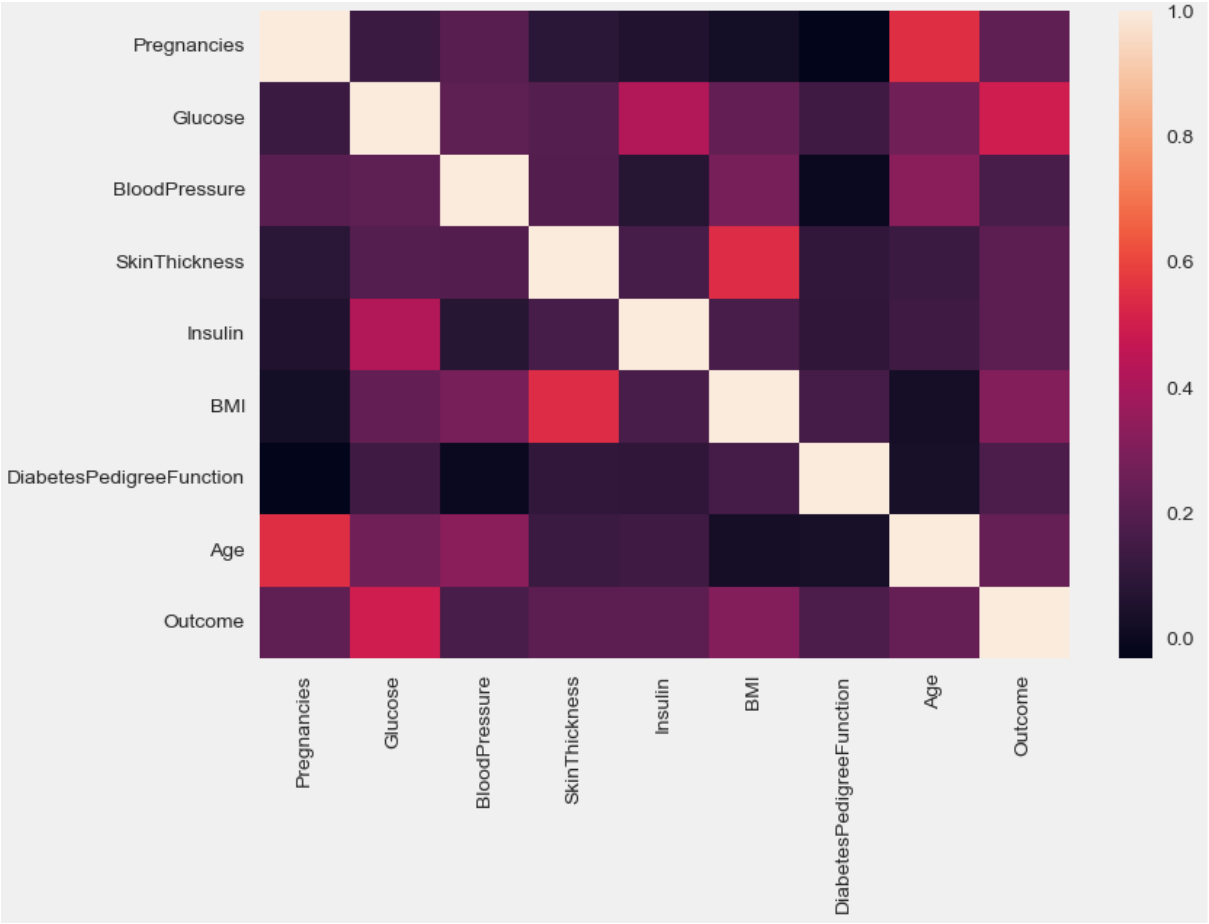
Diabetes

https://github.com/yasking68/Maha-ML-Project/tree/main/Project%20MAHA

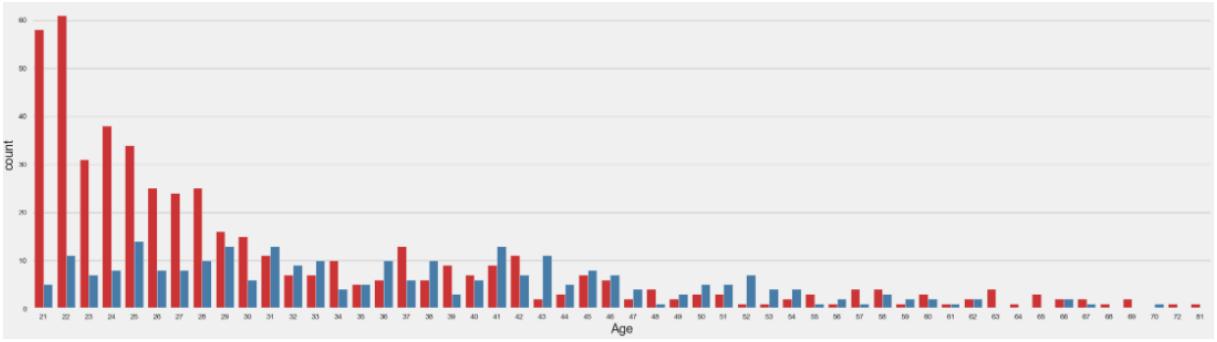
The dataset comprises various medical predictor variables and one target variable, "Outcome". Independent variables encompass factors such as the number of pregnancies, BMI, insulin levels, age, and more. Columns include details like the number of pregnancies experienced by the patient, plasma glucose concentration after an oral glucose tolerance test, diastolic blood pressure, triceps skin fold thickness, serum insulin level, body mass index, diabetes mellitus history in relatives, patient age, and the outcome variable indicating diabetes presence or absence. Specifically, 268 instances are labeled as diabetes-positive (1), while the remainder are diabetes-negative (0). The task at hand is to construct a machine learning model capable of accurately predicting whether patients within the dataset have diabetes or not.

Heat map:



From the heatmap, several insights can be gleaned. There is a noticeable strong correlation between Glucose and Outcome, indicating that plasma glucose concentration is closely related to diabetes presence. Additionally, BMI and SkinThickness exhibit significant correlation, suggesting a relationship between body mass index and triceps skin fold thickness. On the other hand, BloodPressure appears to have weaker correlations with other

features, indicating less influence on diabetes prediction. This visualization aids in understanding the interrelationships between different variables in the dataset, providing valuable insights for further analysis and modeling.



I started by visualizing the data with a bar plot, which depicted the count of different age values, likely segmented by some categorical feature like outcome.

Then, I divided the dataset into training and testing sets using the train_test_split method, assigning 80% of the data for training and reserving 20% for testing.

Following this, I standardized the features using StandardScaler to remove the mean and scale them to unit variance.

Next, I initialized a KNN classifier with 11 neighbors, opting for the Euclidean distance metric, and trained it on the training data.

Subsequently, I made predictions on the test set using the trained model.

For evaluation purposes, I employed a confusion matrix, F1 score, and accuracy score. The confusion matrix revealed 32 true positives, 94 true negatives, 13 false positives, and 15 false negatives.

The F1 score, representing the balance between precision and recall, came out to be 0.696.

Regarding accuracy, the model achieved a score of 0.818, implying that approximately 81.8% of the predictions were correct.

In conclusion, while the KNN classifier showed decent performance, there's potential for improvement, especially in reducing false positives and false negatives. Further exploration, such as parameter tuning or experimenting with alternative algorithms, could lead to enhanced predictive accuracy.