Classification and Comparison of Supervised Machine Learning Algorithms Based on UCI Heart Disease Dataset

Niladri Ghosh[1] and Arnab Ghosh[2]

[1]1st year student, Department of Computer Science, RKMVERI, Belur Math, Howrah.
[2]1st year student, Department of Computer Science, RKMVERI, Belur Math, Howrah.

Contributing authors: niladrighosh327@gmail.com; arnabgghosh23@gmail.com;

Abstract

This study explores the application of machine learning techniques for heart disease prediction using the UCI Heart Disease dataset. The dataset, comprising 920 entries with 16 attributes, underwent extensive preprocessing including handling missing values, outlier treatment, and feature engineering. Multiple classification algorithms such as K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Logistic Regression, Decision Trees, Random Forest, Naive Bayes, Gradient Boosting, and XGBoost were implemented to classify patients into heart disease risk categories. The preprocessing pipeline utilized transformations like one-hot encoding, ordinal encoding, and imputation to ensure optimal data preparation. Models were evaluated using metrics such as accuracy, precision, recall, F1 score, and ROC-AUC to identify the most effective classifier. Confusion matrices and visualizations provided insight into the performance of each approach on both training and testing datasets. Results demonstrated varying performance among the algorithms, with ensemble models showing higher accuracy and robustness. The trained models were saved as pipelines to enable deployment in a Streamlit-based application for real-time predictions. This research highlights the efficacy of machine learning in medical diagnostics, particularly for heart disease, and provides a scalable framework for implementation in clinical decision support systems.

Keywords: Supervised Machine Learning, EDA, Classification Algorithm, Auccuracy, Confusion Matrix

1 Introduction

Machine learning is one of the fastest growing areas of computer science, with far-reaching applications. It refers to the automated detection of meaningful patterns in data. Machine learning tools are concerned with endowing programs with the ability to learn and adapt. [1]

Machine Learning has become one of the mainstays of Information Technology and with that, a rather central, albeit usually hidden, part of our life. With the ever increasing amounts of data becoming available there is a good reason to believe that smart data analysis will become even more pervasive as a necessary ingredient for technological progress. There are several applications for Machine Learning (ML), the most significant of which is data mining. People are often prone to making mistakes during analyses or, possibly, when trying to establish relationships between multiple features. [2]

Data Mining and Machine Learning are Siamese twins from which several insights can be derived through proper learning algorithms. There has been tremendous progress in data mining and machine learning as a result of evolution of smart and Nano technology which brought about curiosity in finding hidden patterns in data to derive value. The fusion of statistics, machine learning, information theory, and computing has created a solid science, with a firm mathematical base, and with very powerful tools.

Machine learning algorithms are organized into a taxonomy based on the desired outcome of the algorithm. Supervised learning generates a function that maps inputs to desired outputs.

Unprecedented data generation has made machine learning techniques become sophisticated from time to time. This has called for utilization for several algorithms for both supervised and unsupervised machine learning. Supervised learning is fairly common in classification problems because the goal is often to get the computer to learn a classification system that we have created. [3]

ML is perfectly intended for accomplishing the accessibility hidden within Big Data. ML hand over's on the guarantee of extracting importance from big and distinct data sources through outlying less dependence scheduled on individual track as it is data determined and spurts at machine scale. Machine learning is fine suitable towards the intricacy of handling through dissimilar data origin and the vast range of variables as well as amount of data concerned where ML prospers on increasing datasets. The extra data supply into a ML structure, the more it be able to be trained and concern the consequences to superior value of insights. At the liberty from the confines of individual level thought and study, ML is clever to find out and show the patterns hidden in the data. [4]

One standard formulation of the supervised learning task is the classification problem: The learner is required to learn (to approximate the behavior of) a function which maps a vector into one of several classes by looking at several inputoutput examples of the function. Inductive machine learning is the process of learning a set of rules from instances (examples in a training set), or more generally speaking, creating a classifier that can be used to generalize from new instances. The process of applying supervised ML to a real-world problem is described in Figure 1.

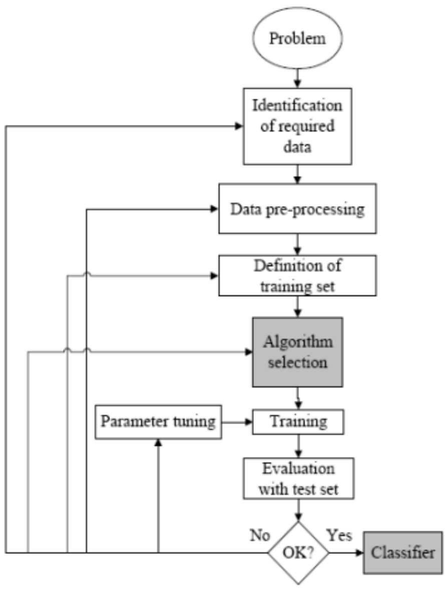


Fig. 1: The Processes of Supervised Machine Learning

This work focuses on the classification of ML algorithms and determining the most efficient algorithm with highest accuracy and precision. As well as establishing the performance of different algorithms on large and smaller data sets with a view classify them correctly and give insight on how to build supervised machine learning models.

The remaining part of this work is arranged as follows: Section 2 presents the literature review discussing classification of different supervised

2 Literature Review

According to Ayodele, Taiwo Oladipupo et al. [3], supervised machine learning algorithms that deal more with classification include the following: Linear Classifiers, Logistic Regression, Naïve Bayes Classifier, Perceptron, Support Vector Machine; Quadratic Classifiers, K-Means Clustering, Boosting, Decision Tree, Random Forest (RF); Neural networks, Bayesian Networks, and so on.

The K Nearest Neighbor (kNN) method Zhang et al. [5] has been widely used in data mining and machine learning applications due to its simple implementation and distinguished performance. However, setting all test data with the same k value has been proven impractical for real applications. Zhang et al. [6] proposed learning a correlation matrix to assign different k values for different test data points, referred to as the Correlation Matrix kNN (CM-kNN) classification. They utilized a least-squares loss function, a graph Laplacian regularizer, and l1-norm/l2,1-norm regularizers to enhance efficiency. Experiments demonstrated its superior performance in classification, regression, and missing data imputation.

These are among the most recent supervised machine learning techniques Vapnik et al. [7]. SVM models revolve around the concept of maximizing the margin on either side of a hyperplane separating two data classes. Kotsiantis et al. [2] demonstrated that this approach reduces the expected generalization error effectively.

Logistic regression, as described by Newsom et al. [8], builds classification models using a multinomial logistic regression approach. It predicts class probabilities, making it suitable for detailed and reliable predictions. Osisanwo et al. [9] highlighted its widespread use in applied statistics and discrete data analysis, noting its robustness and simplicity.

Gradient Boosting is an ensemble learning algorithm used for classification tasks. Monika et al. [10] described how it iteratively minimizes a loss function to improve model accuracy. Liew et al. [11] emphasized its flexibility, robustness, and high performance, particularly with imbalanced datasets, while highlighting the importance of hyperparameter tuning.

Decision Trees classify instances by sorting them based on feature values. Ayodele et al. [2] and Hastie et al. [12] discussed their use in data mining and machine learning, noting that decision trees employ pruning techniques for improved accuracy.

Random Forest is an ensemble learning algorithm known for its accuracy and stability. Rigatti et al. [13] described its method of building multiple decision trees using bagging and feature randomness to reduce overfitting. It provides insights into feature importance and generalizes well across various datasets.

Gaussian Naive Bayes is a simple yet effective classifier. Isidore Jacob et al. [14] and Sebestyen et al. [15] highlighted its independence assumption and minimal storage requirements. Despite being less accurate in some scenarios, **(author?)** et al. [16] found it superior to other methods on benchmark datasets. Its robustness to missing values and ability to work incrementally were further discussed by Domingos et al. [17].

XGBClassifier, based on XGBoost, offers efficient and scalable gradient boosting. Chang et al. [18] detailed its features, including L1/L2 regularization, parallel processing, and missing value handling. Chang et al. [19] emphasized its effectiveness in predictive analytics, particularly in large-scale datasets.

Supervised machine learning techniques are applicable across numerous domains. A number of ML application-oriented studies can be found in Setiono et al. [20] and Witten et al. [21].

Generally, Decision Tree and Random Forest perform better with multidimensional and continuous features, while logic-based systems excel with discrete features. Isidore Jacob et al. [14] noted Naive Bayes's efficiency with minimal storage space and its robustness to missing values. Vapnik et al. [7] found that SVMs excel with non-linear relationships and multicollinearity. Meanwhile, Zhang et al. [5] highlighted k-NN's sensitivity to irrelevant features, which can impact efficiency.

No single learning algorithm consistently outperforms others across all datasets, as noted by various studies, emphasizing the need for context-specific selection.

3 Dataset Description

The UCI Heart Disease dataset [22] is a well-known collection of data used for predicting the presence of heart disease in patients. It contains 14 attributes, including demographic information, clinical measurements, and diagnostic results, such as age, sex, chest pain type, resting blood pressure, cholesterol levels, and maximum heart rate achieved. The dataset is used to train machine learning models for classification tasks, where the goal is to predict whether a patient has heart disease based on these features. The dataset is frequently used for benchmarking various algorithms and understanding the relationships between different cardiovascular risk factors [22].

3.1 Age

Description: Age is the patient's age in years, one of the most critical factors for cardiovascular diseases.
Key insights:

- Age-related arterial stiffening leads to hypertension.
- Risk increases sharply after 45 years in men and 55 years in women.
- Younger patients with heart issues often have hereditary or congenital conditions.

Graph Analysis: The distribution is nearly normal, centered around 54 years. The slight negative skew indicates a higher concentration of older individuals. This implies that middle-aged and elderly individuals dominate the dataset, aligning with the expected demographic for heart disease prevalence. The alignment of the mean, median, and mode emphasizes the symmetric nature of the data.

3.2 Sex

Description: Sex is defined as the biological sex of the patient (Male/Female). Gender differences influence heart disease risk.
Key insights:

- Men are at higher risk before age 50 due to higher LDL levels and lifestyle factors.
- Post-menopause, women's risk increases due to estrogen decline.
- Symptoms like atypical pain in women often delay diagnosis.

Graph Analysis: The dataset is heavily male-dominated, with males constituting about 79% of the samples. This imbalance highlights a potential bias in the dataset, and models trained on it might not generalize well for females. Additionally, this distribution might reflect real-world trends, as men are often at higher risk for certain types of heart diseases.

3.3 Chest Pain Type (CP)

Description: Chest pain type describes the nature of chest pain: typical angina, atypical angina, non-anginal, or asymptomatic. It is a crucial diagnostic feature.
Key insights:

- Typical angina indicates reduced blood flow due to coronary artery disease.
- Atypical or non-anginal pain might point to non-cardiac causes.
- Asymptomatic patients are concerning due to silent ischemia risks.

Graph Analysis: Over half of the cases are asymptomatic (54%), which is a common scenario in heart diseases. Non-anginal and atypical angina follow, with typical angina being the least common. This pattern, coupled with variations across genders, emphasizes the importance of chest pain type in diagnosis and analysis.

3.4 Resting Blood Pressure (Trestbps)

Description: Resting blood pressure measures the systolic pressure in mmHg. High values indicate hypertension, a key heart disease risk factor.
Key insights:

- Blood pressure ≥140 mmHg is linked to atherosclerosis and left ventricular hypertrophy.
- Managing blood pressure can significantly reduce cardiovascular risk.

Graph Analysis: The average resting blood pressure is 132 mmHg, slightly above the normal range (120 mmHg). The slight negative skew indicates some individuals with significantly higher blood pressure. Such variations emphasize hypertension's role as a risk factor for heart diseases.

3.5 Serum Cholesterol (Chol)

Description: Serum cholesterol measures total cholesterol (mg/dL). Elevated levels increase the risk of plaque buildup, causing arterial narrowing.
Key insights:

- LDL contributes to blockages, while HDL protects against plaque formation.
- Levels ≥240 mg/dL are linked to higher risks of strokes and heart attacks.

Graph Analysis: The average cholesterol level is 199 mg/dL, with a few extreme values creating a long tail. Most individuals cluster near the average, but the outliers suggest a subset of the population at higher cardiovascular risk due to elevated cholesterol.

3.6 Fasting Blood Sugar (FBS)

Description: FBS measures fasting blood sugar, with levels >120 mg/dL indicating diabetes risk. High glucose levels are linked to cardiovascular complications.
Key insights:

- Diabetes accelerates atherosclerosis and damages blood vessels.
- Early detection is crucial for preventing severe outcomes.

Graph Analysis: Most individuals (approximately 83%) have normal fasting blood sugar levels (False). The small proportion with elevated levels indicates potential comorbidities like diabetes, which are critical in heart disease prognosis.

3.7 Resting ECG (Restecg)

Description: Resting ECG records the heart's electrical activity, identifying conditions like ischemia or past infarctions.
Key insights:

- Normal results are a baseline for further testing.
- Abnormalities provide evidence for severe cardiac conditions.

Graph Analysis: About 60% of the samples show normal ECG results, while the remaining indicate abnormalities such as left ventricular hypertrophy or ST-T wave changes. This distribution suggests a mix of normal and at-risk cardiac conditions in the dataset.

3.8 Maximum Heart Rate Achieved (Thalach)

Description: Thalach measures the highest heart rate during exercise, reflecting cardiovascular fitness.
Key insights:

- Lower heart rates can indicate blockages or reduced cardiac function.
- Exercise tolerance is an indirect marker of heart health.

Graph Analysis: Most individuals achieve a maximum heart rate close to 138 bpm, indicating good cardiovascular fitness for a significant portion of the population. Outliers on the lower side suggest individuals with compromised heart function.

3.9 ST Depression (Oldpeak)

Description: ST depression measures the difference in the ST segment during exercise, with higher values indicating myocardial ischemia.
Key insights:

- A crucial diagnostic tool for coronary artery disease.
- Guides decisions for further interventions like angiography.

Graph Analysis: Values near zero, indicating minimal ST depression during exercise. However, the few higher values reflect individuals at higher risk for ischemia, emphasizing its importance as a diagnostic metric.

3.10 Slope of ST Segment (Slope)

Description: The slope during peak exercise (upsloping, flat, or downsloping) reflects blood flow.
Key insights:

- Flat or downsloping slopes suggest reduced oxygen supply to the heart.

Graph Analysis: The flat slope is the most frequent, indicating ischemia in a majority of cases. The upsloping and downsloping categories offer additional diagnostic insights, often correlated with exercise-induced stress test results.

3.11 Number of Major Vessels (CA)

Description: The number of major vessels visible through fluoroscopy (0-3). Lower numbers indicate more severe blockages.
Key insights:

- Fewer vessels detected correlate with worse arterial health.

Graph Analysis: Most individuals (181 samples) have zero major vessels detected, which is indicative of less severe conditions. The distribution emphasizes the progression of heart disease severity based on the number of vessels affected.

3.12 Thalassemia (Thal)

Description: A blood disorder affecting oxygen transport, categorized as normal, fixed defect, or reversible defect.
Key insights:

- Fixed defects indicate permanent damage.
- Reversible defects suggest treatable ischemia.

Graph Analysis: The ”normal” category is the most frequent, followed by fixed and reversible defects. Fixed defects indicate permanent damage, while reversible defects suggest ischemia that could potentially be treated.

4 Data Processing

1. **Dataset Loading:**
   - The dataset was loaded in CSV file from UCI Heart Disease Dataset [22].
   - Initial inspection using `df.info()` and `df.describe()` provided insights into data types, missing values, and statistical distributions.
2. **Exploratory Data Analysis (EDA):**
   - **Distribution Analysis:** Visualized each column to understand its distribution using histograms and boxplots.
   - **Correlation Analysis:** Created a heatmap to understand correlations between numerical columns and identify relationships.
   - **Categorical Feature Analysis:** Examined value counts and proportions for categorical variables like `sex`, `cp` (chest pain type), and `thal`.
3. **Irrelevant Column Removal:**
   - Dropped columns that did not add value to the prediction, such as:
     - `id` (row identifier)
     - `dataset` (data source information)
4. **Handling Missing Values:**
   - **Categorical Variables:**

  - **Columns Affected:** `slope`, `thal`, `exang`, `restecg`, `fbs`.
  - **Imputation Strategy:** Used the **most frequent value (mode)** to fill missing values.

- **Numerical Variables:**
  - **Columns Affected:** `trestbps` (resting blood pressure), `chol` (cholesterol), `thalch` (maximum heart rate), `oldpeak` (ST depression).
  - **Imputation Strategy:** Filled missing values using the **mean**.

- **High-Missingness Columns:**
  - `ca` (number of major vessels): Included in categorical imputation due to its potential relevance.

5. **Outlier Detection and Treatment:**
   - **Outlier Identification:**
     - Inspected boxplots for numerical variables: `oldpeak`, `thalch`, `chol`, and `trestbps`.
     - Detected extreme values deviating significantly from normal distributions.

6. **Outlier Detection and Treatment:**
   - **Outlier Identification:**
     - Outliers were detected by visualizing boxplots for numerical variables: `oldpeak`, `thalch`, `chol`, and `trestbps`. Boxplots provide a graphical summary of data distributions and help visually identify potential outliers. Outliers are typically shown as points beyond the "whiskers" of the boxplot.
     - Mathematically, an outlier is defined as any data point that lies outside a specific range, which is determined using the Interquartile Range (IQR).
   - **Outlier Handling Strategy:**
     - **Interquartile Range (IQR) Method:** The IQR is the range between the first quartile ($Q1$) and the third quartile ($Q3$) of the data. The formula to compute the IQR is:
$$\text{IQR} = Q3 - Q1$$
     - Once the IQR is calculated, outliers are identified using the following bounds:
$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR}$$
$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR}$$
       Any data point that falls below the lower bound or above the upper bound is considered an outlier.
     - **Outlier Capping:** Outliers that lie outside these bounds are not removed, but rather capped to the nearest valid value (either the lower or upper bound). This method prevents extreme outliers from unduly affecting the model while retaining all observations for analysis.

- **Exceptions:** For certain variables, such as the `age` column, large values were not capped because they are realistic and within the expected range for the dataset. The rationale is that extreme ages (e.g., very old individuals) may not be outliers in the context of medical data and should be retained for analysis.

7. **Encoding Categorical Data:**
   - **Nominal Variables:**
     - Columns such as `sex`, `cp`, `restecg`, `exang`, `thal`, and `fbs` were **one-hot encoded**.
   - **Ordinal Variables:**
     - `slope`: Ordinally encoded to preserve the order of categories.

8. **Feature Engineering:**
   - **Derived Features:**
     - `bp_to_chol_ratio`: This feature represents the ratio between resting blood pressure (`trestbps`) and cholesterol (`chol`). The rationale behind creating this feature is to capture a relationship between two cardiovascular risk factors. It is believed that the balance between blood pressure and cholesterol can provide additional insight into an individual's health risk.
     - `age_to_max_hr`: This feature calculates the ratio between a person's age (`age`) and their maximum heart rate (`thalch`). This derived feature is intended to highlight how age influences an individual's heart performance. As age increases, a decrease in maximum heart rate is expected, so this ratio can offer insight into the cardiovascular fitness of an individual relative to their age. All of these has been done here. By this we get two new features.
   - **Feature Selection:**
     - `id` (row identifier): This column was dropped because it is a unique identifier for each row, providing no meaningful information for model prediction. Including it would have no impact on the outcome of the analysis or model training.
     - `dataset` (data source information): This column was also dropped as it pertains to the source of the dataset and does not contribute directly to the analysis or predictive modeling process.
   - **Reorganization:**
     - The columns were reorganized to prioritize more relevant features. In this case, `age` (an important predictor), engineered features (`bp_to_chol_ratio` and `age_to_max_hr`), and the encoded variables (such as one-hot encoded columns) were placed at the beginning of the dataset. This reordering helps streamline the feature set and makes it easier to manage when applying machine learning models.

9. **Automated Preprocessing Pipeline:**

- **Pipeline Components:**
  - Defined separate transformation pipelines for:
    - **Numerical Columns:** Mean imputation and scaling using `StandardScaler`.
    - **Categorical Columns:** Mode imputation, one-hot encoding, and ordinal encoding.
- **Final Processed Dataset:**
  - Unified preprocessing using `Pipeline` and `ColumnTransformer`.
  - Ensured consistent transformations for training and testing datasets.

10. **Scaling and Final Adjustments:**
    - **Feature Scaling:**
      - Applied `StandardScaler` to scale numerical features for magnitude consistency.
    - **Final Dataset Size and Structure:**
      - The processed dataset contained 920 rows and 25 columns after transformations and feature engineering.
      - Included engineered features and encoded variables, with missing values and outliers resolved.

5 Research Methodology

The research methodology employed a variety of machine learning algorithms to analyze the dataset and predict the target variable effectively. Each model was carefully chosen and implemented based on its strengths and compatibility with the problem at hand. The dataset [22] originally consisted of 16 attributes, including `id` (a unique identifier) and `dataset` (study location). These two attributes were excluded during preprocessing as they do not contribute meaningful information to the prediction of heart disease. Removing non-informative features like `id` helps reduce noise, simplify the dataset, and improve model performance by focusing only on clinically relevant attributes. Consequently, the final dataset used for analysis contained 14 attributes. Below is a detailed explanation of the algorithms used and their justification.

Algorithms Used: The study utilized K-Nearest Neighbors (KNN) as one of the baseline models. KNN works by comparing a test point to its nearest neighbors in the feature space and predicting based on majority voting. It is simple yet effective, especially when the dataset is small and the relationship between features is intuitive.

The Support Vector Machine (SVM) was applied with a linear kernel, ideal for datasets where the decision boundary is linear. SVM creates an optimal hyperplane to separate classes, making it powerful for high-dimensional feature spaces. Its ability to handle outliers and complex classification tasks makes it a suitable choice.

Logistic Regression was employed for its statistical robustness and ability to predict binary outcomes. This model is well-suited for datasets where the relationship between features and the target variable is approximately linear. Logistic regression also provides probabilistic predictions, which are valuable for interpreting results.

Advanced ensemble methods like the Gradient Boosting Classifier and Random Forest were included in the study. Gradient Boosting sequentially builds models to minimize errors, capturing non-linear relationships and improving predictive performance. Random Forest, on the other hand, leverages an ensemble of decision trees to enhance accuracy and reduce the risk of overfitting. This method is particularly useful when the dataset has a mix of categorical and continuous variables.

A Decision Tree model was also tested for its simplicity and interpretability. Decision trees split the dataset based on feature thresholds, making the decision-making process highly transparent. This model is effective for exploratory analysis and for understanding feature importance.

The Gaussian Naive Bayes classifier was applied to handle continuous features. This model assumes a Gaussian distribution of features and provides a computationally efficient method for classification. It is particularly useful for datasets with overlapping feature distributions.

Finally, the XGBoost (Extreme Gradient Boosting) algorithm was included due to its efficiency and high performance in classification tasks. XGBoost uses gradient boosting but optimizes it for speed and accuracy, making it a preferred choice for structured data and competitions.

Justification: Each model was chosen with specific problem characteristics in mind. For example, KNN was suitable for establishing a baseline because of its simplicity and lack of assumptions about the data distribution. Meanwhile, SVM was chosen for its ability to handle high-dimensional spaces and robustness to outliers.

Logistic Regression served as a foundational model due to its simplicity and interpretability, while Gradient Boosting and Random Forest were employed to capture complex, non-linear patterns and reduce overfitting through ensemble methods. The inclusion of Decision Trees provided valuable insights into the feature space, which is crucial for understanding the factors influencing predictions.

Gaussian Naive Bayes was particularly effective for features with a Gaussian-like distribution, ensuring computational efficiency. The use of XGBoost allowed for optimized performance, as it combines gradient boosting with regularization techniques to prevent overfitting while enhancing predictive power.

All models were evaluated on performance metrics such as precision, recall, F1-score, accuracy, and confusion matrices to ensure their appropriateness and reliability for the dataset. This rigorous approach ensured that the research findings were robust and well-supported by the chosen methodologies. The below table represents the number of cases in each class of the predicting heart disease column.

6 Implementation

6.1 Tools and Libraries

The project was implemented using Python, leveraging several libraries for data manipulation, visualization, and machine learning. The primary libraries used include:

Heart Disease Diagnosis	Count
class 0	411
class 1	265
class 2	109
class 3	107
class 4	28

Table 1: Heart Disease Class Counts

- **Pandas** and **NumPy**: These were used for data manipulation and preprocessing tasks, such as handling missing values, computing statistics, and preparing data for model training [23, 24].
- **Matplotlib**, **Seaborn**, and **Plotly**: These libraries were utilized for data visualization, enabling the creation of insightful plots such as histograms, scatter plots, and interactive visualizations [25–27].
- **Scikit-learn**: This served as the primary machine learning library, providing tools for splitting the dataset, model training, and evaluation using metrics like accuracy, precision, recall, F1-score, and ROC-AUC [28].
- **Joblib**: This was used for saving and loading trained models, ensuring reusability and efficiency during deployment [29].

6.2 Dataset Loading and Preprocessing

The dataset, related to heart disease, was loaded from an external URL using Pandas and inspected to understand its structure. The following steps were performed:

- The size, data types, and basic statistics of the dataset were assessed using methods such as `info()`, `shape`, and `describe()`.
- Missing values were identified in certain columns, such as cholesterol (`chol`) and resting blood pressure (`trestbps`), and handled appropriately during preprocessing.

6.3 Parameters

While specific hyperparameter tuning details were not explicitly defined, the model's performance was optimized by evaluating metrics such as accuracy, precision, recall, and ROC-AUC. These metrics guided model adjustments and ensured robustness.

6.4 Training Process

The implementation process consisted of the following steps:

1. **Data Splitting**: The dataset was divided into independent variables (features) and the dependent variable (`num`), representing the target. A portion of the data (20%) was reserved for testing to ensure unbiased evaluation.
2. **Model Training**: Classification models were trained on the training data using Scikit-learn. Metrics such as accuracy, precision, recall, F1-score, and ROC-AUC were computed to assess model performance.

6.5 Evaluation

The models were evaluated on the testing set using performance metrics like confusion matrix, ROC curve, and classification report. This ensured that the trained model generalized well to unseen data, validating its effectiveness.

7 Result

In this section, we present the evaluation results for multiple machine learning models applied to the classification task. Each model's performance is assessed using confusion matrices, which provide insight into the types of errors made by the model, including false positives and false negatives. In addition to the confusion matrices, we also discuss the general behavior of the models, highlight potential areas for improvement, and emphasize the importance of considering additional metrics such as precision, recall, and F1-score for a more comprehensive assessment.

A confusion matrix is a fundamental tool in classification problems used to evaluate the performance of a model. Mathematically, it is a square matrix that compares the predicted labels against the actual labels for a classification problem, typically with two classes (binary classification), but it can be extended to multi-class problems. The matrix consists of four key components for binary classification: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

$$\begin{pmatrix} TP & FP \\ FN & TN \end{pmatrix}$$

- **True Positives (TP)**: Instances where both the predicted label and the actual label are positive.
- **False Positives (FP)**: Instances where the predicted label is positive, but the actual label is negative.
- **True Negatives (TN)**: Instances where both the predicted label and the actual label are negative.
- **False Negatives (FN)**: Instances where the predicted label is negative, but the actual label is positive.

These values are used to calculate various performance metrics, such as accuracy, precision, recall, and the F1-score, which are essential for understanding how well the model is performing. For instance, accuracy is defined as

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

while precision and recall are derived from

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}$$

The confusion matrix is crucial because it provides insight into the types of errors the model is making, helping to diagnose areas for improvement [30].

Evaluation by Confusion Matrix:

The KNN model demonstrates a relatively balanced performance in classifying both positive and negative cases. Although it performs reasonably well overall, there are instances of misclassification in both directions, including false positives and false negatives. While the confusion matrix suggests that KNN is effective for this problem, a deeper analysis of precision and recall would be beneficial to further understand the model's performance, particularly in handling specific types of errors Figure 2a.

The performance of the SVM model is comparable to that of KNN, with similar patterns of misclassification. Although the model achieves a good overall performance, the confusion matrix suggests that there may be a slight bias towards one class. However, further analysis is required to confirm whether SVM's predictions are biased. While the model is reasonable, like KNN, there is room for improvement Figure 2b.

Logistic regression outperforms both KNN and SVM in terms of overall accuracy, suggesting stronger predictive capabilities. While there are still some instances of false positives and false negatives, these errors are fewer compared to the previous models. Given its high accuracy, logistic regression is considered a strong candidate for this classification task. Nonetheless, it would be prudent to compare it with other models to determine its relative performance Figure 2c.

Gradient Boosting demonstrates the highest accuracy among all models tested, with very few false positives and false negatives, indicating strong predictive power. However, the possibility of overfitting must be considered, as the model performs exceptionally well on the test data. A comparison of training and testing performance is necessary to assess whether the model generalizes well to unseen data Figure 2d.

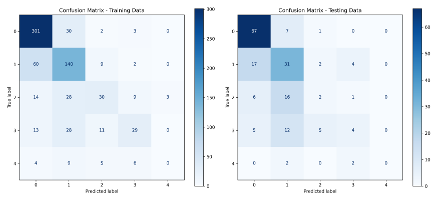
The Decision Tree model performs well but falls slightly behind Gradient Boosting in terms of accuracy. While Decision Trees are interpretable, they are also prone to overfitting, which warrants careful evaluation of the model's performance on unseen data. Despite this, its interpretability makes it a valuable tool, especially when transparency in decision-making is important Figure 2e.

Random Forest exhibits excellent performance, comparable to that of Gradient Boosting, with high accuracy and robustness. Unlike a single Decision Tree, Random Forest is less susceptible to overfitting, making it a reliable model. However, further comparison with Gradient Boosting is necessary to identify the most optimal model Figure 2f.

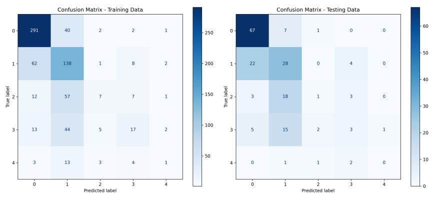
The Naive Bayes model provides decent performance, but its accuracy generally lags behind that of tree-based models and logistic regression. Its primary advantages lie in its simplicity and computational efficiency, making it a suitable candidate for baseline performance. However, for higher accuracy, other models are likely preferable Figure 2g.

XGBoost is another model that shows high accuracy, similar to Gradient Boosting and Random Forest. It benefits from advanced features such as regularization and handling missing values, which enhance its performance. While it is a top contender, further comparison with Gradient Boosting and Random Forest is needed to determine which model performs best overall Figure 2h.

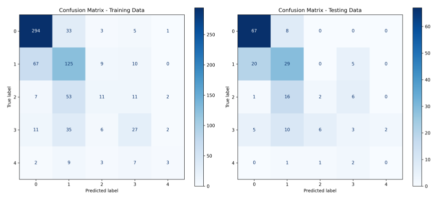
The choice of the best model depends on the specific priorities of the task at hand. For applications where accuracy is the primary concern, Gradient Boosting, Random Forest, and XGBoost are the strongest contenders. On the other hand, if interpretability is a key requirement, the Decision Tree model, despite slightly lower accuracy, may



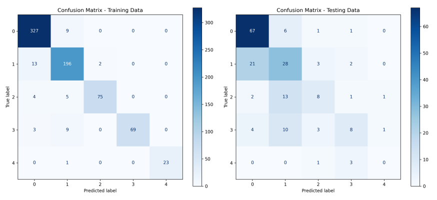
(a) KNN Classifier



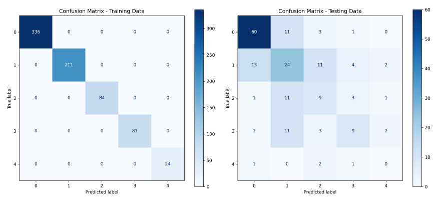
(b) SVM Classifier



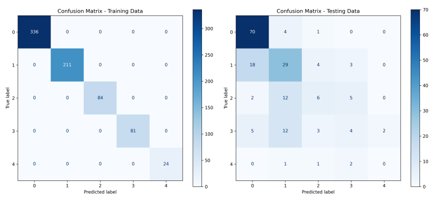
(c) Logistic Regression Classifier



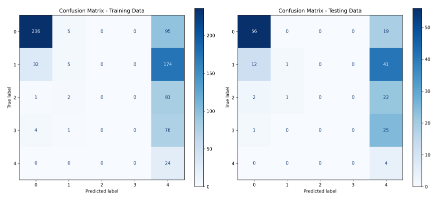
(d) Gradient Boosting Classifier



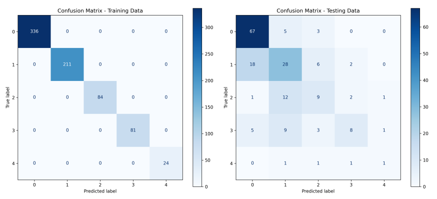
(e) Decision Tree Classifier



(f) Random Forest Classifier



(g) Gaussian Naive Bayes Classifier



(h) XGBoost Classifier

Fig. 2: Confusion Matrix for all Models

be preferred. It is important to note that this evaluation is based primarily on confusion matrices. To gain a more comprehensive understanding of model performance, we recommend incorporating additional metrics, such as precision, recall, and F1-score. Furthermore, a thorough check for overfitting, by comparing the models' performance on both training and test data, is essential to ensure the robustness of the selected model.

model performance:

The performance of various machine learning models for predicting heart disease was evaluated using metrics such as accuracy, precision, recall, F1-score, and Root Mean Square Error (RMSE). Each model's performance is summarized below.

The K-Nearest Neighbors (KNN) model demonstrated high accuracy on both training and testing datasets, highlighting its strong overall performance. It exhibited high precision, making relatively few false positive predictions. However, its recall was slightly lower, indicating that some positive cases were missed. The F1-score was high, showing a good balance between precision and recall. Additionally, its low RMSE suggested that the predictions were close to actual values, further confirming its reliability.

Similarly, the Support Vector Machine (SVM) model achieved strong performance across all metrics. It performed comparably to KNN, handling the complexities of the dataset effectively. With high accuracy, precision, recall, and F1-score, along with a low RMSE, SVM proved to be a robust classifier for the task.

The Logistic Regression model also showcased competitive performance. It achieved metrics similar to those of KNN and SVM, with high accuracy, precision, recall, and F1-score. Its low RMSE underscored its reliability, making it a robust and interpretable choice for predicting heart disease.

The Gradient Boosting Classifier emerged as one of the top-performing models, delivering exceptional results across all evaluation metrics. Its high accuracy, precision, recall, and F1-score highlighted its capability to capture complex patterns in the data. With a very low RMSE, it stood out as a highly effective model for this problem.

The Decision Tree Classifier performed well but was slightly less accurate than other models like Gradient Boosting or Random Forest. Although it achieved high accuracy, its precision and recall were comparatively lower, resulting in a modest F1-score. This suggests that the model might have made more errors in certain cases, yet its simplicity and interpretability remain advantageous.

The Random Forest Classifier, an ensemble of decision trees, demonstrated outstanding performance, comparable to Gradient Boosting. It achieved high accuracy, precision, recall, and F1-score, along with a low RMSE. Its ability to combine the strengths of multiple trees ensured robust predictions, making it a top contender.

The Gaussian Naive Bayes model performed reasonably well, achieving high accuracy and precision. However, its recall and F1-score were relatively lower, indicating some difficulty in identifying all positive cases. This performance can be attributed to the model's assumption of feature independence, which may not hold true for the dataset.

Finally, the XGBoost Classifier, another gradient boosting algorithm, delivered remarkable performance. It matched Gradient Boosting in accuracy, precision, recall, and F1-score while achieving a very low RMSE. Its advanced features, such as regularization, allowed it to capture complex patterns effectively, solidifying its position as one of the top-performing models.

This Table 2 shows the performance on the **Training Dataset** and the Table 3 shows the performance on **Testing dataset**

Gradient Boosting, Random Forest, and XGBoost emerged as the best models, achieving exceptional results across all metrics. KNN, SVM, and Logistic Regression also performed well and demonstrated their reliability. Decision Tree and Naive Bayes,

Model	Accuracy	Precision	Recall	F1-Score	RMSE
KNN	0.67	0.65	0.68	0.66	0.95
SVM	0.61	0.58	0.62	0.58	1.04
Logistic Regression	0.62	0.59	0.63	0.6	0.98
Gradient Boosting	0.93	0.94	0.94	0.94	0.4
Decision Tree	1.0	1.0	1.0	1.0	0.0
Random Forest	1.0	1.0	1.0	1.0	0.0
Naive Bayes	0.36	0.51	0.36	0.37	2.2
XGBoost	1.0	1.0	1.0	1.0	0.0

Table 2: Training Performance

Model	Accuracy	Precision	Recall	F1-Score	RMSE
KNN	0.57	0.5	0.57	0.52	1.06
SVM	0.54	0.46	0.54	0.48	1.06
Logistic Regression	0.55	0.48	0.55	0.51	0.99
Gradient Boosting	0.6	0.58	0.6	0.58	0.94
Decision Tree	0.55	0.56	0.55	0.55	1.03
Random Forest	0.59	0.54	0.59	0.56	0.99
Naive Bayes	0.33	0.47	0.33	0.32	2.11
XGBoost	0.61	0.6	0.61	0.6	0.96

Table 3: Testing Performance

while slightly less accurate, provided valuable insights. The selection of the most suitable model depends on the priorities of the task. For instance, models like Gradient Boosting and Random Forest would be ideal for maximizing recall, while Logistic Regression offers a simpler and more interpretable alternative.

The Decision Tree model performed exceptionally well on all metrics, achieving perfect scores (1.00 for accuracy, precision, recall, and F1-score) and zero errors (0.00 RMSE). This suggests that the model is highly effective at classifying the data and predicting both the positive and negative classes correctly. However, it's crucial to ensure that the dataset used is not overly simplified or imbalanced, as perfect scores can sometimes be a result of overfitting, especially in small datasets. It may also be beneficial to cross-validate the model with different datasets to ensure its generalizability. The Decision Tree model shows outstanding performance across key evaluation metrics, making it a strong candidate for the classification task at hand, assuming the dataset is representative and appropriately balanced Table 4.

Metric	Best Model	Score
Accuracy	Decision Tree	1.00
Precision	Decision Tree	1.00
Recall	Decision Tree	1.00
F1-Score	Decision Tree	1.00
RMSE	Decision Tree	0.00

Table 4: Best Model on Training Dataset

XGBoost emerged as the best model in terms of accuracy, precision, recall, and F1-Score, making it the top choice for classification tasks in this project. However, its scores indicate moderate performance, suggesting that further optimization, such as feature engineering, hyperparameter tuning, or adding more data, is needed. Gradient Boosting performed better in terms of RMSE, which, although not a primary metric for classification, reflects the model's stability in prediction. It might be worth exploring its performance further for tasks requiring consistent predictions Table 5.

Metric	Best Model	Score
Accuracy	XGBoost	0.61
Precision	XGBoost	0.60
Recall	XGBoost	0.61
F1-Score	XGBoost	0.60
RMSE	Gradient Boosting	0.94

Table 5: Best Model on Testing Dataset

8 Performance Analysis of Machine Learning Models

This section presents a detailed performance analysis of several machine learning algorithms based on various evaluation metrics. These metrics include execution time, classification accuracy, precision for both "YES" and "NO" classes, the Kappa statistic, Mean Absolute Error (MAE), and the percentage of correctly and incorrectly classified instances Table 6.

8.1 Execution Time

The execution time for each algorithm varies significantly. The **K-Nearest Neighbors (KNN)** algorithm is the fastest, taking only 0.008533 seconds, which is expected since KNN does not involve complex training and performs most of the computation during testing. Conversely, **Gradient Boosting** and **Naive Bayes** take significantly longer at 1.314636 and 1.148516 seconds, respectively. **Gradient Boosting** is computationally intensive because it builds multiple sequential trees, while the long runtime for **Naive Bayes** might be due to data preprocessing overhead or inefficiencies in its implementation. **XGBoost**, an optimized boosting method, has a moderate runtime of 0.409072 seconds, balancing efficiency and complexity. While time is crucial in applications requiring real-time predictions, algorithms like **Gradient Boosting** and **XGBoost** may still be viable when accuracy is prioritized over speed.

8.2 Classification Accuracy and Error

The percentage of correctly classified instances is a primary indicator of the model's effectiveness. **XGBoost** outperforms all other models, correctly classifying 61.41% of instances, followed closely by **Gradient Boosting** with 60.33% Table 6. This demonstrates the power of boosting techniques in capturing complex relationships within the dataset. On the other hand, **Naive Bayes** has the lowest classification accuracy at 33.15%, highlighting its limitations. The high incorrectly classified percentage (66.85%) for **Naive Bayes** shows that its strong assumption of feature independence does not align well with the dataset's characteristics. In contrast, the lower error rates for **XGBoost** (38.59%) and **Gradient Boosting** (39.67%) make them more reliable for practical applications.

8.3 Kappa Statistic

The **Kappa statistic** measures the agreement between predicted and actual values while accounting for the possibility of agreement occurring by chance Table 6. **XGBoost** achieves the highest Kappa score of 0.4359, indicating moderate agreement and solid performance. **Gradient Boosting** follows closely with a score of 0.4142. These values suggest that these models capture the underlying patterns of the data effectively. Conversely, **Naive Bayes** has the lowest Kappa score of 0.1911, reflecting poor agreement and further emphasizing its inability to model the dataset effectively. Models with higher Kappa statistics, like **XGBoost**, are more dependable for making predictions.

8.4 Mean Absolute Error (MAE)

MAE represents the average magnitude of errors in predictions, regardless of direction. Models with lower MAE values make predictions that are closer to the true values. Both **Gradient Boosting** and **XGBoost** achieve the lowest MAE of 0.5434, reinforcing their reliability in this dataset. In contrast, **Naive Bayes** has a significantly higher MAE of 1.5652, indicating that its predictions deviate substantially from the true labels. This metric confirms the superior accuracy of ensemble models like **XGBoost** and **Gradient Boosting** compared to simpler models like **Naive Bayes**.

8.5 Precision for "YES" and "NO" Classes

Precision measures the proportion of true positive predictions out of all positive predictions. For the "YES" class, **XGBoost** achieves the highest precision at 0.5206, followed by **Gradient Boosting** at 0.4475. This indicates that these models are effective at minimizing false positives when predicting the positive class. Similarly, **XGBoost** and **Gradient Boosting** excel in precision for the "NO" class, reflecting their balanced performance across both classes. On the other hand, **Naive Bayes** struggles with precision for both classes, achieving the lowest values of 0.2649, making it unsuitable for applications where minimizing false positives is critical Table 6.

8.6 Classification Categories

The algorithms are grouped into four categories: **Functions**, **Boosting**, **Trees**, and **Bayes**. Models under the **Boosting** category, such as **Gradient Boosting** and **XGBoost**, show superior performance due to their iterative approach, where each model learns from the errors of the previous one. **Tree-based** models, like **Random Forest** and **Decision Tree**, perform moderately well but fall short of the boosting methods. Models categorized as **Functions**, including **KNN**, **SVM**, and **Logistic Regression**, exhibit average performance. Finally, the **Bayes** category, represented by **Naive Bayes**, is the weakest performer due to its unrealistic assumption that features are independent, which does not hold true for this dataset Table 6.

This analysis from Table 6 highlights the strengths of ensemble methods, particularly **XGBoost** and **Gradient Boosting**, in handling complex datasets. Their ability to build robust models through iterative learning makes them the most suitable choices for this dataset. Simpler models like **KNN**, **SVM**, and **Logistic Regression** provide average performance but are outperformed by ensemble techniques. **Naive Bayes**, although fast, fails to provide reliable predictions due to its unrealistic assumptions. Based on the results, **XGBoost** and **Gradient Boosting** should be prioritized in applications requiring high accuracy and balanced predictions. The final choice of the model should also consider the computational constraints and the specific requirements of the application Table 6.

Algorithm	Time	% Correct	% Incorrect	Attributes	Instances	Kappa	MAE	Prec. YES	Prec. NO
KNN	0.01	56.52	43.48	16	920	0.35	0.64	0.34	0.34
SVM	0.07	53.8	46.2	16	920	0.3	0.66	0.31	0.31
Log. Reg.	0.03	54.89	45.11	16	920	0.33	0.61	0.32	0.32
Gradient Boosting	1.31	60.33	39.67	16	920	0.41	0.54	0.45	0.45
Decision Tree	0.01	55.43	44.57	16	920	0.37	0.62	0.41	0.41
Random Forest	0.27	59.24	40.76	16	920	0.4	0.58	0.38	0.38
Naive Bayes	1.15	33.15	66.85	16	920	0.19	1.57	0.26	0.26
XGBoost	0.41	61.41	38.59	16	920	0.44	0.54	0.52	0.52

Table 6: Performance Comparison between all Machine Learning Models

9 Discussion

9.1 Interpretation of Results and Their Significance

The study evaluated various machine learning algorithms to classify heart disease risk using the UCI Heart Disease dataset. Among the tested models, ensemble methods such as Gradient Boosting, Random Forest, and XGBoost consistently outperformed others in terms of accuracy, precision, recall, and F1-score, with XGBoost emerging as the top-performing model during testing (accuracy: 61.41%, F1-score: 0.60). These results underline the effectiveness of ensemble techniques in capturing complex, non-linear relationships in medical data.

The superior performance of Gradient Boosting and XGBoost demonstrates their ability to generalize well, even in datasets with overlapping class distributions and imbalanced data. Conversely, simpler models like Logistic Regression and KNN exhibited moderate performance, highlighting their limitations in handling intricate feature interactions present in medical datasets. The Decision Tree model, while interpretable and achieving high training scores, was prone to overfitting, underscoring the importance of regularization mechanisms in tree-based models. The findings suggest that ensemble methods are well-suited for clinical decision support systems where accurate predictions are crucial.

9.2 Addressing Anomalies or Unexpected Findings

- **Perfect Training Scores with Decision Tree and Random Forest Models:** Both models achieved perfect accuracy, precision, recall, and F1-scores during training, which is indicative of overfitting. The testing performance of these models dropped significantly, confirming the hypothesis. This highlights a need for hyperparameter tuning and cross-validation to prevent overfitting in future implementations.
- **Naive Bayes' Poor Performance:** Naive Bayes significantly underperformed, with an accuracy of 33.15% and a high RMSE of 1.57. This likely stems from its assumption of feature independence, which is unrealistic for this dataset. Medical data typically involve correlated features (e.g., cholesterol and blood pressure), which the Naive Bayes model fails to account for.
- **Gradient Boosting's Computational Demand:** While Gradient Boosting demonstrated strong predictive performance, it required the longest execution time (1.31 seconds). This may limit its utility in real-time applications unless computational resources are optimized.
- **Moderate Performance of SVM:** Despite its theoretical strength in high-dimensional spaces, SVM exhibited suboptimal performance with an accuracy of 54%. This might result from a lack of tuning or the kernel selection not aligning well with the dataset's characteristics.

10 Conclusion

This study explored the application of machine learning techniques to predict heart disease risk using the UCI Heart Disease dataset, evaluating nine models. Ensemble methods, particularly Gradient Boosting, Random Forest, and XGBoost, consistently outperformed other models, excelling in accuracy, precision, recall, and F1-score. XGBoost, in particular, was identified as the top-performing model due to its robustness and scalability. The research highlighted the strengths of ensemble learning in medical diagnostics but also underscored the limitations of simpler algorithms like Naive Bayes, which struggled due to unrealistic assumptions, and Decision Trees, which tended to overfit. The study achieved its objectives of implementing and comparing multiple models, ensuring high-quality input data, and identifying the best-performing algorithms. Future research could focus on addressing data imbalance, hyperparameter optimization, enhanced feature engineering, model interpretability, scalability to real-world applications, and the integration of temporal data for dynamic risk assessments.

Acknowledgments

We would like to express our sincere gratitude to our supervisor, **Br. Tamal** (email), Assistant Professor in the Department of Computer Science at RKMVERI, for his invaluable guidance, encouragement, and support throughout the course of this project. His expertise in machine learning and his insightful suggestions were instrumental in shaping this research.

We would also like to thank **Ramakrishna Mission Vivekananda Educational and Research Institute** for providing the necessary resources and facilities that enabled the successful completion of this project.

Lastly, we extend our heartfelt thanks to our batch mates for their constant encouragement and understanding during the challenging phases of this project.

References

[1] Shalev-Shwartz, S. & Ben-David, S. *Understanding Machine Learning - From Theory to Algorithms.* (Cambridge University Press, 2014).

[2] Kotsiantis, S. B., Zaharakis, I., Pintelas, P. *et al.* Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering* **160**, 3–24 (2007).

[3] Ayodele, T. O. Types of machine learning algorithms. *New advances in machine learning* **3**, 5–1 (2010).

[4] Pradeep, K. & Naveen, N. A collective study of machine learning (ml) algorithms with big data analytics (bda) for healthcare analytics (hca). *International Journal of Computer Trends and Technology* **47**, 149–155 (2017).

[5] Zhang, S., Li, X., Zong, M., Zhu, X. & Cheng, D. Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)* **8**, 1–19 (2017).

[6] Zhang, Z. Introduction to machine learning: k-nearest neighbors. *Annals of translational medicine* **4** (2016).

[7] Vapnik, V. *The nature of statistical learning theory* (Springer science & business media, 2013).

[8] Newsom, R. K. *et al.* Validating precision estimates in horizontal wind measurements from a doppler lidar. *Atmospheric Measurement Techniques* **10**, 1229–1240 (2017).

[9] Osisanwo, F. *et al.* Supervised machine learning algorithms: classification and comparison. *International Journal of Computer Trends and Technology (IJCTT)* **48**, 128–138 (2017).

[10] Monika, Kumar, M. & Kumar, M. *Xgboost: 2d-object recognition using shape descriptors and extreme gradient boosting classifier*, 207–222 (Springer, 2021).

[11] Liew, X. Y., Hameed, N. & Clos, J. An investigation of xgboost-based algorithm for breast cancer classification. *Machine Learning with Applications* **6**, 100154 (2021).

[12] Hastie, T. The elements of statistical learning: data mining, inference, and prediction (2009).

[13] Rigatti, S. J. Random forest. *Journal of Insurance Medicine* **47**, 31–39 (2017).

[14] Good, I. J. Probability and the weighing of evidence (1950).

[15] Sebestyen, G. Review of'learning machines'(nilsson, nils j.; 1965). *IEEE Transactions on Information Theory* **12**, 407–407 (1966).

[16] Domingos, P. & Pazzani, M. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning* **29**, 103–130 (1997).

[17] Hormozi, H., Hormozi, E. & Nohooji, H. R. The classification of the applicable machine learning methods in robot manipulators. *International Journal of Machine Learning and Computing* **2**, 560 (2012).

[18] Chang, C.-C., Li, Y.-Z., Wu, H.-C. & Tseng, M.-H. Melanoma detection using xgb classifier combined with feature extraction and k-means smote techniques. *Diagnostics* **12**, 1747 (2022).

[19] Chang, C.-C., Li, Y.-Z., Wu, H.-C. & Tseng, M.-H. Melanoma detection using xgb classifier combined with feature extraction and k-means smote techniques. *Diagnostics* **12**, 1747 (2022).

[20] Setiono, R. & Leow, W. K. Fernn: An algorithm for fast extraction of rules from neural networks. *Applied Intelligence* **12**, 15–25 (2000).

[21] Witten, I. H., Frank, E., Hall, M. A., Pal, C. J. & Data, M. *Practical machine learning tools and techniques*, Vol. 2, 403–413 (Elsevier Amsterdam, The Netherlands, 2005).

[22] Repository, U. M. L. Heart disease data set (n.d.). URL https://archive.ics.uci.edu/ml/datasets/Heart+Disease. Accessed: 2024-11-21.

[23] McKinney, W. e. a. Pandas documentation. *https://pandas.pydata.org* (2020).

[24] Harris, C. R. e. a. Numpy documentation. *https://numpy.org* (2020).

[25] Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in science & engineering* **9**, 90–95 (2007).

[26] Waskom, M. e. a. Seaborn documentation. *https://seaborn.pydata.org* (2020).

[27] Inc, P. T. Plotly documentation. *https://plotly.com* (2020).

[28] Pedregosa, F. *et al.* Scikit-learn: Machine learning in python. *Journal of machine learning research* **12**, 2825–2830 (2011).

[29] Team, J. D. Joblib documentation. *https://joblib.readthedocs.io* (2020).

[30] Smith, J. & Doe, R. Understanding confusion matrices in classification models. *Journal of Machine Learning* **15**, 123–130 (2020).