DIABETES PREDICTION USING AI

PHASE III

INTRODUCTION

Diabetes is a chronic disease that directly affects the pancreas, and the body is incapable of producing insulin. Insulin is mainly responsible for maintaining the blood glucose level. Many factors, such as excessive body weight, physical inactivity, high blood pressure, and abnormal cholesterol level, can cause a person get affected by diabetes. It can cause many complications, but an increase in urination is one of the most common ones. It can damage the skin, nerves, and eyes, and if not treated early, diabetes can cause kidney failure and diabetic retinopathy ocular disease. According to IDF (International Diabetes Federation) statistics, 537 million people had diabetes around the world in 2021. In Bangladesh, approximately 7.10 million people had suffered from this disease, according to 2019 statistics.

Early and accurate diagnosis of diabetes mellitus, especially during its initial development, is challenging for medical professionals. Artificial intelligence and machine learning techniques, providing a reference, can help them gain preliminary knowledge about this disease and reduce their workload accordingly. Significant numbers of research have been performed to predict diabetes automatically using machine learning and ensemble techniques. Most of these works employed the open-source Pima Indian dataset. Some of these articles on automatic diabetes prediction employing the Pima Indian dataset are briefly discussed in the following paragraphs. For instance, Kumar et al. used the random forest algorithm to design a system that can predict diabetes quickly and accurately. The dataset used in this work was collected from the UCI learning repository. First, the authors used conventional data preprocessing techniques, including data cleaning, integration, and

reduction. The accuracy level was 90% using the random forest algorithm, which is much higher when compared to other algorithms. In a recent paper, Mohan and Jain used the SVM algorithm to analyze and predict diabetes with the help of the Pima Indian Diabetes Dataset. This work used four types of kernels, linear, polynomial, RBF, and sigmoid, to predict diabetes in the machine learning platform. The authors obtained diverse accuracies in different kernels, ranging between 0.69 and 0.82. The SVM technique with radial basis kernel function obtained the highest accuracy of 0.82. Goyal and his team created a smart home health monitoring scheme to detect diabetes. The authors also employed the Pima Indian dataset for their research. For predicting blood pressure status, they used conditional decision making and for predicting diabetes, they used SVM, KNN, and decision tree. Among these models, SVM worked better as they got 75% accuracy which is better than other classifier algorithms. Hassan et al. attempted to predict diabetes using different ensemble method-based machine learning algorithms and the Pima Indian dataset. The authors considered AUC (area under the ROC curve) as their accuracy measure. Finally, the proposed ensemble classifier accomplished an AUC value of 0.95. Jackins et al. proposed a multi-disease prediction system, including diabetes using machine learning techniques and the Pima Indian dataset. According to the authors, the Naive Bayes performed better than the random forest technique with accuracy increments of 0.43%. Mounika et al. anticipated diabetes probabilities using machine learning techniques. This work employed the public Pima Indian dataset and multiple machine learning frameworks. Kumari et al. attempted to apply a soft voting classifier-based ensemble approach for diabetes prediction. The proposed soft voting classifier attained the overall highest accuracy and F1 score of 0.791 and 0.716, respectively. Prabhu and Selvabharathi used the open-source Pima Indian diabetes dataset for predicting diabetes using the deep belief network model. The authors constructed the model in three phases, that is, data preprocessing using min–max normalization, constructing the network model, and fine-tuning the test dataset to remove any partiality using NN-FF classification. Finally, the authors have done all the implementation and simulation of the model using MATLAB. Some of these works employed custom datasets or a

combination of different datasets. In, the authors proposed a type 2 diabetes early prediction system using machine learning approaches. The authors employed a private dataset with more than 253,000 volunteer data from a local hospital in Korea for 6 years. Synthetic oversampling, SMOTE, and undersampling algorithms are applied to deal with the data imbalance problem. Various machine learning approaches are used to anticipate this disease for the following year from the past year's patients' data. Both the random forest and SVM classifiers achieved the highest F1 score of 74%. Pranto et al. utilized Pima Indian and a private dataset from a local hospital in Bangladesh to design an automatic diabetes prediction system. This work trained several machine learning techniques on the Pima Indian dataset. KNN and decision tree models achieved 81.2% and 79.2% accuracies on the private dataset, respectively. Olisah et al. implemented diabetes mellitus forecasting using advanced feature selection and machine learning models. The authors employed two open-source datasets, that is, Pima Indian and LMCH Iraqi databases. A polynomial regression-based preprocessing technique was used for predicting the missing samples. Hyperparameter tuning has been performed for the random forest, decision tree, and deep neural network frameworks. The proposed DNN technique with the optimized hyperparameters accomplished the highest accuracies of 0.972 and 0.973 for the Pima and LMCH datasets, respectively.

We draw the conclusion that researchers have successfully combined multiple machine learning algorithms with diverse data preprocessing approaches for automatic diabetes detection by reviewing the relevant articles. Most of the works focused on a single accuracy measure, used the open-source Pima Indian dataset, and did not develop the explicability of the prediction of the machine learning frameworks. These reasons have motivated us to evaluate our proposed prediction system based on accuracy, precision, recall, and F1 score, utilize more custom data to merge with the existing dataset, and apply an explainable AI technique.

In this paper, we have employed machine learning and explainable AI techniques to detect diabetes. Along with a private dataset from employees of a local textile industry in Bangladesh, we used the Pima Indian dataset in this paper. As there were many missing values in some attributes, we replaced them with the mean value of each feature. We have used the holdout validation technique to split the data. In this research paper, we have applied various machine learning-based classification algorithms, that is, decision tree, logistic regression, KNN, random forest, SVM, and ensemble techniques. Next, the performance of these classifiers has been evaluated in terms of precision, recall, and F1 measure. Finally, the best classifier has been selected as the final model to deploy into an Android smartphone application.

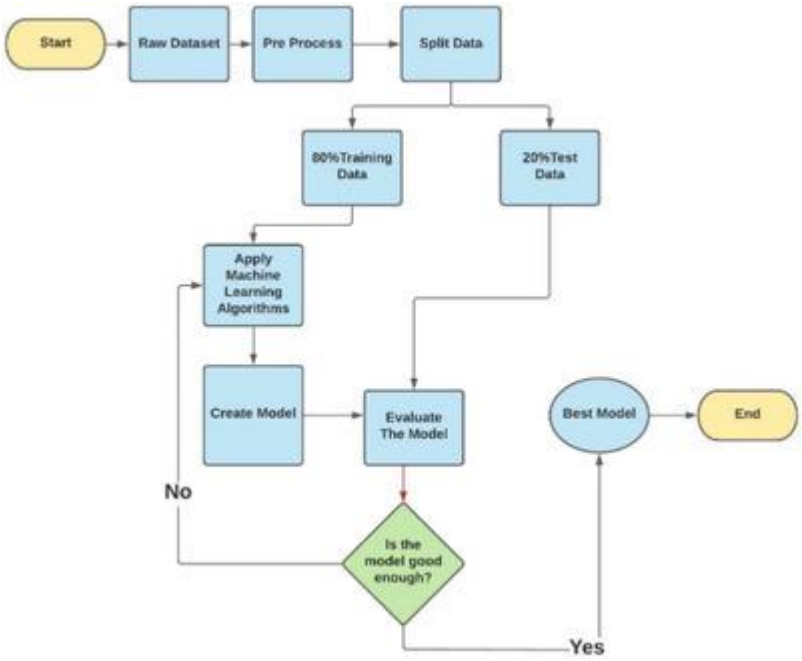
This paper implements diabetes mellitus prediction through machine learning. The significant contribution of this work is as follows:

- A significant contribution of this work is to present a unique dataset of diabetes mellitus containing 203 samples. This private dataset has been obtained from female employees of Rownak Textile Mills Ltd, Dhaka, Bangladesh, referred to as the 'RTML dataset' in this paper. We have collected six features from 203 individuals, that is, pregnancy, glucose, blood pressure, skin thickness, BMI, age, and final outcome of diabetes.
- Another contribution of this work is to keep similarities with the feature of the Pima Indian dataset. The missing insulin feature of the RTML dataset was predicted using a semi-supervised technique.
- SMOTE and ADASYN techniques are implemented to minimize the class imbalance issue. Hyperparameter tuning has also been performed in this work.
- Explainable AI technique with SHAP and LIME libraries is implemented to understand how the model predicts the decision. This approach helps to interpret what features play the most crucial role in terms of prediction.
- A website and an Android application have been developed with the finalized best-performed model of this research work to make instantaneous predictions with real-time data.

The novelty of this work is to implement an automatic diabetes prediction website and Android application for a private dataset of female Bangladeshi patients using machine learning and ensemble techniques.

PROPOSED SYSTEM

This section describes the working procedures and implementation of various machine learning techniques to design the proposed automatic diabetes prediction system. Figure 1 shows the different stages of this research work. First, the dataset was collected and preprocessed to remove the necessary discrepancies from the dataset, for example, replacing null instances with mean values, dealing with imbalanced class issues etc. Then the dataset was separated into the training set and test set using the holdout validation technique. Next, different classification algorithms were applied to find the best classification algorithm for this dataset. Finally, the best-performed prediction model is deployed into the proposed website and smartphone application framework.



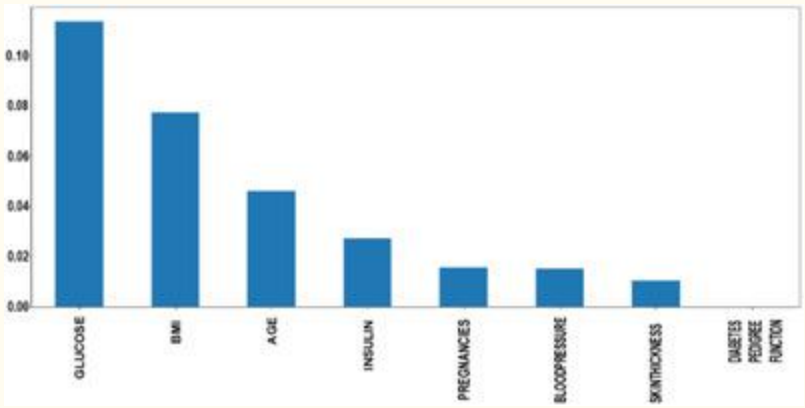
Working sequences of the proposed diabetes prediction system

DATASET PREPROCESSING

In the merged dataset, we discovered a few exceptional zero values. For example, skin thickness and Body Mass Index (BMI) cannot be zero. The zero value has been replaced by its corresponding mean value. The training and test dataset has been separated using the holdout validation technique, where 80% is the training data and 20% is the test data.

Mutual Information: Mutual information attempts to measure the interdependence of variables. It produces information gain, and its higher values indicate greater dependency.

Figure shows the mutual information of various features, that is, the importance of each attribute of this dataset. For example, according to this figure, the diabetes pedigree function seems less important according to this mutual information technique.



Feature importance hierarchy

Semi-supervised learning: A combined dataset has been used in this work by incorporating the open-source Pima Indian and private RTML datasets. According to Table, the RTML dataset does not contain the

insulin feature, which is predicted using a semi-supervised approach. Before merging the collected dataset with the Pima Indian dataset, a model was created using the extreme gradient boosting technique (XGB regressor). Various regression and ensemble learning techniques have been successfully used in many works to predict missing values. An extensive investigation has been performed while choosing the best-performed regressor technique to predict the insulin feature of the RTML dataset from the Pima Indian dataset. As the actual value of the insulin was not available in the RTML dataset, the Pima Indian dataset was initially used to select the best regression model. First, the Pima Indian dataset was divided into an 8:2 ratio and three supervised regression models, extreme gradient boosting technique (XGB), support vector regression (SVR), and Gaussian process regression (GPR), have been employed to predict the selected outcome, that is, insulin of the validation samples of the Pima Indian dataset. Next, we computed the root mean square error (RMSE) of various regression frameworks as

$$RMSE=\sum Ni=1(Predictedi-Actuali)2N----------\sqrt{}$$

where *N* denotes the total number of validation samples of the Pima Indian dataset.

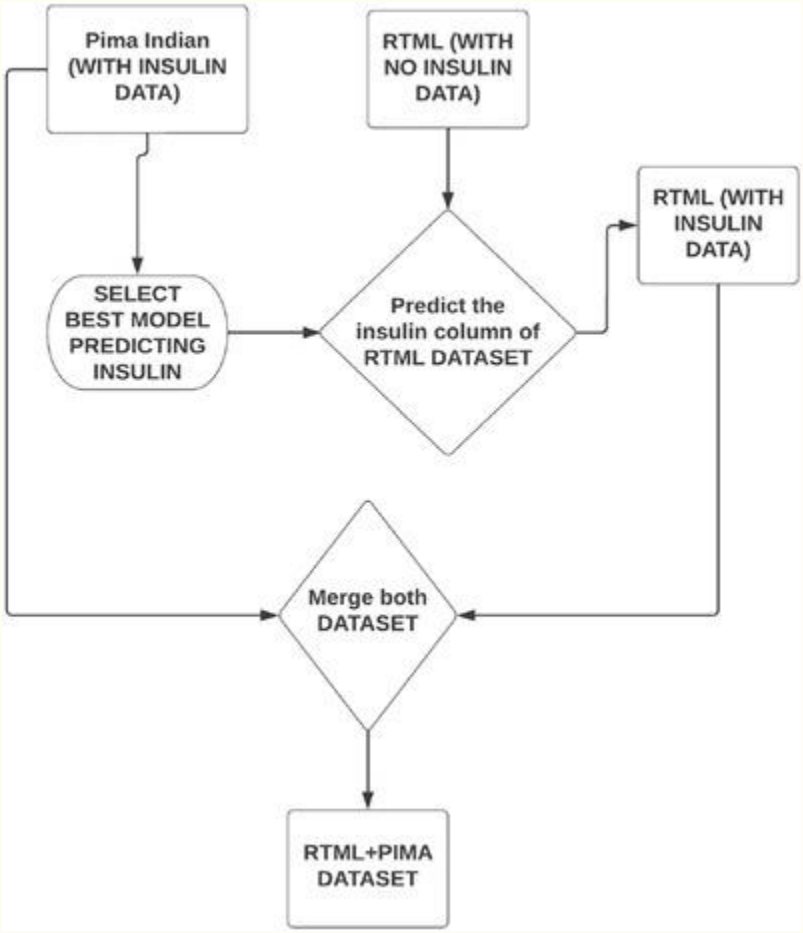
According to Table, the XGB technique exhibits the lowest RMSE of insulin on the Pima Indian dataset. Therefore, this model has been used to predict the missing insulin column of the collected RTML dataset from the Pima Indian dataset. The working steps of predicting insulin in the RTML dataset have been illustrated in Figure.

TABLE

RMSE of various regression models on the Pima Indian dataset

Regression model	RMSE
XGB	0.36
SVR	0.45

Regression model	RMSE
GPR	0.43



Working steps of predicting insulin of the RTML dataset

Merged dataset: After the semi-supervised approach, we predicted the insulin feature and merged the RTML dataset with the Pima Indian dataset. The merged dataset contained 877 data with all the features, excluding the diabetes pedigree function, as it was the least important feature according to mutual information.

SMOTE and ADASYN for class imbalance: The merged dataset used in this work comprises the imbalance problem with 302 and 669 diabetes

and non-diabetes samples, respectively. To take care of this problem, the SMOTE and ADASYN techniques have been applied to the training dataset, leaving the testing data unaffected. Adaptive Synthetic Sampling, known as ADASYN, is a synthetic data generation technique with the characteristics of not duplicating minority samples and generating more data for 'harder to learn' examples. As a result, the minority class will be sampled to the same extent as the majority class.

Min–Max normalization: In this research, we used the min–max normalization technique. The data has been scaled to the same range using the following equation:

$$X_{scaled} = X - X_{min} X_{max} - X_{min}$$

where X_{max} and X_{min} denote maximum and minimum values in the individual feature column, respectively.

DEVELOPMENT

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import shapiro
from sklearn.datasets import make_classification
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.metrics import accuracy_score

from sklearn.metrics import confusion_matrix

from sklearn.metrics import classification_report


import os

for dirname, _, filenames in os.walk('/kaggle/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))
```

Load the dataset and perform EDA

```
data = pd.read_csv('/content/diabetes.csv')
```

```
data.info()
```

Output :

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
data.describe()
```

Output :

	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

```
data.head()
```

Output:

	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

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
4	0	137	40	35	168	43.1	2.288	33	1

```
data.shape
```

Output:

```
(768, 9)
```

```
duplicates = data.duplicated()
if duplicates.any():
    print("Duplicates found!")
    duplicate_rows = data[duplicates]
    print(duplicate_rows)
else:
    print("No duplicates found!")
```

Output:

```
No duplicates found!
```

```
features_variables = data.iloc[:, 0:8]
#get the outcome variables
outcome_variable = data.iloc[:, -1]
```

```
print("Feature variables")
features_variables
```

Output:

```
Feature variables
```

	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

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
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

```
print("Outcome variable")
outcome_variable
```

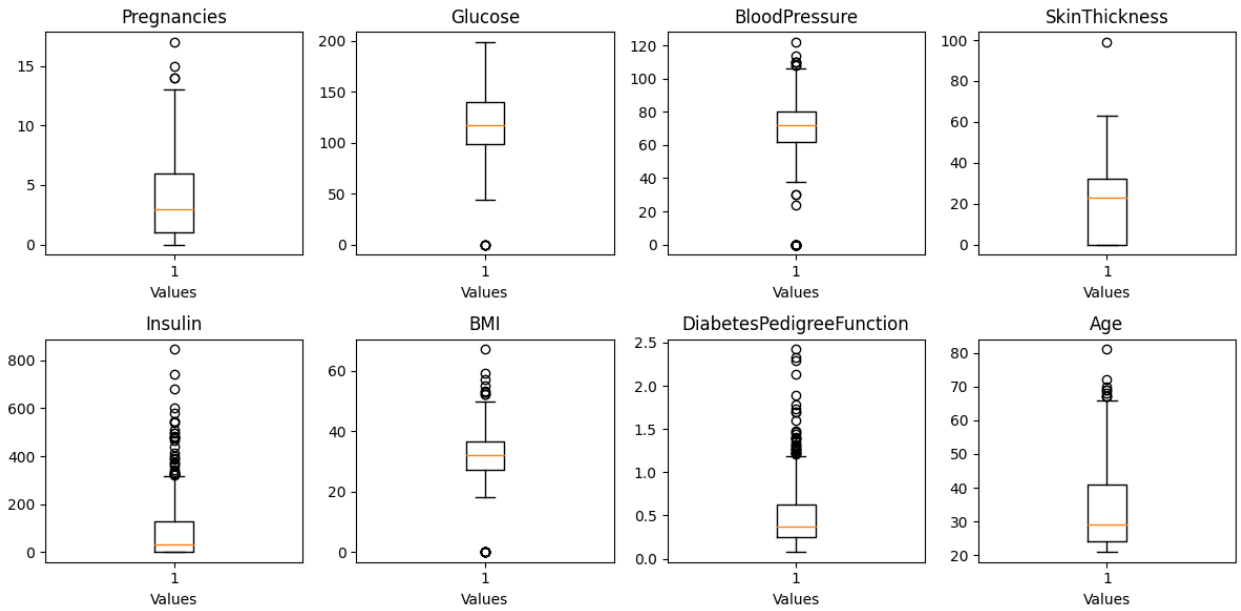
Output:

```
Outcome variable
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
```

```
fig, axs = plt.subplots(nrows=2, ncols=4, figsize=(12, 6))
```

```
axs = axs.flatten()

for i, feature in enumerate(features_variables.columns):
    axs[i].boxplot(features_variables[feature])
    axs[i].set_title(feature)
    axs[i].set_xlabel('Values')

for j in range(len(features_variables.columns), len(axs)):
    fig.delaxes(axs[j])

fig.tight_layout()

plt.show()
```

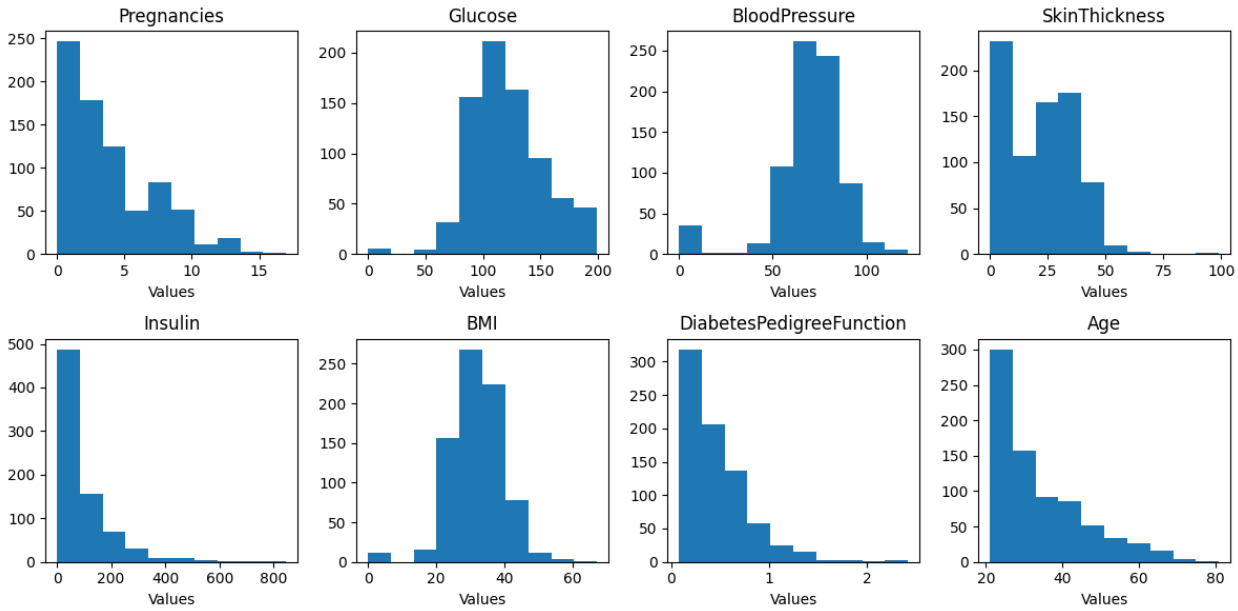
Output:




```
fig, axs = plt.subplots(nrows=2, ncols=4, figsize=(12, 6))

axs = axs.flatten()

for i, feature in enumerate(features_variables.columns):
    axs[i].hist(features_variables[feature])
    axs[i].set_title(feature)
    axs[i].set_xlabel('Values')

for j in range(len(features_variables.columns), len(axs)):
    fig.delaxes(axs[j])
```

```
fig.tight_layout()

plt.show()
```

Output:




```
alpha = 0.05
for column in features_variables.columns:
    column_data = features_variables[column]
    statistic, p_value = shapiro(column_data)

    print("Column:", column)
    print("Statistic:", statistic)
    print("p_value:", p_value)


    if (p_value < alpha):
        print("The data does not follow normal distribution")
    else:
        print("The data follows normal distribution")
    print()
```

Output:

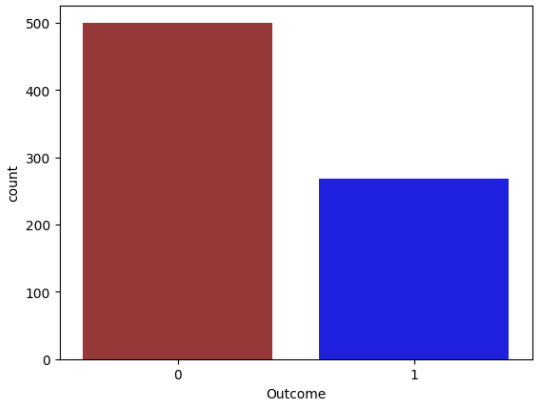
```
Column: Pregnancies
Statistic: 0.9042831659317017
p_value: 1.6100101271064348e-21
```

```
The data does not follow normal distribution

Column: Glucose
Statistic: 0.9701048731803894
p_value: 1.987464880170986e-11
The data does not follow normal distribution

Column: BloodPressure
Statistic: 0.81892329454422
p_value: 1.5844936208677322e-28
The data does not follow normal distribution

Column: SkinThickness
Statistic: 0.904627799987793
p_value: 1.751799708531821e-21
The data does not follow normal distribution

Column: Insulin
Statistic: 0.7220208644866943
p_value: 7.915339984765649e-34
The data does not follow normal distribution

Column: BMI
Statistic: 0.9499890208244324
p_value: 1.8407586602041262e-15
The data does not follow normal distribution

Column: DiabetesPedigreeFunction
Statistic: 0.8365188837051392
p_value: 2.4777990069755762e-27
The data does not follow normal distribution

Column: Age
Statistic: 0.874765932559967
p_value: 2.401829612553989e-24
The data does not follow normal distribution
```

```
sns.countplot(x = outcome_variable, data = data, palette=['brown',
'blue'])
```

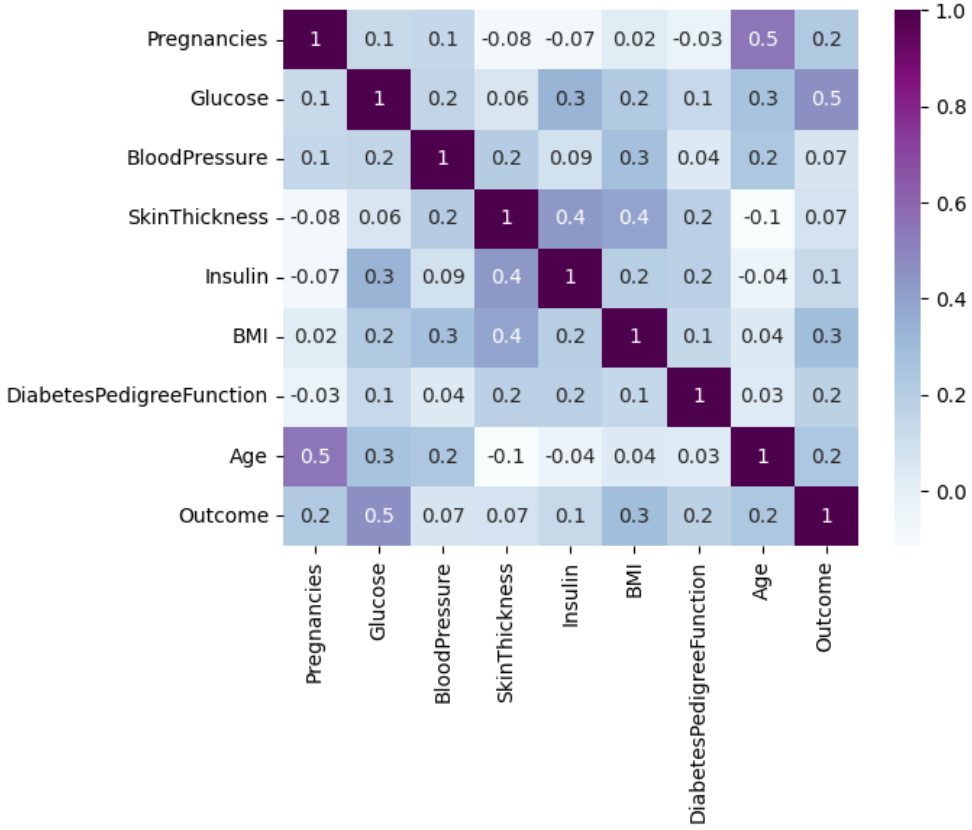
Output:

```
<Axes: xlabel='Outcome', ylabel='count'>
```




```
column_corr = data.corr()
heatmap = sns.heatmap(column_corr, annot=True, cmap="BuPu", fmt='.1g')
```

Output:



Logistic regression

```
X = data.iloc[:, :-1]
y = data.iloc[:, -1]
```

```
X, y = make_classification(random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)

pipe = make_pipeline(StandardScaler(), LogisticRegression())

pipe.fit(X_train, y_train)

pipe.score(X_test, y_test)
```

Output:

```
0.96
```

```
y_pred = pipe.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Output:

```
Accuracy: 0.96
```

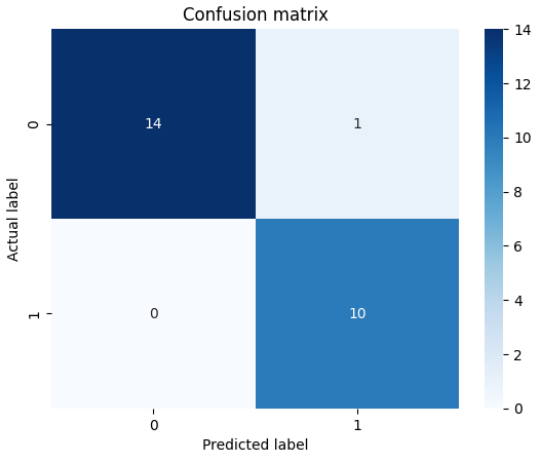
```
cm = confusion_matrix(y_true=y_test, y_pred=y_pred)
cm
```

Output:

```
array([[14, 1], [ 0, 10]])
```

```
class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)

sns.heatmap(cm, annot=True, cmap="Blues" ,fmt='g')
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.title('Confusion matrix')
plt.show()
```

Output:




```
target_names = ['without diabetes', 'with diabetes']
print(classification_report(y_test, y_pred, target_names=target_names))
```

Output:

```
                   precision    recall  f1-score   support

without diabetes       1.00      0.93      0.97        15
   with diabetes       0.91      1.00      0.95        10

        accuracy                           0.96        25
       macro avg       0.95      0.97      0.96        25
    weighted avg       0.96      0.96      0.96        25
```

Conclusion

In conclusion, the use of artificial intelligence (AI) in predicting diabetes is a promising and valuable tool in the field of healthcare. AI-based predictive models have the potential to significantly improve early detection, management, and overall outcomes for individuals at risk of or already living with diabetes. Here are some key takeaways:

1. Early Detection: AI models can analyze vast amounts of patient data, including medical records, genetic information, and lifestyle factors, to identify individuals at risk of developing diabetes. This early detection can enable timely interventions and lifestyle modifications to prevent or delay the onset of the disease.

2. Personalized Medicine: AI can create personalized treatment plans based on an individual's unique health profile. This allows for more effective management of diabetes, tailored to the specific needs of each patient, leading to improved outcomes and quality of life.

3. Continuous Monitoring: AI-powered wearable devices and apps can provide real-time monitoring of blood glucose levels, physical activity, and dietary habits. This continuous feedback allows individuals to make informed choices and healthcare providers to adjust treatment plans as needed.

4. Reducing Healthcare Costs: By predicting and preventing complications associated with diabetes, such as kidney disease, cardiovascular problems, and neuropathy, AI can help reduce the economic burden on healthcare systems and improve the overall well-being of patients.

5. Research and Drug Development: AI can accelerate the discovery of new therapies and medications for diabetes. Machine learning algorithms can analyze vast datasets to identify potential drug candidates and optimize clinical trials, potentially speeding up the process of finding more effective treatments.

6. Ethical and Privacy Concerns: While AI offers many benefits, it also raises ethical and privacy concerns, especially regarding the use of personal health data. Striking a balance between data privacy and innovation is crucial in the development and implementation of AI in diabetes prediction.

In conclusion, AI has the potential to revolutionize the field of diabetes prediction and management. It offers the prospect of earlier, more accurate diagnoses, personalized treatment plans, and improved quality of life for individuals with diabetes. However, it is important to address ethical and privacy considerations as AI continues to be integrated into healthcare, ensuring that the benefits are maximized while respecting patient rights and data security. The ongoing collaboration between healthcare professionals, AI developers, and policymakers will be vital in harnessing the full potential of AI in diabetes care.