Importing the Dependencies

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings('ignore')
```

Importing the Dataset

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

diabetes_dataset.head()
```


```
{"summary":"{\n  \"name\": \"diabetes_dataset\",\n  \"rows\": 768,\n
\"fields\": [\n    {\n      \"column\": \"Pregnancies\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3,\n        \"min\": 0,\n        \"max\": 17,\n
\"num_unique_values\": 17,\n        \"samples\": [\n          6,\n
1,\n          3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Glucose\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 31,\n        \"min\": 0,\n        \"max\": 199,\n
\"num_unique_values\": 136,\n        \"samples\": [\n          151,\n
101,\n          112\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BloodPressure\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 19,\n        \"min\": 0,\n
\"max\": 122,\n        \"num_unique_values\": 47,\n
\"samples\": [\n          86,\n          46,\n          85\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"SkinThickness\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 15,\n        \"min\": 0,\n
\"max\": 99,\n        \"num_unique_values\": 51,\n        \"samples\":
[\n          7,\n          12,\n          48\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Insulin\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 115,\n
\"min\": 0,\n        \"max\": 846,\n        \"num_unique_values\":
186,\n        \"samples\": [\n          52,\n          41,\n
183\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BMI\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 7.8841603203754405,\n        \"min\": 0.0,\n        \"max\":
```

```
67.1,\n        \"num_unique_values\": 248,\n        \"samples\": [\n
19.9,\n          31.0,\n          38.1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"DiabetesPedigreeFunction\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.33132859501277484,\n        \"min\": 0.078,\n        \"max\": 2.42,\
n        \"num_unique_values\": 517,\n        \"samples\": [\n
1.731,\n          0.426,\n          0.138\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 11,\n        \"min\": 21,\n
\"max\": 81,\n        \"num_unique_values\": 52,\n        \"samples\":
[\n          60,\n          47,\n          72\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Outcome\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 0,\n        \"max\": 1,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          0,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"diabetes_dataset"}
```


```
# Checking the shape of the dataset
diabetes_dataset.shape
```

```
(768, 9)
```

Data Preprocessing

```
# Checking if any Missing value is present or not
diabetes_dataset.isnull().sum()
```

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

```
diabetes_dataset.describe()
```


```
{"summary":"{\n  \"name\": \"diabetes_dataset\",\n  \"rows\": 8,\n
\"fields\": [\n    {\n      \"column\": \"Pregnancies\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
269.85223453356366,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
3.8450520833333335,\n          3.0,\n          768.0\n        ],\n
```

```
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Glucose\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
243.73802348295857,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
120.89453125,\n          117.0,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"BloodPressure\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
252.85250535810619,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
69.10546875,\n          72.0,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"SkinThickness\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
263.7684730531098,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          768.0,\n
20.536458333333332,\n          32.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Insulin\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
350.26059167945886,\n        \"min\": 0.0,\n        \"max\": 846.0,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          768.0,\n
79.79947916666667,\n          127.25\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"BMI\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 262.05117817552093,\n
\"min\": 0.0,\n        \"max\": 768.0,\n        \"num_unique_values\":
8,\n        \"samples\": [\n          31.992578124999998,\n
32.0,\n          768.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"DiabetesPedigreeFunction\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 271.3005221658502,\n
\"min\": 0.078,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
0.47187630208333325,\n          0.3725,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 260.1941178528413,\n
\"min\": 11.76023154067868,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
33.240885416666664,\n          29.0,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Outcome\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
271.3865920388932,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 5,\n        \"samples\": [\n
0.3489583333333333,\n          1.0,\n          0.4769513772427971\n
```

```
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}
```


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

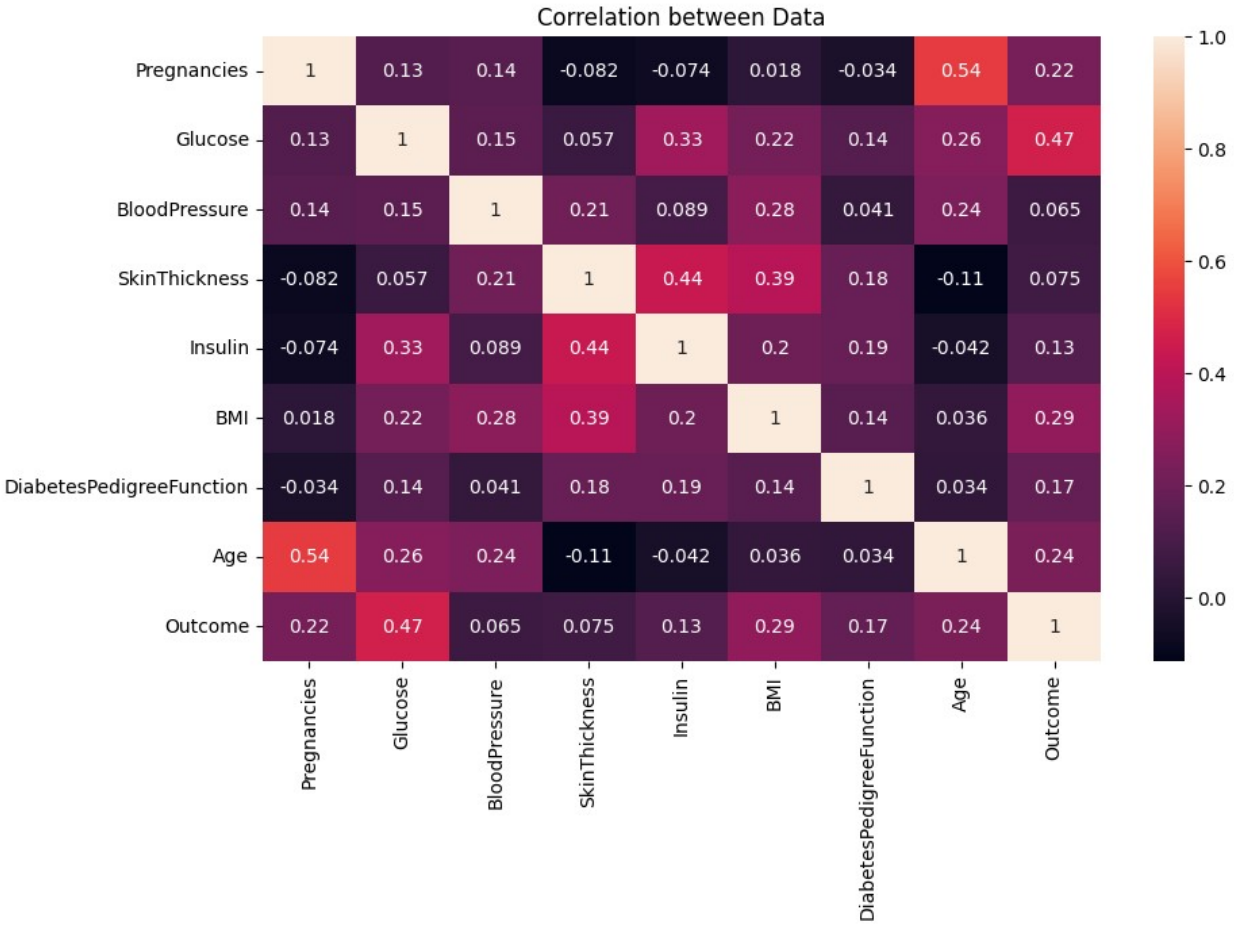
```
Outcome
0    500
1    268
Name: count, dtype: int64
```

```
diabetes_dataset.groupby('Outcome').mean()
```

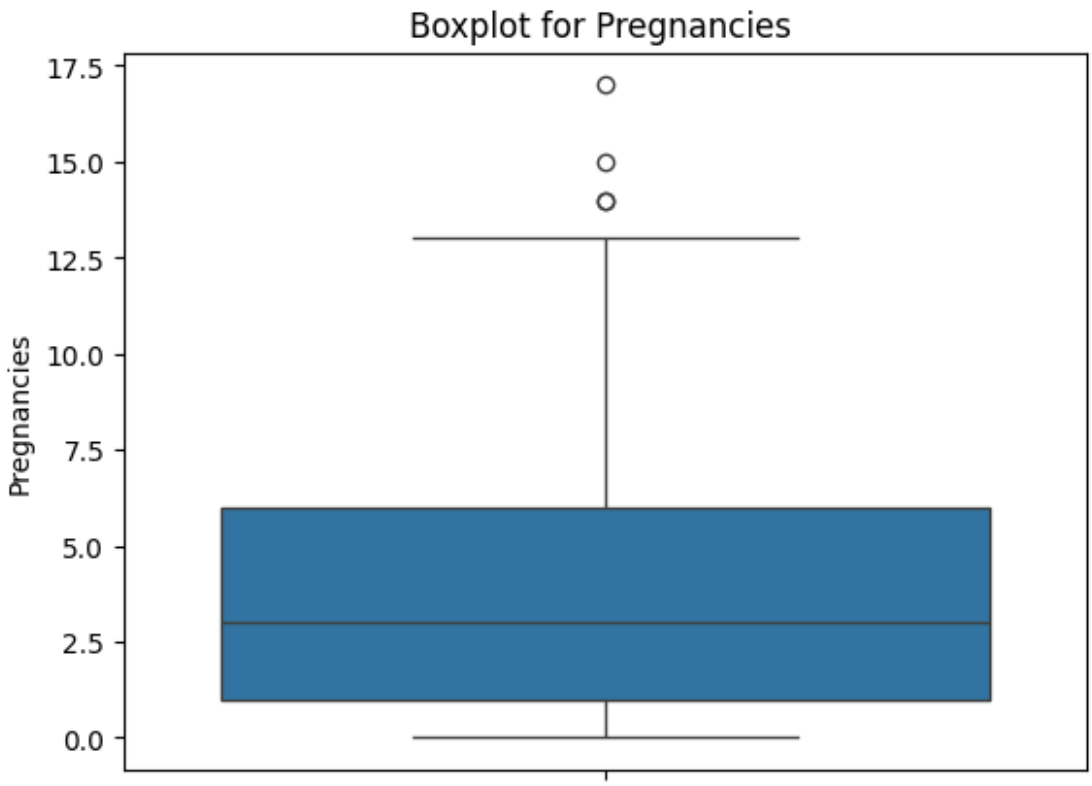

```
{"summary":"{\n  \"name\": \"diabetes_dataset\",\n  \"rows\": 2,\n
\"fields\": [\n    {\n      \"column\": \"Outcome\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Pregnancies\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.108511248584296,\n        \"min\":
3.298,\n        \"max\": 4.865671641791045,\n
\"num_unique_values\": 2,\n        \"samples\": [\n
4.865671641791045,\n          3.298\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Glucose\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
22.116505963980842,\n        \"min\": 109.98,\n        \"max\":
141.25746268656715,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          141.25746268656715,\n          109.98\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"BloodPressure\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.8672051632998017,\n        \"min\": 68.184,\n        \"max\":
70.82462686567165,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          70.82462686567165,\n          68.184\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"SkinThickness\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.7678935989570275,\n        \"min\": 19.664,\n        \"max\":
22.16417910447761,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          22.16417910447761,\n          19.664\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Insulin\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
22.304849659757796,\n        \"min\": 68.792,\n        \"max\":
100.33582089552239,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          100.33582089552239,\n          68.792\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"BMI\",\n      \"properties\": {\
```

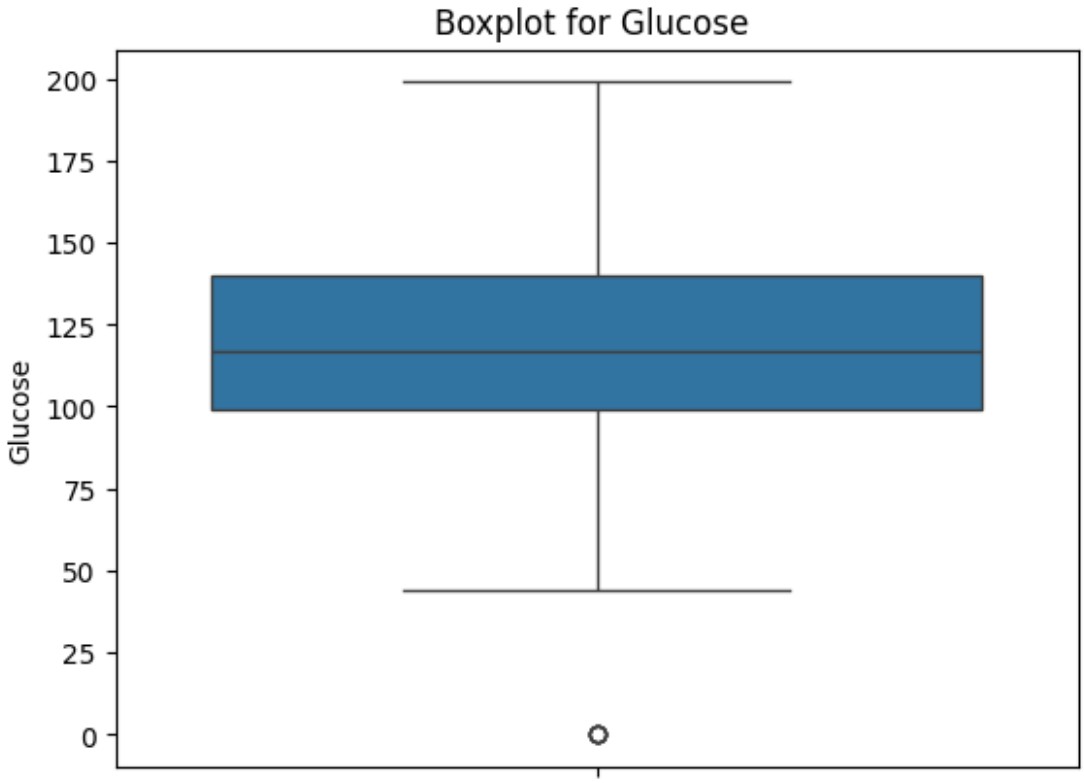
```
n        \"dtype\": \"number\",\n        \"std\": 3.4212211239962618,\
n        \"min\": 30.3042,\n        \"max\": 35.14253731343284,\n
\"num_unique_values\": 2,\n        \"samples\": [\n
35.14253731343284,\n          30.3042\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"DiabetesPedigreeFunction\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.08539445753677459,\n        \"min\": 0.429734,\n        \"max\":
0.5505,\n        \"num_unique_values\": 2,\n        \"samples\": [\n
0.5505,\n          0.429734\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Age\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 4.155782645191446,\n        \"min\":
31.19,\n        \"max\": 37.06716417910448,\n
\"num_unique_values\": 2,\n        \"samples\": [\n
37.06716417910448,\n          31.19\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

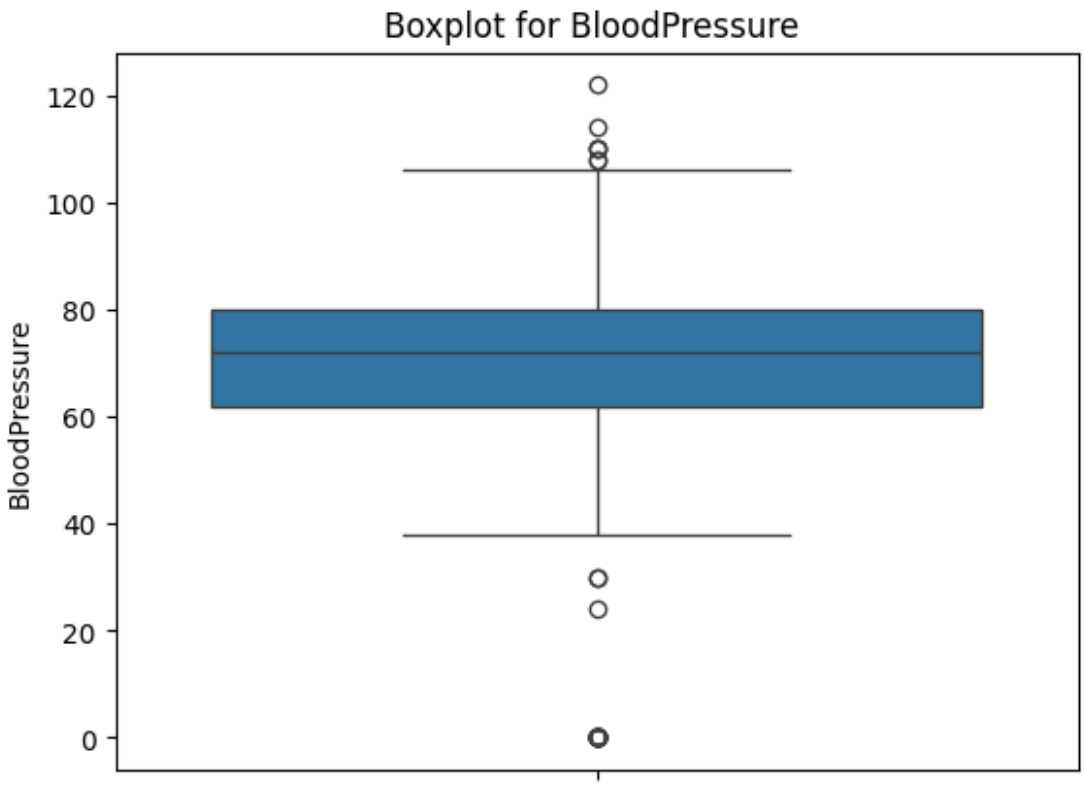

```
# Checking the Correlation

plt.figure(figsize=(10,6))
sns.heatmap(diabetes_dataset.corr(),annot=True)
plt.title("Correlation between Data")
plt.show()
```

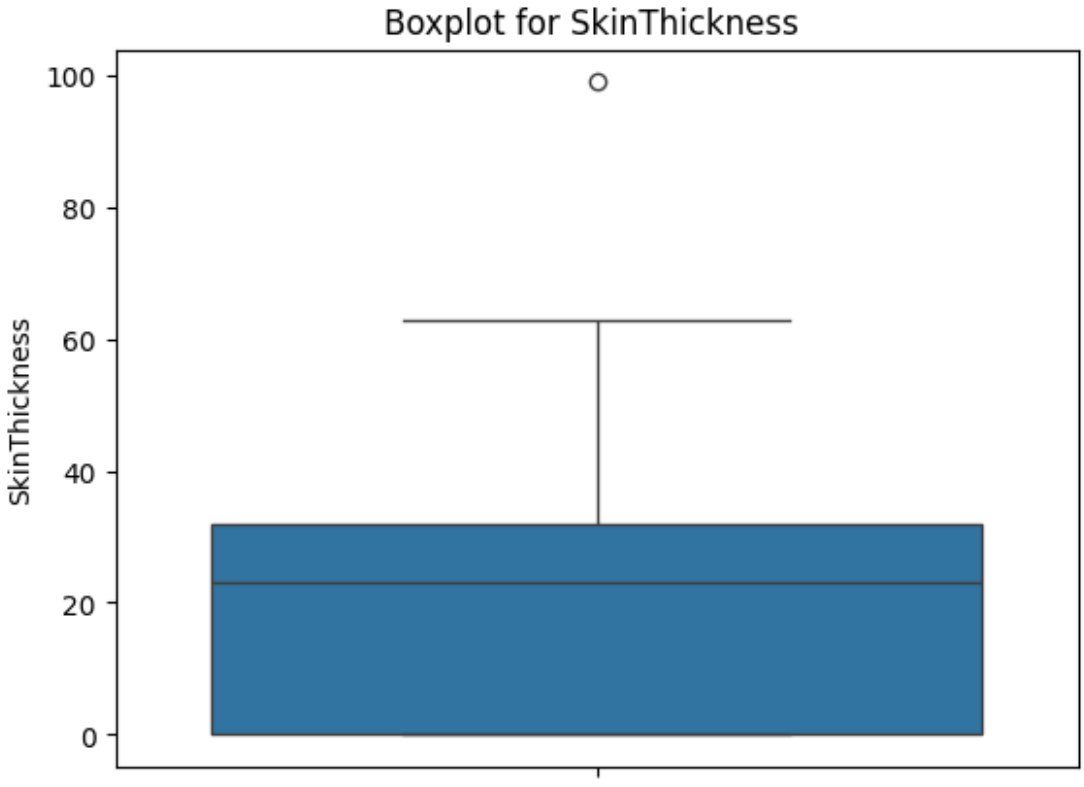


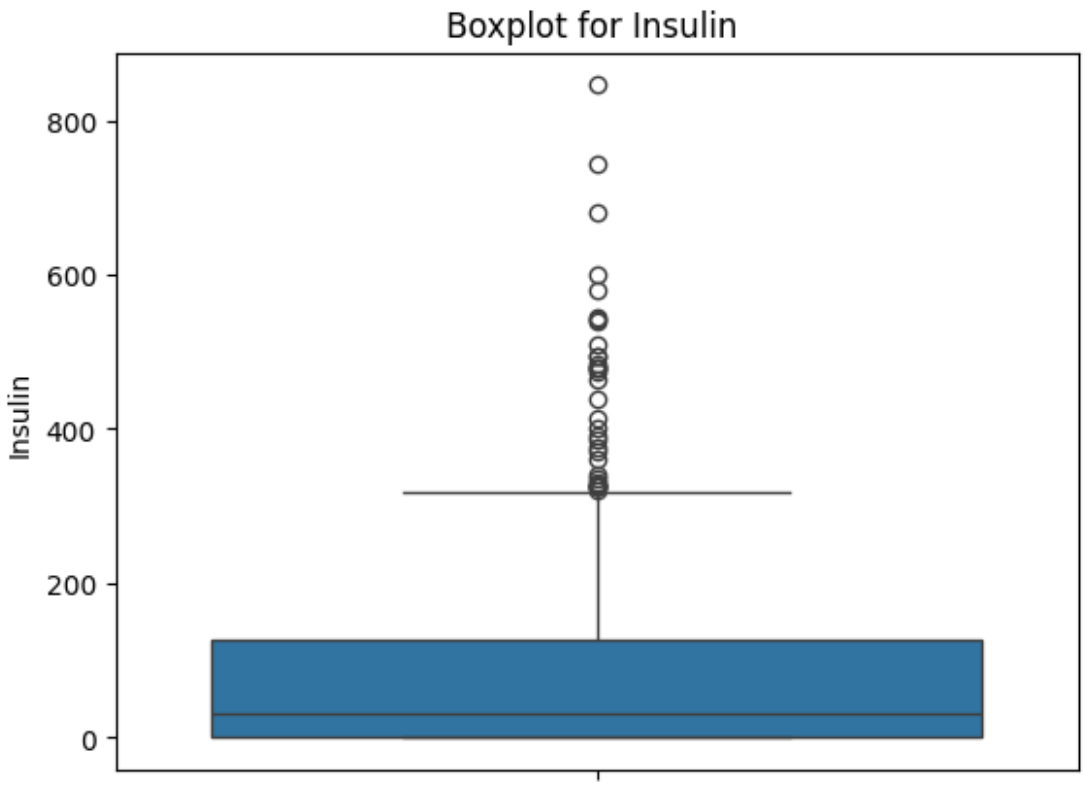

```
list1=diabetes_dataset.columns

for i in list1:
  sns.boxplot(diabetes_dataset[i])
  plt.title(f"Boxplot for {i} ")
  plt.show()
```

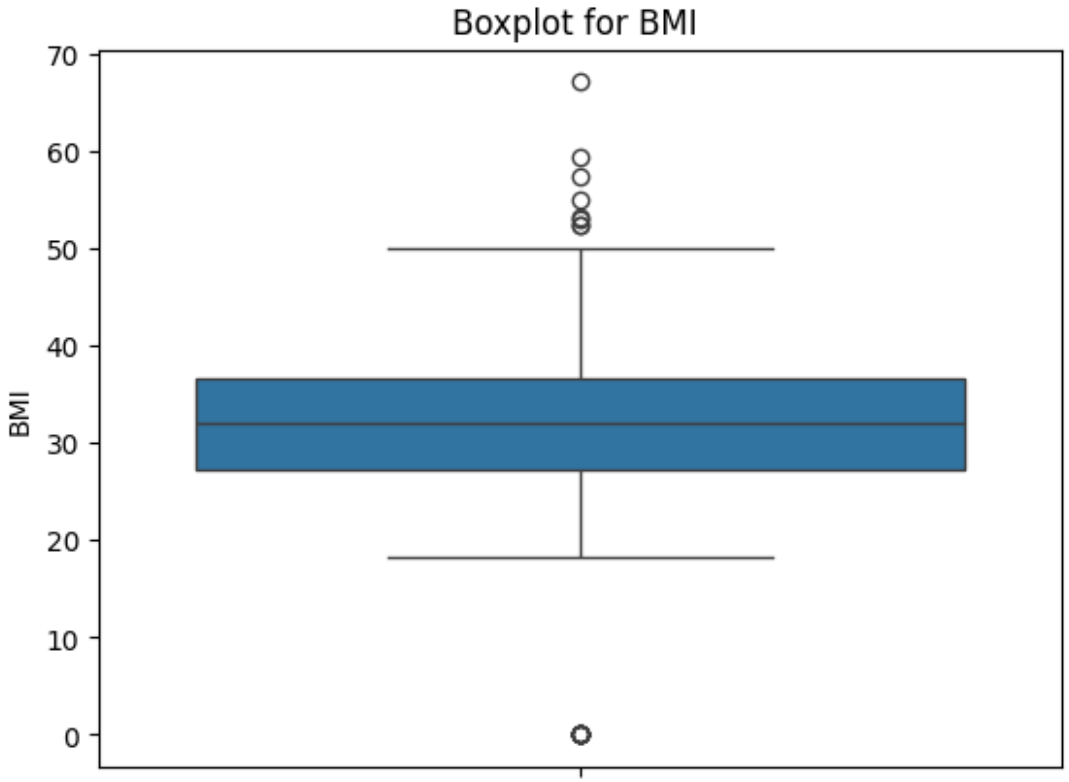

Boxplot for Pregnancies
Pregnancies
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0

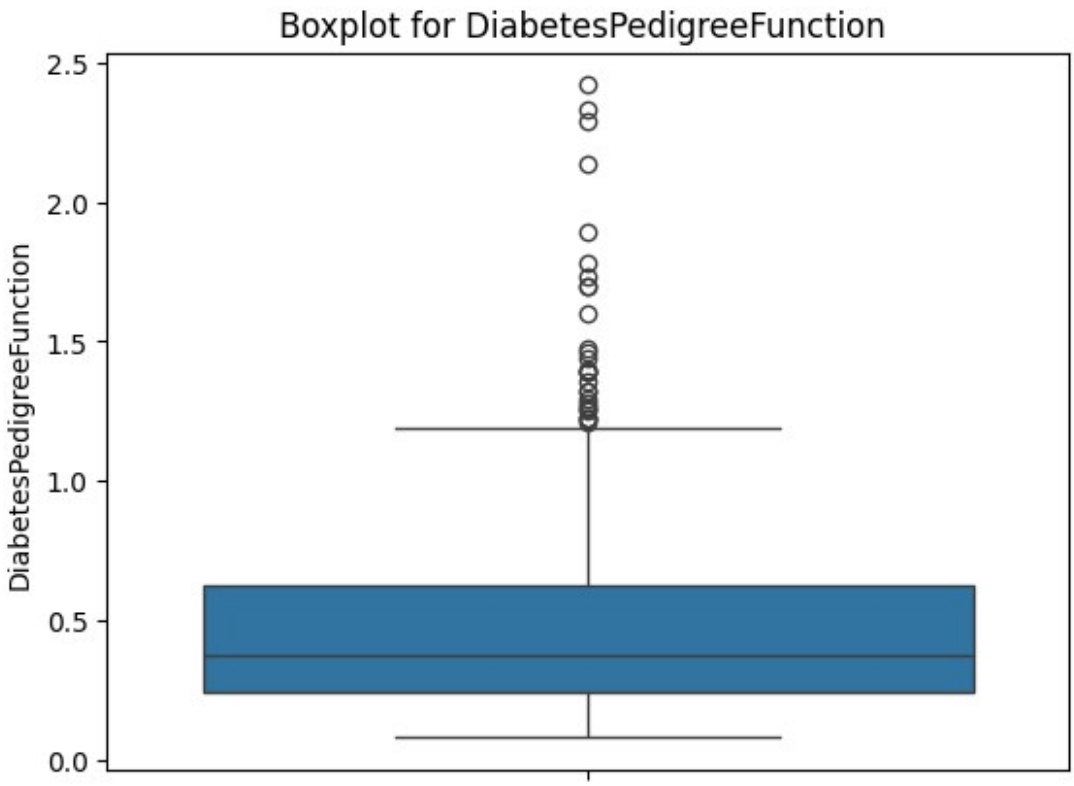

Boxplot for Glucose
Glucose
200
175
150
125
100
75
50
25
0

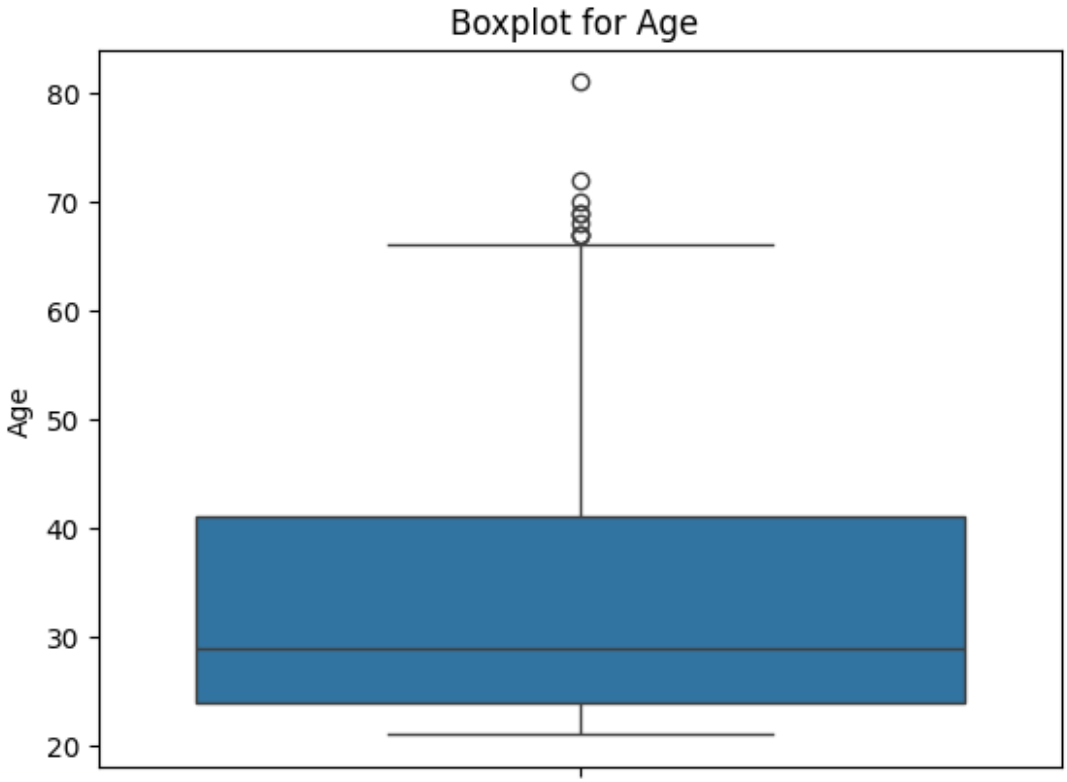

Boxplot for BloodPressure
BloodPressure
120
100
80
60
40
20
0

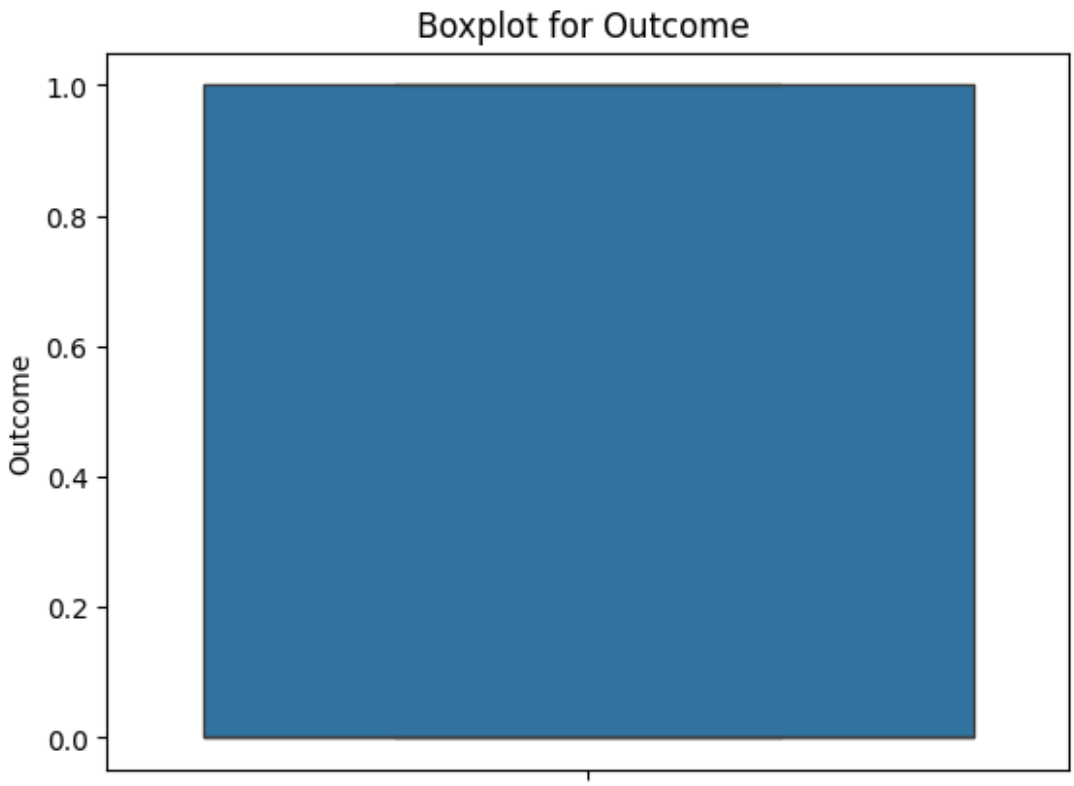

Boxplot for SkinThickness
SkinThickness
100
80
60
40
20
0


Boxplot for Insulin
Insulin
800
600
400
200
0


Boxplot for BMI
BMI
70
60
50
40
30
20
10
0


Boxplot for DiabetesPedigreeFunction
DiabetesPedigreeFunction
2.5
2.0
1.5
1.0
0.5
0.0


Boxplot for Age
Age
80
70
60
50
40
30
20




```
df_no_outliers = diabetes_dataset.copy()

for column in list1:
    Q1 = df_no_outliers[column].quantile(0.25)
    Q3 = df_no_outliers[column].quantile(0.75)
    IQR = Q3 - Q1

    # Define the lower and upper bounds
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Filter out the outliers for the current column
    df_no_outliers = df_no_outliers[(df_no_outliers[column] >=
lower_bound) & (df_no_outliers[column] <= upper_bound)]

print("Shape of the DataFrame before outlier removal:",
diabetes_dataset.shape)
print("Shape of the DataFrame after outlier removal:",
df_no_outliers.shape)
```

```
Shape of the DataFrame before outlier removal: (768, 9)
Shape of the DataFrame after outlier removal: (636, 9)
```

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

```
print(X)
print(Y)
```

```
     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

5              5      116             74              0        0  25.6

..           ...      ...            ...            ...      ...   ...

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
5                       0.201   30
..                        ...  ...
763                     0.171   63
764                     0.340   27
765                     0.245   30
766                     0.349   47
767                     0.315   23

[636 rows x 8 columns]
0      1
1      0
2      1
3      0
5      0
      ..
763    0
764    0
765    0
```

```
766    1
767    0
Name: Outcome, Length: 636, dtype: int64
```

Scaling the Data

```
# Scale the data(like 0 to 1 range)
scalar=StandardScaler()

data_scaled=scalar.fit_transform(X)
print(data_scaled)
```

```
[[ 6.81425557e-01  1.00540261e+00 -6.96199653e-04 ...  2.56438414e-01
   8.19790711e-01  1.60468837e+00]
 [-8.57589551e-01 -1.16896300e+00 -5.32035774e-01 ... -8.40893209e-01
  -3.07223286e-01 -1.43040721e-01]
 [ 1.29703160e+00  2.21338350e+00 -7.09148966e-01 ... -1.35820669e+00
   1.00354299e+00 -5.10549792e-02]
 ...
 [ 3.73622536e-01  7.35316313e-02 -6.96199653e-04 ... -9.03597873e-01
  -7.40061995e-01 -2.35026462e-01]
 [-8.57589551e-01  2.46100331e-01 -1.06337535e+00 ... -2.92227397e-01
  -3.15390054e-01  1.32873114e+00]
 [-8.57589551e-01 -8.92853085e-01 -1.77809391e-01 ... -2.45198899e-01
  -4.54225112e-01 -8.78926652e-01]]
```

```
X=data_scaled
Y=df_no_outliers['Outcome']

print(X)
print(Y)
```

```
[[ 6.81425557e-01  1.00540261e+00 -6.96199653e-04 ...  2.56438414e-01
   8.19790711e-01  1.60468837e+00]
 [-8.57589551e-01 -1.16896300e+00 -5.32035774e-01 ... -8.40893209e-01
  -3.07223286e-01 -1.43040721e-01]
 [ 1.29703160e+00  2.21338350e+00 -7.09148966e-01 ... -1.35820669e+00
   1.00354299e+00 -5.10549792e-02]
 ...
 [ 3.73622536e-01  7.35316313e-02 -6.96199653e-04 ... -9.03597873e-01
  -7.40061995e-01 -2.35026462e-01]
 [-8.57589551e-01  2.46100331e-01 -1.06337535e+00 ... -2.92227397e-01
  -3.15390054e-01  1.32873114e+00]
 [-8.57589551e-01 -8.92853085e-01 -1.77809391e-01 ... -2.45198899e-01
  -4.54225112e-01 -8.78926652e-01]]
0      1
1      0
2      1
3      0
5      0
```

```
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 636, dtype: int64
```

Spliting the data into Train and Test

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

print(X_train.shape,X_test.shape,Y_train.shape,Y_test.shape)
```

```
(508, 8) (128, 8) (508,) (128,)
```

Training the model

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

svm_model.fit(X_train,Y_train)

model_pred=svm_model.predict(X_train)

# Checking the prediction score on seen data
model_accuracy=accuracy_score(model_pred,Y_train)

print("Accuracy of the model on Training Dataset :",model_accuracy)
```

```
Accuracy of the model on Training Dataset : 0.7893700787401575
```

Testing the model

```
# Checking the prediction score on unseen data
model_pred_2=svm_model.predict(X_test)

model_accuracy_2=accuracy_score(model_pred_2,Y_test)

print("Accuracy of the model on Testing Dataset :",model_accuracy_2)
```

```
Accuracy of the model on Testing Dataset : 0.8046875
```

Making the prediction fuction

```
def Prediction(data):
  # convert to array
  data_array=np.asarray(data)

  # reshaping to prediction input
  data_reshaped=data_array.reshape(1,-1)
```

```
  # scaling the input
  data_scal=scalar.transform(data_reshaped)

  #predicting the Outcome
  prediction=svm_model.predict(data_scal)

  if prediction[0]==0:
    print("the person is not diabetic")
  else:
    print("the person is diabetic")

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

```
the person is not diabetic
```

```
input_data2=(5,166,72,19,175,25.8,0.587,51)
Prediction(input_data2)
```

```
the person is diabetic
```

```
input_data3=(5,139,64,35,140,28.6,0.411,26)
Prediction(input_data3)
```

```
the person is not diabetic
```