DATA ANALYSIS

DS3114

PROJECT TITLE:

Diabetes Dataset - Pima Indians

COURSE REPRESENTS :
(DR.Omaima A. Fallatah)

SUBMITTED BY:

Name	ID
Lamar Waleed Fattah	444006719
Rakha Matuq Nooh	444001287

COLLEGE OF COMPUTING UMM AL-QURA UNIVERSITY

Table of content

1. Introduction

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on selecting these instances from a larger database. We utilized all the features including **Age**, **DiabetesPedigreeFunction**, **BMI**, **Insulin**, **SkinThickness**, **BloodPressure**, **Glucose**, and **Pregnancies**. The **Outcome** variable was employed as the target feature to predict whether a patient is diabetic or not diabetic.

2. Data Exploration and Preprocessing

The dataset is first explored to understand its structure, identify missing values, and summarise key statistics The target variable, **Outcome**, is used to predict whether the patient has diabetes based on various health features like **Glucose**, **BloodPressure**, **SkinThickness**, etc.

- Pregnancies: To express the Number of pregnancies
- Glucose: To express the Glucose level in blood
- BloodPressure: To express the Blood pressure measurement
- SkinThickness: To express the thickness of the skin
- Insulin: To express the Insulin level in blood
- BMI: To express the Body mass index
- DiabetesPedigreeFunction: To express the Diabetes percentage
- Age: To express the age
- Outcome: To express the final result 1 is YES o is NO

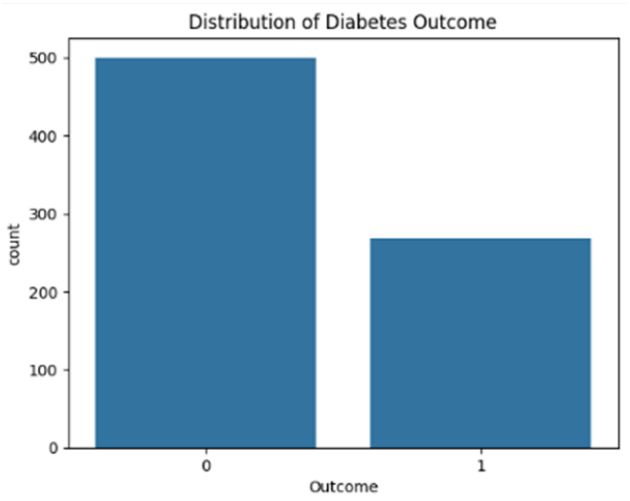
2.1 Data Exploring

After we installed the important Libraries the data set has been thoroughly evaluated and confirmed to be clean and issue-free. After applying the functions f.isnull().sum(), df.dtypes, and .duplicated().sum(), it was determined that there are no missing values, incorrect data types, or duplicated entries. These checks confirm that the data is well-prepared for analysis or further processing. This ensures that the dataset is of high quality, ready for reliable and accurate interpretation or modelling.

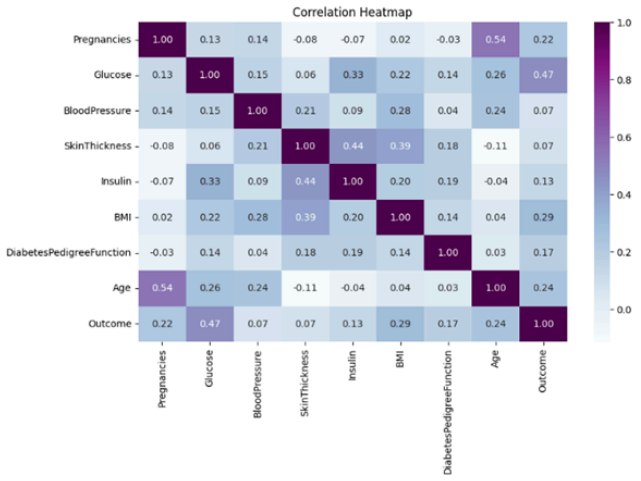
```
df=pd.read_csv('/content/diabetes.csv')
df.head()
```

	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

The Dataset that we aim to analyse



1 as have diabetes , 0 as not have no balance of the target



Explore correlations between features

3.Model Employment

3.1 The Naive Bayes

The Naive Bayes model is a classification algorithm that applies Bayes' Theorem with the assumption that the features (predictors) are conditionally independent given the target variable. Despite this strong assumption, it performs well in many real-world tasks. Types of Naive Bayes classifiers are one that we used in the dataset: Gaussian Naive Bayes is used when features follow a normal distribution (continuous data).

```
[15] X = df.iloc[:, 0:8]  #all columns
     y = df['Outcome']

     # Train Test Split
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

x: This variable contains the feature set, columns from index 0 to 8.

y: The **Outcome** column serves as the target variable, indicating diabetes (1) or not (0).

train_test_split: This function from **sklearn.model_selection** splits the dataset into two parts:

Training set: This is 80% of the data

Testing set: The remaining 20% of the data is used for testing the model's performance.

random_state=42: This ensures that the split is reproducible.

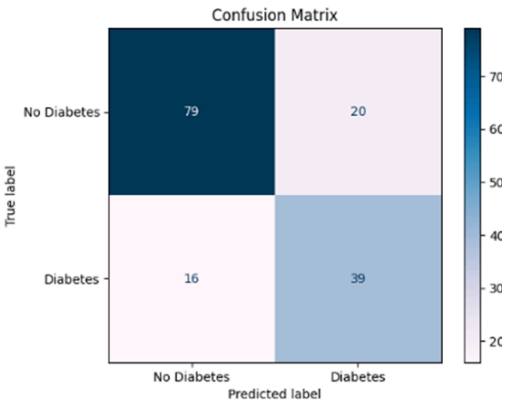
```
# bulid model
model = GaussianNB()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

We build the model using Gaussian

We evaluate the model by calculating the accuracy, confusion matrix, F1-score, and classification




```
Accuracy: 0.7662337662337663
Confusion Matrix:
[[79 20]
 [16 39]]
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.80      0.81        99
           1       0.66      0.71      0.68        55

    accuracy                           0.77       154
   macro avg       0.75      0.75      0.75       154
weighted avg       0.77      0.77      0.77       154
```

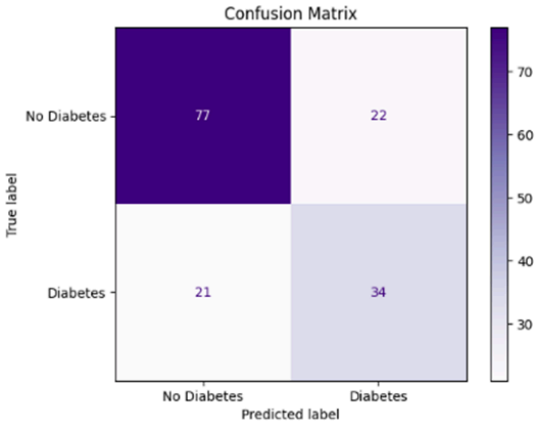
The model performed well!

3.2 Random Forest

*As in the same split we used in **The Naive Bayes** model, it was used in **The Random Forest** model to compare the two models.*

```
[ ]  model = RandomForestClassifier(n_estimators=100, random_state=42)
     model.fit(X_train, y_train)
     y_pred = model.predict(X_test)
```

We evaluate the model by calculating the accuracy, confusion matrix, F1-score, and classification




```
Accuracy: 0.7207792207792207
Confusion Matrix:
[[77 22]
 [21 34]]
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.78      0.78        99
           1       0.61      0.62      0.61        55

    accuracy                           0.72       154
   macro avg       0.70      0.70      0.70       154
weighted avg       0.72      0.72      0.72       154
```

the random forest model performed well too

Evaluation	*Random Forest*	Naive Bayes
Accuracy	0.72	0.78
F1-score	0.82	0.76

4. Result

Both models performed well; however, the Naive Bayes classifier outperformed the Random Forest model by a small margin. The Naive Bayes model achieved an accuracy of 0.77 and an F1-score of 0.81, indicating a more balanced performance in predicting both positive and negative outcomes.

In comparison, the Random Forest model yielded an accuracy of 0.72 and an F1-score of 0.78. While Random Forest is generally a strong and versatile classifier, its performance on this dataset was slightly lower in both metrics.

Based on these results, the Naive Bayes classifier appears to be better suited to this dataset, offering superior predictive accuracy and handling of the class distribution. This suggests that the Naive Bayes model is more effective at capturing the underlying patterns in the data, making it a more appropriate choice for this specific task.