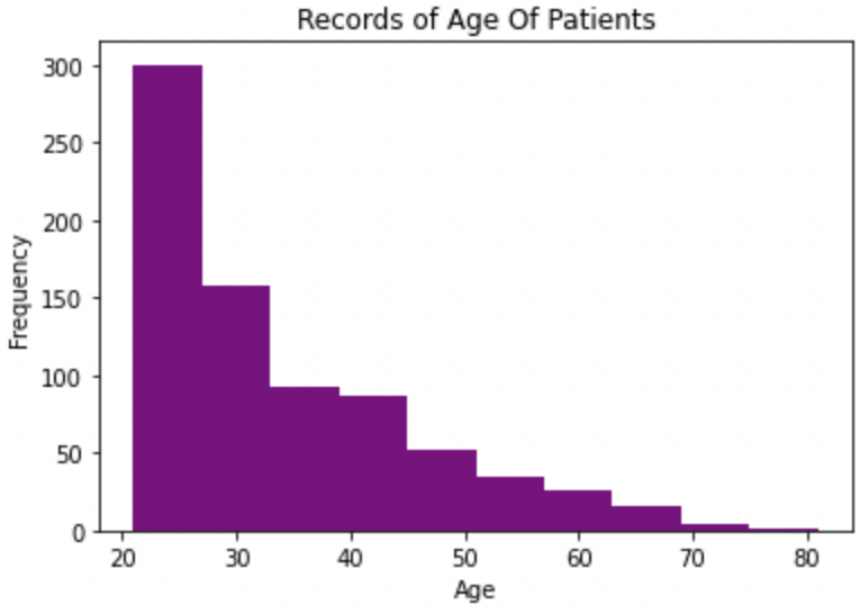
From the histograms below, we see a huge difference in the values. If you have higher insulin levels according to this dataset it is more likely that you have diabetes.

```
plt.hist(df['Insulin'],color='purple')
plt.xlabel("Insulin")
plt.ylabel("Frequency")
plt.title('Records of Insulin')
```



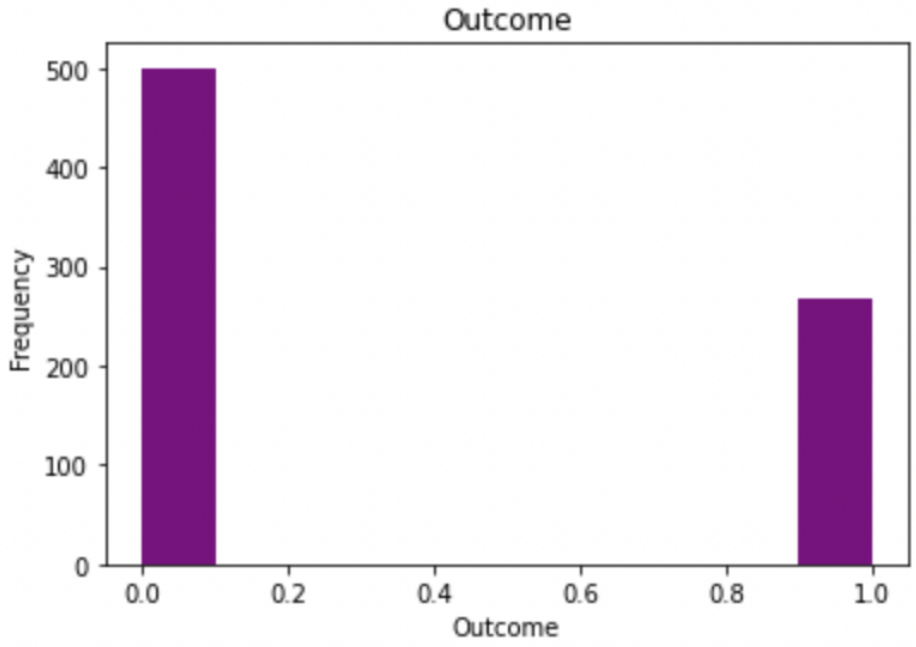
From the below histogram we see that there is a high density of young people testing positive for diabetes.

```
plt.hist(df['Age'],color='purple')
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title('Records of Age Of Patients')
```



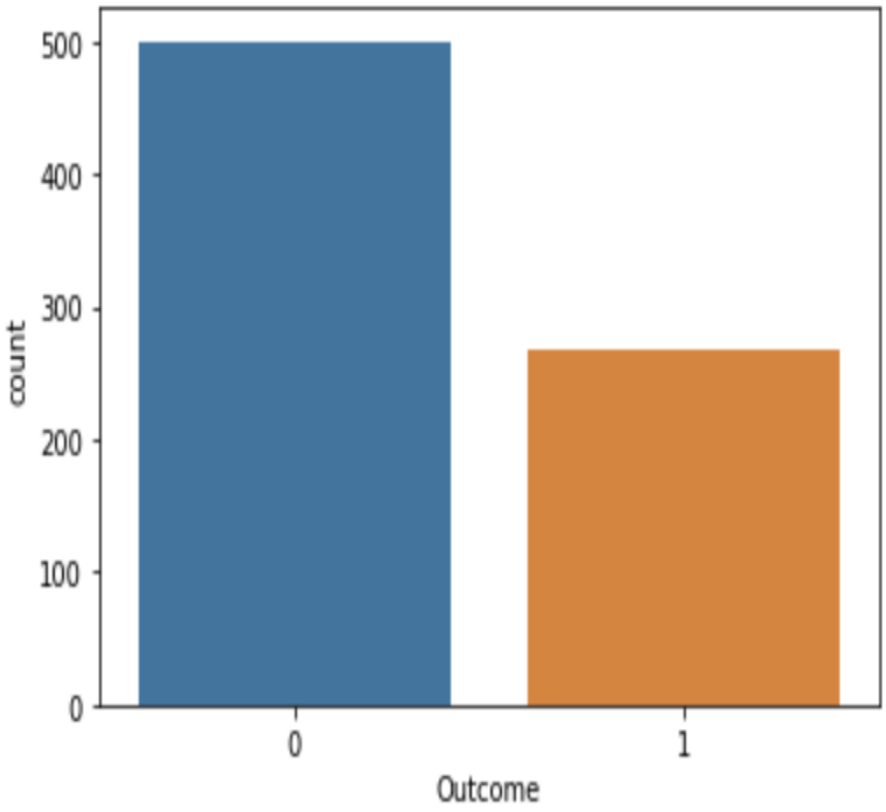
The below graph shows the number of people who have diabetes and who do not have.

```
plt.hist(df['Outcome'],color='purple')
plt.xlabel("Outcome")
plt.ylabel("Frequency")
plt.title('Outcome')
```



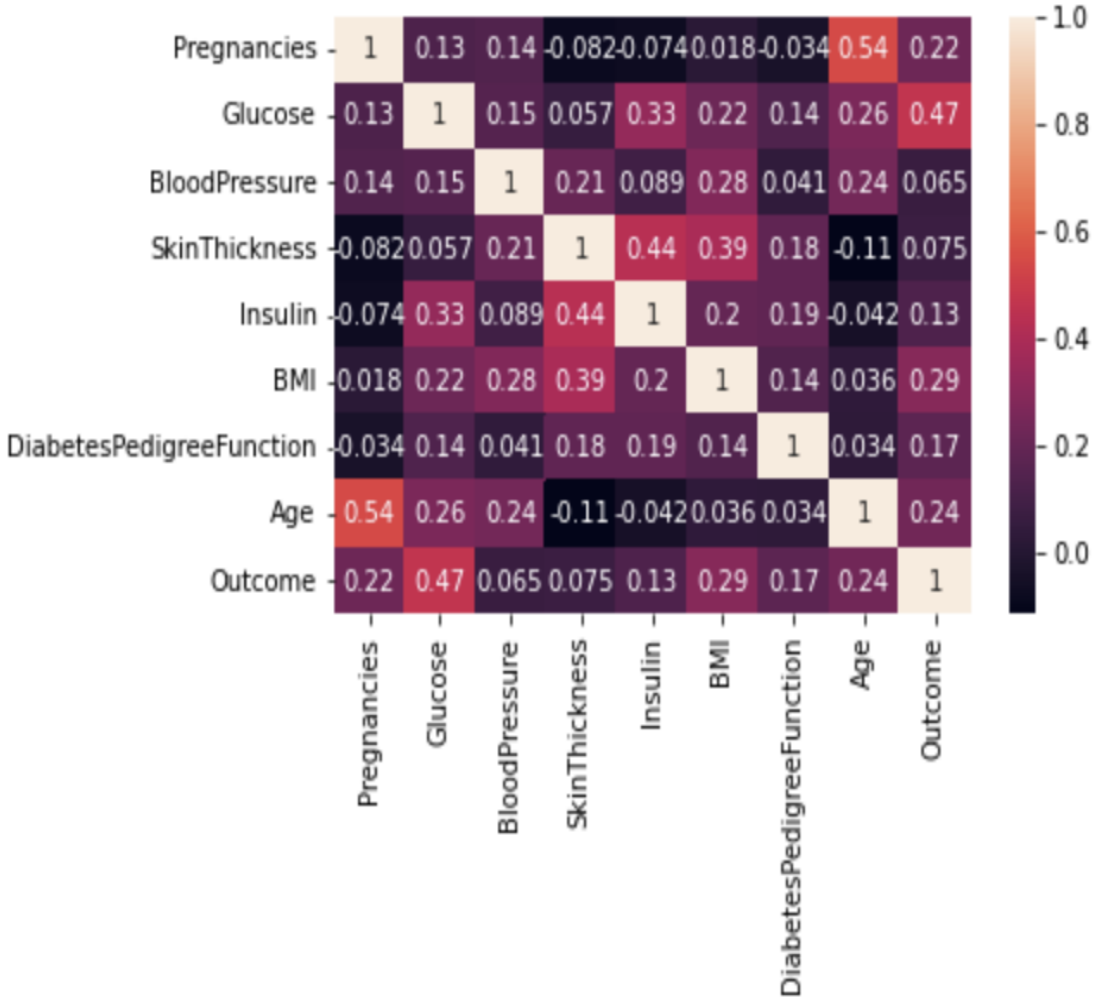
The below graph shows that the data is biased towards data points having outcome value as 0 where it means that diabetes was not present

actually. The number of non-diabetics is almost twice the number of diabetic patients.



We created pairplots for each of the factors so we could visualize the findings for each of the factors.

We created a heatmap to gather a comprehensive overview of the findings.



DATA MANIPULATION

Scaling and Normalization:

In order to make the correct predictions, we needed to standardize the data into a consistent format. Performing feature scaling by normalizes the data from range 0 to 1, boosting the algorithm's calculation speed. Scaling means that you're transforming your data so that it fits within a specific scale, like 0-100 or 0-1.

```
[20] scaler = StandardScaler()
```

```
[21] scaler.fit(x)

     StandardScaler()
```

```
[22] sd = scaler.transform(x)
```

```
[23] print(sd)

     [[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
        1.4259954 ]
      [-0.84488505 -1.12339636 -0.16054575 ... -0.68442195 -0.36506078
       -0.19067191]
      [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
       -0.10558415]
      ...
      [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
       -0.27575966]
      [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
        1.17073215]
      [-0.84488505 -0.8730192   0.04624525 ... -0.20212881 -0.47378505
       -0.87137393]]
```

```
[24] x = sd
     y = df['Outcome']
```

```
print(x)
print(y)
```

```
[[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
   1.4259954 ]
 [-0.84488505 -1.12339636 -0.16054575 ... -0.68442195 -0.36506078
  -0.19067191]
 [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
  -0.10558415]
 ...
 [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
  -0.27575966]
 [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
   1.17073215]
 [-0.84488505 -0.8730192   0.04624525 ... -0.20212881 -0.47378505
  -0.87137393]]
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

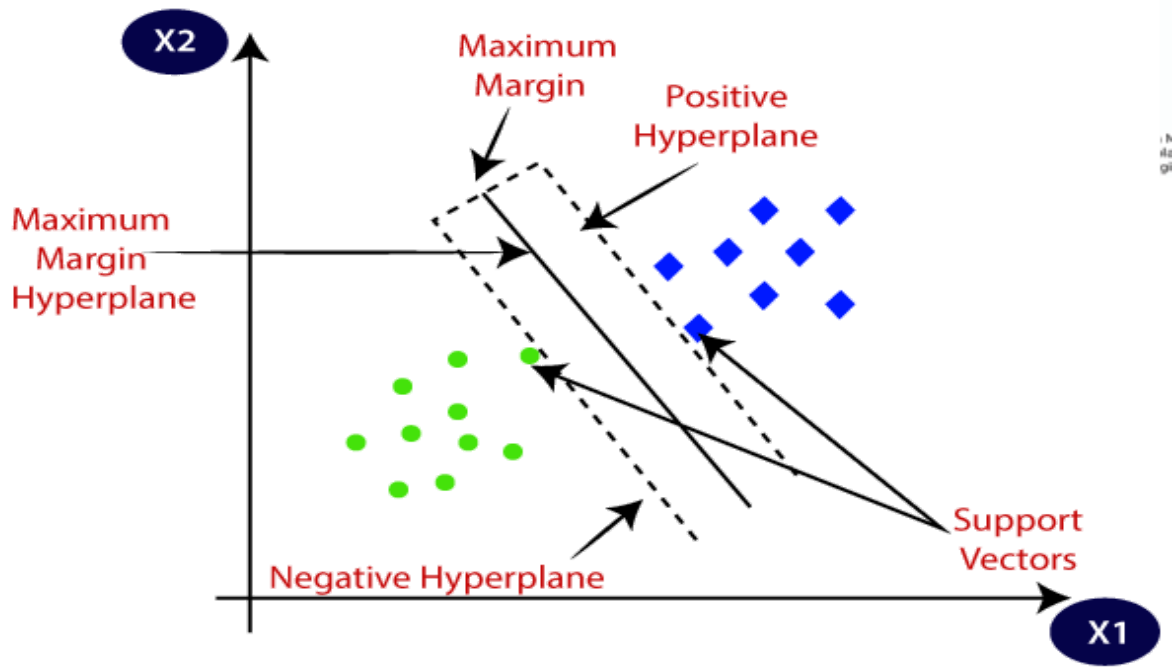
MODEL BUILDING AND ANALYSIS

SUPPORT VECTOR MACHINE :

Support Vector Machine or SVM is a Supervised Learning algorithm, which is used for Classification as well as Regression problems. It is primarily used for Classification problems in Machine Learning.

Goal of SVM algorithm:

To Create the best line or decision boundary called a hyperplane that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.



SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called the margin. **The main goal of SVM is to maximize this margin.** The hyperplane with maximum margin is called the optimal hyperplane.

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

We built and trained a model to return the results of a given input specified to the factors to predict if someone would have diabetes or not. Our model shows to have 77% accuracy.

```
#accuracy score of training data
x_train_prediction = classifier.predict(x_train)
training_data_accuracy = accuracy_score(x_train_prediction, y_train)
```

```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  0.7866449511400652
```

```
#accuracy sccore of test data
x_test_prediction = classifier.predict(x_test)
test_data_accuracy = accuracy_score(x_test_prediction, y_test)
```

```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  0.7727272727272727
```

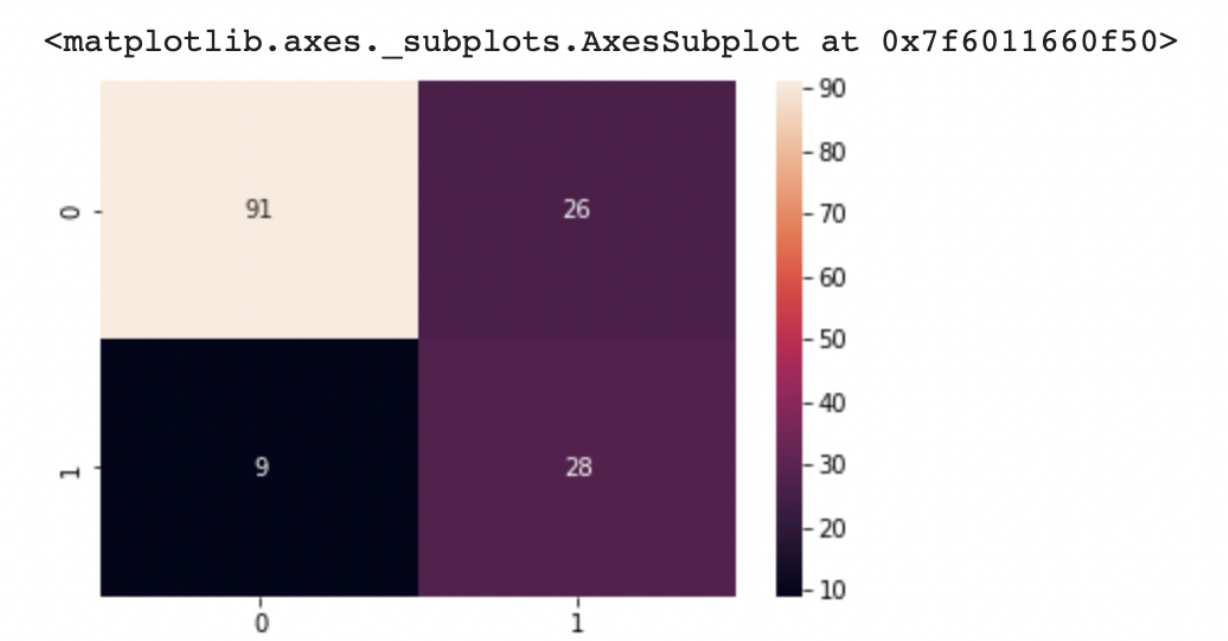
```
# Confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(x_test_prediction, y_test)
cm
```

```
array([[91, 26],
       [ 9, 28]])
```

```
# Heatmap of Confusion matrix
sns.heatmap(pd.DataFrame(cm), annot=True)
```

Confusion matrix

Depicts the output in a matrix format with complete description performance of the model.



The following performance metrics are used:

True positive (TP) – person has diabetes, and the prediction is positive

True negative (TN) – person not having diabetes and the prediction is negative

False positive (FP) – person not having diabetes but the prediction is positive

False negative (FN) – person having diabetes and the prediction is negative

Calculation Measures:

- **True positive and true negative** = used to calculate accuracy rate.
- **Error rates =** used to calculate false positive and false negative values.
- **True positive rate** = used to calculate true positive by a total number of persons who truly have diabetes.
- **False positive rate =** used to calculate false positives by a total number of persons who truly do not have diabetes.
- **Precision =** True positive divided by the total number of people who have a prediction result of true.
- **Accuracy** = Number of correct prediction / total number of predictions made

From the given input example below, our model predicted that this person would not be diabetic.

```
input_data = (4,110,92,0,0,37.6,0.191,30)


#changing the input_data to numpy array
input_data_as_numpy_array =np.asarray(input_data)

#reshape  the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1, -1)

#standardize the input data
std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0) :
  print('The person is not diabetic')
else:
  print('The person is diabetic')
```

CONCLUSION

After analyzing the data, we found that over half the patients in our dataset have diabetes. Since glucose is the main factor contributing to diabetes, we also found out that patients with high glucose levels are diabetic. This data is important because it shows the positive correlation to diabetes in the Pima Indian heritage. The results were surprising because of how prevalent diabetes is among these women. While it is interesting to see the results of this particular group, we would recommend expanding research to more categories of people. Expanding research could lead to finding more relationships between individuals and diabetes.

REFERENCES

- ***https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm***
- ***Sahoo, K.S., et al.: An evolutionary SVM model for DDOS attack detection in software defined networks. IEEE Access 8, 132502–132513 (2020)***