**Project Report**

**Problem Statement**

A binary classification problem: predicting whether an individual is at high risk or low risk of developing diabetes based on a range of medical data. This system offers a powerful tool for early risk assessment and targeted healthcare interventions.

**Design Thinking**

Here's a simplified design thinking approach for the problem.

**Understand the Problem:**

Start by gaining a deep understanding of the problem you want to solve. This involves talking to domain experts and potential users to identify their needs and expectations.

**Data Collection and Cleaning:**

Gather relevant data on diabetes patients, ensuring it's of high quality. This includes removing any missing or inconsistent data.

**Feature Selection and Engineering:**

Identify the most important factors (features) that can predict diabetes. Create new features if needed to capture more insights from the data.

**Model Selection:**

Choose suitable machine learning algorithms for the task. Consider which algorithms are likely to perform well given the data and the problem's nature.

**Model Training:**

Split the data into training and validation sets to train your models. Optimize the model's parameters and evaluate its performance during this phase.

**Model Evaluation:**

Assess how well the model is doing using metrics like accuracy, precision, recall, and F1-score. Understand its strengths and weaknesses. Visualize the model's performance with tools like confusion matrices.

**Phases of Development**

* Data Preparation:
  + Collect relevant, high-quality data on diabetes patients.
  + Clean the data by removing missing or inconsistent values to ensure data quality.
* Feature Selection and Model Development:
  + Identify key predictive factors (features) for diabetes.
  + Create new features if necessary to gain deeper insights from the data.
  + Choose suitable machine learning algorithms that fit the problem and data.
  + Split the data into training and validation sets for model training.
  + Optimize model parameters and evaluate their performance.
* Model Evaluation and Validation:
  + Assess the model's performance using metrics like accuracy, precision, recall, and F1-score.
  + Understand the model's strengths and weaknesses.
  + Visualize its performance using tools like confusion matrices

**Dataset Used**

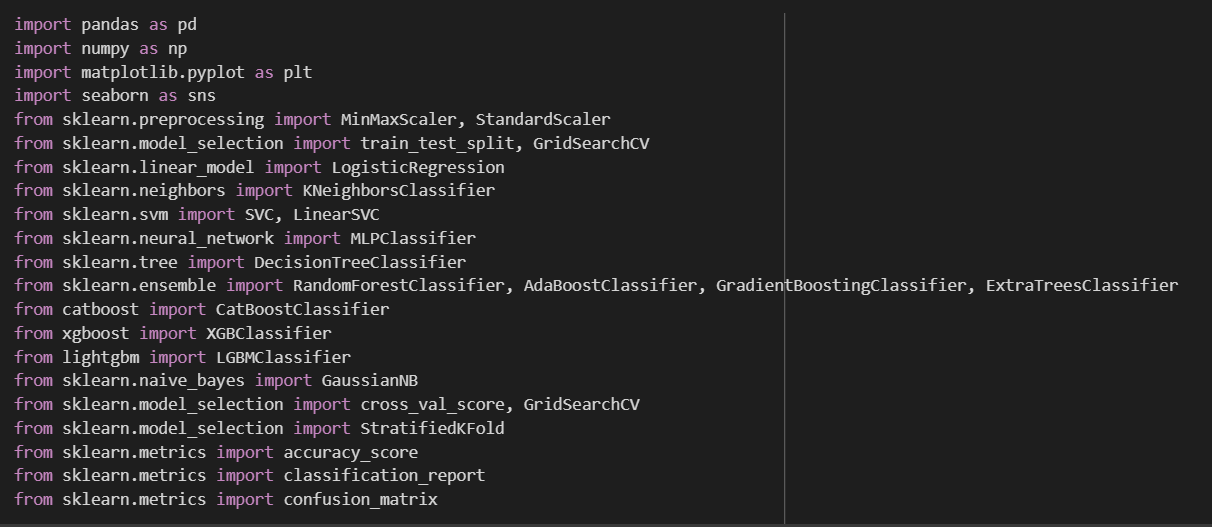
The dataset used for this project is derived from a larger database and consists of medical data about female individuals of Pima Indian heritage who are at least 21 years old. The following attributes are included:

* **Pregnancies**: Number of times pregnant
* **Glucose**: Plasma glucose concentration at 2 hours in an oral glucose tolerance test
* **BloodPressure**: Diastolic blood pressure (mm Hg)
* **SkinThickness**: Triceps skin fold thickness (mm)
* **Insulin**: 2-Hour serum insulin (mu U/ml)
* **BMI**: Body mass index (weight in kg/(height in m)^2)
* **DiabetesPedigreeFunction**: Diabetes pedigree function
* **Age**: Age (years)
* **Outcome**: Class variable (0 or 1)

These attributes provide a comprehensive view of the individual's health profile and serve as the basis for predicting diabetes risk.

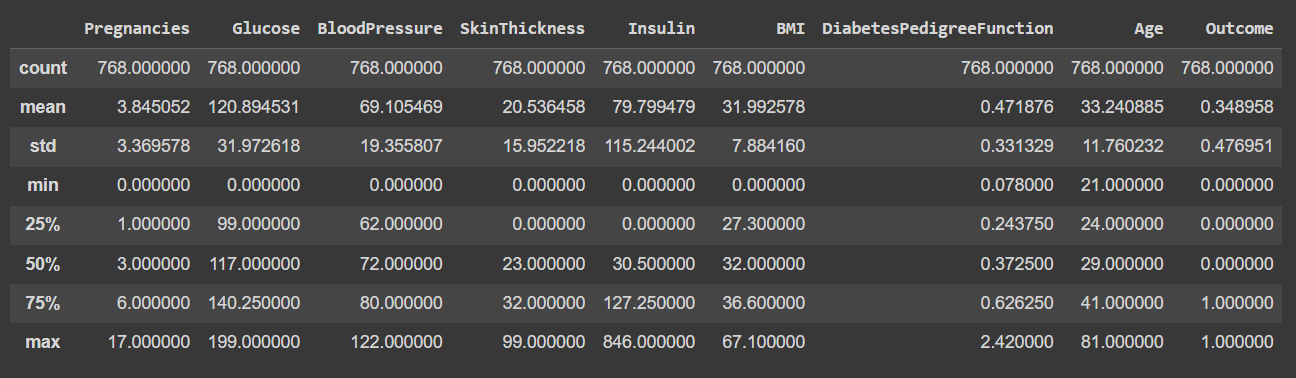
**Dataset Link:** [**https://www.kaggle.com/datasets/mathchi/diabetes-data-set**](https://www.kaggle.com/datasets/mathchi/diabetes-data-set)

**Importing Required Libraries**

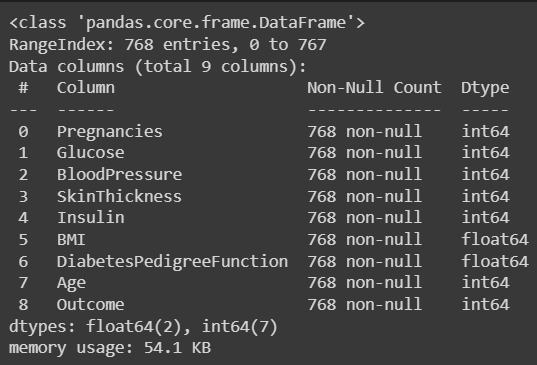
****

**Data Preprocessing**

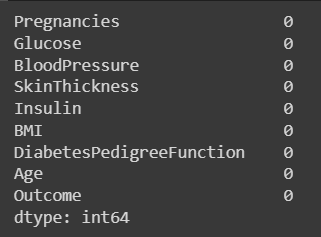
* **Descriptive Statistics (df.describe()):**
  + Descriptive statistics provide an overview of the dataset's numerical attributes. This includes measures such as mean, standard deviation, minimum, maximum, and quartiles for each numeric feature. It helps us understand the central tendency and dispersion of the data. This information can be used to identify outliers, assess data distributions, and gain initial insights into feature importance.



* **Data Information (df.info()):**
  + The "df.info()" method provides a concise summary of the dataset, including the number of non-null entries, the data types of each feature, and the total memory usage. This information is crucial for understanding the completeness of the dataset and for making decisions on data type conversions if necessary.



* **Missing Data Analysis (df.isnull().sum()):**
  + Identifying missing data is vital in data preprocessing. The "df.isnull().sum()" operation is used to count the number of missing values for each feature. This count helps us understand the extent of missing data in the dataset, which can guide decisions on data imputation or, in some cases, the removal of incomplete records.



**Data Visualization**

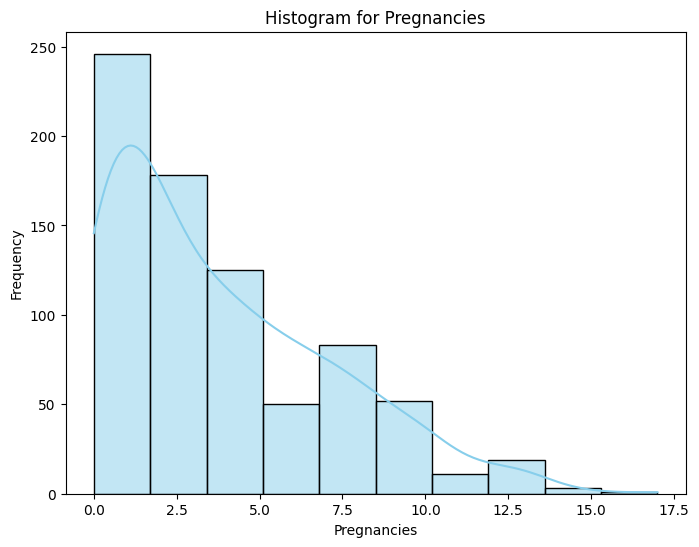
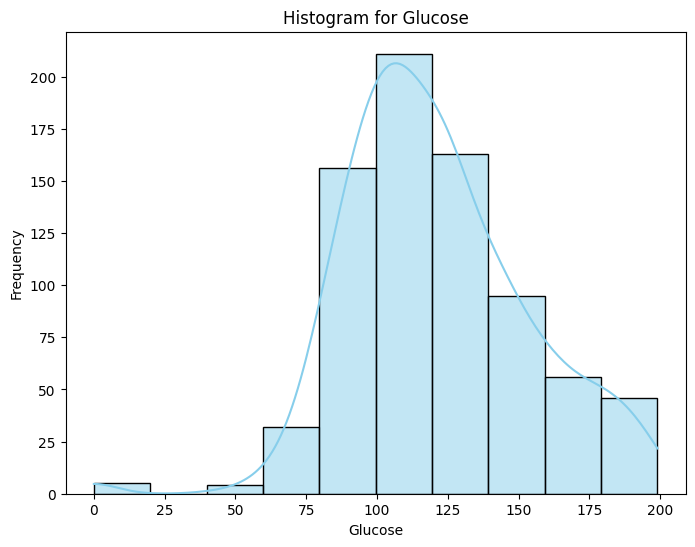
Data visualization is a crucial step in understanding the distribution of variables and the relationship between features.

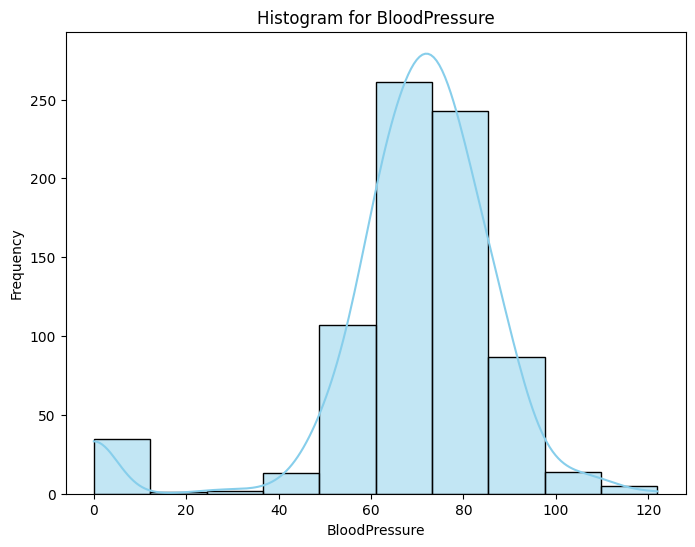
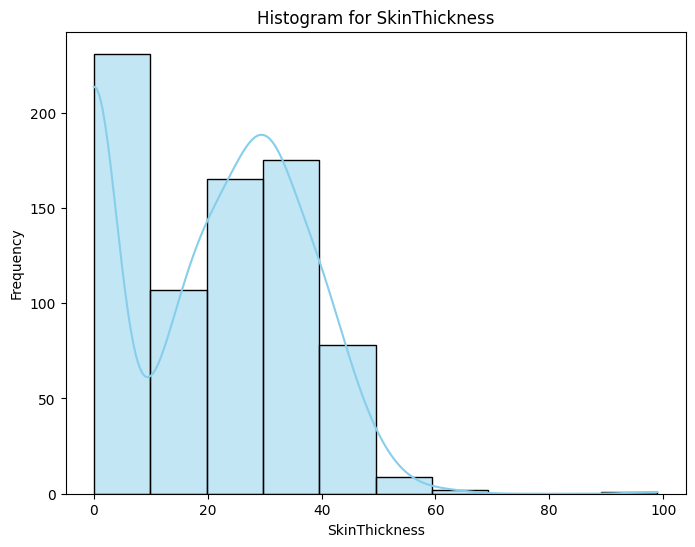
**Numeric Column Distributions:**

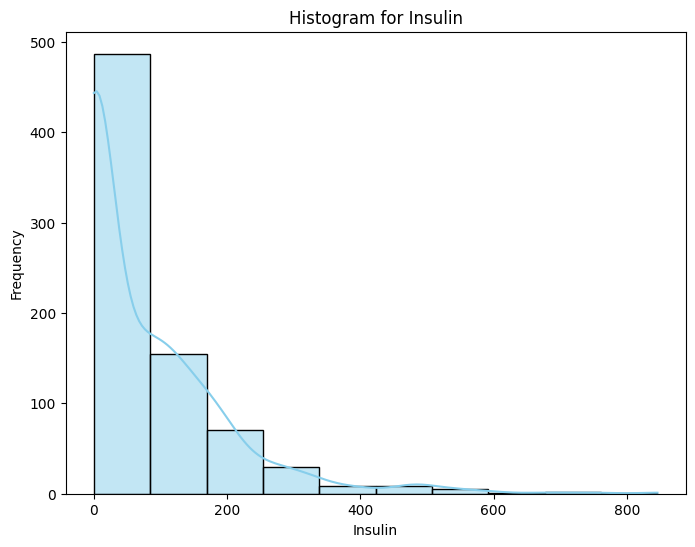
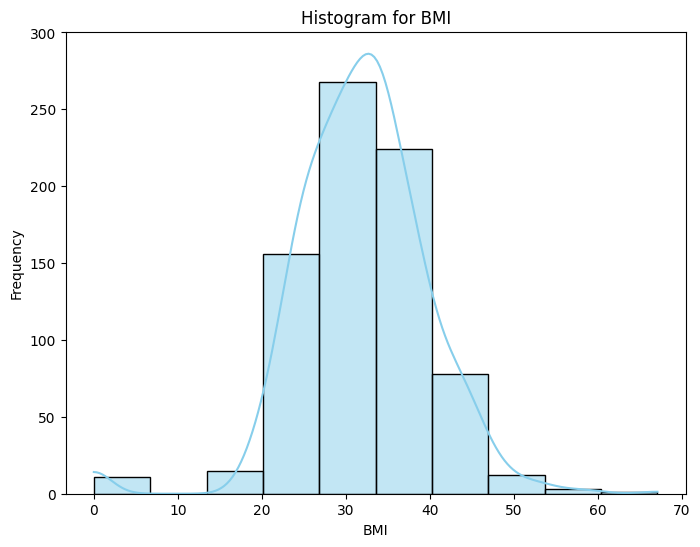
We begin by examining the distributions of numeric columns. For each numeric feature, a histogram is plotted to visualize the frequency distribution. Histograms provide insights into the central tendencies and spread of the data.

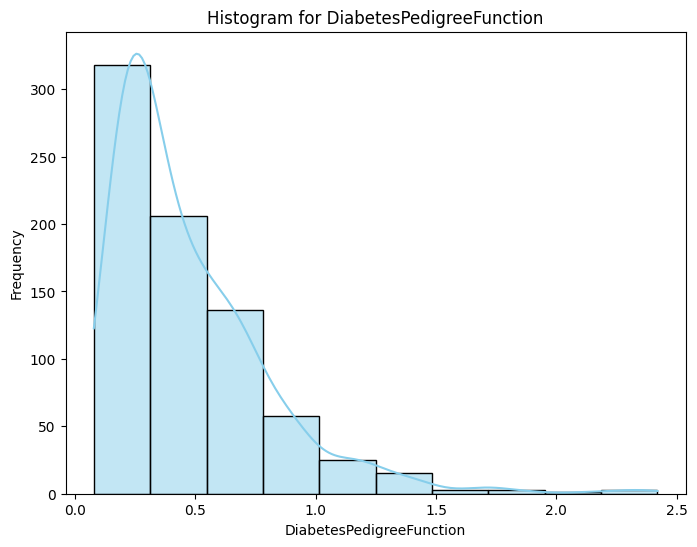
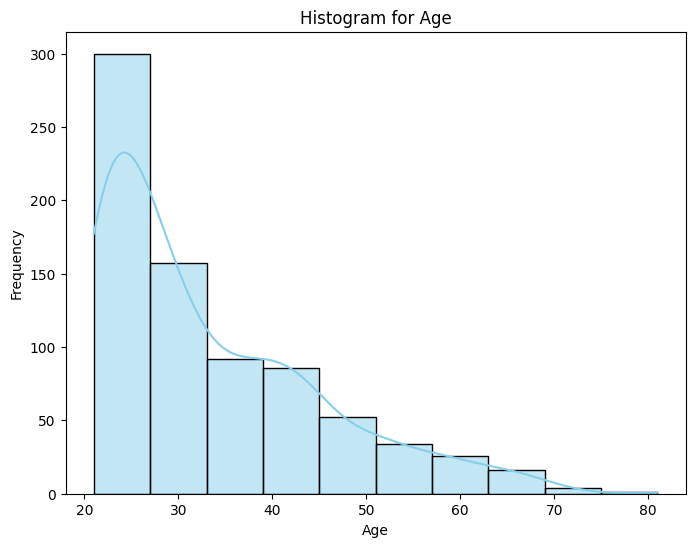
Histograms for the numeric columns:

* Pregnancies: The histogram for the number of pregnancies shows a skewed distribution, with the majority of individuals having fewer pregnancies.
* Glucose: The glucose level histogram exhibits a relatively normal distribution, which is a positive sign for modeling.
* BloodPressure: Blood pressure values appear to follow a somewhat normal distribution.
* SkinThickness: Skin thickness data is right-skewed, with many individuals having lower values.
* Insulin: The insulin level histogram also reveals right-skewed data.
* BMI: Body mass index (BMI) values are somewhat normally distributed.
* DiabetesPedigreeFunction: The diabetes pedigree function values exhibit a skewed distribution.
* Age: The age histogram shows the age distribution in the dataset, with a peak in the middle age range.

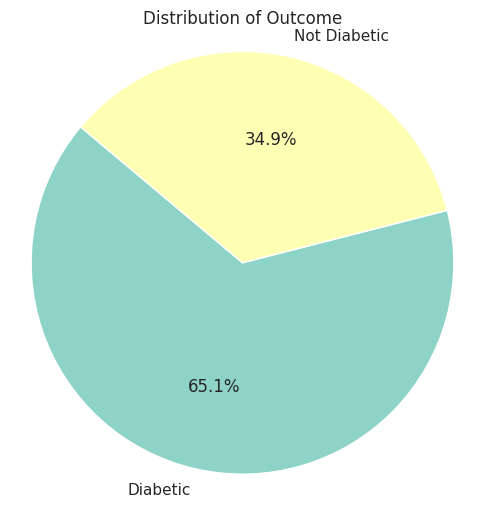
 

**Categorical Column Distribution:**

We also examine the distribution of the target variable, "Outcome," which indicates whether an individual is diabetic or not. This binary outcome variable is visualized using a pie chart.

Summary of the distribution:

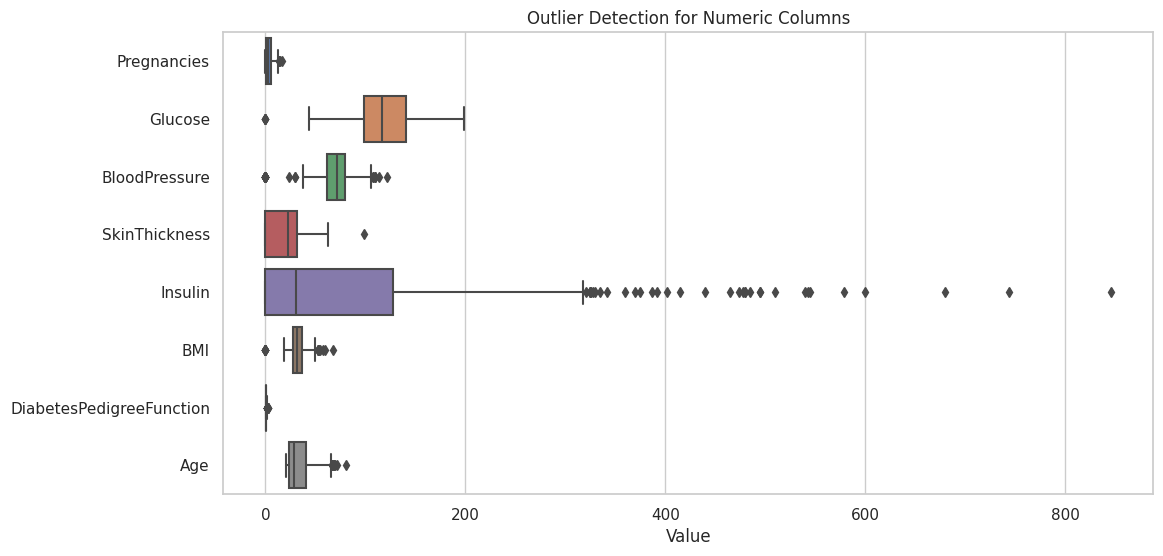
* Outcome Distribution: The pie chart depicts the distribution of outcomes, showing the proportion of individuals who are diabetic and those who are not. It's important to note that the dataset may have class imbalance, which can impact model training.

****

**Boxplot for Outlier Detection:**

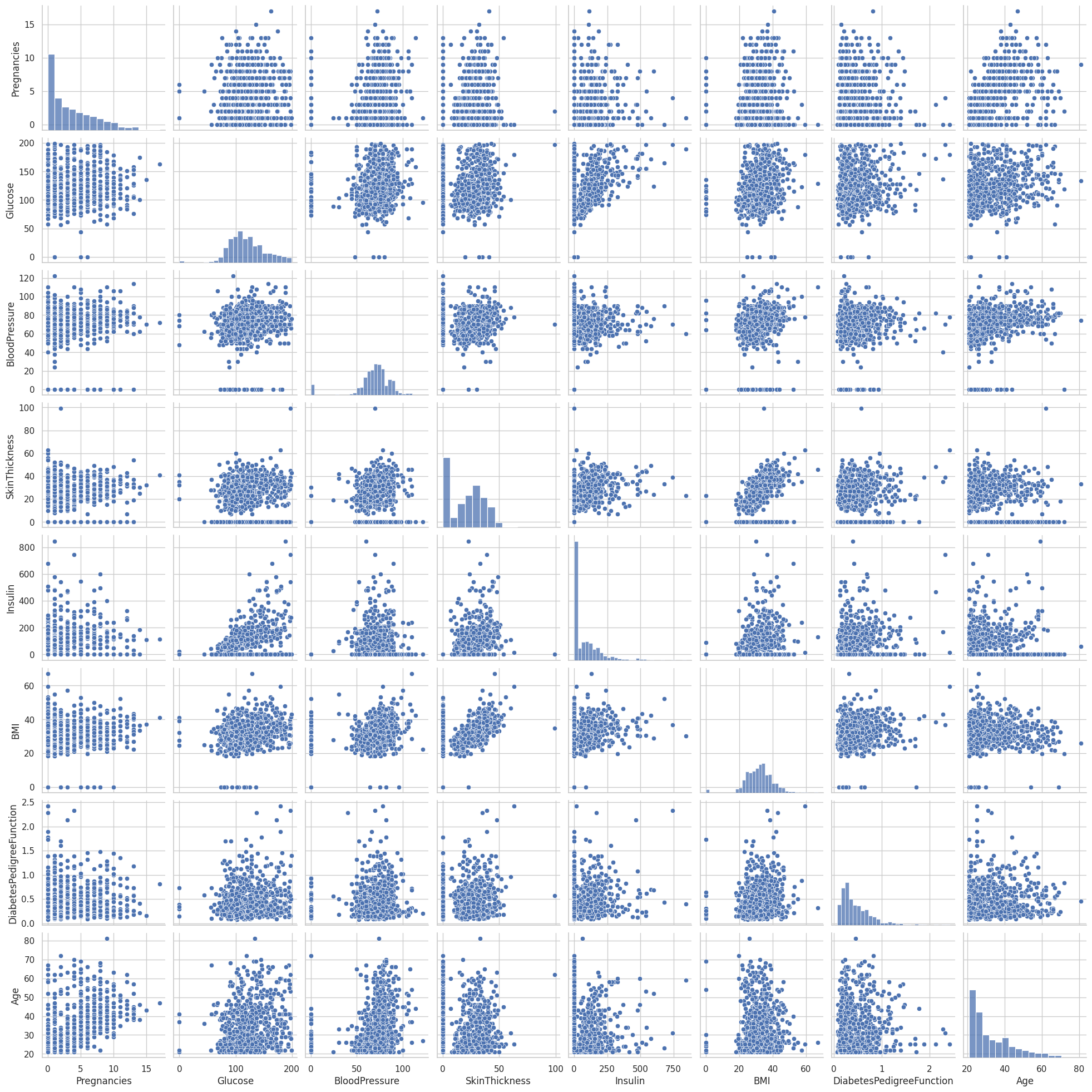
We also utilize boxplots to visually assess the presence of outliers in the numeric columns. Boxplots provide a clear representation of a feature's distribution, including measures such as the median, quartiles, and potential outliers. By examining the spread of data, we can identify extreme values that deviate from the typical distribution, indicating potential outliers.

The boxplot for each numeric column is presented horizontally, allowing us to observe the range of values and the presence of outliers more easily. Identifying and handling outliers is crucial for ensuring the accuracy and robustness of machine learning models.



**Pairplot Analysis:**

We start by creating a pairplot, which is a grid of scatterplots that allows us to visualize the pairwise relationships between numeric columns. The pairplot provides insights into potential correlations or patterns in the data. From the pairplot, we can observe how different numeric features relate to each other. This can help in identifying linear or nonlinear relationships and potential outliers.



**Preprocessing Data**

Feature scaling is an essential step in data preprocessing to ensure that all numeric features are on a consistent scale. In this report, we discuss the use of two common scaling techniques: Min-Max scaling and standardization.

**Min-Max Scaling:**

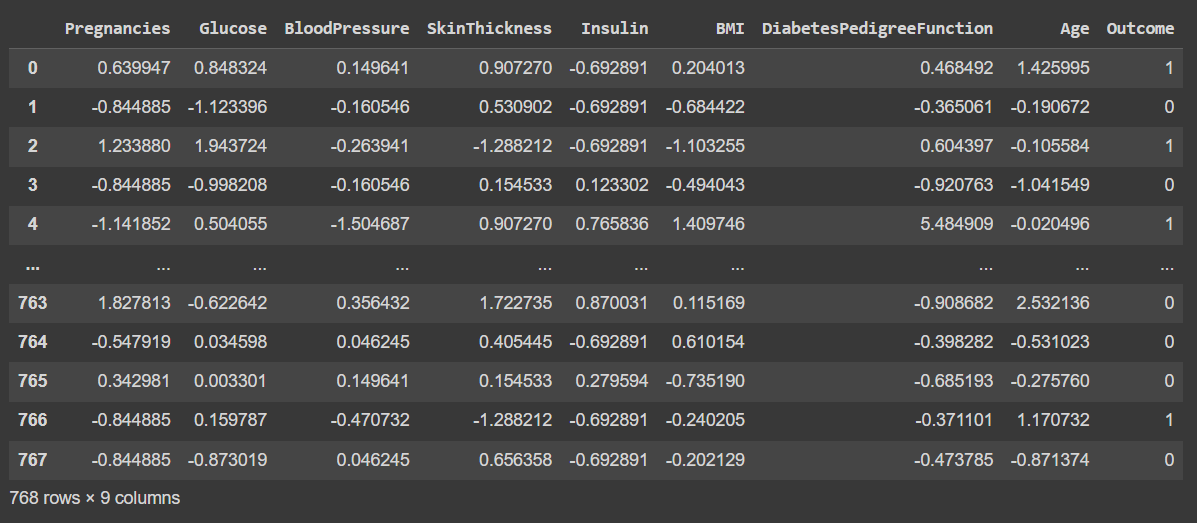
Min-Max scaling, also known as normalization, rescales the values of numeric features to fall within a specified range, typically between 0 and 1. It's achieved by subtracting the minimum value from the feature and then dividing by the range (the maximum value minus the minimum value). This technique is particularly useful when preserving the interpretability of features within a bounded range is important.

In our preprocessing steps, we applied Min-Max scaling to the numeric columns in the dataset using the MinMaxScaler. This process transformed the data, ensuring that all features are within the [0, 1] range. This can be especially useful when working with algorithms that are sensitive to the scale of input features.

**Standardization:**

Standardization, on the other hand, transforms the values of numeric features to have a mean of 0 and a standard deviation of 1. It's achieved by subtracting the mean value from each data point and then dividing by the standard deviation. Standardization helps in making data more Gaussian-like, which can be beneficial for certain algorithms, such as those relying on distances or gradient descent.

Similarly, in our data preprocessing, we applied standardization to the numeric columns using the StandardScaler. This process centered the data around a mean of 0 and scaled it to have a standard deviation of 1, ensuring that all features have a common scale.

****

**Impact on the Dataset:**

After applying Min-Max scaling and standardization to the numeric columns, the data is now consistent in terms of scale. This prepares the dataset for further analysis, as machine learning algorithms often perform better when working with features that have similar scales. It reduces the risk of any feature dominating the learning process due to differences in scales.

These scaling techniques are fundamental in enhancing the effectiveness and performance of machine learning models when applied to this dataset.

**Model Selection and Evaluation**

We’ll discuss the selection of machine learning models and their evaluation for predicting diabetes using the DiabeTrack+ system.

**Data Splitting:**

Before we proceed with model selection and evaluation, the dataset was split into training and testing sets. The training set, which consists of 70% of the data, was used to train the models, while the testing set (30%) was reserved for evaluating their performance. A random seed was set to ensure reproducibility.

**Model Candidates:**

A variety of machine learning models were considered for the task, including:

* Logistic Regression
* Random Forest
* Support Vector Machines (SVM)
* K-Nearest Neighbors
* Naive Bayes
* Neural Network
* Decision Tree
* Gradient Boosting
* AdaBoost
* XGBoost
* LightGBM
* CatBoost
* Extra Trees

**Cross-Validation:**

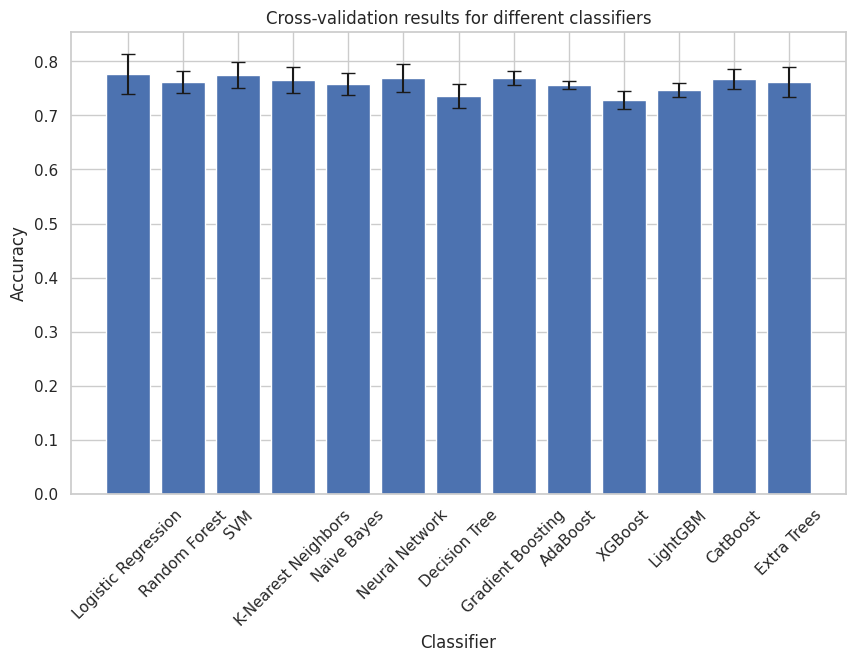
To assess the performance of these models and to prevent overfitting, k-fold cross-validation was employed with k = 4. Stratified k-fold was chosen to ensure that each fold maintains the class distribution. The 'accuracy' metric was used to evaluate the models during cross-validation.

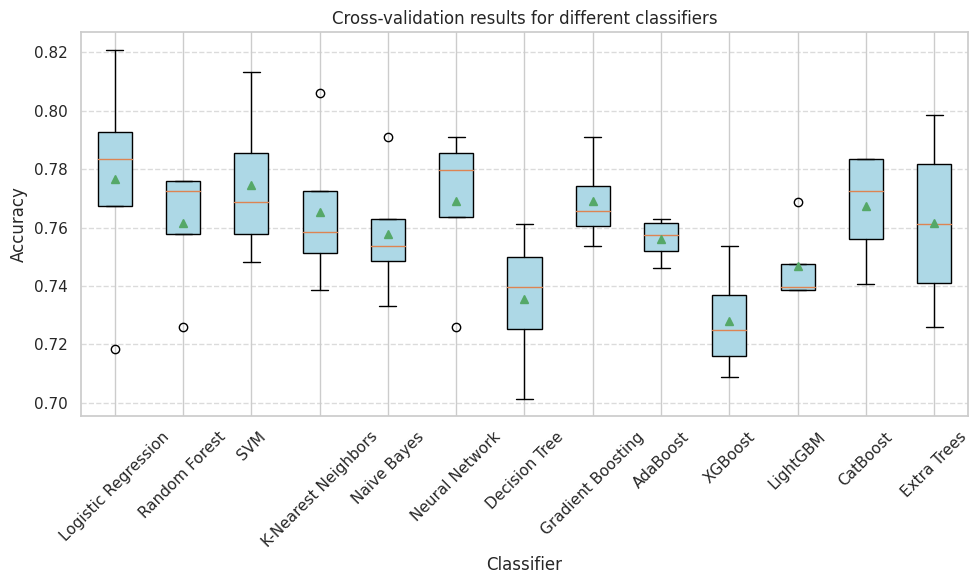
The cross-validation results provide insights into how well each model performs on the training data. This step helps in understanding the potential accuracy and variance of each model.

**Visualizing Cross-Validation Results:**

Two visualizations were created to summarize the cross-validation results:

* A bar chart displaying the mean accuracy and its standard deviation for each model.
* A boxplot showcasing the distribution of accuracy scores for each model, including median accuracy.

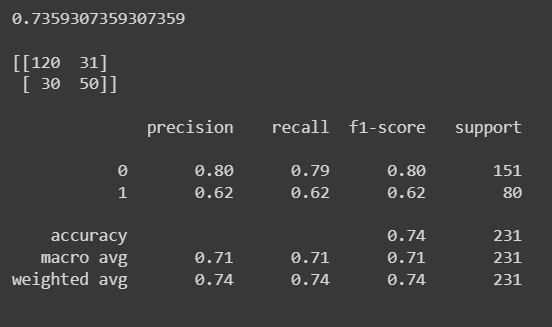




**Model Training and Testing:**

After evaluating the models with cross-validation, a Logistic Regression model was chosen for training and testing. The Logistic Regression model was fitted to the training data and used to predict outcomes on the test data. The following evaluation metrics were calculated:

* Accuracy Score: Measures the proportion of correctly predicted outcomes.
* Confusion Matrix: Provides insight into true positives, true negatives, false positives, and false negatives.
* Classification Report: Offers a detailed breakdown of precision, recall, F1-score, and support for both classes (diabetic and non-diabetic).
* The accuracy score and the confusion matrix help in understanding how well the chosen model performs on the test data and in identifying its strengths and limitations.

****

**Conclusion**

This section concludes the model selection and evaluation phase, showcasing the performance of various models, the chosen model for the DiabeTrack+ system, and the model's performance on the test data. It provides a foundation for further model fine-tuning and deployment.