Predicting Heart Disease Risk Using Clinical Data

Hanzhen Zhu
University of Michigan

Abstract

Heart disease is a major global health challenge, making early detection essential to save lives and reduce costs. Using the UCI Heart Disease dataset (n=303) with 14 clinical features, this study aimed to predict heart disease (0 = no, 1 = yes). After data preprocessing, we first implemented five machine learning models—Logistic Regression, Random Forest, XGBoost, K-Nearest Neighbors, and Support Vector Machines—optimized and evaluated through k-fold cross-validation. Random Forest and XGBoost performed well but struggled with edge cases and feature variability. To address this, we built a more complex hybrid model combining both Random Forest and XGBoost using a soft-voting classifier. The hybrid model achieved 90.16% accuracy, 93.33% precision, and 94.40% AUC-ROC, reducing false negatives and positives compared to individual models. Our research shows the potential of hybrid ensemble models in improving heart disease detection, enabling timely interventions, and saving lives.[1]

1 Introduction

According to the World Health Organization [2021], cardiovascular diseases (CVDs) are the leading cause of death worldwide, causing approximately 17.9 million deaths annually, which is 32% of all global deaths. In the United States, heart disease is the top cause of death across genders and most racial and ethnic groups, with one person dying every 33 seconds from it [Centers for Disease Control and Prevention, 2023]. Detecting heart disease early is vital because timely interventions can lower death rates and reduce healthcare costs. However, traditional diagnostic methods rely on clinical judgment and basic statistics, which may miss subtle but important patterns in patient data. Machine learning offers a way to improve diagnostic accuracy by uncovering complex relationships between clinical features, enabling more precise, data-driven healthcare decisions.

This study focuses on building a binary classification model to predict heart disease using the UCI Heart Disease dataset [Janosi and Detrano, 1989]. The dataset includes 14 clinical features, such as age, serum cholesterol, resting blood pressure, and maximum heart rate (continuous variables) and chest pain type, fasting blood sugar, and thalassemia status (categorical variables). Before modeling, we addressed missing values, standardized numerical features, and applied one-hot encoding for categorical variables. The model predicts heart disease as either 1 (present) or 0 (absent). We developed and fine-tuned several machine learning algorithms—Logistic Regression, Random Forests, XGBoost, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and a self-defined hybrid model—to identify the most effective approach for this task.

2 Related Work

In the previous research, heart disease prediction using machine learning methods can generally be classified into traditional models and ensemble models.

[1]The codes for this project are publicly available on GitHub: Heart-Disease-Prediction Repository

Preprint. Under review.

Traditional models, such as logistic regression and decision trees, often show different levels of accuracy depending on dataset characteristics and feature selection. For example, Rahim et al. [2021] applied logistic regression to the Framingham dataset, achieving 94.2% accuracy, which was better than decision trees (accuracy = 74.3%). This is likely because logistic regression can handle collinear features, such as overlapping clinical indicators, more effectively. On the other hand, Minou et al. [2020] found decision trees performed better (84.4%) than logistic regression (68.9%) on another dataset, possibly due to the decision tree's ability to capture non-linear relationships between variables. These differences show how the choice of model depends heavily on dataset characteristics, including the relationships between features and the distribution of target outcomes.

Ensemble models, like random forests and gradient boosting, combine multiple decision trees to improve predictive performance. For example, Bhatt et al. [2023] showed that random forests achieved 91.1% accuracy with a near-perfect Area Under the Curve (AUC) score. These models excel in handling complex interactions among features, such as how age, cholesterol levels, and chest pain type together influence heart disease risk. By using techniques like bagging and boosting, ensemble models reduce overfitting, making them particularly useful for medical datasets with high variability.

Despite these promising results, we cannot directly compare studies due to differences in datasets and experimental setups. For instance, datasets may vary in size, feature availability, or population demographics, which can significantly affect model performance. Therefore, in our study where we use the UCI Heart Disease dataset, we will compare our models based on baseline performance metrics provided by the dataset's source, showing XGBoost and logistic regression perform best, both having accuracy of 81.6% and precision of 83.2%.

3 Dataset and Features

The UCI Heart Disease Dataset [Janosi and Detrano, 1989] contains 303 patient records, each with 14 clinical features. The target variable is binary, where 1 represents the presence of heart disease, and 0 represents its absence. The dataset is slightly imbalanced, with 164 records (54%) indicating no heart disease and 139 records (46%) indicating heart disease, as shown in Appendix A.1. Only two categorical features have trivial missing values: `ca` (number of major vessels) has 4 missing entries, and `thal` (thalassemia status) has 2.

Data Splitting: The dataset was split into training (80%) and testing (20%) sets for model development. To make the most of this small dataset, we applied 5-fold cross-validation on the training set for hyperparameter tuning and to ensure reliable model evaluation.

Feature Overview: The dataset comprises 5 continuous variables (age, resting blood pressure, serum cholesterol, maximum heart rate, and ST depression) and 9 categorical variables (e.g., chest pain type, fasting blood sugar, resting ECG). For instance, chest pain type `cp` is categorized into typical angina, atypical angina, non-anginal pain, and asymptomatic, informing diagnostic details. These categories provide diagnostic insights into the severity and nature of a patient's symptoms.

Preprocessing: Data preprocessing has 3 main steps. First, to handle missing values in the two categorical variables `ca` (number of major vessels) and `thal` (thalassemia status), we used mode imputation, which replaces them with the most frequent category, so that we can maintain the overall distribution and minimize any bias introduced during model training. Next, continuous features were normalized to have a mean of 0 and a standard deviation of 1, reducing biases caused by different features' units. Lastly, categorical features were one-hot encoded into numerical indicators for model training. Table 1 shows a sample of the processed data:

Table 1: Sample Preprocessed Data

Age	Cholesterol	RestingBP	MaxHR	ChestPain_0	ChestPain_1	Target
57.0	1.23	-0.75	0.45	0	1	1
62.0	-0.25	1.12	-1.02	1	0	0

Correlation Analysis: Correlation analysis (Figure 1)between several features: `thalach` (maximum heart rate achieved) correlates positively with features like cp_4 (asymptomatic chest pain) (r = 0.59) and $slope_2$ (slope of peak ST segment) (r = 0.54), indicating interconnected diagnostic markers.

Conversely, `exang` (exercise-induced angina) and `oldpeak` (ST depression induced by exercise) are negatively correlated with `thalach` (maximum heart rate achieved), with coefficients of -0.44 and -0.51, respectively. These negative correlations align with clinical expectations because people who have exercise habits tend to have lower maximum heart rates.

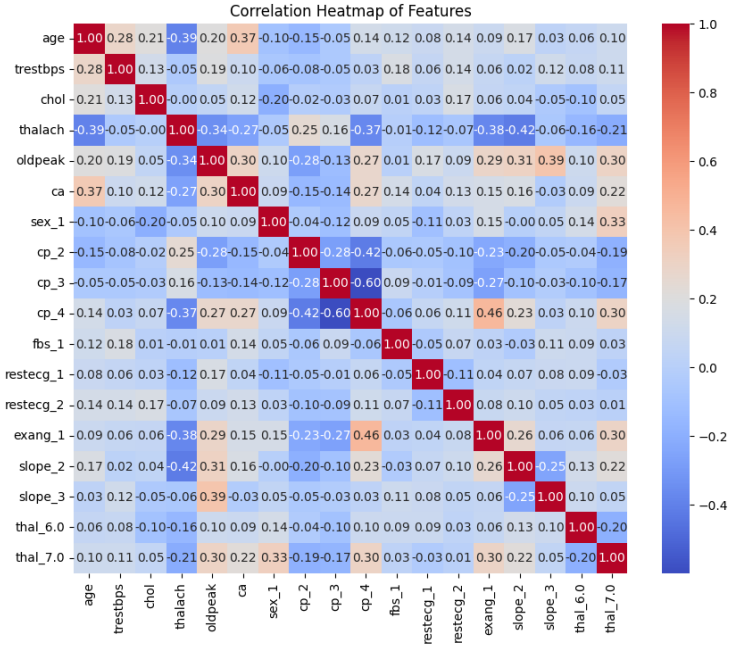

Figure 1: Correlation Heatmap of Features after Normalization

Distribution of Features: As shown in Appendix A.1, continuous features such as `age` (mean = 54) and maximum heart rate achieved (`thalach`, mean = 150 bpm) follow normal distributions, while resting blood pressure (`trestbps`), serum cholesterol (`chol`), and ST depression (`oldpeak`) are slightly skewed with outliers.

The relationships with the target variable (heart disease status) show clear trends. Among the **numerical** features (Figure 2), patients with heart disease generally have higher median values for `age`, `trestbps`, and `chol`, while `thalach` tends to be lower and `oldpeak` higher. For **categorical** features, heart disease cases are more common among males and individuals with asymptomatic chest pain (`cp` type 4), elevated fasting blood sugar (`fbs`), abnormal ECG results (`restecg`), and exercise-induced angina (`exang`). These trends align with clinical expectations, as shown in Figure 3.

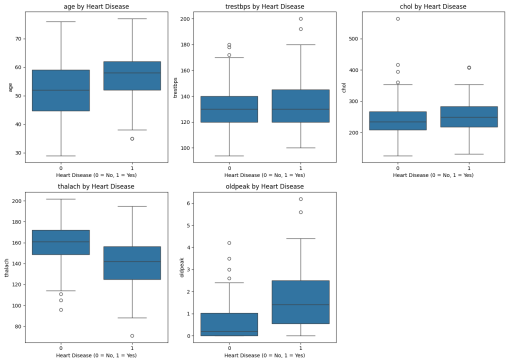

Figure 2: Distribution of Numerical Features in the Dataset by Heart Disease Status

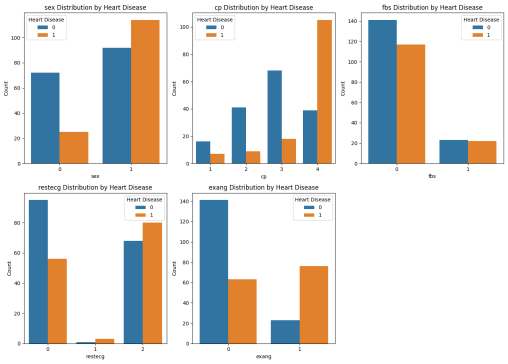

Figure 3: Distribution of Categorical Features in the Dataset by Heart Disease Status

This study explores the performance of five machine learning algorithms: Logistic Regression, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and XGBoost. Logistic Regression served as the baseline model for comparison. Each algorithm was implemented with hyperparameter tuning to optimize performance. The details are outlined below.

3.1 Logistic Regression

Logistic Regression models the probability $P(y = 1|x)$ using the logistic function:

$$P(y = 1|x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}}$$

where β_0 is the intercept, $\beta_1, \ldots, \beta_n$ are coefficients, and $x_1, \ldots, x_n$ are features. The coefficients are estimated using maximum likelihood estimation. Logistic Regression was used as the baseline model due to its simplicity and interpretability, with no additional hyperparameters tuned apart from regularization strength to prevent overfitting.

3.2 Random Forest

Random Forest is an ensemble learning method that builds multiple decision trees and combines their outputs through majority voting for classification. The hyperparameters tuned for this study included the number of trees (`n_estimators`), which balances model stability and computational cost, and the depth of each tree (`max_depth`), which prevents overfitting by limiting the complexity of the trees. Additionally, the minimum number of samples required to split a node (`min_samples_split`) and the minimum number of samples at a leaf node (`min_samples_leaf`) were adjusted to control overfitting. Lastly, the number of features considered for splitting (`max_features`) was fine-tuned to balance generalization and bias.

3.3 Support Vector Machine (SVM)

SVM constructs a hyperplane to separate classes in the feature space. For non-linearly separable data, kernel functions $K(x, x')$ project data into higher dimensions. The optimization problem is:

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$$

subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0.$

We tuned the regularization parameter (`C`) to balance maximizing the margin and minimizing classification errors. We choose the kernel (`kernel`) from linear, polynomial, RBF, and sigmoid options to handle linear or non-linear data separations. The kernel coefficient (`gamma`) was adjusted to set the influence of individual data points in non-linear models.

3.4 K-Nearest Neighbors (KNN)

KNN classifies a data point based on the majority class of its k-nearest neighbors in the feature space. The number of neighbors (`n_neighbors`) directly affects decision boundaries, with larger values leading to smoother boundaries. The weighting scheme (`weights`) was tuned to determine whether all neighbors contribute equally (uniform weighting) or whether closer neighbors have greater influence (distance-based weighting). Finally, the distance metric (`p`) was adjusted between Manhattan (1) and Euclidean (2) to capture similarity.

3.5 XGBoost

XGBoost is a gradient-boosting algorithm that sequentially builds decision trees to minimize the following objective function:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k),$$

where l is the loss function, $\hat{y}_i$ is the predicted value, and Ω regularizes tree complexity.

For XGBoost, we tuned the learning rate (`learning_rate`) to control the step size in gradient updates, ensuring stability. The tree depth (`max_depth`) was adjusted to balance model complexity and prevent overfitting. We also optimize the number of boosting rounds (`n_estimators`) to achieve high accuracy.

3.6 Hybrid Model (Random Forest + XGBoost)

From the preliminary results (Section 4), we created a hybrid model that combines the strengths of RF and XGBoost using a VotingClassifier with soft voting. This method leverages probabilistic predictions from both models to achieve balanced and robust performance.

4 Experiments and Results

This section includes our experimental setup, hyperparameter tuning, evaluation metrics, and model performance analysis. It also includes insights into the successes and failures of different models with data and visualizations.

4.1 Experimental Setup

Data Splitting: The UCI Heart Disease dataset was split into training and testing sets with an 80:20 ratio, where training set shape = (242, 18) and testing set shape = (61, 18).

$$\text{Accuracy} = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP + FP}} \quad \text{Recall} = \frac{\text{TP}}{\text{TP + FN}}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision + Recall}}$$

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR})$$

Here, TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives.

4.2 Hyperparameter Tuning

Before testing, we fine-tuned hyperparameters using `RandomizedSearchCV`, which efficiently searches a wide range of parameter combinations. To ensure robustness and minimize overfitting, we applied 5-fold cross-validation, splitting the training data into five subsets. Each subset was used as a validation set once, while the remaining four were used for training, ensuring all data contributed to both training and validation.

The primary evaluation metric was the Area Under the Receiver Operating Characteristic Curve (AUC-ROC). AUC-ROC is ideal for binary classification because it measures how well the model distinguishes between classes across all decision thresholds. This makes it particularly effective for imbalanced datasets, as it evaluates overall performance beyond accuracy, which can be misleading in such cases.

The parameter choices for each model were informed by prior research and empirical testing. We listed them as well as the final fine-tuned hyperparameters of each model in Appendix B.

4.3 Results of Individual Models

Table 3 summarizes the fine-tuned models' performance.

Random Forest (RF) achieved the best results among individual models, with 90.16% accuracy, 93.33% precision, and an AUC-ROC of 94.83%. XGBoost matched RF in accuracy (90.16%) and precision (93.33%) but had a slightly lower AUC-ROC of 93.97%. Both models showed strong reliability in distinguishing between cases with and without heart disease.

For the other models, K-Nearest Neighbors (KNN) achieved 88.52% accuracy, 90.32% precision, and an AUC-ROC of 92.89%. Its sensitivity to overlapping class distributions limits its use in

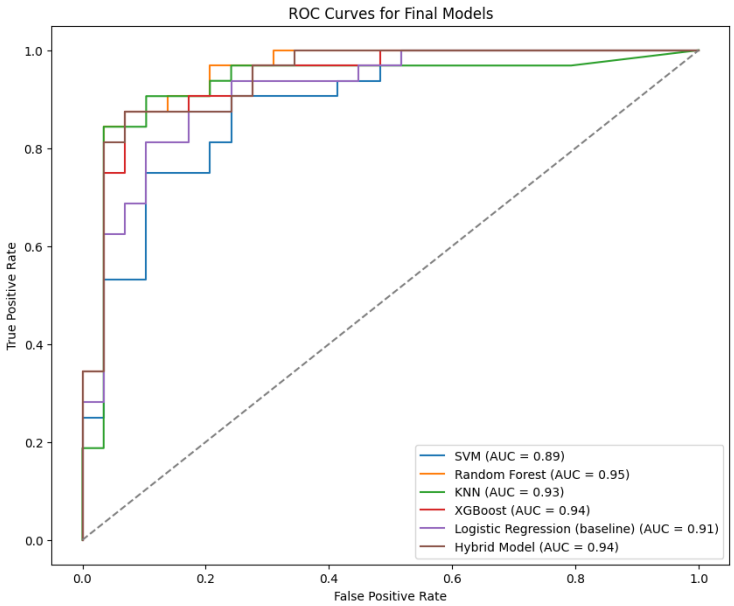

Figure 4: ROC Curves for Final Models

identifying borderline cases. Support Vector Machine (SVM) had the weakest performance, with 78.69% accuracy and an AUC-ROC of 88.79%. Logistic Regression served as a reliable benchmark with 83.61% accuracy and an AUC-ROC of 91.49%.

Figure 4 shows the Receiver Operating Characteristic (ROC) curves for all models, illustrating how well each model distinguishes between cases with and without heart disease.

Random Forest achieved the best performance across all metrics, because it addresses overfitting most effectively by regularizing parameters below. Below we explain the fine-tuned parameters with dataset-specific interpretations and their roles in reducing overfitting that make it stand out:

- `n_estimators=200`: 200 decision trees ensure the model can aggregate insights from multiple subsets of features, important for identifying nuanced patterns in heart disease indicators.
- `min_samples_split=10`: Each node requires at least 10 samples to split, reducing overfitting by ensuring splits in the heart disease dataset are based on meaningful patterns, not noise from rare feature combinations, like unusual chest pain types or ECG patterns.
- `max_features= 'log2'`: Limits the number of features at each split to about 4 ($\log_2(18) \approx 4.17$), preventing overfitting by avoiding reliance on a single dominant feature.

XGBoost also performed similarly well. Below are its fine-tuned parameters with interpretations:

- `subsample=0.6`: Uses 60% of the training data for each boosting round, introducing randomness to prevent overfitting. In the context of heart disease, this parameter ensures that the model generalizes well to unseen cases, particularly given the interpersonal variability in clinical indicators.
- `n_estimators=400`: Performs 400 boosting rounds, providing enough iterations to minimize the loss function. With this dataset, where some features (e.g., `ca` and `thal`) carry significant weight, the higher number of iterations allows the model to refine predictions while avoiding underfitting.

4.4 Failure Analysis

We analyzed the misclassifications by the Random Forest (RF) and XGBoost models to show each of their limitations in predicting heart disease.

The RF model misclassified 8 cases: 5 false negatives (heart disease cases predicted as no heart disease) and 3 false positives (no heart disease cases predicted as heart disease). A closer look at these cases shows that higher average values for the number of major vessels (`ca` = 0.875) and reversible thalassemia defects (`thal_7.0` = 0.625) were common among misclassified cases. These

features likely introduced variability or complexity that the model struggled to interpret accurately, highlighting the need for more nuanced handling of such clinical indicators.

XGBoost misclassified 11 cases: 8 false negatives and 3 false positives. Misclassified instances exhibited greater variability in features like `age` (average = -0.452, variation = 1.130), resting blood pressure (`trestbps`, average = 0.044, variation = 0.809), and cholesterol levels (`chol`, average = -0.240, variation = 1.022). This suggests that XGBoost had difficulty generalizing for patients with extreme or unusual values in these features, which could lead to uncertainty in predictions.

Overall, both models tend to misclassify cases where the values for `ca` and `thal_7.0` were higher. Additionally, XGBoost's misclassifications were more strongly associated with variability in `age`, `trestbps`, and `chol`.

4.5 Further Improvement: Hybrid Model (Random Forest + XGBoost)

Inspired by the limitations found in the failure analysis, we developed a hybrid model combining Random Forest and XGBoost using the `VotingClassifier` with soft voting. This approach leverages the strengths of both models to achieve balanced and robust performance.

We developed the hybrid model through the following steps: First, we defined a parameter grid for both Random Forest and XGBoost to tune their hyperparameters, which enables us to explore different combinations of parameters and identify the optimal settings for each model. Next, we created the hybrid model using `VotingClassifier`, combining Random Forest and XGBoost as base estimators. Then, we used `GridSearchCV` to perform an exhaustive search over the parameter grid. This approach allowed us to find the best combination of hyperparameters for the hybrid model by evaluating its performance through cross-validation. After identifying the best parameters, we trained the hybrid model on the training data. The model was then evaluated on the test data and evaluated with performance metrics, including Accuracy, Precision, Recall, F1 Score, and AUC ROC.

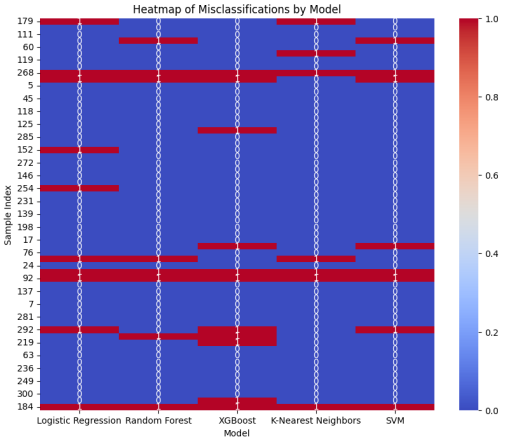

Figure 5: Heatmap of Misclassifications by Model

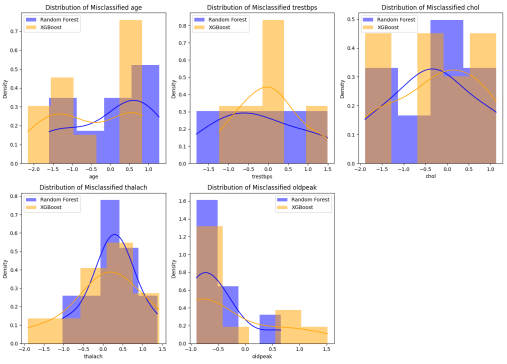

Figure 6: Distribution of Misclassification Features: Random Forest vs XGBoost

4.6 Overall Results

Among the individual models, Random Forest performed the best across all evaluation metrics. Its strength lies in capturing complex feature interactions in clinical data while effectively avoiding overfitting. XGBoost also showed strong performance with comparable accuracy, but its slightly lower robustness in handling edge cases was noted during the failure analysis.

To address the limitations observed in the failure analysis, we developed a hybrid model that combines the strengths of Random Forest and XGBoost. The hybrid model achieved the most balanced performance by reducing false negatives and false positives from each individual model. It leveraged Random Forest's ability to generalize well and XGBoost's gradient-boosting ability to

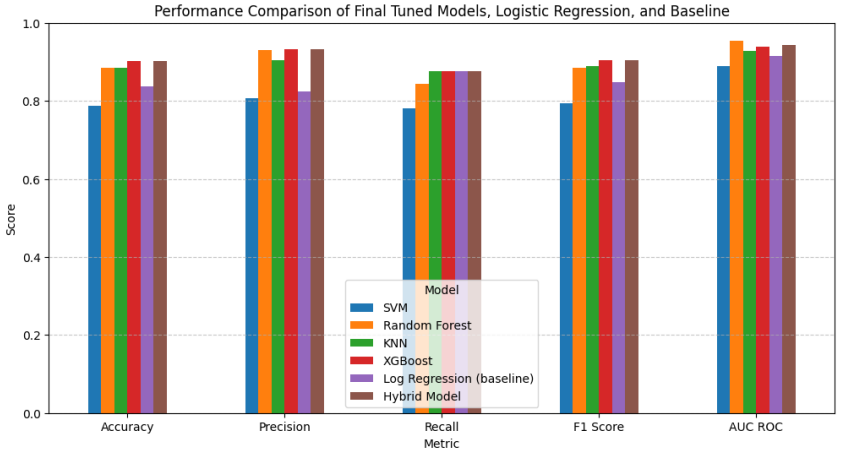

Figure 7: Performance Comparison of Final Tuned Models

handle complex data distributions. Therefore, we identify it as the most reliable model for predicting heart disease, especially for its performance stability across different datasets.

Table 3 in Appendix B summarizes the performance metrics of all models, which is also visualized in Figure 7.

5 Conclusion, Discussion, and Future Work

Our study developed and evaluated machine learning models to predict heart disease using the UCI Heart Disease dataset. While Random Forest and XGBoost emerged as strong individual performers, both models showed limitations in handling edge cases. To address these limitations, we developed a hybrid ensemble model combining Random Forest and XGBoost through a VotingClassifier with soft voting to leverage the strengths of both algorithms with the optimized balance. As a result, the hybrid model showed improved performance in reducing both false negatives and false positives. The results demonstrate the potential of **hybrid models** in clinical datasets, as they better handle complex patterns, collinear variables, and edge cases compared to single models, offering the most stable and reliable performance. One drawback of the hybrid model, however, is the increased training time compared to individual models.

Future research could focus on several improvements. First, to address the hybrid model's runtime issue, we could incorporate parallel processing, such as distributed libraries in TensorFlow, to optimize hyperparameter tuning within the VotingClassifier framework. Second, expanding the dataset to include diverse clinical populations would improve the model's generalization, particularly for underrepresented subgroups with different cardiovascular risk indicators. Third, incorporating advanced feature engineering, such as clinical risk scores or temporal patterns from patient histories, could add domain-specific insights to improve predictions. Finally, integrating recurrent neural networks (RNNs) could further capture patterns in sequential data, such as changes in blood pressure over time, which could help identify early warning signs or track heart disease progression.

Furthermore, to make this model scalable and suitable for real-world clinical use, we propose several ideas. First, incorporating interpretability tools such as SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) could make predictions more transparent and easier to trust, as they enable clinicians to see the factors influencing model decisions [Lundberg and Lee, 2017]. Second, embedding these models into wearable devices could enable continuous tracking of patient health, allowing for the early detection of heart anomalies and timely interventions [Hughes et al., 2023]. For example, detecting sharp changes in blood pressure could help prevent urgent illnesses like heart attacks. These improvements could pave the way for more scalable, efficient, and personalized healthcare delivery, addressing current healthcare challenges such as unequal access, rising costs, and resource shortages, ultimately fostering a healthier population.

References

Chintan M. Bhatt, Parth Patel, Tarang Ghetia, and Pier Luigi Mazzeo. Effective heart disease prediction using machine learning techniques. *Algorithms*, 16:88, 2023.

Centers for Disease Control and Prevention. Heart disease facts, 2023. URL `https://www.cdc.gov/heartdisease/facts.htm`. Accessed: 2024-11-16.

Andrew Hughes, Md Mobashir Hasan Shandhi, Hiral Master, Jessilyn Dunn, and Evan Brittain. Wearable devices in cardiovascular medicine. *Circulation Research*, 132(5):652–670, 2023. doi: 10.1161/CIRCRESAHA.122.322389. URL `https://www.ahajournals.org/doi/abs/10.1161/CIRCRESAHA.122.322389`.

Steinbrunn William Pfisterer Matthias Janosi, Andras and Robert Detrano. Heart Disease. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C52P4X.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017. URL `http://arxiv.org/abs/1705.07874`.

John Minou, John Mantas, Flora Malamateniou, and Daphne Kaitelidou. Classification techniques for cardio-vascular diseases using supervised machine learning. *Medical Archives*, 74(1):39, 2020.

Aqsa Rahim, Yawar Rasheed, Farooque Azam, Muhammad Waseem Anwar, Muhammad Abdul Rahim, and Abdul Wahab Muzaffar. An integrated machine learning framework for effective prediction of cardiovascular diseases. *IEEE Access*, 9, 2021.

World Health Organization. Cardiovascular diseases (cvds), 2021. URL `https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)`. Accessed: 2024-11-16.

Appendix

A Figures and Graphs

A.1 Additional Data Analysis Plots

The target variable is imbalanced, with 164 cases (54%) labeled as “No Heart Disease" and 139 cases (46%) labeled as “Heart Disease."

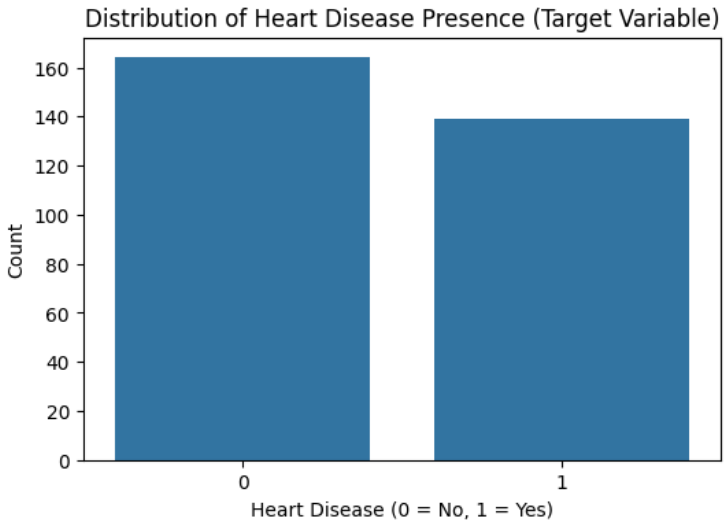

Figure 8: Target Variable Distribution

Distribution of Numerical Features:

- *Age:* Normally distributed with a mean of 54 years, reflecting a varied age range.
- *Resting Blood Pressure (trestbps):* Slightly right-skewed, showing some high outliers.
- *Serum Cholesterol (chol):* Right-skewed with visible outliers, indicating occasional very high levels.
- *Maximum Heart Rate (thalach):* Normally distributed with a mean of 150 bpm, showing typical cardiovascular function.
- *ST Depression (oldpeak):* Right-skewed with a mean of 1.04, showing variation in stress-induced responses.

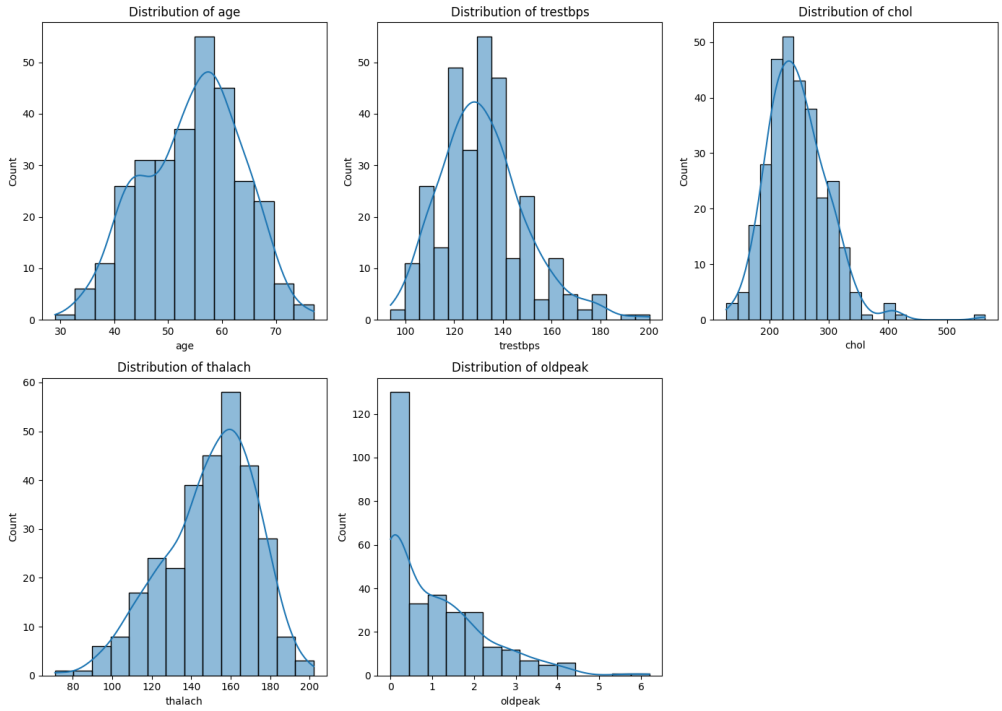

Figure 9: Distribution of Numerical Features

Feature Pair Plots:

- *Thalach* (maximum heart rate) and *oldpeak* (ST depression) show clear separations between heart disease cases and non-cases, making them strong predictors individually.
- *Age* and *chol* (serum cholesterol) have overlapping values across classes but may provide valuable insights when considered together with other features.
- Relationships between other numerical features, such as *trestbps* (resting blood pressure), indicate these subtle patterns can contribute to model's prediction.

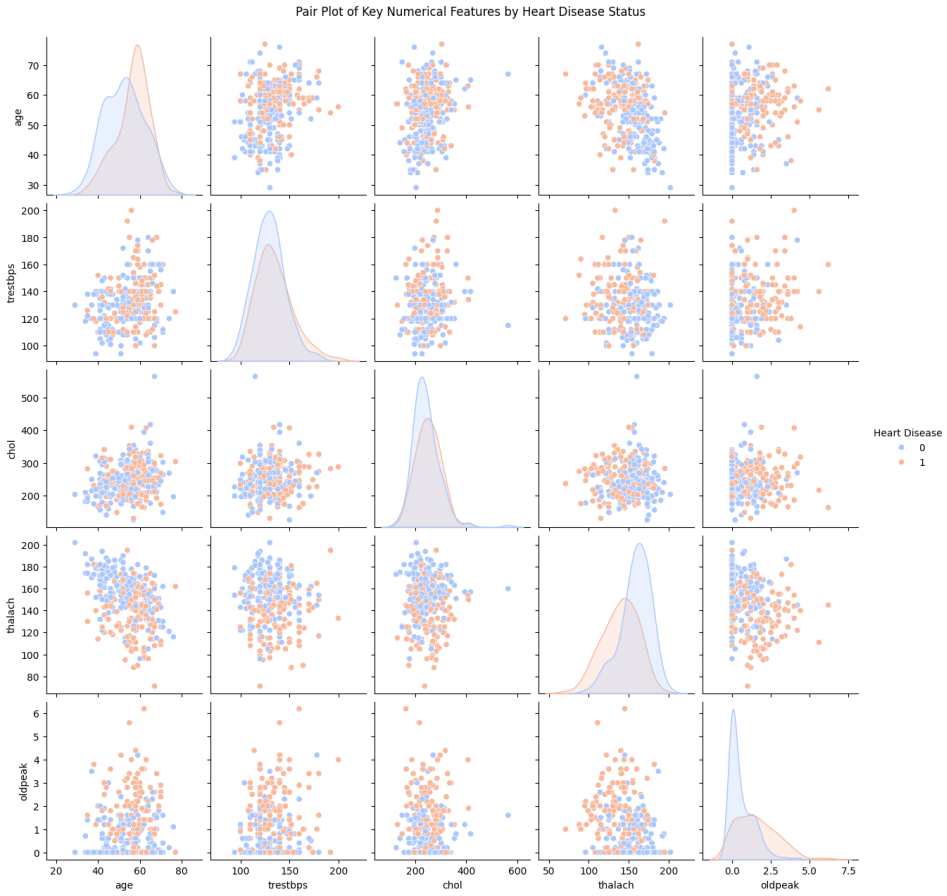

Figure 10: Pair Plot of Numerical Features

A.2 Bias-Variance Tradeoff: Randomized Search on Each Model

This section shows AUC-ROC scores over 10 iterations from the randomized hyperparameter search, comparing training and testing performance to assess the bias-variance tradeoff for each model.

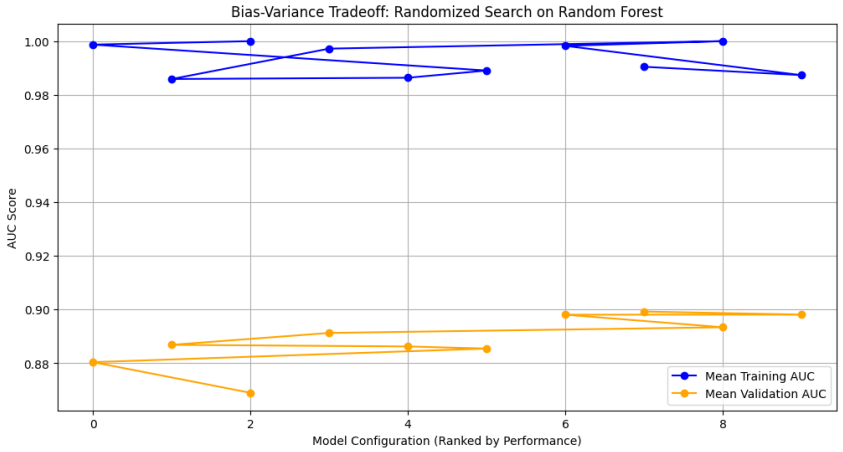

Figure 11: Randomized Search for RF: AUC-ROC over 10 iterations, training vs testing

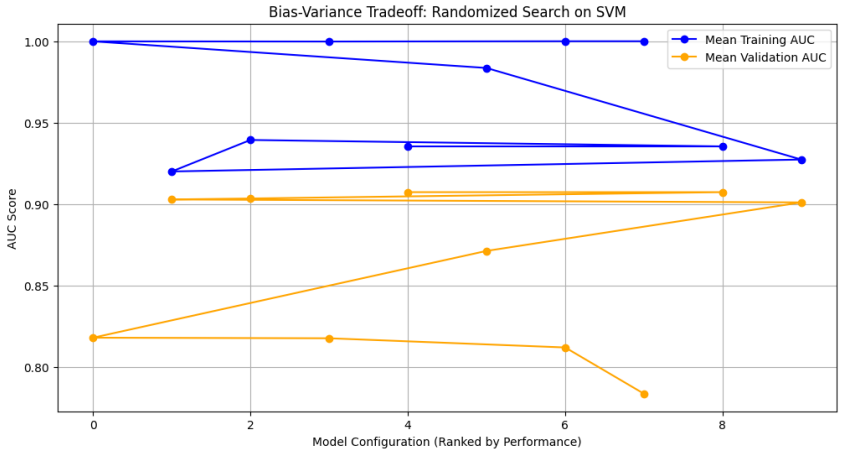

Figure 12: Randomized Search for SVM: AUC-ROC over 10 iterations, training vs testing

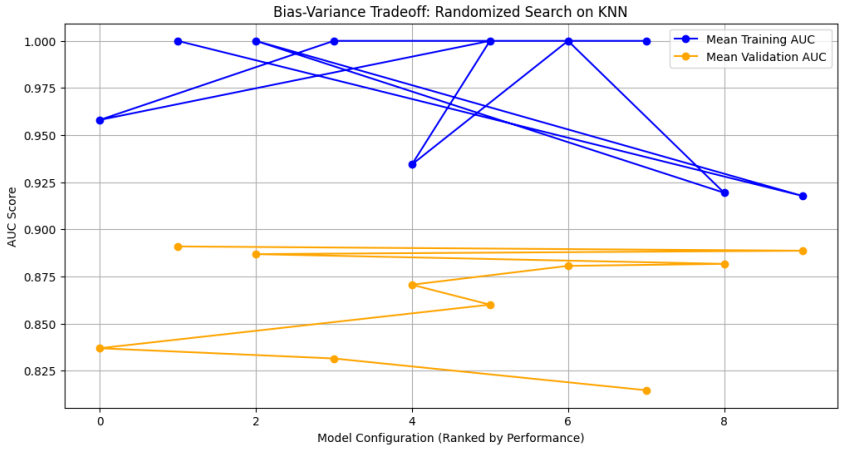

Figure 13: Randomized Search for KNN: AUC-ROC over 10 iterations, training vs testing

B Hyperparameter Tuning Details

This section details the RandomizedSearchCV parameter grid for each model in the hyperparameter tuning process:

Random Forest: These settings fine-tune the model's complexity, especially for small datasets.

- `n_estimators`: [50, 100, 200, 400]. More trees improve stability but make the model slower.
- `max_depth`: [10, 20, 30, 40, None]. Deeper depth increases complexity, more likely to overfit.
- `min_samples_split`: [2, 5, 10]. Smaller numbers allow more splits, improving detail in splits.
- `min_samples_leaf`: [1, 2, 5]. Larger values reduce overfitting by ensuring more data in leaves.
- `max_features`: ['sqrt', 'log2', None]. Fewer features at each split helps improve generalization.

Support Vector Machine (SVM): These settings provide flexibility to handle both linear and non-linear data separations.

- `C`: [0.1, 1, 10, 100]. Balances the trade-off between keeping a wide margin and minimizing errors.
- `kernel`: ['linear', 'rbf', 'poly', 'sigmoid']. Decides how the data is transformed for separation.

- `gamma`: ['scale', 'auto', 0.01, 0.1]. Adjusts how much influence a single data point has on the decision boundary.

K-Nearest Neighbors (KNN): These settings adjust the balance between neighborhood size and how features are weighted.

- `n_neighbors`: [5, 10, 15, 20]. Controls the number of nearby points used for deciding the output.
- `weights`: ['uniform', 'distance']. Distance-based weighting gives closer points more influence.
- `p`: [1, 2]. Manhattan (1) and Euclidean (2) distances determine how similarity is calculated.

XGBoost: These settings ensure the model remains flexible and avoids overfitting.

- `n_estimators`: [100, 200, 300, 400]. More boosting rounds improve accuracy but reduce speed.
- `learning_rate`: [0.01, 0.05, 0.1]. Controls the step size for each update in training.
- `max_depth`: [5, 10, 15]. Deeper trees find more patterns but increase the risk of overfitting.
- `subsample`: [0.6, 0.8, 1.0]. Limits data to be used for training each tree to reduce overfitting.
- `colsample_bytree`: [0.6, 0.8, 1.0]. Restricts the number of features considered per tree to improve generalization.

Table 2: Fine-tuned Hyperparameters and AUC-ROC Scores

Model	**Hyperparameters**	**AUC-ROC**
Logistic Regression	`C=1, penalty='l2'`	0.9149
Random Forest	`n_estimators=200, max_depth=40, max_features='log2'`	0.9483
SVM	`C=1, kernel='linear', gamma='scale'`	0.8879
KNN	`n_neighbors=11, weights='distance', p=1`	0.9289
XGBoost	`n_estimators=400, learning_rate=0.01, max_depth=10`	0.9397
Hybrid Model	`Soft Voting, RF and XGBoost parameters as above`	0.9440

Table 3: Performance Metrics of Fine-Tuned Models

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Logistic Regression	0.8361	0.8235	0.8750	0.8485	0.9149
Random Forest	0.9016	0.9333	0.8750	0.9032	0.9483
SVM	0.7869	0.8065	0.7812	0.7937	0.8879
KNN	0.8852	0.9032	0.8750	0.8889	0.9289
XGBoost	0.9016	0.9333	0.8750	0.9032	0.9397
Hybrid Model (RF+XGBoost)	0.9016	0.9333	0.8750	0.9032	0.9440