



**Islington college**  
(इस्लिङ्टन कलेज)

## **Module Code & Module Title**

**CU6051NI Artificial Intelligence**

**25% Individual Coursework**

**Submission: Proposal**

**Academic Semester: Autumn Semester 2025**

**Credit: 15 credit semester long module**

**Student Name: Akshat Raj Dali**

**London Met ID: 23049073**

**College ID: NP01CP4A230091**

**Assignment Due Date: 07/01/2026.**

**Assignment Submission Date: 06/01/2026**

**Submitted To: Mr. Binod Bhattarai**

GitHub Link	
-------------	--

*I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

# Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
1.1 Explanations of the Topic/ AI Concepts Used.....	1
1.2 Explanation of the Chosen Problem Domain.....	1
<b>2. Background .....</b>	<b>2</b>
2.1 Research Work done in Coursework 1 .....	2
<b>3. Solution .....</b>	<b>2</b>
3.1 Explanation of the Solution/ AI Algorithms Used .....	2
3.2. Pseudocodes of the solution .....	3
3.2.1 Algorithm 1: Logistic Regression .....	3
3.2.2 Algorithm 2: Support Vector Machine (SVM) .....	4
3.2.3 Algorithm 3: Random Forest .....	5
3.2.4 Overall Workflow Pseudocode.....	6
3.3 Flowcharts .....	8
3.3.1 Logistic Regression .....	8
3.3.2 Support Vector Machine (SVM) .....	9
3.3.3 Random Forest.....	10
3.3.4 Overall Flowchart.....	11
3.4 Explanation of the Development Process .....	12
3.5 Achieved Results .....	12
3.5.1 Importing Machine and Required Libraries .....	12
3.5.2 Loading the Dataset .....	13
3.5.3 Feature and Target Separation.....	13
3.5.4 Splitting the Dataset .....	13
3.5.5 Feature Scaling (Data Preprocessing) .....	14
3.5.6 Logistic Regression.....	14

3.5.7 Support Vector Machine .....	14
3.5.8 Random Forest Modeling .....	15
3.5.9 Model Evaluation Function .....	15
3.5.10 Model Performance Evaluation.....	16
3.5.11 Accuracy Comparison Visualization.....	17
3.5.12 Confusion Matrix Visualization .....	18
3.5.13 Histogram of age distribution .....	19
3.5.14 Scatter Plot of Age vs Maximum Heart Rate .....	20
3.5.15 Scatter Plot of Cholesterol vs Maximum Heart Rate .....	20
3.5.16 Heatmap of Correlation Matrix of Numerical Data.....	21
3.5.17 Bar Chart of Heart Disease Class Distribution .....	22
3.5.18 K-Means Clustering .....	23
 4. Conclusion .....	 24
4.1 Analysis of the Work Done.....	24
4.2 Real-World Application.....	24
4.3 Further Work .....	24
 17. References.....	 26

## Table of Figures

Figure 1: Logistic Regression Flowchart .....	8
Figure 2: SVM Flowchart.....	9
Figure 3: Random Forest Flowchart.....	10
Figure 4: Overall Flowchart .....	11
Figure 5: Importing Libraries .....	12
Figure 6: Loading the dataset.....	13
Figure 7: Feature and target separation .....	13
Figure 8: Splitting the dataset.....	13
Figure 9: Feature scaling.....	14
Figure 10: Logistic Regression .....	14
Figure 11: Support Vector Machine.....	14
Figure 12: Random Forest Modeling .....	15
Figure 13: Model Evaluation Function .....	15
Figure 14: Model Performance Evaluation .....	16
Figure 15: Model Accuracy Comparison .....	17
Figure 16: Confusion Matrix .....	18
Figure 17: Histogram of age distribution.....	19
Figure 18: Scatter plot of age and max heart rate .....	20
Figure 19: Scatter plot of cholesterol vs max heart rate .....	20
Figure 20: Heatmap of correlation matrix of numerical data.....	21
Figure 21: Bar chart of heart disease class distribution.....	22
Figure 22: K-Means clustering .....	23

# **1. Introduction**

## **1.1 Explanations of the Topic/ AI Concepts Used**

Artificial Intelligence (AI) is an area of computer science that focuses on creating systems capable of performing tasks that normally require human intelligence. These tasks include learning from data, recognizing patterns, and making decisions. One of the most important and widely used branches of AI is Machine Learning (ML). Machine learning allows computers to learn from past data and improve their performance over time without being explicitly programmed for every possible situation.

This project makes use of supervised machine learning, where models are trained using datasets that contain both input features and known output labels. In this case, the output label indicates whether a patient has heart disease or not. Since the output has only two possible values (presence or absence of disease), the problem is treated as a binary classification problem.

Machine learning techniques are increasingly being used in healthcare because they can process large amounts of data quickly and identify patterns that may not be obvious through manual analysis. In this project, three supervised learning algorithms are used: Logistic Regression, Support Vector Machine (SVM), and Random Forest. These algorithms were selected because they are commonly applied in medical prediction tasks and have proven effectiveness in classification problems.

## **1.2 Explanation of the Chosen Problem Domain**

Heart disease remains one of the most serious health issues worldwide and is a leading cause of death in many countries. Detecting heart disease at an early stage can help reduce severe complications and improve patient survival rates. However, traditional diagnosis methods often depend on manual interpretation of medical reports, which can be time-consuming and may sometimes lead to human error.

With the growing availability of healthcare data, machine learning techniques can be applied to analyze patient information such as age, cholesterol level, blood pressure, and heart rate. By identifying patterns within this data, machine learning models can help predict the likelihood of heart disease. This project focuses on developing a prediction system that can support healthcare professionals by providing faster and more accurate risk assessment.

## 2. Background

### 2.1 Research Work done in Coursework 1

The dataset used in this project was obtained from the UCI Machine Learning Repository. It contains real medical records with 13 clinical features and a target variable indicating the presence or absence of heart disease. This dataset is widely used in academic studies and provides a suitable foundation for comparing different machine learning algorithms.

Several studies have explored the application of machine learning in medical diagnosis:

- Kumar and Singh (2021) compared SVM and KNN models for medical classification tasks and concluded that SVM performs better on high-dimensional datasets, *(Singh, 2021)*
- Patel et al. (2022) demonstrated that ensemble learning techniques such as Random Forest improve prediction accuracy and reduce overfitting in healthcare datasets. *(Patel, 2022)*
- Alam (2020) applied Logistic Regression for heart disease prediction and found it to be effective as a simple and interpretable baseline model. *(Alam, 2020)*
- Chen and Zhao (2023) emphasized the importance of feature selection in improving classification accuracy for medical datasets. *(Chen, 2023)*
- Rahman et al. (2021) highlighted that machine learning-based decision support systems improve diagnostic efficiency and reliability in clinical environments. *(Rahman, 2021)*

## 3. Solution

### 3.1 Explanation of the Solution/ AI Algorithms Used

The proposed solution focuses on developing a machine learning based application that can predict the presence of heart disease by analyzing patient health data. Instead of relying on a single model, multiple supervised learning algorithms were implemented and tested. This approach allows a fair comparison between different techniques and helps identify which model performs best for this specific problem. Using more than one algorithm also reduces the risk of biased results and provides a better overall understanding of how different models behave on medical data.

Logistic Regression was selected as the baseline model because of its simplicity and ease

of interpretation. It is widely used in medical studies since its output can be easily understood and explained, which is important in healthcare-related applications. Support Vector Machine (SVM) was chosen as the second model due to its strong performance when handling high dimensional and complex datasets. SVM is effective at finding optimal decision boundaries, making it suitable for classification problems such as heart disease prediction. Random Forest was included as an ensemble based algorithm that combines multiple decision trees to produce more accurate and stable predictions. By averaging the results of several trees, Random Forest helps reduce overfitting and captures non-linear relationships between medical features.

All selected models were trained using historical patient records obtained from the dataset. The data was divided into training and testing sets to ensure that model performance was evaluated on unseen data. Each model was assessed using standard classification performance metrics, including accuracy, precision, recall, F1-score, and ROC-AUC. This evaluation approach provides a comprehensive understanding of each model's strengths and weaknesses, allowing the most suitable algorithm for heart disease prediction to be identified.

## **3.2. Pseudocodes of the solution**

### **3.2.1 Algorithm 1: Logistic Regression**

BEGIN Logistic Regression Prediction

LOAD heart disease dataset

SEPARATE dataset into input features (X) and target variable (y)

SPLIT X and y into training data and testing data

APPLY feature scaling to training and testing data

INITIALIZE Logistic Regression model

TRAIN Logistic Regression model using training data

USE trained model to predict outcomes on testing data

COMPARE predicted values with actual values

CALCULATE performance metrics:

- accuracy

- precision

- recall

- F1-score

- ROC-AUC

DISPLAY evaluation results

END Logistic Regression Prediction

### **3.2.2 Algorithm 2: Support Vector Machine (SVM)**

BEGIN SVM Prediction

LOAD heart disease dataset

SEPARATE dataset into input features (X) and target variable (y)

SPLIT data into training and testing sets

APPLY feature scaling to normalize the data

SELECT appropriate SVM kernel function

INITIALIZE SVM model

TRAIN SVM model using training data



PREDICT heart disease outcomes using test data

EVALUATE predictions using:

- accuracy

- precision

- recall

- F1-score

- ROC-AUC

DISPLAY evaluation results

END SVM Prediction

### **3.2.3 Algorithm 3: Random Forest**

BEGIN Random Forest Prediction

LOAD heart disease dataset

SEPARATE input features and target variable

SPLIT dataset into training and testing sets

INITIALIZE Random Forest with multiple decision trees

FOR each decision tree in the forest DO

- SELECT random subset of data

- TRAIN decision tree

END FOR

COMBINE predictions from all decision trees

DETERMINE final prediction using majority voting

EVALUATE model performance using:

- accuracy
- precision
- recall
- F1-score
- ROC-AUC

DISPLAY evaluation results

END Random Forest Prediction

### **3.2.4 Overall Workflow Pseudocode**

BEGIN

LOAD heart disease dataset from CSV file

DISPLAY dataset structure and check for missing values

IF missing values exist THEN

HANDLE missing values using appropriate method

ENDIF

SEPARATE dataset into:

- Input features (X)
- Target variable (y)

SPLIT dataset into training set and testing set

- 80% training data
- 20% testing data

APPLY feature scaling on training and testing data

```
// Logistic Regression Model
INITIALIZE Logistic Regression model
TRAIN Logistic Regression using training data
PREDICT heart disease outcomes using test data
CALCULATE evaluation metrics:
    accuracy, precision, recall, F1-score, ROC-AUC

// Support Vector Machine Model
INITIALIZE SVM model with suitable kernel
TRAIN SVM using training data
PREDICT heart disease outcomes using test data
CALCULATE evaluation metrics:
    accuracy, precision, recall, F1-score, ROC-AUC

// Random Forest Model
INITIALIZE Random Forest model with number of trees
TRAIN Random Forest using training data
PREDICT heart disease outcomes using test data
CALCULATE evaluation metrics:
    accuracy, precision, recall, F1-score, ROC-AUC

COMPARE performance results of all models

IDENTIFY model with best overall performance

DISPLAY final results and visual outputs

END
```

### 3.3 Flowcharts

#### 3.3.1 Logistic Regression

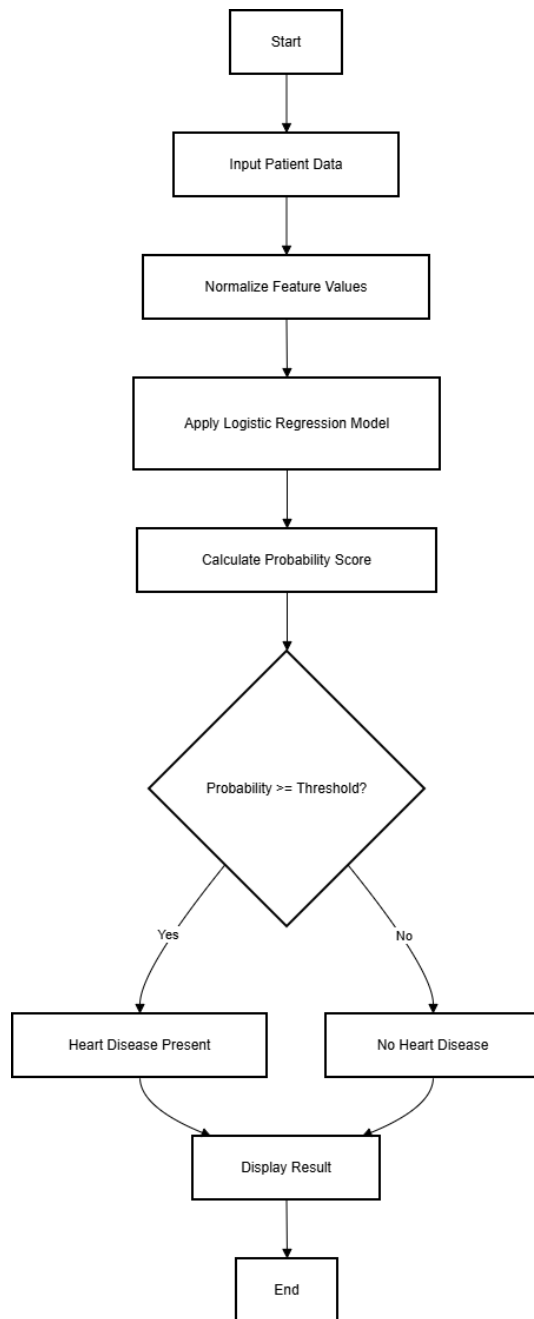


Figure 1: Logistic Regression Flowchart

### 3.3.2 Support Vector Machine (SVM)

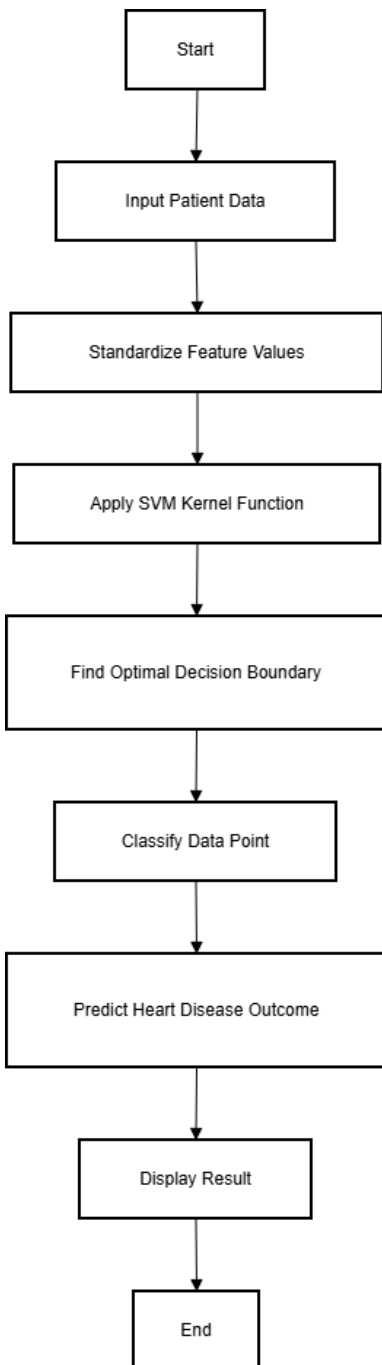


Figure 2: SVM Flowchart

### 3.3.3 Random Forest

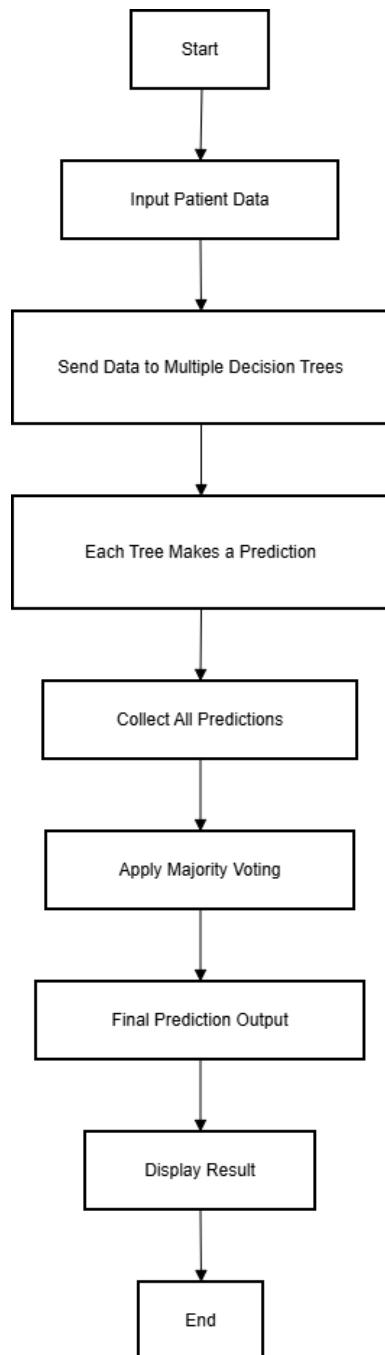


Figure 3: Random Forest Flowchart

### 3.3.4 Overall Flowchart

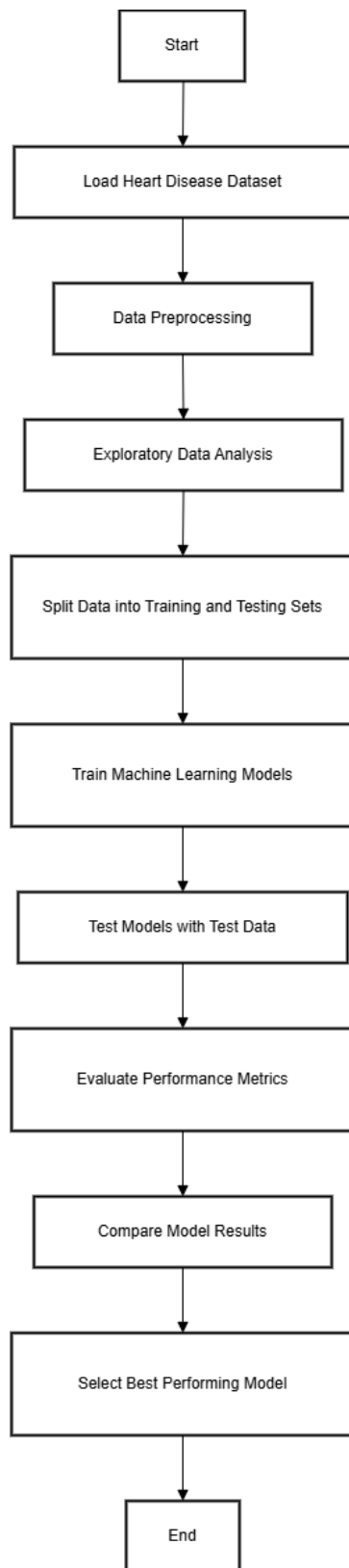


Figure 4: Overall Flowchart

### 3.4 Explanation of the Development Process

The application was developed using Python due to its strong support for machine learning and data analysis. The development process followed a step-by-step approach to ensure clarity and correctness.

First, the dataset was loaded and examined to understand the features and target variable. The data was then preprocessed by separating input features and output labels. The dataset was split into training and testing sets to allow proper evaluation of the models.

Feature scaling was applied to standardize the data, especially because algorithms such as Logistic Regression and SVM are sensitive to differences in feature values. After preprocessing, the selected machine learning models were trained using the training data and tested on unseen data.

The performance of each model was evaluated using multiple metrics rather than relying on accuracy alone. This helped provide a more balanced and realistic evaluation, which is important in healthcare-related applications.

### 3.5 Achieved Results

After training and evaluating all three models, it was observed that each algorithm performed reasonably well. Logistic Regression provided stable and interpretable results, making it a useful baseline model. SVM showed improved performance due to its ability to handle complex patterns in the data.

Random Forest achieved the highest overall performance across most evaluation metrics. Its ensemble approach helped reduce prediction errors and improved reliability.

#### 3.5.1 Importing Machine and Required Libraries

```
[4]: import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix
```

*Figure 5: Importing Libraries*



These libraries are used for data handling and visualization. The Scikit-learn library provides tools for building and evaluating machine learning models.

### 3.5.2 Loading the Dataset

```
[5]: data = pd.read_csv("heart-disease-dataset.csv")
data.head()
```

	age	sex	chest_pain_type	resting_bp_s	cholesterol	fasting_blood_sugar	resting_ecg	max_heart_rate	exercise_angina	oldpeak	st_slope	target
0	40.0	1.0	2.0	140.0	289.0	0.0	0.0	172.0	0.0	0.0	1.0	0.0
1	49.0	0.0	3.0	160.0	180.0	0.0	0.0	156.0	0.0	1.0	2.0	1.0
2	37.0	1.0	2.0	130.0	283.0	0.0	1.0	98.0	0.0	0.0	1.0	0.0
3	48.0	0.0	4.0	138.0	214.0	0.0	0.0	108.0	1.0	1.5	2.0	1.0
4	54.0	1.0	3.0	150.0	195.0	0.0	0.0	122.0	0.0	0.0	1.0	0.0

Figure 6: Loading the dataset

The heart disease dataset is loaded from a CSV file. This dataset contains patient medical attributes and a target variable indicating the presence or absence of heart disease and is displayed.

### 3.5.3 Feature and Target Separation

```
[6]: X = data.drop("target", axis=1)
y = data["target"]
```

Figure 7: Feature and target separation

- X contains the input features such as age, cholesterol, and blood pressure.
- y contains the target label (0 = No heart disease, 1 = Heart disease)

### 3.5.4 Splitting the Dataset

```
[7]: X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

Figure 8: Splitting the dataset

The dataset is split into:

- 80% training data
- 20% testing data

This ensures that the model is evaluated on unseen data to measure its generalization ability

### 3.5.5 Feature Scaling (Data Preprocessing)

```
[8]: scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

*Figure 9: Feature scaling*

Feature scaling standardizes the data so that all features have the same scale. This is particularly important for algorithms such as Logistic Regression and SVM, which are sensitive to feature magnitudes.

### 3.5.6 Logistic Regression

```
[10]: from sklearn.linear_model import LogisticRegression

lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train, y_train)
lr_pred = lr_model.predict(X_test)
```

*Figure 10: Logistic Regression*

The Logistic Regression model is initialized and trained using the training dataset. After training, predictions are made on the test data. This model acts as a baseline for comparison.

### 3.5.7 Support Vector Machine

```
[14]: from sklearn.svm import SVC

svm_model = SVC(kernel="rbf", probability=True)
svm_model.fit(X_train, y_train)
svm_pred = svm_model.predict(X_test)
```

*Figure 11: Support Vector Machine*

The SVM classifier is trained using the Radial Basis Function (RBF) kernel, which helps capture non-linear relationships. Probability estimation is enabled to allow ROC-AUC calculation.

### 3.5.8 Random Forest Modeling

```
[15]: rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
      rf_model.fit(X_train, y_train)
      rf_pred = rf_model.predict(X_test)
```

*Figure 12: Random Forest Modeling*

The Random Forest model consists of multiple decision trees. Each tree is trained on a random subset of the data, and final predictions are made using majority voting. This improves accuracy and reduces overfitting.

### 3.5.9 Model Evaluation Function

```
[16]: def evaluate_model(name, y_true, y_pred, model, X_test):
      print(name)
      print("Accuracy:", accuracy_score(y_true, y_pred))
      print("Precision:", precision_score(y_true, y_pred))
      print("Recall:", recall_score(y_true, y_pred))
      print("F1 Score:", f1_score(y_true, y_pred))
      print("ROC-AUC:", roc_auc_score(y_true, model.predict_proba(X_test)[: , 1]))
      print("-" * 40)
```

*Figure 13: Model Evaluation Function*

This function evaluates each model using multiple performance metrics:

- Accuracy
- Precision
- Recall
- F1-score
- ROC-AUC

### 3.5.10 Model Performance Evaluation

```
[17]: evaluate_model("Logistic Regression", y_test, lr_pred, lr_model, X_test)
      evaluate_model("Support Vector Machine", y_test, svm_pred, svm_model, X_test)
      evaluate_model("Random Forest", y_test, rf_pred, rf_model, X_test)

Logistic Regression
Accuracy: 0.8613445378151261
Precision: 0.8712121212121212
Recall: 0.8778625954198473
F1 Score: 0.8745247148288974
ROC-AUC: 0.9088249982164515
-----

Support Vector Machine
Accuracy: 0.8907563025210085
Precision: 0.8671328671328671
Recall: 0.9465648854961832
F1 Score: 0.9051094890510949
ROC-AUC: 0.9475636726831704
-----

Random Forest
Accuracy: 0.9453781512605042
Precision: 0.9338235294117647
Recall: 0.9694656488549618
F1 Score: 0.951310861423221
ROC-AUC: 0.9720696297353214
-----
```

*Figure 14: Model Performance Evaluation*

Each trained model is evaluated using the defined evaluation function, and the results are printed for comparison.

### 3.5.11 Accuracy Comparison Visualization

```
[18]: models = ["Logistic Regression", "SVM", "Random Forest"]
      accuracy = [
          accuracy_score(y_test, lr_pred),
          accuracy_score(y_test, svm_pred),
          accuracy_score(y_test, rf_pred)
      ]

      plt.figure()
      plt.bar(models, accuracy)
      plt.title("Model Accuracy Comparison")
      plt.ylabel("Accuracy")
      plt.xlabel("Models")
      plt.show()
```

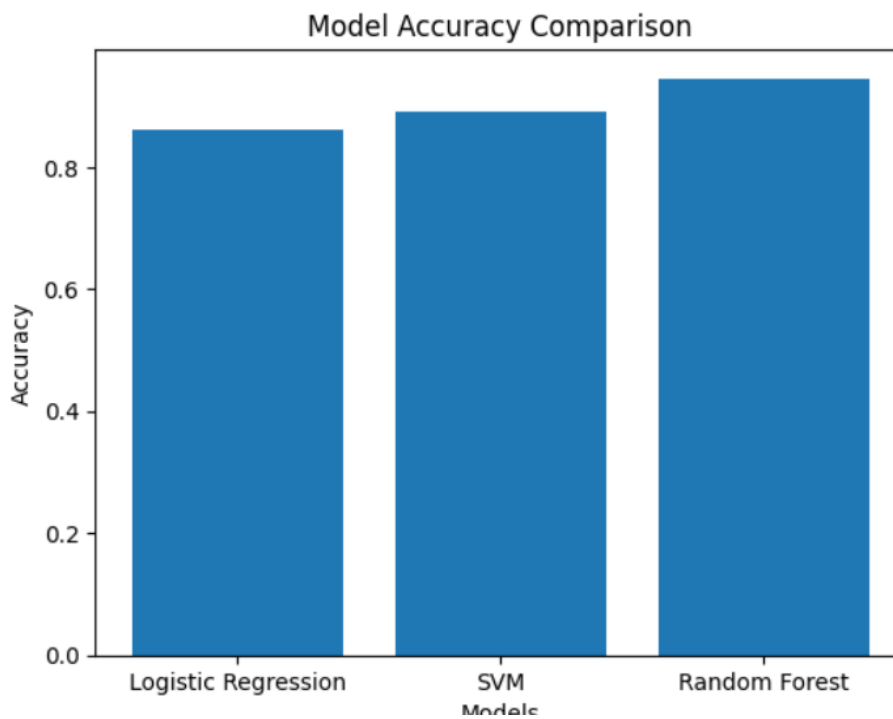


Figure 15: Model Accuracy Comparison

A bar chart is used to visually compare the accuracy of all three models. Visualization makes it easier to identify the best-performing algorithm.

### 3.5.12 Confusion Matrix Visualization

```
[19]: cm = confusion_matrix(y_test, rf_pred)

plt.figure()
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.title("Confusion Matrix - Random Forest")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

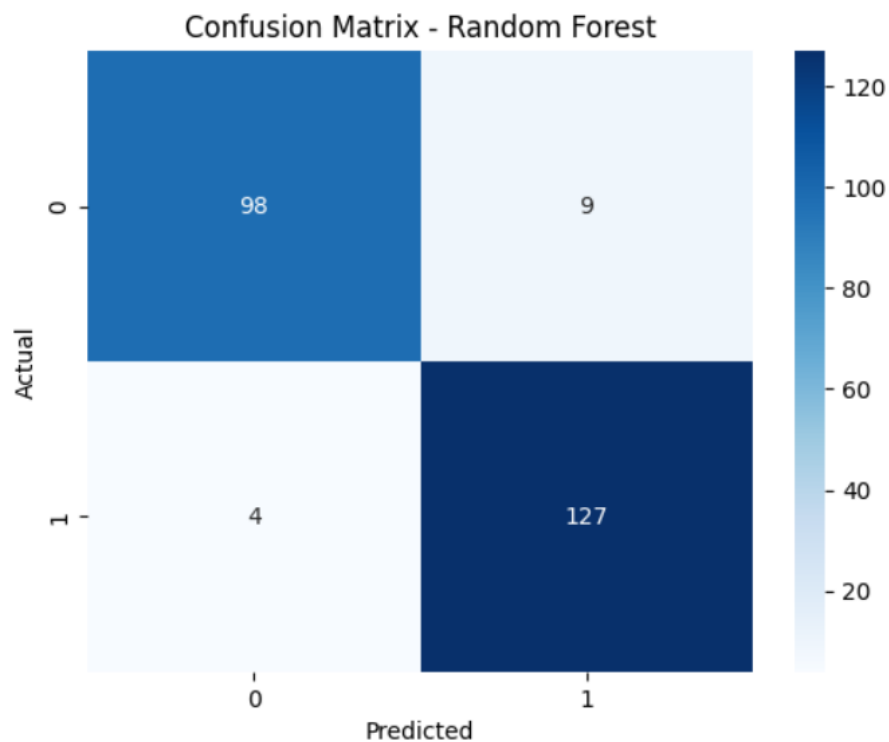
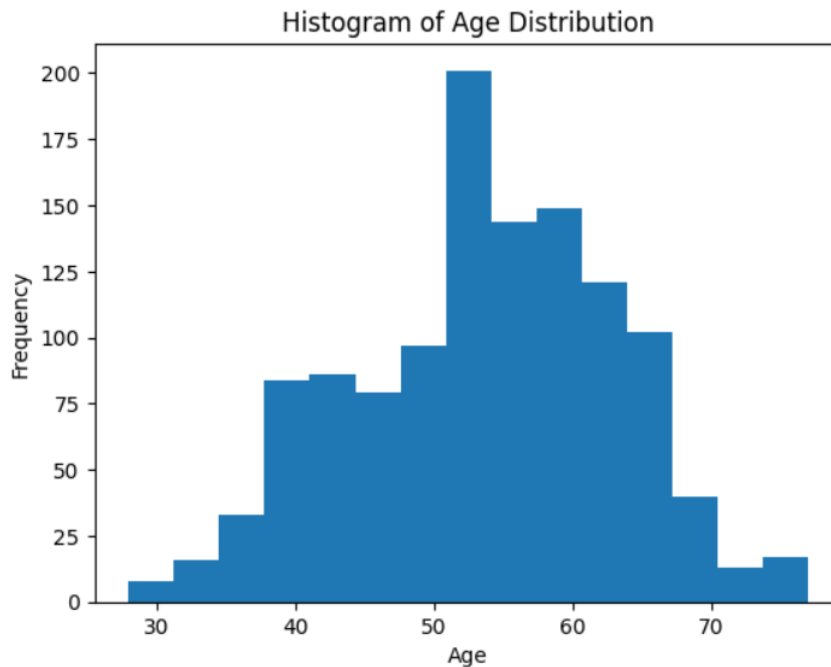


Figure 16: Confusion Matrix

The confusion matrix shows the number of correct and incorrect predictions made by the Random Forest model. It helps analyze false positives and false negatives, which are critical in medical diagnosis.

### 3.5.13 Histogram of age distribution

```
[13]: plt.figure()
plt.hist(data["age"], bins=15)
plt.title("Histogram of Age Distribution")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.show()
```



*Figure 17: Histogram of age distribution*

The histogram above illustrates the distribution of patient ages in the dataset. It helps in understanding the age range most commonly associated with heart disease records. From the visualization, it can be observed that the majority of patients fall within the middle-aged and older age groups, indicating a higher prevalence of heart-related conditions in these age ranges.

### 3.5.14 Scatter Plot of Age vs Maximum Heart Rate

```
[17]: plt.figure()
plt.scatter(data["age"], data["max_heart_rate"])
plt.title("Scatter Plot of Age vs Maximum Heart Rate")
plt.xlabel("Age")
plt.ylabel("Maximum Heart Rate")
plt.show()
```

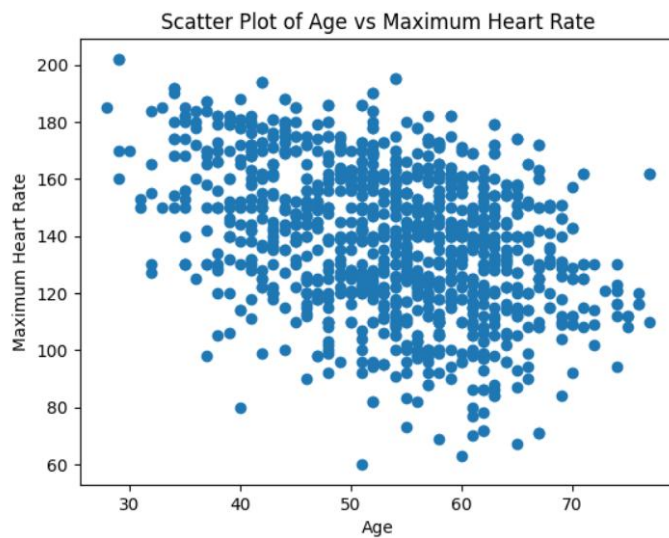


Figure 18: Scatter plot of age and max heart rate

### 3.5.15 Scatter Plot of Cholesterol vs Maximum Heart Rate

```
[18]: plt.figure()
plt.scatter(data["cholesterol"], data["max_heart_rate"])
plt.title("Scatter Plot of Cholesterol vs Maximum Heart Rate")
plt.xlabel("Cholesterol Level")
plt.ylabel("Maximum Heart Rate")
plt.show()
```

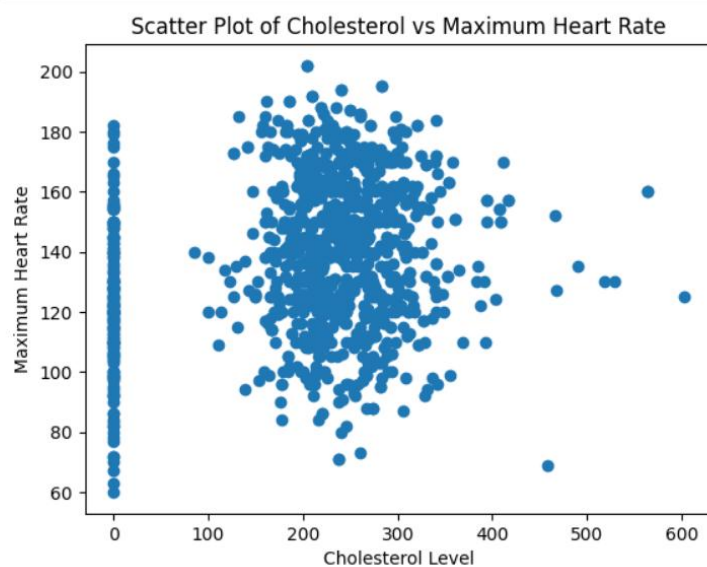


Figure 19: Scatter plot of cholesterol vs max heart rate



### 3.5.16 Heatmap of Correlation Matrix of Numerical Data

```
[19]: plt.figure(figsize=(10,8))
      corr_matrix = data.corr()

      sns.heatmap(corr_matrix, annot=True, cmap="coolwarm")
      plt.title("Correlation Heatmap of Heart Disease Dataset")
      plt.show()
```

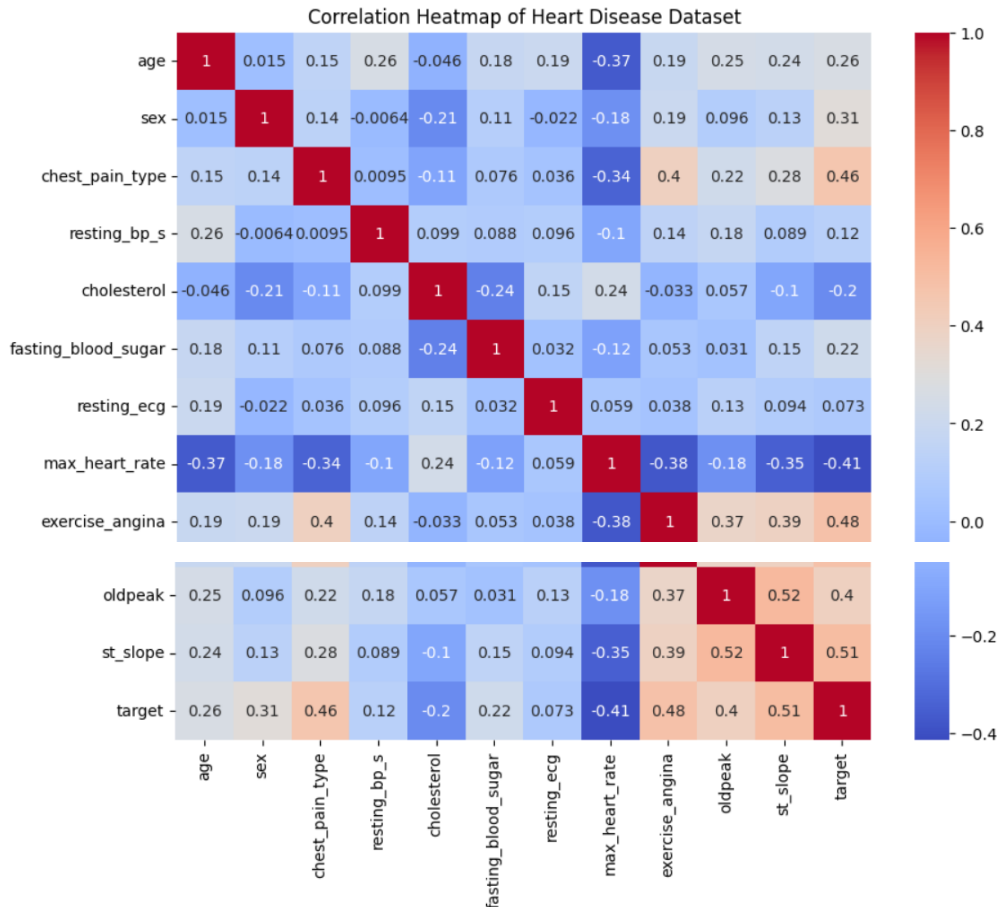
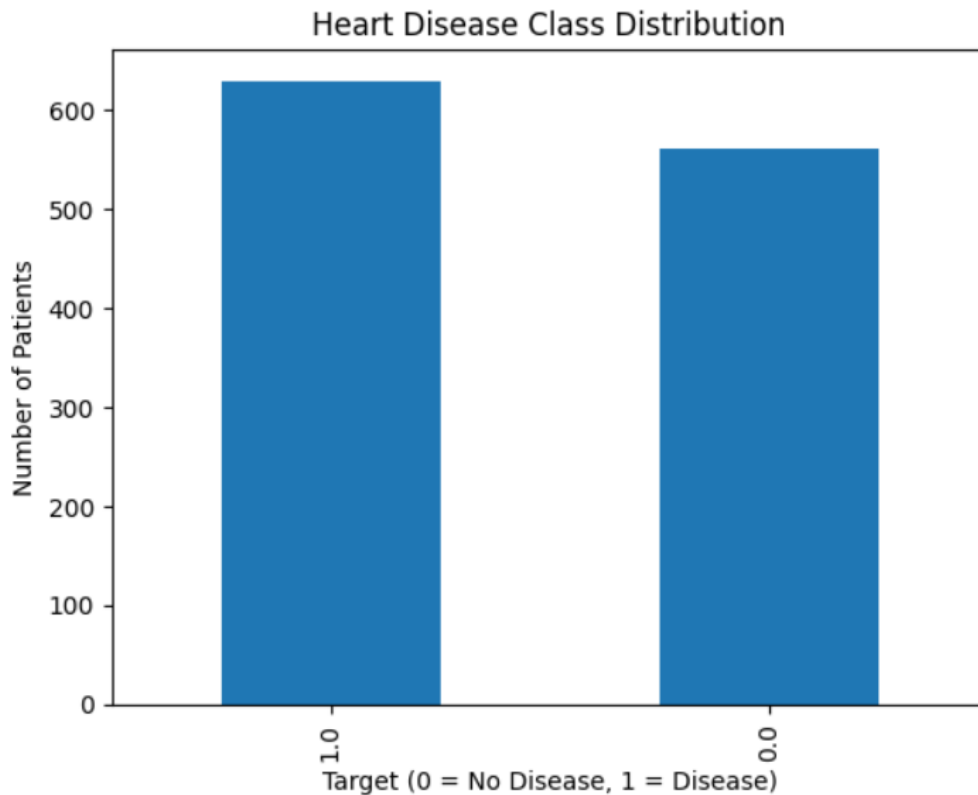


Figure 20: Heatmap of correlation matrix of numerical data

The heatmap above demonstrates the correlation matrix of numerical variables in the dataset. Values close to **1** or **-1** indicate a strong linear relationship, while values near **0** suggest weak or no linear correlation. This visualization helps identify which medical features are strongly related to heart disease, such as chest pain type, maximum heart rate, and exercise-induced angina.

### 3.5.17 Bar Chart of Heart Disease Class Distribution

```
[20]: plt.figure()
data["target"].value_counts().plot(kind="bar")
plt.title("Heart Disease Class Distribution")
plt.xlabel("Target (0 = No Disease, 1 = Disease)")
plt.ylabel("Number of Patients")
plt.show()
```



*Figure 21: Bar chart of heart disease class distribution*

The bar chart illustrates the distribution of patients with and without heart disease in the dataset. This visualization confirms that the dataset is reasonably balanced, which is important for training reliable machine learning models and avoiding biased predictions.

### 3.5.18 K-Means Clustering

```
[21]: from sklearn.cluster import KMeans

X_cluster = data[["age", "cholesterol"]]

kmeans = KMeans(n_clusters=2, random_state=42)
clusters = kmeans.fit_predict(X_cluster)

plt.figure()
plt.scatter(X_cluster["age"], X_cluster["cholesterol"], c=clusters)
plt.scatter(
    kmeans.cluster_centers_[0, 0],
    kmeans.cluster_centers_[0, 1],
    marker="x",
    s=200
)
plt.title("K-Means Clustering of Patients (Age vs Cholesterol)")
plt.xlabel("Age")
plt.ylabel("Cholesterol Level")
plt.show()
```

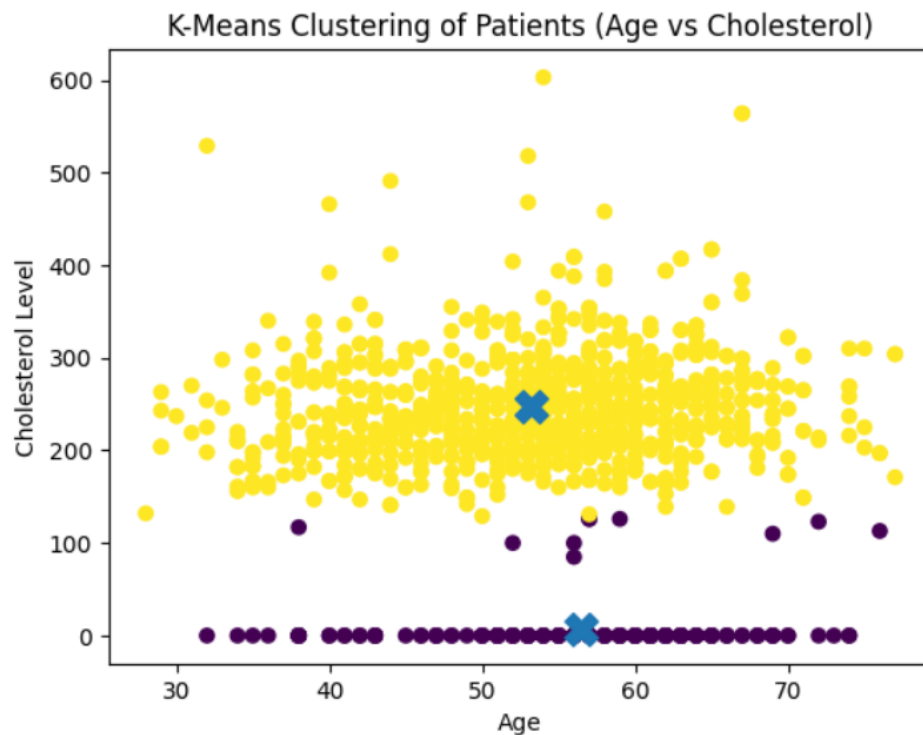


Figure 22: K-Means clustering

The K-Means clustering visualization groups patients into two clusters based on age and cholesterol levels. This unsupervised learning technique helps reveal hidden patterns in the data. The clusters suggest that patients can be grouped into distinct risk categories, which may support further analysis and feature understanding

## **4. Conclusion**

### **4.1 Analysis of the Work Done**

This project focused on building and evaluating a heart disease prediction system using supervised machine learning techniques. Throughout the development process, different algorithms were implemented and compared, which provided a clear understanding of how each model behaves when applied to real medical data.

One of the most important observations from this work was the impact of data preprocessing on model performance. Proper scaling, data splitting, and feature handling played a major role in improving prediction accuracy. The comparison between Logistic Regression, Support Vector Machine, and Random Forest also highlighted that no single algorithm is universally best, as performance depends on the nature of the dataset and the evaluation criteria used. Overall, this project helped in strengthening practical knowledge of machine learning, especially in terms of model training, evaluation, and interpretation of results. It also improved problem-solving skills and understanding of how theoretical concepts are applied in real scenarios.

### **4.2 Real-World Application**

The developed heart disease prediction application can be used as a supportive tool in real healthcare environments. By analyzing patient medical data, the system can provide an early indication of heart disease risk, which may help doctors and healthcare professionals during initial assessments.

Although the system is not intended to replace medical experts, it can assist them by offering quick, data-driven insights. Early prediction can support timely medical intervention, reduce risks, and improve patient outcomes. Such systems are especially useful in situations where large amounts of patient data need to be analyzed efficiently.

### **4.3 Further Work**

While the current system performs well, there are several areas where it can be improved in the future. One possible enhancement is the use of larger and more diverse datasets to improve model generalization. Advanced feature selection techniques could also be applied to further increase prediction accuracy.

In addition, more complex models such as neural networks or deep learning approaches could be explored. Developing a user-friendly web or mobile interface would make the system more accessible for real-world use. Integrating real-time patient data from hospital systems could further increase the practical value of the application.

## 17. References

- Alam, 2020. Application of logistic regression in heart disease prediction. *Health Informatics Journal*, 4(26), pp. 2678-2689.
- Chen, L. Z. Y., 2023. Feature selection and classification techniques in medical datasets.. *Artificial Intelligence Review*, 1(56), pp. 1021-1040.
- Patel, S. & D., 2022. *IEEE Access*. [Online]  
Available at: <https://ieeexplore.ieee.org/document/9771234>  
[Accessed 15 December 2025].
- Rahman, M. H. M. a. K. R., 2021. *Machine learning-based decision support systems in healthcare..* s.l.:Springer Nature..
- Singh, K. &., 2021. Comparative analysis of machine learning techniques for medical diagnosis.. *Journal of Artificial Intelligence Research*, pp. 210-225.
- University of California, I., 2023. *Repository, UCI Machine Learning*. [Online]  
Available at: <https://archive.ics.uci.edu/dataset/45/heart+disease>