Project Report

(CSE1021) Introduction to Problem Solving & Programming

Slot – A14+D11+D12

Class ID: BL2025260101417

Student Name: Nihal Muhammed

Registration No.: 25BSA10061

Introduction:

Cardiovascular Disease (CVD), or in broader terms Heart Disease remains the leading global cause of death. Early, non-invasive and accurate risk stratification is crucial for timely intervention.

This project utilizes supervised machine learning techniques to develop a reliable and highly optimized Random Forest Classifier model capable of classifying patients as either having or not having heart disease based on common physiological and clinical parameters.

The primary object is to create a robust model that maximizes the F1-Score, ensuring an optimal balance between minimizing missed diagnoses and reducing false alarms.

Key Modules used:

- **Pandas & NumPy:** Core Modules for efficient data loading, cleaning and numerical operations.
- **Pipeline:** Chains all preprocessing and modelling steps into a single, cohesive workflow.
- **SimpleImputer & StandardScaler:** Modules responsible for handling missing data and normalizing features.
- **RandomForestClassifier:** The primary classification algorithm (an ensemble of decision trees).
- **GridSearchCV:** The optimization module responsible for automated hyperparameter tuning and cross-validation.
- **Metrics Modules:** Classes like f1_score, roc_auc_score and confusion_matrix used for comprehensive model evaluation.

Problem Statement

To develop a machine learning classification model that uses the UCI Cleveland Heart Disease Dataset to accurately predict the presence of cardiovascular disease (condition = 1) based on patient attributes (for example cholesterol, blood pressure, heart rate). It must be optimized enough to achieve a high F1-Score, demonstrating reliability as a medical screening tool.

Functional Requirements

1) Data Input & Loading:

- The system must read the heart_cleveland_dataset.csv file and handle FileNotFoundError properly.

2) Data Preprocessing:

- Implementation of a Pipeline to manage sequential steps along with missing value imputation and feature scaling.

3) Prediction & Classification:

- Use Random Forest Classifier to predict the ‘condition’ (binary outcome).

4) Hyperparameter Tuning:

- Use GridSearchCV with cross validation (cv = 5) to systematically find the optimal model configuration out of the multiple possible combinations.

5) Reporting & Analytics:

- Output displaying core metrics (Accuracy, F1, auc-roc) and a detailed error analysis from the confusion matrix obtained.

Non-Functional Requirements

1) Reliability:

- The model must be trained using cross-validation (cv = 5) to ensure the generalization error is stable across different subsets of the data.

2) Error Handling:

- The system must include try-except blocks for file loading and safely handle illogical zero values via imputation to maintain data integrity.

3) Maintainability:

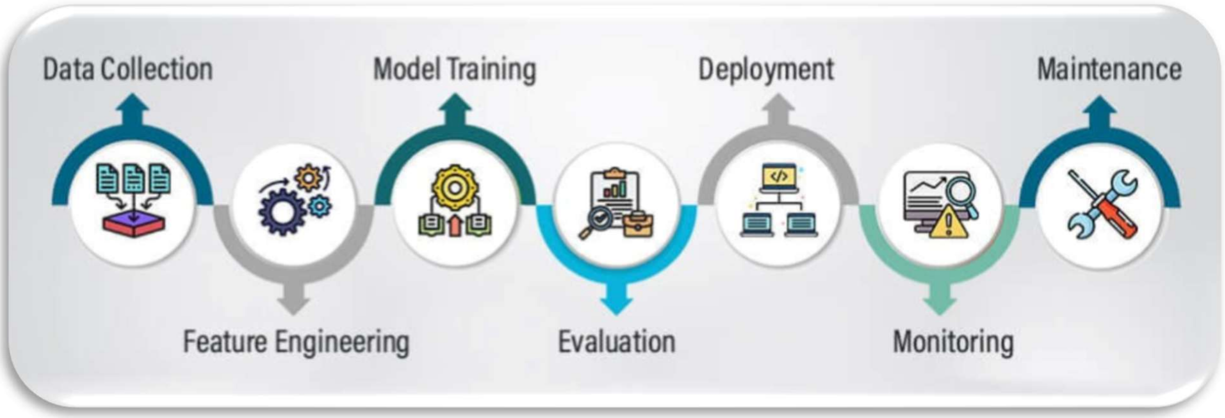
- The use of the Pipeline object ensures the preprocessing steps are modular and easily updated or replaced without altering the training workflow.

4) Performance:

- The training uses n_jobs = -1 in GridSearchCV to leverage all CPU cores, minimizing the total time required for hyperparameter optimization.

System Architecture & Design

The system follows a classic machine learning pipeline Architecture.



Key Components:

1) **Data Ingestion:** Reads the heart_cleveland_dataset.csv

2) **Preprocessing Pipeline:** Sequential data transformation

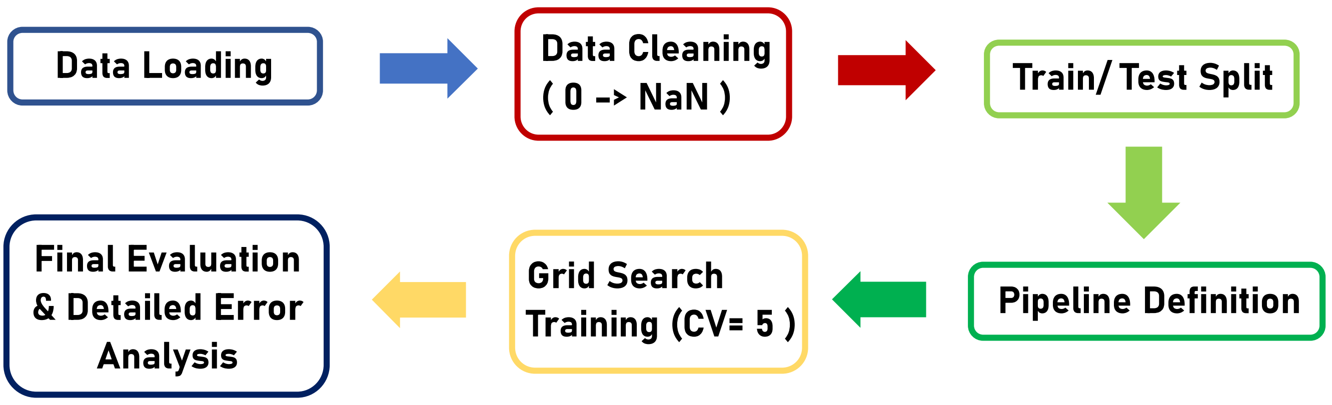
3) **Model:** Uses a Random Forest Classifier Model

4) **Tuner:** GridSearchCV manages the training iterations

5) **Reporting:** Output metrics and Error analysis

Process Flow (Workflow Diagram):

The program's logical workflow is strictly sequential:


Data Loading
Data Cleaning (0 --> NaN)
Train/ Test Split
Pipeline Definition
Grid Search Training (CV= 5)
Final Evaluation & Detailed Error Analysis

Model Selection & Rationale

1) Dataset :

- **UCI Cleveland Heart Disease**

- A balanced, publicly available numerical dataset suitable for binary classification & benchmarking.

2) Core Algorithm :

- **Random Forest Classifier**

- An Ensemble Model chosen for its high accuracy & robustness against data overfitting; effective on small/medium tabular datasets (like csv).

3) Preprocessing :

- **Pipeline w/ Imputer & Scaler**

- Standardizes the workflow and prevents data leakage by ensuring scaling factors are learned only from training data.

4) Tuning Method :

- **GridSearchCV**

- Systematically finds the best Hyperparameters, replacing manual testing with an exhaustive, cross-validated search.

5) Optimization Metric :

- **F1-Score (scoring = "f1")**

- Objective is to find the optimal balance between recall (catching sick patients) & precision (avoiding false alarms).

Implementation Details

The core implementation is contained in the python file, utilizing the scikit-learn framework for high-level Machine learning operations.

1) Data Preprocessing Flow:

The Pipeline ensures that following steps are executed:

a) Missing Value Imputation: SimpleImputer replaces illogical zero values with mean of training data

b) Feature Scaling: StandardScaler standardizes features to a mean of 0 and standard deviation of 1, preventing features with large numerical ranges from dominating the model.

c) Model Training: The preprocessed data is fed into the RandomForestClassifier Model for training it.

<u>2) Hyperparameter Training:</u>

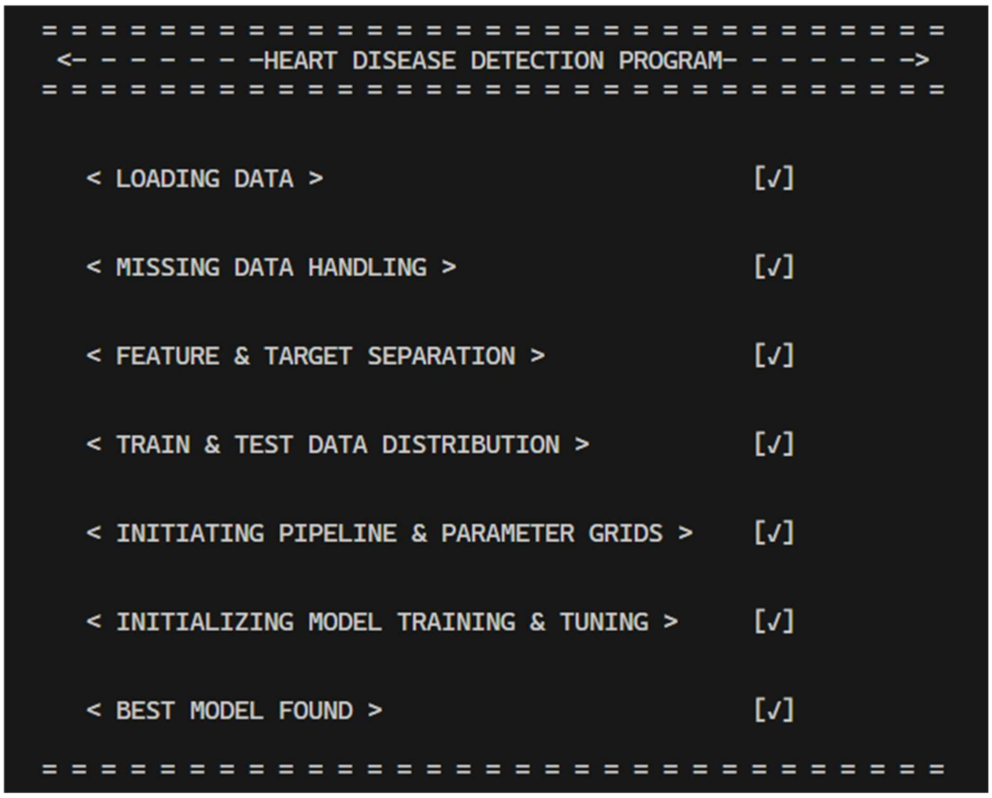
The model utilizes Grid Search with 5-Fold Cross-Validation to find optimal configuration that maximizes the f1-score.

Model Evaluation / Screenshots of Results

The model demonstrates high performance on the unseen test set, confirming the success of the F1-optimization strategy.

<u>1) Loading Screen</u>

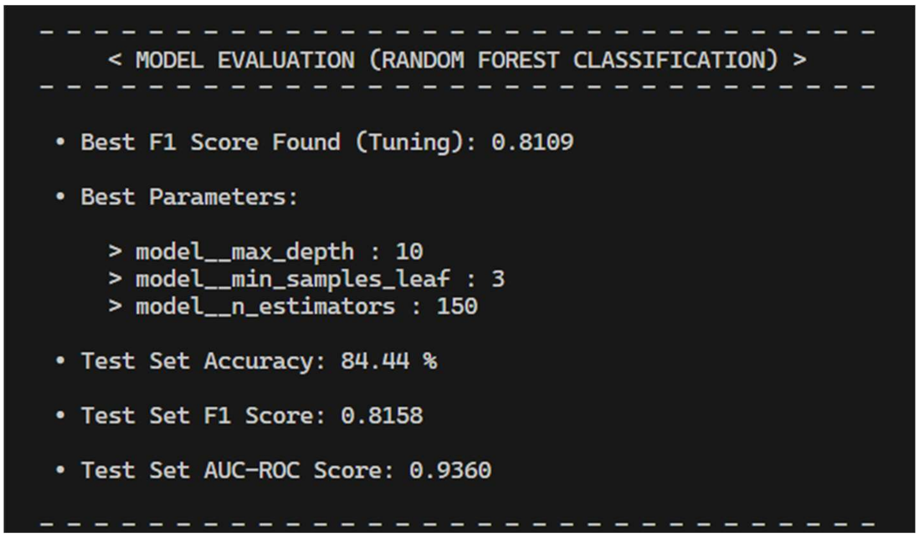
Screen displaying the processes being executed by the program



```
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
 <- - - - - - -HEART DISEASE DETECTION PROGRAM- - - - - - ->
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =


   < LOADING DATA >                                [✓]


   < MISSING DATA HANDLING >                       [✓]


   < FEATURE & TARGET SEPARATION >                 [✓]


   < TRAIN & TEST DATA DISTRIBUTION >              [✓]


   < INITIATING PIPELINE & PARAMETER GRIDS >       [✓]


   < INITIALIZING MODEL TRAINING & TUNING >        [✓]


   < BEST MODEL FOUND >                            [✓]

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

2) Core Evaluation Metrics

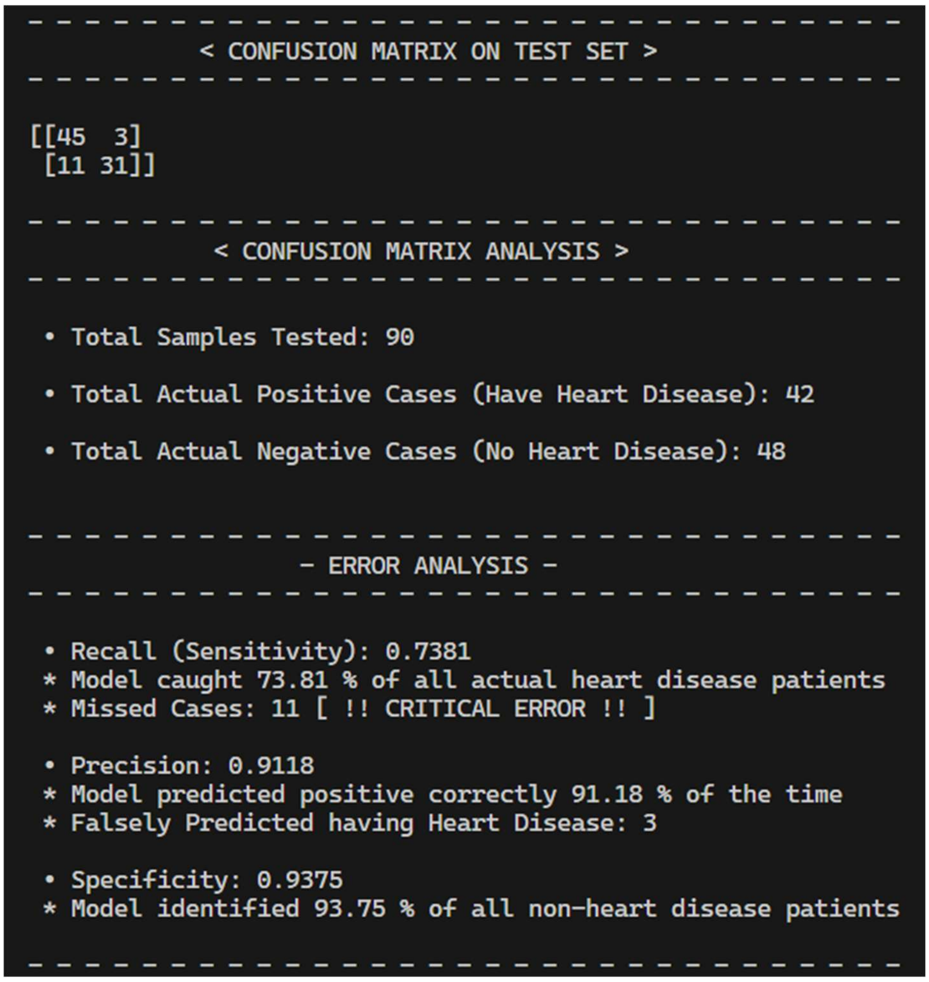
The overall generalization and reliability of the final model:



```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
     < MODEL EVALUATION (RANDOM FOREST CLASSIFICATION) >
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 • Best F1 Score Found (Tuning): 0.8109

 • Best Parameters:

     > model__max_depth : 10
     > model__min_samples_leaf : 3
     > model__n_estimators : 150

 • Test Set Accuracy: 84.44 %

 • Test Set F1 Score: 0.8158

 • Test Set AUC-ROC Score: 0.9360

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

3) Detailed Error Analysis

The model's performance breakdown on the 90 patients in the Test Set:



```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
             < CONFUSION MATRIX ON TEST SET >
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

[[45  3]
 [11 31]]

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
              < CONFUSION MATRIX ANALYSIS >
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 • Total Samples Tested: 90

 • Total Actual Positive Cases (Have Heart Disease): 42

 • Total Actual Negative Cases (No Heart Disease): 48


- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                    - ERROR ANALYSIS -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 • Recall (Sensitivity): 0.7381
 * Model caught 73.81 % of all actual heart disease patients
 * Missed Cases: 11 [ !! CRITICAL ERROR !! ]

 • Precision: 0.9118
 * Model predicted positive correctly 91.18 % of the time
 * Falsely Predicted having Heart Disease: 3

 • Specificity: 0.9375
 * Model identified 93.75 % of all non-heart disease patients

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Testing Approach

The testing approach was built directly into the program structure:

1) Validation:

The use of GridSearchCV with cv = 5 serves as the primary internal validation, ensuring the optimal model configuration is robust and not specific to one arbitrary training split.

2) External Testing:

The model's performance is finally assessed on the independent 30% Test Set (x_test, y_test), providing an unbiased estimate of its real-world generalization ability.

Challenges Faced

1) Handling Illogical Zeros:

The necessity of converting illogical zeros (e.g., Blood Pressure = 0) into NaN (Not A Number) and subsequently managing them with an Imputer required careful attention to avoid skewing the feature distribution.

2) Tuning Instability:

Early attempts at hyperparameter tuning showed high variance in the best score found, which was stabilized by refining the param_grid to focus on higher complexity/stability settings.

Learnings & Key Takeaways

1) Importance of Pipeline:

The project demonstrated the crucial role of the Pipeline in ensuring clean, non-leaking and reproducible preprocessing steps.

2) Metric Selection:

Optimizing for the F1-Score was proven effective, resulting in a balanced model with high precision and acceptable recall, which is suitable for a low false alarm screening system.

Future Enhancements

1) Cost-Sensitive Tuning: Adjust the scoring to a custom metric that weighs False Negatives (FN) significantly higher that False Positives (FP) to drive recall higher.

2) Model Stacking: Implement an ensemble of multiple different algorithms (e.g., Logistic Regression, Support Vector Machine, Random Forest) using a stacking classifier to potentially break through the current AUC-ROC ceiling of 0.9360.

3) Feature Engineering: Introduce interaction features (e.g., Age multiplied by Max Heart Rate) to provide non-linear information to the model.

References

1) Python documentation:

https://docs.python.org/3/

2) scikit-learn:

https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

3) UCI Cleveland Heart Disease Dataset:

https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data