Table of Contents

Section 1: Project Topic Description

Introduction

The primary goal of this project is to develop and validate a prediction model for all-cause in-hospital mortality among admitted patients. This model aims to enhance the predictive capabilities of healthcare systems enabling better patient management and allocation of resources.

Problem Statement

In-hospital mortality is a critical metric that hospitals use to gauge the quality of care. Despite its importance, the predictors of mortality among admitted patients remain poorly characterized. This project seeks to address this gap by leveraging machine learning techniques to predict the likelihood of in-hospital death based on a range of clinical and demographic factors.

Type of Learning

The task at hand is a supervised learning problem because we have labeled data indicating whether patients survived or died during their hospital stay. Supervised learning is appropriate here as it involves using a set of known labels (outcomes of patients) to train models that can predict the outcome of new unseen data.

Algorithms & Task

The specific task within supervised learning for this project is binary classification. Binary classification is used because the output variable (“hospital_death” representing patient mortality) is binary (i.e., the patient either survives or does not survive their hospital stay).

For the project, I chose to explore several algorithms suited for binary classification tasks including:

Logistic Regression: A fundamental technique suitable for baseline comparison due to its simplicity and interpretability.

Random Forests: An ensemble learning method known for its high accuracy and robustness against overfitting.

Support Vector Machines (SVM): Effective in high-dimensional spaces which is typical in medical data sets with many predictors.

Neural Networks: Highly flexible and powerful models capable of learning complex patterns in large datasets. They are particularly useful for capturing nonlinear relationships in data.

XGBoost: A gradient boosting framework that is renowned for its performance and speed in classification problems. XGBoost can handle large data sets and complex features making it highly effective for predictive tasks involving heterogeneous data.

Each of these algorithms has distinct advantages and potential drawbacks and part of this project will involve evaluating which algorithm performs best under various conditions specific to our dataset.

Data Set Explanation and Descriptions

Citation in APA format:

Agarwal, M. (n.d.). Patient. Kaggle. Retrieved from https://www.kaggle.com/datasets/mitishaagarwal/patient

The dataset titled "Patient Survival Prediction" was compiled by Mitisha Agarwal and is hosted on Kaggle which is a platform that allows users to find and publish datasets in a web-based data

science environment. While specific details regarding the methodology of data collection are not provided on the dataset's webpage, it typically includes clinical and demographic information collected from patients admitted to hospitals.

This dataset presented a comprehensive collection of clinical information pertaining to patient admissions in hospitals specifically curated to study in-hospital mortality. Comprising 91,713 individual patient records, the dataset encompasses a broad spectrum of 85 distinct features, which encapsulate both the patient's personal and clinical details. The data set size is 31.41MB.

The data included a diverse range of feature types, including continuous numeric variables such as age, body mass index (BMI), and various physiological parameters monitored during the patients' ICU stay. For example, it records maximum and minimum values of heart rate, blood pressure, and temperature within specific time frames with the first day and first hour of ICU admission. Additionally, there are categorical variables that represent the patient's demographics (like ethnicity and gender), medical status (such as presence of comorbidities including AIDS, cirrhosis, and diabetes mellitus), and healthcare journey (like ICU admission source and type of ICU).

Indicator variables encoded as 0 or 1 flags provide binary data for conditions such as elective surgery status, post-operative status, and life support interventions like intubation and ventilation. Furthermore, APACHE score components are included offering critical insights into the severity of disease and predicted probabilities of hospital and ICU death.

The dataset also includes several encoded identifiers: encounter_id, patient_id, hospital_id, and icu_id, which are unique to each hospital visit, patient, hospital, and ICU, respectively. These features were not used in the modeling phase.

Key diagnostic variables, such as the APACHE II and III diagnosis categories and body system involved, are included enhancing the potential for specialized medical analyses. However, it's noted that the dataset includes an Unnamed: 83 column with zero values (all missing data) and thus was removed from the analysis.

The outcome variable of interest "hospital_death" is a binary indicator of whether the patient succumbed to their ailments during their hospital stay.

Below is a table with features names, data types, description, and variable type (numeric vs categorical) where N/A stands for features not used in training:

Column Name	Data Type	Description	Variable Type
encounter_id	int64	A unique identifier for each patient encounter in the hospital.	N/A

patient_id	int64	A unique identifier for each patient.	N/A
hospital_id	int64	A unique identifier for each hospital.	N/A
icu_id	int64	A unique identifier for each ICU.	N/A
age	float64	The patient's age in years.	Numeric
bmi	float64	The patient's body mass index.	Numeric
elective_surgery	int64	Indicator (0 or 1) whether the surgery was elective.	Categorical
ethnicity	object	The patient's self-reported ethnicity.	Categorical
gender	object	The patient's gender.	Categorical
height	float64	The patient's height in centimeters.	Numeric
icu_admit_source	object	The source from which the patient was admitted to the ICU.	Categorical
icu_stay_type	object	Categorizes the type of ICU stay (e.g., admission, readmission).	Categorical
icu_type	object	The type of ICU (e.g., cardiac, general, neuro).	Categorical

pre_icu_los_days	float64	Length of stay before ICU admission in days.	Numeric
weight	float64	The patient's weight in kilograms.	Numeric
apache_2_diagnosis	float64	The APACHE II diagnosis category.	Numeric
apache_3j_diagnosis	float64	The APACHE III diagnosis category.	Numeric
apache_post_operative	int64	Indicator (0 or 1) if the patient is post-operative.	Categorical
arf_apache	float64	Acute renal failure as per APACHE scoring.	Numeric
gcs_eyes_apache	float64	Glasgow Coma Scale - eye response.	Numeric
gcs_motor_apache	float64	Glasgow Coma Scale - motor response.	Numeric
gcs_verbal_apache	float64	Glasgow Coma Scale - verbal response.	Numeric
gcs_unable_apache	float64	Indicator if GCS was unassessable.	Numeric
heart_rate_apache	float64	Heart rate as per APACHE scoring.	Numeric
intubated_apache	int64	Indicator (0 or 1) if the patient was intubated.	Categorical

map_apache	float64	Mean arterial pressure as per APACHE scoring.	Numeric
resprate_apache	float64	Respiratory rate as per APACHE scoring.	Numeric
temp_apache	float64	Temperature as per APACHE scoring.	Numeric
ventilated_apache	int64	Indicator (0 or 1) if the patient was ventilated.	Categorical
d1_diasbp_max	float64	Maximum diastolic blood pressure on the first day in ICU.	Numeric
d1_diasbp_min	float64	Minimum diastolic blood pressure on the first day in ICU.	Numeric
d1_diasbp_noninvasive_max	float64	Max non-invasive diastolic blood pressure on the first day in ICU.	Numeric
d1_diasbp_noninvasive_min	float64	Min non-invasive diastolic blood pressure on the first day in ICU.	Numeric
d1_heartrate_max	float64	Maximum heart rate on the first day in ICU.	Numeric
d1_heartrate_min	float64	Minimum heart rate on the first day in ICU.	Numeric
d1_mbp_max	float64	Maximum mean blood pressure on the first day in ICU.	Numeric

d1_mbp_min	float64	Minimum mean blood pressure on the first day in ICU.	Numeric
d1_mbp_noninvasive_max	float64	Max non-invasive mean blood pressure on the first day in ICU.	Numeric
d1_mbp_noninvasive_min	float64	Min non-invasive mean blood pressure on the first day in ICU.	Numeric
d1_resprate_max	float64	Maximum respiratory rate on the first day in ICU.	Numeric
d1_resprate_min	float64	Minimum respiratory rate on the first day in ICU.	Numeric
d1_spo2_max	float64	Maximum peripheral oxygen saturation on the first day in ICU.	Numeric
d1_spo2_min	float64	Minimum peripheral oxygen saturation on the first day in ICU.	Numeric
d1_sysbp_max	float64	Maximum systolic blood pressure on the first day in ICU.	Numeric
d1_sysbp_min	float64	Minimum systolic blood pressure on the first day in ICU.	Numeric
d1_sysbp_noninvasive_max	float64	Max non-invasive systolic blood pressure on the first day in ICU.	Numeric

d1_sysbp_noninvasive_min	float64	Min non-invasive systolic blood pressure on the first day in ICU.	Numeric
d1_temp_max	float64	Maximum temperature on the first day in ICU.	Numeric
d1_temp_min	float64	Minimum temperature on the first day in ICU.	Numeric
h1_diasbp_max	float64	Maximum diastolic blood pressure in the first hour in ICU.	Numeric
h1_diasbp_min	float64	Minimum diastolic blood pressure in the first hour in ICU.	Numeric
h1_diasbp_noninvasive_max	float64	Max non-invasive diastolic blood pressure in the first hour in ICU.	Numeric
h1_diasbp_noninvasive_min	float64	Min non-invasive diastolic blood pressure in the first hour in ICU.	Numeric
h1_heartrate_max	float64	Maximum heart rate in the first hour in ICU.	Numeric
h1_heartrate_min	float64	Minimum heart rate in the first hour in ICU.	Numeric
h1_mbp_max	float64	Maximum mean blood pressure in the first hour in ICU.	Numeric

h1_mbp_min	float64	Minimum mean blood pressure in the first hour in ICU.	Numeric
h1_mbp_noninvasive_max	float64	Max non-invasive mean blood pressure in the first hour in ICU.	Numeric
h1_mbp_noninvasive_min	float64	Min non-invasive mean blood pressure in the first hour in ICU.	Numeric
h1_resprate_max	float64	Maximum respiratory rate in the first hour in ICU.	Numeric
h1_resprate_min	float64	Minimum respiratory rate in the first hour in ICU.	Numeric
h1_spo2_max	float64	Maximum peripheral oxygen saturation in the first hour in ICU.	Numeric
h1_spo2_min	float64	Minimum peripheral oxygen saturation in the first hour in ICU.	Numeric
h1_sysbp_max	float64	Maximum systolic blood pressure in the first hour in ICU.	Numeric
h1_sysbp_min	float64	Minimum systolic blood pressure in the first hour in ICU.	Numeric
h1_sysbp_noninvasive_max	float64	Max non-invasive systolic blood pressure in the first hour in ICU.	Numeric

h1_sysbp_noninvasive_min	float64	Min non-invasive systolic blood pressure in the first hour in ICU.	Numeric
d1_glucose_max	float64	Maximum glucose level on the first day in ICU.	Numeric
d1_glucose_min	float64	Minimum glucose level on the first day in ICU.	Numeric
d1_potassium_max	float64	Maximum potassium level on the first day in ICU.	Numeric
d1_potassium_min	float64	Minimum potassium level on the first day in ICU.	Numeric
apache_4a_hospital_death_prob	float64	The predicted probability of hospital death by APACHE IVa.	Numeric
apache_4a_icu_death_prob	float64	The predicted probability of ICU death by APACHE IVa.	Numeric
aids	float64	Indicator (0 or 1) if the patient has AIDS.	Categorical
cirrhosis	float64	Indicator (0 or 1) if the patient has cirrhosis.	Categorical
diabetes_mellitus	float64	Indicator (0 or 1) if the patient has diabetes mellitus.	Categorical

hepatic_failure	float64	Indicator (0 or 1) if the patient has hepatic failure.	Categorical
immunosuppression	float64	Indicator (0 or 1) if the patient is immunosuppressed.	Categorical
leukemia	float64	Indicator (0 or 1) if the patient has leukemia.	Categorical
lymphoma	float64	Indicator (0 or 1) if the patient has lymphoma.	Categorical
solid_tumor_with_metastasis	float64	Indicator (0 or 1) if the patient has a solid tumor with metastasis.	Categorical
apache_3j_bodysystem	object	The body system category as per APACHE III.	Categorical
apache_2_bodysystem	object	The body system category as per APACHE II.	Categorical
Unnamed: 83	float64	This seems to be an unidentified or improperly labeled column and will be deleted.	N/A
hospital_death	int64	Indicator (0 or 1) if the patient died in the hospital.	Categorical

Data Distributions

Visualizing data distributions is a critical step in the data analysis process as it provides immediate insights into the shape, variability, and central tendencies of the data. This

visualization helps in identifying patterns, spotting anomalies, and understanding the underlying structure of the dataset. For categorical data, bar charts highlight the frequency of categories, showing imbalances or dominant categories that could impact predictive modeling. For numeric features, distribution type influences the kind of statistical analysis and modeling techniques that are most appropriate. The visual examinations of these distributions can be used to detect outliers.

To further evaluate how close these distributions adhered to normality, I did run skewness and kurtosis tests. **Skewness** measures the asymmetry of a distribution. A positive skew indicates a tail that stretches towards higher values while a negative skew indicates a tail that stretches towards lower values. A skewness close to zero suggests a symmetric distribution. **Kurtosis** measures the tails' heaviness compared to a normal distribution. High kurtosis means more of the variance is due to infrequent extreme deviations as opposed to frequent modestly sized deviations.

Given the distribution of values found in the table below, I chose to focus my modeling efforts on approaches that do not have normality assumptions.

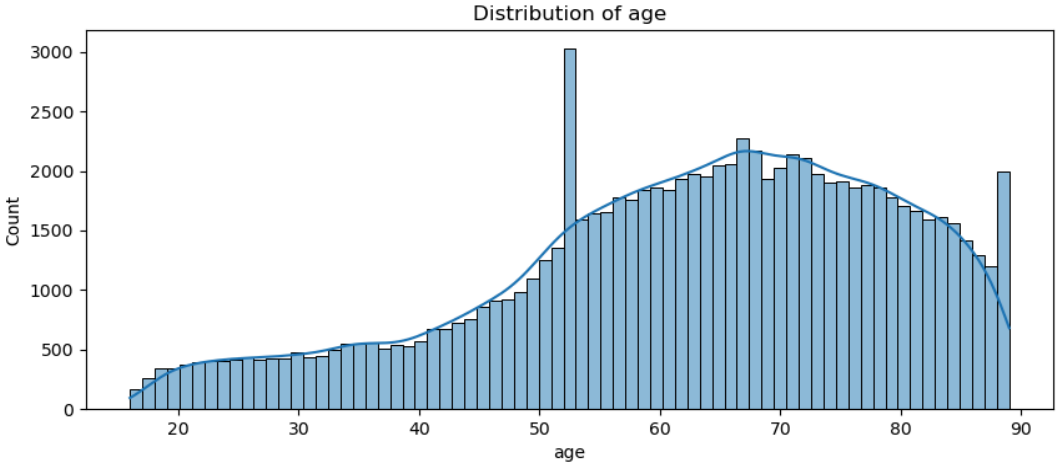
Variable	Skewness	Kurtosis
age	-0.62	-0.21
bmi	1.44	3.41
height	-0.10	-0.39
pre_icu_los_days	10.99	311.72
apache_2_diagnosis	0.51	-1.59
apache_3j_diagnosis	1.01	0.02
apache_4a_icu_death_prob	-2.03	13.83
arf_apache	5.72	30.77
gcs_motor_apache	-2.71	6.32
gcs_verbal_apache	-1.20	-0.30

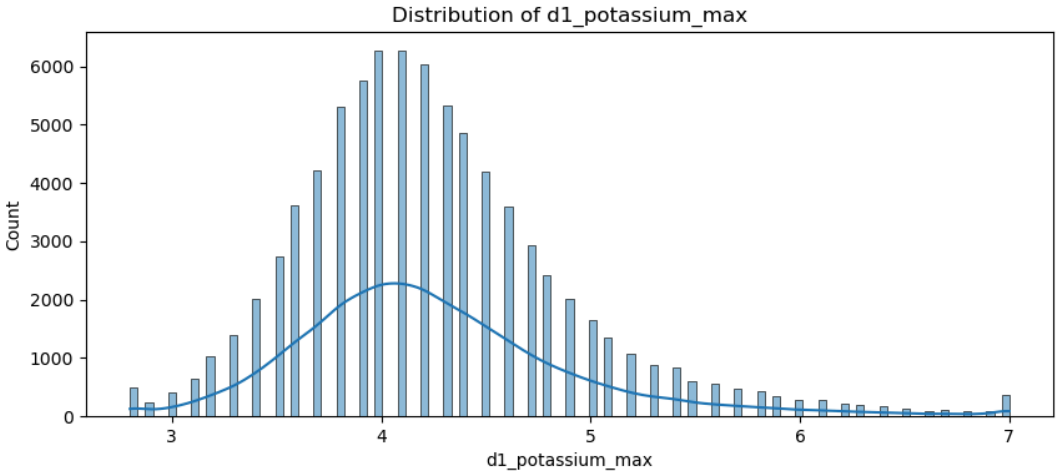
gcs_unable_apache	10.10	99.96
map_apache	0.70	-0.79
resprate_apache	0.26	-0.93
temp_apache	-0.97	8.98
d1_heartrate_max	0.57	0.34
d1_heartrate_min	-0.11	1.82
d1_mbp_noninvasive_max	0.75	0.92
d1_mbp_noninvasive_min	0.20	0.34
d1_resprate_max	2.50	9.54
d1_resprate_min	0.17	4.16
d1_spo2_max	-19.30	875.74
d1_spo2_min	-4.77	31.57
d1_sysbp_max	0.51	0.25
d1_sysbp_min	0.22	0.31
d1_temp_max	0.85	1.93
d1_temp_min	-2.85	12.51
h1_heartrate_min	0.39	-0.03

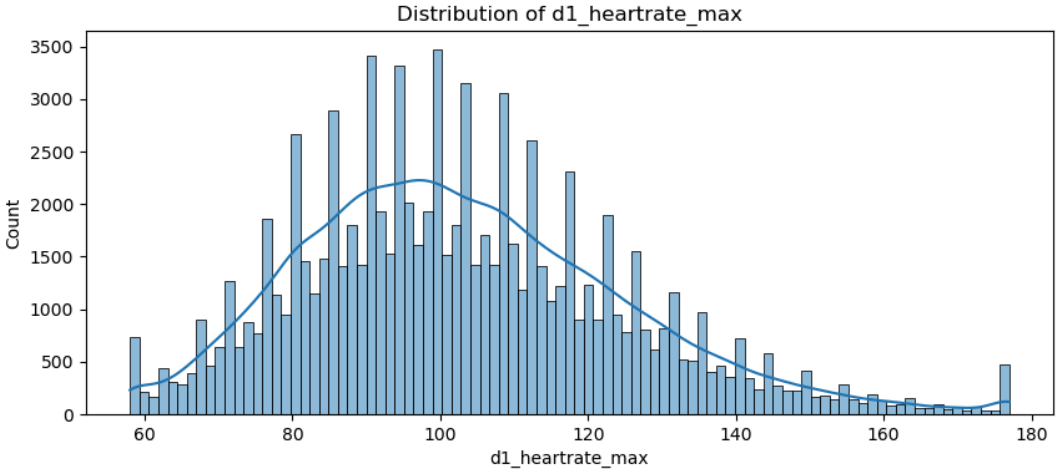
h1_mbp_noninvasive_max	0.63	0.51
h1_resprate_max	1.53	3.79
h1_resprate_min	1.88	20.96
h1_spo2_max	-10.81	252.22
h1_spo2_min	-6.77	73.86
h1_sysbp_max	0.55	0.24
h1_sysbp_min	0.29	-0.04
d1_glucose_max	2.06	5.47
d1_glucose_min	1.38	3.41
d1_potassium_max	1.05	2.07
d1_potassium_min	0.27	0.46

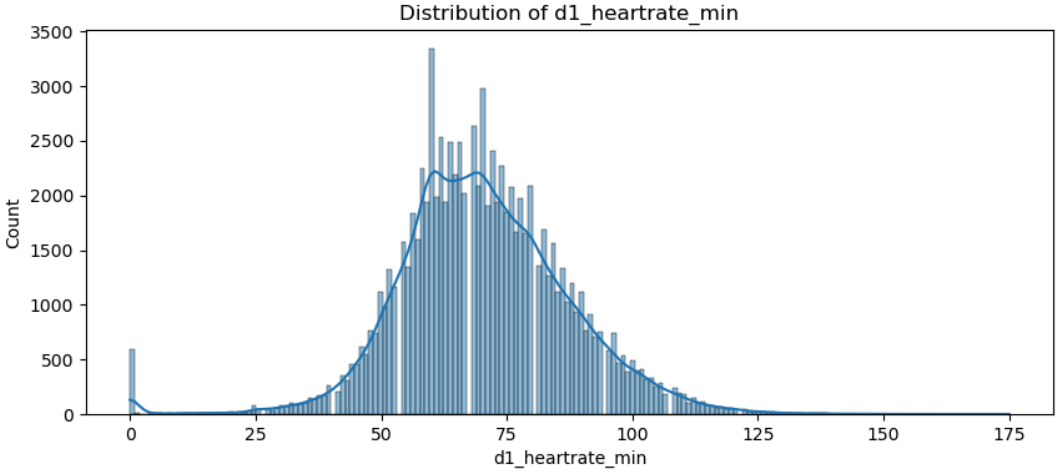
Below are all the distribution plots for both categorical and numerical features:

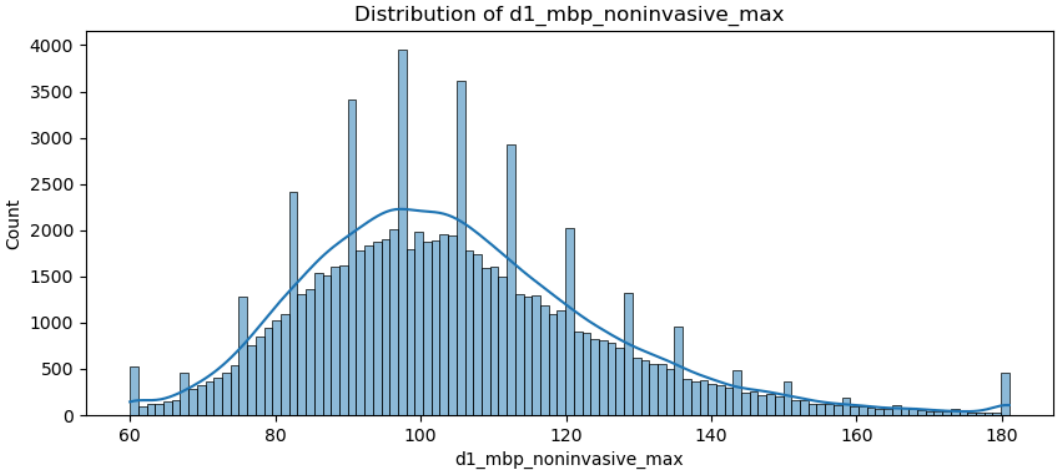
Numeric Feature Distributions

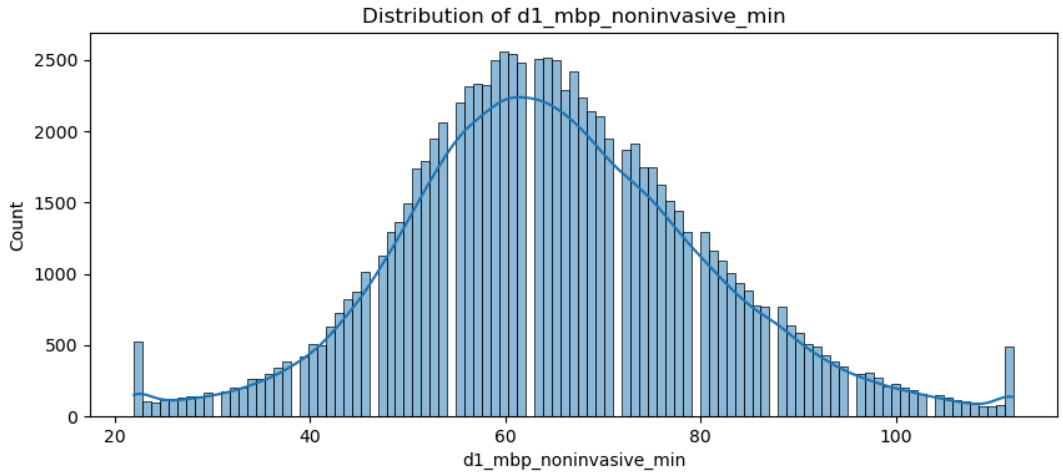

Distribution of age
Count
age
0
500
1000
1500
2000
2500
3000
20
30
40
50
60
70
80
90

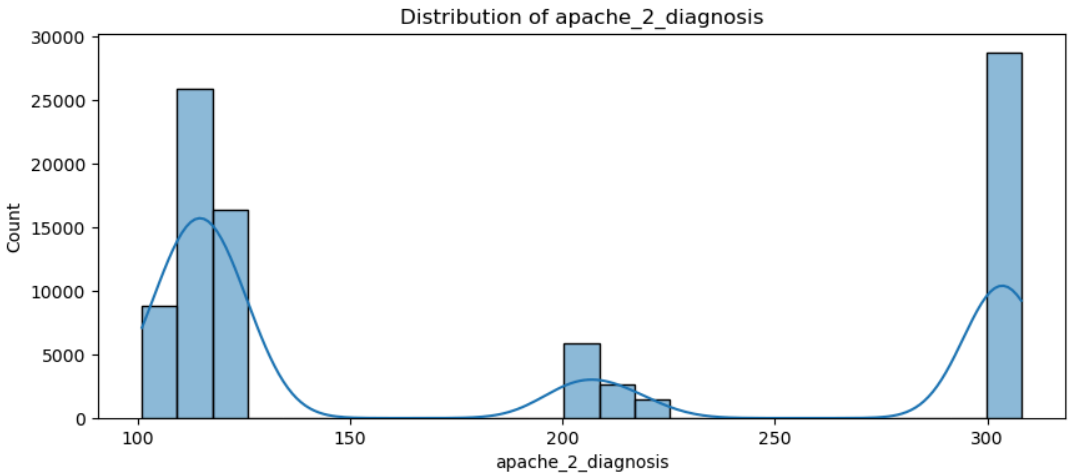

Distribution of d1_potassium_max
Count
d1_potassium_max
0
1000
2000
3000
4000
5000
6000
3
4
5
6
7

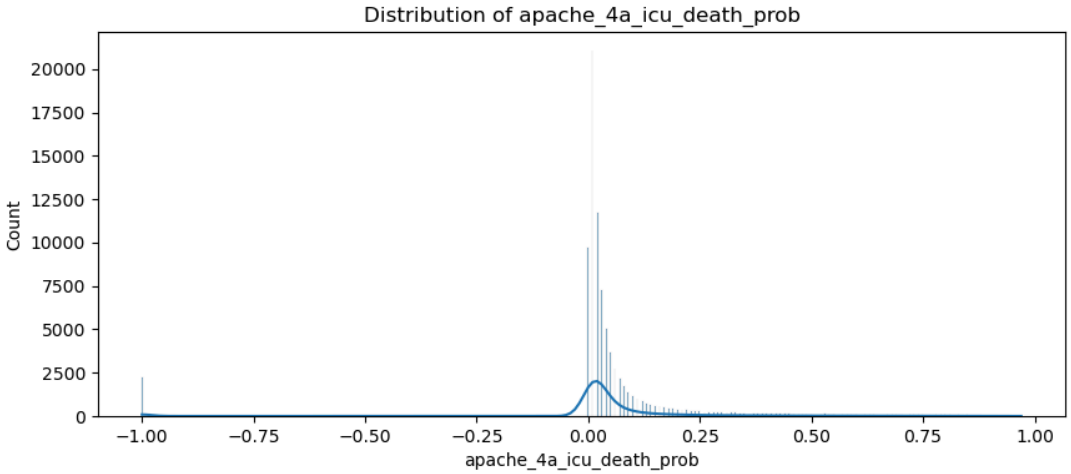

Distribution of d1_heartrate_max
Count
d1_heartrate_max
0
500
1000
1500
2000
2500
3000
3500
60
80
100
120
140
160
180

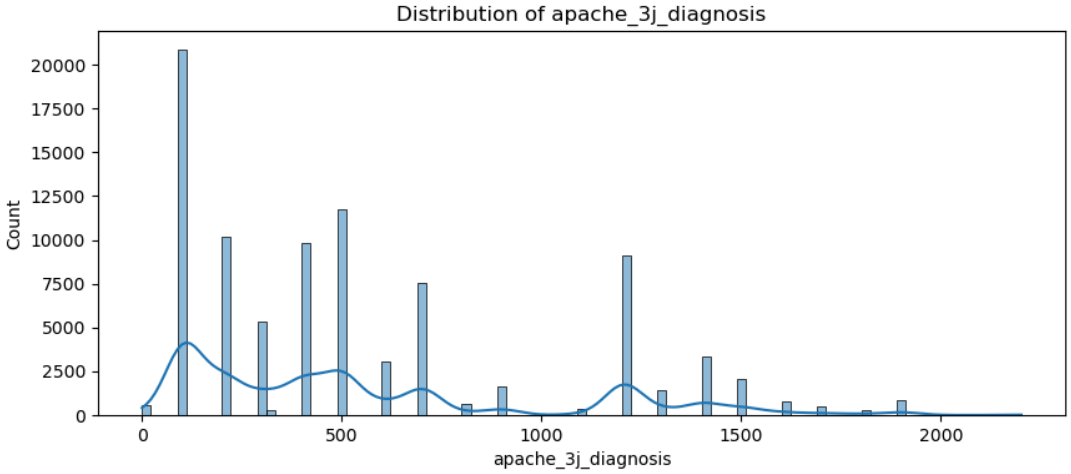

Distribution of d1_heartrate_min
Count
d1_heartrate_min
0
500
1000
1500
2000
2500
3000
3500
0
25
50
75
100
125
150
175

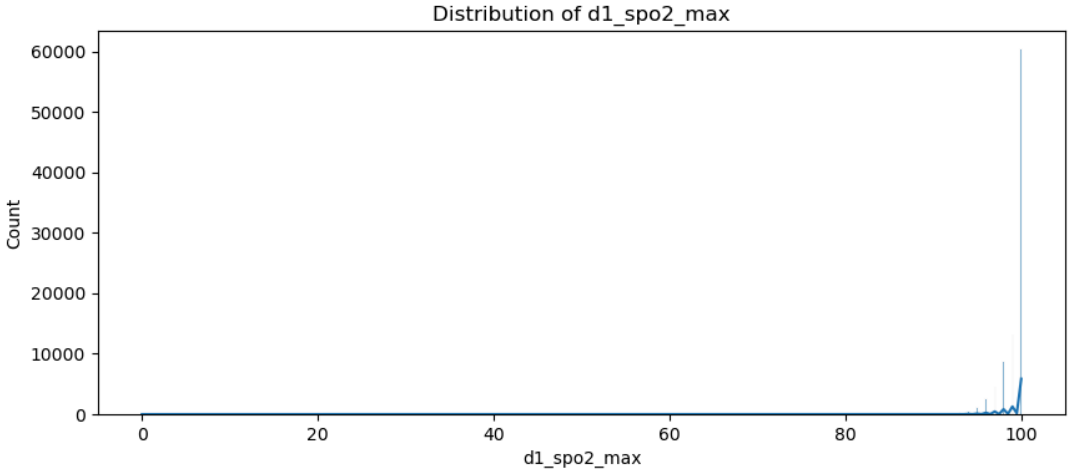

Distribution of d1_mbp_noninvasive_max
Count
d1_mbp_noninvasive_max
0
500
1000
1500
2000
2500
3000
3500
4000
60
80
100
120
140
160
180

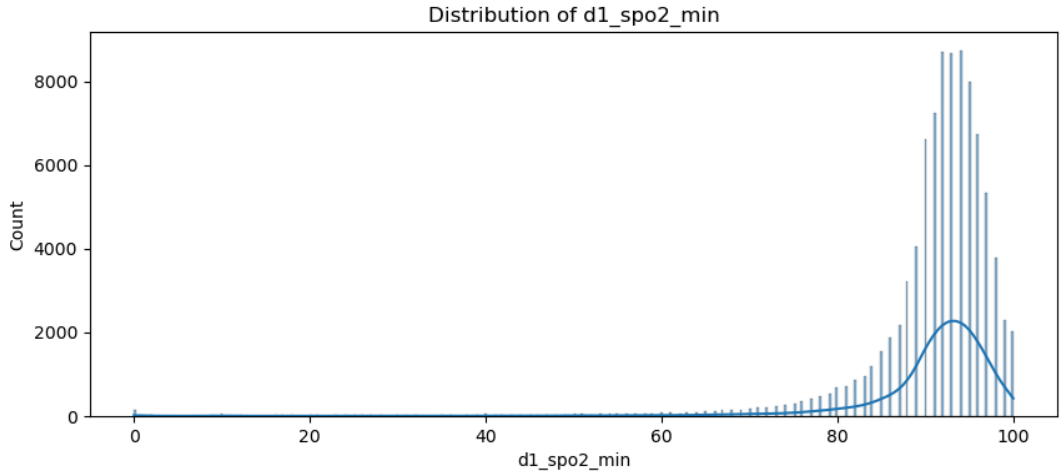

Distribution of d1_mbp_noninvasive_min
Count
d1_mbp_noninvasive_min
0
500
1000
1500
2000
2500
20
40
60
80
100

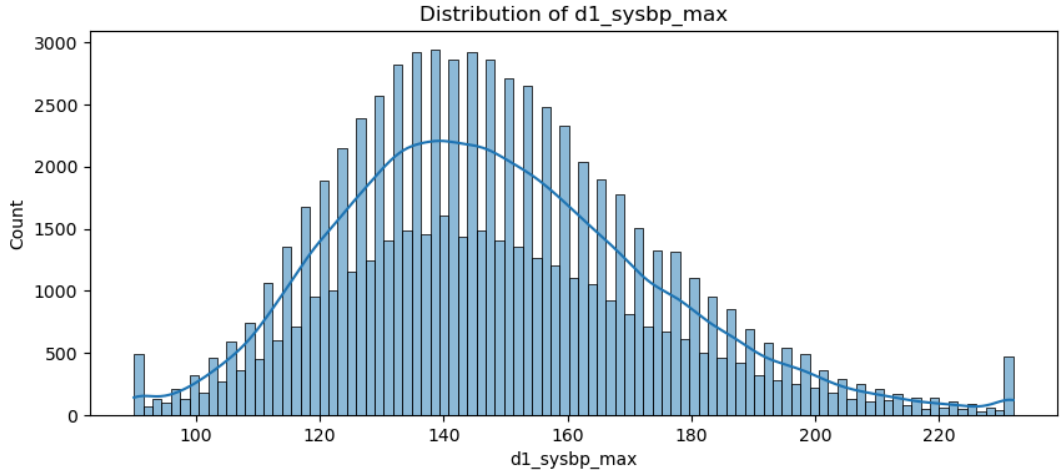

Distribution of apache_2_diagnosis
Count
apache_2_diagnosis
0
5000
10000
15000
20000
25000
30000
100
150
200
250
300

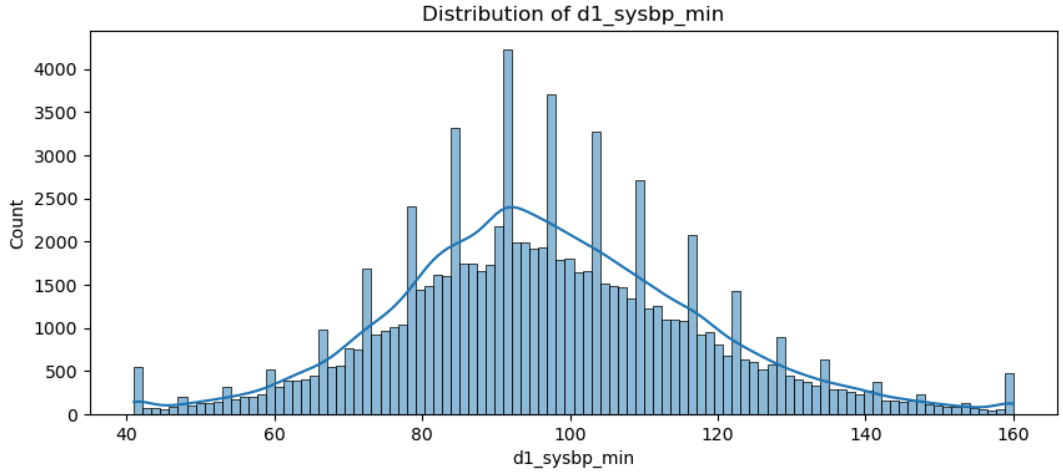

Distribution of apache_4a_icu_death_prob
Count
20000
17500
15000
12500
10000
7500
5000
2500
0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
apache_4a_icu_death_prob

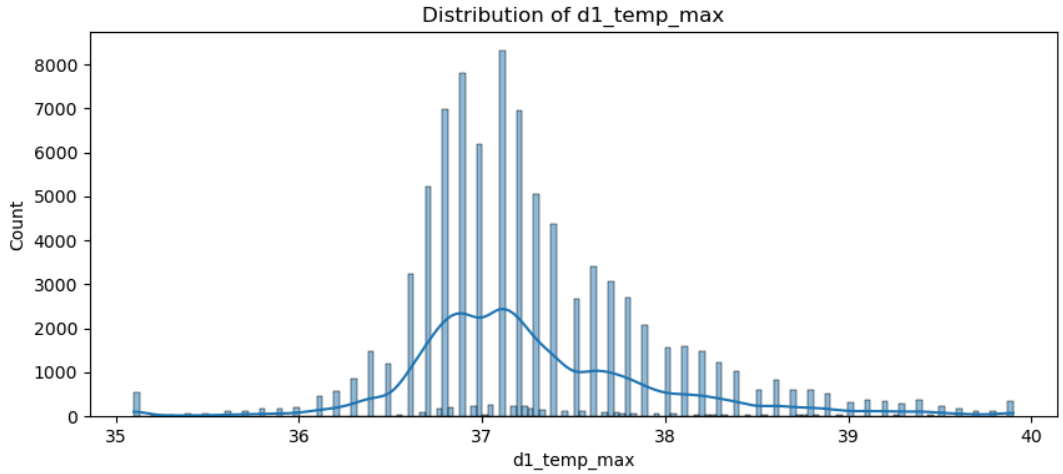

Distribution of apache_3j_diagnosis
Count
20000
17500
15000
12500
10000
7500
5000
2500
0
0
500
1000
1500
2000
apache_3j_diagnosis

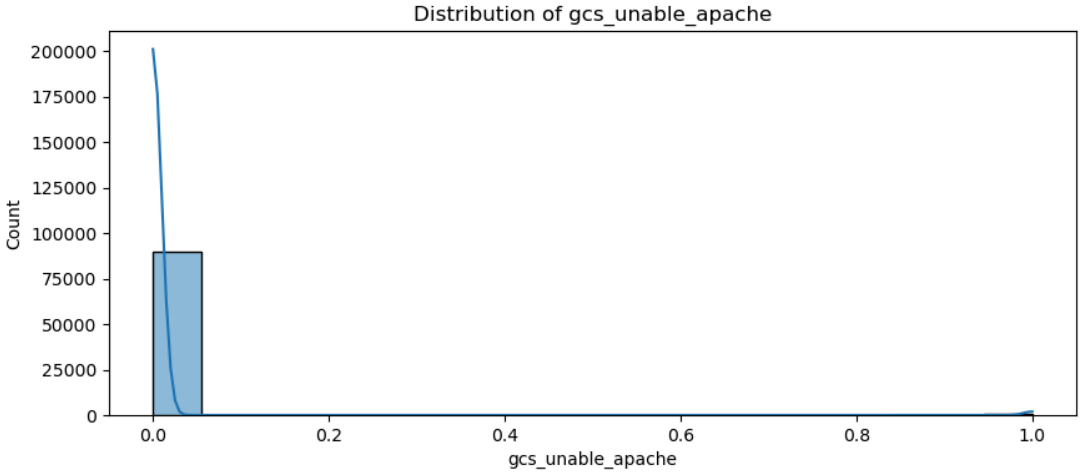

Distribution of d1_spo2_max
Count
d1_spo2_max
0
10000
20000
30000
40000
50000
60000
0
20
40
60
80
100

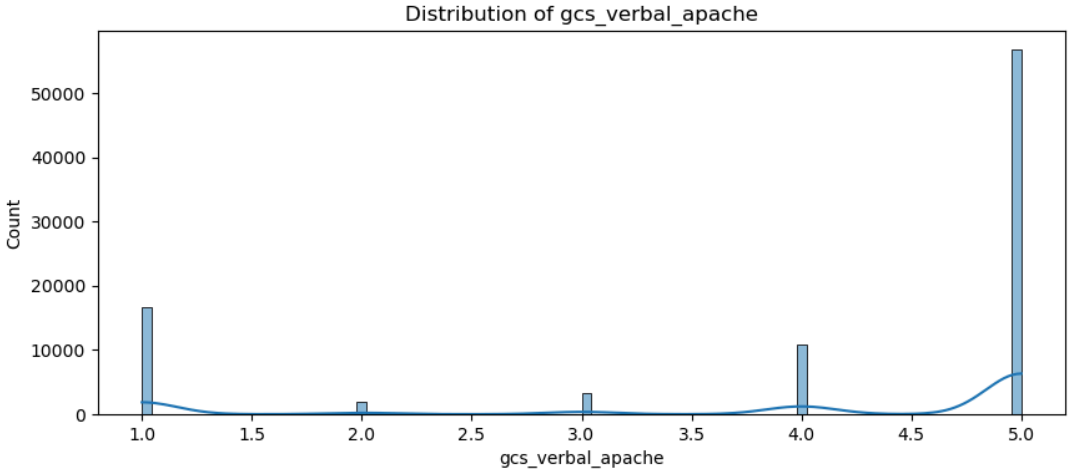

Distribution of d1_spo2_min
Count
d1_spo2_min
0
2000
4000
6000
8000
0
20
40
60
80
100

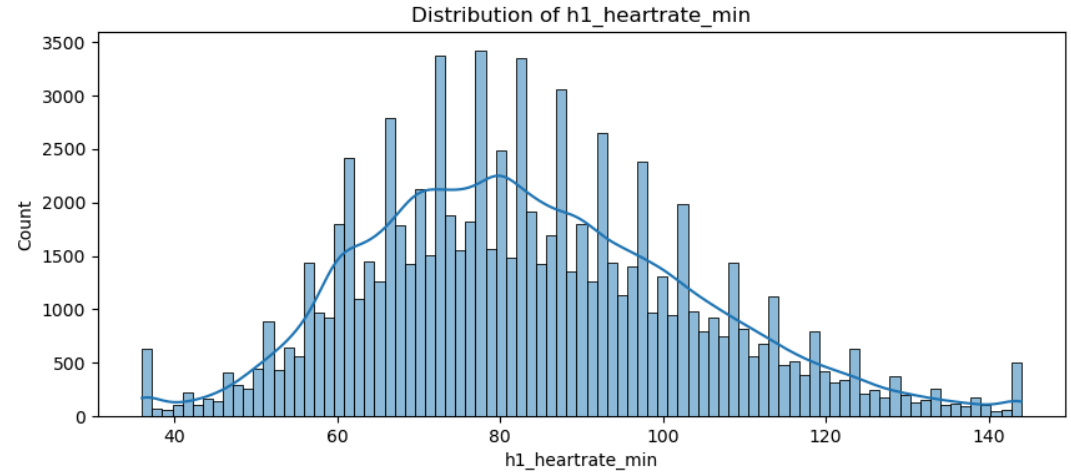

Distribution of d1_sysbp_max
Count
d1_sysbp_max
0
500
1000
1500
2000
2500
3000
100
120
140
160
180
200
220

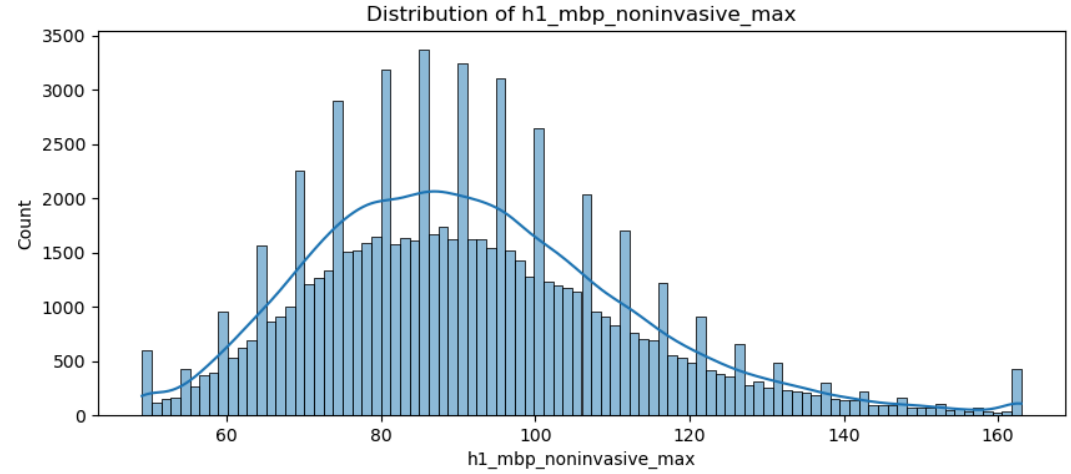

Distribution of d1_sysbp_min
Count
d1_sysbp_min
0
500
1000
1500
2000
2500
3000
3500
4000
40
60
80
100
120
140
160

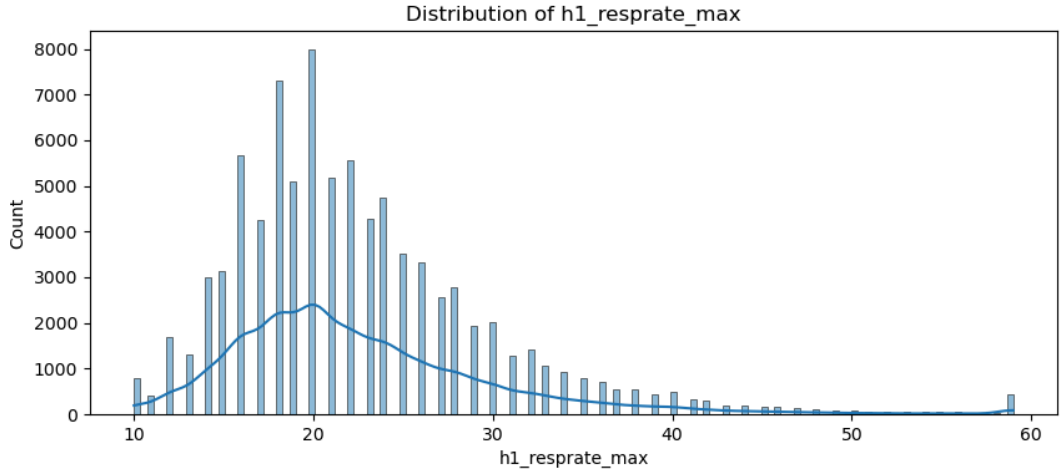

Distribution of d1_temp_max
Count
d1_temp_max
0
1000
2000
3000
4000
5000
6000
7000
8000
35
36
37
38
39
40

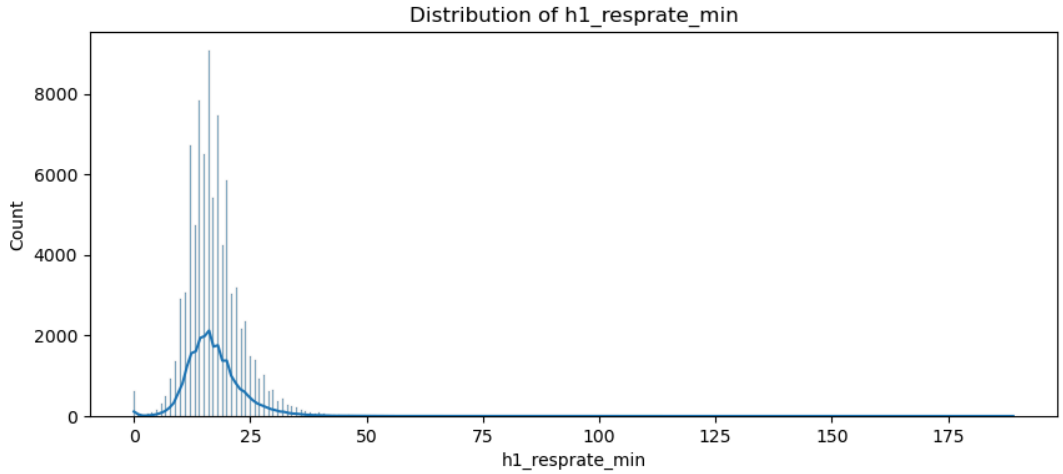

Distribution of gcs_unable_apache
Count
gcs_unable_apache
0
25000
50000
75000
100000
125000
150000
175000
200000
0.0
0.2
0.4
0.6
0.8
1.0

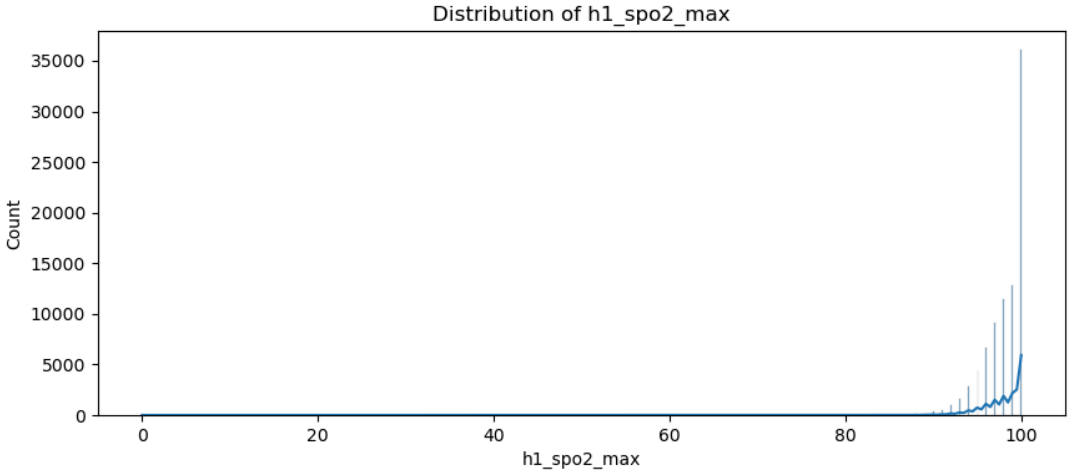

Distribution of gcs_verbal_apache
Count
gcs_verbal_apache
0
10000
20000
30000
40000
50000
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

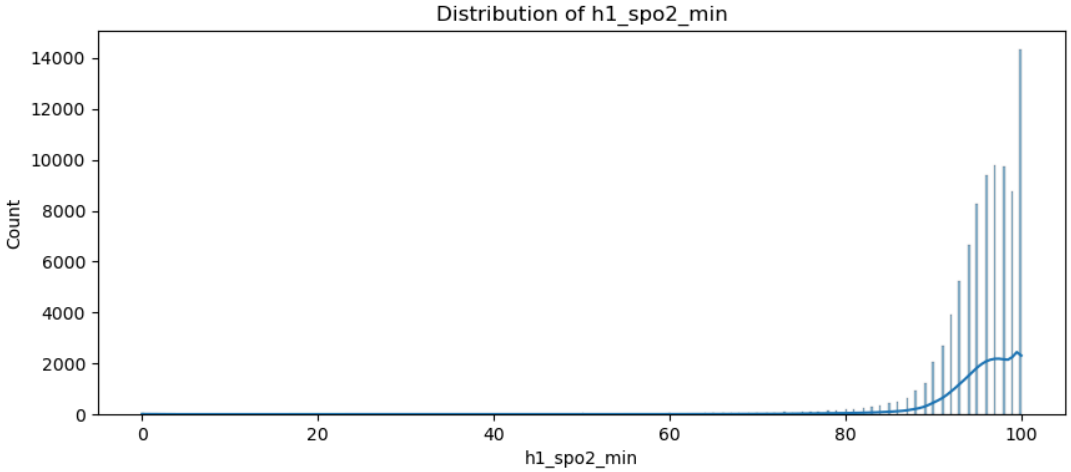

Distribution of h1_heartrate_min
Count
h1_heartrate_min
0
500
1000
1500
2000
2500
3000
3500
40
60
80
100
120
140

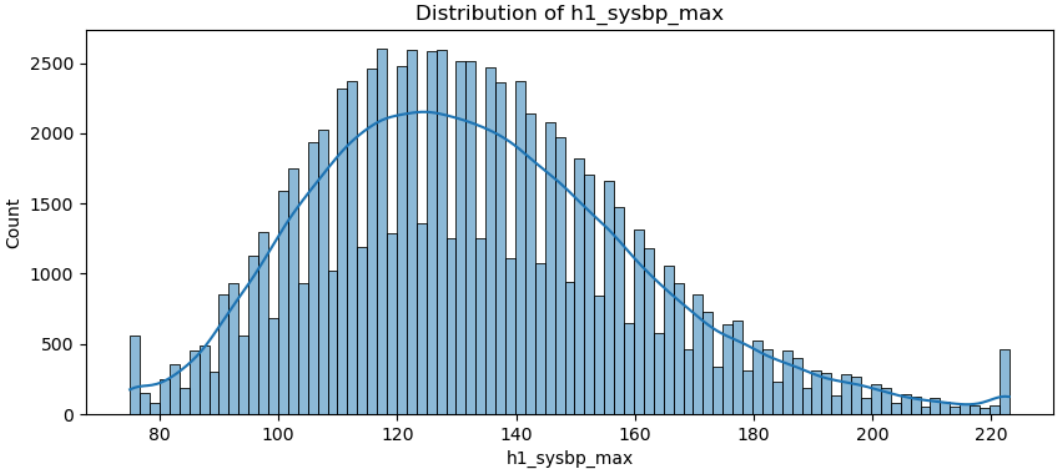

Distribution of h1_mbp_noninvasive_max
Count
h1_mbp_noninvasive_max
0
500
1000
1500
2000
2500
3000
3500
60
80
100
120
140
160

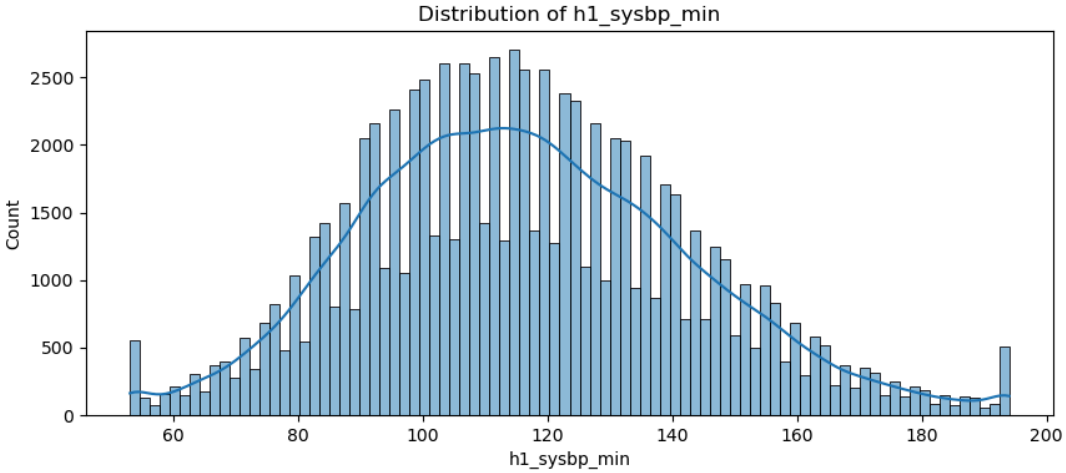

Distribution of h1_resprate_max
Count
h1_resprate_max
0
1000
2000
3000
4000
5000
6000
7000
8000
10
20
30
40
50
60

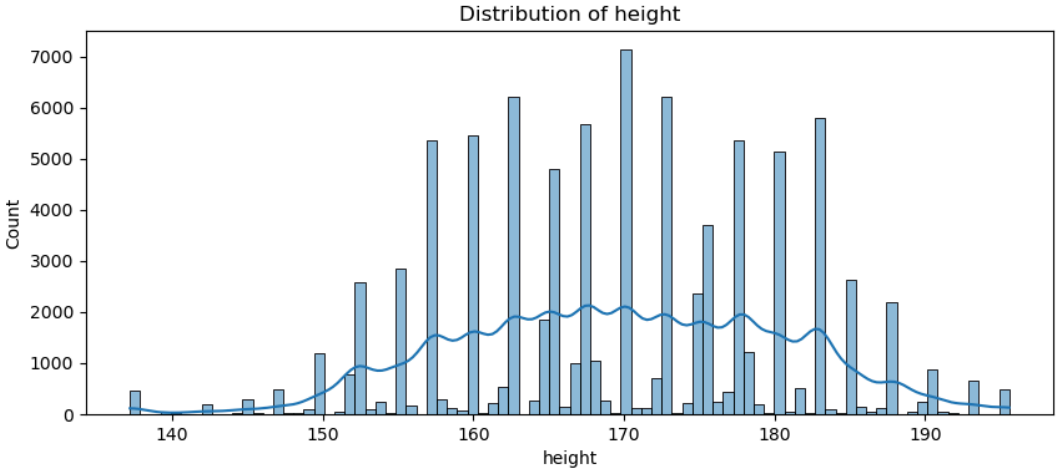

Distribution of h1_resprate_min
Count
h1_resprate_min
0
2000
4000
6000
8000
0
25
50
75
100
125
150
175

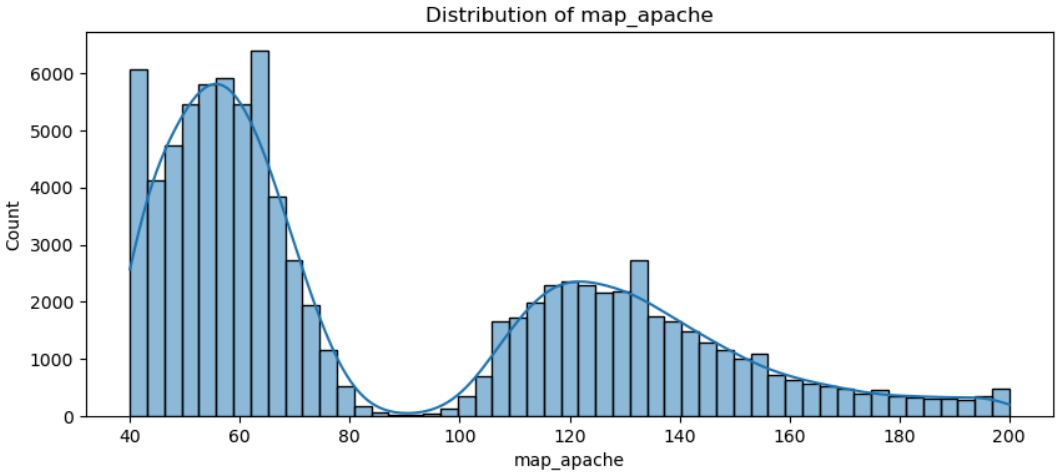

Distribution of h1_spo2_max
Count
h1_spo2_max
0
5000
10000
15000
20000
25000
30000
35000
0
20
40
60
80
100

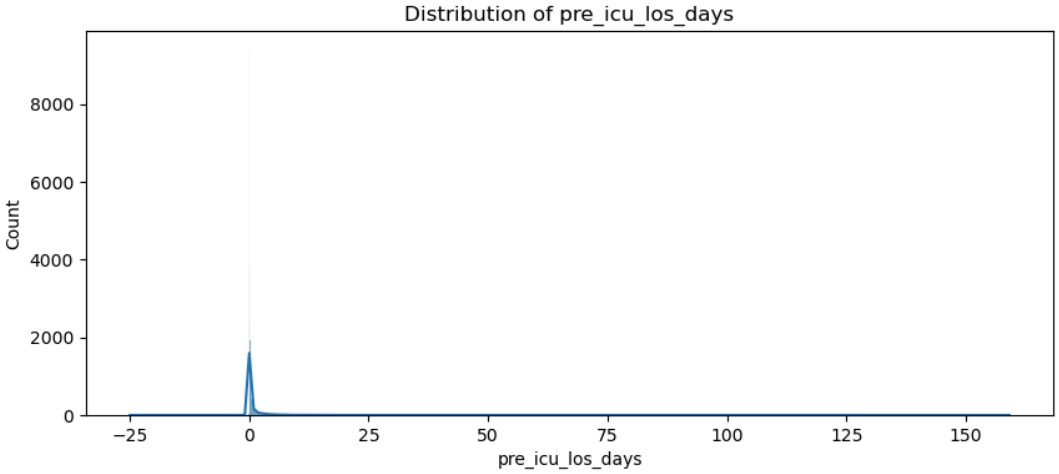

Distribution of h1_spo2_min
Count
h1_spo2_min
0
2000
4000
6000
8000
10000
12000
14000
0
20
40
60
80
100

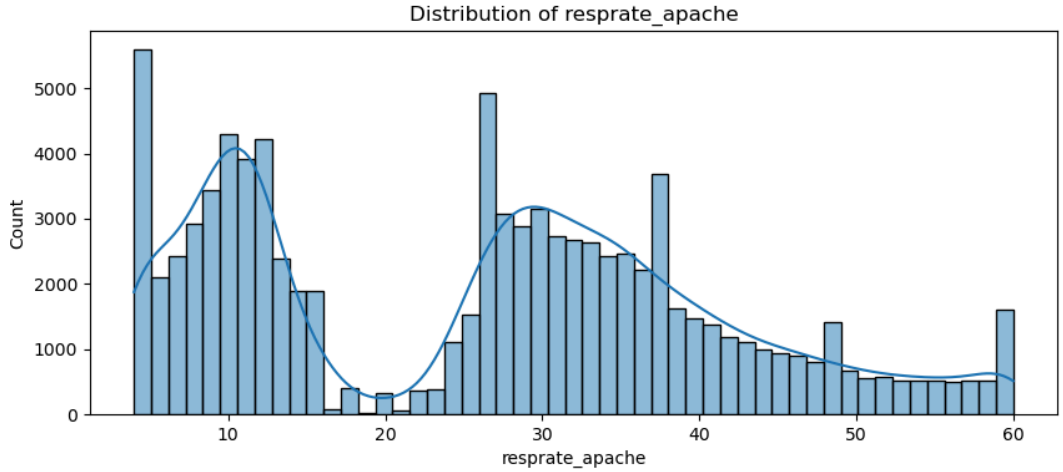

Distribution of h1_sysbp_max
Count
h1_sysbp_max
0
500
1000
1500
2000
2500
80
100
120
140
160
180
200
220

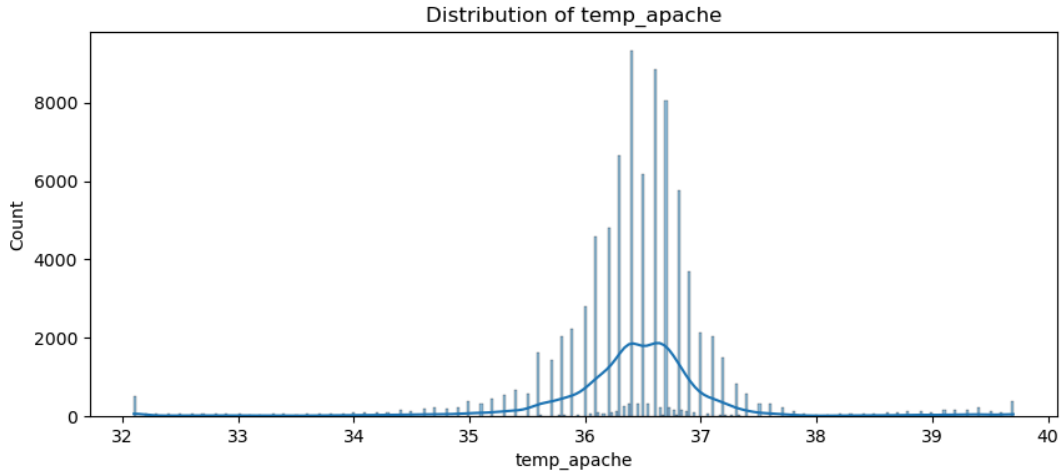

Distribution of h1_sysbp_min
Count
h1_sysbp_min
0
500
1000
1500
2000
2500
60
80
100
120
140
160
180
200

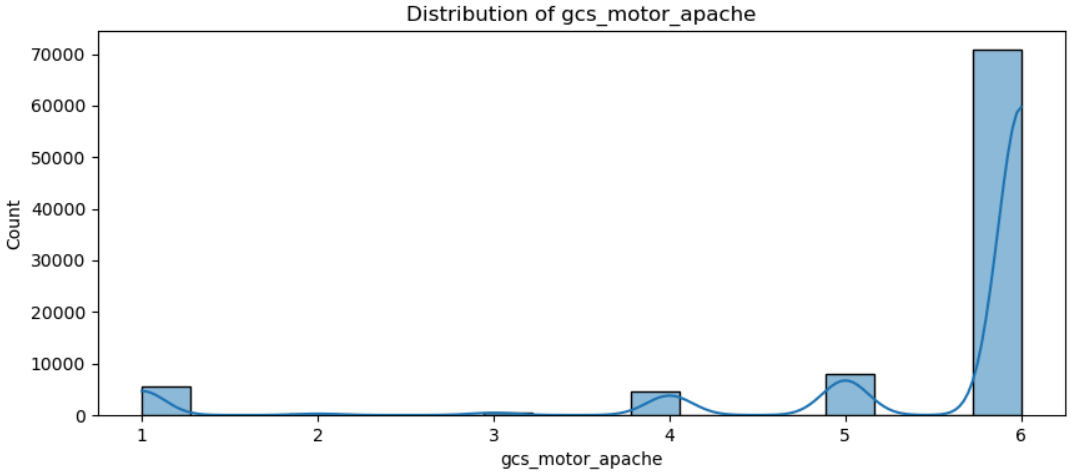

Distribution of height
Count
height
0
1000
2000
3000
4000
5000
6000
7000
140
150
160
170
180
190

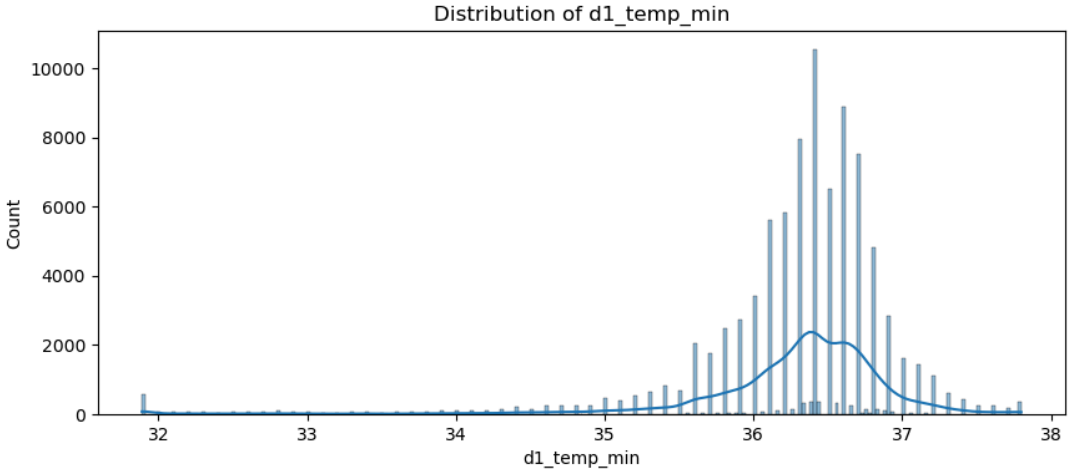

Distribution of map_apache
Count
map_apache
0
1000
2000
3000
4000
5000
6000
40
60
80
100
120
140
160
180
200

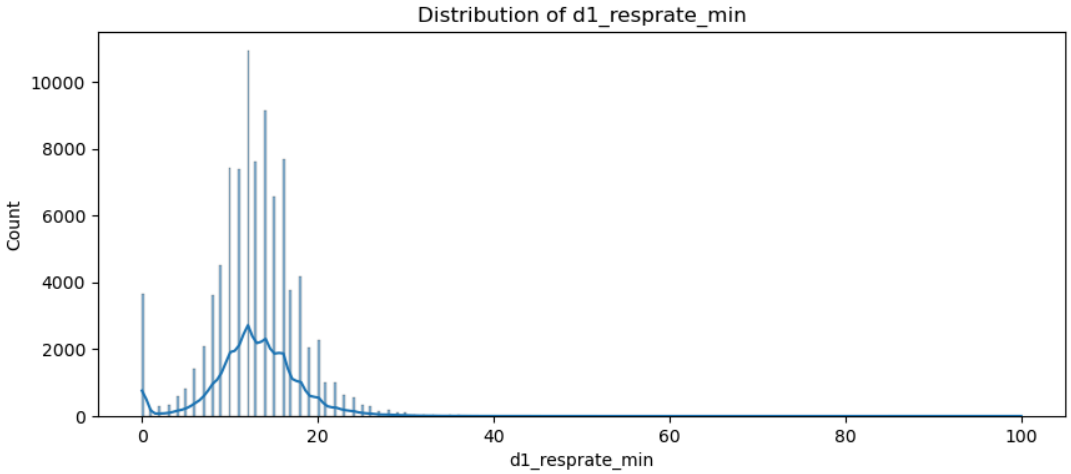

Distribution of pre_icu_los_days
Count
pre_icu_los_days
0
2000
4000
6000
8000
−25
0
25
50
75
100
125
150

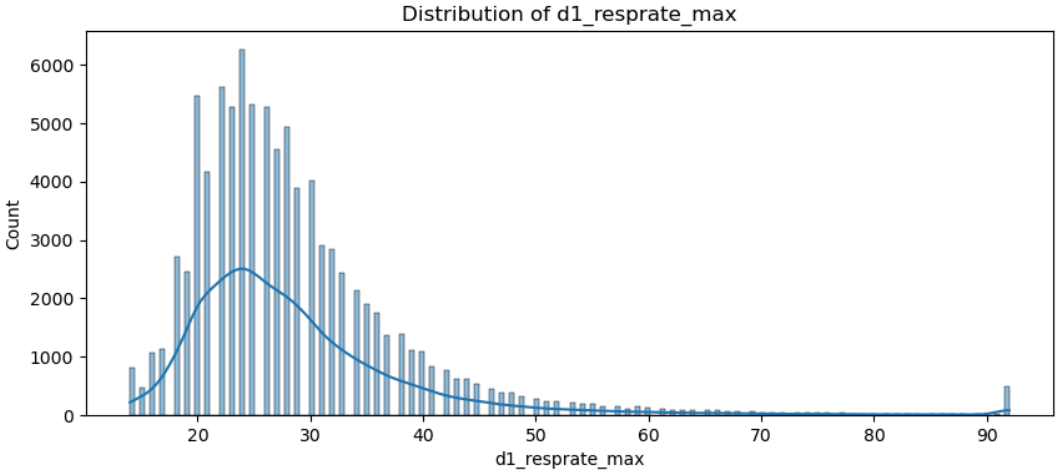

Distribution of resprate_apache
Count
resprate_apache
0
1000
2000
3000
4000
5000
10
20
30
40
50
60

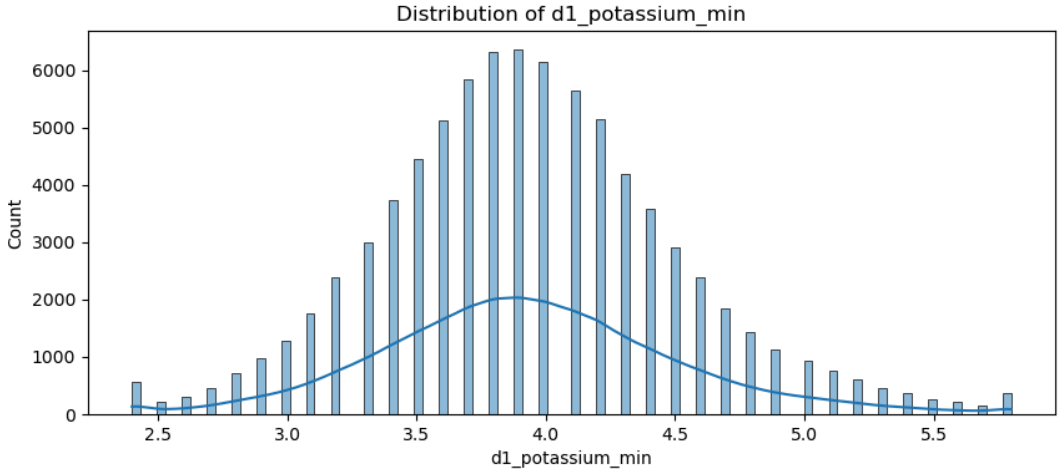

Distribution of temp_apache
Count
temp_apache
0
2000
4000
6000
8000
32
33
34
35
36
37
38
39
40

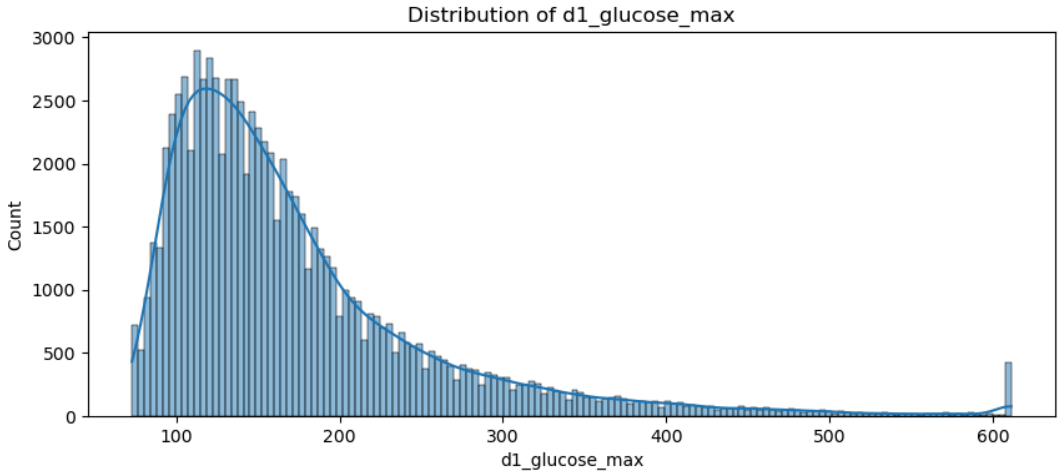

Distribution of gcs_motor_apache
Count
gcs_motor_apache
0
10000
20000
30000
40000
50000
60000
70000
1
2
3
4
5
6

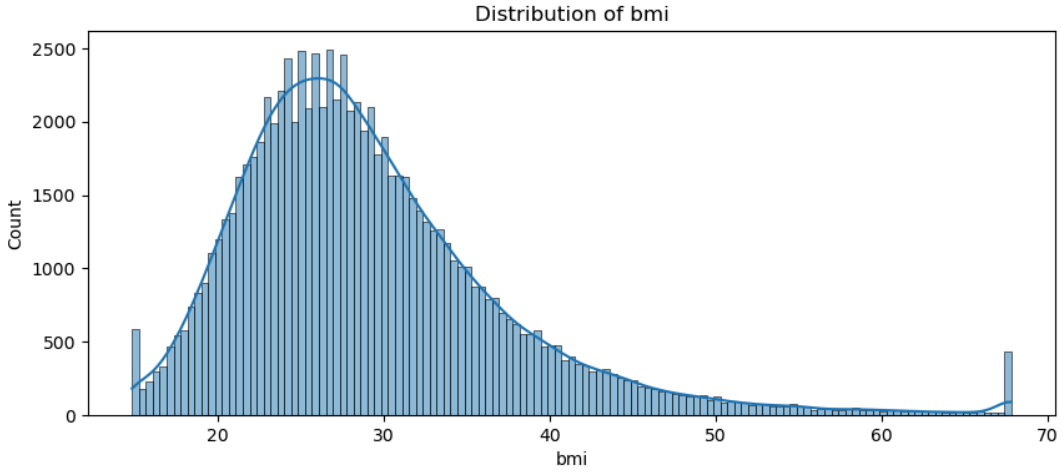

Distribution of d1_temp_min
Count
d1_temp_min
0
2000
4000
6000
8000
10000
32
33
34
35
36
37
38

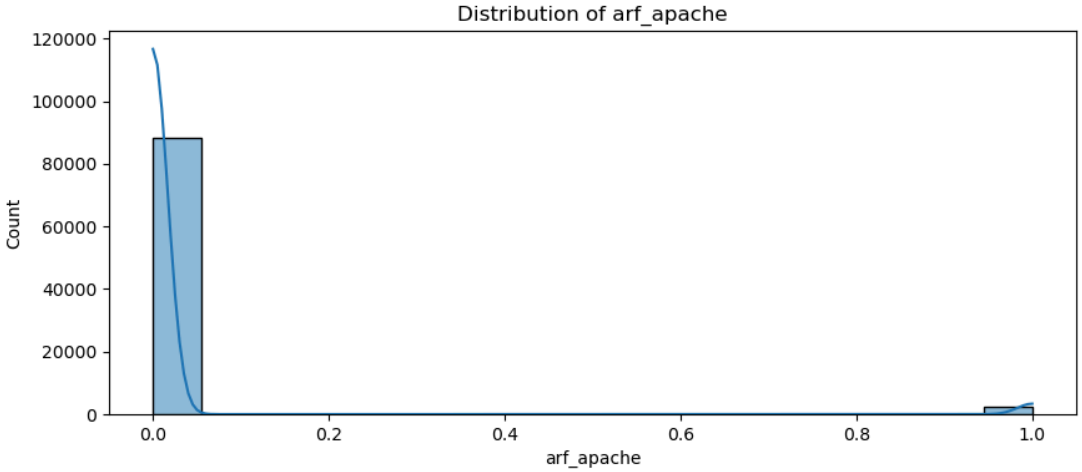

Distribution of d1_resprate_min
Count
d1_resprate_min
0
2000
4000
6000
8000
10000
0
20
40
60
80
100

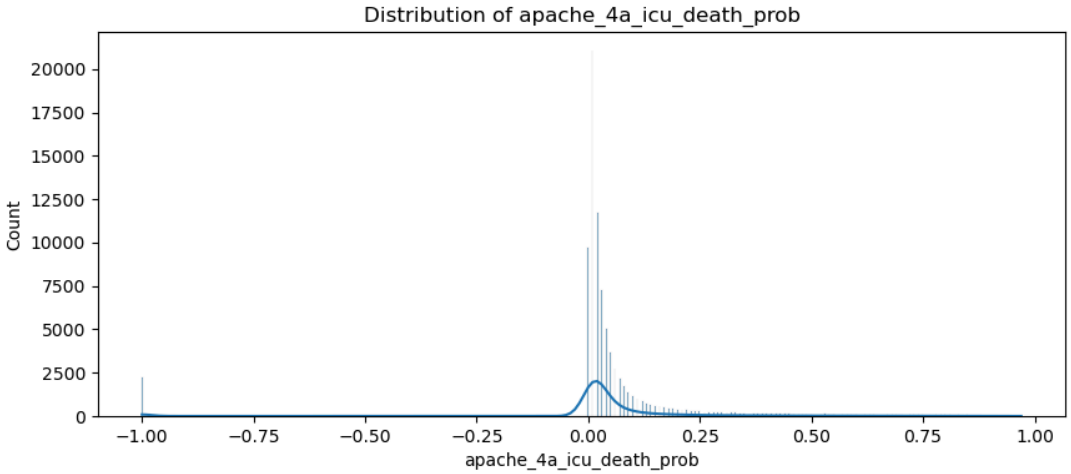

Distribution of d1_resprate_max
Count
d1_resprate_max
0
1000
2000
3000
4000
5000
6000
20
30
40
50
60
70
80
90


Distribution of d1_potassium_min
Count
d1_potassium_min
0
1000
2000
3000
4000
5000
6000
2.5
3.0
3.5
4.0
4.5
5.0
5.5

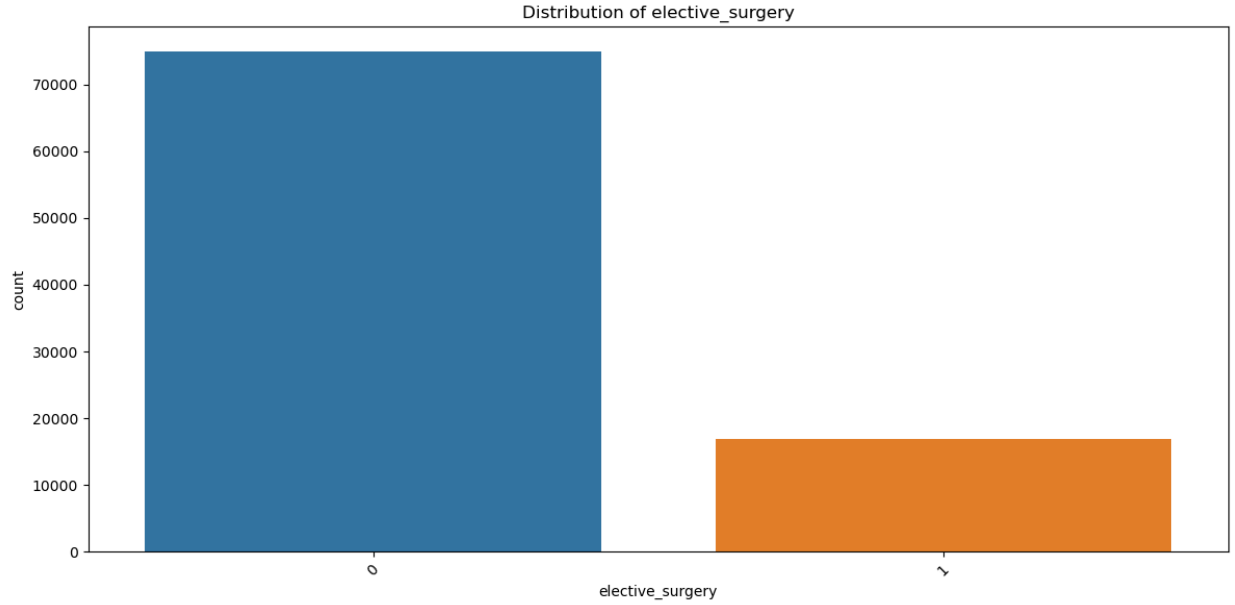

Distribution of d1_glucose_max
Count
d1_glucose_max
0
500
1000
1500
2000
2500
3000
100
200
300
400
500
600

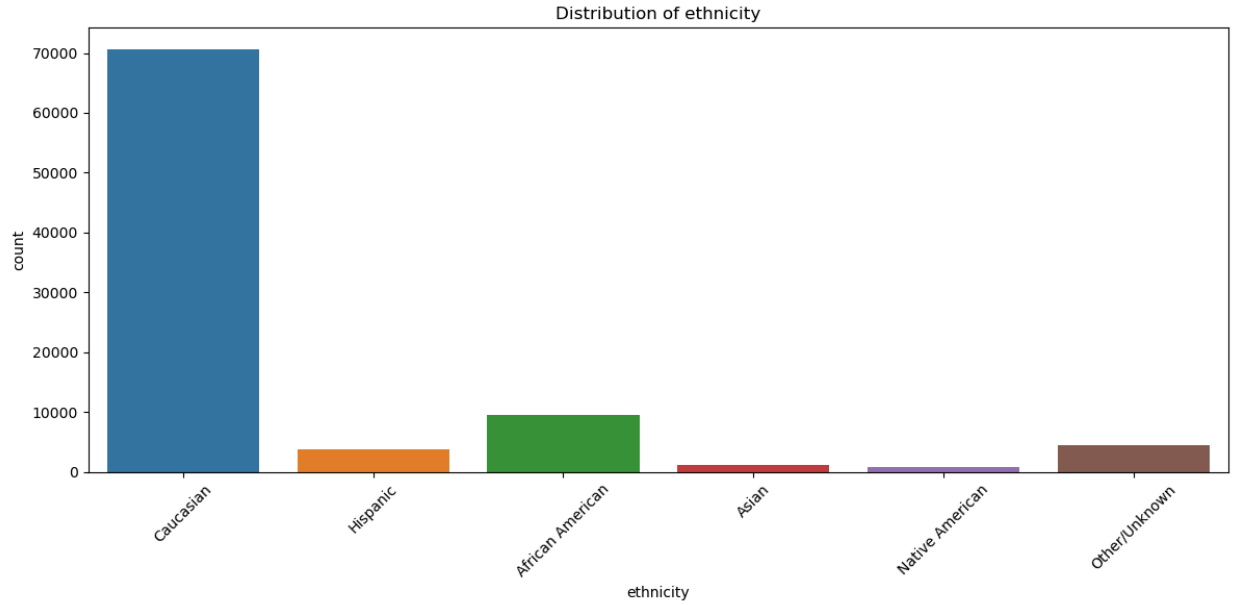

Distribution of bmi
Count
bmi
0
500
1000
1500
2000
2500
20
30
40
50
60
70

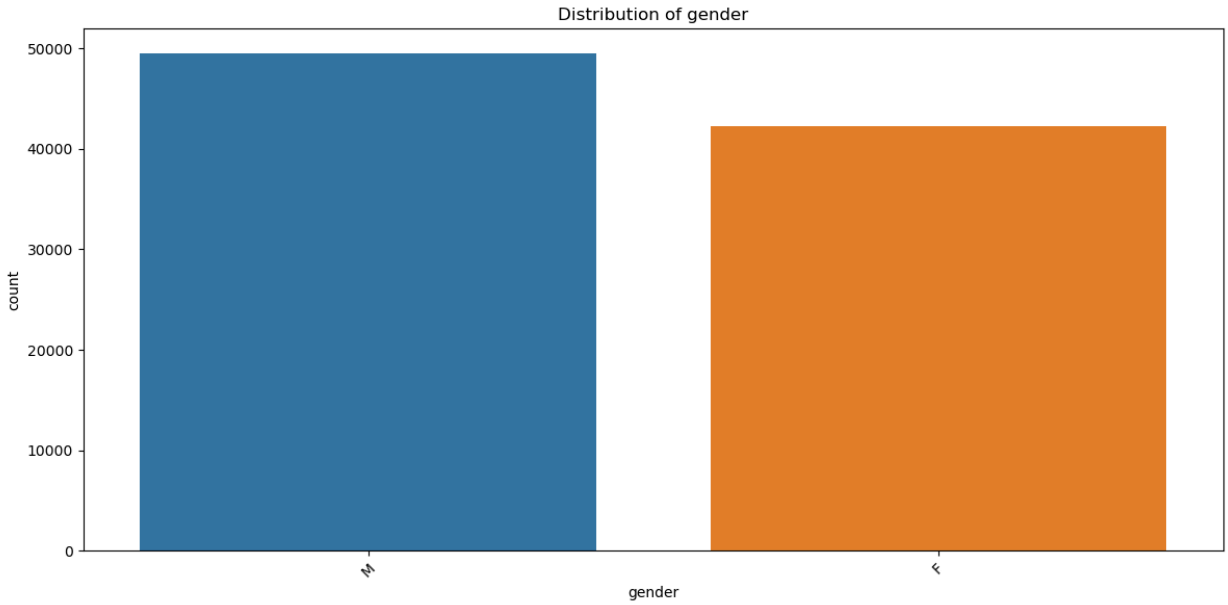

Distribution of arf_apache
Count
arf_apache
0
20000
40000
60000
80000
100000
120000
0.0
0.2
0.4
0.6
0.8
1.0

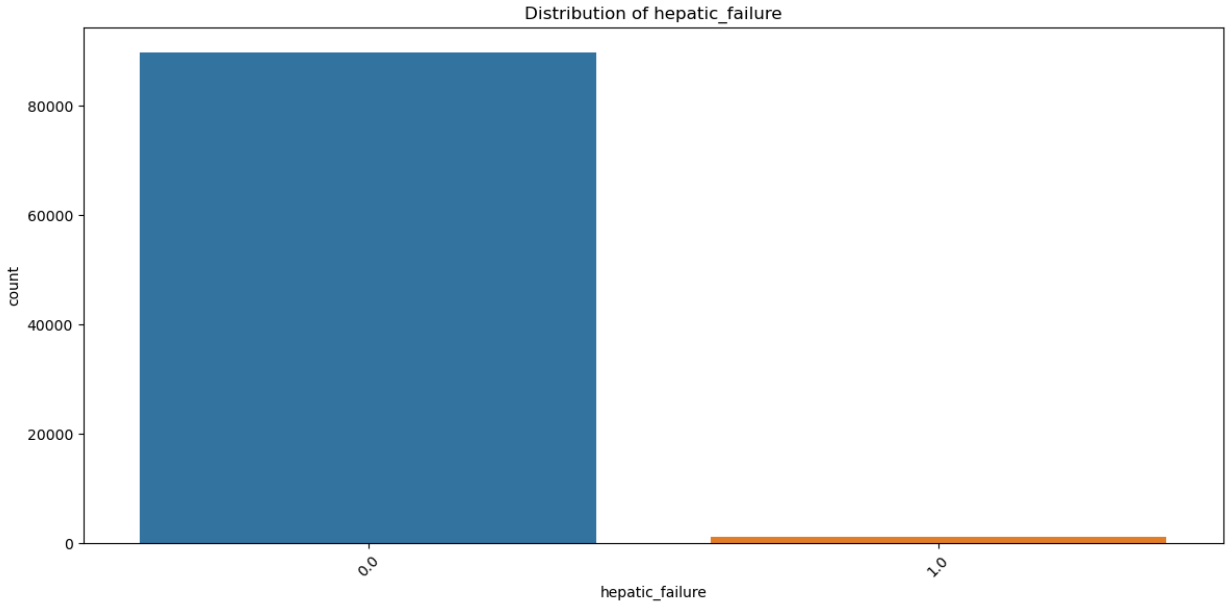

Distribution of apache_4a_icu_death_prob
Count
apache_4a_icu_death_prob
0
2500
5000
7500
10000
12500
15000
17500
20000
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

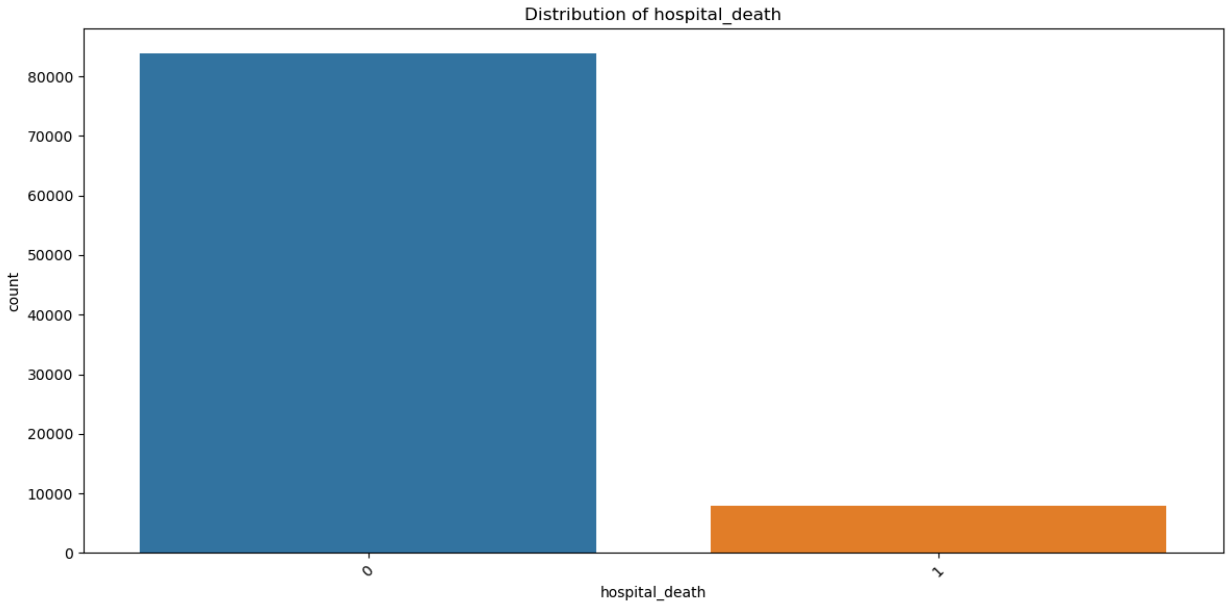
Categorical Feature Distributions

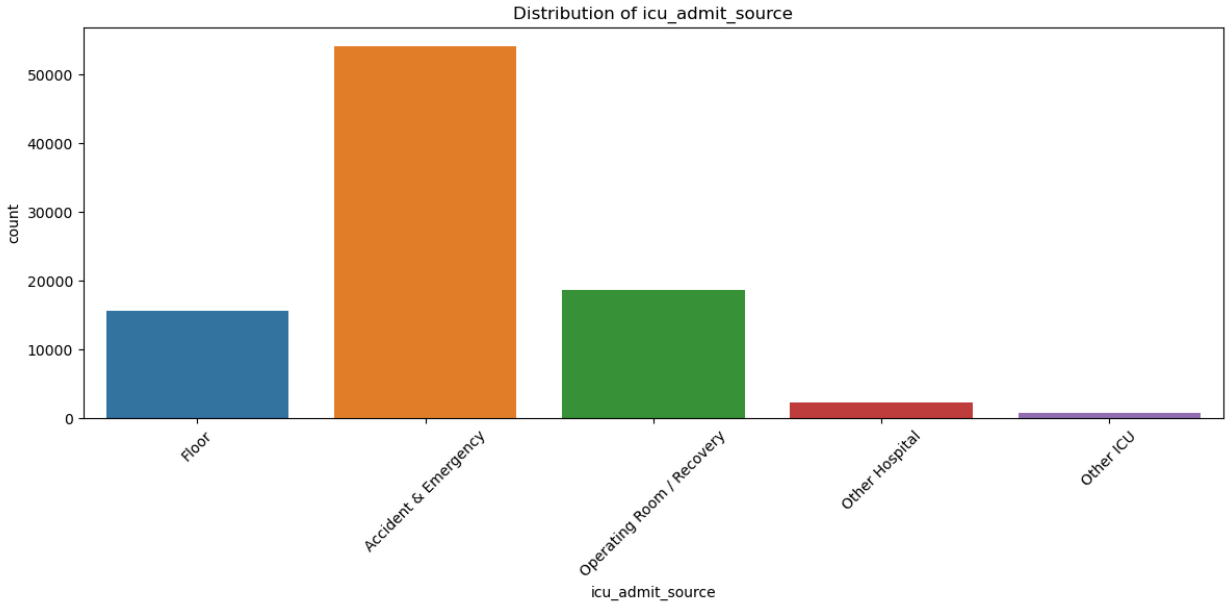


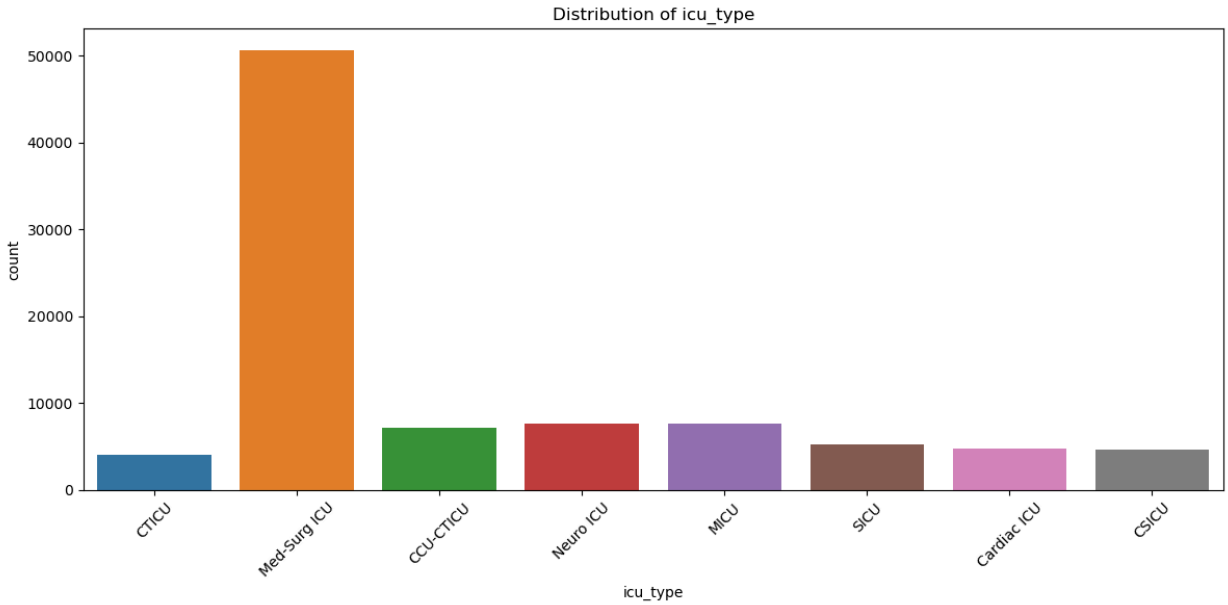


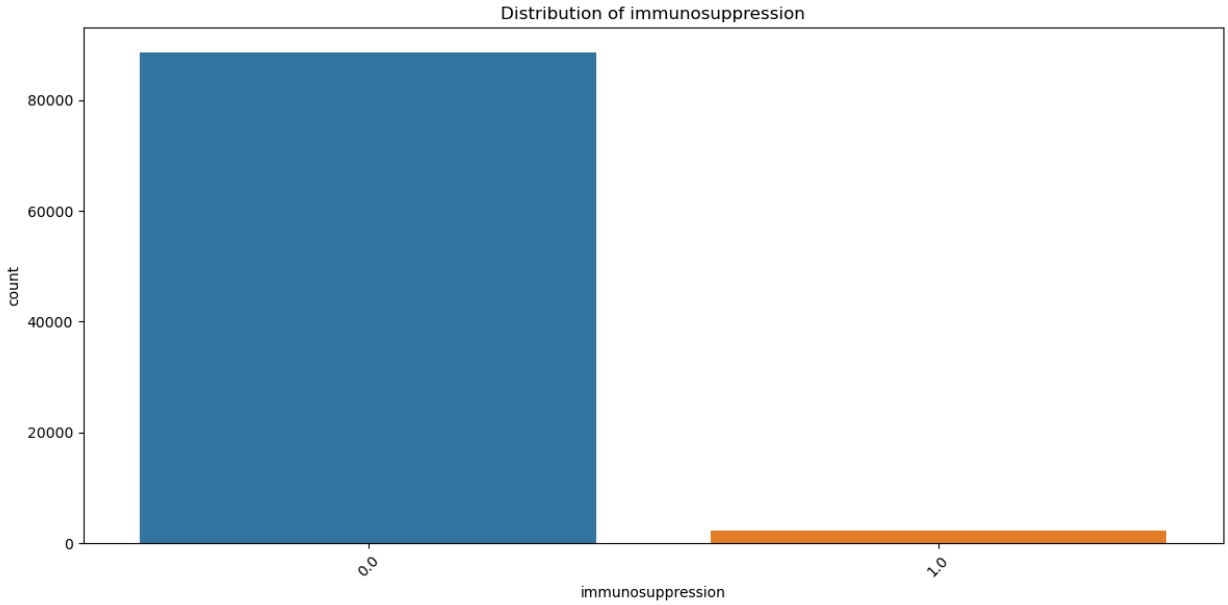

Distribution of gender
50000
40000
30000
20000
10000
0
count
M
F
gender

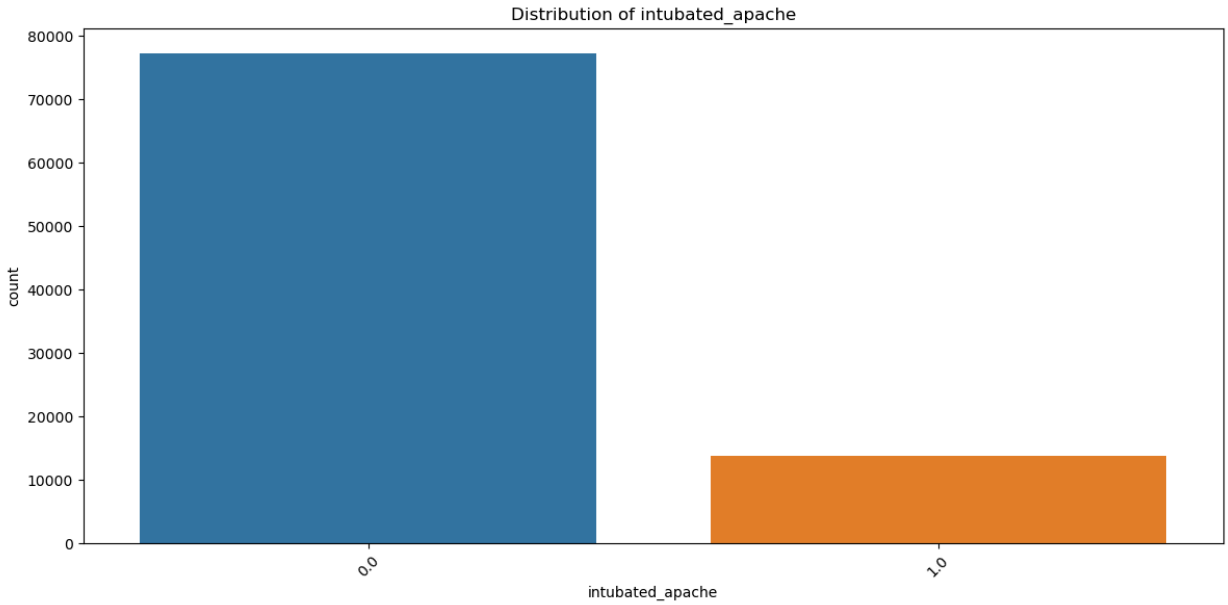

Distribution of hepatic_failure
80000
60000
40000
20000
0
count
0.0
1.0
hepatic_failure

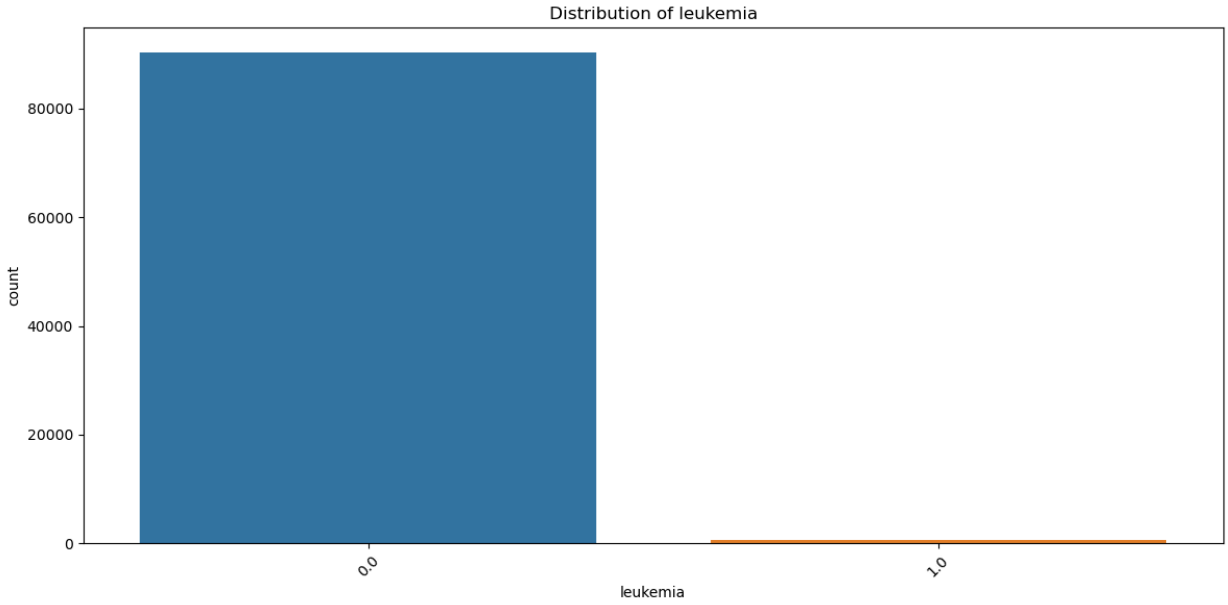

Distribution of hospital_death
count
80000
70000
60000
50000
40000
30000
20000
10000
0
0
1
hospital_death

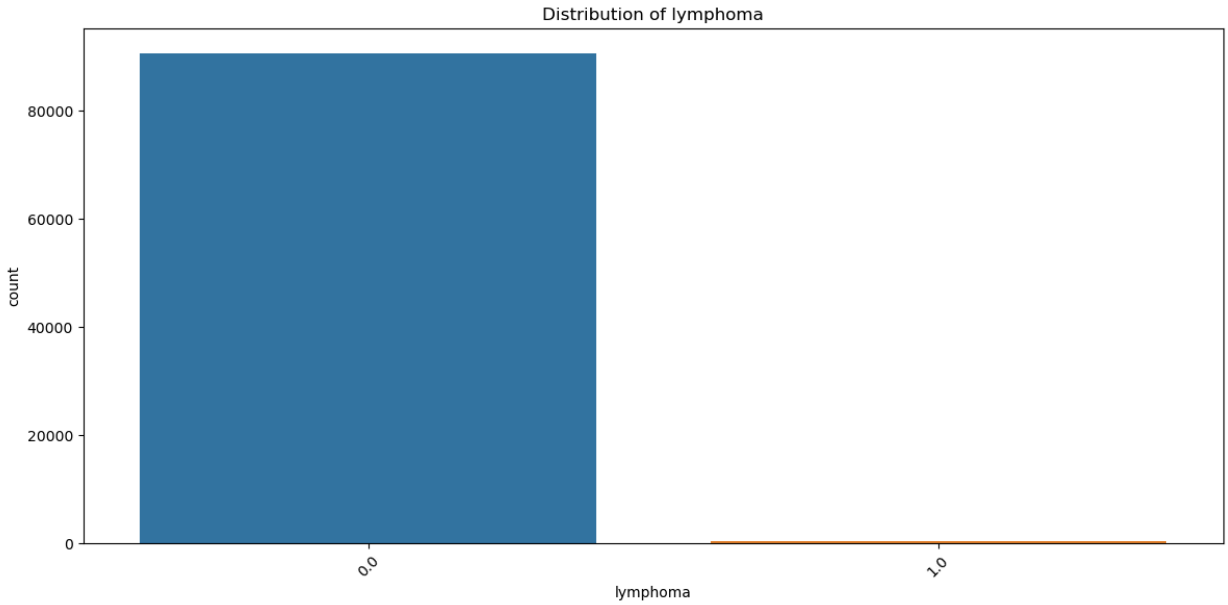

Distribution of icu_admit_source
count
50000
40000
30000
20000
10000
0
Floor
Accident & Emergency
Operating Room / Recovery
Other Hospital
Other ICU
icu_admit_source

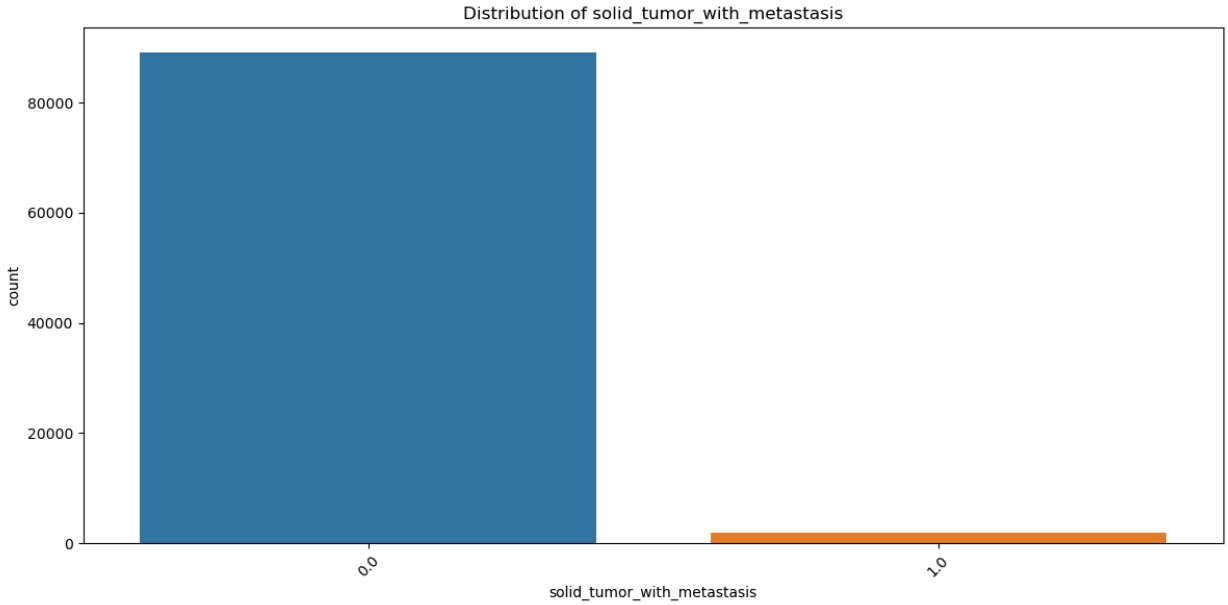

Distribution of icu_type
count
50000
40000
30000
20000
10000
0
CTICU
Med-Surg ICU
CCU-CTICU
Neuro ICU
MICU
SICU
Cardiac ICU
CSICU
icu_type

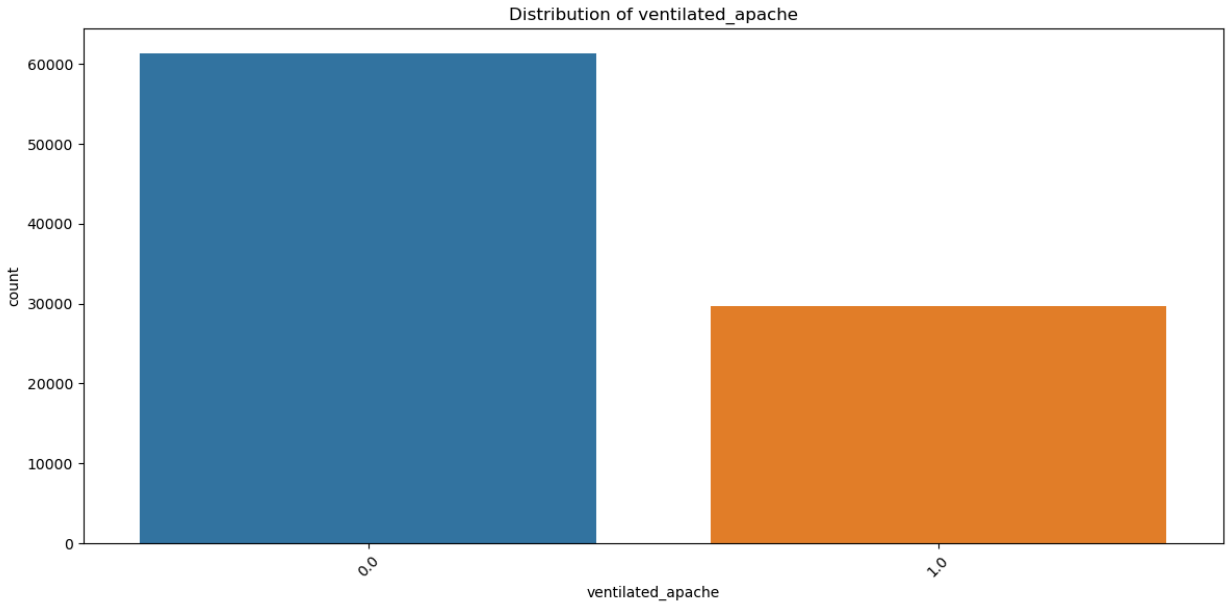

Distribution of immunosuppression
count
80000
60000
40000
20000
0
0.0
1.0
immunosuppression

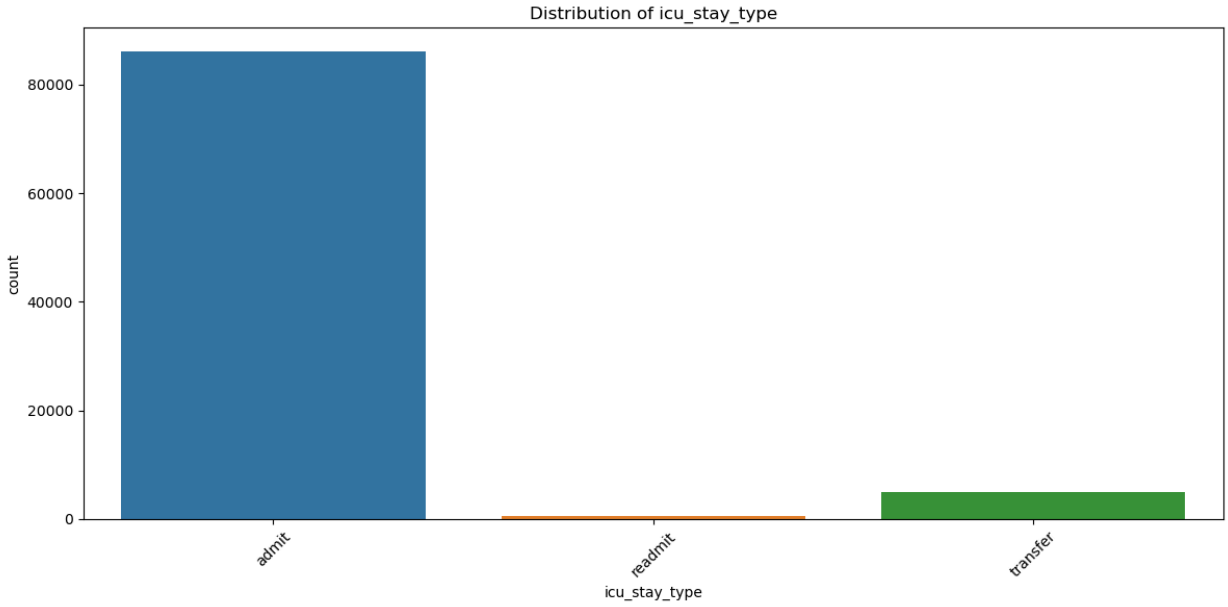

Distribution of intubated_apache
80000
70000
60000
50000
40000
30000
20000
10000
0
count
0.0
1.0
intubated_apache

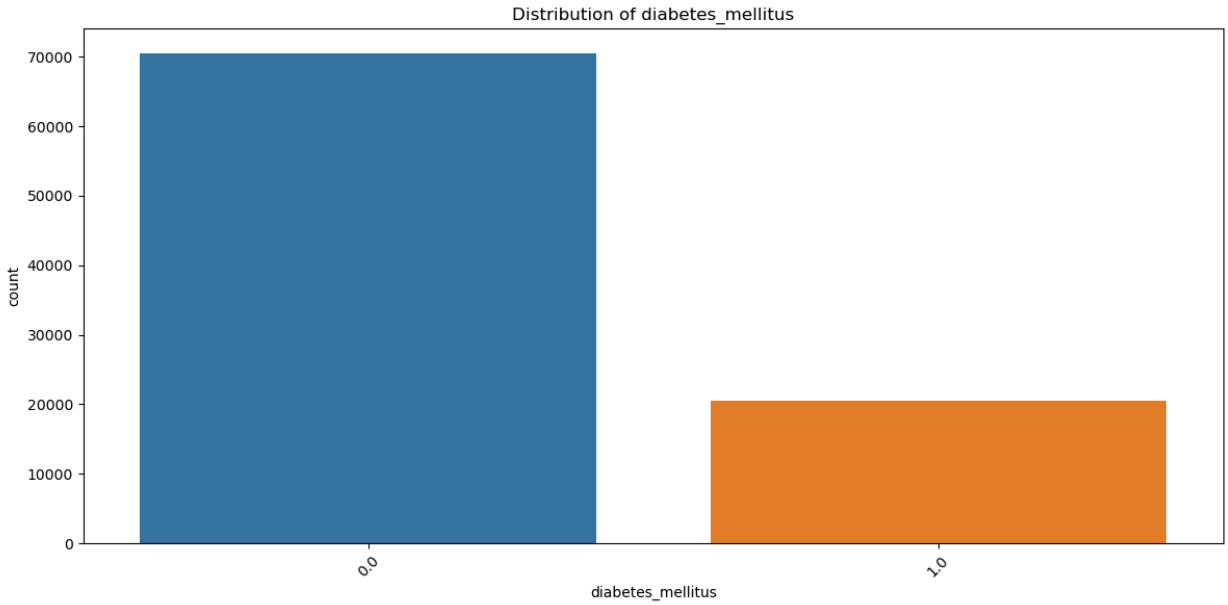

Distribution of leukemia
80000
60000
40000
20000
0
count
0.0
1.0
leukemia

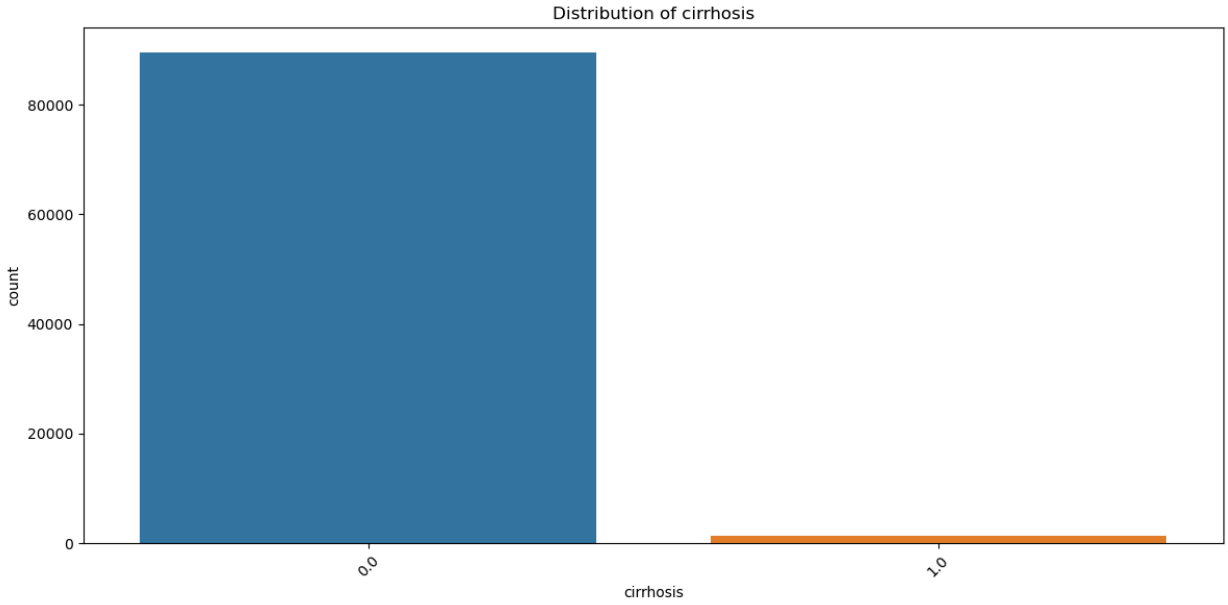

Distribution of lymphoma
count
80000
60000
40000
20000
0
0.0
1.0
lymphoma

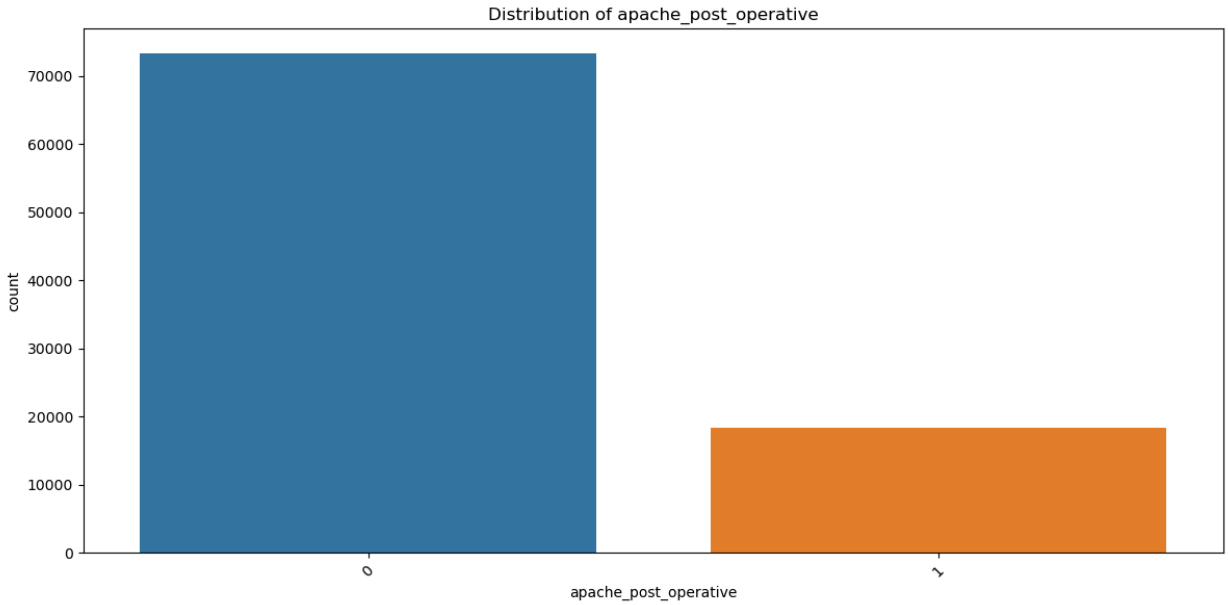

Distribution of solid_tumor_with_metastasis
count
80000
60000
40000
20000
0
0.0
1.0
solid_tumor_with_metastasis

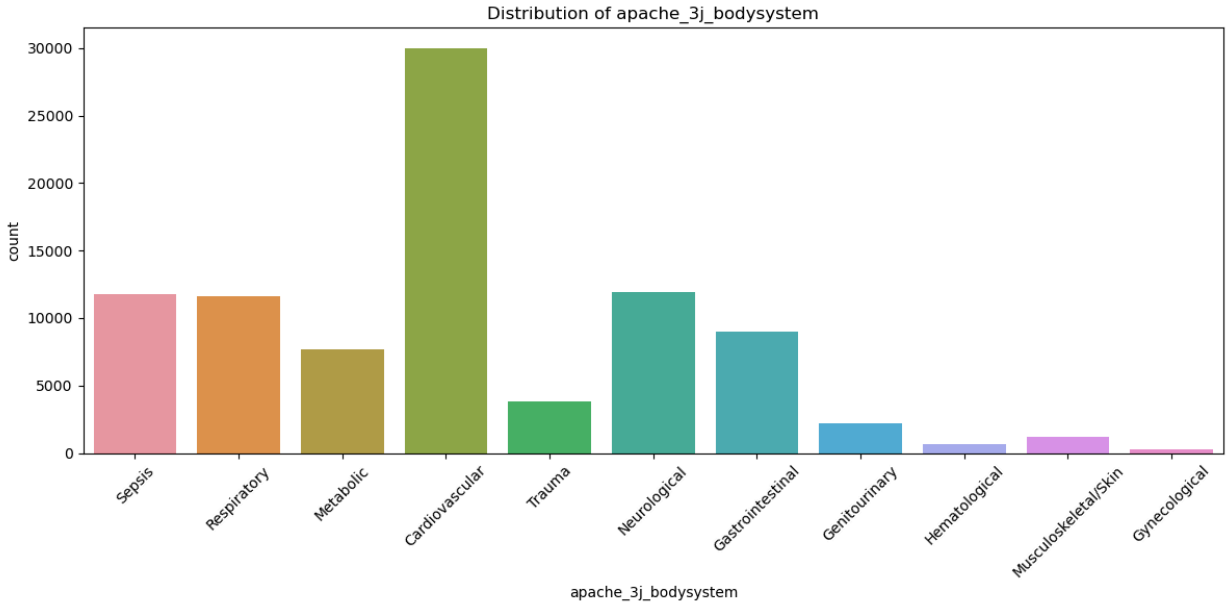

Distribution of ventilated_apache
count
60000
50000
40000
30000
20000
10000
0
0.0
1.0
ventilated_apache

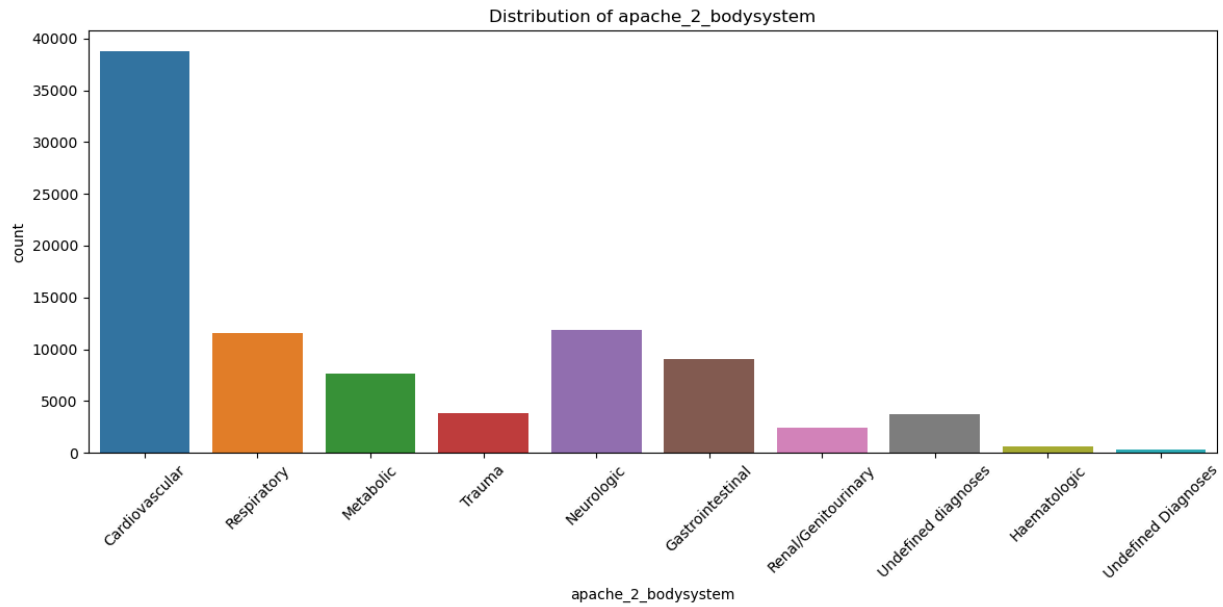

Distribution of icu_stay_type
count
80000
60000
40000
20000
0
admit
readmit
transfer
icu_stay_type


Distribution of diabetes_mellitus
70000
60000
50000
40000
30000
20000
10000
0
count
0.0
1.0
diabetes_mellitus

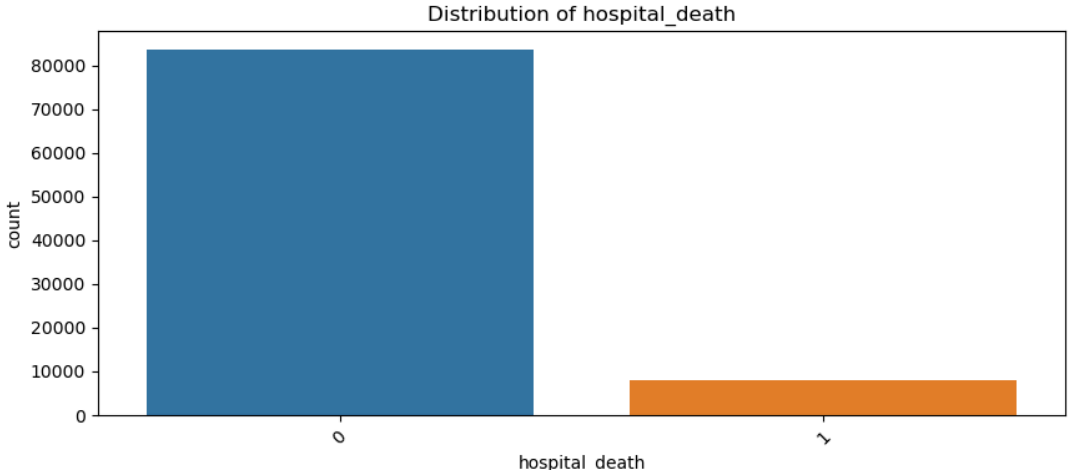

Distribution of cirrhosis
80000
60000
40000
20000
0
count
0.0
1.0
cirrhosis


Distribution of apache_post_operative
count
70000
60000
50000
40000
30000
20000
10000
0
0
1
apache_post_operative


Distribution of apache_3j_bodysystem
count
30000
25000
20000
15000
10000
5000
0
Sepsis
Respiratory
Metabolic
Cardiovascular
Trauma
Neurological
Gastrointestinal
Genitourinary
Hematological
Musculoskeletal/Skin
Gynecological
apache_3j_bodysystem



Target Variable Distribution



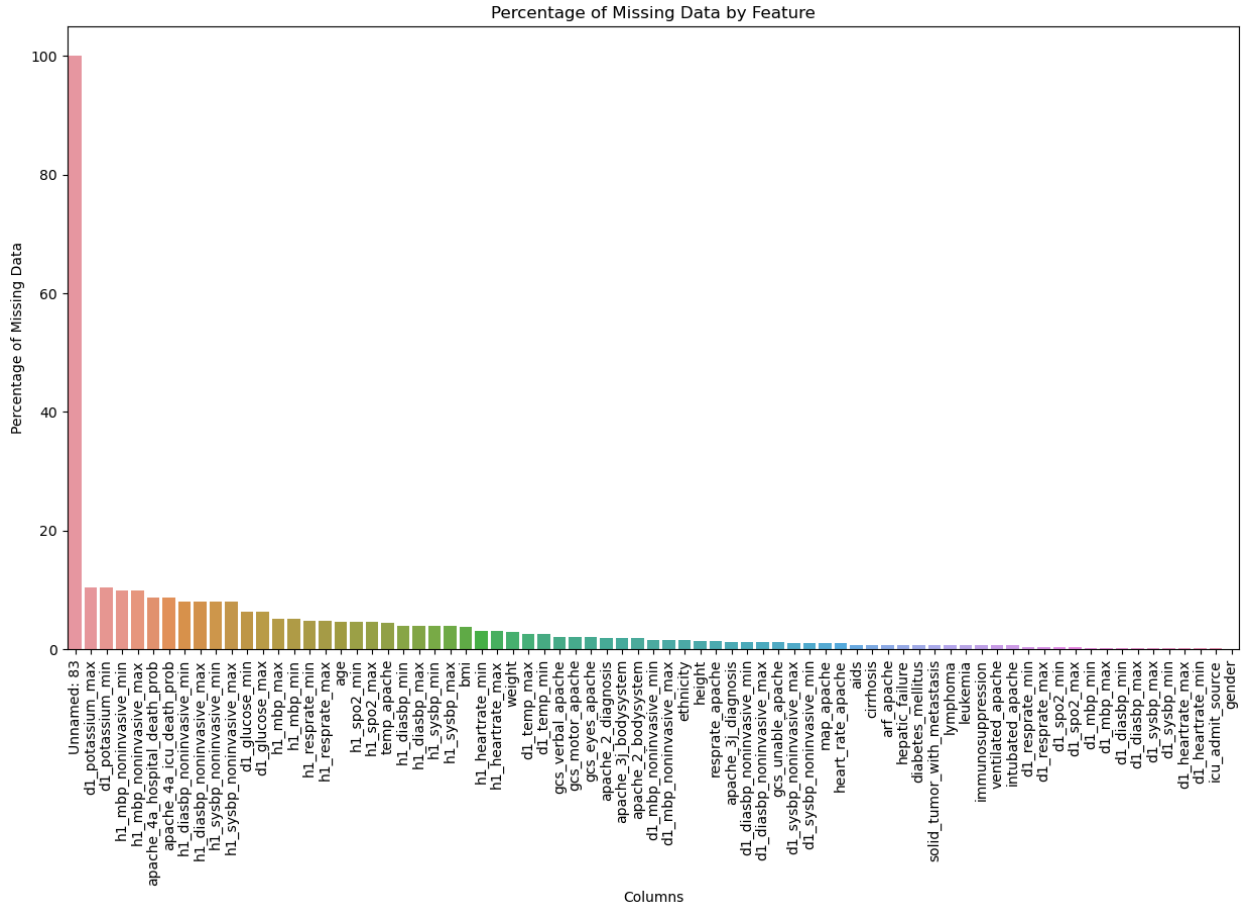
Data Exploratory Analysis & Cleaning

For basic EDA, I looked at metrics from across my dataframe using the describe() and info() accordingly to get a general sense of the data frame. From here I could analyze the following metrics:

- Count of elements
- Mean of each feature
- STD of each feature
- Quartile values (see notebook for specific values)

Missing Data

From here, I then evaluated the number of missing values across my dataset.



Specifically, I looked at the number and percentage of missing values across all my features to understand how much missingness exists in my dataset. From the image above, we can see that column “Unnamed: 83” is completely missing and was subsequently deleted from the dataset. The remaining features all appeared to have less than 11% of missing values across the board. Other columns, such as 'h1_mbp_noninvasive_min' and 'h1_mbp_noninvasive_max,' show a degree of missingness at approximately 9.9% hinting at potential complications in data collection or entry during the early often tumultuous hours of ICU admission.

We see the absence of data in clinically vital features like 'apache_4a_hospital_death_prob' and 'apache_4a_icu_death_prob,' which lack around 8.7% of their values. The missing data here may stem from the non-availability of certain diagnostics or a lag in reporting.

This pattern extends to biological measurements crucial for initial diagnostics in the ICU such as 'd1_glucose_max' and 'd1_potassium_min' where we observe missing data ranging from about 6.3% to 10.5%. Such gaps could be attributed to the immediate medical condition of patients on ICU admission dictating whether specific tests are conducted promptly or deferred.

The implications of these missing data are multifaceted. From a modeling perspective, if the absence of data is systemic it could introduce bias. For instance, models might undervalue the severity of cases if data from more critical conditions are disproportionately absent due to patients' inability to undergo certain tests. Statistically speaking, we need to identify the nature of the missing data whether it would be considered any of the following:

- **MCAR (Missing Completely at Random)**: Data is missing randomly without any relation to other data or the missing data itself.
- **MAR (Missing at Random)**: The missing data has a relationship with other observed data, but not with the missing data's own value.
- **MNAR (Missing Not at Random)**: The missingness is related to the actual value that's missing, suggesting a hidden pattern in the missing data points.

Understanding how data is missing has distinct repercussions on the analysis. MCAR missingness can often be ignored or imputed straightforwardly. In contrast, MAR missingness requires more nuanced imputation methods to account for the observed relationships. MNAR missingness, however, poses the most significant challenge as it can introduce substantial bias, necessitating complex imputation techniques or the reevaluation of the use of affected variables.

I performed statistical measurements to test MCAR, MAR, and MNAR with the following results:

- **Variables Likely MCAR:**
    - elective_surgery
    - icu_stay_type
    - icu_type
    - pre_icu_los_days
    - apache_post_operative
    - Unnamed: 83
- **Variables Potentially MAR or MNAR:**
    - age
    - bmi
    - ethnicity
    - gender
    - height
    - icu_admit_source
    - weight
    - apache_2_diagnosis
    - apache_3j_diagnosis
    - arf_apache
    - gcs_eyes_apache
    - gcs_motor_apache
    - gcs_unable_apache
    - gcs_verbal_apache
    - heart_rate_apache
    - intubated_apache
    - map_apache
    - resprate_apache
    - temp_apache
    - ventilated_apache
    - d1_diasbp_max
    - d1_diasbp_min

- d1_diasbp_noninvasive_max
- d1_diasbp_noninvasive_min
- d1_heartrate_max
- d1_heartrate_min
- d1_mbp_max
- d1_mbp_min
- d1_mbp_noninvasive_max
- d1_mbp_noninvasive_min
- d1_resprate_max
- d1_resprate_min
- d1_spo2_max
- d1_spo2_min
- d1_sysbp_max
- d1_sysbp_min
- d1_sysbp_noninvasive_max
- d1_sysbp_noninvasive_min
- d1_temp_max
- d1_temp_min
- h1_diasbp_max
- h1_diasbp_min
- h1_diasbp_noninvasive_max
- h1_diasbp_noninvasive_min
- h1_heartrate_max
- h1_heartrate_min
- h1_mbp_max
- h1_mbp_min
- h1_mbp_noninvasive_max
- h1_mbp_noninvasive_min
- h1_resprate_max
- h1_resprate_min
- h1_spo2_max
- h1_spo2_min
- h1_sysbp_max
- h1_sysbp_min
- h1_sysbp_noninvasive_max
- h1_sysbp_noninvasive_min
- d1_glucose_max
- d1_glucose_min
- d1_potassium_max
- d1_potassium_min
- apache_4a_hospital_death_prob
- apache_4a_icu_death_prob
- aids
- cirrhosis
- diabetes_mellitus
- hepatic_failure
- immunosuppression
- leukemia
- lymphoma
- solid_tumor_with_metastasis
- apache_3j_bodysystem

- apache_2_bodysystem

For the purposes of this project, given the relatively low amount of missing data combined with the test above, I chose my imputation strategies to leverage the **median** for my numerical features and the **mode** for my categorial features. Median values are preferred for numerical features as it preserves the data's central tendency without being affected by outliers. On the other hand, the mode is adopted for imputing categorical data in order to maintain the sample's statistical characteristics under the assumption that the data is MAR and to minimize disruption to the original distribution and inter-feature relationships.

Outliers

In addressing outliers within the dataset, a robust method involving **z-score** analysis was employed. By setting a z-score threshold of 2.5, the strategy aimed to identify and separate outlier data points that deviated significantly from the mean. Such points are those falling beyond the threshold when measured in terms of standard deviation units from the dataset's mean capturing the essence of what constitutes an extreme variation in the data. This method's flexibility allows for adjustment of the sensitivity of outlier detection. I then segregated from the dataset these outliers resulting in 1273 observations removed from the data set.

Data Correlations

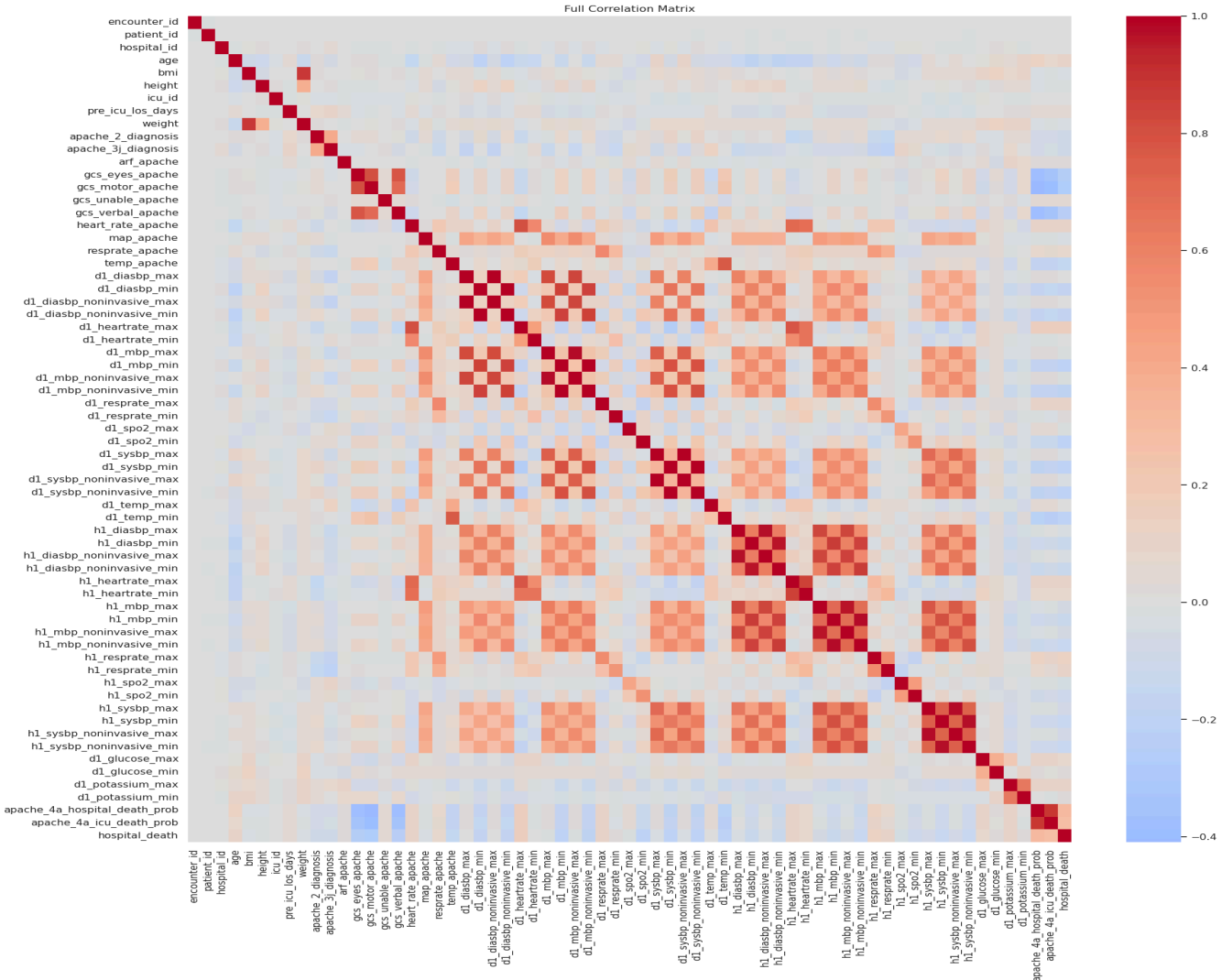
The dataset exhibits a number of significant correlations between various variables highlighting dependencies and redundancies in the data. These correlations can have implications for any analytical modeling such as multicollinearity issues or overfitting in machine learning models.

For instance, the strong correlation (greater than 0.8) between weight and bmi is expected as BMI is a derived metric involving weight and height, indicating that these features are not independent. Similarly, in clinical assessments such as the Glasgow Coma Scale (GCS), components like gcs_motor_apache and gcs_eyes_apache are closely related as they are both integral aspects of the neurological examination hence their high correlation.

Blood pressure readings, such as d1_diasbp_noninvasive_max and d1_diasbp_max, show redundancy since they represent the same physiological measurement but perhaps captured through different methods (invasive vs. non-invasive). This redundancy is mirrored across other pairs of invasive and non-invasive measurements throughout the first and subsequent hours in ICU, e.g., h1_mbp_max and h1_diasbp_noninvasive_max. The repeated high correlations among these pairs validate the consistency of measurement methods but also suggest that one of each pair might suffice for most analytical purposes to reduce dimensionality without losing critical information.

Furthermore, the high correlation between apache_4a_icu_death_prob and apache_4a_hospital_death_prob suggests that these predictive metrics, though perhaps calculated using slightly different parameters or models, often yield similar prognostic information regarding patient outcomes. This can imply that in predictive modeling using both might not provide additional benefit and could lead to unnecessarily complex models.

Below we calculate the correlation matrix then we identify highly correlated variables with a threshold greater than 0.8 and subsequently remove one of the pairs of variables from consideration in the modeling process.



Features with high correlations

First Variable	Second Variable	Correlation Coefficient
weight	bmi	> 0.8
gcs_motor_apache	gcs_eyes_apache	> 0.8
d1_diasbp_noninvasive_max	d1_diasbp_max	> 0.8

d1_diasbp_noninvasive_min	d1_diasbp_min	> 0.8
d1_heartrate_max	heart_rate_apache	> 0.8
d1_mbp_max	d1_diasbp_max	> 0.8
d1_mbp_max	d1_diasbp_noninvasive_max	> 0.8
d1_mbp_min	d1_diasbp_min	> 0.8
d1_mbp_min	d1_diasbp_noninvasive_min	> 0.8
d1_mbp_noninvasive_max	d1_diasbp_max	> 0.8
d1_mbp_noninvasive_max	d1_diasbp_noninvasive_max	> 0.8
d1_mbp_noninvasive_max	d1_mbp_max	> 0.8
d1_mbp_noninvasive_min	d1_diasbp_min	> 0.8
d1_mbp_noninvasive_min	d1_diasbp_noninvasive_min	> 0.8
d1_mbp_noninvasive_min	d1_mbp_min	> 0.8
d1_sysbp_noninvasive_max	d1_sysbp_max	> 0.8
d1_sysbp_noninvasive_min	d1_sysbp_min	> 0.8
h1_diasbp_noninvasive_max	h1_diasbp_max	> 0.8
h1_diasbp_noninvasive_min	h1_diasbp_min	> 0.8
h1_heartrate_min	h1_heartrate_max	> 0.8
h1_mbp_max	h1_diasbp_max	> 0.8

h1_mbp_max	h1_diasbp_noninvasive_max	> 0.8
h1_mbp_min	h1_diasbp_min	> 0.8
h1_mbp_min	h1_diasbp_noninvasive_min	> 0.8
h1_mbp_noninvasive_max	h1_diasbp_max	> 0.8
h1_mbp_noninvasive_max	h1_diasbp_noninvasive_max	> 0.8
h1_mbp_noninvasive_max	h1_mbp_max	> 0.8
h1_mbp_noninvasive_min	h1_diasbp_min	> 0.8
h1_mbp_noninvasive_min	h1_diasbp_noninvasive_min	> 0.8
h1_mbp_noninvasive_min	h1_mbp_min	> 0.8
h1_sysbp_min	h1_mbp_min	> 0.8
h1_sysbp_noninvasive_max	h1_sysbp_max	> 0.8
h1_sysbp_noninvasive_min	h1_mbp_noninvasive_min	> 0.8
h1_sysbp_noninvasive_min	h1_sysbp_min	> 0.8
apache_4a_icu_death_prob	apache_4a_hospital_death_prob	> 0.8

As a result, the following variables were dropped from the modeling process:

List of variables to drop due to high correlation

- 'weight'
- 'gcs_eyes_apache'
- 'd1_diasbp_noninvasive_max'
- 'd1_diasbp_noninvasive_min'
- 'heart_rate_apache'

- 'd1_diasbp_max'
- 'd1_mbp_min'
- 'd1_diasbp_min'
- 'd1_mbp_max'
- 'd1_sysbp_noninvasive_max'
- 'd1_sysbp_noninvasive_min'
- 'h1_diasbp_noninvasive_max'
- 'h1_diasbp_noninvasive_min'
- 'h1_heartrate_max'
- 'h1_diasbp_max'
- 'h1_mbp_min'
- 'h1_diasbp_min'
- 'h1_mbp_max'
- 'h1_sysbp_noninvasive_max'
- 'h1_mbp_noninvasive_min'
- 'h1_sysbp_noninvasive_min'
- 'apache_4a_hospital_death_prob'

Model Training & Results

After removing highly correlated features, implementing our imputation strategy, and removing outliers, I moved on to the model training phase. Here the data was divided into training and testing sets at a 80%/20% split respectively allowing for validation on unseen data to assess model generalizability. I then preprocessed the data to standardize numerical features via a StandardScaler and one-hot encoded categorical features in order to optimize the data for training. Scaling ensured that any features with larger magnitudes wouldn't unduly bias the modeling phase and one-hot encoding allows categorical variables to be considered in the training process.

Dimensionality reduction via UMAP(Uniform Manifold Approximation and Projection) was leveraged to enhance training efficiency and potentially boost model performance by focusing on the most informative features. UMAP is valued for its ability to preserve both the local and global structure of the data in lower dimensional space making it suitable for reducing the dimensionality of the data preserving critical data relationships and structure.

A variety of models including logistic regression, random forest, SVM, XGBoost, and a neural network were then trained with their hyperparameters optimized via GridSearchCV. This diverse approach helps identify the most effective model based on cross-validation accuracy.

Specifically, here were the model parameters explored where we include regularization parameters as show below:

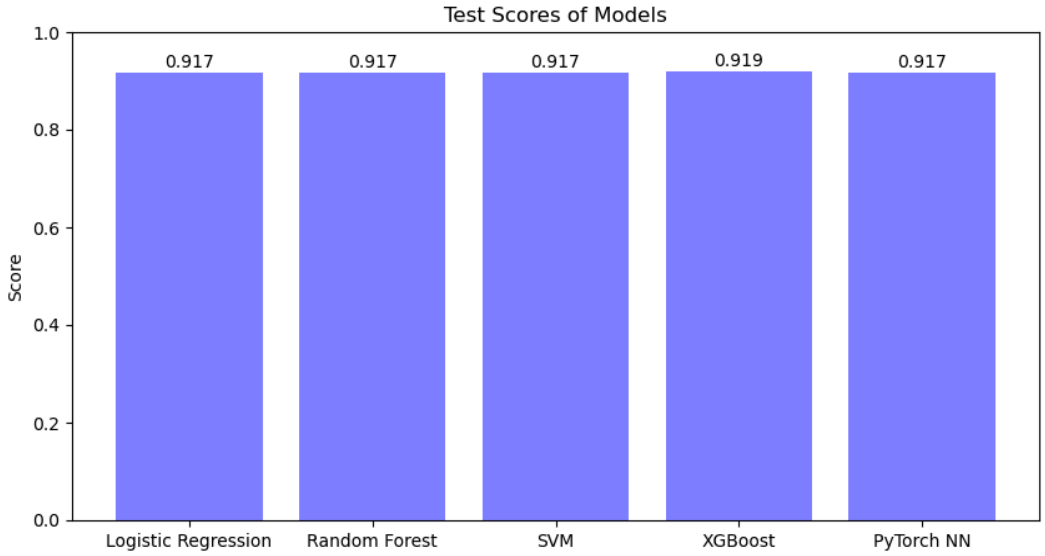
Model	Model Object	Parameters
Logistic Regression	`LogisticRegression(max_iter=2000)`	'C': [0.01, 0.1, 1, 10, 100]
Random Forest	`RandomForestClassifier()`	'n_estimators': [50, 100, 200, 400], 'max_depth': [10, 15, 20, 50]
Linear SVC (Experiment 2)	`LinearSVC()`	'C': [0.1, 1, 10, 100]
XGBoost	`XGBClassifier(tree_method='hist', device='cuda')`	'n_estimators': [50, 100, 200, 500], 'max_depth': [3, 6, 9, 15, 30], 'learning_rate': [0.1, 0.01, 0.001]
PyTorch NN	Neural Network GridSearch	'lr': [0.1, 0.01, 0.001], 'max_epochs': [10, 20, 30, 50, 100], 'batch_size': [64, 128], 'module**num_units': [100, 200], 'module**activation': [ReLU(), Tanh()]
SVM (Experiment 1)	`SVC(kernel='linear', cache_size=2000),`	'C': [0.1, 1, 10, 100] 'kernel': ['linear', 'poly', 'rbf', 'sigmoid']

Lastly, I want to mention that in this phase I ran two experiments. The first focused on modeling on the dataset without dimensionality reduction and without addressing the class imbalance and the second experiment focused on dimensionality reduction and utilizing a strategy to deal with class imbalance.

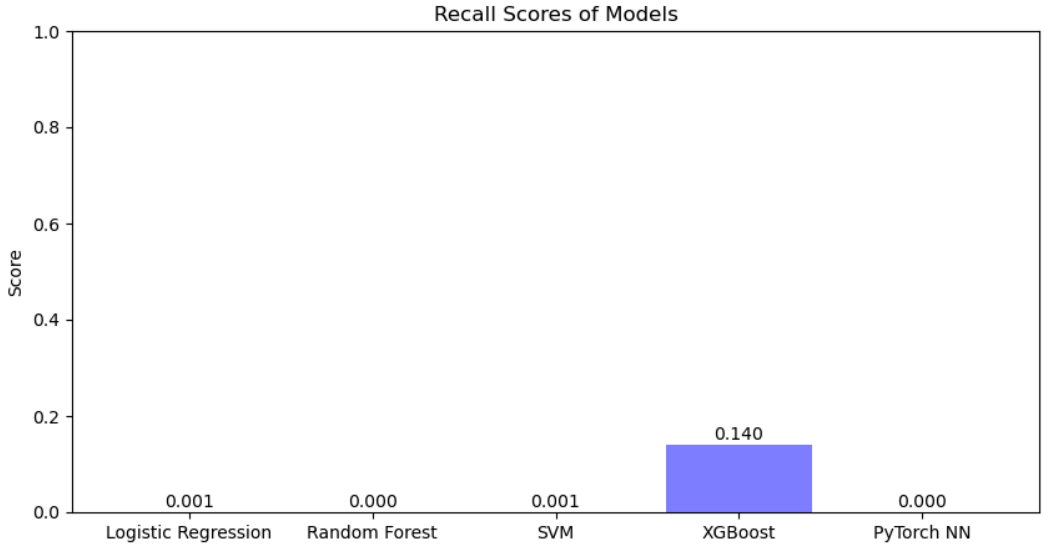
Results Experiment 1

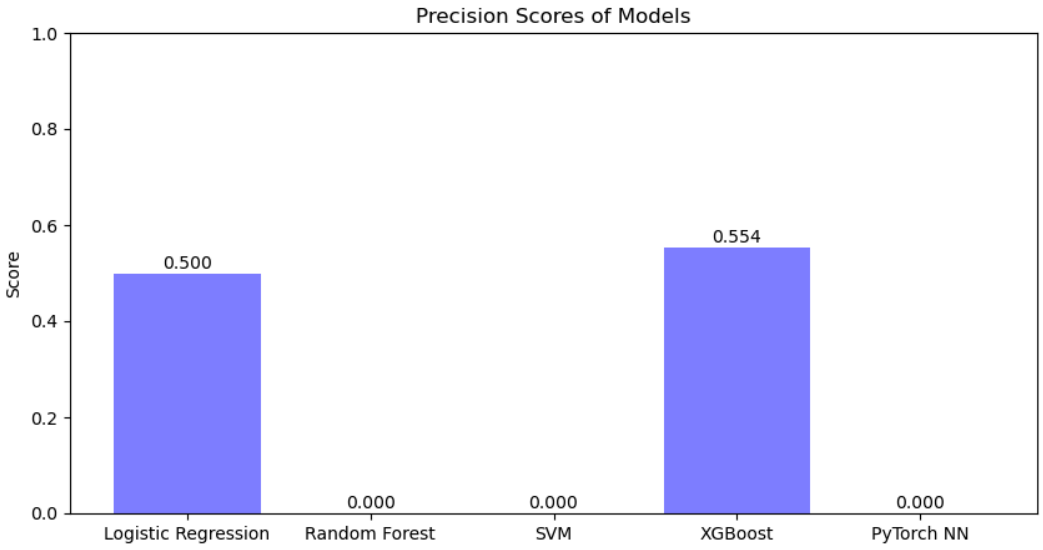
Below are the results of **cross-validation score (**cross validation was leveraged to determine the best set of hyperparameters for each model and used to iteratively improve the overall model performance) and performance on the test set accordingly along with what best parameters were found.

Model	Best CV Score	Best Parameters	Training Time (seconds)	Test Score
Logistic Regression	0.915676139	C: 0.01	0.23	0.916961551
Random Forest	0.915676139	max_depth: 10, n_estimators: 50	15.08	0.916961551
SVM	0.915676139	C: 0.1, kernel: linear	3219.47	0.916961551
XGBoost	0.920665633	learning_rate: 0.01, max_depth: 9, n_estimators: 500	286.61	0.919228218
PyTorch NN	0.916533069	batch_size: 64, lr: 0.001, max_epochs: 100, module activation**: Tanh(), module_**num_units: 200	18998.16	0.916961521



What we see here is exceptional performance across all five model times in terms of accuracy. However, digging deeper into the numbers looking at metrics like precision and recall we find:



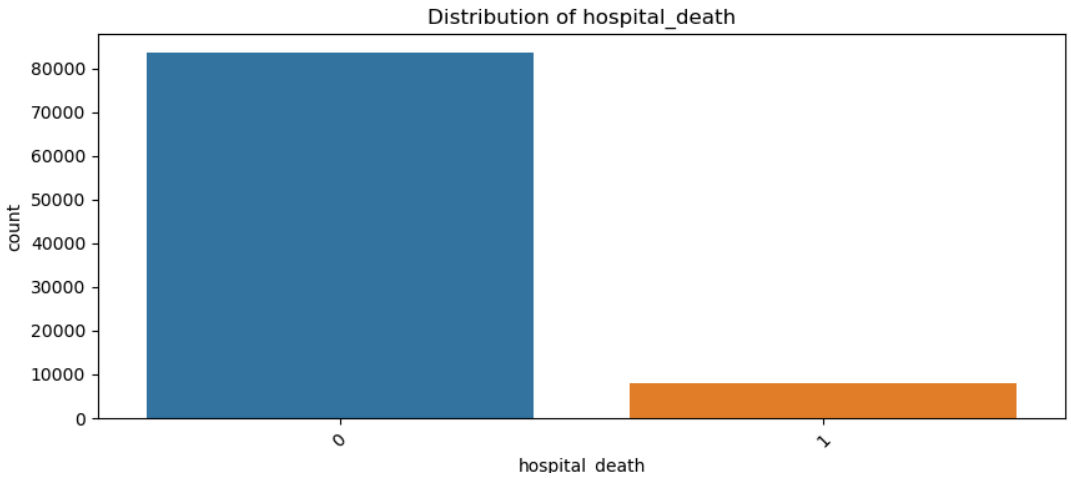


Precision measures the accuracy of the positive predictions made by the model. It is the ratio of true positive predictions to the total number of positive predictions including false positives. High precision indicates a low rate of false positives.

Recall measures the model's ability to identify all relevant instances. It is the ratio of true positives to the total number of actual positive instances including false negatives. High recall indicates that the model is good at catching positive cases.

These metrics were chosen to evaluate model performance as they reflect the necessity that in health systems the cost of false positives (receiving treatment you don't need) and false negatives (not receiving treatment needed) are very high.

We see that the precision and recall performance of these models are abysmal. Why is that? It turns out it has to do with significant class imbalance!



We see that our target variable has substantially more negative class examples than positive ones. Specifically: 83798 examples of the negative class and 7915 of the positive class. Given this imbalance, the models learned simply to predict the Negative class (in general) achieving substantially high overall accuracy rates but terrible precision and recall metrics making this model useless in a real-world setting.

Feature Importance

The following table includes the most significant features used by the XGBoost model to predict outcomes highlighting their respective importance scores. Features not impacting the model's decisions have been consolidated under "Others" with a score of 0.0 for clarity.

Feature	Importance (XGBoost)
icu_admit_source	0.23056102
d1_heartrate_min	0.052775655
ventilated_apache	0.043882713
icu_type	0.03969492
intubated_apache	0.036849055
d1_spo2_min	0.03259673
map_apache	0.03242066
gender	0.03130003
gcs_unable_apache	0.031074235
d1_mbp_noninvasive_min	0.031013478
age	0.029589973
height	0.028464723

gcs_motor_apache	0.027766768
resprate_apache	0.027247055
d1_mbp_noninvasive_max	0.025811188
hepatic_failure	0.022743242
ethnicity	0.021324947
apache_3j_diagnosis	0.020061536
h1_resprate_min	0.019146834
gcs_verbal_apache	0.018818064
temp_apache	0.016734354
d1_heartrate_max	0.015618164
arf_apache	0.012123373
apache_post_operative	0.010803052
bmi	0.009848279
h1_resprate_max	0.008283185
d1_sysbp_min	0.0064551695
d1_temp_min	0.0059397584
h1_mbp_noninvasive_max	0.0059227673
h1_heartrate_min	0.0048030587

d1_resprate_max	0.0035660246
d1_temp_max	0.002627758
Others	0.0

And here's a comparison table of feature importances for Logistic Regression:

Feature	Importance (Logistic Regression)
height	1.2938159
hepatic_failure	0.7874897
cirrhosis	0.68876195
age	0.43689704
apache_3j_bodysystem	0.4163586
apache_4a_icu_death_prob	0.3743671
icu_type	0.26149592
gcs_motor_apache	0.25389513
icu_admit_source	0.24611883
aids	0.24441238
leukemia	0.2359267

intubated_apache	0.23194344
d1_potassium_max	0.23164569
lymphoma	0.22629978
ventilated_apache	0.22437984
d1_mbp_noninvasive_min	0.19959792
h1_spo2_max	0.16629721
h1_spo2_min	0.16257232
d1_potassium_min	0.16217126
d1_heartrate_min	0.16142935
temp_apache	0.1435303
d1_glucose_max	0.13883643
d1_spo2_min	0.13543685
pre_icu_los_days	0.13142563
diabetes_mellitus	0.12625858
ethnicity	0.1258931
d1_glucose_min	0.119905345
h1_resprate_max	0.11064488
apache_post_operative	0.10310288

h1_sysbp_max	0.092504896
bmi	0.07802819
resprate_apache	0.07792106
immunosuppression	0.06840778
d1_heartrate_max	0.060665146
d1_temp_min	0.0598779
h1_heartrate_min	0.0593229
h1_mbp_noninvasive_max	0.054002233
gcs_verbal_apache	0.05296661
apache_3j_diagnosis	0.043653425
apache_2_bodysystem	0.0417597
d1_mbp_noninvasive_max	0.041563842
solid_tumor_with_metastasis	0.0405452
icu_stay_type	0.040104613
d1_resprate_min	0.036936965
map_apache	0.03096487
elective_surgery	0.03003878
d1_sysbp_min	0.026470216

h1_resprate_min	0.026148904
d1_resprate_max	0.022111716
arf_apache	0.01036236
gcs_unable_apache	0.009818635
gender	0.008980301
h1_sysbp_min	0.008634388
d1_sysbp_max	0.0052802474
d1_temp_max	0.0039180275
d1_spo2_max	0.0036403171
Others	0.0

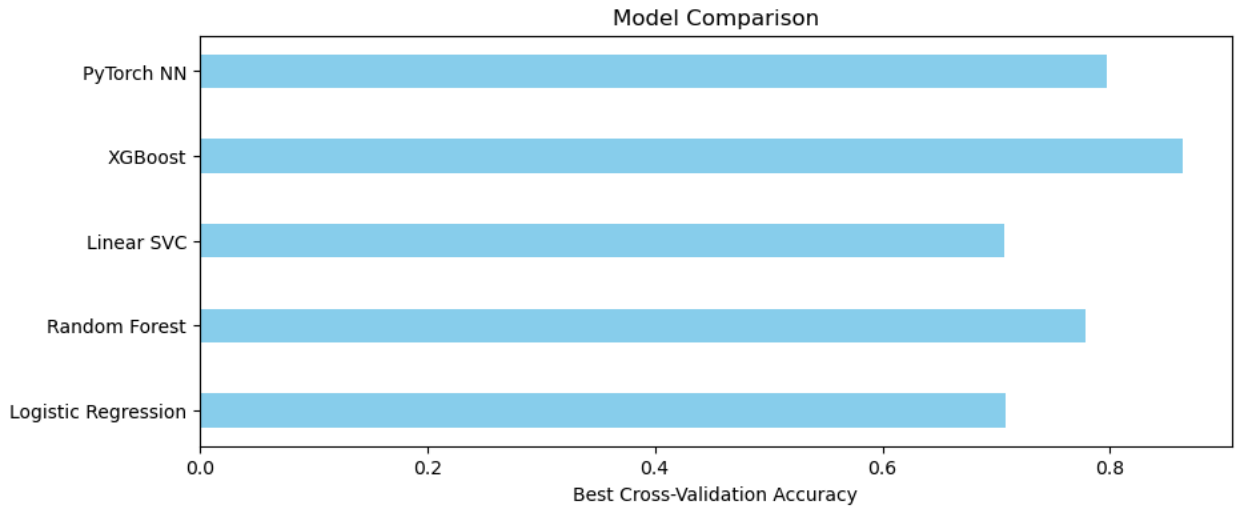
It is interesting to note how two different styles of models will place more weight on different types of features.

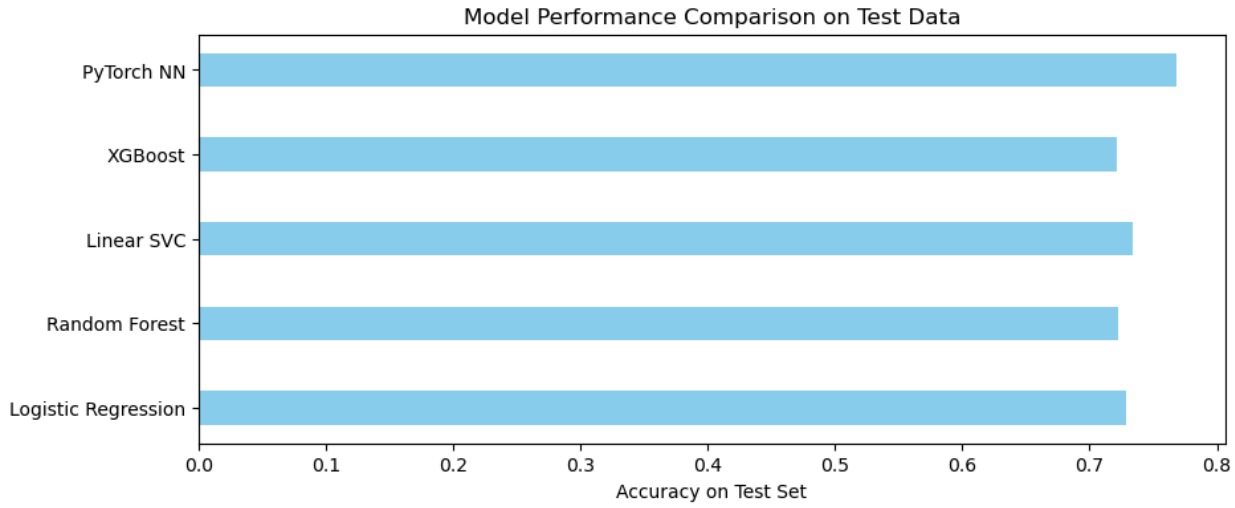
Results Experiment 2

For experiment 2, I sought to address the class imbalance by implementing a data sample strategy called **SMOTE (Synthetic Minority Over-sampling Technique)**. SMOTE is a statistical method used to balance dataset class distributions by artificially synthesizing new instances from the minority class. This technique identifies the minority class that needs augmentation and finds its nearest neighbors in the feature space. New samples are then created by interpolating between each minority class instance and one of its nearest neighbors. This approach helps in forming a more generalizable training dataset avoiding the pitfalls of overfitting that can occur when simply duplicating existing samples.

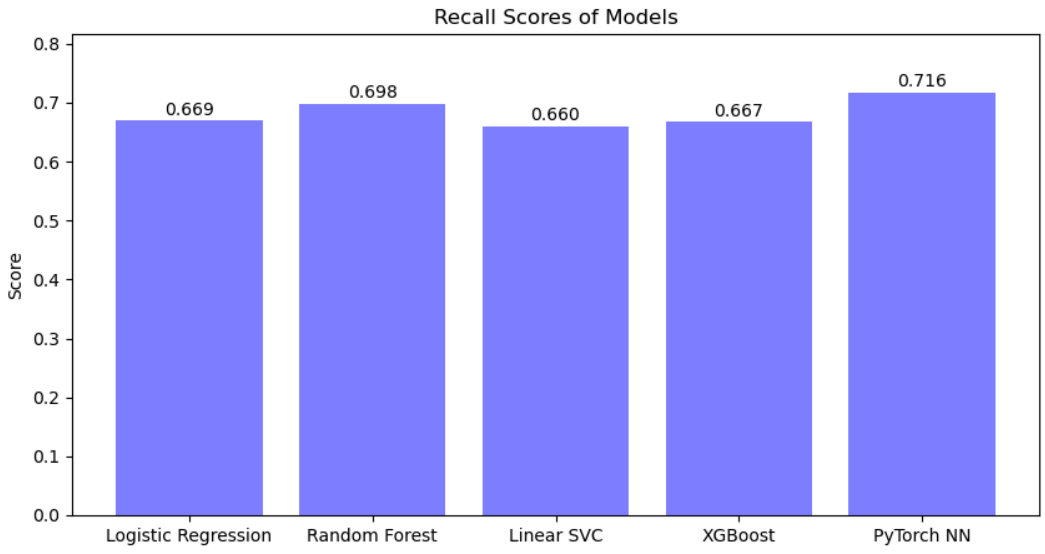
And here were the results:

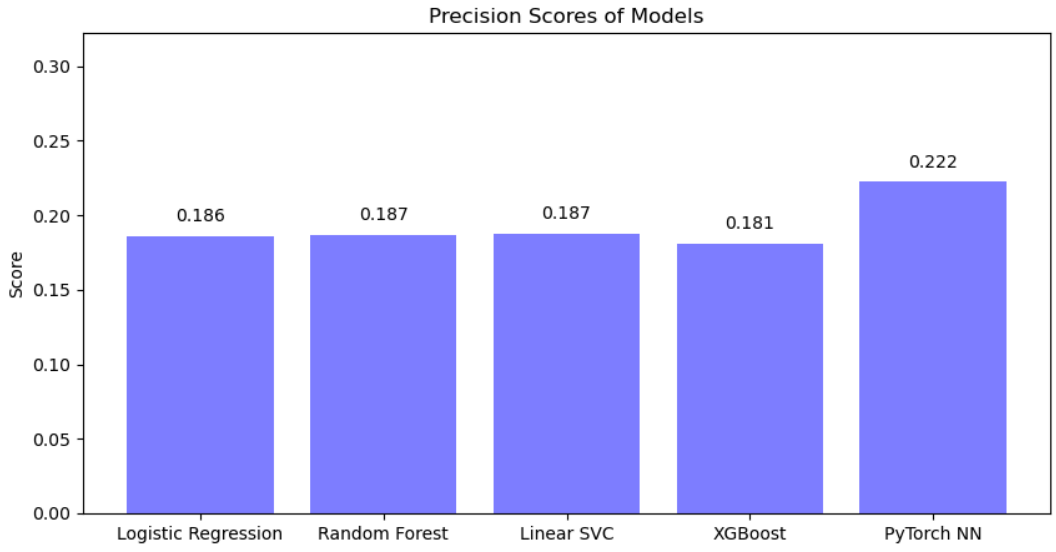
Model	Best CV Score	Best Parameters	Training Time (seconds)	Test Score
Logistic Regression	0.708404389	C: 0.01	0.29	0.728881001
Random Forest	0.778924063	max_depth: 10, n_estimators: 50	1.08	0.722191513
Linear SVC	0.707249695	C: 0.1	0.19	0.733580291
XGBoost	0.863835964	learning_rate: 0.01, max_depth: 30, n_estimators: 500	812.42	0.721638655
PyTorch NN	0.796765292	batch_size: 128, lr: 0.001, max_epochs: 100, module__ activation**: ReLU(), module__**num_units: 200	9841.23	0.768188854





What we notice here is that test set accuracy is down across the board with the best performing model being PyTorch NN at ~77%. However, looking into the precision and recall metrics we see substantial improvement here:





Note: You will also notice that my "SVM" model from the first experiment is different in my second experiment as I switched to a Linear SVC model. SVM models from scikit-learn have exponential runtime as input grows and I found that my SVMs, on downsampled data, had the best performance with a linear kernel. With that, I switched to leveraging cuML Linear SVC model which can train on the entire dataset in a fraction of the time.

The results demonstrate varying performance across the models tested with the PyTorch NN model leading in terms of test score, precision, and recall. All models have somewhat low precision relative to recall indicating a tendency to correctly identify positive instances but at the cost of more false positives. The PyTorch NN model not only has the highest test score but also balances precision and recall more effectively than the others. This suggests that the deep learning approach of the PyTorch NN is particularly well-suited for handling the complexities of this dataset making it the most robust choice among the models evaluated.

However, it should be noted that despite improvement here in performance with experiment 2, none of these models as presently constructed would suffice in a production setting. In the hospital setting, the cost of false positives and false negatives are high and these models would need to achieve higher precision and recall values in order to trust deploying resources to treat patients most equitably and effectively based on these models.

Feature Importance w/ UMAP

Given the above experimental setup, the feature importance table looks a little different. UMAP reduces the set of features down, in this case, to a 3 component representation that explains the majority of the variance in the dataset. Therefore, our feature importance table shows the weight each model placed on each given component.

Model	UMAP Component 1	UMAP Component 2	UMAP Component 3
Logistic Regression	0.13582405	0.25223166	0.00077778
Linear SVC	0.06179278	0.10930464	0.00480672
XGBoost	0.275318	0.42799884	0.2966832
PyTorch NN	0.28725122	0.30263158	0.1730429

Conclusion

In conclusion, in both experimental cases, using a combination of GridSearch, Dimensionality Reduction, and class imbalance strategies with SMOTE, did not produce viable models for production use. In the hospital setting, particularly dealing with patient mortality, the model performance in such instances must be held to the highest of standards to ensure resources deployed to save lives are done effectively. However, despite the shortcomings in approaches, the second experiment did highlight that perhaps deep neural networks offer the most viable approach toward a solution in this space with its ability to model complex relationships.

Lessons Learned and Next Steps

Here were some takeaways from this effort:

- GridSearch is computationally expensive. Many hours were spent on multi-hour and in some cases, multi-day runs of my code which limited iterative exploration. RandomGrid search, more spacing between hyperparameter values, and starting with GPU-accelerated algorithms would be my go to strategy to speed up analysis next.
- Exploring different dimensionality reduction techniques. PCA, incremental PCA, Truncated SVD, and Random Projections are all methods that should be explored. Additionally, while I evaluated the effect of UMAP using 2 and 3 components respectively, I would want to GridSearch over this parameter as well to test the effect on model performance.
- Assessing different imputation strategies. For this effort, I focused on utilizing median and mode impute strategies respectively for numerical and categorical features. Other methods exist like K-nearest neighbors and multiple imputations where multiple values are proposed for the missing value and utilize statistical analysis to combine them would be methods I would want to explore.
- Testing different outlier detection methods. While in the project Z-score was used, methods like IQR and DBScan clustering would be methods I would utilize to evaluate impact on model performance.