CAPSTONE PROJECT REPORT

DATA-DRIVEN EXPLORATION: UNLOCKING PREDICTIVE POTENTIAL AND INSIGHTS INTO ICU OUTCOMES IN SEPSIS

Aaryan Nagpal and Maanas Kejriwal

Supervisor: Dr. Lipika Dey

Department of Computer Science

Ashoka University

Monsoon 2024

ABSTRACT

This study utilizes the MIMIC-IV database to explore key factors influencing in-hospital mortality and ICU Length of Stay (LOS) in sepsis patients. The research begins with a detailed analysis of ICD-9 and ICD-10 diagnosis codes, such as sepsis, acidosis, and ARDS, to identify patterns and relationships that impact patient outcomes. This exploration informed the subsequent development of machine learning models for predicting mortality and LOS.

Several models, including Random Forest, XGBoost, LightGBM, and AdaBoost, were applied to the data, with attention to feature selection, data balance, and model tuning. Performance evaluations were conducted on various train-test splits and Minimal Data Retention Ratios (MDRR), demonstrating promising results. An ensemble approach was also employed to enhance model accuracy and robustness.

Additionally, a user-friendly data visualization tool was created to support clinical decision-making, offering clear insights into model predictions and their relationship to blood parameters and other clinical features.

This study focuses on the exploration of the MIMIC-IV dataset as a foundational step for the upcoming thesis. The aim is to generate insights from electronic health record (EHR) data, ultimately contributing to the development of predictive modeling tools to support clinicians in assessing patient prognosis.

Keywords MIMIC-IV, mortality, length of stay, data preprocessing, data visualization, sepsis, blood parameters

Acknowledgement

We would like to express our deepest gratitude to our advisor and mentor, Dr. Lipika Dey from the Computer Science Department at Ashoka University for her invaluable guidance, insightful contributions, and active involvement throughout the course of this project. Her thoughtful feedback played a crucial role in shaping the direction and outcomes of the research, without which this project would not have been possible.

Our sincere thanks also goes to Dr. Mayank Garg from the Koita Center for Digital Health (KCDH-A) for his generous support and profound biological insights, which helped us draw important connections between the data and clinical context, significantly enriching the quality of this work.

We are also grateful to the Computer Science Department at Ashoka University and to the Koita Center for Digital Health (KCDH-A) for helping us with the computational resources necessary for the execution of this project, when required. The opportunity to leverage these resources greatly contributed to the successful completion of the work.

The successful implementation of this project would not have been possible without the use of essential Python libraries, including scikit-learn, numpy, pandas, Streamlit, Plotly, and Matplotlib, which facilitated efficient data analysis, machine learning, and visualization.

Lastly, we would like to express our heartfelt appreciation to our families and friends for their unwavering support, encouragement, and understanding throughout the course of this research. Their belief in us kept us motivated and focused, and we are deeply grateful for their constant and unconditional presence.

Contents

1 Introduction

The integration of advanced computational techniques with healthcare data has emerged as a transformative approach to improving clinical decision-making and patient outcomes. Despite the vast potential of electronic health records (EHR), challenges in data preprocessing, analysis, and interpretability have limited their full utility.

This project leverages the MIMIC-IV database, a comprehensive repository of anonymized EHRs, to explore the predictive potential of blood parameters in critical care settings, focusing specifically on sepsis patients.

Sepsis remains a critical global health concern, characterized by high mortality rates and significant healthcare resource utilization. By identifying key blood parameters associated with outcomes such as in-hospital mortality and ICU length of stay (LOS), this study aims to bridge the gap between raw data and actionable insights. The primary challenge lies in navigating the complex structure of the MIMIC-IV database, addressing issues of data sparsity, and ensuring biological relevance in the feature selection process.

The objectives of this study are multi-faceted. First, it aims to perform exploratory data analysis (EDA) to identify and preprocess relevant subsets of data, ensuring the selection of biologically significant features. Second, it seeks to develop machine learning models for predicting ICU outcomes with a focus on interpretability and clinical utility. Third, the project incorporates user-friendly visualization tools to present key insights and model predictions, enhancing accessibility for healthcare professionals.

This interdisciplinary endeavor combines expertise in data science, machine learning, and healthcare to lay a robust foundation for the thesis that will follow this project. By focusing on sepsis patients and employing innovative modeling and visualization techniques, this study not only prepares us with essential insights and methodologies but also provides a springboard for exploring more advanced predictive analytics in the upcoming thesis work.

2 Background and Motivation

2.1 Background

2.1.1 Application of Computer Science Techniques in Healthcare

The integration of computer science and healthcare has revolutionized the way patient data is analyzed and utilized. From predictive analytics to real-time monitoring systems, advanced computational techniques have enabled significant improvements in patient care and resource management. Machine learning models, in particular, have shown promise in predicting outcomes, such as disease progression and treatment efficacy, leading to more personalized and effective care pathways.

Considering our literature review, as detailed in the below section, it is clear that the study of health data is greatly benefited by statistical and computer science based techniques.

The large volume of stored health data can contribute directly to improvement of individual lives, as the accessibility of this data ensures backing for procedures, tests, and successful techniques used to cure a certain diagnosis.

2.1.2 Interdisciplinary Nature of the Project

The project extends across not just computer science, but our interests in data engineering and healthcare as well. Not only is the interdisciplinary nature a factor of our interests, but it is valuable for people belonging to both the fields of computer science and medicine.

Clinicians can greatly benefit with easier access to such complex data, given the exploration and even access requires technical knowledge of data science techniques. Being able to present this data in a visual and easy to understand manner will help healthcare professionals take full advantage of the large amounts of unused data that is being collected.

2.1.3 Patient Sub-Group Analysis

Patient sub-group analysis has become an essential focus in healthcare research. By identifying and analyzing specific subsets of patients, such as those with sepsis, researchers can uncover patterns and correlations that may not be apparent in broader datasets. This targeted approach not only enhances the understanding of disease mechanisms but also facilitates the development of tailored interventions and predictive models.

2.2 Motivation

2.2.1 Understanding EHR Data (MIMIC-IV)

The MIMIC-IV database represents a significant advancement in the availability of high-quality electronic health records (EHR) for research. Covering over a decade of ICU admissions, MIMIC-IV provides a comprehensive resource for studying critical care outcomes. Its modular design and inclusion of diverse data types, such as demographics, vital signs, and laboratory results, make it an invaluable tool for machine learning and data-driven healthcare research.

For our thesis specifically, we were interested in the database and we felt that this project would serve as a deep dive into the database. Professor Dey also suggested this exploration for us to get familiar as pre-work for our capstone thesis.

2.2.2 Previous Work with ARDS

One of us has also worked extensively with ARDS demographic related exploration earlier, due to interest in sub-group analysis. This was a further motivation for this project as this context caused an interest in exploring the MIMIC-IV database.

2.2.3 Passion for Healthcare Research

A large part of our motivation to get involved in this project was due to our passion for healthcare research. We wanted to work on a project that could, if worked on in the long term, have a real life impact on healthcare.

We realised that our knowledge of computer sciene and statistical programming could help us create tools and delve into deeper analysis of healthcare data, that could eventually deeply benefit clinicians and make insights from such data more easily accessible.

2.2.4 Interest in Predictive Modelling

Early prevention through prediction is something that greatly interests us. We wanted to be able to use the large amount of data that MIMIC-IV provides in order to try and predict certain targets that, when predicted, could be rectified in advance.

We knew that targets such as length of stay and mortality, if predicted in advance with a decent accuracy, can be used to trace back and compare patients with similar trajectories to suggest interventions that can be taken for the patient's well-being. This itself was our biggest motivation to start this project.

3 Literature Survey

The rapid advancements in data-driven technologies and the increasing availability of large-scale healthcare datasets have revolutionized clinical research and patient care. The Medical Information Mart for Intensive Care (MIMIC) database series, including the recent MIMIC-IV, has emerged as a cornerstone for research in critical care, offering rich, anonymized electronic health records (EHR) from intensive care units (ICUs). These datasets provide a unique opportunity to explore predictive modeling, patient trajectory analysis, and the development of decision-support tools for clinicians.

In this section, we review key studies that have utilized the MIMIC datasets to address various challenges in healthcare research. These studies span a range of applications, including mortality prediction, length-of-stay estimation, and sepsis diagnosis. We emphasize the transition from MIMIC-III to MIMIC-IV and the implications of this evolution for data preprocessing, model development, and fairness evaluation. The survey also highlights gaps in the literature, such as the need for improved model interpretability, handling of multi-modal data, and the development of user-centric visualization tools for clinical applications.

This review sets the stage for understanding the contributions and limitations of existing approaches and aligns them with the objectives of our study, which aims to analyze sepsis patients' blood parameters, develop predictive models for hospital outcomes, and create accessible visualization interfaces.

3.1 Previous Literature

3.1.1 Strategies of Predictive Schemes and Clinical Diagnosis for Prognosis Using MIMIC-III Khope and Elias (2023)

Objective

The paper focuses on using the MIMIC-III dataset to analyse patient trajectories, developing predictive models for prognosis. It emphasises on strengths and gaps in current ML/deep learning approaches for such prediction.

Relevance

There are many data challenges that the paper discusses (missing data, skewed data, reliance on ICD-9). It also talks about the challenge of a lack of granularity in the ICD-9 system, requiring higher levels of data pre-processing and feature selection.

Since our project and thesis involves structuring parts of the the dataset for sepsis analysis, we directly address certain challenges that the paper discusses.

3.1.2 Predicting 30-Day Mortality for MIMIC-III Patients with Sepsis-3: A Machine Learning Approach Using XGBoost Hou et al. (2020)

Objective

The study uses MIMIC-III data to compare predictive ML models (Logistic Regression, SAPS-II, and XGBoost) to predict 30-day mortality. The study uses various predictors such as blood pressure, oxygen saturation, lab values for prediction.

Relevance

The objective of this paper is similar to that of one of our research questions. The paper compares ML models used for mortality prediction within 30 days for sepsis. We instead use the **MIMIC-IV dataset** for **hospital** expiry using **blood parameter values**. The paper also discusses the use of XGBoost, which is a model we have tested too, and surprisingly it has not performed as well as some others we used. The paper also discusses how model interpretability and dataset imbalance are significant issues, with a need for more comprehensive validation in real-world settings, which is something we discovered as well.

3.1.3 MIMIC-IV, a Freely Accessible Electronic Health Record Dataset Johnson et al. (2023)

Objective

This paper goes over the transition from MIMIC-III to MIMIC-IV, discussing how the structure of the data has changed. It introduces the table structure of MIMIC-IV and gives a high-level overview of the table connections. Finally, it also discusses patient trajectory graphs.

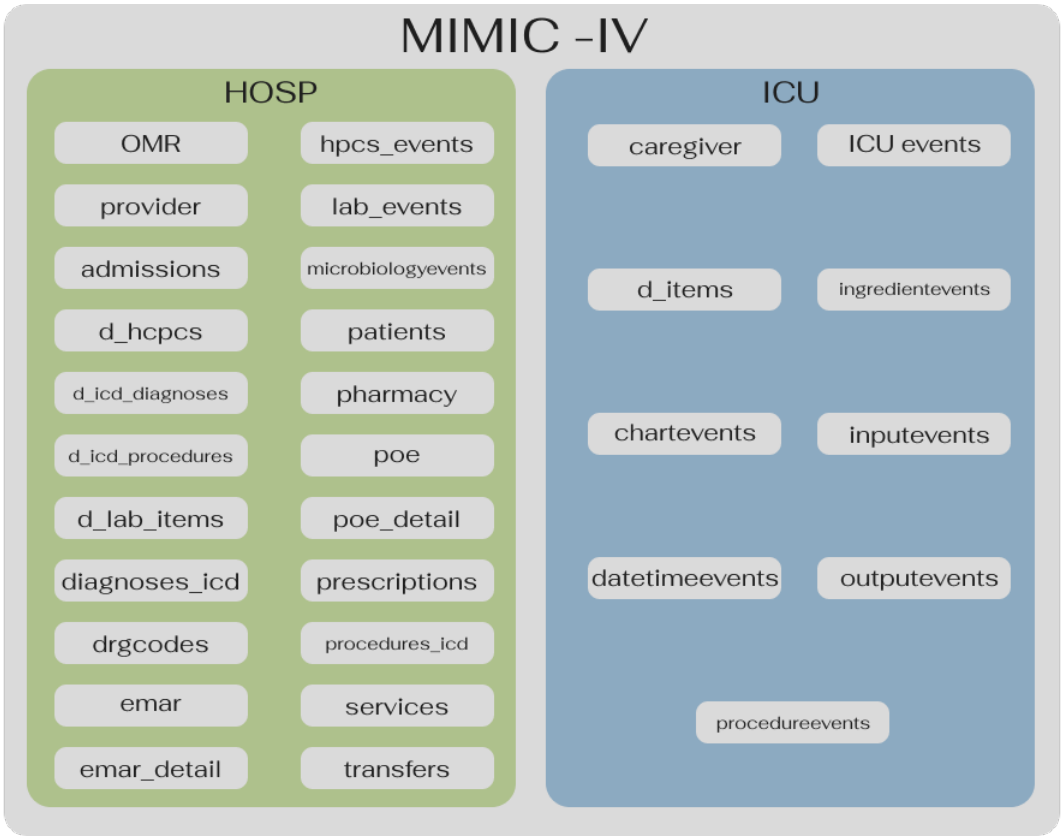
Relevance

The description of MIMIC-IV helped us understand how it is structured, which was helpful for a database of this size. Secondly, the other literature we reviewed used MIMIC-III, and this paper helped connect such research to the new iteration of the dataset. Finally, the paper covered challenges with respect to data sparsity and missing data, which we were introduced to before accessing the data.

3.2 MIMIC-IV Database

MIMIC-IV is a publicly available database of electronic health records (EHR) from Beth Israel Deaconess Medical Center, covering admissions from 2008 to 2019. It includes

data from 383,220 patient admissions to both intensive care units (ICUs) and emergency departments.



3.3 Gap Analysis

3.3.1 Explainability

On exploring the MIMIC-IV dataset, and reading the aforementioned research papers, we found that although the dataset is structured and the information easy to gain access to, it is tough to interpret. This is due to the convuluted structure of the dataset, along with the biological knowledge required to understand it. The issue that arises is the navigation of the tables within the dataset to actually perform analysis requires a fair bit of technical knowledge of tables and data science, which also needs to backed with understanding the biological basis of the dataset.

Due to the large size of the dataset and the complicated and complex nature of the table connections, we decided to perform exploratory data analysis to hone into certain tables that were relevant to our study, aiming to make those convenient to use and explainable.

3.3.2 Visualisation

Despite being officially accessible to people who clear the ethics courses, accessing the MIMIC-IV database requires technical knowledge or some familiarity with querying databases. This serves as a hindrance to many clinicians that lack the technical knowledge to but want to use the database to gain biological insights.

For this reason, it is important to create an easy-to-use visualisation tool to make access to a structured version of the MIMIC-IV database easier. The visualisation tool will also make getting insights from such a large amount of data much more intuitive and simple.

3.4 Research Questions

3.4.1 Structuring MIMIC-IV Data and Exploratory Data Analysis

For the first part of our project, we wanted to **understand the MIMIC-IV data structure**. This involved exploration and understanding how each table relates to the others, common keys, and what information is contained within each table.

After getting familiar with the structure of MIMIC-IV, we then aimed to perform **exploratory data analysis**, to decide on what tables, features and data was important for our study, and what we wanted to explore.

Our end-goal was to do an analysis after finding a certain sub-group of patients based on certain parameters, using machine learning to predict an important target feature.

3.4.2 Predicting Length of Stay on the Basis of Blood Parameters

The next part of our project aims to discuss the comparitive performance of different machine learning models to predict the length of stay of a patient.

This research question formed after exploring the data, and finding that a large subset of the patients in the database were suffering from a form of sepsis. We decided to extract this sepsis data, and find a further subset of features we could use to predict the length of stay of particularly sepsis patients.

We found that there are blood composition and blood behaviour parameters present in the MIMIC-IV database. Since we could see that there seemed to be a degree of correlation of our target variable and certain blood parameters we decided on focusing on these blood parameters to obtain concrete results.

Our research question then became to predict length of stay, with a multi-class, non-numeric approach using machine learning models and comparing the same, similar to one of the papers discussed in the previous literature section.

3.4.3 Predicting In-Hospital Mortality on the Basis of Blood Parameters

Extending from our previous research question, we could also see that in-hospital expiry was a feature that seemed to be greatly affected by blood composition parameters for sepsis patients. We took this as a binary prediction problem and decided on using the same models to try and predict in-hopital mortality of sepsis patients based on blood.

4 Problem Statement and Objectives

4.1 Problem Statement

Hospitals generate a large amount of data in the form of electronic health records (EHRs). However, there is an untapped potential of this data that remains to be reached due to challenges in data pre-processing, analysis, and interpretability. The MIMIC-IV health database, a huge repository of anonymised data can help provide opportunities for predictive modeling and data analysis. Although, it is a complicated task to extract meaningful insights from such a complex database, due to a few challenges.

1. **Complex Data Structure**

   The MIMIC-IV database has a large number of intricately connected tables with varying forms and missingness of data. This makes the navigation and preprocessing a highly technical and resource and time intensive process.

2. **Biological Relevance**

   Currently, effective analysis requires not only data science expertise but also a deep understanding of the biological parameters and clinical significance inherent in the dataset.

3. **Prediction Challenges**

   Developing accurate predictive models for critical outcomes such as in-hospital mortality and length of stay requires addressing issues like data sparsity, imbalanced classes, and the selection of meaningful features.

4. **Missing Interpretability and Explainability**

   Many machine learning models lack interpretability, reducing their utility for clinicians. Moreover, visualizing results in an accessible manner for healthcare professionals remains an unmet need.

4.2 Objectives

The primary objective of this project is to leverage the MIMIC-IV database to develop a comprehensive framework for analyzing sepsis patients' blood parameters and predicting key hospital outcomes such as length of stay and mortality. This involves a structured approach encompassing data preprocessing, exploratory analysis, machine learning, and visualization. The specific objectives are:

4.2.1 Exploratory Data Analysis

1. To find a patient subset and disease/infection subset for analysis, and data to support the analysis.
2. To explore the MIMIC-IV database and identify relevant blood parameters associated with sepsis patients.
3. To generate correlation matrices and statistical summaries to understand relationships between features and clinical outcomes.
4. To preprocess the MIMIC-IV data, including handling missing values, normalizing inconsistent units, and integrating relevant tables.
5. To create structured datasets containing daily blood parameters for sepsis patients.

4.2.2 Machine Learning

1. To develop machine learning models for:
   (a) In-Hospital Mortality/*hospital_expiry_flag* Prediction
   Using blood parameters to predict patient survival during hospitalization as a binary classification problem.
   (b) Length of Stay Prediction
   Estimating patient length of stay as a multi-class classification problem based on blood parameters.
2. To evaluate and compare the performance of different machine learning algorithms and identify the most effective models for each prediction task based on key metric.
3. To visualise the differences between each model for each day.

4.2.3 Data Visualisation

1. To design and implement a simple, user-friendly data visualization tool for plotting key insights from the analysis.
2. To provide clinicians with accessible visual representations of model predictions and correlations, aiding clinical decision-making.

4.2.4 Addressing Model Challenges

1. To ensure that the developed models are interpretable and address potential biases in data representation and predictions.
2. To evaluate the fairness of the models across different demographic groups, ensuring equitable healthcare applications.

5 Scope

This project aims to take a subset of the large MIMIC-IV dataset to provide more concrete analyses on specific data using fixed constraints.

5.1 Aims

1. To identify key blood parameters that influence critical outcomes, such as in-hospital mortality and length of stay, in sepsis patients.
2. To develop predictive models and a data visualization tool to assist clinicians in decision-making, as well as to see the impact of compositional blood parameters on factors like length of stay and mortality in sepsis patients.
3. To provide a framework that bridges the gap between complex data science methods and practical healthcare applications.

5.2 Boundaries of Research

1. **Database:**

   The project is limited to the MIMIC-IV database of EHR data. It focuses on ICU admissions more than hospital admissions. Visualization tools are designed for general accessibility and simplicity, aimed at clinical use cases.

2. **Patient Subset:**

   After exploratory data analysis, we picked the subset of patients suffering from sepsis for our project, as this seemed to have greater variation and less missingness.

3. **Features:**

   We honed into the blood parameters (especially the composition parameters) for our predictive analysis. We used these blood parameters as variables to predict our targets (length of stay and hospital expiry).

4. **Non-inclusion of Hospital Notes:**

   We only used structured data available in the 'hosp' and 'icu' modules, and have not used the 'notes' module as it contains images and would require advanced image parsing and natural language processing.

6 Methodology, Work Done and Challenges

6.1 PhysioNet Training

Since the MIMIC-IV dataset is only given access to for people with certain credentials, we were required to pass the CITI Data or Specimens only Research course with a minimum score of 80%. This course contained 13 modules, relating to data ethics and usage agreements. Taking the course ensured compliance with strict ethical standards, which is a prerequisite for working with such sensitive healthcare data.

6.2 Exploratory Data Analysis

This section outlines the data analysis methodology used to summarize the dataset, identify patterns, and extract meaningful insights to guide the modeling process.

The analysis began with selected portions of the raw MIMIC-IV dataset, which were refined and transformed into a machine-learning-ready format. It provides an overview of the dataset, describes the preprocessing steps undertaken, and highlights key observations about the features and target variables, along with the rationale behind their selection.

6.2.1 Exploration of ICD Diagnoses

The initial dataset for this study was derived from the publicly available MIMIC-IV database, which contains extensive patient information, including demographic details, clinical notes, laboratory test results, and vital signs. At this stage, no additional filtering for specific conditions was performed; instead, the initial focus was on understanding the broader patterns in the dataset to make informed decisions on narrowing the scope.

The exploratory analysis began with the diagnosis records in the MIMIC-IV database, located in `hosp/diagnoses_icd`, where each patient's diagnosis was labeled using ICD-9 and ICD-10 codes.

To better understand the distribution of diagnoses, we generated frequency-based distribution graphs of these ICD codes. The codes were grouped into three frequency buckets: 3,000–5,000 occurrences, 5,000–10,000 occurrences, and 10,000–15,000 occurrences. These groupings allowed us to observe trends and identify conditions of interest across varying levels of prevalence.

The diagnosis distribution graph for the first frequency bucket (3,000–5,000 occurrences) is shown below:

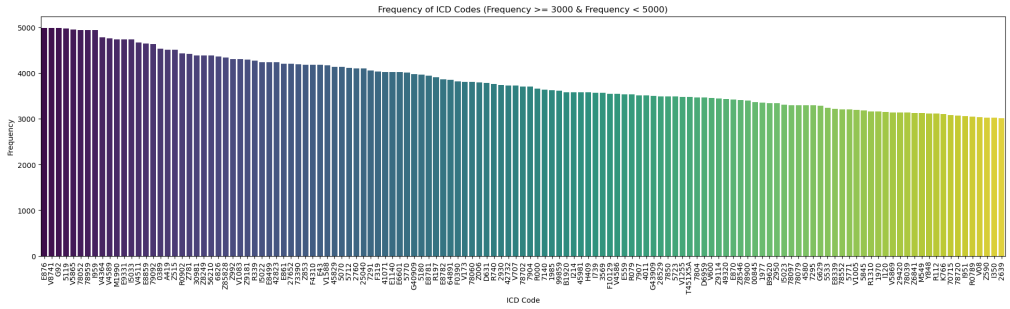
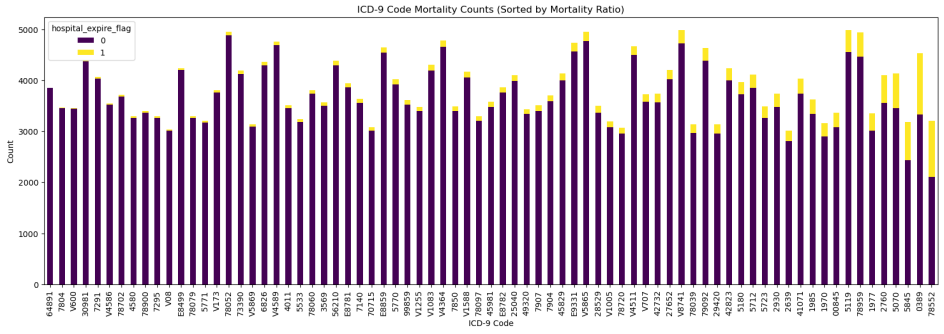
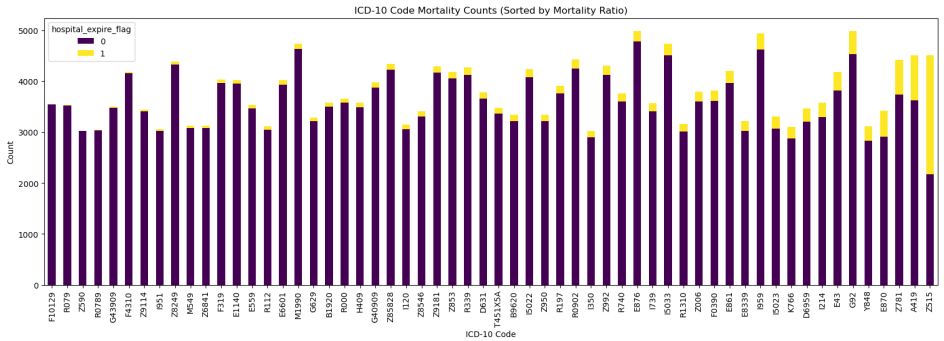


Figure 1: Diagnosis Distribution Graph for ICD Codes (Frequency: 3,000–5,000)

Following the initial analysis, we extracted the mortality flag[1] for each patient within the defined frequency buckets. Using this information, we calculated the mortality ratio for each ICD-9 and ICD-10 code within each bucket, and then created distribution graphs of the mortality ratios, stratified by ICD code, for each frequency bucket.



(a) ICD-9 Mortality Ratio Distribution



(b) ICD-10 Mortality Ratio Distribution

Figure 2: Mortality Ratio Distribution for ICD Codes (Frequency: 3,000–5,000)

[1]This flag, represented by the variable *hospital_expire_flag* from the `hosp/admissions` table, indicates whether the patient had died (1) or survived (0).

The mortality ratio distribution graph for the first frequency bucket (3,000–5,000 occurrences) is shown above. This allowed us to identify conditions with notably high or low mortality rates, offering futher context for selecting ICD codes for further analysis. The graphs illustrating the diagnosis distributions for the other frequency buckets (5,000–10,000 and 10,000–15,000 occurrences) are provided in the Appendix (see Figures 25a, 25b, 26b, 26a, 26d and 26c).

Based on these visualizations, previous work done and recommendation of our advisors, we selected these specific ICD codes for further study:

Diagnosis	ICD-9 Codes	ICD-10 Codes
Sepsis	78552, 0380, 0381, 0382, 0383, 0384, 0388, 0389, 99592	A419
Acidosis	2762	E872
Kidney Failure	5845, 5847, 5848, 5849	-
ARDS	51881, 5185	J80x

6.2.2 Demographic Analysis of Patients with Selected Diagnoses

After identifying diagnoses of interest through the ICD distribution analysis, the next step was to explore the demographic characteristics of patients associated with these diagnoses. This analysis focused on key demographic features, including age, gender, and race, to gain deeper insights into the population affected by these conditions.

The following figures illustrate the demographic distributions specifically for the sepsis cohort:

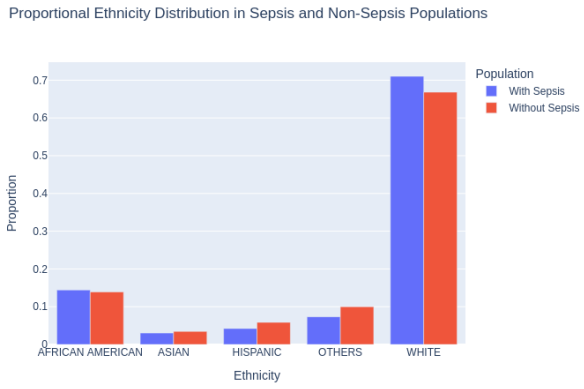


Figure 3: Race Distribution for the Sepsis Cohort

Figure 4: Gender Distribution for the Sepsis Cohort

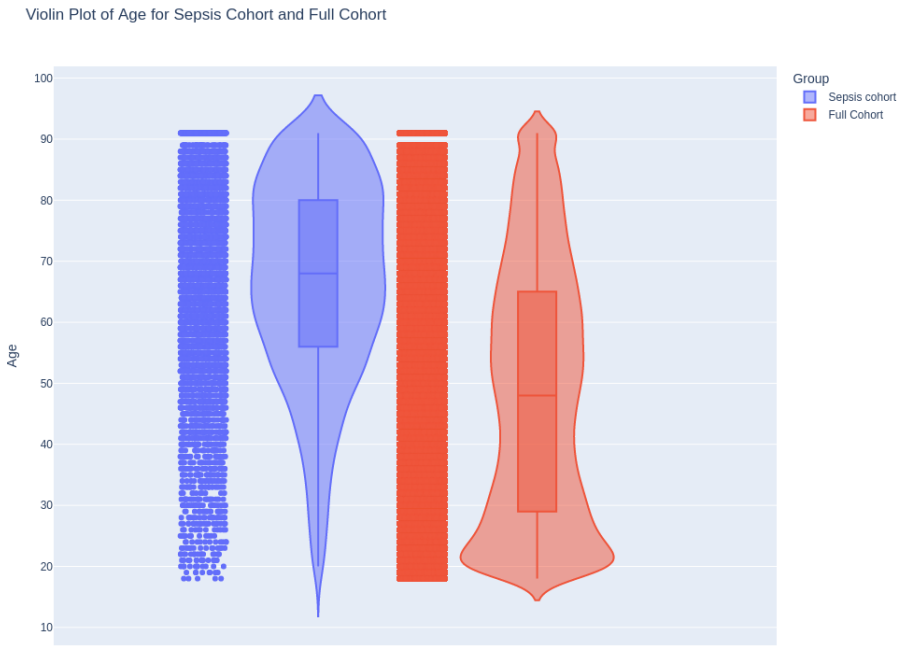


Figure 5: Age Distribution for the Sepsis Cohort

The graphs illustrating the demographic distributions for remaining cohorts (ARDS, Acidosis and Kidney Failure) are provided in the Appendix (see Figures 27, 28, 29, 30, 31, 32, 33, 34, and 35).

This step was undertaken to better understand the composition of the patient population, identify any demographic trends, and evaluate whether the selected diagnoses disproportionately affect specific groups

6.2.3 Exploration of Target Variables

Following the initial exploration of the diagnosis data, the focus was narrowed to two specific cohorts: patients diagnosed with **sepsis** and **acidosis**. The number of patients in these cohorts across all admissions was **7,087** for sepsis and **8,513** for acidosis.

This decision was based on their clinical significance in ICU settings and their relevance to patient outcomes, as highlighted in previous studies. Both sepsis and acidosis are critical factors influencing ICU admissions, and we aimed to explore the relationship between these conditions and the outcomes of **Length of Stay (LOS)** and **Mortality**.

To classify patients, we defined two categories for ICU stay duration:

- Long Stay: Any ICU stay lasting longer than 7 days.
- Short Stay: Any ICU stay lasting 7 days or fewer[2].

We then categorized patients into four distinct groups based on their LOS and survival status:

- `longstay_dead`: Patients with extended ICU stays who did not survive.
- `longstay_alive`: Patients with extended ICU stays who survived.
- `shortstay_dead`: Patients with shorter ICU stays who did not survive.
- `shortstay_alive`: Patients with shorter ICU stays who survived.

Pie charts were created to visualize the distribution of patients across these four categories, helping us understand how the cohort was distributed based on both ICU stay duration and survival outcomes.

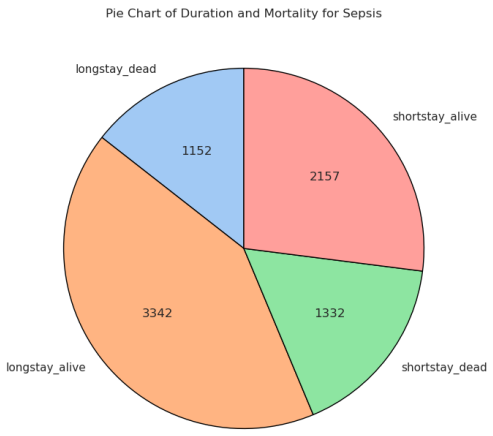


Figure 6: Distribution of Duration and Mortality for the Sepsis Cohort

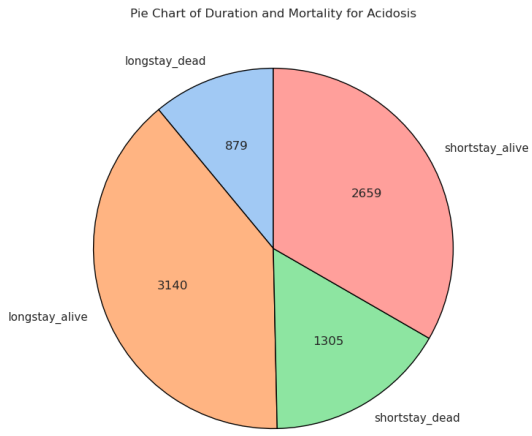


Figure 7: Distribution of Duration and Mortality for the Acidosis Cohort

Following the exploration of ICD diagnoses, we analyzed the distribution of ICU Length of Stay (LOS) specifically for patients diagnosed with sepsis to further identify any

[2]This threshold was chosen based on the recommendation from Professor Lipika, who, in her previous studies, found this division to effectively differentiate between patient cohorts with significantly different prognostic characteristics.

meaningful patterns and validate the feasibility of using LOS as a target variable for prediction.

The distribution of ICU LOS for sepsis patients is visualized below:

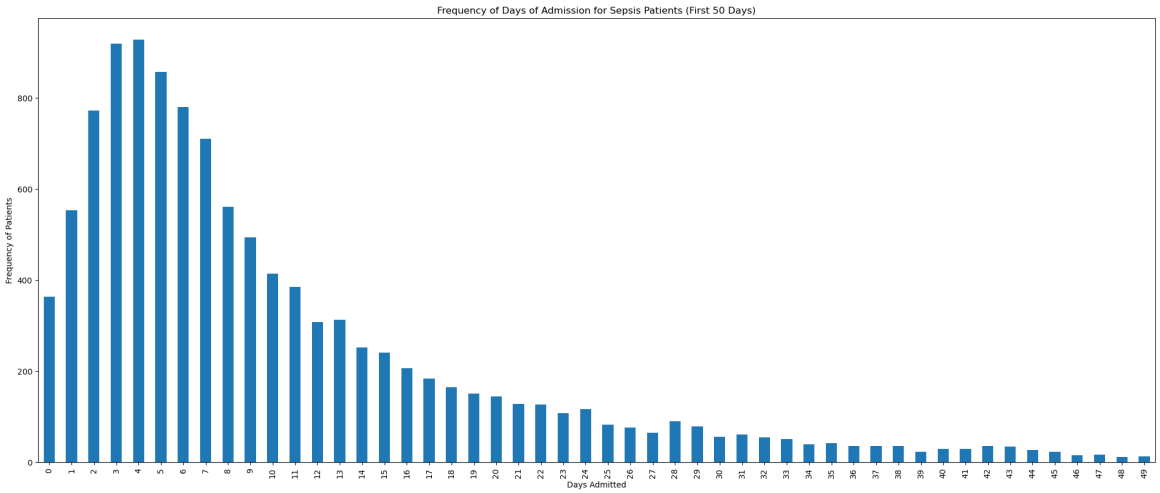


Figure 8: Admission Day Frequency for Sepsis patients

This analysis was complemented by insights from prior work conducted by Professor Lipika and the clinical importance of sepsis to critical care settings. These factors helped narrow the scope of the study to focus **exclusively** on sepsis. By doing so, we ensured that the target variables selected for prediction—LOS and mortality—were both clinically significant and well-represented within the dataset.

6.2.4 Exploration of Patient Procedures

As the next step in our analysis, we focused on consolidating the procedures performed on patients from the cohorts defined in the previous step, abbreviated as `LD`, `LA`, `SD`, and `SD` respectively.

The rationale for examining sepsis patient procedures was to identify patterns or trends that could provide insights into the medical interventions linked to these categories. However, due to the vast number of procedures recorded, plotting all of them was impractical. Instead, we randomly selected five patients from each cohort and analyzed the frequency of procedures performed over three time-frames: 5 days, 10 days, and 20 days.

Below is an example graph showing the number of times `Procedure 221217` (Ultrasound) was performed for sepsis patients in these categories over 20 days:

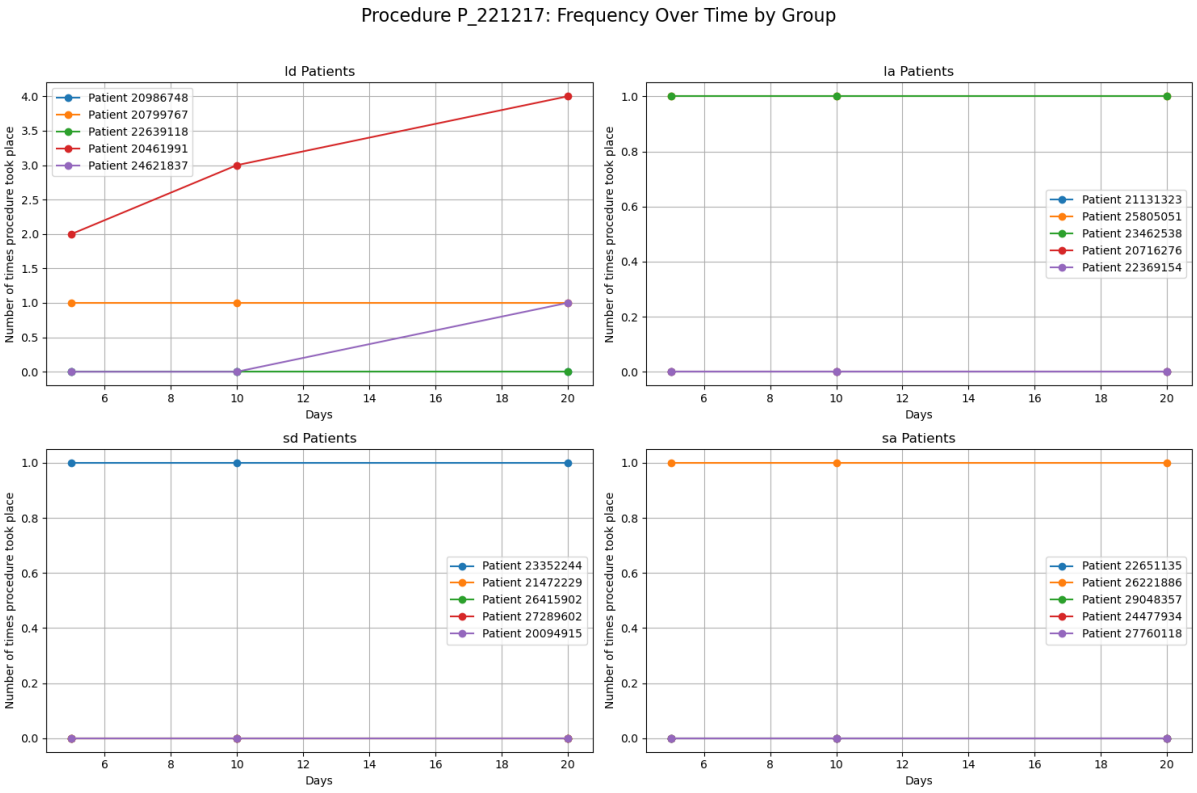


Figure 9: Frequency of Procedure 221217 (Ultrasound) Across Cohorts Over 20 Days

In addition to analyzing ICD diagnoses, the distribution of medical procedures performed on the patients from Sepsis cohort was also studied, aimed to identify the frequency and patterns of specific procedures to further inform the selection of features for the study.

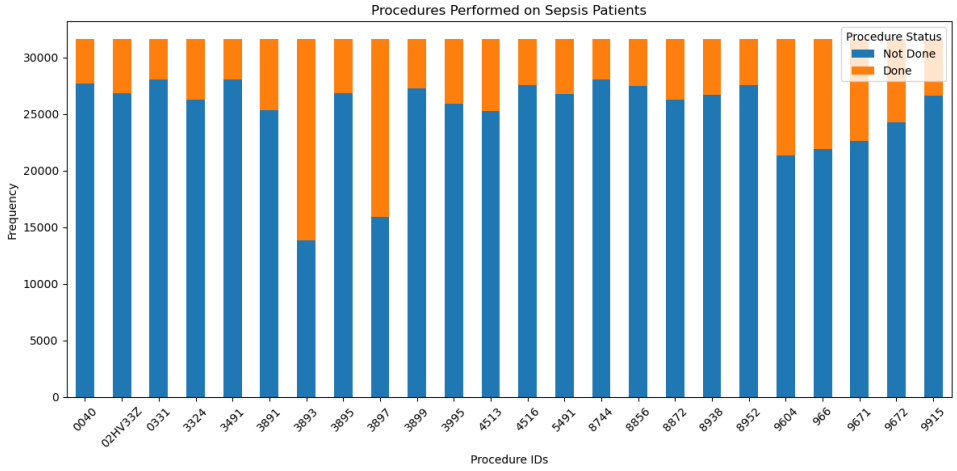


Figure 10: Frequency of Procedures done on Sepsis Cohort

Unfortunately, this analysis did not reveal any correlation or actionable insights with any other parameters. As a result, we decided to shift our focus to laboratory results, specifically those related to **blood parameters** and **bilirubin levels**, which are more directly tied to sepsis-related outcomes.

6.2.5 Feature Extraction for Blood Parameters

For this part of the analysis, we focused on parameters that capture blood behavior and bilirubin levels, as these are critical in understanding the physiological state of sepsis patients. The selected parameters are as follows:

Feature	Parameters
Blood Behaviour	ABPd, ABPm, ABPs Blood Temp CCO (C). Manual BPd L, Manual BPd R, Manual BPs L, Manual BPs R NBPd, NBPm, NBPs
Bilirubin Levels	Bilirubin ApacheIV, Direct Bilirubin, Total Bilirubin

The analysis began by collecting data for four specific days based on the quartiles of the days of admission for sepsis patients: Day 1, Day 3, Day 6, and Day 7. Data for the selected features and their corresponding parameters were then compiled into a structured dataframe. The dataframe included the recorded values of the parameters for each day of observation. Additionally, a new *Status* feature was introduced, categorizing each patient's status for that day as either admitted, released, deceased, or without a recorded status. This feature allowed us to observe the patterns of readmission and track how these parameters changed over time.

To analyze blood behavior and other physiological parameters, we developed a Python function, `take_blood`, which processes data from MIMIC-IV to extract day-wise values for selected blood parameters. This function performs the following key tasks:

- Filters blood-related parameters from the dataset based on their abbreviations and units.
- Extracts laboratory test results for patients admitted with specific ICD codes.
- Groups the data by hospital admission ID and observation day, calculating mean values for each parameter.
- Constructs a pivot table to organize parameter values for specific days (Day 1, Day 3, Day 6, Day 7).

- Adds a *Status* feature to categorize each day's record as admitted, discharged, deceased, or missing.

An example of the distribution for one of the parameters, NBPd on Day 1, is shown below:

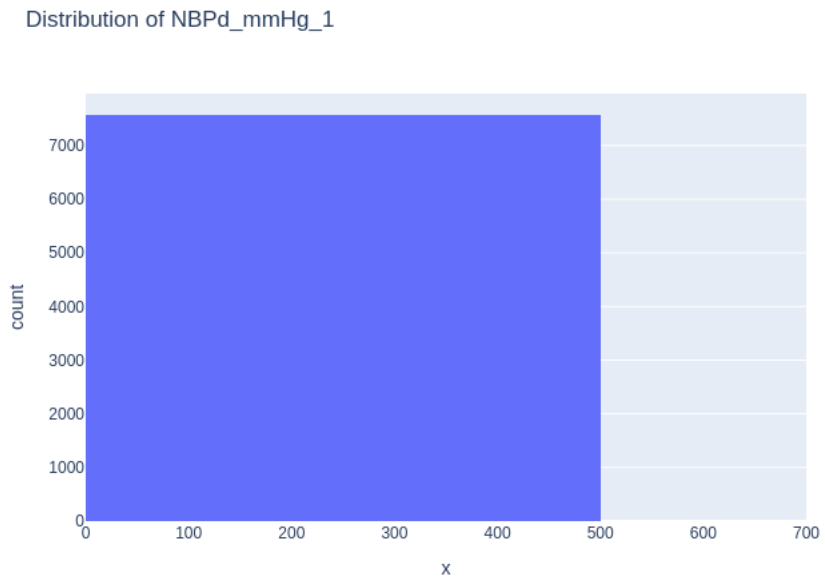


Figure 11: Distribution of NBPd Level for Day 1 in Sepsis Patients

The initial analysis of blood parameters, focused on identifying meaningful behaviors or distributions, did not reveal significant patterns for most individual parameters. Based on these preliminary findings and after consulting with Dr. Garg, we decided to broaden the scope of our analysis to include additional blood composition-specific parameters. This expansion aimed to provide more comprehensive insights into factors potentially influencing ICU Length of Stay and mortality outcomes.

To identify these new parameters, we applied a filter to the `hosp/d_labitems` table, selecting items categorized under *Blood Gas* or with *Blood* as the fluid type. This filtering process also included all parameters for Bilirubin levels, so we consolidated them into a unified list for further analysis.

Some of the selected blood parameters include *CD5 Absolute Count*, *Target Cells*, *pO2*, *Body Fluid*,*Sodium*, *Whole Blood* etc. The complete list of incorporated parameters can be found on this link. The Python function for extracting blood parameters is detailed in Appendix 11.1.3.

6.2.6 Exploration of Blood Parameters

After selecting the relevant parameters, we extracted data for the first seven days of admission for the sepsis cohort. The dataset contained 8,064 records for each day, with 299 features; however, as the days progressed, the amount of missing data for each feature increased, which is a known limitation of the MIMIC-IV database.

This time period was specifically chosen to observe the progression of blood parameters and their potential correlation with patient outcomes early in the ICU stay.

To better understand the relationships between these parameters, we computed a correlation matrix for each day of the cohort's stay. The correlation matrix for Day 1 is shown below:

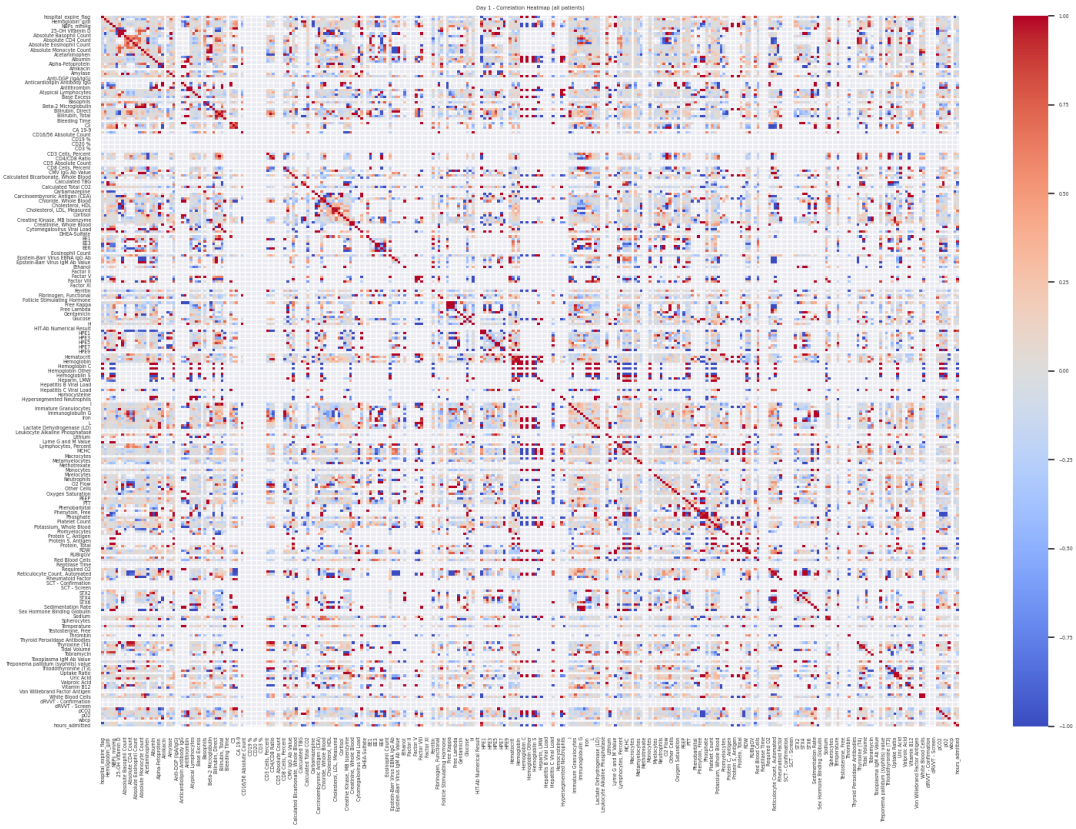


Figure 12: Correlation Matrix for Day 1 of Sepsis Cohort

As observed, many columns in the dataset contained a significant number of missing values, which resulted in a sparse correlation matrix that lacked meaningful information. This sparsity in the matrix could compromise the interpretability of the data.

To address this issue, we introduced a **Minimal Data Retention Ratio (MDRR)** for each column. This ratio quantifies the proportion of non-missing data in a given feature and serves as a constraint to retain only those columns with an adequate amount of filled data, ensuring that analyses and visualizations were based on columns with sufficient data, avoiding the inclusion of features with excessive missing values that could undermine the integrity of the results—especially when building models that rely on robust feature sets[3].

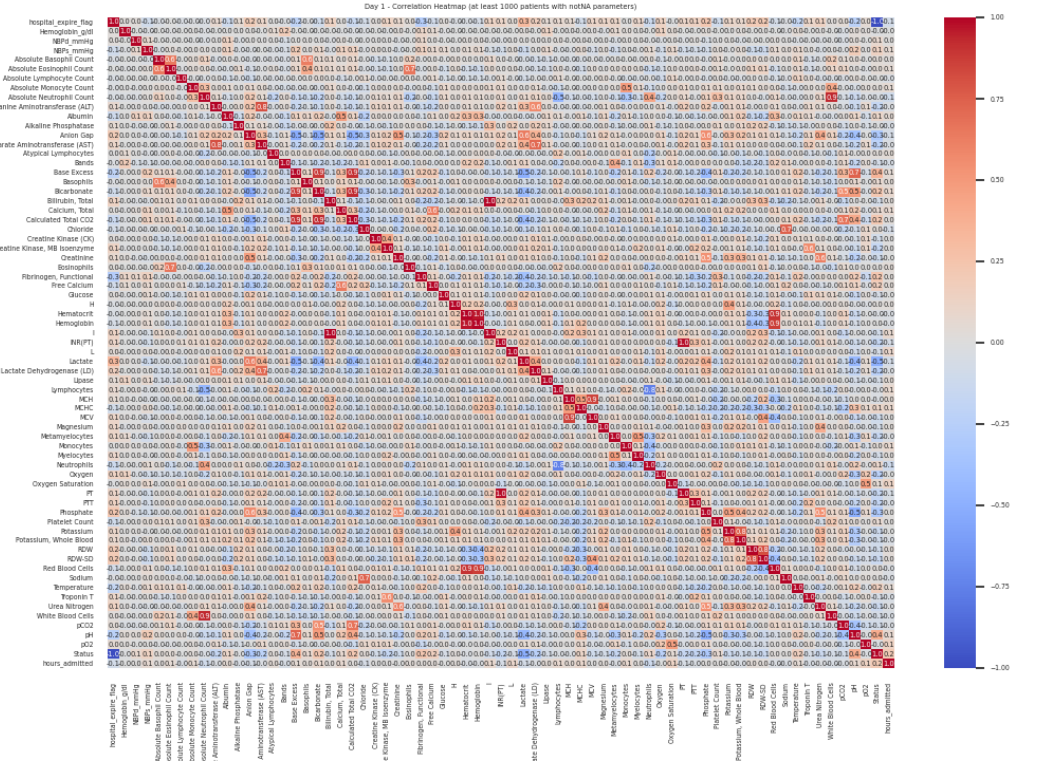


Figure 13: Correlation Matrix for Day 1 after MDRR Constraint (MDRR ≥ 0.125)

After applying the Minimal Data Retention Ratio (MDRR) constraint, we were able to generate correlation matrices with significantly improved visual clarity. One such

[3]For example, consider three columns with the following MDRR values: Column A (MDRR = 0.85), Column B (MDRR = 0.75), and Column C (MDRR = 0.40). Columns A and B, with relatively high MDRRs, were retained, while Column C was discarded due to its substantial proportion of missing values.

constraint applied was an MDRR of 0.125, which required that features have at least 12.5% non-missing values, corresponding to a minimum of 1,000 filled rows per feature. With this constraint, the number of features remaining for Day 1 was 72. The resulting correlation matrix is shown above.

Another constraint applied was an MDRR of 0.5, meaning that features had to have at least 50% non-missing values, or a minimum of 4,000 filled rows per feature. With this stricter constraint, the number of features remaining for Day 1 was 42.

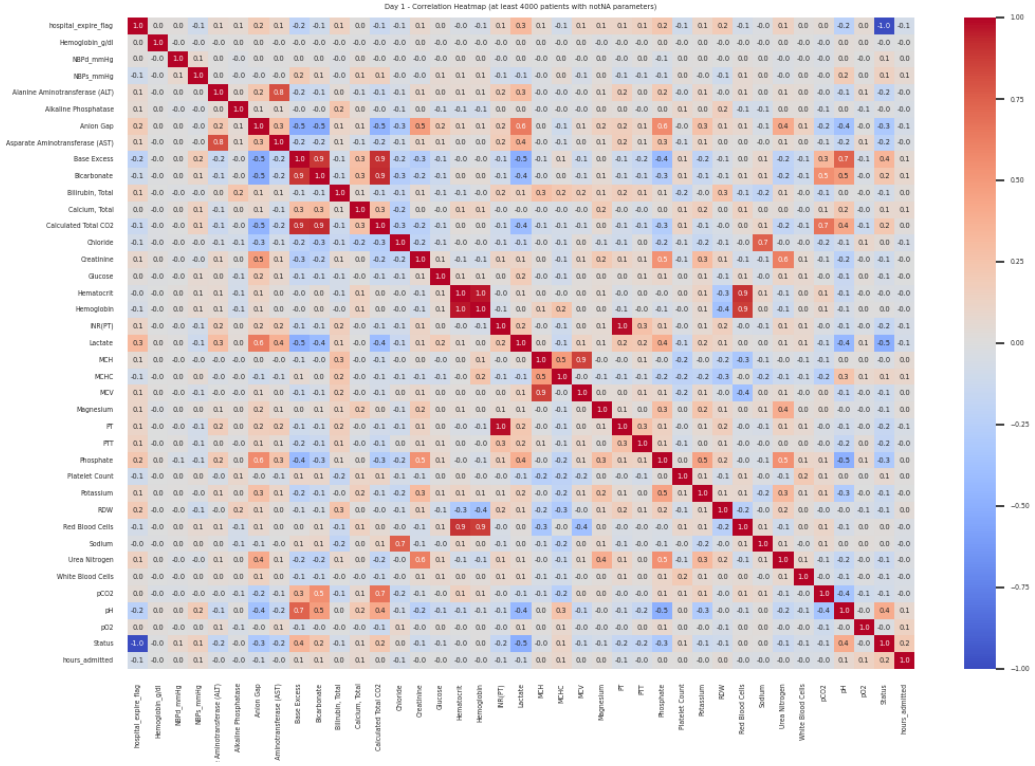


Figure 14: Correlation Matrix for Day 1 after MDRR Constraint (MDRR ≥ 0.5)

As shown in the matrices, the feature sets are now more complete and informative, providing a solid foundation for subsequent machine learning model development and analysis. Additional figures and correlation matrices can be found in the appendix (see figu add).

6.3 Comparative Study of Machine Learning Models

This section outlines the methodology employed to compare various machine learning models aimed at predicting ICU outcomes, specifically Length of Stay (LOS) and hospital

mortality. The process began with a zero-shot learning approach, followed by feature selection using Random Forest to identify the most significant variables. Subsequently, models were trained for both LOS and the hospital expiration flag (HEF). Hyperparameter tuning was then conducted to optimize model performance, which was evaluated using confusion matrices.

Following recommendations, ensemble learning techniques were explored through majority voting, and a thresholding process was applied to select the top features. Additionally, training splits and the Minimal Data Retention Ratio (MDRR) were carefully considered across several models, including Random Forest, XGBoost, LightGBM, and AdaBoost, to ensure optimal performance.

This methodology allowed for a comprehensive comparison of models, ensuring the selection of the best-performing approach for predicting ICU outcomes.

6.3.1 Initial Exploration with Zero-Shot Learning

This section outlines the initial exploration conducted to analyze the dataset and experiment with models to gain preliminary insights. For this exercise, due to the sparsity of data as the number of ICU days increased, we defined the length of stay (LOS) as a binary classification problem: short stays (less than or equal to 3 days) and long stays (greater than 3 days)[4].

For predicting the Length of Stay, we assigned the label 0 for short stays and 1 for long stays. We then performed an equal sampling of both classes to balance the dataset, selected features based on the Minimal Data Retention Rate (MDRR) constraint, with a threshold of 0.7, and imputed the remaining missing values using the mean value for each feature[5].

After preprocessing, we split the data into an 80-20 train-test split and applied a Random Forest classifier to identify the top features contributing to the prediction of LOS. Based on feature importance scores, we selected the top 20 most influential features and ran a Random Forest model using only these features.

[4] This threshold will also be explored further in the thesis to understand its implications on the model's performance.

[5] This approach was chosen to maintain simplicity in our initial exploration; however, we plan to investigate the impact of other imputation strategies, such as median or advanced techniques, in our thesis.

The resulting model achieved an accuracy of 75.5%, with a recall of 0.70 and 0.81, and precision scores of 0.81 and 0.71 for the short and long stay classes respectively. The confusion matrix for the model is shown below:

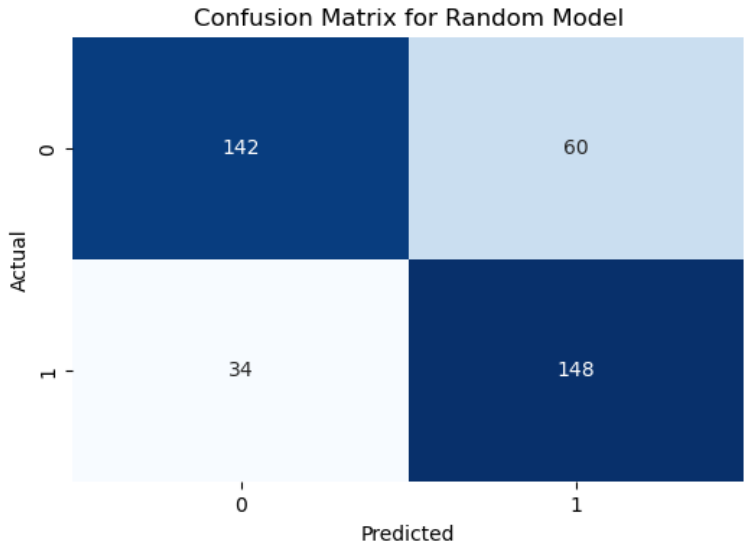


Figure 15: Confusion Matrix for the Random Forest Model on Length of Stay Prediction

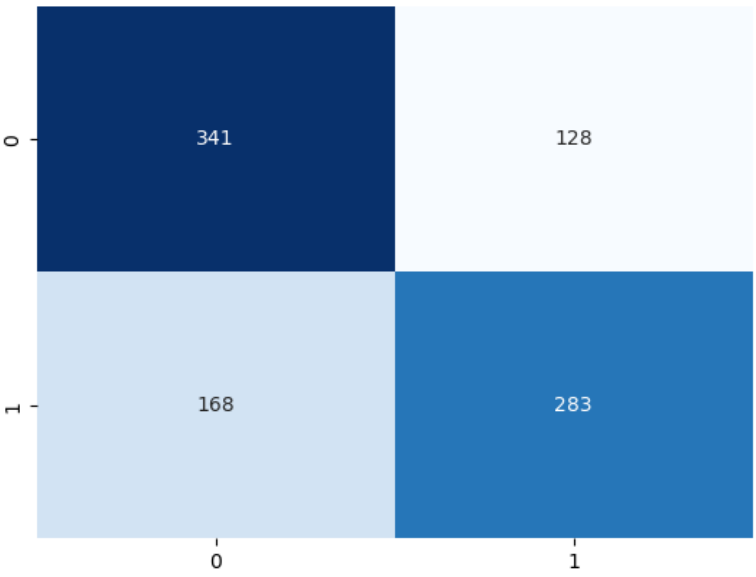


Figure 16: Confusion Matrix for the Random Forest Model on Mortality Prediction

We repeated the same procedure for mortality prediction, using a binary encoding of 0 for alive and 1 for deceased. The initial model achieved an accuracy of 67.83%, with recall scores of 0.73 and 0.63 for the alive and deceased classes, respectively. Precision scores

were 0.67 for the alive class and 0.69 for the deceased class. The confusion matrix for the model's performance is shown above.

Upon reviewing the initial results, we decided to apply hyper-parameter tuning to optimize the model's performance further. For both the prediction tasks, the following parameter grid was used for tuning with 2-fold cross-validation, with results on the optimal parameters for each task:

Hyperparameter	**Candidate Values**	**Best Value (LOS)**	**Best Value (Mortality)**
n Estimators	[50, 100, 150, 200]	100	150
Max Depth	[None, 10, 20, 30]	10	None
Min Samples Split	[2, 5, 10]	2	10
Min Samples Leaf	[1, 2, 4]	4	2
Max Features	['auto', 'sqrt', 'log2']	sqrt	sqrt
Bootstrap	[True, False]	False	True

Table 1: Hyperparameter Candidates and Best Values for Model Tuning

The performance of the models after hyperparameter tuning, is summarized in the tables below for each class for both LOS and mortality prediction.

Metric	**Length of Stay (LOS)**		**Mortality**	
	Short Stay (0)	**Long Stay (1)**	**Alive (0)**	**Deceased (1)**
Accuracy	0.77	0.75	0.67	0.68
Precision	0.83	0.71	0.66	0.68
Recall	0.69	0.85	0.72	0.62
Average Accuracy	**0.765**		**0.67**	

Table 2: Performance Metrics for Length of Stay (LOS) and Mortality Prediction

The confusion matrices for both tasks:

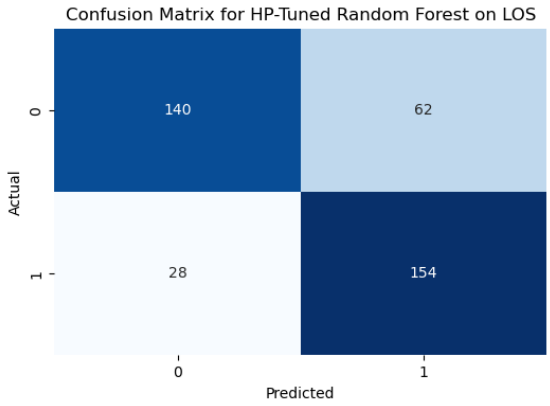


Figure 17: Confusion Matrix for the HP-Tuned Random Forest Model on LOS Prediction

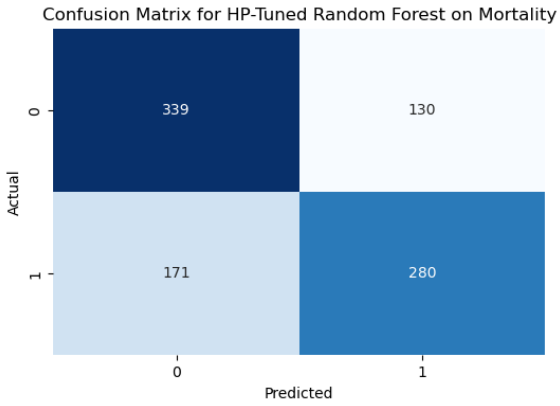


Figure 18: Confusion Matrix for the HP-Tuned Random Forest Model on Mortality Prediction

6.3.2 Comparison of Machine Learning Models for LOS and Mortality Prediction

In this section, we shift focus from our initial exploration with Random Forest to a comparative analysis of different machine learning models for predicting ICU Length of Stay (LOS) and mortality. The models chosen for this study were selected based on their respective advantages and limitations, as outlined below:

1. `Random Forest`
    - Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification.
    - **Advantages:** Robust to overfitting, handles missing data well, and can model non-linear relationships.
    - **Disadvantages:** Computationally expensive for large datasets and less interpretable than simpler models.
2. `Logistic Regression`
    - A simple and widely used model for binary classification that assumes a linear relationship between features and the log-odds of the target variable.
    - **Advantages:** Interpretable coefficients and efficient on small datasets.
    - **Disadvantages:** Limited in capturing non-linear relationships and can struggle with high-dimensional data.
3. `AdaBoost`

- An iterative boosting algorithm that combines weak learners (e.g., decision stumps) to create a strong classifier.
- **Advantages:** Effective in reducing bias and variance, robust to overfitting.
- **Disadvantages:** Sensitive to noisy data and computationally intensive.

4. Gradient Boosting
    - A sequential ensemble method that optimizes a loss function by adding models to correct errors of previous iterations.
    - **Advantages:** High predictive accuracy and flexibility in handling different types of data.
    - **Disadvantages:** Prone to overfitting without proper tuning and computationally expensive.
5. LightGBM (LGBM)
    - A gradient boosting framework that uses tree-based learning algorithms optimized for speed and memory usage.
    - **Advantages:** Faster training compared to traditional gradient boosting, excellent handling of large datasets.
    - **Disadvantages:** Requires careful parameter tuning and can overfit on small datasets.
6. XGBoost
    - An optimized gradient boosting framework designed for efficiency and scalability.
    - **Advantages:** High predictive power, efficient with large datasets, and offers built-in regularization.
    - **Disadvantages:** Can be computationally expensive and complex to tune.

In the end, we also do an Ensemble using Majority Voting, which combines predictions from multiple models by majority vote, improving robustness and reducing bias. Using this, we aim to leverage the strengths of individual models and mitigates their weaknesses. Since it requires diverse models to be effective, we feel this will be a useful addition to this study.

7 Design

To better understand the blood parameter correlations and to make it convenient to analyse this subset of MIMIC-IV, we created a tool for visualisation. This tool is both to make it easier to for accessing the various graphs generated, other than plotting the blood parameter values.

7.1 Machine Learning Results Visualisation

For our machine learning model comparison and confusion matrices, due to there being 4 different parameters. there are graphs generated for all possible combinations of each of these parameters. For our frontend, we used streamlit, a python library to create webapps locally.

Figure 19: Machine Learning Results Visualisation

If "ML Results" is chosen:

1. **Prediction Type Selection:** A user can pick between length of stay and hospital expiry.
2. **Day Selection:** Any day between day 1 and day 6 (the scope of the machine learning in this project) can be selected.

3. **Model Comparison vs. Confusion Matrix:** The user may choose to see the graph depicting the accuracy, F1-Score, Precision and Recall against MDRR (used interchangeably with threshold) or they may choose to see a confusion matrix.
    - If **Model Comparison** is chosen, the user will be prompted to pick a test-train split, and the image will be generated.
    - If **Confusion Matrix** is chosen, the user will be prompted to pick a test-train split and a MDRR value and the confusion matrix image will be displayed.

7.2 Blood Parameter Comparison

Due to our feature subset being blood parameters, we have taken out blood parameter values for each day and grouped them in sheets. To better understand how one blood parameter may relate to another, we have created another part of the tool where a user can pick any two blood parameters and graph them.

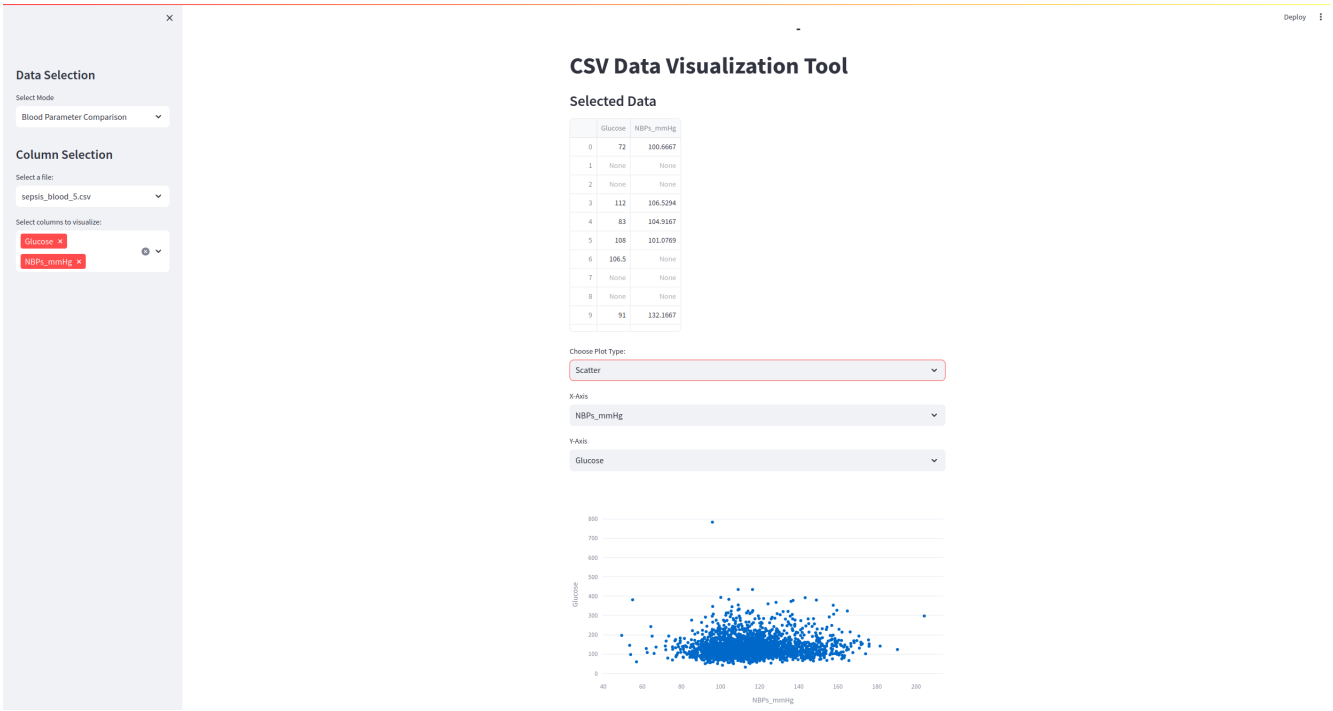


Figure 20: Blood Parameter Comparison Visualisation

If "Blood Parameter Comparison" is chosen:

1. The user will be prompted to enter a day between day 1 and day 7.

2. The user can pick any two blood parameters they would like to visualise. They may pick more than two as well, but the third column will only be shown to the user in the preview CSV widget, and not used in the graphs.
3. The user will then be able to see a preview CSV widget in the web app of the columns they have chosen.
4. They can then choose the plot they would like, choosing between scatter, bar, line and histogram.
5. Lastly, the user can choose which column to plot on which axis, and the visualisation will be presented. The graph will be interactive to increase explainability and interpretability. If histogram is chosen as the plot, then the column that is chosen for the x-axis will be plotted against frequency.

8 Results and Discussions

We decided to evaluate the performance of the models on the dataset for both tasks. To assess model accuracy, we analyzed trends across different days, varying train-test splits, and changing MDRR values.

For predicting mortality and LOS, we present the performance graphs for the models using a 90-10 train-test split, with varying MDRR values, represented as **Threshold** in the graph. The graphs display the model performance on Day 1 and Day 2 of the sepsis cohort, highlighting the differences in trends with respect to the changing MDRR values for the same train-test split.

To evaluate model performance on the dataset, we developed the `run_model` function, which performs the following tasks:

- Balancing the dataset by sampling equal instances from each class.
- Removing columns with excessive missing values and imputes missing data using the mean strategy.
- Using Random Forest to select the top `n` important features based on their importance scores.
- Spiting the data into training and testing sets.
- Training the listed models on the selected features.
- Combining model predictions using a majority vote.
- Computing confusion matrices and classification reports for each model and the ensemble.

The Python function for extracting blood parameters is detailed in Appendix 11.2.

8.1 Mortality Prediction Results

8.1.1 Day 1

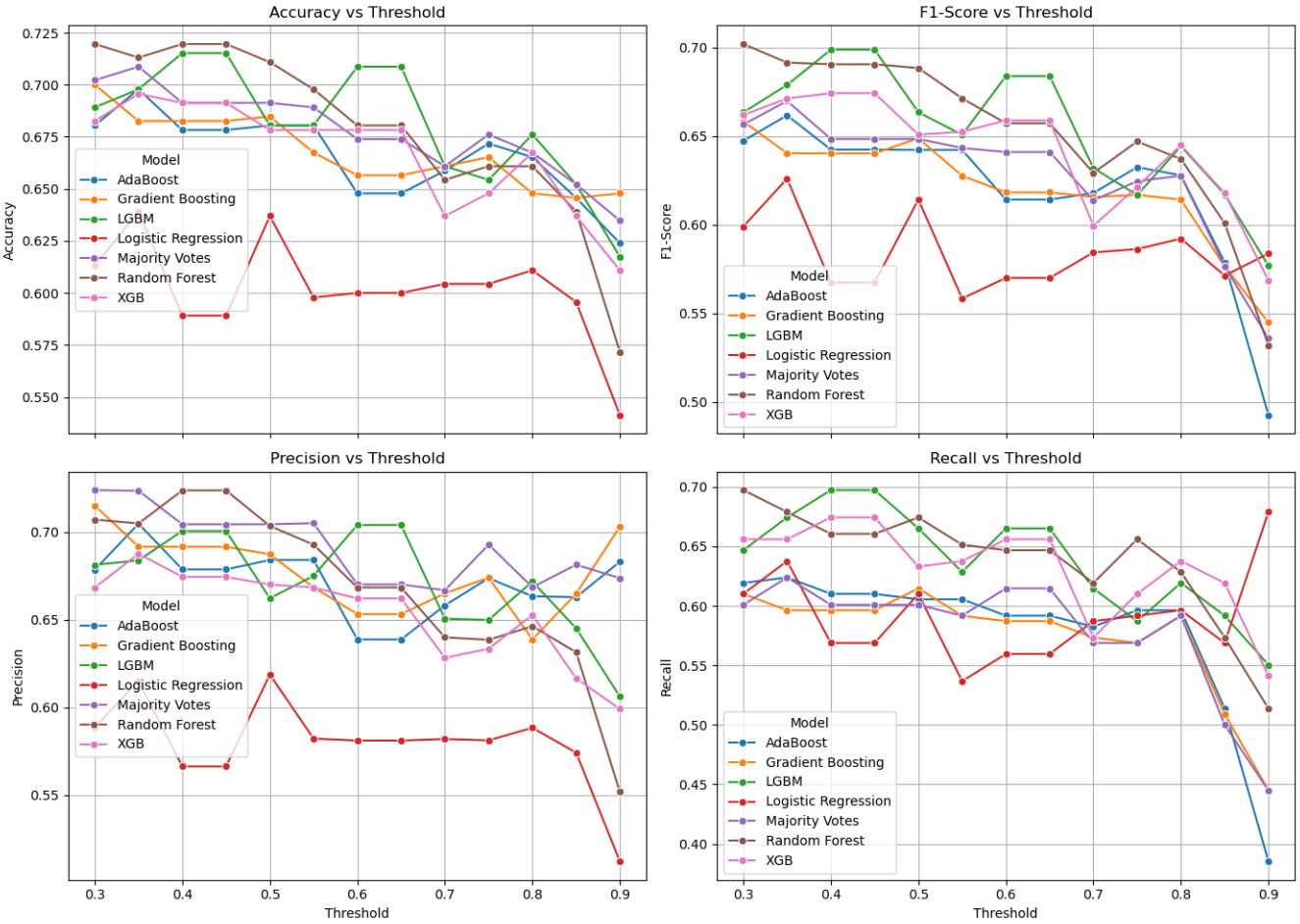


Figure 21: Model performance on Day 1 with varying MDRR values (Threshold) with 90-10 train-test split.

8.1.2 Day 2

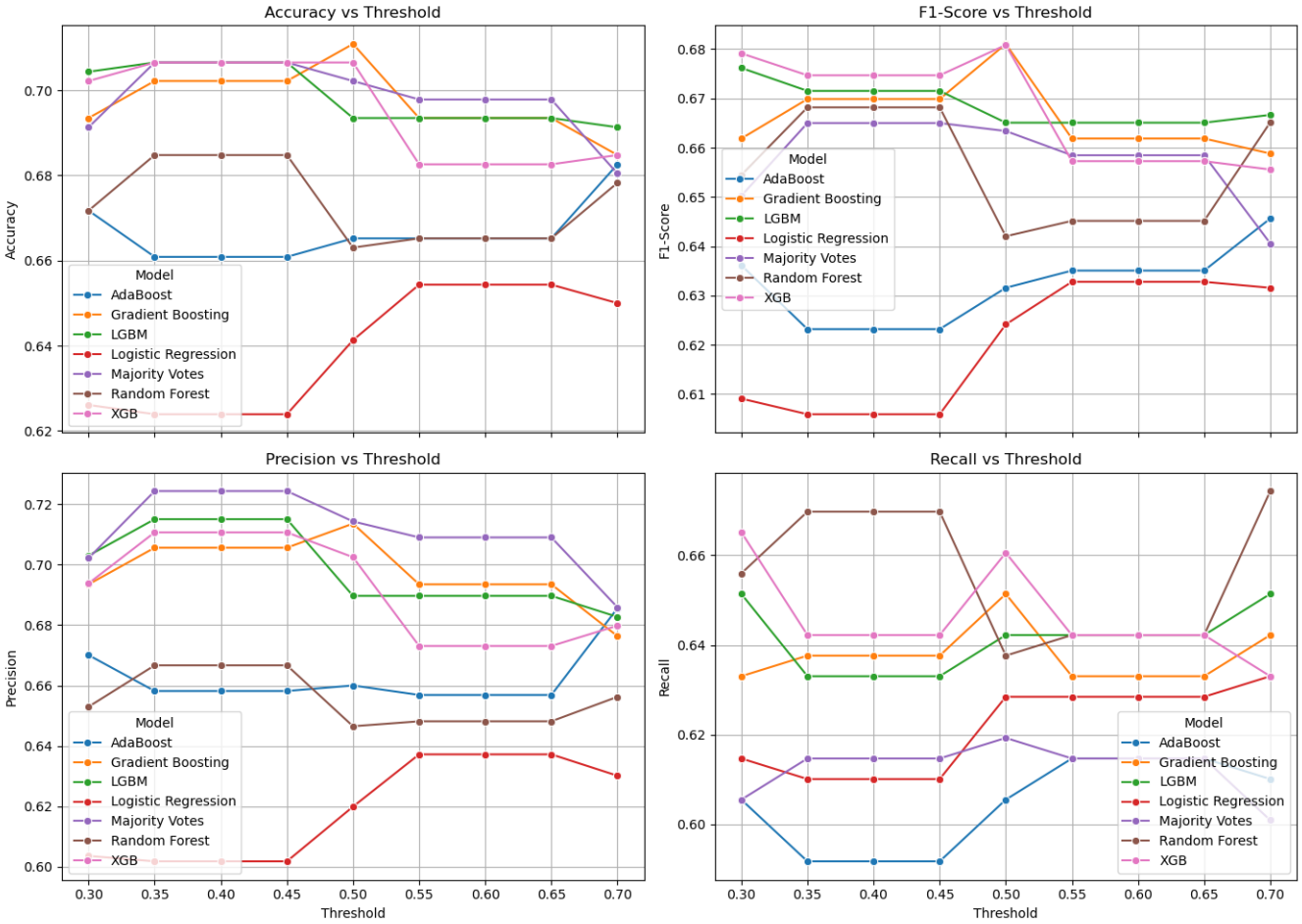


Figure 22: Model performance on Day 2 with varying MDRR values (Threshold) with 90-10 train-test split.

8.2 Length of Stay (LOS) Prediction Results

8.2.1 Day 1

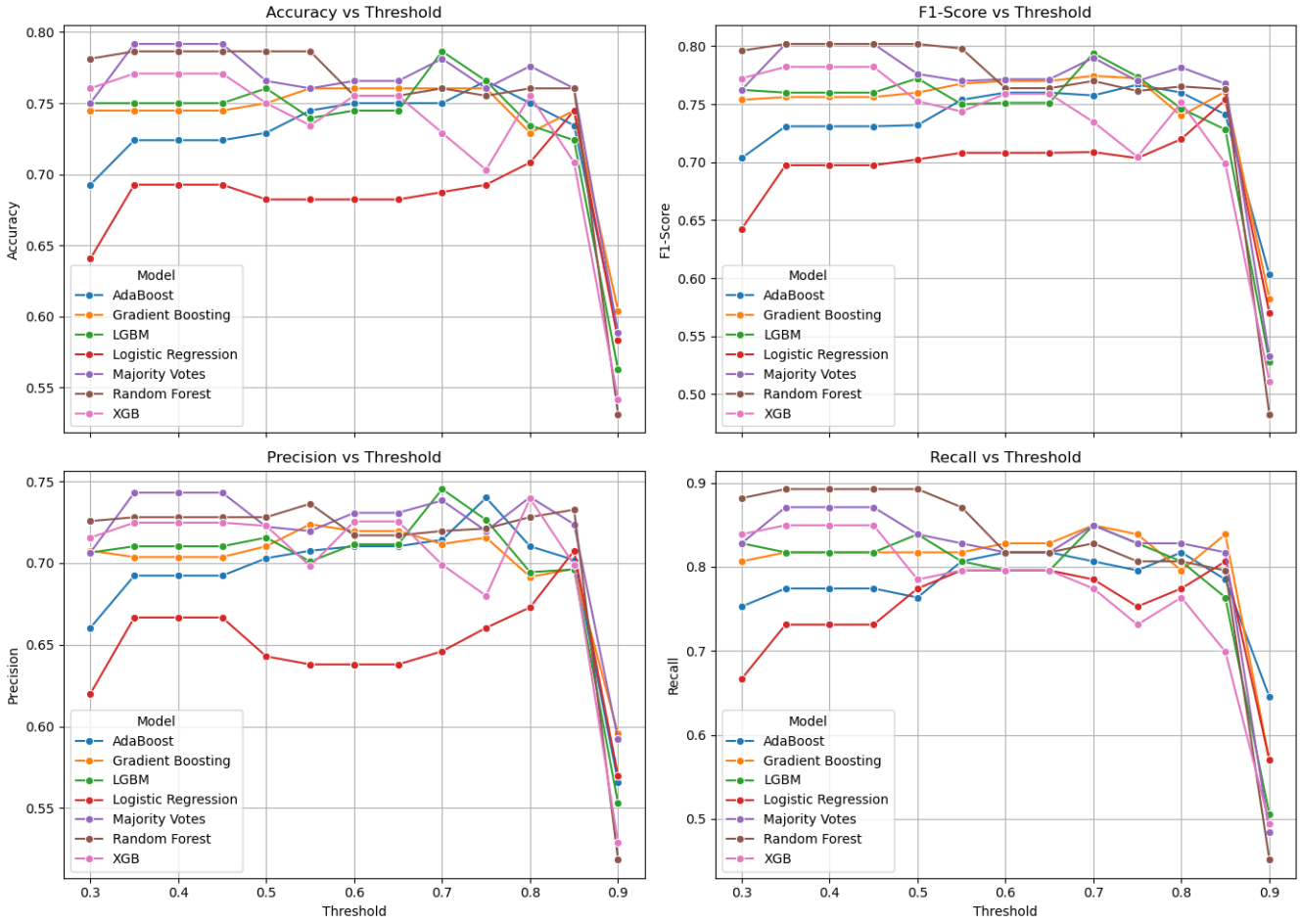


Figure 23: Model performance on Day 1 with varying MDRR values (Threshold) with 90-10 train-test split.

8.2.2 Day 2

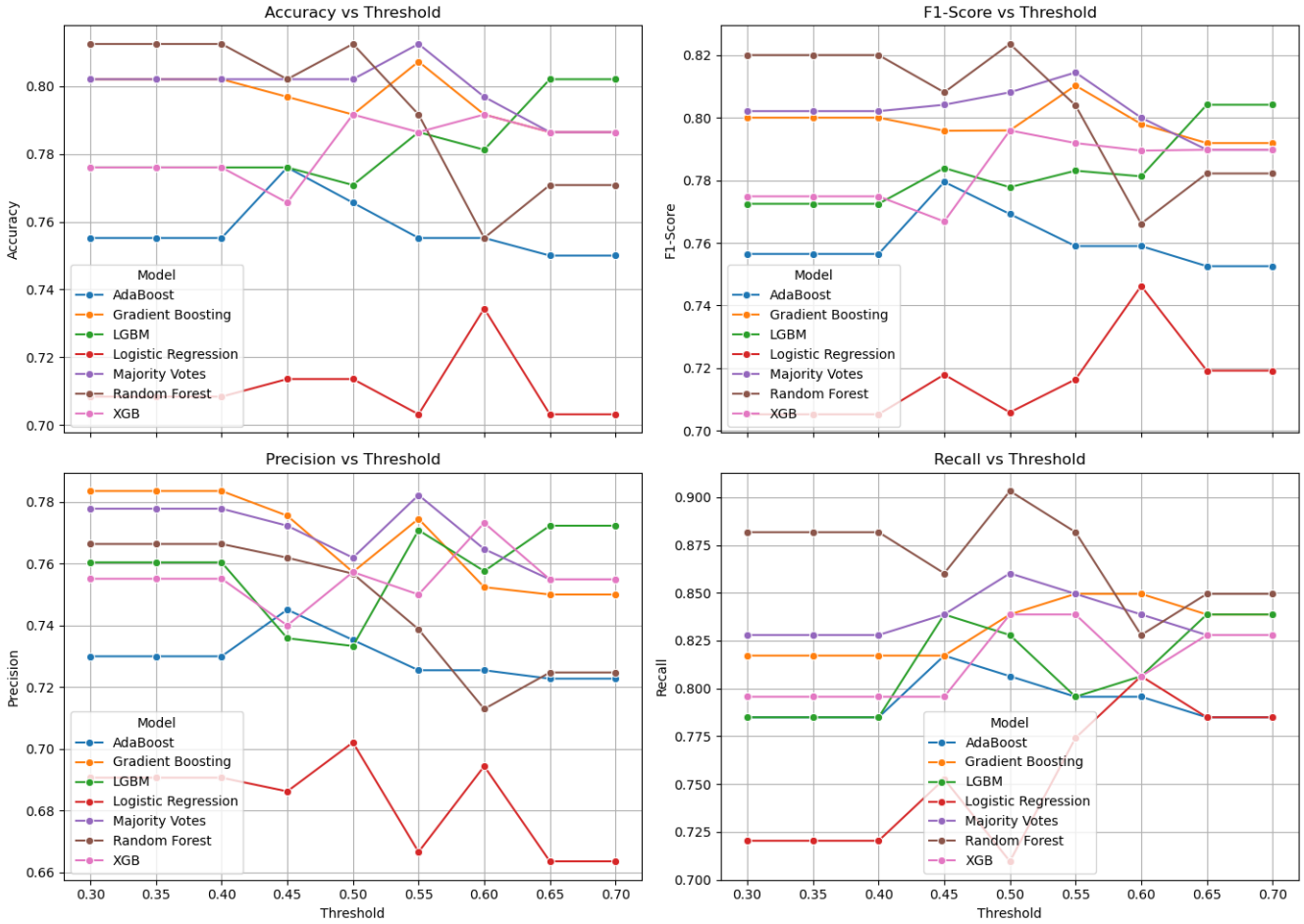


Figure 24: Model performance on Day 2 with varying MDRR values (Threshold) with 90-10 train-test split.

8.3 Insights

The analysis of machine learning model performance on predicting ICU outcomes revealed nuanced and interconnected insights. Higher Minimal Data Retention Ratios (MDRR) generally reduced sparsity and improved reliability, but a **surprising drop in accuracy beyond an MDRR of 0.5 hinted at the loss of critical features under stricter constraints**. This pattern underscores the delicate balance required in feature retention.

Model performance varied significantly between days. **On Day 1, Random Forest (RF) emerged as the strongest performer**, reaching peak accuracy at an MDRR of 0.45. However, its efficacy waned by **Day 2, where Gradient Boosting took the lead**, illustrating

the shifting importance of features over time. The inability to surpass an MDRR of 0.7 on Day 2 further highlighted the limitations imposed by data sparsity.

Interestingly, **Logistic Regression (LR) struggled** across the board, reaffirming its unsuitability for this dataset, while **LightGBM, Random Forest, and ensemble methods like Majority Voting consistently delivered reliable results**. Contrary to expectations and prior literature, **XGBoost did not achieve comparable performance**, emphasizing the importance of dataset-specific validation.

Overall, Day 1 demonstrated better predictive accuracy compared to Day 2, suggesting that **early ICU data holds richer predictive signals**. These findings emphasize the need for flexible, context-aware modeling strategies. By understanding these dynamics, this study sets a robust stage for the thesis, where deeper exploration of ensemble techniques and dynamic feature interactions can enhance the predictive capabilities in critical healthcare applications.

9 Conclusions

This project leveraged the MIMIC-IV database for understanding and predicting critical outcomes in sepsis patients. Through exploratory data analysis, we sought to identify relevant blood parameters and apply preprocessing techniques to better handle data sparsity and complexity. While challenges in feature selection and model tuning emerged, they provided valuable learning opportunities that enriched our approach.

The performance of machine learning models offered insights into the dynamics of ICU data. Random Forest excelled on Day 1, achieving peak accuracy at an MDRR of 0.45, while Gradient Boosting performed better on Day 2, reflecting how feature importance evolves over time. Despite some models like Logistic Regression and XGBoost falling short, others, such as LightGBM and ensemble methods, demonstrated promising consistency. While Day 1 data showed better predictive accuracy compared to Day 2, likely due to richer initial clinical information, our work highlights the importance of adaptability and context-aware strategies in predictive modeling, especially related to electronic health data. The visualization tools developed in this project serve as a step toward making complex analyses more accessible and actionable for healthcare practitioners.

In conclusion, this study lays the groundwork for future research, including the thesis that will build on these findings. We recognize that our work is just one step in a larger journey to advance predictive analytics in critical care. With continued exploration and refinement, we hope to contribute further to this vital field and support efforts to improve patient outcomes through data-driven insights.

10 Extensions and Future Work

10.1 Prediction Improvements

- We want to explore other ML models and ensemble techniques other than majority voting. We also want to implement neural networks to see differences in predictions and improve prediction accuracy and robustness for both length of stay and mortality.
- We also plan on including temporality and dynamic features like time series analysis to capture certain changes across time.

10.2 Addition of Multi-Modal Data

- We wish to expand the scope of the project to further include free-text clinical notes (from the 'notes' module of the MIMIC- IV database) so as to leverage natural language processing and signal processing. Our goal is to enhance predictive power using these techniques.
- In the long term, eventually, we would hope to be able to functionally integrate external datasets (such as AmsterdamUMCdb) to validate and generalise findings beyond the MIMIC-IV cohort. This will also help us analyse the biases and skews present in the MIMIC-IV data.

10.3 Advanced Visualisation Tools

- We wish to develop more sophisticated, intelligent and interactive dashboards to allow clinicians to explore predictions and insights in real-time. This solves the problem of explainability as well as accessibility, and make it convenient to access relevant MIMIC-IV data.
- We hope to enable personalised care pathways by using this data analysis and machine learning techniques to be able to identify sub-groups of patients, suggesting certain types of patients specific pathways.

10.4 Ethics and Fairness

- We aim to further analyze and mitigate biases in the dataset and predictions to ensure equitable treatment across diverse patient demographics.
- Finally, incorporating explainability tools, such as SHAP or LIME, to provide transparency and build trust in machine learning predictions is important.

10.5 Other Diseases - Expansion Beyond Sepsis

- Apply the framework to other critical conditions or patient subgroups, exploring generalizability and scalability.
- Investigate the potential of predictive models in areas like treatment optimization, resource allocation, and **readmission risk prediction**.

10.6 Improved Data Imputation Techniques

- As of now, we have imputed missing data (for machine learning classifiers) using a median fill.
- We plan on using more advanced and 'intelligent' techniques to improve our prediction results, as well as maintain the integrity of the data, estimating the missing values as closely as possible.

10.7 5 Dimensional Data Representation in 3D Graphs

- Currently, we have a large number of graphs due to a lack of a strategy for representing 5 dimensions (accuracy, model_name, MDRR, train-test split, and days) in 3 dimensions, while retaining interpretability.
- We could have either chosen to compromise on interpretation while decreasing the number of graphs, or we could have decided to keep the graphs interpretable but we would need to go over a bulk number of graphs. For now, we picked the second strategy as this was an outlined goal of the paper.
- We plan to look for better ways to represent high dimensional data while retaining all information as well as being able to understand it.

10.7.1 Advanced Ensemble Configurations

- Currently, we are using an ensemble of all models, and are using an unweighted majority voting algorithm for the final predicted value. We want to explore other techniques for ensemble configurations.
- Other than majority voting, based on research we found that model stacking can lead to better results for larger databases.
- We plan on using different models for different features, based on model strengths, and then using a weighted vote (based on both accuracy and type of feature) so as to increase overall prediction accuracy.

10.8 Better Selection of Parameters

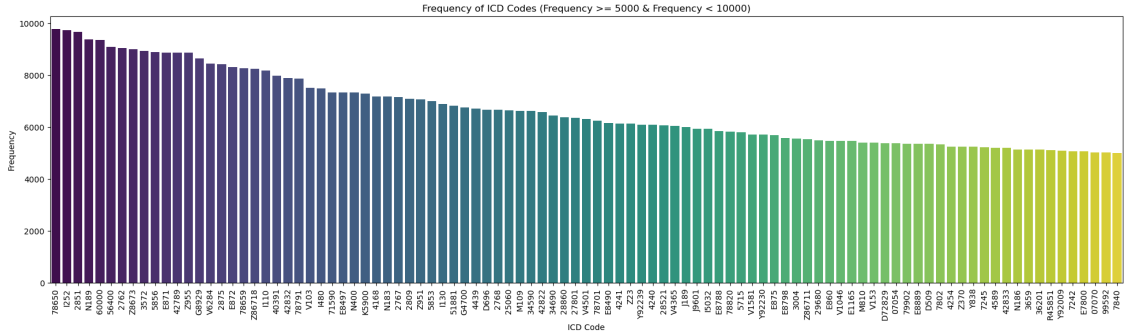
- Currently we have defined certain constraints and variables, such as what amounts to a long stay and a short stay based on certain statistical patterns. We plan on experimenting with these values to compare the changes in prediction and accuracy.
- We would, finally, also want to explore parameters beyond blood composition to see what else we may find that has a high level of correlation for outcomes such as mortality and length of stay.

11 Appendix

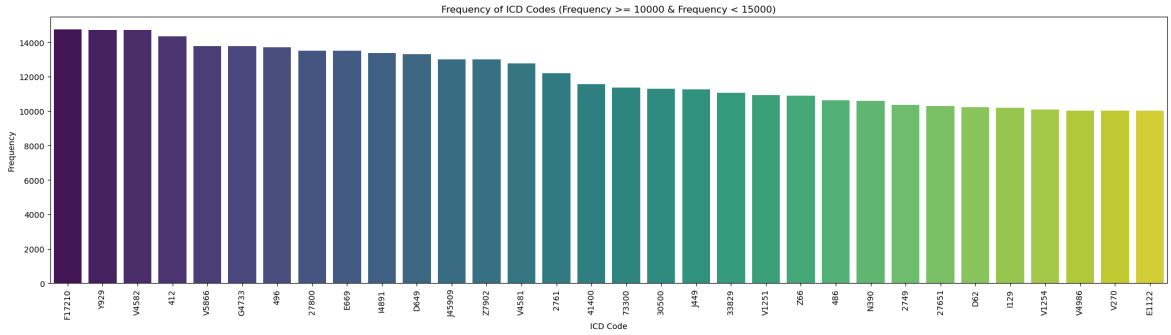
11.1 Figures, Code and Tables from 6.2

11.1.1 Additional Figures from 6.2.1

The following figures display the ICD diagnosis distribution graphs for the other frequency buckets:



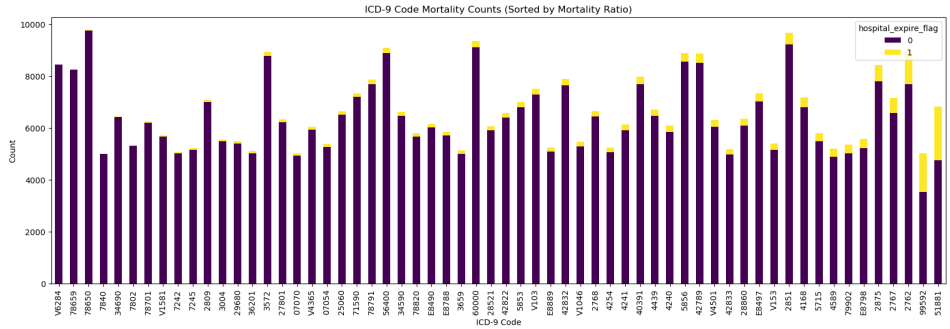
(a) Diagnosis Distribution (Frequency: 5,000–10,000)



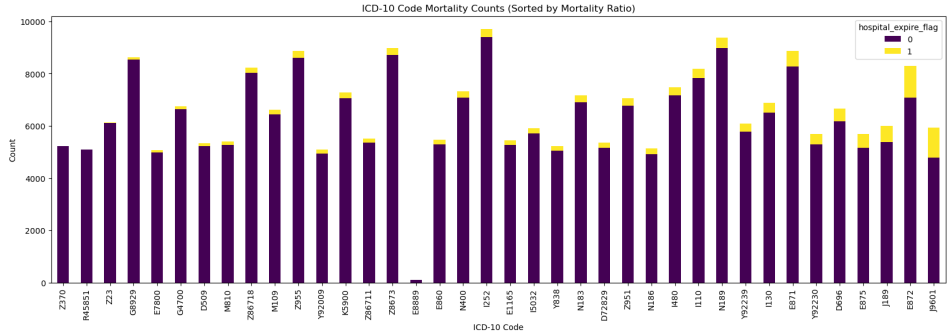
(b) Diagnosis Distribution (Frequency: 10,000–15,000)

Figure 25: Diagnosis Distribution Graphs for ICD Codes at Different Frequencies

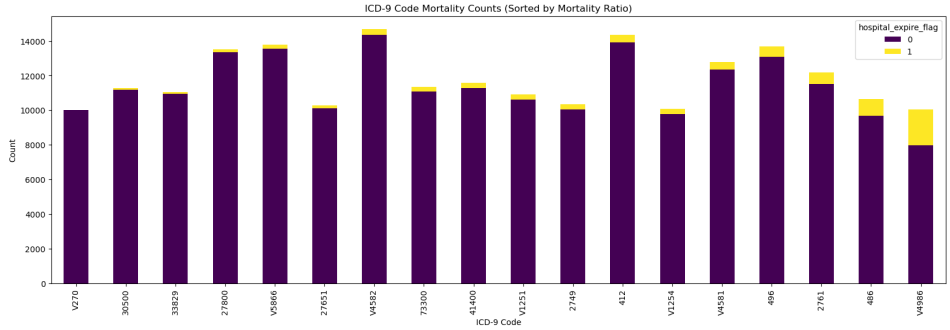
The following figures display the ICD mortality ratio distribution graphs for the other frequency buckets (5,000–10,000 and 10,000–15,000 occurrences):



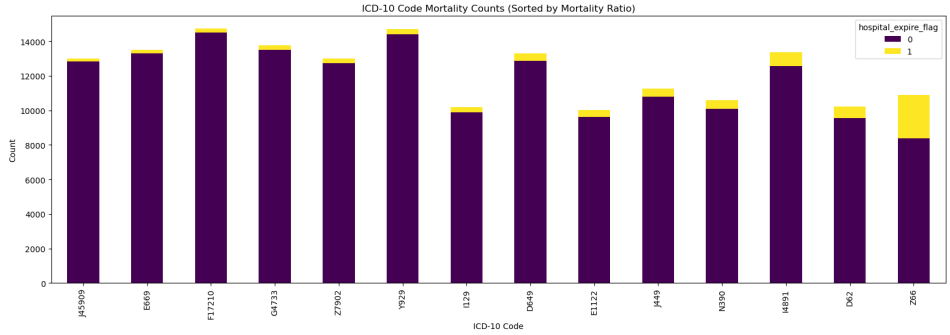
(a) ICD-9 Mortality Ratio Distribution (Frequency: 5,000–10,000)



(b) ICD-10 Mortality Ratio Distribution (Frequency: 5,000–10,000)



(c) ICD-9 Mortality Ratio Distribution (Frequency: 10,000–15,000)



(d) ICD-10 Mortality Ratio Distribution (Frequency: 10,000–15,000)

Figure 26: Mortality Ratio Distribution for ICD Codes (Frequency: 5,000–10,000 and 10,000–15,000)

11.1.2 Additional Figures from 6.2.2

The following figures display the demographic distributions for remaining cohorts (ARDS, Acidosis and Kidney Failure)

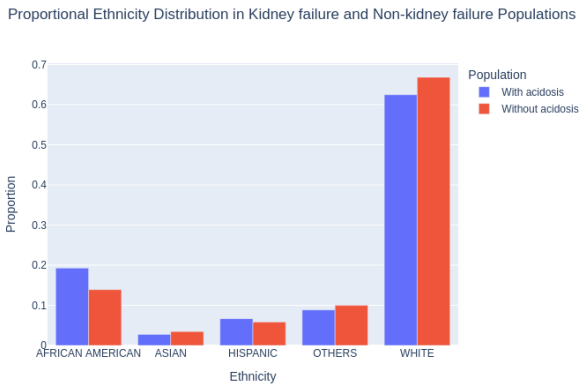


Figure 27: Race Distribution for the Acidosis Cohort

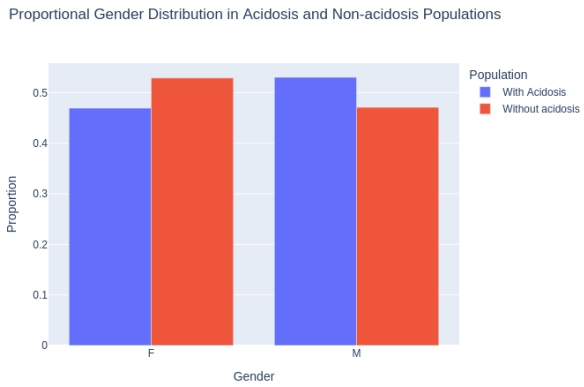


Figure 28: Gender Distribution for the Acidosis Cohort

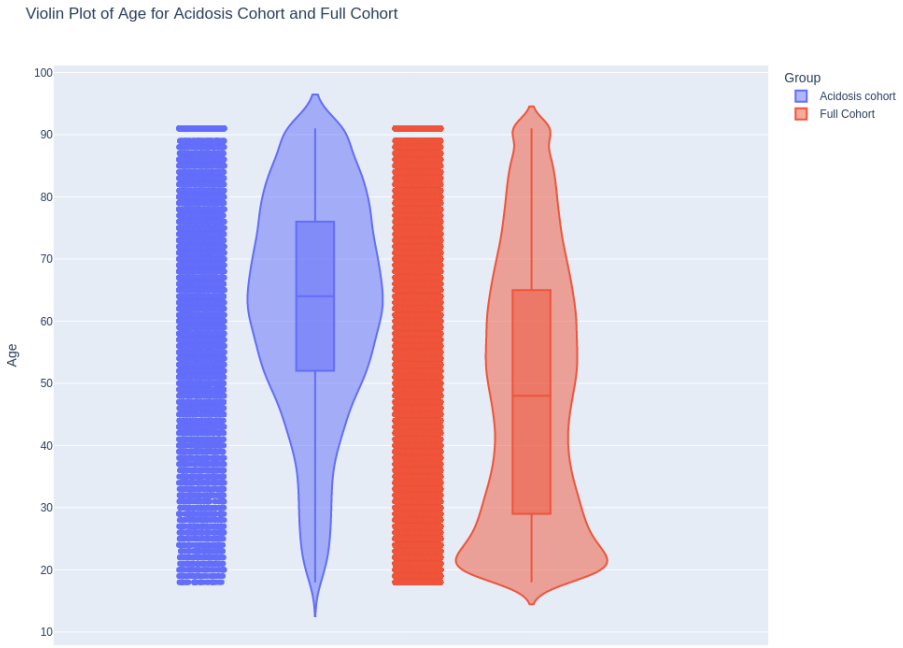


Figure 29: Age Distribution for the Acidosis Cohort

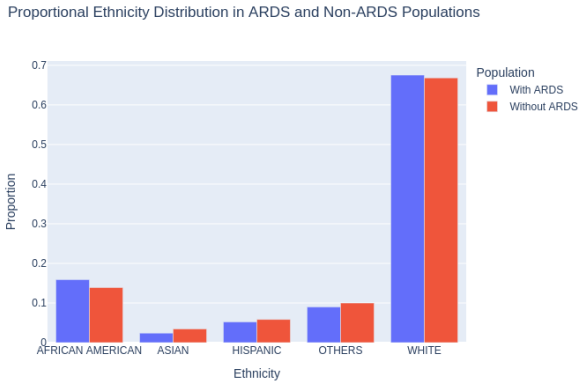


Figure 30: Race Distribution for the ARDS Cohort

Figure 31: Gender Distribution for the ARDS Cohort

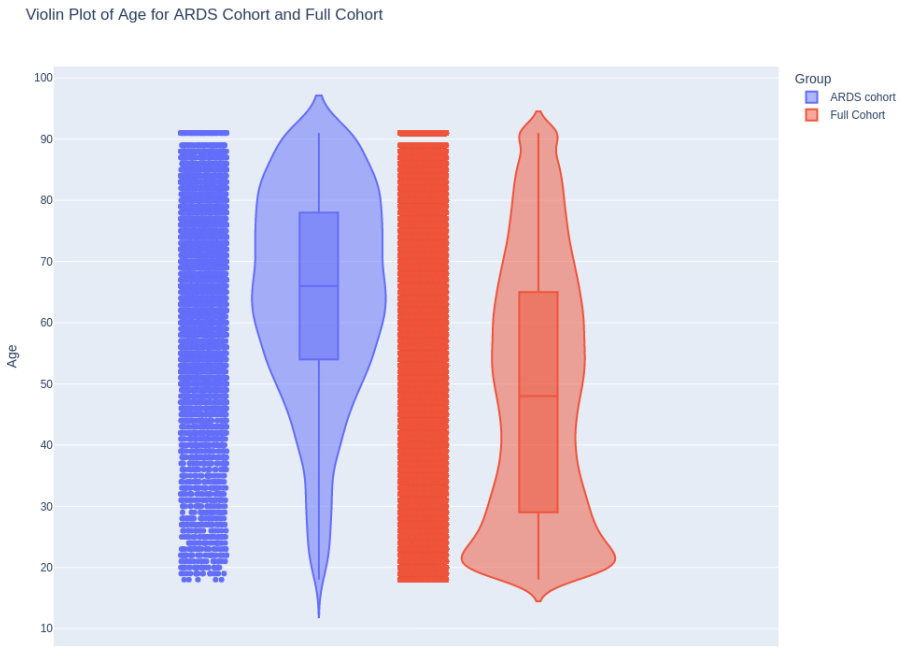


Figure 32: Age Distribution for the ARDS Cohort

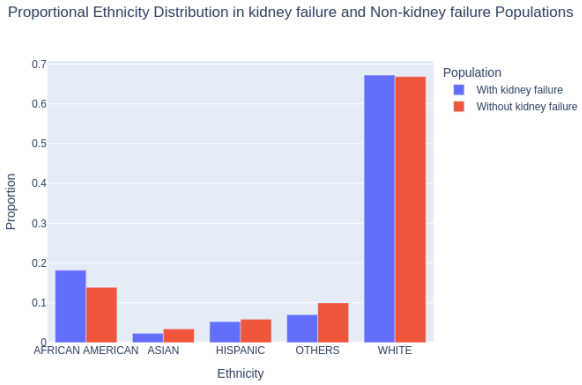


Figure 33: Race Distribution for the Kidney Failure Cohort

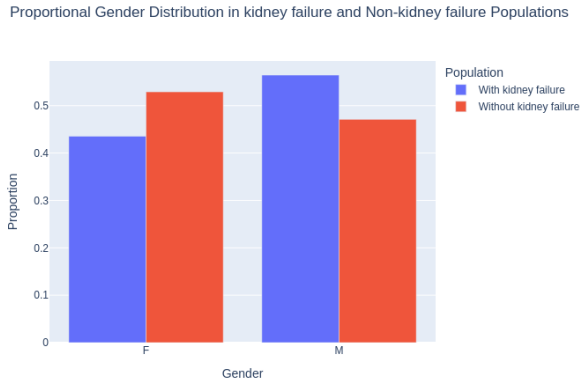


Figure 34: Gender Distribution for the Kidney Failure Cohort

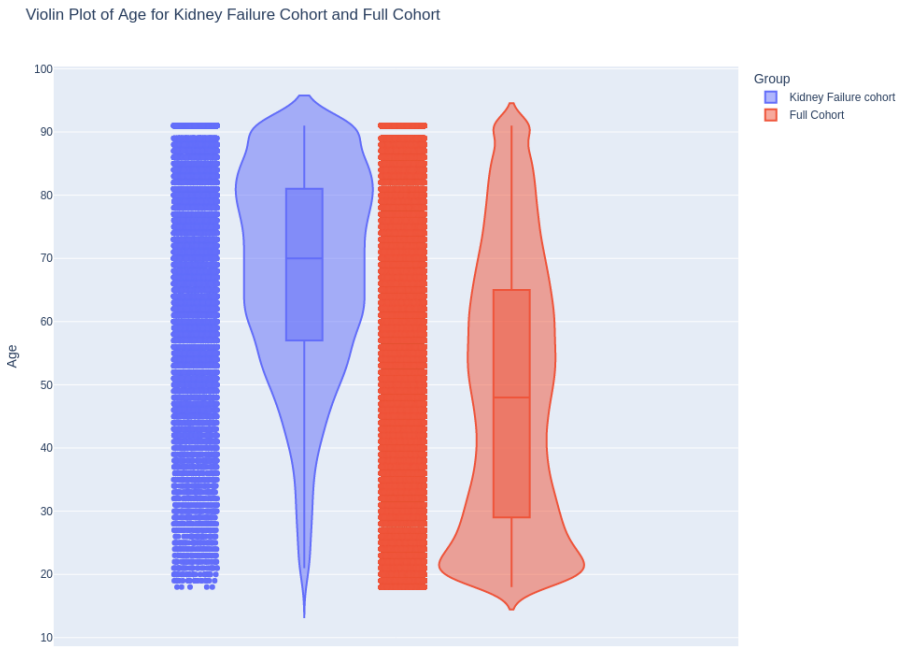


Figure 35: Age Distribution for the Kidney Failure Cohort

11.1.3 Python Code from 6.2.5

The following Python function, `take_blood`, extracts day-wise values for selected blood parameters from the MIMIC-IV dataset. This function processes the data based on ICD codes and the days of admission for sepsis patients.

```
def take_blood(icd_codes, days = [1,3,6,7]):

    # took out blood related param names
    blood_items = d_items[d_items["abbreviation"].isin([i.split("_")[0] for i in
    ↪  ['Blood Temp CCO (C)_°C', 'NBPd_mmHg', 'NBPs_mmHg', 'Hemoglobin_g/dL',
    ↪  'Sodium_mEq/L', 'Potassium_mEq/L', 'Glucose_mg/dL']])]
    blood_lab_items = d_labitems[(d_labitems["category"].str.contains("Blood
    ↪  Gas")) | (d_labitems["fluid"] == "Blood")]

    # making column names for params
    blood_param = pd.DataFrame(columns=["itemid", "testlabel_unit"])
    blood_param["itemid"] = blood_items["itemid"]
    blood_param["testlabel_unit"] =
    ↪  blood_items["abbreviation"].str.cat(blood_items["unitname"].astype(str),
    ↪  sep = '_')

    blood_dparam = blood_lab_items[["itemid", "label"]]

    # take out patients with icd code
    icd_diag = diagnoses[diagnoses["icd_code"].isin(icd_codes)]
    icd_adm =
    ↪  admissions[admissions["hadm_id"].isin(icd_diag["hadm_id"])][['subject_id',
    ↪  'hadm_id', 'admittime', 'dischtime', 'deathtime',
    ↪  'hospital_expire_flag']]
    icd_adm["hours_admitted"] = (pd.to_datetime(icd_adm["dischtime"]) -
    ↪  pd.to_datetime(icd_adm["admittime"]))
    icd_adm["died_in_days"] = (pd.to_datetime(icd_adm["deathtime"]) -
    ↪  pd.to_datetime(icd_adm["admittime"]))


    # time admitted and died in days
    icd_adm["hours_admitted"] = icd_adm["hours_admitted"].dt.days * 24 +
    ↪  icd_adm["hours_admitted"].dt.seconds // 3600
    icd_adm["died_in_days"] = np.ceil((icd_adm["died_in_days"].dt.days * 24 +
    ↪  icd_adm["died_in_days"].dt.seconds // 3600) / 24)
    icd_adm["died_in_days"] = icd_adm["died_in_days"].fillna(-1)
    icd_adm["died_in_days"] = icd_adm["died_in_days"].astype(int)

```

```
    death_df = icd_adm[["hadm_id", "died_in_days", "hours_admitted"]]

    for i in days:
        icd_adm[f"{i}_day_adm"] = icd_adm["hours_admitted"] > 24 * i
        icd_adm[f"{i}_day_adm"] = icd_adm[f"{i}_day_adm"].astype(int)

    # take lab reports for those patients
    blood_lab = chartevents[chartevents["hadm_id"].isin(icd_adm["hadm_id"])]

    blood_dlab = labevents[labevents["hadm_id"].isin(icd_adm["hadm_id"])]
    blood_dlab = blood_dlab[['subject_id', 'hadm_id', 'charttime', 'itemid',
    ↪  'valuenum']]
    blood_dlab.rename(columns={'valuenum' : 'value'}, inplace=True)

    blood_lab = pd.merge(blood_lab, blood_param, on="itemid").drop(columns =
    ↪  ["itemid"])
    blood_lab["charttime"] = pd.to_datetime(blood_lab["charttime"])
    blood_lab["value"] = blood_lab["value"].astype(float)

    blood_dlab = pd.merge(blood_dlab, blood_dparam, on="itemid")#.drop(columns =
    ↪  ["itemid"])
    blood_dlab["charttime"] = pd.to_datetime(blood_dlab["charttime"])
    blood_dlab["value"] = blood_dlab["value"].astype(float)

    # group according to hadm id, chart day and test type
    lab_by_day = blood_lab.groupby(["hadm_id", pd.Grouper(key='charttime',
    ↪  freq='D'), 'testlabel_unit']).mean().reset_index()
    dlab_by_day = blood_dlab.groupby(["hadm_id", pd.Grouper(key='charttime',
    ↪  freq='D'), 'label']).mean().reset_index()

    # calculating day
    lab_by_day['Day'] =
    ↪  lab_by_day.groupby('hadm_id')['charttime'].transform(lambda x: (x -
    ↪  x.min()).dt.days + 1)
    dlab_by_day['Day'] =
    ↪  dlab_by_day.groupby('hadm_id')['charttime'].transform(lambda x: (x -
    ↪  x.min()).dt.days + 1)
    dlab_by_day["hadm_id"] = dlab_by_day["hadm_id"].astype(int)

```

```
    # making pivot
    pivoted_df = lab_by_day.pivot_table(index=['hadm_id', 'Day'],
    ↪  columns='testlabel_unit', values='value')
    pivoted_df.columns = [f'{col}' for col in pivoted_df.columns]
    pivoted_df = pivoted_df.reset_index()

    d_pivoted_df = dlab_by_day.pivot_table(index=['hadm_id', 'Day'],
    ↪  columns='label', values='value')
    d_pivoted_df.columns = [f'{col}' for col in d_pivoted_df.columns]
    d_pivoted_df = d_pivoted_df.reset_index()

    pivoted_df = pd.merge(pivoted_df, d_pivoted_df, how="left",
    ↪  left_on=["hadm_id", "Day"], right_on=["hadm_id", "Day"])

    # filtering on days and merging
    pivoted_df = pivoted_df[(pivoted_df["Day"].isin(days))]
    final = pd.merge(icd_adm[["subject_id", "hadm_id", "hospital_expire_flag"]],
    ↪  pivoted_df, on = "hadm_id")

    # making daynumber-wise fixed rows for all ids, NaNs for ones that did not
    ↪  have a day
    days_to_add = pd.MultiIndex.from_product([final['hadm_id'].unique(), days],
    ↪  names=['hadm_id', 'Day'])
    df_full = final.set_index(['hadm_id',
    ↪  'Day']).reindex(days_to_add).reset_index()
    df_full['subject_id'] =
    ↪  df_full.groupby('hadm_id')['subject_id'].transform(lambda x:
    ↪  x.ffill().bfill())
    df_full['subject_id'] = df_full['subject_id'].astype(int)

    df_full["Status"] = df_full.apply(lambda x: "Admitted" if
    ↪  x["hospital_expire_flag"] == 0 else np.nan, axis=1)
    df_full.hospital_expire_flag =
    ↪  df_full.hospital_expire_flag.fillna(method="ffill")

    df_full = pd.merge(df_full, death_df, on="hadm_id", how = "left")

    df_full["Status"] = df_full.apply(status_assign, axis = 1)

```

```
    pivot_df = df_full.pivot_table(
    index=['subject_id', 'hadm_id', 'hospital_expire_flag'],
    columns='Day',
    values= list(df_full.columns)[4:-2],
    aggfunc='first'
    )

    # Flatten the MultiIndex columns
    pivot_df.columns = [f'{col[0]}_{col[1]}' for col in pivot_df.columns]

    # Reset index to make it a regular DataFrame
    pivot_df = pivot_df.reset_index()

    pivot_df['hospital_expire_flag'] =
    ↪  pivot_df['hospital_expire_flag'].astype(int)

    return pivot_df, df_full
```

11.2 Python Code from 8

The following Python function, `run_model`, processes the dataset, trains multiple machine learning models, applies an ensemble method, and evaluates their performance using confusion matrices and classification reports.

```
def run_model(df, target_column, threshold = 0.7, top_n = 0.9,
↪  feature_split_size = 0.25, model_split_size = 0.3):
    class_0 = df[df[target_column] == 0]
    class_1 = df[df[target_column] == 1]

    sample_size = min(len(class_0), len(class_1))

    class_0_sampled = class_0.sample(n=sample_size, random_state=42)
    class_1_sampled = class_1.sample(n=sample_size, random_state=42)

    sampled_df = pd.concat([class_0_sampled, class_1_sampled])

    sampled_df = sampled_df.sample(frac=1,
    ↪  random_state=42).reset_index(drop=True)

```

```
    df = sampled_df

    df.dropna(axis=1, thresh=int(len(df) * threshold), inplace= True)
    df.shape

    X = df.drop(columns=[target_column])
    y = df[target_column]

    fill_NaN = SimpleImputer(missing_values=np.nan, strategy='mean')
    imputed_DF = pd.DataFrame(fill_NaN.fit_transform(X))
    imputed_DF.columns = X.columns
    imputed_DF.index = X.index

    X = imputed_DF

    pipeline = Pipeline(steps=[
        ('classifier', RandomForestClassifier(n_estimators=100,
        ↪  random_state=42))
    ])

    X_train, X_test, y_train, y_test = train_test_split(X, y,
    ↪  test_size=feature_split_size, random_state=42)

    pipeline.fit(X_train, y_train)

    rf = pipeline.named_steps['classifier']
    feature_importances = rf.feature_importances_

    importance_df = pd.DataFrame({
        'Feature': X.columns,
        'Importance': feature_importances
    })
    importance_df = importance_df.sort_values(by='Importance', ascending=False)
    select_top = int(len(importance_df) * top_n)
    top_features = importance_df['Feature'].head(select_top).values
    X_top = X[top_features]

    X_train, X_test, y_train, y_test = train_test_split(X_top, y,
    ↪  test_size=model_split_size, random_state=42)
```

```
50
51      rf_model = RandomForestClassifier(random_state=42)
52      gb_model = GradientBoostingClassifier(random_state=42)
53      lr_model = LogisticRegression(random_state=42)
54      ada_model = AdaBoostClassifier(random_state=42)
55      lgb_model = lgb.LGBMClassifier(random_state=42, verbose = -1)
56      xgb_model = xgb.XGBClassifier(random_state=42)
57
58
59      rf_model.fit(X_train, y_train)
60      gb_model.fit(X_train, y_train)
61      lr_model.fit(X_train, y_train)
62      ada_model.fit(X_train, y_train)
63      lgb_model.fit(X_train, y_train)
64      xgb_model.fit(X_train, y_train)
65
66      print("Running random forest model")
67      rf_preds = rf_model.predict(X_test)
68      print("Running gradient boosting model")
69      gb_preds = gb_model.predict(X_test)
70      print("Running logistic regression model")
71      lr_preds = lr_model.predict(X_test)
72      print("Running ada boost model")
73      ada_preds = ada_model.predict(X_test)
74      print("Running lgb model")
75      lgb_preds = lgb_model.predict(X_test)
76      print("Running xgb model")
77      xgb_preds = xgb_model.predict(X_test)
78
79      print("Running ensemble model")
80      ensemble_preds = np.array([rf_preds, gb_preds, lr_preds, ada_preds,
        ↪  lgb_preds, xgb_preds]).T
81      majority_vote_preds = [np.bincount(row).argmax() for row in ensemble_preds]
82      conf_matrices = {
83      "Random Forest": confusion_matrix(y_test, rf_preds),
84      "Gradient Boosting": confusion_matrix(y_test, gb_preds),
85      "Logistic Regression": confusion_matrix(y_test, lr_preds),
86      "AdaBoost": confusion_matrix(y_test, ada_preds),
87      "LGBM": confusion_matrix(y_test, lgb_preds),
```

```
    "XGB": confusion_matrix(y_test, xgb_preds),
    "Ensemble (Majority Vote)": confusion_matrix(y_test, majority_vote_preds),
    }

    stats = {}

    for i, j in {'Random Forest': rf_preds, 'Gradient Boosting' : gb_preds,'Logistic Regression' : lr_preds, 'AdaBoost' : ada_preds, 'LGBM' : lgb_preds, 'XGB' : xgb_preds, "Majority Votes" : majority_vote_preds, }.items(): #"Special Ensemble" : ensemble_preds_spec
        stats[i] = classification_report(y_test, j, output_dict=True)

    return conf_matrices, stats
```

This function was run within the following loop for all Days, MDRR values, train-test splits and models to evaluates their performance and find the best one.

```
for i in range(1, 6):
    print(f"\n\n============== Day: {i} ============== \n\n")
    df = pd.read_csv(f"daywise_sepsis/sepsis_blood_{i}")
    df = df.drop(columns=["Unnamed: 0", "hadm_id", "subject_id", "Status"])
    df["hospital_expire_flag"] = df["hospital_expire_flag"].astype(int)
    target_column = 'hospital_expire_flag'
    df.columns = df.columns.str.replace('[^A-Za-z0-9_]', '', regex=True)

    thres_subrange = np.arange(thres_range[i][0], thres_range[i][1], 0.05)
    # thres_subrange = np.arange(thres_range[i][0], 0.35, 0.05)

    for j in [0.1, 0.2, 0.3]:
        print(f"\n\n============== Split Size: {j} ============== \n\n")
        thres_results = {}
        cms = {}
        for k in thres_subrange:
            print(f"\n\n============== Threshold: {k} ============== \n\n")

            cm, stats = run_model(df, target_column, threshold=k, model_split_size = j)
            # print(f"Remaining columns after dropping: {len(df.columns)}")
            # print(f"Shape after dropping: {df.shape}")
```

```
            thres_results[k] = stats
            cms[k] = cm

        plot_data = []

        for threshold, models in thres_results.items():
            for model, metrics in models.items():
                plot_data.append({
                    "Threshold": threshold,
                    "Model": model,
                    "Accuracy": metrics["accuracy"],
                    "F1-Score": metrics["1"]["f1-score"],
                    "Precision": metrics["1"]["precision"],
                    "Recall": metrics["1"]["recall"]
                })
        metrics_df = pd.DataFrame(plot_data)
        metrics_df = metrics_df.sort_values(by=['Model',
        ↪  "Threshold"]).reset_index(drop = True)
        metrics_df["Threshold"] = metrics_df["Threshold"].round(2)
        metrics_df.to_csv(f'ML_results/HEF/Day
        ↪  {i}/model_comparision_split_{j}.csv', index = False)

        fig, axes = plt.subplots(2, 2, figsize=(14, 10), sharex=True)

        sns.lineplot(data=metrics_df, x="Threshold", y="Accuracy", hue="Model",
        ↪  marker="o", ax=axes[0, 0])
        axes[0, 0].set_title("Accuracy vs Threshold")
        axes[0, 0].set_ylabel("Accuracy")

        sns.lineplot(data=metrics_df, x="Threshold", y="F1-Score", hue="Model",
        ↪  marker="o", ax=axes[0, 1])
        axes[0, 1].set_title("F1-Score vs Threshold")
        axes[0, 1].set_ylabel("F1-Score")

        sns.lineplot(data=metrics_df, x="Threshold", y="Precision", hue="Model",
        ↪  marker="o", ax=axes[1, 0])
        axes[1, 0].set_title("Precision vs Threshold")
        axes[1, 0].set_ylabel("Precision")
```

```
56
57          sns.lineplot(data=metrics_df, x="Threshold", y="Recall", hue="Model",
            ↪  marker="o", ax=axes[1, 1])
58          axes[1, 1].set_title("Recall vs Threshold")
59          axes[1, 1].set_ylabel("Recall")
60
61          for ax in axes.flat:
62              ax.set_xlabel("Threshold")
63              ax.grid(True)
64
65          fig.tight_layout()
66          # plt.show()
67          plt.savefig(f'ML_results/HEF/Day {i}/model_comparision_split_{j}.png')
68
69          for threshold, models in cms.items():
70              for model_name, matrix in models.items():
71                  disp = ConfusionMatrixDisplay(confusion_matrix=matrix,
                    ↪  display_labels=[0, 1])
72                  disp.plot(cmap=plt.cm.Blues)
73                  plt.title(f"Confusion Matrix for {model_name} at Threshold
                    ↪  {round(threshold, 2)}")
74                  # plt.show()
75                  plt.savefig(f'ML_results/HEF/Day
                    ↪  {i}/{model_name}/Split_{j}/{round(threshold, 2)}.png')
```

References

Ning Hou, Meng Li, Li He, Bin Xie, Lin Wang, Rong Zhang, Yan Yu, Xiao Sun, Zhenyu Pan, and Kai Wang. Predicting 30-days mortality for mimic-iii patients with sepsis-3: a machine learning approach using xgboost. *Journal of Translational Medicine*, 18(1):462, 2020. doi:10.1186/s12967-020-02620-5. URL https://translational-medicine.biomedcentral.com/articles/10.1186/s12967-020-02620-5. Open-access publication.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-Wei H Lehman, Mengling Feng, Marzyeh Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iv: A freely accessible critical care database. *Scientific Data*, 10(1):1–14, 2023. doi:10.1038/s41597-023-02055-7.

Sarika R Khope and Susan Elias. Strategies of predictive schemes and clinical diagnosis for prognosis using mimic-iii: A systematic review. *Healthcare (Basel)*, 11(5):710, 2023. ISSN 2227-9032. doi:10.3390/healthcare11050710. URL https://www.mdpi.com/2227-9032/11/5/710. Open-access publication.