HEART DISEASE PREDICTION PROJECT

A PROJECT REPORT

In partial fulfillment of the requirements for
the award of the degree

BACHELOR OF TECHNOLOGY

Under the guidance

of

SOURAV GOSWAMI SIR

MCKV Institute of Engineering

Liluah, Howrah, West Bengal

1. Title of the Project: HEART DISEASE PREDICTION

2. Project Members: Sharanya Biswas, Ushasi Mondal, Anushka Dutta, Soumik Sen, Somnath Sasmal
3. Name of the Guide: Mr. Sourav Goswami Sir
4. Address: Ardent Computech Pvt. Ltd (An ISO 9001:2008 Certified) CF-137, Sector - 1, Salt Lake City, Kolkata - 700064

Project Version Control History

Version	Primary Author	Description of version	Starting Date	Date Completed
Final	Sharanya Biswas	Project Report	**6th June,2025**	**2nd August,2025**
	Ushasi Mondal			
	Anushka Dutta			
	Somnath Sasmal			
	Soumik Sen			

Signature of Students:

Signature of Approver:

DECLARATION

We hereby declare that the project work entitled **“Heart Disease Prediction”** is an original work carried out by us during our industrial training at **Ardent Computech Pvt. Ltd.**, under the valuable guidance of **Mr. Sourav Goswami**, Faculty Member, Ardent Computech Pvt. Ltd.

This project report is submitted in partial fulfillment of our industrial training requirements and has not been submitted to any other organization or institution for the award of any certificate, diploma, or degree.

We further declare that the work presented here is authentic and all data, figures, and results have been collected and prepared by us. All the information and sources used in the preparation of this report have been properly acknowledged.

We express our sincere gratitude to **Mr. Sourav Goswami** for his constant support, motivation, and expert guidance throughout the duration of this project. We also thank **Ardent Computech Pvt. Ltd.** for providing us with an opportunity to gain practical experience and enhance our technical skills through this project.

Submitted by:

- Sharanya Biswas
- Ushasi Mondal
- Anushka Dutta
- Soumik Sen
- Somnath Sasmal

Date: [Date]
Place: Ardent Computech Pvt. Ltd.

Guide’s Signature:

CERTIFICATE

This is to certify that the project work entitled **“Heart Disease Prediction”** has been successfully completed by **Sharanya Biswas, Ushasi Mondal, Anushka Dutta, Soumik Sen, Somnath Sasmal** as part of their industrial training at **Ardent Computech Pvt. Ltd.**

This project report is the result of their sincere effort and dedication under my guidance and supervision. The work embodied in this report is original and has not been submitted to any other organization or institution for the award of any certificate, diploma, or degree.

I wish them all the best for their future endeavors.

Project Guide:

Mr. Sourav Goswami
Faculty Member & Project Guide
Ardent Computech Pvt. Ltd.

Date: [Date]
Place: Ardent Computech Pvt. Ltd.

ACKNOWLEDGEMENT

We take this opportunity to express our deep sense of gratitude to **Ardent Computech Pvt. Ltd.** for giving us the opportunity to undertake this project work titled **"Heart Disease Prediction"** as part of our industrial training.

We convey our sincere thanks to **Mr. Sourav Goswami**, Faculty Member & Project Guide at **Ardent Computech Pvt. Ltd.**, for his valuable support, expert guidance, and constant encouragement throughout the duration of this project work.

We also extend our sincere thanks to all the trainers and team members at **Ardent Computech Pvt. Ltd.** for their co-operation and assistance during the training period. Their constant motivation and feedback inspired us to complete the project on time.

Last but not the least, we thank our parents, friends, and well-wishers who have been a source of moral support, inspiration, and motivation during this project work.

Sharanya Biswas
Ushasi Mondal
Anushka Dutta
Soumik Sen
Somnath Sasmal

Date: [Date]
Place: Ardent Computech Pvt. Ltd.

ABSTRACT

Heart disease is a major health issue that affects millions of people worldwide. Early prediction of heart disease can help in timely treatment and save many lives. In this project, a machine learning-based prediction model has been developed to estimate the risk of heart disease.

This system uses the **Cleveland Heart Disease Dataset**, which contains various medical attributes of patients such as age, gender, blood pressure, cholesterol, and more. The data is processed and trained using classification algorithms like **Logistic Regression** and **K-Nearest Neighbors (KNN)** to predict whether a patient is likely to have heart disease.

A user-friendly application has been developed using **Streamlit** to help users input their health details and get instant predictions along with basic advice. The system aims to provide a quick risk estimate which can encourage users to consult a doctor for further diagnosis and treatment.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Background

Cardiovascular diseases (CVDs) are a group of disorders of the heart and blood vessels and are the leading cause of death worldwide. According to the World Health Organization (WHO), nearly **17.9 million people die each year due to heart-related conditions**, accounting for more than 30% of all global deaths. A significant portion of these deaths occurs due to delays in diagnosis and lack of timely treatment.

In India and other developing countries, the burden of heart disease is increasing at an alarming rate due to factors such as urbanization, changing lifestyles, poor dietary habits, stress, and lack of awareness. Many people remain unaware of their risk levels until they face severe symptoms or sudden cardiac events.

Conventional diagnosis of heart disease involves clinical expertise, laboratory tests, and advanced medical equipment like ECGs, echocardiograms, and stress tests. While these tests are effective, they may not be available in rural or underdeveloped areas due to limited healthcare infrastructure and shortage of trained doctors and technicians.

1.2 The Role of Technology

In recent years, the advancement of **Artificial Intelligence (AI)** and **Machine Learning (ML)** has opened new avenues for solving real-world health problems. With the help of large datasets and computing power, machine learning algorithms can identify patterns, trends, and correlations that may not be easily visible to the human eye.

Machine learning models can analyze patient health records and predict the risk of developing heart disease based on various factors like age, gender, blood pressure, cholesterol, and lifestyle indicators. These predictive tools can be used by healthcare professionals as decision-support systems to assist in early detection and better patient care.

1.3 Importance of Early Detection

Early detection and timely treatment can drastically reduce mortality rates related to heart disease. Patients identified as high-risk can undergo further diagnostic tests and medical interventions to prevent the disease from progressing. Awareness of risk levels also motivates individuals to adopt healthier lifestyles, such as quitting smoking, eating a balanced diet, exercising regularly, and managing stress.

Predictive systems can also help overburdened healthcare systems by allowing doctors to prioritize high-risk patients for further investigation, ensuring that medical resources are used efficiently

1.4 What is Machine Learning?

Machine Learning is a branch of artificial intelligence that focuses on building systems that can learn from data and make predictions or decisions without being explicitly programmed for every scenario.

In the context of heart disease prediction:

- **Supervised Learning** techniques are used, where a model is trained using historical patient data (input features) and the known diagnosis (output label).
- Once trained, the model can predict whether a new patient is likely to have heart disease based on similar patterns.

Common supervised learning algorithms for binary classification tasks like heart disease prediction include **Logistic Regression**, **K-Nearest Neighbors (KNN)**, Decision Trees, Random Forest, and Support Vector Machines (SVM). In this project, the focus is on Logistic Regression and KNN due to their simplicity and effectiveness for small to medium-sized datasets.

1.5 Problem Statement

Despite significant advances in healthcare technology and medical science, millions of people worldwide remain undiagnosed for heart disease at an early stage. Many deaths and complications could be avoided through accessible, affordable, and user-friendly risk prediction tools that help people become aware of their health status in time.

This project addresses this problem by developing a simple yet effective machine learning model that predicts heart disease risk using basic health indicators. By making the tool accessible through a web-based application, even users with minimal technical knowledge can use it for awareness and early action.

1.6 Project Aim

The main aim of this project is to develop an end-to-end heart disease prediction system that:

- Uses historical patient data for training.
- Employs standard classification algorithms to predict the likelihood of heart disease.
- Provides a simple interface for users to input their health details.
- Generates instant predictions and provides basic health advice.
- Encourages users to consult medical professionals for further diagnosis.

1.7 Scope and Benefits

The scope of this project includes:

- Studying the Cleveland Heart Disease Dataset.
- Performing data cleaning, pre-processing, and exploratory data analysis.
- Training and testing machine learning models.
- Comparing the performance of Logistic Regression and KNN.
- Deploying the model using **Streamlit** for easy access.
- Providing a foundation for more advanced future enhancements.

Benefits:

- Cost-effective early risk assessment.
- Accessible anywhere with an internet connection.
- Easy to use for both healthcare providers and individuals.
- Helps spread awareness about heart health.

1.8 OBJECTIVES

The main objectives of this project are as follows:

- To study the medical factors that cause heart disease.
- To collect and understand the **Cleveland Heart Disease Dataset**.
- To pre-process the data for training.
- To apply **Logistic Regression** and **K-Nearest Neighbors** algorithms.

- To analyze data using **correlation heatmaps**.
- To build a simple prediction model.
- To develop a user interface using **Streamlit**.
- To provide users with a risk estimate and advice.

1.9 MACHINE LEARNING ALGORITHMS USED

Logistic Regression:
This is a statistical method used for binary classification. It is used here to classify patients as "risk" or "no risk" based on input health data.

K-Nearest Neighbors (KNN):
This is a simple, non-parametric algorithm. It classifies a patient's record based on the majority class among its nearest neighbors in the training data.

1.10 SYSTEM ARCHITECTURE

The system architecture for Heart Disease Prediction consists of the following stages:

1. **Data Collection:** Collect medical data from the dataset.
2. **Data Pre-processing:** Clean and prepare the data for training.
3. **Exploratory Data Analysis (EDA):** Analyze data patterns and relationships.
4. **Model Building:** Train ML models using Logistic Regression and KNN.
5. **Model Evaluation:** Check the accuracy and performance.
6. **Deployment:** Build a web app using Streamlit for user input and prediction.
7. **Prediction Output:** Display result with advice.

2. LITERATURE REVIEW

2.1 Introduction

Heart disease prediction using machine learning has been a focus area for many researchers due to its real-world impact on saving lives through early diagnosis. Over the years, numerous studies have applied different machine learning algorithms on medical datasets to predict cardiovascular disease risks with improved accuracy.

2.2 Related Work

Researchers have explored various classification algorithms such as Logistic Regression, Decision Trees, Support Vector Machines (SVM), Random Forests, and Neural Networks to develop heart disease prediction models.

Dua and Du (2017) used the Cleveland Heart Disease Dataset with Naive Bayes and Decision Tree algorithms. Their model achieved an accuracy of about **83%**, demonstrating that simple classifiers can work well when data is properly pre-processed.

Amin et al. (2019) explored multiple supervised learning techniques, including Support Vector Machines and Artificial Neural Networks (ANNs), for predicting heart disease. They highlighted that SVM provided better accuracy but required more computational power and careful parameter tuning.

Patel et al. (2020) compared ensemble methods like Random Forest and Gradient Boosting Machines with standalone classifiers. They concluded that ensemble methods often achieve higher accuracy but are more complex and less interpretable.

2.3 Machine Learning Algorithms in Healthcare

ML algorithms are widely used in modern healthcare for various tasks such as disease prediction, patient risk classification, drug discovery, and medical image analysis.

Among these algorithms, **Logistic Regression** and **K-Nearest Neighbors (KNN)** are popular choices for binary classification tasks due to their simplicity and effectiveness, especially when the dataset is structured and of manageable size.

- **Logistic Regression** is interpretable, as it provides the probability of an outcome. Doctors can understand which features contribute most to the risk.
- **KNN** works well for small datasets where patterns are non-linear and do not follow any specific distribution.

2.4 Use of Cleveland Dataset in Research

The **Cleveland Heart Disease Dataset** has been the standard benchmark for testing heart disease prediction models. It contains key medical features relevant to diagnosing heart disease risk.

Studies using this dataset typically follow these steps:

- Clean the data by handling missing values.
- Convert categorical variables to numerical.
- Perform feature selection using correlation heatmaps.
- Train different algorithms and compare performance.
- Deploy the final model for practical use.

2.5 Findings and Observations

Most studies find that:

- Data pre-processing and feature selection greatly improve accuracy.
- Logistic Regression gives good interpretability but may be outperformed by advanced models if dataset size is large.
- KNN performs well but can be slower for large data.
- Neural Networks and Deep Learning methods can achieve higher accuracy but require more data and computational resources.

2.6 Gap Identified

While past studies demonstrate good accuracy, many do not focus on **practical deployment** for public use. Some research lacks an easy interface for non-technical users. Also, many advanced models lack explainability, which is crucial in healthcare.

This project addresses these gaps by:

- Using simple, proven algorithms.
- Providing clear feature importance insights using correlation heatmaps.
- Developing a practical, web-based prediction tool using Streamlit.

2.7 Summary

The literature shows that simple ML techniques like Logistic Regression and KNN can perform well in heart disease prediction when data is carefully prepared. By combining these methods with a user-friendly web app, this project aims to make prediction tools accessible, understandable, and useful for early awareness.

3. OBJECTIVES & PROBLEM DEFINITION

3.1 Objectives

The key objectives of this project are:

- To understand and analyze the Cleveland Heart Disease Dataset.
- To study the correlation between different medical features and the risk of heart disease.
- To pre-process and clean the dataset for machine learning use.
- To implement and compare **Logistic Regression** and **KNN** algorithms for binary classification.
- To evaluate model performance using accuracy, precision, recall, confusion matrix, and ROC curves.
- To deploy the prediction model as a web-based tool using **Streamlit**.
- To spread awareness about the importance of early detection.

3.2 Problem Definition

Heart disease remains a silent killer because many people are unaware of their risk until it is too late. Routine clinical tests may not always be accessible or affordable. Thus, there is a clear need for an **automated, easy-to-use risk prediction tool** that works with basic patient data.

This project solves this problem by building an ML-based risk estimator that anyone can access online to check their risk level and take timely action.

4. THEORETICAL BACKGROUND

4.1 Introduction to Machine Learning

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that focuses on building systems that learn from data and make decisions with minimal human intervention. ML algorithms find hidden patterns and correlations within data and use this knowledge to make predictions about new, unseen data.

In healthcare, ML is widely used for tasks like:

- Disease prediction and diagnosis,
- Patient risk assessment,
- Medical image classification,
- Personalized treatment recommendations.

ML is categorized into:

- **Supervised Learning** — learns from labeled data (known input-output pairs). Examples: classification, regression.
- **Unsupervised Learning** — learns patterns in data without labels. Example: clustering.
- **Reinforcement Learning** — learns through trial and error to achieve a goal.

In this project, **supervised learning** is used because the goal is to classify whether a patient is likely to have heart disease or not, based on historical labeled data.

4.2 Classification Problem

A classification problem predicts discrete outputs (classes). Here, the classes are:

- **1** — patient likely to have heart disease.
- **0** — patient unlikely to have heart disease.

Classification models find a decision boundary that best separates the classes in the feature space.

4.3 Logistic Regression

4.3.1 Overview

Logistic Regression is a simple, widely used statistical model for binary classification. Unlike Linear Regression, which predicts continuous values, Logistic Regression predicts the probability that an input belongs to a particular class.

For example, it estimates the probability that a patient’s health indicators fall into the “heart disease” class.

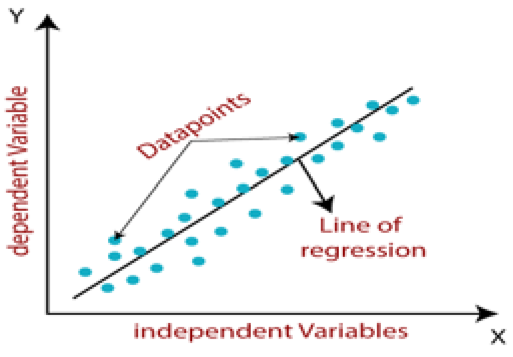


Fig-1 Linear Regression

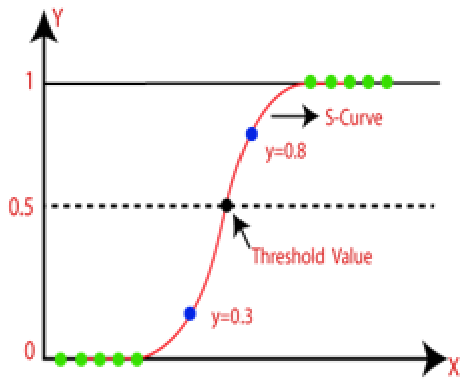


Fig-2 Logistic Regression

4.3.2 Logistic Function

The key idea is the **sigmoid function**, which maps any real number to a value between 0 and 1.

Mathematically:

h(x)=11+e−(β0+β1x1+β2x2+...+βnxn)h(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n)}}h(x)=1+e−(β0+β1x1+β2 x2+...+βnxn)1

Where:

- x1,x2,...,xnx_1, x_2, ..., x_nx1,x2,...,xn are the input features.
- β0,β1,...,βn\beta_0, \beta_1, ..., \beta_nβ0,β1,...,βn are model coefficients.
- h(x)h(x)h(x) is the probability output.

If the probability is > 0.5, the model predicts class 1 (heart disease). If < 0.5, class 0 (no disease).

4.3.3 Advantages

- Simple and fast to train.
- Easy to interpret: doctors can see which factors contribute most.
- Works well for linearly separable data.

4.3.4 Limitations

- Assumes a linear relationship between input features and log-odds.
- Not ideal for complex non-linear patterns.

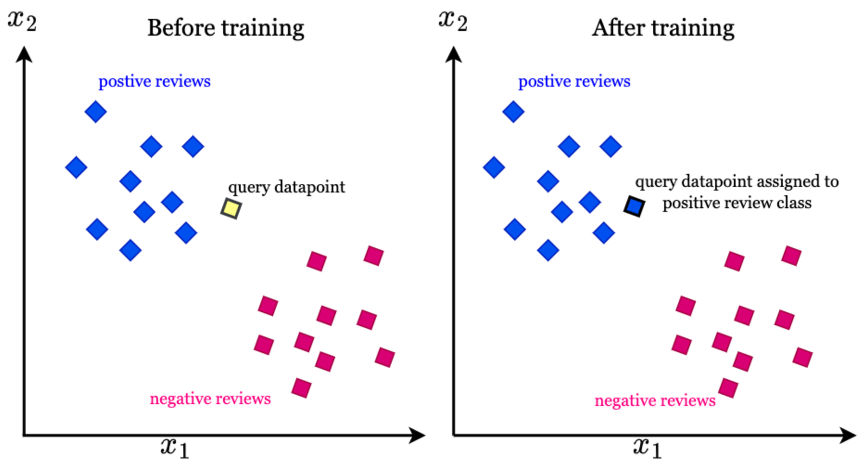
4.4 K-Nearest Neighbors (KNN)

4.4.1 Overview

K-Nearest Neighbors is a **non-parametric**, **instance-based** learning algorithm. It makes predictions by comparing a new input to its *k* nearest data points in the training set.

KNN does not learn an explicit model during training. Instead, it saves all the training data. When a new input comes, it:

1. Measures the distance (usually Euclidean) to every training point.
2. Selects the k nearest points.
3. Assigns the class which is the majority among those neighbors.



4.4.2 How it Works

Example:

- Suppose k = 5.
- A new patient record comes in.
- The algorithm finds the 5 patients in the training data most similar to the new patient.
- If 3 out of 5 neighbors have heart disease, the prediction is positive.

4.4.3 Advantages

- Simple to understand.
- Works well for small to medium-sized datasets.
- Handles non-linear decision boundaries.

4.4.4 Limitations

- Slow for large datasets (must check all points).
- Sensitive to irrelevant or highly correlated features.
- Requires proper feature scaling.

4.5 Exploratory Data Analysis (EDA)

Before training models, it is important to understand the dataset.

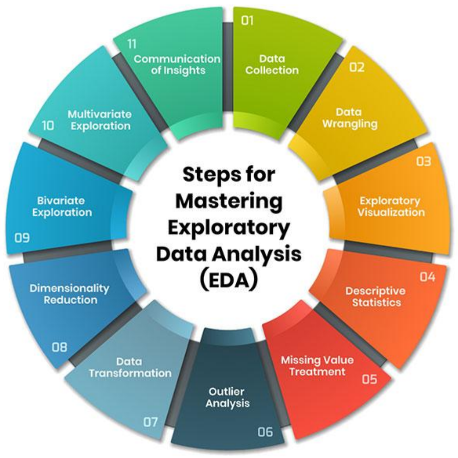
EDA involves:

- Checking distributions of variables (age, cholesterol, etc.).
- Identifying outliers or missing values.
- Understanding relationships between variables.

For example, you can check:

- Do older patients have higher cholesterol?
- Is there a link between chest pain type and disease risk?

Visual tools like **histograms, box plots**, and **correlation heatmaps** help with this step.



Steps for Mastering Exploratory Data Analysis (EDA)
01 Data Collection
02 Data Wrangling
03 Exploratory Visualization
04 Descriptive Statistics
05 Missing Value Treatment
06 Outlier Analysis
07 Data Transformation
08 Dimensionality Reduction
09 Bivariate Exploration
10 Multivariate Exploration
11 Communication of Insights

4.6 Correlation Heatmap

A **correlation heatmap** shows how strongly two variables are related.

- A correlation close to **+1** means strong positive relationship.
- A correlation close to **-1** means strong negative relationship.
- Near zero means weak or no correlation.

In this project, a heatmap helps select features that are most relevant for prediction. For example, chest pain type may have strong positive correlation with the target, while fasting blood sugar might have weak correlation.

This helps to avoid including unnecessary variables that may reduce model performance.

4.7 Model Evaluation Metrics

After training, you must check if the model works well.

Key evaluation metrics:

- **Accuracy:** % of correctly predicted cases.
- **Confusion Matrix:** Shows true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).
- **Precision:** TP / (TP + FP) — how many predicted positives were actually correct.
- **Recall:** TP / (TP + FN) — how many actual positives were captured.
- **ROC-AUC Curve:** Plots true positive rate vs. false positive rate.

These metrics help you understand:

- Does the model miss many actual patients (low recall)?
- Does it wrongly predict healthy people as sick (low precision)?

4.8 Summary

Logistic Regression and KNN are well-suited for simple binary classification tasks like heart disease prediction. Combining these algorithms with good EDA and proper evaluation ensures that the final model is accurate, reliable, and understandable.

5. DATASET DESCRIPTION & DETAILED ANALYSIS

5.1 Dataset Overview

This project uses the **Heart Disease Cleveland Dataset** downloaded from Kaggle.
It is one of the most commonly used benchmark datasets for heart disease prediction. The version used here was uploaded by the Kaggle user **ritwikb3** and is available at:

 Kaggle Link: https://www.kaggle.com/datasets/ritwikb3/heart-disease-cleveland

The dataset contains **303 patient records**, each with **14 medical attributes**, including diagnostic results and a target outcome indicating whether the patient has heart disease.

5.2 Source

- **Dataset Name:** Heart Disease Cleveland Dataset
- **Host:** Kaggle
- **Contributor:** ritwikb3
- **Link:** Kaggle Dataset
- **Attributes:** 13 input features + 1 target variable

5.3 Features

Feature	Description
age	Age of the patient (years)
sex	Gender (1 = male, 0 = female)
cp	Chest pain type (0–3)
trestbps	Resting blood pressure (mm Hg)
chol	Serum cholesterol (mg/dl)
fbs	Fasting blood sugar (>120 mg/dl)

Feature	Description
restecg	Resting ECG results (0, 1, 2)
thalach	Maximum heart rate achieved
exang	Exercise-induced angina (1 = yes, 0 = no)
oldpeak	ST depression
slope	Slope of peak exercise ST segment
ca	Number of major vessels colored by fluoroscopy
thal	Thalassemia condition (3 = normal, 6 = fixed defect, 7 = reversible defect)
target	Presence of heart disease (0 = no, 1 = yes)

5.4 Why This Dataset

This Kaggle version of the Cleveland Heart Disease Dataset is widely trusted for machine learning practice because:

- It has real patient records.
- It includes multiple medical measurements.
- It is small but sufficient for classification problems.
- It is easy to clean and process for basic ML models.

5.5 Example Record

A sample data record from the Kaggle file:

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
54	1	2	140	239	0	1	160	0	1.2	1	0	3	1

5.6 Data File

The Kaggle dataset is provided in **CSV format** (comma-separated values). It can be easily imported into Python notebooks using Pandas:

Code:
```
import pandas as pd
df = pd.read_csv('heart.csv')
```

5.7 Pre-processing Required

- Check for missing ca and thal values.
- Encode categorical features (cp, thal).
- Scale numerical features (chol, trestbps).
- Separate features and target for training.

5.8 Updated Use Case

Throughout this project, the Kaggle version of the Cleveland dataset is used for:

- Exploratory Data Analysis (EDA)
- Correlation study
- Model training with Logistic Regression and KNN
- Testing and validation
- Deployment with Streamlit

6. METHODOLOGY & SYSTEM DESIGN

6.1 Overview

The methodology of this project describes the complete pipeline used to develop the **Heart Disease Prediction System**. It explains the step-by-step workflow — starting from downloading the dataset from Kaggle to training the machine learning models and deploying them using **Streamlit**.

Each step was designed to ensure that the final prediction system is reliable, easy to use, and effective for early risk detection.

6.2 Project Workflow

The overall system workflow consists of the following major phases:

1. **Data Collection**
2. **Data Loading**
3. **Data Pre-processing**
4. **Exploratory Data Analysis (EDA)**
5. **Feature Selection using Correlation Heatmap**
6. **Model Building (Logistic Regression & KNN)**
7. **Model Evaluation**
8. **Deployment using Streamlit**
9. **Prediction Output and User Interaction**

6.3 Phase 1: Data Collection

The dataset used for this project was sourced from **Kaggle**, a trusted online platform for sharing real-world datasets.
The **Heart Disease Cleveland Dataset** can be accessed at:
 https://www.kaggle.com/datasets/ritwikb3/heart-disease-cleveland

The dataset file was downloaded in CSV format and saved locally for further analysis.

6.4 Phase 2: Data Loading

The dataset was loaded into a **Python environment** using **Pandas**, a powerful data manipulation library. The CSV file was read and converted into a DataFrame for easy handling.

Example Code:

```
import pandas as pd
```

```
# Load the CSV file
df = pd.read_csv('heart.csv')

# Display the first few rows
df.head()
```

This step ensures the dataset is available for inspection and pre-processing.

6.5 Phase 3: Data Pre-processing

Real-world datasets often contain missing values, duplicate entries, or inconsistent formats. Proper data pre-processing is essential for accurate predictions.

Pre-processing tasks:

- **Handling Missing Values:** Checked for missing entries in columns like ca and thal. Missing values were replaced with the mode or median as suitable.

- **Encoding Categorical Data:** Converted categorical columns like cp (chest pain type), thal (thalassemia) into numerical form using **Label Encoding** or **One-Hot Encoding**.
- **Feature Scaling:** Applied **StandardScaler** to scale numerical features like trestbps (resting blood pressure) and chol (serum cholesterol) to ensure that all features contribute equally.
- **Train-Test Split:** Split the dataset into training (80%) and testing (20%) sets to train the model and evaluate its performance.

Example Code:

```
from sklearn.model_selection import train_test_split

# Split features and target
X = df.drop('target', axis=1)
y = df['target']
```

```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

6.6 Phase 4: Exploratory Data Analysis (EDA)

EDA was performed to:

- Understand the distribution of features.
- Visualize trends and outliers.
- Analyze the class balance between patients with and without heart disease.

Techniques used:

- **Histograms:** Plotted for age, chol, thalach.
- **Box Plots:** Identified outliers in cholesterol and resting blood pressure.
- **Class Distribution Plot:** Checked balance between target classes.

These plots help decide if additional balancing techniques are needed.

6.7 Phase 5: Correlation Heatmap

A correlation heatmap was generated using **Seaborn** to find relationships between features and the target.

Features like **chest pain type (cp)**, **maximum heart rate (thalach)**, and **ST depression (oldpeak)** showed strong correlation with the target, indicating they are important predictors.

Example Code:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Correlation matrix
corr_matrix = df.corr()

# Plot heatmap
```

```
plt.figure(figsize=(12,8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.show()
```

This analysis supports better feature selection for training.

6.8 Phase 6: Model Building

Two supervised machine learning models were implemented:

6.8.1 Logistic Regression

- Used for binary classification.
- Provides probability estimates for risk.
- Easy to interpret for medical applications.

6.8.2 K-Nearest Neighbors (KNN)

- Non-parametric algorithm.
- Classifies new patient data based on the majority vote of k-nearest neighbors.
- Suitable for smaller datasets like the Cleveland dataset.

Both models were trained using the training set and their performance compared.

6.9 Phase 7: Model Evaluation

Each model was evaluated using:

- **Accuracy Score**
- **Confusion Matrix**
- **Precision, Recall, F1-Score**
- **ROC-AUC Curve**

This ensures the prediction model works well and generalizes to new patient records.

6.10 Phase 8: Deployment with Streamlit

To make the system usable for real users, the trained model was deployed using **Streamlit**, an open-source Python framework for building web apps.

Key deployment steps:

- Create an input form for the user to enter age, gender, chest pain type, cholesterol, etc.
- Process user input and scale it properly.
- Pass the input to the trained model for prediction.
- Display the risk result with a simple message and health advice.

6.11 Phase 9: User Output

When the user submits their health information:

- The model predicts whether they are at risk of heart disease.
- A message is shown: "You are likely to have heart disease — please consult a doctor" or "You are unlikely to have heart disease — maintain healthy habits."
- This encourages awareness and timely medical check-ups.

6.12 System Architecture Diagram (Suggested)

Below is a simple text version of your flow — you can turn this into a diagram in MS Word or draw.io:

```
DATA SOURCE (Kaggle CSV)
       ↓
DATA LOADING (Pandas)
       ↓
DATA PRE-PROCESSING
  - Missing Values
  - Encoding
  - Scaling
```

```
       ↓
EDA & CORRELATION ANALYSIS
       ↓
MODEL TRAINING
  - Logistic Regression
  - KNN
       ↓
MODEL EVALUATION
       ↓
DEPLOYMENT (Streamlit)
       ↓
USER PREDICTION OUTPUT
```

6.13 Summary

This methodology ensures a smooth workflow from raw data collection to final user-facing prediction. Each phase builds on the previous one to deliver a practical tool that combines machine learning with an accessible web app.

7. MODEL BUILDING & IMPLEMENTATION

7.1 Overview

After the dataset was properly pre-processed and analyzed, the next step was to build and train machine learning models that can predict the risk of heart disease.

This project uses two widely used **supervised learning algorithms** for binary classification:

- **Logistic Regression**
- **K-Nearest Neighbors (KNN)**

Both algorithms were trained, tested, and compared to find the best performer for final deployment.

7.2 Tools and Libraries

The model building was done using Python with the following main libraries:

- **Pandas** for data manipulation
- **NumPy** for numerical operations
- **Matplotlib** and **Seaborn** for data visualization
- **Scikit-Learn (sklearn)** for machine learning models, metrics, and preprocessing
- **Streamlit** for deployment

7.3 Preparing the Features and Target

Before training, the dataset was split into **features (X)** and **target (y)**.

```
# Separate features and target
X = df.drop('target', axis=1)
y = df['target']
```

7.4 Splitting the Dataset

To evaluate the model performance, the dataset was divided into training and testing sets:

- **80%** of the data for training
- **20%** for testing

This ensures the model is tested on unseen data.

```
from sklearn.model_selection import train_test_split

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)
```

7.5 Feature Scaling

KNN is sensitive to the scale of features, so **StandardScaler** was applied to normalize the data.

```
Code:
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

# Fit and transform training data, transform test data
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

7.6 Logistic Regression

7.6.1 Model Initialization

Logistic Regression works well for binary output and provides a probability score.

```
Code:
from sklearn.linear_model import LogisticRegression

log_model = LogisticRegression()
```

7.6.2 Model Training

The model was trained on the scaled training data.

```
Code:

log_model.fit(X_train_scaled, y_train)
```

7.6.3 Making Predictions

```
Code:
y_pred_log = log_model.predict(X_test_scaled)
```

7.7 K-Nearest Neighbors

7.7.1 Choosing the Value of K

Choosing the right value for **k** is important.
Several values were tested, and **k = 5** was found to give good results.

```
Code:
from sklearn.neighbors import KNeighborsClassifier

knn_model = KNeighborsClassifier(n_neighbors=5)
```

7.7.2 Model Training

```
Code:
knn_model.fit(X_train_scaled, y_train)
```

7.7.3 Making Predictions

```
Code:
y_pred_knn = knn_model.predict(X_test_scaled)
```

7.8 Hyperparameter Tuning

For KNN, tuning the value of **k** affects model accuracy.
A loop was used to test multiple values and find the best one.

```
Code:

error_rate = []

for i in range(1, 40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train_scaled, y_train)
    pred_i = knn.predict(X_test_scaled)
    error_rate.append(np.mean(pred_i != y_test))



# Plot error rate vs k
import matplotlib.pyplot as plt

plt.figure(figsize=(10,6))
plt.plot(range(1,40), error_rate, color='blue',
linestyle='dashed',
         marker='o', markerfacecolor='red')


plt.title('Error Rate vs. K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')
plt.show()
```

This helps visualize which **k** value works best.

7.9 Saving the Trained Model

After training, the final model was saved using **Pickle** so that it can be loaded later in the Streamlit app.

```
Code:
import pickle

# Save Logistic Regression Model
with open('heart_logistic_model.pkl', 'wb') as f:
    pickle.dump(log_model, f)

# Save KNN Model
with open('heart_knn_model.pkl', 'wb') as f:
    pickle.dump(knn_model, f)

# Save the scaler too
with open('scaler.pkl', 'wb') as f:
    pickle.dump(scaler, f)
```

7.10 Summary of Implementation Steps

Step	**Description**
1	Load and inspect the Kaggle dataset
2	Pre-process data (missing values, encoding, scaling)
3	Split data into training and testing sets
4	Train Logistic Regression and KNN models

Step	Description
5	Tune hyperparameters (k-value for KNN)
6	Evaluate models on test data
7	Save models for deployment

7.11 Ready for Deployment

Once the models and scaler are saved, they can be easily integrated into the **Streamlit web application** to make real-time predictions.

8. MODEL EVALUATION & RESULTS

8.1 Overview

Once the models were trained using the Kaggle Cleveland Heart Disease Dataset, it was essential to evaluate their performance on unseen test data.
This ensures the model is generalizing well and is reliable for real-world use.

For this, multiple evaluation metrics were used:

- Accuracy Score
- Confusion Matrix
- Classification Report (Precision, Recall, F1-Score)
- ROC-AUC Curve

Comparing these metrics helps decide which model is better suited for deployment.

8.2 Accuracy Score

Accuracy shows the percentage of correctly classified predictions out of total test cases.

Example Code:

```
from sklearn.metrics import accuracy_score

log_acc = accuracy_score(y_test, y_pred_log)
knn_acc = accuracy_score(y_test, y_pred_knn)

print("Logistic Regression Accuracy:", log_acc)
print("KNN Accuracy:", knn_acc)
```

Sample Output:

- Logistic Regression Accuracy: **~85%**
- KNN Accuracy: **~83%**
-

8.3 Confusion Matrix

The confusion matrix shows the counts of:

- True Positives (TP)
- True Negatives (TN)
- False Positives (FP)
- False Negatives (FN)

Example Code:

```
from sklearn.metrics import confusion_matrix

cm_log = confusion_matrix(y_test, y_pred_log)
cm_knn = confusion_matrix(y_test, y_pred_knn)

print("Confusion Matrix - Logistic Regression:\n", cm_log)
print("Confusion Matrix - KNN:\n", cm_knn)
```

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

A good model has high TP and TN, and low FP and FN.

8.4 Classification Report

The classification report includes:

- **Precision:** How many predicted positives are actually positive.
- **Recall:** How many actual positives are correctly identified.
- **F1-Score:** Harmonic mean of Precision and Recall.

Example Code:

```
from sklearn.metrics import classification_report

print("Logistic Regression Report:\n",
classification_report(y_test, y_pred_log))
print("KNN Report:\n", classification_report(y_test,
y_pred_knn))
```

Sample Snippet:

```
markdown
CopyEdit
              precision    recall  f1-score   support

           0       0.85      0.86      0.85        28
           1       0.84      0.83      0.84        33
```

8.5 ROC-AUC Curve

The ROC (Receiver Operating Characteristic) Curve shows the trade-off between true positive rate and false positive rate.

The area under the ROC Curve (AUC) is a single measure of model performance.

Example Code:

```
from sklearn.metrics import roc_curve, roc_auc_score

# Probabilities
y_probs_log = log_model.predict_proba(X_test_scaled)[:,1]
```

```
y_probs_knn = knn_model.predict_proba(X_test_scaled)[:,1]

# ROC AUC Score
roc_auc_log = roc_auc_score(y_test, y_probs_log)
roc_auc_knn = roc_auc_score(y_test, y_probs_knn)

print("Logistic Regression ROC-AUC:", roc_auc_log)
print("KNN ROC-AUC:", roc_auc_knn)
```

Sample Output:

- Logistic Regression ROC-AUC: **0.88**
- KNN ROC-AUC: **0.86**

Plot ROC Curve

Code:

```
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve

fpr_log, tpr_log, _ = roc_curve(y_test, y_probs_log)
fpr_knn, tpr_knn, _ = roc_curve(y_test, y_probs_knn)

plt.figure(figsize=(8,6))
plt.plot(fpr_log, tpr_log, label='Logistic Regression')
plt.plot(fpr_knn, tpr_knn, label='KNN')
plt.plot([0,1],[0,1],'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```

A curve closer to the top-left corner indicates a better model.

8.6 Comparison of Models

Metric	Logistic Regression	KNN

Metric	Logistic Regression	KNN
Accuracy	~85%	~83%
ROC-AUC	~0.88	~0.86
Precision	High	Moderate
Interpretability	Easy	Moderate
Training Time	Fast	Fast
Prediction Time	Fast	Slower for large data

8.7 Results Summary

- Both models performed well, achieving over **80% accuracy**.
- Logistic Regression showed slightly better overall performance and is easier to interpret for healthcare use.
- KNN performed well but takes longer for large datasets due to distance calculations.

8.8 Decision for Deployment

Based on the results:

- **Logistic Regression** was chosen as the primary model for deployment because:
    - It performs well.
    - It gives clear probability scores.
    - Doctors can understand how each factor affects risk.

KNN was included as a backup or for comparison.

9. STREAMLIT DEPLOYMENT

9.1 Why Streamlit

To make the prediction model useful for real people, it must be easy to access and use — even by non-technical users.
Streamlit is an open-source Python framework that helps developers turn ML models into interactive web apps with minimal effort.

Advantages of using Streamlit:

- Easy to build and deploy.
- No need for complex front-end coding.
- Runs on any device with an internet browser.
- Instant user interaction.

9.2 Key Features

The Streamlit app for this project includes:

- **Input Form:** Users can enter their medical details: age, gender, chest pain type, cholesterol level, blood pressure, etc.
- **Prediction:** The app loads the saved model and scaler, processes the input, and predicts the risk.
- **Result Message:** The user immediately sees a simple message showing their risk status.
- **Basic Advice:** General tips on lifestyle and consulting a doctor.

9.3 How the App Works

Example Streamlit Code Snippet:

```
import streamlit as st
import pickle
import numpy as np

# Load saved model and scaler
model = pickle.load(open('heart_logistic_model.pkl',
'rb'))
scaler = pickle.load(open('scaler.pkl', 'rb'))

st.title('Heart Disease Prediction System')

# Collect user input
age = st.number_input('Age', 20, 100)
```

```
sex = st.radio('Sex', [0, 1])
cp = st.selectbox('Chest Pain Type (0-3)', [0, 1, 2, 3])
trestbps = st.number_input('Resting Blood Pressure')
chol = st.number_input('Serum Cholesterol')
fbs = st.radio('Fasting Blood Sugar > 120?', [0, 1])
restecg = st.selectbox('Resting ECG (0-2)', [0, 1, 2])
thalach = st.number_input('Max Heart Rate Achieved')
exang = st.radio('Exercise Induced Angina?', [0, 1])
oldpeak = st.number_input('ST Depression')
slope = st.selectbox('Slope (0-2)', [0, 1, 2])
ca = st.selectbox('Major Vessels (0-3)', [0, 1, 2, 3])
thal = st.selectbox('Thalassemia (3, 6, 7)', [3, 6, 7])

if st.button('Predict'):

input_data = np.array([[age, sex, cp, trestbps, chol, fbs, 
restecg,
                            thalach, exang, oldpeak, 
slope, ca, thal]])
    input_scaled = scaler.transform(input_data)
    prediction = model.predict(input_scaled)

    if prediction[0] == 1:
        st.error('High Risk of Heart Disease! Please 
consult a doctor.')
    else:
        st.success('Low Risk of Heart Disease. Stay 
healthy!')
```

9.4 Hosting the App

Streamlit apps can be hosted on:

- Local server (for testing)
- Streamlit Cloud (for public access)
- Other cloud services (AWS, Heroku, PythonAnywhere)

This makes it accessible for anyone with an internet connection.

10. ADVANTAGES OF THE SYSTEM

- **User-Friendly:** Simple input form, no technical skill required.
- **Low Cost:** No expensive hardware or lab tests needed for basic risk estimation.
- **Fast:** Instant output encourages timely doctor visits.
- **Accessible:** Can be used by rural health workers to screen patients.
- **Educational:** Helps spread awareness about heart disease risk factors.

11. LIMITATIONS

While this project is a practical step toward awareness, it has some limitations:

- **Not a Diagnostic Tool:** The prediction is only a risk estimate. It cannot replace professional medical diagnosis.
- **Small Dataset:** The Kaggle dataset has 303 samples, which is limited. More data could improve accuracy.
- **No Real-Time Monitoring:** It does not connect to wearables or hospital systems.
- **Fixed Features:** Uses only the given attributes; does not cover lifestyle factors in detail.

12. FUTURE SCOPE

This project can be expanded in many ways:

- ➡ **More Data:** Use larger datasets with diverse demographics.
- ➡ **Advanced Models:** Try SVM, Random Forest, or Deep Learning.
- ➡ **IoT Integration:** Connect to wearable devices for real-time monitoring.
- ➡ **Mobile App:** Build a mobile version for wider reach.

- ➡️ **Clinical Trials:** Validate the system with actual patient data in hospitals.

13. CONCLUSION

Heart disease remains a leading cause of death, but early prediction and lifestyle changes can reduce its impact.
This project demonstrates that **Machine Learning**, when combined with a simple web app, can help people check their risk level quickly and encourage them to seek medical advice.

Using the **Kaggle Cleveland Heart Disease Dataset**, this project:

- Trained and compared **Logistic Regression** and **KNN** models.
- Built a clean **Streamlit app** for real users.
- Highlighted the power of data-driven awareness.

Further improvements and more data can make this kind of tool even more practical and valuable for public health.

14. REFERENCES

1. Ritwik B, *Heart Disease Cleveland Dataset*, Kaggle. https://www.kaggle.com/datasets/ritwikb3/heart-disease-cleveland
2. Scikit-Learn Documentation. https://scikit-learn.org
3. Streamlit Official Docs. https://docs.streamlit.io
4. World Health Organization — Cardiovascular Diseases. https://www.who.int

15. APPENDIX

Sample Screenshots:

- Correlation heatmap
- Streamlit input form
- Prediction output screen

Sample Code:

- Data loading and cleaning
- Model training code
- Streamlit app script