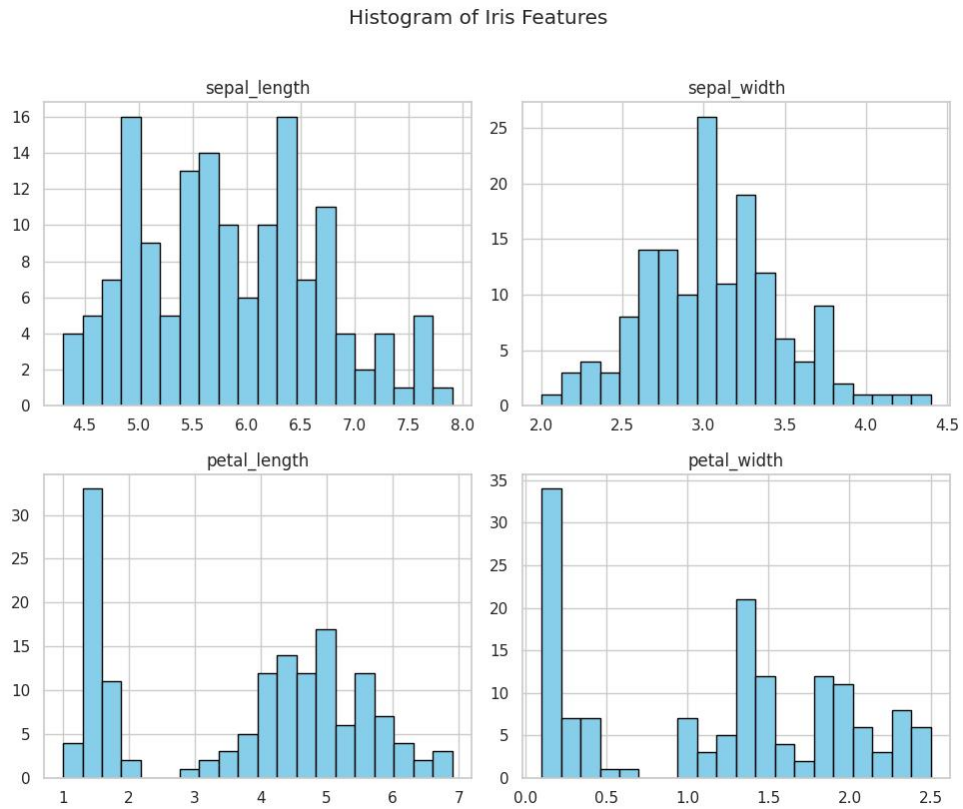
Machine Learning Tasks Report

Gul-e-Rana
DHC-3306
AI & ML Intern

Task 1: Exploring and Visualizing the Iris Dataset

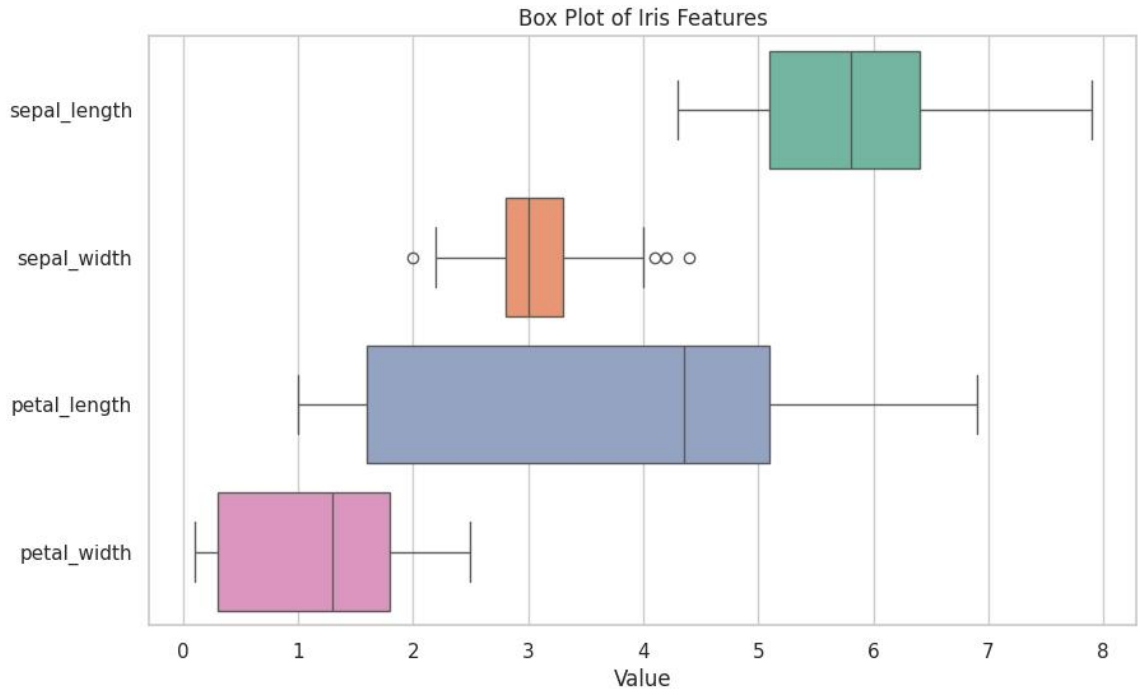


Objective:
Learn how to load, inspect, and visualize the Iris dataset to understand data trends and distributions.

Steps Performed:
- Loaded dataset using seaborn and pandas.
- Printed shape, column names, and first 5 rows using .head().
- Used .info() and .describe() to get summary statistics.
- Visualized relationships using scatter plots, histograms, and box plots to identify trends and outliers.

Tools Used:
- pandas, seaborn, matplotlib

Outcome:



Task 2: Predict Future Stock Prices (Short-Term)

Objective:
Use historical stock data to predict the next day's closing price using Linear Regression and Random Forest.

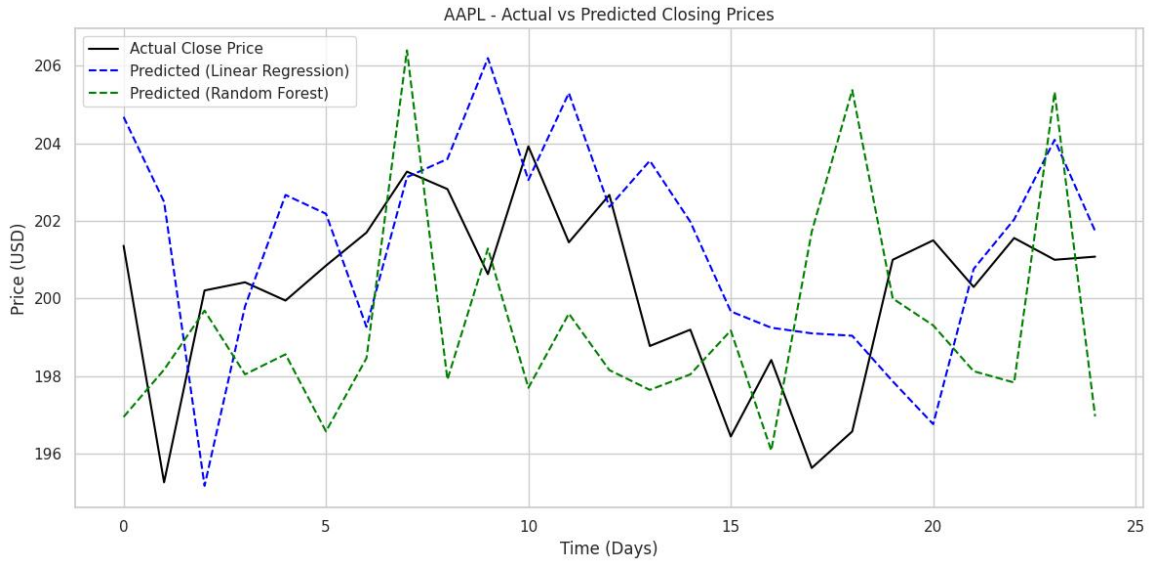
Steps Performed:
- Fetched historical stock data using yfinance (e.g., Apple/Tesla).
- Used features: Open, High, Low, Volume to predict Close price.
- Trained Linear Regression and Random Forest models.
- Compared predictions vs actual prices using line plots.

Tools Used:
- yfinance, pandas, scikit-learn, matplotlib

Outcome:
- Random Forest gave slightly better prediction accuracy due to its handling of non-linear patterns

Explanation:
The models were trained on basic stock indicators to predict the next-day closing

price. Random Forest captured subtle trends better, while Linear Regression showed a linear fit. This visual distinction clarified Random Forest's better performance.



Task 3: Predict Heart Disease (Classification)

Objective:
Build and compare two classification models—**Logistic Regression** and **Decision Tree**—to predict the presence of heart disease using clinical features.

Steps Performed:

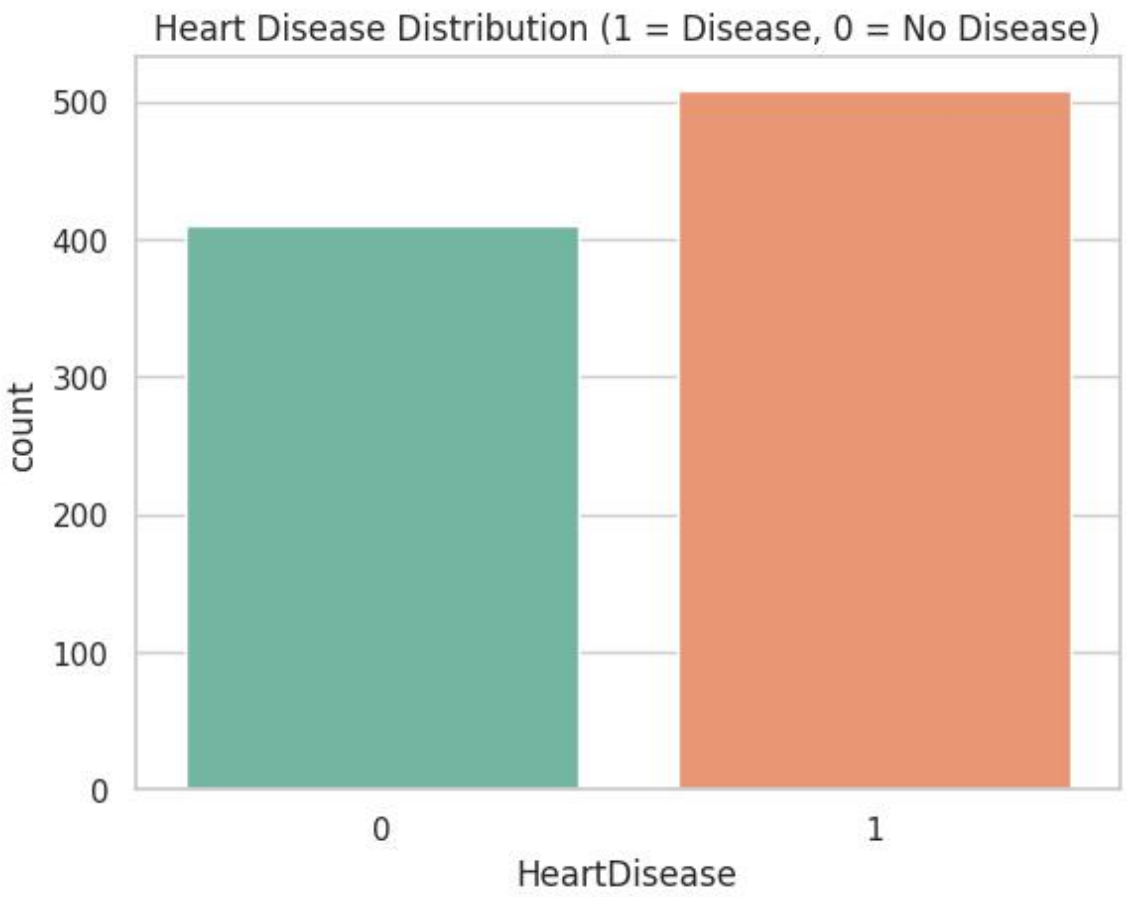
1. Downloaded the “heart-failure-prediction” dataset via `kagglehub` and loaded it into a DataFrame.
2. **Data Inspection & Cleaning:**

Checked dataset shape, columns, and missing values.

Previewed first few rows to verify data integrity.

3. **Exploratory Data Analysis (EDA):**

Plotted the target (`HeartDisease`) distribution.



Feature Preparation:

Separated features (`X`) and target (`y`).

One-hot encoded any categorical columns to ensure all inputs are numeric.

Split into training and testing sets (80/20).

Model Training:

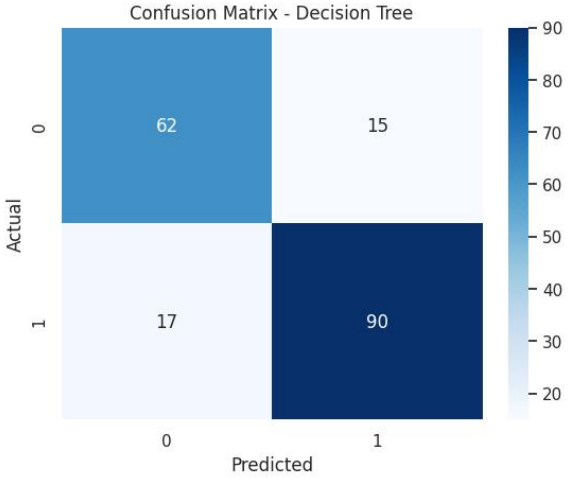
Trained a **Logistic Regression** model (`max_iter=1000`).

Trained a **Decision Tree Classifier** (`random_state=42`).

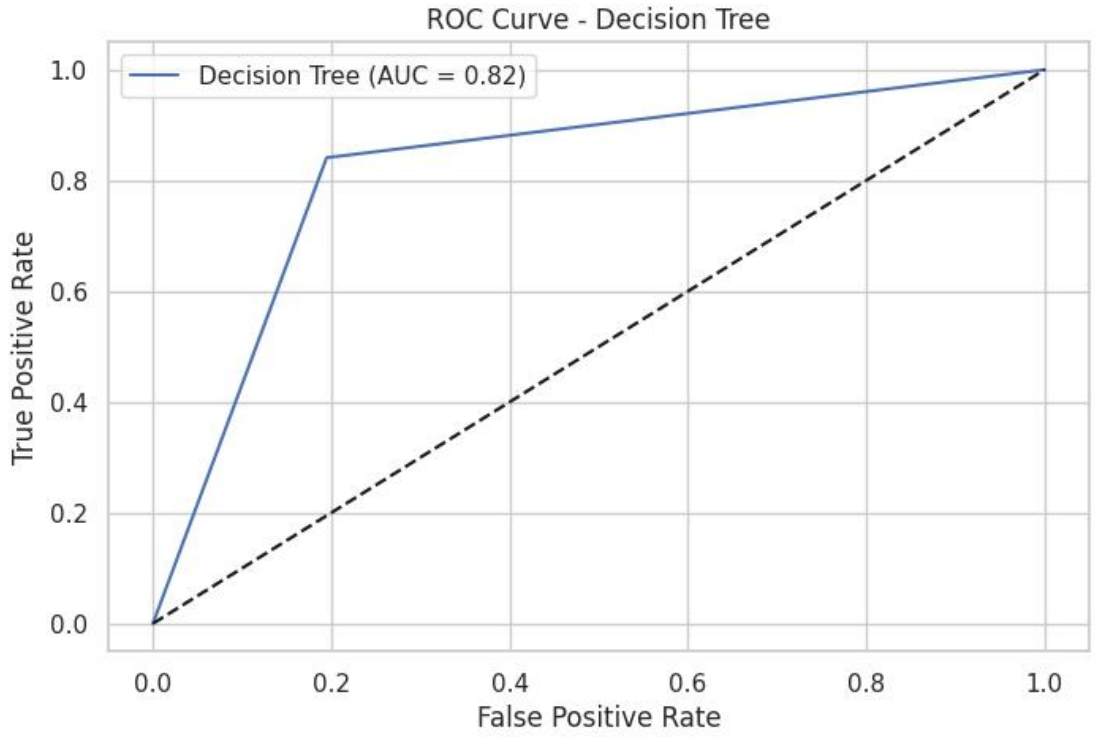
Model Evaluation:

Computed and printed accuracy scores for both models.

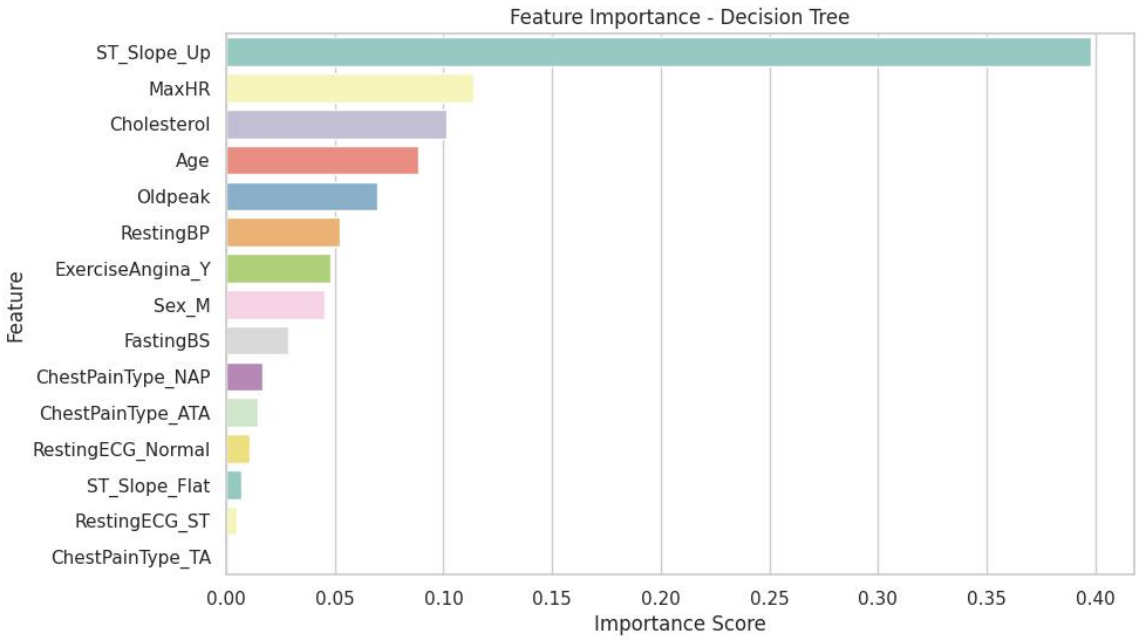
Visualized the Decision Tree's confusion matrix.



Plotted the ROC curve and calculated AUC for the Decision Tree.



Displayed feature importances from the Decision Tree.



Extra Insight:

Compared final accuracies and patterns between the two models.

Tools Used:

`kagglehub`, `pandas`, `numpy`, `seaborn`, `matplotlib`,
`scikit-learn` (for model building and evaluation)

Outcome:

Logistic Regression achieved **85.3%** accuracy.

Decision Tree achieved **82.6%** accuracy.
This indicates a strong linear signal in the data that the Logistic model captured better, while the Decision Tree—though able to fit non-linear patterns—showed slight overfitting.

Explanation:
Logistic Regression fits a linear decision boundary, which generalized well on test data here, reflecting the dataset's underlying linear separability. The Decision Tree model, with its hierarchical splits, picked up more complex interactions but tended to over-specialize on training examples, leading to marginally lower test accuracy. The ROC/AUC and confusion matrix visuals confirm that Logistic Regression offers the more reliable, generalizable classifier for this heart disease dataset.