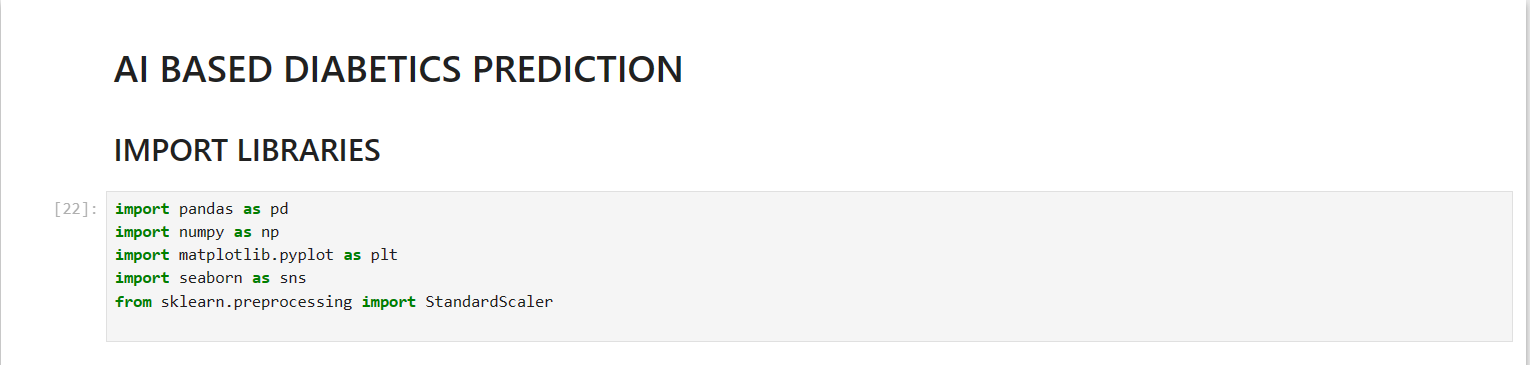
**PHASE – 3**

**DEVELOPMENT – 1**

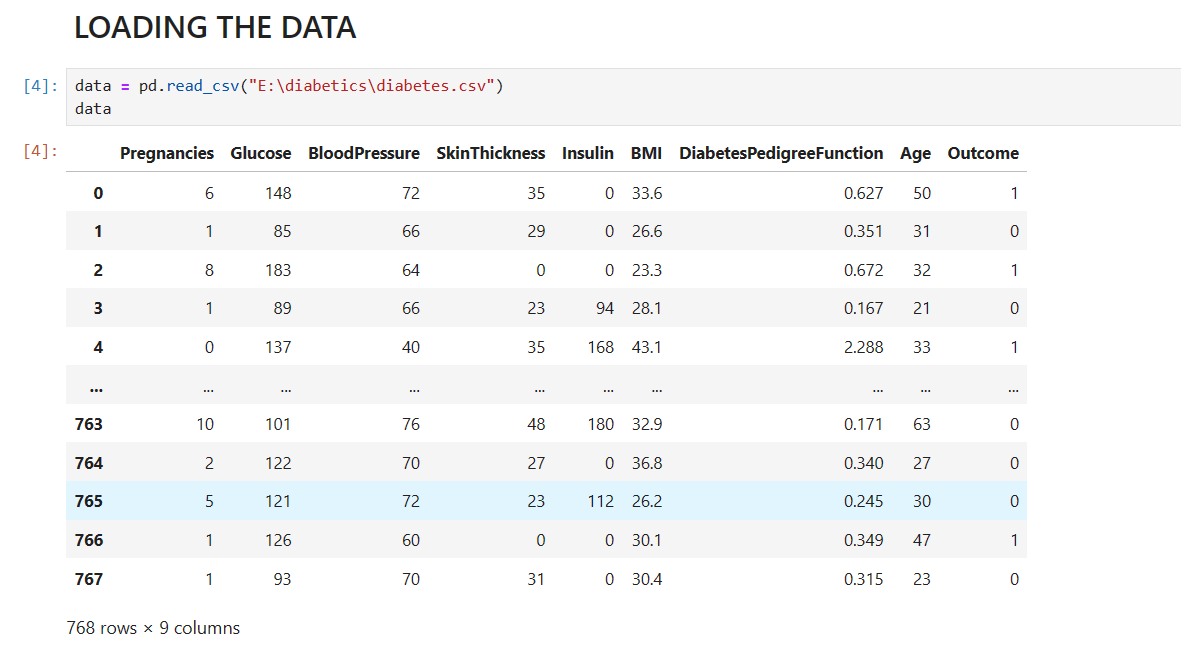
**IMPORTING LIBRARIES:**

Start by importing the necessary Python libraries, including NumPy, Pandas, and Scikit-Learn for data manipulation and machine learning.



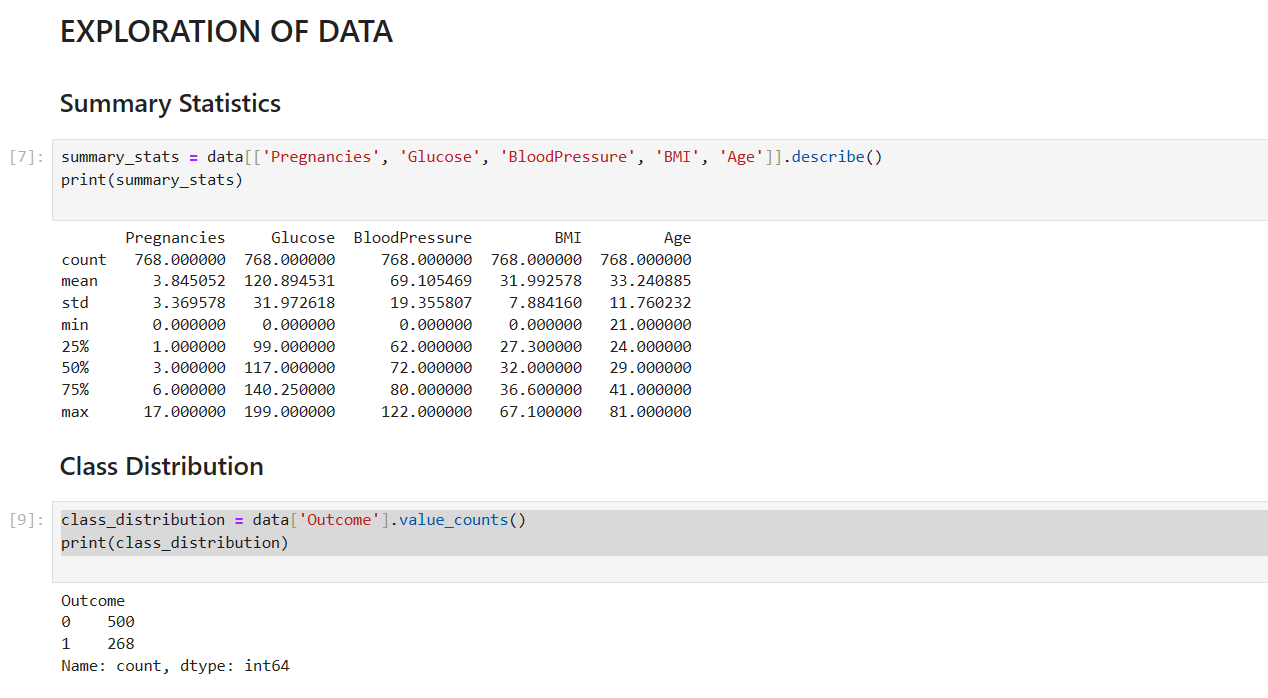
**LOADING THE DATA:**

Load the dataset into a Pandas Data Frame.



**EXPLORATION OF DATASET:**

Examine the dataset to understand its structure and the type of data it contains. This can include checking for missing values, data types, and statistical summaries.



**HANDLING MISSING VALUES:**

Check for missing values and decide how to handle them. You can either impute them with mean, median, or zero or remove rows with missing values.

A screenshot of a computer

Description automatically generated

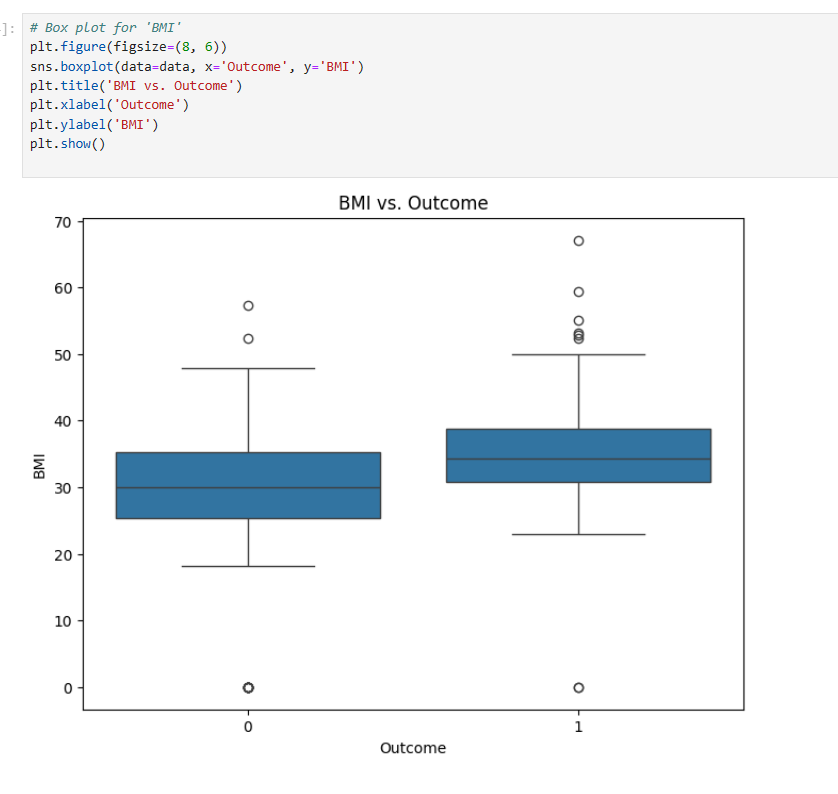
**DATA VISUALIZATION:**

The provided code generates two plots for data visualization. The first is a histogram showing the distribution of 'Glucose' levels, providing insights into the frequency of different glucose values. The second is a box plot illustrating the relationship between 'BMI' and 'Outcome,' allowing for a quick comparison of 'BMI' distributions across different outcome categories. These visualizations aid in understanding the dataset's characteristics and variable relationships.

A screenshot of a graph

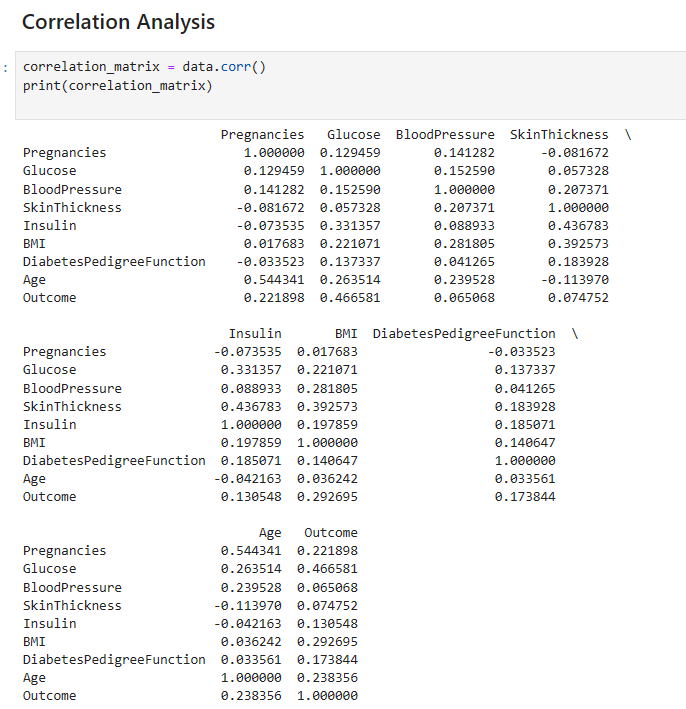
Description automatically generated

**BMI VS OUTCOME:**



**CORRELATION ANALYSIS:**

This code begins by calculating and displaying a correlation matrix for a dataset, showcasing the relationships between its numerical features using a heatmap. It then narrows its focus to investigate the connection between 'Pregnancies' and 'Age' with a scatter plot, allowing for a closer examination of how these two variables relate to one another. This process of data exploration provides valuable insights into the dataset's feature interactions and specific relationships between variables, aiding in understanding the data's characteristics and potential patterns.



A screenshot of a computer screen

Description automatically generated

**FEATURE RELATIONSHIP:**

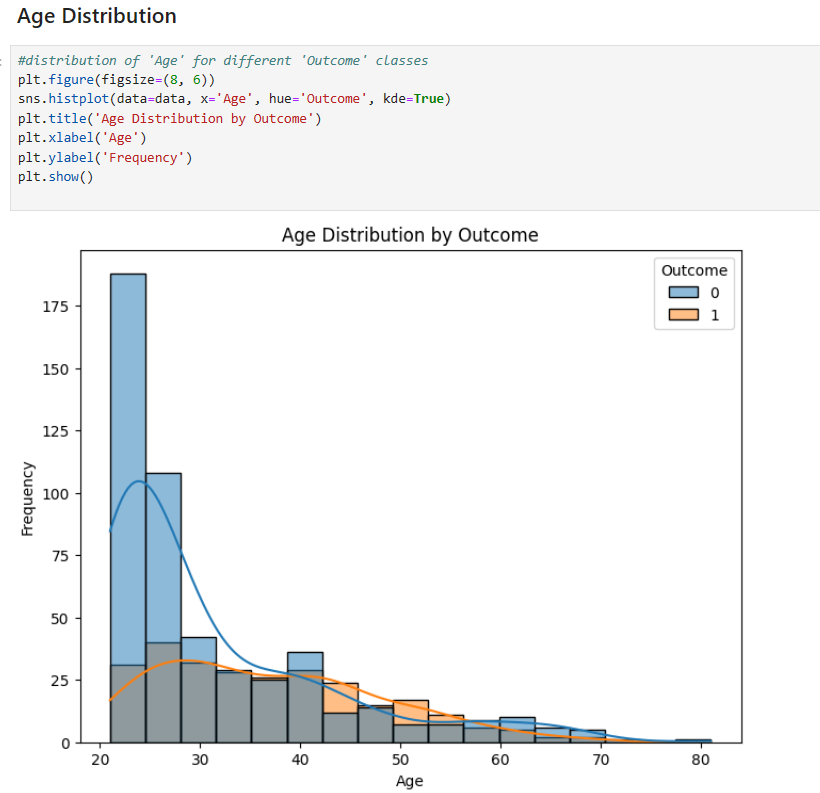
This code creates a scatter plot to visualize the connection between 'Pregnancies' and 'Age' in a dataset. The plot shows individual data points as dots, with 'Pregnancies' on the x-axis and 'Age' on the y-axis. By examining this plot, one can gain insights into how these two variables relate to each other. It helps to identify any patterns, trends, or correlations between the number of pregnancies and a person's age. This kind of visualization is a common exploratory data analysis technique, enabling a quick and intuitive understanding of the relationship between specific features within the dataset

A screen shot of a graph

Description automatically generated

**AGE DISTRIBUTION:**

This code generates a histogram with two different colored distributions to visualize the 'Age' data in relation to different 'Outcome' classes. The plot displays the frequency of 'Age' values on the x-axis, with distinct colors representing different 'Outcome' categories. It provides a quick and informative overview of how the 'Age' variable is distributed within each 'Outcome' group. By using this visualization, you can easily compare and contrast the age distributions for various outcome classes, helping to identify potential age-related patterns or differences within the dataset. This type of plot is valuable in exploring the impact of 'Outcome' on the distribution of 'Age' and understanding any potential trends or relationships.



**VISUALIZING NUMERICAL FEATURES:**

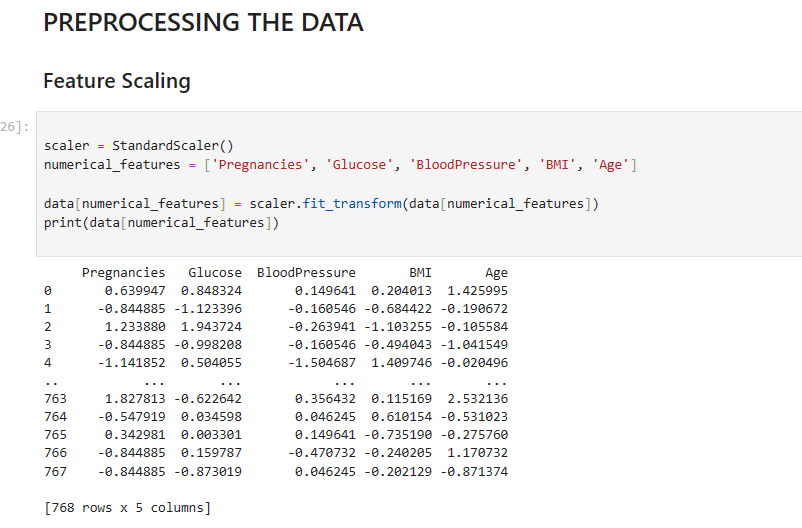
This code snippet creates histograms for standardized numerical features within a dataset. It iterates through a list of numerical features, visualizing the distribution of each feature using Seaborn's histplot. The histograms display the frequency of each standardized feature, along with a Kernel Density Estimation (KDE) plot for a smoothed density curve. It is a helpful way to visualize how data is distributed after standardization, which often involves scaling features to have a mean of 0 and a standard deviation of 1. Additionally, the code saves the preprocessed data, including the standardized features, to a new CSV file named 'preprocessed\_diabetes\_data.csv' in the specified directory 'E:/diabetics'. Finally, it prints the preprocessed data to the console. This code is useful for both visualizing the effects of standardization on data distribution and preserving the preprocessed dataset for further analysis.

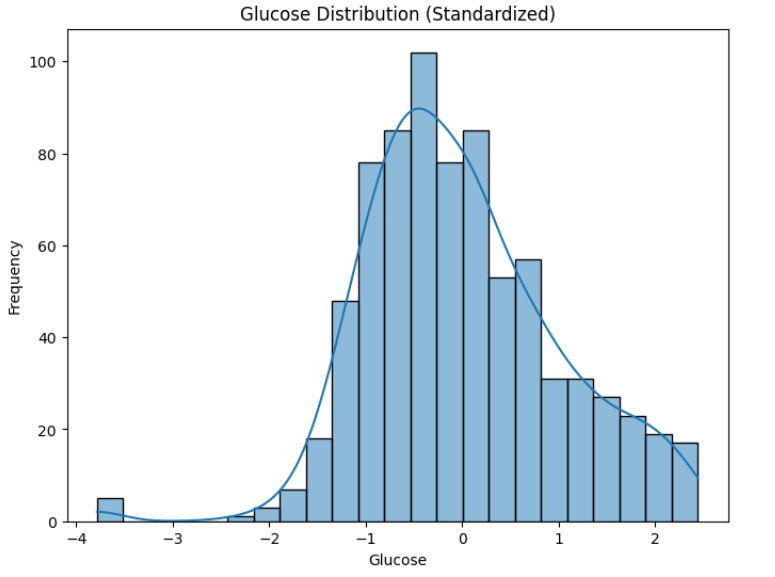
A screen shot of a graph

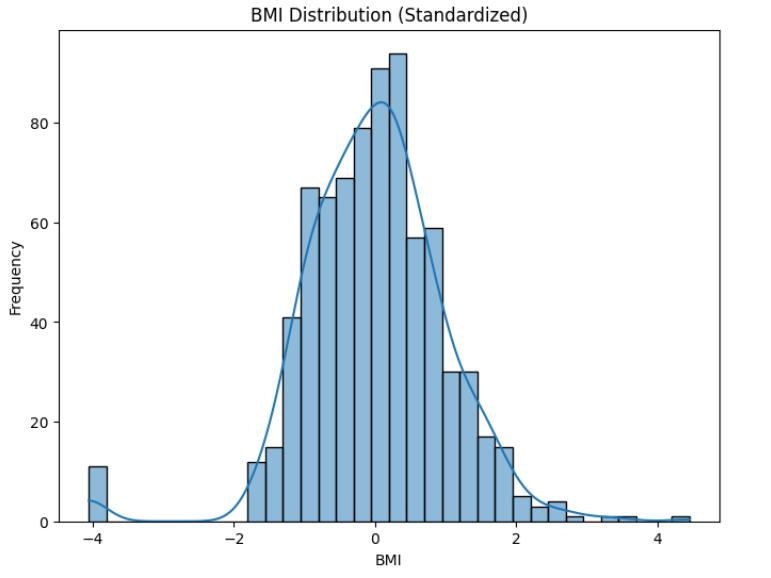
Description automatically generated

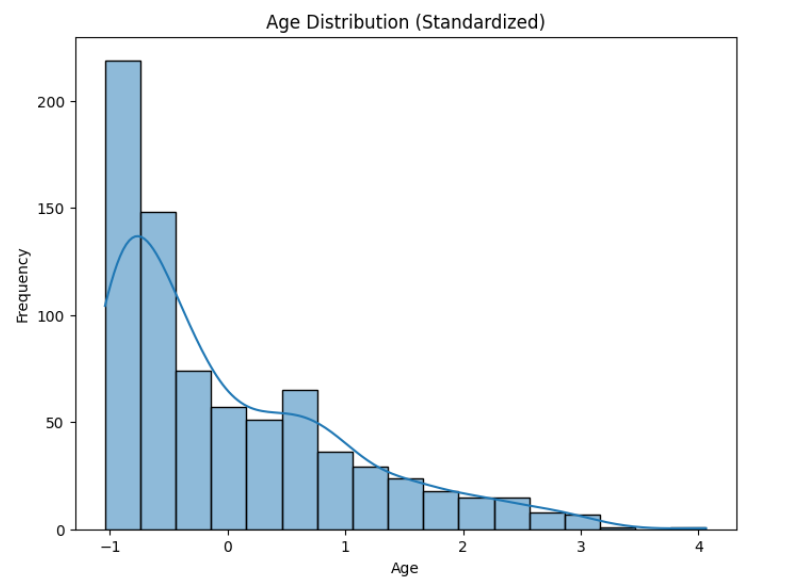
**PROCESSING THE DATA:**

This code snippet standardizes a set of numerical features, including 'Pregnancies,' 'Glucose,' 'BloodPressure,' 'BMI,' and 'Age,' using the **StandardScaler** from scikit-learn. Standardization rescales these features to have a mean of 0 and a standard deviation of 1, ensuring they are on a consistent scale for machine learning. The code then prints the standardized features, preparing the data for modeling by enhancing model stability and performance through consistent feature scaling.









**SAVE PREPROCESSED DATA:**

This code saves the preprocessed dataset to a CSV file and then prints the dataset to the console for verification. It helps ensure that the preprocessing is correctly applied, and it stores the processed data in a file for future use.

