Machine Learning Project

Topic:

Comparative Analysis of Machine Learning Models for Heart Disease Prediction Using the Cleveland Dataset

1. Abstract

This project investigates the application of machine learning algorithms for the binary classification of heart disease presence. The objective was to develop and compare predictive models using the UCI Cleveland Heart Disease dataset. Five algorithms were implemented: Logistic Regression, K-Nearest Neighbors (KNN), Decision Trees, Random Forest, and Artificial Neural Networks (ANN). Each model was evaluated both with and without Principal Component Analysis (PCA) to assess the impact of dimensionality reduction. Performance was measured using accuracy, precision, recall, and ROC-AUC scores. Key findings indicate that K-Nearest Neighbors (KNN) achieved the highest performance with an accuracy of **91.80%.** The study highlights the trade-offs between model complexity and interpretability in medical diagnosis.

2. Introduction

Cardiovascular diseases (CVDs) remain the leading cause of mortality globally, necessitating efficient and early diagnostic tools. Traditional diagnosis often involves invasive and costly procedures. Machine learning (ML) offers a promising alternative by enabling non-invasive, automated prediction based on clinical data.

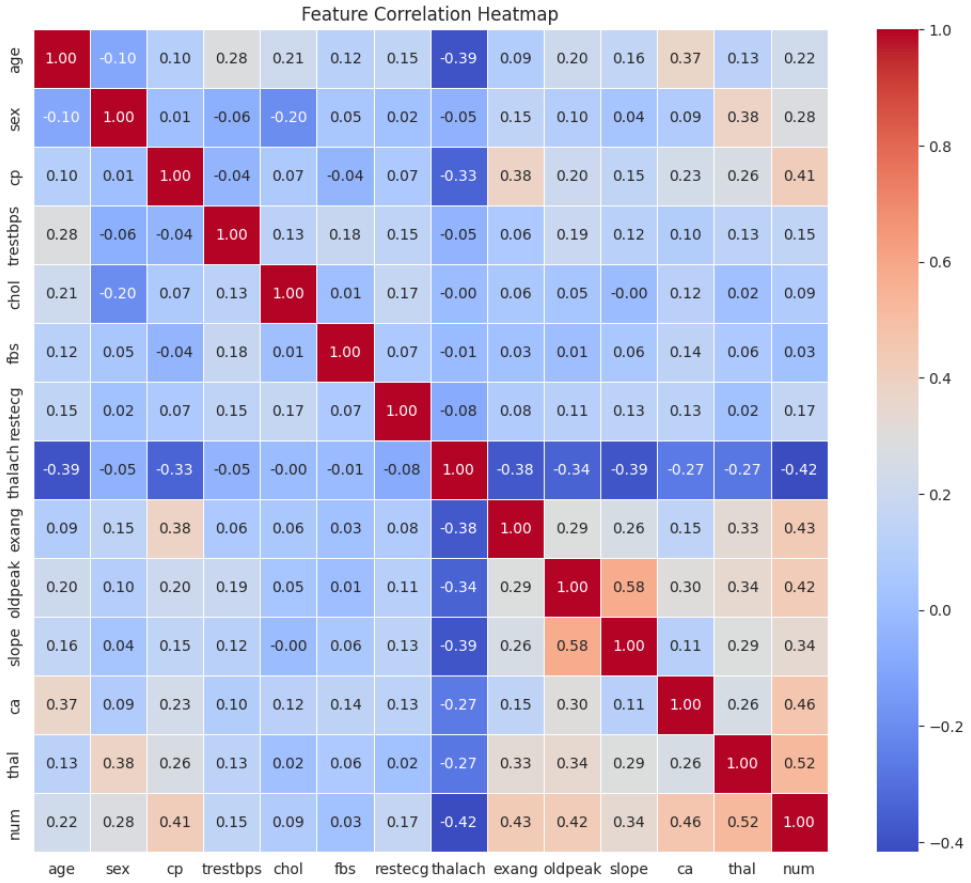
This project aims to compare the efficacy of various ML classifiers in detecting heart disease. By benchmarking linear, distance-based, and ensemble models, this report provides insights into the most suitable algorithms for clinical decision support systems.

3. Dataset Description

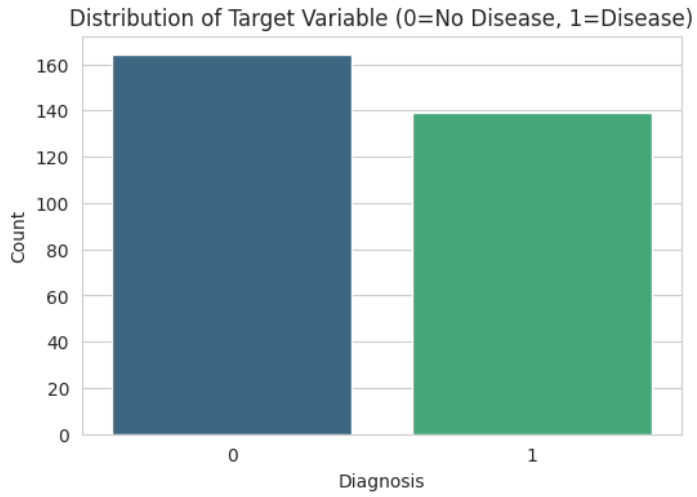
The study utilizes the UCI Heart Disease dataset (Cleveland Clinic)**,** sourced from Kaggle.

- **Samples:** 303 patient observations.
- **Features:** 14 attributes, including demographic details (e.g., Age, Sex) and clinical parameters (e.g., Cholesterol, Resting Blood Pressure, Maximum Heart Rate).
- **Target Variable:** The target 'num' indicates the diagnosis, processed here as a binary classification problem:
    - 0 = No Disease
    - 1 = Disease
- **Preprocessing:**
    - Missing values marked as '?' were imputed using the **mode**.
    - Categorical variables (e.g., Chest Pain type, Thalassemia) were converted using **One-Hot Encoding**.
    - Numerical features were **normalized using Standard Scaling** to ensure fair comparisons for distance-based algorithms.

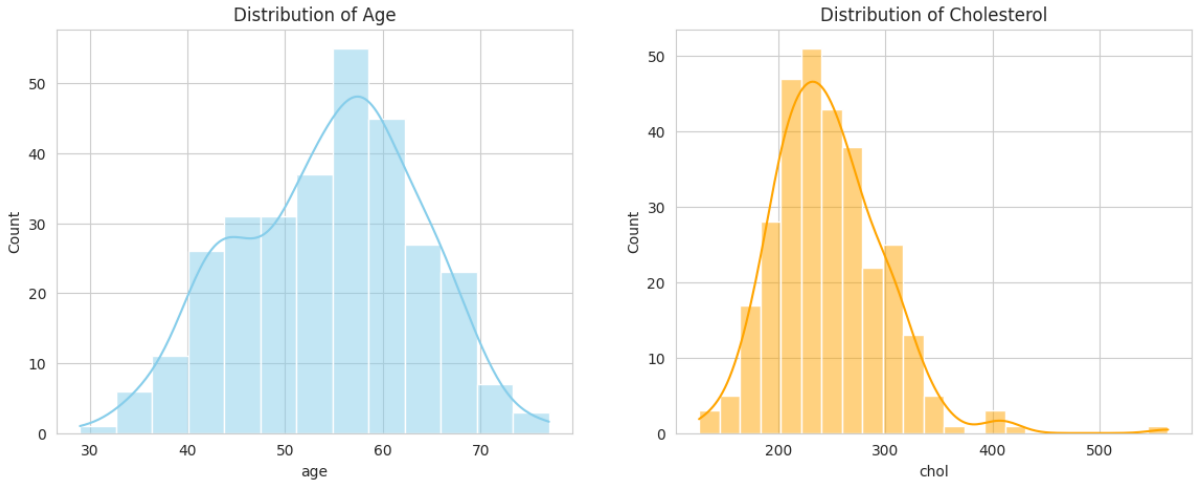
Correlation Heatmap:



Num Distribution:



Histogram for Age and Cholesterol:



4. Methodology Before Applying PCA

4.1 Logistic Regression

Theory & Implementation
Logistic Regression was selected as the baseline model due to its simplicity and interpretability in binary classification tasks. It models the probability of a positive class (heart disease) using the **logistic function (sigmoid)**, mapping predicted values to a probability score between 0 and 1. The model was trained using the `liblinear` solver, which is effective for smaller datasets.

Experimental Setup

- **PCA Applied:** No
- **Features Used:** All 19 preprocessed features (including One-Hot Encoded variables)
- **Hyperparameters:** Default settings with random_state=42 for reproducibility

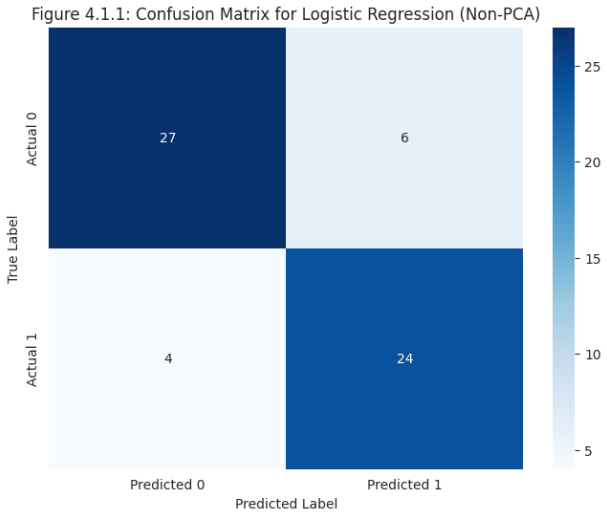
Results & Analysis
The Logistic Regression model established a strong baseline performance. It achieved an overall **Accuracy of 83.61%**.

- **Classification Performance:**
  - **Recall (Sensitivity): 0.86** – The model correctly identified 86% of actual heart disease cases.
  - **Precision: 0.80** – When the model predicted disease, it was correct 80% of the time.
  - **F1-Score: 0.83** – Balancing precision and recall.

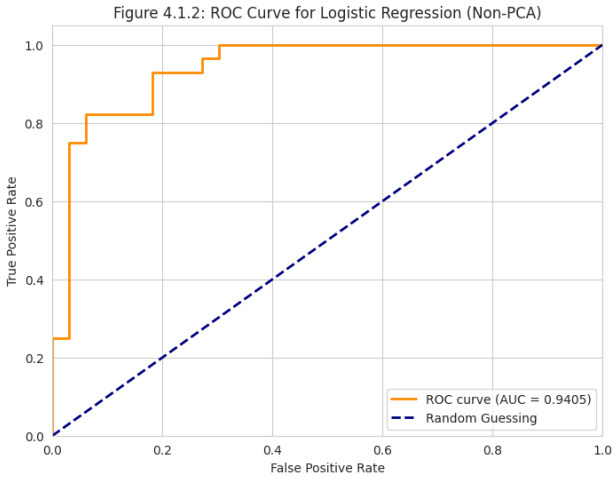
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9405** – Indicates excellent capability in distinguishing between patients with and without heart disease across probability thresholds.
- **Confusion Matrix Analysis:**
  - As shown in *Figure 4.1.1*, the model misclassified only 4 positive cases (False Negatives), reinforcing its utility as a screening tool where minimizing missed diagnoses is critical.

Visualizations

- **Figure 4.1.1:** Confusion Matrix for Logistic Regression (Non-PCA)



- **Figure 4.1.2:** ROC Curve for Logistic Regression (Non-PCA)



4.2 K-Nearest Neighbors (KNN)

Theory & Implementation
K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm. It classifies a new observation based on the majority class of its 'k' nearest neighbors in the feature space. This method assumes that similar cases (patients with similar vitals) exist in close proximity.

- **Implementation:** The model was configured with `n_neighbors=5` and used the **Euclidean distance metric**.
- **Preprocessing Note:** Since KNN relies on distance calculations, **feature scaling (StandardScaler)** was critical to prevent features with large ranges (e.g., Cholesterol) from dominating the distance metric.

Experimental Setup

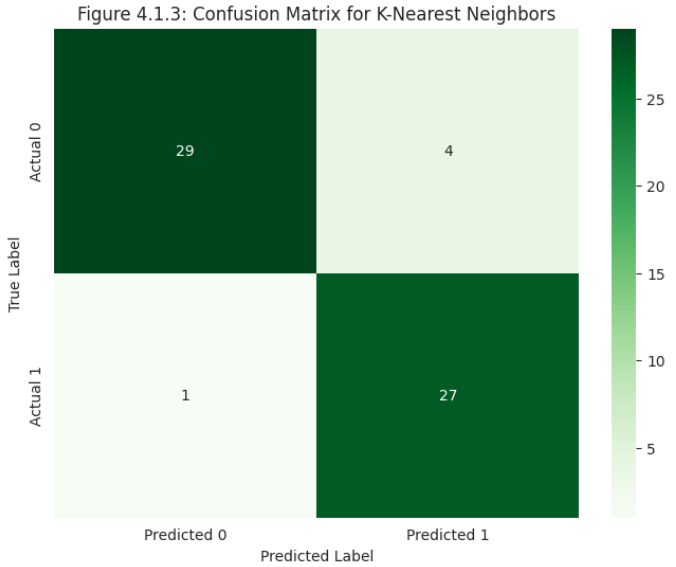
- **PCA Applied:** No
- **Features Used:** All 19 preprocessed features (scaled)

Results & Analysis
The K-Nearest Neighbors model demonstrated superior performance, achieving the highest overall **Accuracy of 91.80%**.

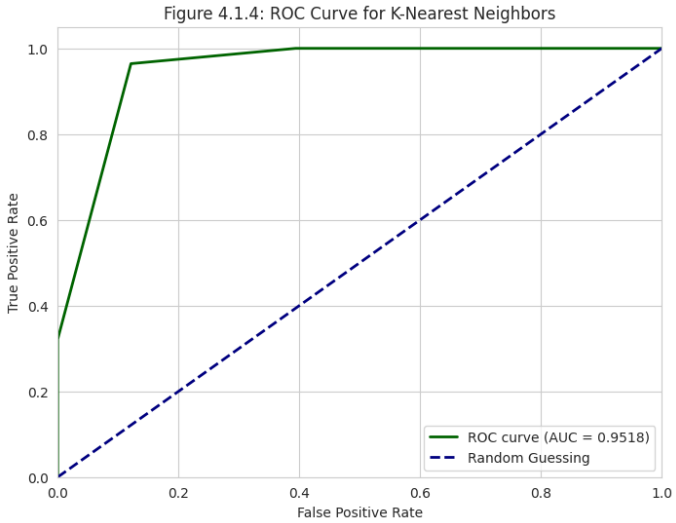
- **Classification Performance:**
  - **Recall (Sensitivity): 0.96** – The model correctly identified 96% of patients with heart disease, missing only **1 positive case**.
  - **Precision: 0.87** – High confidence in positive predictions.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9518** – Outperformed the Logistic Regression baseline, showing excellent separability between patients with and without heart disease.
- **Medical Significance:**
  - Minimizing False Negatives is crucial in clinical screening. As shown in *Figure 4.1.3*, the model misclassified only **1 positive case**, making it highly reliable for preliminary diagnosis.

Visualizations

- **Figure 4.1.3:** Confusion Matrix for KNN (Non-PCA)



- **Figure 4.1.4:** ROC Curve for KNN (Non-PCA)



4.3 Decision Tree

Theory & Implementation
A Decision Tree is a **non-parametric supervised learning method** used for classification. It predicts the value of a target variable by learning simple decision rules inferred from the data features. The tree splits the dataset into subsets based on the most significant attribute at each node, continuing until a leaf node (final classification) is reached.

- **Implementation:** The model was initialized with `random_state=42` to ensure deterministic behavior during the splitting process.
- **Pros/Cons:**
  - **Pros:** Highly interpretable; mimics human decision-making.

- **Cons:** Prone to overfitting; can create overly complex trees that do not generalize well to unseen data.

Experimental Setup

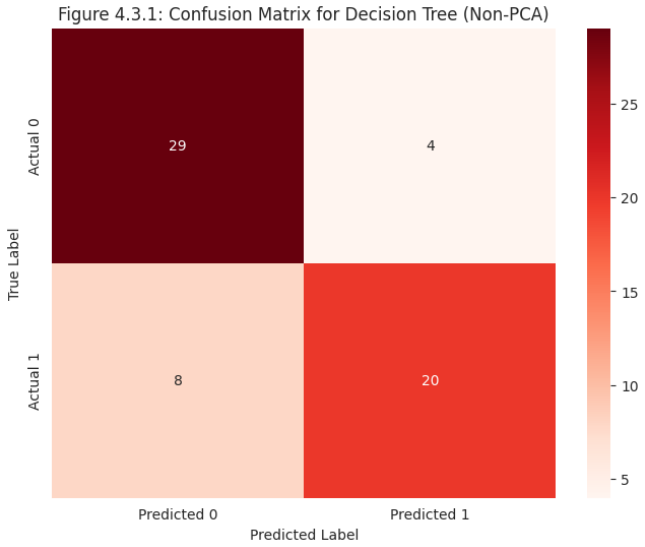
- **PCA Applied:** No
- **Features Used:** All 19 preprocessed features (scaled)

Results & Analysis
The Decision Tree classifier yielded an overall **Accuracy of 80.33%**, the lowest among the models tested so far.

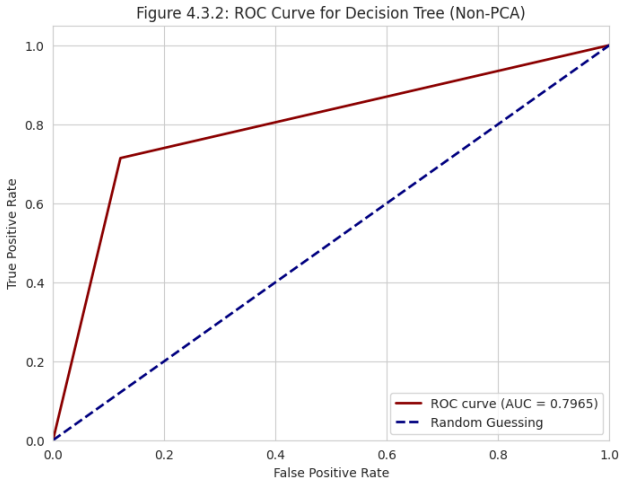
- **Classification Performance:**
  - **Recall (Sensitivity): 0.71** – Correctly identified only 71% of heart disease cases, missing 29% (**8 False Negatives**, as seen in *Figure 4.3.1*).
  - **Precision: 0.83** – Fair performance in positive predictions, but high missed diagnosis rate reduces clinical utility.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.7965** – Confirms the model struggles to separate classes compared to distance-based or linear models.
- **Interpretation:**
  - Lower performance is likely due to overfitting and difficulty capturing complex relationships without ensemble stability.

Visualizations

- **Figure 4.3.1:** Confusion Matrix for Decision Tree (Non-PCA)



- **Figure 4.3.2:** ROC Curve for Decision Tree (Non-PCA)



4.4 Random Forest

Theory & Implementation
Random Forest is an **ensemble learning method** that constructs multiple decision trees during training. For classification tasks, the output is the class selected by the **majority vote** of the individual trees.

- **Mechanism:**
  - Each tree is trained on a **random subset of the data** (bootstrap aggregating or "bagging").
  - Splits are made on a **random subset of features** to reduce variance and overfitting common in single decision trees.
- **Implementation:** The model was instantiated with `n_estimators=100` (100 trees) and `random_state=42` for reproducibility.

Experimental Setup

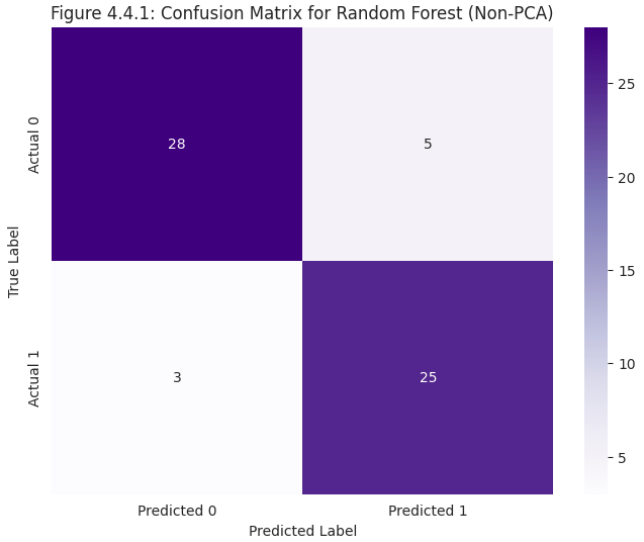
- **PCA Applied:** No
- **Features Used:** All 19 preprocessed features

Results & Analysis
The Random Forest classifier achieved a robust **Accuracy of 86.89%**, marking a significant improvement over the single Decision Tree (80.33%).

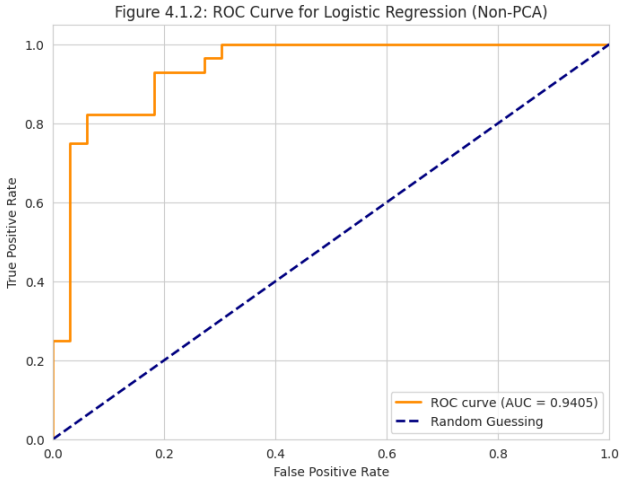
- **Classification Performance:**
  - **Recall (Sensitivity): 0.89** – Correctly identified 89% of disease cases.
  - False Negatives reduced from **8 (Decision Tree)** to **3**, making this model safer for clinical use (*Figure 4.4.1*).
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9524** – Comparable to the KNN model, showing highly reliable probability estimates even if hard classification accuracy is slightly lower.

Visualizations

- **Figure 4.4.1:** Confusion Matrix for Random Forest (Non-PCA)



- **Figure 4.4.2:** ROC Curve for Random Forest (Non-PCA)



4.5 Artificial Neural Network (ANN)

Theory & Implementation
An Artificial Neural Network (ANN) is a **computational model inspired by biological neural networks**. It consists of interconnected layers of nodes ("neurons") that process information using non-linear activation functions.

- **Architecture:**
  - Multi-Layer Perceptron (MLP) with:
    - Input layer: 19 features
    - Hidden Layer 1: 16 neurons, ReLU activation
    - Hidden Layer 2: 8 neurons, ReLU activation
    - Output Layer: 1 neuron, Sigmoid activation for binary classification

- **Implementation:** Trained using the `adam` optimizer with a maximum of **1000 iterations** to ensure convergence.

Experimental Setup

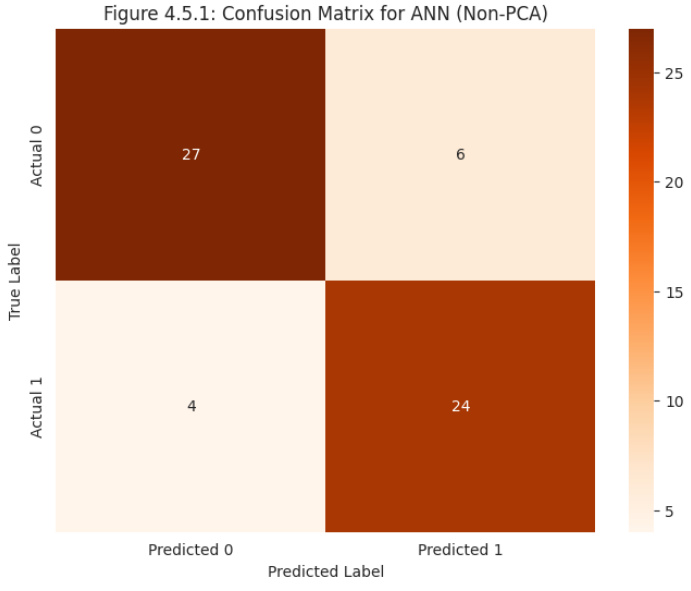
- **PCA Applied:** No
- **Features Used:** All 19 preprocessed features (scaled)

Results & Analysis
The Artificial Neural Network achieved an **Accuracy of 83.61%**, matching the performance of the baseline Logistic Regression model.

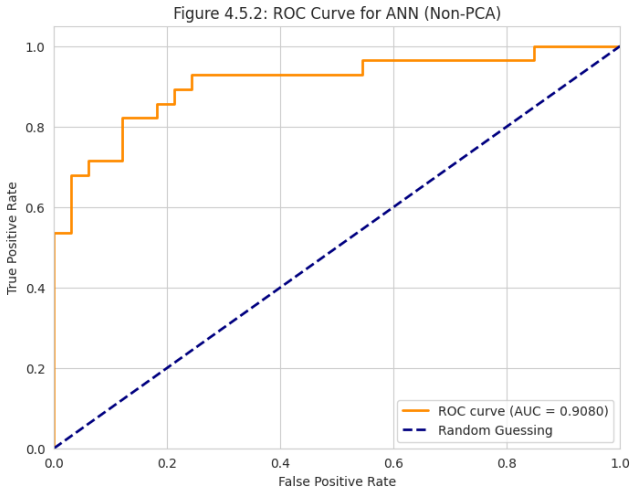
- **Classification Performance:**
  - **Recall (Sensitivity): 0.86** – Correctly identified 86% of disease cases.
  - **False Negatives:** 4 patients misclassified as healthy (*Figure 4.5.1*).
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9080** – Good separability between classes, though slightly below Random Forest (0.9524) and KNN (0.9518).
- **Interpretation:**
  - The ANN matched, but did not exceed, the linear baseline. This suggests that, without dimensionality reduction, the network may have struggled to extract additional non-linear patterns from the limited and potentially noisy feature set.

Visualizations

- **Figure 4.5.1:** Confusion Matrix for ANN (Non-PCA)



- **Figure 4.5.2:** ROC Curve for ANN (Non-PCA)



5. Methodology After Applying PCA

5.1 Logistic Regression (with PCA)

Theory & Implementation
Logistic Regression was re-evaluated using the transformed feature space generated by **Principal Component Analysis (PCA)**. The model operates on the orthogonal components rather than the original features, testing its ability to maintain performance when dimensionality is reduced to **15 components** (explaining 95% of variance).

- **Implementation:** The same liblinear solver and hyperparameters (random_state=42) were used to ensure a direct comparison with the baseline model.

Experimental Setup

- **PCA Applied:** Yes (15 Principal Components)
- **Features Used:** Transformed PCA components derived from scaled data

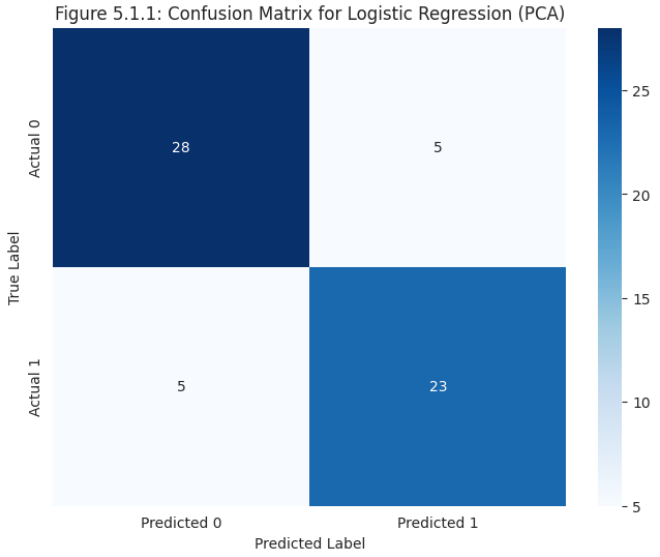
Results & Analysis
The Logistic Regression model with PCA achieved an **Accuracy of 83.61%**, identical to the performance of the model using the full feature set.

- **Classification Performance:**
  - **Recall (Sensitivity): 0.82** – Correctly identified 82% of positive cases.
  - **Precision: 0.82** – Balanced confidence in positive predictions.
  - The **confusion matrix** (*Figure 5.1.1*) shows a slight redistribution of errors compared to the non-PCA version, balancing the **false positives and false negatives (5 each)**.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9394** – Virtually the same as the non-PCA model (0.9405), indicating that performance was maintained.

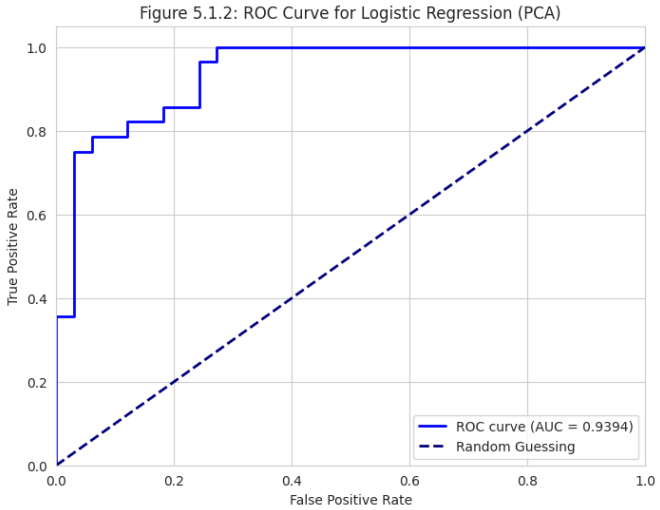
- **Interpretation:**
    - The 15 principal components successfully captured the essential variance required for the linear decision boundary, confirming that the removed dimensions were largely non-informative noise for this algorithm.

Visualizations

- **Figure 5.1.1:** Confusion Matrix for Logistic Regression (PCA)



- **Figure 5.1.2:** ROC Curve for Logistic Regression (PCA)



5.2 K-Nearest Neighbors (with PCA)

Theory & Implementation

The K-Nearest Neighbors (KNN) classifier was applied to the reduced **15-dimensional feature space** obtained through **Principal Component Analysis (PCA)**. Since KNN relies on **Euclidean distance** between data points to determine class membership, this experiment

evaluated whether PCA's denoising effect would improve neighborhood clarity or whether dimensionality reduction would obscure critical local structures.

- **Implementation:** The model retained the same configuration as the non-PCA phase, using `n_neighbors=5` with the **Minkowski distance metric**, ensuring a fair comparison.

Experimental Setup

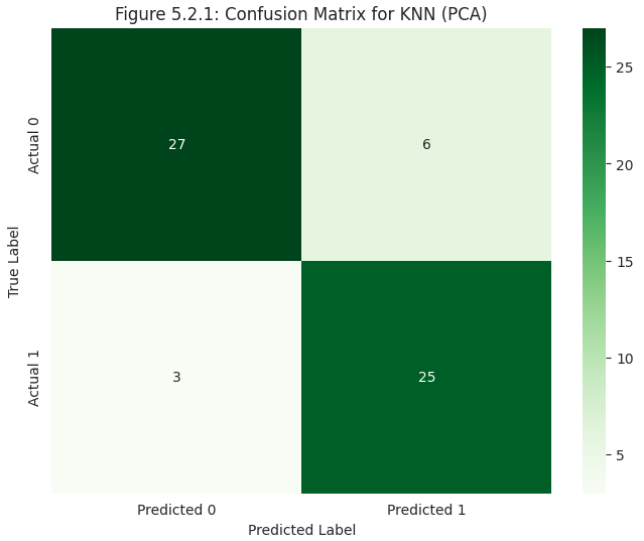
- **PCA Applied:** Yes (15 Principal Components)
- **Features Used:** Transformed PCA components derived from scaled data

Results & Analysis
The KNN model with PCA achieved an **Accuracy of 85.25%**, representing a noticeable decline from the **91.80%** accuracy obtained using the full feature set.

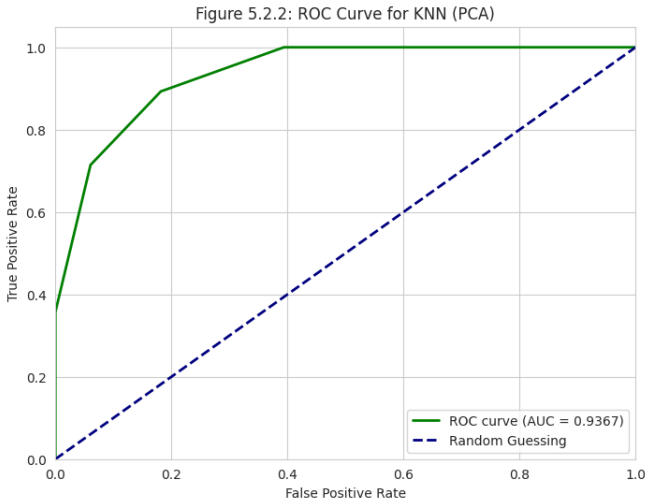
- **Classification Performance:**
  - **Recall (Sensitivity): 0.89** – Lower than the non-PCA model (0.96).
  - **False Negatives:** Increased to **3 cases** (*Figure 5.2.1*), indicating that PCA removed critical spatial information required to correctly classify certain borderline patients.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9367** – Slightly reduced compared to the non-PCA score of 0.9518.
- **Interpretation:**
  - While PCA preserves global variance, KNN depends heavily on **local, fine-grained feature relationships**. Dimensionality reduction appears to have removed subtle but crucial details that define precise neighborhood boundaries between healthy and diseased patients.

Visualizations

- **Figure 5.2.1:** Confusion Matrix for KNN (PCA)



- **Figure 5.2.2:** ROC Curve for KNN (PCA)



5.3 Decision Tree (with PCA)

Theory & Implementation
The Decision Tree classifier was trained on the **15 principal components** to evaluate how orthogonal transformations affect a rule-based model. Since Decision Trees create **axis-aligned decision boundaries**, rotating the feature space through PCA fundamentally alters how features align with the axes used for splitting.

- **Implementation:** The model used the same `random_state=42` to ensure that any performance differences were attributable to the transformed input data rather than initialization variance.

Experimental Setup

- **PCA Applied:** Yes (15 Principal Components)
- **Features Used:** Transformed PCA components

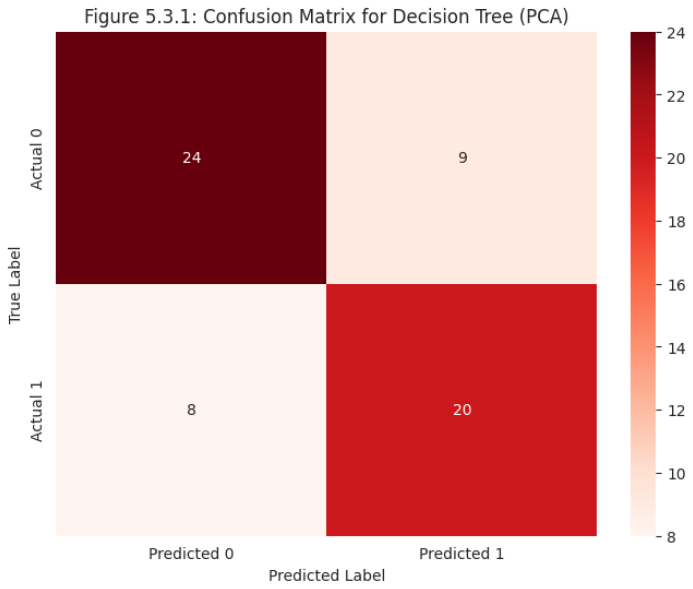
Results & Analysis
The Decision Tree with PCA yielded an **Accuracy of 72.13%**, representing a substantial decline from the **80.33%** achieved using the original feature space.

- **Classification Performance:**
  - **Recall (Sensitivity): 0.71** – Identified only 71% of disease cases.
  - **Precision: 0.69** – Reduced reliability in positive predictions.
  - The **confusion matrix** (*Figure 5.3.1*) shows **17 total misclassifications**, indicating difficulty in forming effective decision rules within the PCA-transformed space.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.7208** – Confirms the model's diminished ability to distinguish between healthy and diseased patients.
- **Interpretation:**

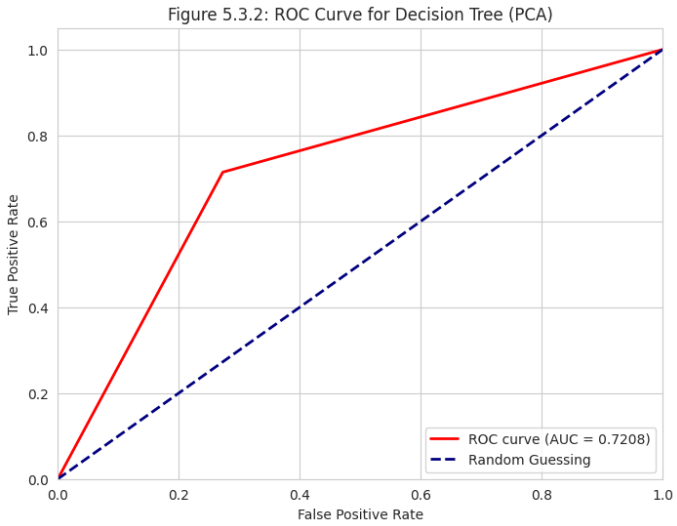
- This outcome highlights a fundamental incompatibility between **PCA and single Decision Trees** for this dataset. PCA generates abstract linear combinations of features (e.g., *0.5 × Age + 0.3 × Cholesterol*), while Decision Trees split on one variable at a time. As a result, the tree struggles to approximate the **diagonal decision boundaries** required in PCA space, leading to deeper, less efficient, and poorly generalizing trees.

Visualizations

- **Figure 5.3.1:** Confusion Matrix for Decision Tree (PCA)



- **Figure 5.3.2:** ROC Curve for Decision Tree (PCA)



5.4 Random Forest (with PCA)

Theory & Implementation
The Random Forest classifier, an **ensemble of 100 decision trees**, was applied to the **15 principal components** to evaluate whether the ensemble voting mechanism could mitigate the challenges individual decision trees face when operating on orthogonal PCA-transformed features.

- **Implementation:** The model retained the same configuration (n_estimators=100, random_state=42) to isolate the impact of dimensionality reduction.

Experimental Setup

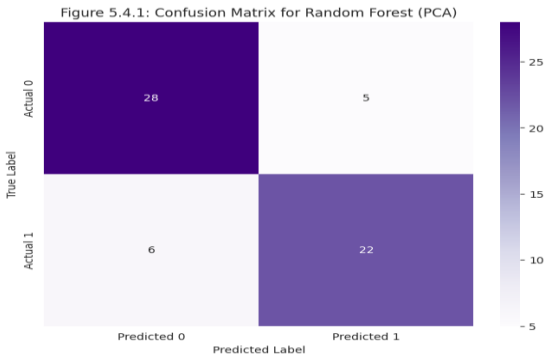
- **PCA Applied:** Yes (15 Principal Components)
- **Features Used:** Transformed PCA components

Results & Analysis
The Random Forest model with PCA achieved an **Accuracy of 81.97%**, representing a clear decrease from the **86.89%** obtained using the original feature set.

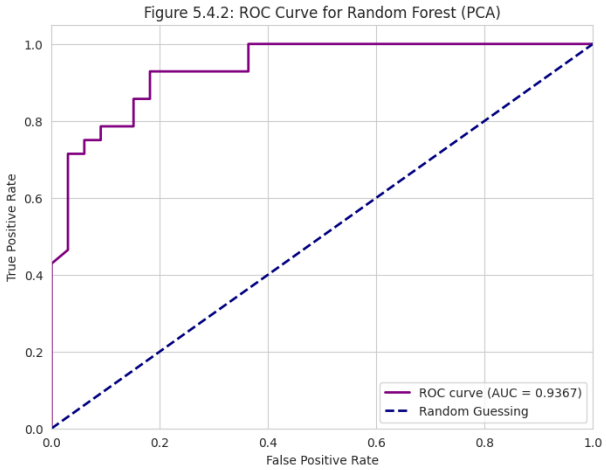
- **Classification Performance:**
  - **Recall (Sensitivity): 0.79** – Reduced from 0.89 in the non-PCA model.
  - **False Negatives:** Increased from **3 to 6**, as shown in *Figure 5.4.1*.
  - While outperforming a single Decision Tree with PCA, the ensemble was still adversely affected by feature space rotation.
- **Discriminative Ability:**
  - **ROC-AUC Score: 0.9367** – Remained strong despite the drop in accuracy, indicating reliable probability ranking even though the default classification threshold led to more misclassifications.
- **Interpretation:**
  - Random Forests rely on **axis-aligned splits**, similar to single Decision Trees. PCA rotates the feature space by creating linear combinations of original features, forcing trees to approximate **diagonal decision boundaries** using a staircase-like sequence of splits. This inefficiency explains the observed reduction in performance.

Visualizations

- **Figure 5.4.1:** Confusion Matrix for Random Forest (PCA)



- **Figure 5.4.2:** ROC Curve for Random Forest (PCA)



5.5 Artificial Neural Network (with PCA)

Theory & Implementation
The Multi-Layer Perceptron (ANN) was trained on the **15 principal components** to evaluate the effect of dimensionality reduction on neural network convergence and generalization. Neural networks are often sensitive to the **curse of dimensionality** and noisy input features; therefore, this experiment assessed whether compressing information into orthogonal components would enable the network to learn more robust patterns.

- **Implementation:** The network architecture (**16 and 8 neurons in the hidden layers**) and optimizer (`adam`) were kept identical to the non-PCA implementation to ensure a fair comparison.

Experimental Setup

- **PCA Applied:** Yes (15 Principal Components)
- **Features Used:** Transformed PCA components

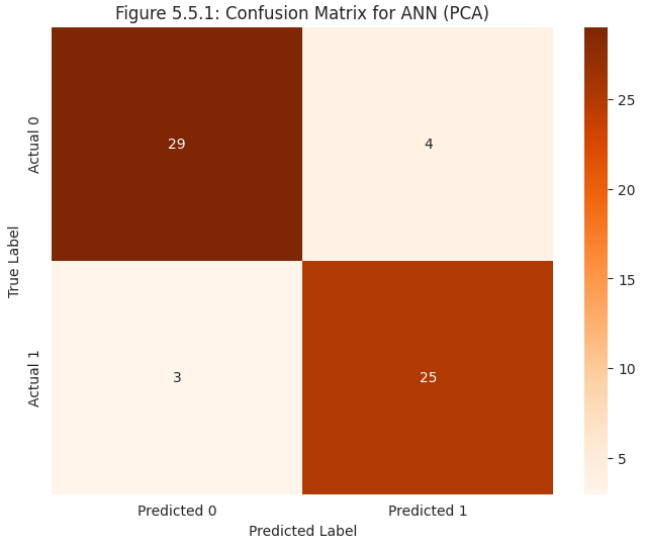
Results & Analysis
The Artificial Neural Network with PCA achieved an **Accuracy of 88.52%**, representing a substantial improvement over the **83.61%** accuracy obtained using the full feature set.

- **Classification Performance:**
  - **Recall (Sensitivity): 0.89** – Improved from 0.86 in the non-PCA model.
  - **False Negatives:** Reduced to **3 cases**, as illustrated in *Figure 5.5.1*.
  - These results indicate that PCA effectively removed noisy and redundant features, allowing the ANN to focus on the most informative variance for classification.
- **Discriminative Ability:**

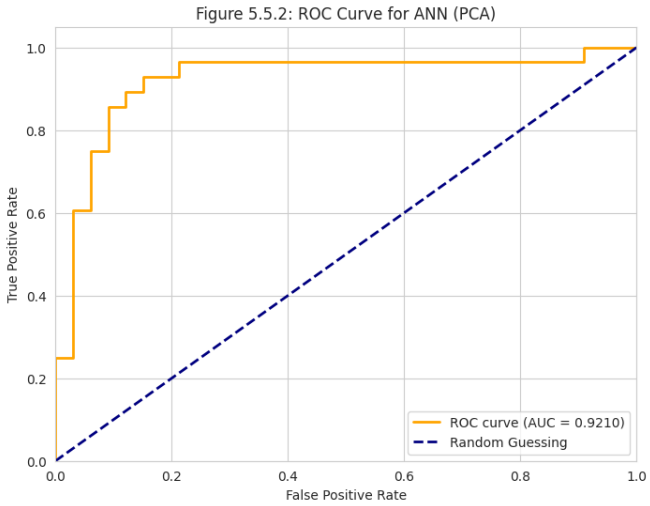
    - **ROC-AUC Score: 0.9210** – Increased from 0.9080, confirming improved probability ranking after dimensionality reduction.
- **Interpretation:**
    - This outcome highlights the advantage of combining **PCA with Neural Networks** for small-to-medium sized datasets. PCA acted as a form of **regularization**, improving generalization by reducing feature redundancy and noise, enabling the ANN to outperform its non-PCA counterpart.

Visualizations

- **Figure 5.5.1:** Confusion Matrix for ANN (PCA)



- **Figure 5.5.2:** ROC Curve for ANN (PCA)



6. Results & Comparison

To systematically evaluate the impact of dimensionality reduction, the performance of all five classifiers was compared under both experimental conditions: The **Original Feature Set** and the **PCA-Transformed Feature Set**. The comparative results are presented in **Table 6.1**.

Table 6.1: Comparative Analysis of Model Accuracy

Model	Accuracy (Original Features)	Accuracy (PCA Features)	Performance Change
Logistic Regression	83.61%	83.61%	No Change
K-Nearest Neighbors (KNN)	**91.80%**	85.25%	Decrease
Decision Tree	80.33%	72.13%	Significant Decrease
Random Forest	86.89%	81.97%	Decrease
Artificial Neural Network (ANN)	83.61%	**88.52%**	**Increase**

Key Observations

1. **Top Performing Model**
   The **K-Nearest Neighbors (KNN)** classifier trained on the original 19 features achieved the highest overall accuracy of **91.80%**, along with the highest recall value (**0.96**). This makes KNN the most effective model for this dataset.
2. **Impact of PCA**
   Principal Component Analysis produced mixed effects across models. PCA caused notable performance degradation in **tree-based models** (Decision Tree and Random Forest) and the **distance-based KNN** algorithm. In contrast, PCA had **no effect** on Logistic Regression and a **positive effect** on the Artificial Neural Network.
3. **Ensemble Model Robustness**
   Although Random Forest consistently outperformed the single Decision Tree in both settings, it was still affected by PCA-induced information loss, experiencing an accuracy reduction of approximately **5%**.

7. Discussion

7.1 Analysis of the Best Performing Model (KNN)

The superior performance of the K-Nearest Neighbors classifier (91.80% accuracy) indicates that the Cleveland Heart Disease dataset contains well-defined local clusters of patients with similar clinical characteristics. Since KNN relies on local neighborhood distances rather than global decision boundaries, it effectively captured these clusters.

Despite the relatively high dimensionality of the dataset (19 features), KNN did not suffer significantly from the curse of dimensionality. This is likely due to the application of **standard scaling during preprocessing**, which ensured that all features contributed proportionally to distance calculations and prevented dominance by any single attribute.

7.2 Effect of PCA on Tree-Based Models

The most pronounced performance degradation following PCA was observed in the **Decision Tree (−8.2%)** and **Random Forest (−4.92%)** models. This behavior can be explained by the fundamental working principles of tree-based algorithms.

PCA transforms original features into linear combinations of variables (e.g., combinations of Age, Blood Pressure, and Cholesterol). Decision Trees, however, rely on **axis-aligned splits** such as *Cholesterol > 200*. When PCA rotates the feature space, these clean, single-feature thresholds are no longer available. Consequently, the trees must approximate diagonal decision boundaries using complex and inefficient structures, leading to poorer generalization performance.

7.3 Success of ANN with PCA

In contrast to other classifiers, the **Artificial Neural Network (ANN)** demonstrated a substantial performance improvement (**+4.91%**) when trained on PCA-transformed features. Neural networks are often sensitive to noisy and redundant inputs, especially when trained on small datasets.

By compressing the original features into **15 orthogonal principal components**, PCA effectively removed redundancy and noise while preserving the most informative variance. This simplified the optimization landscape for the ANN, enabling more stable convergence and improved generalization compared to training on raw features.

8. Conclusion

This study evaluated the performance of five machine learning algorithms for the binary classification of heart disease. Among all evaluated models, **K-Nearest Neighbors (without PCA)** emerged as the most effective, achieving an accuracy of **91.80%** and a recall of **0.96**. High recall is particularly critical in medical diagnostics, where false negatives can have severe clinical consequences.

The application of **Principal Component Analysis (PCA)** yielded mixed outcomes. While PCA successfully reduced dimensionality, the associated loss of feature-specific information negatively impacted tree-based and distance-based models. However, PCA proved to be a valuable preprocessing technique for **Artificial Neural Networks**, significantly enhancing their performance on a small medical dataset.