



MASTER OF SCIENCE IN COMPUTER SCIENCE

# Machine Learning Based Prediction of Critical Events of Intensive Care Heart Failure Patients

*Author:*

KARTIKAY SRIVASTAVA

*(Matriculation No. 1599127)*

*Supervisors:*

PROF. DR.-ING. ROMAN OBERMAISSER (INTERNAL)

M.SC. ABU SHAD AHAMMED (INTERNAL)

DIPL. PHYS. DUBRAVKA UKALOVIC (EXTERNAL)

May 25, 2023

A master thesis written at Embedded Systems Department (University of Siegen) in collaboration with Siemens Healthcare GmbH and submitted in partial fulfillment of the requirements for the degree Master of Science.

**Author:** Kartikay Srivastava

**Title:** Machine Learning Based Prediction of Critical Events of Intensive Care Heart Failure Patients.

**Date:** May 25, 2023

**Supervisors:**

Prof. Dr.-Ing. Roman Obermaisser (internal)

M.Sc. Abu Shad Ahammed (internal)

Dipl.-Phys. Dubravka Ukalovic (external)

# Statutory Declaration

I, Kartikay Srivastava (matriculation number: 1599127) affirm that I have written my thesis (in the case of a group thesis, my appropriately marked part of the thesis) independently and that I have not used any sources or aids other than those indicated, and that I have clearly indicated citations.

All passages that are taken from other works in terms of wording or meaning (including translations) have been clearly marked as borrowed in each individual case, with precise indication of the source (including the World Wide Web and other electronic data collections). This also applies to attached drawings, pictorial representations, sketches and the like. I take note that the proven omission of the indication of origin will be considered as attempted deception.

**Location:** Erlangen (Bavaria)

**Date:** May 25, 2023

**Signature:** 

# Acknowledgements

First and foremost, I have to express my gratitude to my supervisor Prof. Dr.-Ing. Roman Obermaisser for his excellent supervision and support throughout the thesis. His teaching style and enthusiasm from his classes on Embedded System made a strong impression on me.

I also want to thank all professors at Faculty IV at the University of Siegen who taught such important technical topics that aided me in finishing all my courses.

I would like to thank my supervisors M.Sc. Abu Shad Ahammed and Dipl.-Phys. Dubravka Ukalovic for their assistance and dedicated involvement in every step throughout the process. I really appreciate and express my humble gratitude to Mr. Marcus Zimmermann for providing such a healthy atmosphere for budding knowledge and work in Siemens Healthineers.

At the same time, I would like to thank my friends and colleagues O.Hammoud, B.Schmitt, R.Srivastava and A.Garg for their constant moral and day-to-day support. Most importantly none of this would have been possible without the unconditional love, encouragement and support of my family. I am grateful forever.

Kartikay Srivastava

# List of Abbreviations

**AI** . . . . . Artificial Intelligence

**ETL** . . . . . Extract Transform and Load

**CVD** . . . . . Cardiovascular disease

**MIMIC** . . . . . Medical Information Mart for Intensive Care

**EHR** . . . . . Electronic Health Record

**ICU** . . . . . Intensive Care Unit

**HADMID** . . . . . Hospital Admission ID

**KNN** . . . . . K Nearest Neighbour

**XGBoost** . . . . . Extreme Gradient Boosting

**ANN** . . . . . Artificial Neural Network

**AUROC** . . . . . Area Under the Receiver Operating Characteristics

**RAM** . . . . . Random Access Memory

**NVMe** . . . . . Non-Volatile Memory Express

**SSD** . . . . . Solid-State Drive

**TP** . . . . . True Positive

**TN** . . . . . True Negative

**FP** . . . . . False Positive

**FN** . . . . . False Negative

**XAI** . . . . . Explainable Artificial Intelligence

**LIME** . . . . . Local Interpretable Model-Agnostic Explanations

**BMI** . . . . . Body Mass Index

**SHAP** . . . . . SHapley Additive exPlanation

**IOT** . . . . . Internet of Things

**LASSO** . . . . . Least Absolute Shrinkage and Selection Operator

**ICD** . . . . . International Classification of Diseases

**MV** . . . . . MetaVision

**CV** . . . . . CareVue

**SVM** . . . . . Support Vector Machine

**LAN** . . . . . Local Area Network

**MAN** . . . . . Metropolitan Area Network

**WAN** . . . . . Wide Area Network

**GUI** . . . . . Graphical User Interface

**csv** . . . . . Comma-Separated Values

**AUC** . . . . . Area under the ROC Curve

**AUC ROC** . . . . . Area Under Curve Receiver Operating Characteristic

# Abstract

Artificial Intelligence since its inception has sown its seeds deep into every possible field, ranging from complex systems such as self-driving cars to the more traditional ones like YouTube and Netflix recommender systems. Healthcare is one such sector where the demand for the applications of Artificial Intelligence systems has gained tremendous growth over the years. The recent corona pandemic has not only exposed the short comings of an already overburdened healthcare sector but has also made the world to realize the need for automation in this sector that has till now relied heavily on human intervention. The focus of this thesis is to analyze the various factors that can lead to heart failure among patients in a hospital. Heart failure is one of the leading causes of exacerbation/mortality for a patient in a hospital. Early prediction of such factors could lead to prevention of mortality for such patients.

In this Master Thesis, various machine learning algorithms will be tested to predict end points such as mortality and exacerbation for patients that have been admitted in Intensive Care Unit (ICU) of a hospital. Later, XAI methods will be used to explain the factors that could lead to a particular outcome for a patient. The result of this master thesis is a software (patient monitoring system) that can be used by the medical staffs or the caretakers of a patient to address risk contributing factors at an early stage.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Healthcare is a traditional sector that has witnessed revolutionary changes over the years ranging from advancement in tools/machines required for diagnosis of diseases to various methodologies like online consultation, sentiment analysis using sensor-based technology etc. to ease the overall hospital process. With the gigantic increase in population, the burden has fallen badly on this sector causing huge strain on the doctors as well as the associated medical staffs and caretakers. With the advancements in the field of computer science, it is possible to incorporate various machine learning strategies in the diagnosis of ailments and hence automate the process of diagnosis and reduce the need for human intervention.

The primary objective of this study is to evaluate the relationship between various factors associated with a patient (who has been diagnosed with heart failure) by including both static and dynamic features that could lead to exacerbation or mortality for that patient. To create a dataset that can be used in a machine learning model for prediction purposes, it is necessary to transform a patient's data from a hospital record (EHR) to machine interpretable form. An EHR is a patient's digital record. These records were originally created to archive a patient's information and perform administrative tasks like billing. Later due to abundance of such records it was made possible to develop applications that can be used in a clinical scenario [3] [4] [5].

The major part for carrying out this study will be dedicated towards the development of an ETL process, this process will be developed to extract and load the data from the original data source and then convert it into a refined form to perform data analysis and further tasks. To complete all the objectives, python programming language will be used. Jupyter notebook is an Integrated Development Environment (IDE) available for this language and will be used for

coding and programming purposes as it provides a good environment for generating appropriate visualization during the programming process. **Data from MIMIC-III [1] [6] dataset will be used.** The various attributes as well as the advantages of using this dataset will be discussed in detail in the following chapters.

## 1.1 Motivation

The question that must be addressed first is: **Why focus on heart failure patients?**

**Cardiovascular disease (CVD) or heart disease** is one of the leading causes of death in Europe. It is responsible for **3.9 and 1.8 million deaths in Europe and EU (European Union) respectively**. This number accounts for 45% of all deaths in Europe and 37% of all deaths in the EU. In 2015, there were just under 11.3 million new cases of CVD in Europe and 6.1 million new cases of CVD in the EU. The treatment of a heart failure patient requires a hefty price to be paid out. Overall CVD complications are estimated to cost the EU economy an amount of €210 billion a year. Of the total cost of CVD in the EU, about 53% (€111 billion) is due to health care costs [7]. The Continent is also expected to face more challenges in the upcoming decades such as: **aging population, migration as well as risks of new endemics/pandemics compounded with forever increasing cost of clinical technologies**.

After looking at the above numbers, it can be concluded that heart failure patients lay a heavy burden on hospital performance. Thus, predicting the end points for a patient can serve as a critical performance indicator and hence provide deep insight in predicting the outcome for a patient based on vital parameters recorded during their stay in the hospital.

This study has been carried around real time clinical data, this can in turn prepare the hospital staffs or caretakers for a possible future outcome for a patient and hence indicate them in advance regarding any immediate actions that will be required to be completed.

The Ponemon Institute in 2012, estimated that 30% of all the digital data that is stored in the world is associated to healthcare data [8]. The extraction of relevant data from such a large database can be considered as the fastest, the most efficient as well as most inexpensive method to lower the burden on hospitals and at the same time improve their overall performance. This in turn will also include other **benefits** such as: increased patient satisfaction and overall improvement of the healthcare sector.

## 1.2 Objective

The current study can be shortly described as a study to **analyze hospital data to predict mortality or exacerbation rate for a patient admitted in the ICU and has been diagnosed with heart failure.** After thorough study and evaluation of machine learning models, a patient monitoring system will be developed that can enable the medical staffs as well as caretakers of a patient to address any risk increasing or decreasing factors at an early stage.

Thus, the primary objectives for this study can be summarized as:

- Define an ETL structure to **automate** the process of extraction of relevant data from this dataset.
- Apply **labelling** for the end points on the extracted data.
- **Evaluate** various machine learning models to check whether the extracted data can be used to make relevant machine learning based predictions.
- Finally, when all the above-mentioned steps have been completed and results have been deemed admissible, **a software** will be developed that can be used in a clinical setting.

## 1.3 Thesis Structure

This section describes the various chapters that will be followed in order for this report:

- **Chapter 2: Literature Review:** This chapter provides an overview of existing studies related to the current research problem.
- **Chapter 3: Methodology:** This chapter describes the MIMIC-III dataset in detail. This chapter takes into account the kind of data on which this dataset has been built on and why this dataset has been chosen for the ongoing study. This chapter also highlights the potential parameters that have been derived from this dataset.

  This chapter then describes the ETL process that has been followed to extract the data. Finally, this chapter ends with the analysis of the final data that will be used for machine learning purposes.

- **Chapter 4: Results:** This chapter describes the evaluation of various machine learning models that have been developed on the final dataset for prediction purposes. This chapter then describes the selection criteria for choosing the best machine learning model.
- **Chapter 5: Software Development:** This chapter describes the software (patient-monitoring system) developed for this study. This chapter also describes the underlying software architecture and the evaluation of results that can be obtained from this software.
- **Chapter 6: Conclusion and Future Outlook:** This chapter presents the final conclusion to the research problem being addressed, drawbacks and possible areas of improvement.

All these chapters are further followed by appendix A and B to look for additional information.

# Chapter 2

# Literature Review

With the abundance and the availability of EHR data in large volumes, numerous studies have been carried out in the field of application of AI in the healthcare sector. A significant amount of research around MIMIC-III dataset has focused on developing machine learning models for predicting patient mortality, readmission, predicting length of stay of a patient in the hospital or the likelihood of septicemia. However, most of these studies have focused on explaining an outcome by considering possible co-morbid reasons associated with a patient. Such studies have also left out the combination of demographic features (static features) like marital status and gender with vital signs associated with a patient like blood pressure or oxygen saturation to predict the outcome.

**Prediction model of in-hospital mortality in intensive care unit patients with heart failure: machine learning-based, retrospective analysis of the MIMIC-III database** [9] aimed to develop various prediction models for hospital mortality for heart failure patients such as XGBoost and LASSO regression. These models produced above average results for predicting the mortality of such patients by using static signs such as age and gender, and also incorporated vital signs like blood pressure and oxygen saturation. This study considered average lab results for such patients like: urine output and also considered the impact of other co-morbid factors like hypertension, diabetes etc. To accommodate all these values, these values were averaged for the whole stay of the patient at once. Mean imputation technique was used to handle the missing values.

**Early hospital mortality prediction using vital signals** [10] aimed at evaluating machine learning models such as Logistic Regression, KNN etc. for predicting mortality for heart failure

patients. This study used a combination of demographic signs like age and gender with vital signs like blood pressure associated with a patient to be used in the machine learning models by providing them equal weightage. The objective of this study is to reduce the reliance of a machine learning model's predictions on the lab results provided by the hospital. This was done to make robust predictions by considering the fact that the generation of lab results can be time consuming and thus cause a delay in the decision-making process. It also removed the dependence of predictions by a machine learning model on co-morbid factors associated with a patient. However, it suffered a big **short coming**, this study took the average of all the vital signs for the whole stay of a patient at once, thus failing to provide an overall insight for the patients throughout their stay in the hospital, thus eliminating the probability for predicting the exacerbation associated with the patients.

**Independent effects of the triglyceride-glucose index on all-cause mortality in critically ill patients with coronary heart disease: analysis of the MIMIC-III database** [11] is a similar study where independent effects of the triglyceride-glucose index on all-cause mortality in critically ill patients with coronary heart disease was analyzed to predict mortality. This study also focused on predicting the mortality of a patient by considering a combination of the lab results and co-morbid factors associated with a patient.

All these studies that were completed by using MIMIC-III dataset to predict mortality for heart failure patients were based on using both static and non-static vital signs along with or without considering co-morbid reasons associated with a patient. The results of these studies laid the foundation that the vital signs from the heart are an indicator for predicting mortality by using machine learning models.

The current study is focused on overcoming the short comings of the related works by predicting the possible causes of end points (exacerbation or death) for a patient by evaluating the **overall admission cycle of a patient. This will be done by averaging all the vital signs associated with a patient on a daily basis to provide a complete overview.** Moreover, the studies that have been mentioned here have also failed to provide the explanation of an outcome. The current study will try to bridge this gap by utilizing XAI methods to explain the outcome associated with a patient. This can prove to be significant factor in the decision-making process. The current study is focused on predicting the outcome for a patient by considering only demographics and vital signs associated with a patient. This was done to eliminate the reliance of

the results of this study on possible co-morbid reasons and results of lab-based tests associated with a patient.

Averaging of the vital signs associated with a patient was done to provide exacerbation rate for a patient (a prominent outcome that was not considered in similar studies). This study finally ends with the demonstration of a software prototype that can be deployed in real-world scenarios and can be utilized by the hospital staffs or caretakers for performing appropriate actions.

# Chapter 3

# Methodology

The MIMIC-III dataset will be described in detail in this section. This includes the underlying structure, content and points of interest that makes this dataset relevant for the ongoing study on heart failure patients. Once an understanding for the underlying structure for this dataset has been established, this chapter will further describe the methodology followed for the ETL process and finally concludes with the data analysis on the extracted data.

## 3.1 About MIMIC III

Medical data must be handled with utmost care due to privacy concerns. Due to this reason, it is often not possible to get such data directly from a hospital or a clinic and perform data analysis on it. Even if real-time data is available from a hospital or a clinic, it is often constrained with a **Non-Disclosure Agreement (NDA)**, this forbids the user from publicly publishing the results.

To handle such scenarios, the **MIMIC** [1] dataset that has data collected from **Beth Israel Deaconess Medical Center**, has been made publicly available for researchers worldwide. The link for obtaining access to this dataset is:

`https://mimic.physionet.org/gettingstarted/dbsetup/`

The underlying architecture for MIMIC-III dataset can be inferred from figure 3.1. From figure 3.1, it can be inferred that to create this dataset, the data underwent rigorous processes to refine it for research purposes. Few inferences that can be drawn from this architecture (figure 3.1) are:

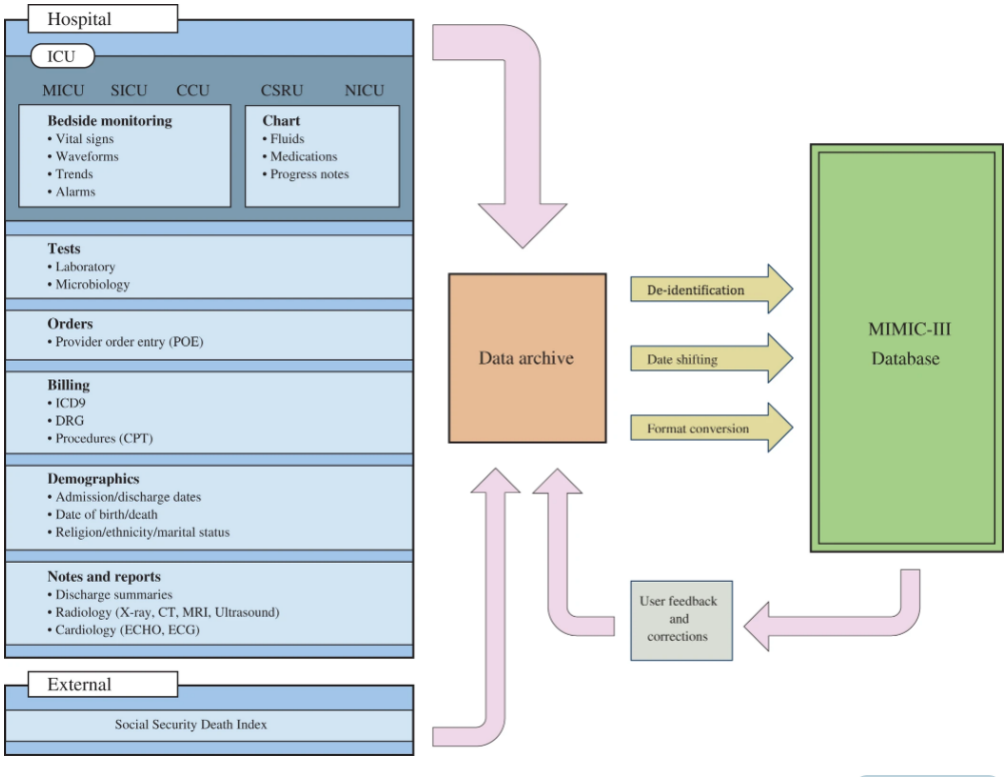


Figure 3.1: MIMIC model adopted from [1].

- This dataset contains information from various **ICU stages** like MICU (Medical ICU), SISU (Surgical ICU), CCU (Cardiovascular ICU) and NICU (Neonatal ICU), thus laying importance on the overall admission cycle for a patient.

- This dataset contains real time **bedside monitor** values like: vital signs, alarms, wave forms etc. along with **charted events** like medications given to a patient, fluid input or output from a patient etc.

- Apart from having the above-mentioned critical information, this dataset also contains billing information from the hospital, thus emphasizing the fact that the information present in the dataset is authentic and can be cross-referenced.

- The major advantage of using this dataset lies in the fact that it contains information about a patient from outside the stay in the hospital in the form of **Social Security Death Index**. This information can be useful in a scenario where a researcher would like to track the information on a patient's condition after the patient has been discharged from the hospital. This information can be used to make predictions for the near future like

readmission or mortality for the patient.

All the information in this dataset has been **de-identified** by the providing agency to adhere to the data privacy laws by shifting the original dates in the dataset. However, the magnitude of the difference between the dates has been left intact to maintain integrity. The MIMIC dataset has also undergone through **numerous feedback** cycles to improve the overall quality of the dataset. Due to these iterations, over the years various versions of this dataset have been released, with each version overcoming the short-comings of the previous version.

As clinical data must be handled with utmost care, a user must undergo a formal process to obtain the MIMIC dataset. This process can been described as:

- Register themself at physionet website.
- Complete a data privacy handling training.
- Sign a non-disclosure agreement.

The providers of MIMIC dataset allow a user to publicly publish the results of their studies. This information can also be found on the following link:

https://mimic.physionet.org/gettingstarted/access/

## 3.2 MIMIC III Main Tables

The MIMIC dataset contains information from **various sources** from the hospital, therefore it has been segregated into various tables/files, with each of them indicating a certain attribute about a patient for their stay in the hospital.

The MIMIC-III dataset contains **26 linked tables (files)** and contains various identifiers that is unique to each table. A flow chart linking the required tables will be presented in the next sections. This chart will display the relationships used among the tables for creating the final dataset. Therefore, it is a pre-requisite to have background knowledge about this dataset. Table 3.1 provides brief and relevant information about the tables/files provided in the MIMIC-III dataset.

| Table Name | Summary |
|---|---|
| ADMISSIONS | Provides first admission details about a patient. |
| CALLOUT | Provides information about the ICU discharge planning. |
| CAREGIVERS | Provides information about the kind of caretaker. |
| CHARTEVENTS | Provides all the Charted information about a patient. |
| CPTEVENTS | Provides the billing information regarding the treatment. |
| D_CPT | Provides information of Current Procedural Technology. |
| D_ICD_DIAGNOSES | Provides the final diagnosis based on the ICD-9 code. |
| D_ICD_PROCEDURES | Provides description of the ICD-9 codes. |
| D_ITEMS | Provides ITEMIDs for measurements recorded. |
| D_LABITEMS | Provides the description of the MIMIC ICU database. |
| DATETIMEEVENTS | Contains all the Date Time events. |
| DIAGNOSES_ICD | Provides description of procedures done on ICD-9. |
| DRGCODES | Provides the description of drug codes. |
| ICUSTAYS | Provides information regarding the ICU stay. |
| INPUTEVENTS_CV | Provides the input events from CV. |
| INPUTEVENTS_MV | Provides the input events from MV. |
| LABEVENTS | Provides information regarding laboratory tests for a patient. |
| MICROBIOLOGYEVENTS | Provides Microbiology information for tests. |
| NOTEEVENTS | Provides information regarding the free notes. |
| OUTPUTEVENTS | Provides the information regarding output by a patient. |
| PATIENTS | Provides hospital independent data. |
| PRESCRIPTIONS | Provides medication entries. |
| PROCEDUREEVENTS_MV | Provides information of the procedures performed in MV. |
| PROCEDURES_ICD | Provides ICD specific procedures provided to a patient. |
| SERVICES | Provides information for services provided to a patient. |
| TRANSFERS | Provides the physical location of a patient. |

Table 3.1: MIMIC 3 table description.

## 3.3 Feature Selection Criteria

Now that the background information about the MIMIC-III dataset has been established, the focus of this study will shift towards the variables that must be extracted and considered for the ongoing research problem. Once such variables have been identified, the ETL process can be established. The ETL process will be described in the next sections. The variables considered for this study are:

- **Vital Signs:** Oxygen Saturation, Blood Pressure, Heart Beat and ECG (Electrocardiogram).
- **Demographics:** Age, BMI, Gender, Disease Duration and Marital Status.

## 3.4 Constructing the ETL

The first step that must be completed before performing any data analysis task is to obtain the raw data and then process it to obtain consistent data that is suitable for analysis purposes. In general, the major part of the time devoted for completing a research problem goes to process of data cleaning and preparation [12].

Figure 3.2 illustrates the generic steps followed for creating an ETL process [13] [2]:

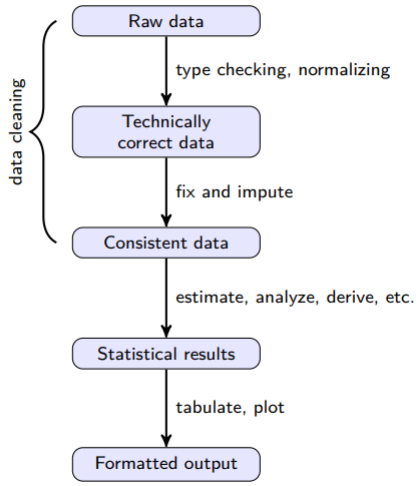


Figure 3.2: Statistical Analysis Value Chain adopted from [2].

## 3.5 Context

From figure 3.2, it can be inferred that the first two steps after obtaining raw data (in this case MIMIC-III) is to find **technically relevant data** that is extracted based on knowledge from the database and then **fixing** this data. These two steps combined are known to be as the data cleaning step. This process of obtaining technical data will be used to obtain the final dataset with required features. To obtain these required features, it will be necessary to merge multiple tables (data sources) [14].

This approach will in turn result into a final table where each row corresponds to various demographics and vital signs of a patient based on an aggregation (daily basis) level. The former kind of variables being static throughout the newly built table and the latter being dynamic based on the aggregation level it corresponds to.

There are various techniques available that can be used to perform this process of data cleaning and transformation. For this study, pandas [15] library in the python programming language will be used. All the files from the MIMIC-III dataset can be downloaded on a local machine in the form of .csv files, where the process of data cleaning will be performed. The process of constructing the ETL will be demonstrated in the upcoming sections.

## 3.6 Technical Details

The first step towards building the final dataset for this study is to understand the requirements and the kind of data that is underlying in the data source that will be used. This must be done to understand the different nature and the capabilities of each table that makes up the entire MIMIC-III dataset and hence locate the tables where a required variable is located. A brief overview of all the tables has been mentioned in the previous sections.

Among all the 26 .csv files in MIMIC-III dataset, they must be first classified in the order of their importance. Several files will be used for this study, some are very large (more than 30 gigabyte in size), while others are small or medium sized.

The small and medium files don't possess any difficulties while reading them through pandas for data merging or analysis purposes. However, some of the constituent data source files are very large and hence cannot be read as a single dataframe due to limitations of the local machine

as it can produce out of memory errors depending upon the capacity of the local machine.

Instead of reading the whole file at once in the form of a dataframe, it can be read in the form of parts of large sizes. These parts can be iterated based on the required conditions i.e. one at time. The size of these parts will thus be under the local machine's computing threshold. The specifications of the machine used for this study are: **16 GB of RAM, Intel Core i5 processor and 256 GB of NVMe SSD**. For reference the size of each part used for this study is **10 million** rows, these parts were read one at a time and the total number of iterations performed for reading the entire file was 34.

Once all the filtered data has been obtained from these parts, the data from each part can then be merged again to create a smaller and filtered dataframe based on the requirements. Among the various advantages of using python for this study, the following will be of the utmost importance:

- Creating fast and efficient dataframes by combining multiple .csv files on the local machine.
- Aggregating the .csv files for creating a dataframe that reflects different time horizons for predicting different labels for a patient.
- Tools for manipulating the different columns/rows in the dataframe from the python programming language. These tools can be later used for creating a final dataframe for machine learning purposes.

## 3.7 Completing the ETL

Figure 3.3 and the steps mentioned in this section will illustrate the complete ETL process that has been followed. The description of the tables mentioned in this section can also be referred from the previous sections.

Figure 3.3 illustrates the tables that have been used to create the final dataset that contains all the important variables and columns required for the ongoing study.

The MIMIC-III dataset contains the data of about **60,000 patients** who were admitted in Beth Israel hospital between 2001 to 2012 [1]. However, these patients could have been diagnosed for various diseases like: kidney failure, cancer, lung infection etc. **To identify the heart**

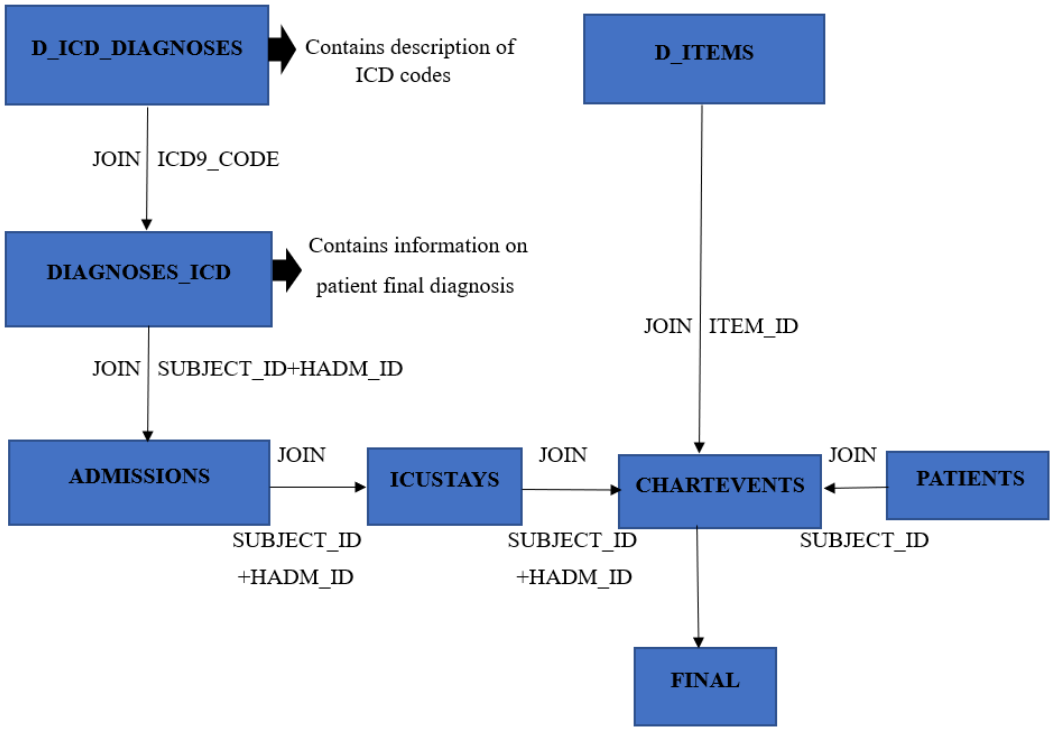


Figure 3.3: ETL.

**failure patients, the ICD-9 codes will be used. The ICD codes are used as diagnostic codes for classifying diseases [16].** By referring to the description of the tables (table 3.1) in this dataset, the following steps must be followed to complete the ETL process:

- The table D_ICD_DIAGNOSES contains the final diagnosis of a patient during a visit to the hospital by using ICD-9 codes.
- The ICD-9 codes must then be referred to those present in the table DIAGNOSES_ICD to identify the exact diagnoses of a patient in the hospital.

  Hence, to identify a patient's diagnosis, the ICD-9 code associated with a patient must be identified first [17].

  The ICD-9 codes used for this study have been reviewed by two reviewers to find all the plausible patients.
- Once the ICD-9 codes for the heart failure patients have been identified, they must be cross-referenced with the ADMISSIONS table.

This step must be performed to obtain the static parameters associated with a patient such as: discharge time, admit time etc.

- The filtered patients obtained by performing the above steps must then be cross-referenced with the ICU_STAYS table, this step must be performed to make sure that the patients obtained from the admissions table had been admitted in the ICU and the vital signs associated with these patients must be extracted during their stay in the ICU.

  The key point that must be kept in mind while extracting the desired set of patients is to use the identifier for a particular patient (SUBJECT_ID) with their unique Hospital Admission ID (HADM_ID).

  This in necessary due to the fact that a patient could be admitted more than once and each time the SUBJECT_ID would be the same, however the HADMID would change.

- Next to identify variables such as type of heart rate, oxygen saturation value and blood pressure, the D_ITEMS tables must be referenced.

  This table shows how a particular vital sign has been associated with a patient by using an item code.

  These ITEM_IDs have also been segregated based upon the data-source (CV or MV) from which they have been derived.

- Once all the required ITEM_IDS have been identified to filter out the vital signs, they must be cross referenced with the CHARTEVENTS table while keeping the required SUBJECT_ID and HADM_ID intact. For this study such vital signs have been aggregated on a daily basis. This aggregation has been performed by averaging all the values of a vital sign on a daily basis.

- The dataset obtained by following the above steps must then be merged with the PATIENTS table (on the basis of SUBJECT_ID) to obtain demographics like GENDER.

  Since, the gender of a patient will remain the same irrespective of the HADMID, this step can be performed based on SUBJECT_ID only.

Once all these steps have been completed, the required dataframe that contains: heart failure patients admitted in the ICU along with vital signs and demographics can be obtained.

Table 3.2 represents the ICD codes selected for this study out of a total of **14,564** codes available in MIMIC-III dataset.

| **ICD9_CODE** | **LONG_TITLE** |
|---|---|
| 4281 | Left heart failure |
| 42821 | Acute systolic heart failure |
| 42822 | Chronic systolic heart failure |
| 42823 | Acute on chronic systolic heart failure |
| 42831 | Acute diastolic heart failure |
| 42832 | Chronic diastolic heart failure |
| 42833 | Acute on chronic diastolic heart failure |
| 42841 | Acute combined systolic and diastolic heart failure |
| 42842 | Chronic combined systolic and diastolic heart failure |
| 42843 | Acute on chronic combined systolic and diastolic heart failure |

Table 3.2: ICD-9 Codes used.

Tables 3.3 and 3.4 represent the ITEMS_IDs that have been selected from MV and CV databases.

| ITEM_ID | DB Source: MetaVision |
|---|---|
| 224167 | Manual Blood Pressure Systolic Left |
| 227242 | Manual Blood Pressure Diastolic Right |
| 227243 | Manual Blood Pressure Systolic Right |
| 227537 | ART Blood Pressure Alarm - High |
| 227538 | ART Blood Pressure Alarm - Low |
| 224751 | Temporary Pacemaker Rate |
| 224639 | Daily Weight |
| 224643 | Manual Blood Pressure Diastolic Left |
| 220045 | Heart Rate |
| 220046 | Heart rate Alarm - High |
| 220047 | Heart Rate Alarm - Low |
| 220050 | Arterial Blood Pressure systolic |
| 220051 | Arterial Blood Pressure diastolic |
| 220052 | Arterial Blood Pressure mean |
| 220056 | Arterial Blood Pressure Alarm - Low |
| 220058 | Arterial Blood Pressure Alarm - High |
| 220224 | Arterial O2 pressure |
| 220227 | Arterial O2 Saturation |
| 226512 | Admission Weight (Kg) |
| 226707 | Height |
| 220277 | O2 saturation pulseoxymetry |
| 223769 | O2 Saturation Pulseoxymetry Alarm - High |
| 223770 | O2 Saturation Pulseoxymetry Alarm - Low |

Table 3.3: MV ITEMIDs.

| ITEM_ID | DB Source: CareVue |
|---|---|
| 442 | Manual BP [Systolic] |
| 443 | Manual BP Mean(calc) |
| 8440 | Manual BP [Diastolic] |
| 51 | Arterial BP [Systolic] |
| 8368 | Arterial BP [Diastolic] |
| 52 | Arterial BP Mean |
| 8555 | Arterial BP #2 [Diastolic] |
| 6701 | Arterial BP #2 [Systolic] |
| 6702 | Arterial BP Mean #2 |
| 53 | Arterial Pressure |
| 490 | PAO2 |
| 646 | SpO2 |
| 733 | Weight Change |
| 3580 | Present Weight kg |
| 762 | Admit Wt |
| 763 | Daily Weight |
| 3693 | Weight Kg |
| 5820 | SpO2 Alarm [Low] |
| 8554 | SpO2 Alarm [High] |
| 211 | Heart Rate |
| 3494 | Lowest Heart Rate |
| 1394 | Height Inches |

Table 3.4: CV ITEMIDs.

## 3.8 Labelling the Data

After following the steps mentioned in the previous sections, a dataframe has been created that reflects a patient's vital signs and demographics on an aggregation level. The last step remaining here is to predict the **time horizon for the death of a patient**. The time horizon chosen for this

study is **1 day** i.e. a patient who can die the next day.

To obtain this labelling, the last chart time for a patient was matched with the death time of that patient that was recorded in the hospital. This was done to ensure that only the patients who died during their ICU stay have been included in this study. By following these steps, an output label has been calculated that signifies the death label of 1 to indicate that the patient dies within 1 day and hence appropriate decision can be taken in due time.

**For a better understanding:** A scenario can be imagined where a patient was admitted for 21 days in the ICU and died on the 21st day. As the dataframe being developed here has been aggregated on a daily level, the output (date of death label) will be **flagged** to 1 on the 20th day for this patient and 0 for all the remaining days.

## 3.9 More about CV and MV database

The MIMIC dataset contains more records for the MV database compared to the CV database due to the records provided by the hospital. However, to have an unbiased view of a patient, it is necessary to consider all the records recorded for a patient. Hence, to accommodate these records it is necessary to find the corresponding values from both the MV database and CV database and then calculate their average values to arrive at the final record that will be used for a particular patient. The corresponding identifiers used from CV and MV database have been mentioned in the previous sections.

This approach can also lead to a situation where a record is present in only one of the databases. In such a situation, the corresponding record from the database where the record is available will be used. This process can also lead to a situation where no record is present in either MetaVision or CareVue database, in such a situation multiple techniques will be used to **handle the missing values**. Such techniques will be discussed in the following sections.

## 3.10 Data Preparation

Patients who were older than 89 years of age at the time of their admission in the MIMIC dataset have had their date of birth shifted by 300 years to conceal their real age and hence comply with regulatory guidelines. As patients who are older than 89 years are easy to identify

based on their real age and diagnostic history. To handle the cases where age cannot be directly calculated based on their date of birth, the age for such patients has been adjusted accordingly by setting them to 90 years [18].

Such an assumption was made on purpose as numerical age is considered as one of the potential members of the feature matrix. Therefore, age could not be dropped as it was lying in an exterior range. Moreover, incorporating a variable that is clearly lying outside the plausible range is considered as noise and thus reduces the performance of machine learning models [18].

## 3.11 Feature Engineering

A total of **4,257 heart failure patients** were separated from 60,000 patients for this study after following the ETL process. Once the required dataframe has been created and contains all the necessary features required for prediction and has the vital signs of the patients averaged on a daily basis, it is necessary to handle the numerical and categorical values.

In the current research problem, marital status and gender are categorical values and are static values like numerical age. One-hot encoding was performed on the categorical values to convert these alphabetical values to numerical values for machine learning purposes. BMI, disease duration and vital signals such as heart rate, blood pressure and oxygen saturation are numerical values and are non-static in nature.

A percentage wise visualization of the missing values can be referred by looking at table 3.5, this table shows the missing values in the original dataframe. These values have not been subjected to any technique that handles the missing data.

As can be inferred from table 3.5, the parameters **height and daily weight are missing a large number of values** (91% and 63% of values are missing respectively). Since, both these values are necessarily required to calculate the BMI of a patient, it is important to handle the missing values for both these parameters first.

Before applying any technique to handle missing data (imputation), it is necessary to check the amount of missing values i.e., it must be checked whether the number of values missing is below a threshold. If the missing values are very high (91% in the case of height), then the imputation techniques can bring bias and hence degrade the machine learning results. Therefore, all the variables used in this study were visualized for their relationship with height and daily

| Variable Name | Missing Value (%) |
|---|---|
| SUBJECT_ID | 0 |
| HADM_ID | 0 |
| Disease_Duration | 0 |
| Height | 91.56 |
| Flag_Height | 0 |
| Daily Weight | 63 |
| Flag_Daily_Weight | 0 |
| AGE | 0 |
| GENDER_M | 0 |
| REL_DAY | 0 |
| Heart_Rate | 0.23 |
| Heart_Rate_Alarm_Low | 40.42 |
| Heart_Rate_Alarm_High | 40.42 |
| Oxygen_Saturation | 0.34 |
| Oxygen_Saturation_Alarm_Low | 8.9 |
| Oxygen_Saturation_Alarm_High | 8.9 |
| Arterial_Blood_Pressure_Systolic | 49.41 |
| Arterial_Blood_Pressure_Diastolic | 49.41 |
| MARITAL_STATUS_DIVORCED | 0 |
| MARITAL_STATUS_LIFE_PARTNER | 0 |
| MARITAL_STATUS_LIFE_MARRIED | 0 |
| MARITAL_STATUS_LIFE_SEPARATED | 0 |
| MARITAL_STATUS_LIFE_SINGLE | 0 |
| MARITAL_STATUS_LIFE_UNKNOWN | 0 |
| MARITAL_STATUS_LIFE_WIDOWED | 0 |

Table 3.5: Missing values in the initial dataframe.

weight, some visualizations can be referred from Figure 3.4:

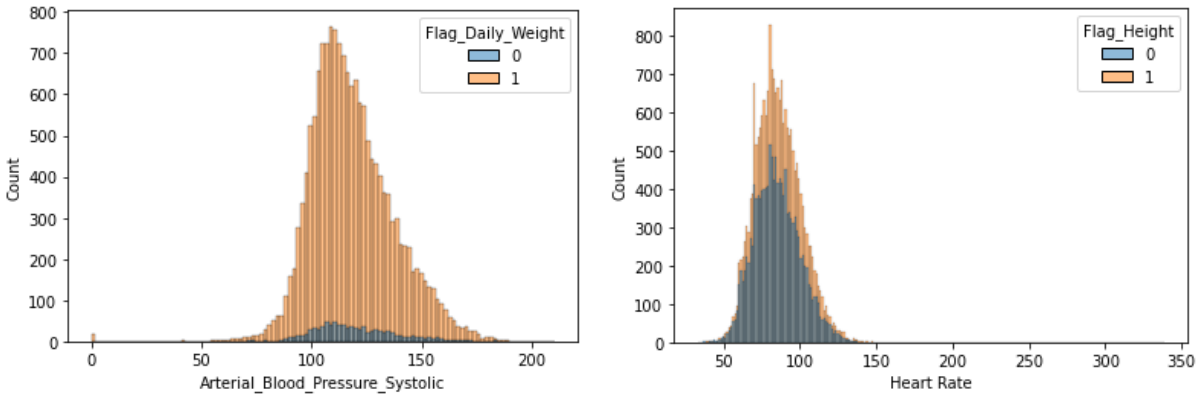


Figure 3.4: Distribution of systolic arterial blood pressure with daily weight on the left and distribution of heart rate with height on the right.

To check the distribution of daily weight and height with other features, two flags were created. These flags indicated whether there are values recorded for daily weight and height for a patient. Therefore, a flag value of 0 for daily weight (Flag_Daily_Weight) indicates that there was no value recorded for that patient for daily weight and vice versa for flag value of 1. Similarly, a flag value of 0 for height (Flag_Height) indicates that no value for height was recorded for that patient and vice versa.

An inference that can be made from figure 3.4 is that the missing values for both daily weight and height for all the patients are uniformly distributed for systolic arterial blood pressure and heart rate. Distribution for other values in the feature matrix can be referred from Appendix A.

However, before the removal of any values, it must be ensured that the removal of such values will not cause any bias in the final data. To ensure that no such bias will be caused, Table1 (summary statistics) [19] technique was used for checking the bias (reference in appendix A). After looking through the Table1 values and the visualizations in figure 3.4 as well as the visualizations in Appendix A, it can be concluded that the patients for whom the flag value for daily weight and height is 0 can be dropped without bringing any bias in the data.

Now that it has been established that the patients for whom the values for height and daily weight are missing values and have been removed, the next step would be to start filling the remaining missing values for all the parameters. For filling these missing values, **forward filling and backward filling techniques** will be used first. This was done to ensure the filling

of missing values with the last accurately recorded value for a patient [20].

Forward filling technique works by filling the missing values in a column by replacing the missing values by the last recorded value in the corresponding column. Similarly, backward filling technique works by replacing the missing values of a column from the next available value in the corresponding column. This was done to ensure that if the medical staffs are facing a situation where the vital signs associated with a patient are missing and need to be fixed before performing any prediction process, the easiest way to handle such missing values will be to replace them by the last recorded accurate value of a patient. The filling techniques thus ensure that the missing values that will be replaced are replaced by the values that represent the most accurate values that can be associated with a patient [20].

As **height** is a static parameter for a patient, the missing values for height were handled by **forward and backward filling-based techniques by grouping the patients by SUBJECT ID**. Similarly for daily weight and other parameters such as heart rate, oxygen saturation and all kinds of arterial blood pressure, the missing values were filled by **forward filling technique by grouping the patients by their SUBJECT ID**.

For handling values in alarm-based columns like heart rate and oxygen saturation, their values were replaced by binary values of 0 and 1. This approach was used to incorporate high readability for machine learning models. A value of 1 was assigned to a column value when a numerical value was recorded in the respective column. A value of 0 was assigned when no value was recorded in the respective column. Therefore, if a numerical value was recorded for a patient for heart rate, it was replaced by 1 and if there was no numerical value recorded for heart rate for a patient, it was replaced by 0.

After these filling techniques have been implemented, the dataframe still contains some missing values that can be referred from the table 3.6. As can be inferred from table 3.6, all the parameters in the feature matrix have less than 30% missing values. This is considered to be below the threshold for applying imputation techniques for handling missing values in the field of medical sciences [20]. Hence, to completely handle the missing values, iterative imputation technique will be implemented. Iterative (Regression) imputation [21] is a technique in which the missing values are replaced by **performing multiple iterations** over the samples of the available data and then finally replacing them by their average.

| **Variable Name** | **Missing Value** (%) |
|---|---|
| SUBJECT_ID | 0 |
| HADM_ID | 0 |
| Disease_Duration | 0 |
| Height | 0 |
| Flag_Height | 0 |
| Daily Weight | 11.28 |
| Flag_Daily_Weight | 0 |
| AGE | 0 |
| GENDER_M | 0 |
| REL_DAY | 0 |
| Heart_Rate | 0.23 |
| Heart_Rate_Alarm_Low | 17.14 |
| Heart_Rate_Alarm_High | 17.14 |
| Oxygen_Saturation | 0.34 |
| Oxygen_Saturation_Alarm_Low | 5 |
| Oxygen_Saturation_Alarm_High | 4.89 |
| Arterial_Blood_Pressure_Systolic | 18.06 |
| Arterial_Blood_Pressure_Diastolic | 18.06 |
| MARITAL_STATUS_DIVORCED | 0 |
| MARITAL_STATUS_LIFE_PARTNER | 0 |
| MARITAL_STATUS_LIFE_MARRIED | 0 |
| MARITAL_STATUS_LIFE_SEPARATED | 0 |
| MARITAL_STATUS_LIFE_SINGLE | 0 |
| MARITAL_STATUS_LIFE_UNKNOWN | 0 |
| MARITAL_STATUS_LIFE_WIDOWED | 0 |

Table 3.6: Missing values after implementing forward and backward filling techniques.

After all the data missing handling techniques have been implemented, the final dataset will be created that has no missing values. Before starting the evaluation of the performance of

machine learning models on this dataset (dataframe), it is necessary to **remove the outliers** from this dataset as it can cause bias and unexpected results in the metrics obtained from the machine learning models.

To obtain and remove outliers, the following technique was implemented:

- Find the inter-quartile range (IQR) for each dynamic value (blood pressure, oxygen saturation, heart rate and BMI).
- Remove any values that lie outside the 1.5*IQR-Lower Limit and 1.5*IQR + Upper Limit.

Once the above technique has been implemented in the dataset, all the outliers have been removed and the newly formed dataset can be used for machine learning purposes.

Figures: 3.5, 3.6, 3.7 and 3.8 represent some plots that represent the distribution of dynamics of the dataframe that will be used for this study.

## 3.12 Binary classification

Now that it has been established that the present study deals with binary output (0 and 1), it is necessary to establish an evaluation criterion for assessing the performance of machine learning models. The most commonly used metric to evaluate the performance is **Accuracy**, accuracy can be defined as:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

From the above formula, it can be inferred that accuracy is the ratio of number of correct predictions to the total number of predictions made by the classifier and thus ranges between 0 and 1. Accuracy can also be defined as:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Where **TP** stands for an instance that was actually positive (1 in the current study) and was correctly classified as positive by the classifier (machine learning model). **TN** implies a negative instance (0 in the current study) and was correctly classified as negative by the classifier (machine learning model). Thus, the numerator in the above formula implies the total number of instances that were correctly classified by the classifier (machine learning model).

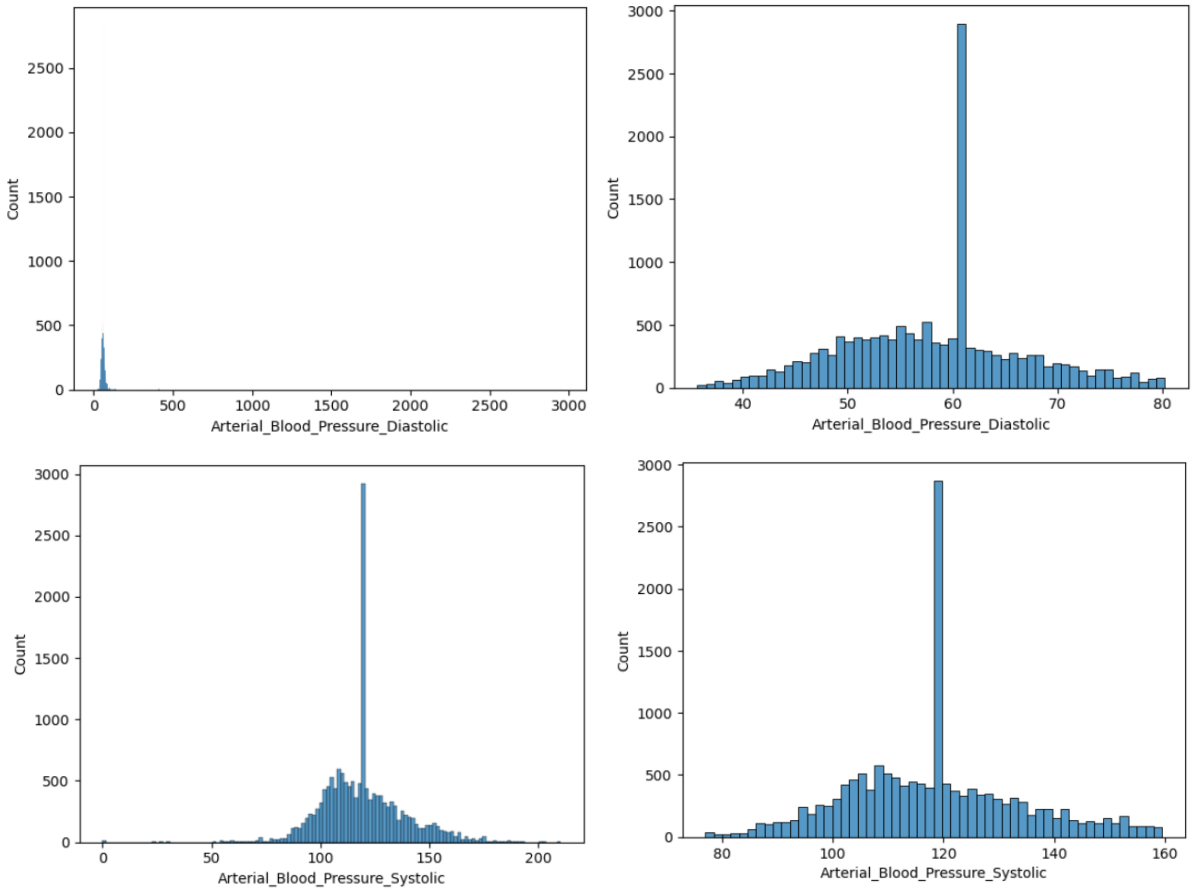


Figure 3.5: Arterial blood pressure with outliers on the left and without outliers on the right.

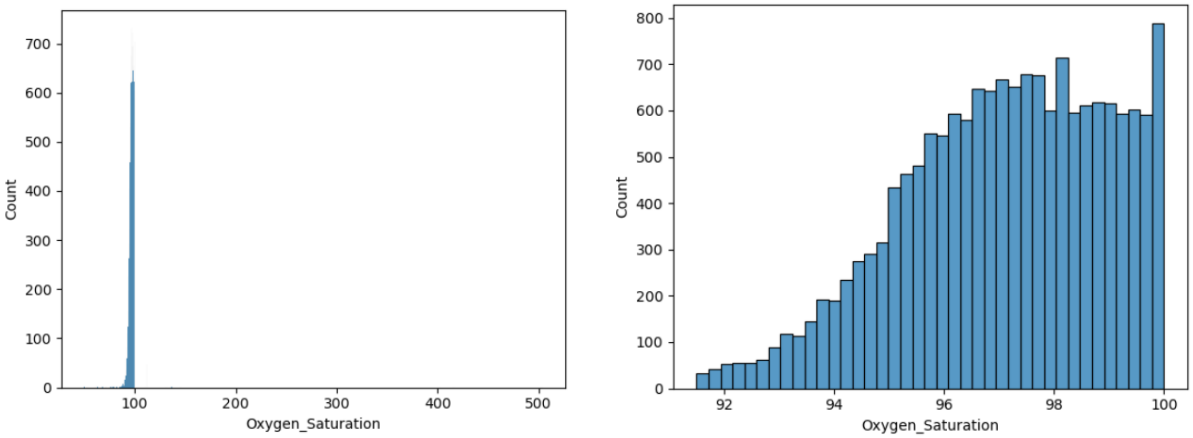


Figure 3.6: Oxygen saturation with outliers on the left and without outliers on the right.

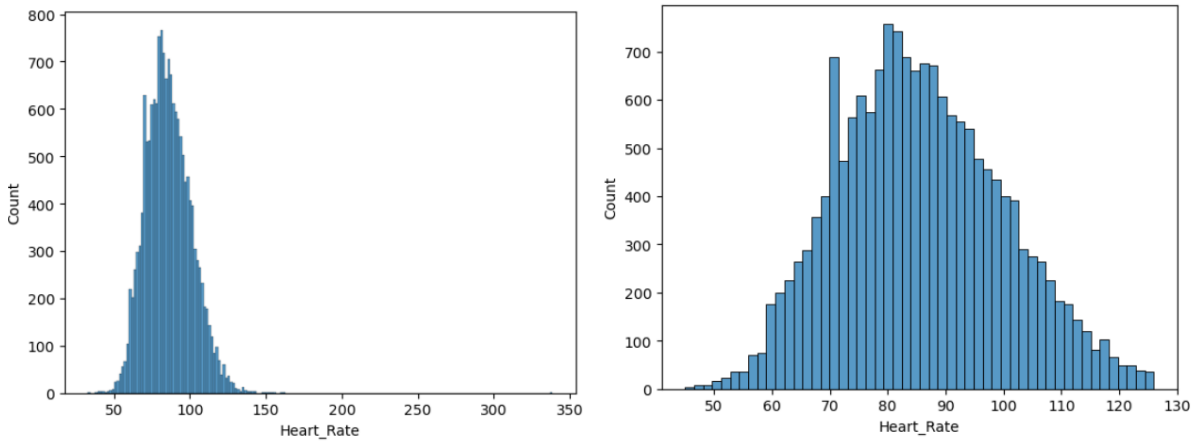


Figure 3.7: Heart rate with outliers on the left and without outliers on the right.

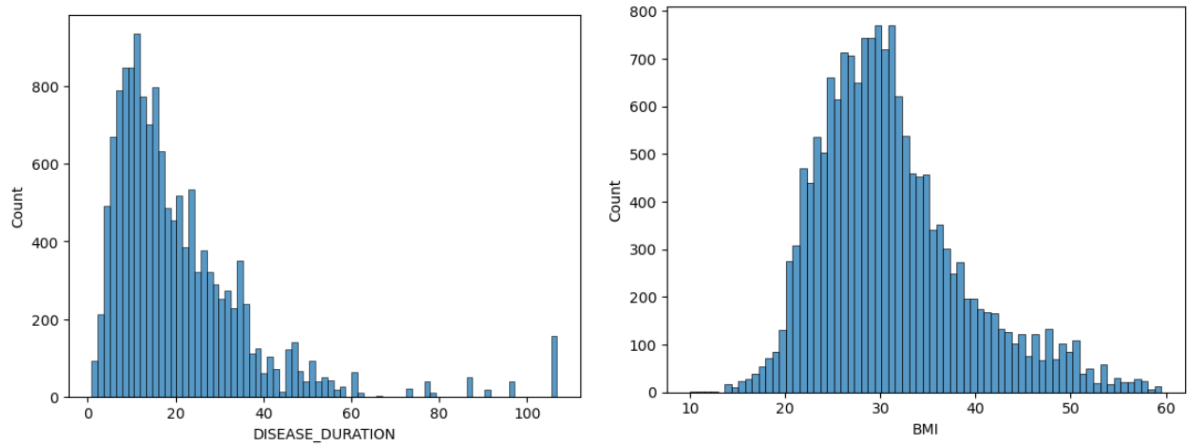


Figure 3.8: Disease duration on the left and BMI on the right for all the patients.

**FP** stands for an instance where the actual output was negative but the classifier classified it as positive and **FN** stands for an instance where the actual output was positive but the classifier incorrectly classified it as negative. A pictorial representation of a confusion matrix can be referred from figure 3.9. A confusion matrix is a table that can be used to define the performance of a classifier.

**Accuracy is a good metric for evaluation when the dataset (dataframe) being used is balanced** (i.e., the number of outputs is similar) or it's good even in terms of unbalanced dataset when the cost of misclassification is not high. However, the current study deals with a **fatal**

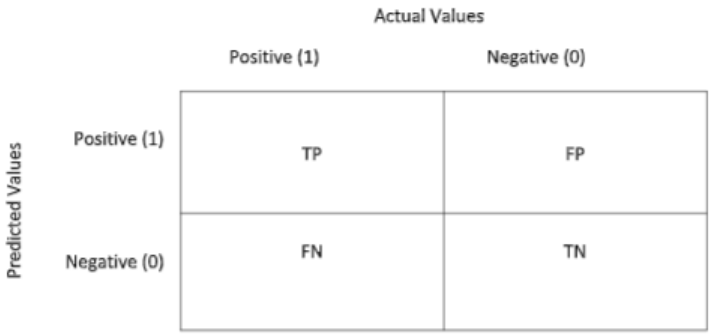


Figure 3.9: Generic Confusion Matrix adopted from [22].

**disease (heart failure)** where the cost of misclassification could be very high i.e., a scenario can be constructed where a machine learning model classified a sick person as healthy and thus remains undiagnosed. This scenario can ultimately lead to mortality due to the incompetency of the machine learning model.

## 3.13 Evaluation Criteria

The machine learning models that will be used for this research will be evaluated by using AUC ROC (Area Under Curve Receiver Operating Characteristic) [23]. This evaluation parameter has been chosen as it informs how good a model is able to differentiate between the classes (binary classification in this study), this implies that the higher the value of AUC, the higher the chances are of a model of being able to distinguish between the classes i.e., 0 and 1.

These values are based on **Sensitivity or True Positive Rate (TPR) and Specificity**. Sensitivity and Specificity can be defined as:

$$Sensitivity = \frac{TP}{TP+FN}$$

$$Specificity = \frac{TN}{FP+TN}$$

In an ROC (Receiver Operating Characteristic) curve, the FPR (False Positive Rate) is plotted along the x-axis and the TPR is plotted along y-axis at different cut-off points. The AUC plots generated for this study will be explained in the upcoming sections.

The problem statement that is being addressed in this study is utilizing an imbalanced dataset. The total number of patients who had output label of 1 turned out to be 20 out of a

total of 1706 patients being used in this study. In such a scenario, accuracy does not turn out to be a good evaluation criterion due to misclassification cost. In the field of medical sciences, predicting a sick patient (heart failure patients in this study) is of the utmost importance, as **the cost of diagnosing a healthy person will be much lower than not diagnosing a sick person who went into the category of a healthy person due to incompetency of a machine learning model** [12].

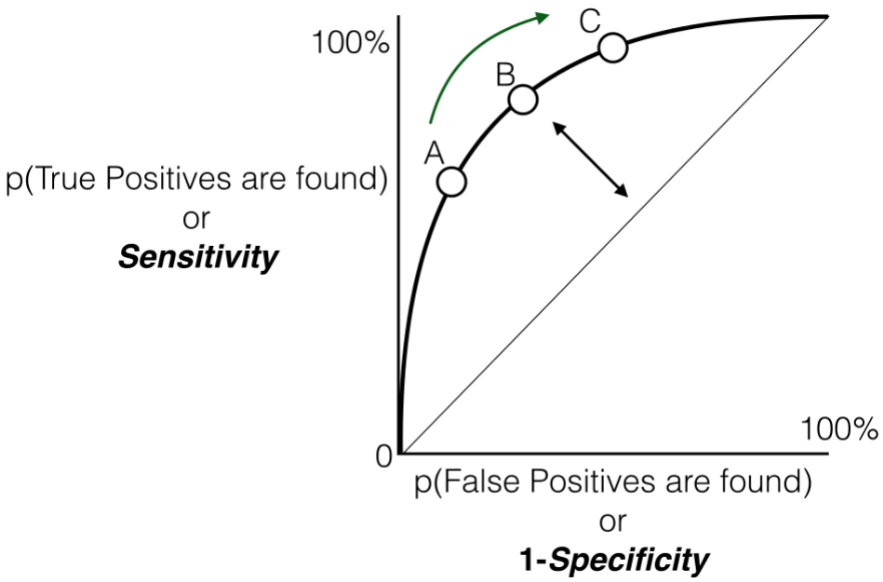


Figure 3.10: Generic AUC ROC plot adopted from [24].

AUC scores have been used as reporting criteria for binary classification in medical science scenarios such as binary classification for hospital mortality, readmission and multi-class classification in terms of length of stay etc. [18]. The results obtained for AUC scores for this study have been obtained by using **4-fold cross-validation technique**.

Other machine learning evaluation metrics that will be used:

$$Precision = \frac{TP}{TP+FP}$$

**Precision is defined as a metric that represents the number of positive class predictions that actually belong to the required positive class.**

$$Recall(Sensitivity) = \frac{TP}{TP+FN}$$

**Recall is defined as a metric that represents the total number of positive output values that are correctly predicted by the classifier (machine learning model).**

$$F1 = \frac{2*TP}{2*TP+FP+FN}$$

**F1 score is defined as a single metric that balances precision and recall values.**

## 3.14 Model Selection

To comply with time requirements in clinical settings, the machine learning models must be chosen wisely and must comply with the following requirements:

- Fast and efficient.
- The machine learning models can be trained in short time.

The following models have been chosen for this study:

- **Logistic Regression:** Logistic Regression [25] [26] is a type of a supervised machine learning algorithm that is primarily used for classification tasks. The primary motive to use logistic regression as a machine learning algorithm is to predict the probability of an instance to belong to a particular class. When used in classification algorithms, this algorithm is referred to as logistic regression as it relies on the output of linear regression function as input and uses a sigmoid function (formulated below) to estimate the probability of a given class.

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

- **Ridge Classifier:** A Ridge Regressor (Classifier) [25] [27] is an optimized and regularized version of a Linear Regression model. In such a model, the original cost function of the regressor is regularized in terms of the forces that are used to fit the data and keep the weights as low as possible.
- **KNN:** K-Nearest Neighbour (KNN) [25] [28] [29] is a supervised non-parametric learning classifier. The primary working basis of this classifier is to use the intuition of proximity to make classification or prediction decisions around the grouping of an individual data point. KNN can be used for both classification or regression problems.

- **Decision Tree:** Decision Tree (DecisionTreeClassifier) [25] [30] is supervised machine learning technique that can be defined as a free-flowing chart like tree structure. The nodes in such a tree/chart denote a test attribute from the given training dataset and each branch represents an output of a particular test and each leaf/terminal node represents a class label.

- **Random Forest:** Random Forest [25] [31] is a kind of ensemble learning technique that can perform both regression and classification tasks by utilizing multiple decision trees and using bootstrap and aggregation techniques that are commonly known as bagging. The primary idea of using Random Forest algorithm is to combine various decision trees to determine the final output rather than relying on the result of an individual decision tree. Therefore, Random Forest contains multiple decision trees as base learning models, where rigorous row and feature sampling are applied on the underlying dataset for each model, this is also known as bootstrapping.

- **SVM:** SVM [25] [32] [29] is a type of a supervised machine learning algorithm that can be used for both classification and regression tasks. The working principle of an SVM model is to find a hyperplane that can separate the input classes.

- **XGBoost:** XGBoost [25] [33] is an optimized distributed gradient boosting technique used for the efficient training of machine learning models. XGBoost works on the principle of ensemble learning i.e., it combines multiple weak models to produce a stronger model.

## 3.15 Hyperparameter Tuning

A machine learning model can be assumed to be a mathematical model that bears a number of parameters that are learnt by it based on the data it is subjected upon [34]. The first step for creating a machine learning model after deciding the input data is to train the model on this data and then fitting the model on the model parameters. These parameters can be assumed to be dependent (internal parameters) upon the selected model and data. However, there is another kind of parameters that cannot be directly learnt from the ongoing training process, such parameters are known as hyperparameters and are defined before the actual training process

starts. Thus, it can be inferred that hyperparameters when once defined are static in nature. These hyperparameters thus carry with them important information about a model such as the model complexity. Therefore, hyperparameters can be defined as a configuration that is external to the model and their values cannot be estimated from the data. **GridSearchCV** technique has been used for finding the best hyperparameters of the selected machine learning models. The GridSearchCV technique works by finding the best hyperparameters for a machine learning model by continuously iterating through a grid of pre-defined hyperparameters and then finding the hyperparameters that provided the best performance.

| **Model name** | **Final hyperparameters** |
|---|---|
| Logistic Regression | C: 0.01, multi_class: multinomial, penalty: l2, solver: newton-cg |
| Ridge Classifier | alpha: 0.5 |
| SVM | C: 0.1, gamma: scale, kernel: poly |
| Random Forest | criterion: gini, max_depth: 12, max_features: sqrt, min_samples_leaf: 4 |
| KNN | algorithm: auto, n_neighbors: 14, p: 1, weights: uniform |
| XGBoost | learning_rate: 0.1, max_depth: 5, min_child_weight: 4, n_estimators: 20 |
| Decision Tree | criterion: entropy, splitter: best |

Table 3.7: Hyperparameter table.

The table 3.7, shows the hyperparameters finally used. All the hyperparameters that were tested for this study have been uploaded in Appendix A.

## 3.16 Threshold Tuning

In terms of mathematics or statistical models, a threshold model is a model where a threshold value has been set to distinguish the output values of a model. Since this study deals with clinical data, it is important to choose a certain threshold where the cost of diagnosing a healthy patient is low compared to a situation where an unhealthy patient passes without any diagnosis and hence will be at risk of mortality.

This process of experimentation of various threshold values to find an optimum value based on the setting where machine learning models will be used is known as **threshold tuning**. Machine learning models are capable of predicting the probability of occurrence of a class

(binary output), this must be evaluated to get the predicted output to be mapped to a crisp class label.

Since the ongoing study deals with the mortality of the patients admitted and death label has been calculated 1 day in advance, it is possible to tune the threshold to find labels that can represent exacerbation of a patient and hence ease the process of decision making. By default, the value of threshold is set to 0.5, this means that all values greater than or equal to this threshold will be mapped to an output class and vice versa. Classification problems that have highly imbalanced output class distribution can result in poor performance if the threshold value has been set to 0.5. The following block will define the use of various threshold tuning techniques:

- **J-statistic:**

  **Youden's J statistic** [35] [36] (can also be referred as Youden's index) is a statistical method that can capture the performance of a dichotomous (two subsets that are jointly exhaustive or mutually exclusive) test. This index can be assumed to be a generalization of a multi class case and hence estimates the probability of an informed decision. The value of J statistic ranges between -1 and 1 (inclusive) and results in value of 0 when a diagnostic test produces the same number of positive results for groups with and without disease (i.e., the test turned out to be worthless) [37] [38]. The major advantage of using J statistic method is that it assumes equal weightage for false positive and false negative cases, so that all the tests performed using this statistic with the same value of the index provide the same proportion of total misclassified results [37].

- **Manual Tuning**:

  Finally comes a threshold setting that depends on the scenario where the machine learning models are actually being used i.e., the user can manually pick a setting where a particular case having a certain threshold will be classified with output 0 or 1, depending on the requirement.

  The results from the above-mentioned techniques will be presented in the next sections.

## 3.17 Cross-Validating Estimator Performance

Now that the final dataset (dataframe) has been created and the baseline for evaluating the performance of machine learning models has been established, the dataset is ready to be evaluated for performance on the selected machine learning models. One of the most established methods to cross check the performance of a model is **Cross Validation** and can be inferred from figure 3.11.

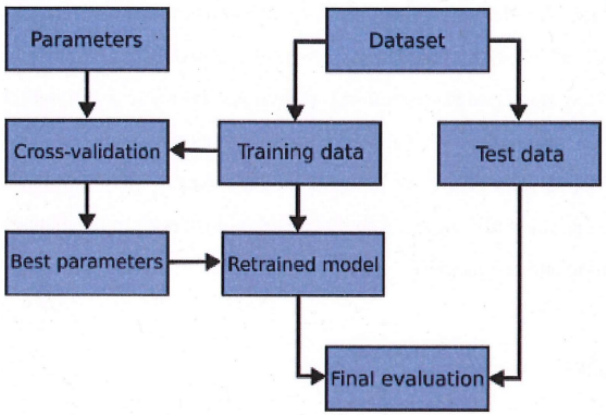


Figure 3.11: Cross Validation for validating performance of machine learning models adopted from [39].

The following conclusions can be inferred from the figure 3.11:

- The final dataset must be first segregated into two completely independent sets.
- The first set (on the left) is also known as **training and validation set**. The data on this set is subjected to various steps like: training this set using cross validation techniques, finding out the best parameters (hyperparameters) and then finally creating the retrained model.
- The other set is known as the **test** set. This is the actual set of data on which these retrained models must be evaluated on.
- The above-mentioned steps ensure that the trained models are tested on a completely new and unknown dataset. Hence, the cross validation technique provides an **unbiased view** of a model's performance.

To adhere to the above-mentioned validation process [39], the dataset being used in this study was segregated by a ratio of **80 to 20**, this implies that 80% of the dataset was separated for training and validation. And the remaining 20% was used for testing purposes.

The **final and the most important factor** that must be considered for this study is the **avoidance of data leakage**. As the dataset created for this study contains continuous data for the patients (due to daily aggregation), it is possible that the data from a patient can leak from the train set to the test set. Therefore, to ensure that such data leakage has been avoided, **GroupShuffleSplit** technique has been implemented. This technique ensures that the train and test splits have been created by ensuring no common SUBJECTID(Patient ID) will exist in both the sets [40].

As this dataset also contains information about patients who have been readmitted, Patient ID will be an incompetent parameter to avoid such a leakage. Therefore, GROUPID that is a combination of the SUBJECTID and HADMID has been introduced in this dataset to be used as a splitter between the train and the test set.

# Chapter 4

# Results

## 4.1 Results

Cross-validation is a technique in which n (n=4 in the ongoing study) equal parts (sets) of the dataset are created. A machine learning model is trained on the n-1 sets combined and then tested on the remaining unseen set. This process (iterations/folds) is repeated till all the n sets have independently been part of the test set and concludes by finding the average performance across all the iterations [41].

Figures: 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 and 4.7 represent the AUC plots for the selected models based on **4-Fold cross-validation technique.** The AUC scores for the selected models can also be referred from the table 4.1. This table has been arranged in decreasing order of the AUC scores obtained for the models.

| **Model Name** | **AUC score (%)** |
|---|---|
| Logistic Regression | 71 |
| Ridge Classifier | 71 |
| Random Forest | 70 |
| XGBoost | 68 |
| SVM | 67 |
| KNN | 56 |
| DecisionTreeClassifier | 52 |

Table 4.1: AUC scores.

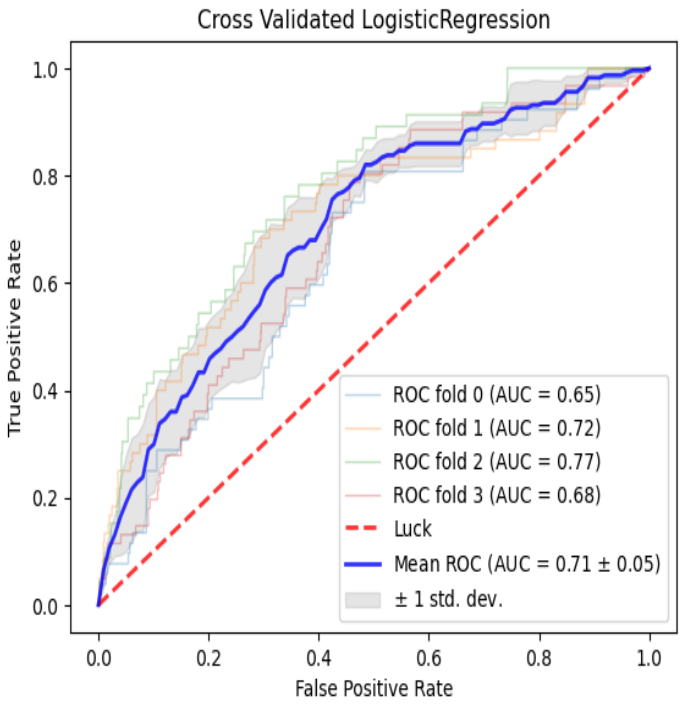


Figure 4.1: 4-Fold Cross Validated AUC plot for Logistic Regression model.

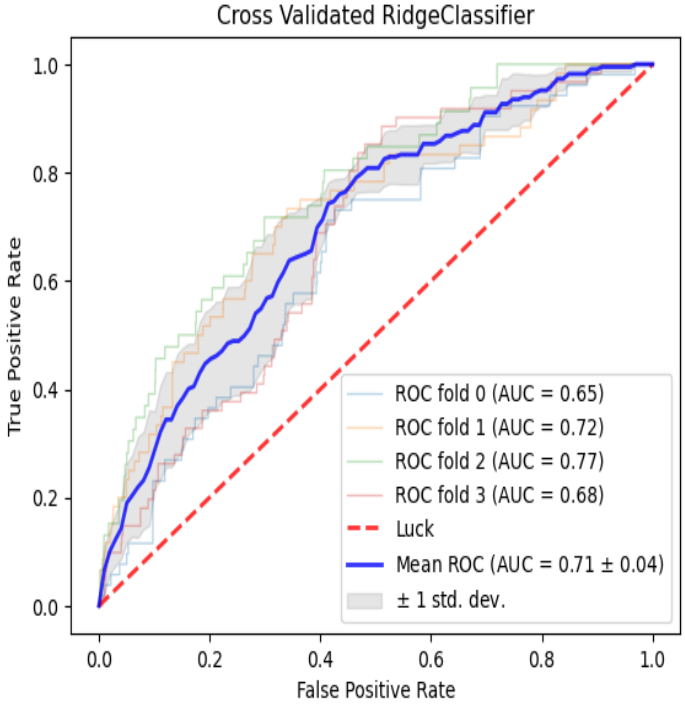


Figure 4.2: 4-Fold Cross Validated AUC plot for Ridge Classifier model.

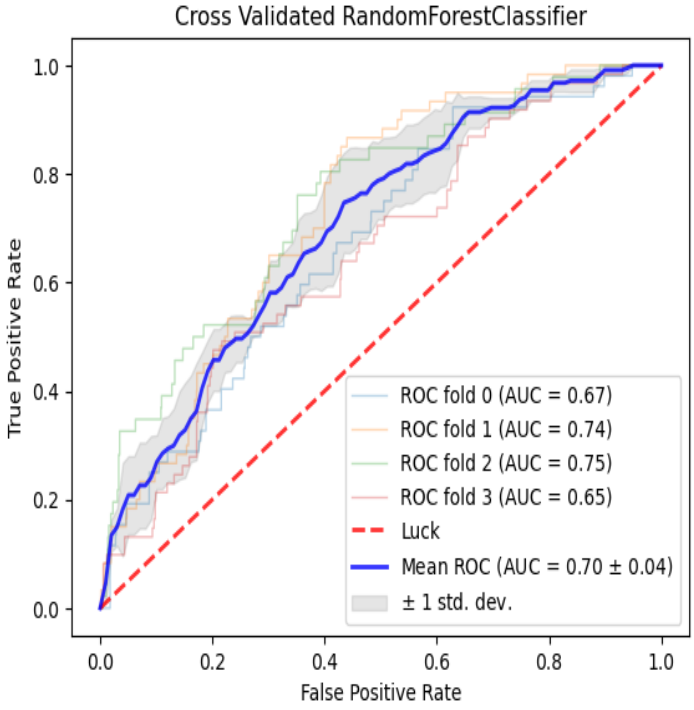


Figure 4.3: 4-Fold Cross Validated AUC plot for Random Forest model.

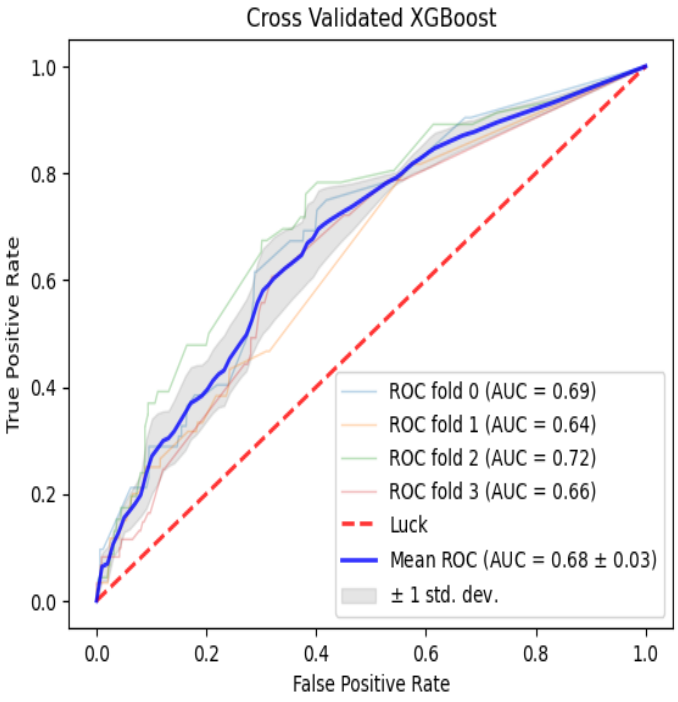


Figure 4.4: 4-Fold Cross Validated AUC plot for XGBoost model.

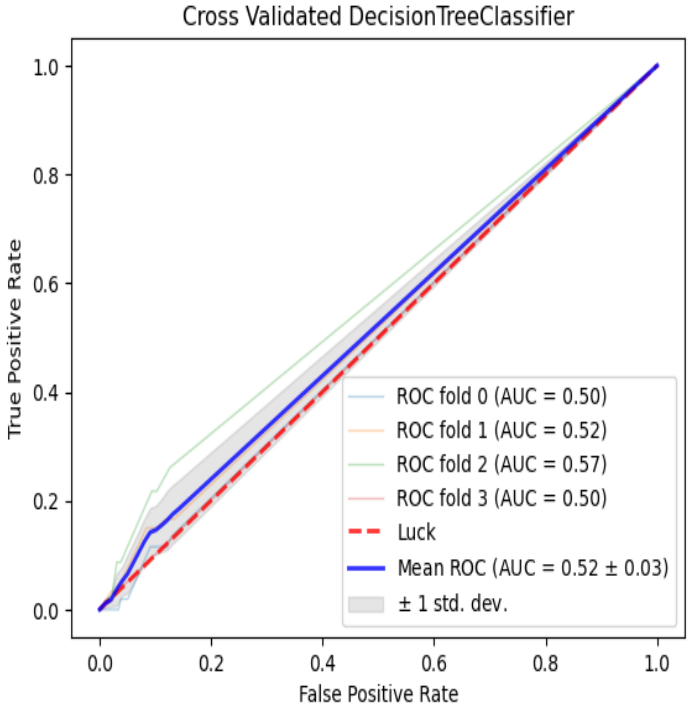


Figure 4.5: 4-Fold Cross Validated AUC plot for DecisionTreeClassifier model.

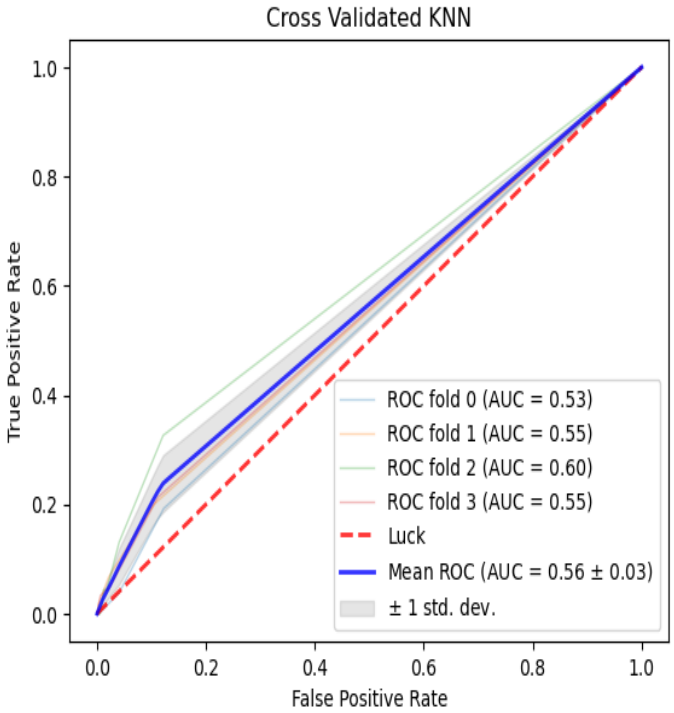


Figure 4.6: 4-Fold Cross Validated AUC plot for KNN model.

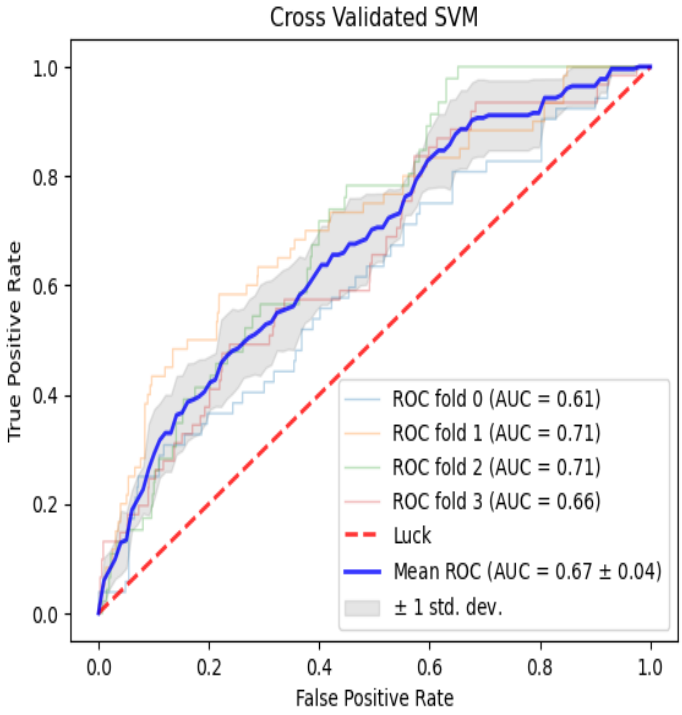


Figure 4.7: 4-Fold Cross Validated AUC plot for SVM model.

| **Model Name** | **Threshold Set** | **Accuracy** | **Precision** | **Recall (Sensitivity)** | **F1** |
|---|---|---|---|---|---|
| Logistic Regression | 0.009 | 0.59 | 0.02 | 0.64 | 0.04 |
| Ridge Classifier | 0.010 | 0.60 | 0.02 | 0.63 | 0.04 |
| Random Forest | 0.022 | 0.82 | 0.03 | 0.45 | 0.06 |
| KNN | 0.071 | 0.96 | 0.03 | 0.07 | 0.04 |
| SVM | 0.009 | 0.84 | 0.02 | 0.30 | 0.05 |
| XGBoost | 0.007 | 0.68 | 0.02 | 0.56 | 0.04 |
| DecisionTreeClassifier | 1.000 | 0.98 | 0.00 | 0.00 | 0.00 |

Table 4.2: Machine learning metrics using J statistics.

The next section will discuss the results obtained from the selected machine learning models.

## 4.2 Discussion

Interpreting area under the receiver operating characteristic curve [42] states that most models that scored more than 70% of AUC score have been deemed as performing in the range of being acceptable to moderately good in diagnostic tests and can be referred from figure 4.8.

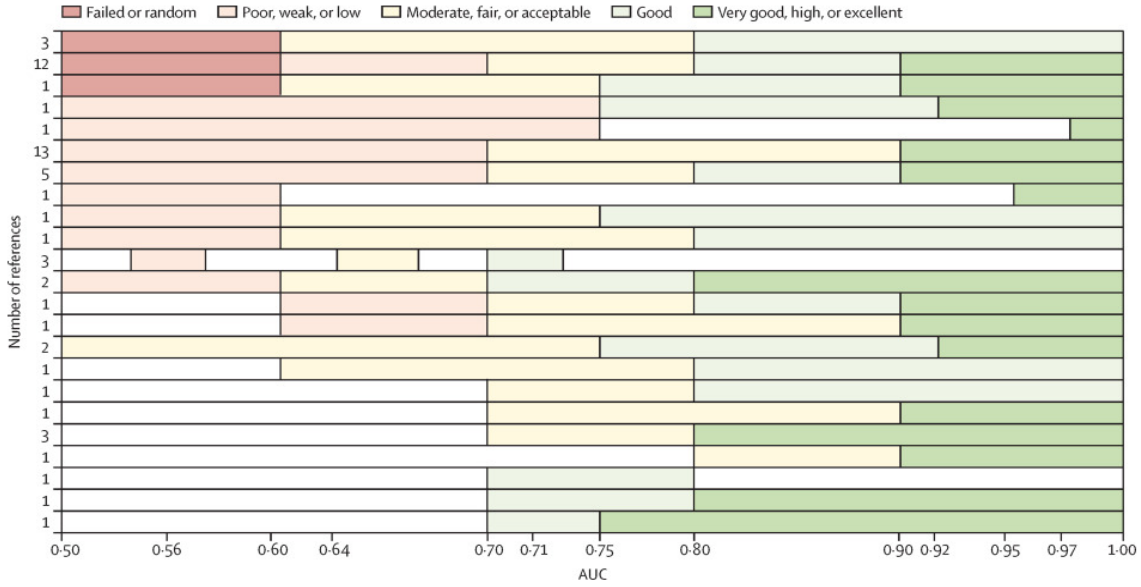


Figure 4.8: Interpreting Area Under Curve adopted from [42].

From the plots and AUC scores obtained for the selected models, it can be concluded that **Logistic Regression, Ridge Classifier and Random Forest models** provided admissible results i.e., above 70% AUC score. Out of these three models, both Logistic Regression and Ridge Classifier can be declared as the winner models.

Table 4.2 represents the scores obtained for machine learning models by using J statistics. One important conclusion that can be drawn from the table 4.2 is that even though Random Forest model had higher accuracy than Logistic Regression and Ridge Classifier models, Logistic Regression and Ridge Classifier had better value for sensitivity i.e., the ability to identify the **True Positive cases**. It is important to have a classifier setting that can identify a positive case which in this scenario turned out to be nearly 66%. This implies that **2 out of 3 cases have been correctly identified by these classifiers** using J statistics method.

Moreover, the threshold set for obtaining the machine learning metrics can be manually changed depending on the requirement (setting where a machine learning model has to be used).

Figures 4.9 and 4.10 represent an example where the threshold has been set to **0.008** and the **Sensitivity/Recall value was evaluated to be 0.67%** for logistic regression model (Note: the position for the output labels i.e., 0 and 1 in the confusion matrices in figures: 4.9 and 4.10 are different to the generic confusion matrix described in the previous sections and the results have been adjusted accordingly).

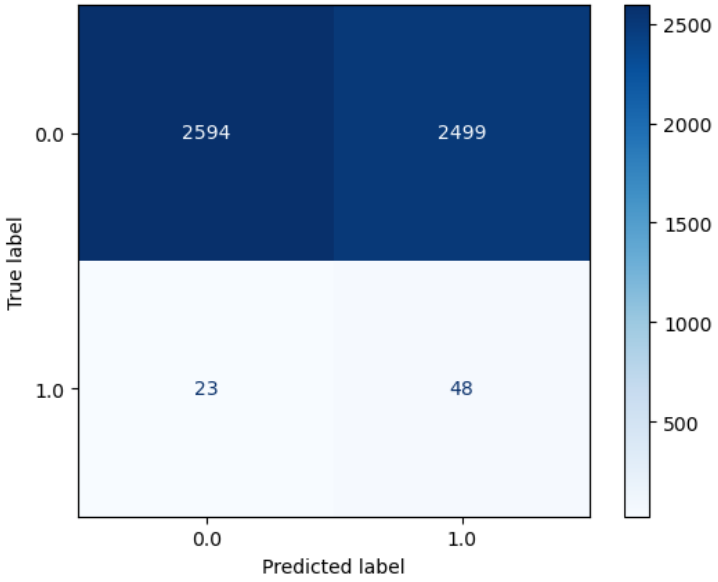


Figure 4.9: Logistic Regression Confusion Matrix for manually set threshold of 0.008.

| Model Name | Threshold Set | Accuracy | Precision | Recall (Sensitivity) | F1 |
|---|---|---|---|---|---|
| Logistic Regression | 0.008 | 0.51 | 0.01 | 0.67 | 0.03 |

Table 4.3: Machine learning metrics for Logistic Regression model for threshold value of 0.008.

**Explaining model performances:** Now that the performance of different machine learning models and the criteria for choosing the best models based on the AUC scores has been discussed, it is necessary to discuss that even though models such as KNN, SVM, XGBoost and DecisionTreeClassifier provided better accuracy, they have been classified as incompetent models in the ongoing study. Accuracy which is the ratio of total number of correct predictions to the total number of predictions made by a classifier, shows how good a model has been able to make the right predictions. However, as the ongoing study is using a highly imbalanced dataset where the total number of positive cases (output label=1) is 1.2%, it is necessary to check how

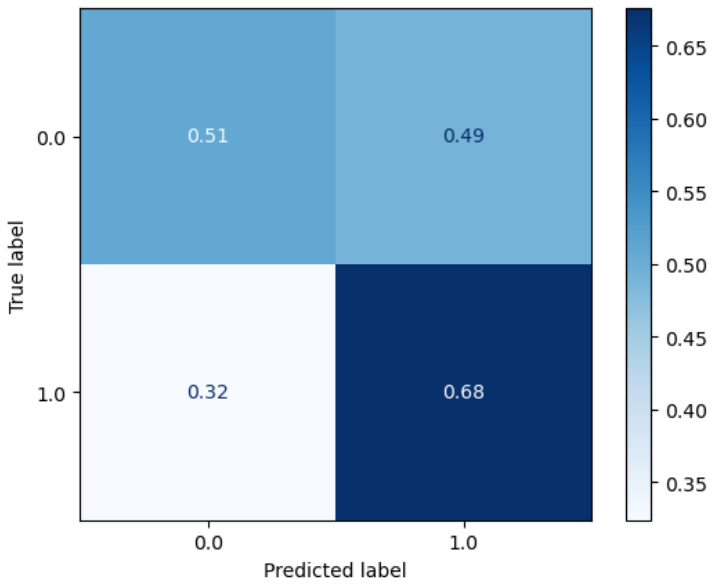


Figure 4.10: Normalized Logistic Regression Confusion Matrix for manually set threshold of 0.008.

well a classifier can identify the positive output labels (1 in the ongoing study). Therefore, even though KNN, SVM, XGBoost and DecisionTreeClassifier models had higher accuracy than Logistic Regression, Ridge Classifier and Random Forest models, the former four models displayed very bad performances in terms of **Recall (Sensitivity)** i.e., the ability to identify the **True Positive Cases** [43]. The focus of this study is to identify the maximum possible number of true positive cases due to the fact that the misclassification of an unhealthy person as a healthy person can lead to mortality.

Due to the threshold (J statistics) set for identifying the predictions made by a model as positive or negative, predictions made by models such as Logistic Regression, Ridge Classifier and Random Forest towards the positive outcome have performed poorly in terms of accuracy as the total number of correct predictions made by these models turned out to be low, but remained high in case of KNN, SVM, XGBoost and DecisionTreeClassifier models. However, good recall values for Logistic Regression, Ridge Classifier and Random Forest make these models more viable for the ongoing study as the number of positive cases correctly identified by these models will allow the identification of maximum number of positive cases.

Therefore, the number of cases (patients) that have been correctly identified as positive (diagnosed with a disease) in case of using Logistic Regression, Ridge Classifier and Random

Forest models will be much higher than using KNN, SVM, XGBoost and DecisionTreeClassifier models.

**For a better understanding,** another scenario can be imagined where 95% output in a dataset is negative (0). A bad classifier can achieve a very good accuracy of more than 90% by correctly predicting the negative output most of the times. However, this classifier can perform very poorly in identifying the positive cases (represented by recall value).

**Note:** Confusion matrices as well as metrics associated with Random Forest and Ridge Classifier models using manual tuning have been added in Appendix A.

Now that it has been proven that the features used in the machine learning models for heart failure patients for this study turned out to be true to predict the output label for such patients based on their AUC scores, it is necessary to check which features had the most and least significant contribution in predicting the output label. To equate these factors, SHAP (game theoretic) approach will be used to conclude the importance of a feature on a machine learning model [44].

LIME explanations for finding importance for a particular model was also considered for this study, however LIME suffers with inconsistent explanation due to its unstable architecture. Hence, SHAP that supports both consistent and robust architecture along with good representation capabilities will be used for explaining the results of machine learning models [45].

SHAP can be defined as an XAI approach that utilizes shapley values from game theory to provide interpretable as well as easily explainable reasons for the factors that led to the most relevant and probable reasons for a model's prediction. Thus, the static and non-static features associated with a patient will be the output for the SHAP values. This can thus help in understanding the importance of the features and the contribution of these features towards the predicted outcome.

SHAP also has the advantage of being able to be used as a global and local (instance level) explanation tool for a model. The latter advantage will be used to provide local explanation for an outcome of a patient at a particular day and hence can be used to address the risk factor for a patient and help in decision making. Figures 4.11, 4.12 and 4.13 demonstrate the plots obtained for the top three best performing models with respect to the outcome (label 0 and 1 imply the output label used in this study). **These graphs also demonstrate the final features used for obtaining the results mentioned in the previous section.**

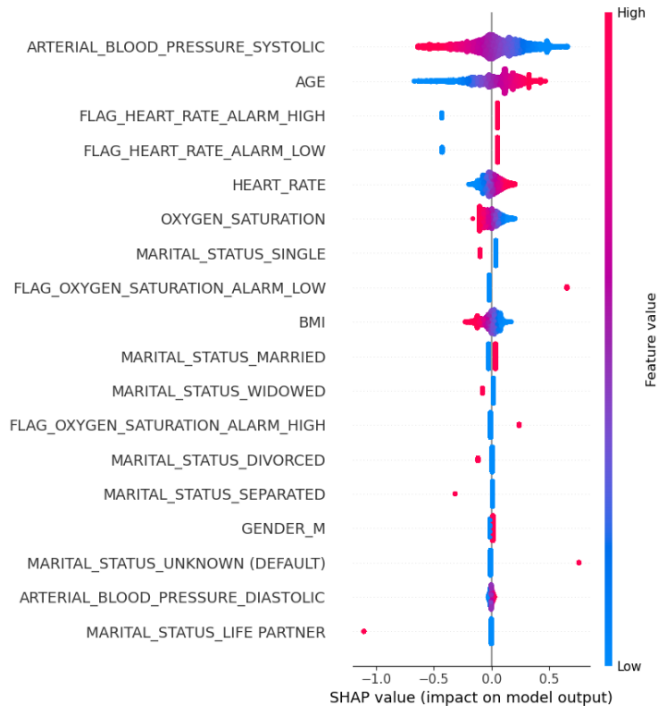


Figure 4.11: SHAP plot displaying feature importance for Logistic Regression model.

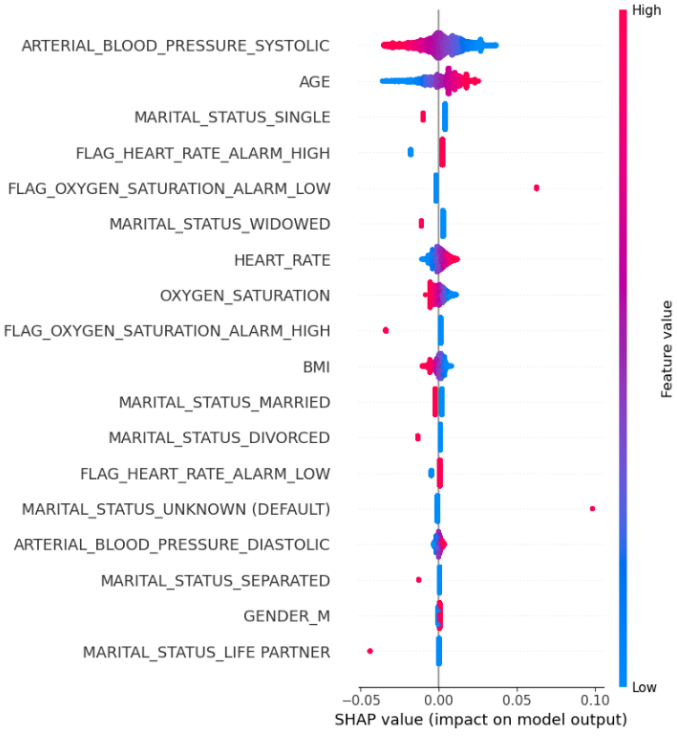


Figure 4.12: SHAP plot displaying feature importance for Ridge Classifier model.

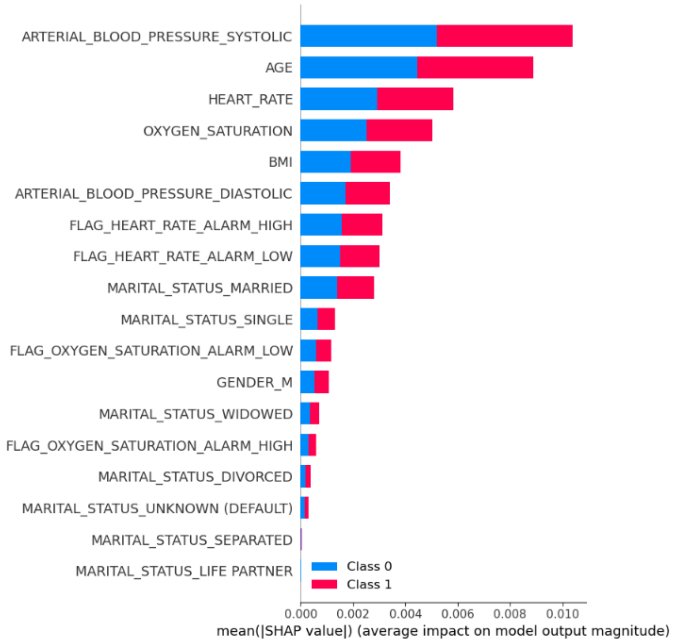


Figure 4.13: SHAP plot displaying feature importance for Random Forest model.

The features displayed in the SHAP plots are arranged in decreasing order of their importance, thus implying that the most important feature will appear at the top and the least important feature will appear at the bottom. A positive SHAP value (displayed in red) indicates a positive contribution towards predicting the actual output. Whereas a negative SHAP value (displayed in blue) indicates a negative contribution towards predicting the actual output.

As can be inferred from figures 4.11, 4.12 and 4.13, features such as **Arterial Blood Pressure, Heart Rate, Oxygen Saturation and Age** continue to appear in the **top-5 features** in contributing towards an output for a patient. This implies that features related to heart and age (which heavily contributes towards heart problems) continue to contribute as major factors for heart failure.

One important inference that can be inferred here is that **BMI**, that has been included as a dynamically changing feature in this study (based on change in daily weight) also appeared in the list of **top-5** contributing factors that led to mortality in the case of **Random Forest** model.

**Note:** SHAP supports different kinds of plots for linear and tree-based models. SHAP plots for the top three performing models have been discussed in this section, the SHAP plots for other models have been uploaded in Appendix A.

# Chapter 5

# Software Development

The previous sections in this report were used to define the research problem, the methodology used to reach a novelty approach to predict the mortality for heart failure patients and the evaluation of machine learning models to lay the foundation for using AI based approach to find a possible solution to the research problem. Now that it has been proven that the machine learning models used in this study have produced admissible results, the focus will shift towards the development of a patient monitoring software that can be used by hospital staffs or caretakers of a patient.

As this software must be viable enough to be used in a clinical setting, it must be designed in such a way that the results produced by this software are not complex and can be interpreted easily. Therefore, Tkinter [46] module in python programming language will be used to create the GUI for this software. This will enable homogeneity between the programming language to be used for the back-end and front-end development.

As the primary motive for using this software is to be used in a real-life scenario, it is necessary that it supports continuous flow of data. To fulfil this requirement, the software has been designed to fetch the data from a relational database (e.g., PostgreSQL, DataBase2, Sybase, MYSQL etc.). The advantage of using a relational database also lies in the fact that a central server can be used in a hospital (or a clinical setting). This server can feed the data from the patients into this software system by using a **LAN, MAN or WAN** network.

PostgreSQL database has been used to develop this software. PostgreSQL has been chosen to develop this software as it provides an interactive GUI to easily feed the data into the database and can be easily connected with python language by using a kerberos (authentication) layer to

fetch the data from the database. PostgreSQL is also freely available, thus allowing a new user to download this software on a local machine and test the results on their own.

## 5.1 Software Design

To keep the design simple and viable, the following approach was followed and can be inferred from figure 5.1:

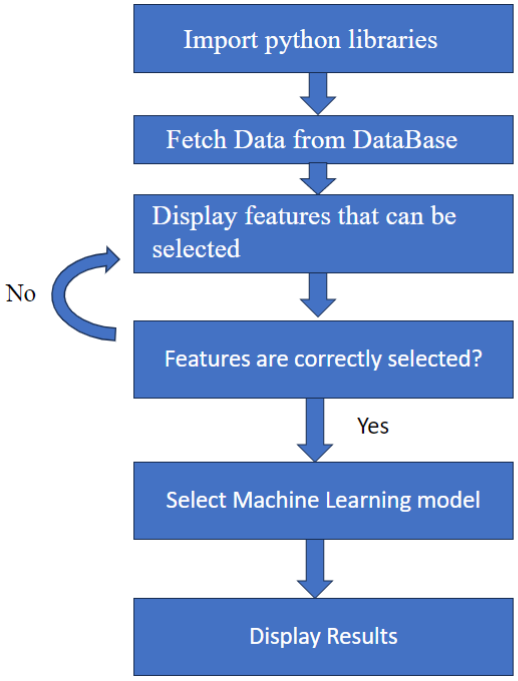


Figure 5.1: User workflow for the software.

- Import python libraries.
- Load the data to be used in the software from the database (PostgreSQL).
- No hard coding was followed to allow different types of back-end data (features) to be incorporated into the software.
- The above-mentioned approach was followed to handle different kinds of scenarios like unavailability of a type of data (vital sign) or in case of handling additional data that can be provided by the hospital (user).

- The above-mentioned approach also allows the user to compare results based on different input features that have been provided to the software.

  This can also prove to be useful in a scenario where the user (e.g. a trained person) already knows which features (static or dynamic signs) can provide a better insight into a patient's condition.

- Allow the user to choose a machine learning model of their choice. This was done to allow the user to compare the results from different machine learning models.

By following the approach that can be inferred from figure 5.1, the launch screen of the software has been designed that can be referred from figure 5.2. As can be inferred from figure 5.2, the launch screen provides the user with the choice of choosing different signs (both static and non-static) for the patients.

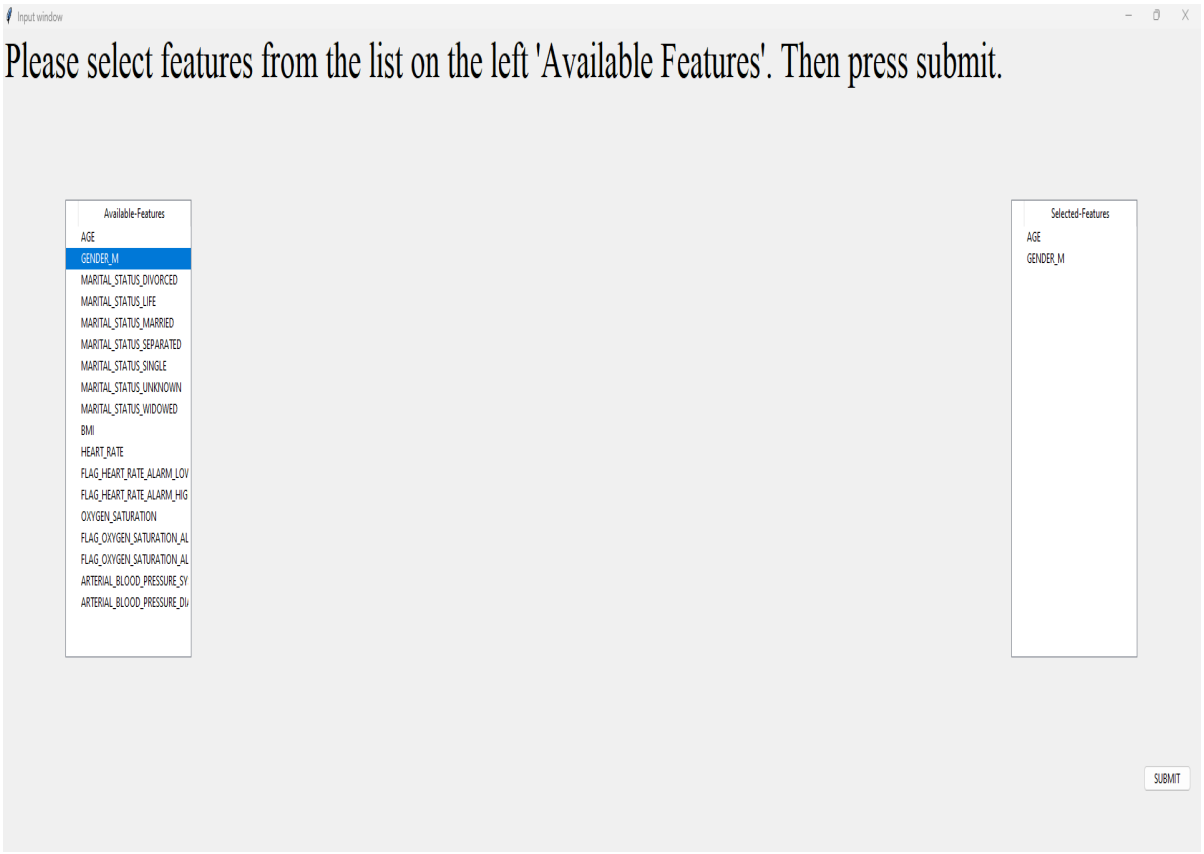


Figure 5.2: Software launch screen.

The launch screen of the software displays two lists: **Available-Features** and **Selected-Features**. The software has been designed in such a way that when a user clicks at an available

feature from the available-features list, it will automatically be displayed in the selected features list on the right, so that the user has a clear view of the selected features. To handle a scenario where the user had accidentally clicked at a feature and would like to remove it now, the user can remove the accidentally selected feature by clicking on it again from the available-features list. Thus, once a user clicks on a feature more than once on the available-features list, that feature will be removed from the selected features list.

The software has also been designed to handle a scenario where the selected-features list has been accidentally left blank by the user by opening an error message window that displays the message: **input features are empty**.

Once the user has selected the required features, the user can proceed further by clicking the **SUBMIT** button as can be inferred from figure 5.2.

The submit button will only work when the features have been properly selected and thus results in a new screen (main screen) that can be inferred from figure 5.3.

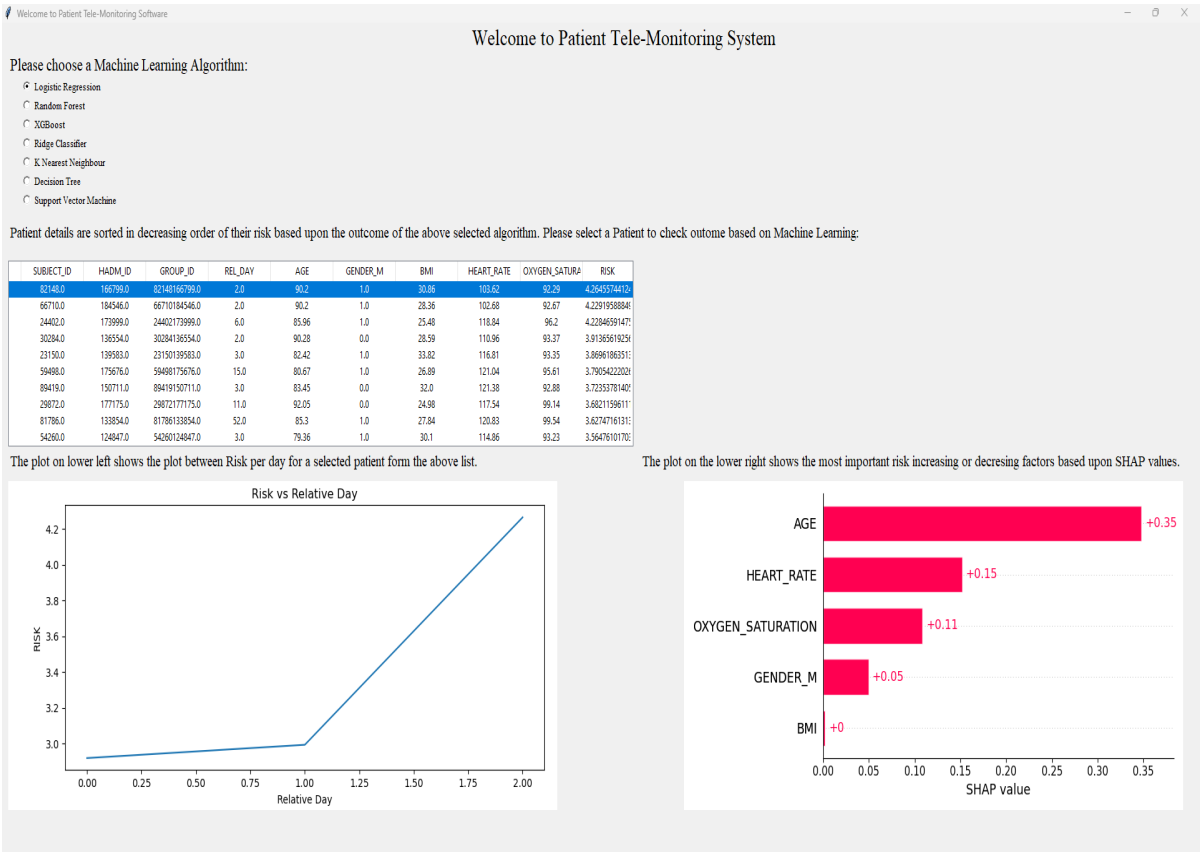


Figure 5.3: Software main screen.

As can be inferred from figure 5.3, the user now reaches the main screen of the software. The

user can now track information about the patients based on the signs selected from the launch screen of the software (in the example shown in figure 5.3: AGE, HEART RATE, OXYGEN SATURATION, GENDER and BMI). The table showing the details of a patient will be blank when the user first reaches the main window. This table will be populated with the information about the patients after the user has clicked on a machine learning model on the main screen. As can be inferred from figure 5.3, the user is provided with a choice of 7 machine learning models: **Logistic Regression, Ridge Classifier, KNN, Random Forest, SVM, XGBoost and Decision Tree (DecisionTreeClassifier)**. Now the user is prompted to choose one of these 7 options to proceed further.

The choice of selecting a machine learning model allows the user to check the result of risk (mortality) for a patient based on the selected machine learning model. Once when the user makes a choice of a machine learning model from the radio buttons, a new the table will appear on the main screen of the software.

This table will be populated with values like: SUBJECT_ID, HADM_ID, GROUP_ID, the vital signs selected by the user and the risk of mortality (denoted by last column in figure 5.3) associated with a patient (the names of all these columns have been dynamically created).

To facilitate risk based interpretation (i.e., patients who have higher risk of mortality), the patient table that can be referred from the figure 5.3, has been sorted in decreasing order of their mortality i.e., **patients who have a higher risk of mortality will appear at the top**, this can enable the caretakers/staff members to easily navigate to patients that require more attention as per their risk assessment.

The **RISK** column has been calculated by **predicting the probability of a selected machine learning model towards the output of 1** i.e., by calculating the exacerbation (mortality) probability of a patient. After following this process, the user is now presented with a table that contains the values of the patients along with the risk associated with each patient. This table has been made in a clickable format to provide an insight analysis.

By referring to figure 5.3, the graph on the lower left displays the risk associated with a patient over the time (days) the patient was admitted in the ICU. Also, by referring to figure 5.3, the graph on the lower right displays the impact of the selected features for the selected patient on the outcome (risk) based on the SHAP values calculated from the selected features. **A positive SHAP value indicates positive contribution towards the output and a negative**

**SHAP value indicates negative contribution towards the output.** The SHAP values have been arranged in the decreasing order of their contribution towards the output.

After clicking on a patient row, the user is presented with a new screen where the user can track the information for the selected patient throughout the admission cycle. This screen has been created to provide an overall insight into the condition of a patient. All the steps mentioned in this section on how to operate the software have also been mentioned in the form of **text tutorial** on all the screens of the software.

## 5.2 Interpreting Software Results

The previous section was dedicated to discuss the design and work flow of the software as well as the results displayed by the software. The current section will be dedicated to discuss the analysis of the results obtained from the software. From the figure 5.3, an inference can be made for the patient having **SUBJECT_ID: 82148**. As can be inferred from figure 5.3, this patient has a **4.26% chance of facing mortality**. This patient has the highest risk of facing mortality among all the patients. By inferring the risk graph (the graph on lower left from figure 5.3), an inference can be made that the risk has been increasing at a sharp rate for this patient. From the graph on the lower right in figure 5.3, an inference can be made about the impact of the features that have been selected for this patient. Features such as oxygen saturation and heart rate have made positive contribution towards this outcome and hence should be checked by the associated person for any possible remedies. Following this, static signs such as AGE and GENDER also contributed positively towards this output.

**Note:** As this software supports the flexibility of choosing the features, it is also possible to incorporate features for other diseases like kidney failure etc. However, to get the best results it is necessary to change the hyperparameters accordingly.

# Chapter 6

# Conclusion and Future Outlook

## 6.1 Conclusion

The thesis **Machine Learning Based Prediction of Critical Events of Intensive Care Heart Failure Patients** had a fixed set of objectives to be completed. These objectives had to be completed within real-life scenario implementation boundaries.

**Firstly,** an ETL process was defined to extract the demographic and vital signs information for patients from the MIMIC-III dataset. **4,257** patients were identified as heart failure patients from a total of 60,000 available patients. After completing the data cleaning process, **1,706** patients were finally separated for completing further tasks. The data on the finally separated patients was then subjected to further analysis and data enrichment process for testing on an output label to predict mortality and exacerbation for a patient.

**Secondly,** this data was tested on various machine learning models. **Logistic Regression, Ridge Classifier, KNN, Random Forest, DecisionTreeClassifier, SVM and XGBoost** models were used to make predictions on the output label. This testing was performed after tuning the hyperparameters for all the models and then cross validating the results. Out of all the tested models, **Logistic Regression, Ridge Classifier and Random Forest** models provided admissible performance by providing an **AUC score of more than 70%**. Both Logistic Regression and Ridge Classifier proved to be the best models out of all models selected for this study.

**Thirdly,** a software was developed that can be used in clinical scenarios. The primary motive for developing this software is to enhance the **decision-making process** by using XAI methods to explain the predictions made on the output by the machine learning models.

## 6.2 Future Outlook

The objective of improving the results of the current research problem can be further discussed. **For example: the ETL process followed to extract information for heart failure patients can be further refined and enriched by using more high-quality data, feature engineering etc. Moreover, further enhancements include the implementation of more machine learning algorithms, refining the hyperparameter tuning process as well as improving the software engineering part.**

What is often considered as an immediate solution in the field of machine learning is the of use more complex machine learning models rather than simpler ones to improve the quality of the predicted outcome. However, such an approach can often lead to scenarios where it takes hours to train the machine learning models. This approach also carries with it high chances of failing again and again based on the combinations of different models being deployed.

Instead, what can be considered as a better approach is to incorporate more relevant data for the problem that needs to be solved. The article **The Unsolvable Effectiveness of Data** [47] claims that refined data plays more significant role than the complexity of a model in terms of improving the quality of the outcome predicted by a machine learning model.

This approach can also prove to be advantageous, as the time taken for making a clinical decision is of the utmost importance and applying less complex models with lower fitting time is much more favourable than designing a highly complex model that can take hours to train and then produce results. Although the results from a more complex model are more than likely to be better than from a less complex model, it is the **fitting time** that can cause the hindrance to be deployed in clinical scenarios.

The software can be enhanced to display the results into an embedded system (e.g., smart watch). These results can be handily used by the staff members to assess the condition of a patient at their convenience rather than monitoring an individual software system. Such embedded systems can be developed by integrating them to an **IOT** based architecture [48]. A patient's information and the predictions made from that information can be made readily available to an embedded system from the software to enhance the decision-making process.

Further enhancement in the software system includes handling **data drift** [49]. Data drift is a scenario where the sudden changes in the quality of data being provided to the software causes

degradation in the quality of predicted outcome. The data drift can be avoided by incorporating data cleaning and data pre-processing techniques in the software before the software starts the machine learning process.

The most important factor that must be considered while evaluating the performance of a machine learning model is the distribution of the output labels. The current study is utilizing a very highly imbalanced dataset where only 20 patients out of 1,706 patients had an output label of 1. This implies that only 1.2% of the total samples tested can be considered as the positive class and the remaining to be negative class. To improve the performance of the machine learning models, **Random over and under sampling techniques** [50] can be used to handle the class imbalance.

# Bibliography


[1] Alistair EW Johnson et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific data* 3.1 (2016), pp. 1–9.

[2] Edwin De Jonge and Mark Van Der Loo. *An introduction to data cleaning with R*. Statistics Netherlands Heerlen, 2013.

[3] Fei Jiang et al. "Artificial intelligence in healthcare: past, present and future". In: *Stroke and vascular neurology* 2.4 (2017).

[4] Benjamin Shickel et al. "Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis". In: *IEEE journal of biomedical and health informatics* 22.5 (2017), pp. 1589–1604.

[5] Erin P Balogh, Bryan T Miller, and John R Ball. "Improving diagnosis in health care". In: (2015).

[6] Alistair Johnson, Tom Pollard, and Roger Mark. "The MIMIC-III Clinical Database 2016". In: *PhysioNet. URL: https://physionet. org/content/mimiciii/1.4/[accessed 2016-09-04]* ().

[7] Jose Leal, Ramon Luengo-Fernandez and Richéal Burns. *CVD Statistics*. 2017. URL: https://ehnheart.org/cvd-statistics.html.

[8] *Ponemon (accessed on 23/05/2023)*. URL: https://buildingbetterhealthcare.com/news/article_page/Comment_Health_networks_delivering_the_future_of_healthcare/94931.

[9] Fuhai Li et al. "Prediction model of in-hospital mortality in intensive care unit patients with heart failure: machine learning-based, retrospective analysis of the MIMIC-III database". In: *BMJ open* 11.7 (2021), e044779.

[10] Reza Sadeghi, Tanvi Banerjee, and William Romine. “Early hospital mortality prediction using vital signals”. In: *Smart Health* 9 (2018), pp. 265–274.

[11] Rongting Zhang et al. “Independent effects of the triglyceride-glucose index on all-cause mortality in critically ill patients with coronary heart disease: analysis of the MIMIC-III database”. In: *Cardiovascular Diabetology* 22.1 (2023), pp. 1–12.

[12] Jordi Albiol Mosegui. “Predicting Intensive Care Unit Length of Stay via Supervised Learning”. In: (2018).

[13] Tamraparni Dasu and Theodore Johnson. *Exploratory data mining and data cleaning*. John Wiley & Sons, 2003.

[14] Olegas Niaksu. “CRISP data mining methodology extension for medical domain”. In: *Baltic Journal of Modern Computing* 3.2 (2015), p. 92.

[15] *Pandas (accessed on: 21/05/2023)*. URL: `https://pandas.pydata.org/`.

[16] *ICD description (accessed on: 21/05/2023)*. URL: `https://en.wikipedia.org/wiki/International_Classification_of_Diseases`.

[17] *ICD-9 Code Description (accessed on: 21/05/2023)*. URL: `https://en.wikipedia.org/wiki/List_of_ICD-9_codes`.

[18] Nandhini Subramanyan and Ranjani Subramanyan. *Patient data representation for outcome prediction of congestive heart failure patients*. 2019.

[19] *Table1 (accessed on: 21/05/2023)*. URL: `https://pypi.org/project/tableone/`.

[20] Lisa Eisenberg et al. “Time-dependent prediction of mortality and cytomegalovirus reactivation after allogeneic hematopoietic cell transplantation using machine learning”. In: *American Journal of Hematology* 97.10 (2022), pp. 1309–1323.

[21] Małgorzata Misztal. “Imputation of missing data using R package”. In: *Acta Universitatis Lodziensis. Folia Oeconomica* 269 (2012).

[22] *Generic Confusion Matrix (accessed on 24/05/2023)*. URL: `https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62`.

[23] Karimollah Hajian-Tilaki. “Receiver operating characteristic (ROC) curve analysis for medical diagnostic test evaluation”. In: *Caspian journal of internal medicine* 4.2 (2013), p. 627.

[24] *Generic AUROC plot (accessed on 23/05/2023)*. URL: `https://www.geeksforgeeks.org/auc-roc-curve/`.

[25] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[26] Michael P LaValley. “Logistic regression”. In: *Circulation* 117.18 (2008), pp. 2395–2399.

[27] Chong Peng and Qiang Cheng. “Discriminative ridge machine: A classifier for high-dimensional data or imbalanced data”. In: *IEEE transactions on neural networks and learning systems* 32.6 (2020), pp. 2595–2609.

[28] Gongde Guo et al. “KNN model-based approach in classification”. In: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings*. Springer. 2003, pp. 986–996.

[29] Muhammad Khan et al. “Drug side-effect prediction using machine learning methods”. In: (2017).

[30] Carl Kingsford and Steven L Salzberg. “What are decision trees?” In: *Nature biotechnology* 26.9 (2008), pp. 1011–1013.

[31] Gérard Biau and Erwan Scornet. “A random forest guided tour”. In: *Test* 25 (2016), pp. 197–227.

[32] Haifeng Wang and Dejin Hu. “Comparison of SVM and LS-SVM for regression”. In: *2005 International conference on neural networks and brain*. Vol. 1. IEEE. 2005, pp. 279–283.

[33] Tianqi Chen et al. “Xgboost: extreme gradient boosting”. In: *R package version 0.4-2* 1.4 (2015), pp. 1–4.

[34] Ziyi Li et al. “Distributed learning from multiple EHR databases: contextual embedding models for medical events”. In: *Journal of biomedical informatics* 92 (2019), p. 103138.

[35] Enrique F Schisterman et al. “Optimal cut-point and its corresponding Youden Index to discriminate individuals using pooled blood samples”. In: *Epidemiology* (2005), pp. 73–81.

[36] *Youden’s J statistic (accessed on: 21/05/2023).* URL: https://en.wikipedia.org/wiki/Youden%27s_J_statistic#:~:text=Youden's%20J%20statistic%20(also%20called,probability%20of%20an%20informed%20decision.

[37] Samuel Ricardo Comar, Mariester Malvezzi, and Ricardo Pasquini. “Evaluation of criteria of manual blood smear review following automated complete blood counts in a large university hospital”. In: *Revista brasileira de hematologia e hemoterapia* 39 (2017), pp. 306–317.

[38] Chris D Skedgel. “Estimating societal preferences for the allocation of healthcare resources using stated preference methods”. PhD thesis. University of Sheffield, 2013.

[39] *Cross Validation (accessed on: 21/05/2023).* URL: https://scikit-learn.org/stable/modules/cross_validation.html.

[40] Asmir Vodencarevic et al. “Prediction of Recurrent Ischemic Stroke Using Registry Data and Machine Learning Methods: The Erlangen Stroke Registry”. In: *Stroke* 53.7 (2022), pp. 2299–2306.

[41] Ilias Tougui, Abdelilah Jilbab, and Jamal El Mhamdi. “Impact of the choice of cross-validation techniques on the results of machine learning-based diagnostic applications”. In: *Healthcare informatics research* 27.3 (2021), pp. 189–199.

[42] Anne AH de Hond, Ewout W Steyerberg, and Ben van Calster. “Interpreting area under the receiver operating characteristic curve”. In: *The Lancet Digital Health* 4.12 (2022), e853–e855.

[43] Brendan Juba and Hai S Le. “Precision-recall versus accuracy and the role of large data sets”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 4039–4048.

[44] Scott M Lundberg and Su-In Lee. “A unified approach to interpreting model predictions”. In: *Advances in neural information processing systems* 30 (2017).

[45] Giorgio Visani et al. “Statistical stability indices for LIME: Obtaining reliable explanations for machine learning models”. In: *Journal of the Operational Research Society* 73.1 (2022), pp. 91–101.

[46] *Tkinter GUI module in python (accessed on: 21/05/2023)*. URL: `https://docs.python.org/3/library/tkinter.html`.

[47] Alon Halevy, Peter Norvig, and Fernando Pereira. “The unreasonable effectiveness of data”. In: *IEEE intelligent systems* 24.2 (2009), pp. 8–12.

[48] Abu Shad Ahammed, Micheal Ezekiel, and Roman Obermaisser. “A Novel Analysis of Performance and Inference Time of Machine Learning Models to Detect Cardiovascular Emergency Situations of Rescue Patients”. In: *2022 International Conference on Artificial Intelligence of Things (ICAIoT)*. IEEE. 2022, pp. 1–6.

[49] T Ryan Hoens, Robi Polikar, and Nitesh V Chawla. “Learning from streaming data with concept drift and imbalance: an overview”. In: *Progress in Artificial Intelligence* 1 (2012), pp. 89–101.

[50] Chris Drummond, Robert C Holte, et al. “C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling”. In: *Workshop on learning from imbalanced datasets II*. Vol. 11. 2003, pp. 1–8.

# A

# Appendix A

The following plots show the distribution of various parameters for daily weight and height:

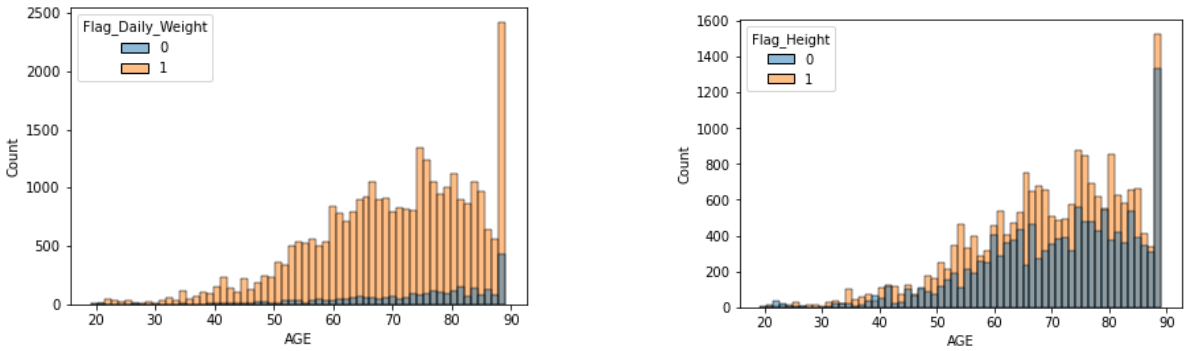


Figure A.1: Distribution of age with daily weight and height.

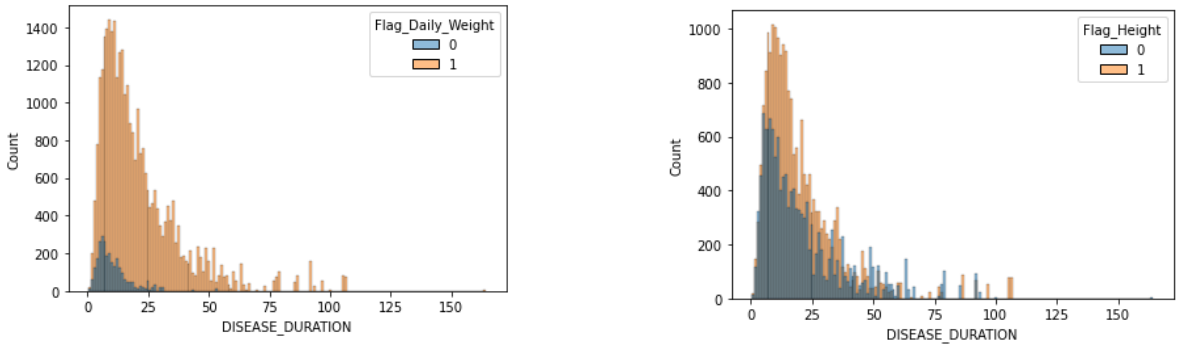


Figure A.2: Distribution of disease duration with daily weight and height.

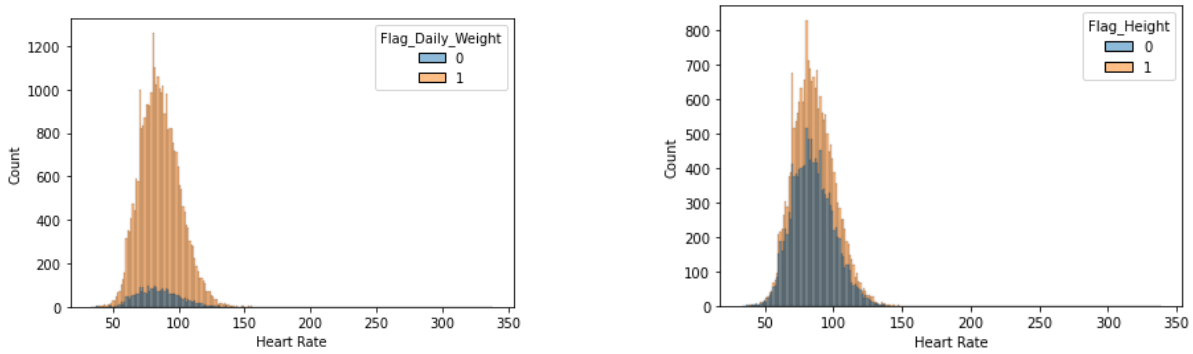


Figure A.3: Distribution of heart rate with daily weight and height.

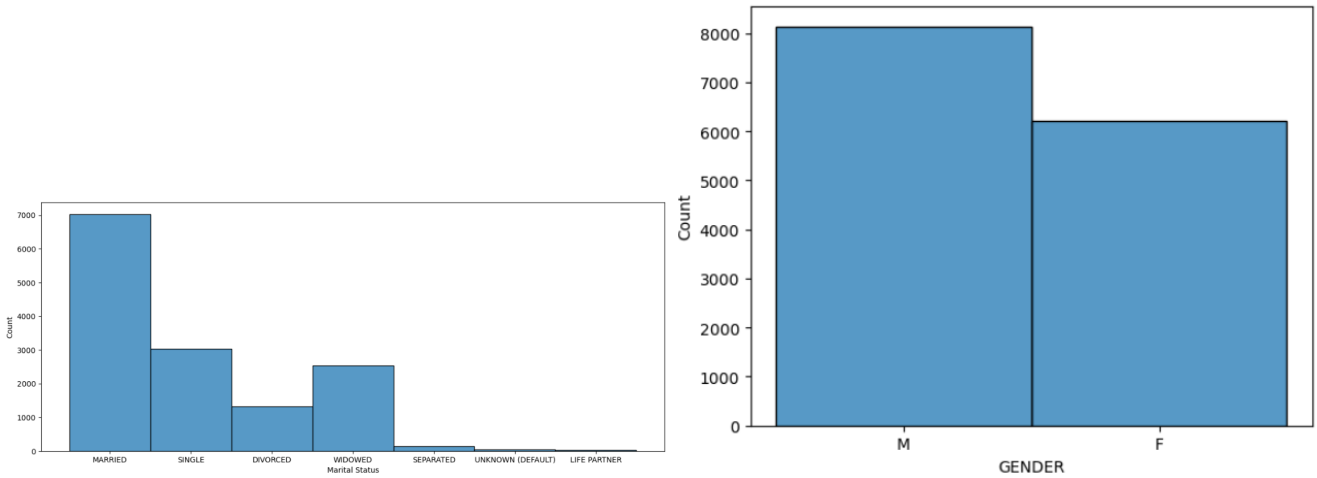


Figure A.4: Distribution of marital status and gender.

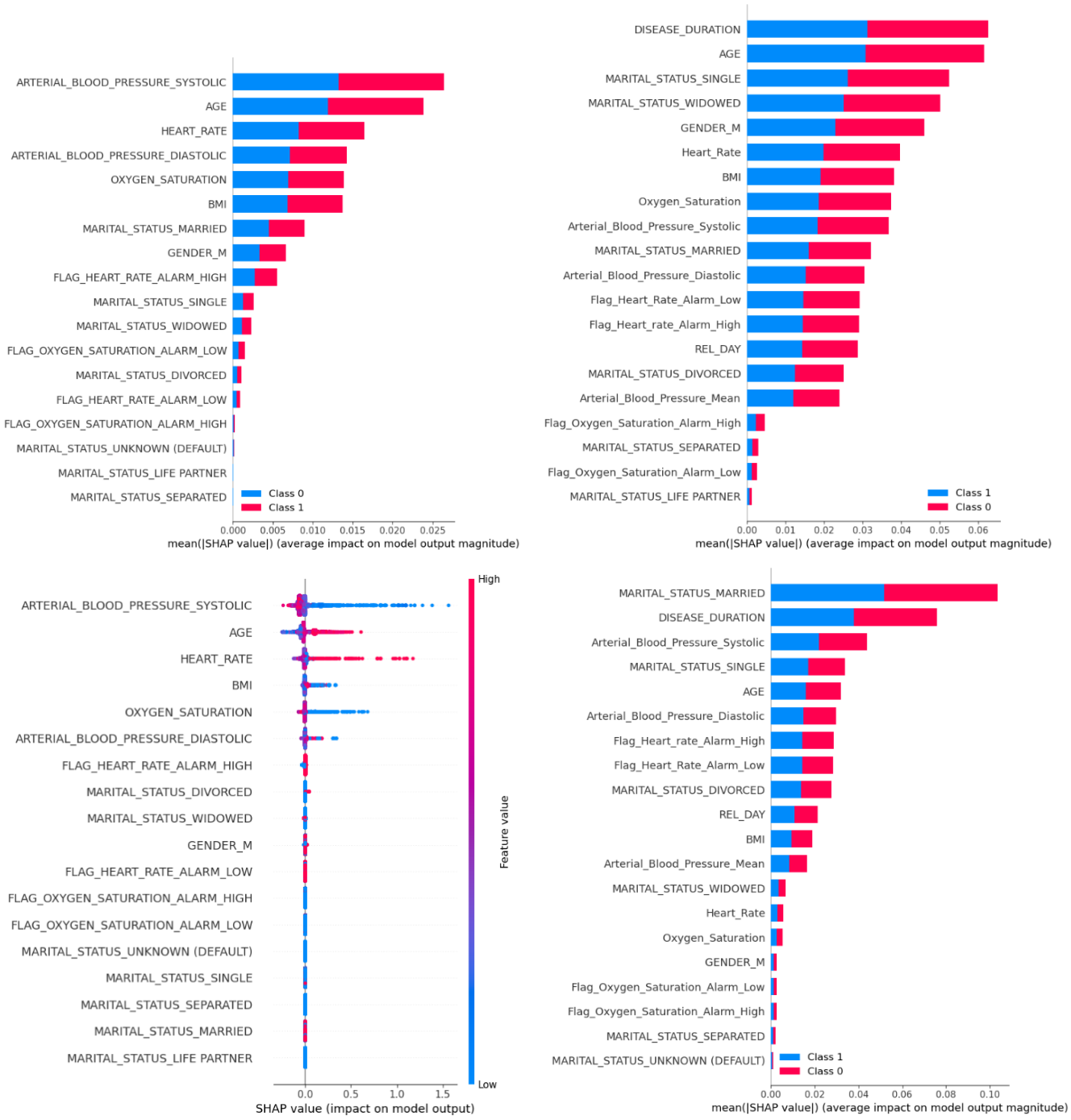


Figure A.5: SHAP plots for remaining models. On the top left is SHAP plot for DecisionTreeClassifier, top right is SHAP plot for KNN model, bottom left is SHAP plot for XGBoost and bottom right is SHAP plot for SVM model respectively.

Table1 for checking importance for feature values will be made available in Appendix B.

**Hyperparameters tested:**

- Logistic Regression:

  'C': np.logspace(-2, 0, 20), penalty': ['none', 'l2', 'l1', 'elasticnet'], 'solver': ['newton-cg', 'lbfgs', 'sag', 'saga'], 'multi_class': ['multinomial']

- RidgeClassifier:

  alpha = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

- SVM:

  kernel = ['poly', 'rbf', 'sigmoid', 'linear'], C = [50, 10, 1.0, 0.1, 0.01], gamma = ['scale']

- Random Forest:

  'criterion': ['gini', 'entropy', 'log_loss'], 'max_depth': [2*n for n in range(1,10)], 'max_features': ['auto', 'sqrt', 'log2'], 'min_samples_leaf': [1, 2, 4], 'min_samples_split': [2, 5, 10]

- KNN:

  'n_neighbors': list(range(1, 20)), 'weights': ['uniform', 'distance'], 'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'], 'p': [1,2]

- XGBoost:

  'max_depth': [4,5,6] , 'min_child_weight':[4,5,6] ,'learning_rate': [0.05,0.1,0.5] ,'n_estimators': [20,50,100]

- DecisionTreeClassifier:

  "max_depth": [3, None], "max_features": randint(1, 9), "min_samples_leaf": randint(1, 9), "criterion": ["gini", "entropy"] ,"splitter":["best"]

  An artificial neural network model was also designed to be used in this study, however due to bad performance of this model(classifier), it was not included in the main study description.

  As can be inferred from Figure A.6, the classifier performed poorly by displaying area under curve score of 54%. Hyperparameters used for this classifier:

  batch size=10, epoch=20, unit=11 and activation=relu

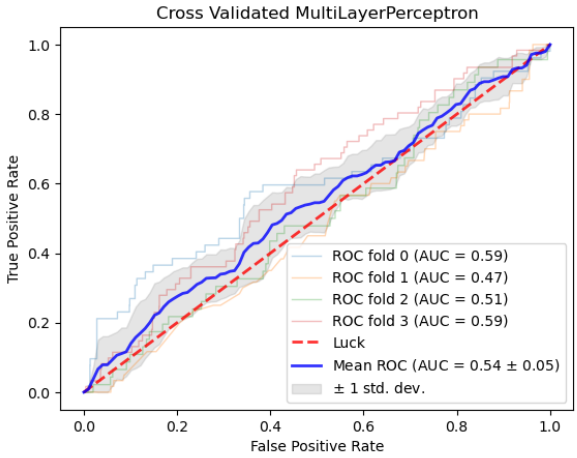


Figure A.6: 4-Fold cross validation result for Artificial Neural Network used for this study.

| Model Name | Threshold Set | Accuracy | Precision | Recall (Sensitivity) | F1 |
|---|---|---|---|---|---|
| Random Forest | 0.010 | 0.57 | 0.02 | 0.69 | 0.04 |
| Ridge Classifier | 0.008 | 0.51 | 0.01 | 0.67 | 0.03 |

Table A.1: Machine learning metrics for Random Forest model and Ridge Classifier model for threshold.

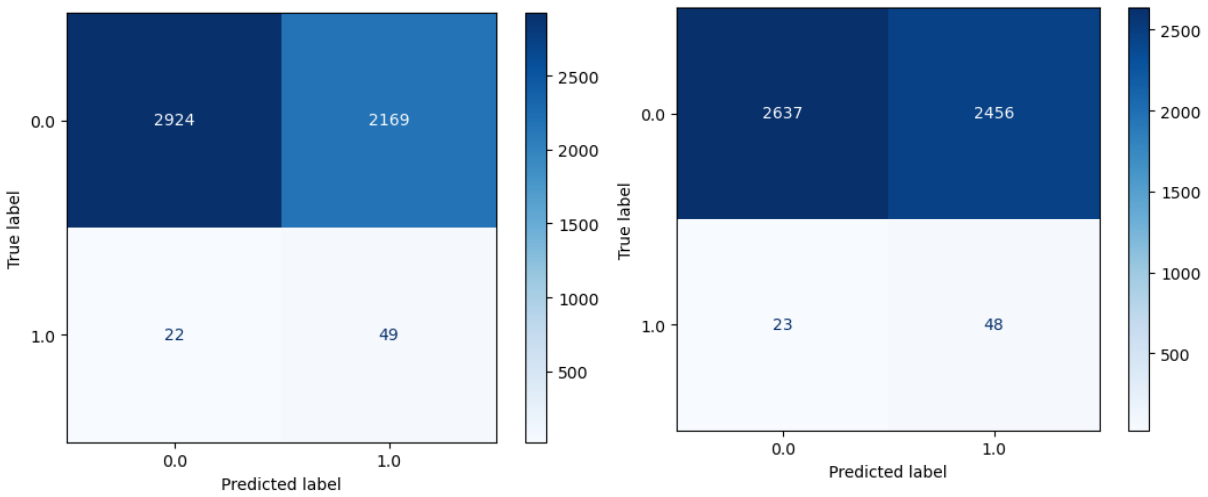


Figure A.7: Confusion matrix for Random Forest on the left (threshold set 0.010) and confusion matrix for Ridge Classifier on the right (threshold set 0.008).

# B

# Appendix B

All the codes, the demo dataset and other necessary documents will be provided in the form of electronic storage medium to the examination office.