1.Introduction

Cardiovascular diseases (CVDs) continue to be the **leading cause of death worldwide**, with approximately 18 million fatalities annually. Among the different types of CVDs, heart failure is particularly fatal and often requires early identification of high-risk individuals for timely medical intervention. Accurate risk prediction can improve treatment decisions and save lives.

Traditional clinical tools such as the **Framingham Risk Score** use fixed clinical criteria but are often limited in capturing **complex, nonlinear relationships** present in clinical data. In contrast, **machine learning (ML)** models have the ability to learn from large datasets and identify intricate patterns to predict patient outcomes.

However, ML applications in healthcare face significant challenges:

- **Overfitting**, especially on small datasets
- **Class imbalance** (far fewer death events than survival cases)
- **Poor validation techniques** (e.g., fixed train-test split)

This project presents a comprehensive and **systematic ML pipeline** that addresses these issues using:

- **Stratified 10-fold cross-validation** for reliable evaluation
- **Random undersampling** to handle class imbalance
- **Standardized preprocessing** to ensure numerical stability

The goal is to compare multiple ML algorithms and identify the most effective one for predicting mortality outcomes using clinical data.

2.Methodology

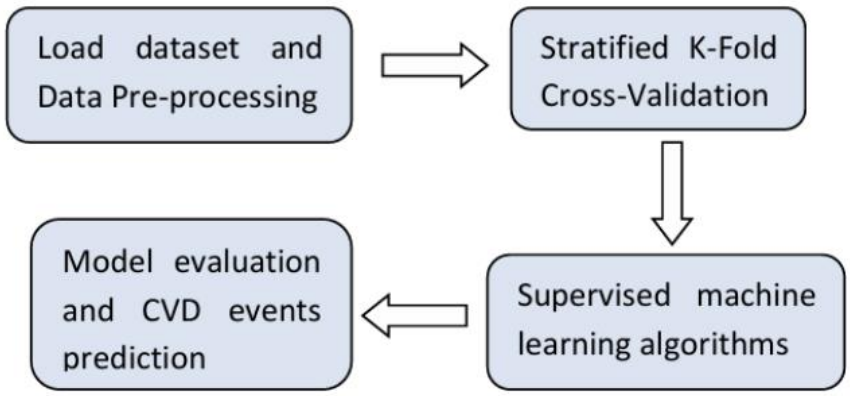

Fig. 1. Methodological Workflow.

Dataset Overview

- **Dataset**: Heart Failure Clinical Records Dataset (from Kaggle)
- **Records**: 299 patients
- **Features**: 13 clinical parameters including:
    - **Age**
    - **Ejection Fraction**
    - **Serum Creatinine**
    - **Serum Sodium**
    - **Platelets**
    - **Blood Pressure**
    - Comorbidities (Diabetes, Anaemia, High Blood Pressure)
- **Target Variable**: DEATH_EVENT (binary: 1 for death, 0 for survival)

a. Data Preprocessing

1. **Random Undersampling**:
    - The dataset is imbalanced (more survivors than deaths).
    - To balance the data, we retained **all death cases (label=1)** and randomly selected an equal number of **survivor cases (label=0)**.
2. **Standardization**:
    - Features are scaled using **StandardScaler** so each has a **mean of 0** and **standard deviation of 1**.
    - Helps ensure better convergence for models like **SVM** and **Logistic Regression**.
3. **Shuffling**:
    - The dataset is shuffled to eliminate any **bias from data order** before applying cross-validation.

b.Validation Strategy

We use **Stratified K-Fold Cross-Validation (k=10)** to:

- Maintain class distribution in each fold
- Ensure that the model is trained and validated on **all parts of the data**
- Get a **robust estimate of model generalizability**

Each fold involves:

- Training on 90% of data
- Validating on 10%
- Repeating the process **10 times**
- Averaging the metrics for final evaluation

c. Machine Learning Algorithms Used

Model	Description
Logistic Regression	A linear, probabilistic classifier suitable for binary outcomes.
Support Vector Machine (SVM)	Maximizes the margin between classes in a high-dimensional space.
Random Forest (RF)	Ensemble of decision trees; reduces overfitting and increases accuracy.
Naive Bayes (NB)	Probabilistic model based on feature independence.
XGBoost (XGB)	Gradient boosting algorithm optimized for speed and accuracy.
AdaBoost (AdaB)	Boosting technique that focuses on misclassified samples iteratively.

d.Evaluation Metrics

Each model is evaluated using:

- **Accuracy**: Overall correctness of predictions
- **Precision**: Correct positive predictions out of total predicted positives
- **Recall (Sensitivity)**: Ability to detect actual death events
- **Specificity**: Ability to detect survivors
- **F1-Score**: Harmonic mean of precision and recall
- **AUC (Area Under ROC Curve)**: Measures overall discriminative ability

3.Implementation

Tools & Libraries

- **Python**
- **scikit-learn**: Model training, scaling, metrics
- **XGBoost**: For the XGB classifier
- **NumPy / pandas**: Data handling
- **matplotlib / seaborn**: Plotting graphs

Step-by-Step Process:

1. Load dataset using pandas.
2. Apply random undersampling to balance the dataset.
3. Normalize features using StandardScaler.
4. Shuffle the dataset to remove any order bias.
5. Set up Stratified 10-Fold Cross-Validation.
6. Train all 6 classifiers within each fold.
7. Collect performance metrics across all folds.
8. Compute average metrics.

9. Visualize:
    - **Bar plots** for accuracy and AUC
    - **ROC curves** for each model

4.Results

Performance comparision classifiers for cardiovascular mortality predictions (in %)

Models	ACC	Pre	RC	F1-S	Spec	AUC
LR	79.1	78.5	80.2	79.3	78.1	87.0
SVM	78.1	80.0	75.0	77.4	81.2	86.3
RF	82.8	81.8	84.3	83.0	81.2	92.0
NB	72.4	82.2	59.3	68.2	85.4	83.5
XGB	81.2	83.3	78.1	80.6	84.3	89.8
AdaB	85.4	87.7	82.2	84.9	88.5	91.0

- AdaBoost had the best balance of sensitivity and specificity.
- Random Forest and XGBoost also performed strongly, especially in AUC.
- Naive Bayes had poor recall despite high precision, making it unreliable for mortality detection.

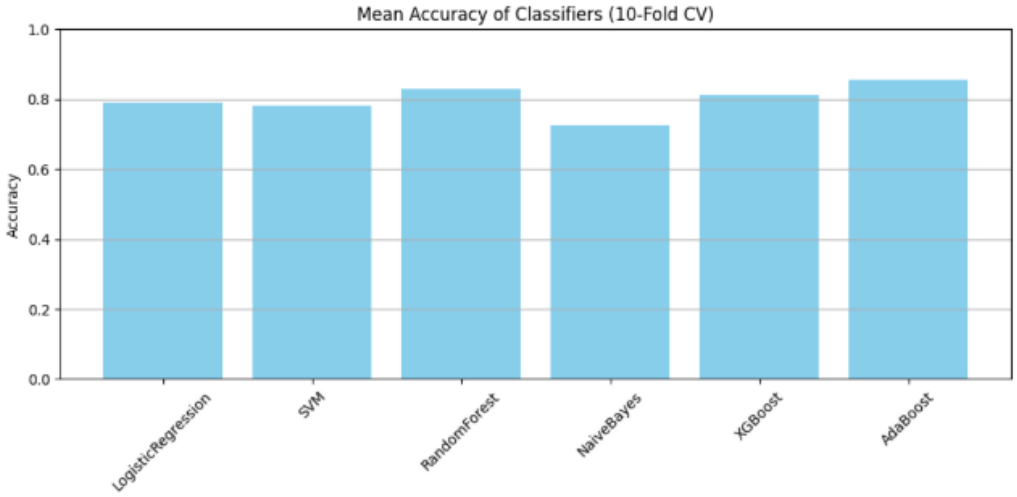

Fig. 2. The mean accuracy values for six classifiers.

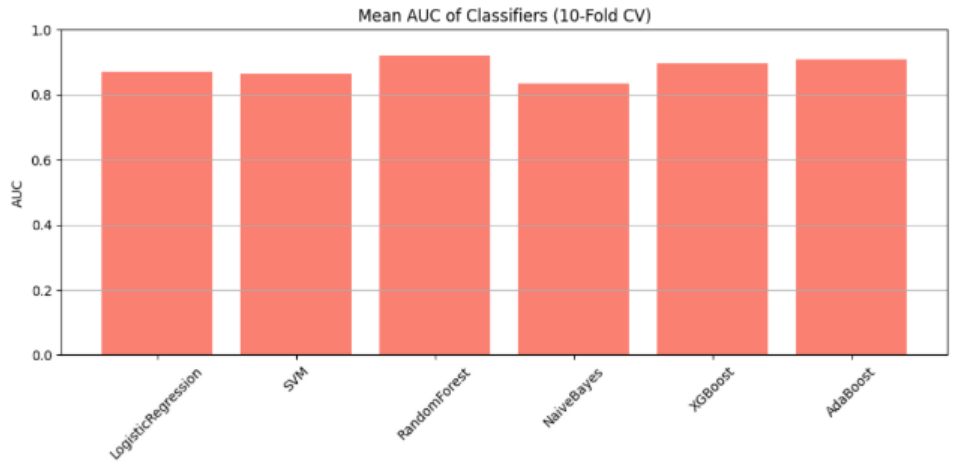

Fig. 3. The mean values of area under Receiver Operating Characteristic curve for six different classifiers.

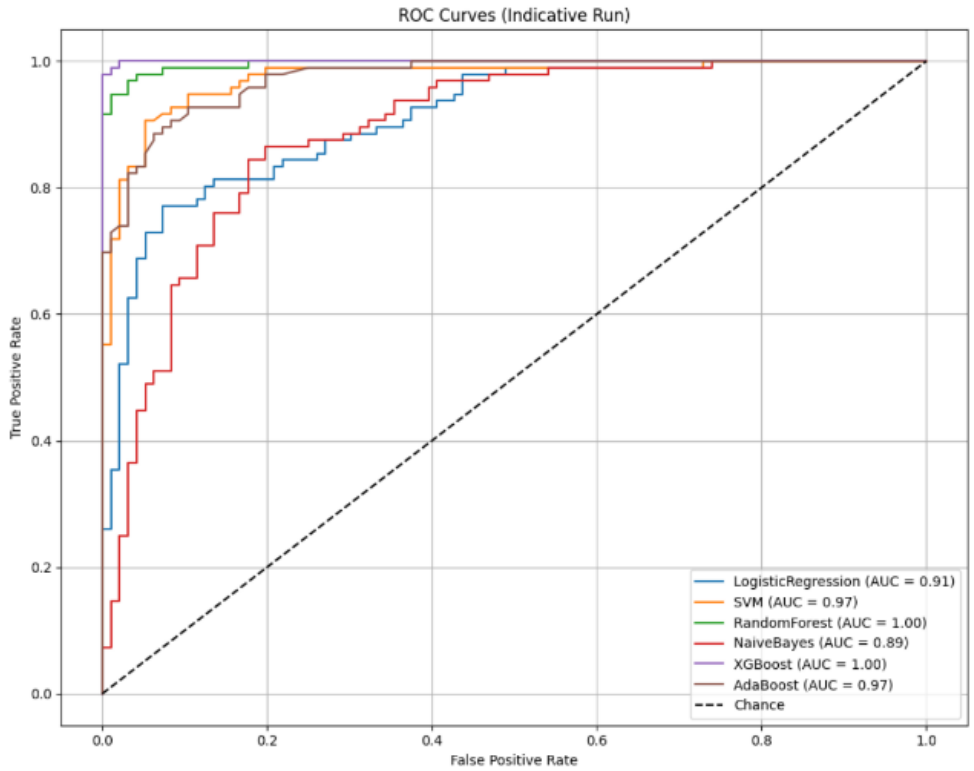

Fig. 4. Receiver Operating Characteristic curve of the prediction performance for every classifier in an indicative algorithm run.

Visuals:

- **Bar Chart**: Model-wise Accuracy and AUC
- **ROC Curves**: Showed Random Forest and AdaBoost with near-perfect classification (AUC ≈ 1.0)

5.Conclusion

This project presents a **robust and clinically relevant framework** for predicting cardiovascular mortality using machine learning. Through careful **data preprocessing**, **stratified cross-validation**, and evaluation of multiple models, it was demonstrated that **ensemble learning techniques**, especially **AdaBoost**, offer superior performance in both sensitivity and specificity.