**Final Project**

**Course: CSIS 260 - Introduction to Artificial Intelligence**

**Prepared by: Jad Zaiter, Hassan Mroueh**

**Presented to:**

**Dr. Michel Chammas**

**May 24, 2024**

**Contents**

1. [Abstract 2](#_TOC_250007)
2. Introduction 2
3. [Phase 1 (Midterm) 2](#_TOC_250006)
4. [Methodology 2](#_TOC_250005)
   1. [Exploratory Data Analysis 2](#_TOC_250004)
   2. [Data pre-processing 3](#_TOC_250003)
   3. [Testing the 4 Models 4](#_TOC_250002)
5. [Phase 2 (Final) 4](#_TOC_250001)
6. [Conclusion 4](#_TOC_250000)

# Abstract

In this paper we will discuss the approach we took while designing, compar- ing and testing our AI model that will detect health problems based on a well- Known dataset *heart disease.csv*. The discussion will be based on two phases 1 (midterm) and 2 (final).

# Introduction:

Due to heart disease's widespread prevalence and significant effects on public health, understanding it is essential. Cardiovascular illnesses are the world's largest cause of death, according to the World Health Organization, which emphasizes the need for thorough study and analysis in this area.

The dataset provided by Dr. Michelle, offers a valuable resource for this purpose. This dataset contains a wide range of features related to heart health, including clinical measurements and lifestyle factors, creating a rich environment for exploration and analysis.

Our main goal in this project is to use model selection, preprocessing methods, and data analysis to get valuable insights from the Heart Disease Dataset. Our goal is to find hidden patterns, identify important risk variables, and create predictive models for heart disease by exploring different machine learning algorithms and techniques. Therefore, our main purpose is to develop models that function well on new data, going beyond simple accuracy and increasing their dependability in practical applications. Our goal is to enhance patient outcomes and further the ongoing efforts in cardiovascular research by means of careful experimentation and thorough evaluation.

# Phase 1 (Midterm)

In this phase we were asked to apply data preparation, pre-processing and regu- larization techniques. Then we were asked to use 3 models and train them on the data then compare their results. We also were asked to use technique such as PCA to check how it affects the models.

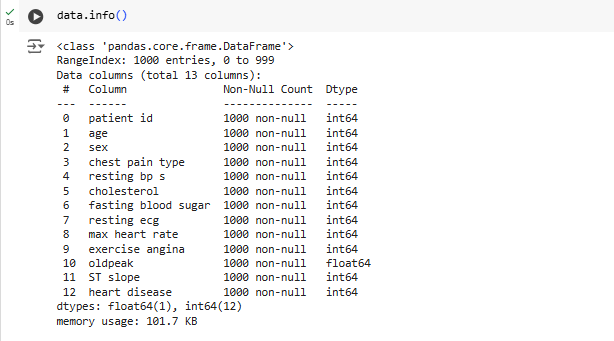
# Methodology

## Exploratory Data Analysis

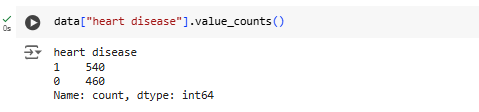
.

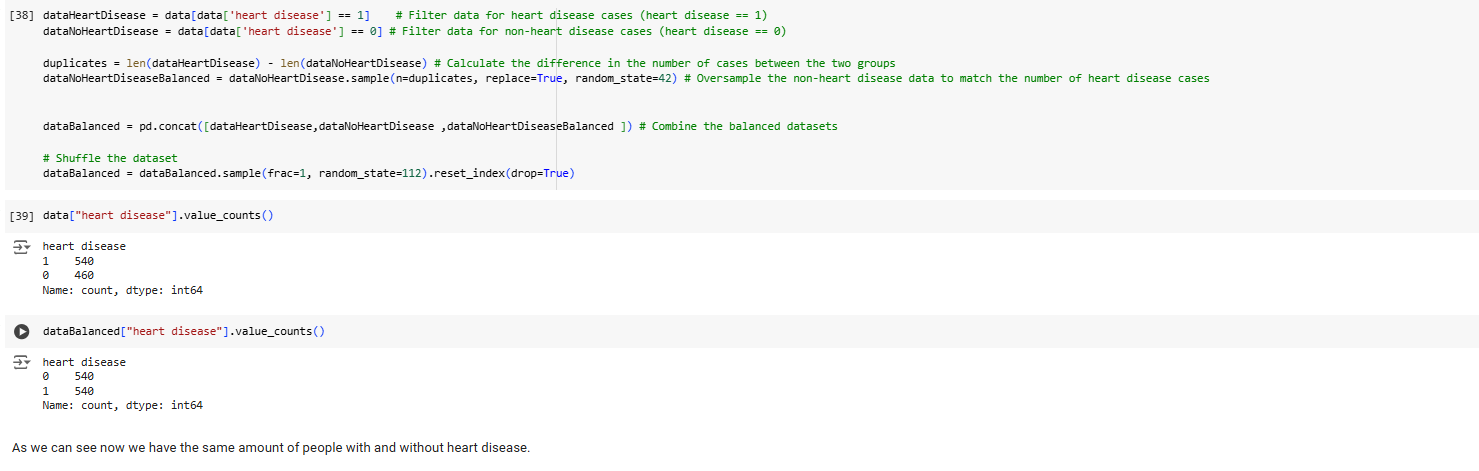
* + 1. We start by loading the dataset then checking if basic information about the features present in the dataset.

**Result:** We found out that all the features are numerical and there are no nulls present.



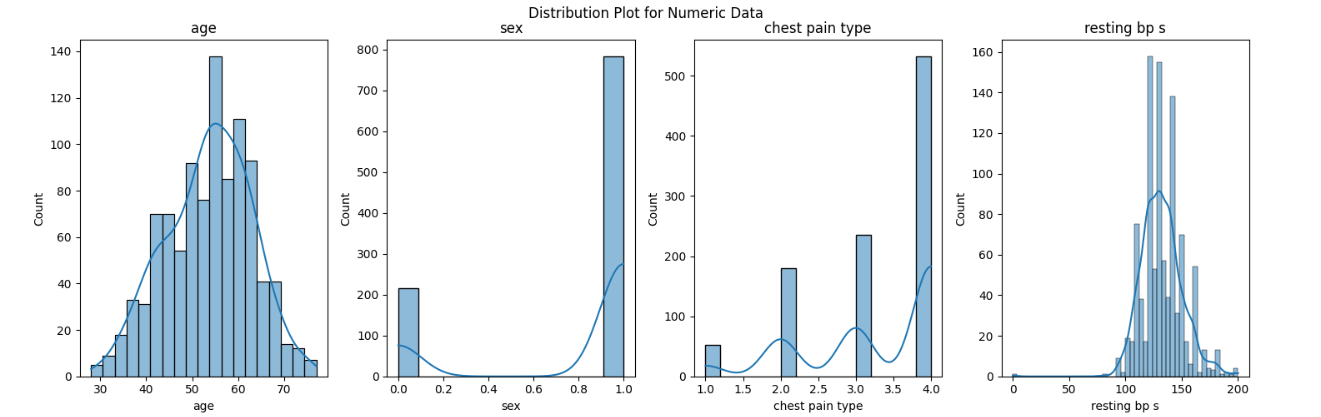
* + 1. After that we checked if the dataset is balanced.

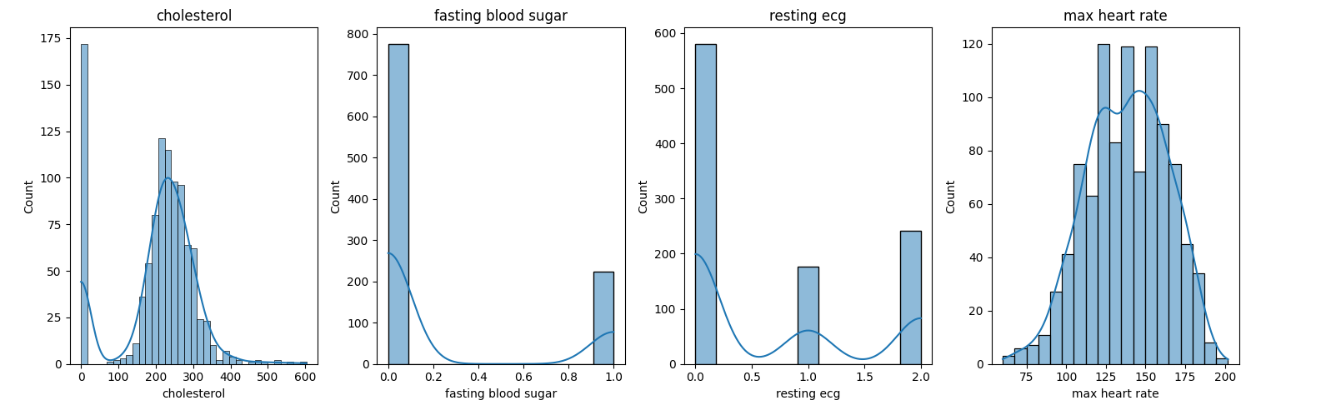
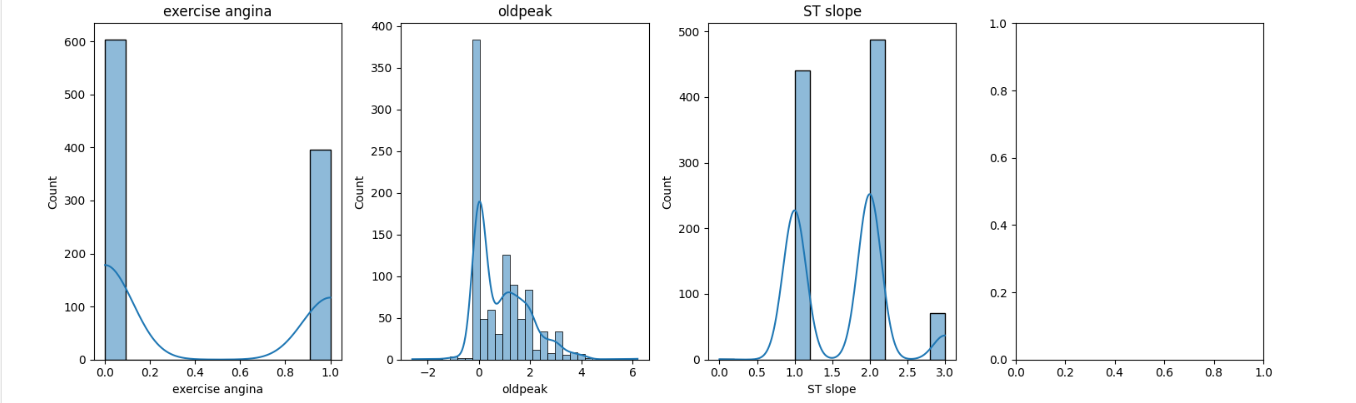




**Result:** We found out that the dataset contains 54% people with heart dis- ease and 46% without. So we duplicated some samples from the people without heart disease until the it reached 50 % and 50% for the two.(This step was not necessary as it data can be considered balanced).

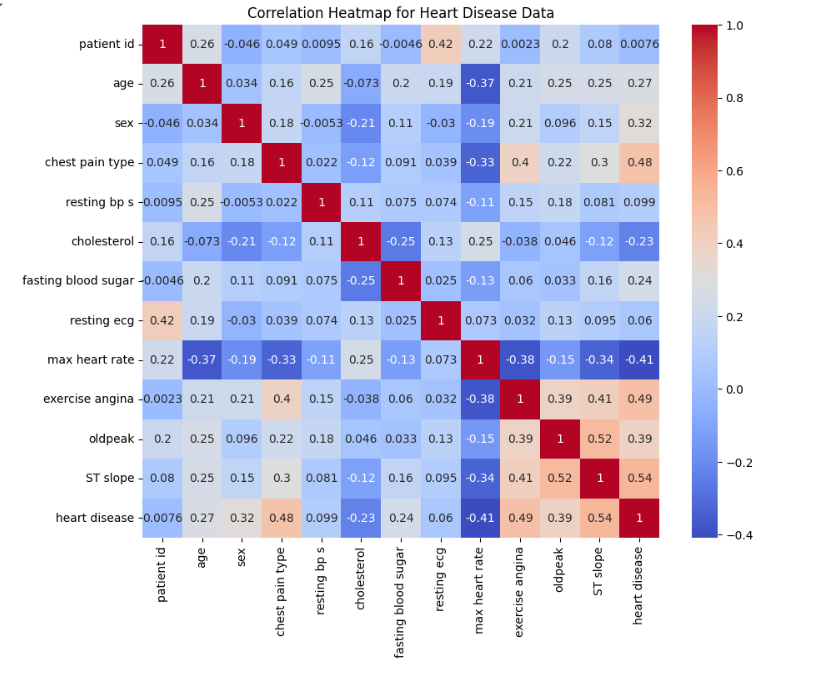
* + 1. After this we got some basic info about the distribution based on count of each feature.





**Result:** We found that multiple features such as (age, max heart rate, resting bp, cholesterol).

* + 1. We checked the correlation between every feature. But we focused on the features that are correlated to the heart disease.



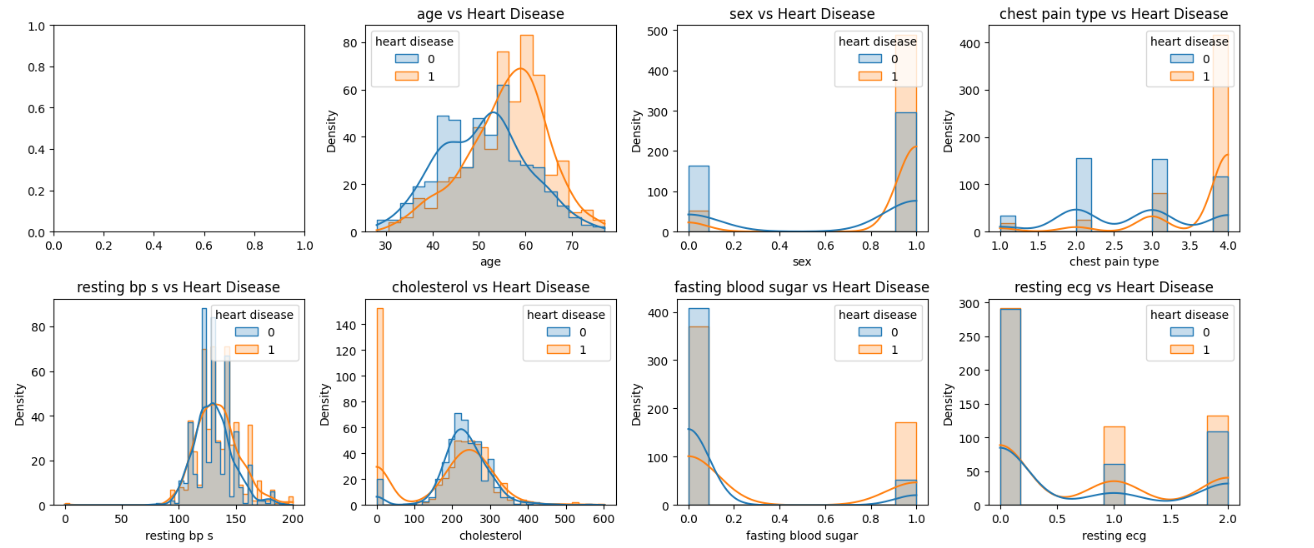
The heatmap shows a dataset's Pearson correlation between column pairs. The linear relationship strength between two variables is measured by Pearson correlation, which has a range of -1 to 1. Greater correlation is denoted by higher absolute values; a perfect positive correlation is represented by a value of 1, and a perfect negative correlation by a value of -1.

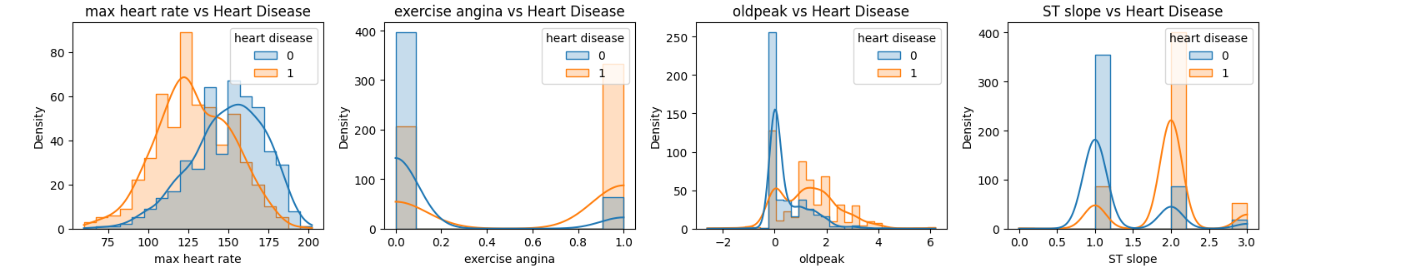
Colors near red in the heat map denote stronger positive association, whereas blue denotes stronger negative correlation. The major diagonal shows the perfect correlation between each column and itself.

Selecting which correlated columns to eliminate is an arbitrary process that is influenced by various factors, including model performance and threshold selection. Setting an arbitrary correlation threshold, merging correlated columns, or utilizing Principal Component Analysis (PCA) to minimize dimensionality while maintaining information are examples of potential tactics. The disadvantage of PCA is that feature interpretability may be lost.

**Result:** We can see that old peak, exercise angina and chest pain type were highly correlated to the target **Heart disease.**

* + 1. Lastly we plotted each feature density versus the target Heart disease.





**Result:**

Age: Between the ages of 55 and 63, a clear peak in heart disease incidence is observed.

Sex: Due to a significant gender imbalance in the dataset, it's challenging to draw definitive conclusions. However, males tend to exhibit a higher prevalence of heart disease compared to females.

Chest pain type: Individuals with chest pain type 4 almost always have heart disease.

Resting blood pressure: Resting blood pressure does not seem to correlate with the presence of heart disease.

Cholesterol: Low cholesterol levels (0) may indicate a higher risk of heart disease.

Fasting blood sugar: Fasting blood sugar levels above 120 mg/dl (value 1) are associated with a greater incidence of heart disease.

Resting ECG: Resting ECG results do not significantly impact the presence of heart disease.

Max heart rate: While maximum heart rate alone doesn't seem to correlate with heart disease, individuals with heart disease tend to have lower initial heart rates.

Old peak: Exercise-induced angina (value 1) suggests a higher risk of heart disease.

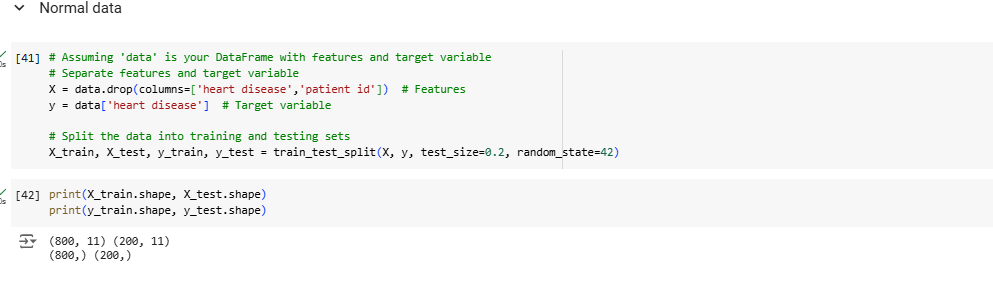
Exercise angina: Values above 1.5 indicate an increased risk of heart disease.

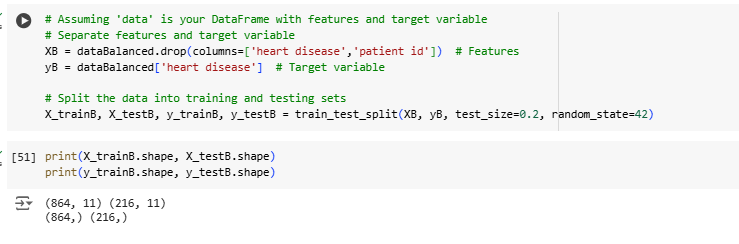
ST slope: Individuals with a flat ST slope (type 2) often have heart disease.

## Data pre-processing

* + 1. So as mentioned before we created duplicates to create 50% occurrence of people with and without heart disease. So now we have two dataset: Balanced dataset and the normal dataset.
    2. We split the data into training data (80%) and testing data (20%).

We used scikit-learn's train\_test\_split tool to split the dataset into training and testing sets. The training set  is used to train machine learning models, whereas the testing set is used for evaluating the models' performance.





* + 1. So for the next step we applied PCA on the data (balanced and normal) with a 95% threshold which resulted in 11 component (this is fixed in the second phase as 11 component is not an improvement for 12 features) but before this we should standerize the dataset, so we applied a stander scaler.

By reducing the mean and scaling the features to unit variance, the StandardScaler preprocessing approach standardizes the features of a dataset. The distribution of the data is essentially changed to have a mean of 0 and a standard deviation of 1.



• 𝑥scaled is the standardized value of the feature 𝑥x.

• 𝑥 is the original feature value.

• 𝜇 is the mean of the feature values in the training set.

• 𝜎 is the standard deviation of the feature values in the training set.

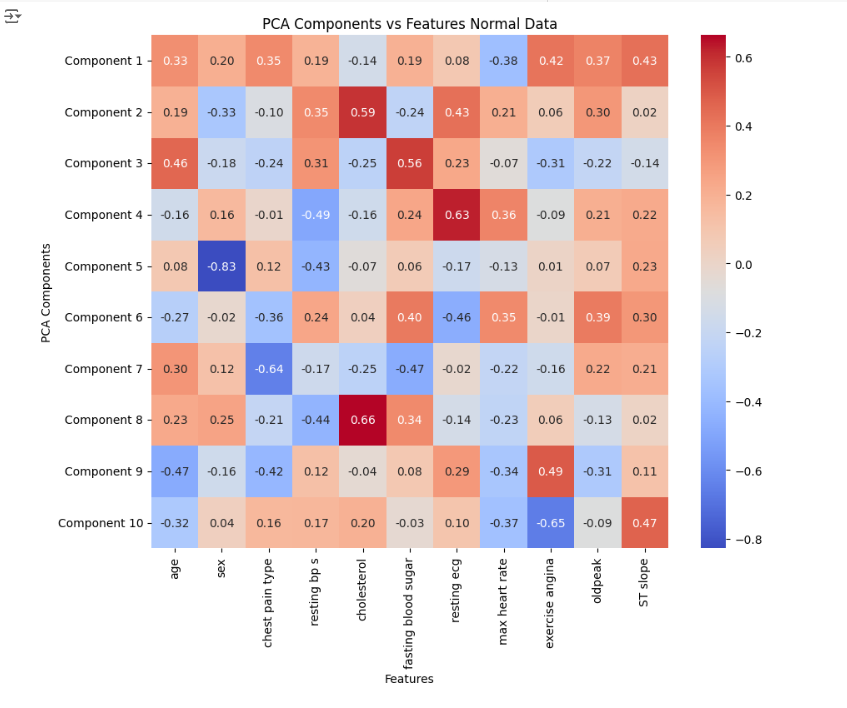
This formula divides each feature's scaled value by the standard deviation after subtracting the mean. This method creates a standardized distribution by mean removal (centering the data around zero) and variance scaling (scaling the data to have a standard deviation of 1).

A dimensionality reduction method called Principal Component Analysis (PCA) seeks to minimize the number of features in a dataset while maintaining its crucial information. Principal components, an additional set of orthogonal (uncorrelated) features created by changing the original features, allow principal component analysis (PCA) to efficiently capture the variation in the data using fewer dimensions.

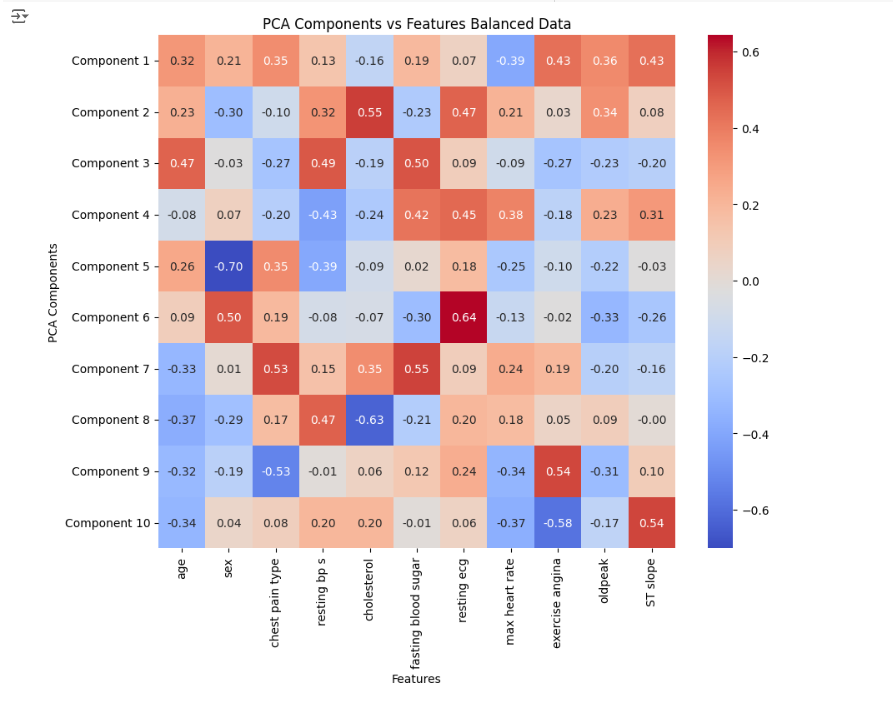
The benefit of PCA is that it can reduce dataset complexity while minimizing information loss. Without requiring extra input or arbitrary decision-making, PCA simplifies the data representation by identifying the most important patterns and correlations among the features.

However, there are several limitations of PCA that should be considered. While it effectively decreases dimensionality, it does so by replacing the original features with a new set of components determined by their covariance and correlation. This can result in a loss of interpretability because the new components may not precisely match to the original features within the context of the dataset's domain.

Normal data:



Balanced data:

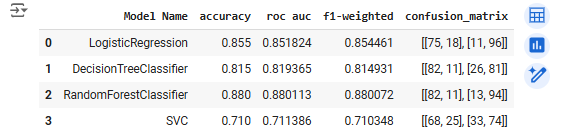


**Result:** Now we have 4 variance of the dataset: normal dataset, Balanced dataset, normal dataset with PCA applied, Balanced datset with PCA ap- plied.

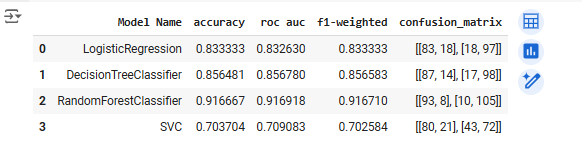
## Testing the 4 Models

* To guarantee the development of an optimal model for predicting heart disease diagnosis, we investigate numerous machine learning algorithms and select the one that exhibits the highest potential.
* First we chose 4 models:
  + LogisticRegression: This modrl is a fundamental classification technique that calculates the likelihood of a binary result based on one or more predictors. Despite its simplicity, Logistic Regression is frequently useful, especially when the relationship between the characteristics and the target variable is linear. It gives interpretable coefficients, allowing you to easily understand how each feature affects the prediction. However, it may not perform well when the relationship is non-linear or when features interact in complex ways.
  + DecisionTreeClassifier: The Decision Tree Classifier is a flexible and understandable method that divides a dataset into smaller subsets based on the most important properties. Each split is intended to optimize the homogeneity of the target variable among the created subsets. Decision trees are simple to depict and grasp, making them ideal for investigating feature importance and decision-making processes. However, they are prone to overfitting, particularly when the tree depth is not well restricted.
  + RandomForestClassifier : This ensemble learning technique uses many decision trees to build a strong prediction model. Each tree is trained on a random selection of data instances and features, which reduces overfitting while increasing accuracy. Combining the predictions of separate trees increases the model's predictive power. Notably, the specific methods for data segmentation and result accumulation can differ, which are often specified via hyperparameters.
  + SVC : The Support Vector Classifier is an extremely effective method for both classification and regression tasks. It works by identifying the best hyperplane to separate various classes in the feature space while optimizing the margin between them. SVC works well in high-dimensional domains and is adaptable since it can utilize multiple kernel functions to accommodate non-linear correlations between features and the target variable. However, SVC is sensitive to kernel and regularization parameter selection, and it can be computationally demanding for large datasets.
* Now for each variation of the dataset mentioned in the data pre-processing section above we applied the 3 models.

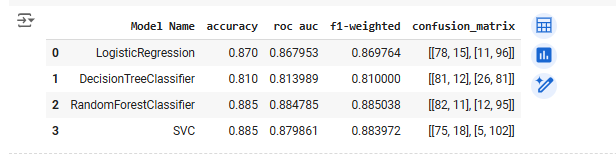
For the normal data:



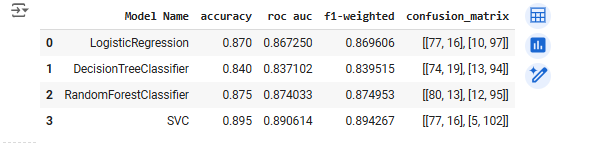
For the balanced data:



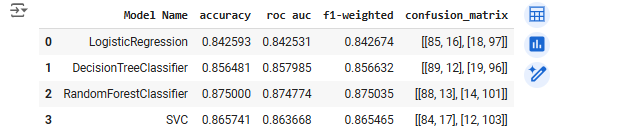
For the Standerized Normal Data:



For Standerized Normal data with PCA:



Standerized Balanced data with PCA:



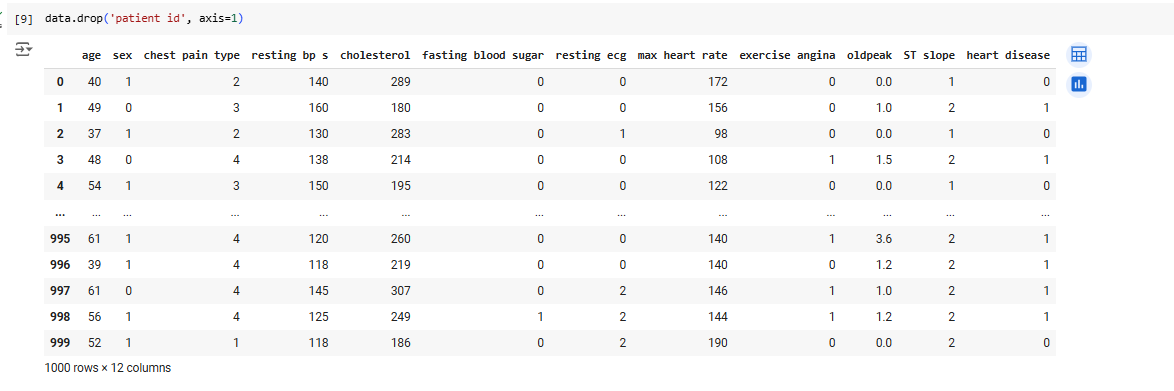
* We plotted the confusion matrix for each model on each variance of the dataset along with accuracy, recall and f1 score.(Which can be seen on the notebook).

**Result:** First we found out that the Random Forest algorithm performed the best (better on the balanced dataset with 93% accuracy). We found also that PCA may decrease the F1, recall etc.. of certain algorithms such as Random Forest , improve the F1, recall etc.. for Decision tree and SVC and not alter some such as Logistic regression.

**Note!!** We also applied L1 regularization but just on one model this will be tackled more in the Final.

# Phase 2 (Final)

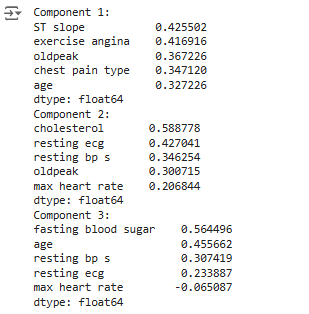
**5.1-** Here we noticed that the 'patient id' feature had no discernible impact on the predictive performance, so before continuing with hyper parameter tuning, we proceeded to remove it from the dataset.

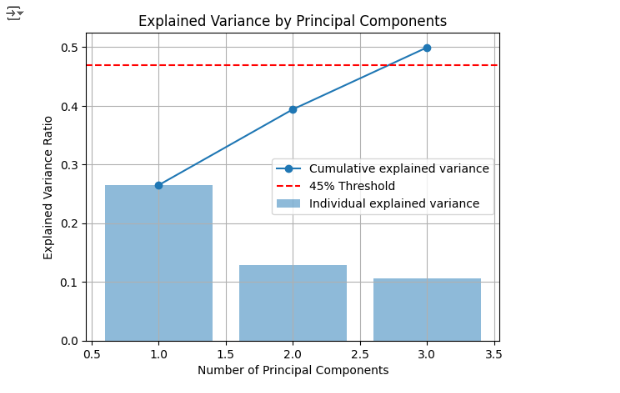


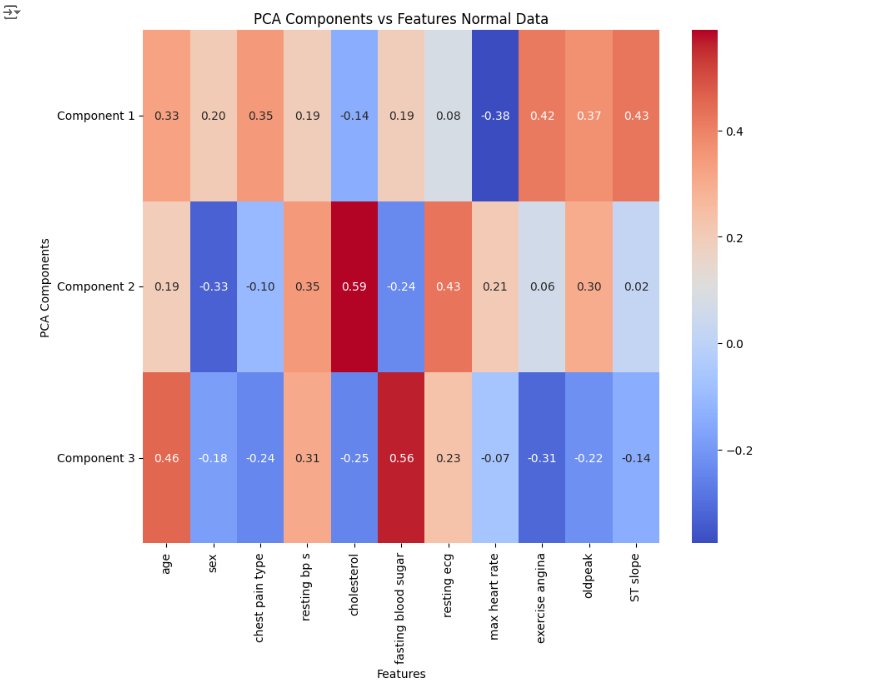
**5.2-** We also conducted another PCA analysis, this time utilizing three components on both the normal data and balanced data:

For the normal data:

The results reveal the most correlated features for each of the three components:

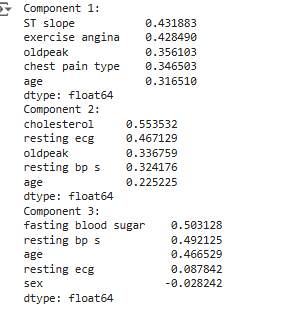


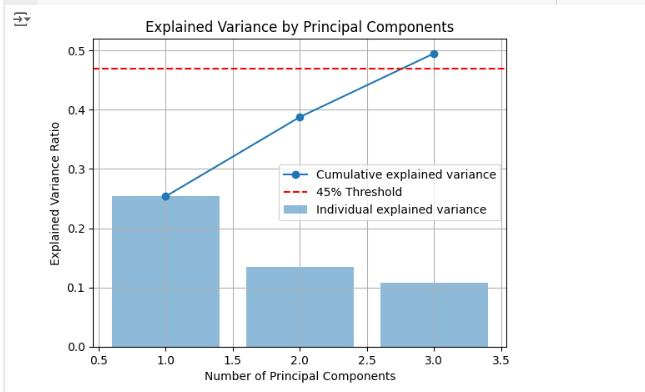


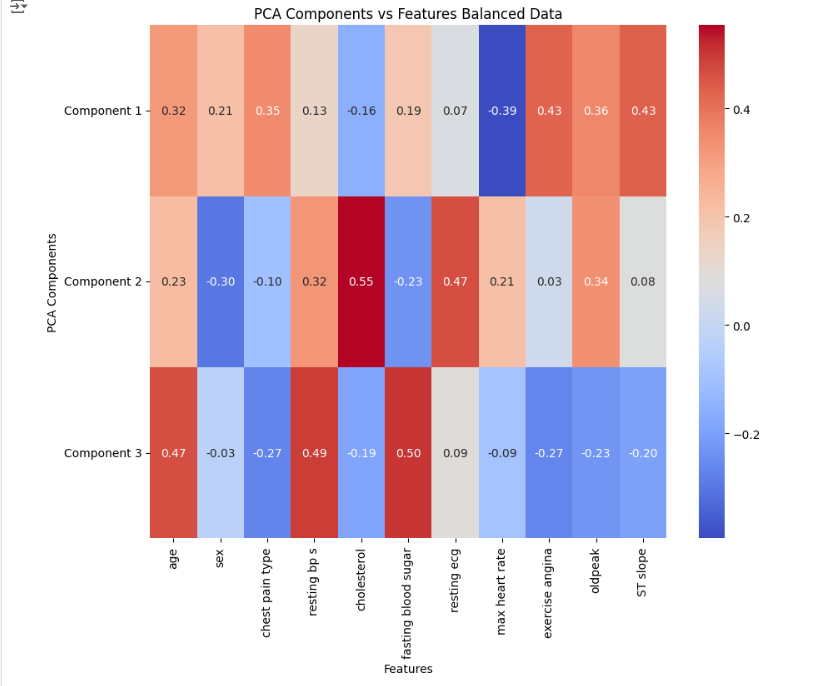


For the balanced data:

The results reveal the most correlated features for each of the three components:





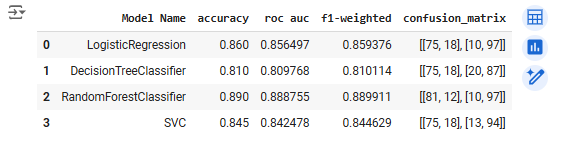


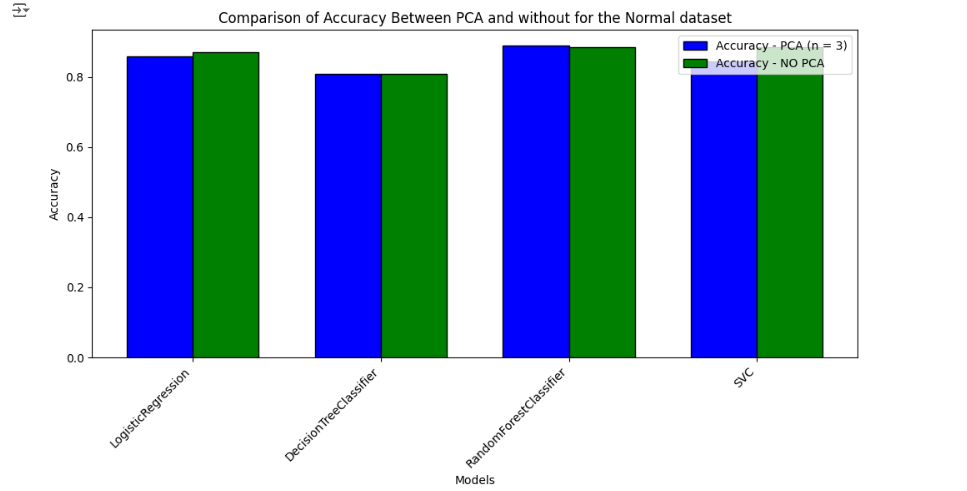
Concerning the graph:

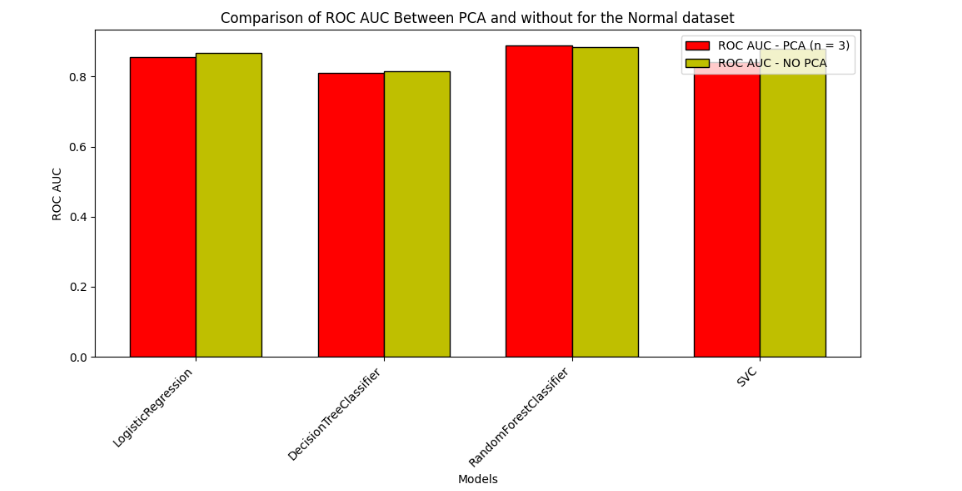
The 2 figures provide insights into the variance of the dataset captured by each component and the cumulative variance explained. It displays the explained variance ratio by principle components in a PCA analysis. It helps establish the ideal amount of principal components to retain while balancing model complexity and variance retention, making it an essential tool for evaluating dimensionality reduction. Setting targets for variance explained is made easier by adding a threshold line, like the widely used 95% threshold. Plotting the explained variance ratio for each component makes it easier to identify significant features or patterns that the PCA caught, which improves decision-making and model interpretation.

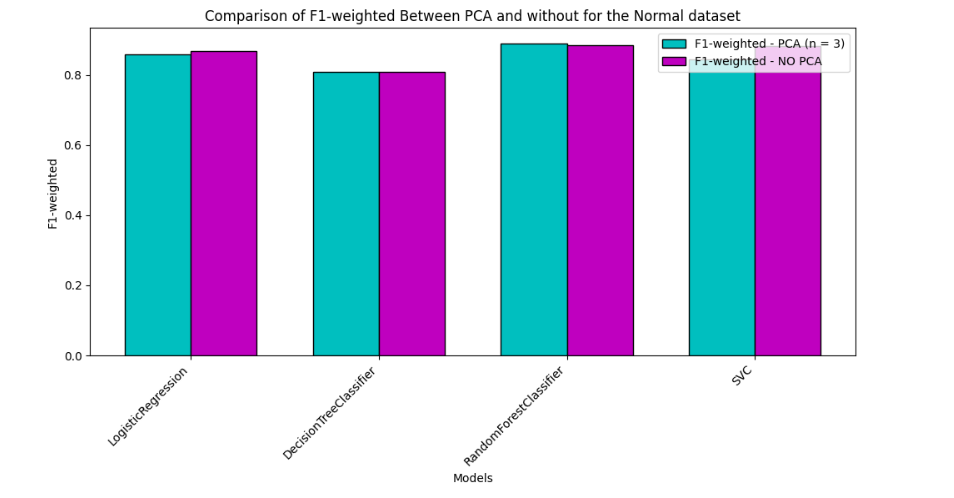
Here, we utilized a 45% threshold to assess the PCA analysis conducted with three components. This threshold indicates the desired level of variance retention, assisting in determining the appropriate number of principal components to include while balancing dimensionality reduction and information preservation.

**5.3-** We Then we went on to test the 4 models on those 2 data the normal data and balanced with PCA with 3 components and plotted their confusion matrices and compared it with the data without pca.

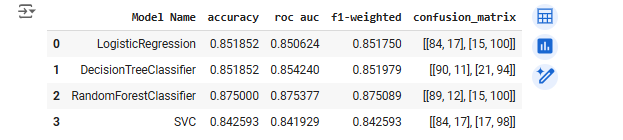
For Standerized normal data with PCA but with 3 components:  


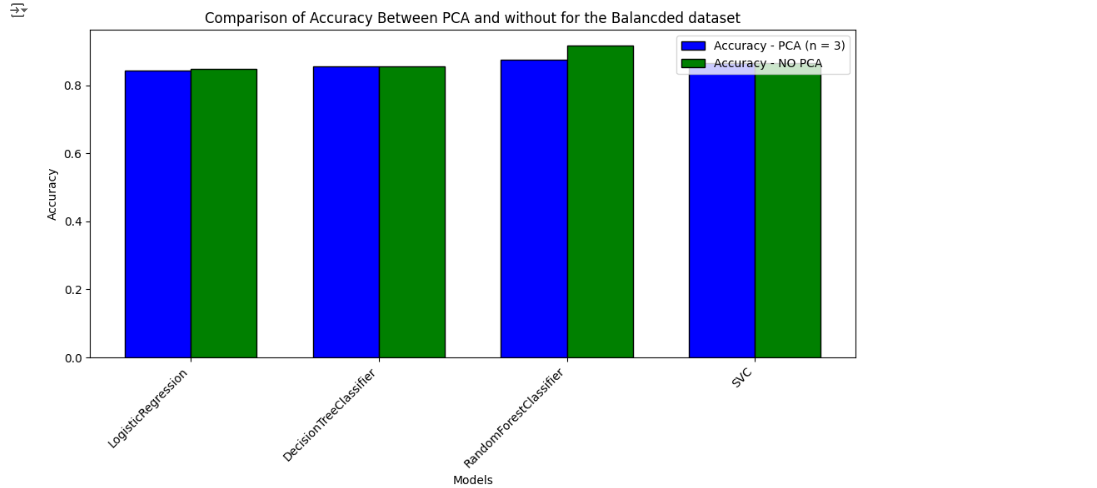


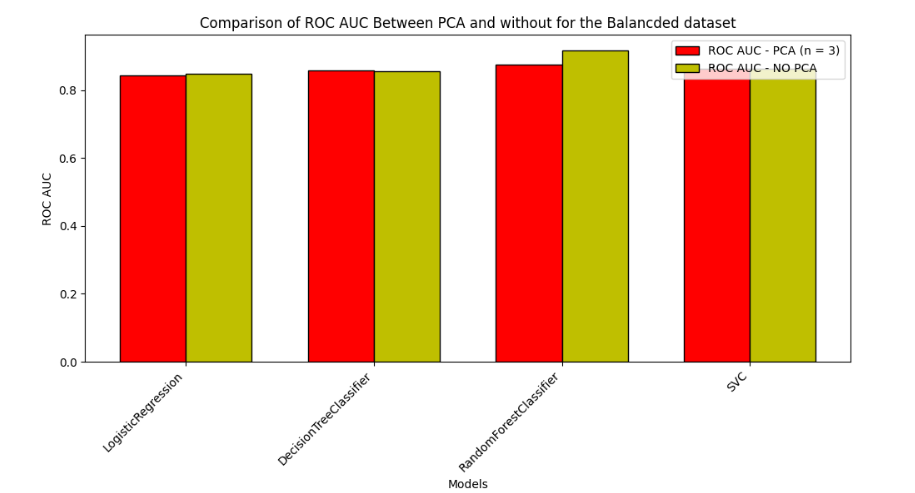


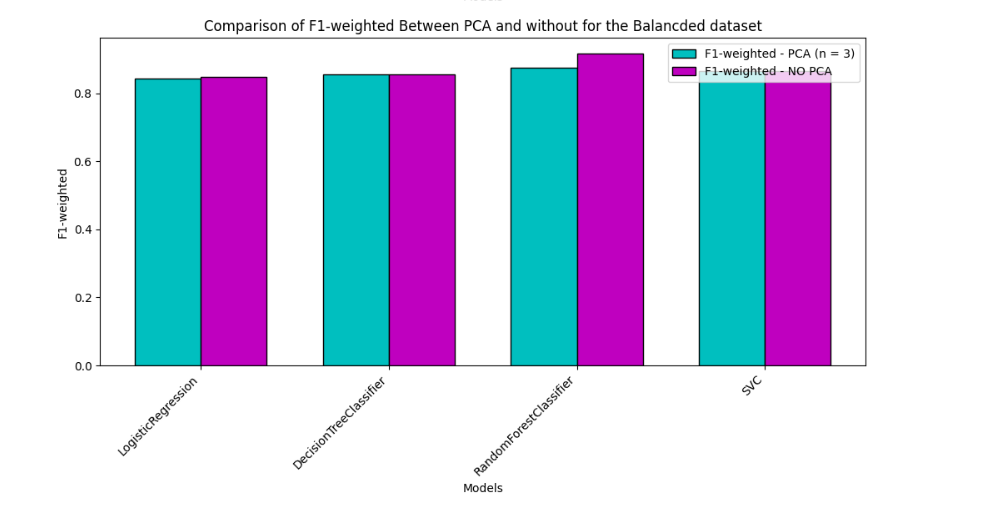


For Standerized balanced data with PCA but with 3 components:









**5.4-Hyperparameter tuning:**

We will perform hyperparameter tuning only on the RandomForestClassifier, DecisionTreeClassifier, and LogisticRegression because they provided high accuracies, and we will not perform it on the SVC due to its longer runtime and already demonstrating the lowest accuracy among the models. So instead of the SVC we used the XGBClassifier. And we will perform the hyperparameter tuning on the balanced data because it showed the highest accuracies during the initial modeling phase.

In our approach to hyperparameter tuning, we looked at three techniques: GridSearchCV, RandomizedSearchCV, and Bayesian optimization. We optimized hyperparameters for four different classifiers: XGBoost, Random Forest, Decision Tree, and Logistic Regression. For each classifier, we defined parameter grids that included essential hyperparameters such as the number of estimators, maximum depth, learning rate, and regularization parameters, which were selected based on their relevance to the algorithms and known impact on performance.

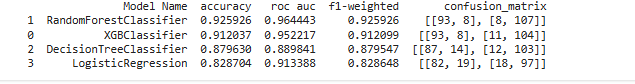
Subsequently, we initialized each classifier with default parameters and stored the outcomes of the hyperparameter tuning process, encompassing metrics such as accuracy, ROC AUC score, and F1-weighted score.

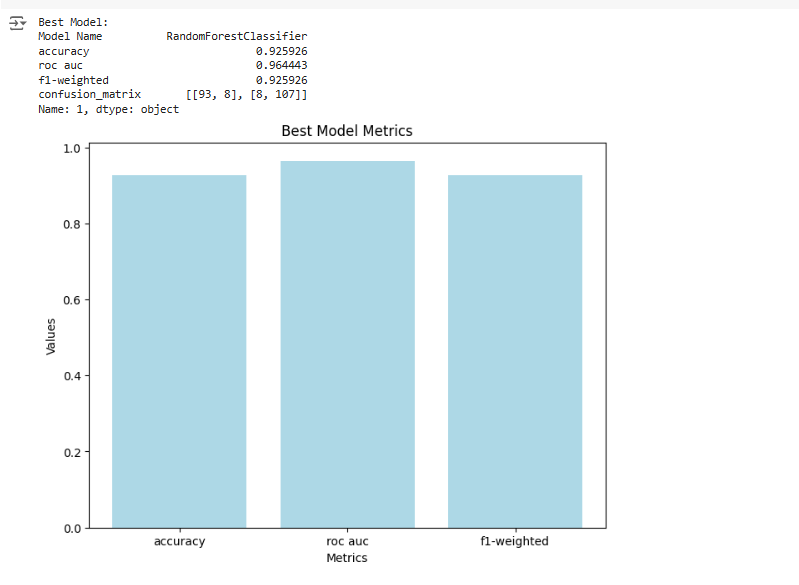
Finally, we plotted the confusion matrices for each model to assess their predictive performance in discriminating between real and false positives and negatives. The findings were organized in a DataFrame and ordered by F1-weighted score, providing insight into each classifier's efficacy in heart disease prediction. This iterative strategy aided in the selection of the best classifier with ideal hyperparameters for our dataset, ultimately improving the accuracy and reliability of our predictive model.

**GridSearchCV:**

GridSearchCV is a strategy for optimizing hyperparameters in machine learning models. It searches systematically through a hyperparameter grid, which is a collection of hyperparameters and their values, to identify the optimal combination that results in the best model performance. This technique uses cross-validation on each combination of hyperparameters to assess the model's performance and choose the optimal set of hyperparameters based on a predetermined score criteria.

We utilize GridSearchCV to automate the hyperparameter tuning process and find the ideal hyperparameters for our model without relying on manual trial and error. GridSearchCV ensures that we use the most effective hyperparameters by thoroughly searching the hyperparameter grid, optimizing our model's performance on unseen data.

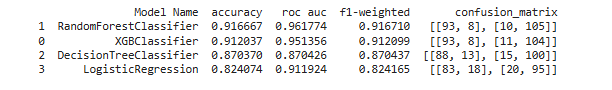


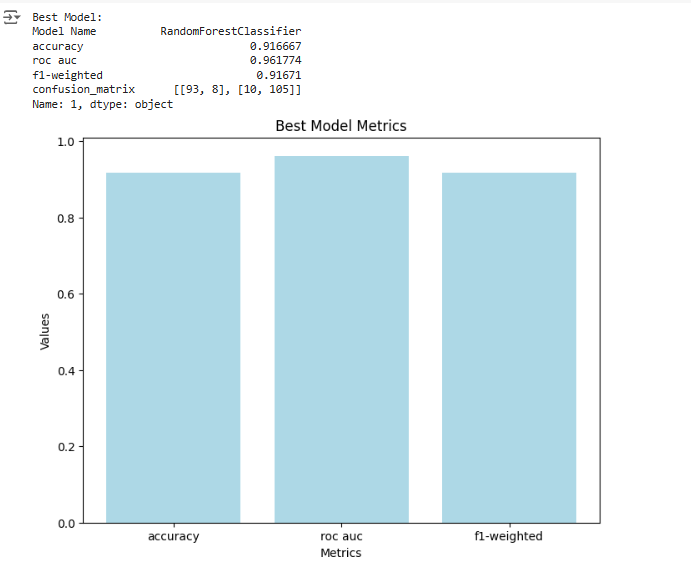
Subsequently, we obtained the optimal model and illustrated its attributes using a bar graph.

**RandomizedSearchCV:**

RandomizedSearchCV is another hyperparameter optimization strategy that investigates a subset of hyperparameters from a given distribution rather than exhaustively searching through all potential combinations, like GridSearchCV does. This strategy is especially effective when the hyperparameter space is big since it decreases computing time while maintaining a high probability of finding optimal hyperparameters.

RandomizedSearchCV can efficiently select hyperparameters from specified distributions, allowing for a more focused exploration of the hyperparameter space. This technique strikes a fair balance between exploration and exploitation, making it appropriate for scenarios with limited computational resources or a large hyperparameter space.

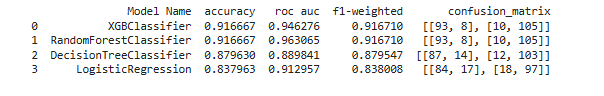


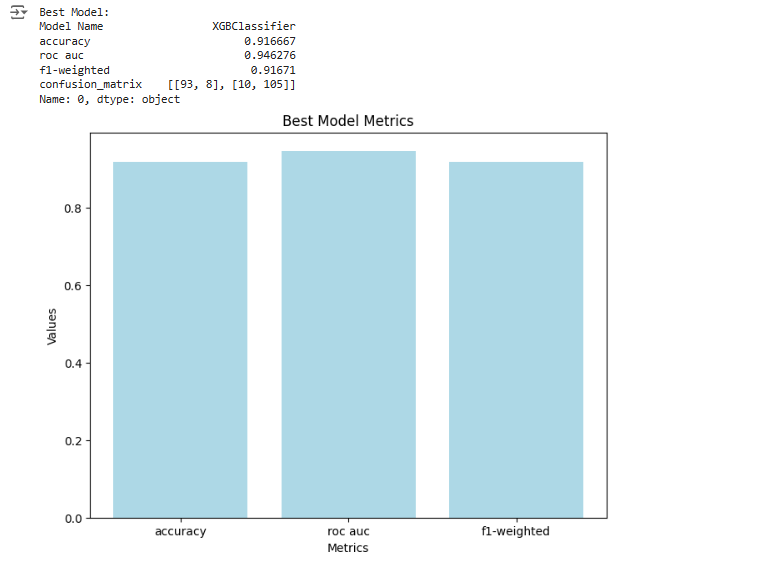


**Bayesian Optimization (using BayesSearchCV):**

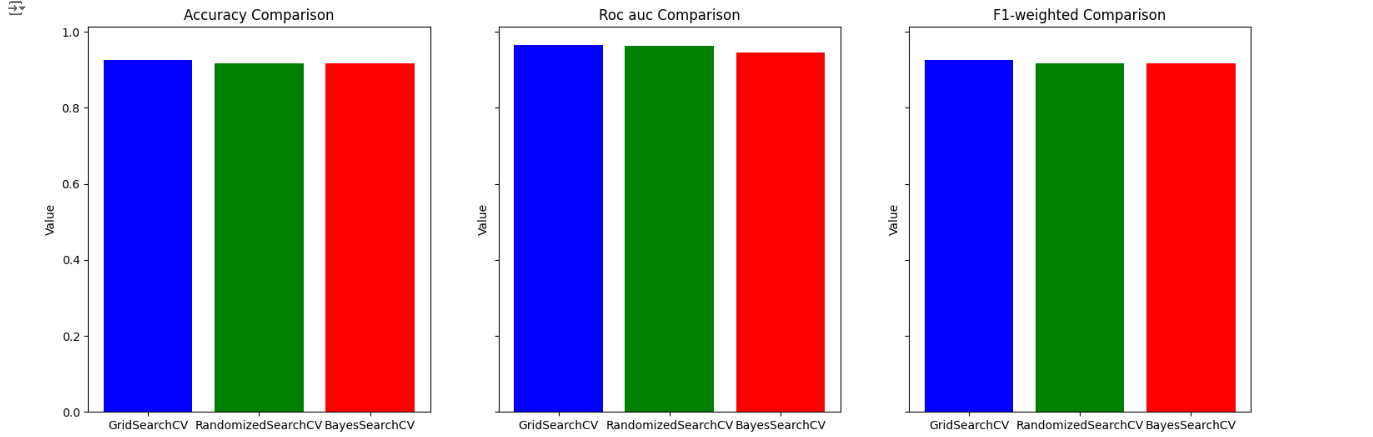
Bayesian optimization, as implemented by BayesSearchCV, is a probabilistic model-based optimization technique that uses information from prior evaluations to direct the search towards regions of the hyperparameter space that are more likely to produce better results. Unlike GridSearchCV and RandomizedSearchCV, which investigate the hyperparameter space in a systematic or random manner, Bayesian optimization use a probabilistic model to make informed decisions about which hyperparameters to examine next.

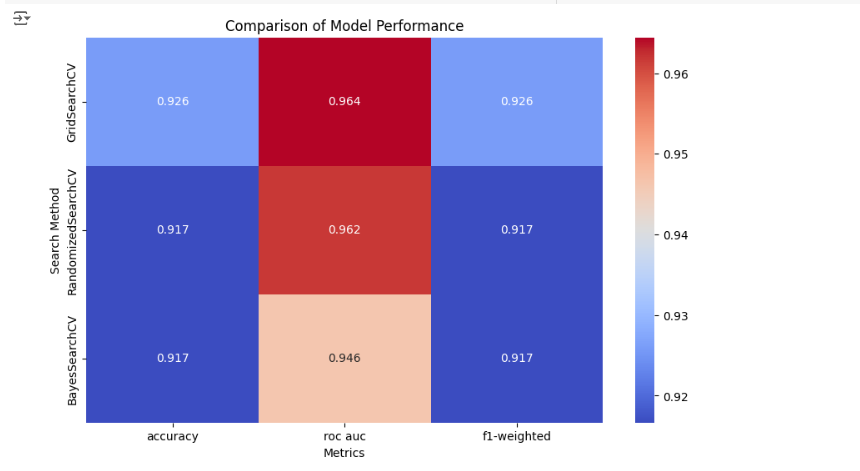
Bayesian optimization is especially useful in cases where evaluating each set of hyperparameters is computationally expensive or time-consuming. Bayesian optimization, which uses knowledge from prior evaluations, may rapidly explore the hyperparameter space and swiftly converge to optimal or near-optimal solutions, making it an effective technique for hyperparameter tuning in machine learning models.

****

****

**5.4- Model Evaluation:**

Here we got the best model of each of the three search techniques used in the hyper parameter tuning and compared them together.



We used seaborn to create a heatmap comparing the performance metrics (accuracy, ROC AUC, and F1-weighted) of the top three models generated by different hyperparameter search methods: GridSearchCV, RandomizedSearchCV, and BayesSearchCV. The heatmap visually represents the performance scores of each model across the selected metrics, allowing for easy comparison of how each search method impacted the resulting models. This visualization aids in identifying which hyperparameter search method led to the best-performing models across the evaluated metrics.



Based on the comparison of average scores across multiple performance metrics including accuracy, ROC AUC, and F1-weighted, the analysis concludes that the best model among GridSearchCV, RandomizedSearchCV, and BayesSearchCV is GridSearchCV. This indicates that the Random Forest Classifier, optimized through hyperparameter tuning using GridSearchCV, exhibited superior performance across the evaluated metrics compared to models generated through alternative hyperparameter optimization methods. Therefore, for this particular dataset and task, the GridSearchCV approach resulted in the most effective model configuration, providing enhanced predictive capabilities for identifying heart disease.

# Conclusion:

We have arrived at a reasonable conclusion following a thorough examination and analysis of the Heart Disease Dataset using methods like feature selection, data preprocessing, and model optimization. After extensive testing and analysis, it is clear that the GridSearchCV method is the best one for predicting heart disease, especially when used to adjust the Random Forest Classifier's hyperparameters. A thorough comparison of average scores for several important performance criteria, such as accuracy, ROC AUC, and F1-weighted scores, led to this conclusion.