Prediction of Mortality Rate of Heart Failure Patients Admitted to ICU

Name:	**Aryan Jain**
Registration No./Roll No.:	21055
Institute/University Name:	IISER Bhopal
Program/Stream:	e.g., DSE
Problem Release date:	August 17, 2023
Date of Submission:	November 19, 2023

1 Introduction

This project addresses a two-class classification problem, aiming to predict the mortality rate of heart failure patients admitted to ICUs. Leveraging data from the MIMIC-III database, encompassing 46,520 patients and 58,976 ICU admissions, the dataset has 48 features and 1058 datapoints. Given, dataset contains 1723 missing values and an imbalanced class distribution, comprising 143 instances of class 0 and 915 instances of class .

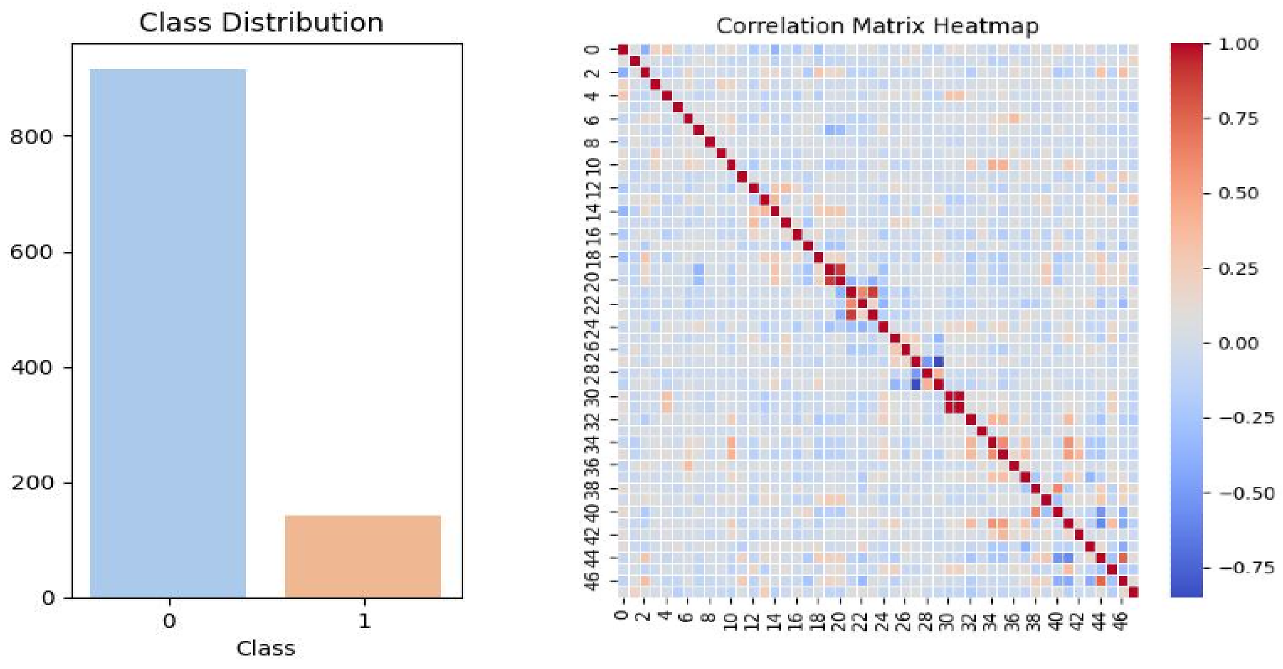

Figure 1: Overview of Data Set

2 Methods

2.1 Data Preprocessing

- **Imputation**
  Imputation methods such as mean, median, mode, and KNN imputation were utilized to replace missing values in the dataset. The mean imputation involved replacing missing values with the mean of the respective features, while median and mode imputation utilized the median and mode, respectively. Additionally, a KNN imputer was implemented to fill missing values based on the values of their k-nearest neighbors.

- **Scaling**
  To ensure the uniformity of feature scales, a standard scaling technique was applied. This method standardizes the features to have a mean of 0 and a standard deviation of 1 using the StandardScaler from the scikit-learn library.

- **One Hot Encoding**
  Categorical features were converted into binary vectors through one-hot encoding. This transformation simplifies the representation of categorical data, making it compatible with different classifier.

- **Feature Selection**
  Three distinct methods were explored: Mutual Information, Recursive Feature Elimination (RFE), and Correlation Matrix analysis. Mutual information was employed to select features based on their relevance to the target variable. RFE, with a Random Forest Classifier, recursively eliminated the least important features. Moreover, a correlation matrix analysis was conducted to identify and remove features highly correlated with others, thereby reducing redundancy.

2.2 Training Classifiers

- **OverSampling**
  After splitting the training data into testing and training set, The Adaptive Synthetic Sampling (ADASYN) technique was applied for oversampling the minority class in the training set. ADASYN generates synthetic samples to balance the class distribution, thereby mitigating the impact of class imbalance on model performance.

- **Classifier training and Hyperparameter Tuning**
  Various Classification Algorithm such as Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), AdaBoost (AB), Logistic Regression (LR), and Linear Support Classifier (LSV) has been used. To refine the performance of these classifiers, hyperparameter tuning was conducted using GridSearchCV. This technique systematically explored a predefined hyperparameter grid for each algorithm, identifying the optimal hyperparameter values that maximize the chosen evaluation metric, the F1-score. This meticulous approach ensures that each classifier is fine-tuned to extract the best possible performance from the dataset.

3 Experimental Setup

Since accuracy isn't appropriate for binary classification with imbalanced classes, I evaluated the model using conventional metrics like F1-Score, Precision, Recall, and Confusion Matrix. Because of its ability to address imbalances by taking into account overall class performance, the F1-Score particularly the macro-averaged variant was given priority. For feature selection, I have used various methods such as Recursive Feature Selection along with Random Forest Classifier, Mutual Information and Correlation Matrix analysis. For handling missing values, I have used Mean, Median, Mode and KNN imputation, after analyzing each combination of feature selection method and imputation method, Correlation matrix analysis in combination KNN imputation is giving the best results for all the classifier except for Adapting boosting , it is giving best results on Mutual Information feature selector along with KNN imputation.The GridSearchCV package is used to optimise each classifier based on the maximum F1-Score through hyperparameter tuning of the classifiers.Since scaling would have rendered the feature continuous, which is not practically conceivable in the case of age, the "Age" feature was removed. There are 10 categorical features on which One Hot Encoding is applied and there was no imputation required for them.

4 Results and Discussion

After trying various classifiers along with imputation method and feature selection methods, I have achieved the best F1 score of 0.66 on Adaptive Boosting Algorithm along with kNN Imputation and Mutual Information Feature selection method. The table below shows the best performance of various other classifiers after hyperparametric tuning.

Table 1: Performance Of Different Classifiers Using All Features

Classifier	Precision	Recall	Feature Selection	F-measure
Adaptive Boosting	0.75	0.62	Mutual Information	0.66
Decision Tree	0.65	0.68	Correlation	0.66
Logistic Regression	0.63	0.73	Correlation	0.65
Random Forest	0.69	0.60	Correlation	0.62
Support Vector Machine	0.43	0.50	Correlation	0.46

Table 2: Confusion Matrices of Different Classifiers

Actual	Predicted Class	
Class	1	0
1	178	5
0	21	8

Adaptive Boosting

Actual	Predicted Class	
Class	1	0
1	160	23
0	15	14

Decision Tree

Actual	Predicted Class	
Class	1	0
1	142	41
0	9	20

Logistic Regression

Actual	Predicted Class	
Class	1	0
1	176	7
0	22	7

Random Forest

Actual	Predicted Class	
Class	1	0
1	183	0
0	29	0

SVM

5 Conclusion

In summary, after analyzing different ways to predict outcomes, the AdaBoost model turned out to be the best for our dataset. It showed high F1-Score in making predictions. Looking ahead, future work could focus on understanding how the AdaBoost model makes decisions and exploring ways to improve predictions further. We might also want to investigate how different ways of filling in missing data affect the model. Additionally, testing our approach on larger datasets and in different situations could be interesting.

References

[1] Github

[2] Feature Selection Techniques

[3] KNN imputation

[4] Missing values Imputation techniques

[5] Model Evaluation