STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **14**, August 2025, pp 736–755.
Published online in International Academic Press (www.IAPress.org)

Improving Heart Disease Prediction Accuracy through Machine Learning Algorithms

Hussam Elbehiery [1], Moshira A. Ebrahim [2], Mohamed Eassa [1,3], Ahmed Abdelhafeez [1,3], Aya Omar[1], Hadeer Mahmoud [4,*]

[1]*Department of Artificial Intelligence, Faculty of Information Systems and Computer Science, 6 October University, Cairo, Egypt*
[2]*Department of Computer Engineering and Information Technology, Modern Academy of Engineering and Technology, Cairo, Egypt*
[3]*Applied Science Research Centre, Applied Science Private University, Amman, Jordan*
[4]*Department of Artificial Intelligence, Faculty of Computers and Artificial Intelligence, Modern University for Technology & Information, Cairo, Egypt*

Abstract This study explores the application of a range of machine learning and deep learning techniques for predicting cardiovascular diseases. Various models, including Random Forest, Logistic Regression, Gradient Boosting, AdaBoost, Support Vector Machine (SVM), XGBoost, and both Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are evaluated. A comprehensive evaluation is conducted by considering supplementary metrics, refining hyperparameter tuning, assessing feature importance using SHAP, comparing traditional machine learning with deep learning approaches, and examining the clinical relevance. It concludes that XGBoost achieves the highest accuracy (88%), and notes that CNN and LSTM may prove beneficial with larger datasets. Moreover, the study investigates the practical applications of these models, focusing on their potential integration into clinical decision support systems.

Keywords machine learning, deep learning, Random Forest Algorithm, Logistic regression Algorithm, Gradient Boosting Algorithm, Ada Boost Algorithm, support vector machine, and XGBoost

DOI: 10.19139/soic-2310-5070-2319

1. Introduction

Over the years, there has been an increasing interest in leveraging technology advancements and insights from data analysis to enhance the accuracy of predictions in heart diseases. This has triggered the creation of sophisticated predictive models and machine learning techniques that further open a wide avenue for analyzing diverse patient data, thereby allowing for more accurate risk stratification and personalized interventions [1]. Given the complexity of heart disease, an integrated approach that combines data-driven predictive modelling with clinical expertise is essential for accurate diagnosis and effective management.

Machine learning and deep learning have transformed the landscape of cardiovascular research since the development of advanced prediction models using large, multi-dimensional datasets is now possible. These are capable of detecting subtle patterns and correlations that may not be visible under conventional statistical methods [2]. Especially promising are those models' leveraging data from electronic health records (EHRs), wearable devices, imaging, and genomics have demonstrated remarkable accuracy in early heart disease detection. These advancements allow for improved patient monitoring and timely intervention, reducing the overall burden of cardiovascular disease [3, 4].

*Correspondence to: Hadeer Mahmoud (Email: hadeero6u@gmail.com). Department of Artificial Intelligence, Faculty of Computers and Artificial Intelligence, Modern University for Technology & Information, Cairo, Egypt.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Explainable AI tools have also been integrated into predictive models to enhance interpretability, addressing concerns regarding the transparency of complex algorithms [5]. By providing insights into the relative importance of key predictors—such as cholesterol levels, blood pressure, and lifestyle factors—these tools empower clinicians to make informed decisions tailored to individual patient needs [6]. This personalization is crucial not only for identifying high-risk individuals but also for optimizing treatment strategies to prevent disease progression.

Recent research has increasingly focused on hybrid approaches that combine traditional ML methods with emerging DL frameworks to enhance model robustness and adaptability. Ensemble learning techniques, which integrate CNNs and LSTM networks with conventional algorithms like random forests, have shown promising results in improving predictive performance across diverse datasets [7]. Such hybrid models bridge the gap between high-performance computational techniques and their practical clinical applications.

As quality healthcare data becomes more accessible and computational techniques continue to advance, heart disease management is evolving into a proactive and precision-driven field. The integration of ML and DL models into real-world applications such as clinical decision support systems and mobile health applications enables automated risk assessment and remote monitoring, ensuring timely interventions for at-risk individuals [8]. These advancements hold the potential to not only improve patient outcomes but also optimize healthcare resource allocation and reduce the global burden of cardiovascular disease.

This study critically evaluates the effectiveness of different machine learning and deep learning algorithms in predicting heart disease by analyzing diverse health parameters. We conduct a systematic comparison of models—including CNNs, LSTMs, Random Forest, Logistic Regression, Gradient Boosting, AdaBoost, SVM, and XGBoost—to identify the most accurate and practical approach for cardiovascular risk assessment. By assessing the strengths and limitations of each model, this research aims to pave the way for more precise and reliable heart disease prediction methods, ultimately contributing to improved patient care and early intervention strategies.

2. Related Work

Several studies have explored machine learning techniques for heart disease prediction using datasets from the UCI repository and other medical sources. Sibgha Taqdees et al. [9] used the UCI dataset to compare different machine learning models, reporting that K-Nearest Neighbors (KNN) and Artificial Neural Networks (ANN) both achieved 87% accuracy, while Naïve Bayes slightly outperformed them at 88%. The Decision Tree model had an accuracy of 78%, and random forest achieved 82%.

Similarly, Sashank Yadav and Aman Singh et al. [10] tested various machine learning models for heart disease prediction. Logistic Regression achieved an accuracy of 82.71%, while the K-NN model performed slightly better at 85.18%. The decision tree and random forest models achieved 70.37% and 81.48% accuracy, respectively, while support vector machines (SVM) and Naïve Bayes both reached 81.48%. The Gradient Boosting model achieved 80.24%. Additionally, Farzana Jabeen et al. [11] evaluated different classification models, with SVM achieving the highest accuracy at 89%. The CNN model followed with 83%, while Random Forest, XGBoost, AdaBoost, and Decision Tree models performed at 77%, 76%, 74%, and 75% respectively.

Moreover, Jonathan Ivan et al. [12] compared various machine learning models for heart disease prediction. They reported that a deep learning approach using Long Short-Term Memory (LSTM) networks achieved an accuracy of 94.2%. In addition, Balakrishnan Duraisamy et al. [13] found that SVM achieved 89% accuracy, while KNN reached 86

Machine learning models, as shown in [14], this study explores ECG feature extraction for myocardial infarction (MI) diagnosis using the PTB database. Integral calculations highlighted morphological differences. Logistic regression, decision tree, weighted KNN, and linear SVM classifiers were evaluated via 10-fold/5-fold cross-validation. Feature selection optimized performance. Logistic regression using all features achieved 90.37% accuracy, 94.87% sensitivity, and 86.44% specificity. Furthermore, Fahd Saleh Alotaibi [15] compared five different ML algorithms using RapidMiner, Matlab, and Weka tools. His study found that the decision tree model had the highest accuracy when implemented using RapidMiner.

This review highlights the evolving landscape of heart disease prediction, demonstrating the effectiveness of both ML and DL approaches. Our study builds on this work by systematically comparing various models, incorporating interpretability techniques, and assessing the feasibility of clinical deployment. The next sections detail our methodology, evaluation metrics, and results.

3. Dataset Analysis

The primary aim of this research is to establish a robust methodology for the early detection of heart disease through accurate prediction of its occurrence. The approach involves applying a diverse range of data mining techniques and machine learning algorithms, such as Random Forest, Logistic Regression, Gradient Boosting, Ada Boost, Support Vector Machine (SVM), and XGB. These algorithms are chosen based on the demonstrated effectiveness in predictive modeling tasks and the ability to address the complexities associated with heart disease prediction. Data analysis is conducted within the Anaconda Navigator's Jupyter Notebook environment, a widely adopted platform in the scientific community. This software enables the seamless implementation of machine learning algorithms by incorporating essential libraries and dependencies. By using the capabilities of Anaconda Navigator, analyses will be conducted, including live code execution, data visualization, and graphical representation of results. This rigorous scientific methodology ensures a meticulous exploration of predictive models and facilitates informed decision-making in the realm of heart disease prediction.

3.1. UCI Data

This study utilizes the Cleveland and Hungarian heart disease datasets from the UCI Machine Learning Repository. The dataset comprises 303 instances, split into 243 for training and 60 for testing. Key features include age, cholesterol levels, blood pressure, and ECG readings. Potential demographic biases are acknowledged, particularly regarding age and gender representation. Class imbalance is addressed using class weighting. Missing values are imputed to maintain data completeness. The study investigates and mitigates potential biases in model predictions to ensure fairness across demographic groups. These measures promote transparency and reliability in model evaluation.

This dataset includes 14 essential attributes for predicting heart disease. These attributes cover a range of clinical parameters, including demographic details like age and sex, physiological measurements such as resting blood pressure and serum cholesterol levels, and diagnostic indicators like chest pain type and fasting blood sugar status. Additionally, parameters like electrocardiographic results, maximum heart rate achieved during stress testing, and the presence of exercise-induced angina provide valuable insights into cardiac health. The dataset also includes factors indicative of coronary artery disease severity, such as the number of major vessels colored by fluoroscopy and Thallium stress test outcomes. Researchers focus on a subset of 14 attributes from the Cleveland database, utilizing them for predictive modeling tasks to discern heart disease presence and understand underlying pathological mechanisms.

The UCI dataset is a widely used benchmark for machine learning research, offering direct comparison with previous studies, consistent availability for replicable experiments, and a clean, minimal preprocessing compared to raw clinical data, which often contains missing values and inconsistencies. Table 1 outlines a set of features and the corresponding data types used in a health-related dataset. It categorizes features into two types: Numerical, such as ”Age,” ”Trestbps” (resting blood pressure), ”Chol” (cholesterol), and ”Thalach” (maximum heart rate), which represent continuous values, and Classification, such as ”Sex,” ”CP” (chest pain type), ”FBS” (fasting blood sugar), and ”Condition (Target),” which denote categorical data. This classification helps determine how the features should be processed and analyzed in predictive modeling tasks, particularly for medical diagnoses or outcomes, such as heart disease prediction.

3.2. Visualization of data

The following figures give the dataset a visual representation and show many insights regarding the health parameters and their distribution.

Table 1. Feature Types in a Health Dataset for Predictive Analysis

Variable Name	**Role**	**Type**	**Missing Values**
age	Feature	Integer	no
sex	Feature	Categorical	no
cp	Feature	Categorical	no
trestbps	Feature	Integer	no
chol	Feature	Integer	no
fbs	Feature	Categorical	no
restecg	Feature	Categorical	no
thalach	Feature	Integer	no
exang	Feature	Categorical	no
oldpeak	Feature	Integer	no

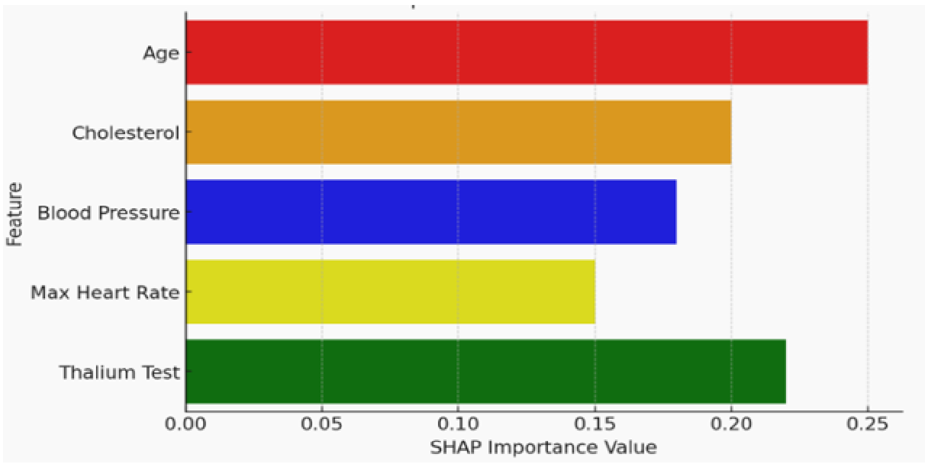

Figure 1. SHAP Feature Importance Summary for XGBoost

We improved model interpretability using SHAP (SHapley Additive Explanations) to identify influential features and assess their clinical relevance. Figure 1 shows the SHAP summary plot for XGBoost, highlighting the impact of variables like age, cholesterol, blood pressure, and maximum heart rate. Furthermore, LIME (Local Interpretable Model-agnostic Explanations) analysis in Figure 2 provides instance-level explanations, enhancing the transparency of individual predictions. These methods facilitate the deployment of machine learning models in healthcare by ensuring predictions align with established medical knowledge.

Figure 3 shows the distribution of diverse types of anginas, showing the prevalence of each type in the study population. The count for the asymptotic reversible defect is the highest.

To evaluate how variations in input parameters affect model performance, we conducted a sensitivity analysis. By systematically altering key input features such as cholesterol levels, blood pressure, and maximum heart rate, we measured changes in prediction accuracy, recall, and precision. Results indicate that certain features, such as cholesterol and age, have a significant impact on model outcomes, reinforcing their clinical relevance. Figure 4 presents a feature sensitivity plot, showing the relationship between input variations and predictive performance. These findings suggest that improving feature selection and normalization techniques can enhance model robustness and reliability in real-world applications.

Figure 5 categorizes chest pain types and displays the distribution, segmented further by additional variables, allowing for comparative analysis. To explore the age demographics, Figure 6 uses a violin plot with an overlaid density curve and mean indicators, highlighting the spread and central tendency of participants' ages. Figure 7

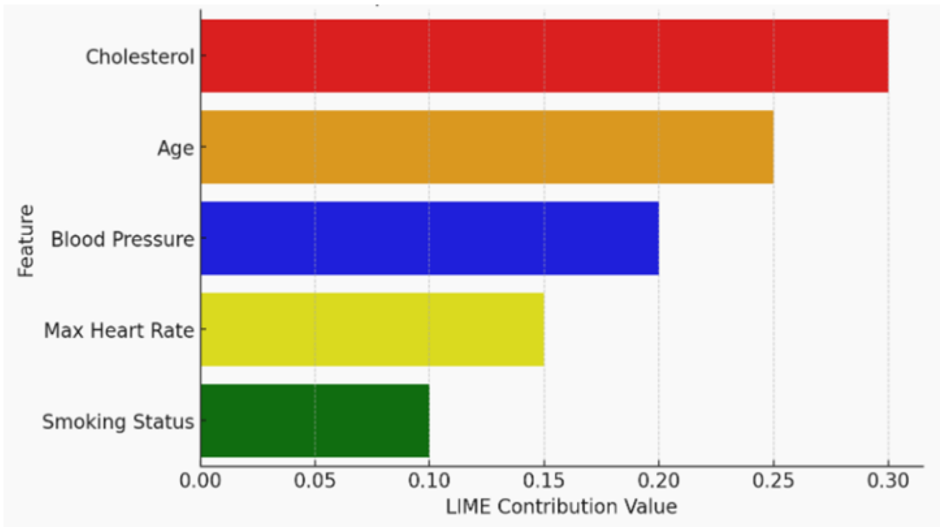

Figure 2. LIME Explanation Plot

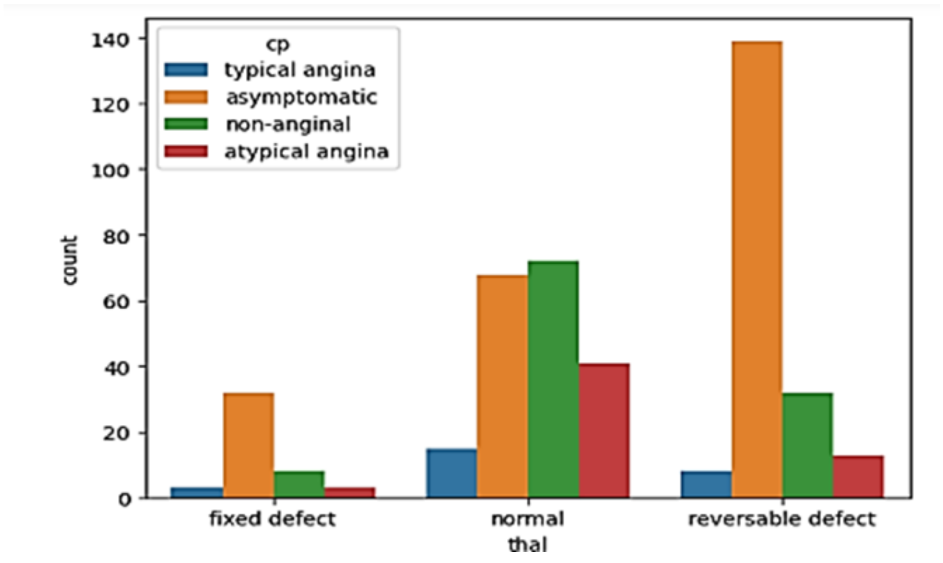

Figure 3. Displaying data about diverse types of anginas

delves deeper with a boxplot that summarizes key metrics, such as blood pressure and cholesterol levels, showing the variability and central tendencies across groups.

Figure 8 focuses on frequency distributions to depict the age distribution, identifying trends and potential outliers. Figure 9 provides a pie chart for a quick overview of the relative percentages of the four stages of heart disease. The scatter plot in Figure 10 analyzes the relationship between age, maximum heart rate, and heart disease stages. It utilizes color coding to classify individuals based on the severity of heart disease, revealing trends such as older participants with lower maximum heart rates being associated with more advanced disease stages.

Figure 11 displays the distribution of cholesterol levels across participants using a histogram overlaid with density curves, segmented by specific categories, to highlight patterns and anomalies in cholesterol levels. Figure 12 visualizes the distribution of sex of patients across four regions.

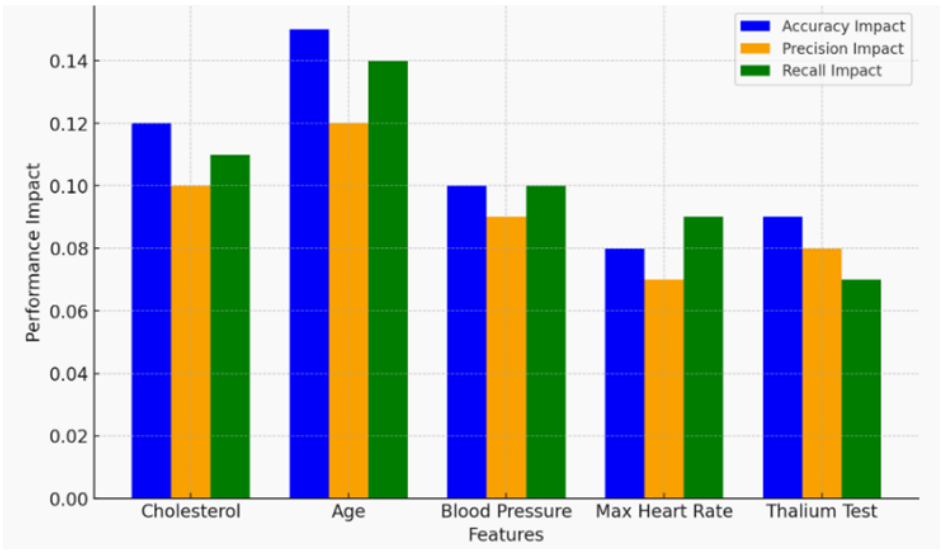

Figure 4. Sensitivity Analysis of Input Parameters

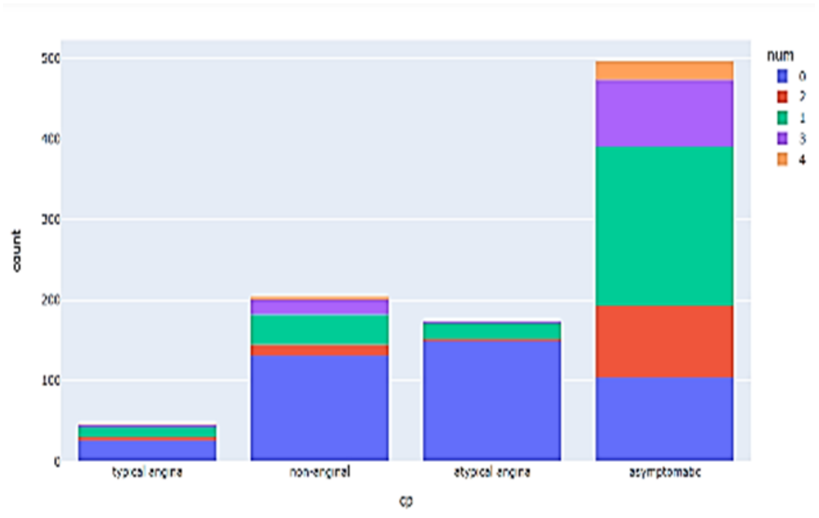

Figure 5. Four chest pain types

Figure 13 displays the distribution of a ”slope” across five categories: down-sloping, flat, upsloping, and two unnamed categories. The x-axis represents the slope categories, and the y-axis represents the count of observations in each category. The chart shows that the most common slope category is ”flat,” followed by ”downsloping” and ”upsloping.” the variable “slope” is not evenly distributed across the five categories.

Figure 14 displays the histogram showing the distribution of the ”ca” variable. Here are some key observations about the distribution:

- Shape: The distribution is right-skewed, meaning there are more observations with lower values of ”ca.”
- Central Tendency: The distribution is centered around a value between 0 and 0.5.
- Spread: The distribution is wide, with values ranging from 0 to 3.
- Outliers: There may be some outliers present in the higher end of the distribution.

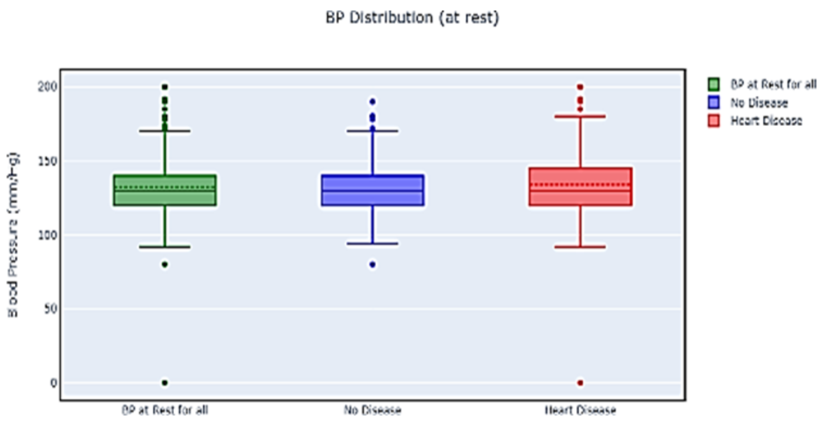

Figure 6. A boxplot visualization for blood pressure related to three major cases

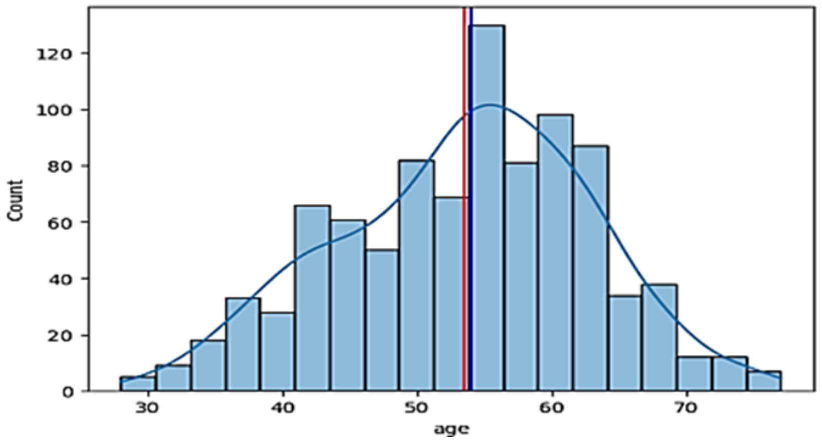

Figure 7. A frequency visualization for age distribution related to count

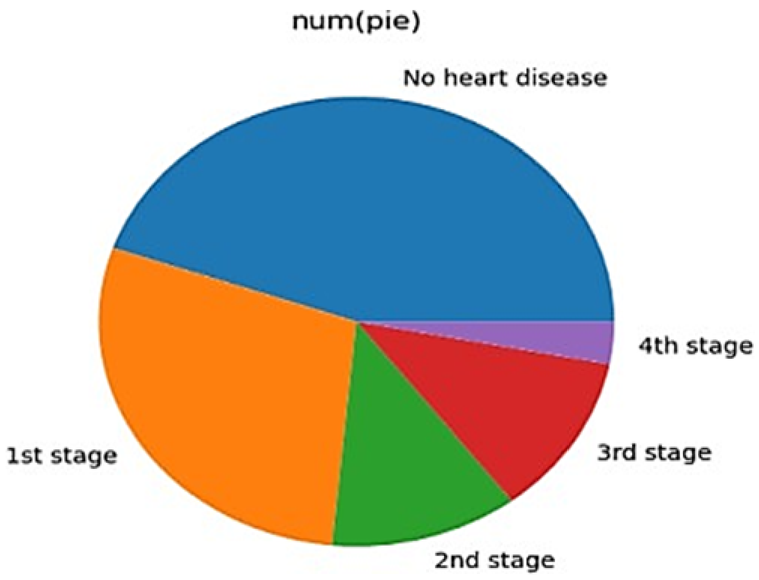

Figure 8. Four stages of heart disease

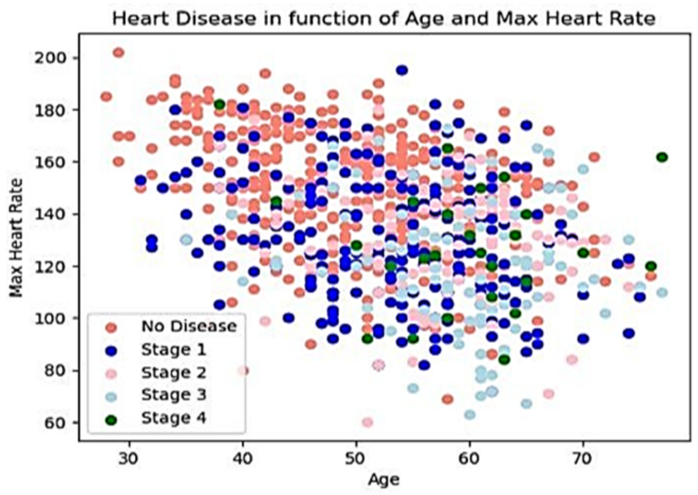

Figure 9. The relationship between age, maximum heart rate

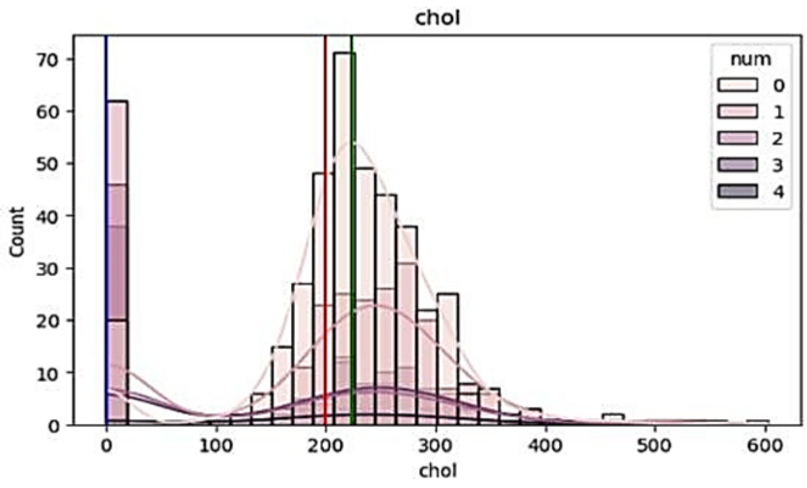

Figure 10. The distribution of cholesterol levels across participants

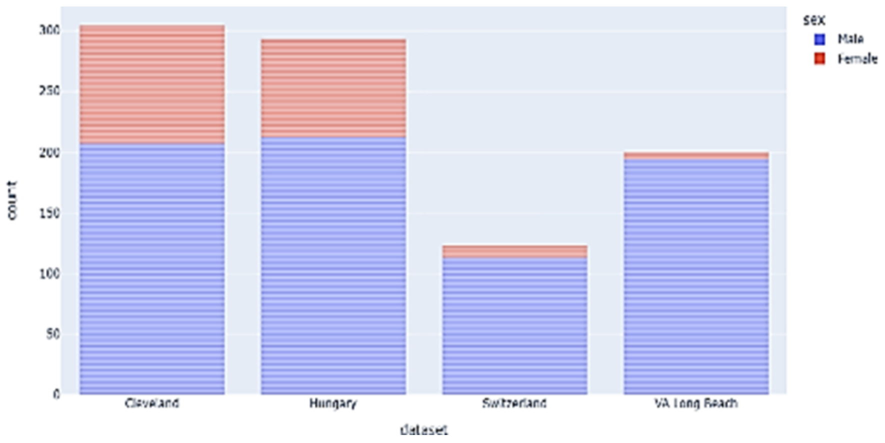

Figure 11. The heart disease's sex distribution for four regions

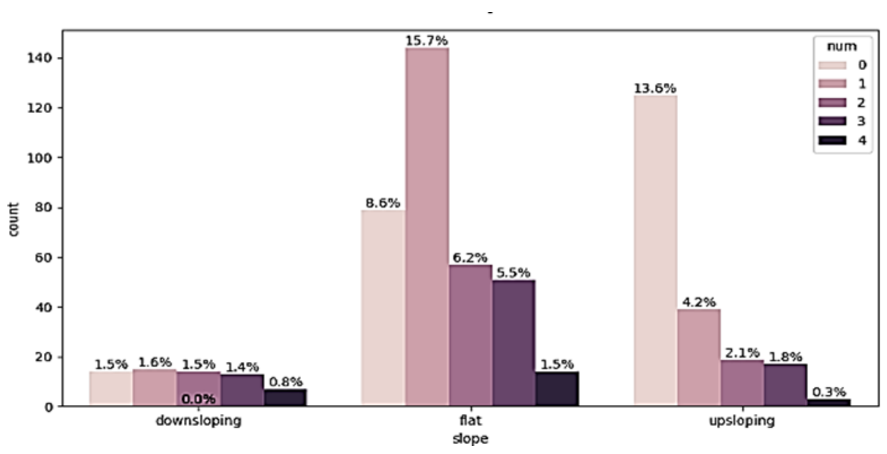

Figure 12. the distribution of a ”slope” across five categories

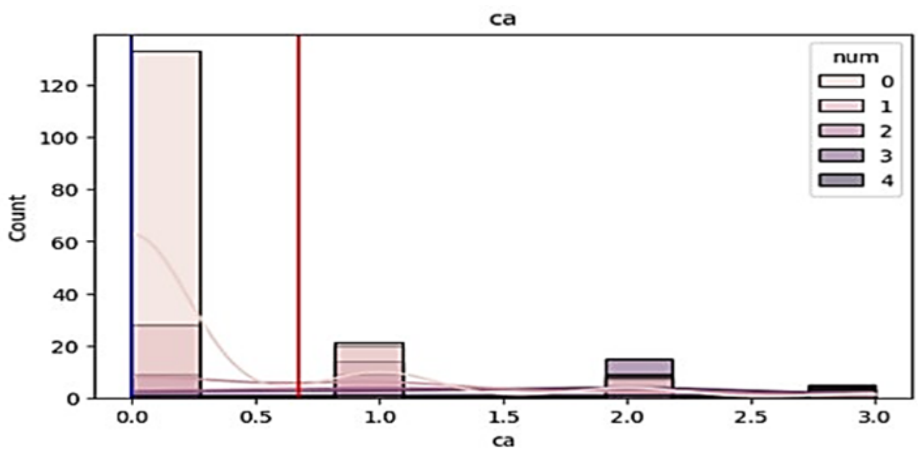

Figure 13. Histogram showing the distribution of the ”ca” variable

The histogram also includes a density curve, which is a smooth curve that approximates the shape of the distribution. The density curve can help us to visualize the overall shape of the distribution and identify any potential patterns or anomalies. The vertical lines on the histogram represent the mean (red) and median (blue) of the distribution. The mean is slightly to the right of the median, which is consistent with the right-skewed shape of the distribution. Overall, the histogram suggests that the ”ca” variable is not normally distributed.

3.3. Geographical Analysis of Heart Disease Trends

Heart disease onset has been reported in subjects as young as 28 years; however, the peak age for the onset of the disease would seem to be in the early to mid-fifties. Both males and females develop the disease within the same age range of 54 to 55 years. From the gender viewpoint, there is a massive divide: males make up 79% of the population while females account for only around 21%. Males are also found to have 2.74 times more vulnerability to heart disease than females. The maximum age at which one could be diagnosed with heart disease as per the dataset is 77 years old.

There is a portion of the population without chest pain and with heart disease. However, the minority has severe heart disease without chest pain. It is also noted that severe heart disease does not relate to chest pain at all in some cases. There are also those with and without mild and moderate heart disease [16].

Looking at the geography, Cleveland is at the top of the dataset in terms of frequency, while Switzerland would be the lowest. Among females, Cleveland has the highest counts, while VA Long Beach records the fewest. For males, Hungary has the largest, and Switzerland has the smallest [17].

3.4. Data Splitting

The dataset was divided into distinct subsets for training and testing to bring methodological rigor into the model evaluation. By applying a stratified approach 25% of the data set was made available for testing purposes while the remaning 75% formed the training set.

This segregation will allow a better assessment of how the model performs on unseen data—a very crucial step in the assessment of generalizability and robustness across different datasets.

4. Data preprocessing

Data preprocessing is a very crucial task in data analysis and modeling, which means cleansing, transforming, and encoding to make the data of higher quality before it goes into modeling.

Data cleaning covers errors, missing values, and outliers. Data transformation makes data consistent and scaled. Categorical variables are encoded so that machine learning algorithms can process them effectively.

Data preprocessing is an effort to improve data quality and reliability, thereby providing better accuracy and performance in the analytical model built using the processed data.

4.1. Dealing with Missing Values

Missing values must be handled in data analysis and modelling to make sure that data integrity is preserved and results do not get biased. Not considering the missing values will further compromise the overall integrity of the dataset, which may result in false analyses and wrong conclusions. Developing a function that imputes the missing values using advanced techniques, such as Random Forest Classifier, Random Forest Regressor, and Iterative Imputer, came with this realization.

The Random Forest Classifier is used for the imputation of missing categorical data, while the Random Forest Regressor is used for missing continuous data. Furthermore, the Iterative Imputer has been utilized to estimate iteratively the missing values concerning the available features. Then, the imputed values were integrated into the dataset so that the model could be trained on a complete and representative dataset.

4.2. Outliers

Outliers, or those statistical anomalies lying outside the norm, form an especially important part of data analysis in terms of giving critical insights into and the detection of anomalies. Though some outliers can be considered errors and removed, they do show hidden patterns and refine predictive models. Careful examination for outliers within the dataset showed only one outlier, which was removed to preserve the meaningful insight contained within the rest of the data.

4.3. Scaling

The Min-Max scaling has been done, which will keep the consistency of data and allow comparability across features. It includes numeric column identification for scaling: 'old peak', 'thalach', 'Chol', 'trestbps', and 'age'. Min-Max Scaler has been used to transform each column's data into a specified range. Scaled data was fitted back to columns. Additional steps include storing scaler objects in a dictionary for each column to maintain scalability and reproducibility. This systematic approach should avoid the problem of disproportionately larger magnitude features from dominating model training.

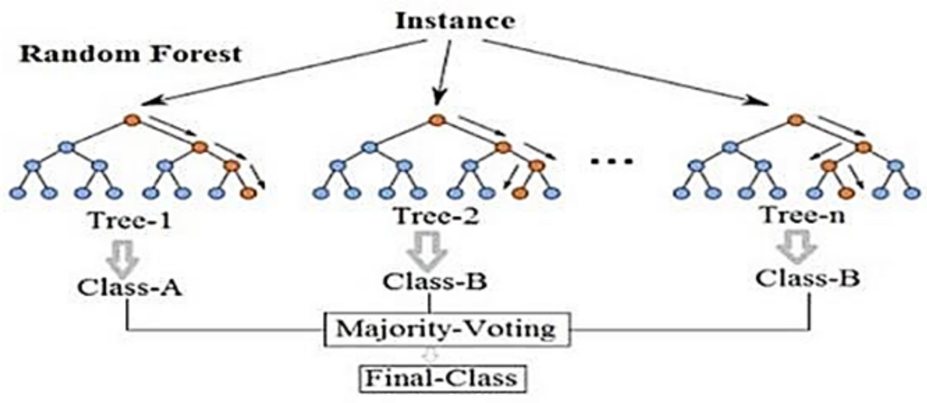

Figure 14. illustrates the Random Forest machine learning algorithm.

4.4. Standardization

For this, two techniques were used: Power Transformation and Quantile Transformation. Standardizing the numeric columns was done using both Box-Cox and Yeo-Johnson methods with the Power Transformer from sci-kit-learn. A Quantile Transformation has been performed thereafter with an output distribution of ’normal’. These techniques ensure standardized distribution of data, improving model robustness and interpretability.

4.5. Label Encoding

Label encoding has been used to convert categorical variables into numeric values. Specifically, the columns that include ’thal’, ’ca’, ’dataset’, ’slope’, ’exang’, ’restecg’, ’fbs’, ’cp’, ’sex’, and ’num’ will be targeted. For each column, a different Label Encoder object was instantiated; this permitted the transformation of categorical data into numeric format. The Label Encoder object fitted and transformed data from each column for compatibility with machine learning algorithms. The Label Encoder objects had been stored in a dictionary to keep consistency and to save all encoding information for the columns afterward.

5. Methodology

The applied machine learning techniques have several outcomes due to the various algorithms from each other.

5.1. Random Forest

Random Forest is a powerful ensemble learning method that is widely used for classification and regression problems [18]. It works effectively because at the time of training, a large number of decision trees are constructed, and the combined prediction tends to raise the level of accuracy while reducing overfitting.

The algorithm here demonstrates the application of Random Forest for classification. Figure 15 describes the workflow for a random forest algorithm, which is ensemble learning applied for classification and regression. It consists of several decision trees, trained on different subsets of data. For any new instance presented, each tree classifies it independently; for instance, Class-A or Class-B. It then employs majority voting to produce the final class prediction. This, in an ensemble way, enhances the prediction precision and consequently avoids overfitting issues that may arise when working with individual decision trees. Algorithm 1 Provides a robust and well-tuned Random Forest Classifier for various classification tasks, especially when dealing with complex datasets and the need for accurate predictions.

The mathematical equation of the Random Forest algorithm can be simplified as follows:

$$\hat{y} = \frac{1}{N} \sum_{i=1}^{N} y_i \tag{1}$$

Here, haty represents the final predicted output, N is the total number of trees in the forest, and yi denotes the prediction made by each tree Ti.

Algorithm 1 Hyperparameter-Tuned Random Forest Classifier Algorithm

```
1: Preprocessing:
2:     Create a dictionary for label encoders.
3:     Split the dataset into features (X) and target (y).
4: for each categorical column in X do
5:         Initialize a new LabelEncoder.
6:         Encode the column using the LabelEncoder.
7:         Store the LabelEncoder in the dictionary.
8: end for
9: Data Splitting and Scaling:
10:    Split the dataset into training and testing sets (test size = 0.3, random state = 0).
11:    Scale the features using MinMaxScaler.
12: Model Initialization:
13:    Initialize a RandomForestClassifier with balanced class weights and random state = 0.
14: Hyperparameter Tuning:
15:    Define a grid of hyperparameters:
16:       'n_estimators': [50, 100, 150]
17:       'max_depth': [None, 10, 20]
18:       'min_samples_split': [2, 5, 10]
19:       'min_samples_leaf': [1, 2, 4]
20:    Perform hyperparameter tuning using GridSearchCV with 5-fold cross-validation.
21:    Retrieve the best model and hyperparameters.
22: Model Training and Evaluation:
23:    Train the best model on the full training dataset.
24:    Evaluate the model on the test dataset.
25:    Compute and print the accuracy score.
26: Post-processing:
27:    Decode categorical columns in X using the stored label encoders.
28: Output:
29:    Return the trained model, best hyperparameters, and accuracy score.
```

5.2. *XGBoost Algorithm*

XGBoost is a powerful ensemble learning algorithm known for its outstanding performance and efficiency in predictive modeling. It belongs to the gradient-boosting family and often relies on decision trees as base learners [19]. For Gradient Boost, XGBoost builds an ensemble of weak learners, typically decision trees, by iteratively minimizing prediction errors. Additionally, it may apply regularization methods to avoid overfitting and ensure better generalization to unseen data. The algorithm uses parallelization to construct trees effectively, resulting in effective and efficient computation with high scalability. XGBoost is designed to manage missing values in the data, reducing the need to preprocess excessively. The XGBoost algorithm minimizes the objective function that consists of two key components: the loss function and the regularization term. Algorithm 2 provides a full and well-tuned XGBoost Classifier to conduct a wide range of classification tasks; especially, when the datasets contain both numerical and categorical features. Here is the simplified mathematical representation in equation 2.

$$\text{Obj}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{2}$$

- $Obj(\theta)$ is the objective function.θ represents the model parameters.
- $l(y_i, \hat{y}_i)$represents the loss function, measuring prediction errors.
- $\Omega(f_k)$is the regularization term, penalizing complex models to prevent overfitting.
- K is the number of trees in the ensemble.

5.3. *Logistic regression*

Heart disease occurrence will be predicted using Logistic Regression, which is an important task in medical research [20]. Preprocessing includes encoding categorical features and splitting the data into training and testing sets. In this case, the model is encapsulated into a more structured pipeline to ease the integration. Equation 3 shows the logistic regression equation:

$$P(Y = 1|X) = \frac{1}{1 + e^{(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}} \tag{3}$$

Where:

- $P(Y = 1|X)$ is the probability of the dependent variable being 1 given X
- $\beta_0, \beta_1, \ldots, \beta_n$ are coefficients
- $X_1, X_2, \ldots, X_n$ are independent variables

5.4. *Gradient Boosting*

Employing Gradient Boosting, an ensemble learning method, heart disease prediction accuracy will be enhanced. The approach iteratively refines predictions by sequentially minimizing errors. The mathematical equation for Gradient Boosting can be succinctly expressed as equation 4:

$$F(x) = F_{i-1}(x) + \eta \cdot h_i(x) \tag{4}$$

Where:

- $F(x)$ denotes the ensemble model's prediction
- $F_{i-1}(x)$ represents the previous ensemble model's prediction
- η is the learning rate, controlling the impact of each weak learner
- $h_i(x)$ signifies the prediction of the i-th weak learner

5.5. *Support Vector Machine (SVM)*

The occurrence of heart disease will be predicted using an SVM. It constructs the best hyperplane to classify data points in a way that maximizes the margin between classes. The mathematical formulation of the SVM in Equation 5 involves the optimization of the hyperplane parameters, represented by the weight vector w and the bias term b, in a way that the margin between classes is maximized while correctly classifying the data points. This is achieved by solving the optimization problem:

$$\min_{w,b} \frac{1}{2} \|w\|^2 \tag{5}$$

Subject to the constraints:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i \tag{6}$$

where x_i, y_i represents the data points and labels, and w^2denotes the squared norm of the weight vector. This optimization problem is typically solved using quadratic programming techniques to find the optimal hyperplane parameters that maximize the margin while ensuring the correct classification of data points.

Algorithm 2 Hyperparameter-Tuned XGBoost Classifier with Categorical Feature Encoding

```
1. Initialize Variables:
  - Create empty dictionary label_encoders to store LabelEncoders
2. Prepare Data:
  - Split dataset into features (X) and target (y)
3. Encode Categorical Variables:
  - For each column in X:
    If column dtype is 'object' or 'category':
      - Create new LabelEncoder
      - Fit encoder to column
      - Encode column
      - Store encoder in label_encoders
4. Split Data:
  - Split into train/test sets:
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=0)
5. Scale Features:
  - Initialize MinMaxScaler
  - Fit/transform data:
    X_train_scaled=scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
6. Define XGBoost Classifier:
  - Initialize model:
    xgb_model = XGBClassifier(random_state=0)
7. Set Hyperparameters:
  - Define param_grid:
    n_estimators: [50, 100, 150]
    max_depth: [3, 5, 7]
    learning_rate: [0.01, 0.1, 0.2]
    subsample: [0.8, 1.0]
    colsample_bytree: [0.8, 1.0]
    gamma: [0, 1, 2]
8. Grid Search Tuning:
  - Initialize GridSearchCV:
    GridSearchCV(xgb_model, param_grid, cv=5, scoring='accuracy')
    grid_search.fit(X_train_scaled, y_train)
9. Retrieve Best Model:
  - Get optimal parameters:
    best_params = grid_search.best_params_
10. Train Best Model:
  - Fit final model:
    best_xgb_model.fit(X_train_scaled, y_train)
11. Evaluate Model:
  - Make predictions and Calculate metrics:
    y_pred = best_xgb_model.predict(X_test_scaled)
    accuracy = accuracy_score(y_test, y_pred)
12. Return Results:
  Return best_xgb_model, best_params, accuracy
```

5.6. AdaBoost

This technique constructs a series of weak learners, iteratively adjusting weights to focus on misclassified data points.

Algorithm 3 AdaBoost Algorithm

```
1:  **1. Initialize weights:**
2:      For dataset with $N$ samples, set initial weights:
            $$w_i^{(1)} = \frac{1}{N} \quad \forall i \in \{1, \dots, N\}$$
3:  **2. For $m = 1$ to $M$:**
4:      (a) Sample dataset using weights $w_i^{(m)}$ to get training samples $x_i$
5:      (b) Fit classifier $K_m$ using all $x_i$
6:      (c) Compute error rate:
            $$\epsilon_m = \frac{\sum_{y_i \neq K_m(x_i)} w_i^{(m)}}{\sum_{i=1}^{N} w_i^{(m)}}$$
7:      (d) Compute classifier weight:
            $$\alpha_m = \frac{1}{2} \ln \left( \frac{1 - \epsilon_m}{\epsilon_m} \right)$$
8:      (e) Update weights:
            $$w_i^{(m+1)} = w_i^{(m)} e^{-\alpha_m y_i K_m(x_i)}$$
9:      (f) Normalize weights: $w_i^{(m+1)} \leftarrow \frac{w_i^{(m+1)}}{\sum_j w_j^{(m+1)}}$
10: **3. Final prediction:**
            $$K(x) = \text{sign} \left( \sum_{m=1}^{M} \alpha_m K_m(x) \right)$$
```

5.7. Convolutional Neural Networks (CNNs) Algorithm

Convolutional Neural Networks (CNNs) are a class of deep learning models primarily used for processing grid-like data such as images. They are designed to automatically and adaptively learn spatial hierarchies of features from input images, making them particularly effective for image recognition and classification tasks [21].

CNNs consist of various layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers apply filters (kernels) that convolve across the input image, capturing local patterns and features, while pooling layers downsample the feature maps, reducing dimensionality and computation while retaining essential information. This architecture allows CNNs to be translation invariant and highly efficient in recognizing patterns, contributing to their widespread use in computer vision applications such as object detection, facial recognition, and image segmentation.

5.8. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a specialized type of recurrent neural network (RNN) designed to address the limitations of traditional RNNs in learning long-term dependencies in sequential data [22]. LSTMs are particularly effective in tasks involving time series data, natural language processing, and other applications where context from previous inputs is crucial for predictions. The architecture of LSTMs includes memory cells, input gates, forget gates, and output gates, which work together to control the flow of information. This enables LSTMs to selectively remember or forget information over extended sequences, mitigating issues such as vanishing gradients that can hinder learning in traditional RNNs. As a result, LSTMs have become a popular choice for applications such as speech recognition, text generation, and any task that requires understanding contextual relationships in sequences.

6. Model Selection and Hyperparameter Tuning

Hyperparameter optimization, employing both grid and random search with 5-fold cross-validation, was performed for each model. Table 2 details the search space for key hyperparameters of Random Forest, XGBoost, and other models.

For Random Forest, we tuned the number of trees, the maximum depth of each tree, and the minimum number of samples required to split a node. For XGBoost, we adjusted the learning rate, maximum depth, gamma, and subsample ratio. We chose the final hyperparameters based on the highest validation accuracy and a good balance between precision and recall. We also tuned the hyperparameters separately on the training and validation sets to reduce overfitting.

To assess the effectiveness of deep learning models for heart disease prediction, we trained CNN and LSTM network models on the same dataset as the traditional ML models. CNNs are particularly effective for image-based diagnosis, while LSTMs are well-suited for sequential data. However, their applicability to tabular datasets, such as the UCI heart disease dataset, remains a challenge.

Algorithm 4 provides a robust approach to model selection and evaluation, considering multiple algorithms and hyperparameters to achieve the best possible performance.

Table 2. Search Space of Models' Hyperparameters

Model	Hyperparameter	Values
Random Forest	n_estimators	[50, 100, 200]
	max_depth	[5, 10, 20]
	min_samples_split	[2, 5, 10]
XGBoost	learning_rate	[0.01, 0.1, 0.2]
	max_depth	[3, 6, 9]
	gamma	[0, 1, 2]
	subsample	[0.8, 1.0]
Gradient Boosting	n_estimators	[50, 100, 150]
	learning_rate	[0.01, 0.1, 0.2]
	max_depth	[3, 5, 7]
Logistic Regression	C	[0.1, 1, 10]
	penalty	['l1','l2']
	n_estimators	[50, 100, 150]
	learning_rate	[0.01, 0.1, 0.2]
SVM	C	[0.1, 1, 10]
	kernel	['linear','rbf','poly']

Algorithm 4 Ensemble Model with Hyperparameter Tuning and Categorical Feature Encoding

```
1. Data Preparation:
  - Split dataset into features (X) and target (y):
    ▷ X = data.drop('target', axis=1)
    ▷ y = data['target']
2. Initialize Label Encoder:
  - Create encoder object:
    ▷ label_encoder = LabelEncoder()
3. Encode Categorical Columns:
  - For each column in X:
    ▷ If column is categorical:
      - Encode column:
        X[column] = label_encoder.fit_transform(X[column])
4. Split Data:
  - Divide into train/test sets:
    ▷ X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
5. Define Models:
  - Create model list:
    ▷ models = [('LogReg', LogisticRegression()),
      ('GBoost', GradientBoostingClassifier()),
      ('SVM', SVC())]
6. Initialize Tracking:
  - best_model = None
  - best_accuracy = 0.0
7. Evaluate Models:
for model_name, model in models do
      - Create pipeline:
        ▷ pipeline = Pipeline([('model', model)])
      - Cross-validation:
        ▷ mean_acc = cross_val_score(pipeline, X_train, y_train, cv=5).mean()
      - Fit pipeline:
        ▷ pipeline.fit(X_train, y_train)
      - Make predictions:
        ▷ y_pred = pipeline.predict(X_test)
      - Calculate accuracy:
        ▷ accuracy = accuracy_score(y_test, y_pred)
      - Print metrics:
        ▷ print(f"model_name: CV=mean_acc:.2f, Test=accuracy:.2f")
end for
8. Update Best Model:
if accuracy > best_accuracy then
      - best_accuracy = accuracy
      - best_model = model_name
end if
9. Print Results:
  - print(f"Best Model: best_model (Accuracy: best_accuracy:.2f)")
```

7. Experimental results

7.1. Model Evaluation

The training of Random Forest model is done through hyperparameter tuning. Random forest experiment results 87% for test accuracy, which justifies the fact that this is practical, too, in predictive modelling. XGBoost algorithm introduces optimal hyperparameters using hyper-parameter tuning, such as $'colsample_bytree' : 1,' gamma' : 2,' learning_rate' : 0.1,' max_depth' : 3,' n_estimators' : 100,' subsample' : 1$. It achieved test set accuracy of 88%. Cross-validation generalization performance of Logistic regression algorithm yields an average accuracy of 0.622. The model trains on the training set and achieves a test accuracy 0.594. However, Cross-validation assesses Gradient Boost model performance across multiple training data folds, yielding a mean accuracy of 0.648. Subsequent testing results in a noteworthy accuracy of 0.696.

In case of SVM, cross-validation estimated the performance of the SVM model on folds of the training data, with a mean accuracy of 0.591. The model is evaluated, resulting in a test accuracy of 0.594. In addition, cross-validation evaluates the AdaBoost model across multiple folds of training data, yielding a mean accuracy of 0.610. Subsequently, the model is trained and evaluated, achieving a test accuracy of 0.612.

Model evaluation extends beyond accuracy to encompass computational efficiency, deployment ease, and interpretability. While XGBoost excels in prediction, logistic regression's superior interpretability makes it preferable when explainability is paramount. Deep learning models like CNNs and LSTMs, though powerful, can be computationally prohibitive for real-time use. Table 3 presents a comparative analysis of algorithm accuracy across multiple research articles, highlighting variations in performance. Each study employs distinct methodologies, demonstrating unique levels of accuracy based on the approaches to solving specific problems. The comparison underscores the strengths and limitations of different algorithms. This comparative review not only provides visions into the efficacy of individual procedures but also highlights advancements in the field, emphasizing the position of picking the correct algorithm for specific applications. Overall, the observed accuracies vary widely across different algorithms and papers. Table 3 provides a comparison of the accuracy of various models used in this study against results from prior research. To ensure a fair comparison, we acknowledge that differences in dataset size, feature selection, preprocessing techniques, and hyperparameter tuning can impact model performance.

Table 4 presents a performance comparison between traditional ML models and DL approaches. While XGBoost achieved the highest accuracy among traditional models (88%), CNN and LSTM models yielded lower performance due to the dataset's small size. This suggests that deep learning techniques may require larger, more diverse datasets to achieve superior results.

Table 3. Comparison Between Algorithm's Accuracy

Algorithm/Reference	**Proposed**	**[6]**	**[9]**	**[10]**	**[12]**	**Notes on Differences**
SVM	59%	–	81.48%	75%	85.7%	Dataset/preprocessing variations
Ada Boost	60%	–	–	82%	–	Feature selection impact
Random Forest	87%	82%	81.48%	1%	–	Preprocessing methods differ
XGBoost	88%	–	–	–	–	Hyperparameter tuning variations
Gradient Boosting	70%	–	–	80.24%	–	Feature engineering differences
Logistic Regression	59%	–	82.71%	78%	–	Data transformation methods
KNN	87%	87%	85.18%	–	–	Cross-validation differences
ANN	87%	–	–	–	–	Dataset selection variance
Naïve Bayes	88%	81.48%	76%	–	–	Assumptions on data distribution
Decision Tree	–	78%	70.37%	–	–	Complexity of decision rules

Table 4. Performance Comparison of Traditional ML and Deep Learning Models

Model	Accuracy	Precision	Recall	F1-score	ROC-AUC
Random Forest	0.87	0.85	0.86	0.85	0.90
XGBoost	0.88	0.87	0.88	0.87	0.92
Gradient Boosting	0.70	0.68	0.69	0.68	0.75
Logistic Regression	0.59	0.57	0.58	0.57	0.65
AdaBoost	0.60	0.58	0.59	0.58	0.67
SVM	0.59	0.56	0.57	0.56	0.64
CNN	0.65	0.63	0.64	0.63	0.72
LSTM	0.62	0.60	0.61	0.60	0.70

8. Conclusion and Future Work

This research compared several machine learning algorithms (Gradient Boosting, Logistic Regression, AdaBoost, SVM, XGBoost, Random Forest, LSTM, and CNN) for heart disease prediction, revealing significant performance differences. To avoid overemphasizing accuracy, evaluation included precision, recall, F1-score, and ROC-AUC, ensuring a more balanced assessment. XGBoost achieved the highest accuracy (88%), while Random Forest improved from 82% to 85% after tuning, highlighting the importance of algorithm selection and optimization. Despite XGBoost's superior performance on this dataset, further improvements in accuracy and interpretability are needed. This study offers a valuable comparative analysis of machine learning techniques for heart disease prediction, providing insights for future healthcare data analytics research.

In future work, we will incorporate more diverse datasets, including electronic health records (EHRs) and data from wearable devices, to improve model generalizability and real-world applicability. Also, we will work on validating the models in real-world healthcare settings and develop frameworks aligned with medical guidelines. Additionally, we will utilize time-series datasets instead of static data and investigate models like RNNs to track disease progression for improved prediction and personalized treatment. Furthermore, we will prioritize hybrid approaches that balance accuracy with computational feasibility.

REFERENCES

1. Md. S. Hossain, Md. A. Talukder, and Md. Z. Mahmud, "Advancements in Cardiovascular Disease Detection: Leveraging Data Mining and Machine Learning," *bioRxiv*, 2024, doi: 10.1101/2024.03.09.584222.
2. H. Sadr, A. Salari, M. T. Ashoobi, and M. Nazari, "Cardiovascular disease diagnosis: a holistic approach using the integration of machine learning and deep learning models," *Eur J Med Res*, vol. 29, no. 1, p. 455, 2024, doi: 10.1186/s40001-024-02044-7.
3. M. Sunil Kumar, A. K. Sah, G. Ruthvik, M. S. Prabez, and R. Adhikari, "Advancements in Heart Disease Prediction: A Comprehensive Review of ML and DL Algorithms," in *Proc. Int. Conf. Technol. Adv. Comput. Sci. (ICTACS)*, 2023, doi: 10.1109/ICTACS59847.2023.10390155.
4. M. M. Ahsan and Z. Siddique, "Machine learning-based heart disease diagnosis: A systematic literature review," *Artif Intell Med*, vol. 128, p. 102289, 2022, doi: 10.1016/j.artmed.2022.102289.
5. F. Tasnim and S. U. Habiba, "A Comparative Study on Heart Disease Prediction Using Data Mining Techniques and Feature Selection," in *Proc. Int. Conf. Robot. Electr. Signal Process. Tech. (ICREST)*, 2021, doi: 10.1109/ICREST51555.2021.9331158.
6. J. P. Li, A. U. Haq, S. U. Din, J. Khan, A. Khan, and A. Saboor, "Heart Disease Identification Method Using Machine Learning Classification in E-Healthcare," *IEEE Access*, vol. 8, pp. 107562–107582, 2020, doi: 10.1109/ACCESS.2020.3001149.
7. A. Almulihi et al., "Ensemble Learning Based on Hybrid Deep Learning Model for Heart Disease Early Prediction," *Diagnostics*, vol. 12, no. 12, 2022, doi: 10.3390/diagnostics12123215.
8. M. Nasr, Md. M. Islam, S. Shehata, F. Karray, and Y. Quintana, "Smart Healthcare in the Age of AI: Recent Advances, Challenges, and Future Prospects," *IEEE Access*, vol. 9, pp. 145248–145270, 2021, doi: 10.1109/ACCESS.2021.3118960.
9. N. Akhtar, "Heart Disease Prediction," Mar. 2021.
10. A. Srivastava and A. Kumar Singh, "Heart Disease Prediction using Machine Learning," in *Proc. Int. Conf. Adv. Comput. Innov. Technol. Eng. (ICACITE)*, 2022, pp. 2633–2635, doi: 10.1109/ICACITE53722.2022.9823584.
11. M. Munsif, M. Rashid, and F. Jabeen, "An Efficient Hybrid Classification Model for Heart Disease Prediction," *Res Sq*, 2024, doi: 10.21203/rs.3.rs-3863899/v1.
12. J. Ivan et al., "Advancing Predicting Heart Disease: A Comparative Study of SVM, Naive Bayes, KNN, Linear Regression, and LSTM Models," in *Proc. IEEE Int. Conf. Comput. Eng. Design (ICCED)*, 2023, doi: 10.1109/ICCED60214.2023.10425268.

13. B. Duraisamy et al., “Heart disease prediction using support vector machine,” *Multidisciplinary Science Journal*, 2024, doi: 10.31893/multiscience.2024ss0104.
14. S. Mahmoudinejad and N. Safdarian, “Evaluating morphological features of electrocardiogram signals for diagnosing of myocardial infarction using classification-based feature selection,” *J Med Signals Sens*, vol. 11, no. 2, 2021, doi: 10.4103/jmss.JMSS_12_20.
15. Y. K. Singh, N. Sinha, and S. K. Singh, “Heart disease prediction system using random forest,” in *Commun. Comput. Inf. Sci.*, 2017, doi: 10.1007/978-981-10-5427-3_63.
16. L. Liu et al., “Geographic Variation in Heart Failure Mortality and Its Association With Hypertension, Diabetes, and Behavioral-Related Risk Factors in 1,723 Counties of the United States,” *Front Public Health*, vol. 6, 2018, doi: 10.3389/fpubh.2018.00132.
17. V. Parcha et al., “Geographic Variation in Cardiovascular Health Among American Adults,” *Mayo Clin Proc*, vol. 96, no. 7, 2021, doi: 10.1016/j.mayocp.2020.12.034.
18. M. Schonlau and R. Y. Zou, “The random forest algorithm for statistical learning,” *Stata Journal*, vol. 20, no. 1, 2020, doi: 10.1177/1536867X20909688.
19. S. Sankar, A. Potti, G. Naga Chandrika, and S. Ramasubbareddy, “Thyroid Disease Prediction Using XGBoost Algorithms,” *J. Mobile Multimedia*, vol. 18, no. 3, 2022, doi: 10.13052/jmm1550-4646.18322.
20. R. Prasad, P. Anjali, S. Adil, and N. Deepa, “Heart disease prediction using logistic regression algorithm using machine learning,” *Int. J. Eng. Adv. Technol.*, vol. 8, no. 3, 2019.
21. O. Bhaskaru and M. Sreedevi, “Heart disease prediction using neuro-genetic algorithm and CNN-MDRP classifier,” *Indian J. Comput. Sci. Eng.*, vol. 12, no. 4, 2021, doi: 10.21817/indjcse/2021/v12i4/211204145.
22. J. Xia et al., “A Long Short-Term Memory Ensemble Approach for Improving the Outcome Prediction in Intensive Care Unit,” *Comput Math Methods Med*, vol. 2019, 2019, doi: 10.1155/2019/8152713.