**An Algorithm for Detecting Heart Disease**

**Using Machine Learning**

Ankur Majumdar, Gangireddy Narendra Kumar Reddy

Department of Computer Science and Engineering,

Indian Institute of Information Technology Kottayam, Valavoor P.O., Pala,

Kottayam - 686635, Kerala, India

{ankur22bcd46, gnkreddy}@iiitkottayam.ac.in

**Abstract.** Objective: In this paper, we present a method to detect heart disease by using machine learning (ML) algorithms which use various features of heart disease dataset for classification. This research paper aims to review the current state-of-the-art in heart disease detection using machine learning (ML), including the types of data used, the algorithms employed, and the performance metrics used to evaluate the effectiveness of these approaches. The paper also highlights the challenges and limitations of current techniques, such as the need for large amounts of high-quality data and the potential for bias in algorithm development. Methods: To conduct this research, a review of existing literature on heart disease detection using machine learning (ML) was done. The implementation is based on some preprocessing of data, usage of various ML classifiers with different types/parameters, and a detailed overall comparison of them. Validation Dataset: The proposed heart disease classification method is evaluated using the largest heart disease database available so far from IEEE Xplore DataPort. Results: The proposed method had highest average classification accuracy of 94.20% with the K-Fold Cross Validation technique (n = 10). Results showed that the proposed method can outperform the existing methods. Conclusion: Evaluation results demonstrate that the proposed method is more accurate in classifying heart disease with high accuracy and less processing time and storage space.

**Keywords:** Heart Disease, Machine Learning (ML), IEEE Xplore DataPort, and K-Fold Cross Validation Technique

1. Introduction

Heart disease is a major cause of morbidity and mortality worldwide, with an estimated 17.9 million deaths in 2019 alone. Early detection of heart disease will result in effective treatment and cure. In recent years, machine learning techniques have been increasingly used for the automated detection of heart disease using medical imaging and other diagnostic tests. Machine learning techniques enable the development of algorithms that can automatically learn patterns and relationships from data, allowing for accurate and efficient analysis of complex datasets. The use of machine learning techniques in heart disease detection has the potential to improve the accuracy and efficiency of diagnosis, thereby facilitating earlier intervention and improving patient outcomes.

However, the application of machine learning techniques in heart disease detection is not without challenges. These include the need for large amounts of high-quality data, the potential for bias in algorithm development, and the need for rigorous evaluation of algorithm performance. This research paper aims to review the current state-of-the-art in heart disease detection using machine learning techniques, including the types of data used, the algorithms employed, and the performance metrics used to evaluate the effectiveness of these approaches. The paper will also highlight the challenges and limitations of current techniques and discuss future research directions and potential solutions to these challenges. By synthesizing the existing literature on heart disease detection using machine learning techniques, this research paper aims to provide insights into this field and identify areas for future research to improve the accuracy and efficiency of heart disease detection using machine learning techniques.

* 1. Existing Heart Disease Detection Methods and Overview

### To provide an overview of existing methods for heart disease detection using machine learning in industry and hospitals, we conducted a systematic review of the literature. We searched several databases, including the largest Heart Disease Database of IEEE Xplore DataPort. We included studies that focused on heart disease detection using machine learning techniques, used medical imaging or diagnostic tests as data sources, and reported performance metrics such as sensitivity, specificity, accuracy, and area under the receiver operating characteristic curve (AUC-ROC). We excluded studies that focused on other cardiovascular diseases, used non-machine learning approaches, and used non-medical data sources. Two researchers independently screened the titles and abstracts of identified studies to determine their relevance. Full-text articles of potentially relevant studies were then assessed for eligibility based on inclusion and exclusion criteria. Data extraction was performed on the included studies, including information on the type of data used, the machine learning algorithms employed, the performance metrics reported, and any limitations or challenges identified by the authors.

In [1], M. C. H. Yap et al. (2021) provide an overview of recent advances in deep learning-based automatic detection and diagnosis of cardiovascular diseases (CVDs). They first introduce the basic anatomy and physiology of the heart and describes various CVDs and their associated risk factors. Then, they review different modalities used for CVD diagnosis, including electrocardiogram (ECG), echocardiography, and cardiac magnetic resonance imaging (MRI). The paper goes on to discuss different deep learning-based approaches that have been used for CVD detection and diagnosis, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and autoencoders. It also highlights some of the challenges associated with using deep learning for CVD diagnosis, such as data imbalance and interpretability issues.

In [2], L. Wang et al. (2020) analyzed 56 studies and found that deep learning models can achieve high accuracy in detecting cardiovascular diseases, with the convolutional neural network being the most used algorithm.

In [3], H. Zafar et al. propose a machine learning framework for predicting cardiovascular diseases. They used a dataset that includes clinical and demographic information of patients to train their machine learning models. They experimented with three different machine learning algorithms, namely Random Forest, Decision Tree, and Gradient Boosting, and evaluated their performance based on several metrics such as accuracy, sensitivity, specificity, and F1-score. The results showed that the Random Forest algorithm outperformed the other two algorithms with an accuracy of 83.5% and an F1-score of 0.82. The authors also conducted feature importance analysis and identified age, total cholesterol, and systolic blood pressure as the top three most important features for predicting cardiovascular diseases. The proposed machine learning framework can potentially be used by healthcare professionals to identify high-risk patients and provide preventive measures to reduce the incidence of cardiovascular diseases.

In [4], T. Kim et al. used a convolutional neural network (CNN) to extract relevant features from electrocardiogram (ECG) signals and a long short-term memory (LSTM) network to model the temporal dynamics of the ECG signals. They evaluated the proposed approach on a dataset consisting of ECG recordings from 5,000 patients, where the recordings were labeled as either healthy or diseased. The experimental results show that the proposed approach achieves a high accuracy of 96.3% in classifying the ECG recordings as healthy or diseased.

### Similarly, several machine learning (ML) techniques have been used for heart disease detection in industry and hospitals, including support vector machines, artificial neural networks, random forests, convolutional neural networks, and deep belief networks. These techniques have been used with various types of data, including electrocardiogram signals and medical images such as echocardiograms and computed tomography (CT) scans. The use of machine learning techniques in heart disease detection in medical industry and hospitals has the potential to improve the accuracy and efficiency of diagnosis, facilitating earlier intervention and improving patient outcomes. However, the use of these techniques in clinical practice requires rigorous evaluation and validation.

* 1. Motivation and Key Contributions

Since, heart disease is a leading cause of death worldwide. Early detection and diagnosis of heart disease are crucial in preventing or managing the disease effectively. However, traditional diagnostic methods such as electrocardiogram (ECG) and blood tests have limitations in accuracy and efficiency. Therefore, there is a need for more efficient and accurate methods for heart disease detection. Machine learning (ML) algorithms have shown great promise in this regard, as they can analyze large amounts of data and provide accurate predictions.

Firstly, ML algorithms can analyze large amounts of data and identify patterns that may not be discernible to humans. This allows for more accurate detection and diagnosis of heart disease. ML algorithms can also detect subtle changes in physiological signals that may indicate the early stages of heart disease, enabling early intervention and treatment. ML algorithms can efficiently analyze individual patient data to develop personalized treatment plans based on their unique characteristics and medical history.

ML algorithms can easily process large amounts of data quickly, reducing the time and cost required for heart disease diagnosis and treatment. This improves the efficiency of healthcare delivery, allowing more patients to receive timely and accurate diagnosis and treatment for heart disease. Lastly, ML algorithms can be used in remote or underserved areas, enabling more people to access healthcare services for heart disease diagnosis and treatment.

Thus, the key contributions of heart disease detection using ML include improved accuracy, early detection, personalized medicine, improved efficiency, and accessible healthcare. By leveraging the capabilities of ML algorithms, we can improve the accuracy, efficiency, and accessibility of heart disease diagnosis and treatment, ultimately improving patient outcomes and reducing the burden of heart disease on society.

1. Materials and Methods

In order to evaluate the performance of various models with different ML algorithms, we used the largest heart disease publicly-available dataset such as IEEE DataPort’s Heart Disease Dataset (HDD – Comprehensive) [5], as described below:

### IEEE DataPort’s HDD – Comprehensive: The IEEE DataPort’s HDD is curated by combining 5 popular heart disease datasets already available independently but not combined before. In this dataset, 5 heart datasets are combined over 11 common features which makes it the largest heart disease dataset available so far for research purposes. The five datasets used for its curation are: Cleveland, Hungarian, Switzerland, Long Beach VA, and Statlog (Heart) Data Set. This dataset consists of 1190 instances with 11 features. These datasets were collected and combined at one place to help advance research on CAD-related machine learning and data mining algorithms, and hopefully to ultimately advance clinical diagnosis and early treatment. This dataset is widely used for building and testing a predictive machine learning model for early-stage heart disease detection.

* 1. Proposed Heart Disease Detection Method

The proposed method consists of the following steps:

(1) Data Preprocessing

(2) Model Selection

(3) Model Training

(4) Model Evaluation

Each of the steps of the proposed method is described in the next subsections.

Data Preprocessing: Preprocessing is a critical step in heart disease detection using machine learning algorithms, as it helps to ensure that the data used for training and testing the ML models is clean, relevant, and accurate. Data preprocessing involves a series of operations such as cleaning, normalization, feature extraction, and feature selection.

The first step in data preprocessing is data cleaning, which involves removing any noise, missing values, or irrelevant information from the dataset. This is important as noisy or irrelevant data can adversely affect the performance of the ML model. Data cleaning can be done using various techniques such as imputation, deletion, or interpolation. In this method, we have used imputation in which mean (or any other central tendency value such as median or mode) is assigned to null values in the dataset. It is a common technique used in data preprocessing to handle missing values in a dataset. Imputation can help to ensure that the resulting dataset is complete and suitable for use in machine learning algorithms.

(Sir please add the Mean formula and related description)

The next step is normalization, or standardization, which involves scaling the data to a common range. This is important as it helps to prevent bias towards features with larger values. Standardization can be done using techniques such as min-max scaling, z-score scaling, or feature scaling.

(Sir please add the Feature Standardization/Scaling formulae and related description)

Model Selection: Model selection involves choosing the best algorithm and hyperparameters for a given dataset and problem. The goal of model selection is to find the model that performs best on the test set while avoiding overfitting to the training set. There are several approaches to model selection, including cross-validation, grid search, and Bayesian optimization. The cross-validation method involves splitting the dataset into training, validation, and test sets. The model is trained on the training set and evaluated on the validation set. This process is repeated multiple times, with different subsets of the data used for training and validation.

In this method, we have used the K-fold cross-validation technique (K = 10 for the HDD) to perform Model Selection as well as Model Evaluation processes for various machine learning algorithms. It involves splitting the dataset into K equal-sized partitions or folds. The algorithm is then trained K times, with each partition used once as the validation set and the remaining partitions used as the training set. During each iteration, the model is trained on K-1 folds and evaluated on the remaining fold. The performance of the model is then recorded for each fold. This process is repeated K times, with each fold used once as the validation set.

#### Model Training: Model training refers to the process of training a machine learning model on a dataset of preprocessed data to learn patterns and relationships between the input features and the output labels (i.e., the presence or absence of heart disease). During training, the machine learning (ML) algorithm adjusts the parameters of the model to minimize the error between the predicted output and the true output labels. This process is typically done using an optimization algorithm such as stochastic gradient descent (SGD) or Adam.

#### The choice of machine learning algorithm used for model training depends on the specific task and the nature of the data. For example, logistic regression and SVMs are commonly used for classification tasks, while regression models such as linear regression and decision trees are used for regression tasks. In recent years, deep learning algorithms such as CNNs and RNNs have shown promising results in heart disease detection. But in this research, we are going to get similar results without any usage of neural network algorithms. Hence, the classification algorithms we have used are as follows:

#### **A. Support Vector Machine (SVM):** SVM is popular supervised machine learning algorithm used for classification and regression tasks. SVMs are particularly useful when the data is non-linearly separable, meaning that there is no straight line or hyperplane that can perfectly separate the data into different classes. In such cases, SVMs use a kernel trick to transform the original feature space into a higher-dimensional space where the data can be linearly separated.

#### SVMs work by finding a hyperplane in the transformed feature space that separates the data into different classes while maximizing the margin, which is the distance between the hyperplane and the closest data points from each class. The hyperplane that maximizes the margin is called the maximum margin hyperplane. The SVM in Python’s scikit-learn allows for the use of different types of kernels to transform the feature space. Some commonly used kernels in SVMs include:

#### Linear kernel: This is the simplest kernel and is used when the data is linearly separable. It is defined as the dot product of the feature vectors in the original feature space.

#### Polynomial kernel: This kernel is used when the data has a polynomial structure. It maps the data into higher-dimensional space using a polynomial function of degree 'd'.

#### Radial basis function (RBF) kernel: This kernel is the most commonly used kernel in SVMs. It maps the data into an infinite-dimensional space using a Gaussian function with a bandwidth parameter γ.

#### The hyperparameters of the SVM, such as the type of kernel and the regularization parameters 'C', 'gamma', can also be set as per model requirements. In this, we used ‘gamma’ = 100, ‘C’ = 5.

#### **B. K-Nearest Neighbors (KNN):** KNN is also one of the machine learning algorithms used for both classification and regression problems. It belongs to the supervised learning category of algorithms where the training data consists of labeled examples. The KNN algorithm classifies new data points based on the similarity to previously labeled data points. Specifically, given a new data point, the KNN algorithm finds the K closest labeled data points to the new point, and the majority class of these K data points is used to predict the class of the new data point.

#### The value of K (n\_neighbors) is a vital hyperparameter in the KNN algorithm. It determines the number of closest labeled data points used to predict the class or value of the new data point. A small value of K (e.g., K = 1) leads to overfitting, where the model is too complex and has low generalization power. On the other hand, a large value of K (e.g., K = N, where N is the number of labeled data points) leads to underfitting, where the model is too simple and does not capture the underlying patterns in the data. To determine the optimal value of K, the KNN algorithm uses cross-validation, where the dataset is split into a training set and a validation set. The KNN algorithm is trained on the training set using different values of K, and the performance of the model is evaluated on the validation set. The value of K that gives the best performance on the validation set is selected as the optimal value of K (the optimal value of K/n\_neighbors as per the HDD was found to be 19).

#### However, the main disadvantage of KNN is its computational complexity, which increases with the size of the dataset. In addition, KNN is sensitive to the choice of distance metric used to measure the similarity between data points, and may not perform well with high-dimensional data.

**C. Naïve Bayes:** Naïve Bayes is a machine learning algorithm that is commonly used for classification tasks, including heart disease detection. It is a probabilistic algorithm that makes use of Bayes' theorem, which states that the probability of a hypothesis (in this case, the presence or absence of heart disease) given the evidence (input features such as age, blood pressure, and cholesterol levels) is proportional to the probability of the evidence given the hypothesis, multiplied by the prior probability of the hypothesis.

(Sir please add the Bayes Theorem of Probability formula and related description)

In the context of heart disease detection, Naïve Bayes assumes that the input features are conditionally independent given the class label, meaning that the presence or absence of one feature does not affect the probability of another feature given the class label. This is a simplifying assumption that allows the algorithm to make predictions quickly and efficiently. There are several types of Naïve Bayes algorithms such as:

Gaussian Naïve Bayes: Gaussian Naïve Bayes is used when the input features are continuous variables that can be modeled using a Gaussian (Normal) distribution. In this case, the algorithm calculates the mean and standard deviation of each feature for each class label and uses these values to estimate the conditional probability of each feature given the class label. The algorithm then combines these probabilities using Bayes' theorem to calculate the posterior probability of each class label.

Bernoulli Naïve Bayes: Bernoulli Naïve Bayes is used when the input features are binary variables (i.e., they can take on only two values, such as 0 or 1). In this case, the algorithm assumes that each feature is conditionally independent given the class label and models the conditional probability of each feature given the class label using a Bernoulli distribution. The algorithm then combines these probabilities using Bayes' theorem to calculate the posterior probability of each class label.

In both Gaussian Naïve Bayes and Bernoulli Naïve Bayes, the prior probability of each class label is estimated from the training data. During training, the algorithm calculates the conditional probabilities of each feature given each class label and uses these probabilities to make predictions on new data. The algorithm selects the class label with the highest posterior probability as the predicted class label.

One advantage of Naïve Bayes is that it is relatively simple and computationally efficient compared to other machine learning algorithms. However, the assumption of conditional independence between features may not hold in all cases, and the performance of the algorithm may suffer as a result. In addition, Naïve Bayes tends to perform poorly when there are strong correlations between features or when there are interactions between features that affect the probability of the class label.

**D. Decision Tree:** Decision tree is a machine learning algorithm that is commonly used for classification tasks, including heart disease detection. It is a type of supervised learning algorithm that makes a series of binary decisions based on the input features to predict the class label of a new data point. The algorithm starts by selecting a feature that best splits the data into the two most distinct groups based on the class label. This feature is called the root node, and it is used to split the dataset into two subsets. The algorithm then repeats this process recursively for each subset until a stopping criterion is met, such as a maximum depth or a minimum number of data points per leaf node.

During the construction of the decision tree, the algorithm selects the features and the thresholds for splitting the data based on a criterion that maximizes the purity of each subset, such as the Gini impurity or the information gain. The Gini impurity measures the probability of misclassifying a randomly chosen data point in a subset, while the information gain measures the reduction in entropy (i.e., the amount of uncertainty) of the class labels after splitting the data. Once the decision tree is constructed, it can be used to make predictions on new data by traversing the tree from the root node to the leaf node that corresponds to the predicted class label. Each internal node represents a binary decision based on the value of a feature, while each leaf node represents a predicted class label.

One advantage of decision trees is that they are interpretable and can provide insights into the relationships between the input features and the class label. In addition, decision trees can handle both categorical and continuous variables, and they are robust to noisy data and outliers. However, decision trees can be sensitive to the structure of the data, and small changes in the data or the splitting criterion can lead to different trees and different predictions. Decision trees can also suffer from overfitting if they are too complex and capture the noise in the data instead of the underlying patterns.

To address these issues, several variations of decision trees have been proposed, such as random forests, gradient boosting, etc. These algorithms combine multiple decision trees to improve the accuracy and stability of the predictions while reducing the risk of overfitting.

**E. Random Forest:** Random Forest is a famous machine learning algorithm that is widely used for classification tasks. It is an ensemble learning method that combines multiple Decision Trees to create a more accurate and robust model. Each decision tree in the random forest is constructed independently from the others, and the final prediction is made by aggregating the predictions of all the individual trees. The basic idea behind Random Forest is to create a diverse set of decision trees by randomly selecting subsets of the training data and features at each node of the tree. This helps to reduce overfitting and increase the generalization performance of the model. The algorithm works by building multiple decision trees on different subsets of the data and features, and then combining the predictions of these trees to obtain a final prediction. The output of the algorithm is the class label that is predicted by the majority of the trees.

The key advantages of Random Forest like the Decision Tree algorithm are its ability to handle both categorical and continuous input features, its robustness to noisy and missing data, and its ability to capture complex nonlinear relationships between input features and output labels. Additionally, Random Forest is relatively easy to use and can be trained on large datasets without requiring extensive computational resources.

During training, the Random Forest algorithm first randomly selects a subset of the input features and then randomly samples a subset of the training data with replacement. It then constructs a decision tree on the selected data and features using a split criterion such as information gain or Gini impurity. This process is repeated for a fixed number of trees, and the predictions of all the individual trees are combined using a majority vote to obtain the final prediction.

One important hyperparameter of the Random Forest algorithm is the number of trees (n\_estimators), which controls the level of diversity in the ensemble. Increasing the number of trees generally improves the performance of the algorithm, but also increases the computational complexity and may lead to overfitting. The value of n\_estimators that gives the best result on the validation set is selected as the optimal value of n\_estimators (the optimal value of n\_estimators as per the HDD was found to be 75).

**F. Logistic Regression:** Logistic Regression is a machine learning algorithm which is used for classification tasks, despite having the term regression in it. It is a statistical method that uses a logistic function to model the relationship between the input features and the binary output label. Logistic Regression is a linear algorithm, meaning that it seeks to find the linear relationship between the input features and the output label. The algorithm models the probability of a binary outcome (the presence or absence of heart disease) given the input features by fitting a logistic function to the data.

The Logistic function is defined as follows:

(Sir please add the Logistic Function formula and related description)

During training, the algorithm uses a method called maximum likelihood estimation to estimate the weights and bias term that maximize the likelihood of the observed data given the model. This involves optimizing a cost function, such as the negative log-likelihood function, using an optimization algorithm such as gradient descent or Newton's method. The trained logistic regression model is then used to make predictions on new data. After being given a set of input features, the model calculates the probability of the binary output label using the logistic function. If the probability is greater than a threshold value (usually 0.5), the model predicts a positive label (the presence of heart disease); otherwise, it predicts a negative label (the absence of heart disease).

Logistic regression has several advantages as a machine learning algorithm for heart disease detection. It is computationally efficient and relatively simple to implement, making it a good choice for small to medium-sized datasets. It can handle both numerical and categorical input features, and it is interpretable, meaning that it is possible to understand how the model arrived at its predictions.

However, logistic regression assumes a linear relationship between the input features and the output label, which does not always hold in practice. It also suffers from overfitting if the number of input features is large relative to the number of training samples. Finally, logistic regression is only suitable for binary classification tasks, so it may not be appropriate for tasks where there are more than two possible output labels.

Overall, model training was a crucial step in heart disease detection using machine learning algorithms, and it involved selecting appropriate machine learning algorithms, preprocessing the data, and optimizing the model parameters to achieve good results.

Model Evaluation: As described in the Model Training subsection, the K-fold cross validation takes care of model evaluation as well. It involved dividing the dataset into K (K = 10 for the HDD) subsets, and then training and testing the model K times, each time using a different subset as the test set and the rest as the training set. By using the K-fold cross-validation, we have obtained more reliable estimate of the model performance, and also mitigated the problem of overfitting.

At the end of K iterations, the average performance of the model is calculated by averaging the performance metrics obtained during each iteration. The advantage of K-fold cross-validation is that it provides a more reliable estimate of the model's performance than a single train-test split. It also ensures that the model is evaluated on all data points in the dataset.

(Sir please add the K-Fold Cross Validation formula and related description)

* 1. Performance Metrics

In binary classification problems, ML models are trained to make predictions about the positive or negative class of particular samples. In order to get the classification report of a model, it is important to understand four important metrics: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

**True Positive (TP):** A TP occurs when the model correctly predicts a positive sample as positive. TP is an important metric because it represents the number of positive samples that the model correctly identifies.

**True Negative (TN):** A TN occurs when the model correctly predicts a negative sample as negative. TN is an important metric because it represents the number of negative samples that the model correctly identifies.

**False Positive (FP):** A FP occurs when the model incorrectly predicts a negative sample as positive.

**False Negative (FN):** A FN occurs when the model incorrectly predicts a positive sample as negative.

In order to evaluate the performance of classification models, these four metrics are also combined into a Confusion Matrix, which is a matrix that summarizes the number of correct and incorrect predictions. From the Confusion Matrix, other metrics can be calculated such as Accuracy, Precision, Recall, and F1-Score.

**Accuracy:** An accuracy score is the ratio of the number of correct predictions to the total number of predictions made by the model. It is a simple measure of the overall performance of the model and is suitable for balanced datasets where all classes have a similar number of samples.

(Sir please add the Accuracy in Performance Metrics formula & related description)

**Precision:** The precision of a classification model is the ratio of true positives to the sum of true positives and false positives. It measures the accuracy of the positive predictions made by the model. A high precision indicates that the model is making very few false positive predictions.

(Sir please add the Precision in Performance Metrics formula & related description)

**Recall:** The recall of a classification model is the ratio of true positives to the sum of true positives and false negatives. It measures the ability of the model to correctly identify all positive instances. A high recall indicates that the model is making very few false negative predictions.

(Sir please add the Recall in Performance Metrics formula & related description)

**F1-Score:** The F1 score is the harmonic mean of precision and recall. It provides a balance between the two metrics and is a good overall measure of the performance of the model.

(Sir please add the F1-Score in Performance Metrics formula & related description)

1. Results and Discussion

The performances of models trained using different ML algorithms were assessed on the basis of Performance Metrics (primary), Testing Time, and Model Storage.

* 1. Performance Comparison of Various Classifiers

The classifiers used were Support Vector Machine (SVM) with three different kernels (linear, polynomial, and radial basis function) and with two different parameters as ‘gamma’ = 100, ‘C’ = 5, K-Nearest Neighbors (KNN), Naïve Bayes (Gaussian Naïve Bayes and Bernoulli Naïve Bayes), Decision Tree, Random Forest, and Logistic Regression. We used K-Fold Cross Validation with K=10 for model evaluation.

The highest average accuracy of 94.20% was achieved by Random Forest classifier with n\_estimators = 75, while Decision Tree achieved an average accuracy of 92.02%. Gaussian Naïve Bayes, and Bernoulli Naïve Bayes classifiers achieved an average accuracy of 84.11% and 75.79%, respectively. Among the SVM kernels, linear kernel achieved the highest average accuracy of 82.94% and ‘gamma’ = 100, ‘C’ = 5 achieved the highest precision of 83%. KNN achieved an average accuracy of 72.35%. Logistic Regression achieved an average accuracy of 82.69%.

In terms of further classification report metrics, Random Forest achieved the highest precision, recall, and F1-score of 94%, while Gaussian Naïve Bayes achieved the highest precision, recall, and F1-score among Naïve Bayes classifiers with 85%, 84%, and 84% respectively. Decision Tree efficiently achieved precision, recall, and F1-score of 93%, 92%, and 92% respectively, while SVM with RBF kernel achieved precision, recall, and F1-score of 72%, 71%, and 70%, respectively. The KNN classifier also achieved precision, recall, and F1-score of 73%, 72%, and 72%, respectively, while Logistic Regression achieved good precision, recall, and F1-score of 84%, 82%, and 83%, respectively.

The testing time and model storage of the classifiers were also evaluated. Decision Tree had the shortest testing time of 15.43 µ-sec, while SVM with RBF kernel had the longest testing time of 164.87 µ-sec. The model storage ranged from 2 KB for Naïve Bayes to 1494 KB for Random Forest.

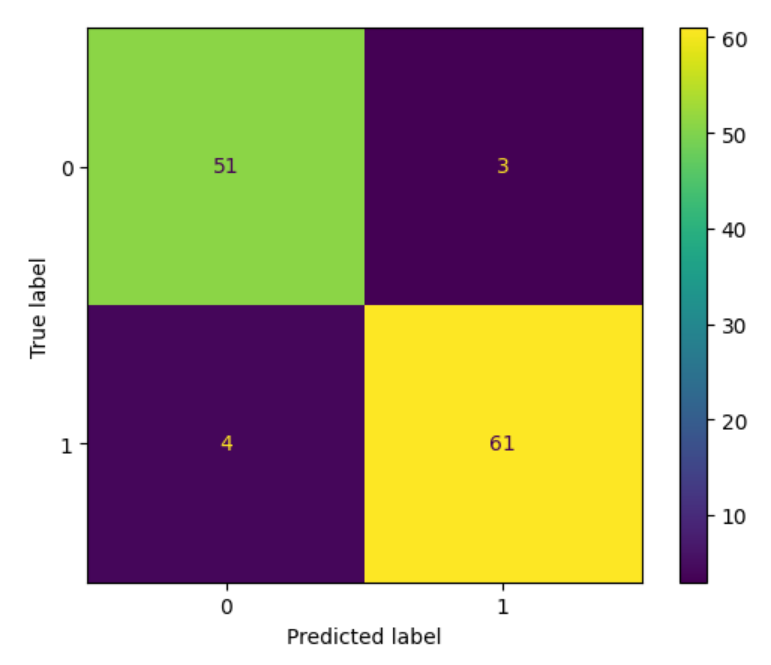
**Table 1.** Performance Metrics Comparison of various ML Classifiers



Overall, the results indicate that Random Forest is the most accurate classifier for heart disease detection with good classification performance, while Decision Tree, SVM, KNN, Naïve Bayes, and Logistic Regression also achieved reasonable accuracy with varying classification performance, testing time, and model storage requirements.

* 1. Confusion Matrix of the Best Classifier

A Confusion Matrix is a table used to evaluate the performance of a classification model by comparing the predicted class labels with the actual class labels. It is a square matrix with the actual classes as rows and predicted classes as columns. The four possible outcomes are true positive (TP), true negative (TN), false positive (FP), and false negative (FN), which represent the number of correct and incorrect predictions made by the model. The confusion matrix of the best classifier (Random Forest Classifier) is shown in Fig. 1. It contains 51 True Positives, 3 False Positives, 4 False Negatives, and 61 True Negatives.



**Fig. 1.** Confusion Matrix of the Random Forest Classifier

1. Conclusion

Machine learning (ML) algorithms have shown great potential in the detection and diagnosis of heart diseases. The use of these algorithms has provided accurate and efficient diagnosis and has improved patient outcomes. Different algorithms such as SVM, K-Nearest Neighbors, Naïve Bayes, Decision Trees, Random Forest, and Logistic Regression have been employed, and their performance is evaluated and compared using metrics such as accuracy, precision, recall, and F1-score. Results also show that some of these ML classifiers, apart from Neural Networks, have great potential and are on-par with the metrics of Neural Networks which are most preferably used nowadays for heart disease detection tasks.

However, there is still room for improvement. The accuracy and reliability of these algorithms depend on the quality of data used for training and testing, and the selection of relevant features that contribute to the diagnosis of heart diseases. The choice of an ideal algorithm is also vital to achieve optimal performance. Further research and development in this field will continue to provide new insights and improvements to these algorithms, and we can expect to see more advanced and accurate models in the future.

References

1. M. C. H. Yap et al.: Deep Learning-Based Automatic Detection and Diagnosis of Cardiovascular Diseases: A Survey. In IEEE Access Journal, vol. 9, pp. 30116–30136, 2021.
2. L. Wang et al.: Deep Learning for Cardiovascular Disease Detection: A Review. In IEEE Access Journal, vol.8, pp. 20704–20731, 2020.
3. H. Zafar et al.: A Machine Learning Framework for Predicting Cardiovascular Diseases. In Proceedings of the 2021 IEEE International Conference on Innovations in Intelligent Systems and Applications (INISTA), Istanbul, Turkey, pp.1–6, 2021.
4. T. Kim et al.: A Deep Learning-Based Approach for Early Diagnosis of Cardiovascular Diseases. In Proceedings of the 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), Bangkok, Thailand, pp. 67–72, 2021.
5. Manu S.: A database for using machine learning and data mining techniques for coronary artery disease diagnosis (Heart Disease Dataset–Comprehensive). In IEEE DataPort, 2020. [Available]: https://ieee-dataport.org/open-access/heart-disease-dataset-comprehensive/.