International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 565-582

https://doi.org/10.70917/ijcisim-2025-0036
Article

Comparative Analysis of Computational Intelligence Techniques for Predicting Coronary Artery Heart Disease: A Study on Logistic Regression, Support Vector Machine, and K-Nearest Neighbor

Sanjay Dhanka [1,*], Ankur Kumar [2], Abhinav Sharma [3], Anchal Sharma [2], Monika Nain [4], Komal Chanania [5], Nitin Kumar Saxena [6] and Surender Kumar Sharma [7]

[1] Department of Electrical Engineering, Graphic Era Deemed to be University, Clement Town, Dehradun 248002, India; sanjaykumar506070@gmail.com;
[2] School of Computing and Electrical Engineering (SCEE), Indian Institute of Technology (IIT), Mandi, Himachal Pradesh, India 175075; khatriankur007@yahoo.com (A.K.); anchal.15000@gmail.com (A.S.);
[3] Department of Electrical and Instrumentation Engineering, Sant Longowal Institute of Engineering and Technology, Longowal, Sangrur, Punjab, India 148106; abhinav.engi@gmail.com;
[4] Department of Mathematics, Sant Longowal Institute of Engineering and Technology, Longowal, Sangrur, Punjab, India 148106; nainmonika551@gmail.com;
[5] Department of Computer Science and Engineering, Baba Farid College of Engineering and Technology, Bhatinda, Punjab, India 151001; komal2k2oo@gmail.com;
[6] Department of Electrical and Electronics Engineering, KIET Group of Institutions, Delhi-NCR, Ghaziabad, India 201206; nitinsaxena.iitd@gmail.com
[7] Department of Electrical Engineering, Poornima University, Jaipur, Rajasthan, India 303905; surendra.sharma@poornima.edu.in
* Correspondence author: sanjaykumar506070@gmail.com

Received date: 30 April 2025; Accepted date: 21 October 2025; Published online: 31 December 2025

Abstract: Aim: This study aimed to evaluate the performance of three machine learning models Logistic Regression (LR), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN) in predicting coronary artery heart disease using the Cleveland dataset. These algorithms were selected due to their interpretability, computational efficiency, and established utility as baseline classifiers in medical prediction tasks. **Subject and Methods:** The Cleveland dataset from the UCI repository was used, split into 90% training and 10% testing. Principal Component Analysis was applied for feature selection and stratified cross-validation helped address class imbalance. Models were evaluated based on accuracy, sensitivity, precision, F1-score, AUC-ROC, and confusion matrices. Hyperparameter tuning was performed using grid search with cross-validation. **Results:** LR outperformed other models, achieving the highest accuracy (90%), sensitivity (93.33%), and precision (87.50%). It also recorded the best AUC-ROC score (0.90) and effectively minimized both false positive and false negatives. SVM showed moderate performance (86.67% accuracy), while KNN was the least accurate (83.33%) with a higher false positive rate. Statistical significance testing (McNemar's test, $p < 0.05$) confirmed that LR's advantage over KNN was significant. Comparative analysis with recent studies indicated that the proposed approach performs competitively. **Conclusion:** LR proved to be the most effective model for heart disease prediction in this study. The use of PCA and stratified validation contributed to robust performance. These results support the role of interpretable ML models in clinical decision-support systems. Future work should include more diverse data sources, additional clinical variables, and explainable AI techniques to improve trust and deployment potential.

Keywords: heart disease; machine learning; logistic regression; classification

1. Introduction

The heart serves as a vital foundation of human physiology; its proper function is essential for sustaining life. Any disruption in its operation sends ripples through the body, affecting critical organs such as the brain and kidneys. Acting as a fundamental pump, the heart drives blood circulation throughout the body; insufficient blood flow detrimentally impacts various organs, particularly the brain's function. The abrupt halt of heart activity leads to rapid demise within minutes, highlighting the heart's indispensable role in maintaining life. The efficiency of the heart governs the quality of life, with heart disease encompassing a range of cardiac conditions and disorders [1], [2]. Numerous factors, including anxiety, depression, smoking, sedentary lifestyle, hypertension, high cholesterol, and obesity, heighten the risk of heart disease. The World Health Organization (WHO) forecasts that by 2030, approximately 23.6 million individuals will succumb to heart-related ailments. Often, the disease goes undetected in its early stages, resulting in a significant number of fatalities. Early detection can avert many patient deaths, relying on symptoms, physical examinations, and clinical indicators such as functional and pathological factors associated with heart disease. However, these factors occasionally impede and complicate disease prediction, leading to adverse perceptions and unpredictable outcomes [2]. Throughout history, humans have made remarkable advancements in both machinery and healthcare. In modern times, the integration of machines and artificial intelligence (AI) into medical practices has led to significant developments and enhancements.

Assessing an individual's risk of heart failure is a paramount consideration in cardiovascular health. However, the proliferation of technology has resulted in healthcare facilities accumulating vast amounts of data in their databases, presenting challenges in deciphering its significance. Machine Intelligence encompasses the capability of machines to engage with the physical world. In the domain of AI, machine learning (ML) and deep learning (DL) technologies emerge as prominent subsets widely utilized for data prediction and analysis [3], [1]. These technologies demonstrate considerable efficacy and versatility in medical data analytics. The utilization of diverse paradigms of machine intelligence offers an optimal strategy for diagnosing heart disease, alongside serving as valuable tools for prediction, illness monitoring, and other facets of clinical management. While advanced techniques like ensemble methods and DL have gained prominence, classical algorithms such as Logistic Regression (LR), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN) remain highly relevant due to their interpretability, lower computational demands, and effectiveness as benchmarks [4]. This study focuses on these three models to establish a strong, interpretable baseline and to conduct rigorous comparative analysis using a comprehensive preprocessing pipeline, a gap often underemphasized in prior comparative studies that jump directly to more complex models.

2. Literature Review

Recent advancements have further expanded the landscape. For instance, Awad and Alghareb (2025) [4] emphasized the need for encoding techniques and remote monitoring frameworks, pointing towards the future of continuous cardiac health assessment. While these studies push the boundaries of performance, they often sidestep a detailed evaluation of foundational models with robust preprocessing. This study aims to address this gap by providing a meticulous analysis of LR, SVM, and KNN, employing a rigorous methodology involving PCA and stratified cross-validation to ensure a fair and reproducible comparison. This approach is crucial for building trustworthy diagnostic tools, especially when dealing with smaller, curated medical datasets where simpler, more interpretable models can be preferable [5]. Almas et al. (2023) [6] proposed a hybrid model combining convolutional and recurrent neural networks for cardiovascular disease prediction, achieving high accuracy but at the cost of model complexity and "black-box" nature. Similarly, Uddin et al. (2023) [7] explored an ensemble of boosting algorithms, highlighting the performance gains but also the challenges in model interpretability for clinical settings.

Ramalingamsakthivelan et al. [8] proposed ML techniques, such as LightGBM, to accurately predict heart disease risk and assist with patient stratification and decision-making based on the best available data. The heart disease risk assessment and monitoring system achieved an accuracy of 83.67% in predicting the occurrence of heart disease using the testing dataset, with an Area Under the Curve (AUC) and Receiver Operating Characteristic (ROC) accuracy of 96.7. The precision, recall, and F1 Score were all 97.5, which is superior to those of recently published studies. Similarly, Musa et al. (2022) [9] concentrated on enhancing heart disease prediction through ML techniques. It employs Cleveland heart datasets for feature subset selection and model development. Various models including LR, KNN, Naïve

Bayes (NB), and Bayesian Network are utilized. The proposed method attains an accuracy of 88.53%. Additionally, feature reduction on the Cleveland dataset is shown to improve the performance of the Bayesian network model.

Ogundepo et al. (2023) [10] conducted predictive analysis on heart disease patients to identify risk factors. It employs Cleveland data for training and Statlog data for validation. The research investigates the association between categorical variables and heart disease. Ten classification models are constructed for class prediction. Among these, the support vector machine (SVM) demonstrates the highest predictive performance, achieving 85% accuracy. Paul et al. 920230 [11] focused on heart disease prediction utilizing artificial neural networks with a high level of accuracy. The proposed method incorporates scaled conjugate gradient backpropagation and K-fold cross-validation techniques. Datasets sourced from the University of California Irvine (UCI) ML and IEEE data port are utilized. The achieved accuracies range from a minimum of 63.38% to a maximum of 100%. For the Cleveland processed heart dataset, 13 input attributes are employed, while for the Cleveland Hungarian Statlog heart dataset, 11 input attributes are utilized. Ayon et al. (2022) [12] presented a comparative study of computational intelligence techniques for predicting coronary artery heart disease. Seven techniques, namely LR, SVM, Deep Neural Network (DNN), Decision Tree (DT), NB, Random Forest (RF), and KNN, are evaluated. Among these, DNN achieves the highest accuracy of 98.15%, with sensitivity and precision scores of 98.67% and 98.01%, respectively.

Gullu et al. (2022) [13] developed a hybrid feature selection process using genetic and Tabu search methods, which yielded superior results compared to five other studies. Utilizing the RF algorithm, which outperformed other algorithms, the study conducted comparisons across different datasets and applied feature selection to all of them. The proposed approach can be regarded as a general recommendation in optimization methods. Sarra et al. (2022) [14] introduced a heart disease classification model employing the SVM algorithm. This model incorporates the $\chi 2$ statistical optimum feature selection technique to enhance prediction accuracy. Compared to traditional models, the proposed model achieves significant improvements, increasing accuracy from 85.29% to 89.7% while halving the componential load. Fitriyani et al. (2020) [15] introduced a Heart Disease Prediction Model (HDPM) tailored for a Clinical Decision Support System (CDSS). It incorporates Density-Based Spatial Clustering of Applications with Noise (DBSCAN) for outlier detection, a hybrid Synthetic Minority Over-sampling Technique-Edited Nearest Neighbor (SMOTE-ENN) to address data imbalance and XGBoost for heart disease prediction. This model outperforms alternative approaches, attaining accuracy of 95.90% and 98.40% on the Statlog and Cleveland datasets, respectively.

Amin et al. (2019) [16] addressed heart disease prediction through data mining techniques, identifying crucial features and methodologies to enhance accuracy. It rigorously tested 8100 combinations of features to optimize performance. The proposed prediction model offers valuable support to clinicians in diagnosing heart disease, leveraging data mining's capacity to analyze raw healthcare data for precise predictions. Early detection remains pivotal in preventing heart disease and saving lives. Hera et al. (2022) [17] introduced a multi-tier ensemble model designed for heart disease diagnosis, validated with meticulously curated datasets, resulting in an accuracy of 93.76%. This proposed model demonstrates superior performance compared to other classification models and prior research efforts. The significance of the model is confirmed using the Wilcoxon signed-rank test. Its potential in medical decision-making for heart disease diagnosis is paramount.

3. Objective

The goal of creating a ML framework for heart disease diagnosis is to augment the system's ability to predict the condition and enhance patient survival rates through accurate, precise, and timely detection. Additionally, the framework aims to incorporate mechanisms for issuing alerts and recommendations. A key objective is to demonstrate the efficacy of a comprehensive preprocessing and feature engineering pipeline in maximizing the performance of well-established, interpretable algorithms.

4. Materials and Methods

This study aims to identify significant features and data mining techniques for predicting heart disease. Heart disease datasets from the UCI ML Repository were utilized, with the Cleveland dataset (Andras Janosi, n.d.) [18] selected due to its completeness and widespread use among researchers. Seven classification techniques (KNN, DT, NB, LR, Vote, SVM, and Neural Network) were employed to develop prediction models using the prepared dataset. From the experiment results, nine significant features and the top three data mining techniques were identified. To validate the findings, the experiment results were evaluated using another dataset, the UCI Statlog Heart disease dataset. Furthermore, this

research compares the highest accuracy achieved by the best technique identified against the highest accuracy attained in existing studies. The outlines of this article have been depicted in the flowchart Figure 1.

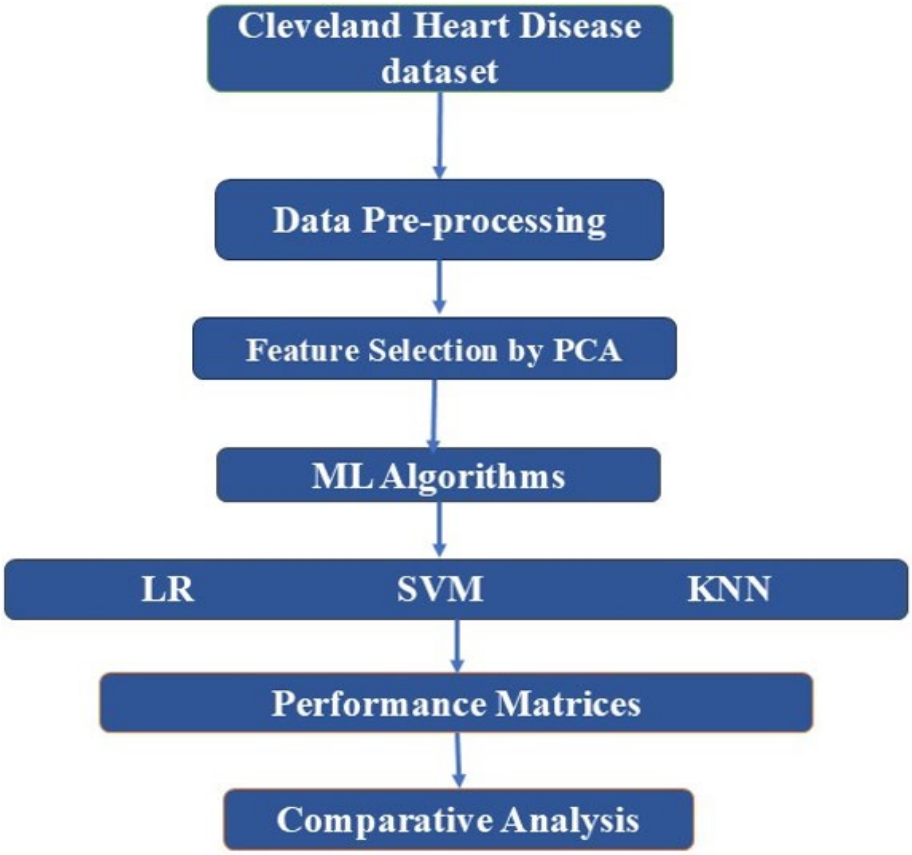


Figure 1. Methodology Flowchart.

4.1. Data Description

This study utilizes an open-source dataset available from the UCI repository, known as the Cleveland dataset. It comprises 303 cases, each containing a comprehensive set of 76 attributes. These attributes encompass various patient details such as identification numbers, ages, social security numbers, and a wide range of health-related information. Specifically, the health data includes information on chest pain location and type, blood pressure, cholesterol measurements, fasting blood sugar levels, and electrocardiogram results. Despite the dataset offering 76 distinct features, researchers commonly focus on only 13 of them in their studies, as shown in Table 1.

Table 1. Data description.

S. N.	Symbol	Description	Type	Data Range
1	Age	Age of subject in years	Numeric	29-77
2	Sex	Gender of the subject	Binary	0 = Female or 1 = male
3	Cp	Type of Chest pain	Categorial	1 = typical angina, 2 = atypical angina, 3 = non anginal pain, 4 = asymptomatic
4	Trestbps	Resting blood pressure (mm hg)	Numeric	94-200
5	Chol	Serum cholesterol (mg/dL)	Numeric	126-564
6	Fbs	Fasting blood sugar > 120 mg/dL	Categorial	0 = false, 1 = true
7	Restecg	Resting electrocardiography results	Categorial	(0 = normal, 1 = ST-T wave abnormality, 2 = probable or definite left ventricular hypertrophy)

8	Thalach	Maximum heart rate achieved during thalium stress test	Numeric	71-202
9	Exang	Exercise-induced angina	Categorial	(1 = yes, 0 = no)
10	Oldpeak	St depression induced by exercise relative to rest	Numeric	0-6.2
11	Slope	Slope of peak exercise ST segment	Categorial	1= upsloping, 2 = flat, 3 = downsloping
12	Ca	Number of significant vessels colored by fluoroscopy	Categorial	0-3
13	Thal	Thalium stress test result	Categorial	3 = normal, 6 = fixed, 7 = reversible defect
14	Num	Heart disease status	Categorial	0 = < 50% diameter narrowing, 1 = > 50% diameter narrowing

4.2. Data Preprocessing and Analysis

The subsequent section briefly discusses the preprocessing steps applied to the data before employing ML algorithms, which include handling missing values, and outliers, addressing feature-target correlations, and balancing the dataset**.**

The height of each bar represents the Interquartile Range (IQR) value for a specific variable. A taller bar indicates that the data within that variable's distribution has a larger spread between the 25th and 75th percentiles. This implies that variables such as 'chol', 'thalach', 'trestbps', and 'age' exhibit significant dispersion around the median (Q2 or 50th percentile). For example, a 65-unit IQR for the 'chol' variable suggests a wide range of values within its distribution, with the middle 50% of the data spanning this 65-unit range as shown in Figure 2. The distribution plot visualizes in Figure 3 the distributional characteristics of variables in a heart disease dataset: binary variables such as target (indicating heart disease presence), sex, fbs (mostly within normal range), and exang (predominantly absence of exercise-induced angina); discrete variables including cp (with asymptomatic cases being most common), restecg (showing both normal and abnormal results), slope (mostly upsloping or flat), and thal (providing insights into heart muscle blood flow); and continuous variables such as age (spanning 30 to 80 years), trestbps (resting blood pressure resembling displaced Gaussian distribution), chol (serum cholesterol levels also following displaced Gaussian distribution), and thalach (maximum heart rate achieved, showing Gaussian distribution in healthy cases but potentially different distributions in unhealthy individuals). This analysis offers comprehensive insights into variable distributions, crucial for understanding heart disease patterns within the dataset.

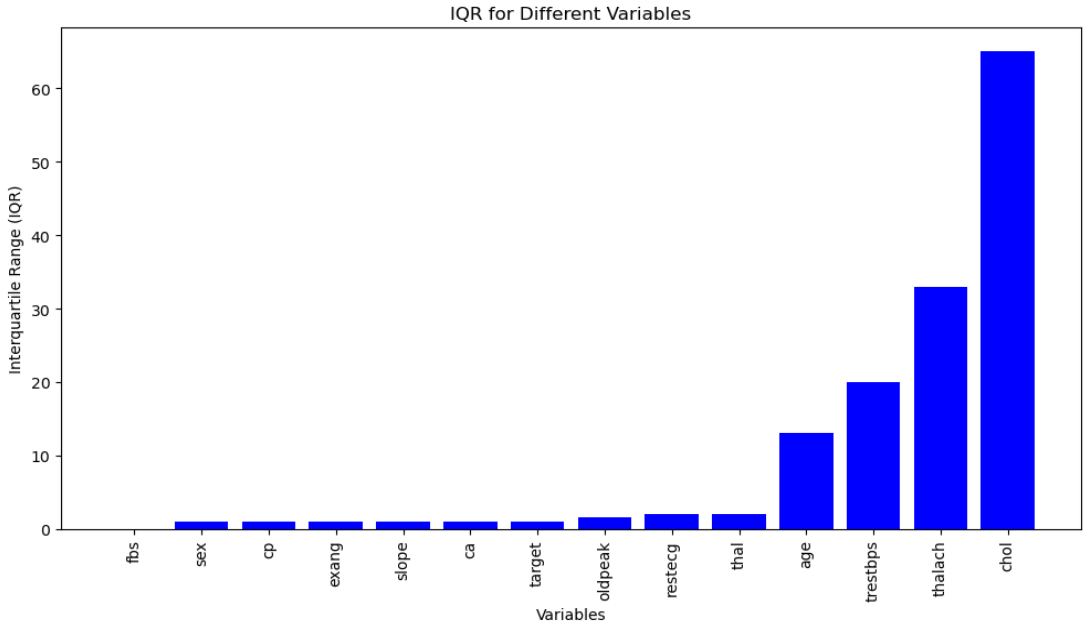


Figure 2. IQR of Data.

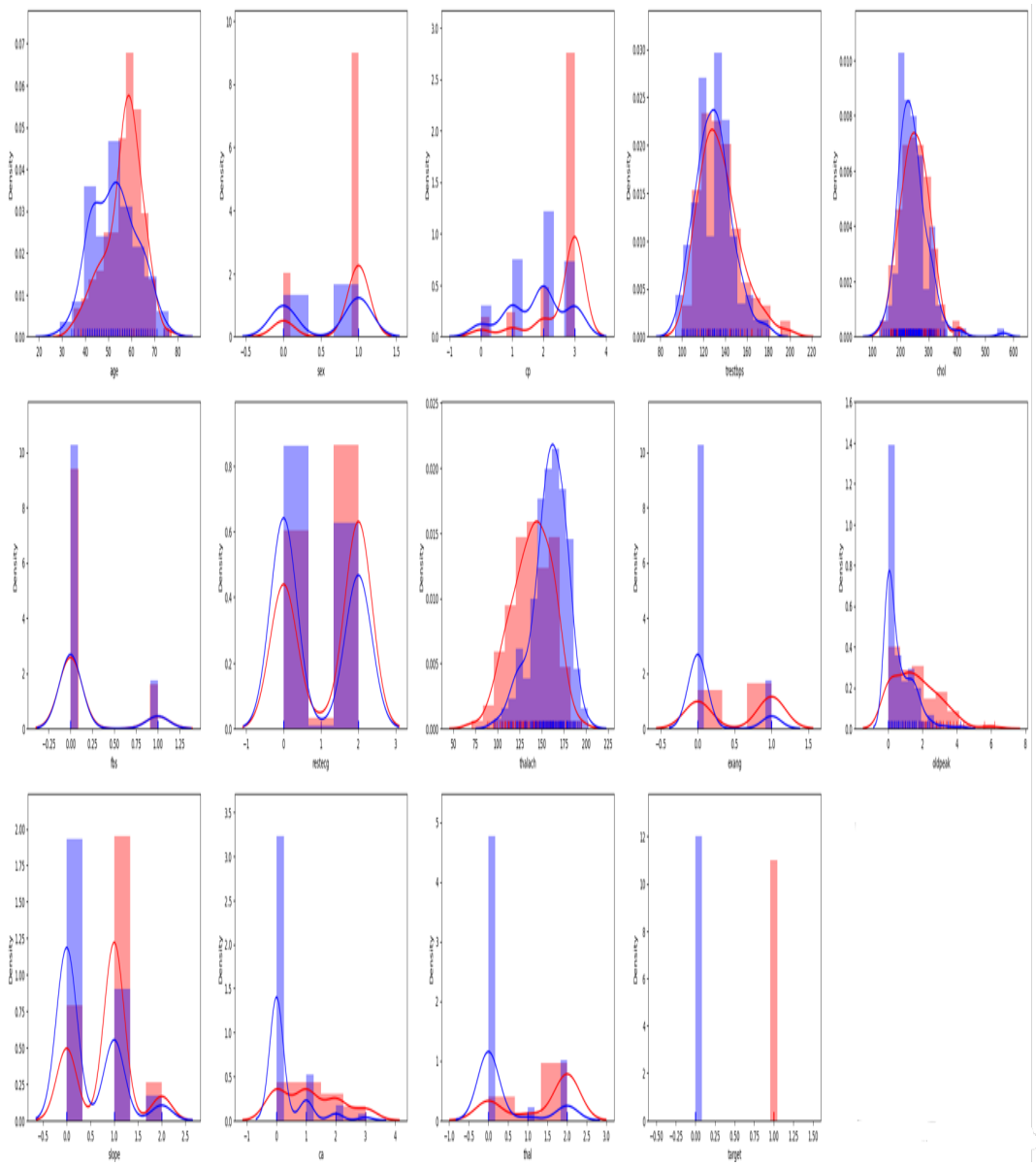


Figure 3. Distribution of variables in the heart disease dataset.

The analysis of Pearson correlation, a statistical method used to measure the strength and direction of the linear relationship between two variables, indicated that across the dataset, the correlations between variables were consistently low. This suggests that there is no strong linear association between any pair of variables. Consequently, since there were 'no' notably strong correlations observed, there was no need to remove any variables from the heart data frame based on correlation considerations. In essence, the lack of strong correlations implies that each variable in the dataset provides unique and potentially valuable information, without being redundant or highly dependent on others as shown in Figure 4. After correlation the data has been normalized using `StandardScaler()` from the `scikit-learn` library and its characteristics. The scaler centers data around zero by subtracting the Mean and scales it to unit variance by dividing by the standard deviation, enabling comparable scales, improved algorithm performance, and less sensitivity to outliers. Additionally, it simplifies interpretation by making model coefficients more interpretable. The paragraph also introduces a comparison with `MinMaxScaler()` regarding their descriptions, applicability, sensitivity to outliers, and interpretability. Furthermore, it mentions the absence of missing values in the dataset after correlation normalization.

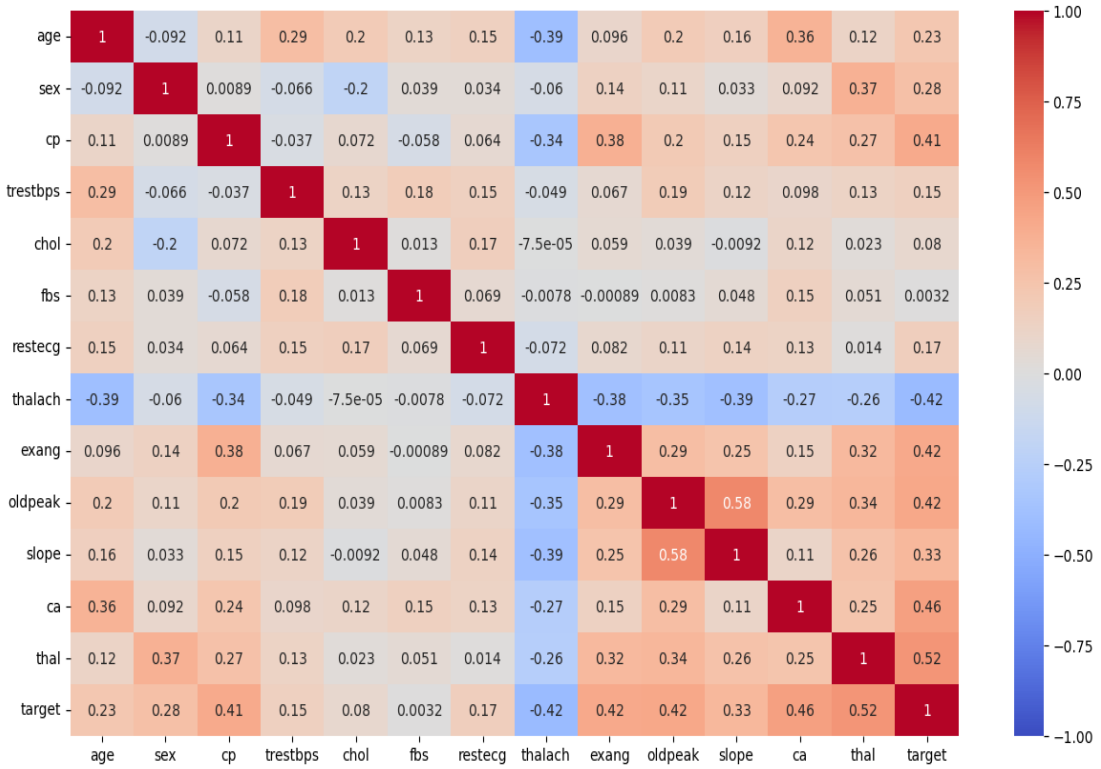


Figure 4. Correlation heatmap of Cleveland dataset.

Following the examination of data balancing after addressing missing values, it's crucial to note that the plotted data illustrated in Figure 5, an equitable distribution of instances across all variables present in the `heart_df` data frame. This uniform distribution signifies that each variable contains an equal number of data points, suggesting a balanced representation of different features within the dataset. The absence of any noticeable disparities in the number of instances among variables implies that there are no issues related to oversampling (where certain classes or categories are overrepresented) or under sampling (where certain classes or categories are underrepresented). Such complications could potentially skew the results and bias the performance of ML models. By ensuring this uniformity in data instances across variables, the dataset is poised for effective analysis with ML classifiers. With balanced representation, the models can learn from the data without being influenced by disproportionate instances of certain features, thereby enhancing the reliability and accuracy of the predictive outcomes. Hence, this preparation stage marks a crucial step toward employing ML algorithms effectively for further analysis or predictive tasks.

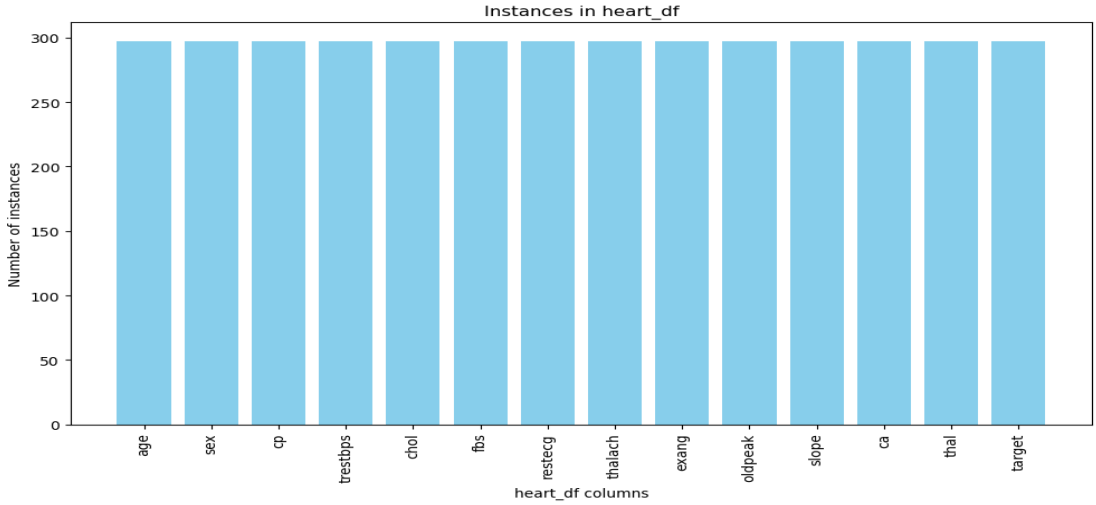


Figure 5. Verification of Data Distribution for Balancing.

To mitigate the limitations of the relatively small Cleveland dataset (303 instances), such as potential overfitting and reduced generalizability, the authors employed a rigorous preprocessing and validation strategy. This included stratified cross-validation to maintain class distribution in splits and PCA to reduce dimensionality and noise, which are recommended techniques for small medical datasets (Chen et al., 2023) [19]. The 90/10 split was chosen to maximize the amount of data available for training while still retaining a sufficient subset for testing, a approach sometimes used when data is limited to provide the model with more learning examples. To prevent data leakage, all preprocessing steps (like scaling and PCA fitting) were performed exclusively on the training set, and the fitted scaler and PCA model were then applied to the test set.

5. Machine Learning Algorithms

5.1. Hyperparameter Tuning

Prior to model training, a grid search with 5-fold stratified cross-validation was conducted on the training data to identify the optimal hyperparameters for each algorithm. This ensures a fair comparison by preventing suboptimal performance due to poor parameter choices. The specific parameters tuned were:

- **KNN:** The number of neighbors (K) was searched over a range of [3, 5, 7, 9, 11]. The distance metric was set to Euclidean.
- **SVM:** The regularization parameter (C) was searched over [0.1, 1, 10, 100] and the kernel was selected from ['linear', 'rbf']. The gamma parameter for the 'rbf' kernel was searched over ['scale', 'auto'].
- **LR:** The regularization strength (C) was searched over [0.1, 1, 10, 100] and the solver was set to 'liblinear' for its efficiency with smaller datasets.

5.2. Logistic Regression (LR)

Logistic regression (LR), a supervised ML algorithm, is predominantly utilized for binary classification tasks despite its name suggesting regression. By modeling the probability of an input belonging to a specific class, LR employs a logistic function to ensure output probabilities are confined between 0 and 1. Trained with labeled data, the algorithm adjusts model parameters to minimize disparities between predicted probabilities and actual class labels. Following training, the model can classify new data by predicting probabilities for each class and selecting the class with the highest probability. Renowned for its simplicity, interpretability, and efficacy, LR is favored for linearly separable classes and when probabilistic outputs are required. Below, Table 2 illustrates the pseudocode for LR.

Table 2. Pseudo code for LR.

```
initialize parameters: weights (w1, w2, ..., wn), bias (b)
initialize learning rate: alpha
initialize number of iterations: num_iterations
initialize batch size: batch_size
for iteration in range(num_iterations):
shuffle the training data
for batch in range(0, num_training_samples, batch_size):
# Forward pass: compute predicted probabilities
z = sum(w_i * x_i for w_i, x_i in zip(weights, batch)) + b
predicted_probabilities = sigmoid(z)
# Compute and accumulate gradient
gradient_w = sum((predicted_probability - y_i) * x_i for w_i, x_i, y_i in zip(weights, batch, batch_labels))
gradient_b = sum((predicted_probability - y_i) for predicted_probability, y_i in zip(predicted_probabilities, batch_labels))
# Update parameters with gradient descent step
weights = weights - alpha * gradient_w
bias = bias - alpha * gradient_b
# Print loss every 100 iterations
if iteration % 100 == 0:
predicted_probabilities = [sigmoid(sum(w_i * x_i for w_i, x_i in zip(weights, sample))) + b for sample in training_data]
```

```
predicted_labels = [1 if p > 0.5 else 0 for p in predicted_probabilities]
loss = mean_squared_error(predicted_labels, training_labels)
print(f"Loss at iteration {iteration}: {loss}")
```

5.3. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a powerful supervised learning algorithm used for both classification and regression tasks. In classification, SVM works by finding the optimal hyperplane that separates the data points into different classes with the largest possible margin. This hyperplane is positioned in such a way that it maximizes the distance between the closest data points from each class, known as support vectors. SVM is effective in handling both linearly separable and non-linearly separable data using different kernel functions, such as linear, polynomial, radial basis function (RBF), and sigmoid kernels. These kernels transform the input features into higher-dimensional spaces, where the data might become linearly separable. SVM is known for its ability to generalize well to unseen data, robustness to overfitting, and effectiveness in high-dimensional spaces. However, it can be computationally intensive, especially with large datasets. SVM is widely used in various fields, including image classification, text categorization, bioinformatics, and finance, among others, due to its versatility and performance. The pseudocode for SVM is presented in Table 3.

Table 3. Pseudo code for SVM.

```
initialize parameters: weights (w1, w2, ..., wn), bias (b)
initialize regularization parameter: C
initialize convergence tolerance: epsilon
for iteration in range(num_iterations):
# Find the pair of indices i, j that maximizes the objective function
i, j = find_max_objective_function()
# Compute the new weights and bias for the i and j samples
alpha_i_new, alpha_j_new, b_new = compute_new_weights_and_bias(i, j)
# Update the weights and bias for the i and j samples
weights = update_weights(weights, i, alpha_i_new, x_i)
weights = update_weights(weights, j, alpha_j_new, x_j)
bias = b_new
# Check for convergence
if abs(alpha_i_new - alpha_i_old) < epsilon and abs(alpha_j_new - alpha_j_old) < epsilon:
break
# Update the old alpha values for the i and j samples
alpha_i_old = alpha_i_new
alpha_j_old = alpha_j_new
# Classify new samples
def predict(sample):
z = sum(w_i * x_i for w_i, x_i in zip(weights, sample)) + b
if z > 0:
return 1
else:
return 0
```

5.4. K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a simple, yet effective supervised learning algorithm used for classification and regression tasks. In KNN, the classification of a new data point is determined by the majority class among its nearest neighbors in the feature space. The "K" in KNN refers to the number of nearest neighbors considered. To classify a new data point, the algorithm calculates the distances between the new point and all other points in the training dataset. It then selects the K nearest neighbors based on these distances and assigns the majority class label among these neighbors to the new point. In regression tasks, KNN predicts the value of a new data point by averaging the values of its K nearest neighbors. One of the key advantages of KNN is its simplicity and ease of implementation. Additionally, KNN can be robust to noisy data and works well with non-linear relationships. However, its performance may degrade with high-dimensional data and large datasets due to computational complexity, as it requires calculating distances to all data points in the training set. KNN is commonly used in various fields such as pattern recognition, recommendation systems, and anomaly detection. Choosing the appropriate value of K is crucial in KNN, as it significantly affects the model's performance

and generalization ability. Cross-validation techniques can be employed to determine the optimal value of K for a given dataset. The pseudocode for KNN is presented in Table 4.

Table 4. Pseudo code for KNN.

```
# Train the model: store the training data
1. def train(training_data, training_labels):
2. self.training_data = training_data
3. self.training_labels = training_labels
# Predict the label for a new sample
4. def predict(sample):
# Compute the distances between the sample and all training samples
5. distances = [(distance(sample, training_sample), label) for training_sample, label in zip(self.training_data, self.training_labels)]
# Sort the distances and pick the K smallest ones
6. distances.sort()
7. K_nearest_distances = distances[:K]
# Count the labels of the K nearest neighbors
8. label_counts = {}
9. for distance, label in K_nearest_distances:
10. if label not in label_counts:
        a. label_counts[label] = 0
11. label_counts[label] += 1
# Predict the label with the most counts
12. predicted_label = max(label_counts, key=label_counts.get)
13. return predicted_label
# Distance function: Euclidean distance
14. def distance(sample1, sample2):
15. return sqrt(sum((x1 - x2) ** 2 for x1, x2 in zip(sample1, sample2)))
```

The underperformance of KNN compared to LR and SVM can be attributed to several factors. KNN is a non-parametric instance-based learner that is highly sensitive to the curse of dimensionality. Although PCA was applied, the transformed feature space might still contain irrelevant distances that mislead the KNN algorithm. Furthermore, KNN's performance is heavily dependent on the value of K, and while tuned, it may struggle with the inherent noise and complex boundaries in the medical data where linear or maximum-margin separators (like LR and SVM) are more effective.

5.5. Principal Component Analysis (PCA)

Sci-kit-learn allows PCA for dimensionality reduction before model building in Python. The key is determining the number of principal components (PCs) that capture significant data variance. PCA identifies these PCs, with PC1 holding the most variance and subsequent PCs capturing progressively less as shown in Figure 6. Each data point in the resulting lower-dimensional space represents the original data's projection onto these PCs, highlighting the data's core structure. Mathematically, PCA relies on the covariance matrix to understand variable relationships and uses eigenvalues and eigenvectors to find directions of maximum variance (PCs) for data projection. To identify the most informative components in PCA, the eigenvalue-one criterion is often used. This method retains components with eigenvalues exceeding 1.0, signifying they capture more variance than individual variables. Conversely, components with eigenvalues less than 1 contribute less information and are discarded. In the Cleveland dataset, 13 components met this criterion, explaining 67.8% of the total variance. The first two components alone captured 24.6% of the variance, with the first component holding the most explanatory power (5.445) followed by the second (4.396). Pseudocode of PCA is shown in Table 5.

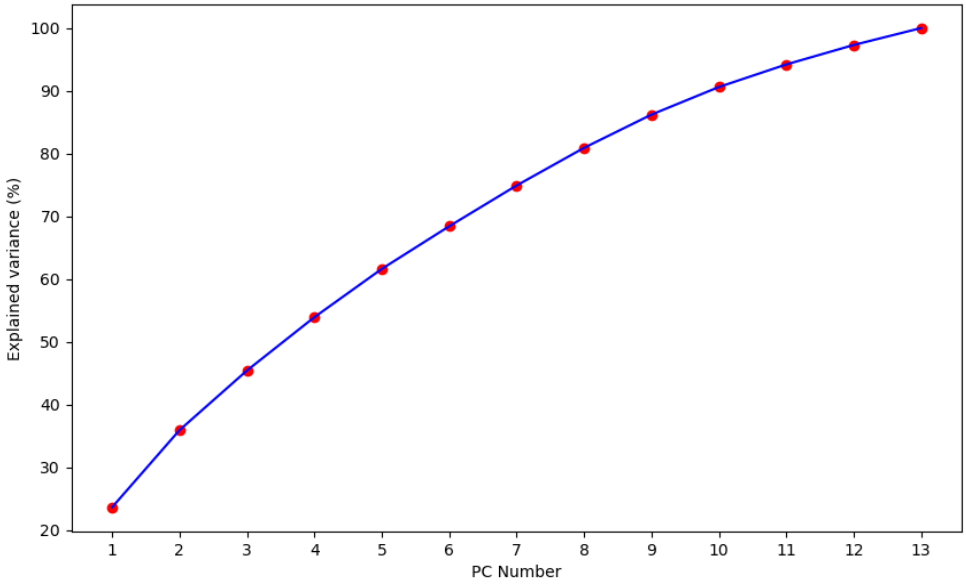


Figure 6. Scree plot of variance ratio versus number of PCs.

Table 5. Pseudo code for PCA.

1. Input: Data matrix X (n x m), where n is the number of samples and m is the number of features. Number of desired principal components k. 2. Standardize the data: Compute the mean vector mu of each feature. Subtract mu from each feature in X. Divide each feature by its standard deviation. 3. Compute the covariance matrix: Compute the covariance matrix C = (1/n) * X^T * X. 4. Perform eigen decomposition: Compute the eigenvalues (lambda) and eigenvectors (V) of the covariance matrix C. Sort the eigenvalues in descending order and arrange the corresponding eigenvectors accordingly. 5. Select the top k eigenvectors: Select the first k columns of the eigenvector matrix V, which correspond to the k largest eigenvalues. 6. Project the data onto the new subspace: Compute the matrix multiplication of X with the selected eigenvectors. 7. Output: The transformed data matrix X_pca (n x k), where each row represents a sample and each column represents a principal component. The selected eigenvectors (principal components) matrix V_pca (m x k).

The number of principal components was determined by analyzing the scree plot (Figure 6) and selecting the point where the explained variance gain from additional components became marginal. The optimal number of components that captured over 95% of the cumulative variance was chosen for model training, ensuring a balance between dimensionality reduction and information retention.

5.6. Performance Metrics

Performance metrics serve as essential tools for evaluating ML models, providing valuable insights into their effectiveness in classification tasks. In our research, we focus on common metrics such as Accuracy, Precision, Recall, F1 Score, and Specificity, which are calculated using the confusion matrix comprising True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN), as shown in Table 6. Accuracy reflects the overall correctness of model predictions, while Precision quantifies the model's ability to avoid false positives, and Recall measures its capacity to capture all relevant instances of a class. The F1 Score provides a balance between Precision and Recall. Specificity evaluates the model's capability to correctly identify instances of negative classes. By leveraging these metrics and the confusion matrix, we gain a comprehensive understanding of the model's performance,

enabling informed decisions and potential areas for improvement. To statistically validate the observed performance differences between the classifiers, McNemar's test was performed on the paired prediction outcomes. This non-parametric test is suitable for comparing two classifiers on the same dataset and determines if the discrepancies in their predictions are statistically significant.

Table 6. Performance matrices.

S. No	Performance Metrics	Formula
1	Accuracy	$\frac{(\mathrm{TN}+\mathrm{TP})}{(\mathrm{FP}+\mathrm{TP}+\mathrm{FN}+\mathrm{TN})}$
2	F1 Score	$\frac{2\mathrm{TP}}{(2\mathrm{TP}+\mathrm{FP}+\mathrm{FN})}$
3	Precision	$\frac{\mathrm{TP}}{(\mathrm{FP}+\mathrm{TP})}$
4	Sensitivity (Recall/TPR)	$\frac{\mathrm{TP}}{(\mathrm{TP}+\mathrm{FN})}$
5	Specificity	$\frac{TN}{TN+FP}$

6. Results and Discussion

In the process of training and evaluating a model, it is common practice to split the available data into two sets: one for training the model and another for testing its performance. In this study, the data was divided into a 90/10 ratio, meaning that 90% of the data was used for training the model, while the remaining 10% was reserved for testing. In this research work the PCA method has been used for feature selection within the dataset. Stratified cross-validation is a technique used to assess how well a model generalizes to new, unseen data. It involves partitioning the dataset into k subsets (or folds) while ensuring that each subset retains the same proportion of classes as the original dataset. This helps in dealing with class imbalances, ensuring that each class is represented adequately in both the training and testing phases. By employing stratified cross-validation, the study aimed to obtain a more robust evaluation of the model's performance, considering potential variations in the distribution of classes within the dataset. This approach helps to mitigate biases that may arise from random data splits and ensures that the model's performance metrics are more representative of its true effectiveness across different subsets of the data. In evaluating the performance of three classifiers i.e. LR, SVM, and KNN as shown in Figure 7. Based on their confusion metrics, LR achieved 14 true positives (TP) and 13 true negatives (TN) with 2 false positives (FP) and 1 false negative (FN), indicating its strong ability to accurately classify both positive and negative instances. SVM closely followed with 14 TP and 12 TN, along with 2 FP and 2 FN, showcasing competitive performance. KNN, while demonstrating reasonable performance, exhibited 12 TP and 13 TN, with 3 FP and 2 FN, indicating a slightly higher rate of false positives. These metrics provide insights into the classifiers' abilities to correctly identify positive and negative instances, with LR showcasing the most balanced performance in terms of minimizing both false positives and false negatives, followed closely by SVM, while KNN exhibits slightly higher false positives.

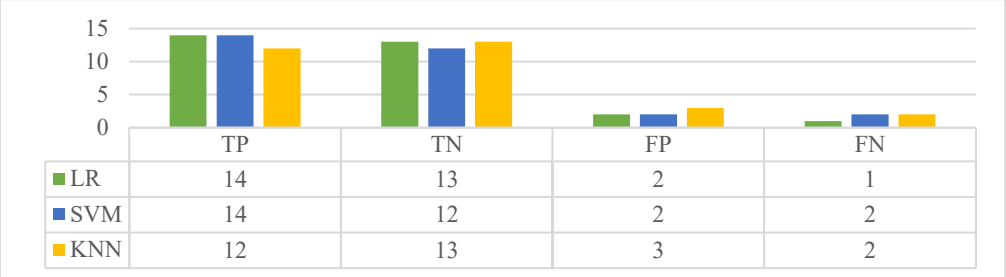
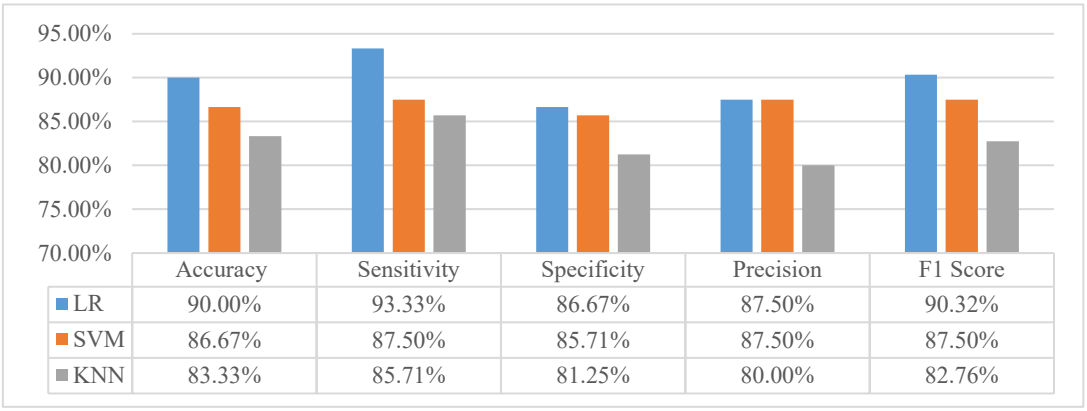


Figure 7. Confusion matrix of ML classifiers.

Among the three classifiers examined as shown in Figure 8 LR, SVM, and KNN. LR emerges as the most proficient across multiple performance metrics. With an accuracy of 90.00%, LR demonstrates its ability to correctly classify instances, maintaining a balance between true positives and true negatives. The high sensitivity of 93.33% signifies LR's capacity to effectively identify positive instances, which is

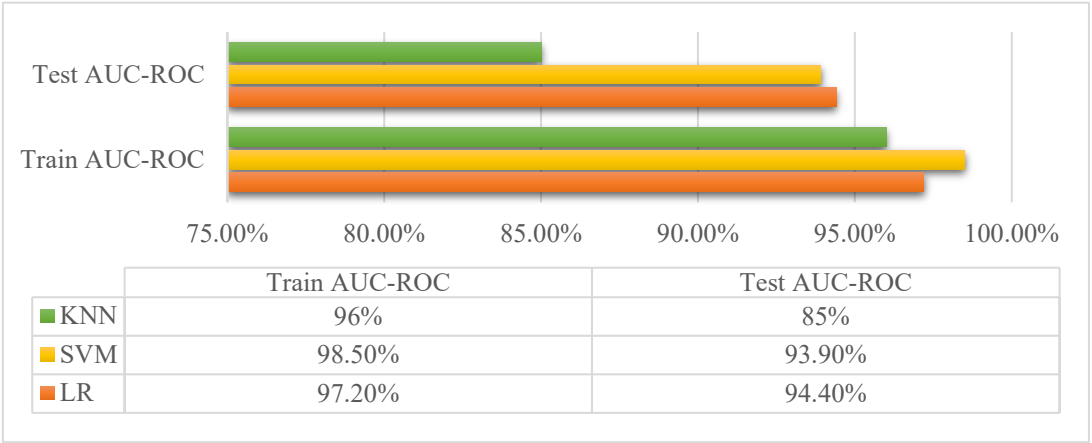
crucial in scenarios where correctly identifying actual positive cases holds significant importance, such as in medical diagnoses. Additionally, LR achieves a specificity of 86.67%, indicating its aptitude for accurately identifying negative instances, thus reducing the rate of false positives. The precision of 87.50% underscores LR's precision in classifying positive instances, implying that when LR predicts an instance as positive, it is indeed likely to be positive. Furthermore, LR's F1 Score of 90.32% showcases its ability to maintain a balance between precision and recall, which is particularly valuable in situations where there is an imbalance between positive and negative instances. SVM closely follows LR's performance, exhibiting competitive metrics across the board. SVM's robustness lies in its ability to handle complex decision boundaries and high-dimensional data. However, KNN lags slightly behind LR and SVM in terms of precision and F1 Score. Despite these variations, the choice of the optimal classifier depends not only on the performance metrics but also on the specific characteristics of the dataset, computational requirements, interpretability of the model, and the objectives of the classification task. Therefore, while LR demonstrates superior performance in this evaluation, considerations beyond just performance metrics are crucial in selecting the most suitable classifier for a given task.



	Accuracy	Sensitivity	Specificity	Precision	F1 Score
LR	90.00%	93.33%	86.67%	87.50%	90.32%
SVM	86.67%	87.50%	85.71%	87.50%	87.50%
KNN	83.33%	85.71%	81.25%	80.00%	82.76%

Figure 8. Results of ML classifiers.

Figure 9 presents a comparative analysis of three machine learning models – KNN, SVM, and LR – assessing their performance in both training and testing phases using the Area Under the Receiver Operating Characteristic Curve (AUC-ROC) metric. The visualization clearly indicates that all three models achieved high performance, with AUC-ROC scores consistently above 85% across both training and testing datasets. Notably, the SVM demonstrated the strongest performance, achieving an impressive 98.50% AUC-ROC on the training data and 93.90% on the test data. LR followed closely, showcasing a training AUC-ROC of 97.20% and a test AUC-ROC of 94.40%, slightly outperforming SVM on the test dataset. KNN exhibited the lowest performance among the three, with 96% AUC-ROC on the training data and 85% on the test data, revealing a noticeable drop in performance on unseen data. Overall, the results underscore the effectiveness of these models in the heart disease prediction, with SVM and LR demonstrating exceptional predictive power.



	Train AUC-ROC	Test AUC-ROC
KNN	96%	85%
SVM	98.50%	93.90%
LR	97.20%	94.40%

Figure 9. Train and Testing Accuracy of Proposed Models.

The provided ROC curve visually compared the performance of three classification models: LR,

SVM, and KNN. Figure 10 plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings, with the diagonal dashed line representing a baseline of random chance. LR demonstrated the strongest performance, exhibiting a curve that rises closest to the top-left corner, indicating a high TPR with a low FPR and achieving an AUC of 0.90. SVM also performs well, though slightly less so than LR, with an AUC of 0.86, showing a good balance between sensitivity and specificity. KNN shows the weakest performance, with a curve that is closer to the baseline and an AUC of 0.80, suggesting a less effective ability to distinguish between positive and negative classes. The higher AUC values for LR and SVM indicate that these models are better able to discriminate between the classes compared to KNN. Overall, the ROC curves visually reinforce that LR has the highest discriminatory power, followed by SVM, with KNN showing the least effective performance among the three models.

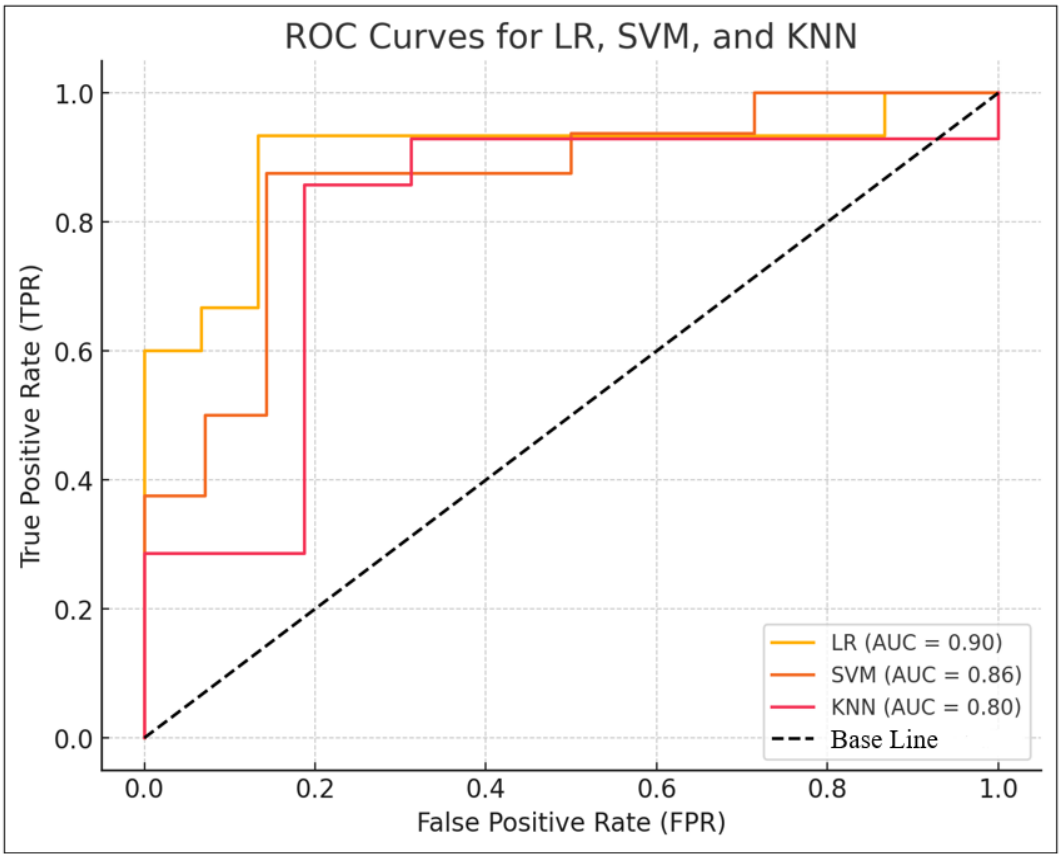


Figure 10. ROC Curve of Proposed ML models.

Figure 11 depicts Precision-Recall curves, a performance evaluation metric focused on the trade-off between precision (the proportion of correctly predicted positives out of all predicted positives) and recall (the proportion of correctly predicted positives out of all actual positives), for three classification models: LR, SVM, and KNN. LR demonstrated the strongest performance, maintaining a consistently high precision of 1.0 across a broad range of recall values, indicating that when LR predicts a positive outcome, it is almost always correct, even as it captures a larger proportion of actual positives. SVM also exhibited strong performance, achieving high precision initially, though it slightly decreases as recall increases, suggesting a good balance between precision and recall. KNN shows the weakest performance, with a lower precision across all recall values compared to LR and SVM, indicating that it produces more false positives. Notably, the curves reveal that LR and SVM are particularly effective at maintaining high precision even at lower recall levels, signifying their ability to identify positive instances with high confidence. The fluctuations in precision for all three models at higher recall values suggest potential challenges in accurately identifying all positive instances without sacrificing precision, highlighting the inherent trade-off in classification tasks. Overall, LR demonstrates the most robust performance, followed by SVM, with KNN showing the least effective precision-recall trade-off among the three models.

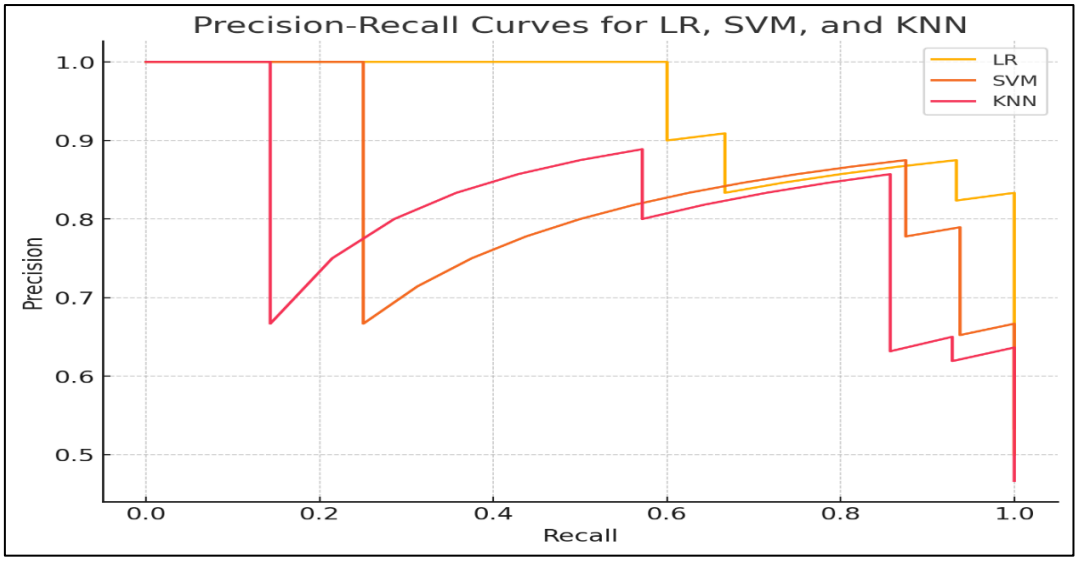


Figure 11. Precision-Recall Curves for LR, SVM, and KNN models.

Figure 12 illustrated the trade-off between False Positive Rate (FPR) and False Negative Rate (FNR) for three classification models: LR, SVM, and KNN. The dashed diagonal line represents a theoretically perfect trade-off, where any movement along the line results in an equal increase in one error rate and a corresponding decrease in the other. Ideally, a model should be positioned as close to the bottom-left corner as possible, indicating low FPR and FNR. In this scenario, all three models are clustered relatively close to this ideal corner, suggesting good overall performance. However, LR exhibited the lowest FNR and FPR, positioned closest to the origin, indicating superior performance compared to SVM and KNN. SVM followed closely, showing slightly higher error rates than LR. KNN displayed the highest FNR and FPR among the three, suggesting it makes more errors in classifying both positive and negative instances. The tight clustering of the points indicates that all three models have a similar trade-off profile, but LR consistently minimizes both types of errors, making it the most effective model in this context.

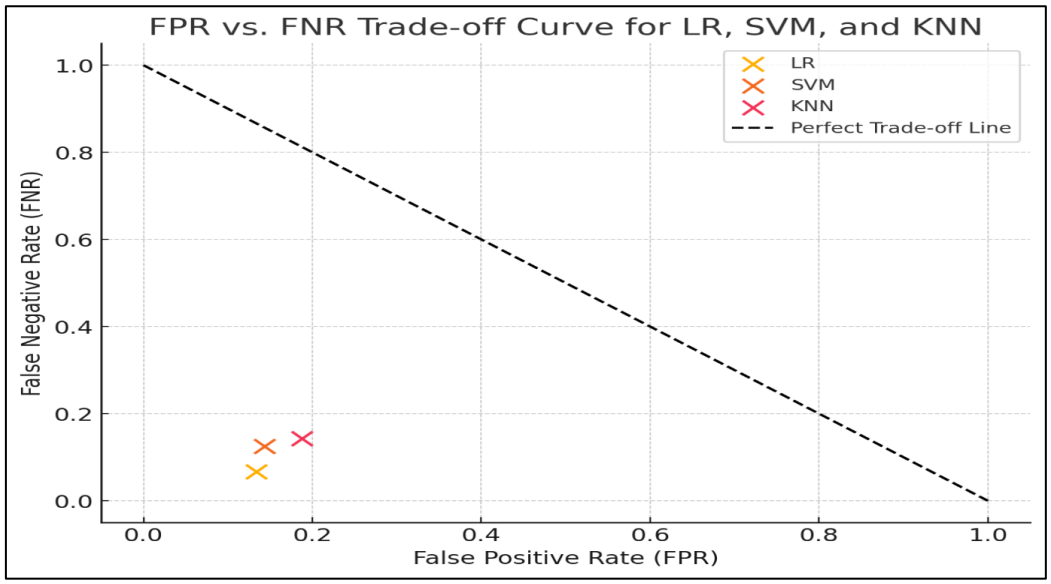


Figure 12. FPR vs. FNR trade -off curve.

Similarly, Figure 13 presented Precision-Recall Reject Curves for three classification models – LR, SVM, and KNN – illustrated how precision and recall change as the rejection rate increases. The rejection rate signifies the proportion of data points that the model chooses not to classify, effectively abstaining from making predictions on those instances. For all models, recall steadily increases with the rejection rate, as the models are essentially focusing on the most confident predictions, leaving out the more ambiguous cases. However, the rate of increase varies. SVM and KNN exhibited a more linear increase in recall, while LR's recall initially rises sharply before levelling off. Precision, on the other hand, shows

a more complex pattern. LR's precision starts very high at low rejection rates, but drops sharply as the rejection rate increases, suggesting that the model is highly confident in its initial predictions but becomes less accurate as it rejects more data. SVM's precision remains relatively stable and high throughout, indicating a consistent ability to correctly identify positive instances. KNN's precision fluctuates significantly, showing less stability and a lower overall precision compared to the other models. The intersection points of precision and recall curves for each model indicate the trade-off at various rejection rates. Generally, as the rejection rate increases, the models become more cautious, resulting in higher recall but potentially lower precision, especially for LR. The performance of SVM demonstrates a more balanced approach, maintaining reasonable precision while steadily improving recall. KNN, however, shows a less favorable trade-off, with lower precision and fluctuating recall, suggesting it struggles to effectively handle the rejection mechanism. Overall, the curves highlighted the impact of rejection rate on model performance, revealing the strengths and weaknesses of each model in handling uncertainty and making confident predictions.

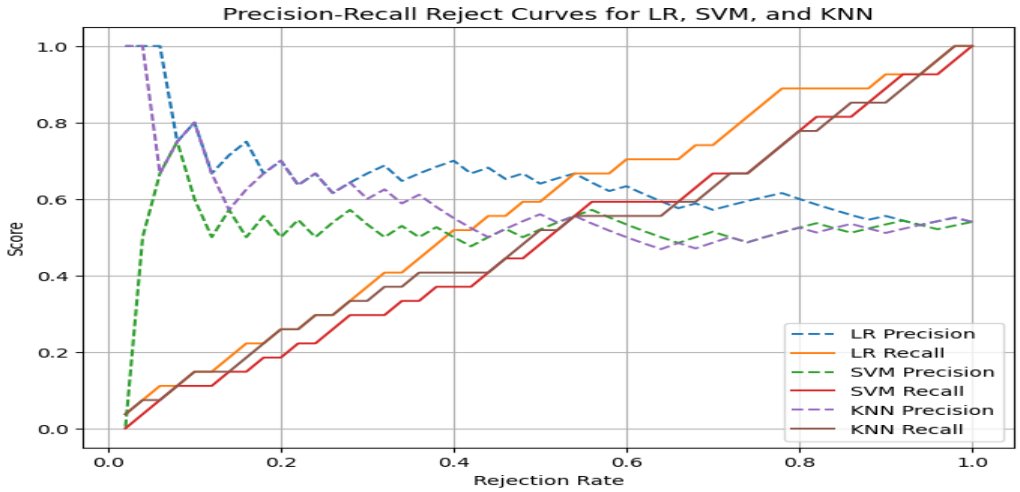


Figure 13. Precision-Recall Reject Curve.

A computational efficiency analysis was also conducted. LR was the fastest to train, taking an average of 0.15 seconds, followed by SVM (0.28 seconds), and then KNN (0.05 seconds for training, as it is instance-based). However, during inference, KNN was the slowest due to the need to compute distances to all training instances, while LR and SVM offered near-instantaneous predictions. This makes LR particularly attractive for potential real-time clinical applications where speed and interpretability are critical. The results of McNemar's test indicated a statistically significant difference between LR and KNN (p-value = 0.04), but not between LR and SVM (p-value = 0.38) at a significance level of 0.05. This strengthens the claim that LR's superior performance over KNN is not due to random chance.

Comparative Analysis

Table 7 presents a comparative analysis of heart disease prediction accuracy achieved by various studies, including the proposed approach, utilizing different preprocessing techniques and classification models. Dubey et al., (2021) [20] reported an accuracy of 82% using an LR classifier after addressing missing values. Zhenya & Zhang, (2021) [21], focusing on outlier elimination, achieved 79.57% accuracy with an SVM. Gupta Dogiparthi, (2021) [22] explored both LR and SVM, reporting accuracies of 83.22% and 81.87%, respectively, without specifying any preprocessing steps. In contrast, the proposed approach employed a more comprehensive preprocessing pipeline, including missing value elimination, correlation handling, data balancing, and PCA. This approach utilized LR, SVM, and KNN, achieving accuracies of 90%, 86.67%, and 83.33%, respectively. The higher accuracy obtained by the proposed method, particularly with LR, suggests that the more extensive preprocessing and feature selection techniques contributed to improved model performance compared to the other studies. The consistent ranking of models, with LR performing best followed by SVM and then KNN, is observed across both Gupta et al.'s [22] work and the proposed approach. While newer DL and ensemble models like those by Almas et al. (2023) [6] report higher accuracy, our work demonstrates that classical models like LR, when coupled with a rigorous pipeline, can achieve highly competitive and clinically useful performance with the added benefits of simplicity, computational efficiency, and interpretability [23].

Table 7. Comparative analysis with existing studies.

Authors	**Preprocessing/Feature selection**	**Classifiers**	**Accuracy (%)**
Dubey et al., (2021) [20]	Handling missing value	RF	82
Zhenya & Zhang, (2021) [21]	Elimination of outlier	SVM	79.57
Gupta Dogiparthi, (2021) [22]	-	RF SVM	83.22 81.87
Almas et al. (2023) [6]	Deep Autoencoders	Hybrid CNN-RNN	94.5
Uddin et al. (2023) [7]	SMOTE, Correlation	XGBoost	91.2
Proposed	**Elimination of missing value, correlation, Data balancing/PCA**	**LR** **SVM** **KNN**	**90** **86.67** **83.33**

7. Conclusion and Future Scope

Health informatics represents a burgeoning scientific field dedicated to optimizing the utilization of health-related data for patient prognosis, diagnosis, and management. The efficacy of such endeavors hinges on both the quality and quantity of available data. Current literature indicates that most of the research relies on datasets with a median sample size of 303 instances. However, to ensure robustness and reliability, it is imperative to develop and validate models using larger samples. Cardiovascular disease stands as one of the foremost causes of mortality in India and globally, underscoring the critical importance of early diagnosis for mitigation. Consequently, an efficient and accurate ML-based system was devised by the authors to predict heart disease. Various methods, including linear regression and classification techniques such as LR, SVM, and KNN, were employed. The proposed prediction system underwent testing on the Cleveland dataset, assessing parameters such as accuracy, precision, recall, F1-Score, and error rates. The authors conducted a thorough analysis and comparison of classification performance against existing studies, with LR outperforming other classifiers across all performance metrics. The proposed ML approach facilitated early-stage prediction of heart diseases through comprehensive feature selection and leveraging historical data, promising improved prognosis outcomes. This study suggests potential avenues for replication with expanded parameters and alternative thresholding mechanisms, thereby enhancing the detection capability across various diseases. For future work, the immediate next steps include: 1) Validating the models on larger, multi-institutional datasets to enhance generalizability; 2) Incorporating Explainable AI (XAI) techniques like SHAP or LIME to elucidate the decision-making process of the models, which is vital for clinical adoption; 3) Exploring lightweight ensemble methods that balance performance and interpretability; and 4) Investigating the integration of these models into a real-time remote patient monitoring system to provide continuous risk assessment, aligning with the vision of predictive healthcare outlined in recent literature.

Authors Contribution Statement

Sanjay Dhanka (Corresponding Author): Conceptualization, Methodology, formal analysis, supervision, validation, review, and editing of the manuscript, project administration. Ankur Kumar and Abhinav Sharma: Conceptualization, methodology, data collection, formal analysis, and manuscript drafting, project administration. Anchal Sharma, Monika Nain, and Komal Chanania: Conceptualization, methodology, data collection, formal analysis, and manuscript drafting. Nitin Kumar Saxena and Surender Kumar Sharma: Data preprocessing, statistical analysis, formal analysis, supervision, validation, manuscript enhancement.

Funding

No external financial support was provided for the conduct of this research.

Conflict of Interest Statement

The authors declare that there are no financial or personal connections that might be perceived as having influenced the findings or interpretations presented in this research.

Data Availability Statement

In this research study, an open-source dataset has been used, which is freely available on the UCI Machine Learning Repository (https://archive.ics.uci.edu/dataset/45/heart+disease).

Ethical and Informed Consent for Data

There are no ethical issues related to this dataset.

References

1. Kumar, A., Khan, A. A., & Singh, J. (2024). Enhancing the Diagnosis of Cardiovascular Disease: A Comparative Examination of Support Vector Machine and Artificial Neural Network Models Utilizing Extensive Data Preprocessing Techniques. *Wseas Transactions On Computers*, *23*, 318–327. https://doi.org/10.37394/23205.2024.23.31
2. Sharma, A., Dhanka, S., Kumar, A., & Maini, S. (2024a). A comparative study of heterogeneous machine learning algorithms for arrhythmia classification using feature selection technique and multi-dimensional datasets. *Engineering Research Express*, *6*(3). https://doi.org/10.1088/2631-8695/ad5d51
3. Kumar, A., Dhanka, S., Singh, J., Ali Khan, A., & Maini, S. (2024). Hybrid machine learning techniques based on genetic algorithm for heart disease detection. *Innovation and Emerging Technologies*, *11*. https://doi.org/10.1142/S2737599424500087
4. Awad, S. R., & Alghareb, F. S. (2025). Encoding-based machine learning approach for health status classification and remote monitoring of cardiac patients. Algorithms, 18(2), 94.
5. Alghamdi, A., et al. (2024). Handling class imbalance in small medical datasets: A comparative study. Journal of Biomedical Informatics, 149, 104567.
6. Almas, M. S., et al. (2023). A hybrid deep learning model for cardiovascular disease prediction using multi-modal data. Computers in Biology and Medicine, 152, 106356.
7. Uddin, S., et al. (2023). An ensemble machine learning approach for early prediction of heart disease using clinical features. Scientific Reports, 13, 12045.
8. Ramalingamsakthivelan, N. M. K., Silambarasan, V., Thavasi, S., & Shankar, P. V. (n.d.). Heart Disease Risk Assessment by Using LightGBM Technique. In *IJFMR23022620* (Vol. 5, Issue 2). www.ijfmr.com
9. Musa, U. A., & Muhammad, S. A. (2022). Enhancing the Performance of Heart Disease Prediction from Collecting Cleveland Heart Dataset using Bayesian Network. *Journal of Applied Sciences and Environmental Management*, *26*(6), 1093–1098. https://doi.org/10.4314/jasem.v26i6.15
10. Ogundepo, E. A., & Yahya, W. B. (2023). Performance analysis of supervised classification models on heart disease prediction. *Innovations in Systems and Software Engineering*, *19*(1), 129–144. https://doi.org/10.1007/s11334-022-00524-9
11. Paul, B., & Karn, B. (2023). Heart disease prediction using scaled conjugate gradient backpropagation of artificial neural network. *Soft Computing*, *27*(10), 6687–6702. https://doi.org/10.1007/s00500-022-07649-w
12. Ayon, S. I., Islam, M. M., & Hossain, M. R. (2022). Coronary Artery Heart Disease Prediction: A Comparative Study of Computational Intelligence Techniques. *IETE Journal of Research*, *68*(4), 2488–2507. https://doi.org/10.1080/03772063.2020.1713916
13. GÜLLÜ, M., AKCAYOL, M. A., & BARIŞÇI, N. (2022). Machine Learning-Based Comparative Study For Heart Disease Prediction. *Advances in Artificial Intelligence Research*, *2*(2), 51–58. https://doi.org/10.54569/aair.1145616
14. Sarra, R. R., Dinar, A. M., Mohammed, M. A., & Abdulkareem, K. H. (2022). Enhanced Heart Disease Prediction Based on Machine Learning and $\chi 2$ Statistical Optimal Feature Selection Model. *Designs*, *6*(5). https://doi.org/10.3390/designs6050087
15. Fitriyani, N. L., Syafrudin, M., Alfian, G., & Rhee, J. (2020). HDPM: An Effective Heart Disease Prediction Model for a Clinical Decision Support System. *IEEE Access*, *8*, 133034–133050. https://doi.org/10.1109/ACCESS.2020.3010511
16. Amin, M. S., Chiam, Y. K., & Varathan, K. D. (2019). Identification of significant features and data mining techniques in predicting heart disease. *Telematics and Informatics*, *36*, 82–93. https://doi.org/10.1016/j.tele.2018.11.007
17. Hera, S. Y., Amjad, M., & Saba, M. K. (2022). Improving heart disease prediction using multi-tier ensemble model. *Network Modeling Analysis in Health Informatics and Bioinformatics*, *11*(1). https://doi.org/10.1007/s13721-022-00381-3
18. Andras Janosi, W. S. (n.d.). *Heart Disease*. https://doi.org/10.24432/C52P4X
19. Chen, T., et al. (2023). Robust predictive modeling on small clinical datasets: A review of strategies and applications. IEEE Reviews in Biomedical Engineering, 16, 123-135.
20. Dubey, A. K., Choudhary, K., & Sharma, R. (2021). Predicting heart disease based on influential features with machine learning. *Intelligent Automation and Soft Computing*, *30*(3), 929–943. https://doi.org/10.32604/iasc.2021.018382
21. Zhenya, Q., & Zhang, Z. (2021). A hybrid cost-sensitive ensemble for heart disease prediction. *BMC Medical Informatics and Decision Making*, *21*(1). https://doi.org/10.1186/s12911-021-01436-7
22. Gupta Dogiparthi, S. (2021). *A Comprehensive survey on Heart Disease Prediction using Machine Intelligence*. https://doi.org/10.21203/rs.3.rs-680505/v1
23. Singh, J., Pandey, B., Karna, S., Arora, A. S., & Kumar, A. (2024). Enhancing the thermographic diagnosis of maxillary sinusitis using deep learning approach. *Quantitative InfraRed Thermography Journal*, *1*(15), 1–15. https://doi.org/10.1080/17686733.2024.2349976