**THE SUPERIOR UNIVERSITY LAHORE**

**Assignment 1**

**Semester: BSAI 4 Section: B**

**Session: Spring 2025**

**Faculty of Computer Science and Information Technology**

**Subject: Programming for Artificial Intelligence Total Marks: 10**

Instructor: Muhammad Hassan Total Marks: 50

****

**dd**

***Instructions:***

1. Time management is the key of success.
2. No extra time shall be given.

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roll No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Advice and Submission Guidelines for Assignment**

* Make sure that you read and understand each and every instruction. If you have any questions or comments you are encouraged to discuss (only) with your colleagues and instructor.
* **Upload PDF file on the Google classroom** and keep files with you (better in your own email) as it will be used for evaluation and viva.
* All the submitted evaluation instruments (quizzes, assignments, lab work, exams, and the project) will be checked for plagiarism.
* Your word/pdf file should be named as your complete roll number e.g. bsse-f23-001.
* Copied assignments will be graded as zero.
* Late Submission will not be accepted.

*While assignments are designed to assess your understanding and skills, they are also opportunities to learn and grow. Enjoy the process of solving problems and building solutions!*

## **Objective:**

The purpose of this project is to develop a machine learning model capable of accurately predicting the likelihood of diabetes in a person based on their clinical and medical records.

**Dataset Selection:**

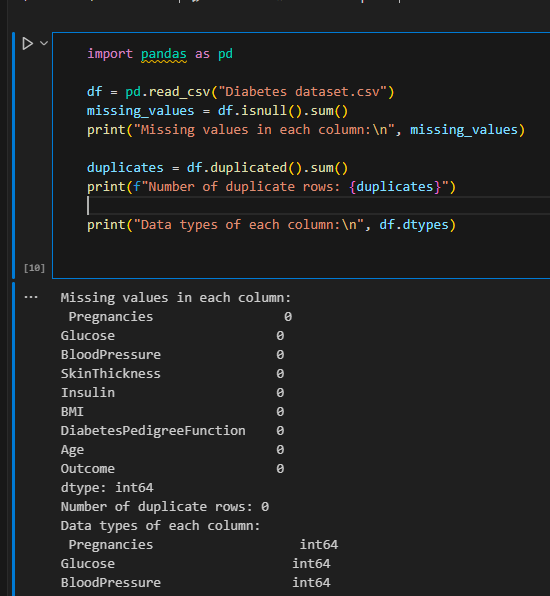
We choose the Pima Indians Diabetes Dataset for our machine learning task. This dataset is well known and widely used in machine learning community.

The dataset contains the medical records of women around the age of 21 who could be the victim of diabetes. So based on the features of dataset, our model will be able to predict the chances of diabetes.  
The dataset’s clear structure and format make it helpful for the team to practice important concepts of machine learning such as preprocessing, feature engineering and classification algorithms.

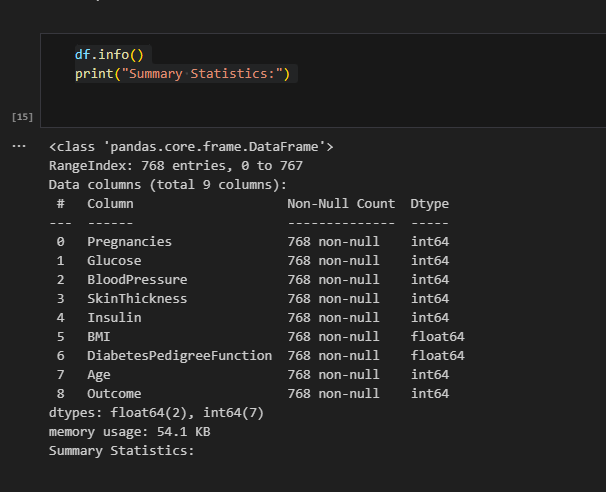
## **Assignment Sections:**

**Initial Data Exploration:**

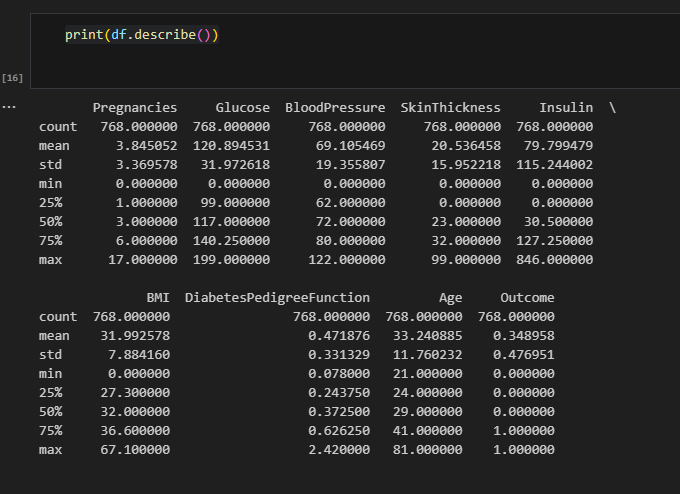
- **Data Loading and analyzing the null and duplicate values:**

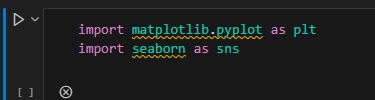
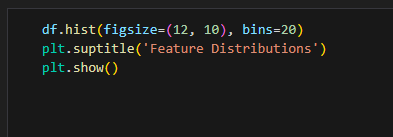
****

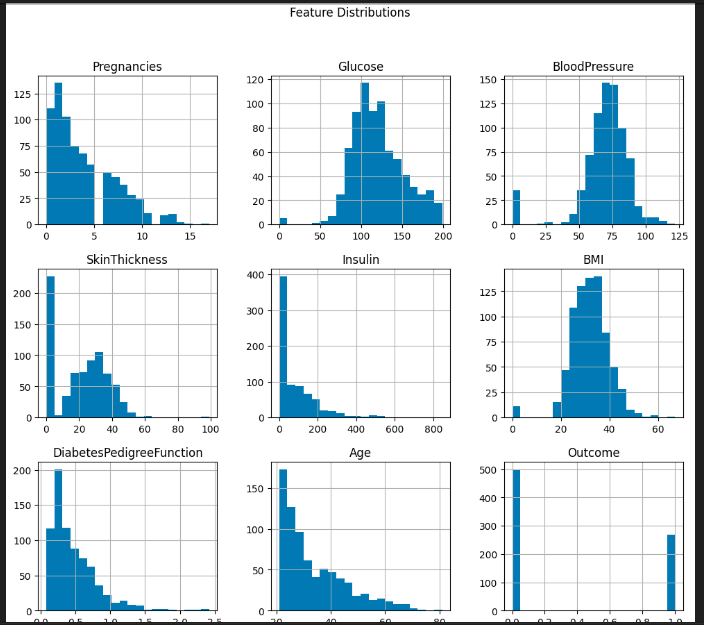
* We are using the pandas library for data manipulation and loading the Diabetes prediction dataset as dataframe of pandas.
* Using the “isnull().sum()” method, we are able to see the null values in each and every column.  
  - For viewing the duplicate values, we use the method called “duplicated()” from pandas.
* These operations allow us to identify the null and duplicates which would help us to avoid inaccurate and biased results by the model.  
  - Using ”dtypes” function, we can inspect the datatypes of columns to identify whether any column needs to be converted or not, i.e: categorical to numerical.  
    
     
   **Dataset Summary:**



**-** Using the “info()” method, we can see summarized information of the dataset.  
 **Statistical Information:**

  
 - The ‘describe()’ method allows us to view the statistical info of the columns. These statistical values such as min, max, percentiles help us to identify outliers or scaling requirements for model training.

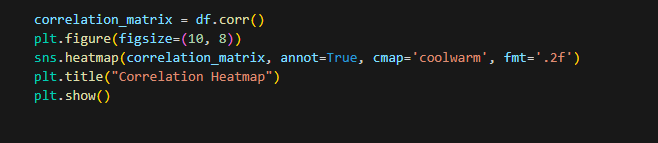
* + **Exploring the dataset using Graphs:**  
     
* For data visualization, we use two important libraries; “matplotlib” and “seaborn”.
* Matplotlib is the library use for creating visual, interactive or animated static graphs  
   Whereas “seaborn” is a high-level interface which is built on matplotlib provides interactive plot graphs in python **Feature Distribution using a Histogram:**
* 

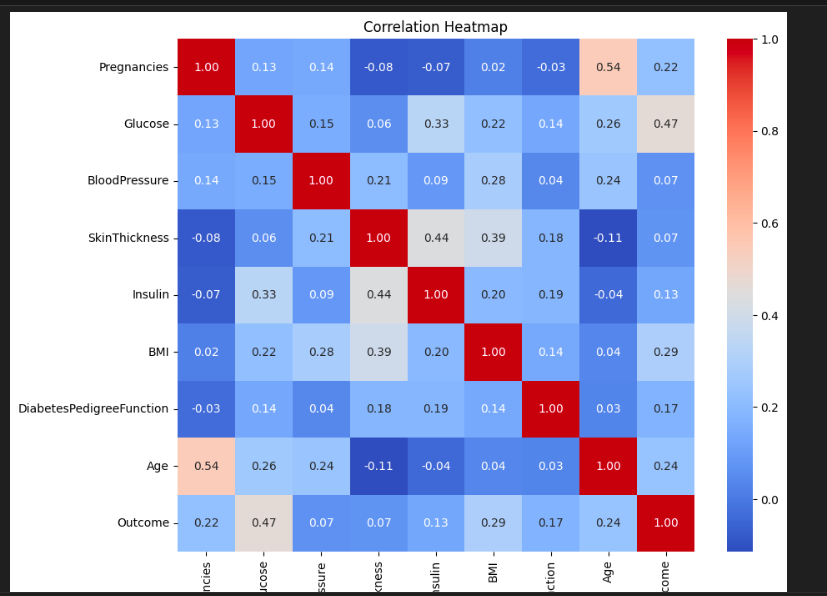


Using a hist() method, a histogram of all numerical features has appeared.

* The purpose of Histogram is to Understand the Data Distribution.
* The x-axis shows the the values of the feature and the y-axis shows the number of people. By seeing this we can detect outliers or can assess the quality of data.
* For example: We can see in the Blood pressure feature, there some values which shows blood pressure is 0 which is biologically incorrect. Hence it is possibly a missing data entry.

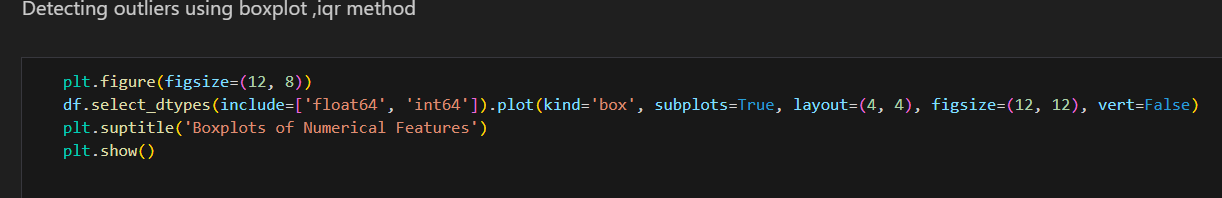
**Correlation Analysis using Heatmap:**

****

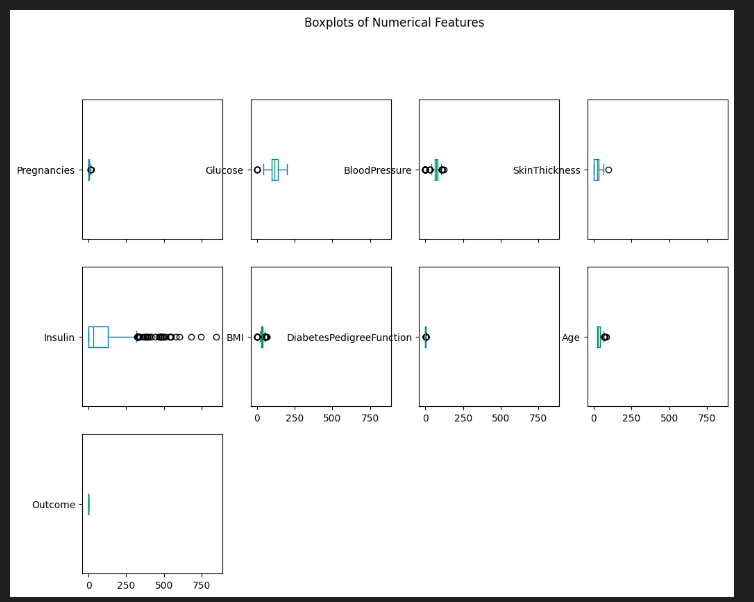
****

* Using “corr()” method to calculate the pairwise correlation between all the numerical features.
* For visualizing the heatmap, we use “sns.heatmap()” function. ‘annot=True’ means the values should appear on the heatmap, ‘cmap=’coolwarm’’ is for color gradient, “fmt=.2f” is for two decimal places of values.
* It helped us selecting the features such as priortizing Glucose and BMI rather than BloodPressure which is not that relevant.
* By Lessing the multicollinearity like between the SkinThickness and Insulin, helps model to perform better to unseen data.
* We can see from the mapt that the ‘Age’(0.24) feature is a positive correlation as the older people have more chances of diabetes.
* Also SkinThickness and Insulin (0.44) are related, so we can remove one.
* And BloodPressure have weak correlation just (0.07) so it is less important.

**Outlier Detection using Boxplots:**

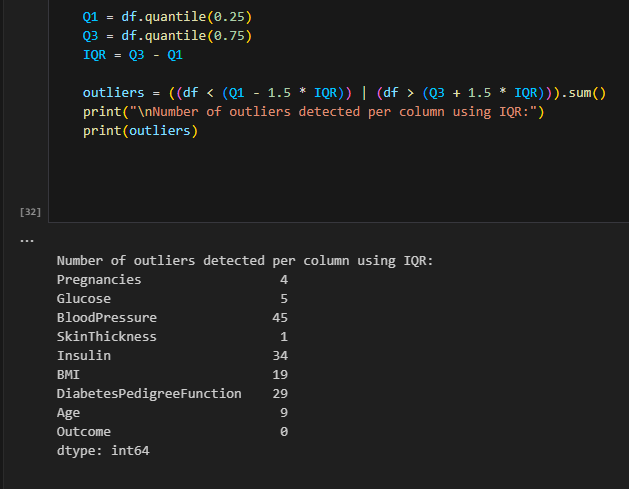
****

* The df.select\_dtypes is used to select only the features that have numerical values, i.e. int and float dtypes.
* “.plot” contains the parameters to create a box plot.

****

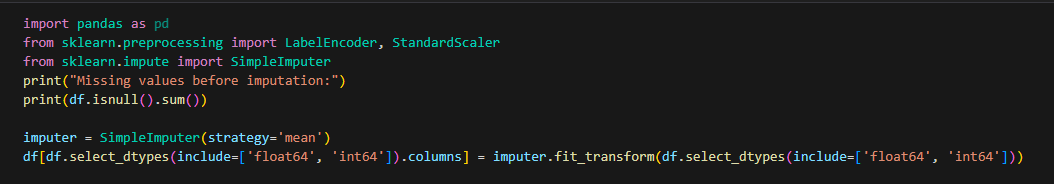
* The boxplot helps us to identify extreme values which are outliers.
* It highlights potential errors or anomalies in dataset.
* We can see in the **Glucose Feature,** there are few extreme values greater than 200, indicates severe diabetic cases. These can be outliers and should remove.
* Similarly we can see extreme values in other features like in **Blood Pressure Feature,** there is a value at 0 which is an outlier because BP can’t be 0.

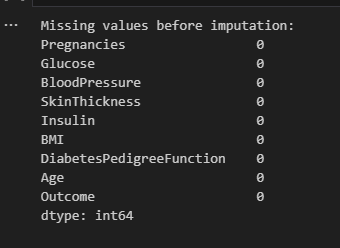
**Removing Outliers:**

****

* **Q1 and Q2**  are the 25 and 75 percent of the data.
* **IQR** Inter quartile range calculated as Q3- Q1.
* **outliers = ((df < (Q1 - 1.5 \* IQR)) | (df > (Q3 + 1.5 \* IQR))).sum()** Identifies the outliers of lower limit and upper limit.
* As we can see it provides the numerical count of outliers in each feature.
* Removing the outliers ensures the quality of the data and leads to better performance of model.

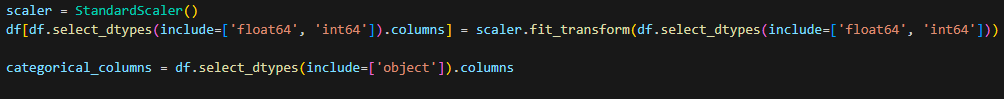
**Data Imputation:**

****

****

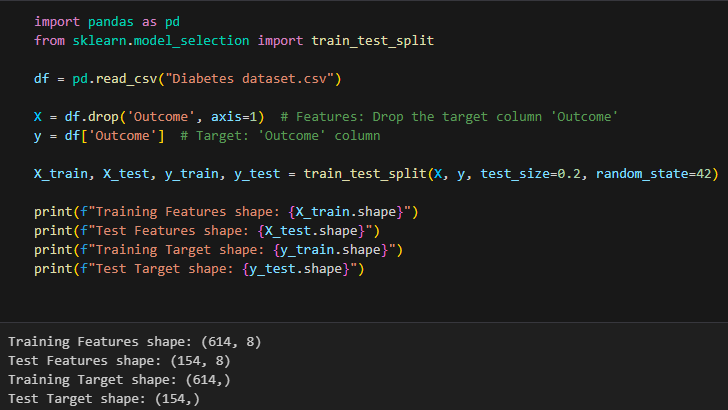
* **SimpleImputer** helps to fill missing values with any strategy such as “mean” in this case.
* **Fit\_transform** computes the mean of every column and filled with the missing ones.
* By doing this, we ensure the data completeness, hence avoiding error during model training.

**Feature Scaling:**

****

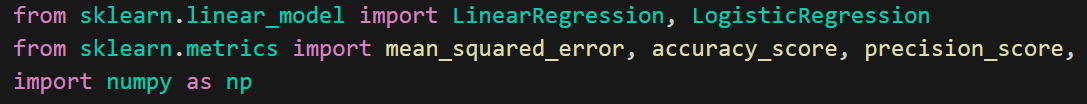
* **StandardScaler() :** Remove the mean and scale to unit variance.
* **df[df.select\_dtypes(include=['float64', 'int64']).columns] = scaler.fit\_transform(df.select\_dtypes(include=['float64', 'int64'])) :** Computes the mean of each column and standard deviation and scale it to convert into zero mean and standard deviation.
* **df.select\_dtypes(include=['object']).columns** : selecting categorical columns
* It ensures all the numerical columns have equal contributions in front of model, i.e. Glucose and Age are on the same scale.
* Note that the features like Insulin ranging from 0-846 and BMI(0-67) will now be on comparable scale preventing biaseness.
* All numerical features have now mean = 0 and std = 1 after scaling.
* Confirmed no categorical columns exist

**Splitting:**



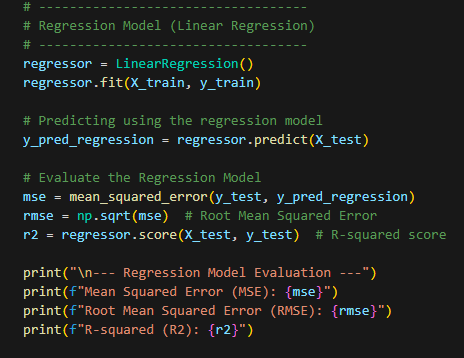
* First we import the train\_test\_split module from the sklearn library for testing and splitting.
* X contains all the input features
* y contains the outcome feature(Target column)
* train\_test\_split: Splitts the data into train test subsets, test\_size is 20%(testing data) and random\_state= 42 ensures the reporductibility by fixing the random seed.
* It will prepare the dataset for the model on supervised learning containing input features labeled as X\_train and target column labeled as y\_train on 80 by 20 ratio for training and testing.
* Lastly, the print statements are printing the shape of each test train subsets, displaying number of rows and columns.

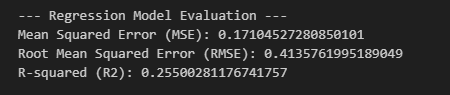
**Libraries for model training and evaluation:**

****

* We have used two algorithms which are Linear Regression and Logistic Regression.
* The Linear Regression is used to predict continuous values, the purpose to include this is just for completion, as it is not directly applicable to classification task.
* Logistic Regression is a classification algorithm used to classify the outcome i.e. resulting in binary values 0 or 1. As we have to detect whether the patient is diabetic or not, so it is important for us to use this classficiation model.
* **mean\_squared\_error** : A matric used to measure the average square difference between original values and the resulting values. It is used for comparison for regression approach.
* **accuracy\_score:** To know the accuracy of the model
* **precision\_score:** To know the precision of the model, how precisely it predicted the diabetes.
* **Recall\_score:** To know how many diabetic patients are missed by the model.
* **F1\_score:** It balances the precision and recall score.

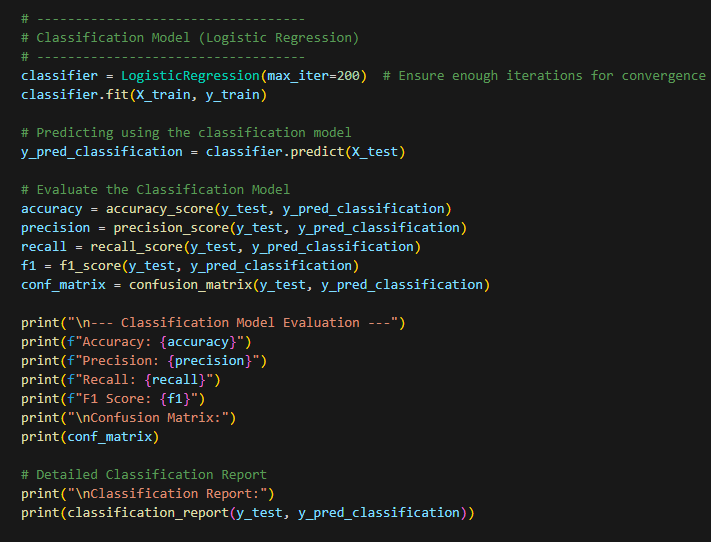
**Regression model(linear regression):**

****

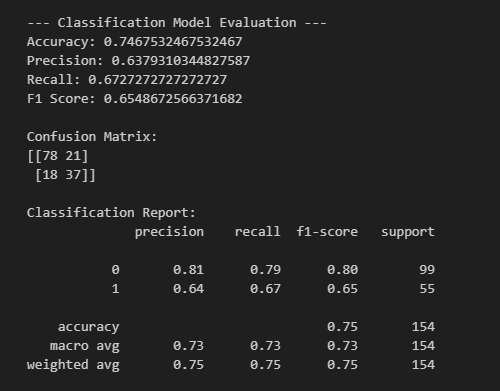
****

* **regressor.fit(X\_train, y\_train):** Trains Linear Regression model on training and testing data.
* **y\_pred\_regression = regressor.predict(X\_test):** Using the trained model to predict on target data.,
* **For evaluation;**
* “mse” computes the mean squared error. Here it is ‘0.17’, it means the model result will be ‘0.17’ units away from the actual resulting value.
* **The Root mean square value** gives the average error of the prediction, here ‘0.41’ means the error is in between 0 and 1, the model will struggle to predict the perfect value as 0 or 1.
* R^2 ranges from 0 to 1, it shows how much variation the model gets in predicting the output value, the value closer to 1 is ideal as it indicates, how much data’s patterns are captured by the model.

**Classification Model:**



* **LogisticRegression(max\_iter=200):** uses a logistic regression model with max of 200 iterations to ensure it covers everything.
* **classifier.fit(X\_train, y\_train):**  Train the classification algo on training data.
* **y\_pred\_classification = classifier.predict(X\_test):** use to predict result on X\_test data.
* Then for evaluation, as explained before, we check the accuracy, precision, recall, f1\_score and confusion matrix to check the model’s performance.
* **Confusion\_matrix:** gives a matrix of correct and incorrect predictions.
* And lastly, we have **classification report** which gives the complete information about model’s performance.



* Here we can see the accuracy of model is about 0.74 which is almost 75%. It means out of 100 diabetic patients, the model will able to detect 75 of them correctly.
* Then precision is 63%, it means when the model was predicting, it was correct 63% of the time.i.e. if it predicts 10 patients diabetic, then it means 6 are truly diabetic.
* Recall value is 67%, means the model identifies 67% patients of all actual diabetic patients.
* Then the f1\_score is 65% balances the percentage between precision and recall. It is a moderate performance by the model.

**Confusion Matrix:**

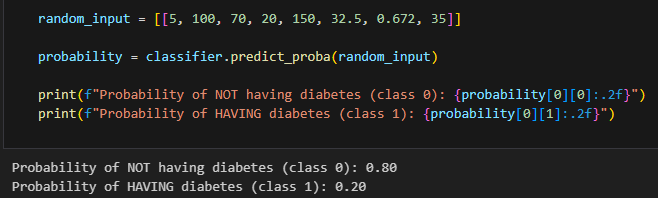
**Predicted 0 Predicted 1**

**Actual 0 78 21**

**Actual 1 18 37**

* + True Negatives (TN): 78 (correctly predicted non-diabetic).
    - False Positives (FP): 21 (non-diabetic patients wrongly flagged as diabetic).
    - False Negatives (FN): 18 (diabetic patients missed by the model).
  + True Positives (TP): 37 (correctly predicted diabetic).
  + Then the **Classficiation Report:**
  + We can see for Non Diabetic patients(0) the precision is 81% which is high. And similarly, the recall and f1 score percetange are also good, so support is 99, it means the model is reliable for detecting non-diabetic patients.
  + Whereas; in Diabetes(1), the models, performance is average. Upto 60% correct.

**Predicting probability for random values:**

****

* + - As we can see, on giving random input of patient with feature values:
    - Pragnencies=5
    - Glucose= 100
    - BP=70
    - SkinThickness=20
    - Insulin= 150
    - BMI= 32.5
    - DiabetesPedigreeFunction=0.672
    - Age= 35
    - Using predict\_proba() function to predict .

Here we can see the probability of having diabetes is 20% and not having diabetes is 80%.

* + - Based on the glucose and BMI values which are 100(which is less than 126) and 32.5(indicates obesity a risk factor but just about 32%), these values are moderate

range which indicates that patient should have less risk factor to have diabetes. So we can say that model predicted correctly.