Advancing ICU Resource Planning

Machine Learning Analysis and Prediction of Mortality & Pre-Admission Length of Stay

Author: Nithin Alluru

Contents

List of Figures

List of Tables

Abstract

The integration of machine learning techniques in healthcare analytics offers the potential for improved predictive modeling and actionable insights. This project explores a comprehensive analysis of a real-world ICU dataset to predict critical outcomes, including *hospital death* (binary classification) and *pre ICU length of stay (LOS)* (regression). The dataset, consisting of 91,713 observations and 85 features, was preprocessed to address missing values, multicollinearity, and class imbalance. Feature engineering techniques, including dimensionality reduction using Principal Component Analysis (PCA) and Random Forest analysis, enhanced the dataset's quality and predictive power.

The classification phase evaluated multiple machine learning algorithms, including Logistic Regression, Decision Trees, Support Vector Machines, and ensemble methods. The Post-Pruned Decision Tree emerged as the best classifier, achieving a test accuracy of 90%, a specificity of 95%, and a robust ROC-AUC score of 84%. Association rule mining using the Apriori algorithm identified significant relationships, such as high *apache 4a ICU death prob* being strongly associated with hospital mortality. Clustering analysis with K-means and DBSCAN revealed two distinct clusters, providing insights into patient subgroups.

Regression analysis for *pre ICU LOS* using multiple linear regression explained approximately 14% of the variance, highlighting the need for non-linear models to better capture complex patterns. The project recommends exploring advanced regression techniques, hybrid classification models, and improved parameter optimization strategies to enhance predictive performance. These findings emphasize the importance of feature selection, class imbalance handling, and explainable AI in deploying machine learning solutions in healthcare. Future work will focus on extending the dataset, integrating advanced architectures, and refining model interpretability to support clinical decision-making and operational efficiency.

1 Introduction

The field of healthcare analytics has seen significant advancements with the integration of machine learning techniques, which allows more precise predictions and decision-making processes. This report outlines the steps and methodologies employed in analyzing a patient data set in the ICU to predict critical outcomes and extract actionable insights (1). The objectives of this project align with the broader goals of healthcare innovation, focusing on both descriptive and predictive modeling.

The analysis is divided into four key phases: Feature Engineering and Exploratory Data Analysis (EDA), Regression Analysis, Classification Analysis, and Clustering with Association Rule Mining. Each phase is designed to address specific questions about the dataset, refine its features, and develop robust models for prediction and inference.

- **Phase I: Feature Engineering and EDA** The initial phase focuses on pre-processing the data set to handle missing values, detect and address multicollinearity, and enhance the predictive power of the data set. Techniques such as Principle Compnent Analysis, Singular Value Decomposition, Random Forest Anaysis (RFA), High-Correlation Filter, and encoding methods are applied to ensure that the data are clean and optimized for analysis.

- **Phase II: Regression Analysis** This phase aimed to predict the continuous numerical feature `pre_icu_los_days` using multiple linear regression. A systematic approach was used to best develop a linear regression model, ensuring that the model captures the maximum possible relationships.
- **Phase III: Classification Analysis** The primary task of this phase is to predict the binary outcome of `hospital_death`, leveraging multiple machine learning classifiers, including logistic regression, decision trees and neural networks. Advanced techniques such as grid search, cross-validation, and performance metrics (e.g. ROC-AUC, precision, recall) are utilized to identify the best-performing model.
- **Phase IV: Clustering and Association Rule Mining** The final phase involves exploratory research into unsupervised learning techniques, including K-means and DBSCAN for clustering, and the Apriori algorithm for association rule mining. These methods aim to uncover hidden patterns and relationships within the data set.

2 Description of the Dataset

The selected data set is a comprehensive real-world dataset from the healthcare industry, which contains **91,713 observations** and **85 features**. This data set satisfies the project's criteria, as it is a multivariate dataset with a mix of **numerical** and **categorical variables**, suitable for descriptive and predictive analyzes. Furthermore, the dataset is publicly available, ensuring compliance with non-classified data usage requirements.

2.1 Target and Independent Variables

- **Phase II: Regression Analysis Target Variable**: `pre_icu_los_days` This variable represents the length of stay in the ICU prior to admission. It is a continuous numerical variable, making it ideal for regression modeling. Predicting this variable provides actionable insights into pre-admission patient care efficiency and resource planning.
- **Phase III: Classification Analysis Target Variable**: `hospital_death` This binary categorical variable indicates whether the patient outcome was fatal (1) or non-fatal (0). Classifying this variable helps identify critical factors contributing to mortality, enabling proactive healthcare interventions.
- **Independent Variables**: The dataset includes a variety of numerical, categorical, and binary features that serve as potential predictors for the target variables. Key independent variables include:
  - **Numerical Variables**: `age`, `bmi`, `height`, `pre_icu_los_days`, `weight`, `apache_2_diagnosis`, `apache_3j_diagnosis`, `heart_rate_apache`, `map_apache`, `resprate_apache`, `temp_apache`,

apache_4a_icu_death_prob, apache_4a_hospital_death_prob, d1_diasbp_max, d1_diasbp_min
d1_diasbp_noninvasive_max, d1_diasbp_noninvasive_min, d1_heartrate_max,
d1_heartrate_min, d1_mbp_max, d1_mbp_min, d1_mbp_noninvasive_max, d1_mbp_noninvasive_m
d1_resprate_max, d1_resprate_min, d1_spo2_max, d1_spo2_min, d1_sysbp_max,
d1_sysbp_min, d1_sysbp_noninvasive_max, d1_sysbp_noninvasive_min, d1_temp_max,
d1_temp_min, d1_glucose_max, d1_glucose_min, d1_potassium_max, d1_potassium_min.

– **Categorical Variables**: ethnicity, icu_admit_source, icu_type, icu_stay_type, apache_3j_bodysystem, apache_2_bodysystem, gcs_verbal_apache, gcs_motor_apache, gcs_eyes_apache.

– **Binary Variables**: elective_surgery, apache_post_operative, arf_apache, gcs_unable_apache, intubated_apache, ventilated_apache, aids, cirrhosis, diabetes_mellitus, hepatic_failure, immunosuppression, leukemia, lymphoma, solid_tumor_with_metastasis, hospital_death.

2.2 Industry Relevance

This dataset holds significant importance in the healthcare industry, as it enables the development of predictive models that can improve patient care and operational efficiency. By predicting critical outcomes like mortality (hospital_death) or pre-admission ICU duration (pre_icu_los_days), healthcare providers can allocate resources more effectively and plan for patient care needs. Additionally, the dataset's diverse features, ranging from demographic and clinical information to ICU metrics, offer valuable insights into risk factors and patterns, supporting both clinical decision-making and medical research. The ability to apply machine learning techniques to such a rich and real-world dataset aligns with the growing emphasis on data-driven healthcare innovation.

3 Phase I: Feature Engineering and Exploratory Data Analysis

In this phase, we focus on preparing the dataset for analysis by performing data preprocessing, handling missing values, encoding categorical variables, and standardizing numerical variables. The goal is to clean and transform the data to ensure its suitability for modeling, and to extract meaningful features for the subsequent analysis.

3.1 Data Preprocessing

3.1.1 Handle Missing Values

The dataset contains missing values that need to be addressed to ensure the robustness of the models. Missing values were handled using different strategies based on the variable type:

- **Numerical Variables**: For numerical features, missing values were imputed with the median value of the respective column. This approach is robust against outliers and ensures that the imputation does not introduce significant bias. Numerical columns such as `age`, `bmi`, `height`, `pre_icu_los_days`, and others were processed in this way.

- **Categorical Variables**: For categorical variables, missing values were imputed using the mode (most frequent value). Variables such as `gender`, `ethnicity`, and `icu_admit_source` were handled in this manner.
- **Binary Variables**: Clinical binary features such as `elective_surgery`, `intubated_apache`, and `hospital_death` were imputed with a value of 0 (assumed to indicate absence).

The total number of missing values was carefully tracked before and after imputation. Initially, the dataset contained **102,381 missing values**, spread across various features. After applying appropriate imputation techniques tailored to the data type, the dataset is now fully complete, with **no missing values remaining**. This ensures a robust foundation for subsequent analyses.

3.1.2 Check for Duplicates

Duplicate rows in the dataset have the potential to distort analysis and negatively impact model performance. Upon examination, the dataset was found to contain **no duplicate rows**, ensuring the integrity of the data for subsequent analyses.

3.1.3 Encoding Categorical Variables

To prepare categorical variables for machine learning models, *One-Hot Encoding* was applied to nominal categorical variables. This transformation converts categorical values into binary columns, which can be effectively processed by most machine learning algorithms. The categorical variables encoded include `gender`, `icu_admit_source`, `icu_type`, and others. After encoding, the first column in each feature set was dropped to avoid the dummy variable trap.

- **Nominal Variables**: These variables include `gender`, `icu_admit_source`, `icu_type`, `icu_stay_type`, `apache_3j_bodysystem`, `apache_2_bodysystem`, and `ethnicity`.

The data was successfully transformed, and the first five rows of the encoded data were verified for correctness.

3.1.4 Feature Aggregation

Certain columns, such as `d1_diasbp_max` and `d1_diasbp_min`, contained both maximum and minimum values for specific clinical metrics. These were consolidated into new features representing the range between the maximum and minimum values. For instance, `d1_diasbp_range` was created to capture the range of diastolic blood pressure (`d1_diasbp_range` = `d1_diasbp_max` − `d1_diasbp_min`). Once the range features were computed, the original max/min columns were dropped.

This aggregation not only enhances the dataset by consolidating redundant information but also serves to reduce the total number of features, simplifying the dataset and improving computational efficiency for subsequent analysis. Similar steps were applied to other features with maximum and minimum values, ensuring a cleaner and more informative dataset.

3.1.5 Normalization and Standardization

Since numerical variables in the dataset can have vastly different scales, **Standardization** was applied to all numerical features using the `StandardScaler` from the `sklearn` library. This process transforms the data to have a mean of 0 and a standard deviation of 1, ensuring that all features contribute equally to the model.

3.1.6 Dimensionality Reduction and Feature Selection

Dimensionality reduction and feature selection were applied to improve the interpretability and performance of the models. Employed a variety of techniques to address issues such as feature redundancy and collinearity, as outlined below. The results for both targets (`pre_icu_los_days` for Phase 2 and `hospital_death` for Phase 3) were analyzed and presented through graphical depictions.

- **Principal Component Analysis (PCA)**: PCA was applied to transform the dataset into a set of orthogonal components that capture the maximum variance in the data. The number of principal components was determined based on the cumulative explained variance reaching 95%. A graph of the cumulative explained variance is presented, showing the proportion of variance retained by each principal component.

  Observations for `pre_icu_los_days`: The PCA revealed that only 29 components captured the majority of the variance in the dataset, demonstrating that dimensionality could be effectively reduced without losing significant information. This also indicates that there's an underlying collinearity in the data matrix that needs to be dealt with. (Figure 1) illustrates the cumulative explained variance along the number of components for `pre_icu_los_days`.

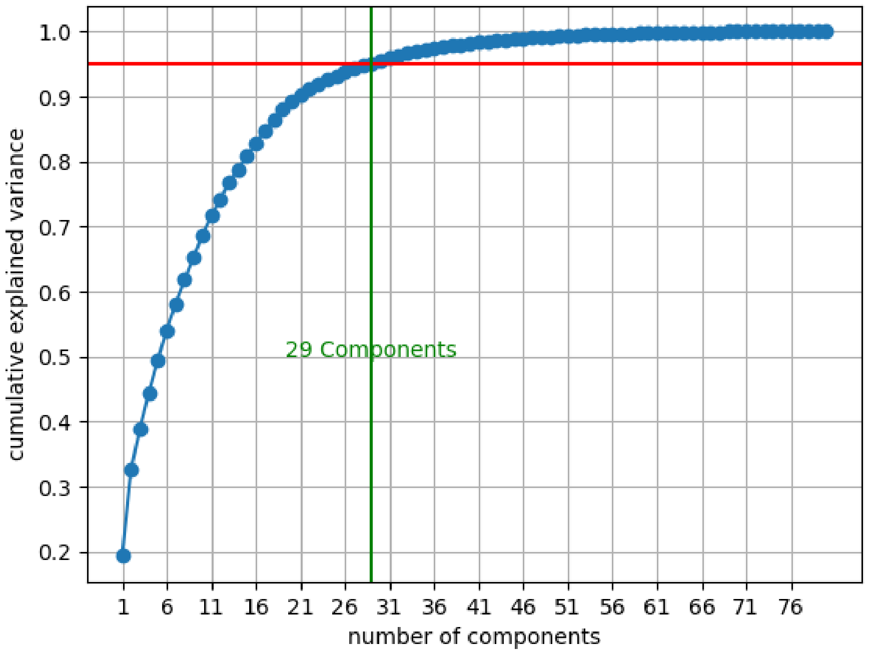

Figure 1: Cumulative Explained Variance from PCA (`pre_icu_los_days`)

Observations for `hospital_death`: Similarly, for predicting hospital death, PCA indicated 29 components explained 95% of the variance. This again indicates that there's an underlying collinearity in the data matrix. (Figure 2) shows the cumulative explained variance for `hospital_death`.

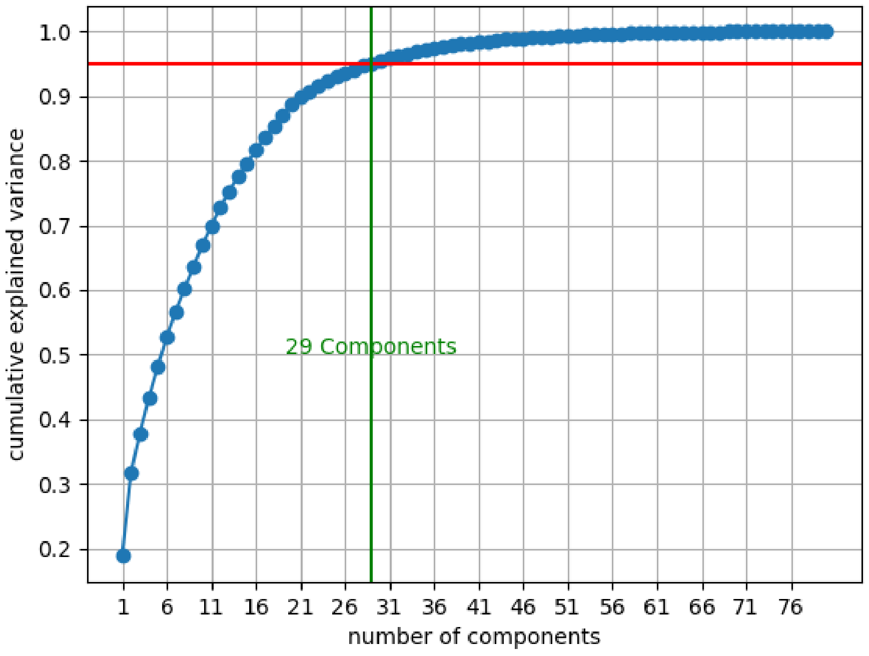

Figure 2: Cumulative Explained Variance from PCA (`hospital_death`)

- **Singular Value Decomposition (SVD)**: SVD was used to decompose the dataset into its constituent components, focusing on the most significant singular values. This technique complements PCA by providing an alternative decomposition approach. The explained variance by the SVD components is visualized in the graphs, highlighting how much variance is retained by the most significant components.

  Observations for `pre_icu_los_days`: SVD provided similar results to PCA, showing that a small number of components were sufficient to capture most of the variance in the dataset. The explained variance ratio for the first few components was high, indicating that dimensionality reduction could be applied effectively for this target. (Figure 3) illustrates the explained variance by SVD components for `pre_icu_los_days`.

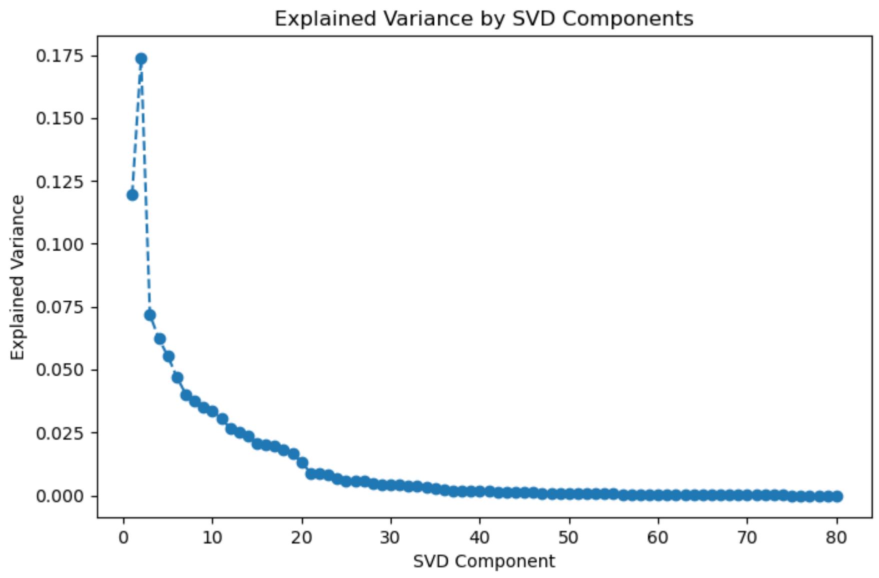

Figure 3: Explained Variance by SVD Components (`pre_icu_los_days`)

Observations for `hospital_death`: The SVD analysis for predicting hospital death also showed that the top components explained most of the variance, with the first few components capturing a significant portion of the information. (Figure 4) shows the explained variance by SVD components for `hospital_death`.

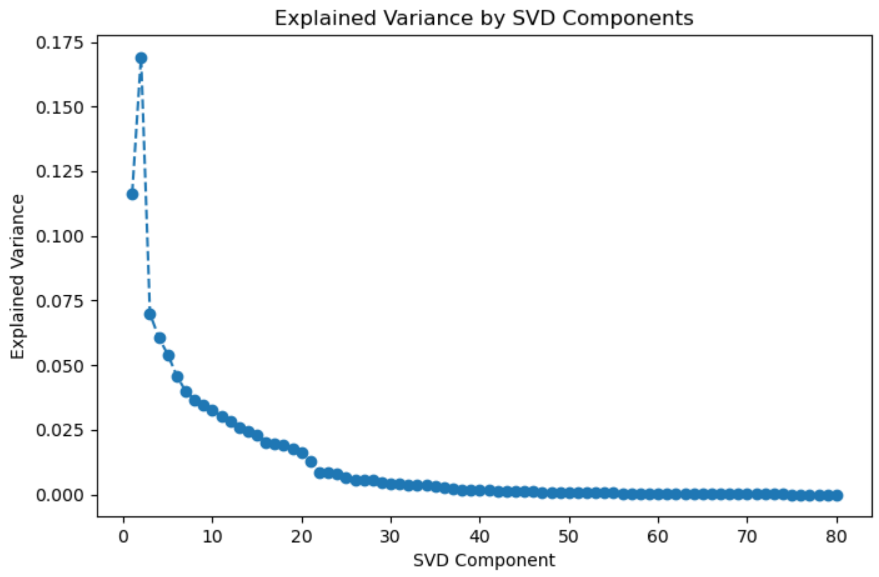

Figure 4: Explained Variance by SVD Components (`hospital_death`)

- **Random Forest Analysis**: A Random Forest model was used to estimate the importance of each feature in predicting the target variable. Features with low importance scores were considered redundant and removed from the dataset. This method helps focus on the most informative features for modeling. The resulting feature importance is shown for both targets, indicating which variables contribute most to prediction.

Observations for `pre_icu_los_days`: The Random Forest model identified key features contributing to the prediction of ICU length of stay before admission. Features like `icu_admit_source`, `apache_4a_hospital_death_prob`, and `apache_4a_icu_death_prob` were found to be the most important predictors, while other features had minimal impact. (Figure 5) shows the feature importance for `pre_icu_los_days`.

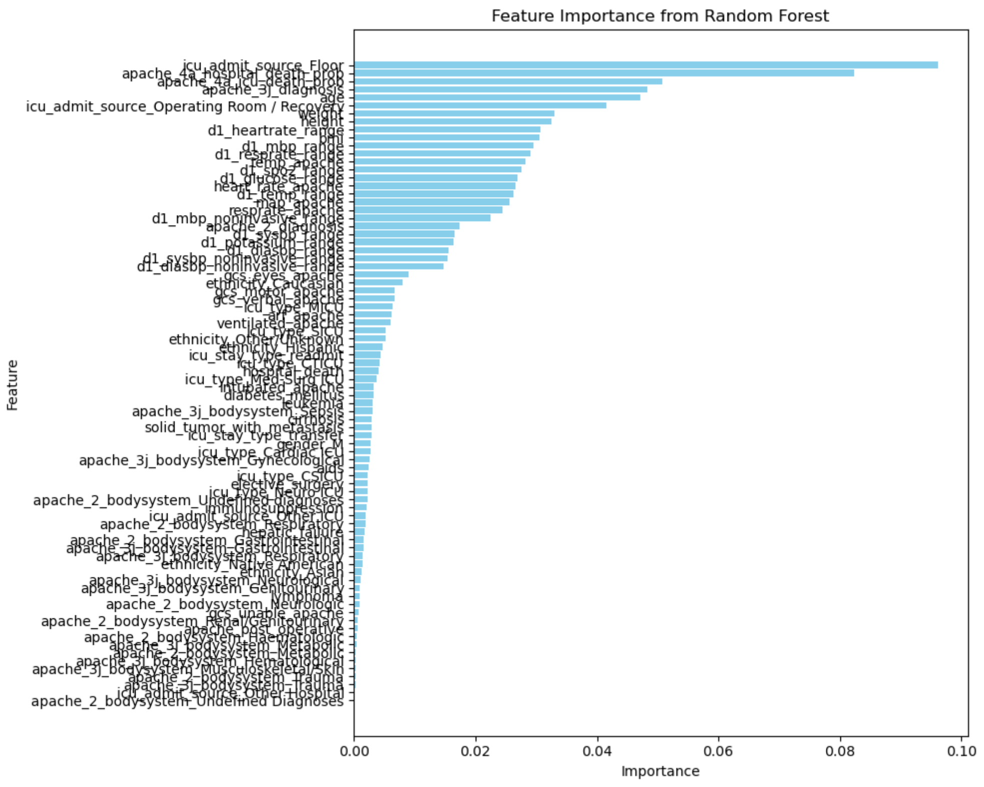

Figure 5: Random Forest Feature Importance (`pre_icu_los_days`)

Observations for `hospital_death`: For predicting hospital death, the Random Forest model highlighted the importance of features such as `apache_4a_hospital_death_prob`, `apache_4a_icu_death_prob`, and `d1_spo2_range`. These variables had the strongest impact on mortality prediction. (Figure 6) depicts the feature importance for `hospital_death`.

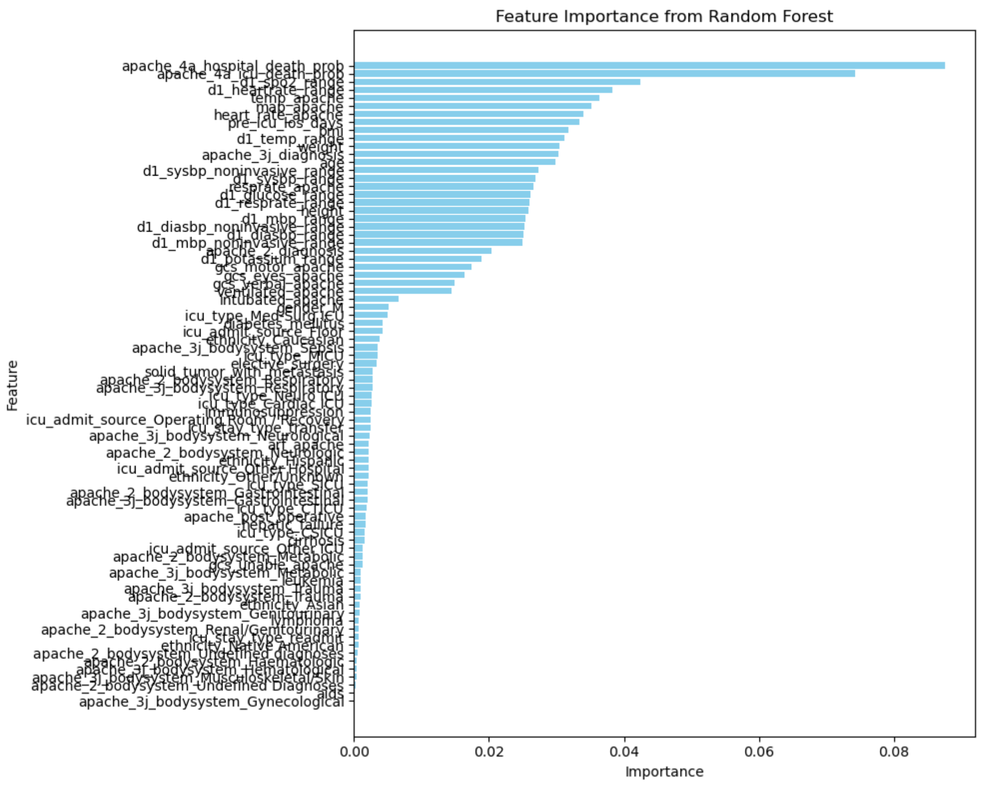

Figure 6: Random Forest Feature Importance (`hospital_death`)

- **High Correlation Filter**: High correlation filtering was applied to eliminate features that were highly correlated with others, to avoid redundancy and multicollinearity. The threshold for high correlation was set at 0.8, meaning that any pair of features with a correlation greater than 0.8 was removed.

  Observations for `pre_icu_los_days`: For predicting `pre_icu_los_days`, several features were identified as highly correlated and subsequently removed to simplify the dataset. The dropped features included: `apache_3j_bodysystem_Gastrointestinal`, `apache_3j_bodysystem_Respiratory`, `apache_3j_bodysystem_Metabolic`, `apache_3j_bodysystem_H`
  `apache_3j_bodysystem_Trauma`, `apache_3j_bodysystem_Genitourinary`, `apache_3j_bodysystem_Ne`
  `apache_3j_diagnosis`, `bmi`, `elective_surgery`, `apache_post_operative`, `d1_diasbp_range`, `d1_mbp_range`, `d1_sysbp_range`, `d1_diasbp_noninvasive_range`, `gcs_eyes_apache`, `apache_4a_hosp`

  Removing these features simplified the dataset, reduced multicollinearity and dimentions. Plots depicting correlation matrix heatmaps before and after applying the high correlation filter for `pre_icu_los_days` are shown in (Figure 7) and (Figure 8).

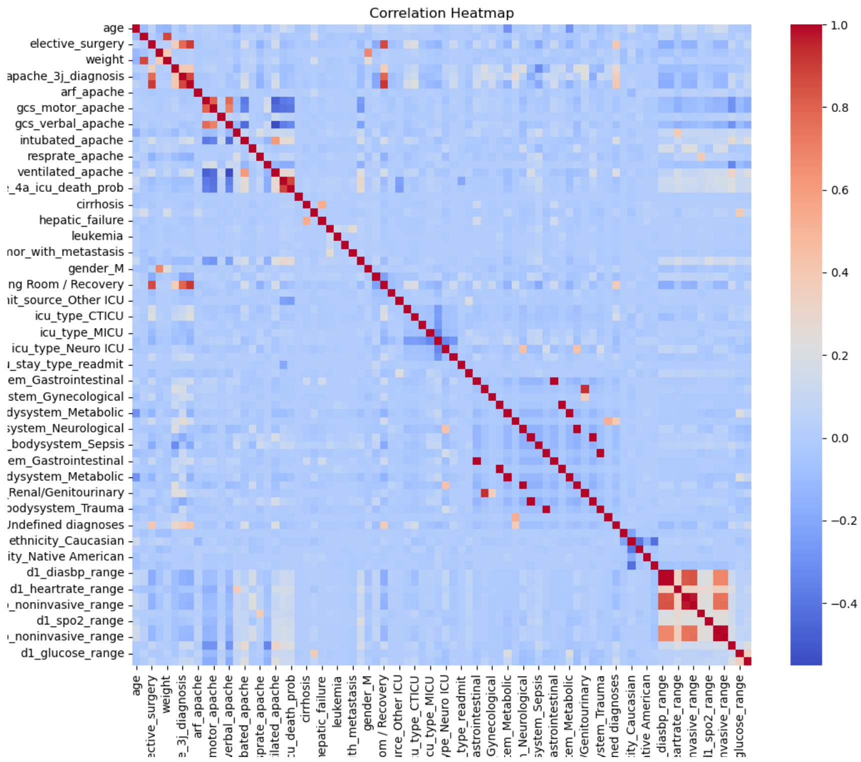

Figure 7: Correlation Matrix Before Applying High Correlation Filter (`pre_icu_los_days`)

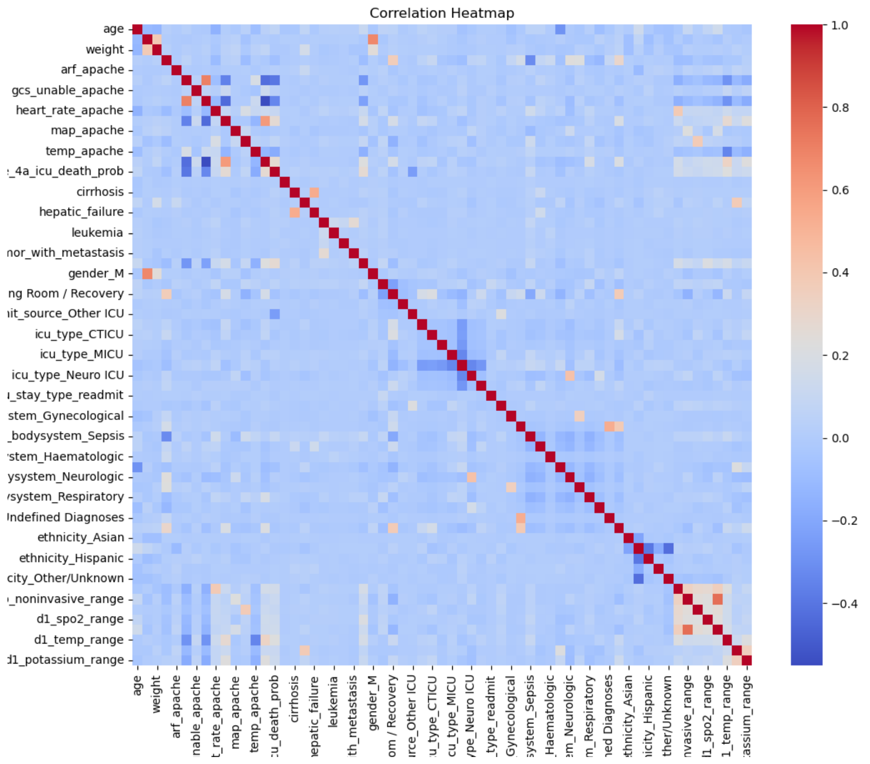

Figure 8: Correlation Matrix After Applying High Correlation Filter (`pre_icu_los_days`)

Observations for `hospital_death`: For hospital death prediction, several features were found to have high correlations and were removed during the filtering process. The dropped features included: `gcs_eyes_apache`, `d1_diasbp_range`, `d1_mbp_range`, `apache_4a_hospital_death_prob`, `apache_3j_bodysystem_Respiratory`, `apache_3j_bodysystem_Ga` `bmi`, `elective_surgery`, `apache_post_operative`

Removing these features simplified the dataset and mitigated issues caused by multicollinearity. The correlation matrix heatmaps before and after filter are provided to illustrate the impact of the high correlation filter on the dataset.

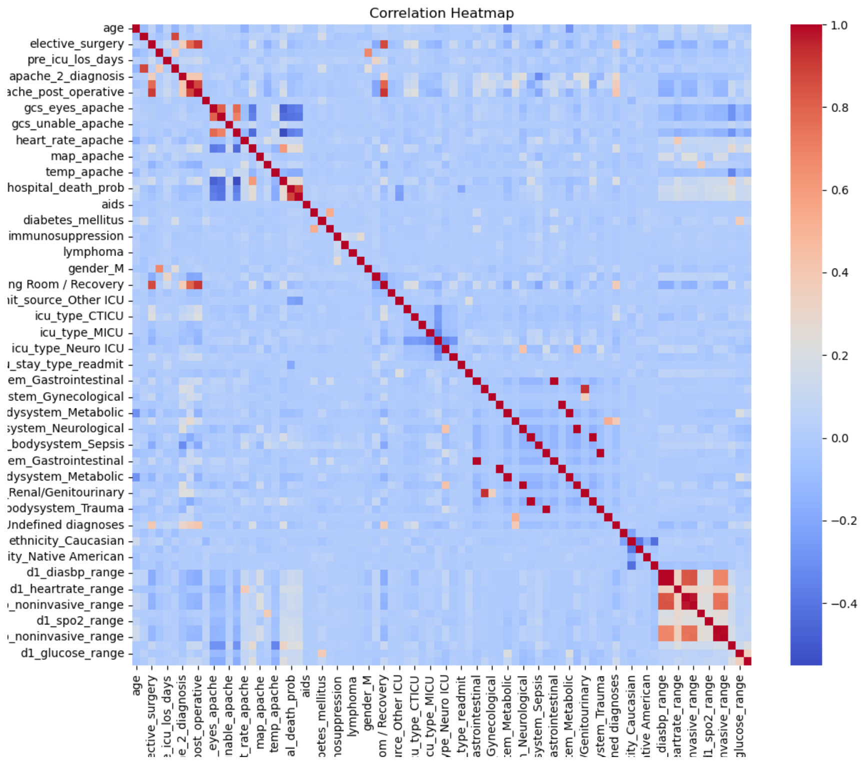

Figure 9: Correlation Matrix Before Applying High Correlation Filter (`hospital_death`)

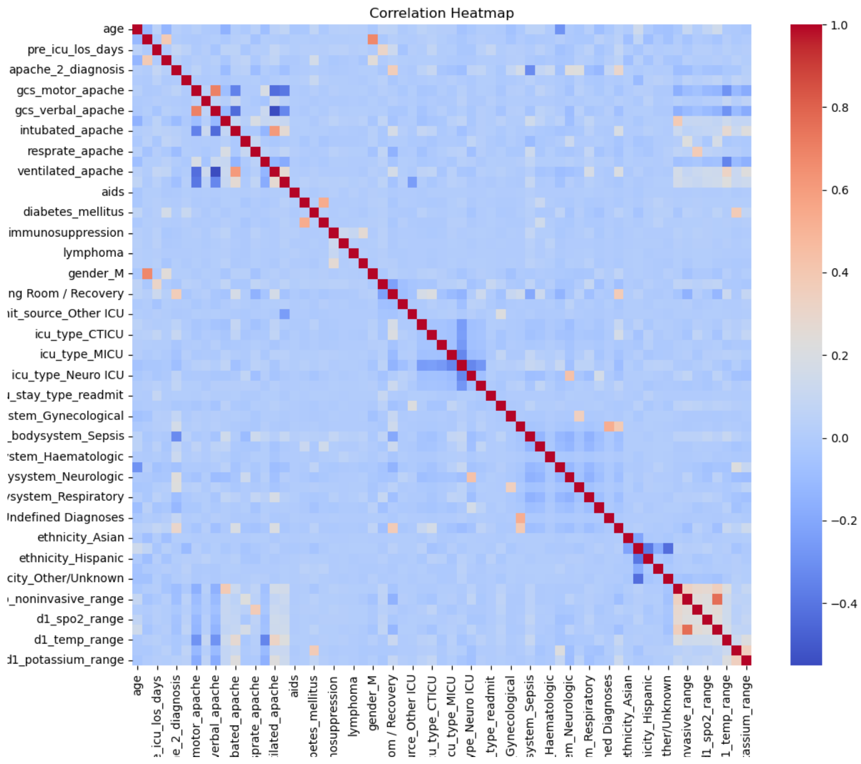

Figure 10: Correlation Matrix After Applying High Correlation Filter (`hospital_death`)

3.1.7 Chosen Dimensionality Reduction Method

After analyzing the results of various dimensionality reduction techniques, the combination of **High Correlation Filter** followed by **Random Forest Analysis** was chosen for this project.

The high correlation filter effectively addressed multicollinearity by removing redundant features that were strongly correlated with each other. This step simplified the dataset, making it more interpretable and reducing the risk of overfitting in subsequent modeling steps. By setting a correlation threshold of 0.8, features that contributed little new information were eliminated while retaining the dataset's core structure.

Following the high correlation filter, Random Forest analysis was used to identify the most important features for each target. This method is robust to noise and provides a clear ranking of feature importance, enabling the selection of the most predictive variables. By focusing on these key features, the dataset was further refined to ensure that only the most informative variables were retained for modeling.

This combined approach of High Correlation Filter and Random Forest balances dimensionality reduction and predictive power, making it an ideal choice for the objectives of this project.

3.1.8 Dimensionality Reduction Results for `pre_icu_los_days`

Dimensionality reduction for the Phase 1 target variable, `pre_icu_los_days`, was conducted using a combination of **High Correlation Filter** and **Random Forest Analysis**.

- **High Correlation Filter**: The High Correlation Filter removed 17 features with a correlation threshold of 0.8. These features, which were highly correlated with others, included clinical and categorical variables such as `apache_3j_bodysystem_Gastrointestinal`, `d1_diasbp_noninvasive_range`, and `bmi`. Removing these redundant features simplified the dataset and effectively addressed multicollinearity.

- **Random Forest Analysis**: Following the High Correlation Filter, Random Forest Analysis was applied to identify the most important features based on their contribution to predicting `pre_icu_los_days`. Using an importance threshold of 0.02, 19 features were selected. These included key clinical metrics such as `age`, `apache_4a_icu_death_prob`, `d1_glucose_range`, and `heart_rate_apache`, as well as categorical features like `icu_admit_source_Fl` and `icu_admit_source_Operating Room / Recovery`.

The dimensionality reduction process reduced the feature set from the original size to 19 features, achieving a significant simplification of the dataset while retaining the most predictive variables. The final selected features are as follows:

`icu_admit_source_Floor`, `weight`, `apache_4a_icu_death_prob`, `d1_glucose_range`, `age`, `d1_sysbp_noninvasive_range`, `height`, `d1_mbp_noninvasive_range`, `d1_heartrate_range`, `icu_admit_source_Operating Room / Recovery`, `d1_spo2_range`, `heart_rate_apache`, `temp_apache`, `d1_temp_range`, `resprate_apache`, `map_apache`, `d1_resprate_range`, `apache_2_diagnosis`, `d1_potassium_range`.

The final correlation matrix for the selected feature set is shown in figure 11.

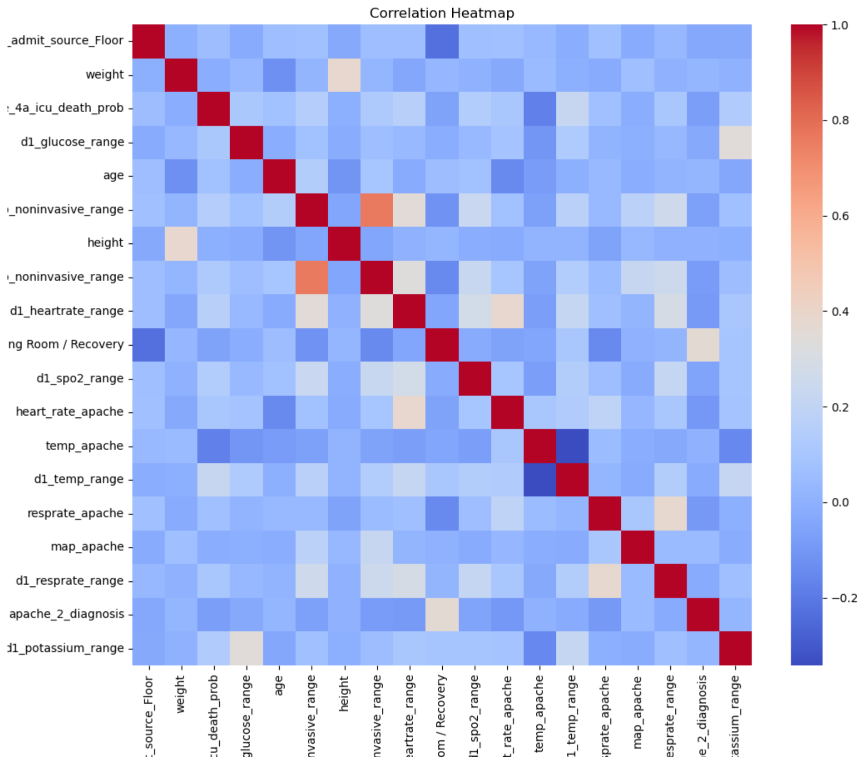

Figure 11: Final Correlation Matrix After Dimensionality Reduction (`pre_icu_los_days`)

3.1.9 Dimensionality Reduction Results for `hospital_death`

Dimensionality reduction for the Phase 3 target variable, `hospital_death`, was conducted using a combination of **High Correlation Filter** and **Random Forest Analysis**.

- **High Correlation Filter**: The High Correlation Filter removed 17 features with a correlation threshold of 0.8. These features, which were highly correlated with others, included clinical and categorical variables such as `apache_3j_bodysystem_Gastrointestinal`, `d1_diasbp_noninvasive_range`, and `bmi`. Removing these redundant features simplified the dataset and addressed multicollinearity effectively.

- **Random Forest Analysis**: Following the High Correlation Filter, Random Forest Analysis was applied to identify the most important features based on their contribution to predicting `hospital_death`. Using an importance threshold of 0.02, 19 features were selected. These included key clinical metrics such as `apache_4a_icu_death_prob`, `d1_spo2_range`, `map_apache`, and `heart_rate_apache`, as well as demographic features like `age` and `height`.

The dimensionality reduction process reduced the feature set from the original size to 19 features, achieving a significant simplification of the dataset while retaining the most predictive variables. The final selected features are as follows:

apache_4a_icu_death_prob, d1_spo2_range, d1_heartrate_range, map_apache, temp_apache, heart_rate_apache, pre_icu_los_days, weight, d1_temp_range, age, d1_sysbp_noninvasive_range, d1_mbp_noninvasive_range, d1_glucose_range, resprate_apache, d1_resprate_range, height, apache_2_diagnosis, gcs_motor_apache, d1_potassium_range.

The final correlation matrix for the selected feature set is shown in Figure 12:

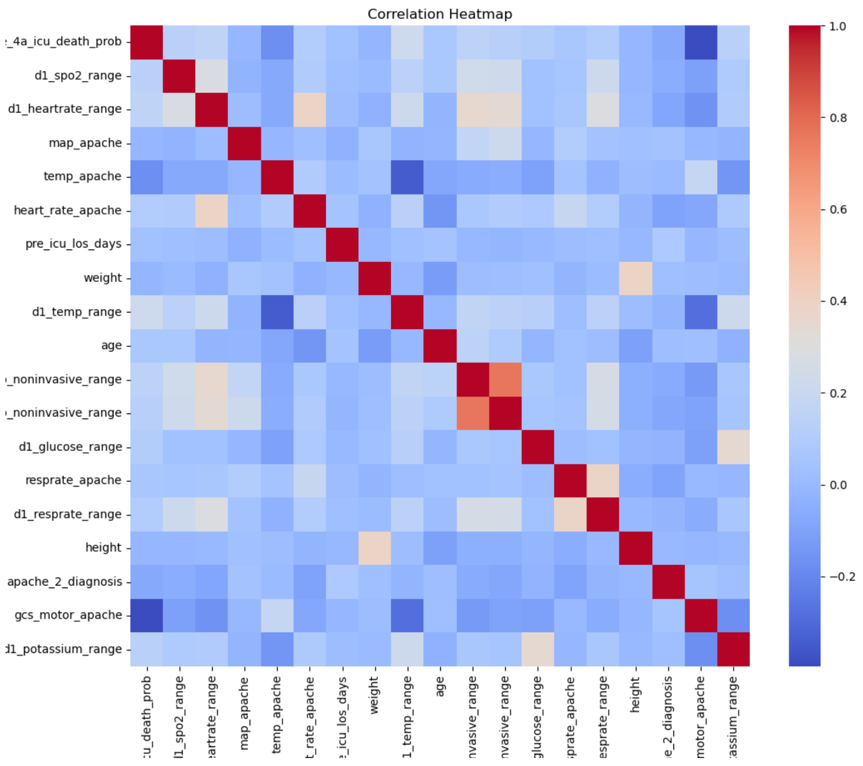

Figure 12: Final Correlation Matrix After Dimensionality Reduction (hospital_death)

3.1.10 Handling Imbalance for Target Variable (hospital_death)

The target variable hospital_death was identified as imbalanced, with a significantly lower proportion of positive cases (1) compared to negative cases (0). This imbalance can bias models to favor the majority class, reducing their ability to correctly predict rare events. To address this issue, the following strategy was applied:

- **Synthetic Minority Oversampling Technique (SMOTE)**: SMOTE was used to oversample the minority class, generating synthetic samples to balance the class distribution in the training dataset. After applying SMOTE, the training set contained an equal representation of both classes, mitigating the imbalance and enabling the model to better learn patterns for both outcomes.

This approach helps to prevent model bias towards the majority class while ensuring that the model can accurately predict both outcomes. Figure 13 and Figure 14 illustrates the class distribution of `hospital_death` before and after applying SMOTE. From the results it is clear tha the imbalance has been dealth with using SMOTE.

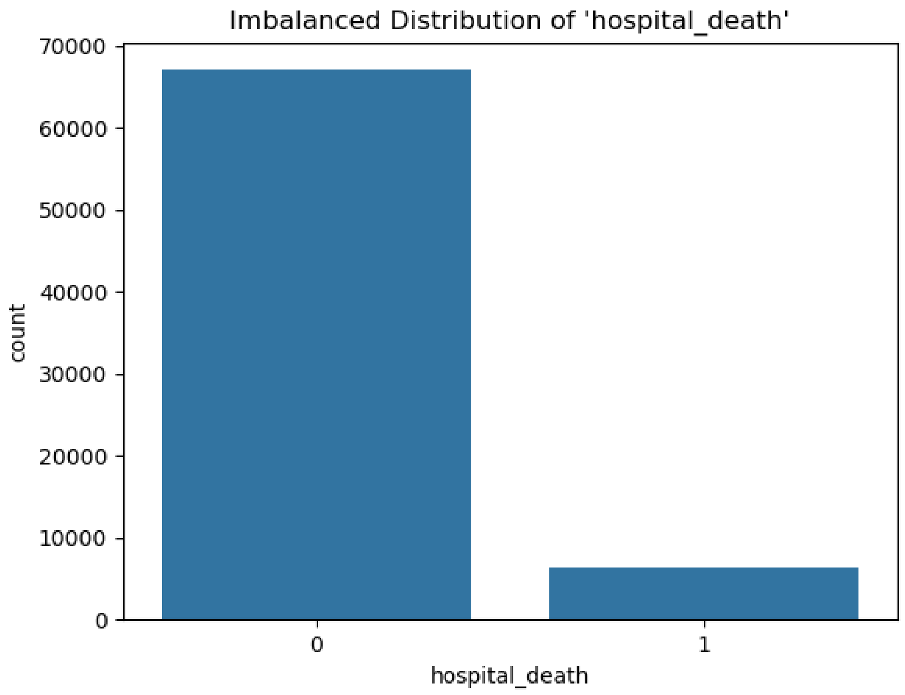

Figure 13: Class Distribution of `hospital_death` Before Applying SMOTE

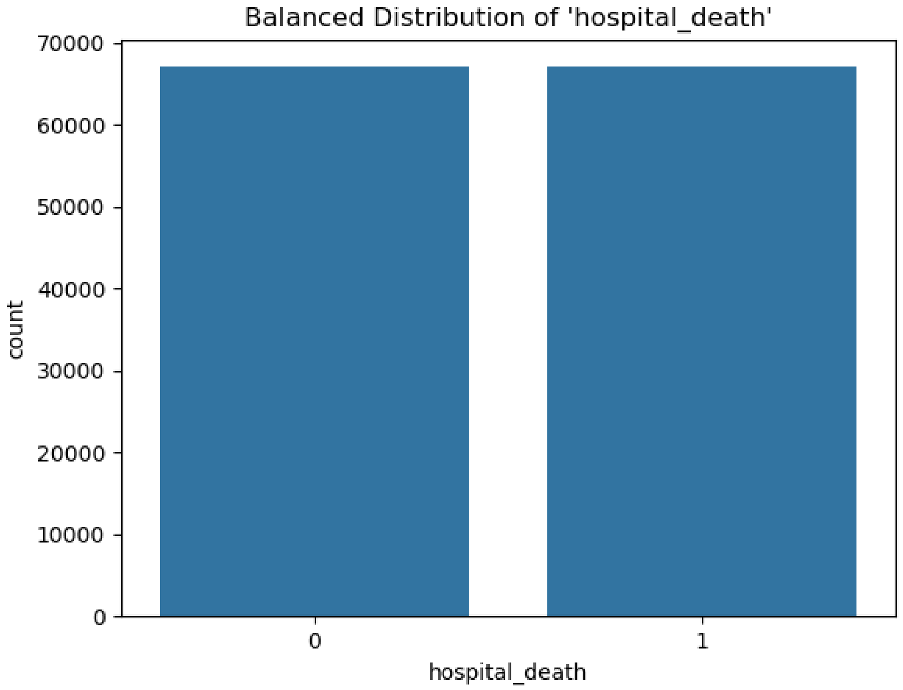

Figure 14: Class Distribution of `hospital_death` After Applying SMOTE

After completing the data cleaning and transformation steps, the dataset was fully preprocessed, with no missing values, duplicates, or irrelevant columns. The numerical features were standardized, categorical variables encoded, and redundant information aggregated. The Dimentionality Reduction techniques applied removed the high collinearity within the data set, and the transformed data set is now ready for the next phases of analysis.

4 Phase II: Regression Analysis

4.1 Overview

This phase focuses on predicting the continuous target variable, `pre_icu_los_days`, using multiple linear regression. The regression model was evaluated using statistical metrics, including R-squared, AIC, BIC, and MSE, and various tests such as T-tests, F-tests, and confidence intervals were used to analyze the predictors. Despite rigorous modeling, the overall results indicate that the linear regression model struggles to capture the complexities in the data, leading to suboptimal performance.

4.2 Model Fitting and Evaluation

The Ordinary Least Squares (OLS) regression model was fitted using selected predictors from the dataset. The evaluation metrics, summarized in Table 1, reveal that the model explains only approximately 14% of the variance in the target variable, as indicated by the

R-squared and Adjusted R-squared values. Furthermore, the minimal difference between training and testing metrics suggests that the model is neither overfitted nor underfitted but is inherently limited in capturing the underlying relationship.

Metric	**Value**
R-squared (Training)	0.1408
R-squared (Test)	0.1476
Adjusted R-squared (Training)	0.1405
Adjusted R-squared (Test)	0.1467
AIC	197042.29
BIC	197226.35
MSE (Training)	0.8583
MSE (Test)	0.8561

Table 1: Model Evaluation Metrics

4.3 T-test Analysis

Table 2 provides the p-values for each predictor variable. Several features, such as `icu_admit_source_Floor` `apache_4a_icu_death_prob`, and `age`, are statistically significant with p-values below 0.05, indicating their relevance in explaining the target variable. However, many predictors, including `weight`, `height`, and `d1_potassium_range`, exhibit high p-values, highlighting their negligible contribution to the model.

Feature	**p-value**
`icu_admit_source_Floor`	0.00
`weight`	0.29
`apache_4a_icu_death_prob`	0.00
`d1_glucose_range`	0.01
`age`	0.00
`d1_sysbp_noninvasive_range`	0.48
`height`	0.70
`d1_mbp_noninvasive_range`	0.01
`d1_heartrate_range`	0.00
`icu_admit_source_Operating Room / Recovery`	0.00

Table 2: T-test Results for Coefficients

4.4 F-test Analysis

The F-test yielded an F-statistic of 632.4758 with a p-value of 0.0000, confirming that the regression model is statistically significant compared to a baseline model with no predictors. However, statistical significance does not translate into practical utility, as reflected in the low R-squared values.

4.5 Confidence Interval Analysis

Table 3 outlines the 95% confidence intervals for the regression coefficients. Predictors like `icu_admit_source_Floor` and `apache_4a_icu_death_prob` have narrow confidence intervals that exclude zero, confirming their statistical significance. However, several features, such as `weight` and `height`, have intervals overlapping zero, suggesting minimal contribution to the model.

Feature	Lower Bound	Upper Bound
`icu_admit_source_Floor`	0.91	0.95
`weight`	-0.01	0.00
`apache_4a_icu_death_prob`	0.01	0.03
`d1_glucose_range`	0.00	0.02
`age`	0.01	0.03
`d1_sysbp_noninvasive_range`	-0.01	0.01
`height`	-0.01	0.01

Table 3: Confidence Intervals for Coefficients

4.6 Stepwise Regression and Feature Selection

Backward stepwise regression retained features such as `icu_admit_source_Floor`, `apache_4a_icu_death_pr` and `age`. Despite this refinement, the Adjusted R-squared remained low at 0.1405, underscoring the limited explanatory power of the model.

4.7 Plotting the Actual vs. Predicted Values

The plot in Figure 15 compares the actual and predicted values for the training and testing sets. Significant deviations from the ideal line highlight the model's inability to accurately predict the target variable.

4.8 Conclusion

The analysis indicates that the relationship between `pre_icu_los_days` and the selected predictors is complex and cannot be adequately captured by a linear model. Future research should explore alternative approaches, such as nonlinear regression or machine learning models, to better represent the intricacies of the data.

5 Phase III: Classification Analysis

In this phase, the goal is to predict the binary outcome of `hospital_death` using various classification techniques. The analysis involves applying multiple classifiers, evaluating their performance using various metrics, and comparing their results to identify the best-performing model.

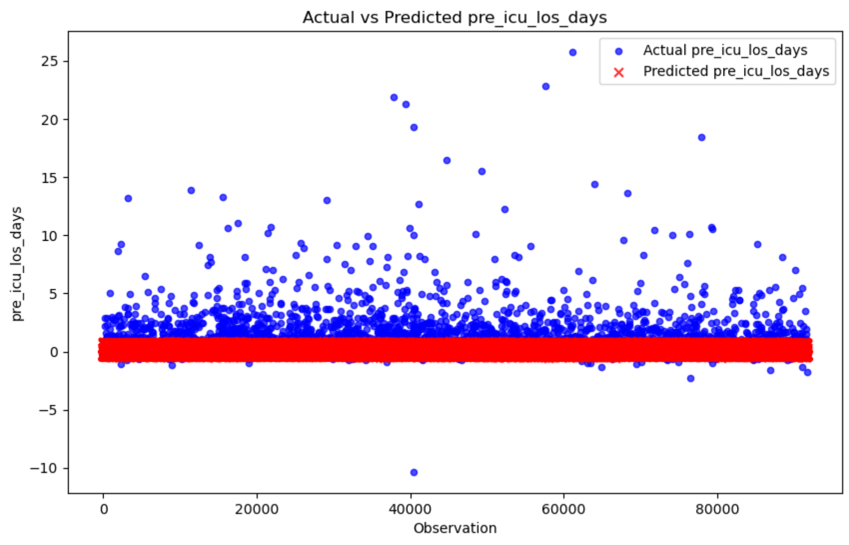

Figure 15: Actual vs Predicted Values for Training and Testing Sets

5.1 Evaluation Metrics

The performance of each classifier was assessed using the following metrics:

- **Accuracy**: Overall proportion of correct predictions.
- **Precision**: Proportion of true positives among all positive predictions.
- **Recall (Sensitivity)**: Proportion of true positives correctly identified.
- **Specificity**: Proportion of true negatives correctly identified.
- **F1-score**: Harmonic mean of precision and recall, balancing the two.
- **ROC-AUC**: Area under the Receiver Operating Characteristic curve, summarizing the trade-off between true positive rate and false positive rate.
- **Confusion Matrix**: A detailed breakdown of true positives, false positives, true negatives, and false negatives.

5.2 Classifiers Applied

The following classifiers were implemented and evaluated:

- **Logistic Regression**: Logistic Regression was employed as a baseline linear classifier to model the relationship between the features and the binary target variable (`hospital_death`). This approach provides interpretability and a robust foundation for comparison with more complex models.

  To optimize the performance of the logistic regression model, a grid search was conducted with the following hyperparameters:

- **Parameter Grid:**
    - `C`: [0.01, 0.1, 1, 10]
    - `solver`: ['liblinear', 'lbfgs']
- **Best Parameters:** `C = 10`, `solver = 'lbfgs'`

Using the best parameters found, the logistic regression model was trained and evaluated. Below are the results:

- **Evaluation Metrics:**
    - Train Accuracy: 0.7446
    - Test Accuracy: 0.7689
    - Confusion Matrix:
$$\begin{bmatrix} 12965 & 3781 \\ 458 & 1139 \end{bmatrix}$$
        - **Observation:** The confusion matrix highlights that the logistic regression model performs well on the majority class with a specificity of 0.7742, while maintaining moderate recall (0.7132) for the minority class. However, the precision (0.2315) indicates room for improvement in correctly predicting positive cases.
    - Precision: 0.2315
    - Recall (Sensitivity): 0.7132
    - Specificity: 0.7742
    - F1-score: 0.3495
    - Stratified K-fold CV Accuracy: 0.9219 ± 0.0020
        - **Observation:** The cross-validation accuracy reflects the model's stable and consistent performance across different splits, indicating good generalizability.
- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of 0.82, demonstrating the model's ability to effectively discriminate between positive and negative classes. The performance aligns well with the overall evaluation metrics, validating logistic regression as a reliable baseline.

Conclusion The logistic regression model serves as a solid baseline with a test accuracy of 0.7689 and an AUC score of 0.84. Its simplicity make it a valuable comparison point against more complex models. However, the relatively low precision (0.2315) and F1-score (0.3495) suggest that the model struggles with imbalanced class predictions. Further exploration with feature engineering or ensemble methods could improve its ability to handle minority class predictions.

- **Decision Tree Classifier**: Decision Tree Classifiers were applied to predict `hospital_death`, using both pre-pruning and post-pruning techniques to enhance performance and interpretability.

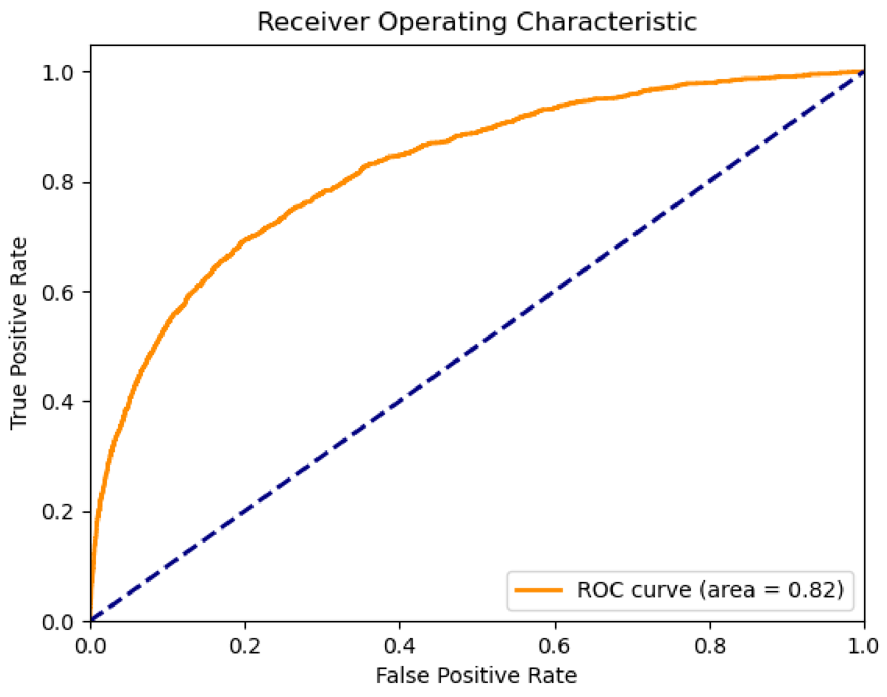

Figure 16: ROC-AUC Curve for Logistic Regression

Pre-Pruned Decision Tree Pre-pruning was implemented to control the complexity of the decision tree by setting constraints such as maximum depth, minimum samples per split, and minimum samples per leaf. This method effectively prevents the tree from overfitting while maintaining a balance between model complexity and accuracy.

To optimize the pre-pruning parameters, a grid search was conducted over the following parameter space:

- `max_depth`: [1, 2, 3, 4, 5]
- `min_samples_split`: [20, 30, 40]
- `min_samples_leaf`: [10, 20, 30]
- `criterion`: ['gini', 'entropy', 'log_loss']
- `splitter`: ['best', 'random']
- `max_features`: ['sqrt', 'log2']

The grid search identified the following optimal configuration for the pre-pruned decision tree:

- **Best Parameters:**
  * `criterion`: gini
  * `max_depth`: 5

  * `max_features`: sqrt
  * `min_samples_leaf`: 10
  * `min_samples_split`: 20
  * `splitter`: best

This configuration was used to train the pre-pruned decision tree, ensuring a balance between complexity and performance.

- **Evaluation Metrics:**
  * Train Accuracy: 0.7870
  * Test Accuracy: 0.7811
  * Confusion Matrix:

$$\begin{bmatrix} 13205 & 3541 \\ 475 & 1122 \end{bmatrix}$$

    · **Observation:** The confusion matrix indicates that the classifier performs well in identifying the majority class (specificity = 0.7885). However, the precision of the model for the minority class is relatively low (0.2406), indicating a higher rate of false positives.
  * Precision: 0.2406
  * Recall (Sensitivity): 0.7026
  * Specificity: 0.7885
  * F1-score: 0.3585
  * Stratified K-fold CV Accuracy: 0.9200 ± 0.0018
    · **Observation:** The cross-validation accuracy demonstrates that the model generalizes well across different folds, with low variability in performance. The high recall suggests the model is capable of correctly identifying a significant proportion of positive instances, though at the cost of a lower precision.

- **Observation from the Tree Visualization:** The pre-pruned decision tree demonstrates a controlled structure with limited depth and fewer nodes, indicative of the constraints imposed during training. The simplified structure reduces the risk of overfitting. However, this also limits the model's capacity to capture more complex relationships within the data.

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) of 0.82, indicating a good trade-off between sensitivity and specificity. This value reflects a reasonably strong classifier performance, even with the constraints imposed by pre-pruning. The upward curve signifies effective discrimination between classes, with room for improvement in reducing false positives.

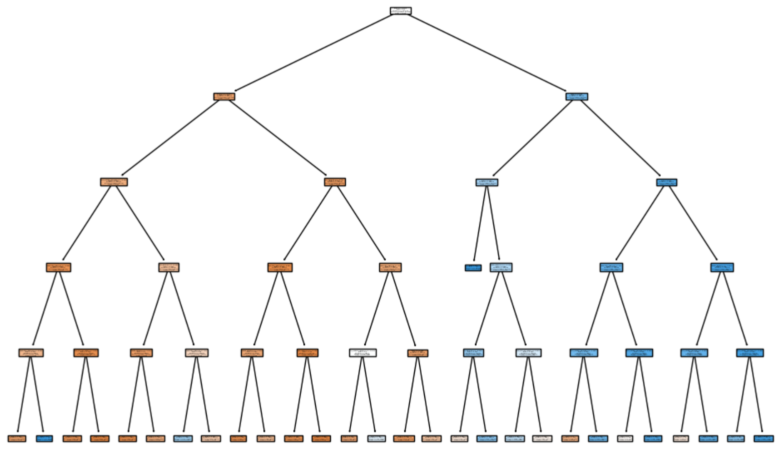

Figure 17: Visualization of Pre-Pruned Decision Tree

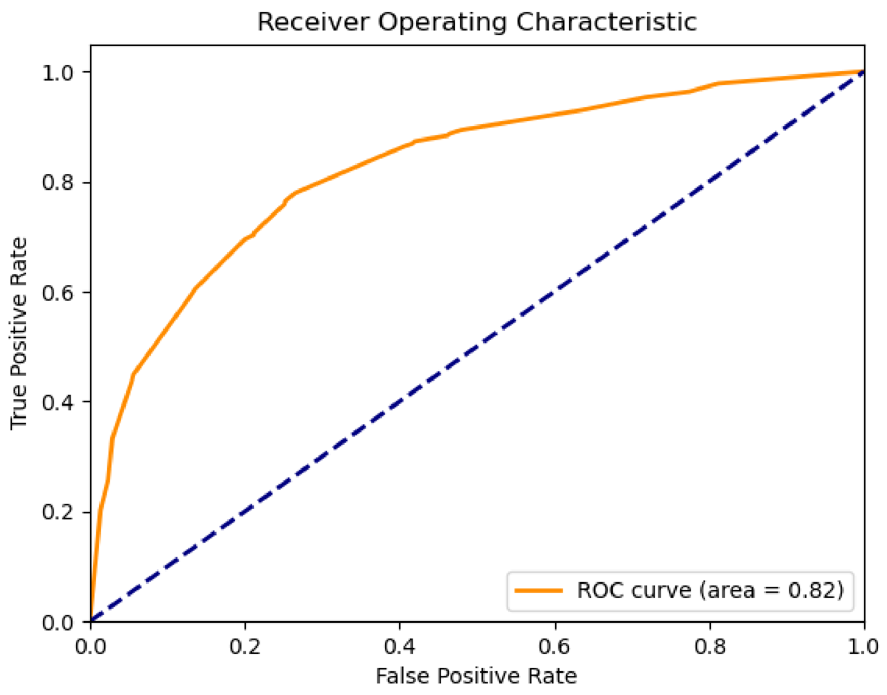

Figure 18: ROC-AUC Curve for Pre-Pruned Decision Tree

Conclusion The pre-pruned decision tree balances model complexity and predictive performance, as shown by its moderate accuracy (Train: 0.7870, Test: 0.7811) and robust cross-validation scores (0.9200 ± 0.0018). While it avoids overfitting through pre-pruning, the model exhibits limitations in handling the minority class, as observed from its lower precision and F1-score. The ROC-AUC score of 0.82 demonstrates the model's overall reliability.

Post-Pruned Decision Tree Post-pruning was applied using cost-complexity pruning, which involves pruning the tree after it has been fully grown, based on a complexity parameter (`ccp_alpha`). This approach balances model complexity and performance by removing less critical splits.

To determine the optimal `ccp_alpha`, a grid search was conducted, and the best value was identified as:

- **Optimal `ccp_alpha`:** $7.80065675359549 \times 10^{-5}$

Using this optimal parameter, the post-pruned decision tree was trained and evaluated. Below are the results:

- **Evaluation Metrics:**
  - Train Accuracy: 0.9316
  - Test Accuracy: 0.9012
  - Confusion Matrix:
$$\begin{bmatrix} 15837 & 909 \\ 903 & 694 \end{bmatrix}$$
    - **Observation:** The confusion matrix indicates a strong performance in identifying the majority class (specificity = 0.9457), with moderate precision and recall for the minority class. This is a marked improvement in class balance compared to the pre-pruned tree.
  - Precision: 0.4329
  - Recall (Sensitivity): 0.4346
  - Specificity: 0.9457
  - F1-score: 0.4337
  - Stratified K-fold CV Accuracy: 0.8825 ± 0.0043
    - **Observation:** The cross-validation accuracy shows lower variability and indicates consistent performance across different data folds. The higher specificity suggests better handling of false positives compared to the pre-pruned tree.

- **Observation from the Tree Visualization:** The post-pruned decision tree exhibits a more compact structure compared to the fully grown tree, highlighting the effectiveness of cost-complexity pruning. This reduced complexity improves generalizability while maintaining robust predictive performance.

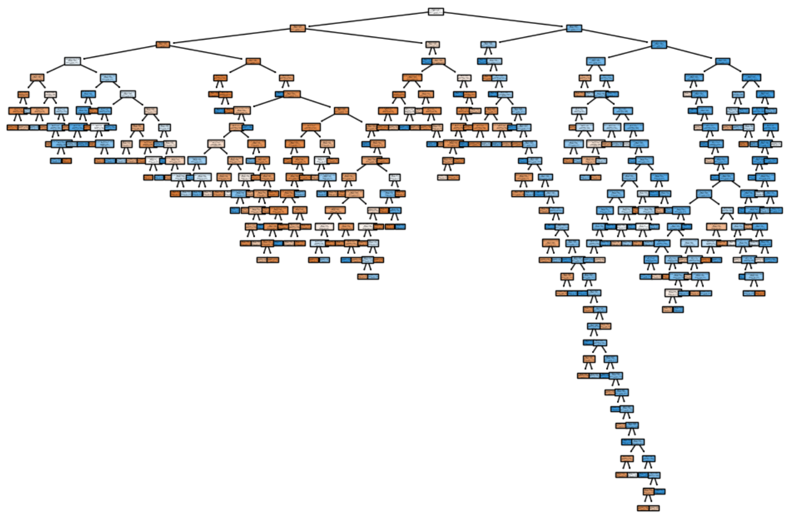

Figure 19: Visualization of Post-Pruned Decision Tree

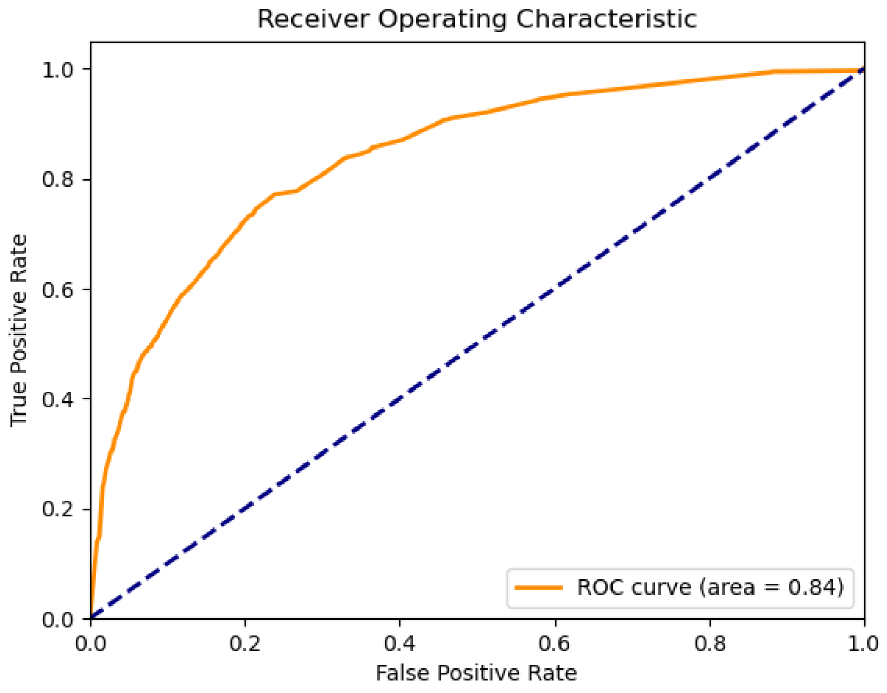

Figure 20: ROC-AUC Curve for Post-Pruned Decision Tree

– **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) of 0.84, which demonstrates an improvement over the pre-pruned tree. This highlights the model's ability to balance sensitivity and specificity effectively, with a better trade-off between true positive and false positive rates.

Conclusion The post-pruned decision tree demonstrates superior performance compared to the pre-pruned model, with higher accuracy (Train: 0.9316, Test: 0.9012) and a more balanced classification of the minority class, as shown by improved precision and recall. The use of cost-complexity pruning (`ccp_alpha` $= 7.80065675359549 \times 10^{-5}$) ensured that the tree avoided overfitting while maintaining robustness. The improved ROC-AUC score (0.84) further underscores the effectiveness of the model.

- **K-Nearest Neighbors (KNN)**: K-Nearest Neighbors (KNN) is a distance-based algorithm that predicts the class of a data point by considering the majority class among its nearest neighbors. The optimal number of neighbors (`K`) was determined using the Elbow Method, balancing the trade-off between overfitting and underfitting.

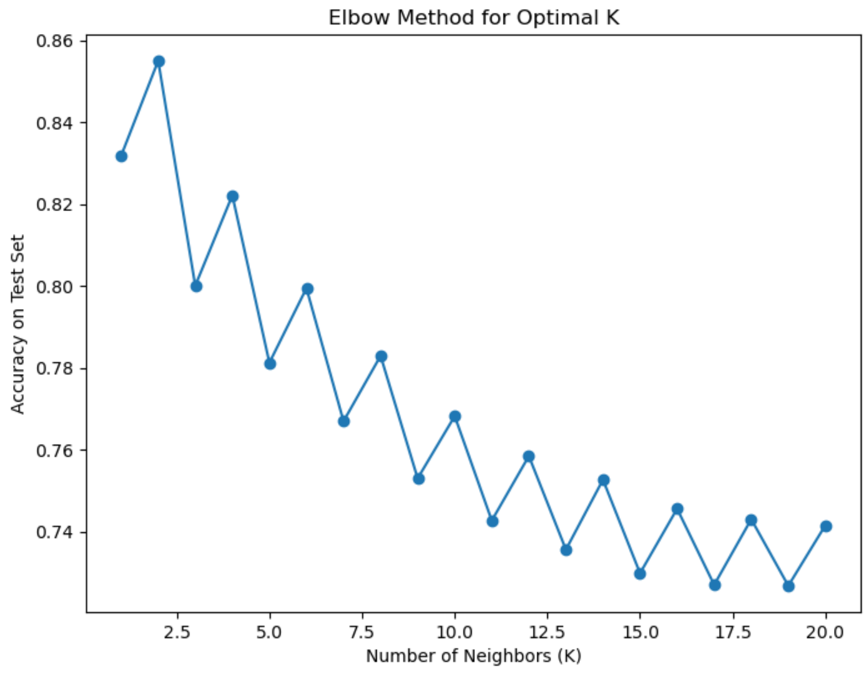

Figure 21: Elbow Method for Optimal K in KNN: Number of Neighbors vs Test Accuracy

Based on the Elbow Method, the optimal number of neighbors (`K`) was found to be **2**.

  - **Evaluation Metrics:**
    - Train Accuracy: 0.9999
    - Test Accuracy: 0.8550

- Confusion Matrix:

$$\begin{bmatrix} 15089 & 1657 \\ 1002 & 595 \end{bmatrix}$$

  - **Observation:** The confusion matrix highlights KNN's excellent performance in identifying negative cases (specificity: 0.9011), but its recall (0.3726) indicates limited ability to predict positive cases accurately.
- Precision: 0.2642
- Recall (Sensitivity): 0.3726
- Specificity: 0.9011
- F1-score: 0.3092
- Stratified K-fold CV Accuracy: 0.9153 ± 0.0013
  - **Observation:** The high cross-validation accuracy reflects KNN's stability across folds, but the large gap between test recall and specificity suggests that the model overemphasizes the majority class.

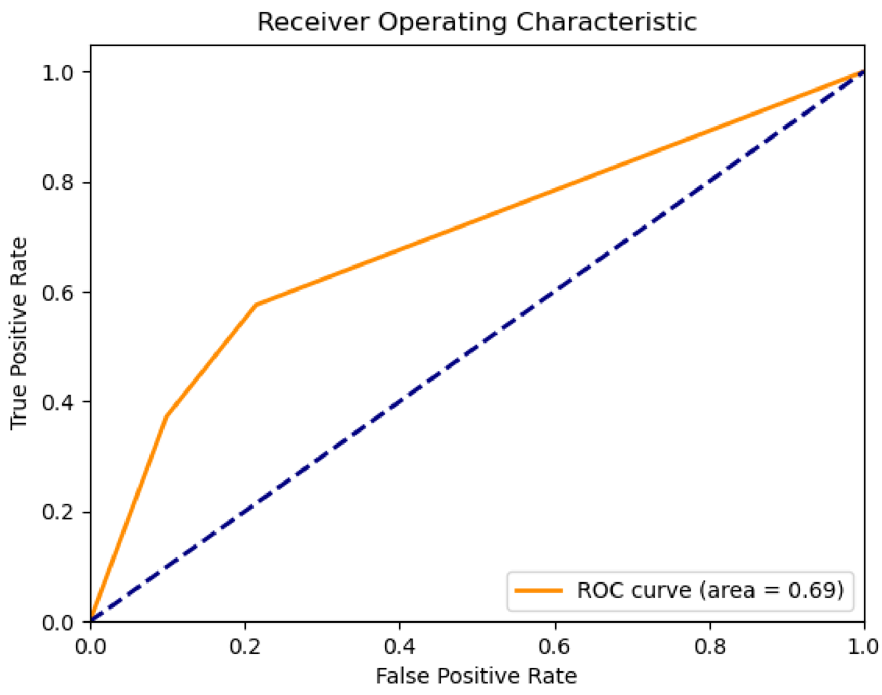

Figure 22: ROC-AUC Curve for K-Nearest Neighbors (KNN)

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of approximately 0.69, reflecting KNN's trade-off between sensitivity and specificity. While the model is highly specific, its lower recall limits its effectiveness in predicting the minority class.

Conclusions: The optimal K value of 2, determined using the Elbow Method (Figure 21), highlights the trade-off between model complexity and performance. KNN demonstrates impressive specificity (0.9011) and cross-validation accuracy (0.9153 $\pm$ 0.0013), effectively identifying negative cases in the dataset. However, its limited recall (0.3726) and low precision (0.2642) indicate challenges in predicting the minority class accurately. While KNN performs well for balanced datasets or when negative class identification is prioritized, its applicability for highly imbalanced datasets like `hospital_death` is constrained without additional techniques to improve recall and precision. The AUC score of 0.69 further underscores these limitations.

- **Support Vector Machine (SVM) with Linear Kernel**: The SVM classifier with a linear kernel was employed to evaluate the relationship between the features and the binary target variable (`hospital_death`). This kernel was chosen for its simplicity and effectiveness in handling linearly separable data.

  To optimize the model's performance, a grid search was conducted over the hyperparameter `C`, which controls the regularization strength.

  - **Parameter Grid:**
    - `C`: [0.1, 1, 10]
  - **Best Parameter:** `C = 1`

  Using the best parameter, the SVM model was trained and evaluated. Below are the results:

  - **Evaluation Metrics:**
    - Train Accuracy: 0.7446
    - Test Accuracy: 0.7689
    - Confusion Matrix:

      $$\begin{bmatrix} 12965 & 3781 \\ 458 & 1139 \end{bmatrix}$$

      Observation: The confusion matrix highlights that the logistic regression model performs well on the majority class with a specificity of 0.7742, while maintaining moderate recall (0.7132) for the minority class. However, the precision (0.2315) indicates room for improvement in correctly predicting positive cases.
    - Precision: 0.2315
    - Recall (Sensitivity): 0.7132
    - Specificity: 0.7742
    - F1-score: 0.3495
    - Stratified K-fold CV Accuracy: 0.9219 $\pm$ 0.0020 **Observation:** The cross-validation accuracy reflects the model's stable and consistent performance across different splits, indicating good generalizability.

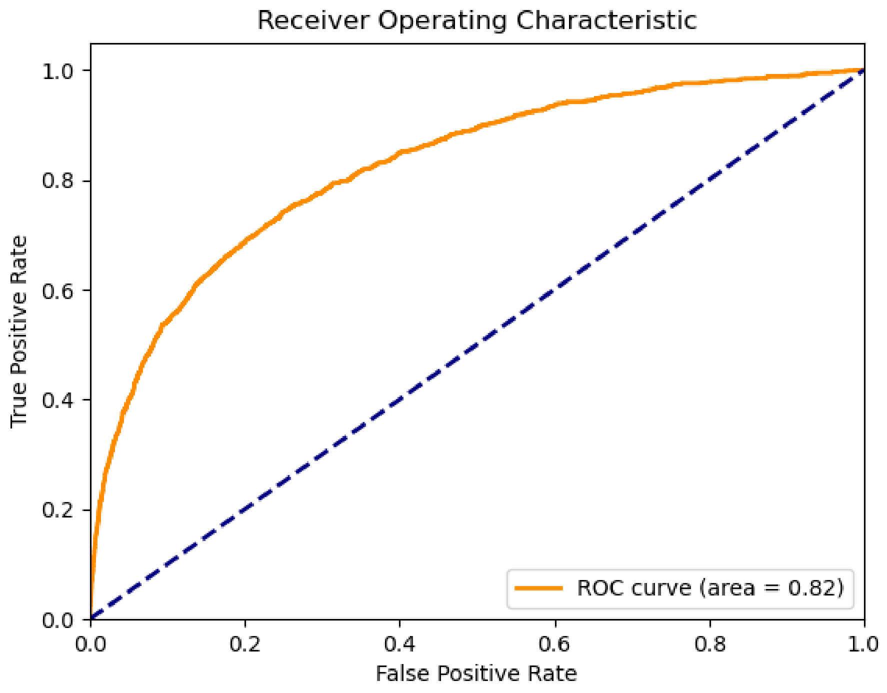

Figure 23: ROC-AUC Curve for Linear Kernal SVM

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of 0.82, demonstrating the model's ability to effectively discriminate between positive and negative classes. The performance aligns well with the overall evaluation metrics, validating logistic regression as a reliable baseline.

• **Support Vector Machine (SVM) - Polynomial Kernel**: The Polynomial Kernel SVM was utilized to capture non-linear relationships in the data by transforming the input features into a higher-dimensional space. The grid search optimized hyperparameters to enhance the model's performance:

- **Parameter Grid:**
  * `C`: [0.1, 1]
  * `degree`: [2, 3]
- **Best Parameters:** `C = 1`, `degree = 3`

Using the optimal parameters, the polynomial kernel SVM was trained and evaluated. Below are the results:

- **Evaluation Metrics:**
  * Train Accuracy: 0.7961

- Test Accuracy: 0.7886
- Confusion Matrix:

$$\begin{bmatrix} 13301 & 3445 \\ 432 & 1165 \end{bmatrix}$$

  - **Observation:** The model effectively distinguishes the majority class (specificity = 0.7943) while maintaining a recall of 0.7295 for the minority class, indicating its ability to balance sensitivity and specificity.
- Precision: 0.2527
- Recall (Sensitivity): 0.7295
- Specificity: 0.7943
- F1-score: 0.3754
- Stratified K-fold CV Accuracy: 0.9218 ± 0.0023
  - **Observation:** The cross-validation results show stable performance across folds, with high accuracy and low variability, demonstrating good generalizability.

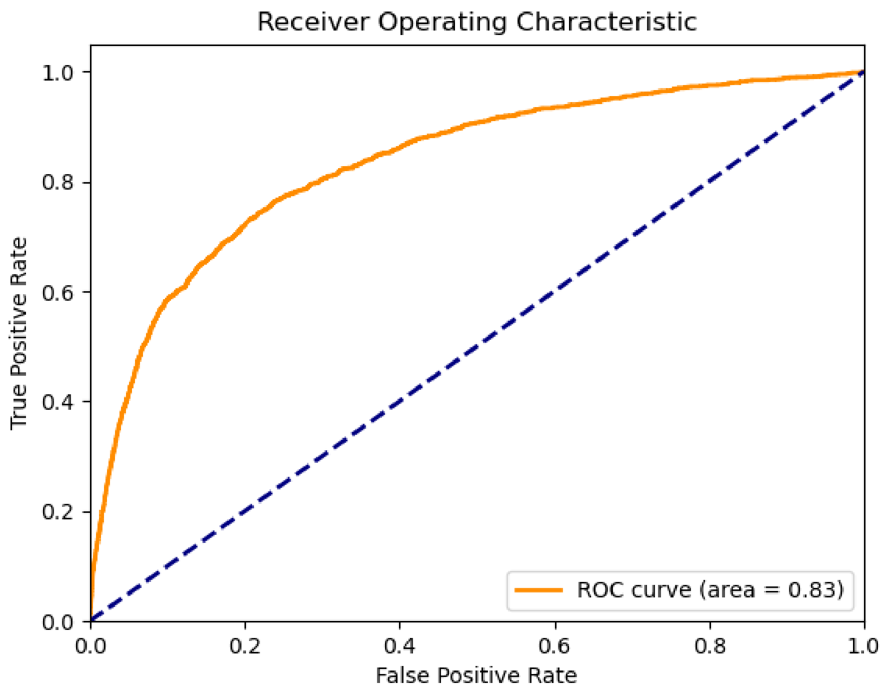

Figure 24: ROC-AUC Curve for Polynomial Kernel SVM

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of approximately 0.83, reflecting the model's capability to balance false positive and true positive rates effectively. This aligns with the overall evaluation metrics, confirming the polynomial kernel SVM's utility in capturing non-linear patterns in the data.

Conclusions: The polynomial kernel SVM, optimized with a degree of 3 and `C` = 1, successfully balances sensitivity and specificity, achieving a good overall test accuracy of 0.7886. With an F1-score of 0.3754 and an AUC of 0.83, it proves to be an effective choice for datasets where non-linear relationships are prevalent.

- **Support Vector Machine (SVM) - Radial Basis Function (RBF) Kernel**: The Radial Basis Function (RBF) kernel was employed to capture complex non-linear relationships in the data by mapping the input features into a higher-dimensional space. A grid search was performed to identify the optimal hyperparameters for the model:

  - **Parameter Grid:**
    - `C`: [0.1, 1]
    - `gamma`: ['scale', 'auto']
  - **Best Parameters:** `C = 1`, `gamma = 'auto'`

  Using the best parameters, the RBF kernel SVM was trained and evaluated. Below are the results:

  - **Evaluation Metrics:**
    - Train Accuracy: 0.8390
    - Test Accuracy: 0.8140
    - Confusion Matrix:
      $$\begin{bmatrix} 13836 & 2910 \\ 502 & 1095 \end{bmatrix}$$
      - **Observation:** The confusion matrix highlights that the RBF kernel effectively discriminates the majority class (specificity = 0.8262) while maintaining a moderate recall (0.6857) for the minority class. Precision for positive predictions is 0.2734, indicating a relatively high rate of false positives.
    - Precision: 0.2734
    - Recall (Sensitivity): 0.6857
    - Specificity: 0.8262
    - F1-score: 0.3909
    - Stratified K-fold CV Accuracy: 0.9222 ± 0.0027
      - **Observation:** The cross-validation results demonstrate a strong performance with low variability, indicating the RBF kernel SVM's stability and generalizability across different folds.

  - **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of 0.83, reflecting the model's ability to effectively balance true positive and false positive rates. The upward curve illustrates a strong trade-off between sensitivity and specificity, validating the RBF kernel's ability to capture complex patterns in the data.

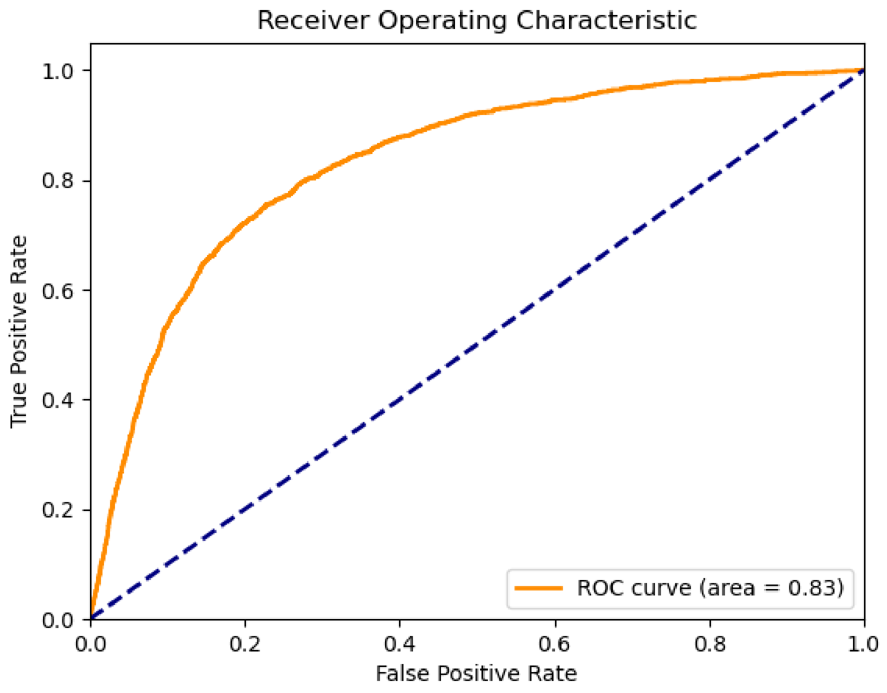

Figure 25: ROC-AUC Curve for SVM with RBF Kernel

Conclusion The RBF kernel SVM demonstrates robust performance with a test accuracy of 0.8140 and a ROC-AUC score of 0.83. The high cross-validation accuracy (0.9222 ± 0.0027) confirms the model's stability and generalizability. However, the model shows limitations in handling imbalanced classes, as evidenced by the relatively low precision (0.2734) and F1-score (0.3909).

Comparison of SVM Kernels : The evaluation of the three SVM kernels—linear, polynomial, and radial basis function (RBF)—reveals distinct strengths and weaknesses. The RBF kernel outperforms the others in terms of test accuracy, achieving 0.8140 compared to the polynomial kernel's 0.7886 and the linear kernel's 0.7689. It also achieves the highest train accuracy (0.8390) and demonstrates excellent generalizability with a cross-validation accuracy of 0.9222 ± 0.0027. Both the polynomial and RBF kernels capture non-linear relationships better than the linear kernel, as reflected in their higher test accuracies and F1-scores.

The linear kernel has the lowest precision (0.2315) and F1-score (0.3495), indicating it struggles the most with imbalanced class predictions. The polynomial kernel slightly improves precision (0.2527) and F1-score (0.3754) while maintaining balanced recall and specificity. However, the RBF kernel delivers the best F1-score (0.3909), precision (0.2734), and specificity (0.8262), showcasing its effectiveness in handling the complex patterns within the dataset.

All three kernels achieve comparable ROC-AUC scores of approximately 0.83, indicating strong discrimination capabilities across the board. However, the RBF kernel's combination of high accuracy, generalizability, and ability to capture non-linear relationships makes it the most effective choice among the three for this dataset, particularly where complex patterns are prevalent.

- **Naïve Bayes**: The Naïve Bayes classifier is a probabilistic model that applies Bayes' theorem, assuming feature independence. Despite this strong assumption, the simplicity of the model allows for fast and effective classification, especially in high-dimensional datasets. The Naïve Bayes model was evaluated on the dataset, and the results are as follows:

  - **Evaluation Metrics:**
    - Train Accuracy: 0.6961
    - Test Accuracy: 0.8164
    - Confusion Matrix:
      $$\begin{bmatrix} 14063 & 2683 \\ 685 & 912 \end{bmatrix}$$
      - **Observation:** The confusion matrix highlights the model's strong performance in predicting the majority class (specificity = 0.8398). However, its recall (0.5711) for the minority class is relatively low, indicating a higher rate of false negatives.
    - Precision: 0.2537
    - Recall (Sensitivity): 0.5711
    - Specificity: 0.8398
    - F1-score: 0.3513
    - Stratified K-fold CV Accuracy: 0.8605 ± 0.0063
      - **Observation:** The cross-validation results show moderate variability, with an average accuracy of 0.8605 and a standard deviation of ±0.0063. This indicates that the Naïve Bayes classifier performs consistently across different data splits but might have limitations in handling complex or imbalanced datasets.

  - **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of approximately 0.79, reflecting moderate capability in distinguishing between the positive and negative classes. While the Naïve Bayes classifier is effective in handling simple patterns, its performance is limited for the minority class, as indicated by its relatively low precision and recall.

Conclusion The Naïve Bayes classifier achieves a test accuracy of 0.8164, with a specificity of 0.8398 and a ROC-AUC score of 0.79. These results demonstrate the model's strength in predicting the majority class and its overall computational efficiency. However, the relatively low recall (0.5711) and F1-score (0.3513) indicate challenges in effectively handling the minority class. For this dataset, the Naïve Bayes

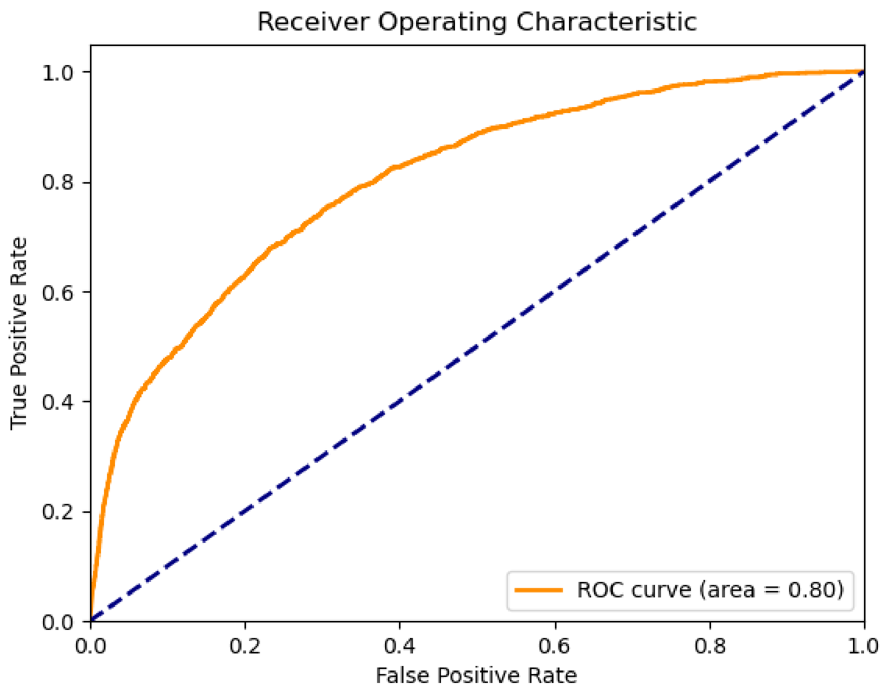

Figure 26: ROC-AUC Curve for Naïve Bayes Classifier

classifier serves as a fast and interpretable baseline, but its performance could be enhanced with techniques such as feature selection, class balancing, or hybrid models.

- **Random Forest with Bagging**: The Random Forest model with bagging was implemented to enhance the base model's predictive performance by combining multiple decision trees through bagging (bootstrap aggregating). This approach increases robustness and reduces overfitting. A grid search was conducted to optimize hyperparameters, and the results are as follows:

  - **Grid Search Parameters:**
    - `n_estimators`: [10, 50]
    - `max_samples`: [0.5, 1.0]
    - `max_features`: [0.5, 1.0]
  - **Best Parameters:**
    - `n_estimators`: 10
    - `max_samples`: 1.0
    - `max_features`: 1.0

  Using the best parameters, the model was trained and evaluated on the dataset. Below are the results:

  - **Evaluation Metrics:**

- Train Accuracy: 0.6829
- Test Accuracy: 0.9257
- Confusion Matrix:

$$\begin{bmatrix} 16584 & 162 \\ 1200 & 397 \end{bmatrix}$$

  - **Observation:** The confusion matrix highlights the model's strong performance in identifying the majority class, achieving a high specificity of 0.9903. However, the relatively low recall (0.2486) for the minority class indicates difficulty in correctly identifying positive cases, resulting in a lower F1-score (0.3683).
- Precision: 0.7102
- Recall (Sensitivity): 0.2486
- Specificity: 0.9903
- F1-score: 0.3683
- Stratified K-fold CV Accuracy: 0.9242 ± 0.0023
  - **Observation:** The cross-validation results demonstrate consistent performance across folds, with an average accuracy of 0.9242 and low variability (± 0.0023), indicating good generalizability.

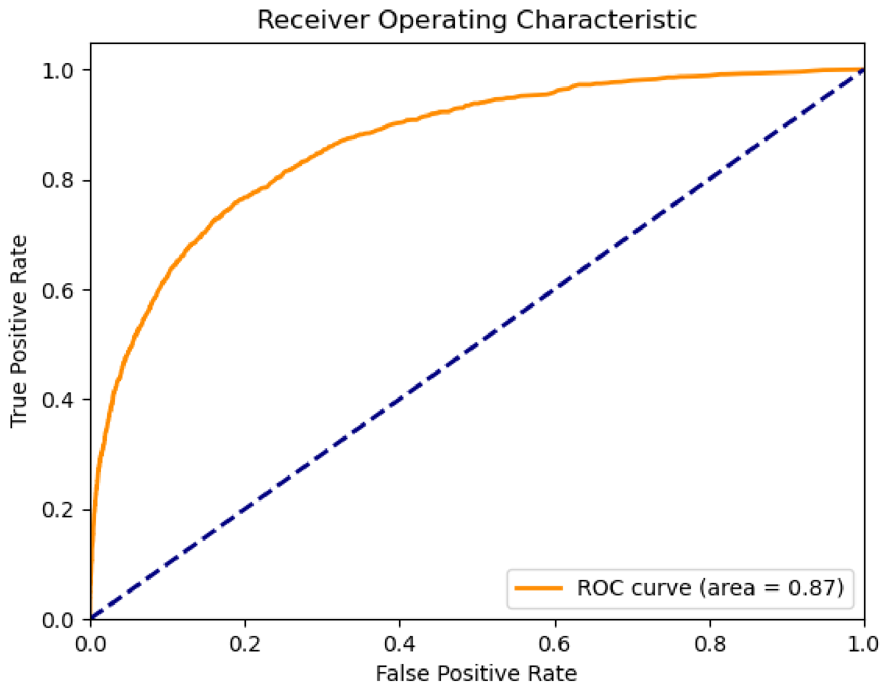

Figure 27: ROC-AUC Curve for Random Forest with Bagging

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of approximately 0.87, reflecting the model's strong ability to balance true positives and false positives. The steep initial rise in the curve indicates a strong separation between the classes.

Conclusion The Random Forest with Bagging achieves a high test accuracy of 0.9257 and specificity of 0.9903, demonstrating its effectiveness in identifying the majority class. The cross-validation accuracy of 0.9242 $\pm$ 0.0023 further supports the model's stability and generalizability. However, the model struggles with minority class predictions, as shown by the relatively low recall (0.2486) and F1-score (0.3683). Future improvements could focus on addressing class imbalance through techniques such as oversampling or cost-sensitive training.

- **Random Forest with Stacking**: Stacking combines multiple classifiers to leverage their individual strengths by training a meta-classifier on their outputs. For this implementation, a Random Forest and Support Vector Machine (SVM) were used as base learners, with a Logistic Regression model as the meta-classifier. Hyperparameters for the base models were optimized using grid search.

  - **Grid Search Parameters:**
    - `rf__n_estimators`: [100, 200]
    - `svc__C`: [0.1, 1, 10]
  - **Best Parameters:**
    - `rf__n_estimators`: 100
    - `svc__C`: 10

  Using the best parameters, the model was trained and evaluated on the dataset. Below are the results:

  - **Evaluation Metrics:**
    - Train Accuracy: 0.7651
    - Test Accuracy: 0.9247
    - Confusion Matrix:
$$\begin{bmatrix} 16475 & 271 \\ 1110 & 487 \end{bmatrix}$$
      - **Observation:** The confusion matrix highlights that the stacking model performs well on the majority class, achieving a high specificity of 0.9838. However, it struggles with correctly identifying the minority class, as indicated by the relatively low recall (0.3049).
    - Precision: 0.6425
    - Recall (Sensitivity): 0.3049
    - Specificity: 0.9838
    - F1-score: 0.4136
    - Stratified K-fold CV Accuracy: 0.9232 $\pm$ 0.0022

· **Observation:** The cross-validation results demonstrate consistent performance across folds, with an average accuracy of 0.9232 and low variability (±0.0022), reflecting the stacking model's robustness and generalizability.

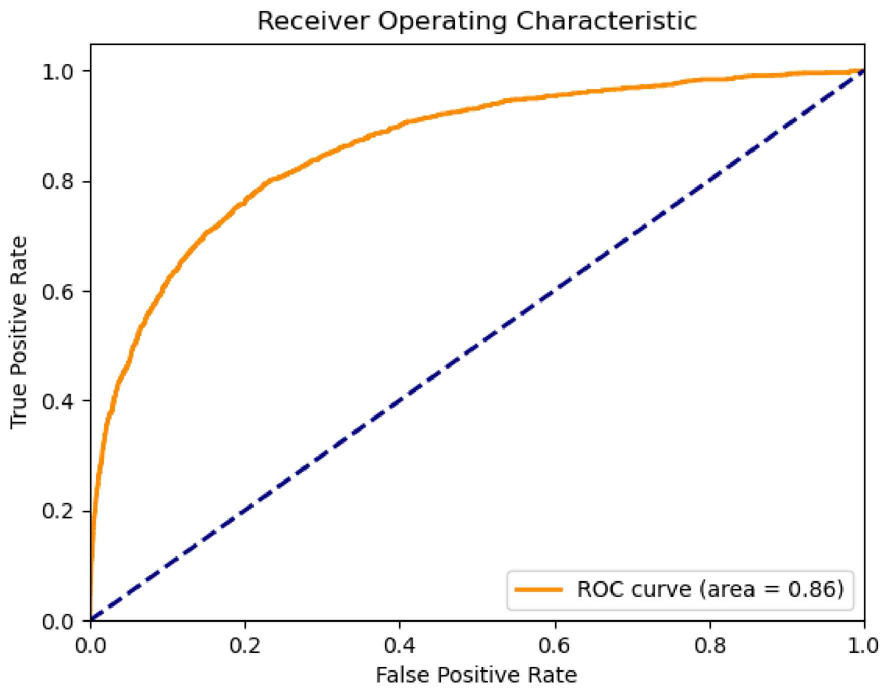

Figure 28: ROC-AUC Curve for Random Forest with Stacking

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of 0.86, reflecting the model's ability to balance true positive and false positive rates effectively. The performance aligns with the evaluation metrics, confirming the stacking model's utility for combining classifiers.

Conclusion The Random Forest with Stacking achieves a test accuracy of 0.9247 and specificity of 0.9838, demonstrating its strength in predicting the majority class. The cross-validation accuracy of 0.9232 ± 0.0022 supports the model's stability. However, the relatively low recall (0.3049) and F1-score (0.4136) highlight challenges in identifying the minority class. Future improvements could involve experimenting with additional base learners, using different meta-classifiers, or addressing class imbalance directly.

- **Random Forest with Boosting**: Boosting is an ensemble method that combines weak learners sequentially, where each subsequent learner focuses on the errors made by its predecessors. In this implementation, the XGBoost algorithm was used, and hyperparameters were optimized using grid search to achieve the best configuration.

- **Grid Search Parameters:**
  - n_estimators: [100, 200]
  - learning_rate: [0.01, 0.1]
  - max_depth: [3, 5, 7]
  - subsample: [0.8, 1.0]
  - colsample_bytree: [0.8, 1.0]
- **Best Parameters:**
  - colsample_bytree: 0.8
  - learning_rate: 0.1
  - max_depth: 7
  - n_estimators: 200
  - subsample: 1.0

Using the best parameters, the model was trained and evaluated on the dataset. Below are the results:

- **Evaluation Metrics:**
  - Train Accuracy: 0.6645
  - Test Accuracy: 0.9256
  - Confusion Matrix:
    $$\begin{bmatrix} 16519 & 227 \\ 1137 & 460 \end{bmatrix}$$
    - **Observation:** The confusion matrix highlights the model's high specificity (0.9864) in identifying the majority class. However, the relatively low recall (0.2880) indicates challenges in predicting the minority class correctly, reflected in the lower F1-score (0.4028).
  - Precision: 0.6696
  - Recall (Sensitivity): 0.2880
  - Specificity: 0.9864
  - F1-score: 0.4028
  - Stratified K-fold CV Accuracy: 0.9217 ± 0.0020
    - **Observation:** The cross-validation results show consistent performance across folds, with an average accuracy of 0.9217 and low variability (± 0.0020), demonstrating good generalizability of the boosting model.

- **Observation from the ROC-AUC Curve:** The ROC curve achieves an area under the curve (AUC) score of 0.87, reflecting the model's strong ability to distinguish between positive and negative classes. The performance aligns with the evaluation metrics, validating the boosting approach's utility for this dataset.

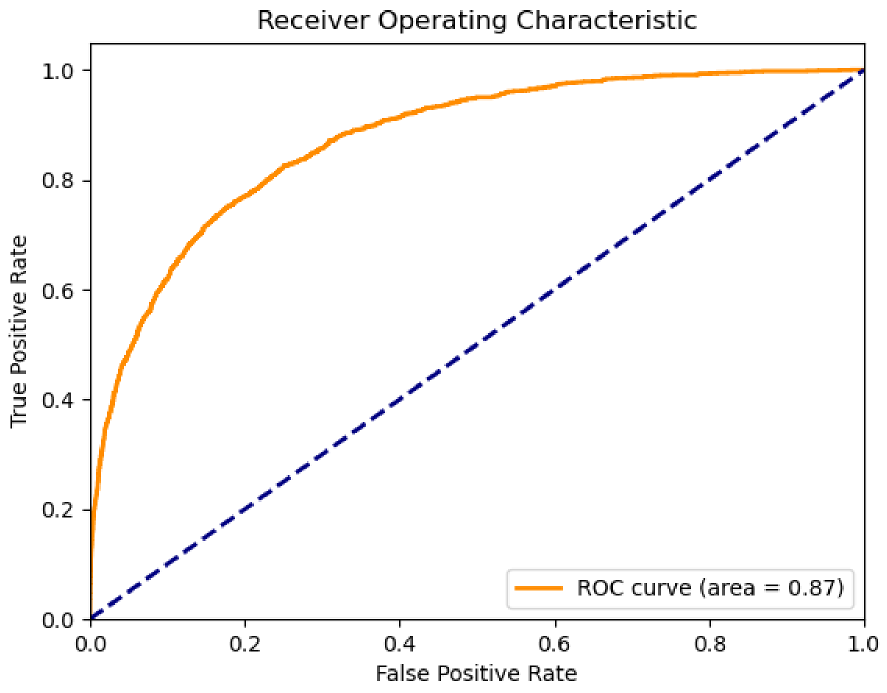

Figure 29: ROC-AUC Curve for Random Forest with Boosting

Conclusion The Random Forest with Boosting achieves a high test accuracy of 0.9256 and specificity of 0.9864, making it effective at predicting the majority class. The cross-validation accuracy of 0.9217 ± 0.0020 further confirms the model's robustness and generalizability. However, the model struggles with minority class predictions, as shown by its relatively low recall (0.2880) and F1-score (0.4028). To improve performance, future work could explore adjusting the boosting strategy, addressing class imbalance, or tuning additional hyperparameters.

- **Neural Networks**: A multi-layer perceptron (MLP) was employed to model the complex, non-linear relationships present in the dataset. Grid search was used to tune hyperparameters and identify the optimal architecture and training configuration.
  - **Grid Search Parameters:**
    - Hidden Layer Sizes: `(50,)`, `(100,)`, `(50, 50)`
    - Activation Functions: `tanh`, `relu`
    - Solvers: `adam`, `sgd`
    - Learning Rate Initialization: `0.001`, `0.01`
  - **Best Parameters:**
    - Activation: `relu`
    - Hidden Layer Sizes: `(100,)`
    - Solver: `sgd`

    * Learning Rate Initialization: `0.001`

Using the optimal parameters, the MLP was trained and evaluated on the dataset. Below are the results:

- **Evaluation Metrics:**
    * Train Accuracy: 0.6165
    * Test Accuracy: 0.9239
    * Confusion Matrix:
$$\begin{bmatrix} 16528 & 218 \\ 1177 & 420 \end{bmatrix}$$
        · **Observation:** The confusion matrix demonstrates the model's strong specificity (0.9870) in predicting the majority class. However, its low recall (0.2630) for the minority class indicates difficulty in identifying positive cases, reflected in the low F1-score (0.3758).
    * Precision: 0.6583
    * Recall (Sensitivity): 0.2630
    * Specificity: 0.9870
    * F1-score: 0.3758
    * Stratified K-fold CV Accuracy: 0.9228 ± 0.0031
        · **Observation:** The cross-validation accuracy reflects the model's robustness and consistent performance across different folds, with minimal variability (± 0.0031).

- **Observation from the ROC-AUC Curve:** The ROC-AUC curve achieves an area under the curve (AUC) score of approximately 0.81, illustrating the model's capability to discriminate between positive and negative classes. However, the trade-off between sensitivity and specificity is evident, suggesting potential room for improvement.

Conclusion The neural network, configured with one hidden layer of 100 neurons, `relu` activation, the `sgd` solver, and a learning rate of 0.001, achieves a high test accuracy of 0.9239 and exceptional specificity (0.9870). The strong cross-validation accuracy (0.9228 ± 0.0031) further confirms its reliability. However, the low recall (0.2630) and F1-score (0.3758) highlight challenges in predicting the minority class. To address this, techniques such as oversampling, class weighting, or more advanced architectures could be explored.

5.3 Model Comparisons and Results

Each classifier was tuned using hyperparameter optimization and evaluated using stratified k-fold cross-validation. The results of the analysis are provided by Table 5.3

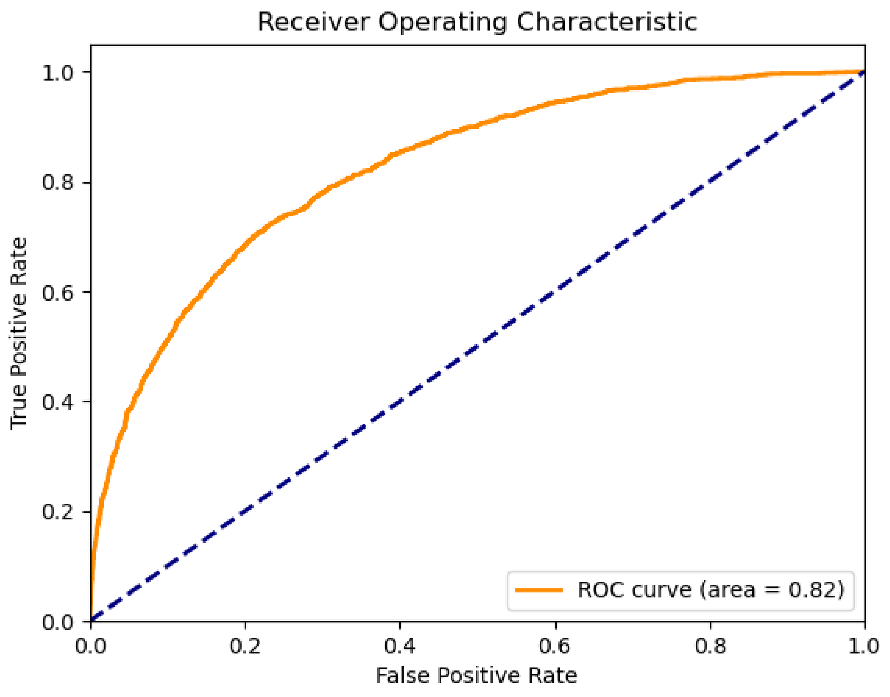

Figure 30: ROC-AUC Curve for Neural Networks (MLP)

Classifier	Precision	Recall	Specificity	F1-score	ROC-AUC Score	Stratified K-fold CV	Test-Accuracy	Train-Accuracy
Logistic Regression	0.23	0.71	0.77	0.35	0.82	0.92 ± 0.0020	0.77	0.74
Pre Pruned Decision Tree	0.24	0.70	0.79	0.36	0.82	0.92 ± 0.0018	0.78	0.79
Post Pruned Decision Tree	0.43	0.43	0.95	0.43	0.84	0.88 ± 0.0043	0.90	0.93
KNN	0.26	0.37	0.90	0.31	0.69	0.92 ± 0.0013	0.86	1.00
Linear Kernel SVM	0.23	0.71	0.77	0.35	0.82	0.92 ± 0.0020	0.77	0.74
Polynomial Kernel SVM	0.25	0.73	0.79	0.38	0.83	0.92 ± 0.0023	0.79	0.80
Radial Base Kernel SVM	0.27	0.69	0.83	0.39	0.83	0.92 ± 0.0027	0.81	0.84
Naïve Bayes	0.25	0.57	0.84	0.35	0.80	0.86 ± 0.0063	0.82	0.70
Random Forest-Bagging	0.71	0.25	0.99	0.37	0.87	0.92 ± 0.0023	0.93	0.68
Random Forest-Stacking	0.64	0.30	0.98	0.41	0.86	0.92 ± 0.0022	0.92	0.77
Random Forest-Boosting	0.67	0.29	0.99	0.40	0.87	0.92 ± 0.0020	0.93	0.66
Neural Networks	0.66	0.26	0.99	0.38	0.82	0.92 ± 0.0031	0.92	0.62

Table 4: Overview of Evaluated Classifiers

5.4 Best Classifier Recommendation

Based on the results in Table 5.3, the Post Pruned Decision Tree is the best choice for predicting hospital mortality (`hospital_death`). This model performs well on key measures that are important in healthcare, where it is critical to correctly identify serious cases and avoid unnecessary alarms.

The Post Pruned Decision Tree has:

- **High Specificity (0.95)**: It is very good at correctly identifying patients who will survive, reducing false alarms.
- **Balanced Precision and Recall (0.43 each)**: This balance means it is reasonably good at finding actual deaths without predicting too many incorrect cases.

- **Strong F1-Score (0.43)**: This shows it handles the trade-off between precision and recall well.
- **Highest ROC-AUC Score (0.84)**: This indicates it can reliably tell the difference between patients who will die and those who will not.
- **Stable Performance**: Its consistent cross-validation score (0.88 ± 0.0043) and high test accuracy (0.90) show it works well on new data.

Although Random Forest models like Boosting have slightly better ROC-AUC scores, they have very low recall (e.g., 0.25-0.30), meaning they miss many serious cases. This makes them less suitable for a healthcare application.

In summary, the Post Pruned Decision Tree provides a good balance of accuracy, reliability, and interpretability, making it the best option for this task.

6 Phase IV: Clustering and Association Rule Mining

This section details the clustering and association rule mining analysis conducted on the dataset, including the application of K-means (with silhouette analysis), DBSCAN, and the Apriori algorithm.

6.1 K-Means Clustering

K-means clustering was performed on the dataset to partition the data into distinct clusters. Silhouette analysis and within-cluster sum of squares (inertia) were used to determine the optimal number of clusters.

6.1.1 Optimal Number of Clusters

The silhouette scores and within-cluster variation (inertia) were computed for different values of k (from 2 to 10). Based on the plots of these metrics (Figures 31 and 32), the optimal number of clusters was determined to be $k = 2$.

6.1.2 Clustering Results for $k = 2$

K-means clustering with $k = 2$ was performed, and the following metrics were used to evaluate the quality of clustering:

- **Silhouette Score:** 0.22
- **Davies-Bouldin Index:** 2.22
- **Calinski-Harabasz Index:** 13049.43
- **Adjusted Rand Index (ARI):** 0.16
- **Mutual Information Score:** 0.03

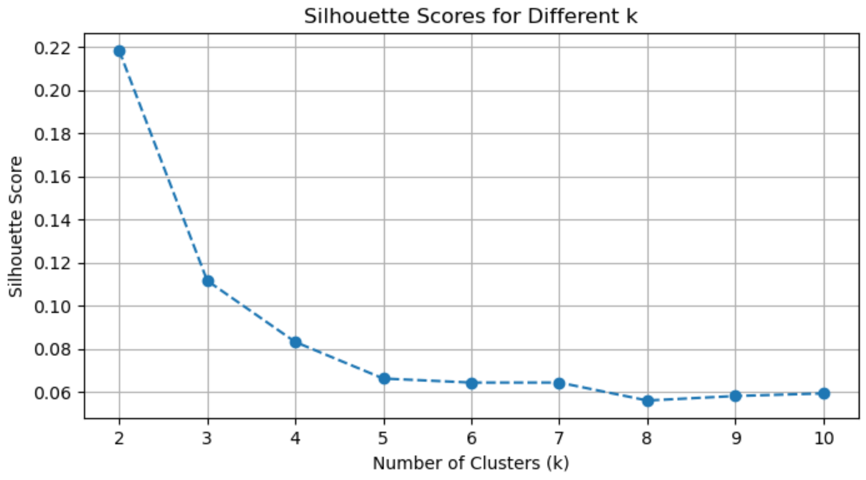

Figure 31: Silhouette Scores for Different Values of k

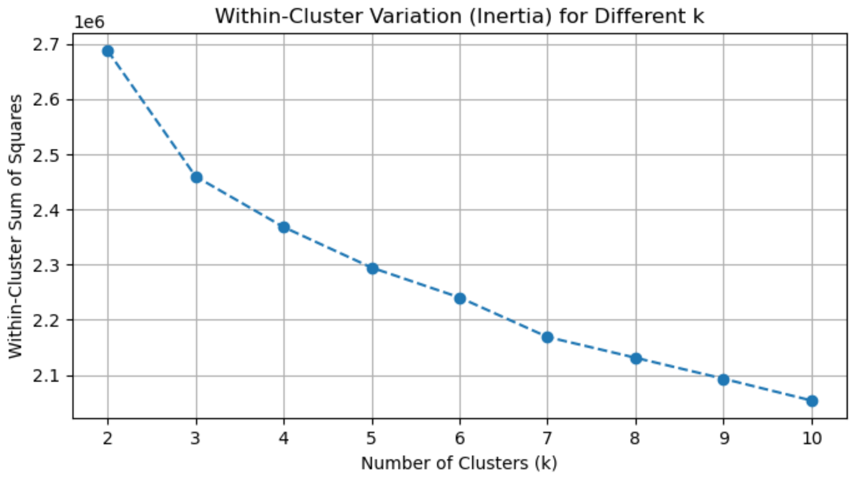

Figure 32: Within-Cluster Sum of Squares (Inertia) for Different Values of k

The clustering results for $k = 2$ show that the clusters are not very well-separated. The silhouette score of 0.22 suggests weak separation, meaning that many data points are not clearly assigned to one cluster. The Davies-Bouldin Index of 2.22 also indicates that the clusters overlap slightly. However, the high Calinski-Harabasz Index of 13,049.43 shows that the clusters are spread out and somewhat distinct from each other. The low Adjusted Rand Index (0.16) and Mutual Information Score (0.03) suggest that the clustering does not align well with any known labels. Overall, the results indicate some structure in the data, but the clusters are not clearly defined.

6.2 DBSCAN Clustering

The DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm was implemented to identify clusters based on density. This approach is especially useful for discovering clusters of arbitrary shapes and isolating noise points.

- **Implementation Details:**
  - ε (Neighborhood Radius): 0.5
  - Minimum Samples: 5
  - Distance Metric: Euclidean Distance
- **Results:**
  - Number of Clusters: 0
  - Number of Noise Points: 91713
  - Cluster Visualizations: The visualization of the DBSCAN clustering is shown below, where all points were classified as noise.

Observation: In this case, DBSCAN classified all points as noise, resulting in no clusters being detected. This indicates that the selected parameters for ε and `min_samples` may not be well-suited to the data, and further tuning or a different approach may be required.

6.3 Association Rule Mining using the Apriori Algorithm

For the Association Rule Mining task, only the binary features in the dataset were used. These binary columns represent various clinical and demographic features related to patients, and are listed below:

`elective_surgery, apache_post_operative, arf_apache, gcs_unable_apache, intubated_apach ventilated_apache, aids, cirrhosis, diabetes_mellitus, hepatic_failure, immunosuppression leukemia, lymphoma, solid_tumor_with_metastasis, hospital_death`

The Apriori algorithm was applied with the following parameters:

- **Minimum Support**: 0.05 (to ensure frequent itemsets)
- **Metric for Rule Evaluation**: `lift`
- **Minimum Lift Threshold**: 1 (to capture only meaningful rules)

After applying the Apriori algorithm, association rules were generated, which highlight the relationships between different clinical variables in the dataset.

Top 5 Association Rules:

- Rule 1: If a patient is `intubated_apache`, then they are likely to be `ventilated_apache` with 100% confidence and a lift of 3.09.
- Rule 2: If a patient is `intubated_apache` and `apache_post_operative`, they are likely to be `ventilated_apache` with 100% confidence and a lift of 3.09.
- Rule 3: If a patient is `ventilated_apache` and `elective_surgery`, they are likely to undergo `apache_post_operative` with 98% confidence and a lift of 4.89.

- Rule 4: If a patient has `elective_surgery`, they are likely to have undergone `apache_post_operative` with 97% confidence and a lift of 4.82.
- Rule 5: If a patient has `apache_post_operative`, they are likely to have `elective_surgery` with 88% confidence and a lift of 4.82.

These rules suggest strong associations between key clinical conditions. For example, the rule that connects `intubated_apache` and `ventilated_apache` with a confidence of 100% suggests that these two conditions almost always co-occur in the dataset. Similarly, the relationship between `elective_surgery` and `apache_post_operative` with high confidence and lift indicates that surgery-related conditions often lead to post-operative care.

The Apriori algorithm identified several interesting relationships, such as the high-confidence rule between `intubated_apache` and `ventilated_apache`, suggesting a direct link between intubation and ventilation in the clinical context. Additionally, the relationships between `elective_surgery` and `apache_post_operative` could provide insights into the treatment and post-operative recovery patterns of patients.

7 Recommendations

This project explored multiple machine learning techniques applied to healthcare data, uncovering actionable insights for predicting *hospital death* and *pre ICU length of stay (LOS)*. The analysis covered regression, classification, clustering, and association rule mining, highlighting both successes and opportunities for future improvements.

7.1 Key Learnings

The regression analysis aimed to predict the continuous target variable *pre ICU LOS*. The multiple linear regression model achieved an R-squared value of approximately 0.14, with features such as *apache 4a ICU death prob*, *age*, and *ICU admit source* showing statistical significance. However, the model's limited explanatory power suggested the inadequacy of linear models for capturing the underlying complexities in the data.

For the classification task, the Post-Pruned Decision Tree emerged as the best-performing model. It achieved a test accuracy of 0.90, specificity of 0.95, and a ROC-AUC score of 0.84, striking a balance between minimizing false alarms and maintaining recall. Features such as *apache 4a ICU death prob*, *d1 SPO2 range*, and *age* were identified as key predictors. While ensemble models like Random Forest with Boosting offered high accuracy, their low recall made them less suitable for healthcare scenarios requiring balanced class predictions.

Association rule mining using the Apriori algorithm revealed meaningful relationships within the dataset. Rules like *high apache 4a ICU death prob* $\rightarrow$ *hospital death* provided interpretable insights into the factors contributing to mortality. These associations align with clinical understanding, supporting their practical relevance in patient risk assessment.

Clustering analysis revealed the presence of two optimal clusters in the feature space, aligning with the dataset's binary outcome. This exploratory step uncovered latent structures that could guide future unsupervised learning applications.

7.2 Future Work and Improvements

To build on the results of this project, the following strategies are recommended to enhance both regression and classification tasks, as well as the broader analytical framework:

1. **Regression Analysis:** To enhance the fit for *pre ICU LOS*, future work should explore non-linear regression models such as support vector regression (SVR) with non-linear kernels or decision tree-based regression techniques. Incorporating polynomial features or interaction terms may also improve the model's ability to capture complex relationships.

2. **Classification Models:** Advanced ensemble techniques, such as gradient boosting or hybrid models that combine tree-based methods with neural networks, could improve recall while maintaining high specificity.

3. **Parameter Optimization:** Exhaustive hyperparameter searches, including Bayesian optimization and evolutionary algorithms, could identify optimal configurations for both regression and classification models.

4. **Feature Engineering and Selection:** Developing domain-specific feature selection could improve model inputs. Additionally, exploring time-series representations of patient data might uncover temporal patterns.

7.3 Conclusion

This project demonstrates the utility of machine learning in healthcare analytics, with the Post-Pruned Decision Tree standing out as the most reliable classifier for predicting hospital mortality. The association rule mining results provided interpretable relationships that align with clinical knowledge, while clustering uncovered latent structures in patient data. However, the modest performance of the regression model highlights the need for non-linear approaches to better capture complex patterns.

Future work should focus on integrating advanced models, expanding datasets, and improving class imbalance handling. These steps, along with the use of explainability frameworks, will strengthen the practical applicability of machine learning models in healthcare decision-making, bridging the gap between data science innovations and clinical practice.

References

[1] Kaggle. (n.d.). Patient Survival Prediction. Available at: https://www.kaggle.com/datasets/mitishaagarwal/patient/data. Accessed: 2024-12-08.