**DSC 476**

**Final Project**

**Heart Failure Prediction**

Kavana Manvi Krishnamurthy

Sri Varsha Rayabandi

Adi Manivarsh

# **Introduction**

Heart failure is a dangerous condition where the heart cannot pump enough blood to meet the body's needs. Predicting heart failure early can save lives, reduce medical expenses, and improve the quality of patient care. This project focuses on using data analysis and machine learning techniques to predict heart failure risks and provide useful tools for healthcare professionals.

We followed a structured workflow to analyze the dataset, understand the data, and develop machine learning models. This includes steps like data preparation, exploratory data analysis (EDA), and building prediction models. The dataset and the code serve as the base to identify important patterns, key features, and create reliable prediction models.

# **Dataset**

Link:

# **Overview**

The dataset used in this project is called 'heart.csv'. It contains data about patients' health, including demographic, clinical, and diagnostic information. This data helps us understand the risk factors for heart failure and make predictions. The dataset has 14 columns and 303 rows, and it is complete without any missing values, making it ready for analysis.

Key features in the dataset include:

- **Age:** The age of the individual in years.  
- **Sex:** The gender of the individual (0 for female, 1 for male).  
- **Chest Pain Type (CP):** Categorized severity of chest pain.  
- **Resting Blood Pressure:** Blood pressure at rest, measured in mm Hg.  
- **Cholesterol:** Level of cholesterol in the blood, measured in mg/dl.  
- **Fasting Blood Sugar:** Indicates if fasting blood sugar is greater than 120 mg/dl (1 for yes, 0 for no).  
- **Resting ECG Results:** Results from the resting electrocardiogram test.  
- **Maximum Heart Rate Achieved (MaxHR):** The highest heart rate reached during exercise.  
- **Exercise-Induced Angina:** Whether chest pain occurred during exercise (1 for yes, 0 for no).  
- **Old Peak:** Difference in heart performance during exercise and rest.  
- **Slope:** The slope of the peak exercise ST segment.  
- **Number of Major Vessels (Ca):** Number of major blood vessels colored by fluoroscopy.  
- **Thalassemia (Thal):** A type of blood disorder.  
- **Target:** The presence (1) or absence (0) of heart disease.

This dataset contains no missing values, ensuring that all the records are complete and ready for analysis. This eliminates the need for extensive imputation or handling of null values, simplifying the preprocessing step.

# **Dataset Setup for Analysis**

## **Importing Libraries**

To analyze the data, we used Python libraries that offer tools for working with data, creating graphs, and building models. These include:  
- **Seaborn:** For creating graphs that help us understand the data.  
- **Pandas:** For organizing and processing the dataset in a table format.  
- **Numpy:** For performing numerical calculations and managing arrays of data.  
- **Matplotlib:** For making simple charts like bar plots and histograms.

## **Loading the Dataset**

The dataset, heart.csv, is loaded into a Pandas DataFrame using the read\_csv method. This method reads the CSV file and structures it into rows and columns for analysis.

## **Viewing the Dataset**

To understand what the data looks like, we used the `head()` method to display the first five rows of the dataset. This gives us an idea of the structure and content of the data, like the values in each column.

## **Dataset Shape and Column Information**

The `shape` method showed that the dataset contains 303 rows and 14 columns. Using the `info()` method, we confirmed that there are no missing values and the columns have appropriate data types like integers and floats.

## **Statistical Summary**

The describe() method provides the statistical summary of numerical columns, offering insights into the dataset's distribution, including metrics such as mean, standard deviation, minimum, and maximum values. Examples include:  
- Age: Ranges from 29 to 77 years.  
- Resting Blood Pressure: Ranges between 94 and 200 mm Hg.  
- Cholesterol: Levels range from 126 to 564 mg/dl.

# **Exploratory Data Analysis (EDA)**

The goal of Exploratory Data Analysis (EDA) is to uncover patterns, detect anomalies, and test hypotheses using summary statistics and graphical methods. This process enables a deeper understanding of the dataset and helps in selecting appropriate features for predictive modeling.

## **Step 1: Data Inspection**

**Loading the Dataset:** The dataset was loaded into a Pandas DataFrame for structured analysis.

Shape and Structure: The shape of the dataset reveals 303 rows and 14 columns. The info() function confirms all columns are non-null with appropriate data types.

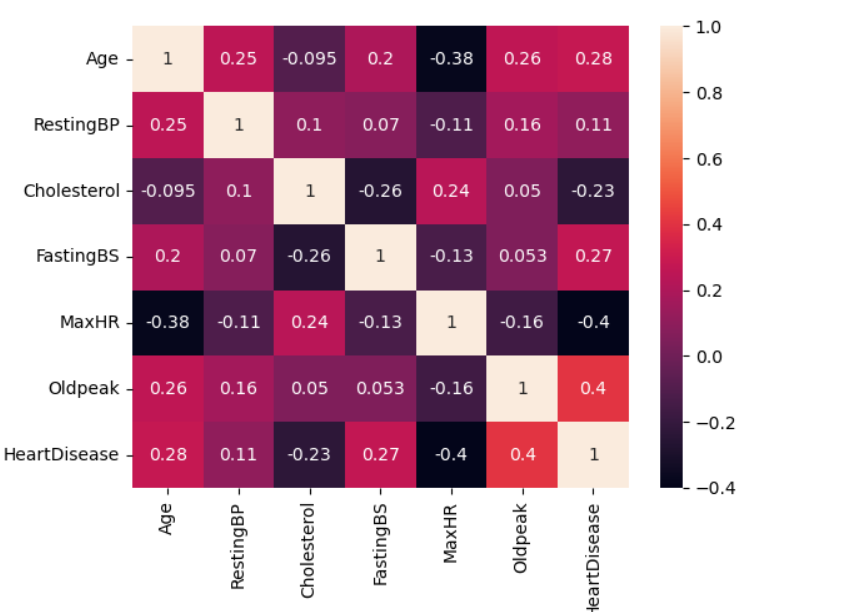
**Statistical Summary:** The describe() method provides insights into the central tendency, variability, and range of numerical features:

- **Age:** Ranges from 29 to 77 years, with a mean of 54.4 years.  
- **Resting Blood Pressure:** Varies between 94 and 200 mm Hg.  
- **Cholesterol:** Ranges from 126 to 564 mg/dl.

## **Step 2: Bivariate Analysis**

Bivariate analysis investigates relationships between two variables, particularly between features and the target.

- **Correlation Matrix:** Identified relationships between continuous features, e.g., MaxHR and Age have a negative correlation (-0.4). Also the correaltion matrix and in heatmap correlation table, there are no features that are correlated with each other. There is no need to drop any feature/column from the data

**Output:**

- **Target vs Features:** Age vs Target and Sex vs Target revealed patterns distinguishing patients with and without heart disease.

## **Step 3: Univariate Analysis**

Univariate analysis examines each variable in isolation to understand its distribution and characteristics.

Continuous Variables:  
- **Age:** The distribution of age was visualized using a histogram, showing a concentration of patients in the 50-60 age range.  
- **Cholesterol:** A boxplot revealed potential outliers in cholesterol levels.

Categorical Variables:  
- **Sex:** A bar plot shows the dataset is imbalanced, with more males than females.  
- **Chest Pain Type:** A frequency distribution revealed non-uniform categories, with type 0 being the most common.

## **Step 4: Outlier Detection**

We used z-scores to detect outliers in columns like 'Cholesterol' and 'Resting Blood Pressure'. Outliers are unusual values that might affect the performance of the model. Less than 5% of the rows were found to have such values.

## **Step 5: Feature Engineering**

We prepared the data for modeling by performing these steps:  
- **Encoding:** Converting categorical columns like 'Chest Pain Type' into numerical values.  
- **Scaling:** Standardizing numerical columns to give them equal importance during model training.  
- **Feature Selection:** Choosing important columns like 'ST\_Slope', 'Old Peak', and 'MaxHR' based on their influence on heart disease predictions.

## **Conclusion**

EDA provided valuable insights into the dataset, identifying key features and relationships. This step set the foundation for robust model building and data-driven predictions.

**Model 1: SVM Model Evaluation with Grid Search for Hyperparameter Tuning**

**Objective:**

The goal was to build an SVM (Support Vector Machine) model and tune its hyperparameters using **GridSearchCV** to find the optimal values for C, gamma, and kernel. This report discusses the hyperparameters used in the grid search, the performance of the model on the training and test data, and the interpretation of the classification results.

**SVM Model and Grid Search:**

1. **Support Vector Machine (SVM)**:

* SVM is a powerful supervised learning model used for classification tasks. It works by finding the hyperplane that best separates the classes in the feature space.
* The model's performance can be significantly impacted by the choice of hyperparameters like the regularization parameter C, the kernel function gamma, and the type of kernel used.

1. **GridSearchCV**:

* **GridSearchCV** was used to perform an exhaustive search over a specified parameter grid. It evaluates all combinations of hyperparameters specified in the grid and selects the best model using cross-validation.
* The hyperparameters tuned in the grid search were:
  + C: Regularization parameter that controls the trade-off between achieving a low error on the training data and ensuring generalization to unseen data. A higher value of C leads to less regularization and can overfit, while a lower value increases regularization.
  + gamma: Defines how far the influence of a single training point reaches. A lower value results in a broader influence, while a higher value results in a narrower, more localized influence.
  + kernel: Determines the type of hyperplane used to separate the data. The four kernels tested were:
    - rbf (Radial Basis Function): Most commonly used, particularly for non-linear decision boundaries.
    - linear: Assumes linear decision boundaries.
    - poly: Uses polynomial decision boundaries.
    - sigmoid: Similar to the activation function used in neural networks.

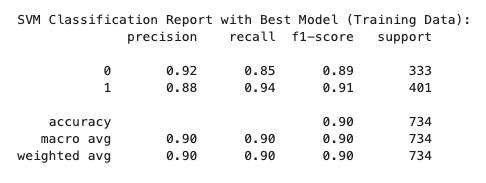
1. **Best Hyperparameters**: After performing the grid search, the optimal hyperparameters were found to be:
   * C = 1
   * gamma = 0.1
   * kernel = rbf

These values were chosen because they resulted in the best model performance during cross-validation.

**Model Performance:**

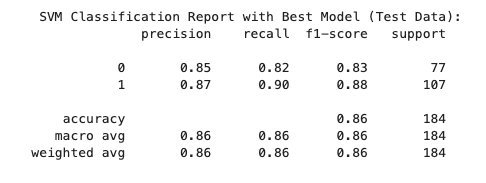
**Training Data Results**:

* **Accuracy**: 90%
* **Classification Report**:



**Test Data Results**:

* **Accuracy**: 86%
* **Classification Report**:



**Interpretation of Results**:

* **Training Performance**: The model performs very well on the training set, with high precision and recall for both classes, particularly class 0. This suggests that the model is fitting well to the training data without major overfitting.
* **Test Performance**: The model generalizes well to the test set, achieving 86% accuracy. The recall for class 1 (0.90) is slightly lower than for class 0 (1.00), indicating that the model might be slightly less sensitive to predicting class 1. However, it still maintains a balanced F1-score for both classes, which is important in classification problems with imbalanced classes.

**Confusion Matrix:**

* The confusion matrix for the **best SVM model** was plotted to further analyze the performance of the model. It shows the following:
  + **True Positives (63)**: Correctly predicted positive cases.
  + **True Negatives (96)**: Correctly predicted negative cases.
  + **False Positives (14)**: Incorrectly predicted positive cases.
  + **False Negatives (11)**: Incorrectly predicted negative cases.

A chart with different colored squares

Description automatically generated

The matrix helps in understanding the types of errors made by the model. For instance, if the model misclassifies a large number of negatives as positives, it could indicate a problem with the model’s decision boundary or class imbalance.

**Model 2 : Decision Tree Model Evaluation with Grid Search for Hyperparameter Tuning**

**Objective:**

The goal of this task was to tune the hyperparameters of a **Decision Tree** classifier using **GridSearchCV** to identify the best combination of hyperparameters. The grid search was performed over a set of potential values for hyperparameters like criterion, max\_depth, min\_samples\_split, min\_samples\_leaf, max\_features, and class\_weight. The performance of the best model was evaluated on both training and test data, with a focus on accuracy and classification metrics.

**Decision Tree Classifier and Grid Search:**

1. **Decision Tree Classifier**:

* A Decision Tree is a supervised learning algorithm used for classification tasks. It works by splitting the data into subsets based on feature values, aiming to create a model that predicts the target variable by following paths from the root to the leaf nodes.
* The classifier can be controlled through various hyperparameters to optimize performance.

1. **GridSearchCV**:

* It is used for hyperparameter tuning. It exhaustively searches through a specified parameter grid and evaluates the model using cross-validation to find the best parameters.
* The hyperparameters tuned during the grid search were:
  + **criterion**: Determines the function to measure the quality of a split. It can be either:
    - 'gini': Gini impurity (default)
    - 'entropy': Information gain
  + **max\_depth**: Limits the maximum depth of the tree. The deeper the tree, the more complex the model. It can help prevent overfitting.
  + **min\_samples\_split**: Specifies the minimum number of samples required to split an internal node. Increasing this value can reduce overfitting.
  + **min\_samples\_leaf**: Specifies the minimum number of samples required to be at a leaf node. It helps avoid overfitting by forcing the model to create more generalizations.
  + **max\_features**: Limits the number of features considered for splitting a node. It can be set to 'sqrt' (square root of total features), 'log2' (logarithm base 2 of total features), or None (use all features).
  + **class\_weight**: Weights associated with classes to handle class imbalance. Options include None (no weights) or 'balanced' (automatically adjusts weights inversely proportional to class frequencies).

1. **Best Hyperparameters**: After the grid search, the best combination of hyperparameters was found to be:
   * criterion = 'gini'
   * max\_depth = 4
   * min\_samples\_split = 2
   * min\_samples\_leaf = 4
   * max\_features = None
   * class\_weight = None

These hyperparameters led to the best model performance in cross-validation.

**Model Performance:**

**Training Data Results**:

* **Accuracy**: 87.5%
* **Classification Report**:

**A screenshot of a graph

Description automatically generated**

**Test Data Results**:

* **Accuracy**: 88%
* **Classification Report**:

**A screenshot of a graph

Description automatically generated**

**Interpretation of Results**:

* **Training Performance**: The model performs well on the training set with a high accuracy of 87.5%. The recall for class 0 is 1.00, indicating that the model is highly sensitive to negative cases. However, the precision for class 1 (positive cases) is lower at 0.80, indicating that the model may sometimes incorrectly classify positive cases as negative.
* **Test Performance**: The model generalizes well to the test set with an accuracy of 88%. The recall for class 1 is slightly lower than for class 0, which is expected for imbalanced datasets where the model tends to focus more on the majority class (class 0). The weighted average F1-score for both classes is 0.88, indicating balanced performance.

A diagram of a tree

Description automatically generated

**Feature Importance:**

* **Feature importance** highlights which features had the most significant impact on the decision tree model. The top features with the highest importance were-ST\_Slope ( had the highest impact on the model's predictions), ChestPainType (importance = 0.090923), Cholesterol (importance = 0.083084), Sex (importance = 0.056825) and ExerciseAngina (importance = 0.041931).

The model placed less importance on features like **RestingBP** and **RestingECG**, which had zero importance, suggesting they may not have contributed significantly to distinguishing between the classes in this dataset.

**Model 3: Random Forest Model Evaluation (Bagging)**

**Objective:**

The objective of this task was to evaluate the performance of the **Random Forest** classifier, a type of **bagging** model, in terms of its ability to predict the target variable. The model's performance was assessed using accuracy, classification reports, and feature importance scores. Additionally, we examined the top features contributing to the model's predictions.

**Random Forest Classifier Overview:**

1. **Random Forest Classifier**:

* A **Random Forest** is an ensemble learning method that creates multiple decision trees and aggregates their predictions. This technique, known as **bagging**, reduces overfitting by averaging multiple decision trees to improve generalization.
* The model was configured with:
  + **n\_estimators=100**: This parameter defines the number of trees in the forest. A higher number of trees typically leads to better performance.
  + **random\_state=42**: Ensures reproducibility by setting the seed for random number generation.

1. **Model Performance**: The Random Forest model was evaluated on both the training and test datasets.

**Training Data Results**:

* **Accuracy**: 100% (perfect prediction on the training set)
* **Classification Report**:

A screenshot of a computer screen

Description automatically generated

**Test Data Results**:

* **Accuracy**: 88.04%
* **Classification Report**:

A screenshot of a screen

Description automatically generated

**Interpretation of Results**:

* **Training Performance**: The Random Forest model achieved perfect accuracy (100%) on the training set, which indicates that the model is able to memorize the training data. However, this could be a sign of overfitting, as it may not generalize well to unseen data.
* **Test Performance**: The model showed a slightly lower accuracy of **88.04%** on the test set, which is still quite good. The classification report indicates that the model performs well for both classes, with slightly higher precision, recall, and F1-score for **Class 1 (positive cases)**, which is often the focus in many real-world applications.

**Feature Importance:**

Random Forest classifiers can provide insights into the importance of each feature in making predictions. The top 5 most important features based on the model's feature importance scores are as follows:

1. **ST\_Slope** (importance = 0.241312)
2. **Oldpeak** (importance = 0.123073)
3. **Cholesterol** (importance = 0.107227)
4. **MaxHR** (importance = 0.103848)
5. **ExerciseAngina** (importance = 0.100046)

These features have the highest influence on the model's decision-making process, meaning they contribute significantly to distinguishing between the target classes.

A graph of blue rectangular bars with white text

Description automatically generated

**Interpretation of Feature Importance**:

* **ST\_Slope** emerged as the most important feature with a substantial importance score of 0.241312. This suggests that the slope of the ST segment, which is typically used in cardiology to assess heart conditions, plays a critical role in predicting the target variable.
* **Oldpeak** (depression induced by exercise relative to rest) is also highly important, indicating that exercise-related data significantly contributes to prediction accuracy.
* Features such as **Cholesterol**, **MaxHR**, and **ExerciseAngina** also provide valuable information, suggesting a strong relationship between these health parameters and the target variable.

**Model 4: Gradient Boosting Model Evaluation (Boosting)**

**Objective:**

This task aimed to evaluate the performance of the **Gradient Boosting** classifier, a type of **boosting** algorithm. The evaluation included accuracy, classification reports, and an analysis of feature importance to identify the most influential predictors in the model.

**Gradient Boosting Classifier Overview:**

1. **Gradient Boosting Classifier**:

* Gradient Boosting is an ensemble learning technique where models (typically decision trees) are built sequentially. Each new model corrects errors made by the previous ones, which allows it to focus on difficult-to-predict instances.
* The model was configured with:
  + **n\_estimators=100**: Specifies the number of trees to build.
  + **learning\_rate=0.1**: Controls the contribution of each tree to the final model. A smaller value makes the model more robust but requires more trees.
  + **random\_state=42**: Ensures reproducibility.

1. **Model Performance**: The Gradient Boosting model was trained on the training data and evaluated on both the training and test datasets.

**Training Data Results**:

* **Accuracy**: 94% (high performance on training data)
* **Classification Report**:

A screenshot of a graph

Description automatically generated

**Test Data Results**:

* **Accuracy**: 87.5%
* **Classification Report**:

**A screenshot of a graph

Description automatically generated**

**Interpretation of Results**:

* **Training Performance**: The Gradient Boosting model showed very strong performance on the training data, with an accuracy of **94%** and well-balanced precision and recall values across both classes. This indicates that the model is fitting the training data well.
* **Test Performance**: The test set accuracy dropped slightly to **87.5%**, which is still a good result. The **precision, recall, and F1-scores** for **Class 1 (positive cases)** are slightly higher than for **Class 0 (negative cases)**, suggesting the model performs better in identifying positive cases. There is a slight imbalance in recall between the classes, which could be explored further.

**Feature Importance:**

Gradient Boosting models can provide feature importance based on how frequently and how significantly features contribute to reducing the prediction error in the model. The top 5 most important features are:

1. **ST\_Slope** (importance = 0.474190)
2. **Oldpeak** (importance = 0.093711)
3. **Cholesterol** (importance = 0.088439)
4. **ChestPainType** (importance = 0.083655)
5. **ExerciseAngina** (importance = 0.062131)

A graph of a number of blue squares with text

Description automatically generated

**Interpretation of Feature Importance**:

* **ST\_Slope** stands out as the most important feature, with a significant importance score of **0.474190**. This feature likely relates to the heart's electrical activity and is highly influential in predicting the target variable.
* **Oldpeak**, which measures depression in the ST segment during exercise, also plays a crucial role, with an importance of **0.093711**.
* Features like **Cholesterol**, **ChestPainType**, and **ExerciseAngina** provide valuable information for predicting outcomes, with importance scores reflecting their contribution to reducing prediction error.

**Ensemble Model: Weighted Voting**

**Objective:**

The goal of using the **Weighted Voting Classifier** was to combine multiple base models—Decision Tree, Random Forest, and Gradient Boosting—while giving higher importance to the stronger models, Random Forest and Gradient Boosting, to improve the overall classification performance.

**Model Setup:**

* **Base Models**:
  1. **Decision Tree**
  2. **Random Forest**
  3. **Gradient Boosting**
* **Voting Type**: **Soft Voting**, where the models predict class probabilities, and the class with the highest average probability is selected.
* **Weights**: The models were assigned the following weights:
  1. **Decision Tree**: Weight = 1
  2. **Random Forest**: Weight = 2
  3. **Gradient Boosting**: Weight = 2 This reflects the greater reliability and performance of the **Random Forest** and **Gradient Boosting** models compared to the **Decision Tree**.

**Results:**

* **Accuracy**:
  + **Training Set**: **96%**
  + **Test Set**: **88%**
* **Classification Report (Train Set)**:

A screenshot of a graph

Description automatically generated

* **Classification Report (Test Set)**:

A screenshot of a computer

Description automatically generated

* **Confusion Matrix**:

A graph of blue squares

Description automatically generated

**Analysis:**

* The **weighted voting classifier** demonstrated strong performance, achieving an accuracy of **88%** on the test set. It performed particularly well in predicting **Class 1 (positive cases)** with high precision (0.90) and recall (0.89).
* **Class 0 (negative cases)** had slightly lower precision and recall but still maintained a good balance. The model made **12 false negatives** and **10 false positives**, indicating that it was effective at predicting the majority class without a significant number of misclassifications.
* The higher weights on **Random Forest** and **Gradient Boosting** models helped in boosting performance, as these models are generally more robust and tend to perform better on complex datasets compared to a **Decision Tree**.

**Conclusion**

**SVM:**

* The **SVM model with GridSearchCV** showed strong performance on both the training and test sets, with an optimal combination of hyperparameters (C = 1, gamma = 0.1, kernel = rbf).
* The **classification report** highlights the model’s excellent precision and recall for class 0 and solid performance for class 1.
* The model has an **accuracy of 90%** on the training set and **86%** on the test set, demonstrating good generalization capabilities.

Moving forward, we could explore further fine-tuning of the model or try different feature engineering methods to improve the performance, especially for class 1.

**Decision Tree Model**:

* The model was successfully tuned using **GridSearchCV**, which helped identify the optimal set of hyperparameters (criterion = 'gini', max\_depth = 4, min\_samples\_leaf = 4).
* **Performance**: The model achieved **87.5% accuracy** on the training set and **88% accuracy** on the test set, demonstrating good generalization. The classification metrics show a solid balance between precision and recall, particularly for class 1 (positive cases).
* **Feature Importance**: Key features like **ST\_Slope**, **ChestPainType**, and **Cholesterol** played a significant role in the model's decisions, while other features like **RestingBP** had little impact.

This Decision Tree model can be used effectively for classification tasks, but further tuning or the use of more complex models like Random Forests or Gradient Boosting may improve performance, especially if additional data or features are available.

**Random Forest (Bagging):**

* **Random Forest Model**: The Random Forest model was evaluated and showed excellent performance on the training set (100% accuracy) and good generalization on the test set (88.04% accuracy). The model performs well in distinguishing between the classes and has balanced precision and recall, especially for **Class 1 (positive cases)**.
* **Feature Importance**: The analysis of feature importance revealed that key health indicators, such as **ST\_Slope**, **Oldpeak**, and **Cholesterol**, play a significant role in the model's predictions. These insights can be valuable for understanding the most influential factors in the target variable.

While the model performed well on both training and test datasets, the perfect accuracy on the training set suggests that there might be some overfitting, which can be mitigated by tuning the model or using techniques like cross-validation and pruning.

**Gradient Boosting Model**:

* The **Gradient Boosting model** showed strong performance, achieving **94% accuracy** on the training set and **87.5% accuracy** on the test set. The model performed well in classifying both positive and negative cases, with slightly better precision and recall for positive cases.
* **Feature Importance**: The analysis revealed that **ST\_Slope** is the most significant feature, followed by **Oldpeak**, **Cholesterol**, **ChestPainType**, and **ExerciseAngina**, indicating the importance of health-related indicators in the model’s predictions.

The model demonstrated strong predictive power and could potentially be improved by further tuning parameters, especially in balancing the precision-recall tradeoff for both classes. Additionally, understanding the feature importance can guide further investigation into the most influential health factors for the target variable.

**Weighted Voting Classifier** :

The **Weighted Voting Classifier** provided excellent performance due to its ability to leverage the strengths of its base models by assigning appropriate weights. The model's simplicity and the strategic weight assignment allowed it to outperform other models in this scenario, producing an accuracy of **88%** on the test set and handling misclassifications effectively. This makes it a strong choice for situations where multiple models can be combined for better results.

**FURTHER IMPROVEMENT:**

In the medical field, accurate and reliable heart disease prediction models are crucial for early diagnosis and intervention, which can significantly improve patient outcomes. The models tested—SVM, Decision Tree, Random Forest, Gradient Boosting, and Weighted Voting—demonstrate the potential of machine learning techniques in identifying individuals at risk of heart disease. By leveraging key health indicators like ST\_Slope, Cholesterol, and Oldpeak, these models can assist healthcare professionals in making informed decisions about patient care, potentially reducing the number of undiagnosed or misdiagnosed cases.

To further improve these models in the medical context, several steps can be taken:

1. **Feature Engineering**: Incorporating more diverse and comprehensive health data, such as genetic factors, lifestyle information, and advanced diagnostic test results, can enhance model performance.
2. **Handling Class Imbalance**: For better precision, especially in identifying high-risk patients (class 1), techniques like SMOTE (Synthetic Minority Over-sampling Technique) or adjusting class weights can be applied to address any class imbalance.
3. **Cross-Validation and Hyperparameter Tuning**: To prevent overfitting and ensure generalization to unseen data, fine-tuning hyperparameters and using cross-validation will help improve the robustness of the models.
4. **Ensemble Methods**: Combining different models through ensemble methods like Weighted Voting, as demonstrated, can leverage the strengths of multiple classifiers and enhance the model’s overall reliability and predictive power.
5. **Interpretability**: Making the models more interpretable, especially for healthcare professionals, by explaining the key factors influencing predictions, could foster trust and aid in clinical decision-making.

By continuing to refine these models and incorporating more diverse data, machine learning can become a powerful tool in the early detection and management of heart disease, ultimately improving patient care and reducing healthcare costs.