Programming Assignment-1 (BIOENG 223A, Fall 2024)

Predicting Heart Failure Clinical Outcomes Using Machine Learning

Task Overview

The task for this assignment is to build a **classification model** that can predict whether a patient is likely to **survive** during the follow-up period based on several clinical features. This is a **binary classification problem**, where the target variable is `DEATH_EVENT`, indicating if the patient survived or passed away during the follow-up period.

Dataset Description

The dataset consists of **900 patient records**, each with **13 clinical features**. The columns in the dataset are as follows:

- **age**: The age of the patient (years) [Integer].
- **anaemia**: Indicates whether the patient has a decrease in red blood cells or hemoglobin [Boolean].
- **creatinine phosphokinase (CPK)**: The level of the CPK enzyme in the blood (mcg/L) [Integer].
- **diabetes**: Indicates whether the patient has diabetes [Boolean].
- **ejection_fraction**: Percentage of blood leaving the heart at each contraction (percentage) [Integer].
- **high_blood_pressure**: Indicates whether the patient has hypertension [Boolean].
- **platelets**: Platelet count in the blood (kiloplatelets/mL) [Continuous].
- **sex**: The sex of the patient [Binary: 1 for male, 0 for female].
- **serum_creatinine**: Level of serum creatinine in the blood (mg/dL) [Continuous].
- **serum_sodium**: Level of serum sodium in the blood (mEq/L) [Integer].
- **smoking**: Indicates whether the patient is a smoker [Boolean].
- **time**: Follow-up period (in days) [Integer].
- **DEATH_EVENT**: The target variable, which indicates whether the patient died during the follow-up period [Boolean].

Tools and Libraries

If you choose to work with Python (recommended, but not required), we suggest you use the following Python libraries:

- **Pandas**: For data manipulation and handling.
- **NumPy**: For efficient numerical computations.
- **Matplotlib**: For creating plots and visualizations.
- **Scikit-learn**: For implementing machine learning models and evaluation metrics.

Feel free to use other additional libraries or tools as needed for the task.

Assignment Steps

To help you complete this assignment, the following steps are provided as a guideline. You are encouraged to explore additional techniques and methods to improve model performance.

Step 1: Data Preprocessing

- Load the dataset using Pandas and inspect it to understand its structure.
- Handle any missing data if necessary (although the dataset may be clean).
- Perform exploratory data analysis (EDA) to gain insights into the data distribution.

Step 2: Train-Test Split

- Split the dataset into **training** and **test** sets. In machine learning, the training set is used to train the model, while the test set is kept hidden from the model (during training) and is used to evaluate its performance, post training. A typical split could be 80% for training and 20% for testing, but you may adjust this based on your analysis.
- Ensure that the target variable (`DEATH_EVENT`) is well represented in both sets.

Important: Ensure that all subsequent steps (feature engineering, model training, etc.) are performed on the train set only. Using the test set for anything other than final evaluation constitutes data leakage and must be avoided.

Step 3: Feature Engineering and Selection

- Consider whether certain features could be removed or transformed to improve model performance.
- For example, it may be beneficial to explore feature selection techniques such as **correlation analysis**, **Recursive Feature Elimination (RFE)**, or **Principal Component Analysis (PCA)** to reduce the feature space.

Step 4: Model Selection and Training

- Use traditional machine learning algorithms for this task (e.g., **Logistic Regression**, **Random Forest**, **Support Vector Machine (SVM)**, **K-Nearest Neighbors (KNN), etc**). This is not an exhaustive list, and you are free to explore other models.
- You are encouraged to experiment with different models and hyperparameters to find the best fit for the data.

Important: Avoid using deep learning models or any techniques that require GPU resources.

Step 5: Evaluation

After training the models, evaluate their performance on the **test set** using the following metrics:

- **Accuracy**: The proportion of correctly predicted instances.
- **Precision, Recall, F1-Score**: Key metrics for evaluating binary classification tasks.
- **Confusion Matrix**: Show the distribution of true positives, false positives, true negatives, and false negatives.
- **ROC-AUC Curve**: Plot the Receiver Operating Characteristic curve and calculate the AUC (Area Under the Curve).

Deliverables

The expected submission includes: **completed Jupyter notebook** that includes:

- A report document (pdf) that contains:
    - A summary of the steps you followed to solve the problem.
    - A summary of the results and key insights from your analysis. This would include the **final accuracy and other binary classification metrics** (precision, recall, F1-score) on the test set. It would also include a **confusion matrix** and an **ROC-AUC curve** for the final model.
    - Any additional conclusions or visualizations from your exploratory analysis.
- Your code files, that contain the code for all the steps you followed to solve the problem, including evaluation and visualization. If you are using Python, you can submit a Jupyter notebook.

Important Notes:

- The dataset is already cleaned and ready for analysis.
- You are encouraged to try different feature engineering techniques and models to maximize the performance of your classifier.
- Ensure that your report and code base is well-documented and easy to follow.