Heart Disease Prediction

Using Classification Trees

I sincerely apologize. Due to personal health reasons, I am unable to speak clearly, which makes it difficult for me to record a presentation video. I have made the written explanations in the Jupyter notebook as clear as possible. Thank you for being so understanding.

Project Overview

- **Focus**: Predicting heart disease presence using clinical measurements.
- **Type**: Supervised learning with classification trees.
- **Objective**: Build a decision tree model to classify patients (target: 0 = no disease, 1 = disease).
- **Dataset**: Cleveland Heart Disease from UCI Machine Learning Repository.

GIT: https://github.com/Gear2382/Heart-Disease-Prediction

Motivation and Goal

- **Context**: Heart disease is a leading global cause of death; early detection improves outcomes.
- **Aim**: Develop an interpretable decision tree model for predicting risk from routine tests.
- **Goal**: Achieve high accuracy, identify key features, and support preliminary screenings.
- **Evaluation**: Use accuracy, precision, recall, and F1-score; visualize tree structure.

Data Description

- **Source**: UCI Machine Learning Repository [Heart Disease Dataset]
- **Samples**: 303
- **Features**:
  age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal
- **Target**: num
- **Size**: ~5 KB, single table.

Data Import

- **Action**: Load `processed.cleveland.data` and view the first five rows.
- **Code Snippet**:

```
import pandas as pd
df = pd.read_csv('processed.cleveland.data')
print(df.head())
```

- **Purpose**: Initial data exploration to understand structure.

Data Cleaning - Identifying Missing Data

- **Investigation**: Check unique values.
- **Finding**: Question marks (?) represent missing values in `ca` and `thal`.
- **Code Snippet**:

```
print("ca:", df['ca'].unique())
print("thal:", df['thal'].unique())
```

Data Cleaning - Handling Missing Data

- **Approach**: Decide between deleting rows or imputing missing values.
- **Analysis**:
  - 6 rows (1.98%) have missing values in 303 total.
  - 297 rows remain after removal, sufficient for modeling.
- **Action**: Remove rows with missing values.
- **Verification**: Confirm no missing values in the cleaned dataset.
- **Code Snippet**:

```
df_no_missing = df.loc[(df['ca'] != '?’) & (df['thal'] != '?')]
print("shape:", df_no_missing.shape)
print("ca:", df_no_missing['ca'].unique())
print("thal:", df_no_missing['thal'].unique())
```

Exploratory Data Analysis

- **Visualizations**:
  - Correlation matrix: Highlights `ca` and `thal` as key predictors.
  - Box plots: Higher `age` and `cp`, lower `thalach` in diseased patients.
- **Analysis**: Disease linked to older age, higher chest pain types, and lower max heart rate.
- **Imbalance**: ~55% no disease, 45% disease.
- **Conclusions**: Use stratified split; prioritize recall for disease detection.

Model Building and Training

- **Model**: Decision Tree.
  - Suitable for interpretable, non-linear classification.
  - Handles mixed data types, no linearity assumption.
- **Pre-processing**: 80/20 train-test split, stratified.
- **Handling Issues**: Robust to collinearity.
- **Comparison**: Tested against Logistic Regression baseline.
- **Feature Importance**: `thal` and `cp` are most significant.
- **Imbalance**: Mild; evaluated with balanced metrics.

Results and Analysis

- **Summary**: Logistic Regression (LR) outperforms Decision Tree (DT) in heart disease prediction.
- **Metrics**: Accuracy, Precision, Recall, F1-Score, AUC-ROC
- **Rationale**: Accuracy measures overall performance; recall prioritizes disease detection. AUC-ROC shows LR's superior discrimination.
- **Comparison**: LR excels with linear relationships and calibration; DT offers interpretability but struggles with dataset structure.

Discussion and Conclusion

- **Learnings**: LR outperforms DT; key features (`thal`, `ca`, `oldpeak`) align with clinical risks. LR: F1=0.81, AUC=0.95.
- **DT Limitations**: F1=0.79, AUC=0.82; sensitive to small dataset and imbalance, leading to overfitting despite pruning.
- **LR Advantage**: Handles linear patterns better, less overfitting, robust modeling of continuous features (e.g., `chol`, `thalach`), higher AUC (0.95).
- **Suggestions**:
  - Ensemble methods (e.g., voting classifier, Random Forest).
  - Larger dataset or cross-validation to reduce DT overfitting.
- **Conclusion**: LR is more reliable for this dataset due to better performance and generalization. Future work: Investigate hybrid approaches for interpretability and accuracy.