Researchers Journal of Science and Technology (REJOST)
https://rejost.com.ng

Optimized Stacking Ensemble of Random Forest and XGBoost for Heart Disease Prediction

U. Sani[1], A.A Abubakar [2], D.G. Lazarus[1], A.A. Babajo[1]

Abstract
Heart disease remains a leading cause of death worldwide, highlighting the need for reliable tools to support early diagnosis and clinical decision-making. Although machine learning methods have been widely applied to heart disease prediction, many existing studies rely on default model settings, limited evaluation metrics, and a narrow range of algorithms. This study develops an optimized stacking ensemble model that combines Random Forest and XGBoost, with hyperparameters tuned using Optuna. The models were evaluated using a publicly available clinical dataset comprising 918 patient records with demographic and cardiovascular attributes. An 80:20 train-test split was applied, and performance was assessed using multiple metrics, including accuracy, macro and weighted F1-score, ROC-AUC, PR-AUC, and balanced accuracy. The results show that the optimized Random Forest achieved an accuracy of 89.1% and a ROC-AUC of 93.4%, while XGBoost achieved an accuracy of 88.6% with a similar ROC-AUC. The stacking ensemble provided the best overall performance, with an accuracy of 89.7%, macro F1-score of 89.6%, ROC-AUC of 93.6%, and PR-AUC of 93.5%. These findings suggest that combining optimized ensemble models can improve predictive performance compared to individual classifiers. However, the results are based on a secondary dataset and have not been validated in real clinical settings, which may limit their direct applicability in practice. Overall, this study contributes by demonstrating the value of hyperparameter optimization and model stacking, while adopting a broader evaluation framework for heart disease prediction.

✉ U. Sani: kulthumummu@ymail.com

[1]Department of Informatics, Kaduna State University, Nigeria

[2]Department of Secure Computing, Kaduna State University, Nigeria

Keywords: Heart disease prediction; Random Forest; XGBoost; stacking ensemble; hyperparameter optimization

Article history

Received: February 13, 2026
Received in revised form: April 12, 2026
Accepted for publication: April 15, 2026
Available online: April 19, 2026

1.0 Introduction

Heart disease remains a major global health concern, affecting millions of individuals worldwide and placing significant pressure on healthcare systems due to its high rates of morbidity and mortality (Shahim et al., 2023). Common clinical manifestations, including shortness of breath, fatigue, chest discomfort, and peripheral edema, often reflect underlying cardiovascular dysfunction. Early and accurate prognostication is therefore essential for guiding clinical decision-making and improving patient outcomes (Argulian & Narula, 2021; Arunachalam & Rekha, 2022). Despite advances in medical treatment, the prediction for many heart disease patients remains suboptimal, highlighting the

need for more effective predictive models to support timely and targeted interventions (Lam et al., 2023).

Traditional prognostic approaches, such as Logistic Regression and Support Vector Machines, have been widely used in cardiovascular risk prediction. However, these methods often struggle to capture the complex, nonlinear relationships present in clinical data (Kokori et al., 2024). Recent studies have further emphasized the limitations of such conventional techniques, particularly in handling high-dimensional and heterogeneous healthcare datasets (Saqib et al., 2024; Liang et al., 2024). As a result, there is increasing interest in leveraging more advanced computational methods to improve predictive performance.

Machine learning (ML) has emerged as a promising alternative due to its ability to model complex patterns and interactions within large datasets. Among ML approaches, ensemble learning techniques have gained prominence for their ability to improve prediction accuracy by combining multiple models. Methods such as Random Forest (a bagging approach) and XGBoost (a boosting approach) have demonstrated strong performance in various classification tasks, as they effectively balance bias and variance while enhancing generalization (Javaid et al., 2022; Khan et al., 2023).

Despite these advancements, several gaps remain in the existing literature. First, many studies rely on default hyperparameter settings, which can lead to suboptimal model performance. Second, evaluations are often limited to basic metrics such as accuracy, without considering more informative measures like ROC-AUC, PR-AUC, and balanced accuracy. Third, most studies focus on individual models rather than exploring optimized ensemble combinations that can leverage complementary strengths. These limitations reduce the robustness and practical relevance of existing models in clinical contexts.

To address these gaps, this study proposes an optimized stacking ensemble model that integrates Random Forest (RF) and XGBoost as base learners, with Logistic Regression as the meta-learner. The models are systematically tuned using the Optuna hyperparameter optimization framework and evaluated on a publicly available dataset comprising 918 patient records with clinical and demographic features.

The main objective of this study is to improve the predictive performance of heart disease prediction models through optimized ensemble learning. Specifically, the study aims to:

i. Develop and optimize Random Forest and XGBoost models using automated hyperparameter tuning.
ii. Construct a stacking ensemble model that combines the strengths of both algorithms.
iii. Evaluate model performance using an extended set of metrics, including accuracy, F1-score, ROC-AUC, PR-AUC, and balanced accuracy.
iv. Compare the performance of the proposed model with individual base learners.

The key contributions of this study are as follows:

I. The development of an optimized stacking ensemble framework for heart disease prediction.
II. The application of systematic hyper-parameter tuning using Optuna to enhance model performance.
III. The use of a comprehensive evaluation framework that goes beyond traditional accuracy-based assessment.
IV. An empirical demonstration of the effectiveness of ensemble learning in improving predictive reliability on clinical data.

2.0 Materials and Methods

2.1 Research Design

This study adopts an experimental research design focused on the comparative evaluation of machine learning models for heart disease

prediction. The design enables systematic testing of different algorithms and configurations to determine their predictive accuracy and clinical relevance. In particular, the study develops and evaluates an optimized stacking ensemble comprising Random Forest (RF) and Extreme Gradient Boosting (XGBoost), tuned using Optuna for hyperparameter optimization. The results of the proposed models are compared against a baseline lightweight metamodel. The flow diagram is illustrated in Fig. 1.

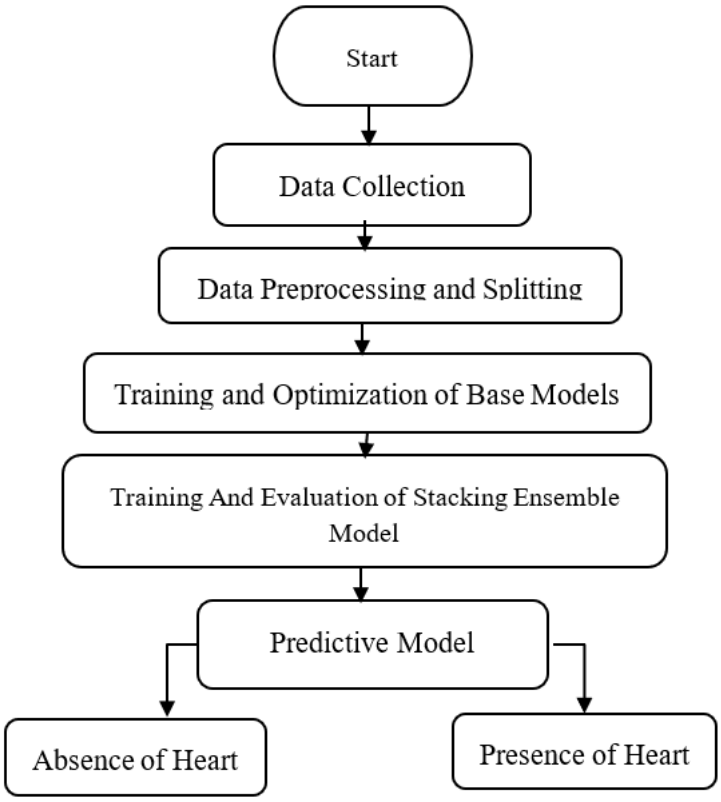

Fig. 1: Research Process Flow Diagram

2.2 Dataset Description

The dataset employed in this study consists of 918 patient records and 12 clinical attributes, sourced from Kaggle (*Heart Failure Prediction Dataset*, 2021). The features include demographic variables (Age, Sex), symptomatic indicators (Chest Pain Type, Exercise Angina, ST_Slope), and clinical measurements (Resting BP, Cholesterol, Fasting BS, Resting ECG, Max HR, Old peak). The target variable, heart disease, is binary, and coded as 1 (presence of heart disease) and 0 (absence of heart disease). The dataset offers a representation of features relevant to cardiovascular prognosis. However, as with many healthcare datasets, it presents challenges such as class imbalance, categorical heterogeneity, and the risk of overfitting due to limited sample size. Presented in Fig. 2.

2.3 Data Preprocessing

Preprocessing was carried out in multiple steps to ensure data quality and suitability for machine learning models:

i. Handling Categorical Variables: Features such as Sex, Chest Pain Type, Resting ECG, Exercise Angina, and ST_Slope were encoded using one-hot encoding to convert categorical attributes into machine-readable binary vectors.
ii. Normalization: Features such as Age, Resting BP, Cholesterol, Max HR, and old peak were standardized using Standard Scaler to ensure comparability and improve the convergence of algorithms.
iii. Data splitting: The dataset was divided into training and test sets using an 80:20 stratified split to maintain class distribution. The training set was used for

model fitting and optimization, while the test set was reserved for final evaluation.

2.4 Model Development

2.4.1 Random Forest (RF)

Random Forest is an ensemble learning algorithm that constructs multiple decision trees and aggregates their predictions to reduce variance and improve accuracy. In this study, Random Forest was tuned using Optuna across hyperparameters including the number of estimators, maximum depth, minimum samples per split, and feature selection strategy.

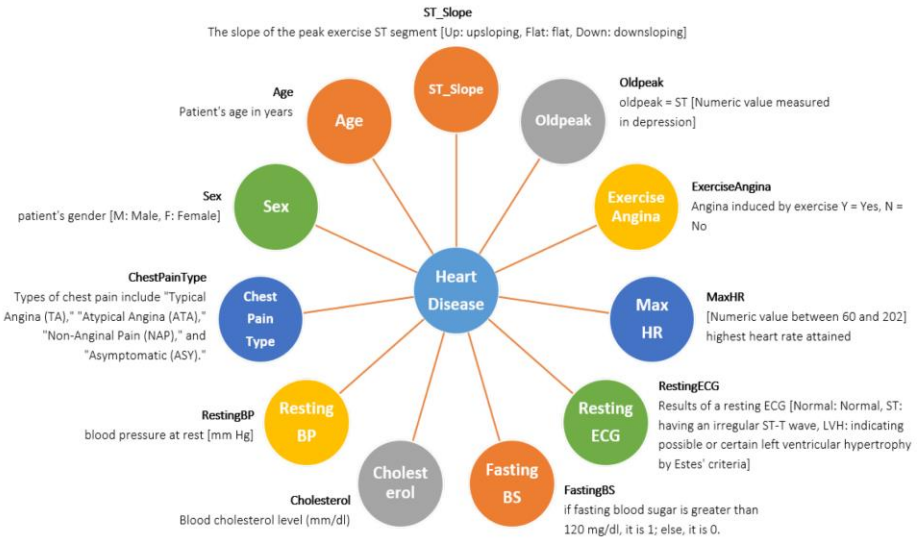

Fig. 2: Details of Features in the Dataset (Mahmud et al., 2023)

2.4.2 XGBoost

XGBoost is a gradient boosting framework known for its scalability and superior performance in structured data classification. The hyperparameters optimized using Optuna included the number of estimators, maximum depth, learning rate, subsample ratio, and column sampling ratio. The imbalance in the dataset was addressed by setting scale_pos_weight.

2.4.3 Stacking Ensemble

To exploit complementary strengths, a stacking ensemble was implemented to combine the strengths of RF and XGBoost. The base learners were the optimized RF and XGBoost models, while Logistic Regression served as the meta-learner. Logistic Regression was chosen as the meta-learner for its simplicity, interpretability, and effectiveness in combining base model predictions. This approach was chosen because stacking often yields higher predictive performance by leveraging complementary strengths of base classifiers.

2.4.4 Hyperparameter Optimization

Hyperparameter optimization was conducted using Optuna, a state-of-the-art framework that applies Bayesian optimization for efficient search. Each trial evaluated a unique set of hyperparameters, and the objective function maximized the macro F1-score on the validation set. Thirty trials were executed for each model to balance computational cost and optimization depth. Unlike grid or random search, Optuna dynamically adjusts the search space to converge more quickly on optimal solutions, making it particularly suitable for computationally intensive ensemble methods. The specific hyperparameters tuned for Random Forest (RF) and XGBoost (XGB) are summarized in Table 1.

2.4.5 Performance Metrics

Model performance was assessed using both traditional and extended evaluation metrics to ensure comprehensive analysis. These included: Accuracy: Overall proportion of correctly classified cases.

Table 1: Hyperparameters Tuned for RF and XGB Models

Model	Hyperparameter	Search Space / Values Explored	Rationale
Random Forest	n_estimators	100 – 500 (integer)	Controls the number of trees in the forest; more trees reduce variance.
	max_depth	3 – 20 (integer)	Prevents overfitting by limiting tree depth.
	min_samples_split	2 – 20 (integer)	Minimum samples required to split an internal node.
	min_samples_leaf	1 – 10 (integer)	Minimum samples required at a leaf node; prevents very small splits.
	max_features	{"sqrt", "log2"}	Number of features considered for best split; balances bias-variance.
	class_weight	"balanced" (fixed)	Addresses class imbalance by weighting the minority class.
XGBoost	n_estimators	100 – 500 (integer)	Number of boosting rounds (trees).
	max_depth	3 – 10 (integer)	Maximum depth of individual trees; controls complexity.
	learning_rate	0.01 – 0.3 (float)	Step size shrinkage; balances bias and variance.
	Subsample	0.5 – 1.0 (float)	Fraction of training samples used per tree; prevents overfitting.
	colsample_bytree	0.5 – 1.0 (float)	Fraction of features sampled per tree; improves generalization.
	scale_pos_weight	(# negatives / # positives) (computed from dataset)	Addresses class imbalance by weighting the minority class.
	eval_metric	"logloss" (fixed)	Chosen for probabilistic classification and consistency across folds.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

where TP is True Positive, which represents the number of correctly predicted positive cases, TN is True Negative representing the number of correctly predicted negative cases, FP is False positive representing the number of incorrectly predicted positive cases and FN is False Negative representing the number of incorrectly predicted negative cases

Precision (macro and weighted): The proportion of true positives among predicted positives.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall (macro and weighted): The proportion of true positives identified among actual positives.

$$Sensitivity\ (Recall) = \frac{TP}{TP + FN} \quad (3)$$

F1-score (macro and weighted): The harmonic mean of precision and recall.

ROC-AUC: Area under the receiver operating characteristic curve. Measures overall separability between classes, and robust to class imbalance.

PR-AUC: Area under the precision-recall curve, particularly relevant in imbalanced datasets. Balanced Accuracy: Average of sensitivity and specificity, useful for imbalanced classification. Confusion Matrix: To visualize classification errors across classes. Additionally, learning curves were generated to analyze model training behavior. Additionally, learning curves were generated to analyze model training behavior.

2.5 Tools and Environment

All experiments were conducted using Python 3.12. Key libraries included Scikit-learn data preprocessing, Random Forest, and evaluation metrics; XGBoost for gradient boosting; and Optuna for hyperparameter optimization. Visualizations were implemented with Matplotlib and Seaborn. Computations were executed in a Jupyter Notebook environment running on Google Colab. The experiments were executed on a CPU-based Google Compute Engine instance with 13 GB of RAM and a disk size of 108 GB.

2.6 Ethical Considerations

The dataset employed in this study was obtained from Kaggle, where it is made publicly available by the data provider. As the data is open-access and does not contain any personally identifiable information, explicit consent from participants was not required for this research. Nonetheless, care was taken to ensure that all data handling procedures adhered to best practices for data privacy, including the secure storage and processing of the dataset. Researchers were mindful of maintaining the confidentiality of the information and ensured that the analysis was conducted in a manner that respects the ethical considerations inherent in handling health-related data.

3.0 Results

3.1 Model Performance Results

This section presents the experimental results of the optimized Random Forest (RF), XGBoost (XGB), and the Stacking Ensemble models. Each model was trained and evaluated on the clinical heart disease dataset, with preprocessing and hyperparameter optimization steps discussed in section 2. The results provide strong evidence that optimized ensemble methods are effective for heart disease prediction, with the stacking ensemble achieving the highest predictive performance across most metrics.

3.1.1 Performance Results of Random Forest Model

The optimized Random Forest model demonstrated robust classification capabilities. The classification report indicated balanced performance across accuracy, precision, recall, and F1-score, suggesting the model effectively captured both positive and negative cases of heart disease. The ROC-AUC of 0.934 and PR-AUC of 0.930 further highlight its discriminative power as shown in Table 2.

The confusion matrix revealed that most predictions were correctly classified, with only a modest number of false negatives as shown in Fig. 3. This is particularly important in healthcare contexts where underdiagnosis of heart disease can have severe implications.

3.1.2 Performance Results of XGBoost Model

The optimized XGBoost model achieved similar performance compared to optimized Random Forest across most metrics, reflecting its strength as a gradient boosting algorithm capable of capturing non-linear feature interactions.

Table 2: Optimized Random Forest Model Classification Report

Precision	Recall	F1-Score	Accuracy	ROC-AUC	PR-AUC	Balanced Accuracy
0.8909	0.8888	0.8897	0.8913	0.9338	0.9304	0.8888
0.8912	0.8913	0.8912	—	—	—	—

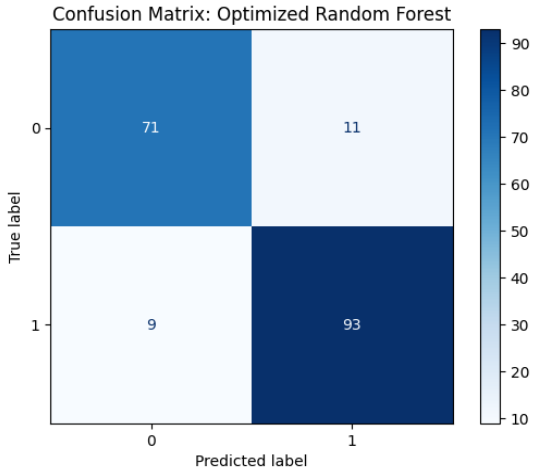

Fig. 3: Optimized Random Forest Model Confusion Matrix

From Table 3, precision and recall scores were notably strong, suggesting that the model not only reduced false positives but also maintained sensitivity to true disease cases. The ROC-AUC (0.934) and PR-AUC (0.932) were comparable to optimized Random Forest. The confusion matrix in Fig. 4 showed slightly more misclassifications of the negative class compared to that of the optimized Random Forest model, resulting in a balanced accuracy of 0.884

Table 3: Optimized XGBoost Model Classification Report.

Precision	Recall	F1-Score	Accuracy	ROC-AUC	PR-AUC	Balanced Accuracy
0.8849	0.8839	0.8844	0.8859	0.9340	0.9321	0.8839
0.8858	0.8859	0.8858	—	—	—	—

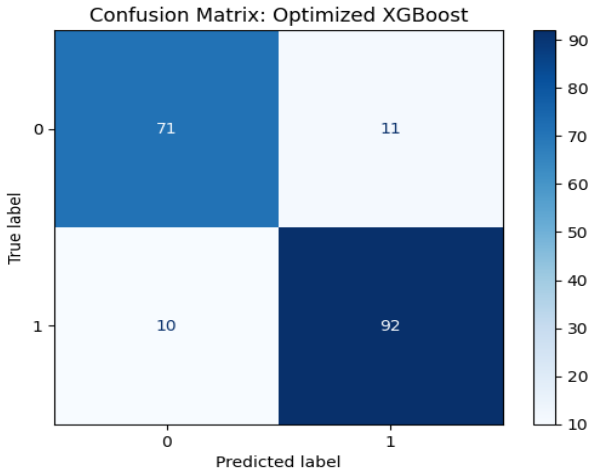

Fig. 4: Optimized XGBoost Model Confusion Matrix

3.1.3 Performance Results of Stacking Ensemble Model

The stacking ensemble, which combined optimized RF and XGB models with Logistic Regression as the meta-learner, delivered the highest overall performance. This result shows that stacking leverages complementary strengths of base learners to improve generalization. The ROC-AUC of 0.936 and PR-AUC of 0.935 were the highest recorded across all experiments, demonstrating superior predictive reliability as shown in Table 4. The confusion matrix shown in Fig. 5 highlighted improved classification balance across both classes, with fewer errors compared to that of the individual models.

Table 4: Stacking Ensemble Model Classification Report

Precision	Recall	F1-Score	Accuracy	ROC-AUC	PR-AUC	Balanced Accuracy
0.8952	0.8961	0.8956	0.8967	0.9360	0.9354	0.8961
0.8969	0.8967	0.8968	—	—	—	—

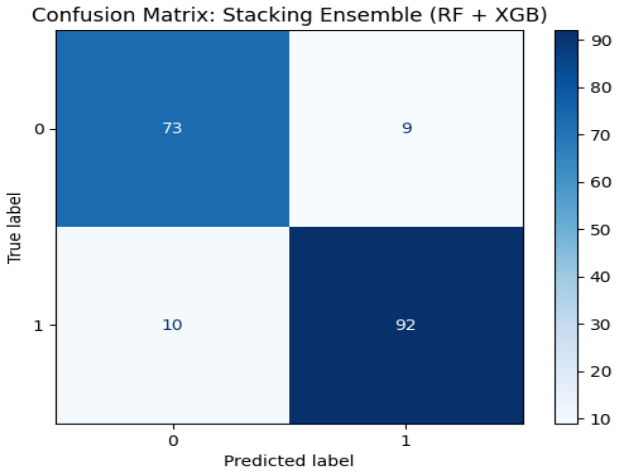

Fig. 5: Stacking Ensemble Model Confusion Matrix

3.2 Comparison with Existing Study

To assess the contribution of this study, the results were compared against the benchmark study by Mahmud et al. (2023), which employed a lightweight metamodel comprising RF, DT, and GNB as base learners with KNN as the meta-learner. Their model achieved an overall accuracy of approximately 87%, but their approach did not include hyperparameter optimization and extended evaluation metrics. The comparison in Table 5 highlights the improvements achieved by this study. The results demonstrate that the optimized stacking ensemble surpassed the benchmark study in terms of accuracy, F1-score, and extended metrics. Furthermore, by incorporating hyperparameter optimization and advanced evaluation, this study provides a more reliable and clinically relevant model for heart disease prediction.

3.3 Learning Curve Analysis

Learning curves were generated for each model to assess generalization behavior as the number of training samples increased.

Optimized Random Forest showed steady convergence between training and validation curves, indicating that the model achieved good balance between bias and variance. Illustrated in Fig. 6. The learning curve in Fig. 7 showed that the optimized XGBoost model exhibited slightly higher variance in early stages but converged smoothly as the training size increased, reflecting its robustness with increased data. Stacking Ensemble achieved the most stable learning curve as shown in Fig. 8, with training and validation scores converging quickly and maintaining good performance across varying training sizes.

Table 5: Comparison of Proposed Models with Mahmud et al. (2023)

Model	Accuracy	Macro F1	ROC-AUC	PR-AUC	Balanced Accuracy
Mahmud et al. (2023) Metamodel	0.87	0.87	Not reported	Not reported	Not reported
Optimized Random Forest (proposed)	0.891	0.887	0.934	0.930	0.889
Optimized XGBoost (proposed)	0.886	0.884	0.934	0.932	0.884
Stacking Ensemble (RF + XGB) (proposed)	0.897	0.896	0.936	0.935	0.896

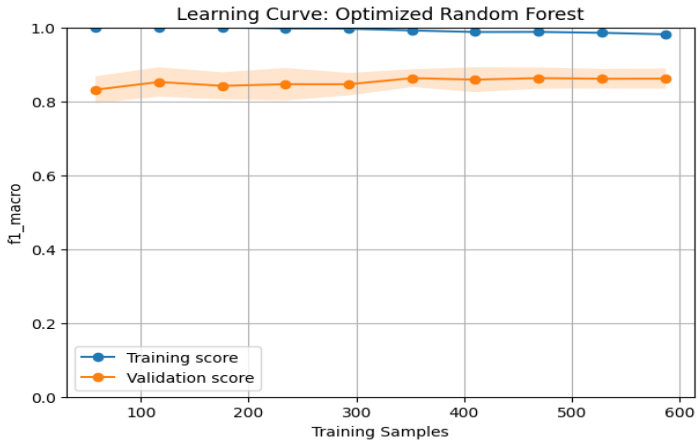

Fig. 6: Optimized Random Forest Model Learning Curve

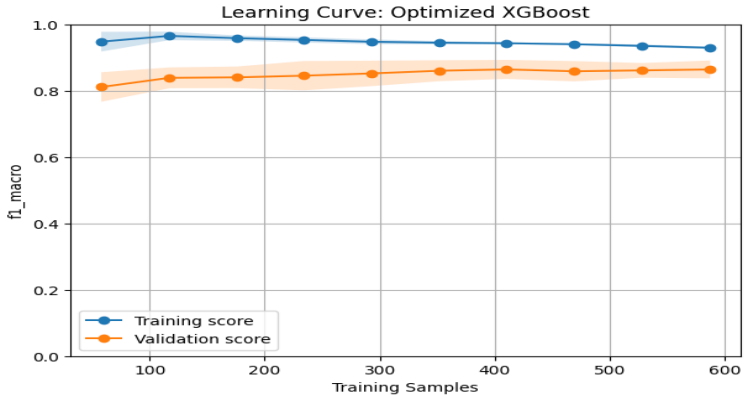

Fig. 7: Optimized XGBoost Model Learning Curve

These findings validate the hypothesis that stacking ensemble models can leverage the complementary strengths of the base models to achieve enhanced performance in heart disease prediction. The consistency of the learning curves suggests that all models benefited from the dataset size and were not overly dependent on additional samples to maintain accuracy.

4.0 Discussion of Findings

The findings of this study indicate that optimized ensemble methods can improve predictive performance in heart disease prediction. Among the models evaluated, the stacking ensemble of Random Forest and XGBoost achieved the highest performance across most metrics, with an accuracy of 89.7%, a macro F1-score of approximately 89.7%, and ROC-AUC and PR-AUC values of 93.6% and 93.5%, respectively. These results suggest that combining the complementary strengths of optimized Random Forest and XGBoost may enhance predictive capability compared to using individual models.

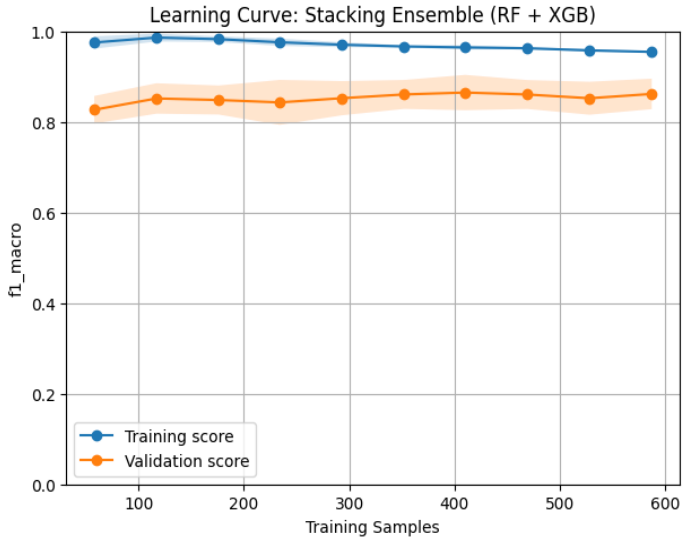

Fig. 8: Stacking Ensemble Model Learning Curve

In comparison with the benchmark study by Mahmud et al. (2023), which reported an accuracy of approximately 87% using a lightweight metamodel with default parameters, the present study demonstrates modest performance improvements. These gains may be attributed to several methodological enhancements. First, the use of hyperparameter optimization via Optuna likely contributed to improved model tuning, potentially reducing both underfitting and overfitting. Second, the stacking ensemble approach appears to have improved generalization by integrating the predictions of two well-performing classifiers. Additionally, the use of extended evaluation metrics provided a more comprehensive assessment of model performance beyond overall accuracy.

Analysis of the confusion matrices indicates that the optimized models, particularly the stacking ensemble, reduced the number of false negatives compared to the individual models. This is an encouraging result, as correctly identifying patients with heart disease is an important consideration in predictive modeling. At the same time, the models maintained relatively balanced specificity, suggesting that the reduction in false negatives did not come at the expense of a substantial increase in false positives. The learning curve analysis further suggests that the models exhibit stable learning behavior, with no clear evidence of severe overfitting within the scope of the dataset used. However, given the dataset size, these observations should be interpreted with caution, as model performance may vary with different data distributions or larger samples.

Overall, while the results demonstrate the potential of optimized ensemble methods for improving predictive performance, the findings are based on a single dataset and experimental setup. As such, they should be viewed as indicative rather than conclusive, and further validation on independent and real-world clinical datasets is necessary before drawing strong conclusions about practical applicability.

5.0 Conclusion

This study investigated the use of optimized ensemble learning techniques for heart disease prediction. The results indicate that combining Random Forest and XGBoost within a stacking framework, alongside systematic hyper-parameter tuning and extended evaluation metrics, can improve predictive performance compared to individual models and the benchmark study. These findings highlight the value of methodological rigor, particularly in model optimization and comprehensive evaluation, when applying machine learning techniques to healthcare data.

However, while the proposed approach demonstrated encouraging results, the findings

should be interpreted with caution. The study was conducted using a single publicly available dataset and did not involve real-world clinical validation. As such, the results are indicative of potential rather than definitive evidence of practical applicability. Further validation in diverse and clinical settings is required before such models can be reliably integrated into healthcare decision-making processes.

Limitations of the Study

Despite its contributions, the study is subject to several limitations:

i. Dataset Size: Although larger than some earlier studies, the dataset of 918 records remains relatively small compared to large-scale clinical datasets, which may limit the robustness of the models.
ii. Feature Scope: The dataset was restricted to clinical and demographic features, omitting other potentially relevant factors such as genetic information, imaging data, and lifestyle variables that could improve predictive performance.
iii. Generalisability: As the dataset is publicly available and not derived from a specific clinical environment, the model's performance may not fully generalize to real-world hospital settings with different patient populations and data distributions.
iv. Risk of Data Leakage: The hyperparameter optimization process may have inadvertently introduced data leakage by utilizing information from the test set during model tuning. This can lead to overly optimistic performance estimates and reduces the validity of the reported results. However, it is important to note that the test data were not used for model training, but only for evaluation across a limited number of optimization trials.
v. Absence of Cross-Validation Strategy: The study relied on a single train-test split without incorporating a cross-validation framework. This limits the reliability and stability of the findings, as model performance may vary depending on the specific data partition used.

5.3 Recommendations for Practice

i. Cautious Integration into Decision Support Systems: While the proposed models show potential, their use in clinical environments should be approached cautiously and only after thorough validation on real-world datasets.
ii. Comprehensive Model Evaluation: Future clinical AI systems should incorporate multiple evaluation metrics, including ROC-AUC and PR-AUC, to ensure balanced performance assessment.
iii. Focus on Error Trade-offs: Model deployment should carefully consider the trade-off between false negatives and false positives, particularly in high-risk medical contexts.

5.3.1 Recommendations for Future Work

To address the identified limitations and further improve the robustness of heart disease prediction models, the following recommendations are proposed:

i. Use of Validation Set for Hyperparameter Optimization: Future studies should adopt a three-way data split (training, validation, and testing) or nested validation approach, ensuring that hyperparameter tuning (e.g., with Optuna) is performed exclusively on the training/validation data to eliminate the risk of data leakage.
ii. Incorporation of Cross-Validation: Implementing k-fold cross-validation, particularly stratified cross-validation, would provide more reliable and stable performance estimates by reducing dependency on a single data split.
iii. Larger and More Diverse Datasets: Future research should validate the proposed models on larger, multi-institutional datasets to improve generalizability across different populations and clinical settings.
iv. Explainability and Interpretability: Integrating model interpretability techniques such as SHAP or LIME would enhance transparency and provide clinicians with actionable insights into model predictions.

v. Hybrid and Deep Learning Approaches: Further research could explore hybrid models that combine ensemble methods with deep learning techniques to capture more complex patterns in clinical data.
vi. Incorporation of Additional Features: Including richer feature sets, such as lifestyle, genetic, or imaging data, may improve predictive performance and provide a more comprehensive assessment of cardiovascular risk.

5.4 Contributions

This study makes several important contributions to the field of machine learning for healthcare, particularly in the domain of heart disease prediction:

i. Methodological Advancement: Unlike the benchmark study, which relied on default parameters and conventional classifiers, this research introduced systematic hyper-parameter optimization using Optuna, thereby ensuring that models were tuned for maximum predictive performance.
ii. Optimized Stacking Ensemble Framework: The study developed and validated a stacking ensemble that combines Random Forest and XGBoost as base learners with Logistic Regression as the meta-learner. This configuration demonstrated superior predictive accuracy and discriminative power compared to both individual models and the benchmark metamodel.
iii. Extended Evaluation Metrics: Beyond accuracy, this research incorporated macro and weighted averages of precision, recall, and F1-score, as well as ROC-AUC, PR-AUC, and balanced accuracy. These metrics provided a more comprehensive evaluation of model performance, especially in the presence of class imbalance, ensuring results are clinically meaningful.
iv. Reduction of False Negatives: By emphasizing recall and sensitivity in both optimization and evaluation, the models developed in this study reduced the risk of false negatives, a critical improvement in clinical contexts where failure to detect high-risk patients could have life-threatening consequences.
v. Foundation for Future Research: The findings provide a basis for further investigation into optimized ensemble methods, while highlighting key methodological considerations such as validation strategies and dataset limitations.

Declarations

Ethics approval and consent to participate

Not applicable

Consent for publication

Not applicable

Availability of data and materials

The dataset employed in this study consists of 918 patient records and 12 clinical attributes, sourced from Kaggle (*Heart Failure Prediction Dataset*, 2021).

Competing interests

The authors declare that they have no competing interests, regarding the research, authorship or publication of this manuscript.

Funding

No funding was received for this study.

Authors' contributions

U.S. (the student) conducted the data collection, preprocessing, model development, analysis, and interpretation of the results. A.A. served as the major supervisor, providing overall guidance, conceptual input, and critical revisions of the study. D.G. acted as the second supervisor, offering additional supervision, reviewing the methodology, critical revisions and making editorial corrections. All authors read and approved the final manuscript.

Acknowledgements

The authors wish to thank the Department of Informatics and Secured Computing, Kaduna State University, for providing computing resources needed for this research.

References

Argulian, E., & Narula, J. (2021). Advanced cardiovascular imaging in clinical heart failure. *JACC Heart Failure*, *9*(10), 699–709. https://doi.org/10.1016/j.jchf.2021-.06.016

Arunachalam, S. K., & Rekha, R. (2022). A novel approach for cardiovascular disease prediction using machine learning algorithms. *Concurrency and Computation Practice and Experience*, *34*(19). https://doi.org/10.1002/cpe.7027

Heart Failure Prediction Dataset. (2021, September 10). Kaggle. https://www.-kaggle.com/datasets/fedesoriano/heart-failure-prediction

Javaid, M., Haleem, A., Singh, R. P., Suman, R., & Rab, S. (2022). Significance of machine learning in healthcare: Features, pillars and applications. *International Journal of Intelligent Networks*, *3*, 58–73. https://doi.org/10.1016/j.ijin.2022-.05.002

Khan, A. A., Chaudhari, O., & Chandra, R. (2023). A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation. *Expert Systems with Applications*, *244*, 122778. https://doi.org/10.1016/j.eswa.2023.122778

Kokori, E., Patel, R., Olatunji, G., Ukoaka, B. M., Abraham, I. C., Ajekiigbe, V. O., Kwape, J. M., Babalola, A. E., Udam, N. G., & Aderinto, N. (2024). Machine learning in predicting heart failure survival: a review of current models and future prospects. *Heart Failure Reviews*. https://doi.org/10.1007/s10741-024-10474-y

Lam, C. S. P., Docherty, K. F., Ho, J. E., McMurray, J. J. V., Myhre, P. L., & Omland, T. (2023). Recent successes in heart failure treatment. *Nature Medicine*, *29*(10), 2424–2437. https://doi.org/10.-1038/s41591-023-02567-2

Liang, M., Singh, S., & Huang, J. (2024). Implementing Machine Learning to Predict Survival Outcomes in Patients with Resected Pulmonary Large Cell Neuroendocrine Carcinoma. *Expert Review of Anticancer Therapy*, *24*(10), 1041–1053. https://doi.org/10.1080/-14737140.2024.2401446

Mahmud, I., Kabir, M. M., Mridha, M. F., Alfarhood, S., Safran, M., & Che, D. (2023). Cardiac failure forecasting based on clinical data using a lightweight machine learning metamodel. *Diagnostics*, *13*(15), 2540. https://doi.-org/10.3390/diagnostics13152540

Saqib, M., Perswani, P., Muneem, A., Mumtaz, H., Neha, F., Ali, S., & Tabassum, S. (2024). Machine learning in heart failure diagnosis, prediction, and prognosis: review. *Annals of Medicine and Surgery*. https://doi.org/10.1097/ms9.0000000000002138

Shahim, B., Kapelios, C. J., Savarese, G., & Lund, L. H. (2023). Global Public Health Burden of heart Failure: An updated review. *Cardiac Failure Review*, *9*. https://doi.org/10.15420/cfr.2023.05

Appendix A: Code Implementation for Data Preprocessing

```
!pip install optuna
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import optuna

from sklearn.model_selection import
train_test_split, StratifiedKFold,
cross_val_predict
from sklearn.preprocessing import
StandardScaler, OneHotEncoder
from sklearn.compose import
ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import (confusion_matrix,
classification_report,
                    f1_score, precision_score,
recall_score,
                    accuracy_score, roc_curve,
precision_recall_curve,
                    roc_auc_score,
average_precision_score,
                    balanced_accuracy_score,
ConfusionMatrixDisplay)

from sklearn.ensemble import
RandomForestClassifier, StackingClassifier
from xgboost import XGBClassifier
from sklearn.linear_model import
LogisticRegression
from google.colab import drive
drive.mount('/content/drive')

# Load Data
df = pd.read_csv
('/content/drive/MyDrive/HeartFailurePredictio
n/heart.csv')
df.describe()
df.info()
df.head()

df.tail()
plt.figure(figsize=(10,6))
sns.heatmap(df.select_dtypes(include=np.numb
er).corr(),cmap='seismic', annot=True)

## Features and Target
X = df.drop("HeartDisease", axis=1)
y = df["HeartDisease"]

## Identify Categorical and Numerical
Columns

cat_features =
X.select_dtypes(include=["object"]).columns.to
list()
num_features =
X.select_dtypes(include=["int64",
"float64"]).columns.tolist()

## Preprocessor: Scale Mumerics, One-hot
Encode Categoricals
preprocessor = ColumnTransformer(
   transformers=[
      ("num", StandardScaler(), num_features),
      ("cat", OneHotEncoder(drop="first",
handle_unknown="ignore"), cat_features)
   ]
)

## Train-Test Split (80:20)
X_train, X_test, y_train, y_test =
train_test_split(
   X, y, test_size=0.2, stratify=y,
random_state=42
)
```

Appendix B: Code Implementation for Model Training and Optimization.

```
## Optuna Optimization: Random Forest

def rf_objective(trial):
   params = {
      "n_estimators":
trial.suggest_int("n_estimators", 100, 500),
      "max_depth":
trial.suggest_int("max_depth", 3, 20),
      "min_samples_split":
trial.suggest_int("min_samples_split", 2, 20),
      "min_samples_leaf":
trial.suggest_int("min_samples_leaf", 1, 10),
      "max_features":
trial.suggest_categorical("max_features",
["sqrt", "log2"]),
      "class_weight": "balanced",
      "random_state": 42,
```

```
        "n_jobs": -1,
    }
    model = Pipeline(steps=[("preprocessor",
preprocessor),
                            ("classifier",
RandomForestClassifier(**params))])
    model.fit(X_train, y_train)
    preds = model.predict(X_test)
    return f1_score(y_test, preds,
average="macro")

study_rf =
optuna.create_study(direction="maximize")
study_rf.optimize(rf_objective, n_trials=30)
best_rf = Pipeline(steps=[("preprocessor",
preprocessor),
                          ("classifier",
RandomForestClassifier(**study_rf.best_param
s,

class_weight="balanced",

random_state=42, n_jobs=-1))])
## Optuna Optimization: XGBoost

def xgb_objective(trial):
    params = {
        "n_estimators":
trial.suggest_int("n_estimators", 100, 500),
        "max_depth":
trial.suggest_int("max_depth", 3, 10),
        "learning_rate":
trial.suggest_float("learning_rate", 0.01, 0.3),
        "subsample":
trial.suggest_float("subsample", 0.5, 1.0),
        "colsample_bytree":
trial.suggest_float("colsample_bytree", 0.5,
1.0),
        "scale_pos_weight": (len(y_train) -
sum(y_train)) / sum(y_train),
        "random_state": 42,
        "eval_metric": "logloss",
        "n_jobs": -1
    }
    model = Pipeline(steps=[("preprocessor",
preprocessor),
                            ("classifier",
XGBClassifier(**params))])
    model.fit(X_train, y_train)
    preds = model.predict(X_test)
    return f1_score(y_test, preds,
average="macro")
study_xgb =
optuna.create_study(direction="maximize")
study_xgb.optimize(xgb_objective,
n_trials=30)
best_xgb = Pipeline(steps=[("preprocessor",
preprocessor),
                           ("classifier",
XGBClassifier(**study_xgb.best_params,

random_state=42, n_jobs=-1,

eval_metric="logloss"))])

## Stacking Ensemble (RF + XGB → Logistic
Regression)
stacking_model = StackingClassifier(
    estimators=[
        ("rf", best_rf),
        ("xgb", best_xgb)
    ],

final_estimator=LogisticRegression(class_weig
ht="balanced", max_iter=1000),
    n_jobs=-1
)
```

Appendix C: Code Implementation for Model Evaluation

```
## Evaluation Function

def evaluate_and_report(model, X_train,
X_test, y_train, y_test, name):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    y_prob = model.predict_proba(X_test)[:, 1]

    print(f"\n=== {name} ===")
    print(classification_report(y_test, y_pred,
digits=4))  # macro & weighted included
```

```
    print("ROC-AUC:", roc_auc_score(y_test,
y_prob))
    print("PR-AUC:",
average_precision_score(y_test, y_prob))
    print("Balanced Accuracy:",
balanced_accuracy_score(y_test, y_pred))

    cm = confusion_matrix(y_test, y_pred)
    disp =
ConfusionMatrixDisplay(confusion_matrix=cm
)
    disp.plot(cmap="Blues")
    plt.title(f"Confusion Matrix: {name}")
    plt.show()

## Run Evaluations:
evaluate_and_report(best_rf, X_train, X_test,
y_train, y_test, "Optimized Random Forest")
evaluate_and_report(best_xgb, X_train, X_test,
y_train, y_test, "Optimized XGBoost")
evaluate_and_report(stacking_model,  X_train,
X_test, y_train, y_test, "Stacking Ensemble (RF
+ XGB)")
```

Appendix D: Code Implementation for Model Learning Curves

```
## Learning Curve Function
from sklearn.model_selection import
learning_curve
def plot_learning_curve(model, X, y, name,
scoring="f1_macro"):
    """Plot learning curve for given model."""
    train_sizes, train_scores, test_scores =
learning_curve(
        model, X, y, cv=5, scoring=scoring,
n_jobs=-1,
        train_sizes=np.linspace(0.1, 1.0, 10),
shuffle=True, random_state=42
    )
    train_mean = np.mean(train_scores, axis=1)
    train_std = np.std(train_scores, axis=1)
    test_mean = np.mean(test_scores, axis=1)
    test_std = np.std(test_scores, axis=1)

    plt.figure(figsize=(7,5))
    plt.plot(train_sizes, train_mean, "o-",
label="Training score")
    plt.plot(train_sizes, test_mean, "o-",
label="Validation score")
    plt.fill_between(train_sizes, train_mean -
train_std, train_mean + train_std, alpha=0.2)
    plt.fill_between(train_sizes, test_mean -
test_std, test_mean + test_std, alpha=0.2)
    plt.title(f"Learning Curve: {name}")
    plt.xlabel("Training Samples")
    plt.ylabel(scoring)
    plt.ylim(0.0, 1.0)
    plt.yticks(np.linspace(0, 1, 6))
    plt.legend(loc="best")
    plt.grid()
    plt.show()

## Run Learning Curves:
plot_learning_curve(best_rf, X_train, y_train,
"Optimized Random Forest")
plot_learning_curve(best_xgb, X_train, y_train,
"Optimized XGBoost")
plot_learning_curve(stacking_model, X_train,
y_train, "Stacking Ensemble (RF + XGB)")
```