IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

A Novel Hybrid Pre-Processing Framework for Cardiovascular Disease Detection with Reduced False Negatives

[1]T. Jayasudha,
[1]Ph.D Research Scholar,
[1]PG & Research Department of Computer Science,
[1]Sri Sarada College for Women (Autonomous), Salem-16.

[2]Dr. R. Uma Rani,
[2]Principal (Retired),
[2]Sri Sarada College for Women (Autonomous), Salem-16.

Abstract

Cardiovascular disease (CVD) is one of the serious health issues in the world that causes the death of many people annually. Early CVD diagnosis is important in enhancing survival. The proposed research will contribute to creating a novel system to detect CVDs based on the synthesis of innovative hybrid techniques to impute missing data, detect outliers, balance the classes, select features, and classify them. It is also aimed at improving accuracy and reliability using a new combination of techniques, hyperparameter optimization and prevention of Type II errors.

This framework applies a real-time dataset that belongs to the Salem Private Hospital and a benchmark dataset that is found in the UCI repository to identify CVDs. The Heuristic-SHAP Adaptive MissForest (HSAMF) approach addressed missing data by adopting a hybrid technique of imputation that involves the combination of MissForest, SHAP and heuristic rules. Outlier handling was done through the Hybrid Global-Local-Structural Outlier Detection (HGLS-OD) method, which is a combination of Local Outlier Factor (LOF), Isolation Forest (IF) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm. Target encoding was implemented on categorical data. Class imbalance was addressed using optimized K-Means and Synthetic Minority Oversampling TEchnique (OKSMOTE), which includes K-Means, SMOTE, optuna and RF. The Min-MaxScaler was used to perform data normalization. The features were selected using the Cluster-Weighted Mutual Information - Genetic Algorithm (CWMI-GA) methodology that comprised the application of Mutual Information (MI), clustering and Genetic Algorithm (GA). Various classification methods had been used, such as traditional methods, such as Optimized Random Forest (ORF), Optimized XGBoost (OXGB) and ensemble techniques, such as Optimized Bagging-Boosting Stacked Ensemble (OBBSE), Optimized Heterogeneous Soft Voting Ensemble (OHSVE), Optimized Feature-Augmented Heterogeneous Stacking (OFAHS), Optimized Heterogeneous Bootstrap-Ensemble (OHBE) and Optimized Heterogeneous Sequential Boosting (OHSB). Optuna was used to maximize the classification and threshold tuning procedures.

The OFAHS model was superior to all the other models with the default threshold of 99.1% when applied to real-time data and 97% when applied to benchmark data. The ideal threshold of 0.47 greatly minimized Type II errors. With this threshold, accuracy increased further to 99.4 and 98.5 on real-time and benchmark data, respectively. Thus, not only were the Type II errors lowered by modifying the threshold, but also the reliability of the forecasts was enhanced. This study demonstrates significant improvement in cardiovascular disease detection through advanced methods that led to improved data handling, accuracy, and performance. The findings are the precursors of the prospective advancements in cardiovascular diagnostics and health management.

Index Terms: Cardiovascular Disease, Pre-Processing, Ensemble Classification, Tuning, Type 2 Error Reduction

1. Introduction

More people die each year due to CVD as compared to all other diseases (1). Cardiovascular disease is a collection of conditions that influence the heart and blood vessels, and it consists of hypertension, heart failure, stroke and coronary artery disease. Due to its risk factors, which include smoking, poor diet, high blood pressure, high cholesterol, inactivity, and diabetes, it has continued to be one of the top causes of death across the world. Early CVD diagnosis is significant to reduce the rate of mortality and improve treatment outcomes. Machine learning and statistical models have become very important in the medical field, particularly in the prediction of heart diseases. Exploring historical training information allows the machine learning of the disease diagnosis (2). Through analysis of patient data, the models are useful in identifying trends and predicting the likelihood of the occurrence of CVD. The attention of many medical researchers is devoted to the development of new machine learning-based predictive models in terms of disease prediction (3, 4, 5). Hospitals tend to employ classification techniques to enhance the precision of disease prediction and diagnosis (6, 7, 8). Random Forest, Decision Tree, Logistic Regression, Naive Bayes, and Support Vector Machine are some of the methods that are used to detect the disease (9, 10).

Pre-processing in medical data analytics is required as it purifies and organises unstructured or variant health data into a standard and trustworthy format. Part of this process includes cleaning up the errors and the null values, dealing with outliers, class balancing, choosing the key features and so on. Dealing with missing data in medical data is essential to eliminate false results and reduce bias. Effective management of null values makes the dataset complete and reliable and enhances healthcare decision-making. The outliers may skew the results and draw the wrong conclusions; that is why it is necessary to cope with them properly. A proper outlier control increases the accuracy of data, and it promotes reliable medical findings. Class balancing ensures equal learning in all the classes so that the majority class does not benefit from the models. This leads to increased general accuracy of prediction and increases detection. The feature selection is important as it eliminates redundant features that make the models faster and more accurate. It enhances prediction and decision-making in medical analysis as it pays attention to the most valuable information. The feature selection algorithms could be classified into supervised, unsupervised and semi-supervised.

1.1 Types of CVD:

- Coronary Artery Disease (CAD): Blockage of the coronary arteries as a result of the deposits of plaque.
- Hypertension: High blood pressure over an extended period of time.
- Heart Attack: A Casualty of the heart caused by a blockage in the blood flow.
- Heart Failure: This is the inability of the heart to pump blood sufficiently.
- Arrhythmias: The heartbeat of the body is affected.
- Stroke: This is a condition that leads to the loss of cells due to interference with the blood flow to the brain.
- Peripheral Artery Disease (PAD): The lack of blood supply to the limbs because the arteries are constricted.
- Cardiomyopathy: A disease that prevents the heart muscle from functioning properly.
- Congenital Heart Disease: “Congenital disabilities of the heart.

- Valvular Heart Disease: The heart damage affects the flow of blood through the heart valves.
- Deep vein thrombosis (DVT): A blood clot within one of the veins, usually in the legs, is termed deep vein thrombosis.
- Endocarditis: It is an infection of the inner surface of the heart.

1.2 Motivation of Cardiovascular Disease Detection:

- High Mortality and Prevalence: Cardiovascular diseases are the number one cause of death in the world, and therefore, early and accurate diagnosis of the disease is essential to curb mortality.
- Personalized Treatment: Accurate classification will enable the creation of treatment plans that are specific to a patient and will enhance patient outcomes and reduce unnecessary operations.
- Improved Knowledge of CVDs: Detection of CVDs is an improvement in medical research and treatment through the identification of illness patterns.
- Key Resource Allocation: Categorizing the high-risk events assists healthcare professionals in prioritizing the high-risk events so that they can allocate resources in the best way possible.
- Reducing Healthcare Costs: This is achieved through early and accurate detection that reduces the long-term costs.

2. Literature Study:

The most current research in heart disease prediction is summarised in Table 1.

Table 1: Related Studies

S.No	Author	Paper Title	Dataset	Pre-processing	Classification	Tuning	Best Accuracy
1	Jingyi Zhang, Huolan Zhu, Yongkai Chen, et al., 2021 [11]	Ensemble machine learning approach for screening of coronary heart disease based on echocardiography and risk factors	Clinical trial dataset	DR: PCA	Stacking Using Many Classifiers	-	87.7
2	Ya-Han Hu et al., 2024 [12]	A novel missforest-based missing values imputation approach with recursive feature elimination in medical applications	Multiple Dataset	Imputation: RFE-MissForest (MF), Mean/Mode, kNN, MICE, MissForest	NRMSE, PFC	-	1stRank RFE-MF
3	Aljee AK, Mukherjee A, et al. 2013 [13]	Comparison of imputation methods for missing laboratory data in medicine	Inflammatory Bowel Disease, Cirrhosis Cohort	Imputation: MissForest, Mean, NN, MICE	LR, RF	-	MissForest produced the lowest error.
4	K. Senthamarai Kannan et	A comparative study of outlier detection methods in	Heart Disease dataset HND0 to HND4	Outlier Handling: k-NN, LOF	NB, SVM	-	83.1% (LOF, NB, HND2)

	al.,2024 [14]	heart disease data					
5	Bilal Ahmad et al., 2025 [15]	Feature selection strategies for optimized heart disease diagnosis using ML and DL models	Cleveland Heart Disease Dataset	Feature Selection: MI, ANOVA F-test, Chi-Square test	NN, LR, RF, GBoost, AdaBoost, DT, LDA, SVM, Nu-SVC, KNN, NB	-	82.3 (MI with NN)
6	Moiz Ur Rehman, Shahid Naseem et al., 2025 [16]	Predicting coronary heart disease with advanced machine learning classifiers for improved cardiovascular risk assessment	UCI CHD Dataset	Imputation: (mean/mode), encoding: One-hot, Balancing: SMOTE, FS: MI, DR: PCA	KNN, NB, SVC, RF, LR, PSO-ANN	Grid search	96.1 PSO-ANN
7	Ghalia A. Alshehri et al., 2023 [17]	Prediction of Heart Disease using an Ensemble Learning Approach	StatLog, Z-Alizadeh Sani, CVD	Normalization: MinMaxScaler, Balancing: SMOTE, FS: (Forward/Backward Wrapper)	AdaBoost, SVM (Linear Kernel), DT, RF, Ensemble (ELA)-Adaptive boosting, SVM, DT, and RF	Manual Tuning	Z-Alizadeh Sani: 91% (ELA) StatLog: 83% (ELA) CVD: 73% (ELA)
8	Chandralekha E, S. Vinodhini, et al., 2025 [18]	Heart Rate Anomaly Detection in Healthcare Using Elliptic Envelope and Local Forest	Three Synthetic Heart Rate Data datasets	Normalization: Min-Max Scaling, Outlier: IF, LOF, OCSVM, and EE	-	-	93 (IF)
9	Vaishali M Deshmukh 2019 [19]	Heart Disease Prediction using Ensemble Methods	Cleveland	Normalization: Min-Max Scaling, Standardization: Standard Scaler, FS: ET	Majority Voting with Bagging (DT, LR, ANN, KNN, NB)	-	87.78
10	Jeevan Babu Maddala et al., 2024 [20]	Heart Failure Prediction Using Machine Learning	Z-Alizadeh Sani, SPECTF, Kaggle CVD dataset	Feature Extraction: RF, Balancing: SMOTE	RF, GB, ADB, ET, XGB, Hybrid Model (ET, RF, XGBoost)	Grid Search CV	89.82 (Hybrid Model)

2.1 Research Gap:

It is evident in the interrelated research papers that many critical research gaps exist. These are:

- ✓ Real-Time Data: The classification of cardiovascular diseases based on real-time data is also an important gap in the research.
- ✓ Reduction of Type II errors: Changing threshold values should be more evident to reduce Type II errors.
- ✓ Imputation: The importance of features and heuristic rules coupled with imputation algorithms has not been studied comprehensively.
- ✓ Outlier Detection: To date, no study has been carried out on hybrid models of detecting local, global, and structural outliers of the aortic heart data.
- ✓ Feature Selection: Selecting variables, more complex methods like MI and GA, are not used effectively during the pre-processing of hybrid pipelines.
- ✓ Optuna Hyperparameter Tuning: There is a gap in the adoption of Optuna for pre-processing and classification processing hybrid models optimization.
- ✓ Boosting and Bagging: It is possible to tap into a relatively untapped opportunity in fusing boosting and bagging in detecting CVD.
- ✓ Stacking models: It has not been thoroughly explored in the literature on CVD detection research whether combinations of various models of learning aid optimality of stacking using advanced methods of hyperparameter optimization in feature-enhanced stacking.
- ✓ Bagging Ensemble: Bagging, Optuna tuning, and Soft Voting have been previously applied individually in CVD detection; however, there is no study offering a combination of these models.
- ✓ Sequential boosting: It involves sequential boosting techniques based on numerous algorithms, which need further studies to enhance CVD detection rates.

To overcome these dilemmas, we have come up with a formidable and new framework for detecting cardiovascular disease.

3. Novel CVD Detection Framework:

We provide a machine learning-based system that integrates numerous techniques to enhance cardiovascular disease risk identification. Data cleaning, imputation procedure, outlier handling, encoding, class balancing, scaling, feature inference, classification, and type 2 error reduction procedure are steps implemented in this structure. The framework of CVD detection is shown in Figure 1.

Step 1: Data Collection: In the detection of CVD, real-time and benchmark data were obtained.

Step 2: Pre-processing:

- ✓ Imputation: To impute missing values, MissForest, SHAP, and a heuristic rule were utilized.
- ✓ Outlier Management IF was applied to detect the global outliers; LOF was used for local outliers, and DBSCAN was applied to detect the cluster-based outliers.
- ✓ Class Balancing: The SMOTE and K-means clustering algorithms were used to solve the class imbalance. In order to maximize the K-means hashing and SMOTE parameters, Optuna tested the performance of the classifier on the balanced dataset.

Step 3: Feature Selection: MI, Clustering and GA hybrid algorithm were employed in the selection of features that are globally and intrinsically significant to the target variable.

Step 4: Classification: Various ensemble techniques are utilized in classification, including feature augmented stacking, bootstrap ensemble, sequential boosting and soft voting. These methods have been trained and optimized with the help of the Optuna hyperparameter tuner to give the perfect classification accuracy. The current paper is the continuation of our previous classifier project (21).

Step 5: Optimisation and evaluation of the Threshold: To reduce Type II mistakes, Optuna was used to optimise the classification threshold. The model was then comprehensively checked with the help of wide evaluation measures.

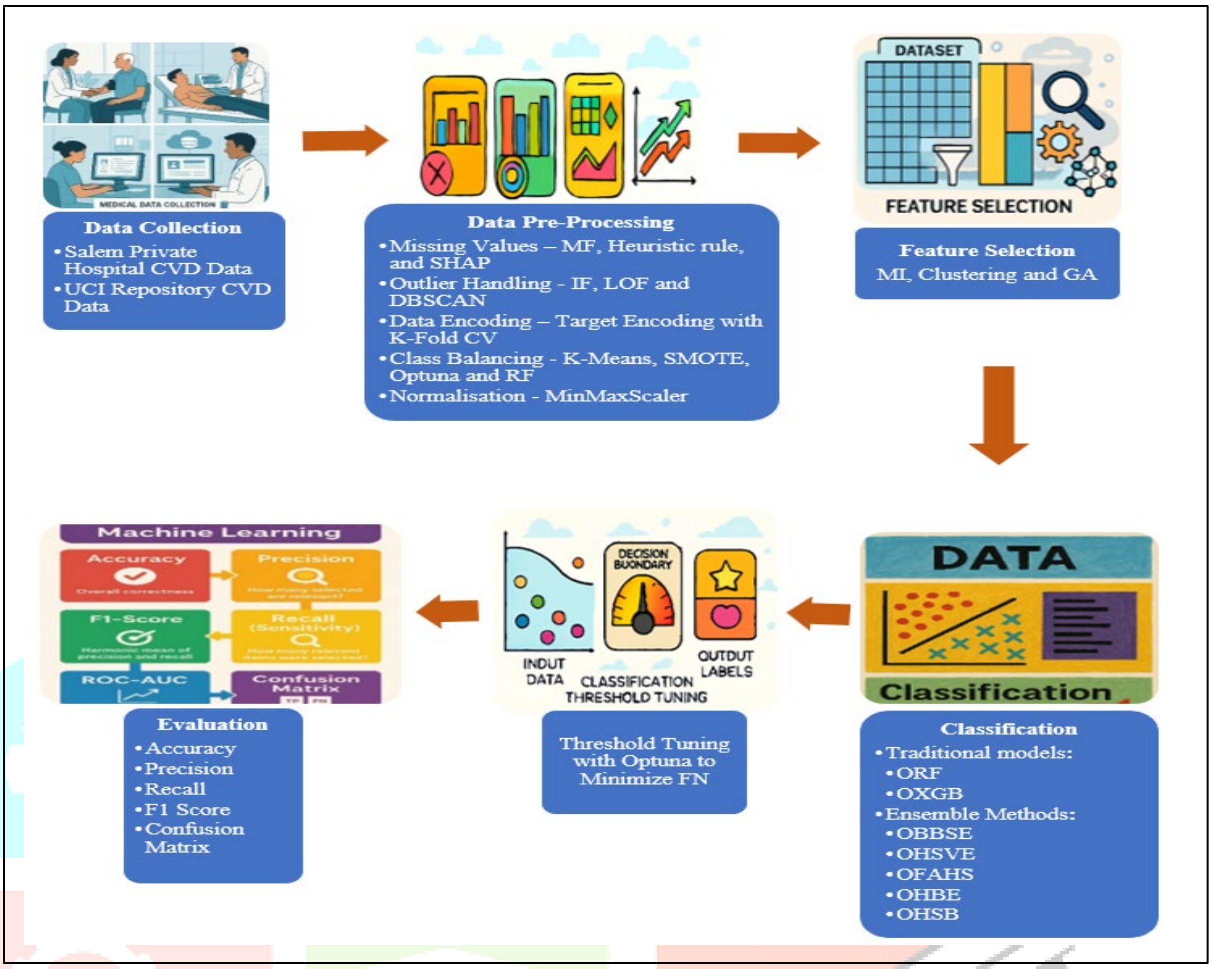
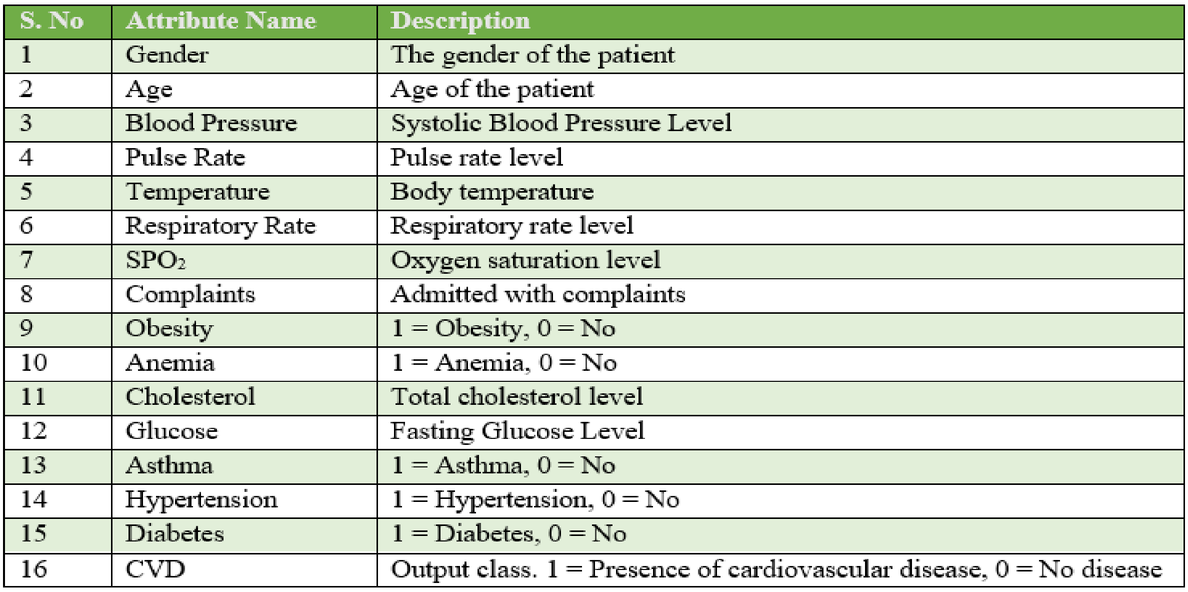


Figure 1: Novel Framework for CVD Detection

4. Data Collection

4.1 Data Description of Real-Time Dataset:

Figure 2 contains the trait information of the real-time data of a private hospital in Salem. This dataset contains 16 features: 15 input features, 1 output feature and 2300 records.

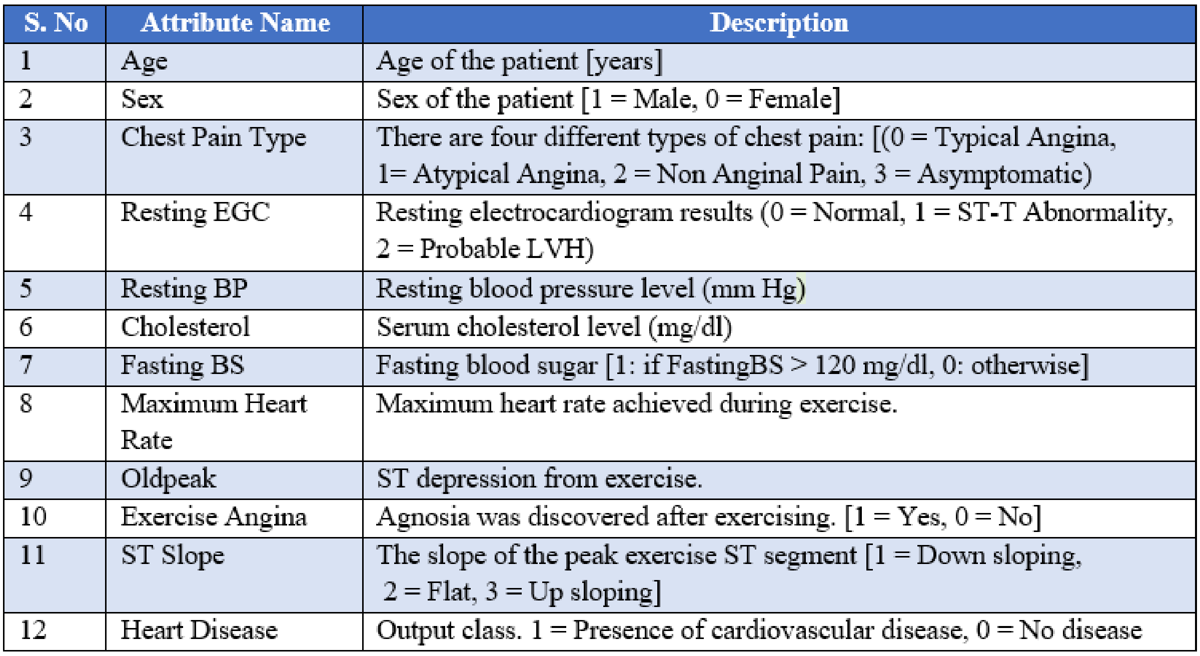


S. No	Attribute Name	Description
1	Gender	The gender of the patient
2	Age	Age of the patient
3	Blood Pressure	Systolic Blood Pressure Level
4	Pulse Rate	Pulse rate level
5	Temperature	Body temperature
6	Respiratory Rate	Respiratory rate level
7	SPO_2	Oxygen saturation level
8	Complaints	Admitted with complaints
9	Obesity	1 = Obesity, 0 = No
10	Anemia	1 = Anemia, 0 = No
11	Cholesterol	Total cholesterol level
12	Glucose	Fasting Glucose Level
13	Asthma	1 = Asthma, 0 = No
14	Hypertension	1 = Hypertension, 0 = No
15	Diabetes	1 = Diabetes, 0 = No
16	CVD	Output class. 1 = Presence of cardiovascular disease, 0 = No disease

Figure 2: Attributes of Real-time Dataset

4.2 Data Description of Benchmark Dataset:

Figure 3 demonstrates the attribute information of the benchmark dataset that is taken from the UCI repository. This dataset consists of the Cleveland, Hungarian, Switzerland, Long Beach VA, and Statlog heart disease datasets. It has one output feature, eleven input features, and 1190 records.



S. No	Attribute Name	Description
1	Age	Age of the patient [years]
2	Sex	Sex of the patient [1 = Male, 0 = Female]
3	Chest Pain Type	There are four different types of chest pain: [(0 = Typical Angina, 1= Atypical Angina, 2 = Non Anginal Pain, 3 = Asymptomatic)
4	Resting EGC	Resting electrocardiogram results (0 = Normal, 1 = ST-T Abnormality, 2 = Probable LVH)
5	Resting BP	Resting blood pressure level (mm Hg)
6	Cholesterol	Serum cholesterol level (mg/dl)
7	Fasting BS	Fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]
8	Maximum Heart Rate	Maximum heart rate achieved during exercise.
9	Oldpeak	ST depression from exercise.
10	Exercise Angina	Agnosia was discovered after exercising. [1 = Yes, 0 = No]
11	ST Slope	The slope of the peak exercise ST segment [1 = Down sloping, 2 = Flat, 3 = Up sloping]
12	Heart Disease	Output class. 1 = Presence of cardiovascular disease, 0 = No disease

Figure 3: Attribute Description

5. Pre-processing:

5.1 Heuristic-SHAP Adaptive MissForest (HSAMF) Imputation:

The HSAMF method aims at attaining strong imputation of absent values through a combination of feature-importance and data-driven parameter optimization. In this method, the MissForest algorithm is used to infer the missing data, and SHAP (SHapley Additive explanations) is used to determine which features are salient and dictate the imputation procedure. MissForest fills in missing values in a dataset using random forests, depending on patterns in other features to predict the missing values. The process is repeated until the maximum accuracy in the prediction of all the missing values is achieved. SHAP uses Shapley values of cooperative game theory to assign an importance score to each variable for a particular prediction. Figure 4 shows the HSAMF workflow.

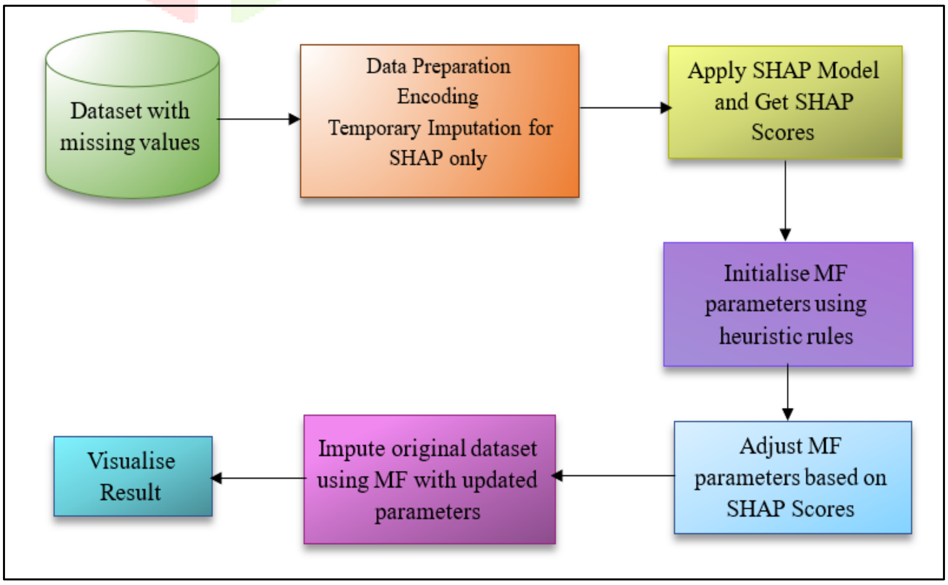

Dataset with missing values
Data Preparation Encoding Temporary Imputation for SHAP only
Apply SHAP Model and Get SHAP Scores
Initialise MF parameters using heuristic rules
Adjust MF parameters based on SHAP Scores
Impute original dataset using MF with updated parameters
Visualise Result

Figure 4: Workflow of HSAMF

HSAMF Pseudocode:

```
Input: A Dataset with missing values.
Output: Imputed dataset.
Step 1: Setup and Data Preparation
    • Import libraries and load the dataset.
    • Check missing values and cardinality in the dataset.
    • Apply cross-validated target encoding on categorical features.
    • Temporarily impute with median for SHAP analysis only.
    • Train SHAP model on temporarily imputed data and get SHAP scores.
Step 2: MissForest Parameter Initialisation and SHAP-based Adjustment
    • Calculate base parameters of MF using the heuristic rules.
    • Adjust MF's base parameters using SHAP importance.
    • Store tuned parameters.
Step 3: Iterative MissForest Imputation
    • Train MF with updated parameters on the encoded dataset with missing
      values.
    • Impute missing values in the dataset.
    • Save the final imputed dataset.
Step 4: Results and Visualization
    • Plot the results.
```

Steps of HSAMF:

Step 1: Setup and Data Preparation:

After loading the dataset, missing values along with cardinality are identified.

Determine the missingness for each feature (f):

$$M_f = \frac{\text{Count of missing values in f}}{\text{Total Samples}}$$

Cross-validated target encoding is utilised to encode categorical features.

To calculate SHAP values, which indicate the most influential factors for prediction, the dataset is temporarily imputed using the median.

Step 2: MissForest Parameter Initialisation and SHAP-based Adjustment:

Heuristically, MissForest initialises the number of trees n_{trees} in the random forest models in accordance with the size of the dataset N:

$$n_{trees} = \min(100, \max(10, 10 * \log_2(N)))$$

The importance score for feature j is:

$$I_j = \frac{1}{N}\sum_{i=1}^{N}|\phi_{i,j}|$$

Where $\phi_{i,j}$ denotes SHAP values.

The number of trees parameter is modified to better capture intricate patterns based on the relative importance of a subset S of top features:

$$n_{trees}^{adj} = n_{trees} \times \left(1 + \lambda \times \frac{\sum_{j \in S} I_j}{\Sigma_j I_j}\right)$$

Where the tuning hyperparameter that controls the amplitude of the adjustment is denoted by λ.

MissForest hyperparameters are initially set heuristically and then fine-tuned via SHAP importance scores. Like the tree structure, SHAP values were used to optimise MissForest's other hyperparameters to direct the tuning procedure and increase imputation accuracy. Here, both the heuristic rules and this SHAP result are used to adjust the MissForest imputation settings to emphasize more important features. Then, the MissForest with adjusted settings is used to impute missing values.

Step 3: Iterative MissForest Imputation:

MissForest uses iterative imputation with the modified parameters. Every feature *f* with missing entries is predicted by a random forest RF_f trained on other features at iteration t + 1, producing updated imputations:

$$x_i^{(f),t+1} = RF_f\left(x_i^{(-f),t}\right)$$

Until convergence, the process is repeated under observation by:

$$Error^{(t)} = \frac{1}{|M|} \sum_{(i,f)\in M} \left| x_i^{(f),t} - \boldsymbol{x}_{\boldsymbol{i}}^{(\boldsymbol{f}),\boldsymbol{t-1}} \right|$$

Where the set of missing elements is represented by M. Iterations come to an end at $\text{Error}^{(t)} < \epsilon$.

Step 4: Result:

Finally, plots of the data before and after imputation are displayed. These plots validate the efficacy of the suggested approach by evaluating the decrease in missingness.

The use of the HSAMF method is innovative since it provides direction to the imputation of the MF by means of synthesizing SHAP-based feature importance with heuristic knowledge. The method adapts the indispensability of each feature to the model by adjusting the parameters of MissForest imputation, and this provides a smart and cost-efficient way to work with the missing data compared to the traditional ways of using constant parameters.

5.2 Hybrid Global-Local-Structural Outlier Detection (HGLS-OD):

This HGLS-OD method is designed to identify and address local, global and structural outliers in an imputed dataset in an efficient way. This approach provides a complete outlier detection strategy by integrating the strengths of LOF, which is good at detecting local abnormalities, IF, which is good at detecting global outliers, and DBSCAN, which is good at detecting structural or cluster-based outliers.

IF is an outlier detection method with no supervision, which isolates outliers with a tree-like structure of random partitions. Reductions in the number of splits to separate outliers are fewer than in the rest of the data, which decreases tree path length. The amount of anomaly is increased to a greater level when data points have a lower and lower path length over multiple of these isolation trees. The LOF is an unsupervised anomaly detection method that is useful for estimating the local deviation of the density of a data point compared to its neighbors to determine the presence of outliers. When the local density of a data point is considerably smaller than that of its neighbours, then it is considered an outlier. Due to this reason, LOF can be applied in the detection of outliers in datasets of different densities. One popular density-based clustering approach for outlier detection is DBSCAN. It classifies closely spaced points according to a density criterion and marks points in low-density areas as noise or outliers. The workflow of the HGLS-OD method is presented in Figure 5.

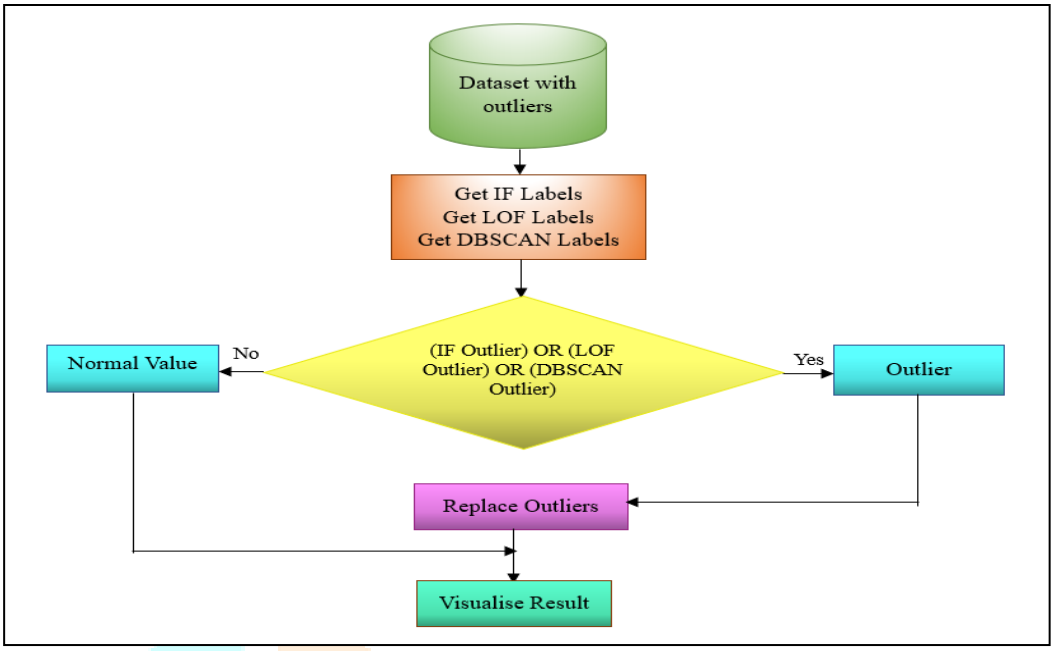


Figure 5: Workflow of HGLS-OD

HGLS-OD Pseudocode:

```
Input: Imputed dataset with outliers
Output: Outlier-handled dataset
Step 1: Setup and Data Preparation
    • Import libraries and load the dataset.
Step 2: Detect Outliers
    • Run Isolation Forest → Get IF labels.
    • Run Local Outlier Factor → Get LOF labels.
    • Run DBSCAN → Get DBSCAN labels.
    • Outlier flag = (IF Outlier) OR (LOF Outlier) OR (DBSCAN Outlier).
Step 3: Handle Outliers
    • For each feature, replace outlier values with the median of non-outliers.
    • Save the cleaned dataset.
Step 4: Result
    • Plot data before and after outlier handling.
```

Steps of HGLS-OD:

Step 1: Setup and Data Preparation:

In this workflow, the imputed dataset is used to detect and address the outliers using three procedures. The initial phase is the setup and data preparation.

Step 2: Detect Outliers:

First, globally isolated anomalies are detected with the help of the IF methodology. Then the anomaly in the local density changes is identified using the LOF, and the atypical points or structurally disconnected points are then determined using the DBSCAN.

IF Score:

$$S(x,n) = 2^{-\frac{E[h(x)]}{c(n)}}$$

Where:

- ✓ For data point x, the average path length across all trees is E[h(x)].
- ✓ c(n) is a normalisation factor.

LOF Score:

$$LOF_k(x) = \frac{1}{|N_k(x)|} \sum_{y \in N_k(x)} \frac{l_r\, d_k(y)}{l_r d_k(x)}$$

Where:

- ✓ The nearest neighbours of *x* are referred to as $N_k(x)$.
- ✓ $l_r d_k(x)$ is the local reachability density of the data point *x*.
- ✓ The local reachability density of a nearby data point y is defined as $l_r\, d_k(y)$. within the k-nearest neighbours of *y*.

DBSCAN Flag:

The simple formula for DBSCAN outlier detection is based on neighborhood density:

$$\text{Outlier } (p)=\begin{cases} 1 \text{ if } |N\varepsilon(p)|<\text{min\_samples and } p \text{ is not reachable from any core point,} \\ 0 \text{ otherwise} \end{cases}$$

- ✓ For each data point *p*:

  Count the number of points within radius ε (eps) from *p*, called the neighborhood $N_\varepsilon(p)$.
- ✓ Define core points, border points, and outliers:
  - Core point: If |*Nε(p)*|≥min_samples, then *p* is a core point.
  - Border points: If |*Nε(p)*|<min_samples but reachable from core point.
  - Outliers: If |*Nε(p)*|<min_samples and *p* is not reachable from any core point, then *p* is labeled as an outlier (noise).

A data point is classified as an outlier if any one model is identified as outlier.

Step 3: Handle Outliers:

On all the flagged records, we substitute each feature value with the median of the non-outlier values. This approach helps maintain data consistency while preserving the dataset's overall distribution.

Step 4: Result:

Finally, visualizations compare data before and after outlier handling to confirm effective outlier handling.

This HGLS-OD approach is regarded as novel since it combines several outlier detection models (IF, LOF and DBSCAN) in a complementary manner, guaranteeing more resilience by flagging an instance as an outlier if any one model discovers it. Furthermore, compared to utilising a single detection method or straightforward removal techniques, substituting the column median for found outliers offers innovation while maintaining the dataset's statistical distribution and minimising information loss.

5.3 Optimized K-Means and SMOTE (OKSMOTE) Class Balancing:

The aim of the objective is to equalise the dataset and enhance the classification performance with Optuna by streamlining the K-Means clustering algorithm and SMOTE. In this work, a fully automated, end-to-end optimisation pipeline that concurrently optimises K-Means clustering and SMOTE oversampling parameters with the help of Optuna is described. K-Means approach is an unsupervised learning algorithm that is often applied in clustering data points that do not have labels associated with them, and is based on similarity to group data points into K clusters. An uneven dataset is one where the underlying classes of output have unequal distributions. The SMOTE approach is considered to be among the most reliable ways of managing unequal data.

SMOTE solves the problem of class imbalance in a dataset by synthesizing samples of the minority class rather than merely duplicating data of the minority class. Optuna is a free open-source framework of hyperparameter tuning that was created to accomplish the process of determining the ideal hyperparameters for machine learning and deep learning frameworks and models. The workflow of the OKSMOTE approach is presented in Figure 6.

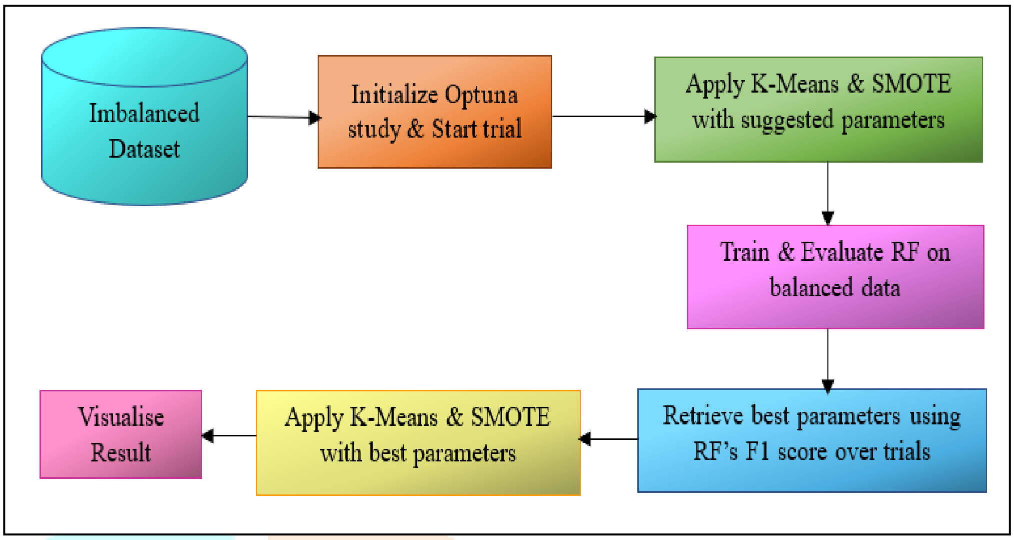

Imbalanced Dataset
Initialize Optuna study & Start trial
Apply K-Means & SMOTE with suggested parameters
Train & Evaluate RF on balanced data
Retrieve best parameters using RF's F1 score over trials
Apply K-Means & SMOTE with best parameters
Visualise Result

Figure 6: Workflow of OKSMOTE

OKSMOTE Pseudocode:

```
Input: Imbalanced Dataset.
Output: Balanced dataset.
Step 1: Setup and Data Preparation
        •  Import libraries and load the dataset.
Step 2: Optimise K-Means and SMOTE Parameters
        •  Use Optuna to find the best K-Means and SMOTE
           parameters.
        •  For each trial:
           ✓  Apply K-Means clustering on the dataset.
           ✓  Apply SMOTE to balance the classes based on K-Means
              clusters.
           ✓  Train the RF model with the balanced dataset.
           ✓  Evaluate the model's performance.
        •  Track and store the best K-Means and SMOTE parameters
           based on RF F1 score.
Step 3: Final Model
        •  Apply the best K-Means and SMOTE parameters to balance
           the entire data.
Step 4: Result
        •  Print and plot the final results.
```

Steps of OKSMOTE:

This hybrid method works by first dividing the minority group into small clusters with the K-Means algorithm.

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$$

Where k is the number of clusters, x is a data point, C_i is cluster i, μ_i is the centroid of the cluster i. After that, SMOTE generates new samples within every cluster.

$$x_{new} = x + \lambda(x_{nn} - x), \lambda \sim U(0,1)$$

Where:

- x→ Sample of the initial minority class.
- x_{nn}→ From the minority class, one of x's k closest neighbours.
- λ→ Picking a random number between 0 and 1 from a uniform distribution.
- x_{new} artificial data point produced between x and x_{nn}.

According to the F1 score of RF, Optuna automatically decides the best settings of this process. When the optimum setup is identified, the whole data set is balanced, and the results are presented.

The specialty of this OKSMOTE method is that all these procedures, such as automatic tuning, clustering, oversampling, and validation, are combined into one automated process, which is rare for predicting cardiovascular disease. Although K-Means, SMOTE and hyperparameter optimisation are better known individually, our hybrid approach, which modifies both clustering and balancing hyperparameters within the same optimisation process under the guidance of feedback from the classifier, is better. This combined optimisation technique enhances the performance of classification using unbalanced data sets, as it helps search through data to find the most suitable balancing technique.

6. Cluster-Weighted Mutual Information - Genetic Algorithm (CWMI-GA) Feature Selection:

In this research, Mutual information, K-Means clustering, and the genetic algorithm are used to determine the most important features to detect CVD. Figure 7 shows the workflow of the CWMI-GA technique.

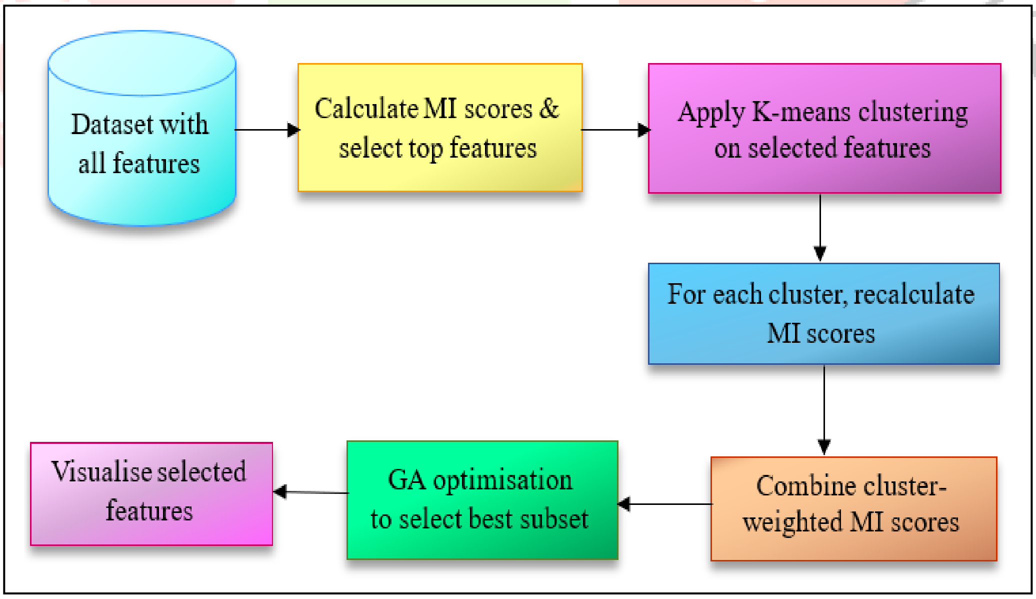

Dataset with all features
Calculate MI scores & select top features
Apply K-means clustering on selected features
For each cluster, recalculate MI scores
Combine cluster-weighted MI scores
GA optimisation to select best subset
Visualise selected features

Figure 7: Workflow of CWMI-GA

MI finds the most informative features on the label of the classes. The clustering of the data is done by the K-means according to similarities. Genetic Algorithm is a genetically inspired search and optimization algorithm inspired by natural evolution. The process of choosing the best features that can enhance the performance of a model is done using GA.

CWMI-GA Pseudocode:

```
Input: A Dataset with all features.
Output: Selected Features.
Step 1:  Setup and Data Preparation
    • Import libraries and load the data.
Step 2:  Global MI Scoring
    • Calculate MI Scores for all features.
    • Select top 'n' highest-scoring features.
Step 3:  Cluster Generation
    • Takes the top 'n' features from Step 2.
    • Applies K-means clustering to group similar samples.
Step 4:  Cluster-Specific Scoring
    • For each Cluster:
        ✓ Recalculate MI Scores.
        ✓ Store Cluster-adjusted importance.
Step 5:  Cluster-Weighted MI Scores
    • Combine MI scores from all clusters into a single importance score.
Step 6:  GA Optimization
    • Initialize random feature sets using the features obtained from step 5.
    • Evaluate using:
        ✓ Prediction accuracy.
        ✓ Avg weighted Score.
    • Evolve via crossover/mutation.
    • Return the best subset across all generations.
Step 7: Result
    • Print and plot the final results.
```

Working Steps of CWMI-GA:

Step 1: Setup and Data Preparation:

The importation of the necessary libraries and loading of the data are carried out first.

Step 2: Global Mutual Information (MI) Scoring:

Calculating the Mutual Information of each feature X_i with the target Y:

$$MI(X_i, Y) = \sum_{x_i \in X_i} \sum_{y \in Y} p(x_i, y) log \frac{P(x_i, y)}{p(x_i)p(y)}$$

Where:

- ✓ The marginal probability of the feature X_i is represented by $p(x_i)$.
- ✓ The marginal probability of the target Y is represented by p(y).

The variables were ranked through Mutual Information on the significance they contribute to the target prediction. Then, the most suitable *n* features were determined by the MI score.

Step 3: Cluster Generation:

The K-means clustering was then used to cluster the samples together based on these chosen characteristics.

Step 4: Cluster-Specific MI Scoring:

Mutual information scores were recalculated, and the most relevant characteristics specific to each cluster C_j were identified.

$$MIc_j(X_i, Y) = \sum_{x_i, y} PC_j(x_i, y)\, log \frac{PC_j(x_i, y)}{PC_j(x_i)PC_j(y)}$$

Where:

- ✓ $PC_j(x_i)$: The probability of the value x_i for given feature X_i, considering only the samples in cluster C_j.
- ✓ $PC_j(y)$: Within cluster C_j, the probability of the desired value y.
- ✓ $PC_j(x_i, y)$: The joint likelihood that goal Y = y and feature $X_i = x_i$ occur simultaneously within cluster C_j.
- ✓

Step 5: Cluster-Weighted MI Scores:

For every feature Xi, aggregate the MI scores from every cluster into a single score that is weighted by the size or significance of the clusters:

$$MI_{weighted}(X_i) = \sum_{j=1}^{k} w_j \times MI_{Cj}(X_i, Y)$$

Where w_j is the cluster C_j's weight.

Step 6: Genetic Algorithm Optimisation:

A genetic algorithm was used to find the optimal feature subset. Before selecting the best-performing subset, random feature sets have to be initialised, evaluated using importance and prediction accuracy scores, and then iteratively improved through crossover and mutation procedures across many generations.

Step 7: Result:

To obtain a detailed visualisation, the obtained results, containing the feature mutual information scores, clusters obtained, and the final set of selected features, were plotted.

The combination of global and cluster-specific mutual information scoring, followed by genetic algorithm optimisation across many generations, is novel since this multi-stage pipeline that combines global and local feature relevance with evolutionary search has not been used in prior investigations.

7. Classification:

7.1 Traditional Models:

Random Forest (RF) and Extreme Gradient Boosting (XGBoost) were the standard machine learning methods involved in the diagnosis of cardiovascular disease. Hyperparameter optimization was conducted using Optuna to optimize the performance of the model.

Traditional Model (ORF and OXGBoost) working steps:

The initial stage entailed the establishment and preparation of the data to an extent that the model could be analyzed and trained with the right data and environment. Subsequently, an objective function was defined using Optuna along with K-Fold CV to train the model, evaluate the accuracy of the model, select hyperparameters based on a dynamically chosen set, and get the optimisation results. Optuna used several experiments to determine the most suitable hyperparameters and get improved results using the model. The best was then printed, and a plot was produced to indicate the accuracy of all the trials run.

7.2 Optimised Bagging-Boosting Stacked Ensemble (OBBSE):

Optimised Bagging-Boosting Stacked Ensemble is an advanced ensemble learning algorithm, which fuses the benefits of boosting as well as bagging together in a stacked format. This method optimises overall predictive performance by using the Extra Trees (ET) and LightGBM models. The optuna will optimise both models' hyperparameters.

Working Steps of OBBSE:

After the data was prepared, the ET and LightGBM models were created and trained and set to be used in prediction. The likelihoods of prediction generated through the training of ET by K-Fold cross-validation were attached to the original data to produce new training and test sets. LightGBM was trained on this enriched data, and it acted as a meta-model. Optuna was used to test different hyperparameters to identify the best ones in both models. After selecting the most optimal hyperparameters, the models have been retrained and tested on the test set with the most optimal hyperparameters, and the results have been shown.

7.3 Optimised Heterogeneous Soft Voting Ensemble (OHSVE):

An Optimised Heterogeneous Soft Voting Ensemble is a prediction method based on a set of various machine learning models, each of which is optimised to work well. The soft voting is used to come up with the final decision. In this approach, several optimised models are summed up to augment the total forecast. This paper employed the Gaussian Naive Bayes (GNB), Logistic Regression (LR), Support Vector Machine (SVM), and CatBoost as tools to implement OHSVE. They also get optimized to the best performance by optimization of their hyperparameters using Optuna.

Working Steps of OHSVE:

After data preparation and the division into the training and test sets, four models were created: GNB, LR, SVM and CatBoost classifiers and integrated into a soft voting ensemble. K-fold cross-validation has been used to predict and test the soft voting classifier using the training set, and average measurements were calculated. Optuna was applied to optimise the hyperparameters by using a trial. Once the optimal hyperparameters were identified, the best hyperparameters were then utilized to retrain the final soft voting model on the entire training set, test on the test set and present the performance results. This approach will guarantee that the model is modified and analysed. Each trial was then plotted using the results of the experiment.

7.4 Optimised Feature-Augmented Heterogeneous Stacking (OFAHS):

Unlike traditional stacking, feature-augmented stacking combines both original predictor variables and prediction probabilities of the first-level models as inputs to the meta-learner to produce a richer and more informative feature space. This process combines the advantages of several machine learning models by stacking them. Here, LR, ET, SVM, and KNN are base models. The meta-model (XGBoost) derives knowledge from the forecasts made by base models to provide improved outcomes. To improve accuracy and dependability for challenging detection jobs, Optuna ensures that all models are optimised for optimal performance.

Working Steps of OFAHS:

The data was pre-prepared, and four base models, namely LR, ET, SVM, and KNN, and XGBoost as the meta-model, were deployed. The base models were trained on K-fold cross-validation to get prediction probabilities. They were then combined with original features to produce new datasets. A series of trials on the latest sets of data using Optuna was performed to determine the optimal hyperparameters to use with each model. After the optimal parameters were chosen, the models were retrained and tested on the test set with the optimal parameters and the outcomes were presented. The process helped refine and test the models.

7.5 Optimised Heterogeneous Bootstrap-Ensemble (OHBE):

The Optimised Heterogeneous Bootstrap-Ensemble uses hyperparameter optimisation with a number of heterogeneous machine learning models, which are trained on various bootstrapped subsets of the training data. This method improves the predictive performance as well as model resilience. In this case, QDA, SVM, KNN, and RF are applied.

Working Steps of OHBE:

In the case of QDA and both SVM and KNN, bagging models were created, and an RF classifier was directly added due to the bagging mechanism inherent in it. These models were combined to create a soft voting ensemble. The hyperparameters of the model were automatically optimized with the help of Optuna, after which the accuracy of the ensemble was compared with the help of K-fold cross-validation. The final model was then trained using the entire training set and evaluated using the test set. The plots were then drawn to determine the performance of the ensemble technique.

7.6 Optimised Heterogeneous Sequential Boosting (OHSB):

Sequential boosting tries to improve the performance of the model and also reduce bias through successive training of the models, whereby each new model tries to address the mistakes that the previous model made. In this study, LR, DT, SVM, and XGBoost models are used in sequential boosting.

Working Steps of OHSB:

For Optuna, an objective function is developed that recommends model settings. K-fold cross-validation is performed in this operation, with models being trained sequentially with sample weights that are adjusted by error. The validation fold takes the available predictions of the models and evaluates them. Based on many trials, Optuna gets the best settings by averaging the accuracy of all folds. The last model is retrained using all the training data and assessed using the test set after the best hyperparameters are identified. Finally, there are graphs that are presented to show the performance of the model.

8. Implementation on Real-Time Dataset:

8.1 Imputation:

SHAP importance scores of top 9 features are shown in Figure 8. Then these features are used in the imputation process.

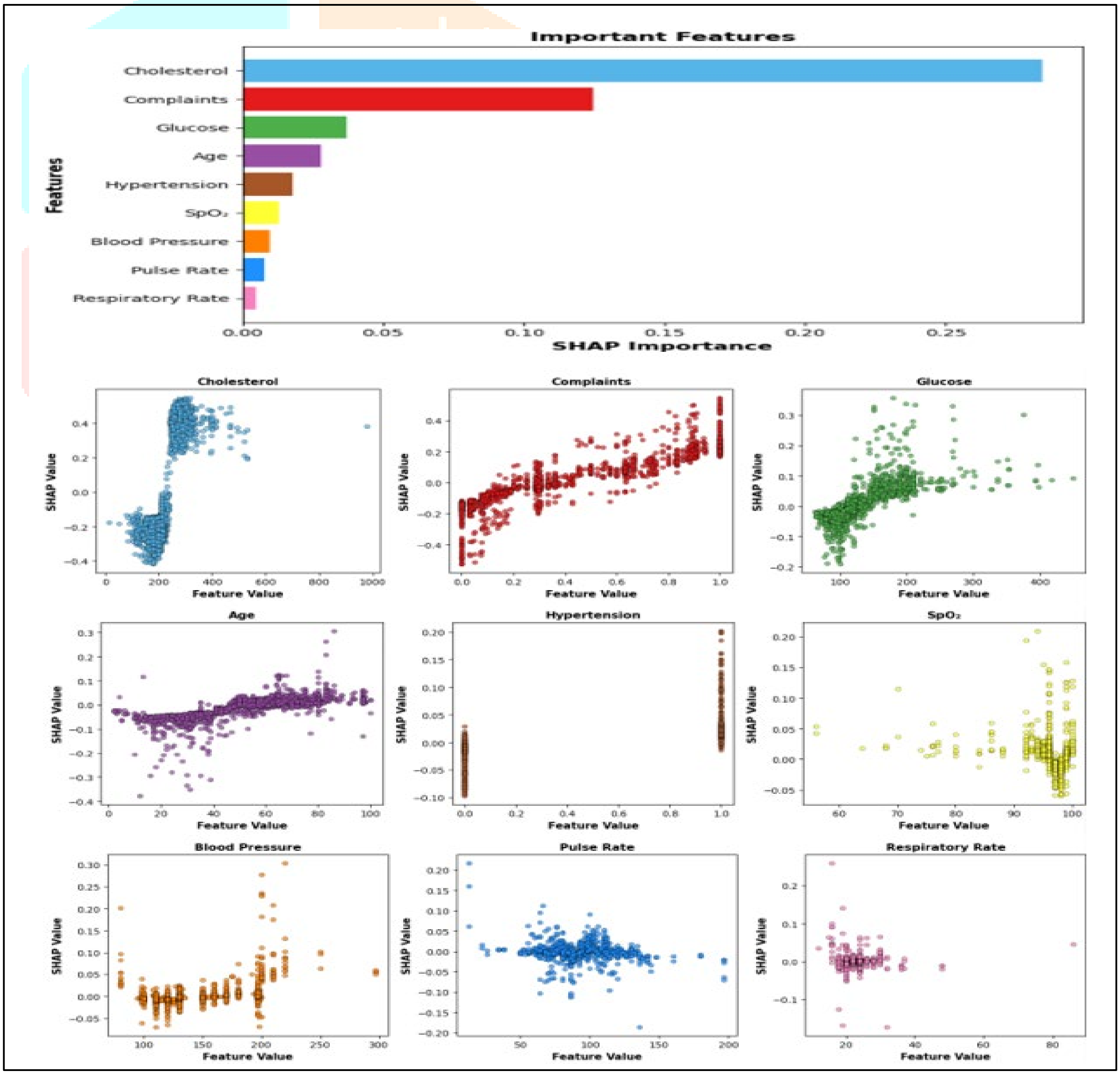
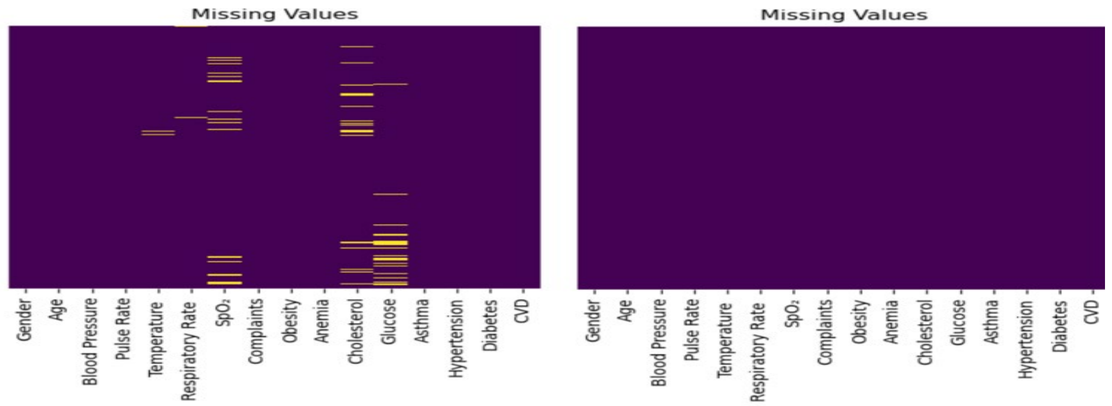


Figure 8: SHAPE Score of Top Nine Features

The missing values before and after imputation using the MF, heuristic rules and SHAP combination are displayed in Figure 9.



a) Before Imputation b) After Imputation

Figure 9: Imputation on Real-time Data

8.2 Handling Outliers:

The dataset's outliers are highlighted in Figure 10.

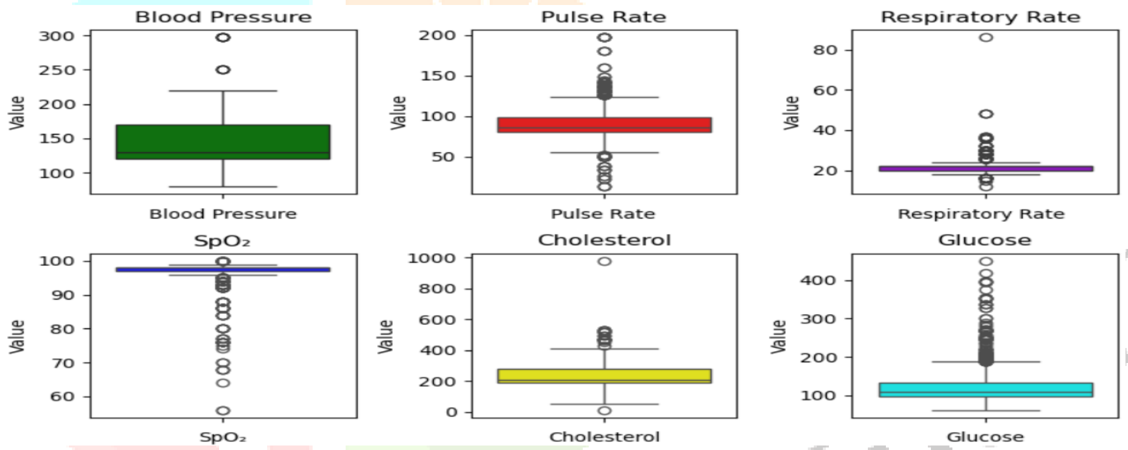


Figure 10: Outliers of Real-time Data

Figure 11 illustrates how the combination of IF, LOF and DBSCAN successfully handles the outliers.

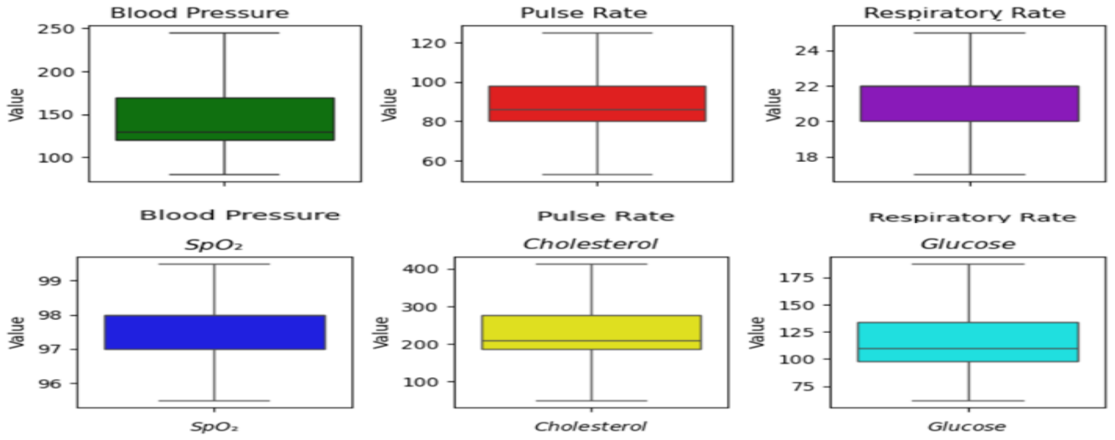
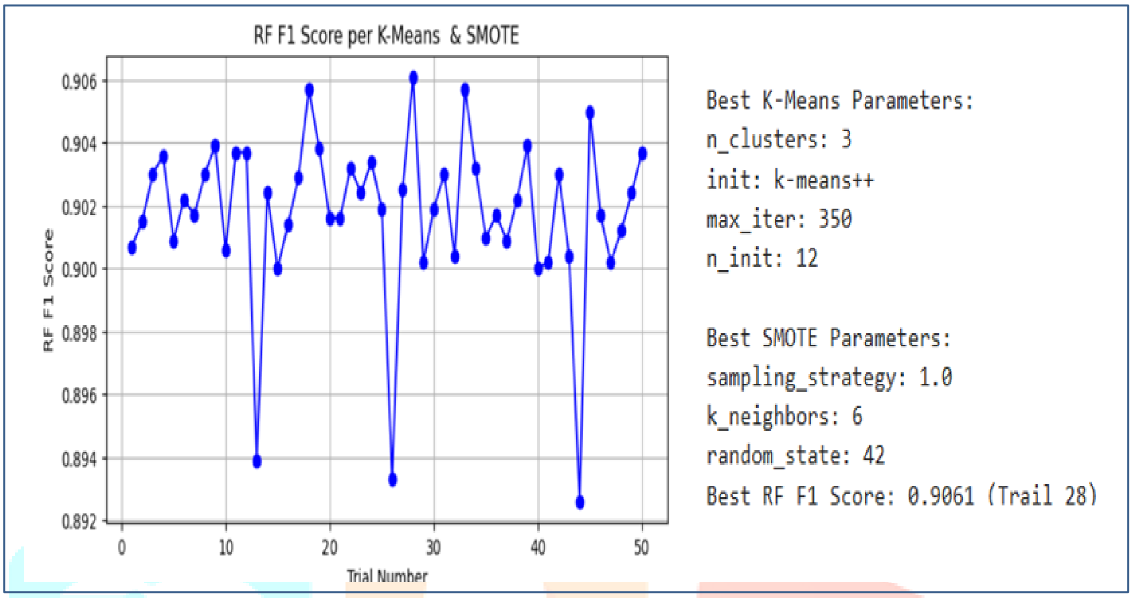


Figure 11: Outliers Handled on Real-Time Data

8.3 Class Balancing:

Figure 12 displays the optimal parameters and scores for the K-Means and SMOTE in class balancing across several trials. The 28th trial achieved the maximum score of 0.906 with the optimal parameters as follows: For K-Means, n_cluster =3, init = k-means++, max_iter =350, and n_init =12; For SMOTE, sampling_strategy =1, n_neighbors = 6 and random_state =42.



```
Best K-Means Parameters:
n_clusters: 3
init: k-means++
max_iter: 350
n_init: 12

Best SMOTE Parameters:
sampling_strategy: 1.0
k_neighbors: 6
random_state: 42
Best RF F1 Score: 0.9061 (Trail 28)
```

Figure 12: Optimisation Result for Class Balancing on Real-Time Data

Figure 13 shows the class distribution before and after balancing using K-Means and SMOTE. Before balancing, Class 1 accounts for 36%, and Class 0 accounts for 64%. After balancing, both classes have an equal distribution of 50% each.

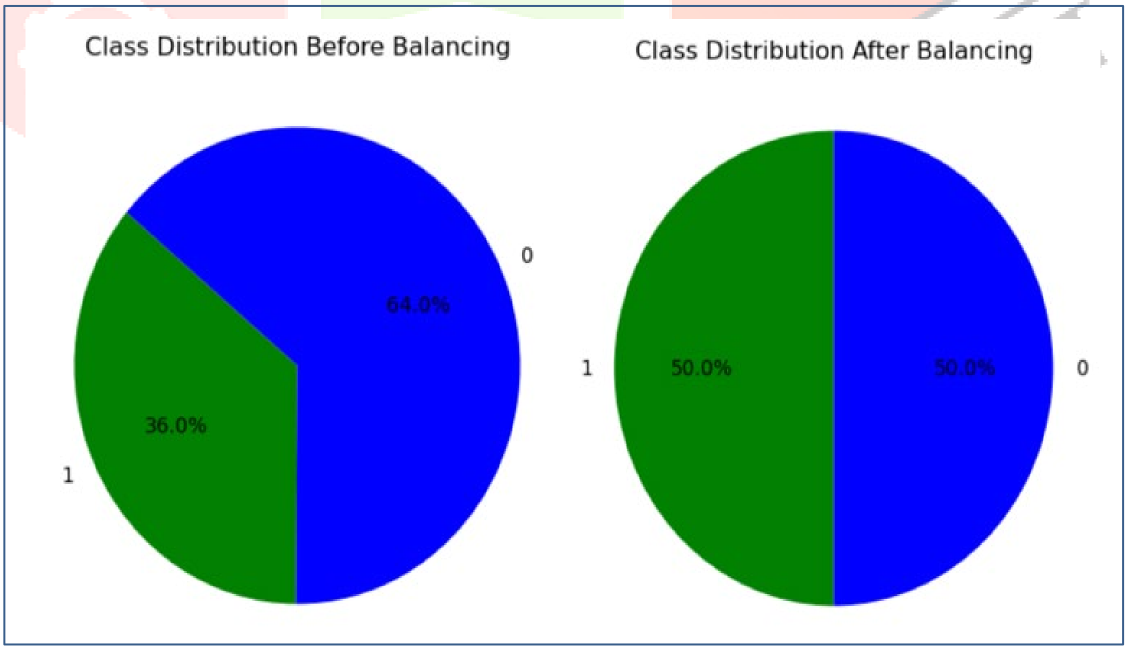


Figure 13: Class Distribution on Real-time Data

8.4 Feature Selection:

Figure 14 displays the top n features that were chosen based on the MI scores, clustering outcomes, mutual information scores for every cluster, cluster-weighted MI scores, and the final feature subset.

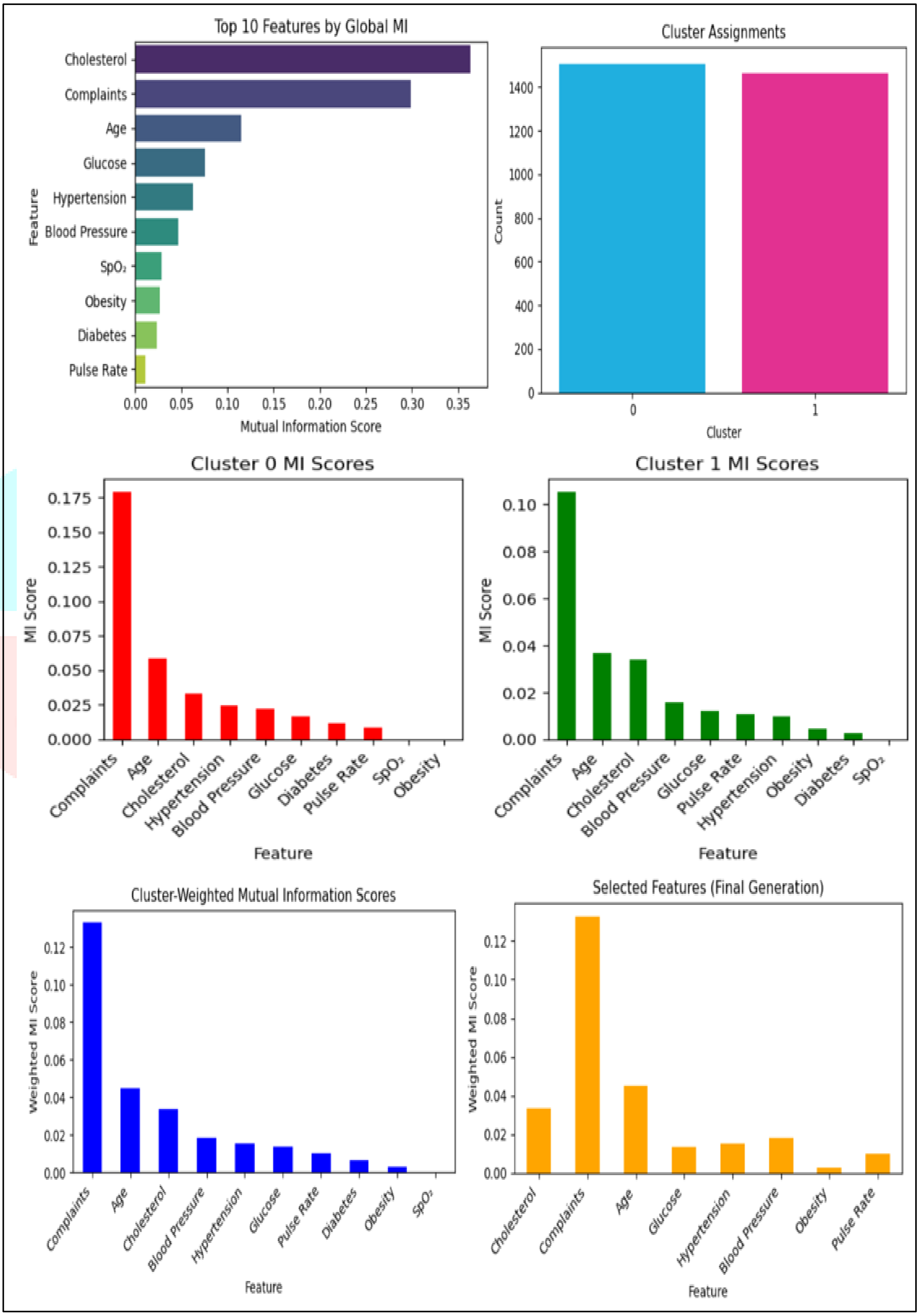
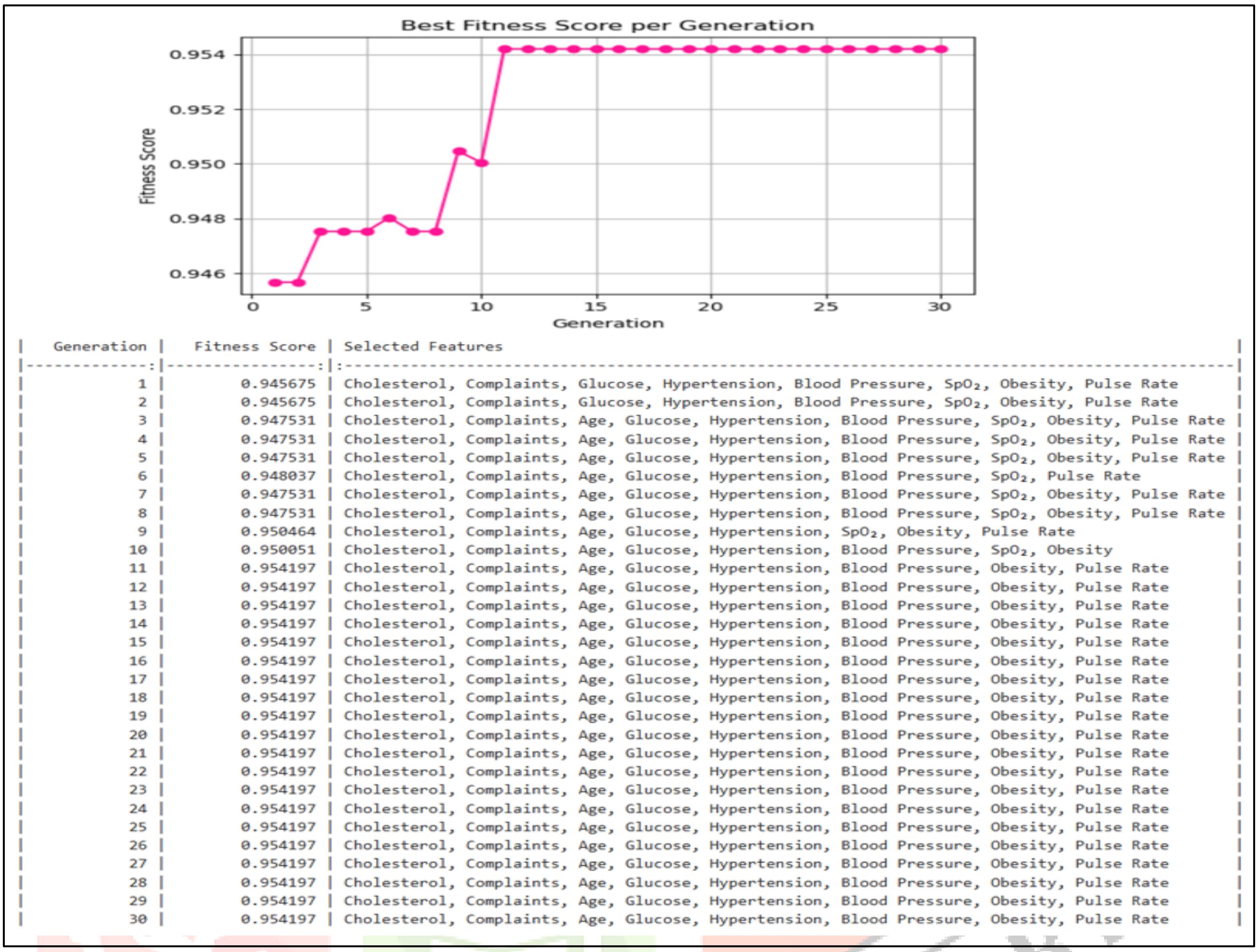


Figure 14: Feature Selection via Mutual Information, Clustering and GA

Figure 15 illustrates the evolution of the feature subsets chosen at each generation throughout 30 generations. It shows how the feature sets are tuned and eventually stabilised with progression in the approach with each generation, reflecting the iterative nature behind the selection process. This shows how the chosen qualities were cultivated and how they were transformed over time.



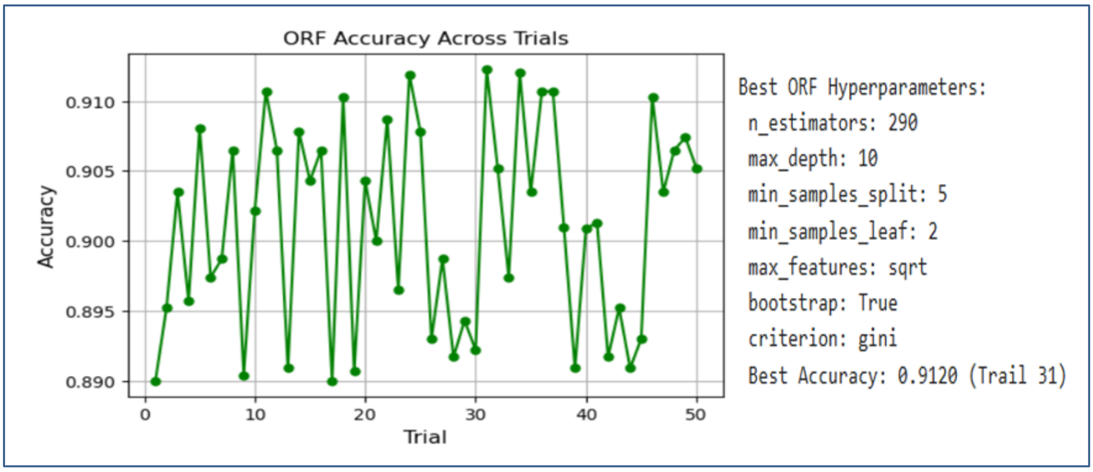
```
|   Generation |   Fitness Score | Selected Features                                                                       |
|-------------:|----------------:|:----------------------------------------------------------------------------------------|
|            1 |        0.945675 | Cholesterol, Complaints, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate      |
|            2 |        0.945675 | Cholesterol, Complaints, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate      |
|            3 |        0.947531 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate |
|            4 |        0.947531 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate |
|            5 |        0.947531 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate |
|            6 |        0.948037 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Pulse Rate          |
|            7 |        0.947531 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate |
|            8 |        0.947531 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity, Pulse Rate |
|            9 |        0.950464 | Cholesterol, Complaints, Age, Glucose, Hypertension, SpO₂, Obesity, Pulse Rate                 |
|           10 |        0.950051 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, SpO₂, Obesity             |
|           11 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           12 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           13 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           14 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           15 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           16 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           17 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           18 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           19 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           20 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           21 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           22 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           23 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           24 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           25 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           26 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           27 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           28 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           29 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
|           30 |        0.954197 | Cholesterol, Complaints, Age, Glucose, Hypertension, Blood Pressure, Obesity, Pulse Rate       |
```

Figure 15: Evolution of Features Subset Across Generation

8.5 Classification:

8.5.1 Optimised Random Forest:

Optimised random forest achieved the highest level of accuracy of 0.912 on trial 31, with n_estimators = 290, max depth = 10, min samples split =5, min samples leaf =2, bootstrap=True, max features= sqrt, and criterion= gini parameters (Figure 16).

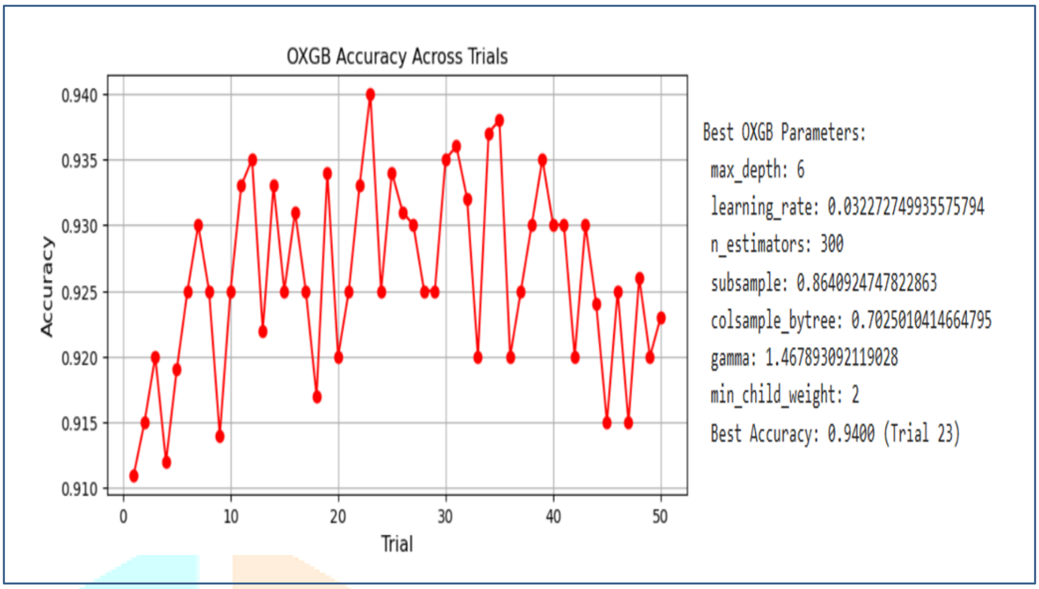


```
Best ORF Hyperparameters:
 n_estimators: 290
 max_depth: 10
 min_samples_split: 5
 min_samples_leaf: 2
 max_features: sqrt
 bootstrap: True
 criterion: gini
 Best Accuracy: 0.9120 (Trail 31)
```

Figure 16: ORF Result on Real-time Data

8.5.2 Optimised XGBoost:

Figure 17 displays the best parameters and accuracy of the optimised XGBoost in several trials.



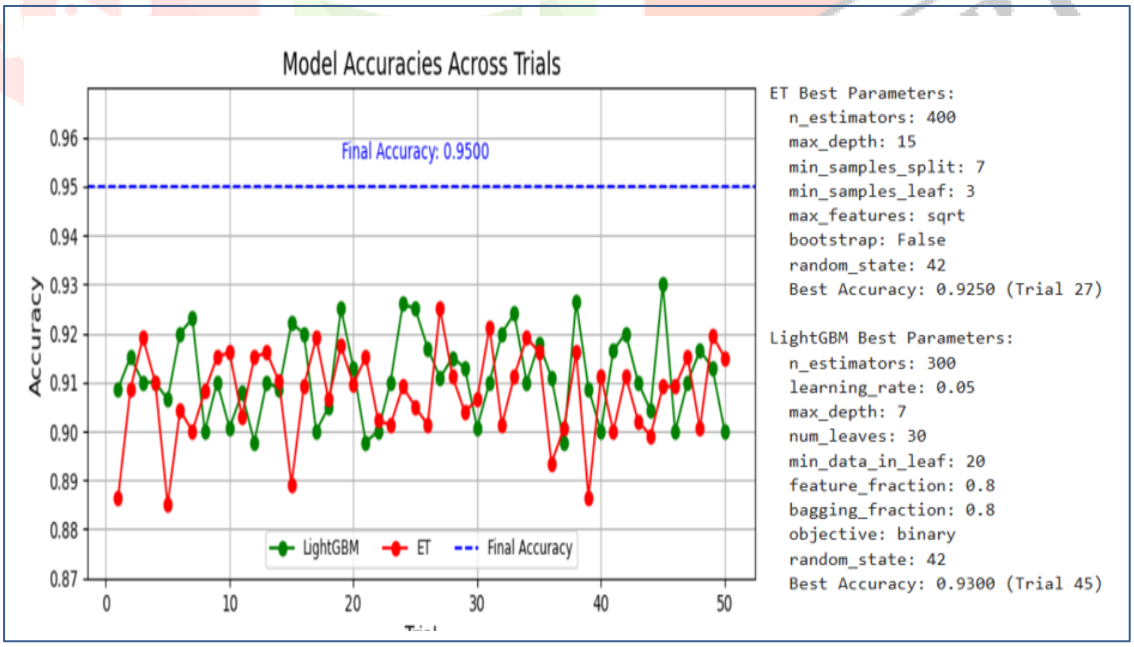
```
Best OXGB Parameters:
 max_depth: 6
 learning_rate: 0.032272749935575794
 n_estimators: 300
 subsample: 0.8640924747822863
 colsample_bytree: 0.7025010414664795
 gamma: 1.467893092119028
 min_child_weight: 2
 Best Accuracy: 0.9400 (Trial 23)
```

Figure 17: **OXGB Result on Real-time Data**

The highest accuracy of 0.94 was obtained at the 23rd attempt with the best parameters being as shown below: max depth = 6, learning rate = 0.03, nestimators = 300, subsamples = 0.86, minchildweight = 2, etc.

5.3.3 Optimised Bagging-Boosting Stacked Ensemble (OBBSE):

Figure 18 displays the ideal parameters and scores for the OBBSE over several trials. At the 27th trial, ET produced the best accuracy of 0.925, and at the 45th trial, LightGBM produced the highest accuracy of 0.93. Finally, after using these optimised parameters, OBBSE reached 0.95 accuracy.

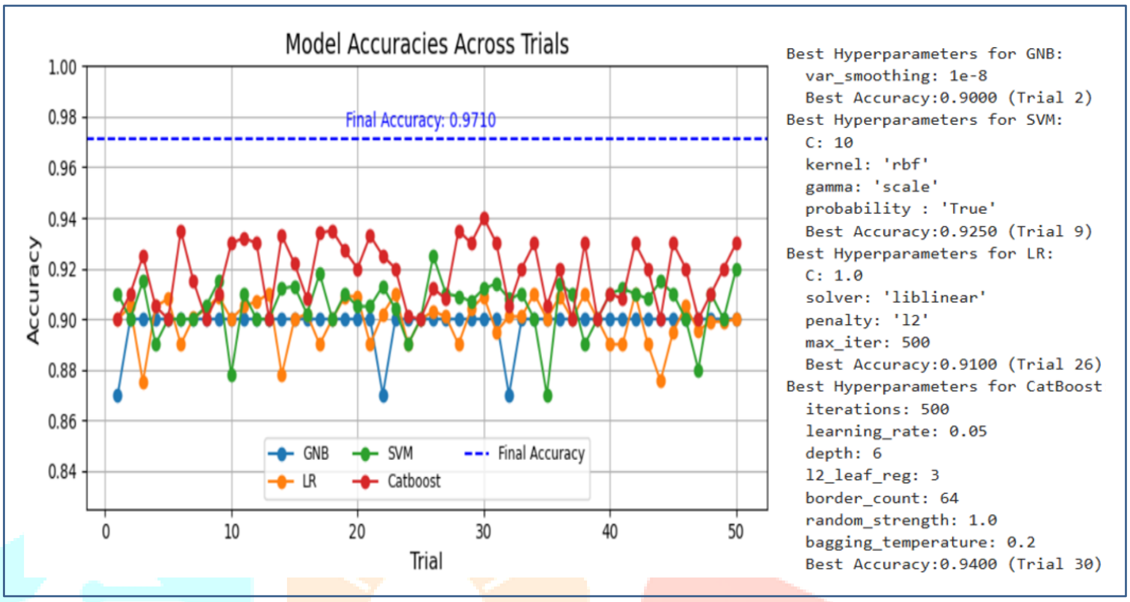


```
ET Best Parameters:
  n_estimators: 400
  max_depth: 15
  min_samples_split: 7
  min_samples_leaf: 3
  max_features: sqrt
  bootstrap: False
  random_state: 42
  Best Accuracy: 0.9250 (Trial 27)

LightGBM Best Parameters:
  n_estimators: 300
  learning_rate: 0.05
  max_depth: 7
  num_leaves: 30
  min_data_in_leaf: 20
  feature_fraction: 0.8
  bagging_fraction: 0.8
  objective: binary
  random_state: 42
  Best Accuracy: 0.9300 (Trial 45)
```

Figure 18: OBBSE Result on Real-time Data

8.5.4 Optimised Heterogeneous Soft Voting Ensemble (OHSVE):

Figure 19 provides the optimal parameters and accuracy of OHSVE at multiple trials. The GNB was the most accurate with trial 2, with an accuracy of 0.90; the LR was the most accurate at 26, with 0.91; the SVM model was the most accurate at trial 9, with 0.925; and the CatBoost was the most accurate at trial 30, with 0.94. The models were retrained and estimated on the test set with the optimum parameters, and the ultimate soft-voting accuracy was obtained as 0.971.



```
Best Hyperparameters for GNB:
  var_smoothing: 1e-8
  Best Accuracy:0.9000 (Trial 2)
Best Hyperparameters for SVM:
  C: 10
  kernel: 'rbf'
  gamma: 'scale'
  probability : 'True'
  Best Accuracy:0.9250 (Trial 9)
Best Hyperparameters for LR:
  C: 1.0
  solver: 'liblinear'
  penalty: 'l2'
  max_iter: 500
  Best Accuracy:0.9100 (Trial 26)
Best Hyperparameters for CatBoost
  iterations: 500
  learning_rate: 0.05
  depth: 6
  l2_leaf_reg: 3
  border_count: 64
  random_strength: 1.0
  bagging_temperature: 0.2
  Best Accuracy:0.9400 (Trial 30)
```

Figure 19: **OHSVE Result on Real-time Data**

8.5.5 Optimised Feature-Augmented Heterogeneous Stacking (OFAHS):

Figure 20 represents the optimal parameters and classification accuracy of the LR, ET, SVM, KNN, and XGBoost using a series of trials. The ET model achieved its maximum accuracy in the 19th trial of 0.9303, and the LR model achieved the maximum accuracy in the 26th trial of 0.9187. The SVM model achieved its maximum accuracy of 0.926 in the 14th trial, and the KNN model achieved its maximum accuracy of 0.9202 in the 35th trial. The XGBoost model reached a peak accuracy of 0.9455 in the 37th trial. Once the models had been retrained and tested on the test set using their respective optimal parameters, their stacking accuracy was 0.991. The XGBoost will act as the meta-learner in this case.

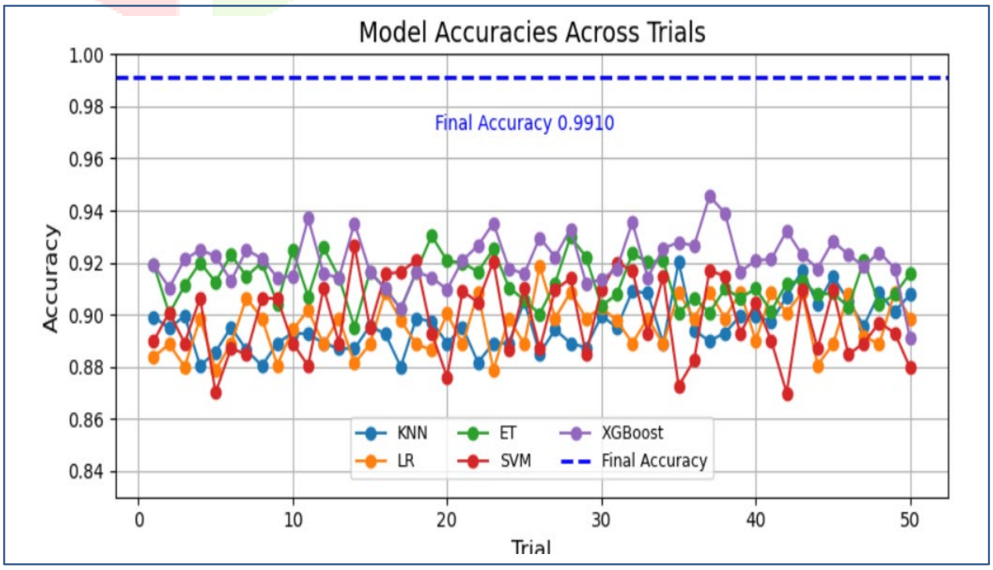
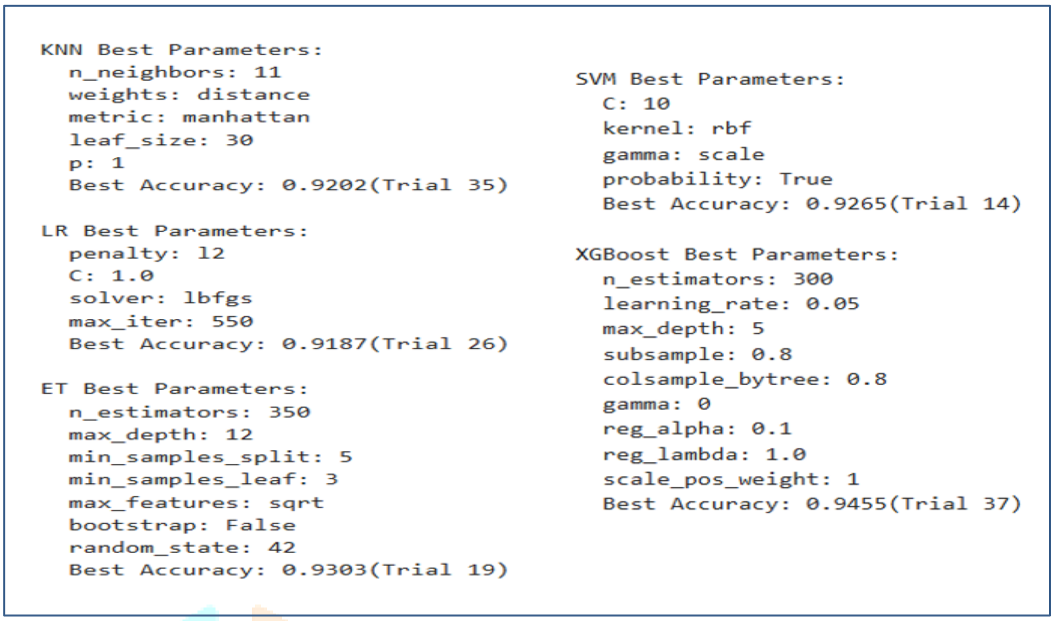


Figure 20: **OFAHS Result on Real-time Data**

The plot of the stacking result is shown in Figure 21.

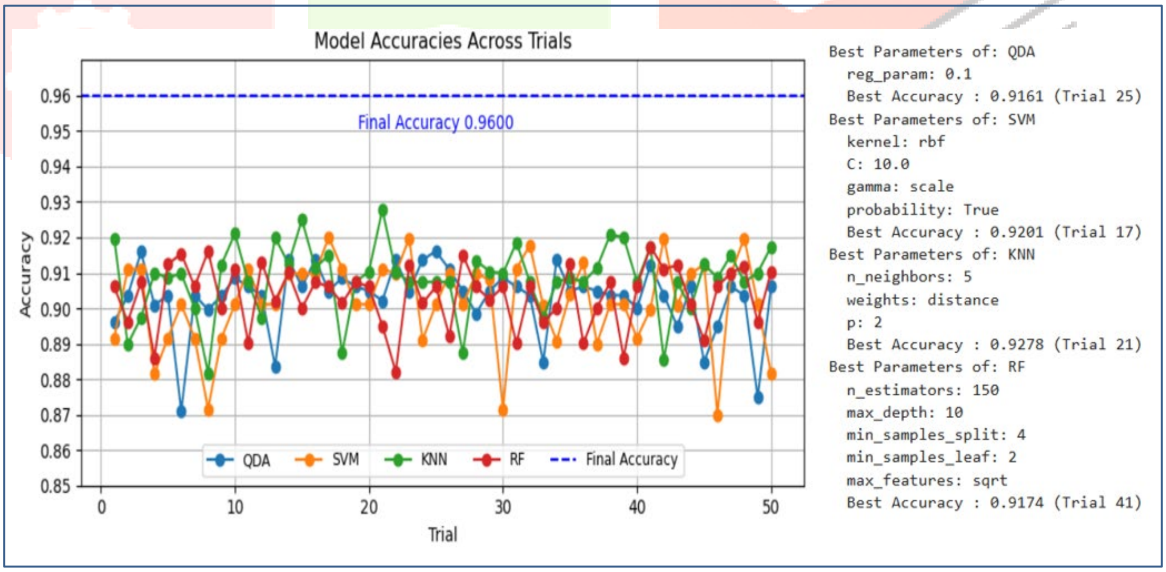


```
KNN Best Parameters:
  n_neighbors: 11
  weights: distance
  metric: manhattan
  leaf_size: 30
  p: 1
  Best Accuracy: 0.9202(Trial 35)

LR Best Parameters:
  penalty: l2
  C: 1.0
  solver: lbfgs
  max_iter: 550
  Best Accuracy: 0.9187(Trial 26)

ET Best Parameters:
  n_estimators: 350
  max_depth: 12
  min_samples_split: 5
  min_samples_leaf: 3
  max_features: sqrt
  bootstrap: False
  random_state: 42
  Best Accuracy: 0.9303(Trial 19)

SVM Best Parameters:
  C: 10
  kernel: rbf
  gamma: scale
  probability: True
  Best Accuracy: 0.9265(Trial 14)

XGBoost Best Parameters:
  n_estimators: 300
  learning_rate: 0.05
  max_depth: 5
  subsample: 0.8
  colsample_bytree: 0.8
  gamma: 0
  reg_alpha: 0.1
  reg_lambda: 1.0
  scale_pos_weight: 1
  Best Accuracy: 0.9455(Trial 37)
```

Figure 21: Best Parameters of OFAHS on Real-Time Data

8.5.6 Optimised Heterogeneous Bootstrap-Ensemble (OHBE):

The ideal parameters and classification accuracy attained by combining QDA, KNN, SVM, and RF over several trials are shown in Figure 22. In the 25th trial, the QDA bagging model achieved its best accuracy of 0.916, whereas in the 17th trial, the SVM bagging model achieved its highest accuracy of 0.92. In the 21st attempt, the KNN achieved its highest accuracy of 0.928, while the RF reached its highest accuracy of 0.917 at the 41st trial. The final soft voting accuracy was 0.96 when the models were retrained and assessed on the test set using their optimal settings.

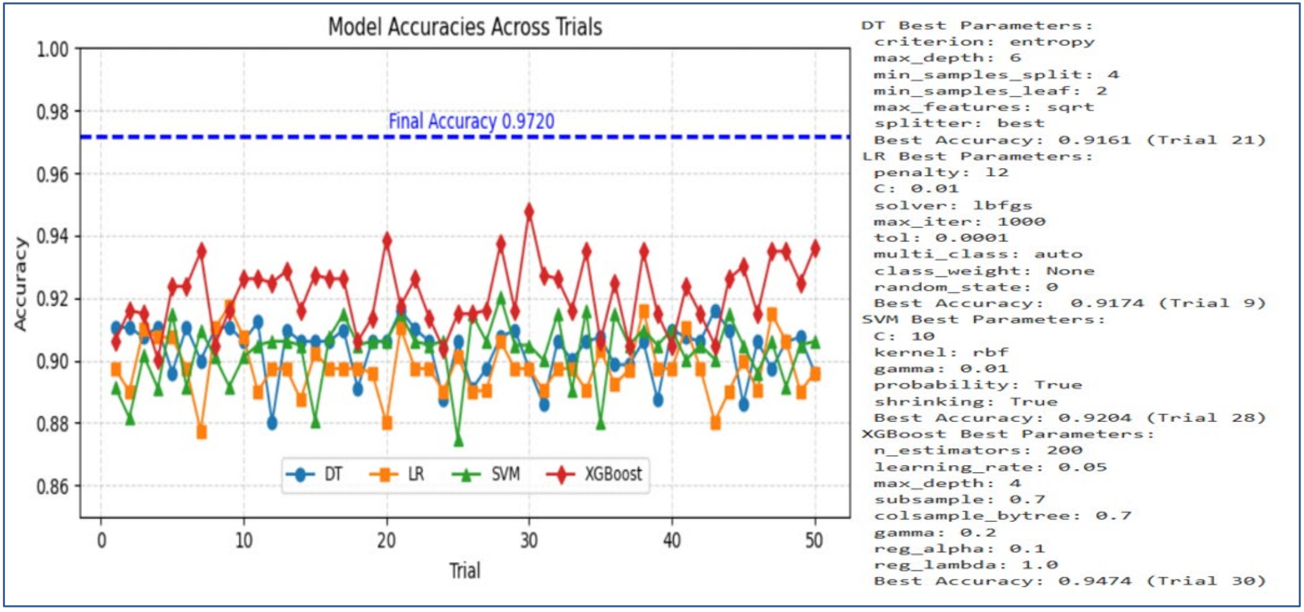


```
Best Parameters of: QDA
  reg_param: 0.1
  Best Accuracy : 0.9161 (Trial 25)
Best Parameters of: SVM
  kernel: rbf
  C: 10.0
  gamma: scale
  probability: True
  Best Accuracy : 0.9201 (Trial 17)
Best Parameters of: KNN
  n_neighbors: 5
  weights: distance
  p: 2
  Best Accuracy : 0.9278 (Trial 21)
Best Parameters of: RF
  n_estimators: 150
  max_depth: 10
  min_samples_split: 4
  min_samples_leaf: 2
  max_features: sqrt
  Best Accuracy : 0.9174 (Trial 41)
```

Figure 22: **OHBE Result on Real-time Data**

8.5.7 Optimised Heterogeneous Sequential Boosting (OHSB):

The best parameters and the classification accuracy of the DT, LR, SVM and XGBoost achieved after several trials are presented in Figure 23. The highest accuracy of the DT model was 0.9161 in the 21st trial, and the LR model had the highest accuracy of 0.9174 in the 9th trial. The SVM model achieved its best accuracy of 0.9204 during the 28th trial, and the XGBoost model achieved its best accuracy of 0.9474 during the 30th trial (22). The last sequence with the highest rate of detection was 0.972 during retraining of the models, which was tested on the test set using the models' respective optimal parameters.

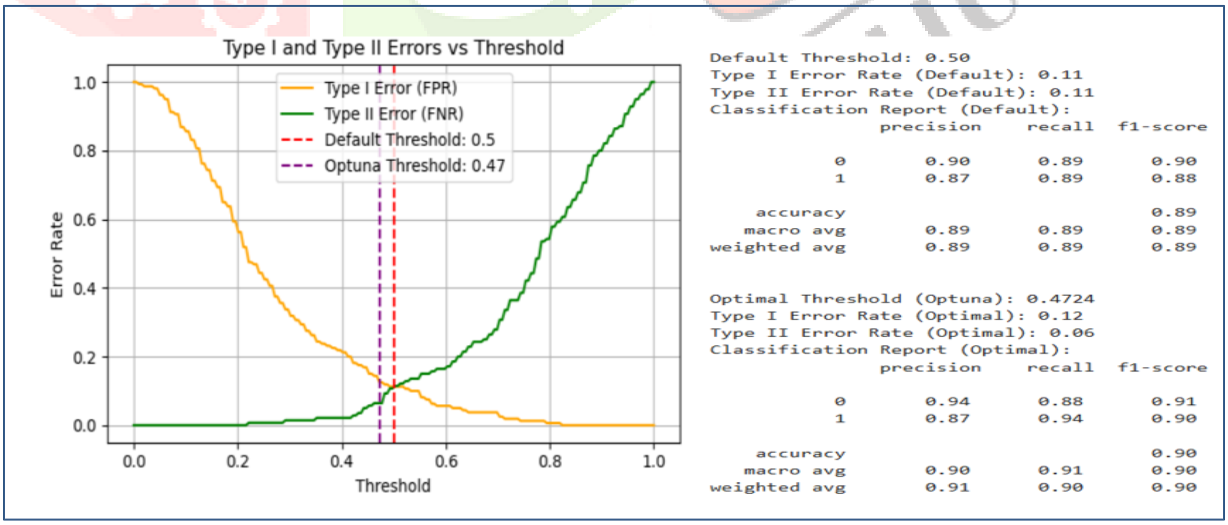


```
DT Best Parameters:
 criterion: entropy
 max_depth: 6
 min_samples_split: 4
 min_samples_leaf: 2
 max_features: sqrt
 splitter: best
 Best Accuracy: 0.9161 (Trial 21)
LR Best Parameters:
 penalty: l2
 C: 0.01
 solver: lbfgs
 max_iter: 1000
 tol: 0.0001
 multi_class: auto
 class_weight: None
 random_state: 0
 Best Accuracy:  0.9174 (Trial 9)
SVM Best Parameters:
 C: 10
 kernel: rbf
 gamma: 0.01
 probability: True
 shrinking: True
 Best Accuracy: 0.9204 (Trial 28)
XGBoost Best Parameters:
 n_estimators: 200
 learning_rate: 0.05
 max_depth: 4
 subsample: 0.7
 colsample_bytree: 0.7
 gamma: 0.2
 reg_alpha: 0.1
 reg_lambda: 1.0
 Best Accuracy: 0.9474 (Trial 30)
```

Figure 23: OHSB -Result on Real-time Data

9. Threshold Optimisation for Reducing Type 2 Errors:

A type II error or a false negative in medical diagnostics is a failure to identify a disease or other medical condition. It gives the patients the feeling that they are healthy. Therefore, this makes them continue with bad habits such as eating junk food, failing to exercise, smoking, taking alcohol, failing to receive prompt treatment, living in a lot of stress and failing to receive the help they require. It is possible to unknowingly relax, and the disease will spread without anyone noticing. This may contribute to the aggravation of the condition, the loss of early treatment and in extreme cases, avoidable death. Therefore, Type 2 mistakes are more problematic than Type 1 mistakes in the healthcare industry; thus, it is essential to minimize them. The reduction of type 2 mistakes can be achieved by reducing the threshold, and the optimal approach towards this is optimisation. The optimal threshold value is obtained with the help of Optuna in this case.



```
Default Threshold: 0.50
Type I Error Rate (Default): 0.11
Type II Error Rate (Default): 0.11
Classification Report (Default):
              precision    recall  f1-score

           0       0.90      0.89      0.90
           1       0.87      0.89      0.88

    accuracy                           0.89
   macro avg       0.89      0.89      0.89
weighted avg       0.89      0.89      0.89


Optimal Threshold (Optuna): 0.4724
Type I Error Rate (Optimal): 0.12
Type II Error Rate (Optimal): 0.06
Classification Report (Optimal):
              precision    recall  f1-score

           0       0.94      0.88      0.91
           1       0.87      0.94      0.90

    accuracy                           0.90
   macro avg       0.90      0.91      0.90
weighted avg       0.91      0.90      0.90
```

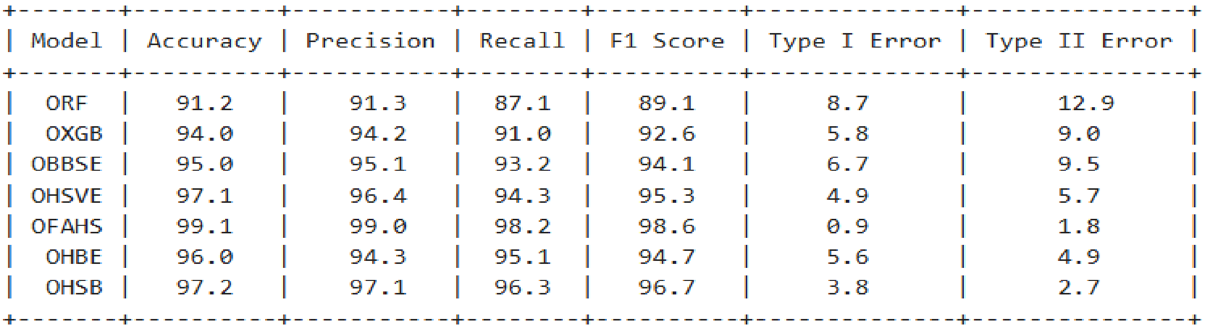
Figure 24: Threshold Optimisation

An LR is utilised to determine the ideal threshold value (Figure 24). With the default threshold value, the accuracy is 89%, the type 1 error rate is 0.11, and the type 2 error rate is 0.11. The type 1 error rating increased from 0.11 to 0.12, the type 2 error rating dropped from 0.11 to 0.06, and the accuracy increased from 0.89 to 0.90 at the ideal threshold value of 0.47.

10. Result Comparison:

10.1 Result Comparison on Real-Time Data:

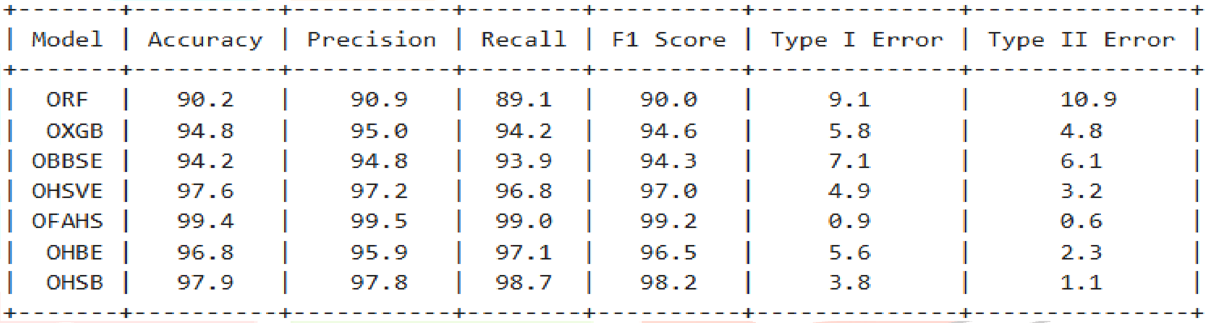
All the pre-processing and classification processes were done on real-time information. Eight out of fifteen traits were selected for detection. Figure 25 depicts the performance of each model at the default threshold value. It indicates that Type II error was steadily declining in all models, as required to minimise the quantity of false negatives.



```
+-------+-----------+-----------+--------+----------+--------------+---------------+
| Model | Accuracy  | Precision | Recall | F1 Score | Type I Error | Type II Error |
+-------+-----------+-----------+--------+----------+--------------+---------------+
|  ORF  |   91.2    |   91.3    |  87.1  |   89.1   |     8.7      |     12.9      |
|  OXGB |   94.0    |   94.2    |  91.0  |   92.6   |     5.8      |     9.0       |
| OBBSE |   95.0    |   95.1    |  93.2  |   94.1   |     6.7      |     9.5       |
| OHSVE |   97.1    |   96.4    |  94.3  |   95.3   |     4.9      |     5.7       |
| OFAHS |   99.1    |   99.0    |  98.2  |   98.6   |     0.9      |     1.8       |
|  OHBE |   96.0    |   94.3    |  95.1  |   94.7   |     5.6      |     4.9       |
|  OHSB |   97.2    |   97.1    |  96.3  |   96.7   |     3.8      |     2.7       |
+-------+-----------+-----------+--------+----------+--------------+---------------+
```

Figure 25: Performance Metrics Using Default Threshold Value on Real-time Data

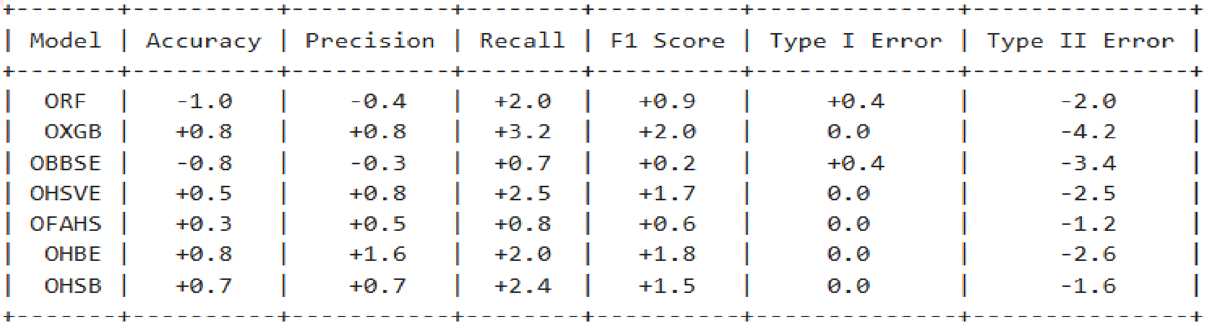
Figure 26 demonstrates the performance metrics of each model using the optimal threshold value.



```
+-------+-----------+-----------+--------+----------+--------------+---------------+
| Model | Accuracy  | Precision | Recall | F1 Score | Type I Error | Type II Error |
+-------+-----------+-----------+--------+----------+--------------+---------------+
|  ORF  |   90.2    |   90.9    |  89.1  |   90.0   |     9.1      |     10.9      |
|  OXGB |   94.8    |   95.0    |  94.2  |   94.6   |     5.8      |     4.8       |
| OBBSE |   94.2    |   94.8    |  93.9  |   94.3   |     7.1      |     6.1       |
| OHSVE |   97.6    |   97.2    |  96.8  |   97.0   |     4.9      |     3.2       |
| OFAHS |   99.4    |   99.5    |  99.0  |   99.2   |     0.9      |     0.6       |
|  OHBE |   96.8    |   95.9    |  97.1  |   96.5   |     5.6      |     2.3       |
|  OHSB |   97.9    |   97.8    |  98.7  |   98.2   |     3.8      |     1.1       |
+-------+-----------+-----------+--------+----------+--------------+---------------+
```

Figure 26: Performance Metrics Using Optimal Threshold Value on Real-time Data

Figure 27 illustrates the differences in performance metrics between the default and ideal threshold values for the models.



```
+-------+-----------+-----------+--------+----------+--------------+---------------+
| Model | Accuracy  | Precision | Recall | F1 Score | Type I Error | Type II Error |
+-------+-----------+-----------+--------+----------+--------------+---------------+
|  ORF  |   -1.0    |   -0.4    |  +2.0  |   +0.9   |     +0.4     |     -2.0      |
|  OXGB |   +0.8    |   +0.8    |  +3.2  |   +2.0   |     0.0      |     -4.2      |
| OBBSE |   -0.8    |   -0.3    |  +0.7  |   +0.2   |     +0.4     |     -3.4      |
| OHSVE |   +0.5    |   +0.8    |  +2.5  |   +1.7   |     0.0      |     -2.5      |
| OFAHS |   +0.3    |   +0.5    |  +0.8  |   +0.6   |     0.0      |     -1.2      |
|  OHBE |   +0.8    |   +1.6    |  +2.0  |   +1.8   |     0.0      |     -2.6      |
|  OHSB |   +0.7    |   +0.7    |  +2.4  |   +1.5   |     0.0      |     -1.6      |
+-------+-----------+-----------+--------+----------+--------------+---------------+
```

Figure 27: Performance Metrics Differences Between the Default and Ideal Threshold Values

Though Type I error rates grew to a slight extent in two of the models (ORF and OBBSE), this is a justifiable and expected cost, because reducing Type II error rates (false negatives) is more crucial to patient safety. Comprehensively, the accuracy remained the same or improved, which proves the usefulness of adjusting the threshold to detect actual positive cases more efficiently.

All methods are compared using the optimal threshold value on real-time data in terms of accuracy, precision, recall, F1 score and Type 1 error and Type 2 error (Figure 28). OFAHS proved to be better than any other model, with an accuracy of 99.4.

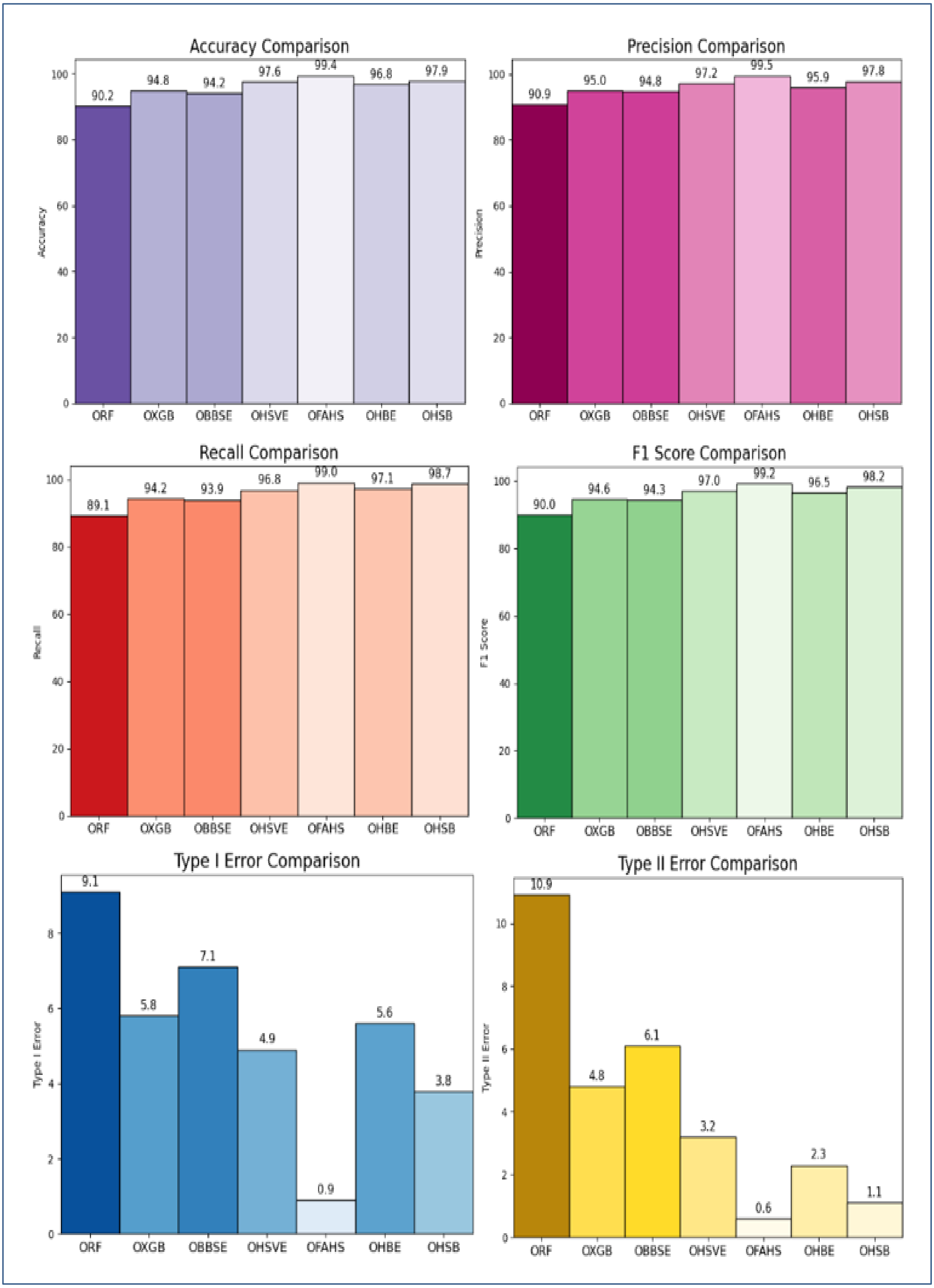


Figure 28: **Result Comparison on Real-Time Data**

Figure 29 compares the confusion matrices of all models using real-time data.

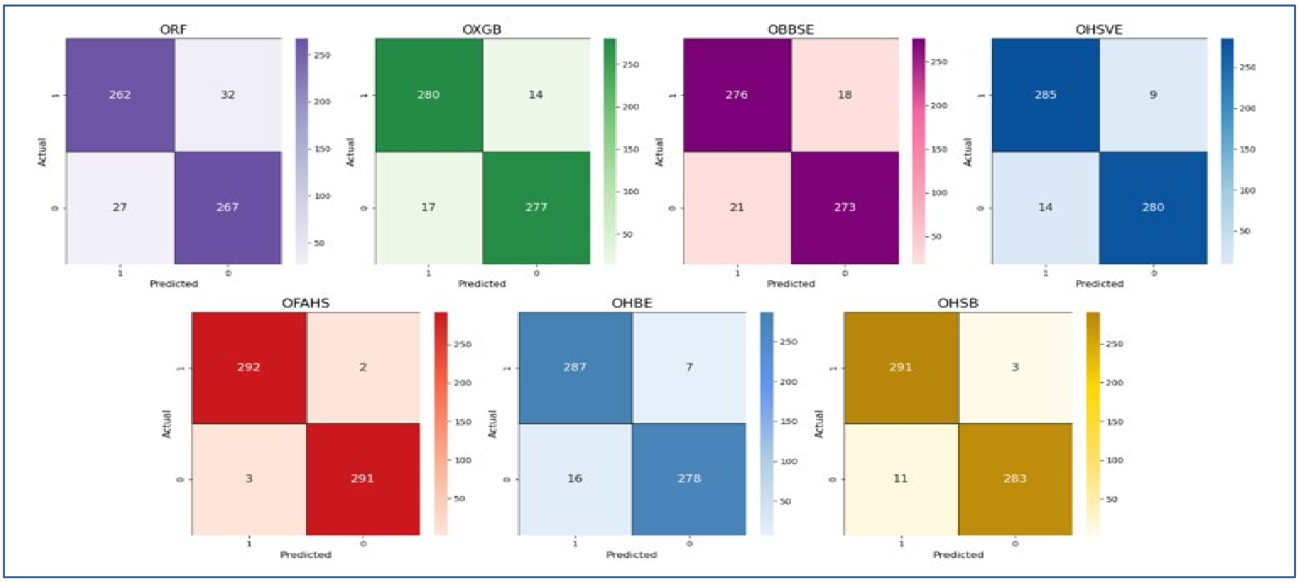
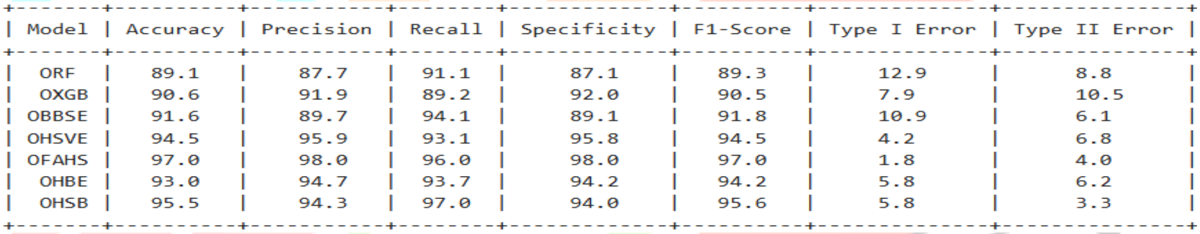


Figure 29: Confusion Matrices of Real-Time Data

10.2 Result Comparison on Benchmark Data:

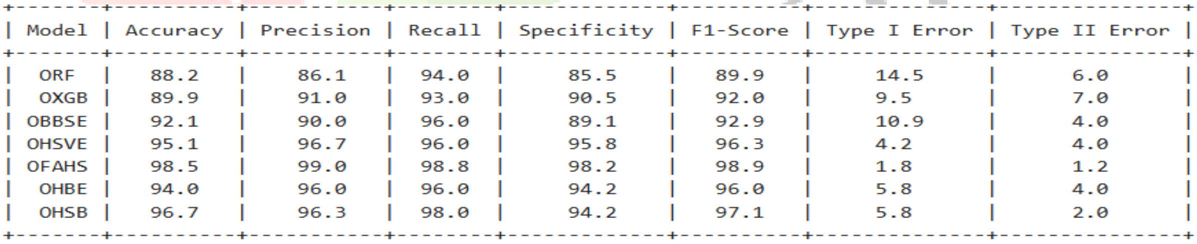
All the pre-processing steps and classification were done on benchmark data. Eight characteristics out of eleven were selected for detection. Figure 30 compares the accuracy, precision, recall, F1 score, Type 1 error, and Type 2 error of all models that apply the baseline decision boundary to the benchmark data.



```
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
| Model | Accuracy | Precision | Recall | Specificity | F1-Score | Type I Error | Type II Error |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
|  ORF  |   89.1   |    87.7   |  91.1  |     87.1    |   89.3   |     12.9     |      8.8      |
|  OXGB |   90.6   |    91.9   |  89.2  |     92.0    |   90.5   |     7.9      |      10.5     |
| OBBSE |   91.6   |    89.7   |  94.1  |     89.1    |   91.8   |     10.9     |      6.1      |
| OHSVE |   94.5   |    95.9   |  93.1  |     95.8    |   94.5   |     4.2      |      6.8      |
| OFAHS |   97.0   |    98.0   |  96.0  |     98.0    |   97.0   |     1.8      |      4.0      |
|  OHBE |   93.0   |    94.7   |  93.7  |     94.2    |   94.2   |     5.8      |      6.2      |
|  OHSB |   95.5   |    94.3   |  97.0  |     94.0    |   95.6   |     5.8      |      3.3      |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
```

Figure 30: Performance Metrics Using Default Threshold Value on Benchmark Data

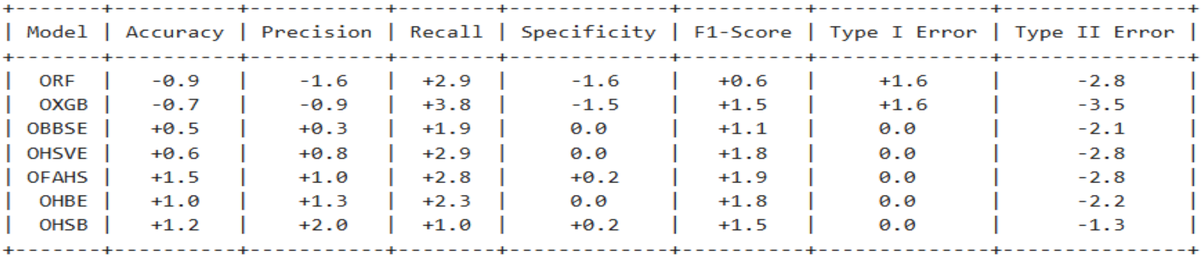
Figure 31 illustrates the performance metrics of all models using the optimal decision boundary on the benchmark data.



```
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
| Model | Accuracy | Precision | Recall | Specificity | F1-Score | Type I Error | Type II Error |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
|  ORF  |   88.2   |    86.1   |  94.0  |     85.5    |   89.9   |     14.5     |      6.0      |
|  OXGB |   89.9   |    91.0   |  93.0  |     90.5    |   92.0   |     9.5      |      7.0      |
| OBBSE |   92.1   |    90.0   |  96.0  |     89.1    |   92.9   |     10.9     |      4.0      |
| OHSVE |   95.1   |    96.7   |  96.0  |     95.8    |   96.3   |     4.2      |      4.0      |
| OFAHS |   98.5   |    99.0   |  98.8  |     98.2    |   98.9   |     1.8      |      1.2      |
|  OHBE |   94.0   |    96.0   |  96.0  |     94.2    |   96.0   |     5.8      |      4.0      |
|  OHSB |   96.7   |    96.3   |  98.0  |     94.2    |   97.1   |     5.8      |      2.0      |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
```

Figure 31: Performance Metrics Using Optimal Threshold Value on Benchmark Data

The differences in the models' performance metrics between the default and ideal threshold values are displayed in Figure 32. All models showed a decrease in Type II error. Accuracy improved for all models, except two (ORF, OXGB). It is acceptable that just two models displayed a minor rise in Type I error because, in this case, minimising Type II mistakes is more important.



```
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
| Model | Accuracy | Precision | Recall | Specificity | F1-Score | Type I Error | Type II Error |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
|  ORF  |   -0.9   |    -1.6   |  +2.9  |     -1.6    |   +0.6   |     +1.6     |      -2.8     |
|  OXGB |   -0.7   |    -0.9   |  +3.8  |     -1.5    |   +1.5   |     +1.6     |      -3.5     |
| OBBSE |   +0.5   |    +0.3   |  +1.9  |     0.0     |   +1.1   |     0.0      |      -2.1     |
| OHSVE |   +0.6   |    +0.8   |  +2.9  |     0.0     |   +1.8   |     0.0      |      -2.8     |
| OFAHS |   +1.5   |    +1.0   |  +2.8  |     +0.2    |   +1.9   |     0.0      |      -2.8     |
|  OHBE |   +1.0   |    +1.3   |  +2.3  |     0.0     |   +1.8   |     0.0      |      -2.2     |
|  OHSB |   +1.2   |    +2.0   |  +1.0  |     +0.2    |   +1.5   |     0.0      |      -1.3     |
+-------+----------+-----------+--------+-------------+----------+--------------+---------------+
```

Figure 32: Benchmark Data Performance Differences at Default and Ideal Thresholds

Figure 33 compares the accuracy, precision, recall, F1 score, and types of mistakes (Type 1 and Type 2) using the ideal threshold value for each model on the benchmark data. OFAHS fared better than any other model, with an accuracy of 98.5%.

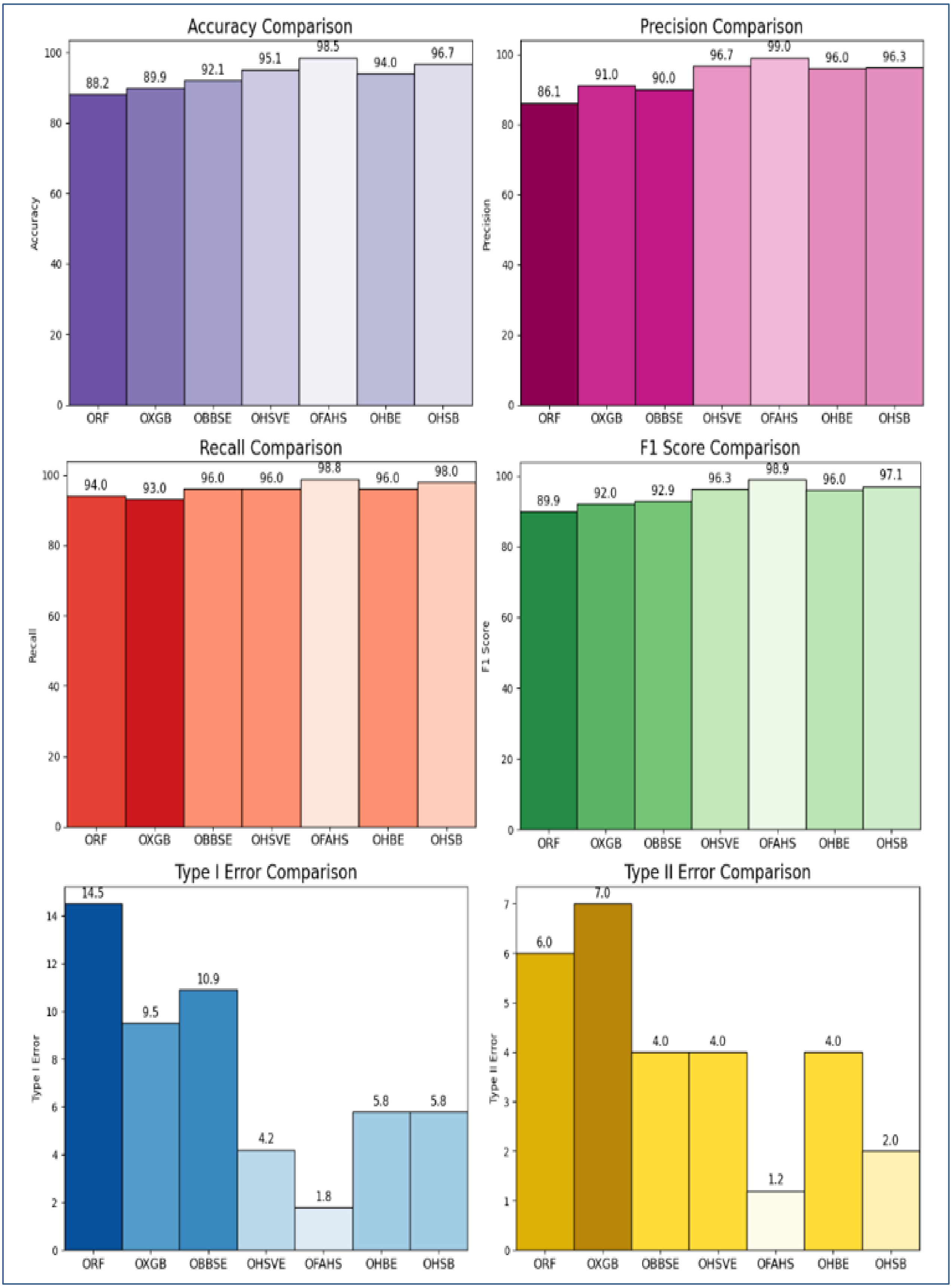


Figure 33: **Result Comparison on Benchmark Data**

Using benchmark data, Figure 34 compares the confusion matrices of each model.

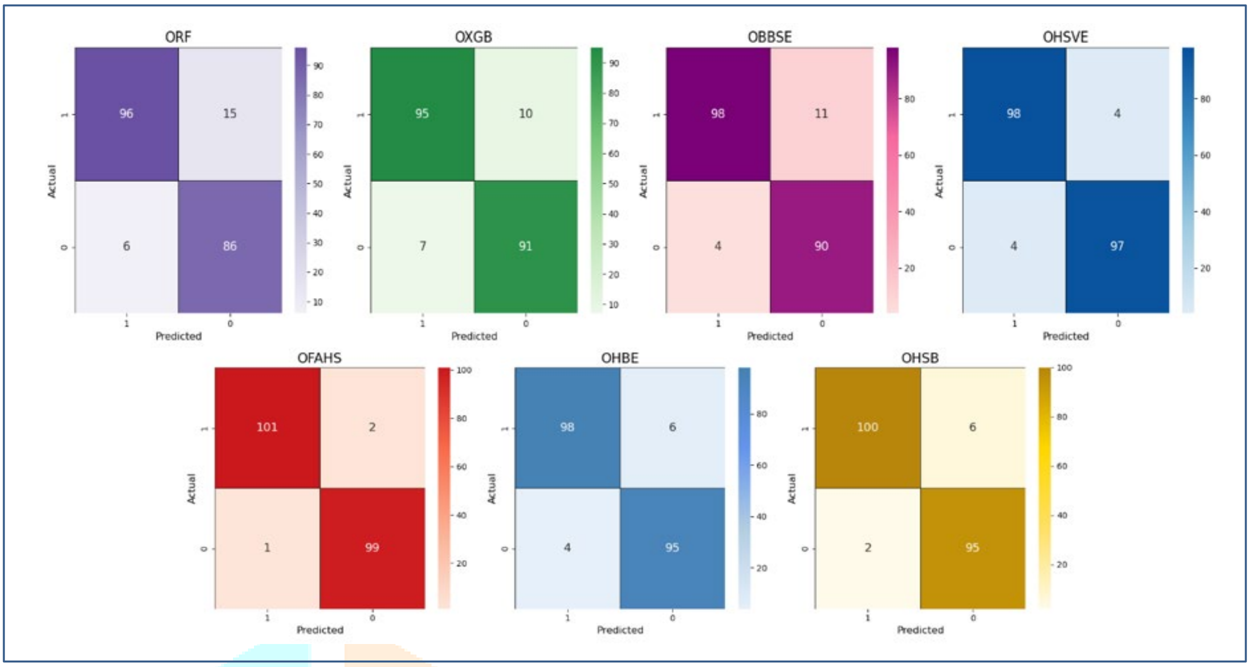
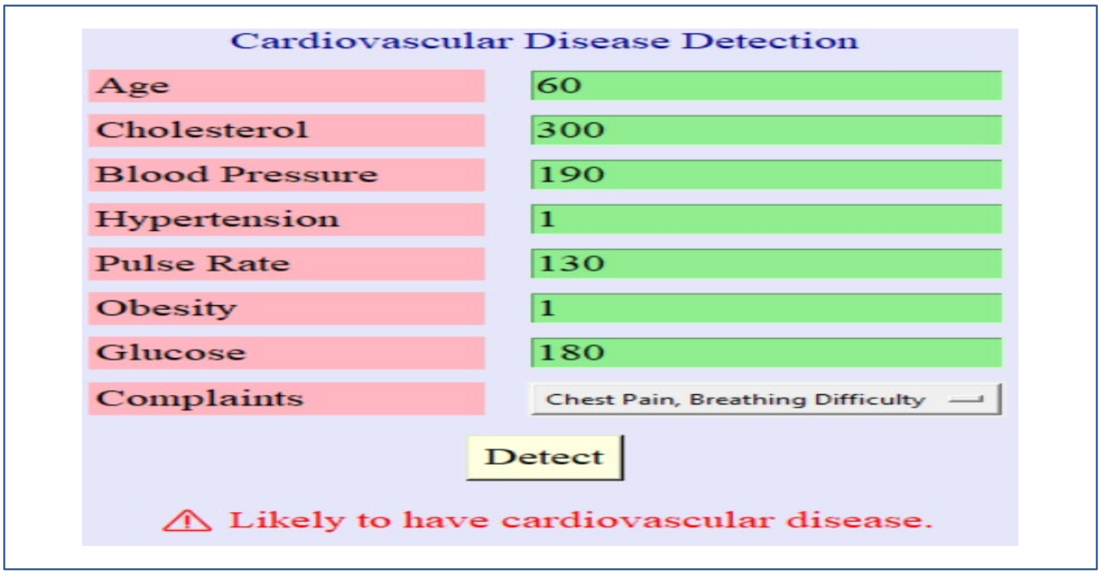


Figure 34: Confusion Matrices of Benchmark Data

11. CVD Predictive System:

The cardiovascular disease detection system has been developed using the Python programming language. A desktop application was developed that uses Tkinter, which comes as part of the Python standard library, to increase the usability of the system by making it a graphical user interface (GUI) application. It does not require any technical expertise, making it easier to interact with the system. It does not need an internet connection as it is fully offline, and is applicable in places with low connectivity. Also, data privacy is enhanced by the fact that the local storage of patient data boosts privacy. The system has low hardware requirements, enabling it to run on PCs due to its lightweight and easy maintenance. Its modularity further gives it the opportunity to be enhanced in the future, such as providing more advanced preprocessing and feature selection methods, or re-training the predictive model using larger data sets. The combination of the ease of use, offline functionality, and secure local data handling of the system makes it a feasible decision-support tool for the identification of cardiovascular disease. The solution is particularly beneficial to urban clinics and the local hospitals since it presents a cost-effective, easily implemented, and readily available technology to assist in patient management and early diagnosis.



Cardiovascular Disease Detection

Age	60
Cholesterol	300
Blood Pressure	190
Hypertension	1
Pulse Rate	130
Obesity	1
Glucose	180
Complaints	Chest Pain, Breathing Difficulty

⚠ Likely to have cardiovascular disease.

Figure 35: CVD Predictive System on the Real-Time Dataset

The OFAHS model used in our predictive system uses the selected traits that are incorporated to detect individuals at risk of cardiovascular disease. Figure 35 indicates that the patient with age (60), cholesterol (300), blood pressure (190), hypertension (1), pulse rate (130), obesity (1), glucose (180), and complains of "chest Pain, Breathing Difficulty" is positively diagnosed with cardiovascular disease.

12. Accuracy Compared with other studies:

Our innovative methods obtained the highest accuracy on both datasets compared to prior relevant studies. Among our methods, OFAHS reached the maximum accuracy (Table 2).

Table 2: Accuracy Comparison with Related Studies

S. No	Author	Pre-processing	Classification	Best Accuracy
1	Jingyi Zhang et al.,2021 [11]	DR: PCA	Stacking Using Many Classifiers	87.7
2	K. Senthamarai Kannan et al., 2024 [14]	Outlier Handling: k-NN, LOF	NB, SVM	83.1% (LOF, NB, HND2)
3	Bilal Ahmad et al., 2025 [15]	Feature Selection: MI, ANOVA F-test, Chi-Square test	NN, LR, RF, GBoost, AdaBoost, DT, LDA, SVM, Nu-SVC, k-NN, NB	82.3 (MI with NN)
4	Ghalia A. Alshehri et al., 2023 [17]	Normalisation: MinMaxScaler, Balancing: SMOTE, FS: forward / backward	AdaBoost, SVM (Linear Kernel), DT, RF, Ensemble (ELA)-Adaptive boosting, SVM, DT, and RF	Z-Alizadeh Sani: 91% (ELA) StatLog: 83% (ELA) CVD: 73% (ELA)
5	Vaishali M Deshmukh 2019 [19]	Normalisation: MinMaxScaler, Standardization: StandardScaler Feature selection: ET	Ensemble Methods**:** Majority Voting with Bagging (DT, LR, ANN, KNN, NB)	87.78
6	**Novel Work**	**Imputation: HSAMF** **Outlier Handling: HGLS-OD** **Encoding: Target Encoding** **Balancing: OKSMOTE** **Normalisation: MinMax** **FS: CWMI-GA**	**Real-Time Data** **OBBSE** **OHBE** **OHSVE** **OHSB** **OFAHS** **Benchmark Data** **OBBSE** **OHBE** **OHSVE** **OHSB** **OFAHS**	**94.2** **96.8** **97.6** **97.9** **99.4** **92.1** **94** **95.1** **96.7** **98.5**

13. Conclusion and Future Enhancements:

The system implements a number of strategies to improve the process of CVD detection, such as advanced data pre-treatment methods, which guarantee the consistency and quality of the data, such as hybrid imputation, outlier management, class balancing, and feature selection, for both datasets. To be classified, ORF, OXGB, OBBSE, OHSVE, OFAHS, OHBE, and OHSB were taken. Optuna helps to define the best parameters used to implement a classification job successfully. The most precise classification models on the two data sets were OFAHS. By implementing the best threshold value, the percentage of type 2 errors has been reduced. With the optimal threshold value, the accuracy of real-time data increased to 99.4% and benchmark data had an increase of 98.5%. It was revealed through threshold tuning that the framework was robust and could classify well by enhancing prediction reliability, especially by minimizing Type II errors. The model will be improved using advanced AI techniques to make it more scalable and adaptable in a broad range of areas.

Reference:

[1] Krishnamoorthy Natarajan, V. Vinoth Kumar et al, “Efficient Heart Disease Classification Through Stacked Ensemble with Optimized Firefly Feature Selection”, Volume 17, article number 173, (2024).

[2] A Review on Disease Diagnosis Using Machine Learning Techniques," International Journal of Pure and Applied Mathematics, Volume 117, No. 16, 2017.

[3] Reem A. Alassaf et al, "Preemptive Diagnosis of Chronic Kidney Disease Using Machine Learning Techniques", International Conference on Innovations in Information Technology (I.T.), IEEE, 2018.

[4] Gopi Battineni, Getu Gamo Sagaro, Nalini Chinatalapudi, and Francesco Amenta, "Applications of Machine Learning Predictive Models in the Chronic Disease Diagnosis", Journal of Personalized Medicine, 2020.

[5] Diyar Qader Zeebare, Habibollah Haron, Adnan Mohsin Abdulazeez, and Dilovan Asaad Zebari, "Machine learning and Region Growing for Breast Cancer Segmentation", International Conference on Advanced Science and Engineering, IEEE, 2019.

[6] Joel Jacob, Joseph Chakkalakal Mathew, Johns Mathew, and Elizabeth Issac, "Diagnosis of Liver Disease Using Machine Learning Techniques", International Research Journal of Engineering and Technology, Vol. 05, Issue: 04, Apr-2018.

[7] Siddhesh Iyer, Shivkumar Thevar, Priyamurgan Guruswamy, Ujwala Ravale, "Heart Disease Prediction Using Machine Learning", International Research Journal of Modernization in Engineering Technology and Science, Vol. 02, Issue 07, July 2020.

[8] Dilovan Asaad Zebari, Diyar Qader Zeebaree, Adnan Mohsin Abdulazeez, Habibollah Haron, and Haza Nuzly Abdul Hamed, "Improved Threshold Based and Trainable Fully Automated Segmentation for Breast Cancer Boundary and Pectoral Muscle in Mammogram Images, IEEE Access, Vol. 8, 2020.

[9] Sneha Grampurohit and Chetan Sagarnal, "Disease Prediction using Machine Learning Algorithms", International Conference for Emerging Technology (I.N.C.E.T.), Belgaum, India. Jun 5-7, 2020.

[10] Pahulpreet Singh Kohli and Shriya Arora, "Application of Machine Learning in Disease Prediction", International Conference on Computing Communication and Automation (I.C.C.C.A.), IEEE, 2018.

[11] Jingyi Zhang et al., “Ensemble machine learning approach for screening of coronary heart disease based on echocardiography and risk factors”, BMC Med Inform Decis Mak (2021) 21:187 https://doi.org/10.1186/s12911-021-01535-5

[12] Hu et al. “A novel MissForest-based missing values imputation approach with recursive feature elimination in medical applications”, BMC Medical Research Methodology (2024) 24:269 https://doi.org/10.1186/s12874-024-02392-2

[13] Waljee AK, Mukherjee A, et al. “Comparison of imputation methods for missing laboratory data in medicine”, BMJ Open 2013;3: e002847. doi:10.1136/bmjopen-2013002847.

[14] K. Senthamarai Kannan, D. Kabinath et al." A Comparative Study of Outlier Detection Methods in Heart Disease Data", Journal of Computational Analysis and Applications VOL. 33, NO. 2, 2024, 10.48047/jocaaa.2024.33.02.27

[15] Bilal Ahmad et al., "Feature selection strategies for optimized heart disease diagnosis using ML and DL models", https://doi.org/10.48550/arXiv.2503.16577.

[16] Moiz Ur Rehman, Shahid Naseem, "Predicting coronary heart disease with advanced machine learning classifiers for improved cardiovascular risk assessment", PMCID: PMC12006408, PMID: 40247042, 2025 Apr 17;15:13361. doi: 10.1038/s41598-025-96437-1

[17] Ghalia A. Alshehri et al., "*Prediction of Heart Disease using an Ensemble Learning Approach*", International Journal of Advanced Computer Science and Applications (IJACSA), Volume 14, Issue 8, 2023,
DOI: 10.14569/IJACSA.2023.01408118

[18] Chandralekha E, S. Vinodhini, "Heart Rate Anomaly Detection in Healthcare Using Elliptic Envelope and Local Forest", International Conference on Machine Learning and Data Engineering, Procedia Computer Science 258 (2025) 1677–1687

[19] Vaishali M Deshmukh, "*Heart Disease Prediction using Ensemble Methods*", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-3, September 2019.

[20] Jeevan Babu Maddala, Bhargav Reddy Modugulla, "Heart Failure Prediction Using Machine Learning", DOI Link: https://doi.org/10.22214/ijraset.2024.59236

[21] T. Jayasudha, Dr R. Uma Rani, "An Innovative Machine Learning Framework for Cardiovascular Disease Detection", Submitted.

[22] Jenny Elizabeth Price et al., "Xgboost and Support Vector Machines: Comparing the Interpretability of Machine Learning Models". http://dx.doi.org/10.2139/ssrn.4336957