Addis Ababa University
አዲስ፡አበባ፡ዩኒቨርሲቲ

Seek Wisdom, Elevate your Intellect and Serve Humanity

College of Natural and Computational Sciences, School of Information Science.

Predicting Emergency Department Mortality Risk Using Machine Learning Algorithms:

The Case of Yekatit 12 Hospital Medical College

BY

ALEMAYEHU GIRMA

ADVISORS

AWGICHEW K.

(MSc IN BIOSTATISTICS, PhD CANDIDATE)

AND

CO-ADVISOR: - DR. MERGA B. (PhD)

Addis Ababa, Ethiopia

September 20, 2025

DECLATATION

I, Alemaychu Girma, the principal investigator of this research, declare that this thesis is my original work and that it has not been submitted partially or fully by any other person for an award of a degree in any institution.

Name of the investigator: Alemayehu Girma

Signature: _____ Date: _____

Name of 1st Advisor: Mr. Awgichew K. (MSc in Biostatistics, PhD candidate)

Signature: _____ Date: 11/11/2025

Name of 2nd Advisor: Dr. Merga Belina. (PhD)

Signature: _____ Date: 20/09/2025

Name of 1st Examiner: Dr. Martha Yifiru. (PhD)

Signature: _____ Date: 11/11/2025

Name of 2nd Examiner: Dr. Yimer Seid (PhD)

Signature: _____ Date: 11/11/2025

ACKNOWLEDGMENT

First and foremost, I would like to give thanks to the almighty GOD who helped me in all situations.

I want to express my gratitude to APHEREA-DST for all of their help with this thesis. My advisers, Mr. Awgichew K. (MSc in Biostatistics, PhD candidate) and Dr. Merga B. (PhD) , deserve special recognition for their tremendous contributions, patience, encouragement, and advice, all of which have been crucial to finishing this study.

Last but not least, I would want to express my sincere gratitude to my family for their unwavering moral support, sage counsel, help, and ongoing inspiration.

ABSTRACT

The increasing burden of emergency department (ED) mortality in resource-constrained settings, such as Ethiopia, underscores the need for innovative approaches to enhance clinical decision-making. Though its use in low- and middle-income countries (LMICs) is still relatively unexplored, machine learning (ML) holds great promise for improving mortality risk prediction using electronic medical record (EMR) data. While machine learning (ML) has been widely used in high-income healthcare settings, little is known about how well it works for predicting mortality risk in LMIC emergency departments (EDs), especially when using locally relevant datasets and overcoming issues like data imbalance and unstructured EMR data.

The goal of the study is to identify important predictive variables, evaluate the effectiveness of different machine learning models in a resource-constrained setting, and develop and assess ML-based model for predicting mortality risk among emergency department patients at Yekatit 12 Hospital Medical College in Addis Ababa, Ethiopia.

A two-year dataset of EMRs from Yekatit 12 Hospital (October 2022–September 2024, or 2015–2016 in Ethiopian calendar) was used for a retrospective analysis. Attribute selection, Data cleaning, Text mining, Data transformation, and Feature engineering were all used in the methodology to glean insights from unstructured data. Four machine learning models: Random Forest, k-Nearest Neighbors, Support Vector Machines, and XG Boosting - were trained and evaluated. Undersampling, hybrid approaches, and the Synthetic Minority Oversampling Technique (SMOTE) were used to rectify the data imbalance. Evaluation parameters such as accuracy, precision, recall, F1-score, and Area Under the Receiver Operating Characteristic Curve (AUC-ROC) were used to evaluate the model's performance. To improve model robustness, RandomizedSearchCV was used for hyperparameter optimization.

The Random Forest model, optimized with RandomizedSearchCV, performed better than other models, with an F1-score of 0.68, precision of 0.90, recall of 0.55, and AUC-ROC of 0.81. Patients' clinical diagnoses, age, vital signs, and some laboratory result were critical predictive variables. The use of SMOTE and hybrid sampling techniques greatly enhanced

model performance on imbalanced data, with Random Forest showing superior generalization across evaluation metrics.

This study shows that machine learning (ML), specifically Random Forest with hyperparameter tuning, can provide a scalable tool for clinical decision support by accurately predicting the risk of ED mortality in an LMIC setting using routine EMR data. In order to improve predictive accuracy, the results emphasize how critical it is to address data imbalance and use text mining for unstructured data.

Keywords: - Electronic Medical Record (EMR); Predicting Mortality Risk; Machine Learning;

TABLE OF CONTENT

List of figures: -

List of Tables: -

List of Acronyms: -

AUROC	Area Under the Receiver Operating Characteristic Curve
CBC	Complete blood count
ED	Emergency Department
ED-LOS	Emergency Department Length of Stay
EHR	Electronic Health Records
EMOH	Ethiopia Ministry of Health
EMR	Electronic Medical Record
GS	GridSearchCV
k-NN	k-Nearest Neighbors
LMIC	Low- and middle-income country
LOS	Length of stay
MAP	Mean Arterial Pressure
MCV	Mean Corpuscular Volume
ML	Machine Learning
PR-ROC	Precision-Recall Receiver Operating Characteristics
RF	Random Forest
RS	RandomizedSearchCV
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machines
WBC	White Blood Cell
XGBoost,	eXtreme Gradient Boosting
Y12HMC	Yekatit 12 Hospital Medical College

CHAPTER 1: Introduction

1.1 Background of the Study

The emergency department provides services for patients with conditions that are life-threatening or potentially life-threatening over 24 hours a day, every day of the year. Emergency Department patient mortality is a serious public health issue that has gotten worse recently [1]. The impact of emergency department death on the individual, society, and the overall health system is significant. It is quickly rising to the top of the list of fatalities in hospitals. It accounts for 15–16% of all hospital deaths worldwide. In sub-Saharan Africa's low and middle-income countries (LMICS), particularly in the Central East and Western regions, the value is significantly higher at 5.1% than in high-income countries [2].

Despite the fact that emergency care systems can directly affect more than half of mortality in LMICs, the development of ED systems is still in its infancy throughout Africa, even though more than ten years have passed since the 60th World Health Assembly called on all member states to implement "formal, emergency medical-care systems." Organized prehospital care services are lacking in many nations, and even when patients do make it to hospitals, many of them lack emergency personnel with the necessary training. The ability of even the most sophisticated referral centers on the continent to treat patients with acute diseases varies widely, as do the infrastructure, services, and equipment they offer [3].

Rising patient numbers and an increase in the demand for hospital admissions have made overcrowding in hospital emergency rooms a global concern in recent decades. This has resulted in more medical errors, greater fatality rates, less patient satisfaction, and longer waiting times for critically ill patients [4]. High rates of disease and mortality are frequently associated with poor leadership, scarce resources, and gaps in provider competence, according to research conducted in LMICs. Furthermore, noncommunicable illnesses and injuries are becoming more prevalent in LMICs [5].

There is currently little information available on traumatic and medical causes of death in Ethiopia's emergency rooms. Nonetheless, a high incidence of illness and injury, restricted access to high-quality healthcare, and a lack of qualified medical staff and necessary resources are thought to be contributory reasons. Although increasing access to emergency treatment is frequently thought of as a remedy, its effect on patient outcomes has not been sufficiently demonstrated [2].

Since substantial morbidity and mortality can result from delays in prompt, accessible, and cheap treatment, emergency care is essential to reducing deaths and disabilities. The World Health Organization has prioritized upgrading acute care systems in LMICs due to the fact that competent emergency and trauma care has been shown to reduce patient fatalities globally. ED patient outcomes are a crucial gauge of the standard of treatment provided by a medical facility, and the mortality should be less than 1.5 % by 2024. However, many LMICs experience acute care constraints, including inadequate infrastructure and staffing gaps, in spite of initiatives by health ministries and partners. Financial limitations, high expenses, and the complexity of healthcare delivery make it difficult to improve population health and allocate healthcare resources optimally. Effective resource allocation depends on early clinical deterioration identification, but emergency physicians frequently find this difficult because it necessitates the expert interpretation of large amounts of clinical data to avert preventable fatalities [4].

Many conventional grading schemes have been created in recent decades to measure the severity of illnesses and forecast patient outcomes, such as morbidity and death. To provide a standardized measure that clinicians may use consistently, these scoring methods seek to objectively evaluate pathophysiological changes in critically ill patients. [6].

Artificial intelligence tools can help with pattern identification and classification issues in medicine, such as early disease detection, diagnosis, and medical decision-making. They are also being used more in public health concerns [7]. Particularly, ML approaches have demonstrated promise in forecasting patient outcomes, frequently surpassing traditional regression models in this regard. Numerous studies have shown

how well machine learning (ML) predicts mortality, particularly in high-risk groups like sepsis patients in intensive care units [8]

1.2 Statement of the Problem

Globally, in the last two decades, there has been an imbalance between emergency service supply and demand, due to the annual ED visits rapidly increasing with the growing population, which affects emergency care. ED patient mortality is a serious public health issue that has gotten worse recently. Many ED deaths take place in LMICs with limited resources, with a global ED mortality rate estimated to be 15–16%. According to litterateurs, morbidity and mortality burden in LMICs unacceptably high. This has led to increased focus on emergency services in Africa, where acute communicable and non-communicable disease mortality is still a major burden [5, 1].

If quality emergency care services are available and individuals use them promptly, millions of deaths and permanent disabilities from acute illnesses and injuries could be avoided. It is estimated that better emergency care might prevent up to 54% of deaths in LMICs each year [9].

The Ethiopian Federal Ministry of Health (EMOH) has worked to increase access to emergency care by offering skilled medical professionals, sustainable, efficient, and high-quality service. However, still the pooled prevalence of mortality in the Ethiopian ED was 7.71%. According to the subgroup analysis utilizing the country's regions, Dire Dawa had the highest ED patient mortality rate, 16.70%, followed by the Amhara area, 12.89%. In a similar vein, a different subgroup analysis based on the study population's age showed that adult ED patient mortality is higher (8.23%) than pediatric mortality (4.48%) [1].

Early identification of hospitalized patients at high risk of death may enhance clinical and organizational decision-making and aid in estimating the number of medical resources needed. On contrary, it might be difficult to predict a patient’s outcome when they are admitted to ED. This is true since a wide range of illnesses and disease severity are present in ED patients. In order to identify high-risk patients who may

require intense intervention, several algorithms have been utilized. Among the algorithms, the Rapid Emergency Medicine Score (REMS), Rapid Acute Physiology Score (RAPS), Worthing Physiological Scoring system (WPS), Routine Laboratory Data (RLD), and Admission Laboratory Tests (ALT) have been widely used [10, 11].

The Quick Sequential Organ Failure Assessment (qSOFA), Systemic Inflammatory Response Syndrome (SIRS), the Sequential Organ Failure Assessment (SOFA), the Acute Physiology and Chronic Health Evaluation II (APACHE II), the Simplified Acute Physiology Score II and III (SAPS II/III), the National Early Warning Score (NEWS), Mortality in Emergency Department Sepsis (MEDS), the Charlson Comorbidity Index (CCI), and the Glasgow Coma Scale (GCS) are the clinical indicators that have been proposed specifically for the prognosis of sepsis patients. Although each of these approaches combines laboratory and clinical data to forecast results, their efficacy differs, underscoring the need for more indicators to improve mortality prediction [12].

More accurate findings have been obtained in the medical profession in recent years as a result of the gradual use of various artificial intelligence applications through machine learning. In recent years, a variety of statistical and machine learning techniques have been proposed to improve ED activity forecasting. Importantly, by making clinical interpretation and analysis easier, machine learning models can be utilized to develop intelligent medical decision support tools that save practitioners' time. The provision of helpful prognostic information improved clinical decision-making, and the effective utilization of medical resources are all made possible by accurate ED mortality prediction [13, 14, 15].

Not alone in LMICs', even in more developed healthcare systems, the application of ML in areas such as ED mortality risk prediction remains relatively limited. In sub-Saharan Africa, challenges such as limited access to reliable EMR systems, fragmented data collection practices, and resource constraints have hindered the development and implementation of advanced predictive models. Most ED mortality prediction tools in Africa rely on conventional statistical algorithms, which often lack the complexity and adaptability needed to account for the unique health challenges and resource limitations in the country. These conventional methods, while useful,

may fail to fully capture the dynamic and multifaceted nature of patient outcomes in under-resourced settings. And the absence of previous work that shows which ML algorithm is effective in predicting ED mortality risk using EMR data was the triggering point of this research.

This research aims to bridge this gap by being one of the first in the region to create an ML model for ED mortality. The study will not only demonstrate the feasibility of ML in these settings but also provide a framework for how such models can enhance decision-making in resource-constrained environments. Moreover, the findings will add to the expanding evidence on the effectiveness of machine learning in enhancing patient outcomes. This work could also serve as a foundation for future innovations, encouraging the adoption of data-driven healthcare solutions across the region.

1.3 Research Questions

This study tries to answer the following research questions: -

- Which machine learning model is most effective in using EMRs and forecasting patient mortality risk at ED?
- What are the key predictors of mortality in the ED?

1.4 General and Specific objectives

General Objective: -

- The general objective of the study is to design or develop a ML model from EMR to predict mortality risk at the ED.

Specific Objectives: -

- To develop a machine learning model that predicts patient mortality risk in the ED.
- To identify key predictors of mortality at ED using data from EMRs.
- To collecting and preparing data for machine training and model development.
- To evaluate and compare the performance of various machine learning algorithms in predicting patient outcomes at the ED

1.5 Significance of the study

The growing problem of the rise in patient mortality, especially in low and middle-income countries like Ethiopia, is addressed by this study. Globally, the increase in emergency department visits has put a strain on healthcare systems, resulting in longer wait times, overcrowding, and delayed admissions—all of which have a detrimental impact on patient outcomes.

This study aims to create a prediction model to detect high-risk patients by utilizing machine learning and electronic health records. This will allow for prompt medical treatments and optimize the use of available resources. In contrast to conventional statistical techniques, ML models are flexible and accurate, which makes them ideal for environments with limited resources. This strategy seeks to increase the effectiveness of emergency care, lower mortality, and improve clinical decision-making, all of which will improve patient outcomes and the standard of healthcare.

Furthermore, this study is remarkable in its use of a ML model to predict ED mortality risk based on the first EMR data in Ethiopia. It offers a solution that is specific to the region's particular problems and creates a framework for implementing comparable advances in LMICs. The results could change the way emergency care is delivered by influencing healthcare policies, encouraging investment in EMR systems, and promoting the use of AI-driven solutions. This research lays the path for future innovations and life-saving interventions in under-resourced areas by adding to the worldwide body of knowledge on machine learning in healthcare.

1.6 Ethical Considerations.

The Institutional Review Board (IRB) of Addis Ababa University's College of Natural Sciences granted this study complete ethical clearance, guaranteeing adherence to participant protection and research integrity guidelines. After clearance, the Y12HMC Research and Publication Department formally granted access to the necessary datasets. Strict respect to confidentiality protocols was maintained during the data gathering process, which was carried out in conjunction with the hospital data center. To preserve patient privacy, all personal identifiable information was eliminated.

Only password-protected systems were used to store the carefully selected datasets. These procedures, which were put in place to protect data integrity and stop illegal use, demonstrate the study's dedication to moral and responsible research methods.

1.7 Thesis Organization.

There are five chapters in this thesis. An introduction is given in Chapter 1, which covers the problem statement, background, and goals of the study. The second chapter examines pertinent research on machine learning techniques, mortality scoring systems, and ER mortality. The study technique, including data collection, preparation, preprocessing, and feature engineering, is covered in Chapter 3. The experimental data are presented and examined in Chapter 4. The investigation is finally concluded in chapter five, which also provides recommendations based on the results.

CHAPTER 2: - Literature Review

2.1 Emergency Department Mortality and Causes

The ED is a multipurpose space where patients at any level of a medical facility are assured of access around-the-clock. By offering the initial line of care upon arrival, the department serves as the foundation for both the general public and medical facilities. To put it briefly, ED serves as the health service's "shop window" [16].

Mortality in emergency rooms has a significant effect on people, society, and the health system as a whole. It is responsible for 15–16% of hospital deaths worldwide. In LMICs, the value is significantly greater. In Sub-Saharan Africa, it is 5.1% higher than in higher-income nations. In recent years, ED has contributed significantly to the burden of disease and patient mortality [16].

The World Health Organization (WHO) recommends that all countries, regardless of resources or economic development, start by establishing a comprehensive emergency care system and monitor progress . Because of this, emergency services are receiving more attention in Africa, where the burden of mortality from acute communicable and non-communicable diseases is still significant [1].

Mortality from medical diseases and traumatic injuries in Ethiopian hospitals' emergency rooms was mostly uncharacterized, but the causes were believed to be complex and included a high burden of trauma, illnesses, and a lack of access to high-quality healthcare services. Cancer, trauma, road accidents, and cardiovascular disease are the main causes of this mortality. According to reports, these causes account for 15–60% of all fatalities, even if the etiologies vary depending on the area [14, 16].

2.2 Different Scoring Systems.

In recent decades, several grading systems have been developed to provide a measuring method for evaluating the severity of illness and predicting patients' outcomes, morbidity, and mortality. Such scores aim to bring the severity of the

specific emergency and certain disease into an objective measurement, which is applicable and differentiable by any physician all over the world [17].

The severity of illness classification system APACHE II (Acute Physiology and Chronic Health Evaluation), described by Knaus et al. in 1985, uses a point score based on 12 routine physiologic measurements, together with age and previous health status, for use on intensive care patients. However, the APACHE II score includes several blood chemistry variables and is therefore not suitable for quick scoring in the ED, it remains impractical for rapid injury severity assessment required in the ED or in the field [18].

The Rapid Emergency Medicine Score (REMS), an attenuated version of APACHE II, allows for prompt calculation. REMS is a composite score consisting of the GCS, RR, oxygen saturation, MAP, hazard ratio, and age. Among non-surgical patients who present to the ED, REMS has proven to be a valid predictor of mortality. REMS was also found to be a strong predictor of in-hospital mortality for the trauma population. But injury type is an important predictor of mortality in patients with trauma, which was not included in the REMS calculation [19].

The mortality in emergency department sepsis (MEDS) score, developed in 2003, was derived and validated in patients presenting to the ED with suspected sepsis, and aims to risk-stratify patients based on their risk for 28-day in-hospital mortality (IHM). It includes basic demographic, clinical, and laboratory variables that can be attained during a patient's time in the ED, which are weighted to give a final score and mortality risk assessment [20].

Furthermore, SOFA scores also reveal predictive information on past medical problems and sepsis severity indices based on a multivariate study. SOFA ratings are based exclusively on physiological and laboratory features, and do not take into account host factors such as ethnicity, age, and comorbidities, which are important predictors of death in sepsis. SOFA and MEDS scores, which use medical and laboratory data to measure the severity of organ failure, have been shown to relate to mortality in sepsis patients treated in the emergency department [21].

2.3 Machine Learning and the Healthcare Systems

Due to several issues, healthcare systems around the world are investigating the possibilities of ML classifiers to make more objective judgments and lessen their dependency on doctors' subjective assessments. High-quality predictive models can be extracted from large datasets using ML, a subfield of artificial intelligence (AI). Machine learning is being used more in medical research to improve predictive modeling and identify novel characteristics that influence outcomes. Because AI models can identify intricate patterns in massive datasets that the human brain finds difficult to identify, they can be useful tools for increasing the accuracy of clinical predictions. The large number of variables already entered EMRs can be used for thorough data analysis using machine learning to avoid overfitting and make it easier to create prediction models that are updated on a regular basis [12, 23].

Numerous studies have used machine learning algorithms to identify multiple risk indicators among laboratory tests and vital signs to predict adult patients' in-hospital mortality [10].

2.4 Machine Learning Algorithms

One kind of technique that uses data to automatically generate models is the machine learning algorithm [24]. These methods can be applied to tasks including clustering, classification, and prediction. Numerous methods and approaches for machine learning problems have been developed over time [25]. These machine learning algorithms—KNN, SVM, RF Decision Tree, and XGBoost—were employed in this investigation.

2.4.1 K-Nearest Neighbor (KNN)

Based on the supervised learning technique, K-Nearest Neighbor is one of the most basic machine learning algorithms. The K-NN algorithm places the new case into the category that is most like the available categories based on the assumption that the new case and data are comparable to the available cases. The K-NN method classifies a new data point based on similarity after storing all the relevant data. This implies that the K-NN algorithm makes it simple to classify newly discovered data into an

appropriate category. Although the K-NN technique is mostly utilized for classification problems, it can also be employed for regression. Since K-NN is a nonparametric approach, it doesn't assume anything about the underlying data. Because it stores the dataset and then executes an action on it at the time of classification, it is also known as a lazy learner algorithm. This is because it does not learn from the training set right away. During the training phase, the KNN algorithm simply saves the dataset and, upon receiving new data, classifies it into a category that is substantially comparable to the new data [26].

2.4.2 Random Forest

There are many variations in the behavior of the decision trees that make up Random Forest. Comparing Random Forest to a single tree classifier reveals a notable boost in performance. Random Forest makes use of the bagging concept. Random Forest selects a random sample from the data set using an ensemble technique called bagging, sometimes referred to as bootstrap aggregation. As a result, each model is created using the bootstrap samples—a step of row sampling with replacement—that were supplied by the original data. Every model is trained separately and produces outcomes. The process of integrating all results and producing output based on majority voting is known as aggregation.

RF is a forest of decision trees that are generated at random, with each node calculating the output by analyzing a random subset of features. The result of each decision tree is then combined by the random forest using majority voting to produce the final output. Hype parameters are used by random forests either to speed up the model or to improve their performance and prediction ability. Random forests are far more effective than choice trees if the trees are acceptable and diversified. Compared to a single decision tree, Random Forest produces more accurate and exact findings and improves classification performance when there are many instances. Additionally, it solves the over-fitting and missing values issues brought on by the dataset’s missing values. Consequently, Random Forest is suggested for a range of classification issues with confidence if one must select from several tree-based classifiers for classification problems [27,28].

2.4.3 Support Vector Machine.

Finding a hyperplane in an N-dimensional (N—the number of features) space that clearly classifies the data points is the aim of the support vector machine method. Decision boundaries known as hyperplanes aid in the data points' classification. The number of input features determines the hyperplane's dimension; if there are two, the hyperplane is a line; if there are three, it is a two-dimensional plane; however, if there are more than three, the hyperplane's dimension becomes challenging. The goal is to identify the hyperplane with the largest margin among the several potential hyperplanes to divide classes. By increasing the margin distance, which offers some reinforcement, future data points can be categorized with greater assurance [29].

2.4.4 XGBoost.

Chen and Guestrin proposed the machine learning technique known as XGBoost, or eXtreme Gradient Boosting. An enhanced version of the gradient boosting method is called XGBoost. It is a machine-learning algorithm for supervised learning issues that is incredibly powerful, scalable, and optimized. XGBoost is an ensemble of decision trees rather than an artificial neural network, in contrast to the LSTM method. Furthermore, because XGBoost employs decision trees to determine the most crucial variables and create models, it is more effective at handling huge datasets and can produce more accurate results than artificial neural networks, which can be computationally costly and require more training time.

The most popular technique for creating predictive models right now is XGBoost because of its exceptional accuracy, effectiveness, and versatility. When compared to other algorithms, XGBoost offers several benefits, including the capacity to manage huge and complicated datasets, highly parallelizable code, and missing data. To further increase its accuracy, XGBoost also has a number of adjustable parameters. Moreover, XGBoost is a reliable approach for issues involving both regression and classification [30].

2.5 Related Works.

In a study that forecast unfavorable outcomes in hospitals, A non-disease-specific Bayesian network (BN) model was created by Cai et al. to forecast death, readmission, and length of stay (LOS) from electronic health records. Using data from 32,634 patients admitted via the emergency department to a Sydney hospital between 2008 and 2011, the model achieved an average daily accuracy of 80% with an area under the receiver operating characteristic curve (AUROC) of 0.82 for mortality, which was the most predictable result. The Genetic Algorithm-Optimized Convolutional Neural Network (GAOCNN) is a deep hybrid model that was introduced by Tavakolian et al. to predict hospital readmission and length of stay. Tested on three different healthcare datasets, this model demonstrated remarkable prediction accuracy, reaching 97.2% for predicting hospital readmissions in diabetic patients and 89.0%, 99.4%, and 94.1% for predicting length of stay (LOS) in diabetic, COVID-19, and intensive care unit (ICU) patients, respectively [31].

Another study intended to predict emergency department mortality by comparing several models in the selected Tehran hospital in Iran, which was done in 2023. Over one month one-month, demographic data and records were gathered from 1,000 patients admitted to the emergency department. To make predictions, Random Forest and Cat Boost models were used. The Cat Boost model demonstrated an outstanding mean accuracy of 0.94 (standard deviation: 0.03), outperforming Random Forest by a significant amount. The study demonstrates how machine learning outperforms traditional methods in forecasting mortality at emergency departments with remarkable accuracy and efficiency. Early diagnosis and intervention can be greatly enhanced by putting such models into practice. In turn, this makes it possible for the emergency room to allocate resources as efficiently as possible, avoiding overuse and ultimately saving lives while improving patient outcomes [32].

According to this study, predicting patient deterioration and averting unplanned deaths in the ED are difficult tasks that require clinical expertise and sophisticated interpretation of complex data. This study compared ED mortality prediction models, concentrating on the RF and CB algorithms. Out of 46 features, RF determined that diabetes mellifluous (DM), ejection fraction (EF), and UREA were the most

significant factors. Clinical experts then agreed on the top 22 features. RF showed remarkable predictive ability, especially when handling high-dimensional data, even though both models (RF and CB) performed well in accuracy, precision, recall, and F1-score. Notably, the CB model had a slightly higher recall (94% vs. 92%) and F1-score (94% vs. 93%), but the strength of RF is its strong feature selection, giving priority to important markers such as EF (a key cardiac function indicator), UREA (a measure of renal/metabolic status), and DM (a major comorbidity). The reliability of RF in ED risk stratification is further validated using ROC curves and k-fold cross-validation [32]. Given the high cost and limited availability of these factors in most LMIC emergency departments, most of them are important for CVD patients, not for all ED patients', the RF model is restricted to use for CVD patients, making it unsuitable for broad application in LMICs' ED settings.

To improve early risk detection among patients with thrombocytopenia, the study used ML approaches. RF outperformed the other models in the test, providing more prediction accuracy than traditional techniques. Important mortality trends were found in the study: the 2-day death rate was lowest in Group E (least severe) at 7.93%, while the highest was in Group A (those with the most severe thrombocytopenia) at 18.99%. The overall mortality rate for all critical platelet groups was 10.63%, underscoring the pressing need for improved risk classification techniques. With an AUC of 0.848, RF showed outstanding discrimination. Its positive predictive value (PPV) of 46.1% was almost twice as high as the Naive Bayes model's (28.4%), and its specificity of 99.1% greatly decreased false alarms. The reliability of RF was further supported by the uneven performance of other models, including stochastic gradient descent (SGD) and artificial neural networks (ANN), with testing sensitivity as low as 3.6%.

The RF model gave the highest weight to abnormal hematologic markers, particularly blast cells and atypical lymphocytes/monocytes, according to a feature importance analysis (which is displayed in the study's bar chart). This suggests that the presence of immature or dysplastic blood cells is a crucial risk signal. Platelet counts, hemoglobin levels, and RBC indices such as MCV and MCHC also had a moderate impact, suggesting that anemia and clotting dysfunction play a part in patient outcomes. Interestingly, RF gave morphological abnormalities more weight than

standard metrics like total white cell count, which is consistent with clinical knowledge of leukemia and marrow failure.

What makes RF unique is its well-balanced performance: with a sensitivity of 15.1%, it still detects significant clinical risk while providing excellent specificity to cut down on pointless warnings. Although it has more interpretability, its strong AUC rival's logistic regression (LR) and performs better than SVM and decision trees (DT). Significantly, the RF model's predictions are accurate and actionable, as nearly half of the cases it flags are genuinely high-risk and require intervention (PPV of 46.1%) [33]

CHAPTER 3: Methodology

3.1 Study Setting

The study was conducted in Y12HMC, which is an academic institution found in Addis Ababa, Ethiopia, under the Addis Ababa City Administration. The hospital treats over 310000 people from the city and its surrounding areas each year and has about 500 beds. It has over twenty-one departments, wards, and outpatient clinics. Around the end of 2019 G.C., the hospital began using Electronic Medical Record.

3.2 Study Design and Period

In this study, experimental research design was employed, incorporating retrospective cohort data from Y12HMC EMR. The study was conducted over two-year period of data collected over a period from October 2022 to September 2024 (2015 and 2016 E.C).

3.3 Data Sources

The dataset was obtained from the EMR of Y12HMC and includes all adult emergency department visits between October 2022 and September 2024. It is structured in a tabular format, comprising 12,440 rows, each representing an individual patient visit, and 34 columns corresponding to specific variables such as age, sex, vital signs, laboratory results, and clinical outcomes. Data extraction was performed using Structured Query Language (SQL) and exported as CSV files. To ensure confidentiality, all identifiable patient information was removed prior to analysis. This study employed a comprehensive enumeration approach, analyzing data from all eligible patients without the need for sampling.

The study included adult patients aged 18 years and above who presented to the emergency department within the defined study period. Patients with partially completed medical records available in the electronic medical record (EMR) system were also included. However, patients who arrived at the emergency department after

death, records with substantial missing data (exceeding 50%) or missing target variables, and duplicated entries, all of which were removed from the analysis.

3.4 Operational Definitions

ED-mortality: - The death of the patient during the service provision at ED.

Co-morbidity: - The underlying illnesses that patients have often precipitate or contribute significantly to the main causes of death.

Triage time: - is the time from the patient's arrival to ED until seen by the triage officer, which should not exceed 5 minutes of arrival to ED [5]

3.5 Attribute Selection

From the initial 34 attributes only 26 variables were chosen with the help of domain experts (emergency and critical care seniors) and a literature review. Because it might not be particularly effective to take all variables in the dataset to identify the best predictors. Another reason is that the more variables a model has, the longer it takes to construct, and improper models may result. To make modeling easier, it is therefore required to exclude attributes that are not crucial for analysis. The removed variables are: MRN, the names of the staff who provide the data, laboratory result arrived data, vital sign measurement data and time, patients' database ID (which is different from MRN), and the remark column.

Table 1: Categories of Variables.

Category	Examples
Demographics	Age, gender, (gives 2 categorical variable)
Clinical	Case Category, Triage color, Vital Sign:(Heart rate, respiratory rate, BP, SpO_2, temperature, RBS) (gives 2 categorical variable and gives 7 numerical variable)
Labs	CBC (gives 11 numerical variable)
Comorbidities	Physician Assessment (gives 1 categorical variable)
Temporal	Triage waiting time, Visit type, (gives 2 categorical variable)
Outcome	Discharge Outcome (binary: alive vs. dead) (gives 1 categorical variable)

3.6 Data Preprocessing

Because real-world databases are usually very large, they are susceptible to noise, missing data, and inconsistencies. Knowledge discovery during the training phase is more challenging if there is a large amount of redundant, irrelevant, and missing information available, or if the input is noisy and untrustworthy. Making the data suitable for the selected machine learning tasks and algorithms is one of the most crucial aspects of conducting this type of experimental research. This stage's goal is to clean up the data as much as possible and transform it into a format that will be used in later phases.

There are various limitations with the data that were gathered and used in this investigation. These consist of inconsistency and missing values in different attributes. In order to ensure that the data met the needs of the chosen machine learning models, the researcher prepared it accordingly. The researcher examined the quantity and distribution of missing values, noisy data (like: human or recording errors, irrelevant features, outliers, e.t.c.) and the significance of individual attribute values and types during the data preparation process. Following that, all limitations associated with the data were avoided through various methods (like: Correct or removing the value, Transformation and Normalization, winsorization/capping,) in-order to meet the requirements of the chosen algorithms.

3.6.1 Data Cleaning

Although it takes a lot of time and effort, data cleaning is a crucial step in the creation of a successful model. Many of the current data are dirty and cleaning them out requires more time and hard work from the researcher. Filling in missing numbers, smoothing noisy data, locating and eliminating outliers, and resolving discrepancies are all steps in the data cleaning process.

Dropping raw's and columns

This step involves removing rows where this critical dependent variable is missing to ensure the data is suitable for analysis. By focusing only on the 'Disposal Outcome' column, any row lacking a value in this field is excluded, creating a new dataset that preserves the integrity of the original while avoiding modifications to it. The process also provides feedback on the number of rows removed and confirms that no missing values remain in the target variable afterward. This was necessary because predictive models rely on complete target data for training and evaluation, and without a known outcome for each data point, those instances cannot contribute to learning or validation. As a result, this cleaning step ensures the dataset contains only valid examples, with 3611 rows dropped due to missing 'Disposal Outcome' values, making it ready for effective model development.

Next, columns with more than 50% missing values were dropped to simplify the dataset and enhance model performance. High missingness in columns like temperature and random blood sugar (each exceeding 70% missing data) offered little useful information and could introduce noise if imputed. Removing these reduced the dataset's complexity, conserved computational resources, and ensured the remaining data was more robust for analysis, resulting in a streamlined dataset with two fewer columns.

Outlier Detection and Handling.

To visualize the outliers, the researcher first attempted to create boxplots for the various continuous variables. For example attributes in the dataset, like heart rate (HR), systolic blood pressure (SBP), white blood cell count (WBC), hemoglobin (Hg), and platelet count (PLT), contained extreme values (outliers). According to the data from the boxplots, the outlier treatment technique is applied to those attributes that had outliers. Thus, using domain knowledge and the particular values those percentiles indicated during the analysis, the greatest and lowest percentiles were chosen for each attribute, respectively, to establish the upper and lower boundaries. This method made sure that the thresholds selected were data-driven and clinically meaningful. Finally, values above the upper boundaries and below the lower

boundaries were then capped or substituted with the greatest percentiles and lowest percentiles, respectively.

For the heart rate (HR) variable, the lower and upper thresholds were defined using the 0.5th and 99.5th percentiles, respectively. Any HR value below the 0.5th percentile was replaced by the 0.5th percentile value, and any value above the 99.5th percentile was replaced by the 99.5th percentile value. For systolic blood pressure (SBP), the capping thresholds were determined at the 0.5th and 99.9th percentiles, replacing values beyond these limits with their respective bounds. For oxygen saturation (SpO_2), outlier capping was similarly performed using the 0.1st percentile as the lower bound and the 99.2nd percentile as the upper bound. Observations below or above these cut-off points were replaced by the corresponding threshold values. This procedure ensured that extreme observations were adjusted to fall within a reasonable range, thereby reducing the impact of outliers on subsequent statistical analyses and predictive modeling while maintaining the integrity and interpretability of the dataset.

Handling Missing Values.

Finding and fixing missing values is crucial to getting the dataset ready for analysis. To identify missingness trends and determine the scope of the problem, this method starts with exploratory data analysis utilizing programs like Microsoft Excel and Python. The distribution of missing data within the dataset is shown using a missingness matrix. The variables that have missing values are clearly visible with this graphical tool, as is the presence of any trends, such as if missing values in one column frequently occur with missing values in another. The data can be classified as Missing Completely at Random (MCAR), Missing at Random (MAR), or Missing Not at Random (MNAR) based on these patterns.

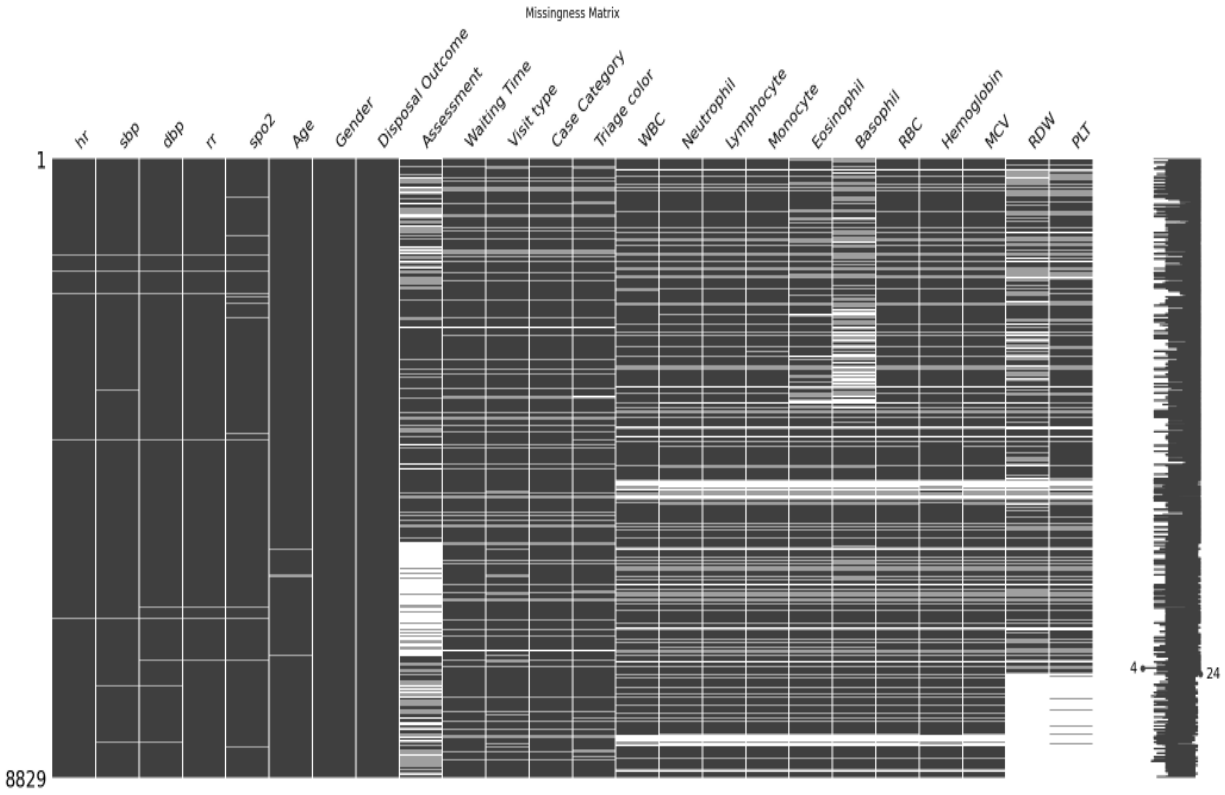

Figure 1:- Missingness Matrix

The dataset's completeness is visually represented by the missingness matrix, where present values are shown in black and missing values are shown in white. The missing data density may be quickly identified because each row is represented by a horizontal line and each column by a vertical line. The percentage of missing values is larger in columns with more white space. Additionally, the plot aids in identifying missingness trends, such as relationships between missing entries in different columns.

For example, a number of lab result variables, such as neutrophil, lymphocyte, monocyte, eosinophil, basophil, red blood cell, hemoglobin, MCV, RDW, and PLT, showed significant connections in missingness. The assumption of MCAR is highly unlikely because of this pattern, which implies that if one lab value is missing, others are probably missing as well.

Imputation techniques that take into account inter-feature correlations were applied, including blood panel values, Assessment, Waiting Time, Visit Type, Case Category, Triage Color, SpO2, Age, SBP, DBP, RR, and HR. Specifically, Multivariate

Imputation by Chained Equations (MICE) was used, which was customized for each variable's data type (continuous, nominal, or ordinal).

1. Ordinal Variable Imputation (e.g., Triage Color)

Using MICE with an IterativeImputer, a BayesianRidge estimator, and an OrdinalEncoder, the ‘Triage Color’ variable were imputed. Prior to imputation, the OrdinalEncoder was used to numerically encode the categories, along with a placeholder for missing data. MICE need numerical input; thus, this is required. The IterativeImputer then used all other features as predictors and treated each feature with missing values as a regression target to carry out step-by-step imputations. The Bayesian Ridge regression model was used in this procedure to estimate the missing data. After imputation was finished, the encoder's inverse transform was used to return the numerical values to their original categorical representations.

2. Continuous Variable Imputation

A two-stage method that incorporated basic and multivariate imputation techniques was used to impute continuous data. To ensure a complete matrix that could be scaled using StandardScaler, missing values were first handled by a Simple Imputer technique using a median. The iterative process's stability and performance are enhanced by this scaling stage. Multivariate imputation was then performed using IterativeImputer with a RandomForestRegressor estimator internally. To forecast the missing entries for each continuous variable with missing values, a random forest model was trained using the remaining characteristics. Inverse scaling was used to return the data to its initial range following the imputation process.

3. Nominal Variable Imputation (e.g., Gender, Visit Type, Case Category)

Using MICE with an IterativeImputer and a RandomForestClassifier estimator internally, nominal variables were imputed. OneHotEncoder converted the nominal categories into binary numeric vectors before imputation, which is a prerequisite for MICE. To control unseen categories and prevent multicollinearity, the encoder was set up with handle_unknown='ignore' and drop='first'. Classification models that predicted the presence (1) or absence (0) of a category based on other features were then used to impute each binary column that represented a category. Following the

completion of the missing values, the encoder's inverse transformation was used to return the binary vectors to their initial nominal categories.

This strong and customized imputation framework guarantees that the dataset preserves the greatest amount of information while suitably resolving the issue of missing data according to its structure and characteristics.

3.6.2 Text Mining

It is crucial to use text mining techniques to extract significant values from the disorganized 2000+ unique values, which are in the form of strings, and for medical terminology to be appropriately identified and mapped to their appropriate categories during analysis, spelling must be consistent. The free-text descriptions in the 'Assessment' column frequently contain mistakes, misspellings, spelling variants, and inconsistent abbreviations, which are common in real-world medical data. These problems occur because various people could type incorrectly or enter information differently. This is fixed by standardizing the text using a typo_corrections dictionary, which substitutes the right spellings (as values) for all known errors (as keys). The same medical word or ailment is uniformly represented throughout the dataset.

The function preprocess_text2 is designed to process and analyze medical text, such as clinical notes or symptom descriptions, in order to extract relevant medical categories based on the terms found within the text. It starts by taking care of punctuation, making sure that commas are placed correctly to divide words that are next to each other, and cleaning up the text by substituting spaces for special characters like +, ?, _, and -. The content is then normalized by converting it to lowercase, deleting numerals and ordinal suffixes (such as "1st" or "2ry"), removing single quotes, and removing most punctuation. It then does typo correction by mapping common misspellings to their right forms (for example, "pnemonia" to "pneumonia") using a predetermined dictionary.

After cleaning, non-alphabetic tokens are removed from the text by tokenizing it into individual words. The inclusion of stopword removal as a commented-out phase is optional, although it may help weed out common terms like "and" or "the." To capture

a broad variety of medical phrases, the function then extracts bigrams and trigrams (such as "chest pain" or "shortness of breath") in addition to single-word tokens. Following that, these terms are compared to a pre-established lexicon of medical categories, each of which corresponds to a list of pertinent terms. To guarantee uniqueness, any related categories are gathered into a set. Lastly, the function returns to an empty list if no relevant categories are found, a list of categories if several categories are detected, or a single category if only one is found.

```
def preprocess_text2(text):
    # Step 1: Add spaces around commas to separate attached words
    text = re.sub(r',', ' , ', str(text))

    # Step 2: Replace specific characters with spaces
    text = re.sub(r'[+?_-]', ' ', text)

    # Step 3: Lowercase, remove punctuation (except protected commas) and numbers
    text = text.lower()
    text = re.sub(r'[^a-zA-Z0-9\s,]', '', text)
    text = re.sub(r'[^\w\s?\-,]', '', text)
    text = re.sub(r'\d+', '', text)
    text = re.sub(r'\d+(?:st|nd|rd|th|ry)', '', text)
    text = re.sub(r"'\b|\b'", '', text)
```

Figure 2: Text Cleaning Code

3.6.3 Feature Engineering

Several new variables were developed in this feature engineering process to improve the dataset's prediction ability by identifying associations that have clinical significance. A more accurate measure of the average blood flow to organs was obtained by calculating the Mean Arterial Pressure (MAP) using the systolic and diastolic blood pressure readings.

MAP = dbp + (1/3) * (sbp - dbp).

The Systemic Inflammation Response Index (SIRI), as a measure of systemic inflammation metric, was calculated by dividing the neutrophil and monocyte counts by the lymphocyte count.

SIRI = (Neutrophil * Monocyte) / Lymphocyte.

Lastly, the Oxygen Delivery Index—which measures the blood's ability to carry oxygen to tissues—was created by multiplying hemoglobin concentration by oxygen saturation (SpO_2).

Oxygen Delivery Index = Hemoglobin * Spo2.

To prevent multicollinearity and redundancy, the original features (sbp, dbp, neutrophil, monocyte, lymphocyte, hemoglobin, and spo2) that were used to generate these new engineered features were removed from the DataFrame.

The goal of these manufactured characteristics is to enhance model performance and offer more complex clinical insights.

```
[ ]
    # Create new features
    Mydf33['MAP'] = Mydf33['dbp'] + (1/3) * (Mydf33['sbp'] - Mydf33['dbp'])

    Mydf33['SIRI'] = (Mydf33['Neutrophil'] * Mydf33['Monocyte']) / Mydf33['Lymphocyte']

    Mydf33['Oxygen Delivery Index'] = Mydf33['Hemoglobin'] * Mydf33['spo2']

    print(Mydf33)
```

Figure 3 = Feature Engineering Code

3.6.4 Data transformation

Consolidating or changing data to make it suitable for the development of machine learning models is known as data transformation. Data transformation techniques include generalization, and smoothing. Normalization is an operation carried out with attributes to scale the values that fall within a narrow range [34].

Based on domain knowledge, several of the continuous variable values in this study were converted into categorical ranges. Then the columns were binned.

For example: -

- Age (Divided into 3 clinically relevant groups: ' Young', 'Adult', 'Geriatric'),
- Vital Signs: Used standard medical classifications:(Heart Rate: -'Bradycardia', 'Normal', 'Tachycardia', Mean Arterial Pressure (MAP):-'Hypotension', 'Normal', 'Hypertension'),
- Laboratory Values: Based on reference ranges (White Blood Cells (WBC): -'Leukopenia', 'Normal', 'Leukocytosis')

Conditional Binning was used for more complex categorizations: -

- Neutrophil-to-Lymphocyte Ratio (NLR) ('Low', 'Intermediate', 'High')

```
Mydf5_2['Age_group'] = pd.cut(Mydf5_2['Age'], bins=[15, 35, 60, 105], labels=['Young', 'Adult', 'Geriatric'])
Mydf5_2['Waiting Time_group'] = pd.cut(Mydf5_2['Waiting Time'], bins=[0, 5, 25, 40], labels=['Immediate', 'Urgent', 'Non-urgent'])

Mydf5_2['hr_category'] = pd.cut(Mydf5_2['hr'], bins=[40, 60, 100, 160], labels=['Bradycardia', 'Normal', 'Tachycardia'])
Mydf5_2['rr_category'] = pd.cut(Mydf5_2['rr'], bins=[10, 20, 30, 40], labels=['Bradypnea', 'Normal', 'Tachypnea'])
Mydf5_2['MAP_category'] = pd.cut(Mydf5_2['MAP'], bins=[0, 70, 100, 180], labels=['Hypotension', 'Normal', 'Hypertension'])

Mydf5_2['WBC_category'] = pd.cut(Mydf5_2['WBC'], bins=[0, 4.5, 12, 35], labels=['Leukopenia', 'Normal', 'Leukocytosis'])
Mydf5_2['MCV_category'] = pd.cut(Mydf5_2['MCV'], bins=[70, 80, 100, 110], labels=['Microcytic', 'Normocytic', 'Macrocytic'])
Mydf5_2['RDW_category'] = pd.cut(Mydf5_2['RDW'], bins=[11, 14.5, 24], labels=['Normal', 'High'])
Mydf5_2['PLT_category'] = pd.cut(Mydf5_2['PLT'], bins=[20, 150, 450, 505], labels=['Thrombocytopenia', 'Normal', 'Thrombocytosis'])
Mydf5_2['Eosinophil_category'] = pd.cut(Mydf5_2['Eosinophil'], bins=[0.0, 0.5, 12.0], labels=['Normal', 'High'])
Mydf5_2['Basophil_category'] = pd.cut(Mydf5_2['Basophil'], bins=[0.0, 0.2, 2.0], labels=['Normal', 'High'])
Mydf5_2['RBC_category'] = pd.cut(Mydf5_2['RBC'], bins=[2.0, 4.5, 6.1,7.0], labels=['Low','Normal', 'High'])
```

Figure 4: Data transformation Code

3.6.5 Data Encoding

The encoding process has proceeded following the classification of the continuous variables into ordinal variables. It manages both ordinal and nominal types when encoding categorical variables. A dictionary is established to indicate the order of categories for ordinal variables, such as "Triage color," where the categories carry significant information (for example, green, yellow, orange, red, and black, from least to most severe). First, the code determines whether the dataset contains these ordinal variables (Mydf5_2). It uses this predetermined order to apply ordinal encoding if it is present. To maintain resilience during model deployment or subsequent data processing, the encoder additionally handles unknown or unseen categories by assigning them a placeholder value (for example, -1).

One-hot encoding is used for nominal variables, which are categorical but do not have an intrinsic order (such as "Gender," "Visit type," and "Case Category"). To prevent multicollinearity, this approach turns each category into a binary column, leaving off the first category as a reference. The script converts the dataset's nominal variables into a new set of binary columns after first confirming which ones are there. The original nominal variables are then substituted with these new columns in the dataset. Because machine learning methods typically require numerical input, this procedure guarantees that the dataset is numerically encoded and prepared for use.

3.6.6 Data Splitting

After completing all preprocessing steps, the final dataset consisted of 8,829 records and 36 features. The number of records was reduced from the original 12,440 to 8,829

after excluding 3,611 entities that lacked values for the target variable. Similarly, the number of features increased from 26 to 36 as a result of generating dummy variables during the categorical encoding process.

Starting with the column 'Discharge Outcome' as the goal variable, the splitting algorithm divides the dataset into predictor variables (X) and the target variable (y). Every other column is regarded as a feature that is utilized to forecast the result. The train_test_split function from scikit-learn is then used to divide the dataset into training and testing sets. With 20% set aside for testing (X_tes, y_tes) and 80% designated for training (X_tra, y_tra), there are enough test samples (~185 "Died" cases) to assess the model's performance on the minority class.

The stratify=y section preserves the target variable's original class distribution in both training and testing sets, which is crucial when the classes are unbalanced. Random_state=1489 guarantees that the split is repeatable across runs. This configuration reliably and consistently prepares the data for machine learning model evaluation and training. Based on comparative model performance, the 80–20% split ratio was selected over the 70–30% ratio after training and evaluating all models on both dataset partitions.

Table 2 = Training and testing dataset samples.

Class	Size of Training Sample	Size of Testing Sample	Total Sample size
Alive	6323 (89.5%)	1581 (89.5)	7904
Died	740 (10.5)	185 (10.5%)	925
Total number of samples = 8829			
Total Training samples		= 7063	
Total Testing samples		= 1766	

3.6.7 Dimensionality Reduction

To improve model performance and cut down on redundancy, a two-step feature selection and dimensionality reduction technique was employed in this phase. First, multicollinearity between two highly correlated features—'Oxygen Delivery Index_risk_Encoded' with 'RBC_category_Encoded' was addressed using Principal Component Analysis (PCA).

```
[205] from sklearn.decomposition import PCA

      # Assuming these are your 3 correlated pairs (6 total features)
      corr_pairs = [
          ['Oxygen Delivery Index_risk_Encoded', 'RBC_category_Encoded'],
          ['Leukocyte', 'Blood'],
          #['Protein', 'Blood']
      ]

      # Store PCA transformers for each pair
      pca_transformers = {}

      # Transform training data
      X_train1 = X_tra.copy()
      for pair in corr_pairs:
          pca = PCA(n_components=1)
          X_train1[f"PCA_{pair[0]}"] = pca.fit_transform(X_tra[pair])
          X_train1.drop(columns=pair, inplace=True)
          pca_transformers[tuple(pair)] = pca  # Save the fitted PCA
```

Apply the Same PCA to X_test

```
[206] X_test1 = X_tes.copy()
      for pair in corr_pairs:
          pca = pca_transformers[tuple(pair)]  # Reuse the PCA fitted on X_train
          X_test1[f"PCA_{pair[0]}"] = pca.transform(X_tes[pair])
          X_test1.drop(columns=pair, inplace=True)
```

Figure 5 = Applying PCA for the training and test dataset code.

Lists of feature names that are significantly correlated are included in the "Correlated pairs" list. It's ['Oxygen Delivery Index', 'Red Blood Cells'] in this case. The fitted PCA objects for every pair are stored in an empty dictionary called pca_transformers. To later apply the same transformation to the test data, it is imperative that the fitted transformers be saved. After looping through each pair in the "Correlated pairs" list, the code initializes a PCA object for the current pair using the formula pca = PCA(n_components=1). When n_components=1, PCA will distill each pair of features into a single principal component, which will capture the greatest amount of variance in that pair. What comes next is X_train1[f"PCA_{pair[0]}"]. = pca.fit_transform(X_tra[pair]): chooses the two columns in the current pair from the

initial training data, converts these two features into a single new feature (the first principal component), and computes the principal components based on the chosen pair of features in X_tra (fit). In X_train1, the resultant single primary component is allocated to a new column. Using an f-string, the new column is dynamically named, for instance, "PCA_Oxygen Delivery Index". The X_train1 DataFrame's two initial associated features are eliminated using X_train1.drop(columns=pair, inplace=True). The single PCA feature has taken its place.

The test data has been subjected to the identical PCA transformations that were discovered using the training data. To guarantee that the transformation is consistent between the training and test sets, avoid data leakage, and guarantee that the model trained on the converted training data will function properly on the transformed test data, it is imperative to use the same fitted PCA items. This guarantees that, with parameters learned exclusively from the training data, the test data experiences the same dimensionality reduction as the training data. Finally, after applying for PCA, the verification code is executed to confirm that the shapes of the converted training and test DataFrames have the same number of columns.

Following this, the most predictive features were then extracted from the dataset, which now contained the PCA components and the remaining uncorrelated variables, using Recursive Feature Elimination with Cross-Validation (RFECV) and a RandomForestClassifier. To assess model performance using 5-fold stratified cross-validation based on accuracy, RFECV iteratively eliminated the least significant features. The procedure determined the ideal feature count that produces the best cross-validated accuracy.

This combined approach—using PCA for dimensionality reduction and RFECV for feature selection—enhanced model efficiency, reduced noise, and improved predictive performance.

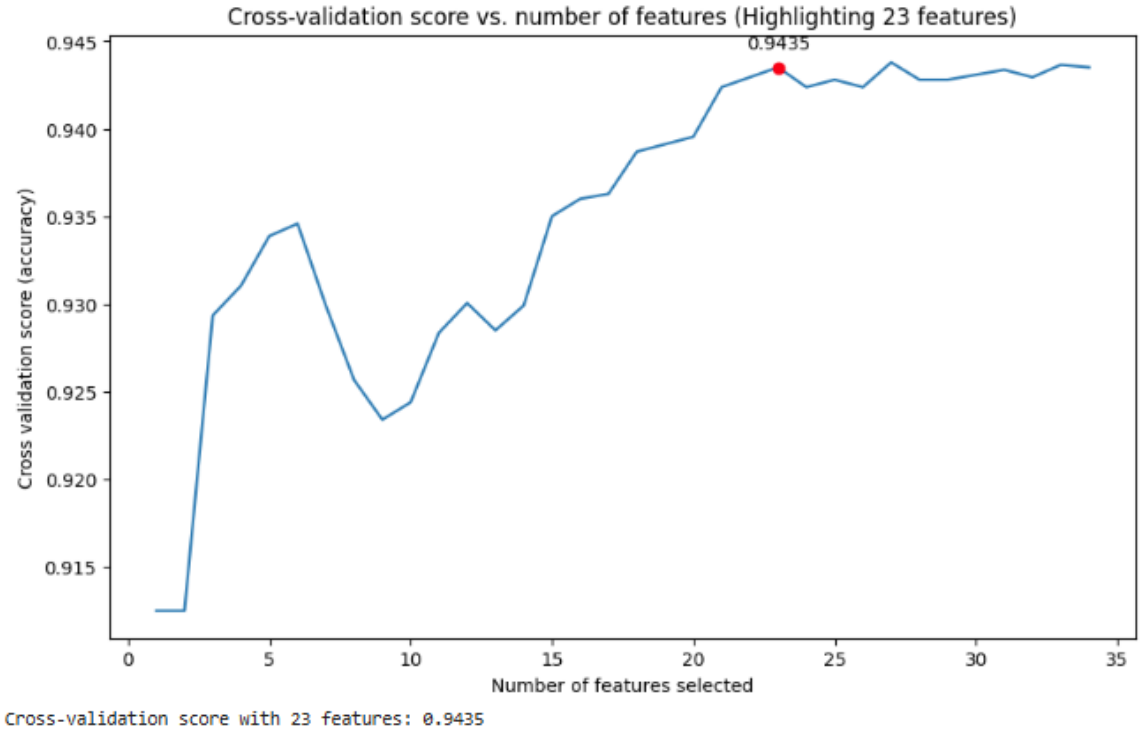

Figure 6: Recursive Feature Elimination with Cross-Validation (RFECV)

To see how model accuracy changed with different numbers of selected features, a performance curve was plotted. The above picture, which precisely indicates the point corresponding to 23 features on that curve, graphically depicts a model's performance (using cross-validation) as the number of features varies over the RFE process. This method ensures the model is both efficient and generalizable by removing noise and duplication (via PCA) and keeping only the most valuable variables (with RFE). The identical transformation process is used on the test data, ensuring consistent pretreatment and model deployment preparedness.

3.7 Model Training and Hyperparameter Tuning

Model creation is the next stage after data preparation. Since the goal of this study is to create a predictive model for mortality risk in emergency departments, the model has been constructed using classification algorithms. Python includes a variety of methods and algorithms. Since these algorithms have been extensively used in previous studies to create mortality predicting machine learning models, Random Forest, XGBoost, Support Vector Machines (SVM), and k-Nearest Neighbors (k-NN) have been used in this study. Because it can identify non-linear associations in the

data and produce a function that translates input variables (predictors) to the intended outputs (mortality risk), supervised machine learning is especially well-suited for this purpose.

This model was developed using a methodical, multi-phase approach that addresses class imbalance and hyperparameter optimization in machine learning. Data loading, model training, and evaluation utilizing various ways to address class imbalance, and lastly, establishing the best-performing model and its features for prospective deployment are all steps in the process of constructing a model for predicting mortality based on the data that has been provided from the outset.

The process starts from Data Preparation, where the necessary libraries for machine learning, data manipulation, and visualization are imported. The training and testing datasets (features `X` and target `Y`) are loaded from Excel files stored in Google Drive. Then, a baseline Classifier model (`model_1`) is initialized with default parameters. The baseline model is trained on the imbalanced training data. The model's performance is evaluated on the test set using a classification report, confusion matrix, and AUC-ROC curve. Precision-Recall (PR) curve analysis is performed, and an optimal threshold is determined based on maximizing the F1-score. The classification report and confusion matrix are presented again using predictions binarized at the optimal threshold.

Then, to identify the best hyperparameters, a Randomized Search Cross-Validation (`RandomizedSearchCV`) and a Grid Search Cross-Validation (`GridSearchCV`) hyperparameter tuning are set up independently in each model. A parameter distribution (`param_dist`) with a range of values for key hyperparameters, like Stratified k-fold Cross-Validation, is defined for the optimization technique. The `RandomizedSearchCV` and `GridSearchCV` objects are fitted to the training data to find the best combination of hyperparameters based on the F1-score. The best estimators (`best_`) are trained on the entire training dataset. The AUC-ROC curve, confusion matrix, and classification report are used to assess the improved model's performance on the test set. PR curve analysis is performed, and an optimal threshold is determined. Predictions binarized at the ideal threshold are used to display the classification report and confusion matrix. The whole step until now was essential for

finding out how much the model performance was improved by hyperparameter tuning changes alone before implementing any data balancing techniques.

Finally, a number of data balancing techniques were used to evaluate the impact of each one alone and in combination with hyperparameter adjustment on model performance. A cautious SMOTE, which balances by duplicating minority class samples, was applied just on the training dataset because pure oversampling techniques, like SMOTE, can raise the risk of overfitting. Other strategies included undersampling techniques like RandomUnderSampler, which reduce the majority class to achieve balance, and hybrid approaches like SMOTEENN, which combine selective undersampling of noisy or overlapping samples in the majority class with oversampling of the minority class. A straightforward and equitable comparison of the effects of each balancing strategy on predictive performance was made possible by the modeling process's adherence to the same structured steps used in the baseline model development: data preprocessing, model training, hyperparameter optimization, and evaluation using clinically relevant metrics.

3.8 Model Evaluation and Interpretation

3.8.1 Performance Metrics

Several complementary metrics, such as accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUROC), were used to thoroughly assess the modified model's performance on the test set. Recall givess priority in order to capture as many actual mortality instances as feasible, and precision assists in determining the percentage of predicted high-risk patients that were truly at risk. The F1-score guarantees a good trade-off between identifying true high-risk cases and preventing excessive false alarms. In imbalanced datasets where false negatives (missing high-risk patients) can have serious consequences, the F1-score was given specific attention because it strikes a compromise between precision and recall, making it particularly useful for mortality prediction.

The model's overall discriminatory capacity was measured by AUROC, which was independent of thresholds, and accuracy, which was interpreted cautiously because of class imbalance, was a general indicator of correct predictions. Additionally, a

precision-recall (PR) curve analysis was conducted to visualize trade-offs between precision and recall across different thresholds, and the optimal decision threshold was determined based on the maximum F1-score. This set of metrics made sure that the model's prediction performance was evaluated in a fair and clinically relevant way.

3.8.2 Overfitting Assessment

Overfitting checks were routinely carried out to make sure the created models were not just learning patterns unique to the training set but also generalized well to new data. Initially, the model's performance was contrasted between the test, validation, and training sets. Overfitting was assumed to be the cause of a significant disparity, such as excellent accuracy on the training data but subpar performance on the test data.

Second, to assess model stability across various dataset subsets, k-fold cross-validation was used during training and hyperparameter tweaking. This method lessened the possibility that a single train-test split would produce unrealistically optimistic findings.

3.8.3 Model Interpretation

To improve the model's interpretability and confirm that its predictions are consistent with accepted medical domain knowledge, SHAP (SHapley Additive Explanations) values and feature importance analysis were used. A deeper comprehension of how and why the model produces results is made possible by SHAP values, which offer a consistent, model-agnostic way to quantify the contribution of each feature to specific predictions. In clinical settings, where openness is essential for adoption and trust, this is especially crucial. To verify if the model's decision-making process accurately represents established clinical risk factors for mortality, feature importance analysis was used in conjunction with SHAP to find the most significant variables throughout the dataset. When combined, these methods guaranteed that the model's predictions were both clinically acceptable and statistically sound.

3.9 Deployment and Practical Considerations

Creating a high-performing model is only the first step; a successful deployment is essential to converting predictions into practical inputs for patient treatment. Deployment might vary from creating a straightforward, static report that summarizes model results to putting in place a fully automated system that is connected to an Electronic Medical Record (EMR) platform for real-time decision support, depending on the intended use case. For this study, the results are provided in a clinical review-ready manner together with suggestions derived from model outputs. These recommendations aim to guide healthcare providers in identifying high-risk patients earlier, prioritizing timely interventions, and ultimately improving patient outcomes.

CHAPTER 4: Results and Discussion

4.1 Data Understanding.

Any model-building process must start with an understanding of the data itself because the quality of the data greatly influences the final product. At this stage, a concise description of the original data gathered for the study was given. Listing attributes, spotting outliers, and missing numbers are all included in this explanation. Experts in the field were consulted in order to understand patient medical records. Additionally, to become familiar with the data and gain preliminary insights into the dataset, exploratory data analysis was carried out.

4.1.1 Descriptive Statistical Summary.

Following the completion of all preprocessing steps, the final dataset comprised 8,829 records and 24 features. The number of records decreased from the original 12,440 to 8,829 due to the exclusion of 3,611 entries that lacked values for the target variable. During preprocessing, the number of features initially increased from 26 to 36 as a result of generating dummy variables for categorical encoding. Subsequently, this number was reduced to 24 following feature selection using the Recursive Feature Elimination with Cross-Validation (RFECV) technique.

Among the 8,829 patients, 7,904 patients (89.5%) were alive, and 925 patients (10.5%) had died, corresponding to an overall mortality rate of 10.5% in the cohort. Both the training and testing sets preserved the class distribution, with 89.5% alive and 10.5% deceased in each subset, ensuring balanced representation for model evaluation.

The dataset "Mydf [numerical]" was used to create a statistical summary for each of the numerical variables in the "Mydf" dataset. Metrics like the mean, standard deviation, lowest and maximum values, count of non-missing values, and the 25th, 50th (median), and 75th percentiles are calculated using the "describe()" code. These figures aid in identifying general trends and problems with data quality. For instance, there is a large age distribution among the patients, with an average age of 45 years

and a standard deviation of 20. Potential outliers or data input problems are highlighted by certain variables, such as temperature (temp) and random blood sugar (rbs), which display strange values. The maximum temperature is 96°C, and the highest rbs is 8,990 mg/dL.

Table 3 = Numerical Variables Summary

	count	mean	std	min	25%	50%	75%	max
Heart Rate	8778	93.32	24.18	1	80	90	105	896
Systolic Blood Pressure	8757	121.68	104.31	0	104	116	131	9137
Diastolic Blood Pressure	8753	73.49	91.48	0	62	70	80	8364
Respiratory Rate	8764	20.53	6.69	0	18	20	21	265
Temperature	2024	36.13	2.12	0	36	36	36.5	96
Oxygene Saturation	8611	94.21	14.12	2	93	95	96	933
Random Blood Sugar	2717	207.37	207.75	0	122	161	249	8990
Age	8752	44.98	20.06	0	28	42	61	107
Waiting Time	8054	5.12	13.42	1	3	3	5	482
White Blood Cell	7387	10.08	6.26	0.1	6.13	8.8	12.52	223.63
Neutrophil	7272	75.73	81.10	0	64.5	76.6	85.9	5878
Lymphocyte	7271	18.99	43.00	0	8.2	15.1	24.8	3385
Monocyte	7232	5.65	9.99	0	3.3	5	6.9425	731
Eosinophil	6938	1.61	3.71	0	0.1	0.5	1.8	152
Basophil	6063	0.26	1.09	0	0	0.1	0.3	66
Red Blood Cell	7280	16.28	73.71	0.1	3.91	4.56	5.11	822
Hemoglobin	7390	18.51	56.03	1.5	12	14.1	15.7	3202
Mean Corpuscular Volume	7271	109.91	128.11	9.3	85.1	89.6	93.9	1227
Red Cell Distribution Width	5187	16.11	15.56	0.131	12.7	13.5	14.9	248
Platelet	5821	244.98	118.71	0	171	234	302	1275

Finally, some columns with numerical labels need to be handled carefully, such as temperature (temp), random blood sugar (rbs), and others. The temperature has an

unphysiologically high value of 96°C and is absent from the majority of entries (just 2,024 valid data). Although clinically significant, random blood sugar can have values much outside of normal limits (e.g., 8,990 mg/dL), which emphasizes the necessity of data cleansing and addressing outliers prior to analysis.

4.1.2 Univariable Analysis

Using "sns.histplot", "sns.countplot", and “sns.boxplot”the distribution plots show the data's shape and aid in determining skewness, spotting outliers, and looking for odd trends. For example, the age distribution is skewed to the right, indicating that many patients are younger. On the other hand, oxygen saturation (spo2) is left-skewed and closely clusters around the higher range (95–100%).

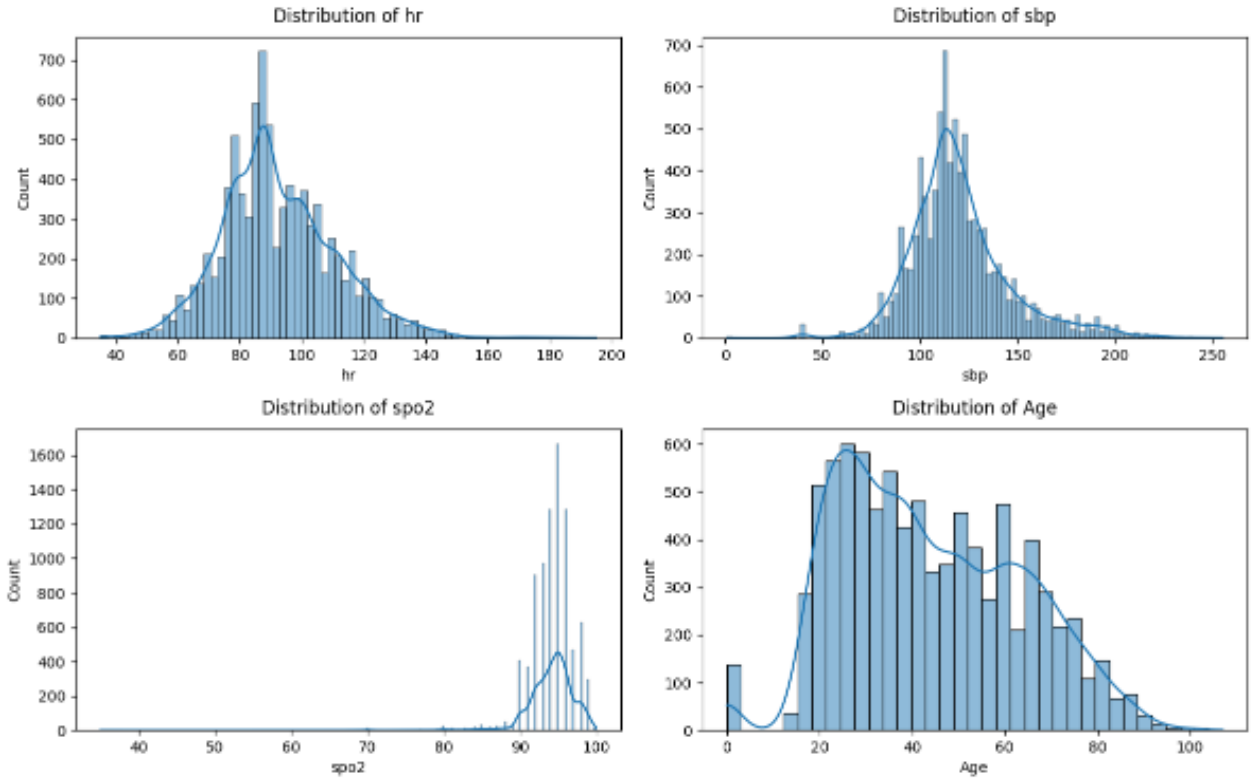

Figure 7 = Numerical Variables Bar Graph

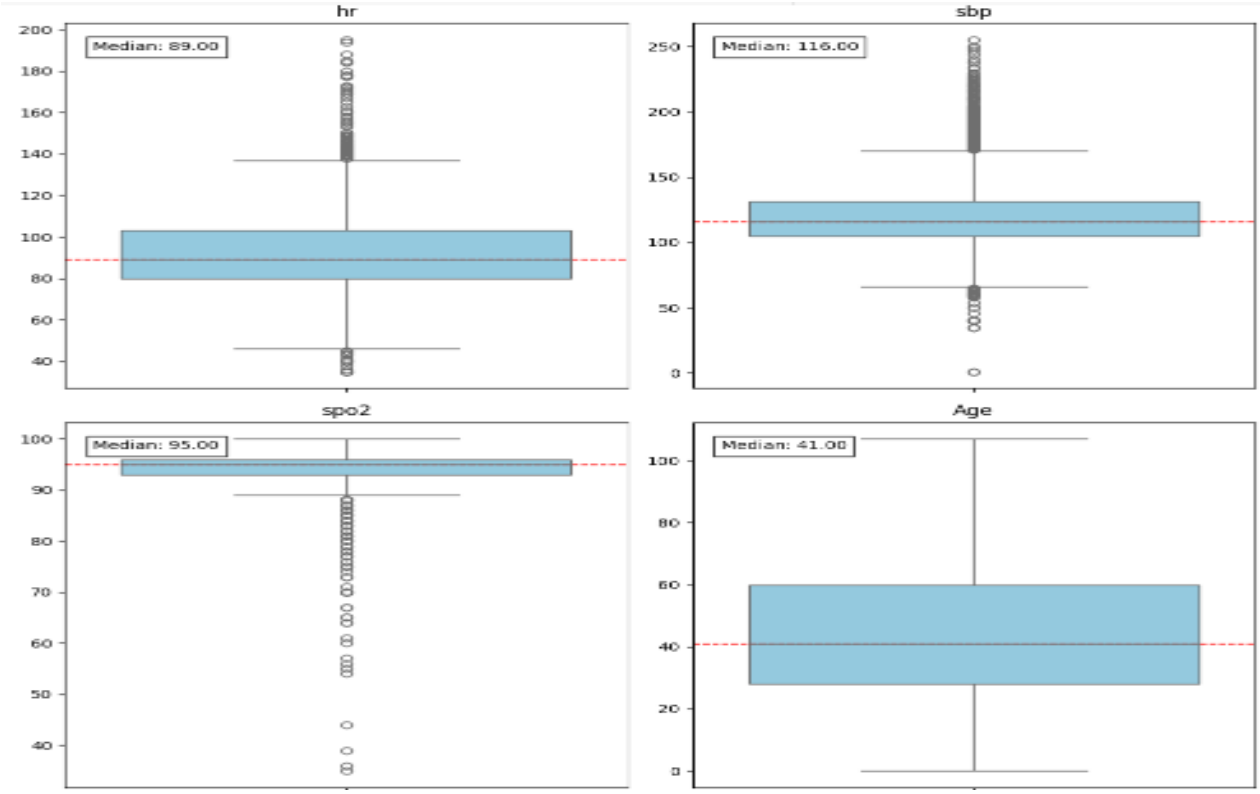

Figure 8 = Numerical Variables Boxplots

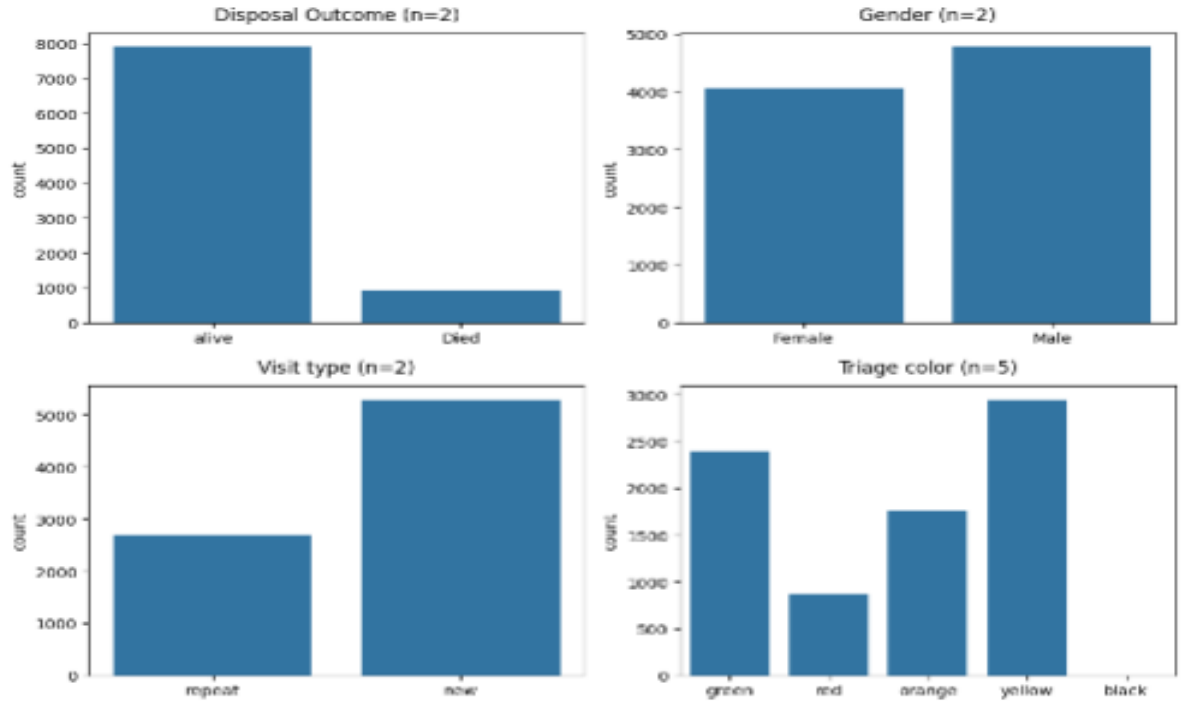

Figure 9 = Categorical Variables Bar Graph

4.1.3 Multivariable Analysis

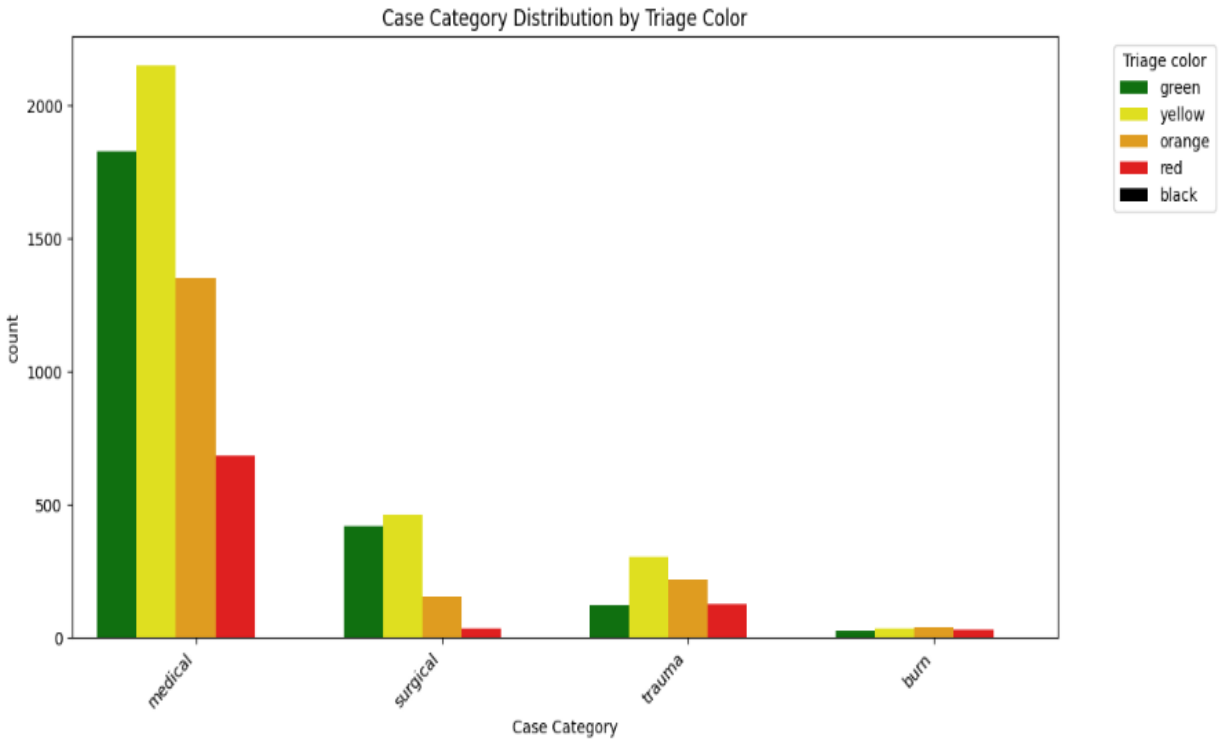

Figure 10 = Case category by triage color

The bar chart highlights how various emergency case types are categorized according to urgency by showing the distribution of case categories versus triage color. Medical cases range greatly in severity, from non-urgent to critical. The majority of cases fall under the "medical" category, which displays a wide variety of triage levels, with the largest counts in the yellow, green, and orange categories. There are many fewer cases of trauma and surgery. Although some instances still fall into the urgent and very urgent categories, most trauma cases are classified as yellow and green, indicating that they are often less severe. Similar trends are seen in surgical cases, with the majority falling into the yellow and green categories and comparatively fewer falling into the red and orange ones. This suggests that many surgical cases may be elective or less serious in nature. The green and yellow triage levels are typically assigned to burn cases, which are the least common of all categories.

By employing the chi-square (χ^2) statistic to evaluate the independence of categorical variable pairs, the heatmap offers insights into the relationships between them. A greater association between variables is indicated by higher χ^2 values, represented by

each cell in the heatmap. Significant associations ($p < 0.05$) are highlighted with color, while statistically insignificant relationships ($p > 0.05$) are marked with white cells.

The heatmap indicates a substantial correlation (high Chi-Squared value) between the case category and visit type, which makes sense given that the case category may be related to the kind of visit (e.g., new patient vs. repeat visit for a chronic ailment). There is also a substantial correlation between case category and triage color, and a statistically significant correlation between visit type and triage color. It's interesting to note that Disposal Outcome (the target variable) has a weaker but statistically significant correlation with Visit type and Triage color, but not with Case Category or Gender. This implies that gender and case category may not be as closely associated with the test's result as triage color and visit type, which may be helpful predictors.

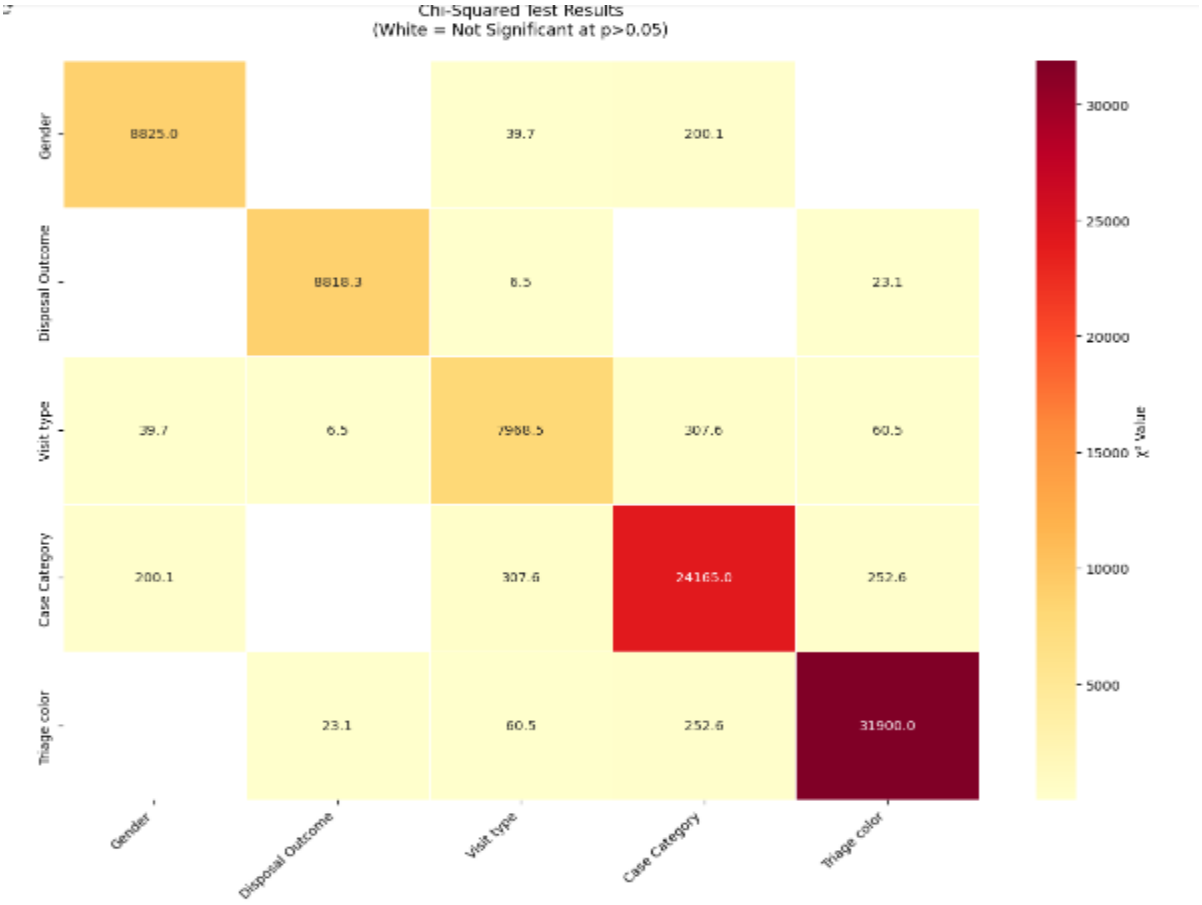

Figure 11 = Chi-square (χ^2) statistic for categorical variable

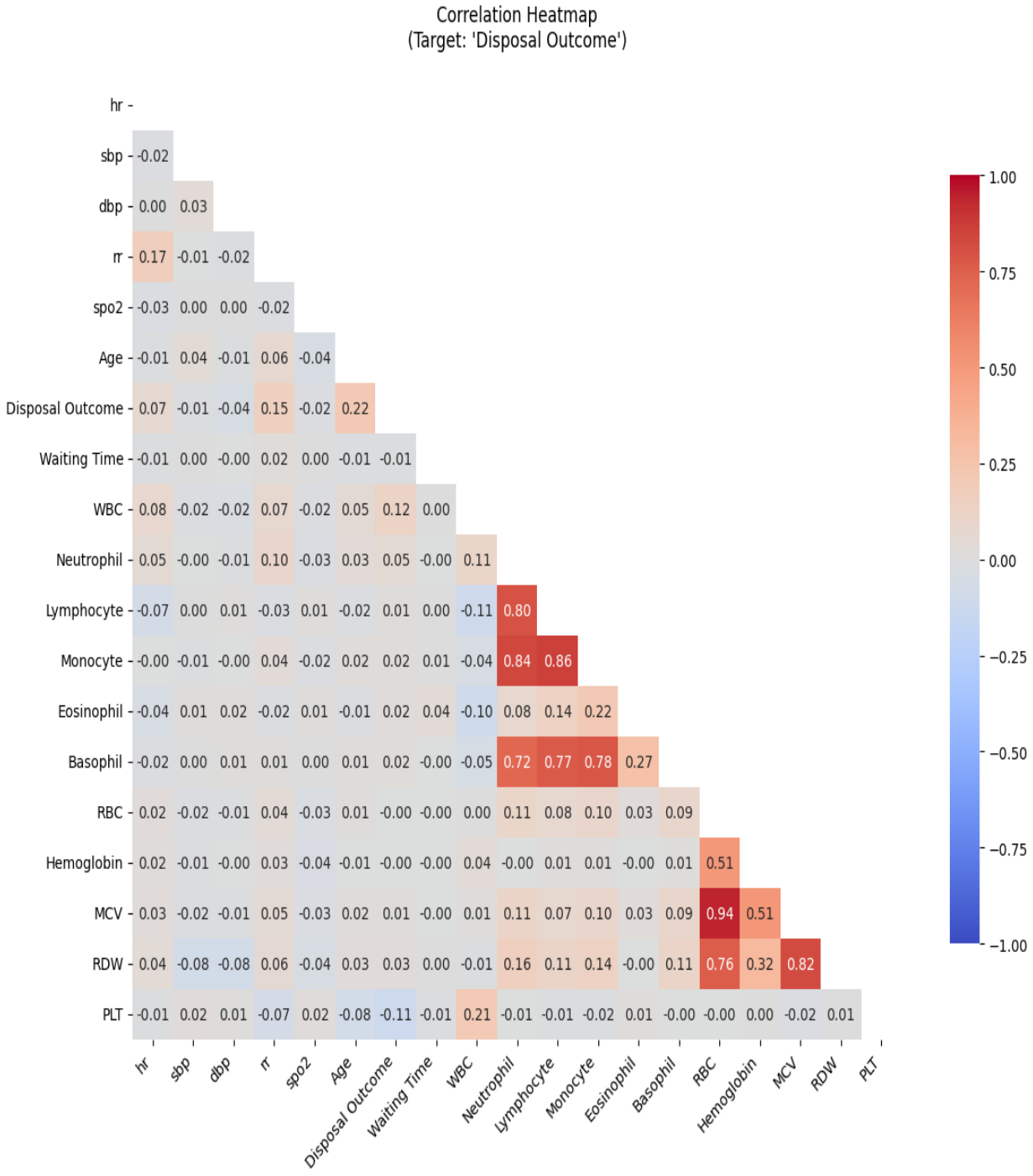

Figure 12 = Heatmap for numerical variable correlation

This correlation heatmap illustrates the pairwise relationships among numerical variables using Pearson's correlation coefficients. The color gradient ranges from dark blue to dark red, representing correlations from –1 to +1, respectively. A dark

red color indicates a strong positive correlation, meaning that as one variable increases, the other tends to increase as well. Conversely, a dark blue color represents a strong negative correlation, implying that as one variable increases, the other decreases. Lighter shades around zero suggest weak or no correlation, signifying that the variables are largely independent.

In the numerical correlation_heatmap, notable inter-variable correlations are also highlighted. The 'Disposal Outcome' study identifies several significant relationships. Age has the greatest positive connection with "Died" (around 0.22) of all the factors, suggesting that older patients have a higher chance of passing away. White blood cell count (WBC) and respiration rate (rr) also show positive associations (about 0.12 and 0.15, respectively), indicating that greater levels of these variables are linked to a higher chance of dying. The positive association with heart rate (hr) is weaker, at roughly 0.07. Platelet count (PLT), on the other hand, exhibits the largest negative association (about -0.11), suggesting that greater counts are typically linked to survival, whereas lower counts are linked to the "Died" outcome. Additionally, there are negative relationships between diastolic blood pressure (dbp) and oxygen saturation (spo2) (about -0.04 and -0.02, respectively), suggesting that lower levels are more prevalent among the deceased. Numerous factors, including waiting time, hemoglobin, systolic blood pressure (sbp), and red blood cell count (RBC), show correlations that are nearly zero, indicating a poor or nonexistent linear association with the disposal outcome. Significant inter-variable correlations are also present; for example, the sbp and dbp have a positive correlation, which is to be expected given that they are both blood pressure measurements. As a result of their common function in blood cell composition, WBC, neutrophils, lymphocytes, monocytes, eosinophils, and basophils also exhibit a variety of connections with one another.

4.2 Experimental Setup Overview

The experimental setup and findings are described in depth in this section, which is used to assess the suggested model's performance. The datasets that have finished the data preparation process are kept as Excel files that will be uploaded and used independently in each model in the subsequent steps.

The experimentation employs all chosen algorithms to systematically investigate various strategies for dealing with class imbalance in a binary classification task. It starts with a baseline model that doesn't handle imbalances and then gradually applies complex methods. While SMOTE and undersampling alter the distribution of training data, the class weighting strategy directly modifies the internal weights of the model. Oversampling and undersampling are both used in the hybrid technique. In combination with this, each technique was tested using both randomized and grid search for hyperparameter optimization.

For fair comparison, the evaluation keeps the metrics (precision, recall, F1, specificity, and ROC-AUC) the same across all approaches. Recall (Sensitivity) and F1_score have been chosen as the primary evaluation metrics for forecasting mortality risk because of their crucial significance in high-risk circumstances such as medical cases. Thus, obtaining a high recall is necessary to guarantee that the model flags the majority of true high-risk cases while maintaining a good F1 score. In contrast to accuracy, which can be deceptive in unbalanced datasets(deaths are relatively uncommon), ROC-AUC is a reliable indicator of how well the model can differentiate between those who will survive and those who won't. Applications, like mortality prediction in emergency or critical care settings, are especially well-suited for this evaluation metric combination, which promotes safer and more efficient clinical decision-making.

The experiments were executed on google colab, which is a free cloud-based platform for data analysis and machine learning, accessing CPU, GPU, and TPU RAM.

4.3 Model Experimentation

4.3.1 Experiment 1: KNN Classifier

Starting with a simple unbalanced model and moving through different class balancing strategies and hyperparameter tuning procedures, the model assesses a KNN classifier algorithm for unbalanced binary classification in several stages.

Significant differences in performance between the two outcome classes are shown by the results. The original unbalanced model achieved a precision of 0.94, a recall of

0.96, and an F1-score of 0.95, indicating good performance for the "Alive" (0) class. It performed poorly in the "Died" (1) class, though, with an F1-score of 0.53, recall of 0.47, and precision of 0.60. The metrics for the "Died" class were somewhat improved by hyperparameter tuning using RandomizedSearchCV and GridSearchCV; recall remained low at 0.41 for both approaches, while precision increased to 0.68 and 0.65, respectively.

The "Died" class's recall increased to 0.41 when SMOTE oversampling was used alone, but the precision (0.44) and F1-score (0.42) decreased. With a recall of 0.50, a precision of 0.77, and an F1-score of 0.61, the combination of SMOTE with RandomizedSearchCV produced excellent results for the minority class. Additionally, undersampling methods demonstrated promise, especially when paired with RandomizedSearchCV, which produced an F1-score of 0.62, a recall of 0.54, and a precision of 0.73 for the "Died" class.

Hybrid balancing techniques showed further advancements. Hybrid Balancing + RandomizedSearchCV maintained good metrics for the "Alive" class (precision: 0.95, recall: 0.99, F1-score: 0.97), while achieving the highest precision (0.81) and a competitive F1-score (0.64) for the "Died" class. In a similar vein, Hybrid Balancing + GridSearchCV demonstrated robust performance for the majority class (precision: 0.94, recall: 0.98, F1-score: 0.96) and strong results for the minority class (precision: 0.76, F1-score: 0.61).

Classification Report at Optimal Threshold:

	precision	recall	f1-score	support
Alive	0.95	0.99	0.97	1581
Died	0.81	0.52	0.64	185
accuracy			0.94	1766
macro avg	0.88	0.75	0.80	1766
weighted avg	0.93	0.94	0.93	1766

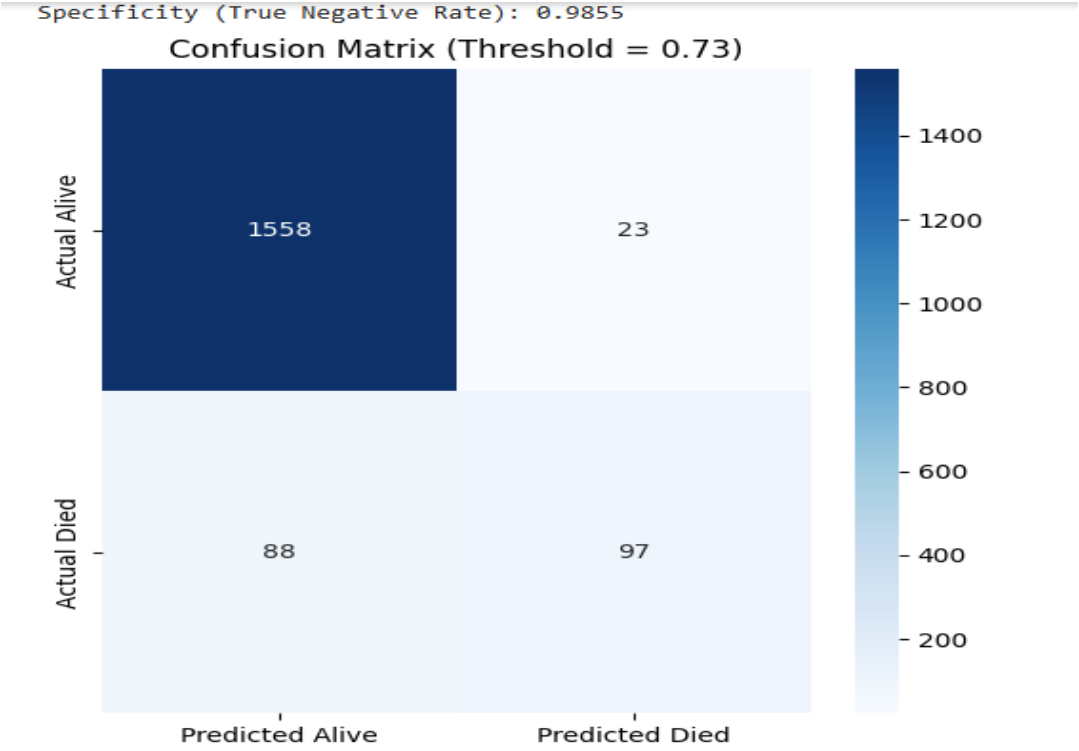

Figure 13 = KNN + Hybrid Balancing with RS Classification Report and Confusion Matrix.

The image you provided is a confusion matrix, which is a tool used to evaluate the performance of a classification model, in this case, likely related to predicting patient mortality ("Alive" or "Died") with a threshold of 0.73. The color code in this confusion matrix represents the magnitude of the values in each cell, with the intensity of the color corresponding to the number of instances. The Dark Blue Color: Indicates the highest values. In this matrix, the darkest blue is in the top-left cell (1558), representing the number of patients correctly predicted as "Alive" who were actually "Alive" (True Negatives). The Light Blue to White Color: Represents progressively lower values. The top-right cell (23) and bottom-left cell (88) are lighter, showing fewer instances of misclassifications (False Positives and False Negatives, respectively). White or Near-White Color: Indicates the lowest values or the absence of data in that cell. The bottom-right cell (97) is lighter, showing the number of patients correctly predicted as "Died" who actually "Died" (True Positives). Color Gradient Scale: The sidebar on the right (ranging from -200 to 1400) serves as a reference for the color intensity. Higher numbers (e.g., 1558) are darker blue, while lower numbers (e.g., 23 or 88) are lighter, with the scale likely centered around zero or adjusted to the data range.

Overall, Hybrid Balancing with RandomizedSearchCV produced the best balance between precision and recall for the minority class, outperforming other approaches in precision (0.81) while keeping a respectable recall (0.52), resulting in an F1 Score of 0.64. These findings demonstrate how well hybrid balancing strategies and hyperparameter optimization work together to improve model performance on unbalanced datasets.

4.3.2 Experiment 2: SVM Classifier

In the original imbalanced model, the SVM classifier showed near-perfect recall (0.99) and good precision (0.95) in identifying the majority "Alive" (0) class across all techniques. With a recall of 0.52 and a precision of 0.85, the minority "Died" (1) class, on the other hand, performed noticeably worse, suggesting bias in favor of the majority class. The modest discriminatory power and propensity to favor the "Alive" class were further supported by the ROC-AUC score of 0.79. Both RandomizedSearchCV and GridSearchCV marginally improved the balance during hyperparameter tuning, raising "Died" recall to 0.54–0.55 and precision to 0.78 while keeping high precision for the "Alive" class (~0.95). The minority class's ROC-AUC scores (0.78–0.80) indicated slight improvements in discriminative ability, whereas the F1-score was constant at 0.64.

SMOTE oversampling decreased "Died" recall to 0.46 and precision to 0.65, but increasing false positives. For ROC-AUC (0.78–0.79), there was no discernible improvement. Undersampling: Provided a better balance, with precision at 0.78 and "Died" recall at 0.52–0.54. With a precision of 0.81 and an F1-score of 0.62, Undersampling + GridSearchCV produced the best undersampling performance; however, recall decreased to 0.51.

Hybrid Balancing with RandomizedSearchCV obtaining the highest metrics for the "Died" class: F1-score: 0.64, ROC-AUC: 0.79, Precision: 0.82, Recall: 0.53. Additionally, this approach maintained outstanding results for the "Alive" class (precision: 0.96, recall: 0.99). Hybrid Balancing with GridSearchCV, on the other hand, fared poorly (precision: 0.72, recall: 0.44), indicating that hybrid approaches need to be careful. For the "Died" class, the Hybrid Balancing with

RandomizedSearchCV method yields good metrics. However, GridSearch without a balancing method performed better than all other techniques.

Classification Report at Optimal Threshold:

	precision	recall	f1-score	support
Alive	0.95	0.98	0.96	1581
Died	0.78	0.55	0.64	185
accuracy			0.94	1766
macro avg	0.86	0.76	0.80	1766
weighted avg	0.93	0.94	0.93	1766

Specificity (True Negative Rate): 0.9817

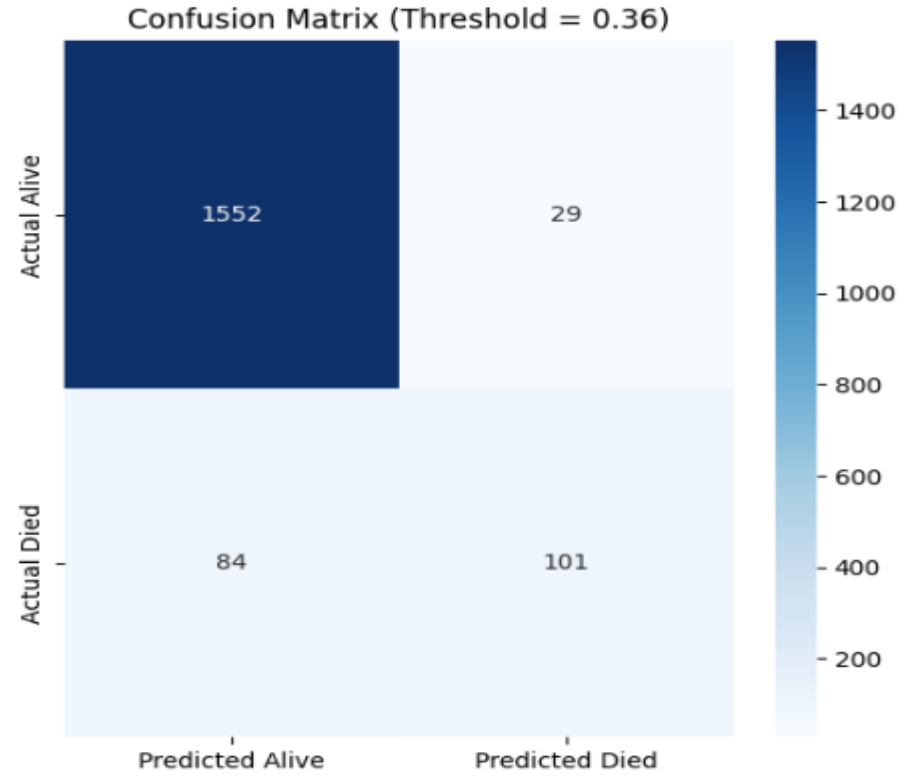

Figure 14 = RF + SMOTE with RS Classification report and Confusion matrix.

4.3.3 Experiment 3: XGBoost Classifier

In order to improve the F1 Score for the minority "Died" class while maintaining high overall performance metrics like precision and ROC-AUC, the XGBoost classifier's performance was evaluated using a binary classification task ("Alive" vs. "Died"). The effects of data balancing techniques and hyperparameter tuning were also

investigated. The greatest overall performance was obtained by the baseline (initial) model without balancing, which obtained a precision of 0.95, a recall of 0.99, an F1-score of 0.97, a specificity of 0.99, an ROC-AUC of 0.80, and an accuracy of 0.95 for the "Alive" class. The baseline model was the most balanced performer when looking at F1-score first, followed by recall and ROC-AUC, and it obtained the best F1-score (0.68) for the "Died" class across all configurations. It also had good precision (0.90) and recall (0.55).

For the "Alive" class, hyperparameter tuning using RandomizedSearchCV and GridSearchCV produced nearly comparable performance (precision: 0.94–0.95, recall: 0.99); for the "Died" class, it produced somewhat lower precision (0.89–0.91) while maintaining recall at 0.54–0.55. These adjusted models fell a little short of the baseline in terms of overall balance, with ROC-AUC improving slightly to 0.81.

F1-scores decreased to 0.60–0.62, with ROC-AUC remaining constant at 0.80, while SMOTE oversampling slightly increased "Died" recall (0.49–0.51) but decreased precision (0.75–0.83). Similar patterns were obtained when SMOTE and tuning were combined, with recall staying restricted to 0.49–0.53 and precision ranging from 0.77 to 0.94.

Higher precision for "Died" (0.83–0.88) and recall of 0.51–0.54 were maintained by undersampling, resulting in competitive F1-scores (0.63–0.67) and ROC-AUC (0.80–0.81). The greatest recall (0.64) for "Died" was obtained via hybrid balancing with GridSearchCV, but at the cost of decreased precision (0.78) and no ROC-AUC increase.

In conclusion, when the F1-score was given priority, the baseline model and its modified variations performed best, followed by recall and ROC-AUC. The difficulty of increasing minority-class sensitivity without compromising overall predictive power was highlighted by other balancing strategies that introduced trade-offs that reduced precision or failed to significantly increase recall.

Classification Report at Optimal Threshold:

	precision	recall	f1-score	support
Alive	0.95	0.99	0.97	1581
Died	0.90	0.55	0.68	185
accuracy			0.95	1766
macro avg	0.93	0.77	0.83	1766
weighted avg	0.94	0.95	0.94	1766

Specificity (True Negative Rate): 0.9930

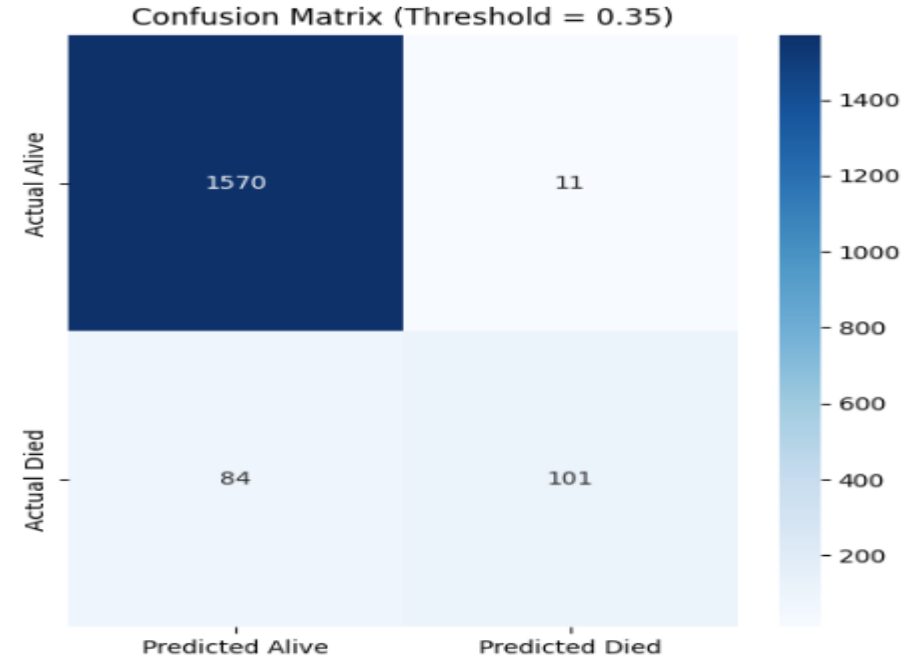

Figure 15 = XGB + Undersampling Classification report and Confusion matrix.

4.3.4. Experiment 4: RF Classifier

The main tuning parameters for the Random Forest (RF) models were imbalance handling (class_weight) and tree complexity (max_depth, min_samples_split). Cross-validation is also used to make sure that the results are accurate and not the result of chance differences in the data splits.

The findings showed a significant difference in performance between the two outcome classes. The model continuously produced good accuracy (≥0.94), recall (≥0.96), and F1 scores (≥0.95) for the "Alive" class, while maintaining robust specificity (≥0.96) in all configurations. The "Died" class, on the other hand,

performed poorly, with F1 scores ranging from 0.57 to 0.68, illustrating the difficulty of striking a balance between recall and precision at a high F1 score. The initial unbalanced model's bias toward properly classifying "Alive" cases was highlighted by its high F1 score for the majority class (0.97) and much lower one for the minority class (0.65).

Several approaches were tried to fix the imbalance: Due to poor precision (0.59), SMOTE alone produced the lowest F1 score (0.57) while significantly improving recall (0.55). For the "Died" class, RandomizedSearchCV (RS) and GridSearchCV (GS) without balancing obtained F1 scores of 0.68 and 0.66, respectively, demonstrating that hyperparameter optimization by itself produced very slight gains. Compared to SMOTE alone, SMOTE plus RS or GS produced higher F1 scores (0.61–0.65), with precision increasing to 0.86–0.90 but recall staying low (~0.46–0.52). Undersampling methods yielded F1 scores ranging from 0.64 to 0.66, indicating slight improvements over the unbalanced model, whether used alone or in conjunction with hyperparameter tuning methods.

The consistent trade-off between sensitivity and specificity was further supported by hybrid balancing techniques (such as Hybrid + GS and Hybrid + RS), which displayed F1 scores ranging from 0.61 to 0.65.

With the greatest F1 score (0.68) for the minority "Died" class and strong performance on all other measures, RandomizedSearchCV without balancing was the best-performing strategy out of all approaches that were assessed. It produced competitive accuracy (0.95), good precision (0.90), moderate F1 Score (0.68), specificity (0.99), and ROC-AUC (0.81). This approach outperformed balancing strategies like SMOTE and undersampling, which either reduced precision or failed to significantly enhance the F1 Score. According to the results, the best configuration for this RF model is hyperparameter tuning alone (using RandomizedSearchCV), which offers a better trade-off between sensitivity and specificity than class-balancing techniques.

Classification Report at Optimal Threshold:

	precision	recall	f1-score	support
Alive	0.95	0.99	0.97	1581
Died	0.90	0.55	0.68	185
accuracy			0.95	1766
macro avg	0.93	0.77	0.83	1766
weighted avg	0.94	0.95	0.94	1766

Specificity (True Negative Rate): 0.9930

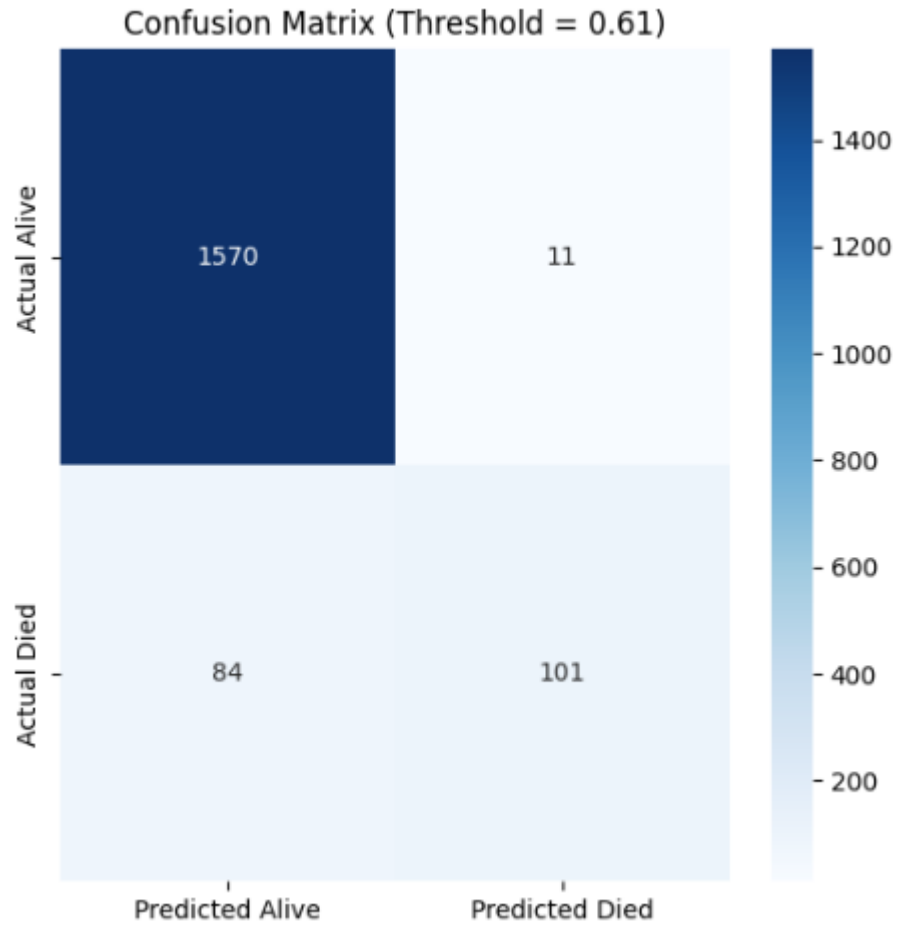

Figure 16 = RF + SMOTE with RS Classification report and Confusion matrix.

4.3.5 Overfitting Assessment Across Models

Four machine learning models are compared in the table according to their generalization gap—the discrepancy between training and test performance, which is a crucial sign of overfitting—as well as their training and test accuracy.

Table 4: Overfitting Assessment Comparison table.

Model	Configuration	Training Accuracy	Test Accuracy	Generalization Gap (Train - Test)
KNN	Hybrid + RandomSearch	99.38%	90.49%	8.89%
SVM	Unbalanced + GridSearch	93.70%	91.96%	1.74%
XGBoost	Unbalanced	95.20%	94.68%	0.52%
Random Forest	Unbalanced + RandomSearch	95.67%	94.05%	1.62%

The KNN 8.89% generalization gap is a huge difference. A classic indicator of severe overfitting is this. Despite having nearly flawless memorization of the training data, the KNN model struggles to generalize to new inputs. Despite employing a "Hybrid" strategy and RandomSearch to adjust hyperparameters, it is obviously too complicated for this purpose. The model may be overly sensitive to noise in the training set if the value of k (number of neighbors) is too low. With tiny differences between training and test accuracy (1.74%, 0.52%, and 1.62%, respectively), the other models—SVM, XGBoost, and Random Forest—all show good generalization and little overfitting. Effective methods for managing model complexity are responsible for this performance: GridSearch's ability to identify the best hyperparameters gave the SVM its resilience, XGBoost's strong built-in regularization by default allowed it to achieve superior balance even on unbalanced data, and Random Forest's ensemble nature and successful hyperparameter tuning through RandomSearch were responsible for Random Forest's dependability.

4.4 Clinically Important Features

The most important features of the two top-performing models in this study, Random Forest (RF) and XGBoost, were determined and shown for comparison in order to give a better picture of the variables as well as the model performance.

The most significant predictor of mortality risk in the RF best model was Metabolic_Case, suggesting that metabolic emergencies such as diabetic ketoacidosis, electrolyte imbalances, and acid-base disorders are highly correlated with unfavorable outcomes. Following shortly behind was Oncologic_Case, which demonstrated the elevated susceptibility of cancer patients because of immunosuppression, treatment problems, and disease progression. Other significant predictors were Mean Arterial Pressure (MAP), which was associated with both hypotension and hypertension crises, and Age_group, which showed a higher mortality risk in older patients.

Considerable weight was also assigned to respiratory rate, cardiovascular case, and gastrointestinal case, underscoring the significance of acute presentations involving major organ systems. Laboratory measurements include platelet and white blood cell counts, as well as oxygen delivery indices (PCA Oxygen Delivery Index), which helped predict mortality risk by highlighting the interaction of infection, tissue hypoxia, and physiological instability. In addition to vital signs and lab findings, the RF ranking indicates a significant impact of clinical presentation categories, providing a well-rounded and comprehensible feature set for clinical use.

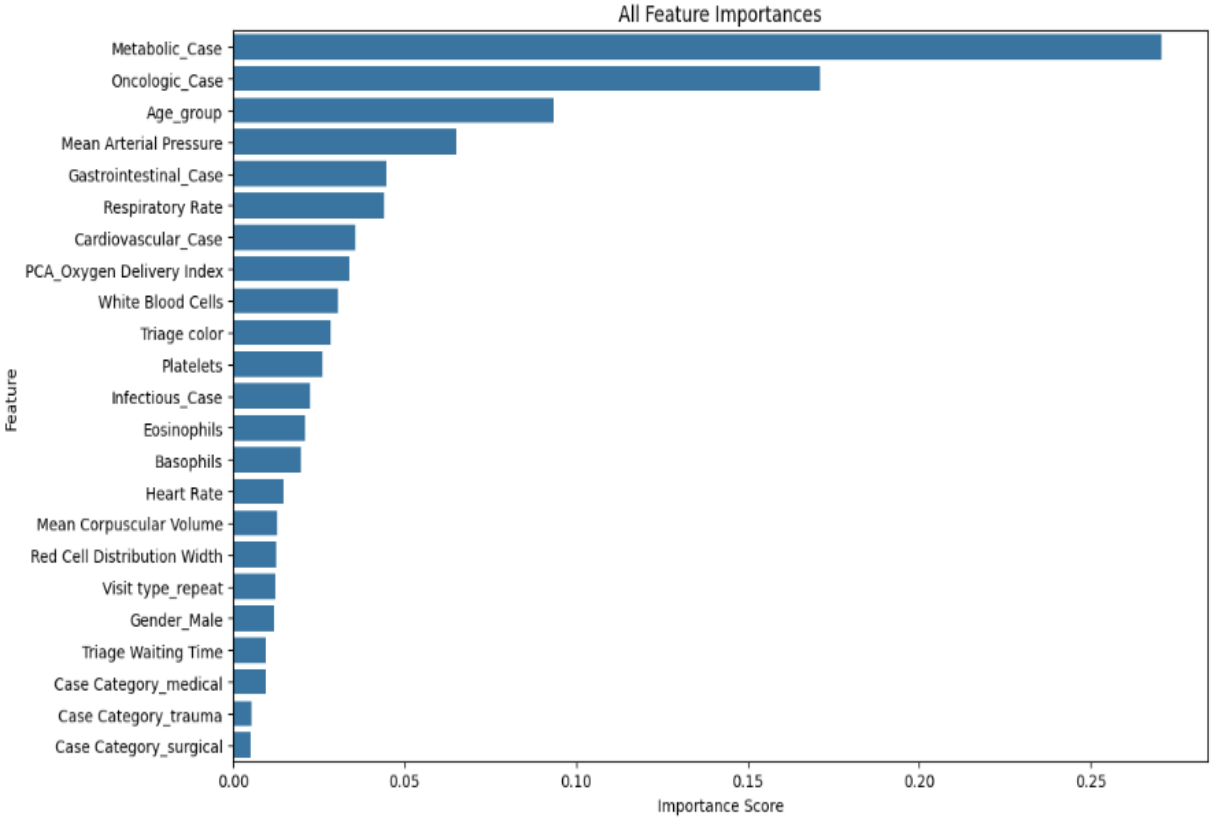

Figure 17 = RF best model base feature importances

The conclusion that metabolic emergencies have the greatest influence on mortality risk is supported by the fact that Metabolic_Case once again emerged as the top predictor in the XGBoost, the second-best model. Oncologic_Case, which continued to play a significant role in forecasting unfavorable outcomes for cancer patients, came next. Cardiovascular_Case was ranked third by XGBoost, in contrast to the RF model, highlighting the acute lethality of cardiac events such as myocardial infarction, arrhythmias, and cardiogenic shock. Age group and Mean Arterial Pressure continued to be among the leading factors, indicating the significance of patient fragility and hemodynamic stability.

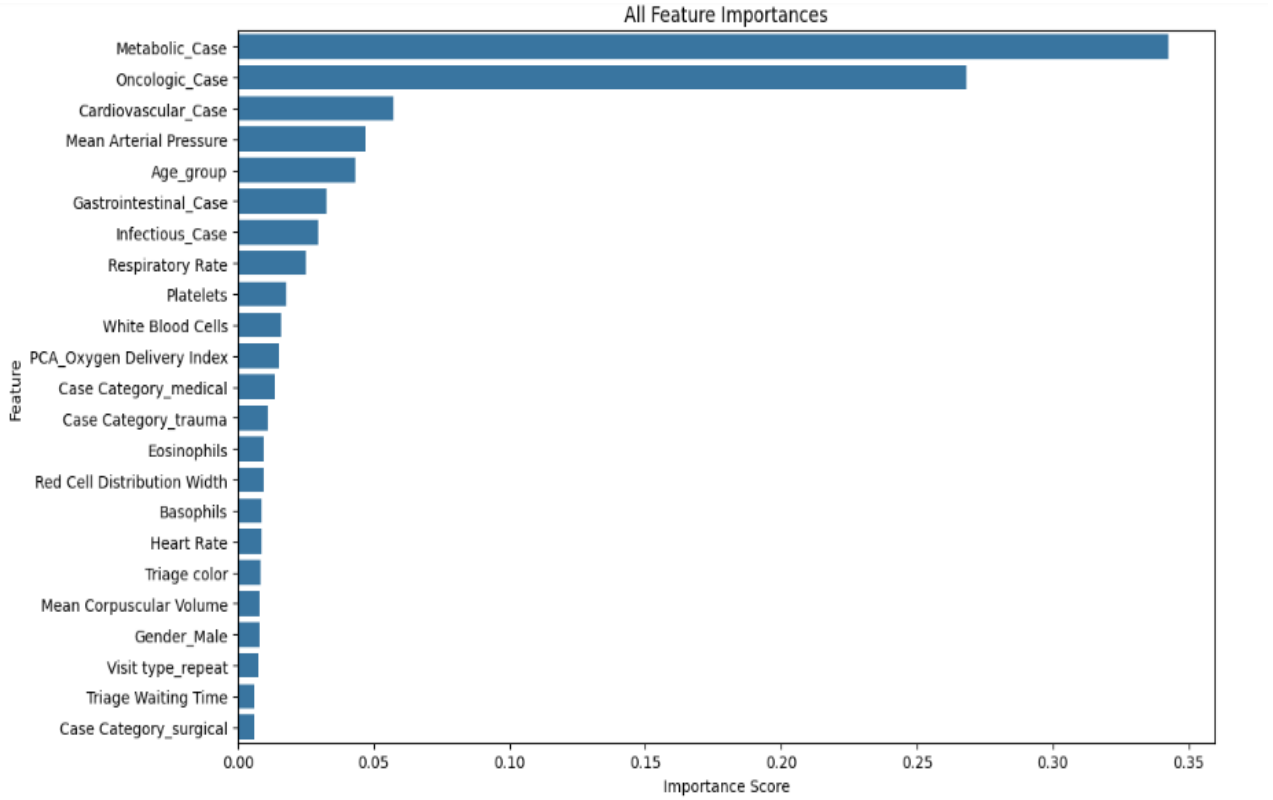

Figure 18 = XGBoost best model base feature importances

Gastrointestinal_Case, Respiratory Rate, and Infectious_Case also scored highly, capturing presentations that are frequently urgent in emergencies. Laboratory markers such as platelets, white blood cells, and oxygen delivery measures maintained their predictive importance, while triage-related variables (Case Category_medical, Case Category_trauma, and Triage color) played a more obvious role than the RF model. Overall, the XGBoost ranking highlights the significant influence of certain acute case categories, especially cardiovascular, in addition to vital signs and specific laboratory

values, indicating that this model might be more responsive to mortality patterns specific to a given illness.

4.5 Discussion of Results

In this study, the relative effectiveness of four machine learning classifiers—K-Nearest Neighbors (KNN), Random Forest (RF), Support Vector Machine (SVM), and XGBoost—for mortality risk prediction in emergency department (ED) settings was assessed. Special attention was paid to managing class imbalance between "Alive" and "Died" outcomes. Performance for the majority "Alive" class was consistently high across all models (precision and recall > 0.90). However, with initial recall scores ranging from 0.41 to 0.55, all baseline models had trouble identifying the minority "Died" class, indicating a systemic bias toward survivorship prediction.

Several balancing techniques were used to lessen this imbalance. SMOTE (Synthetic Minority Oversampling Technique) increased the rate of false positives by slightly improving minority-class recall, but frequently at the expense of decreased precision (as low as 0.44). Similarly, undersampling techniques also provided a low improvement in recall. SMOTE and undersampling hybrid approaches produced no discernible benefit and frequently worsened performance, indicating that greater complexity did not equate to improved generalization.

4.5.1 Comparative Analysis of Undersampling-Based Models

The performance of these four algorithms was evaluated using undersampling to solve class imbalance, with and without hyperparameter adjustment via RandomizedSearchCV and GridSearchCV, in a comparative analysis of undersampling-based models.

Table 5: Undersampling-based Model Performance Ranking.

Rank	Method	F1 Score	Precision	Recall	Notes
1	XGB-Undersampling	0.67	0.88	0.54	Best F1, high precision
2	RF-Undersampling+RS	0.66	0.83	0.55	Balanced precision-recall
3	RF-Undersampling	0.64	0.86	0.51	High precision, low recall
4	SVM-Undersampling	0.63	0.78	0.54	Moderate performance
5	KNN-Undersampling+RS	0.62	0.73	0.54	Tuning improved F1 over baseline
6	XGB-Undersampling+RS	0.63	0.76	0.54	Tuning reduced F1 vs. baseline

Predicting the minority "Died" class—a crucial component of clinical risk stratification—was the focus of the evaluation. With the highest F1 score (0.67) and precision (0.88), XGBoost without tuning was the best performer; nevertheless, its recall was only 0.54, which meant that it missed roughly 46% of actual "Died" occurrences. With F1 scores between 0.64 and 0.66 and precision between 0.83 and 0.88, RF models trailed closely behind XGBoost, where tuning resulted in modest improvements in recall but no change in F1.

With F1 values ranging from 0.57 to 0.63, SVM and KNN had moderate performance; however, KNN had the lowest precision (0.59–0.73), and adjusting only slightly enhanced outcomes. The trade-off between accuracy and recall was noticeable across all approaches: high precision (0.78–0.88) decreased false alarms, but recall remained below 0.55, which limited the models' use for screening but increased their dependability for verifying high-risk patients. While RF-Undersampling with RS hyperparameter tuning provides a superior precision-recall balance, XGB-Undersampling is better for deployment when reducing false positives is the top concern. All models' persistently low recall value points to the necessity for alternate

approaches, such as increased feature engineering, cost-sensitive learning, or hybrid sampling, to increase sensitivity without significantly compromising precision.

4.5.2 Comparative Analysis of SMOTE-Based Models

Table 6: SMOTE-based Model Performance Ranking.

Rank	Method	F1 Score	Precision	Recall	Key Insight
1	RF-SMOTE+GS	0.65	0.86	0.52	Best F1, robust precision
2	XGB-SMOTE+RS	0.62	0.74	0.53	Balanced performance
3	XGB-SMOTE Only	0.60	0.75	0.51	Strong baseline
4	RF-SMOTE+RS	0.61	0.90	0.46	High precision, low recall
5	SVM-SMOTE+RS	0.55	0.65	0.48	Moderate trade-off
6	KNN-SMOTE Only	0.53	0.44	0.41	Worst precision-recall balance

These four algorithms, using the Synthetic Minority Oversampling Technique (SMOTE) to address class imbalance, were evaluated in the comparative analysis of SMOTE-based models, both with and without hyperparameter tuning via RandomizedSearchCV and GridSearchCV. While recall was still limited at 0.52, meaning that over half of real "Died" cases were missed, RF in conjunction with GridSearchCV produced the best F1 score (0.65) and strong precision (0.86) among the tested models. With balanced precision (~0.74–0.75) and recall (~0.51–0.53), XGBoost models showed consistent performance (F1 around 0.60–0.62), with RandomizedSearchCV offering the best overall balance.

In comparison to RF and XGBoost, SVM demonstrated moderate precision and recall, but lower F1 scores (0.53–0.55). With the lowest precision (0.44) and recall (0.41), KNN continuously performed poorly, and adjusting made it even less effective. Although the excellent precision of RF and XGBoost in clinical settings makes them

appropriate for verifying high-risk patients with fewer false alarms, their low recall across all models restricts their applicability for screening, as 47–54% of "Died" cases were overlooked. SMOTE enhanced precision for RF and XGBoost models; however, undersampling performed marginally better than SMOTE in F1 score and recall (0.67 and 0.54, respectively, for XGBoost with undersampling). When it comes to precision, RF-SMOTE+GS is the best option overall, but XGB-SMOTE+RS provides a superior balance between precision and recall.

4.5.2 Comparative Analysis of Unbalanced-Based Models

Table 7: Unbalanced-based Model Performance Ranking.

Rank	Method	F1 Score	Precision	Recall	Key Insight
1	XGB-W/O Balancing+RS	0.68	0.89	0.55	Best F1, high precision
2	RF-W/O Balancing+RS	0.68	0.90	0.55	Matches XGBoost
3	XGB-W/O Balancing+GS	0.68	0.91	0.54	Slightly higher precision
4	RF-W/O Balancing	0.65	0.81	0.55	Strong baseline
5	SVM-W/O Balancing	0.64	0.85	0.52	Competitive but lower recall
6	KNN-W/O Balancing	0.53	0.60	0.47	Worst performer

Again, these four algorithms were compared without the use of class-balancing strategies. Baseline models and those adjusted with RandomizedSearchCV or GridSearchCV were assessed. According to the results, the tuned XGBoost and RF models missed around 45% of real "Died" occurrences, achieving the maximum F1 score (0.68) with precision above 0.89 and recall plateauing at 0.55. While KNN severely lagged (F1=0.53) with low recall (0.41–0.47) and precision (0.60–0.68), and tuning further deteriorated its performance, SVM performed decently (F1=0.64, precision up to 0.85), but shared the recall constraint. While hyperparameter adjustment had little effect on recall, it mostly increased precision.

Although their limited recall reduces their screening efficacy, clinically, XGBoost and RF's high precision indicates they are dependable for verifying high-risk patients with fewer false alarms. The top unbalanced models scored marginally better in F1 score (0.68 vs. 0.65–0.67) than models trained with undersampling or SMOTE, most likely because of the preserved data integrity, but recall was still low for all approaches. This emphasizes the necessity of using several approaches, like cost-sensitive learning or hybrid sampling, to increase sensitivity without compromising accuracy.

4.5.4 Comparative Analysis of Best Model Across Algorithms

When examining the models that performed the best overall in terms of precision and recall, Random Forest (RF) and XGBoost were the best, which both had the greatest F1-scores (0.68) for the minority "Died" class. Both RF and XGBoost had the same results when precision and recall were compared (precision = 0.90, recall = 0.55), indicating that they reduced false alarms while collecting slightly more than half of the real deaths.

Table 8 = Comparative Performance Evaluation Across the Best Model of the Algorithms

Comparison Metric	KNN (Hybrid + RandomSearch)	SVM (Unbalanced + GridSearch)	XGB (Unbalanced)	RF (Unbalanced + RandomSearch)
Precision	0.81	0.78	0.90	0.90
Recall	0.52	0.55	0.55	0.55
F1-Score	0.64	0.64	0.68	0.68
ROC-AUC	0.79	0.80	0.80	0.81
Accuracy	0.94	0.94	0.95	0.95

Nonetheless, RF demonstrated great discriminative performance with a little better ROC-AUC (0.81 vs. 0.80) and equaled XGBoost's high accuracy (0.95). Crucially,

RF's feature importance rankings are typically more consistent and comprehensible than XGBoost's, which is a significant benefit in a clinical context where clear decision-making and explanation of model results are crucial.

This interpretability enhances confidence and makes incorporation into clinical workflows easier by enabling physicians to comprehend which patient characteristics influence mortality risk estimations. When model explainability is a top concern, RF—with RandomizedSearch tuning—offers competitive performance and more lucid insights than XGBoost, which didn't require any hyperparameter adjustment to function at its best.

4.5.5 RF RandomizedSearch Optimal Threshold Analysis

The probability threshold at which the model maximizes the F1-score, which treats precision and recall equally, and strikes the optimal balance between the two, is 0.61. The precision of the model is 90% at this threshold. This indicates that the model is 9 out of 10 accurate when it forecasts a patient's death. For instance, roughly 90 patients face a high risk of death if the program flags 100 individuals as such. In clinical contexts, this high precision is useful because it lowers false positives, or individuals who are mistakenly classified as high risk. For patients who otherwise survive, this helps save needless harsh treatments like lengthy intensive care unit stays or invasive operations, which can be expensive and distressing.

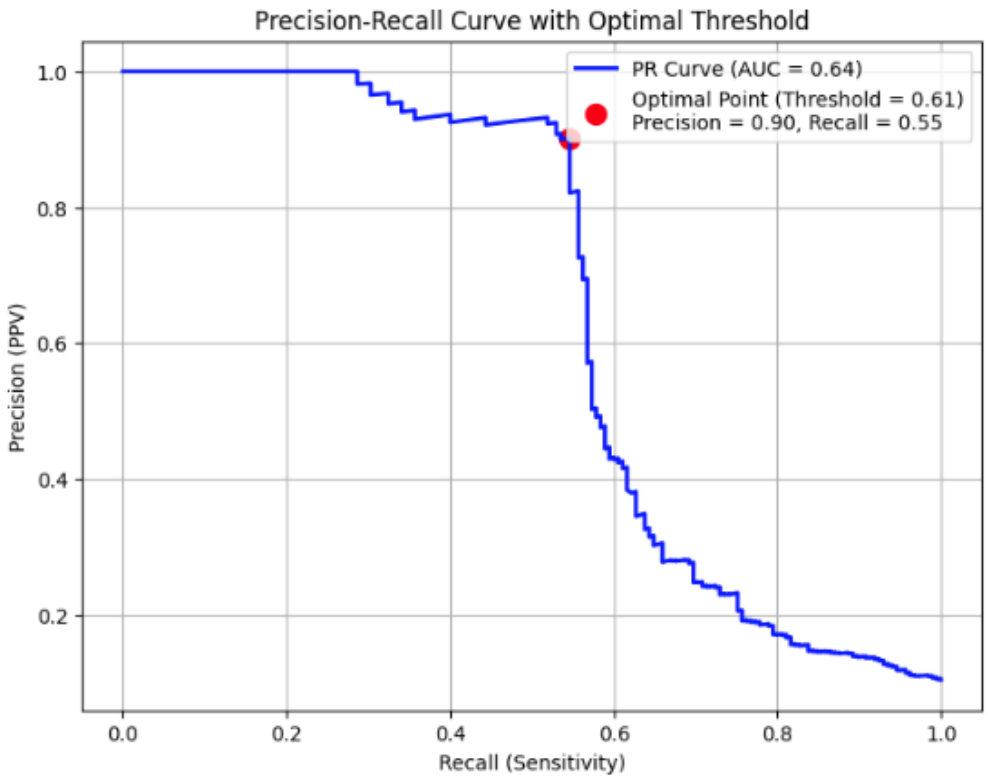

Figure 19 = Precision-Recall Curve for the Best Model.

However, the algorithm accurately detects slightly more than half of all real fatalities, with a recall of only 55%. In other words, the model fails to account for 45% of patients who ultimately pass away. This is a serious drawback since the patients who were overlooked might not get life-saving treatments in a timely manner, such as intensive monitoring or emergency surgery. For instance, worse outcomes may result from a failure to identify at-risk patients early in sepsis or acute heart failure instances. A reasonable ability to balance precision and recall is shown by the Precision-Recall Area Under the Curve (PR AUC), which is 0.64. The difficulty of forecasting uncommon occurrences, such as death, in a highly unbalanced dataset—where the majority of patients survive—is reflected in this score. PR AUC is a more pertinent statistic in this case since it concentrates on the model's performance for the minority (death) class, in contrast to ROC-AUC.

Clinically, selecting this threshold represents a trade-off between recall (capturing every real instance) and precision (preventing false alarms). This method works well in resource-constrained environments where overtreatment could be dangerous or

expensive. Avoiding needless admissions due to false positive results, for example, can enable a hospital with a limited number of intensive care unit beds to maintain capacity for patients who need it.

4.5.6 Performance Comparison with Models from Literature

Thus, having accomplished the first and third research goals, which were: (1st) to develop the best ED mortality risk prediction model and (3rd) to evaluate several machine learning algorithms in predicting mortality, the study now contextualizes its contributions by contrasting its top-performing model with previous studies. Like, a study conducted at Chang Gung Memorial Hospital in Taiwan focused particularly on predicting 2-day mortality in patients with thrombocytopenic episodes is one comparative study[33].

The RF model created in this study performs noticeably better than the Chang Gung Random Forest model when compared based on the majority of clinically significant measures. Its precision is significantly higher (0.90 vs. 0.46), meaning that only 10% of its positive predictions are wrong, while the Chang Gung model's is above 50%. In addition, its recall is significantly higher (0.55 vs. 0.15), detecting more than half of true positive instances, while the Chang Gung model misses 85% of them. A significantly higher F1-score (0.68 vs. 0.23) is the consequence of this precision and recall advantage, which shows a better balance between detecting positive cases and preventing false alarms. Both models accurately identify almost all negative occurrences, with an equally high specificity of 0.99. Although the Chang Gung model's ROC-AUC is marginally higher (0.85 vs. 0.81), this advantage is probably due to the model's performance on the negative class and does not make up for its low sensitivity and precision.

Considering all things, this study's RF model is more practically applicable and dependable, particularly in situations when failing to identify high-risk patients or producing an excessive number of false positives could have detrimental effects. Despite achieving a little better ROC-AUC (0.85 vs. 0.81), which indicates slightly better class separation, their model's practical relevance is limited due to the significantly poorer recall. This research model was created using a larger ED

population and performs better in a variety of scenarios, and it is more useful in terms of clinical application. While the much-improved recall implies fewer critical patients are overlooked, the increased precision guarantees that alarms are reliable. In the future, adding domain-specific characteristics dynamics might improve this research model's discriminatory ability without lowering its sensitivity.

Table 9 = Performance Comparison: Optimized RF Model vs. Chang Gung Memorial Hospital RF Model.

Comparison Metric	This research RF Model	Chang Gung RF model [33]	Interpretation
Precision	0.90	0.46	This research model correctly identifies 90% of true positives (vs. 46% in the Taiwan study), meaning fewer false alarms.
Recall (Sensitivity)	0.55	0.15	This research model detects 55% of actual mortality cases, significantly outperforming the Taiwan model (15%), which misses many true positives.
F1-Score	0.68	0.23	This research model achieves a better balance between precision and recall, indicating stronger overall performance.
Specificity	0.99	0.99	Both models excel in ruling out non-fatal cases (99% specificity).
ROC-AUC	0.81	0.84	The Taiwan model has a marginally better discriminatory power (AUC), suggesting it differentiates between classes slightly better, despite lower recall.

In another emergency department (ED) mortality prediction study done at Tehran Hospital in Iran, the Tehran model significantly outperformed this research model in key metrics: it obtained a precision of 0.94 and a recall of 0.92, as opposed to this research model's 0.90 and 0.55, respectively. This suggests that, in contrast to this research model, which may miss more than 45% of true positives, Tehran's model not only reduces false positives but also captures the great majority of actual mortality cases. Their F1-score of 0.93, which shows a better balance between recall and precision, is also significantly higher than that of this research model (0.68). However, this research model showed a moderate ROC-AUC of 0.81 and strong specificity (0.99), indicating excellent ability to rule out non-fatal cases[32].

This performance discrepancy could be explained by several things. More sophisticated feature engineering probably helped Tehran's model, maybe adding dynamic clinical factors like lab levels, which were not used in this research, or the vital sign they used was temporal patterns, not a one-time vital sign like this research. It's also possible that their data, which came from a single category of patients with CVD, was more homogeneous than this research model, which was trained on a more diverse set of disease categories with unbalanced dataset, although lately it have balanced used different balancing techniques. It's also possible that their model was overfit to their specific dataset, but this would need to be confirmed by outside validation.

Table 10 = Performance Comparison: Optimized RF Model vs. Tehran Hospital RF Model

Comparison Metric	This research RF Model	Tehran Hospital RF Model[32]	Interpretation
Accuracy	0.95	0.93	Each model demonstrates near-perfect classification performance.
Precision	0.90	0.94	Tehran’s model has fewer false positives (only 6% incorrect mortality alerts vs. 22% in this research

			model).
Recall (Sensitivity)	0.55	0.92	Tehran's model captures 92% of true mortality cases, significantly outperforming this research model (59%).
F1-Score	0.68	0.93	Indicates Tehran's model has a far better balance between precision and recall.

The poorer recall of this research model may limit its applicability for screening applications, even though its high specificity can still be helpful for targeted resource utilization.

4.5.7 Feature Importance Comparison

The second key objective of this research was to identify predictors of emergency department (ED) mortality risk, and the study has highlighted several critical contributing factors. In addition, the feature rankings also provide the foundation for assessing how closely or differently the results of another published studies match or differ from them. A more thorough comparison is made possible by presenting the feature sets of both models, the first and second performer (RF with RS and XGBoost), which show the consistency of several clinically significant predictors as well as model-specific variations that can affect their suitability in various emergency department scenarios.

Metabolic_Case and Oncologic_Case are the two most important predictors of mortality according to both the Random Forest (RF) and XGBoost models. This is clinically congruent with the serious, frequently fatal nature of both diseases in ED. The models' subsequent ranks, however, differ. In the top tier, RF highlights Age_group and Mean Arterial Pressure, emphasizing hemodynamic stability and patient fragility as key factors in outcome prediction. Given that age and blood pressure are well -recognized mortality factors, this strategy is in line with

conventional emergency and critical care concepts.Cardiovascular_Case, on the other hand, is ranked significantly higher by XGBoost than by RF, indicating a greater sensitivity to acute heart events. Additionally, RF allocates relevance more uniformly across case types, vital signs, and laboratory data, whereas XGBoost provides slightly greater weight to clinical presentation categories (such as Case Category_medical and Case Category_trauma).

Both models are strong from a purely predictive perspective; however, the Random Forest model has a small advantage when it comes to matching clinically significant variables that are generally relevant across patient populations. This is due to the fact that it provides a comprehensive and comprehensible risk profile by balancing the many acute illness categories with basic physiological markers (Age group, MAP, Respiratory Rate, WBC, Platelets). The feature set of RF is more directly applicable to bedside decision-making since it reflects the multidimensional assessment that physicians employ in actual triage.

There is a noticeable difference between the scope and clinical applicability of this research RF model and the feature importances stated in a study carried out in Tehran, Iran, and Chang Gung Memorial Hospital. While all models highlight certain overlapping hematological markers such as platelet count, hemoglobin, white blood cell count, red blood cell indices (RDW, MCV) and glucose, this research RF model extends beyond laboratory values to integrate a broader set of clinically relevant variables—including comorbid conditions (e.g., metabolic and oncologic presentations) and vital signs such as mean arterial pressure and respiratory rate. This research's RF model is especially well-suited for general emergency department (ED) mortality prediction because it incorporates patient history, physiological characteristics, and laboratory results to provide a more complete picture of patient state.

Table 11 = Feature Importance Comparison with Different Studies

Content	This research Model	Tehran Model[32]	Taiwan Model[33]
Vital Signs	Mean Arterial Pressure (MAP), Respiratory Rate (RR)	Ejection Fraction (EF)	None reported
Comorbidities / Presentation Type	Metabolic cases, Oncologic cases	Cardiac conditions (via EF)	None reported
Age	Included	Not included	Included
Laboratory – Hematology	Platelet count, Hemoglobin, RBC indices (MCV, RDW), WBC count	Platelet count, Hemoglobin	WBC count, Platelet count, RDW, MCV, MCH, Blast cells, Abnormal lymphocytes
Laboratory – Biochemistry	Not included	Urea, Creatinine	Creatinine
Overall, Scope	Broad: combines vital signs, comorbidities, and laboratory data	Moderate: mainly cardiac and hematological + some biochemical	Narrow: primarily hematological and biochemical
Population Focus	General ED patients (mixed presentations)	Cardiac-oriented ED patients	Mixed ED but hematology-heavy focus

The Tehran RF model, on the other hand, is more specifically focused on hematological and cardiac-related indicators, such as urea and ejection fraction (EF), which may be less relevant for broader prognostic tasks in varied ED populations, albeit being useful for specific patient categories. The Chang Gung model, on the contrary, is primarily concerned with hematopathological indicators, such as aberrant lymphocytes and blast cells, which are very useful for identifying particular blood illnesses but less instructive for more general prognostic tasks in diverse ED

populations. Because acute deterioration and mortality in the ED are multifactorial, this research's RF model feature set more closely matches this reality, allowing it to be applied to a greater variety of clinical circumstances where prompt and precise risk stratification is essential.

4.5.8 SHAP Values Interpretation

With features arranged from most to least significant along the y-axis, the SHAP Beeswarm Plot offers a thorough understanding of how different clinical characteristics affect mortality risk. An individual patient is represented by each dot on the plot, and the x-axis position shows whether the feature raises (right) or lowers (left) the patient's mortality risk. A thorough analysis of how various medical problems and physiological data affecting the overall risk profile is made possible by this representation.

To begin with, Metabolic_Case is the most important element and the best indicator of elevated mortality risk. Almost all the dots are on the right, mostly in red, suggesting that individuals with metabolic diseases (Metabolic_Case = 1) are at significantly higher risk. With dots grouped on the right and primarily red, Oncologic_Case, the second most significant attribute, likewise exhibits a substantial positive influence, indicating that cancer patients (Oncologic_Case = 1) similarly face a markedly elevated chance of death. The third important factor, age_group, has a gradient pattern, with the majority of the dots pointing to the right, indicating that younger ages (blue) correlate with lower risk and older age groups (red) with higher risk.

A more complicated picture with conflicting effects is shown by mean arterial pressure. It displays a bimodal distribution, where high values (red) may reduce risk, perhaps as a result of hypertension's protective effects in some situations, and low values (blue) increased risk, which is consistent with the risks of hypotension. Given that their dots are primarily on the positive side, which suggests that these conditions increase mortality risk, Gastrointestinal Case and Infectious Case, which are of moderate importance, generally increase risk.

The fact that respiratory rate exhibits an even distribution on both sides suggests that the impact it has on risk varies according to the number. On the other hand, with left-skewed distributions and primarily negative SHAP values, white blood cells, platelets, and triage color tend to reduce risk. Certain triage colors (presumably signifying less urgency) and higher white blood cell and platelet counts seem to be protective factors. Remarkably, despite their clinical significance, Cardiovascular_Case and Heart Rate likewise exhibit a primarily negative effect on risk, with the dots pointing to the left. This surprising result implies that cardiovascular diseases and elevated heart rates might be linked to a lower chance of death in this model, which calls for more research.

4.6 Prototype Mobile Application for Mortality Risk Prediction

To show the potential for real-world deployment of the mortality risk prediction model, a prototype mobile application was constructed as an extension of the Random Forest (RF) model produced in this research. This application's goal is to give healthcare workers a useful tool that enables them to rapidly evaluate patient mortality risk at the emergency department (ED). The trained RF model, which was exported into the ONNX format for mobile compatibility, is integrated into the application, which was created with Android Studio using the Java language. The application's user interface was also created using XML layouts. Vital signs, laboratory findings, age, and triage information are among the clinical and demographic characteristics that clinicians might enter into the predictive model. The application generates an estimated mortality risk score, represented as a likelihood percentage, after processing the inputs through the embedded RF model.

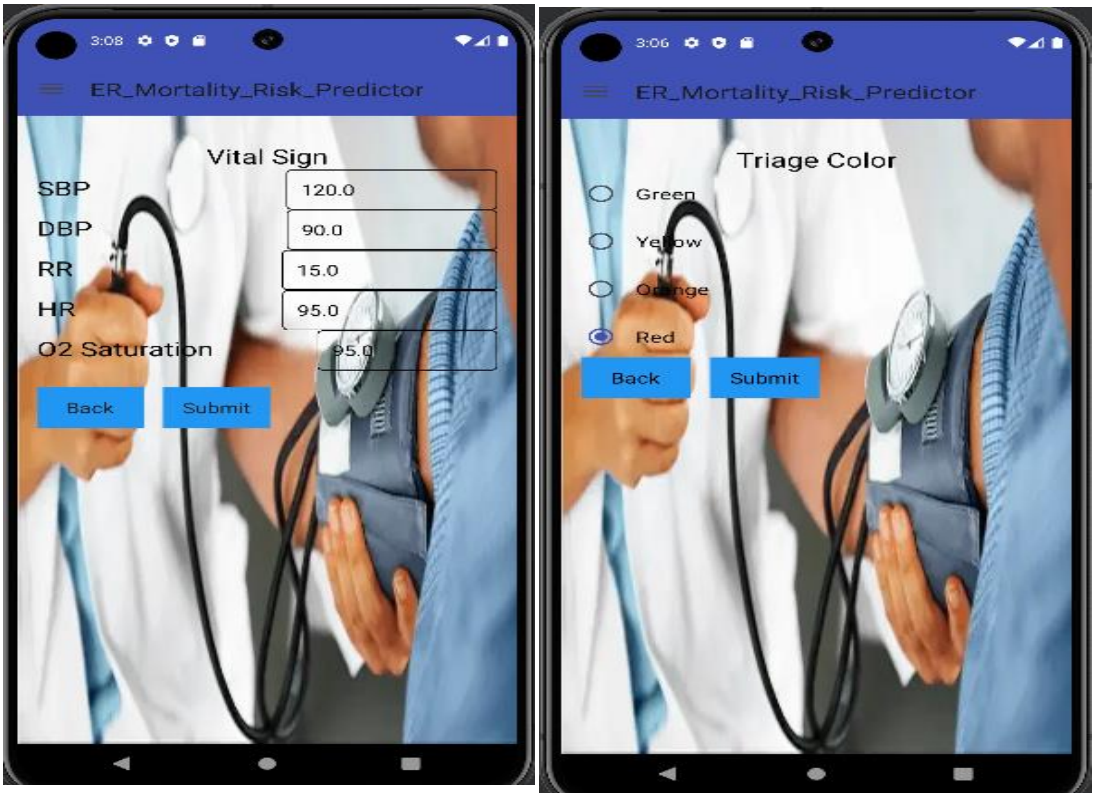

Figure 20: Mobile application input interface for patient data entry.

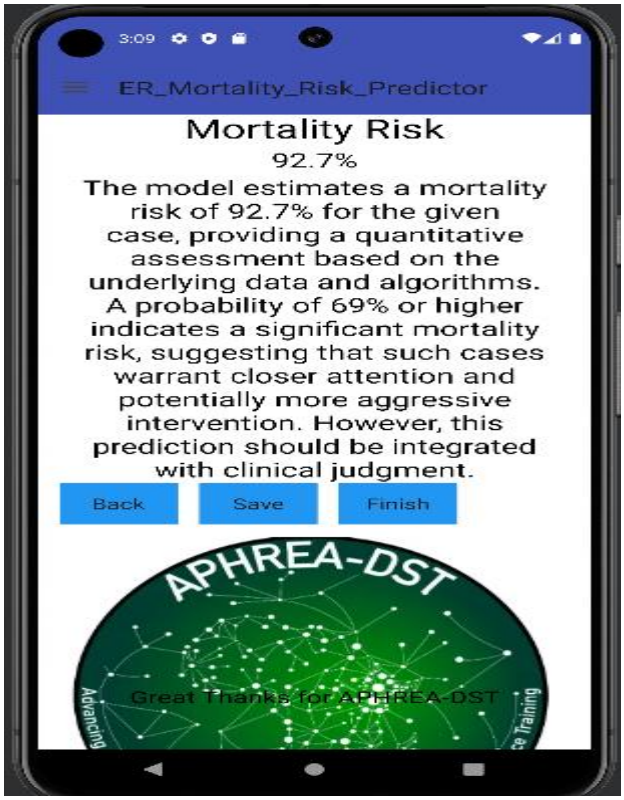

Figure 21: Mortality risk prediction output screen.

In addition to predicting mortality risk, the prototype includes a basic data management system. The patient's input attributes, the anticipated mortality risk

percentage, and the actual clinical result (if known) are all stored locally for each prediction event. With the help of this feature, the application can be used as a feedback-driven tool for improvement, comparing actual results with forecasts to enhance the model. Additionally, the application has an export feature that enables the export of all saved entries to external storage in CSV format. As a result, the prototype serves as both a data-collecting tool and a decision-support tool, facilitating the gathering of locally pertinent datasets for additional training and validation.

The technological viability of using machine learning models for clinical decision support on mobile devices is demonstrated by this prototype. The application functions as proof-of-concept, demonstrating how predictive analytics might be included in routine medical procedures, especially in environments with limited resources where instant access to sophisticated computational resources would not be possible. The creation of this application shows how predictive models may be converted into easily navigable digital health resources that can facilitate quick decisions in emergencies.

4.7 Scope, Strengths, and Limitations of the Study

This study's scope is purposefully narrowed, influenced by both conscious choices and real-world constraints. Python offers a wide range of machine learning methods for classification and prediction, but because of their demonstrated efficacy in medical data analysis, this study only uses four popular algorithms: Random Forest, Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and XGBoost. Furthermore, to focus exclusively on forecasting patient outcomes in urgent care settings, the study is restricted to electronic health records (EHRs) gathered in emergency rooms.

This study exhibits several important strengths. Even though there are problems with data quality because of the novelty of the dataset, it set the groundwork for further research. The study goes beyond condition-specific methods by creating a generalizable mortality prediction model for emergency patients. The model's applicability in a variety of clinical settings is increased by its reliance on frequently gathered or inexpensive variables. Additionally, to close a significant gap in the use of EMR data, the study uses cutting-edge text mining techniques to extract disease

categories from unstructured and frequently "messy" assessment variables. This method demonstrates analytical sophistication.

Despite these strengths, the study has a number of drawbacks that could compromise the effectiveness of the machine learning algorithms. During data preparation, several significant features that are frequently included in well-established mortality prediction models had to be removed because they had more than 50% missing values. The accuracy of the model might have been affected by this essential omission. Even though they are hard to avoid, these limitations emphasize the need for careful result interpretation and the significance of more thorough and well-integrated datasets in subsequent studies.

CHAPTER FIVE: Conclusion and Recommendation

5.1 Conclusions

While there are still obstacles to overcome, such as managing diverse medical diagnosis categories, handling imbalanced datasets, and guaranteeing high-quality data, the creation of trustworthy ML models for predicting mortality risk in ED patients offers substantial opportunities to enhance clinical decision-making. To solve class imbalance, four machine learning algorithms were assessed in this work employing a variety of balancing strategies and hyperparameter tuning methodologies. The findings show that machine learning techniques can reliably forecast mortality risk in hospital emergency departments, by quickly and more precisely identify high-risk patients, shorten evaluation times, facilitate quicker and better therapeutic decisions.

The study successfully collected and prepared a comprehensive two-year dataset of electronic medical records (EMR) from the Emergency Department of Yekatit 12 Hospital Medical College, spanning October 2022 to September 2024. This process involved meticulous data cleaning to address imbalanced datasets through techniques such as SMOTE, undersampling, and hybrid methods, alongside handling diverse medical diagnosis categories and ensuring high-quality data inputs. These efforts ensured the dataset was robust and suitable for machine training, enabling the development of an effective mortality risk prediction model tailored to the ED context.

With equal precision (0.90) and recall (0.55), Random Forest (RF) and XGBoost outperformed the other methods in terms of F1-scores (0.68) for the minority "Died" class. This means that both models reduced false alarms while identifying slightly over half of real deaths. However, RF demonstrated a marginally better ROC-AUC (0.81 vs. 0.80) and matched XGBoost’s high overall accuracy (0.95). Crucially, in clinical settings where clear, intelligible outcomes are essential for physician trust and adoption, RF's feature importance rankings are more consistent and interpretable than XGBoost's.

This work created a model that strikes a balance between interpretability and prediction performance by integrating the RF algorithm with RandomizedSearchCV hyperparameter tuning. This balance guarantees the efficient identification of high-risk patients without overburdening physicians with false positives. As a result, the RF model is a useful and therapeutically applicable tool for predicting mortality risk in EDs, providing both accurate predictions and clear insights into the patient characteristics driving those predictions.

The balanced predictions of the RF model may assist in detecting high-risk patients who might otherwise go unnoticed while lowering alert weariness, a problem that frequently arises in busy ERs. Its emphasis on regularly gathered, inexpensive data improves effectiveness and makes adoption easier in environments with limited resources. Metabolic_Case was the most significant predictor in the RF model, emphasizing the elevated mortality risk linked to metabolic emergencies such as acid-base disorders, diabetic ketoacidosis, and electrolyte imbalances. Following shortly behind was Oncologic_Case, which represented the susceptibility of cancer patients as a result of immunosuppression, treatment side effects, and the advancement of the disease. Demographic characteristics like age group and vital signs also showed predictive significance, capturing age related susceptibility and physiological stress.

The model's applicability for general ED mortality risk prediction is improved by this thorough integration of patient history, vital signs, laboratory results, and diagnostic categories. It provides actionable insights that can direct prompt clinical decision-making while retaining robustness in settings with limited resources.

The implementation of SHAP explanation analysis was critical to this study's translational value. It ensured the model's fidelity to clinical reality by verifying that its predictions were driven by medically plausible features such as metabolic and oncologic comorbidities. Furthermore, its ability to provide individualized explanations paves the way for integration into clinical decision support systems, where it could aid in triage and resource allocation by transparently highlighting the dominant risk factors for a specific patient. Finally, the technique identified unexpected relationships that merit further investigation, demonstrating how AI can serve not only as a predictive tool but also as a catalyst for scientific inquiry.

5.2 Practical Extension: Mobile Application Prototype

A mobile application prototype was created as a useful extension of this study, even though it was not a part of the primary research method. By integrating the Random Forest model into a user-friendly offline platform that will be used on Android smartphones, the application illustrates the approach's translational value. By this extension, the study not only demonstrates methodological advancements in machine learning but also provides insight into its practicality.

By offering quick, data-driven assessments of patient mortality risk at the bedside, the prototype may help emergency department physicians or any HCW work at ED. This could help prioritize high-risk patients, direct the distribution of resources, and eventually raise the standard of emergency care. Additionally, because it is a lightweight mobile solution, it is especially well-suited for healthcare settings with limited resources, where quick access to expensive laboratory results that were used for known predictive analysis is frequently absent.

The prototype's role is expanded from prediction to prospective data gathering with its added capability, which involves storing input-output pairings with real patient outcomes and exporting them to CSV. By producing local datasets that represent the patient population in Ethiopian emergency settings, this capability may make it easier to continuously refine the model.

5.3 Recommendations

The following recommendations are put forth to enhance future implementation and optimize clinical impact considering the study's findings and their practical extension through the creation of a mobile application:

- To enable real-time mortality risk predictions at the point of care, it is advised to utilize the mobile application that was created, after rigorous testing. Additionally, the model should be integrated into the Electronic Medical Records (EMR) system. The model can be a dependable clinical decision support tool for

detecting high-risk patients while reducing false positives due to its high accuracy (90%) and balanced performance. Its interpretability also ensures transparency and physician trust in emergency department operations. However, because recall is still only modest (~55%), the model should be utilized as an additional tool rather than as a substitute for clinical judgment in borderline or uncommon cases.

- Cost-sensitive methods that impose more severe penalties for incorrectly categorizing high-risk patients should be used in future research. This increases sensitivity without unduly compromising precision by ensuring that clinically significant instances are not missed. To improve predictive power and model generalizability, more research into advanced feature engineering techniques—such as automated feature generation, temporal feature extraction, and the integration of outside clinical knowledge—may be able to identify intricate, nonlinear patterns that are frequently overlooked by traditional preprocessing methods.
- Successful deployment will require local capacity building, which includes training ED staff on interpreting model outputs and encouraging physician-data scientist collaboration for ongoing enhancement and contextual adaptation. To optimize patient survival while fostering equity and effectiveness in model deployment, healthcare organizations should standardize important predictors, support policy-aligned validation, and make sure that model selection is in line with therapeutic objectives. In order to facilitate broader applications like real-time surveillance, clinical decision support, and population-level health analytics, as well as to support machine learning models, governments should give top priority to the development and fortification of EMR systems. This will lay the groundwork for long-term, data-driven healthcare innovation.
- The mobile application's clinical relevance could be further enhanced by new features, including role-based access for physicians, real-time synchronization with hospital systems, and automatic alarm messages for high-risk patients. Furthermore, for sustainability, interoperability, and broad acceptance, the app must be in line with changing EMR standards and compatible with a range of mobile platforms. To remain relevant and dependable in clinical practice, the development cycle should also incorporate regular updates, security fixes, and feedback-driven modifications.

REFERENCES.

[1] M. A. Messelu et al., "Mortality and its determinants among patients attending in emergency departments," BMC Emergency Medicine, vol. 24, no. 125, pp. 1–10, 2024. doi.org/10.1186/s12873-024-01050-6.

[2] F. Abebe, A. Habtamu, and A. Workina, "Risks of early mortality and associated factors at adult emergency department of Jimma University Medical Center," Open Access Emergency Medicine,

[3] C. Bae et al., "Evaluating emergency care capacity in Africa: An iterative, multicountry refinement of the Emergency Care Assessment Tool," BMJ Global Health, vol. 3, no. 6, p. e001138, 2018, doi: 10.1136/bmjgh-2018-001138.

[4] S. Moosavi and S. Zargar, "Mortality prediction in emergency department using machine learning models," Journal of Archives in Military Medicine, Oct. 2023, doi: 10.5812/jamm-140442.

[5] W. Dode et al., "Pattern and predictors of mortality in emergency department of Saint Paul Hospital Millennium Medical College Addis Ababa, Ethiopia: Retrospective study," St. Paul's Hospital Millennium Medical College, Sep. 9, 2022. doi.org/10.21203/rs.3.rs-1920924/v1.

[6] A. Ala, S. S. Vahdati, M. Jalali, and S. Parsay, "Rapid Emergency Medicine Score as a Predictive Value for 30-day Outcome of Nonsurgical Patients Referred to the Emergency Department," 2020. : http://creativecommons.org/publicdomain/zero/1.0/.

[7] S. Pujari et al., "Artificial intelligence for global health: Cautious optimism with safeguards," Bulletin of the World Health Organization, vol. 101, no. 5, pp. 364-364A, May 2023. doi: 10.2471/BLT.23.290215.

[8] C. Hu et al., "Interpretable machine learning for early prediction of prognosis in sepsis: A discovery and validation study," Infectious Diseases and Therapy, vol. 11, pp. 1117-1132, Apr. 2022, doi: 10.1007/s40121-022-00628-6.

[9] A. G. Belayneh et al., "Prolonged length of stay and its associated factors at adult emergency department in Amhara region comprehensive specialized hospitals, northwest Ethiopia," BMC Emergency Medicine, vol. 23, no. 34, pp. 1–9, 2023. doi.org/10.1186/s12873-023-00804-y.

[10] D. Stoessel et al., "Early prediction of in-hospital mortality utilizing multivariate predictive modelling of electronic medical records and socio-determinants of health of the first day of hospitalization," BMC Medical Informatics and Decision Making, vol. 23, no. 259, pp. 1–12, 2023. doi.org/10.1186/s12911-023-02356-4.

[11] D. T. Ha et al., "Prognostic performance of the Rapid Emergency Medicine Score (REMS) and Worthing Physiological Scoring system (WPS) in emergency department," Int. J. Emerg. Med., vol. 8, no. 18, pp. 1–7, 2015. doi.org/10.1186/s12245-015-0066-3.

[12] S. Luka et al., "Can we improve mortality prediction in patients with sepsis in the emergency department?," Medicina, vol. 60, no. 8, p. 1333, Aug. 2024, doi: 10.3390/medicina60081333.

[13] F. Kadri, A. Dairi, F. Harrou, and Y. Sun, "Towards accurate prediction of patient length of stay at emergency department: a GAN-driven deep learning framework," J. Ambient Intell. Humaniz. Comput., vol. 14, pp. 11481–11495, Feb. 2023. doi.org/10.1007/s12652-022-03717-z.

[14] J.-W. Perng et al., "Mortality prediction of septic patients in the emergency department based on machine learning," J. Clin. Med., vol. 8, no. 11, p. 1906, Nov. 2019. doi.org/10.3390/jcm8111906.

[15] A. J. Zeleke et al., "Machine learning-based prediction of hospital prolonged length of stay admission at emergency department: a Gradient Boosting algorithm

analysis," Front. Artif. Intell., vol. 6, p. 1179226, Jul. 2023. doi.org/10.3389/frai.2023.1179226.

[16] H. D. Yosha et al., "A two-year review of adult emergency department mortality at Tikur Anbesa specialized tertiary hospital, Addis Ababa, Ethiopia," BMC Emergency Medicine, vol. 21, no. 33, pp. 1–7, 2021. doi.org/10.1186/s12873-021-00429-z.

[17] A. Ala et al., "Rapid Emergency Medicine Score as a predictive value for 30-day outcome of nonsurgical patients referred to the emergency department," Open Access, 2020. https://creativecommons.org/licenses/by-nc/4.0/.

[18] T. Olsson and L. Lind, "Comparison of the Rapid Emergency Medicine Score and APACHE II in nonsurgical emergency department patients," Acad. Emerg. Med., vol. 10, no. 10, pp. 1040–1048, Oct. 2003.

[19] B. F. Imhoff et al., "Rapid Emergency Medicine Score (REMS) in the trauma population: a retrospective study," BMJ Open, vol. 4, p. e004738, 2014. https://doi.org/10.1136/bmjopen-2013-004738.

[20] J. Z. Pong et al., "Validation of the mortality in emergency department sepsis (MEDS) score in a Singaporean cohort," Medicine, vol. 98, no. 34, p. e16962, 2019. https://doi.org/10.1097/MD.0000000000016962.

[21] N. Maryani and C. F. R. Wisudarti, "The comparison score of SPEEDS, MEDS, SOFA, APACHE II, and SAPS II as predictor of sepsis mortality: A systematic review," AJOSH, vol. 2, no. 01, Oct. 2023. https://ajosh.org/.

[22] M. S. Alie et al., "Machine learning algorithms for predicting COVID-19 mortality in Ethiopia," BMC Public Health, vol. 24, no. 1728, pp. 1–10, 2024. https://doi.org/10.1186/s12889-024-19196-0.

[23] M. H. Choi et al., "Mortality prediction of patients in intensive care units using machine learning algorithms based on electronic health records," Scientific Reports, vol. 12, no. 7180, pp. 1–9, 2022. https://doi.org/10.1038/s41598-022-11226-4.

[24] "Machine learning algorithms explained InfoWorld," Jul. 22, 2019. https://www.infoworld.com/article/3394399/machine-learning-algorithms-explained.html.

[25] V. Jakkula, "Tutorial on Support Vector Machine," Appl. Comput. Math., vol. 6, no. 4, p. 13, 2016. [Online]. https://doi.org/10.11648/j.acm.s.2017060401.11.

[26] Q. An et al., "A Comprehensive Review on Machine Learning in Healthcare Industry: Classification, Restrictions, Opportunities and Challenges," Sensors, vol. 23, no. 9, p. 4178, May 2023. https://doi.org/10.3390/s23094178.

[27] B. G. Abebaw, "ACCEPTANCE Stroke Risk Prediction using Machine Learning," Internal Examiner External Examiner.

[28] "Machine Learning: Algorithms and Applications," Jan. 23, 2025. : https://www.researchgate.net/publication/303806260_Machine_Learning_Algorithms_and_Applications.

[29] "Support Vector Machine — Introduction to Machine Learning Algorithms," Jan. 23, 2025. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[30] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., New York, NY, USA, 2016, pp. 785–794. https://doi.org/10.1145/2939672.2939785.

[31] R. Bopche et al., "In-hospital mortality, readmission, and prolonged length of stay risk prediction leveraging historical electronic patient records," JAMIA Open, vol. 7, no. 3, p. ooae074, 2024. https://doi.org/10.1093/jamiaopen/ooae074.

[32] S. M. Kashani and S. Zargar, "Mortality prediction in emergency department using machine learning models," J. Arch. Mil. Med., vol. 11, no. 3, p. e140442, Sep. 2023. https://doi.org/10.5812/jamm-140442.

[33] Y.-H. Wang et al., "Predicting 2-day mortality of thrombocytopenic patients based on clinical laboratory data using machine learning," Medical Care, vol. 59, no. 3, pp. 245-254, Mar. 2021,

[34] S. Chakrabarti et al., Data Mining Know It All. Burlington, MA, USA: Morgan Kaufmann, 2009.

ANNEXES

Appendix 1: Different Machine Learning Algorithm Performance Table.

A comprehensive comparison table of the performance of different machine learning algorithms is provided in this appendix. The assessment ranges from preliminary experiments with unbalanced datasets without hyperparameter adjustment to more sophisticated strategies that integrate hybrid balancing techniques with Grid Search (GS) and Random Search (RS) optimization.

K-NN Model Results Comparison Table.

Method	Class	Precision	Recall	F1 Score (CV)	Specificity	ROC-AUC	Accuracy
Initial Model W/O Balancing	Alive(0)	0.94	0.96	0.95	0.96	0.77	0.91
	Died(1)	0.60	0.47	0.53			
RandomizedSearchCV	Alive(0)	0.93	0.98	0.95	0.98	0.76	0.92
	Died(1)	0.68	0.41	0.51			
GridSearchCV	Alive(0)	0.93	0.97	0.95	0.97	0.76	0.92
	Died(1)	0.65	0.41	0.50			
SMOTE Only	Alive(0)	0.93	0.94	0.94	0.94	0.74	0.94
	Died(1)	0.44	0.41	0.42			
SMOTE+RS	Alive(0)	0.94	0.98	0.90	0.98	0.79	0.93
	Died(1)	0.77	0.50	0.61			
SMOTE+GS	Alive(0)	0.93	0.97	0.95	0.97	0.76	0.92
	Died(1)	0.65	0.41	0.50			
Undersampling	Alive(0)	0.95	0.96	0.85	0.96	0.80	0.91
	Died(1)	0.59	0.55	0.57			
Undersampling+RS	Alive(0)	0.95	0.98	0.96	0.98	0.79	0.93
	Died(1)	0.73	0.54	0.62			
Undersampling+GS	Alive(0)	0.95	0.97	0.96	0.97	0.78	0.93
	Died(1)	0.69	0.55	0.61			

Hybrid Balancing + RS	Alive(0)	0.95	0.99	0.97	0.98	0.79	0.94
	Died(1)	0.81	0.52	0.64			
Hybrid Balancing + GS	Alive(0)	0.94	0.98	0.96	0.98	0.78	0.93
	Died(1)	0.76	0.51	0.61			

RF model Results Comparison Table.

Method	Class	Precision	Recall	F1 Score (CV)	Specificity	ROC-AUC	Accuracy
Initial Model W/O Balancing	Alive(0)	0.95	0.99	0.97	0.99	0.77	0.94
	Died(1)	0.81	0.55	0.65			
RandomizedSearchCV	Alive(0)	0.95	0.99	0.97	0.99	0.81	0.95
	Died(1)	0.90	0.55	0.68			
GridSearchCV	Alive(0)	0.95	0.98	0.97	0.98	0.80	0.94
	Died(1)	0.81	0.55	0.66			
SMOTE Only	Alive(0)	0.95	0.96	0.95	0.96	0.79	0.91
	Died(1)	0.59	0.55	0.57			
SMOTE+RS	Alive(0)	0.94	0.99	0.97	0.99	0.78	0.94
	Died(1)	0.90	0.46	0.61			
SMOTE+GS	Alive(0)	0.95	0.99	0.97	0.99	0.81	0.94
	Died(1)	0.86	0.52	0.65			
Undersampling	Alive(0)	0.95	0.99	0.97	0.99	0.80	0.94
	Died(1)	0.86	0.51	0.64			
Undersampling+ RS	Alive(0)	0.95	0.99	0.97	0.98	0.80	0.94
	Died(1)	0.83	0.55	0.66			
Undersampling+ GS	Alive(0)	0.95	0.99	0.97	0.98	0.80	0.94
	Died(1)	0.88	0.51	0.65			
Hybrid Balancing + RS	Alive(0)	0.95	0.99	0.97	0.99	0.79	0.94
	Died(1)	0.90	0.51	0.65			
Hybrid Balancing + GS	Alive(0)	0.94	0.99	0.96	0.99	0.80	0.94
	Died(1)	0.82	0.49	0.61			

SVM Model Results Comparison Table.

Method	Class	Precision	Recall	F1 Score (CV)	Specificity	ROC-AUC	Accuracy
Initial Model W/O Balancing	Alive(0)	0.95	0.99	0.97	0.99	0.79	0.94
	Died(1)	0.85	0.52	0.64			
RandomizedSearchCV	Alive(0)	0.95	0.98	0.96	0.98	0.78	0.94
	Died(1)	0.78	0.54	0.64			
GridSearchCV	Alive(0)	0.95	0.98	0.96	0.98	0.80	0.94
	Died(1)	0.78	0.55	0.64			
SMOTE Only	Alive(0)	0.94	0.97	0.96	0.97	0.79	0.92
	Died(1)	0.66	0.46	0.54			
SMOTE+RS	Alive(0)	0.94	0.97	0.95	0.97	0.78	0.92
	Died(1)	0.65	0.48	0.55			
SMOTE+GS	Alive(0)	0.94	0.97	0.95	0.97	0.78	0.92
	Died(1)	0.66	0.44	0.53			
Undersampling	Alive(0)	0.95	0.98	0.96	0.98	0.78	0.94
	Died(1)	0.78	0.54	0.63			
Undersampling+ RS	Alive(0)	0.95	0.98	0.96	0.98	0.79	0.93
	Died(1)	0.78	0.52	0.63			
Undersampling+ GS	Alive(0)	0.94	0.99	0.97	0.99	0.78	0.94
	Died(1)	0.81	0.51	0.62			
Hybrid Balancing + RS	Alive(0)	0.95	0.99	0.97	0.99	0.79	0.94
	Died(1)	0.82	0.53	0.64			
Hybrid Balancing + GS	Alive(0)	0.94	0.98	0.96	0.98	0.79	0.92
	Died(1)	0.72	0.44	0.55			

XGB model Results Comparison Table.

Method	Class	Precision	Recall	F1 Score (CV)	Specificity	ROC-AUC	Accuracy
Initial Model W/O Balancing	Alive(0)	0.95	0.99	0.97	0.99	0.80	0.95
	Died(1)	0.90	0.55	0.68			
RandomizedSearchCV	Alive(0)	0.95	0.99	0.97	0.99	0.80	0.95
	Died(1)	0.89	0.55	0.68			
GridSearchCV	Alive(0)	0.95	0.99	0.97	0.99	0.81	0.95
	Died(1)	0.91	0.54	0.68			
SMOTE Only	Alive(0)	0.94	0.98	0.91	0.98	0.80	0.93
	Died(1)	0.75	0.51	0.60			
SMOTE+RS	Alive(0)	0.95	0.94	0.99	0.99	0.80	0.93
	Died(1)	0.74	0.53	0.62			
SMOTE+GS	Alive(0)	0.94	0.99	0.97	0.99	0.80	0.94
	Died(1)	0.83	0.49	0.62			
Undersampling	Alive(0)	0.95	0.99	0.97	0.99	0.80	0.94
	Died(1)	0.88	0.54	0.67			
Undersampling+ RS	Alive(0)	0.95	0.98	0.96	0.98	0.81	0.93
	Died(1)	0.76	0.54	0.63			
Undersampling+ GS	Alive(0)	0.94	0.99	0.97	0.99	0.80	0.94
	Died(1)	0.83	0.51	0.63			
Hybrid Balancing + RS	Alive(0)	0.95	0.98	0.96	0.98	0.80	0.93
	Died(1)	0.74	0.52	0.61			
Hybrid Balancing + GS	Alive(0)	0.90	0.98	0.94	0.98	0.80	0.93
	Died(1)	0.78	0.64	0.63			

Undersampling -Based Models Assessment.

Method	Class	Precision	Recall	F1 Score (CV)
KNN-Undersampling	Alive(0)	0.95	0.96	0.85
	Died(1)	0.59	0.55	0.57
KNN-Undersampling+RS	Alive(0)	0.95	0.98	0.96
	Died(1)	0.73	0.54	0.62
KNN-Undersampling+GS	Alive(0)	0.95	0.97	0.96
	Died(1)	0.69	0.55	0.61
RF- Undersampling	Alive(0)	0.95	0.99	0.97
	Died(1)	0.86	0.51	0.64
RF-Undersampling+RS	Alive(0)	0.95	0.99	0.97
	Died(1)	0.83	0.55	0.66
RF-Undersampling+GS	Alive(0)	0.95	0.99	0.97
	Died(1)	0.88	0.51	0.65
SVM-Undersampling	Alive(0)	0.95	0.98	0.96
	Died(1)	0.78	0.54	0.63
SVM-Undersampling+RS	Alive(0)	0.95	0.98	0.96
	Died(1)	0.78	0.52	0.63
SVM-Undersampling+GS	Alive(0)	0.94	0.99	0.97
	Died(1)	0.81	0.51	0.62
XGB-Undersampling	Alive(0)	0.95	0.99	0.97
	Died(1)	0.88	0.54	0.67
XGB-Undersampling+RS	Alive(0)	0.95	0.98	0.96
	Died(1)	0.76	0.54	0.63
XGB-Undersampling+GS	Alive(0)	0.94	0.99	0.97
	Died(1)	0.83	0.51	0.63

SMOTE -Based Models Assessment.

Method	Class	Precision	Recall	F1 Score (CV)
KNN- SMOTE Only	Alive(0)	0.93	0.94	0.95
	Died(1)	0.44	0.41	0.53
KNN- SMOTE+RS	Alive(0)	0.94	0.98	0.95
	Died(1)	0.77	0.50	0.51
KNN- SMOTE+GS	Alive(0)	0.93	0.97	0.95
	Died(1)	0.65	0.41	0.50
RF- SMOTE Only	Alive(0)	0.95	0.96	0.95
	Died(1)	0.59	0.55	0.57
RF- SMOTE+RS	Alive(0)	0.94	0.99	0.97
	Died(1)	0.90	0.46	0.61
RF- SMOTE+GS	Alive(0)	0.95	0.99	0.97
	Died(1)	0.86	0.52	0.65
SVM- SMOTE Only	Alive(0)	0.94	0.97	0.96
	Died(1)	0.66	0.46	0.54
SVM- SMOTE+RS	Alive(0)	0.94	0.97	0.95
	Died(1)	0.65	0.48	0.55
SVM- SMOTE+GS	Alive(0)	0.94	0.97	0.95
	Died(1)	0.66	0.44	0.53
XGB- SMOTE Only	Alive(0)	0.94	0.98	0.91
	Died(1)	0.75	0.51	0.60
XGB- SMOTE+RS	Alive(0)	0.95	0.94	0.99
	Died(1)	0.74	0.53	0.62
XGB- SMOTE+GS	Alive(0)	0.94	0.99	0.97
	Died(1)	0.83	0.49	0.62

Unbalanced-Based Models Assessment.

Method	Class	Precision	Recall	F1 Score (CV)
KNN- W/O Balancing	Alive(0)	0.94	0.96	0.95
	Died(1)	0.60	0.47	0.53
KNN- W/O Balancing+ RandomizedSearchCV	Alive(0)	0.93	0.98	0.95
	Died(1)	0.68	0.41	0.51
KNN- W/O Balancing + GridSearchCV	Alive(0)	0.93	0.97	0.95
	Died(1)	0.65	0.41	0.50
RF- W/O Balancing	Alive(0)	0.95	0.99	0.97
	Died(1)	0.81	0.55	0.65
RF- W/O Balancing + RandomizedSearchCV	Alive(0)	0.95	0.99	0.97
	Died(1)	0.90	0.55	0.68
RF- W/O Balancing+ GridSearchCV	Alive(0)	0.95	0.98	0.97
	Died(1)	0.81	0.55	0.66
SVM- W/O Balancing	Alive(0)	0.95	0.99	0.97
	Died(1)	0.85	0.52	0.64
SVM- W/O	Alive(0)	0.95	0.98	0.96

Balancing+ RandomizedSearchCV	Died(1)	0.78	0.54	0.64
SVM-W/O Balancing+ GridSearchCV	Alive(0)	0.95	0.98	0.96
	Died(1)	0.78	0.55	0.64
XGB- W/O Balancing	Alive(0)	0.95	0.99	0.97
	Died(1)	0.90	0.55	0.68
XGB- W/O Balancing + RandomizedSearchCV	Alive(0)	0.95	0.99	0.97
	Died(1)	0.89	0.55	0.68
XGB- W/O Balancing +GridSearchCV	Alive(0)	0.95	0.99	0.97
	Died(1)	0.91	0.54	0.68

Appendix 2: Sample Code for Best-Performing Models

Implementation examples of the best-performing machine learning models assessed in this study are included in this appendix. A distinct classification algorithm's fundamental implementation, including data preprocessing, model training, and evaluation stages, is shown in each code sample.

1 = Imputing numerical data

```
# 3. Iterative Imputation (MICE with RandomForest)
imputer = IterativeImputer(
    estimator=RandomForestRegressor(n_estimators=100, random_state=42),
    max_iter=20,
    random_state=123,
    verbose=2
)
X_imputed_scaled = imputer.fit_transform(X_scaled)
```

2 = Imputing ordinal_categories

```
 # MICE Imputation (Using BayesianRidge as in the original intention)
    imputer = IterativeImputer(
        estimator=BayesianRidge(),
        max_iter=20,
        random_state=123,
        min_value=min_val_imputer,
        max_value=max_val_imputer
    )
    X_imputed = imputer.fit_transform(X_encoded)
```

3 = Imputing nominal variables

```
# Step 2: MICE Imputation
mice_imputer = IterativeImputer(
    estimator=LogisticRegression(max_iter=1000),
    max_iter=20,
    random_state= 123)
X_nominal_imputed_encoded = mice_imputer.fit_transform(X_nominal_encoded)
```

4 = Text Mining

```
def preprocess_text2(text):
    # Step 1: Add spaces around commas to separate attached words
    text = re.sub(r',', ' , ', str(text))

    # Step 2: Replace specific characters with spaces
    text = re.sub(r'[+?_-]', ' ', text)

    # Step 3: Lowercase, remove punctuation (except protected commas), and numbers
    text = text.lower()
    text = re.sub(r'[^a-zA-Z0-9\s,]', '', text)
    text = re.sub(r'[^\w\s?\-,]', '', text)
    text = re.sub(r'\d+', '', text)
    text = re.sub(r'\d+(?:st|nd|rd|th|ry)', '', text)
    text = re.sub(r'''\b|\b'''', '', text)

    # Step 4: Correct typos (applied BEFORE tokenization)
    for typo, correct in typo_corrections2.items():
        text = re.sub(r'\b' + typo + r'\b', correct, text)

    # Step 5: Tokenize and clean
    tokens = word_tokenize(text)
    tokens = [word for word in tokens if word.isalpha()]

    # Step 6: Remove stopwords (optional)
    #tokens = [word for word in tokens if word not in stop_words]

    # Step 7: Generate n-grams (bigrams/trigrams) and combine with single words
    bigrams = extract_ngrams(tokens, n=2)
    trigrams = extract_ngrams(tokens, n=3)

    # Combine all terms (single words + n-grams)
    all_terms = tokens + bigrams + trigrams
```

```
    # Step 8: Filter for predetermined terms and their categories
    extracted_categories = set()
    for term in all_terms:
        for category, terms in medical_categories.items():
            if term in terms:
                extracted_categories.add(category)

    # Step 9: Return categories based on the number of unique categories found
    if len(extracted_categories) == 1:
        return list(extracted_categories)[0]
    elif len(extracted_categories) > 1:
        return list(extracted_categories)
    else:
        return []
```

5 = Applying PCA for training and test data.

```
# Store PCA transformers for each pair
pca_transformers = {}

# Transform training data
X_train1 = X_tra.copy()
for pair in corr_pairs:
    pca = PCA(n_components=1)
    X_train1[f"PCA_{pair[0]}"] = pca.fit_transform(X_tra[pair])
    X_train1.drop(columns=pair, inplace=True)
    pca_transformers[tuple(pair)] = pca

# Transform test data
X_test1 = X_tes.copy()
for pair in corr_pairs:
    pca = pca_transformers[tuple(pair)]  # Reuse the PCA fitted on X_train
    X_test1[f"PCA_{pair[0]}"] = pca.transform(X_tes[pair])
    X_test1.drop(columns=pair, inplace=True)
```

6 = Recursive Feature Elimination

```
# Create RFECV object
rfecv = RFECV(
    estimator=estimator,
    step=1,
    cv=StratifiedKFold(5),  # 5-fold cross-validation
    scoring='accuracy'  # or other appropriate metric
)

# Fit RFECV
rfecv.fit(X_train1, y_tra)

# Plot number of features vs. cross-validation scores
plt.figure(figsize=(10, 6))
plt.xlabel("Number of features selected")
plt.ylabel("Cross validation score (accuracy)")
plt.plot(range(1, len(rfecv.cv_results_['mean_test_score']) + 1),
rfecv.cv_results_['mean_test_score'])
plt.show()
print(f"Optimal number of features: {rfecv.n_features_}")
```

7= RF + SMOTE with RS.

```
 # Define pipeline (SMOTE → RF)
pipeline = Pipeline([('smote', SMOTE(random_state=42)),
    ('rf', RandomForestClassifier(random_state=42, class_weight=None)) ])
    param_dist = {
        'rf__n_estimators': randint(50, 200),          # Wider range than GridSearch
        'rf__max_depth': [None, 6, 10],                # Mix of unlimited and constrained
        'rf__min_samples_split': randint(2, 10),       # Random integer in range
        'rf__max_features': ['sqrt', 0.8]              # Feature subset strategies
    }
    rf_os_rs = RandomizedSearchCV(
        estimator=pipeline,
        param_distributions=param_dist,
```

```
    n_iter=20,
    scoring='f1',
    cv=3,
    n_jobs=-1,
    random_state=42
)
```

8 = SVM + GS (without balancing)

```
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': [0.001, 0.01, 0.1],
    'kernel': ['rbf']
}
svc_gs = GridSearchCV(
    base_svc,
    param_grid=param_grid,
    scoring='f1',
    cv=3,
    verbose=1,
    n_jobs=-1
)
```

9 = XGB + Undersampling (without hyperparameters)

```
# Define and train a model
xgb_us = XGBClassifier(
    objective='binary: logistic',
    eval_metric=['logloss', 'aucpr', 'error'],
    n_estimators=200,
    max_depth=6,
    learning_rate=0.05,
    early_stopping_rounds=20,
    subsample=0.8,
    colsample_bytree=0.8,
    reg_alpha=0.5,
```

```
    reg_lambda=1,
    gamma=0.1,
    random_state=42,
    use_label_encoder=False
    )
xgb_us.fit(
    X_resampled, y_resampled,
    sample_weight=sample_weights,
    eval_set=[(X_fold_val, y_fold_val)],
    verbose=0
)
```

10 = KNN + Undersampling with GS.

```
# 2. Define medical-optimized parameter grid
param_grid = {
    'undersampler__sampling_strategy': [0.7, 0.8, 0.9],  # Undersampling ratios
    'knn__n_neighbors': [5, 7, 9, 11],  # Odd numbers for binary classification
    'knn__weights': ['distance'],  # Emphasize closer neighbors
    'knn__p': [1],  # Manhattan distance (better for medical data)
    'knn__leaf_size': [20, 30]  # Balance speed vs accuracy
}
scoring = {
    'f1': make_scorer(f1_score, pos_label=1),
    'recall': make_scorer(recall_score, pos_label=1),  # Critical for disease detection
    'roc_auc': 'roc_auc'
}
knn_us_gs = GridSearchCV(
    estimator=knn_pipe,
    param_grid=param_grid,
    scoring=scoring,
    refit='f1',  # Optimize for F1 then refit best model
    cv=StratifiedKFold(n_splits=5, shuffle=True, random_state=42),
    n_jobs=-1,
    verbose=1)
```

Appendix 3: Export The Model to ONNX Format

```
from skl2onnx import convert_sklearn
from skl2onnx.common.data_types import FloatTensorType
import joblib  # If your model is saved as a .joblib file

# Load the model trained with selected features
model = joblib.load('best_rf_rs_selected_features.joblib')

# Get the number of features from the loaded model (assuming it's a pipeline)
if hasattr(model, 'named_steps') and 'classifier' in model.named_steps:
    num_features = len(model.named_steps['classifier'].feature_names_in_)
elif hasattr(model, 'feature_names_in_'):
     num_features = len(model.feature_names_in_)
else:
    # Fallback or raise an error if feature names cannot be determined
     print("Warning: Could not automatically determine number of features from the model.")
      num_features = X_train_selected.shape[1] # Assuming X_train_selected is available and correct

# Define the input type for the ONNX model
initial_type = [('input_features', FloatTensorType([None, num_features]))]

# Convert the loaded model to ONNX format
onnx_model = convert_sklearn(model, initial_types=initial_type, target_opset=12)

# Save the ONNX model to a file
with open("mortality_model_selected_features.onnx", "wb") as f:
    f.write(onnx_model.SerializeToString())

print("Model converted to ONNX format and saved as 'mortality_model_selected_features.onnx'")
```

Appendix 4: Mortality Risk Prediction Code (ONNX Model Integration).

```
private float runONNXPrediction(float[] inputFeatures) throws Exception {
   if (session == null) throw new IllegalStateException("ONNX session not initialized");

   Log.d("ONNX", "Input features: " + Arrays.toString(inputFeatures));

   long[] shape = {1, inputFeatures.length};
   OnnxTensor inputTensor = OnnxTensor.createTensor(
         OrtEnvironment.getEnvironment(),
         FloatBuffer.wrap(inputFeatures),
         shape
   );

   try (inputTensor) {
      String inputName = session.getInputNames().iterator().next();
      Log.d("ONNX", "Input name: " + inputName);

      OrtSession.Result results = session.run(Collections.singletonMap(inputName, inputTensor));

      float riskPercentage = 0f;

      // Get output names properly - use getOutputNames() instead of keys()
      for (String outputName : session.getOutputNames()) {
         Optional<OnnxValue> valueOptional = results.get(outputName);
         if (valueOptional.isPresent()) {
            try (OnnxValue value = valueOptional.get()) {
               Log.d("ONNX", "Output: " + outputName + ", Type: " + value.getType());

               // Try to extract the risk percentage from different output types
```

```
                riskPercentage = extractRiskFromValue(value, outputName);
                if (riskPercentage > 0f) {
                    break; // Found valid risk value
                }
            }
        }
    }

    Log.d("ONNX", "Final Mortality Risk: " + riskPercentage + "%");
    return riskPercentage;
  }
}
```