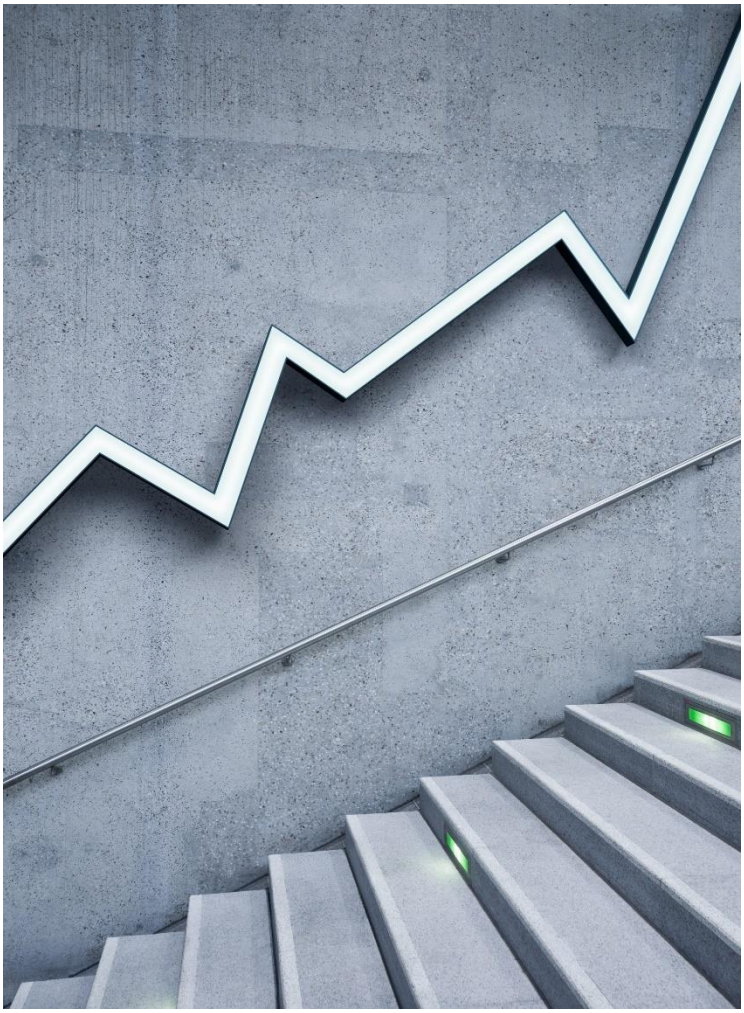


Diabetes Prediction Model

Submitted by-

Mr. Mayur Dev Sewak

General Manager, Operations
Eisystems Services

&

Ms. Mallika Srivastava
Trainer, Data Science & Analytics Domain
Eisystems Services

Submitted to-

Harshit Singh

[Vishveshwarya Group of Instituitions]

Content Table

Link of Image-https://miro.medium.com/max/1200/1*INSggrGiQ1lCgU8YTsfEVw.png

Diabetes Prediction model

Overview

In this article ,we will be predicting that whether the patient has diabetes or not on the basis of the basis of the features we will provide to our machine learning model , and for that ,we will be using the datasets from Kaggle [website for explore and build data models]

Introduction

Information mining is a powerful way to drawing meaningful information from datasets containing massive embedded data. Data mining may be fruitfully exploited in hospitals where a huge volume of data exists. These hospital datasets often need to be clustered, or even further classified to gain a meaningful analysis of the underlying data. Soft computing techniques such as pattern recognition (PR) and machine learning (ML) have been used for identifying statistical parameters from the dataset. A World Health Organization (WHO) report claims that almost 422 million people worldwide are suffering from both TYPE-1 and TYPE-2 diabetes (https://www.who.int/health-topics/diabetes). Also, as released by the WHO, India's profile shows that 46% of the total population have prevailing diabetes and related risk factors.

Diabetes Mellitus (DM) is mainly sorted as three types: (a) diabetes mellitus or DM, (b) insulin resistance, and (c) the third one is gestational diabetes generally found amid pregnant ladies. DM is generally caused by high blood glucose levels and is a typical disease that influences those individuals who have imbalance in blood glucose levels, especially for pregnant women. In fact, past research has demonstrated that pregnant women with diabetes are increasingly inclined to have newborns with birth defects than those without diabetes. Kanguru, et al. (2014) notes that, in such situation, the child may be influenced by prevailing illness conditions, for example, coronary illness and spina bifida. The beginning stage for living admirably with DM is an early diagnosis. Hence, the primary aim of our work is to predict diabetic or non-diabetic using ML and deep learning algorithms. A major study of the classification systems gives high accuracy with high handling time, though few strategies have yielded low precision even with enormous dataset. Along these lines, our work aims for high accuracy and less processing time with immense dataset.

Abstract

Diabetes is a chronic disease with the potential to cause a worldwide health care crisis . According to international Diabetes Federation 382 million people are living with diabetes Across the world . By 2035 this will doubled as 592 million. Diabetes mellitus or simply diabetes is a disease caused due to the increase level of blood glucose. Various traditional methods , based on physical and chemical tests, are available for diagnosing diabetes. Diabetes. However , early prediction of diabetes is quite challenging task for medical practitioners due to complex interdependence on various factors as diabetes affects human organs such kidney, eye ,heart on common questions . One such task is to help make predictions on medical data . Machine learning is an emerging scientific field in data science dealing with the ways in which machine learn from experience . The aim of the project is to develop a system

which can perform early prediction of diabetes for a patient with a higher accuracy by combining the results of different machine learning techniques . This project aims to predict diaetes via svm[support vector machine algo].

Project Summary

1.
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
```

Importing Basic Libraries

- Numpy-
  - NumPy (Numerical Python) is a **python package**, consisting of multi-dimensional array objects and a collection of routines for processing these array objects. In this article, we will cover frequently used NumPy operations used in ML Firstly, we need to import the NumPy library using the following code: import numpy as np

- Pandas-
  - Pandas ml is a **package** which integrates pandas, scikit-learn, xgboost into one package for easy handling of data and creation of machine learning models. By data scientists, for data scientists ANACONDA

- Sklearn-
  - Sklearn is unanimously the favorite **Python library** among data scientists. As a newcomer to machine learning, you should be comfortable with sklearn and how to build ML models, including: Linear Regression using sklearn Logistic Regression using sklearn, and so on.

- Standard Scaler-
  The standard scaling is calculated as: **z = (x - u) / s**. Where, z is scaled data. x is to be scaled data. u is the mean of the training samples; s is the standard deviation of the training samples. Sklearn preprocessing supports StandardScaler() method to achieve this directly in merely 2-3 steps.

- train_test_split is a function in Sklearn model selection for **splitting data arrays into two subsets: for training data and for testing data**. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.
- The support vector machine algorithm is a supervised machine learning algorithm that is often used for **classification problems**, though it can also be applied to regression problems.
- Accuracy Score = **(TP+TN)/ (TP+FN+TN+FP)** Here we can also calculate accuracy with the help of the accuracy_score method from sklearn. accuracy_score (y_true, y_pred, normalize=False) In multilabel classification, the function returns the subset accuracy.

❖
```
diabetes_dataset = pd.read_csv('diabetes.csv')
```

In this line basically document or dataset is loaded which is in csv type file.

❖
```
diabetes_dataset.describe()
```

Here In this line of code we basically can get the basic information regarding our dataset used

	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

❖

This is basically a overview regarding information of our Diabetes Data.

```
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
```

❖

Separating X and Y for further training
Here x is whole data without outcome column
And y is only outcome column of our Dataset and this separation is basically used further for Data training.

❖

```
model=StandardScaler()
```

```
model.fit(X)
```

```
StandardScaler()
```

```
standardized_d = model.transform(X)
```

```
X = standardized_d
```

```
Y = diabetes_dataset['Outcome']
```

Standard Scaler Used here

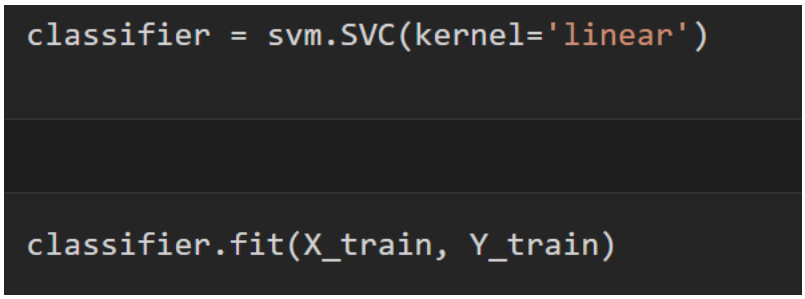
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)
```

```
X.shape, X_train.shape, X_test.shape
```

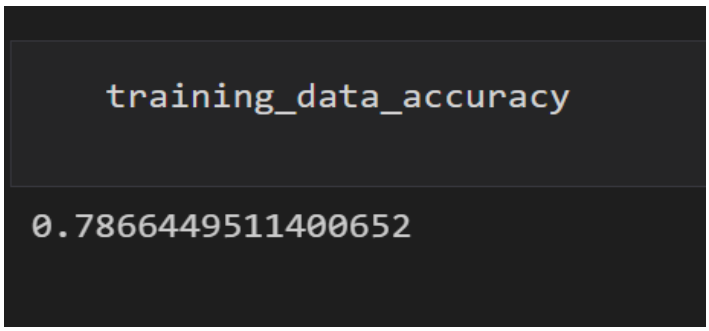
```
((768, 8), (614, 8), (154, 8))
```

❖

This is basically a part of sklearn library for both training and testing data and 20% of data is test and 80% of the data is for training purpose and these value assigned in different variables used in above diagram

- 
```
classifier = svm.SVC(kernel='linear')


classifier.fit(X_train, Y_train)
```
classifier used from sklearn for training and fitting of our data
SVM used with linear kernel.

- 
```
    training_data_accuracy

0.7866449511400652
```
Accuracy of our model is approximately of 79%.

```
    std_data = model.transform(id)
    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')
```

```
[[ 0.3429808   1.41167241  0.14964075 -0.09637905  0.82661621 -0.78595734
   0.34768723  1.51108316]]
[1]
The person is diabetic

C:\Users\91790\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base
names, but StandardScaler was fitted with feature names
```

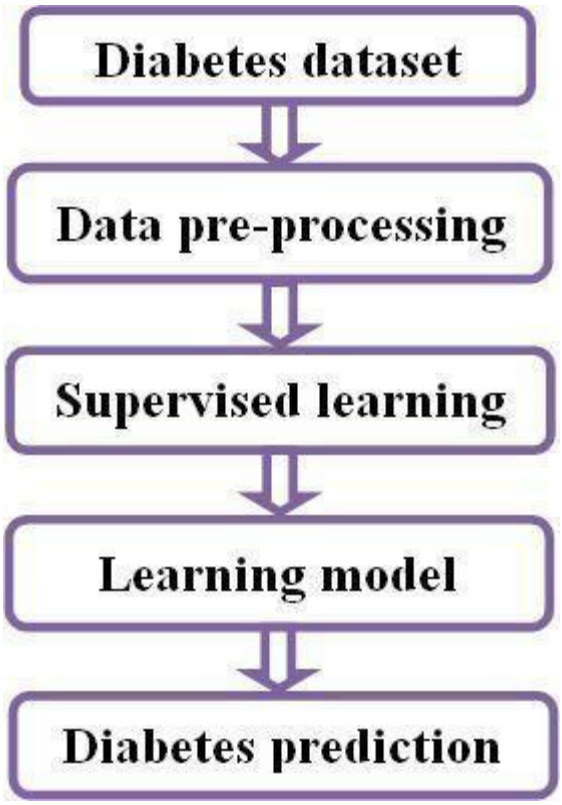
❖ Checking randomly a person details for finding whether is Diabetic or not.

Objective—

We propose a diabetes risk prediction model, DLPD, which can not only **predict whether someone will have this disease in the future but also determine the type of disease that a person may have in the future**

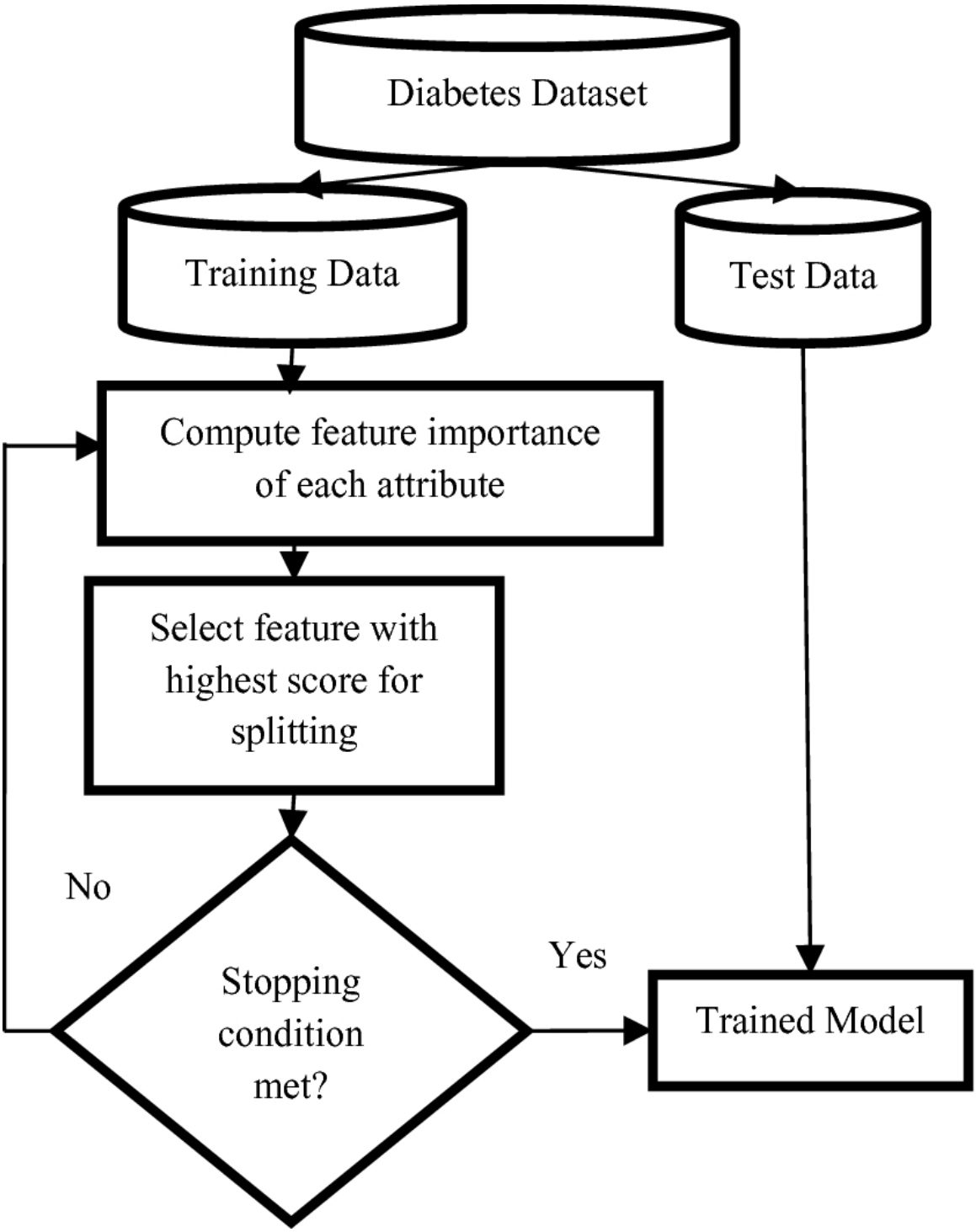
The objective is to **predict whether or not a patient has diabetes**, based on certain diagnostic measurements included in the dataset. 0 – Absence of Diabetes 1 – Presence of Diabetes

Data Flow Diagram of This Model [working]--

1. 

- Taking a dataset from website [Kaggle].
- Step of Data Preprocessing
- Supervised Learning algorithm used [Support Vector Machine] for classifying data and make predictions further
- Learning Model or Simply Trained Accordingly
- Making Prediction through normal Randomly chosen features of a person.

Data Flow Diagram of This Machine Learning Model



Input dataset sample screenshot—[The training dataset]—

	A	B	C	D	E	F	G	H	I
1	Pregnanci	Glucose	BloodPres	SkinThickr	Insulin	BMI	DiabetesP	Age	Outcome
2	6	148	72	35	0	33.6	0.627	50	1
3	1	85	66	29	0	26.6	0.351	31	0
4	8	183	64	0	0	23.3	0.672	32	1
5	1	89	66	23	94	28.1	0.167	21	0
6	0	137	40	35	168	43.1	2.288	33	1
7	5	116	74	0	0	25.6	0.201	30	0
8	3	78	50	32	88	31	0.248	26	1
9	10	115	0	0	0	35.3	0.134	29	0
10	2	197	70	45	543	30.5	0.158	53	1
11	8	125	96	0	0	0	0.232	54	1
12	4	110	92	0	0	37.6	0.191	30	0
13	10	168	74	0	0	38	0.537	34	1
14	10	139	80	0	0	27.1	1.441	57	0
15	1	189	60	23	846	30.1	0.398	59	1
16	5	166	72	19	175	25.8	0.587	51	1
17	7	100	0	0	0	30	0.484	32	1
18	0	118	84	47	230	45.8	0.551	31	1
19	7	107	74	0	0	29.6	0.254	31	1
20	1	103	30	38	83	43.3	0.183	33	0
21	1	115	70	30	96	34.6	0.529	32	1
22	3	126	88	41	235	39.3	0.704	27	0
23	8	99	84	0	0	35.4	0.388	50	0
24	7	196	90	0	0	39.8	0.451	41	1
25	9	119	80	35	0	29	0.263	29	1
26	11	143	94	33	146	36.6	0.254	51	1
27	10	125	70	26	115	31.1	0.205	41	1
28	7	147	76	0	0	39.4	0.257	43	1
29	1	97	66	15	140	23.2	0.487	22	0
30	13	145	82	19	110	22.2	0.245	57	0
31	5	117	92	0	0	34.1	0.337	38	0
32	5	109	75	26	0	36	0.546	60	0

Output /Prediction--->

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


[[ 0.3429808   1.41167241  0.14964075 -0.09637905  0.82661621 -0.78595734
   0.34768723  1.51108316]]
[1]
The person is diabetic
```

Showing the Women's details entered is a diabetic Women.

Code-

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

```
diabetes_dataset = pd.read_csv('diabetes.csv')
```

```
diabetes_dataset.head()
```

	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

```
diabetes_dataset.describe()
```

```
diabetes_dataset.describe()
```

	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

```
diabetes_dataset['Outcome'].value_counts()
```

```
0    500
1    268
Name: Outcome, dtype: int64
```

```
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
```

```
X
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148	72	35	0	33.6	0.627	50
1	1	85	66	29	0	26.6	0.351	31
2	8	183	64	0	0	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33
...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63
764	2	122	70	27	0	36.8	0.340	27
765	5	121	72	23	112	26.2	0.245	30
766	1	126	60	0	0	30.1	0.349	47
767	1	93	70	31	0	30.4	0.315	23

768 rows × 8 columns

```
model=StandardScaler()
```

```
model.fit(X)
```

```
standardized_d = model.transform(X)
```

+ Code + Markdown

```
X = standardized_d
```

```
Y = diabetes_dataset['Outcome']
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)
```

```
X.shape, X_train.shape, X_test.shape
```

```
((768, 8), (614, 8), (154, 8))
```

```
classifier = svm.SVC(kernel='linear')
```

```
classifier.fit(X_train, Y_train)
```

```
    X_train_prediction = classifier.predict(X_train)
    training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

```
    training_data_accuracy
```

```
0.7866449511400652
```

```
    X_test_prediction = classifier.predict(X_test)
    test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
    a=np.array([5,166,72,19,175,25.8,0.587,51])
```

```
    id=a.reshape(1,-1)
```

```
    std_data = model.transform(id)
    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')
```

```
[[ 0.3429808   1.41167241  0.14964075 -0.09637905  0.82661621 -0.78595734
   0.34768723  1.51108316]]
[1]
The person is diabetic
```

References-

Sidhhardhan [youtube]

Link--https://www.youtube.com/watch?v=xUE7SjVx9bQ&t=570s

Manas jain [Other Supports]