Heart Disease Classification

Hana Jurić Fot, Pedro Cruz
Complements of Machine Learning, prof. Petia Georgieva
Departamento de Eletrónica, Telecomunicações e Informática
Universidade de Aveiro
{hana.juric.fot, pedro.p.cruz}@ua.pt

***Abstract*—Predicting heart disease is essential for early detection and effective treatment, ultimately improving patient outcomes and lowering mortality rates. This study evaluates the performance of various machine learning models, including Logistic Regression, Support Vector Machine (SVM), Random Forest, K-Nearest Neighbors (KNN), XGBoost, and Artificial Neural Network (ANN), using the Cleveland Heart Disease dataset. The models were evaluated using standard classification metrics, including accuracy, precision, recall, F1 score and ROC-AUC. The Artificial Neural Network demonstrated the highest accuracy at 90%, followed closely by the K-Nearest Neighbors (KNN) model with an accuracy of 88%. However, in terms of ROC-AUC, SVM outperformed all models with a score of 0.9417, followed by ANN with 0.9406. These findings highlight the potential of machine learning models to support clinical diagnosis of heart disease, though further research and validation are needed before such systems can be reliably implemented in real-world healthcare settings.**

***Index Terms*—heart disease prediction, Cleveland heart disease dataset, machine learning, deep learning**

I. Introduction

One of the big challenges in the medical sciences field is the diagnostic and prediction of heart related diseases. As heart diseases are the leading cause of death in the world it is important to develop tools and technologies that help and facilitate the prevention and diagnosis of these diseases.

And it is in this context that a possible solution to the problem is the usage of Machine Learning algorithms. Given certain conditions about a patient the objective is to predict whether the patient has a certain condition or not. Compared to traditional methods, these algorithms can easily pick up on very complex patterns in the data that would be extremely hard for a human person to understand, and this way it could help with predicting diseases that are hard to diagnose via testing.

The main objective of this study is to develop and study how effectively Machine Learning algorithms would perform when given the task of classification, for that purpose some traditional Machine Learning models and a Artificial Neural Network was prepared and compared with the results of other studies.

II. Related work

The application of machine learning in the medical domain has gained significant attention in recent years, driven by its potential to improve diagnostic accuracy and uncover patterns within complex healthcare data. Numerous studies have explored the feasibility of using machine learning algorithms for the early detection and classification of various diseases, including cardiovascular conditions. The Cleveland Heart Disease dataset is one of the most frequently used benchmarks for evaluating machine learning techniques in this area, offering a consistent basis for model comparison.

Paul [1] evaluated several classification algorithms, including Support Vector Machine (SVM), Random Forest, XGBoost, and Artificial Neural Network (ANN). All models demonstrated solid performance, with accuracy exceeding 80% and ROC-AUC values above 0.85. The best results were achieved using XGBoost, with an accuracy of 89.1% and a ROC-AUC score of 0.93.

Anderies et al. [2] compared SVM, Naive Bayes, Logistic Regression, ANN, Decision Tree and K-Nearest Neighbors (KNN). SVM achieved the best performance with an F1 score of 0.87 and an accuracy of 85%. Interestingly, despite its simplicity, Naive Bayes yielded a competitive accuracy of 83.33%.

Suryawanshi [3] explored Logistic Regression, SVM and Gradient Boosting, and further combined these models into a Voting Classifier ensemble. This hybrid approach achieved an impressive accuracy of 97.9%, highlighting the potential of ensemble methods in improving predictive performance.

Shrestha [4] experimented with Logistic Regression, Random Forest, Gradient Boosting, XGBoost and Long Short-Term Memory (LSTM) networks. To accommodate LSTM model, which is typically suited for sequential data, the originally tabular dataset was reshaped into a 3D format, transforming each sample into a pseudo-sequence. Despite this unconventional application, LSTM achieved an accuracy of 85%. However, it was outperformed by simpler models, with Logistic Regression achieving 89% accuracy and XGBoost recording a ROC-AUC of 0.94.

Anderson et al. [5] compared Logistic Regression and Gradient Boosting, concluding that Gradient Boosting consistently outperformed Logistic Regression across all evaluation metrics, reinforcing its effectiveness in structured data scenarios.

Finally, Lin et al. [6] investigated the performance of Convolutional Neural Networks (CNNs) versus standard feed-forward Neural Networks (NNs) on the Cleveland dataset. Their study involved multiple experiments, varying parameters such as the inclusion of categorical variables, class balancing techniques, and network architecture (number of hidden layers and neurons). They found that standard NNs outperformed CNNs, likely due to the limited size of the dataset. The optimal

configuration consisted of two hidden layers with fewer than 20 neurons each.

Based on insights from these previous studies, we selected six commonly used and high-performing models for further evaluation: Logistic Regression, Support Vector Machine (SVM), Random Forest, K-Nearest Neighbors (KNN), XGBoost, and Artificial Neural Network (ANN). Our goal was to systematically assess their performance on the Cleveland Heart Disease dataset using multiple evaluation metrics.

III. Dataset

The dataset chosen was "Heart Disease Cleveland UCI", taken from Kaggle, it is publicly available, and it is processed version of another dataset, "Heart Disease" that was collected at Cleveland Clinic, USA, and donated in 1988, the original dataset presented 76 features but only 14 of them seemed prevalent and used for machine learning purposes. As it was data collected in 1988, it may not reflect in nowadays clinical context, but given the objective of the study it suffices.

The dataset has data that was collected on 297 patients with the focus on studying if given certain features about a patient, it could be accurately measured if that patient had a heart condition or not. Given the objective of the dataset all 297 patients have been measured or tested on 14 different features, age; gender; chest pain type; resting blood pressure; serum cholesterol; fasting blood sugar; resting electrocardiographic results; maximum heart rate achieved; exercise induced angina; ST depression induced by exercise relative to rest; slope of the peak exercise ST segment; number of major vessels colored by fluoroscopy; presence of thalassemia and finally the fourteenth feature, the condition, of which each patient has their own set of features.

Excluding the condition feature, on this dataset we have 7 categorical and 6 numerical features, all scaled differently.

Given the dataset size, there may some bias on the performance of some models, such as the Neural Network and ensemble models. Besides the size there may be limitations on the use of this dataset as all data is from Cleveland, USA, which may present some bias if tested with another demographic group.

A. Data Preprocessing

Preprocessing is an important part of training classification models and neural networks, as these models only work as well as the data allows them to. Preprocessing data is the process of preparing data to be used, and for data to be usable, missing values should be checked as some models do not work with missing values. In this case the dataset presents no missing values so no further processing in that regard is needed.

Histograms were created, showing the distribution of data on all the features. It's clear that numerical categories present a somewhat Gaussian curve, resembling a normal distribution apart from the number of major vessels colored by fluoroscopy (ca). In addition to numerical features, it is interesting to observe the distribution of categorical features as some seem unbalanced, these distributions shown are a feature of our data set, so no balancing will be applied. The most important feature to analyze, is the condition feature as it is our target feature, it doesn't really seem unbalanced given the ratio of patients with condition being 0.46 and the ratio with patients with no condition being 0.54.

Then, the next step would be to prepare the categorical data to be used to train our neural network and classification models, in order to do that, One-Hot Encoding was applied to all the suitable categorical variables. One-Hot Encoding is the process of separating a variable into several dummy variables, depending on the number of categories the feature has, this way it's common to end up with a different and bigger dataframe shape than when we started. Further processing was applied by eliminating the first dummy resulting of the One-Hot Encoding to avoid redundancy as any dummy can be easily predicted given all the others, that's known as multicollinearity.

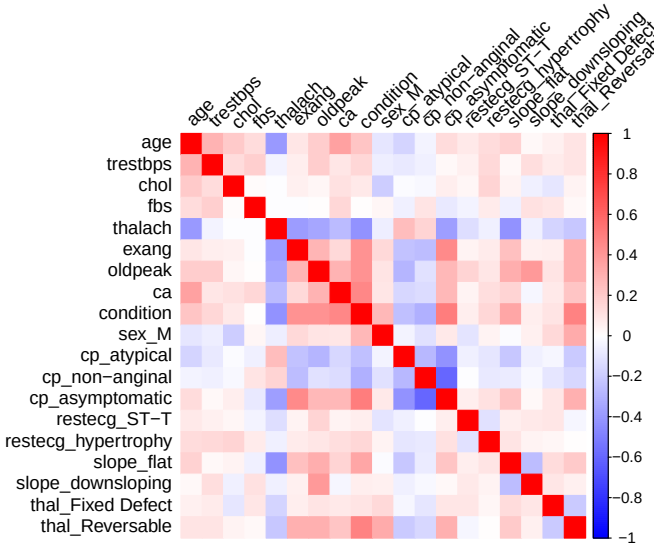

Fig. 1. Features Correlation Heatmap.

In figure 1 it's noticeable that most features present low correlations between each other. The highest correlation seems to be the asymptomatic chest pain with non anginal chest pain with a correlation ration of 0.596. As all the categorical features already had dropped a feature to avoid multicollinearity, no further feature removal was practiced.

It was proceeded with the normalization of data, the sklearn Standard Scaler was used as the numerical presented normal behavior, it scales data to having mean 0 and standard deviation of 1. Normalization of data is a must for machine learning practices as it avoids bias by not attributing higher or lower weights to features in different scales.

The data was then divided into the target and the rest of the dataframe. Was also applied shuffling on the data. Then the data was split into training and testing with a ratio of 80 of the data on the training set and 20 on the testing set, also for reproducibility, a seed was set on 42. As a final processing of

the data the models were prepared to run on a Stratified K-Fold of 5 splits, the K-Fold was also shuffled and set a seed on 42 for reproducibility. At some point during this process some models were tested and were not performing well, only then was the data shuffled, and the models performed a lot better. This behavior could mean that the data was ordered into stratified sets of data, and training models under those conditions could develop biased results.

IV. Methodology

The use of machine learning (ML) in medicine has the potential to significantly enhance diagnostic accuracy, enable earlier detection of diseases, and support clinical decision making. In the context of heart disease, where timely diagnosis is critical, ML models can help identify patterns and risk factors that may not be immediately evident to human practitioners. By leveraging patient data such as clinical measurements and personal health attributes, ML algorithms can assist in predicting the likelihood of heart disease.

For this project, we framed the task as a binary classification problem, aiming to predict whether a patient has heart disease based on features from the Cleveland UCI Heart Disease dataset. We decided to evaluate the following machine learning algorithms: Logistic Regression, Support Vector Machine (SVM), Random Forest, K-Nearest Neighbors (KNN), XGBoost and Artificial Neural Network.

A. Logistic Regression

Logistic regression is a classification algorithm used for binary classification. It uses the sigmoid function to map linear combinations of the inputs to values between 0 and 1, representing probabilities. The model is trained by minimizing the binary cross-entropy loss, which measures the difference between predicted probabilities and actual class labels. This approach is efficient and interpretable, making it suitable for many real-world applications.

Logistic Regression has demonstrated strong performance in numerous studies [2], [4], [5], making it a suitable candidate for our analysis. Motivated by these findings, we included Logistic Regression in our experiments. Using grid search for hyperparameter tuning, we identified the optimal regularization parameter as $C = 1$.

B. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm used for classification tasks by finding the optimal hyperplane that best separates data points of different classes. It maximizes the margin between the classes, which improves generalization. During training, SVM minimizes hinge loss, penalizing misclassified and marginally classified samples. With the use of kernel functions, SVM can also model non-linear decision boundaries effectively.

Support Vector Machines (SVM) achieved the best performance in the study by Anderies et al. [2], which motivated us to include SVM in our experiments as well. For hyperparameter tuning, we considered a grid of values for the regularization parameter C, kernel type, and the kernel coefficient γ, as follows:

- $C \in \{0.01, 0.1, 1, 10, 100\}$
- Kernel: `linear`, `rbf`
- $\gamma \in \{\texttt{scale}, \texttt{auto}\}$ (applicable to the `rbf` kernel)

Grid search identified the optimal hyperparameters as $C = 1$, `kernel` = `rbf`, and $\gamma =$ `auto`.

C. Random Forest

Random Forest is a machine learning algorithm based on ensemble learning, it is commonly applied to classification problems but also performs well in regression problems. Random Forest is based on decision trees but has been improved with a bootstrap aggregating mechanism, that randomly picks any samples into a bootstrap dataset, then picks a number of random features and trains a split using the bootstrap dataset and the picked features, then repeats any arbitrary number of times. The way Random Forest predicts the outcome, in a classification problem, is by keeping track of the number of trees that vote each outcome, and the classification that gets predicted is the one that has the highest number of votes.

The motivation on including Random Forest on this study, is that it is a model that behaves well in classification problems, but unlike in the study by Shresta et al. [4], it performs the worst of all models.

In the attempt to find the best parameters, the following grid of parameters was run on a grid search:

- `n_estimators` $\in \{50, 100, 200\}$
- `max_depth` $\in \{3, 5, 7, \texttt{none}\}$
- `min_samples_split` $\in \{2, 5, 10\}$
- `min_samples_leaf` $\in \{1, 2, 4\}$
- `max_features` $\in \{\texttt{sqrt}, \texttt{log2}, \texttt{none}\}$
- `bootstrap` $\in \{\texttt{True}, \texttt{False}\}$

The following best parameters were found: `bootstrap` = `True`, `max_depth` = 5, `max_features` = `sqrt`, `min_samples_leaf` = 4, `min_samples_split` = 10, and `n_estimators` = 200.

D. K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm used for classification tasks. It assigns a class label based on the majority vote of the k nearest neighbors in the feature space, typically determined by a distance metric. Key hyperparameters include the number of neighbors (k), the choice of distance metric (Euclidean or Manhattan), and the weighting scheme (uniform or distance-based). As KNN has no explicit training phase, its performance is highly dependent on the quality and scale of the input features.

The K-Nearest Neighbors (KNN) algorithm also demonstrated strong performance in the study by Anderies et al. [2], achieving a precision of 0.90 and an F1-score of 0.81. Motivated by these results, we included KNN in our evaluation. For hyperparameter tuning, we explored the following parameter grid:

- `n_neighbors` $\in \{5, 7, 9, 15, 30\}$
- `weights` $\in \{$`uniform`, `distance`$\}$
- `algorithm` $\in \{$`auto`, `ball_tree`, `kd_tree`, `brute`$\}$
- `p` $\in \{1, 2\}$
- `metric` $\in \{$`euclidean`, `manhattan`$\}$

Grid search identified the best-performing combination as: `n_neighbors` = 5, `weights` = `uniform`, `metric` = `manhattan`, and $p = 1$.

E. XGBoost

Extreme Gradient Boosting, also known as XGBoost is a multipurpose machine learning algorithm that is widely used for classification and regression. It stems from Gradient Boosting but has built in regularization parameters. The way it works is by averaging the target feature, which has to be attributed to the model as the default value to start the process is 0.5, then measuring the residual values and iteratively creating trees that minimize the residual value by fixing the previous trees errors.

XGBoost was added in this study because it is regarded as one of the best classification models given its various vantages, such as easy regularization and versatility, when comparing to Shrestha's et al. [4] results both models performed about the same.

The following parameters were tested to find the best possible combination to achieve the highest accuracy:

- `learning_rate` $\in \{0.01, 0.05\}$
- `max_depth` $\in \{3, 4\}$
- `n_estimators` $\in \{100, 200\}$
- `subsample` $\in \{0.6, 0.8\}$
- `colsample_bytree` $\in \{0.6, 0.8\}$
- `reg_alpha` $\in \{0.1, 0.5\}$
- `reg_lambda` $\in \{5, 10\}$
- `gamma` $\in \{0.1, 0.5\}$

The best parameters found on the grid search were: `colsample_bytree` = 0.6, `gamma` = 0.5, `learning_rate` = 0.05, `max_depth` = 4, `n_estimators` = 100, `reg_alpha` = 0.1, `reg_lambda` = 5 and `subsample` = 0.6.

It's important to note that these parameters were not the first to be applied, the first group of parameters that were checked into the grid search overfitted the model so the parameters had to be changed to generalize better.

F. Artificial Neural Network

Artificial Neural Networks (ANNs) are powerful machine learning models inspired by the structure and function of the human brain. They consist of layers of interconnected nodes (neurons), where each connection carries a weight that is adjusted during training. Through iterative optimization methods such as backpropagation and gradient-based algorithms, ANNs can effectively learn complex, non-linear relationships in data. This flexibility makes them highly effective for a wide range of applications.

In the context of heart disease prediction, ANNs have shown strong performance in several studies. Lin et al. [6] reported an accuracy of 93% using a neural network with two hidden layers and fewer than 20 neurons per layer, outperforming convolutional architectures on the same dataset. Similarly, Anderies et al. [2] achieved a competitive accuracy of 80% with a standard ANN architecture.

While ANNs offer significant potential, their performance is highly dependent on the choice of hyperparameters such as the number of hidden layers, the number of neurons per layer, batch size, number of epochs, dropout rate, and regularization strength.

To identify the optimal architecture for our ANN model, we initially fixed several hyperparameters to commonly used baseline values: 50 training epochs, a batch size of 32, a dropout rate of 0.3 (`dropout_rate` = 0.3), and L_2 regularization with $\lambda = 0.01$. With these settings in place, we focused on exploring different network architectures by varying the number of hidden layers (1, 2, and 3) and the number of neurons in each layer.

Using 5-fold stratified cross-validation, we found that the best performance was achieved with a two-layer architecture (consistent with the findings of Lin et al. [6]). However, in contrast to their model, which used fewer than 20 neurons per layer, our best-performing configuration used 64 and 32 neurons in the first and second layer, respectively. This architecture yielded a mean cross-validation accuracy of 0.8485 with a standard deviation of 0.0545.

After establishing the optimal architecture, we proceeded to fine-tune the remaining hyperparameters. Specifically, we evaluated a grid of values for batch size (`batch_size` $\in \{16, 32, 64\}$), number of training epochs (`epochs` $\in \{50, 100, 150\}$), L_2 regularization strength ($\lambda \in \{0.001, 0.01, 0.1\}$), and dropout rate (`dropout_rate` $\in \{0.3, 0.5\}$).

Grid search was used to identify the best combination of these parameters, which resulted in the following optimal values: `batch_size` = 64, `epochs` = 50, λ = 0.1, and `dropout_rate` = 0.5. These parameters were then used in our final model training and evaluation.

V. Results

A. Evaluation Metrics

To evaluate and compare the performance of different classification models, we used several standard metrics commonly applied in machine learning. In the formulas below, TP (true positives), TN (true negatives), FP (false positives) and FN (false negatives) represent the counts of correct and incorrect predictions with respect to the actual class labels.

- **Accuracy**: Measures the proportion of correct predictions among the total number of predictions.

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

- **Precision**: Measures the proportion of true positives among all predicted positives.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall (Sensitivity or True Positive Rate)**: Measures the proportion of true positives among all actual positives.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1 Score**: Harmonic mean of precision and recall. It balances the two metrics.

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

- **ROC-AUC (Receiver Operating Characteristic - Area Under Curve)**: Measures the area under the ROC curve, which plots the true positive rate against the false positive rate at various threshold settings.

B. Performance Evaluation

This section presents the outcomes of our experiments, highlighting the performance of the evaluated machine learning models on the Cleveland Heart Disease dataset. Each model was assessed using standard classification metrics described in the previous subsection.

To evaluate the risk of overfitting, we measured the accuracy of each model on both the training and test sets. Table I presents a comparison of these results. For most models, the difference between training and test accuracy is minimal, indicating good generalization and low risk of overfitting. An interesting observation arises with the Neural Network model, where the test accuracy (0.90) exceeds the training accuracy (0.81). This is due to the use of dropout during training, which acts as a regularization technique by randomly deactivating neurons and reducing overfitting. During testing, dropout is disabled, allowing the entire network to contribute, which results in improved performance.

TABLE I
TRAINING AND TEST ACCURACY OF ALL MODELS

Model	Train Accuracy	Test Accuracy
Logistic Regression	0.88	0.87
SVM	0.89	0.87
Random Forest	0.92	0.85
KNN	0.85	0.88
XGBoost	0.89	0.85
Neural Network	0.81	0.90

Table II summarizes the performance of all models on the test set using four key evaluation metrics: accuracy, precision, recall, and F1 score. All models performed well, achieving accuracies above 0.85. The Neural Network delivered the strongest overall performance, with an accuracy of 0.90, precision of 0.90, recall of 0.89, and an F1 score of 0.90. The K-Nearest Neighbors (KNN) model also yielded competitive results, attaining both an accuracy and F1 score of 0.88.

Figures 2 to 7 display the confusion matrices for each of the evaluated models. These matrices provide a detailed view of the models' classification performance, including the number of true positives, true negatives, false positives, and false negatives.

TABLE II
PERFORMANCE ON TEST SET FOR ALL MODELS (MACRO AVERAGES)

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.87	0.87	0.86	0.86
SVM	0.87	0.87	0.86	0.86
Random Forest	0.85	0.85	0.84	0.84
KNN	0.88	0.88	0.89	0.88
XGBoost	0.85	0.85	0.84	0.84
Neural Network	0.90	0.90	0.89	0.90

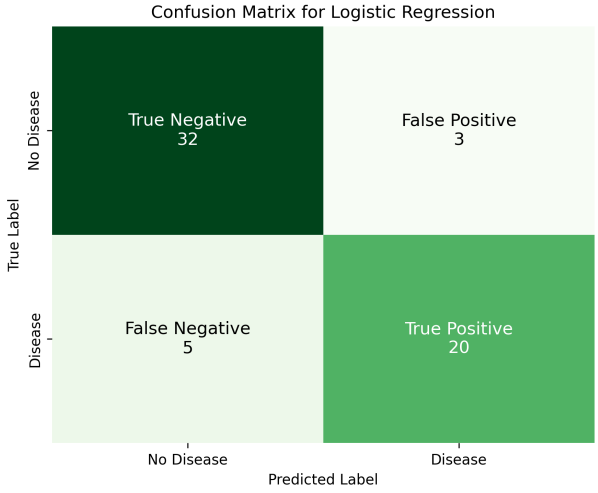

Fig. 2. Confusion matrix for Logistic Regression

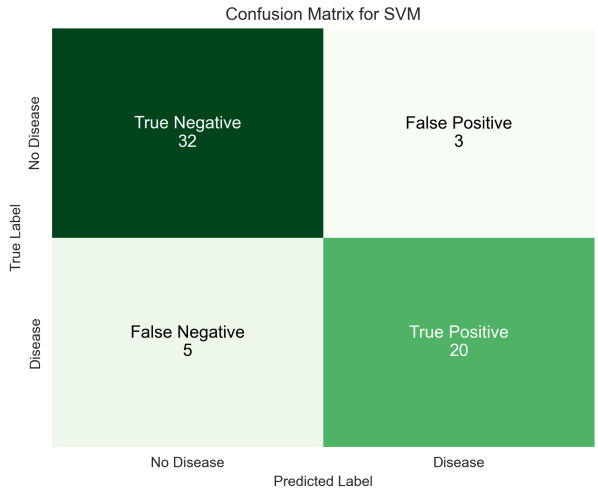

Fig. 3. Confusion matrix for Support Vector Machine (SVM)

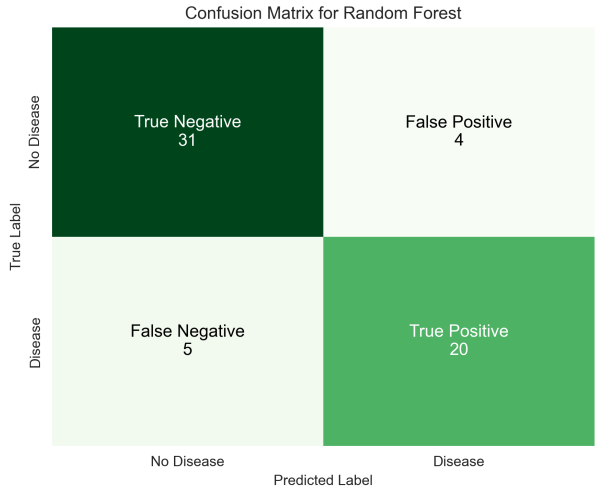

Fig. 4. Confusion matrix for Random Forest

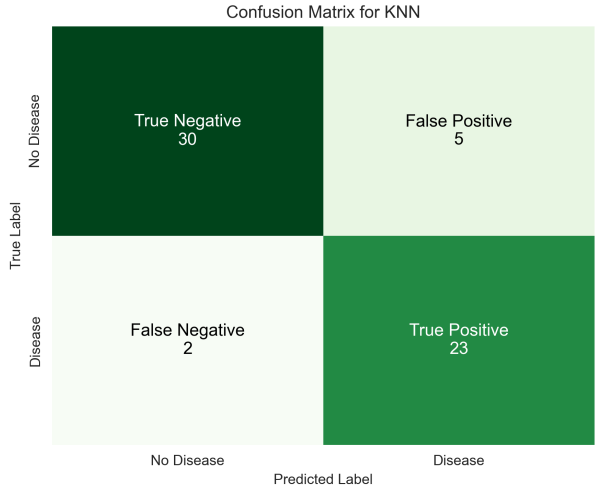

Fig. 5. Confusion matrix for K-Nearest Neighbors (KNN)

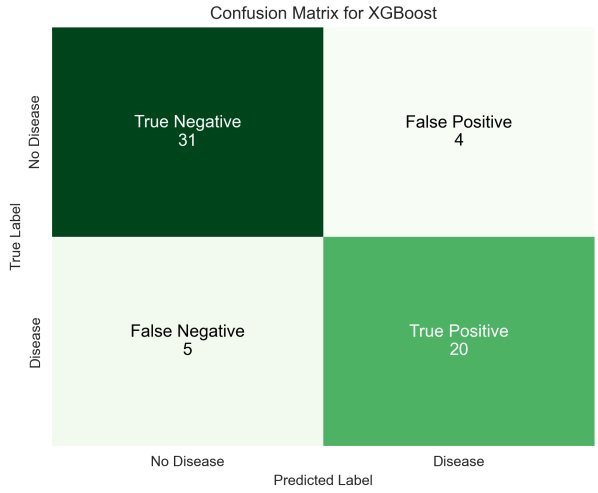

Fig. 6. Confusion matrix for XGBoost

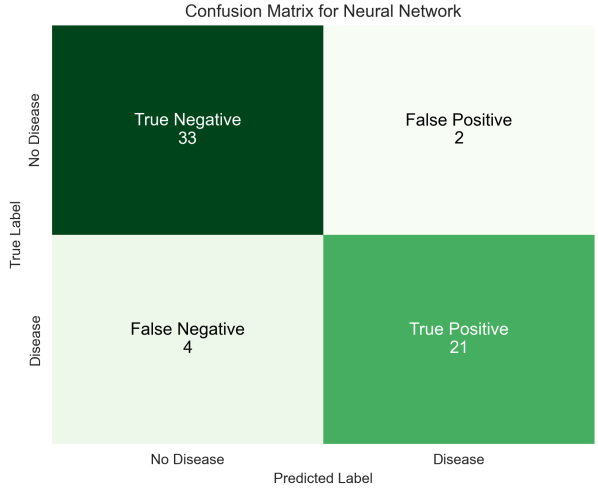

Fig. 7. Confusion matrix for Artificial Neural Network

C. Model Comparison

To further compare the models, we plotted a graph showing the ROC curves for all classifiers, which can be seen in Figure **??**. The ROC curve illustrates the trade-off between the true positive rate and false positive rate at various classification thresholds. A higher area under the curve (AUC) indicates better overall classification performance. As shown, all models perform well, with the curves closely approaching the top-left corner. Among the models, the Support Vector Machine (SVM) achieved the highest AUC of 0.9417, followed closely by the Artificial Neural Network (ANN) with an AUC of 0.9406.

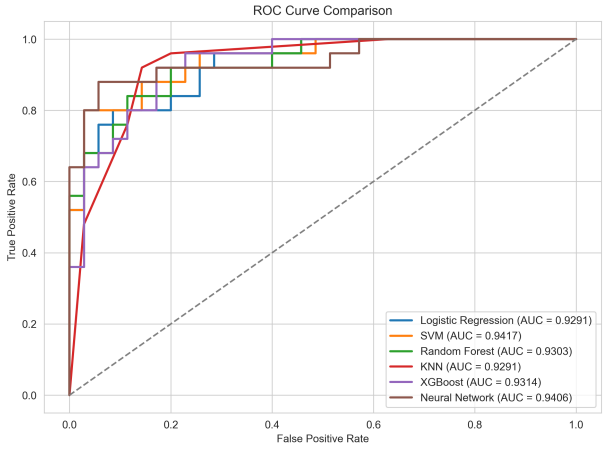

Fig. 8. Comparison of models using ROC curves

VI. Conclusion

This study aimed to explore the application of machine learning algorithms for predicting heart disease. The results indicate that the Artificial Neural Network (ANN) achieved

the highest test accuracy of 0.90, followed closely by the K-Nearest Neighbors (KNN) model with a test accuracy of 0.88.

While these models performed well, it is important to note that the dataset used in this study, sourced from 1988, may not fully reflect current medical practices or advances. Additionally, the relatively small size of the dataset (297 samples) limits the models' potential, particularly for algorithms like ANN and XGBoost that typically require larger datasets for optimal performance.

For future work, it is recommended that these models be tested on a larger, more recent dataset with additional features that could further improve predictive accuracy. Such improvements could lead to more robust and reliable models for early detection and diagnosis of heart disease.

ACKNOWLEDGMENT

The contributions of each team member are as follows: Pedro Cruz was responsible for the description and preprocessing of the dataset, implemented the Random Forest and XGBoost models, and wrote the conclusion section. Hana Jurić Fot implemented the Logistic Regression, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Neural Network models, wrote the related work and results sections, and conducted an extensive review of relevant literature.

REFERENCES

[1] J. Paul, "A comprehensive study of advanced machine learning algorithms for predicting heart disease using the cleveland dataset," *researchgate.net*, 2024.

[2] A. Anderies, J. A. R. W. Tchin, P. H. Putro, Y. P. Darmawan, and A. A. S. Gunawan, "Prediction of heart disease uci dataset using machine learning algorithms," *Engineering, MAthematics and Computer Science Journal (EMACS)*, vol. 4, no. 3, pp. 87–93, 2022.

[3] N. S. Suryawanshi, "Accurate prediction of heart disease using machine learning: A case study on the cleveland dataset," *researchgate.net*, 2024.

[4] D. Shrestha, "Advanced machine learning techniques for predicting heart disease: A comparative analysis using the cleveland heart disease dataset," *Applied Medical Informatics*, vol. 46, no. 3, 2024.

[5] K. Anderson, "Techniques using the cleveland dataset predictive modeling in healthcare: Comparing logistic regression and gradient boosting for heart disease detection," *researchgate.net*, 2024.

[6] C.-H. Lin, P.-K. Yang, Y.-C. Lin, and P.-K. Fu, "On machine learning models for heart disease diagnosis," in *2020 IEEE 2nd Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS)*. IEEE, 2020, pp. 158–161.