Diabetes Prediction

Using Machine Learning

By
C K Anand Prakash

Introduction

- 10.5% of the US population (34.2 million) has diabetes
- 3 types:
  - Type I
    - Occurs mostly in children & teens
    - Body produces little to no insulin
  - Type II
    - 90% of all diabetes cases, mostly adults
    - Body doesn't use the insulin efficiently or properly
  - Gestational
    - High blood glucose during pregnancies
- Being able to predict diabetes based on certain factors like BMI, age, etc. can help with improved treatment down the line

Dataset

	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

Predictor variables:

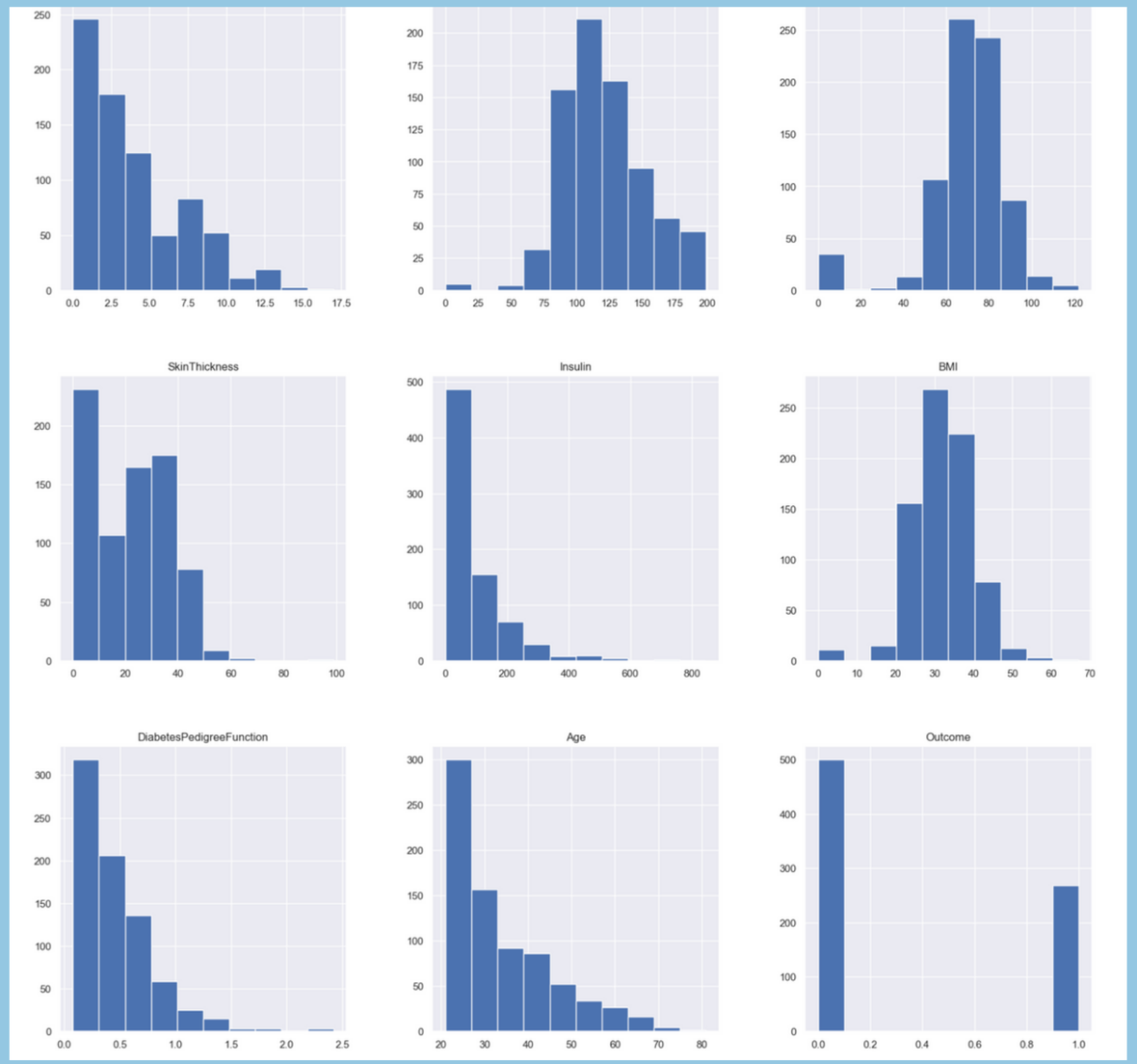
1. Pregnancies
2. Glucose
3. BloodPressure
4. Skin Thickness
5. Insulin
6. BMI
7. DiabetesPedigreeFunction
8. Age

Outcome variable: Outcome (0 or 1)

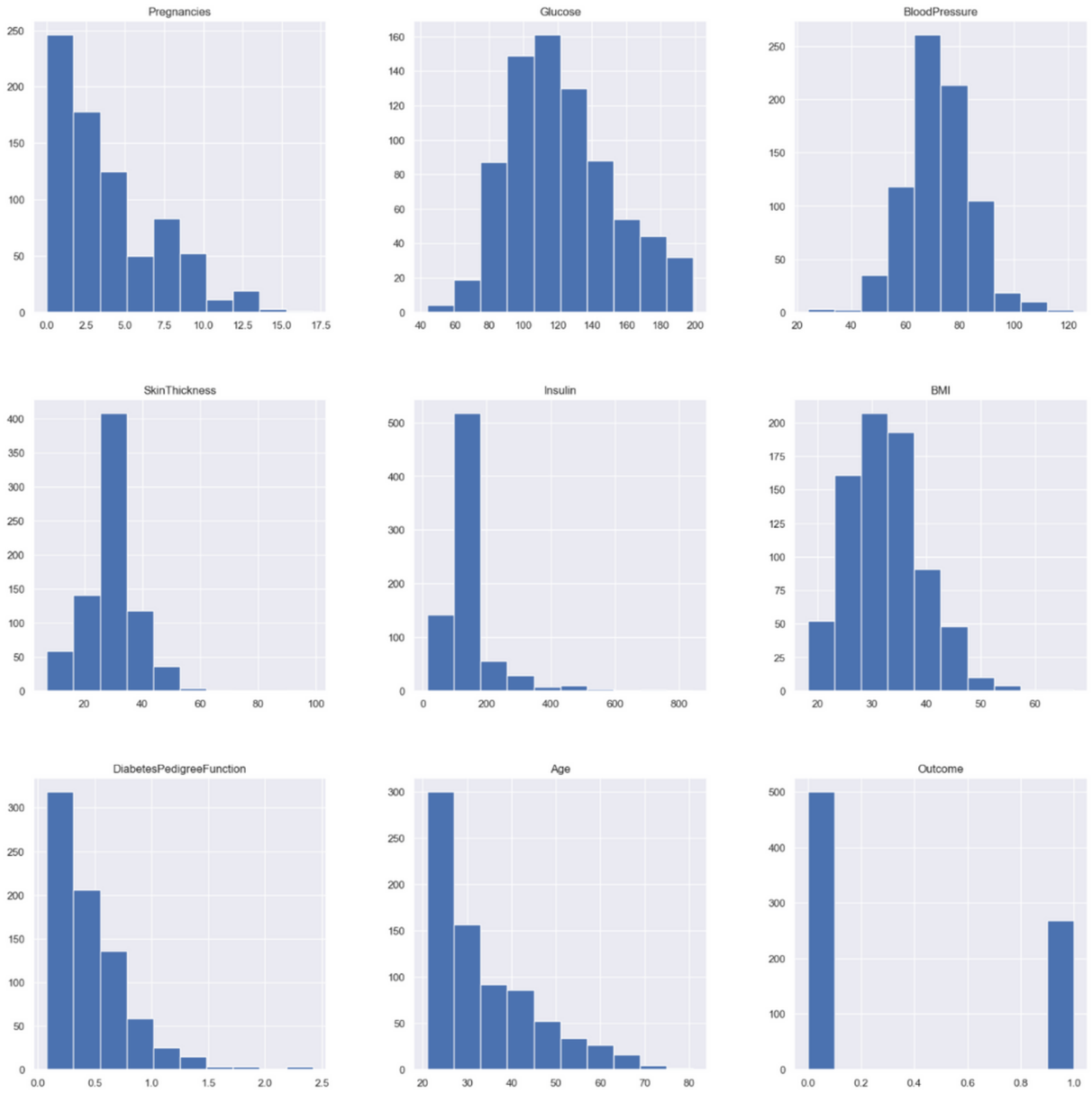
Shape: (768, 9)

Benchmark #1

Training the model with raw data



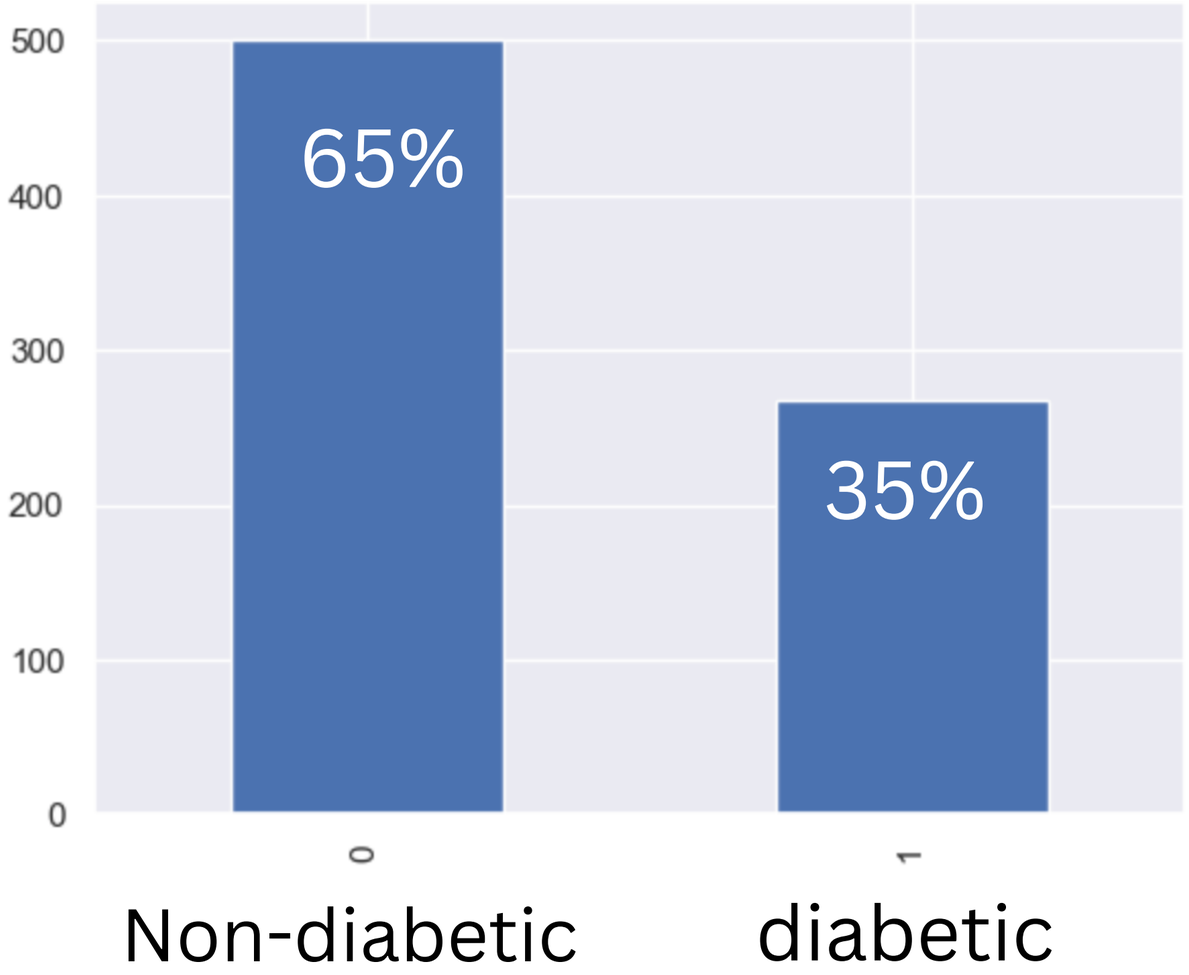
- Used Support Vector Machine (SVM) Classifier to predict accuracy score
- Accuracy score of the test data: 75.3%



Replacing Null Values

- Missing values in columns: Glucose, BloodPressure, SkinThickness, Insulin, & BMI
- Replaced null values for columns using Iterative Imputer

```
lucose','BloodPressure','SkinThickness','Insulin','BMI']
ativeImputer()
pute_it.fit_transform(diabetes_data_copy[null_columns])
py[null_columns] = impute_data
```



Class Variable distribution

- Number of non-diabetic patients (0 -> 500) is almost twice the amount of diabetic patients (1 -> 268)

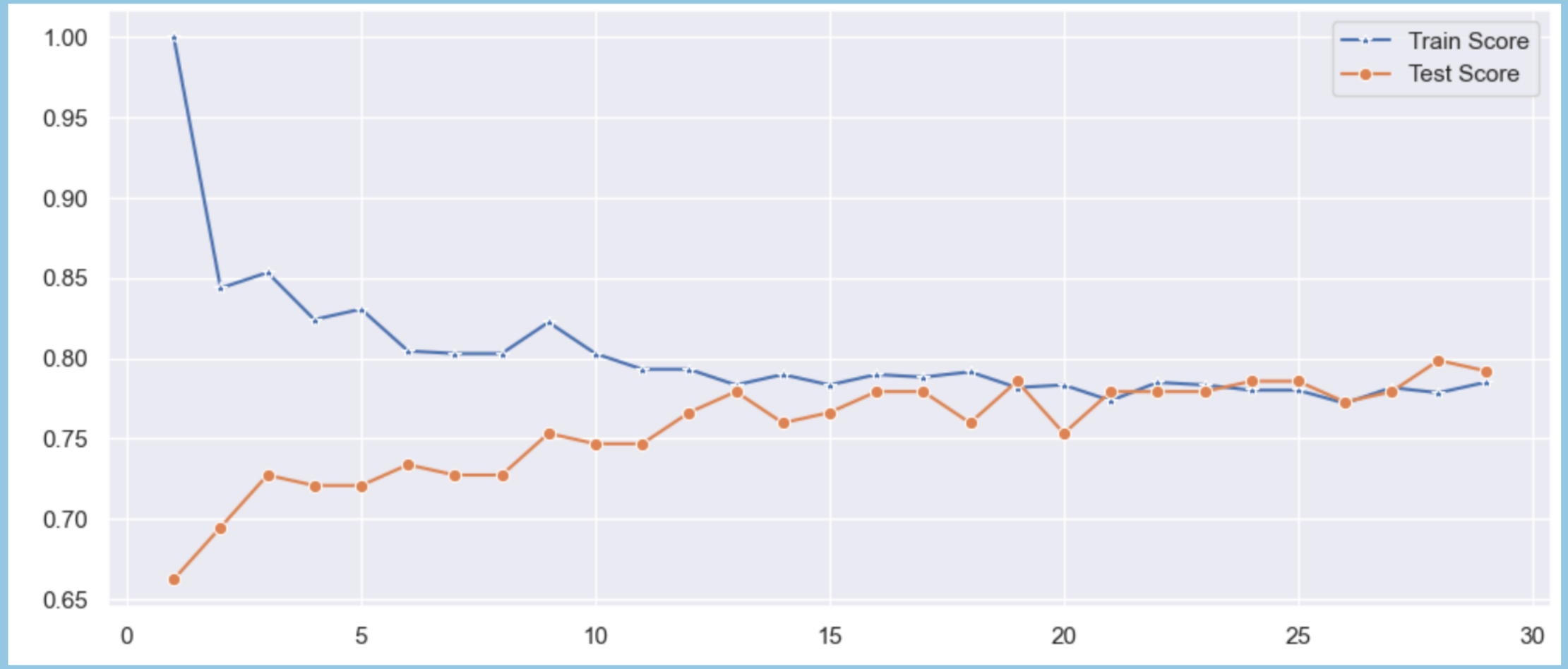
Standardization

- Used StandardScaler to remove the mean and scale each variable to unit variance
- Put predictor variables in X (pregnancy to age)
- Put outcome variable in Y (outcome)

```
[[ 0.63994726  0.86510807 -0.03351824 ...  0.16661938  0.46849198
   1.4259954 ]
 [-0.84488505 -1.20616153 -0.52985903 ... -0.85219976 -0.36506078
  -0.19067191]
 [ 1.23388019  2.0158134  -0.69530596 ... -1.33250021  0.60439732
  -0.10558415]
 ...
 [ 0.3429808  -0.0225789  -0.03351824 ... -0.910418   -0.68519336
  -0.27575966]
 [-0.84488505  0.14180757 -1.02619983 ... -0.34279019 -0.37110101
   1.17073215]
 [-0.84488505 -0.94314317 -0.19896517 ... -0.29912651 -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
```

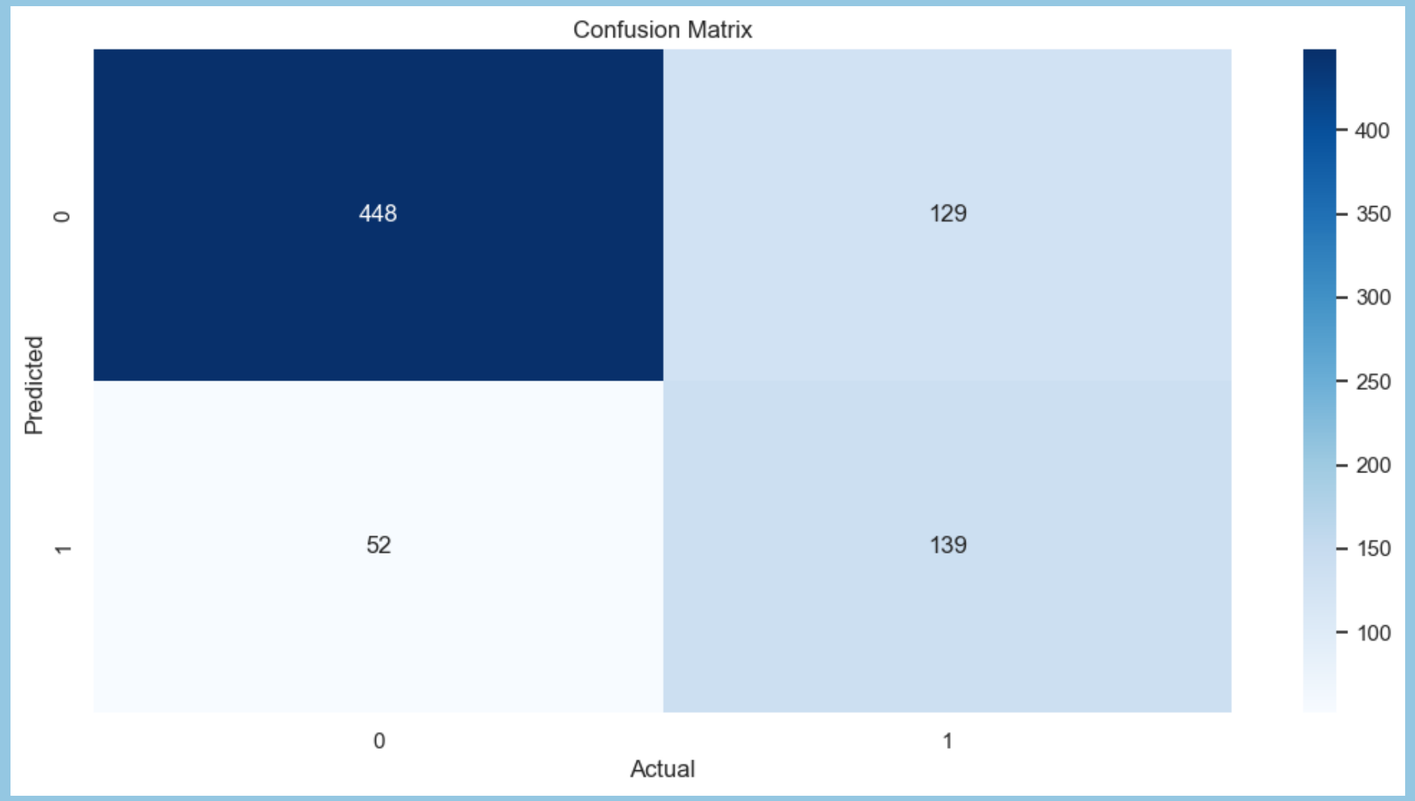
Model 1
KNearestNeighbour Classification

Setting up the N_Nearest for KNN



The highest score from nearest neighbors is 28

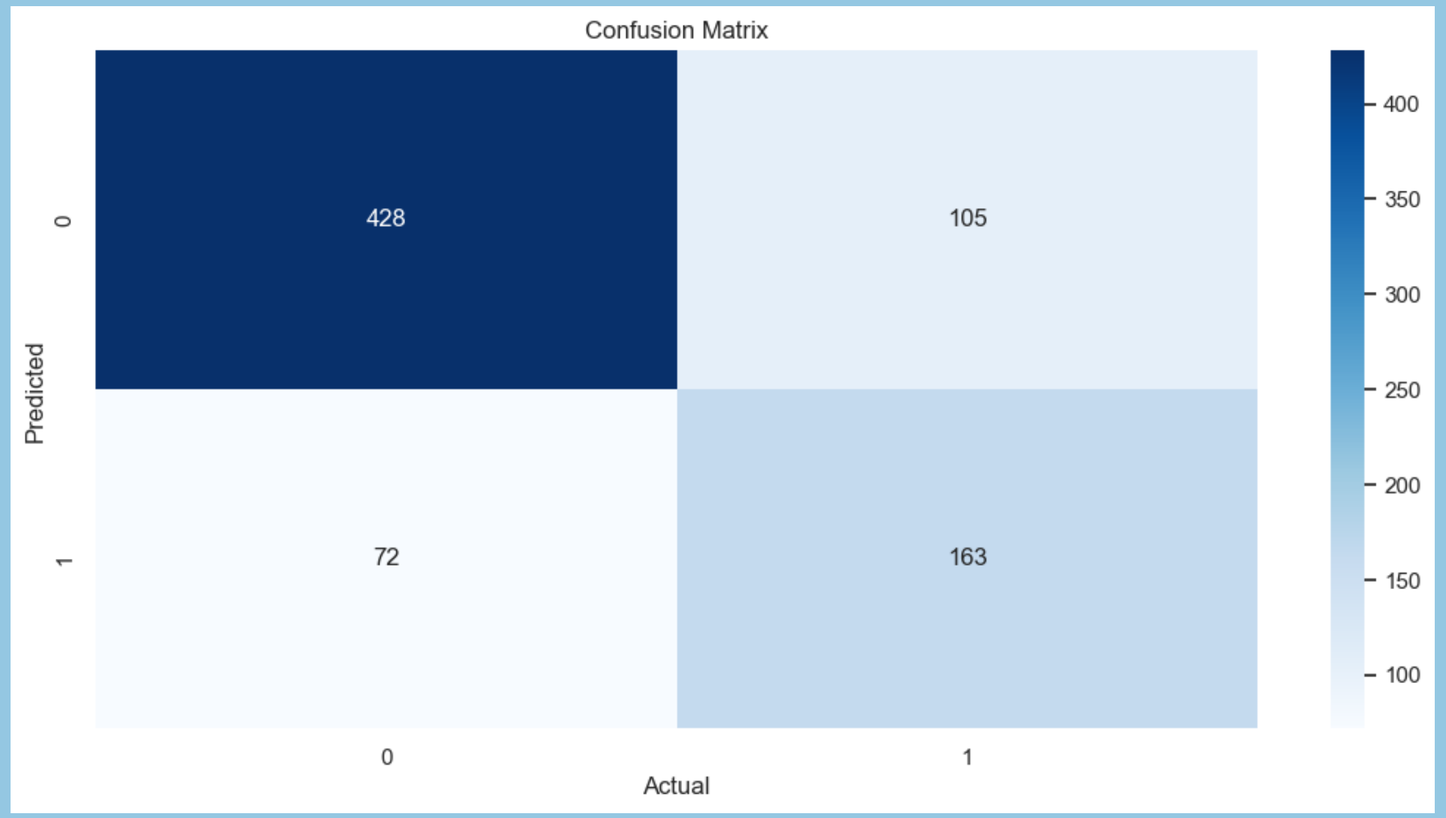
Confusion Matrix: KNN (K- Fold Cross Validation)



- True positives: 448, predicted as Non Diabetic when the person is actually Non Diabetic
- True negatives: 139, predicted as Diabetic when the person is actually Diabetic
- False positives: 129, predicted as Non Diabetic when the person is actually Diabetic
- False negatives: 52, predicted as Diabetic when the person is actually Non Diabetic

Model 2: Random Forest Classification

Confusion Matrix-Random Forest (K Fold Cross Validation)



- True positives: 428, predicted as Non Diabetic when the person is actually Non Diabetic
- True negatives: 163, predicted as Diabetic when the person is actually Diabetic
- False positives: 105, predicted as Non Diabetic when the person is actually Diabetic
- False negatives: 72, predicted as Diabetic when the person is actually Non Diabetic

*Used GridSearchCV for Hyper Parameter Optimization

Random Forest Classifier

		Actual		
		Non Diabetic	Daibetic	Total
Predicted	Non Diabetic	428	105	533
	Diabetic	72	163	235
	Total			**768**

Overall accuracy: 77%

Stratified accuracy:

- Non Diabetic: 80%
- Diabetic: 30.6%

FP's: 105

P(FP): 0.196

K-Nearest Neighbors

		Actual		
		Non Diabetic	Daibetic	Total
Predicted	Non Diabetic	448	129	577
	Diabetic	52	139	191
	Total			**768**

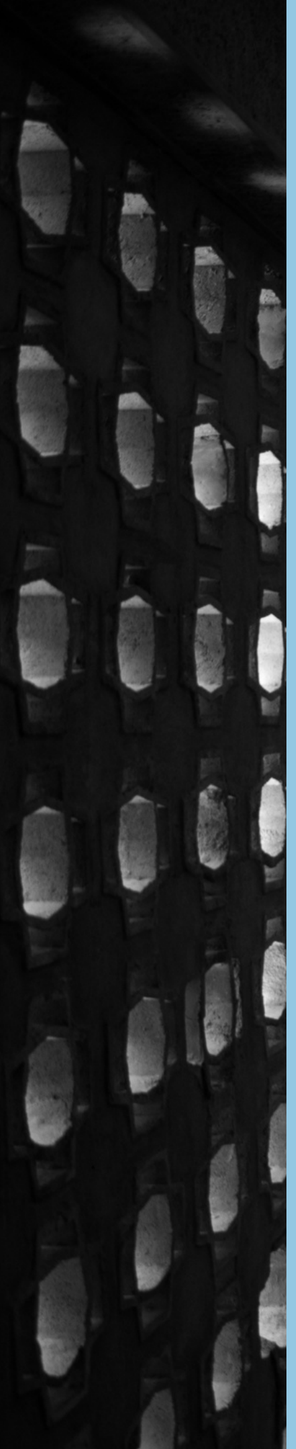
Overall accuracy: 76.4%

Stratified accuracy:

- Non Diabetic: 77.6%
- Diabetic: 27.2%

FP's: 129

P(FP): 0.223



Model Evaluation: Expected value gain per insttance

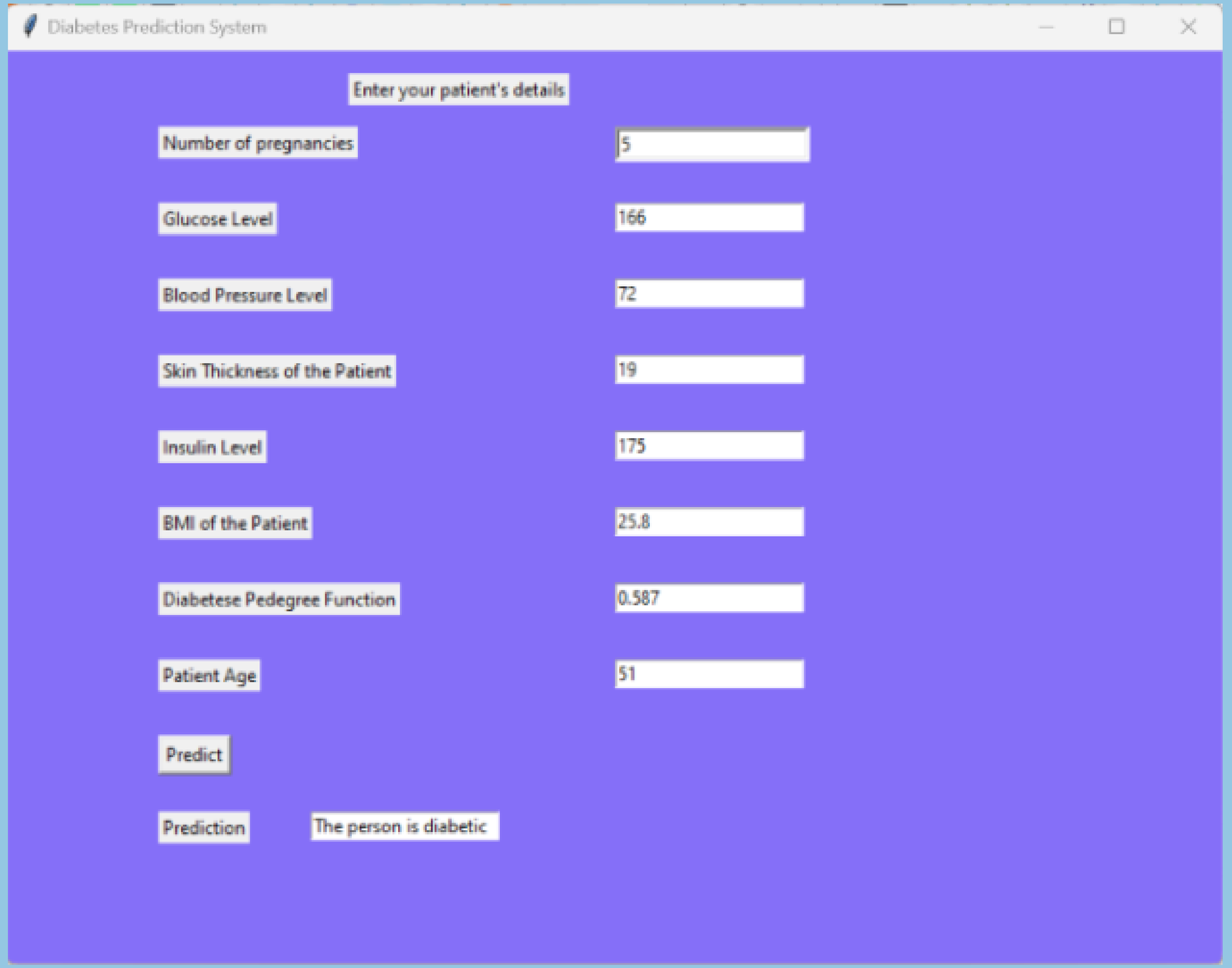
- Assumptions:
  - For correct predictions i.e predicting diabetic and non-diabetic for the actual diabetic and non-diabetic patients' resp. Per instance, gain is 80 units
  - For False Positives i.e predicting as non-diabetic when actually diabetic. Per instance, loss is 40 units

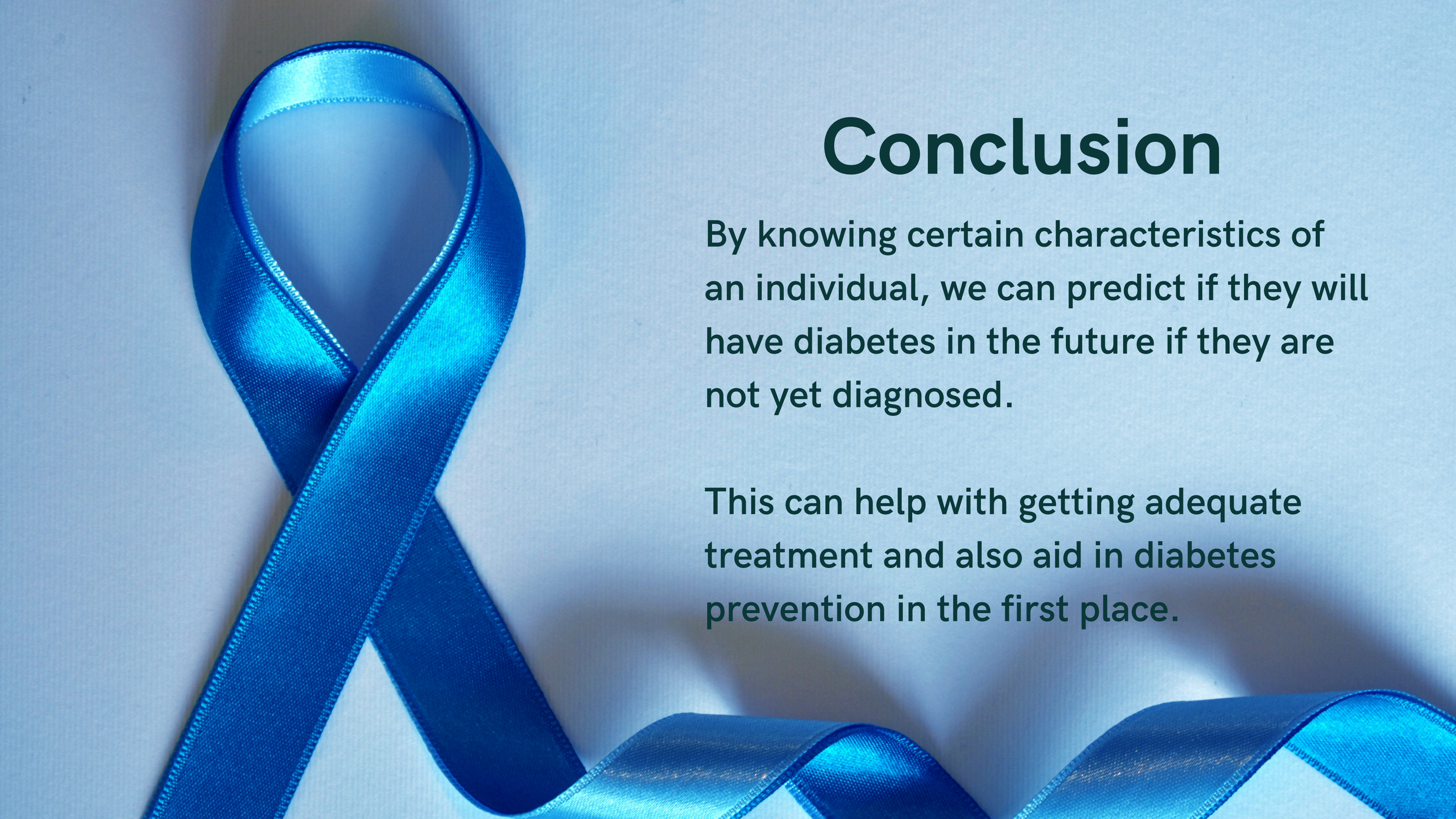
- Expected value gain per instance:
  - Random forest: 56.09 units
  - KNN: 54.43 units

INTERPRETATIONS

- Random forest model has fewer number of false positives (diabetic but predicted as non-diabetic) than KNN
- Overall accuracy of random forest model is better than KNN: 77% vs. 76.4%
- Recommend using random forest model to accurately predict diabetes

PREDICTIVE SYSTEM using Random-Forest Classification


Diabetes Prediction System
Enter your patient's details
Number of pregnancies
5
Glucose Level
166
Blood Pressure Level
72
Skin Thickness of the Patient
19
Insulin Level
175
BMI of the Patient
25.8
Diabetese Pedegree Function
0.587
Patient Age
51
Predict
Prediction
The person is diabetic



Conclusion

By knowing certain characteristics of an individual, we can predict if they will have diabetes in the future if they are not yet diagnosed.

This can help with getting adequate treatment and also aid in diabetes prevention in the first place.

Thank You!
Q&A