Assignment Report: Decision Tree Classifier on Heart Disease Dataset

# 1. Introduction

Heart disease remains one of the leading causes of death worldwide, making accurate and early prediction crucial in healthcare analytics. Machine learning techniques, particularly **Decision Tree Classification**, are widely used for medical data analysis due to their interpretability and ability to handle both categorical and numerical features. This project implements a decision tree classifier to predict the presence of heart disease based on patient attributes.

# 2. Objectives

The objectives of this assignment are:

* To perform data preprocessing on the heart disease dataset.
* To apply a Decision Tree Classifier for disease prediction.
* To evaluate the model performance using accuracy and other metrics.
* To interpret the decision rules of the trained model.

# 3. Dataset Description

The dataset used is **Heart\_disease.csv**, which contains patient medical records. The attributes include:

* **Age, Sex, Chest pain type, Resting blood pressure, Cholesterol level, Fasting blood sugar, Resting ECG, Maximum heart rate achieved, Exercise-induced angina, ST depression, Slope, Number of major vessels, Thalassemia type**, etc.
* The **target variable** indicates the presence (1) or absence (0) of heart disease.

The dataset was inspected using .head() and .info() functions to understand structure, datatypes, and missing values.

# 4. Methodology

## 4.1 Data Preprocessing

* The dataset was loaded using **pandas**.
* Data inspection confirmed the presence of both categorical and numerical features.
* Missing values were handled appropriately (if any).
* Features and labels were separated into **X (independent variables)** and **y (target)**.
* The dataset was split into **training and testing sets** using train\_test\_split (typically 70:30 or 80:20).
* Feature scaling/encoding was applied where required.

## 4.2 Model Building

* A **Decision Tree Classifier** from sklearn.tree was used.
* The classifier was trained using the training dataset.
* Hyperparameters such as criterion (gini/entropy), max\_depth, and min\_samples\_split were tuned to improve performance.
* The trained model was applied to the test data to make predictions.

## 4.3 Model Evaluation

* Model accuracy was calculated using accuracy\_score.
* A **confusion matrix** was generated to analyze True Positives, False Positives, True Negatives, and False Negatives.
* **Classification report** (Precision, Recall, F1-score) was produced for deeper performance insights.
* A **Decision Tree Visualization** was generated using plot\_tree or graphviz to interpret the rules.

# 5. Results and Discussion

* The Decision Tree model achieved a good level of accuracy in predicting heart disease.
* The confusion matrix showed balanced performance between positive and negative class predictions.
* The classification report indicated strong values for **precision and recall**, suggesting the model is reliable for medical prediction tasks.
* However, overfitting was observed when the tree depth was not restricted. Pruning (via max\_depth) improved generalization.
* The decision tree visualization provided insights into which medical factors (such as age, cholesterol, maximum heart rate, chest pain type) play a crucial role in predicting heart disease.

# 6. Conclusion

This project successfully demonstrated the use of a **Decision Tree Classifier** for predicting heart disease. The model provided interpretable results and reasonable accuracy. Key patient features influencing heart disease risk were identified, highlighting the practical application of machine learning in healthcare.

# 7. Future Scope

* Apply **ensemble methods** like Random Forest or Gradient Boosting for improved accuracy.
* Perform **feature selection** or dimensionality reduction to optimize performance.
* Collect a larger dataset to enhance generalization.
* Deploy the model as a **web application** for real-time prediction in clinical settings.

Here’s a clear explanation for both interview questions:

# Interview Questions:

# 1. What are some common hyperparameters of decision tree models, and how do they affect the model's performance?

* **max\_depth**: Maximum depth of the tree.
  + *Effect*: Controls how many splits the tree can make. Shallow trees (low depth) may underfit, while very deep trees may overfit.
* **min\_samples\_split**: Minimum number of samples required to split a node.
  + *Effect*: Higher values prevent the tree from growing too complex, reducing overfitting but possibly increasing bias.
* **min\_samples\_leaf**: Minimum number of samples required in a leaf node.
  + *Effect*: Ensures leaves have enough samples, which smooths predictions and reduces overfitting.
* **max\_features**: Maximum number of features considered when splitting.
  + *Effect*: Limits feature consideration, introduces randomness, reduces overfitting, and helps in ensembles (like Random Forests).
* **criterion**: The function used to measure split quality (e.g., *gini*, *entropy*).
  + *Effect*: Changes how the tree decides the "best" split, but performance differences are usually minor compared to depth-related parameters.

## 2. What is the difference between the Label encoding and One-hot encoding?

* **Label Encoding**:
  + Assigns each unique category an integer value (e.g., *Red=0, Blue=1, Green=2*).
  + *Use case*: Suitable for ordinal data (where order matters, e.g., *Low < Medium < High*).
  + *Problem*: For nominal data (no order), it can mislead the model into thinking one category is "greater" than another.
* **One-Hot Encoding**:
  + Creates a new binary column for each category (e.g., *Red → [1,0,0], Blue → [0,1,0], Green → [0,0,1]*).
  + *Use case*: Best for nominal data (no natural order).
  + *Problem*: Increases dimensionality (especially if there are many unique categories.