In [1]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
```

```
C:\Users\sanja\anaconda3\lib\site-packages\pandas\core\computation\expres
sions.py:21: UserWarning: Pandas requires version '2.8.0' or newer of 'nu
mexpr' (version '2.7.3' currently installed).
  from pandas.core.computation.check import NUMEXPR_INSTALLED
C:\Users\sanja\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWar
ning: A NumPy version >=1.16.5 and <1.23.0 is required for this version o
f SciPy (detected version 1.24.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
# loading the diabetes dataset to a pandas DataFrame
diabetes_dataset = pd.read_csv('diabetes.csv')
```

In [3]:
```
# printing the first 5 rows of the dataset
diabetes_dataset.head()
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunc
0	6	148	72	35	0	33.6	0
1	1	85	66	29	0	26.6	0
2	8	183	64	0	0	23.3	0
3	1	89	66	23	94	28.1	0
4	0	137	40	35	168	43.1	2

In [4]:
```
# number of rows and Columns in this dataset
diabetes_dataset.shape
```

Out[4]: (768, 9)

In [5]:
```
# getting the statistical measures of the data
diabetes_dataset.describe()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Dial
count	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	
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	

In [6]:
```
diabetes_dataset['Outcome'].value_counts()
```

Out[6]:
```
Outcome
0    500
1    268
Name: count, dtype: int64
```

0 --> Non-Diabetic

1 --> Diabetic

In [7]:
```
diabetes_dataset.isnull().sum()
```

Out[7]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [8]:
```
diabetes_dataset.groupby('Outcome').mean()
```

Out[8]:

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	D
0	3.298000	109.980000	68.184000	19.664000	68.792000	30.304200	
1	4.865672	141.257463	70.824627	22.164179	100.335821	35.142537	

In [9]:
```
# separating the data and labels
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
```

In [10]:
```
print(X)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6   
1              1       85             66             29        0  26.6   
2              8      183             64              0        0  23.3   
3              1       89             66             23       94  28.1   
4              0      137             40             35      168  43.1   
..           ...      ...            ...            ...      ...   ...   
763           10      101             76             48      180  32.9   
764            2      122             70             27        0  36.8   
765            5      121             72             23      112  26.2   
766            1      126             60              0        0  30.1   
767            1       93             70             31        0  30.4   

     DiabetesPedigreeFunction  Age  
0                       0.627   50  
1                       0.351   31  
2                       0.672   32  
3                       0.167   21  
4                       2.288   33  
..                        ...  ...  
763                     0.171   63  
764                     0.340   27  
765                     0.245   30  
766                     0.349   47  
767                     0.315   23  

[768 rows x 8 columns]
```

In [11]:
```
print(Y)
```

```
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
```

In [12]:
```
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
```

In [13]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, s
```

In [14]:
```
print(X.shape, X_train.shape, X_test.shape)
```

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

In [15]:
```
classifier = svm.SVC(kernel='linear')
```

In [16]:
```
#training the support vector Machine Classifier
classifier.fit(X_train, Y_train)
```

```
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:75
7: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if not hasattr(array, "sparse") and array.dtypes.apply(is_sparse).any
():
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:59
5: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype):
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:60
4: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype) or not is_extension_array_dtype(pd_dtype):
```

Out[16]:
```
SVC(kernel='linear')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [17]:
```
# accuracy score on the training data
X_train_prediction = classifier.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

```
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:75
7: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if not hasattr(array, "sparse") and array.dtypes.apply(is_sparse).any
():
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:59
5: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype):
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:60
4: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype) or not is_extension_array_dtype(pd_dtype):
```

In [18]:
```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  0.7833876221498371
```

In [19]:
```
X_test_prediction = classifier.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:75
7: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if not hasattr(array, "sparse") and array.dtypes.apply(is_sparse).any
():
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:59
5: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype):
C:\Users\sanja\anaconda3\lib\site-packages\sklearn\utils\validation.py:60
4: FutureWarning: is_sparse is deprecated and will be removed in a future
version. Check `isinstance(dtype, pd.SparseDtype)` instead.
  if is_sparse(pd_dtype) or not is_extension_array_dtype(pd_dtype):
```

In [2]:
```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  93.7
```

In [21]:
```
input_data = (5,166,72,19,175,25.8,0.587,51)

# 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)

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

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

```
[1]
The person is diabetic

C:\Users\sanja\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarni
ng: X does not have valid feature names, but SVC was fitted with feature
names
  warnings.warn(
```

In [24]:
```
import pickle
filename = 'diabetes_model.pkl'
pickle.dump(classifier, open(filename, 'wb'))
```

In [ ]:
```
```

In [25]:
```
# loading the saved model
loaded_model = pickle.load(open('diabetes_model.pkl', 'rb'))
```

In [26]:
```
input_data = (5,166,72,19,175,25.8,0.587,51)

# 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)

prediction = loaded_model.predict(input_data_reshaped)
print(prediction)

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

```
[1]
The person is diabetic

C:\Users\sanja\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarni
ng: X does not have valid feature names, but SVC was fitted with feature
names
  warnings.warn(
```

In [27]:
```
for column in X.columns:
    print(column)
```

```
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
```

In [ ]:

In [ ]:

In [ ]: