Data Analytics and Visualization

Heart Disease Predictive System

Student No: Q102543890

Anandhakrishnan Madathil Remesh

Table of Contents

INTRODUCTION **2**

AIMS AND OBJECTIVE **4**

General layout of the proposed prediction system 4

METHODS **5**

Dataset 5

ANALYSIS AND RESULTS **7**

Dataset Preparation 7

Data preprocessing 8

EXPLORATORY DATA ANALYSIS **10**

DATA MODELLING, VISUALIZATION AND HYPERPARAMETERS TUNING **15**

Data Preprocessing 16

Feature Scaling 16

Logistic Regression Model 17

K-Nearest Neighbors (KNN) 17

Random Forest 18

Confusion Matrix and ROC Curve 19

Predictions for New Data 20

EVALUATION **20**

LIMITATIONS AND CHALLENGES **24**

CONCLUSION **25**

REFERENCE LIST **26**

APPENDIX **27**

Table of Figures

Figure 1 Layout Steps 4
Figure 2 Conversion 7
Figure 3 Handling Missing values 8
Figure 4 Outlines detection 9
Figure 5 Encode Categories 9
Figure 6 Feature Engineering 9
Figure 7 Overall Statistics 10
Figure 8 The age distribution Chart 11
Figure 9 Boxplot for Cholesterol level 11
Figure 10 Gender Distribution 12
Figure 11 Chest Pain Type Bar Chart 13
Figure 12 Scatter Plot 14
Figure 13 Correlation Heatmap 14
Figure 14 Boxplot for Old peak vs Target 15
Figure 15 preprocessing Step 1 16
Figure 16 preprocessing Step 2 17
Figure 17 Training Logistic Model 17
Figure 18 Training KNN 18
Figure 19 Training Random Forest 19
Figure 20 Plotting ROC 19
Figure 21 New Data For predictions 20
Figure 22 Evaluations 20
Figure 23 Confusion matrix plot 1 21
Figure 24 Confusion matrix plot 2 22
Figure 25 previous work Results on Testing set 23
Figure 26 Tree Map of Cholesterol by Age 28
Figure 27 Logistic Model ROC 30

Introduction

Every year, 17.9 million lives are lost to heart-related illnesses, making them one of the most significant health concerns worldwide. According to the World Health Organization (WHO), (1) these conditions contribute to 32% of all global deaths, highlighting their widespread impact.
Traditional ways of diagnosing heart diseases (CVDs) involve physical exams, medical histories, and lab tests. But these methods can be time-consuming, expensive, and sometimes wrong. On the other hand, Machine Learning (ML) and Deep Learning (DL) techniques, which are part of Artificial Intelligence (AI), have shown great promise in predicting and diagnosing diseases early. ML systems can analyze a lot of medical data and find patterns and connections that might not be obvious right away. This means they can detect diseases faster and more accurately. Early diagnosis can save lives, lower death rates, and reduce the global impact of CVDs.

In this research study, a predictive model for cardiovascular disease was developed. The Cleveland Heart Disease dataset was sourced from the UC Irvine Machine Learning Repository. The study employed three primary machine learning algorithms: Random Forest (RF), Logistic Regression (LR), and K-Nearest Neighbors (KNN), (2). Each algorithm's performance was evaluated using key metrics, including accuracy, precision, recall, and F1 score. To further enhance predictive performance, hyperparameter tuning was applied to optimize each model. The results demonstrated that ML algorithms can effectively predict heart disease, providing a foundation for developing robust, accurate, and practical systems to assist healthcare professionals in the early diagnosis and treatment of cardiovascular diseases.

Aims and objective

This study aims to develop a predictive system for heart disease using machine learning (ML) techniques and Tableau for data visualization. The Cleveland heart disease dataset is utilized, with feature engineering and preprocessing to optimize performance. Interactive Tableau dashboards provide insights into heart disease risk factors, while ML models are implemented and fine-tuned. The system offers accurate early predictions, supporting healthcare professionals in proactive diagnosis and treatment, combining advanced ML techniques with intuitive visual analysis.

General layout of the proposed prediction system

This is the general layout of a heart disease prediction model: it involves data collection, pre-processing, model training, selection, and prediction with interpretable results.

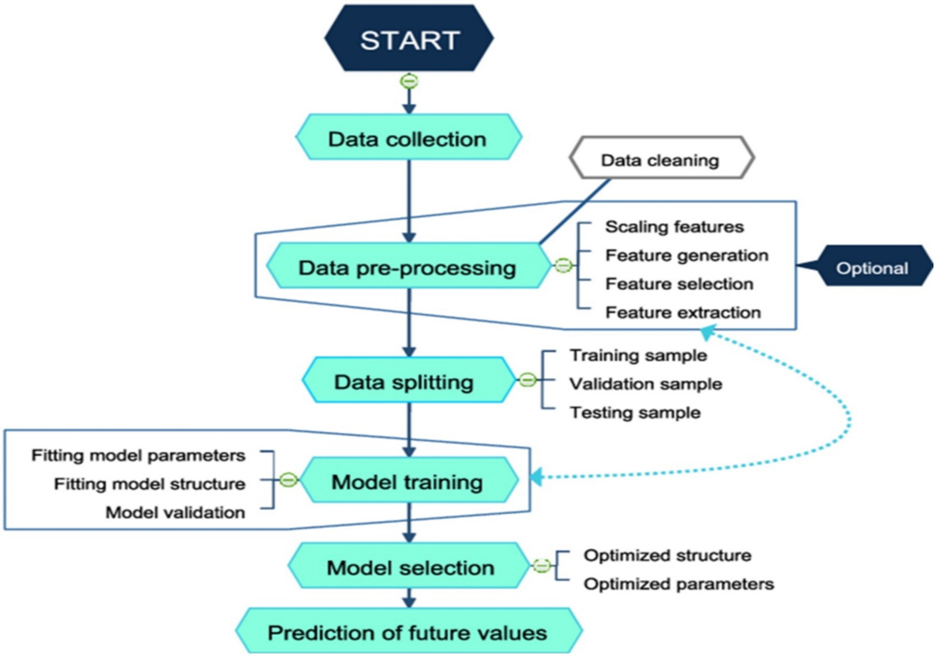


Figure 1 Layout Steps

Methods

Dataset

The heart disease prediction study leverages the Heart Disease Dataset from the UCI Machine Learning Repository, (3) a well-known and reputable source for machine learning datasets. Originally collected in 1987 by the Cleveland Clinic Foundation, this dataset is a cornerstone in medical machine learning research, providing 303 samples with 76 features from patients across multiple sources, including Cleveland, Hungary, Switzerland, and the VA Long Beach datasets. The dataset contains a wealth of information, including patient demographics, medical conditions, and test results. However, for this study, the processed version of the dataset was utilized, which narrows the scope to 14 relevant features: 13 attributes and a target variable indicating the presence or absence of heart disease. The processed version simplifies the data, focusing on the most meaningful features, making it highly suitable for machine learning workflows while preserving the dataset's integrity.

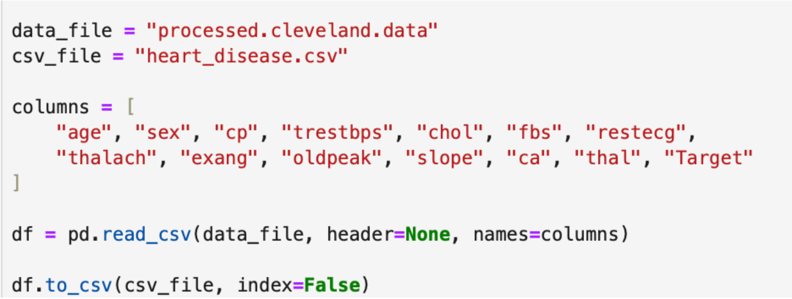
The processed dataset includes a balanced mix of demographic data, such as age and gender, medical measurements like blood pressure and cholesterol levels, and test results such as electrocardiographic findings and exercise-induced angina. The target variable categorizes patients into “sick” (165 samples labeled as 1) and “healthy” (138 samples labeled as 0), allowing for binary classification tasks. These features are crucial in predicting heart disease and provide the foundation for robust modeling. A summary of the dataset features is presented below

Feature Name	Description	Type
age	Age of the patient (in years).	Numeric
sex	Gender of the patient (1 = male; 0 = female).	Categorical
cp	Chest pain type (1 = typical angina; 2 = atypical angina; 3 = non-anginal pain; 4 = asymptomatic). Categorical	Categorical
trestbps	Resting blood pressure (in mm Hg).	Numeric
chol	Serum cholesterol (in mg/dl).	Numeric
fbs	Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)	Categorical
restecg	Resting electrocardiographic results (0 = normal; 1 = ST-T wave abnormality; 2 = hypertrophy).	Categorical
thalach	Maximum heart rate achieved	Numeric
exang	Exercise-induced angina (1 = yes; 0 = no)	Categorical
oldpeak	Slope of the peak exercise ST segment	Categorical
slope	Slope of the peak exercise ST segment	Numeric
ca	Number of major vessels (0-3) colored by fluoroscopy	Numeric
thal	Thalassemia (3 = normal; 6 = fixed defect; 7 = reversible defect).	Categorical
target	Diagnosis of heart disease (1 = presence; 0 = absence).	Categorical

Analysis and Results

Dataset Preparation

The Heart Disease dataset was prepared to ensure high-quality, structured, and meaningful data for analysis. Initially, the dataset, sourced from the UCI Machine Learning Repository, was in a non-CSV format. It was converted into a CSV file using Python, with appropriate column names assigned for better usability. This step ensured the dataset was well-organized and accessible for further processing.



```
data_file = "processed.cleveland.data"
csv_file = "heart_disease.csv"

columns = [
    "age", "sex", "cp", "trestbps", "chol", "fbs", "restecg",
    "thalach", "exang", "oldpeak", "slope", "ca", "thal", "Target"
]

df = pd.read_csv(data_file, header=None, names=columns)

df.to_csv(csv_file, index=False)
```

Figure 2 Conversion

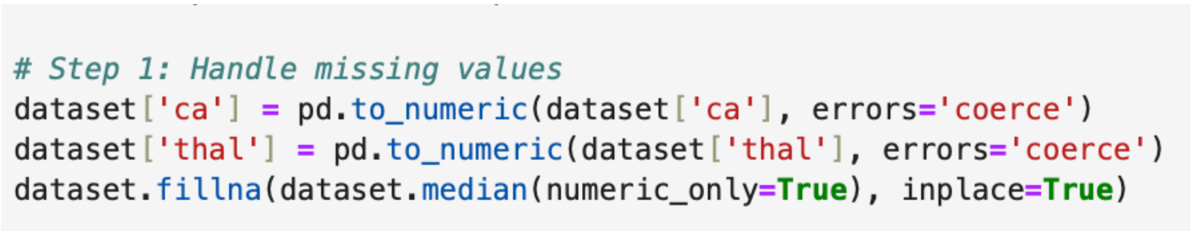
This prepared file served as the foundation for subsequent cleaning and preprocessing tasks. The dataset consisted of 14 features, including patient demographics, medical measurements, and test results, along with a target variable indicating the presence of heart disease.

Data preprocessing

The preprocessing of the heart disease dataset involved multiple steps to ensure data quality and suitability for predictive modeling. These steps focused on addressing missing values, handling outliers, encoding categorical variables, and enhancing feature interpretability. The outlier detection findings were integrated into the preprocessing workflow to retain medically significant data points while mitigating their impact on the model.

1. *Handling Missing Values*

Missing values were identified in the *ca* and *thal* columns. To address this, non-numeric values in these columns were first converted to NaN for easier handling. The missing values were then imputed using the median of their respective columns. This approach was chosen to maintain data consistency and avoid skewing the overall distribution.

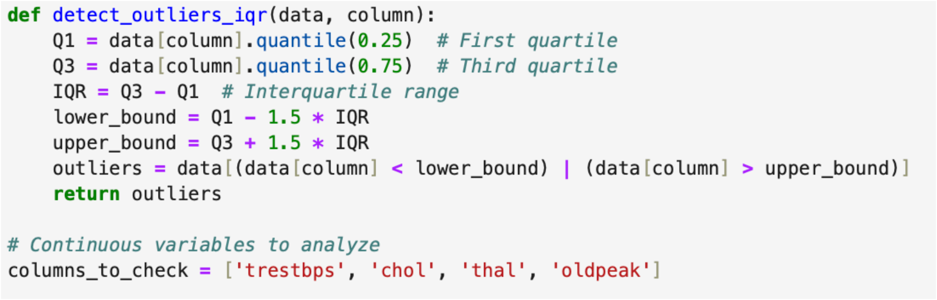


```
# Step 1: Handle missing values
dataset['ca'] = pd.to_numeric(dataset['ca'], errors='coerce')
dataset['thal'] = pd.to_numeric(dataset['thal'], errors='coerce')
dataset.fillna(dataset.median(numeric_only=True), inplace=True)
```

Figure 3 Handling Missing values

2.*Outliners Detection*

Outliers in trestbps, chol, and oldpeak columns were detected using the Interquartile Range (IQR) method. These medically significant outliers were retained to capture edge cases. The Robust Scaler was applied to minimize their influence during model training for a balanced analysis.

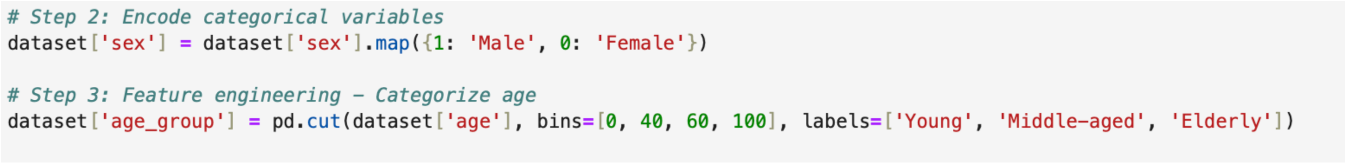


```python
def detect_outliers_iqr(data, column):
    Q1 = data[column].quantile(0.25)  # First quartile
    Q3 = data[column].quantile(0.75)  # Third quartile
    IQR = Q3 - Q1  # Interquartile range
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = data[(data[column] < lower_bound) | (data[column] > upper_bound)]
    return outliers

# Continuous variables to analyze
columns_to_check = ['trestbps', 'chol', 'thal', 'oldpeak']
```

Figure 4 Outlines detection

3. Encoding

Categorical variables were encoded for interpretability and analysis. The sex column was converted to “Male” and “Female,” and the target column to binary (0 for absence, 1 for presence of heart disease). These transformations made the data meaningful and suitable for modeling.

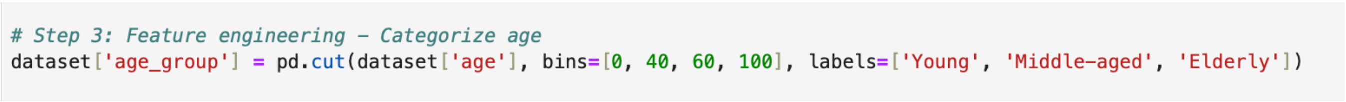


```python
# Step 2: Encode categorical variables
dataset['sex'] = dataset['sex'].map({1: 'Male', 0: 'Female'})

# Step 3: Feature engineering - Categorize age
dataset['age_group'] = pd.cut(dataset['age'], bins=[0, 40, 60, 100], labels=['Young', 'Middle-aged', 'Elderly'])
```

Figure 5 Encode Categories

4.Feature Engineering

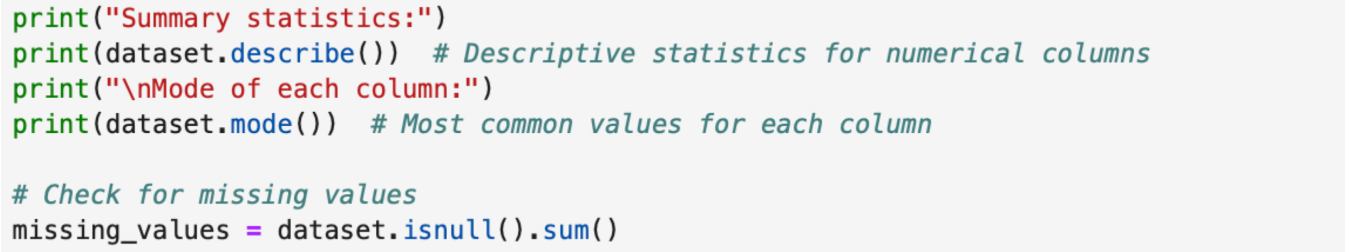
Feature engineering improved dataset interpretability and provided insights. Age was grouped into Young (0-40), Middle-aged (40-60), and Elderly (60-100) groups for demographic analysis and uncovering heart disease patterns across age ranges.



```python
# Step 3: Feature engineering - Categorize age
dataset['age_group'] = pd.cut(dataset['age'], bins=[0, 40, 60, 100], labels=['Young', 'Middle-aged', 'Elderly'])
```

Figure 6 Feature Engineering

Dataset underwent quality assessment, including descriptive statistics and missing value analysis. Outliers were retained as they represent medically significant edge cases. Mode analysis showed higher Male participation and middle-aged dominance



```
print("Summary statistics:")
print(dataset.describe())  # Descriptive statistics for numerical columns
print("\nMode of each column:")
print(dataset.mode())  # Most common values for each column

# Check for missing values
missing_values = dataset.isnull().sum()
```

Figure 7 Overall Statistics

Cleaned dataset saved in enhanced formats for analysis and visualization. Preprocessing ensured dataset consistency and readiness for predictive modeling.

Exploratory Data Analysis

The EDA of the heart disease dataset was conducted using both Python and Tableau to gain a thorough understanding of the data. The analysis focused on feature distributions, relationships, and trends, with visualizations highlighting key insights. (4)

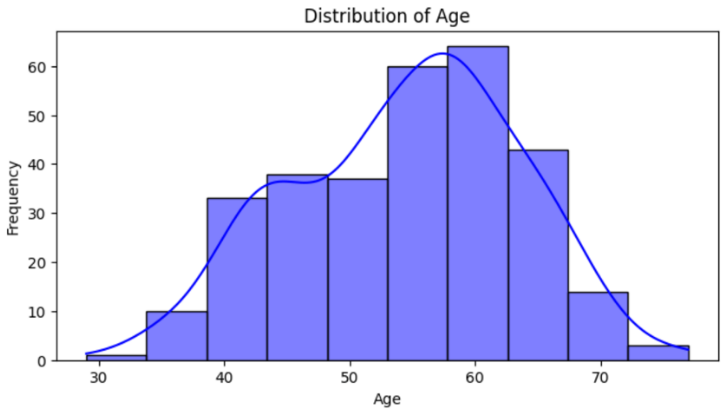


Figure 8 The age distribution Chart

The numerical feature distributions provided key insights. Most patients are between 50-60 years old, with fewer cases in younger and older demographics. Cholesterol values are skewed, with notable outliers above 400 mg/dL, suggesting hypercholesterolemia. Oldpeak (ST depression) values are concentrated between 0 and 2, with higher values indicating severe ischemia. Visualizations, like a histogram for age and a boxplot for cholesterol, clearly illustrate these patterns.

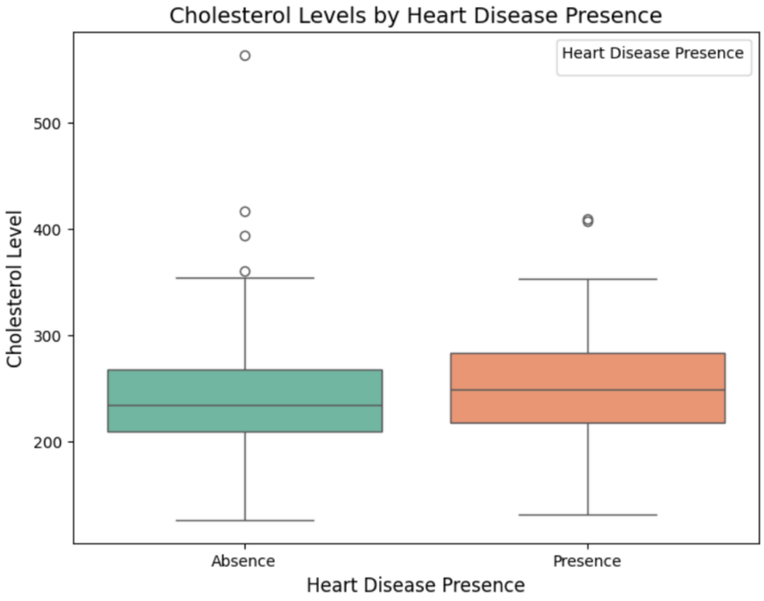


Figure 9 Boxplot for Cholesterol level

The dataset revealed key insights into categorical features. Males dominate the sex distribution (200 vs. 100 females). “Asymptomatic” chest pain type is most common, often associated with severe heart conditions. Most patients have normal resting ECG results. Visualizations, including a Tableau bar chart highlighting gender imbalance and a Python count plot displaying chest pain type distribution, provide an in-depth understanding of the dataset’s categorical features.

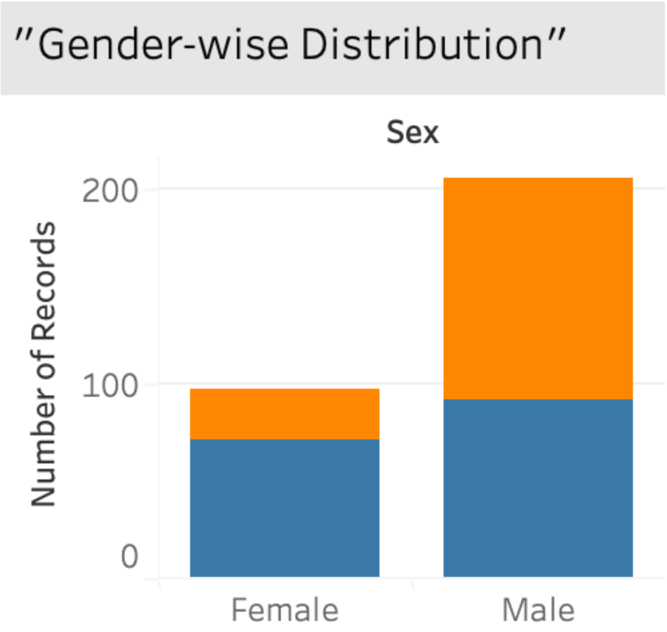


Figure 10 Gender Distribution

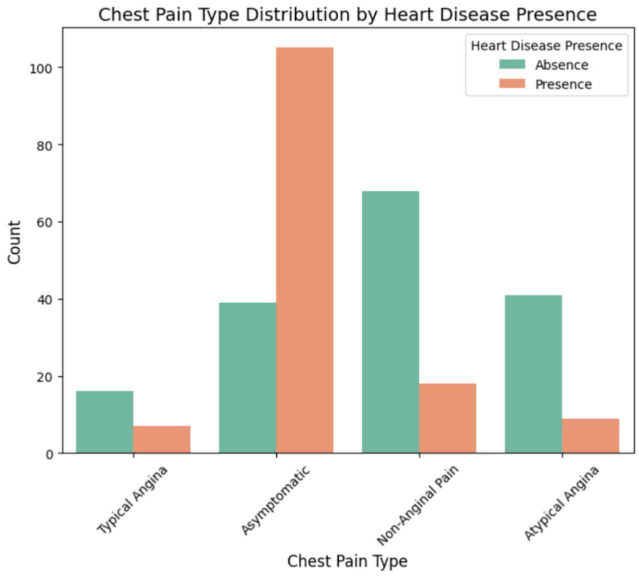


Figure 11 Chest Pain Type Bar Chart

Correlation analysis revealed moderate correlations between features and the target variable. Oldpeak (ST depression) and ca (number of major vessels) showed significance as predictors. Age negatively correlated with max heart rate (-0.39), reflecting age-related decline. A Python-generated correlation heatmap and Tableau scatterplot visualized these relationships. These visualizations provide a comprehensive view of feature interactions and predictive value.

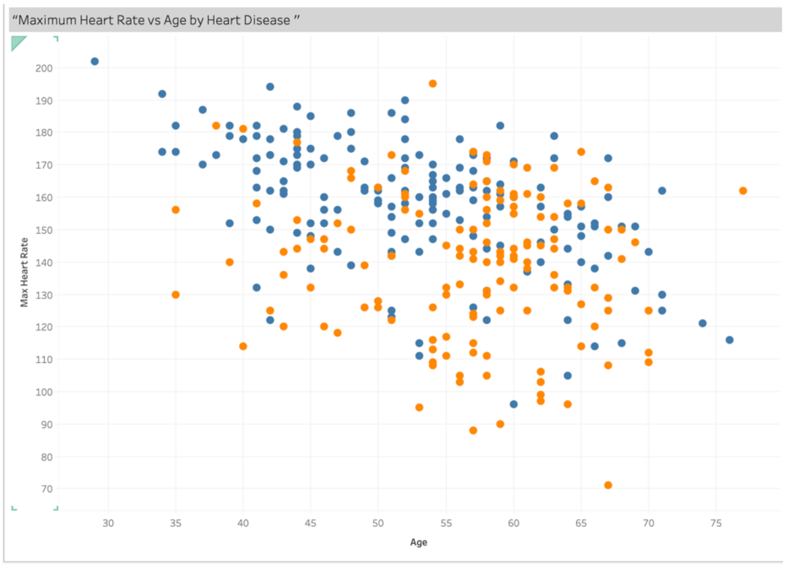


Figure 12 Scatter Plot

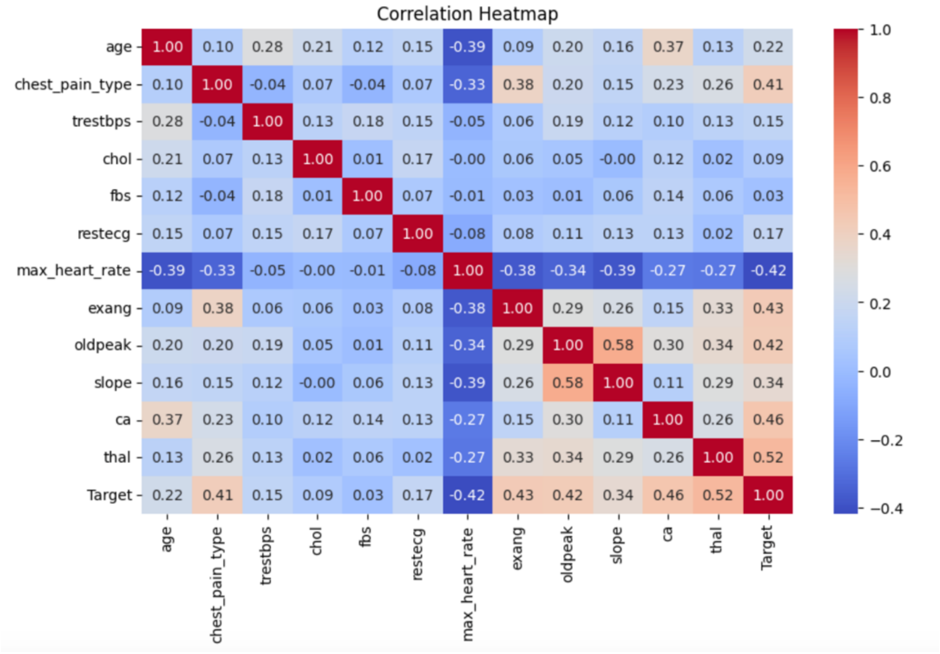


Figure 13 Correlation Heatmap

The analysis revealed that higher oldpeak values and a greater number of ca vessels were strongly linked to heart disease. Additionally, patients with lower maximum heart rates were more likely to have the disease. To illustrate these findings, a Python boxplot showed the correlation between higher oldpeak values and heart disease, while a pairplot highlighted the relationships and interactions between age, cholesterol, and heart disease. These visualizations effectively demonstrated the predictors of heart disease in the dataset.

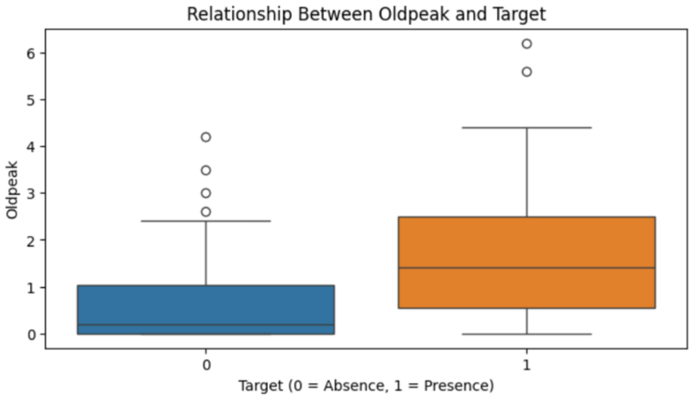


Figure 14 Boxplot for Old peak vs Target

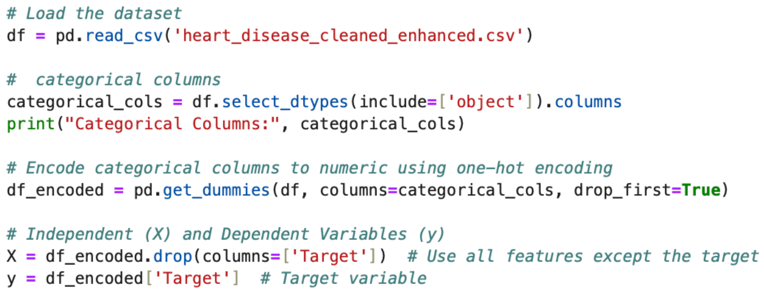
Data Modelling, Visualization and Hyperparameters Tuning

Data modeling and visualization are key for extracting insights and patterns from datasets. I analyzed a heart disease prediction dataset with the dependent variable (target) indicating heart disease and independent variables like age, cholesterol etc. Categorical variables like sex and thalassemia were encoded using one-hot encoding.

Three machine learning models—Logistic Regression, K-Nearest Neighbors (KNN), and Random Forest—were developed. Each model was trained on a dataset scaled using Standard Scaler for improved performance. The dataset was split into 80% training and 20% testing for robust evaluation.

Data Preprocessing

Data preprocessing cleans the input for the model. We load the dataset, identify categorical features, and convert them to numerical format using one-hot encoding.

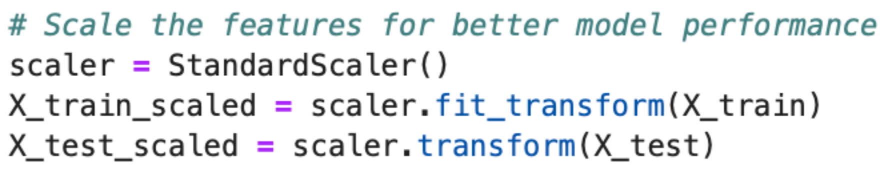


```
# Load the dataset
df = pd.read_csv('heart_disease_cleaned_enhanced.csv')

#  categorical columns
categorical_cols = df.select_dtypes(include=['object']).columns
print("Categorical Columns:", categorical_cols)

# Encode categorical columns to numeric using one-hot encoding
df_encoded = pd.get_dummies(df, columns=categorical_cols, drop_first=True)

# Independent (X) and Dependent Variables (y)
X = df_encoded.drop(columns=['Target'])  # Use all features except the target
y = df_encoded['Target']  # Target variable
```

Figure 15 preprocessing Step 1

Feature Scaling

StandardScaler is used to standardize features before training, ensuring they are on the same scale and improving model convergence and accuracy.

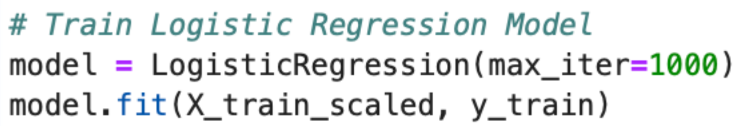


```
# Scale the features for better model performance
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Figure 16 preprocessing Step 2

Logistic Regression Model

The Logistic Regression model is a widely used linear classification method that predicts probabilities using the sigmoid function, mapping inputs to values between 0 and 1. It is particularly effective for binary classification problems and provides interpretable results.(5) The model is trained on scaled data to ensure equal feature contribution, with a max_iter of 1000 for efficient convergence and minimal errors.



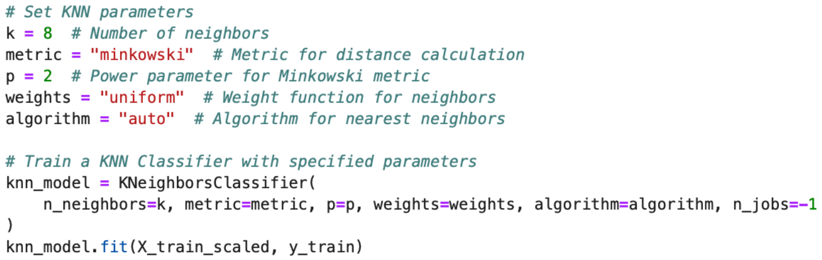
```
# Train Logistic Regression Model
model = LogisticRegression(max_iter=1000)
model.fit(X_train_scaled, y_train)
```

Figure 17 Training Logistic Model

K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a straightforward and intuitive machine learning method used for both classification and regression. KNN predicts the outcome by considering the 'k' closest data points to an input and determining the result based on the majority class or average value of those neighbors. It's easy to understand and suitable for smaller datasets, but scaling features ensures equal contribution to distance calculations. After systematic hyperparameter tuning, the optimal configuration was determined as k=8 (number of neighbors) (6), Minkowski distance metric with p=2 (Euclidean distance), and uniform weights.

KNN is distance-based, so feature scaling ensures all features contribute equally to distance calculations. Without scaling, features with larger ranges (e.g., chol) dominate distance calculations, leading to biased results.



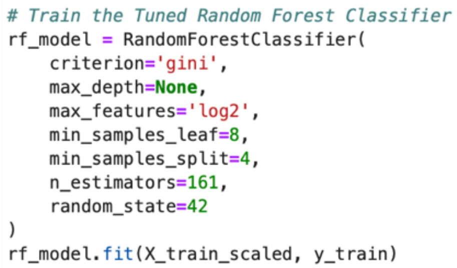
```
# Set KNN parameters
k = 8  # Number of neighbors
metric = "minkowski"  # Metric for distance calculation
p = 2  # Power parameter for Minkowski metric
weights = "uniform"  # Weight function for neighbors
algorithm = "auto"  # Algorithm for nearest neighbors

# Train a KNN Classifier with specified parameters
knn_model = KNeighborsClassifier(
    n_neighbors=k, metric=metric, p=p, weights=weights, algorithm=algorithm, n_jobs=-1
)
knn_model.fit(X_train_scaled, y_train)
```

Figure 18 Training KNN

Random Forest

The Random Forest model, an ensemble learning technique, combines multiple decision trees to improve accuracy and reduce overfitting. It creates a “forest” of trees trained on random data and features. The final prediction is determined by aggregating results from all trees, making it robust and effective for classification.

The Random Forest Classifier was trained with optimized hyperparameters (7) for high performance and generalizability. It used the gini criterion to measure impurity and create splits, with 161 trees in the ensemble for accuracy. Randomly selecting features for each split promoted diversity and reduced overfitting. Minimum of 4 samples for splitting and 8 for forming leaves-controlled tree complexity. These tuned parameters resulted in a robust and reliable model.

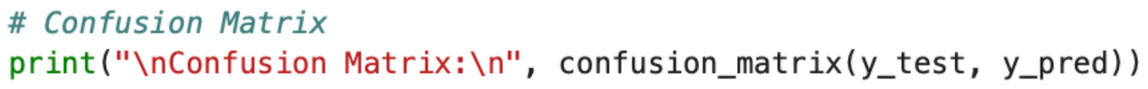


```
# Train the Tuned Random Forest Classifier
rf_model = RandomForestClassifier(
    criterion='gini',
    max_depth=None,
    max_features='log2',
    min_samples_leaf=8,
    min_samples_split=4,
    n_estimators=161,
    random_state=42
)
rf_model.fit(X_train_scaled, y_train)
```

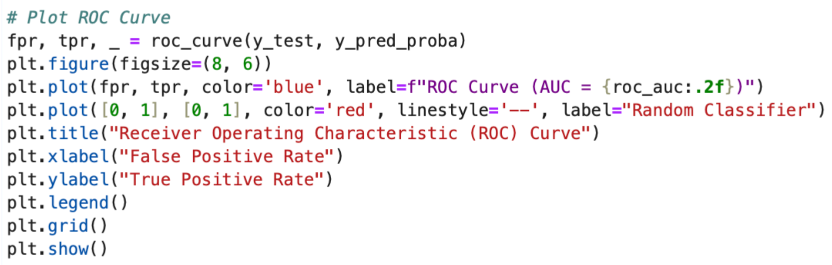
Figure 19 Training Random Forest

Confusion Matrix and ROC Curve

The confusion matrix visually shows the number of correct and incorrect predictions, while the ROC curve evaluates the trade-off between sensitivity and specificity.



```
# Confusion Matrix
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

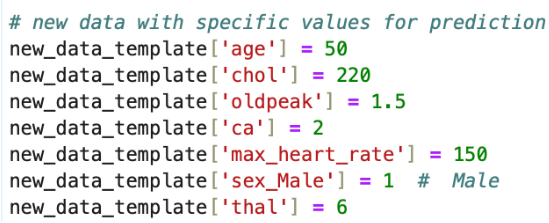


```
# Plot ROC Curve
fpr, tpr, _ = roc_curve(y_test, y_pred_proba)
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='red', linestyle='--', label="Random Classifier")
plt.title("Receiver Operating Characteristic (ROC) Curve")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend()
plt.grid()
plt.show()
```

Figure 20 Plotting ROC

Predictions for New Data

Model tested on new data to simulate real-world application. Feature values updated to reflect hypothetical scenarios.

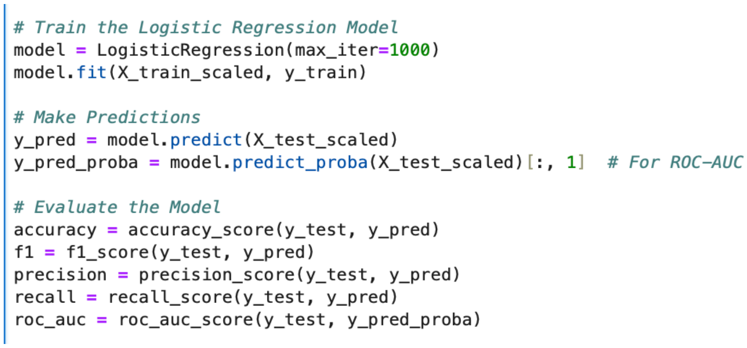


```
# new data with specific values for prediction
new_data_template['age'] = 50
new_data_template['chol'] = 220
new_data_template['oldpeak'] = 1.5
new_data_template['ca'] = 2
new_data_template['max_heart_rate'] = 150
new_data_template['sex_Male'] = 1  #  Male
new_data_template['thal'] = 6
```

Figure 21 New Data For predictions

Evaluation

To enhance our proposed system, i conducted experiments on the Cleveland heart disease database using three ML algorithms. These algorithms were evaluated using various performance metrics: accuracy, recall, precision, F1 score, and AUC. I trained multiple algorithms and selected the best model. The goal was to assess the predictive model's performance in classifying patients with or without heart disease. The results were evaluated using key classification metrics and compared to previous work.



```
# Train the Logistic Regression Model
model = LogisticRegression(max_iter=1000)
model.fit(X_train_scaled, y_train)

# Make Predictions
y_pred = model.predict(X_test_scaled)
y_pred_proba = model.predict_proba(X_test_scaled)[:, 1]  # For ROC-AUC

# Evaluate the Model
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred_proba)
```

Figure 22 Evaluations

The Logistic Regression model accurately predicted heart disease based on clinical parameters, achieving an 89% accuracy and a 0.92 ROC-AUC score. A confusion matrix heatmap showed 27 true positives, 27 true negatives, 5 false negatives, and 2 false positives. The ROC curve validated the model's reliability, outperforming a random classifier. Predictions on new data confirmed its ability to identify moderate and high-risk cases. Thus, it's a valuable tool for heart disease prediction and clinical decision support.

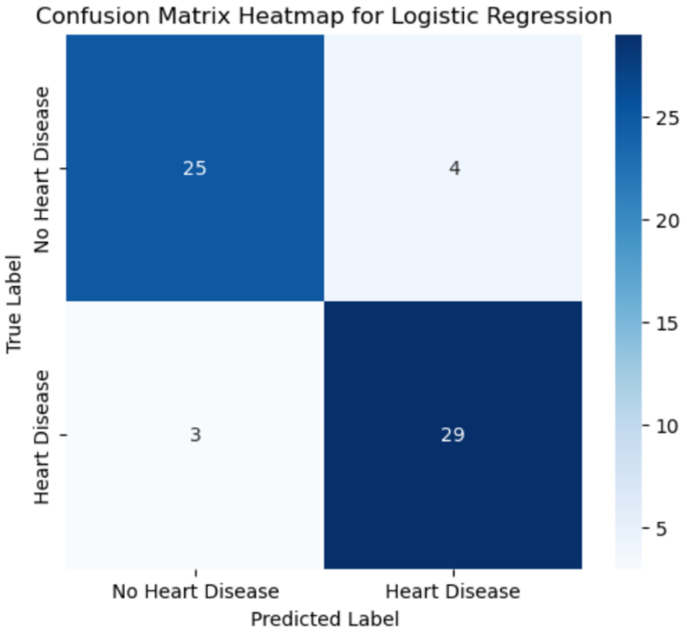


Figure 23 Confusion matrix plot 1

K-Nearest Neighbors (KNN) achieved 89% test accuracy with cross-validation confirming k=8's robustness. ROC curves showed sensitivity and specificity trade-offs, capturing non-linear data relationships.

The Random Forest model also achieved 89% accuracy and a 94% ROC-AUC score, demonstrating strong predictive power.

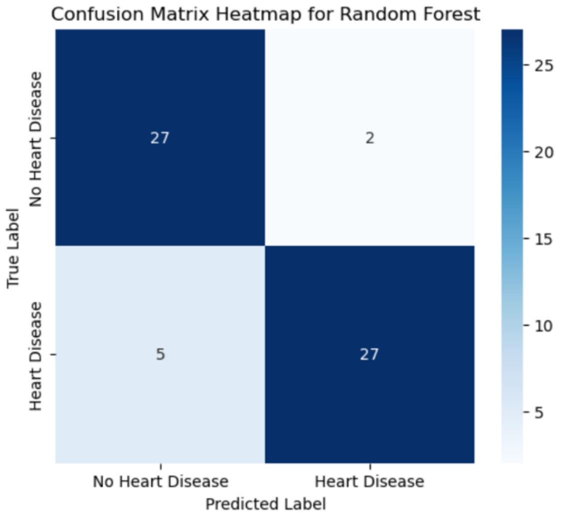


Figure 24 Confusion matrix plot 2

The confusion matrix showed 27 true positives, negatives, false negatives, and positives. The model's precision was 93%, its recall 84%, and its F1-score 88%.

The ROC curve confirmed the model's ability to distinguish classes. Feature importance analysis revealed key factors driving heart disease predictions. While strong, improving recall by reducing false negatives enhanced the model's utility for critical healthcare decisions.

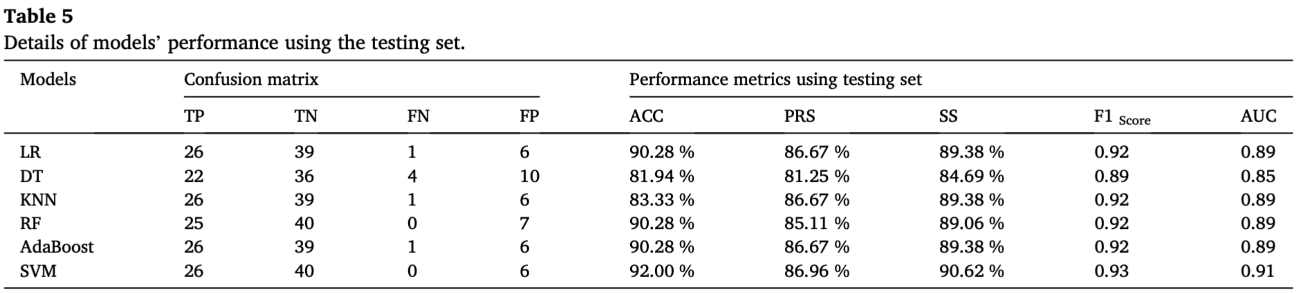


Table 5
Details of models' performance using the testing set.

Models	Confusion matrix				Performance metrics using testing set				
	TP	TN	FN	FP	ACC	PRS	SS	F1 $_{\text{Score}}$	AUC
LR	26	39	1	6	90.28 %	86.67 %	89.38 %	0.92	0.89
DT	22	36	4	10	81.94 %	81.25 %	84.69 %	0.89	0.85
KNN	26	39	1	6	83.33 %	86.67 %	89.38 %	0.92	0.89
RF	25	40	0	7	90.28 %	85.11 %	89.06 %	0.92	0.89
AdaBoost	26	39	1	6	90.28 %	86.67 %	89.38 %	0.92	0.89
SVM	26	40	0	6	92.00 %	86.96 %	90.62 %	0.93	0.91

Figure 25 previous work Results on Testing set

Performance metrics Testing set

Models	Accuracy	Precision	Recall	F1-score
Logistic Regression	0.89	0.88	0.91	0.89
KNN	0.89	0.93	0.84	0.89
Random Forest Model	0.89	0.93	0.84	0.89

These are the results from the prediction models using the three models. KNN has better accuracy from previous research, so I'll choose KNN or Random Forest for creating a predicting system.

Limitations and Challenges

Balancing precision and recall, managing false positives and negatives, and addressing false negatives, which could delay treatment, were key challenges in developing Logistic Regression, K-Nearest Neighbors (KNN), and Random Forest models. Logistic Regression had 3 false negatives and 4 false positives, KNN and Random Forest had 5 false negatives and 2 false positives each.

Hyperparameter tuning and feature scaling improved performance, but achieving perfect balance remained challenging, especially in complex healthcare scenarios. Random Forest had strong results with 89% accuracy and a 94% ROC-AUC score but struggled with false negatives. Logistic Regression was simpler and more interpretable but had slightly lower sensitivity. Testing on unseen data highlighted limitations in generalizability, suggesting further validation on larger, more diverse datasets for reliable real-world deployment.

Conclusion

This project provided a comprehensive learning experience in developing machine learning models for heart disease prediction. I selected and analyzed datasets, identifying key features and framing meaningful questions. Implemented models like Logistic Regression, KNN, and Random Forest, understanding data preprocessing, feature scaling, and hyperparameter tuning. Random Forest offered higher accuracy, Logistic Regression simplicity, and KNN computational efficiency.

I used Tableau to create visualizations highlighting trends and relationships between features, providing clear insights and then, optimized model performance for real-world applications, balancing precision and recall.

This work can be expanded into a user-friendly software application predicting heart disease likelihood based on input parameters, integrating real-time data and advanced deep learning models for improved accuracy and scalability. Overall, the project strengthened my technical, analytical, and visualization skills, showcasing the impact of machine learning and data visualization in healthcare.

Reference List

1. **World Health Organization** (2021) *cardiovascular diseases (CVDs): Key facts*. Available at: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) (11 June 2021).

2. H.V. Denysyuk, R.J. Pinto, P.M. Silva, R.P. Duarte, F.A. Marinho, L. Pimenta, A.J. Gouveia, N.J. Gonçalves, P.J. Coelho, E. Zdravevski, P. Lameski, V. Leithardt, N.M. Garcia, I.M. Pires, (2023) Algorithms for automated diagnosis of cardiovascular diseases based on ECG data: a comprehensive systematic review, Heliyon 9 e13601, Available at :https://www.cell.com/heliyon/fulltext/S2405-8440(23)00808-3?_returnURL=https%3A%2F%2Flinkinghub.elsevier.com%2Fretrieve%2Fpii%2FS2405844023008083%3Fshowall=true

3. **UC Irvine Machine Learning Repository** (2024) *Heart Disease Dataset*. Available at: https://archive.ics.uci.edu/dataset/45/heart+disease(October 15, 2024).

4. **Indrakumari, R., Poongodi, T. and Jena, S.R.** (2020) 'Heart disease prediction using exploratory data analysis', *Procedia Computer Science*, 173, pp. 130–139. Available at: https://pdf.sciencedirectassets.com

5. Yingjie Zhang et al 2021 'Logistic Regression Models in Predicting Heart Disease' *Journal of Physics: Conference Series*, 1769, 012024 available at :https://iopscience.iop.org/article/10.1088/1742-6596/1769/1/012024/pdf

6. **Mrs. Mini Jain1 , Prof. Chetan Gupta** August 2018 Vol. 7, Issue 8, A Review and Analysis of Centroid Estimation in k-means Algorithm pp.43-45 https://ijarcce.com/wp-content/uploads/2018/09/IJARCCE.2018.789.pdf

7. Justus A Ilemobayo, ,Olamide Durodola (2021) 'Hyperparameter tuning techniques in machine learning for medical predictions', Journal of Engineering Research and Reports, vol. 26, no. 6, pp. 388-395 Available at:https://www.researchgate.net/publication/381255284_Hyperparameter_Tuning_in_Machine_Learning_A_Comprehensive_Review

Appendix

https://ssu-my.sharepoint.com/:v:/g/personal/0madaa90_solent_ac_uk/Ec35xzorpaFFqEmoLGQggEQBdMEbp7rdEsVo1JwcWv6V1g?email=raza.hasan%40solent.ac.uk&e=drgqZw

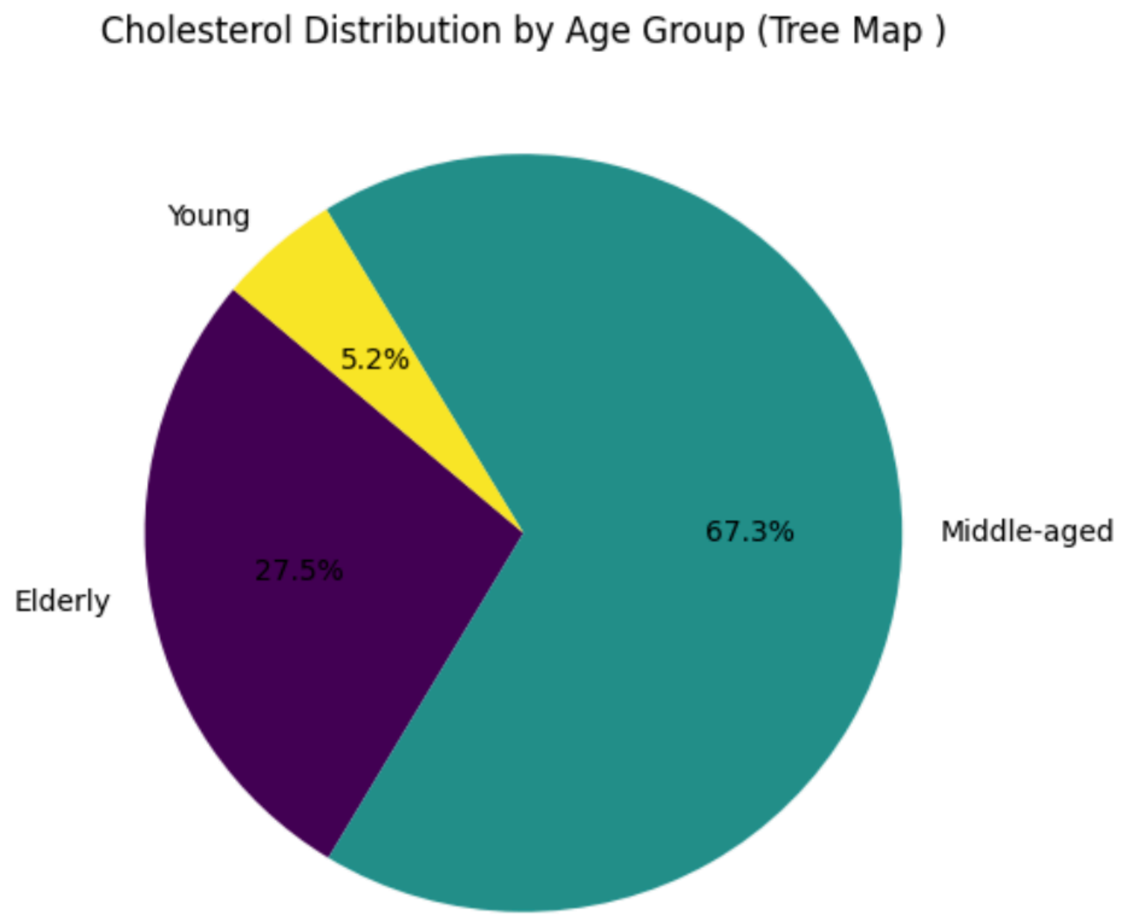
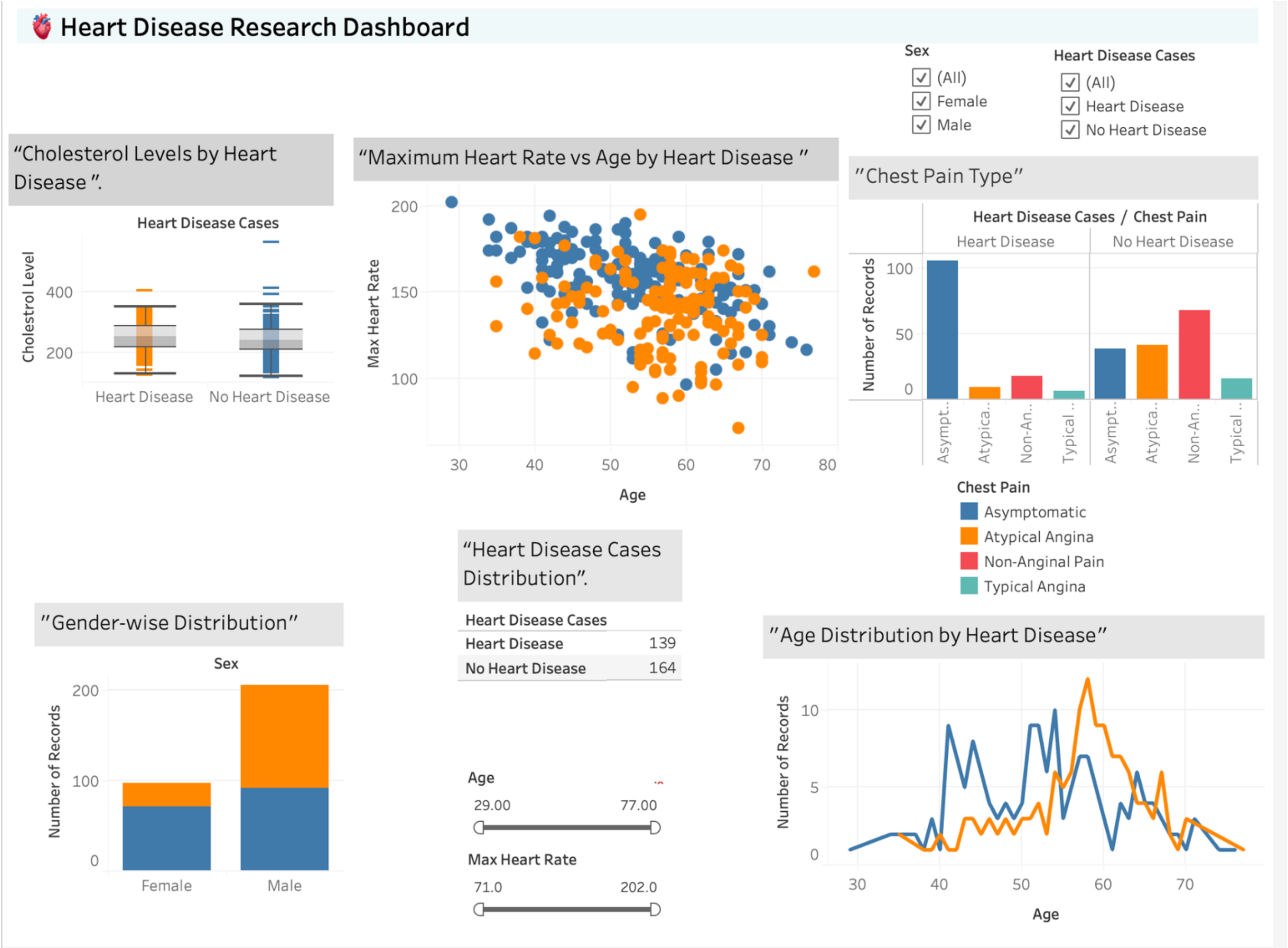


Figure 26 Tree Map of Cholesterol by Age



Heart Disease Research Dashboard

Sex

☑ (All)
☑ Female
☑ Male

Heart Disease Cases

☑ (All)
☑ Heart Disease
☑ No Heart Disease

“Cholesterol Levels by Heart Disease ”.

“Maximum Heart Rate vs Age by Heart Disease ”

”Chest Pain Type”

“Heart Disease Cases Distribution”.

Heart Disease Cases	
Heart Disease	139
No Heart Disease	164

Age

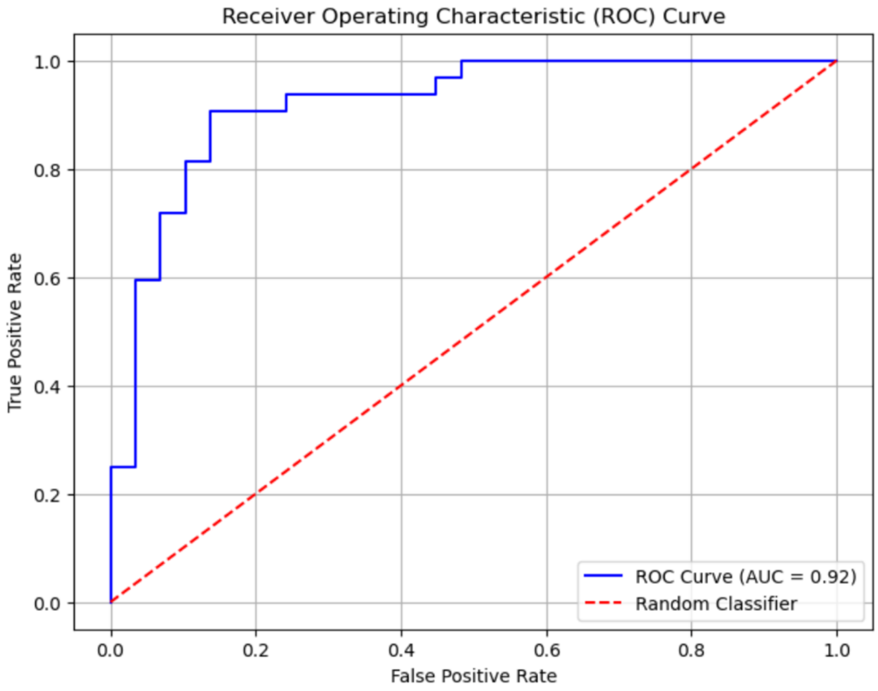
29.00 77.00

Max Heart Rate

71.0 202.0

”Gender-wise Distribution”

”Age Distribution by Heart Disease”



```
Prediction for new data: No Heart Disease

Prediction for new data: Heart Disease Present
```

Figure 27 Logistic Model ROC