```
print("Accuracy (Test) score of KNN ", knn.score(X_test,y_test)*100)
print("Accuracy score of KNN ", accuracy_score(y_test,knn_pred)*100)

# Train & Test Scores of Naive-Bayes
print("Accuracy (Train) score of Naive Bayes ",nb.score(X_train,y_train)*100)
print("Accuracy (Test) score of Naive Bayes ", nb.score(X_test,y_test)*100)
print("Accuracy score of Naive Bayes ", accuracy_score(y_test,nb_pred)*100)

# Train & Test Scores of SVM
print("Accuracy (Train) score of SVM ",sv.score(X_train,y_train)*100)
print("Accuracy (Test) score of SVM ", sv.score(X_test,y_test)*100)
print("Accuracy score of SVM ", accuracy_score(y_test,sv_pred)*100)

# Train & Test Scores of Decision Tree
print("Accuracy (Train) score of Decision Tree
",dt.score(X_train,y_train)*100)
print("Accuracy (Test) score of Decision Tree ", dt.score(X_test,y_test)*100)
print("Accuracy score of Decision Tree ", accuracy_score(y_test,dt_pred)*100)

# Train & Test Scores of Random Forest
print("Accuracy (Train) score of Random Forest
",rf.score(X_train,y_train)*100)
print("Accuracy (Test) score of Random Forest ", rf.score(X_test,y_test)*100)
print("Accuracy score of Random Forest ", accuracy_score(y_test,rf_pred)*100)
```

Step 10: Making a prediction System

```
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)

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

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

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