Analysis of MIMICIII ICU Data for Sepsis Cases

What decisions, and what business/societal value will be impacted

Our project focuses on refining ICU patient monitoring systems by developing a specialized, real-time machine learning model. This model will be designed to analyze a range of clinical indicators specifically for sepsis patients, predicting when their conditions might escalate into life-threatening situations.

The alarm system in our model will be highly targeted: it will only activate in cases of imminent, critical danger to sepsis patients, avoiding the issuance of alerts for less severe conditions. This focused approach aims to prevent the overload of false alarms, thereby reducing unnecessary stress on ICU staff and ensuring that medical attention is directed swiftly and accurately to patients in urgent need.

By ensuring that alarms are only triggered by truly critical situations, the model will enhance the effectiveness of patient monitoring in the ICU, leading to better patient outcomes, fewer medical errors, and improved overall efficiency in patient care management.

1. Table Join and Feature Description

Tables used:

- `patients`
- `icustays`
- `chartevents`
- `outputevents`
- `d_items`

Personal Info Features

- `subject_id` : Number for each patient
- `icustay_id` : Number for each ICU case
- `age` : Patient's age
- `gender` : Patient's gender
- `marital_status` : Marital status of the patient
- `ethnicity` : Ethnicity of the patient
- `icu_duration_hour` : Time spent in ICU
- `icu_times` : The ith time being admitted to ICU
- `icu_times_total` : Total number of times admitted to ICU
- `icu_times_sepsis` : The ith time admitted to ICU because of sepsis
- `icu_times_total_sepsis` : Total times admitted to ICU because of sepsis
- `hospital_expire_flag` : Died in hospital
- `expire_flag` : Died eventually
- `died_immediately` : Died in hospital or within 24h after leaving hospital

2. Test Labels Selection

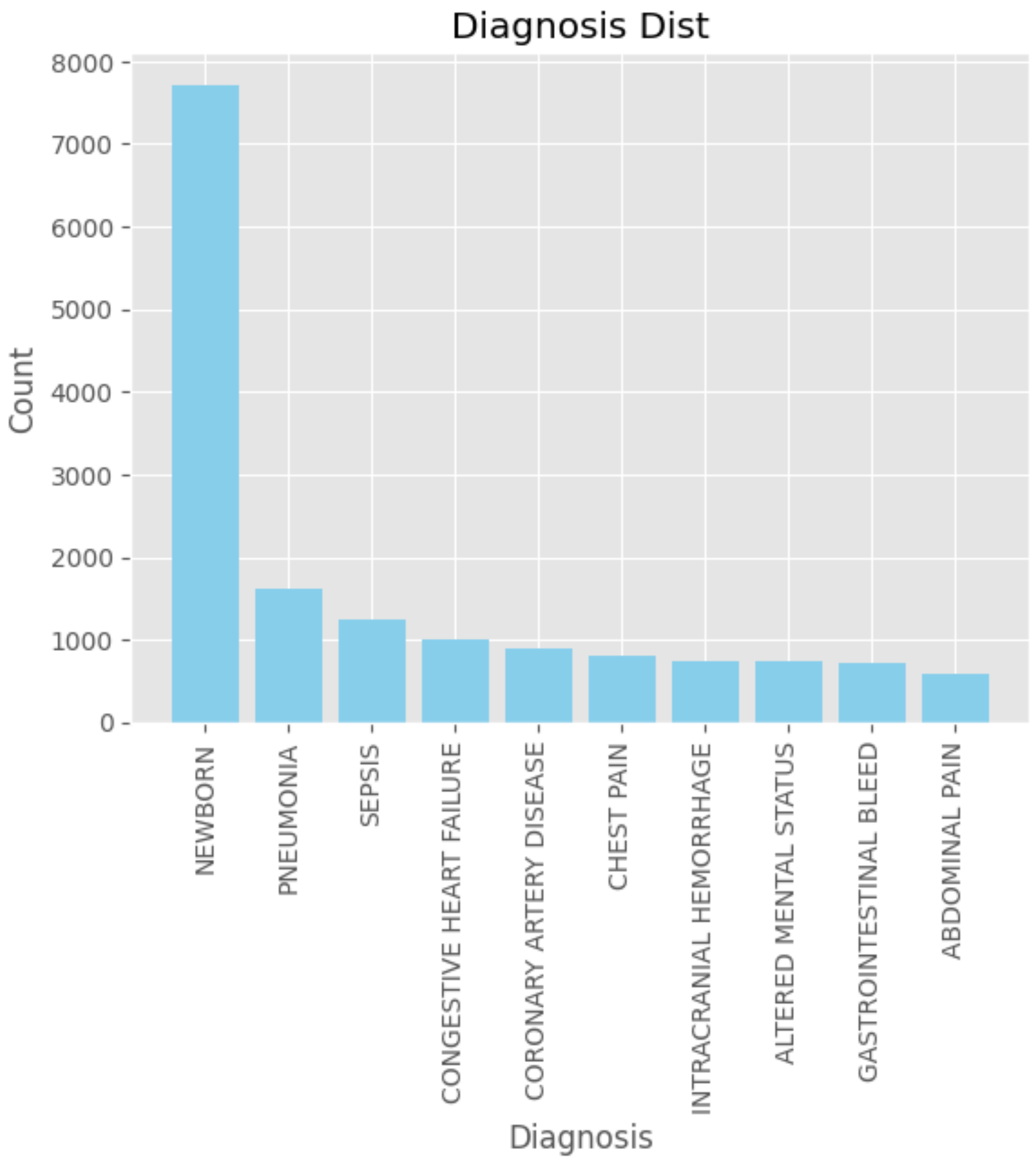
What desease should we analyze? Why sepsis?

Since there are many different diseases in MIMICIII and it would be more efficient if we choose only one disease to analyze.

First we analyze the number of different diseases.

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import psycopg2
        from IPython.display import display, HTML

        plt.style.use('ggplot')

        sqluser = 'postgres'
        dbname = 'mimic'
        schema_name = 'mimiciii_generate'
        password = '012500'
        con = psycopg2.connect(dbname=dbname, user=sqluser, password=password)
        query_schema = 'set search_path to ' + schema_name + ';'

        diag_only_query = '''
        select distinct diagnosis, count(distinct icustay_id) as num
        from mimiciii_generate.patients_all
        where subject_id is not null
        group by diagnosis
        order by diagnosis asc;
        '''
        diagnosis_num = pd.read_sql_query(diag_only_query, con)
        diagnosis_num = diagnosis_num.sort_values(by='num',ascending=False).reset_in
        print(diagnosis_num)
        #diagnosis_num.to_csv(r'F:\STUDY\python_code\Pract_DA\prepocess\all_diag.csv

        sepsis_top10 = diagnosis_num.head(10)
        plt.bar(sepsis_top10['diagnosis'], sepsis_top10['num'],color='skyblue')
        plt.title('Diagnosis Dist')
        plt.xlabel('Diagnosis')
        plt.xticks(rotation=90)
        plt.ylabel('Count')
        plt.show()
```

```
                                diagnosis   num
0                                 NEWBORN  7718
1                               PNEUMONIA  1619
2                                  SEPSIS  1250
3                CONGESTIVE HEART FAILURE   993
4                 CORONARY ARTERY DISEASE   890
...                                   ...   ...
14983          ETOH WITHDRAWAL;CHEST PAIN     1
14984           ETOH WITHDRAWAL;CIRRHOSIS     1
14985         ETOH WITHDRAWAL;EKG CHANGES     1
14986   ETOH WITHDRAWAL;FAILURE TO THRIVE     1
14987                                None     1

[14988 rows x 2 columns]
```



It turns out that, except for newborn, sepsis has the second highest number of cases. Besides, sepsis is one of the most deadly diseases in the United States. Therefore, we choose **sepsis** as our subject.

What tests should be considered?

Since there are hundreds of medical tests in MIMICIII, it would be very time consuming to process all of these test values and take all of them into account. Therefore, we will select a small part of these tests to be our features.

We calculated the proportion of patients participating in each type of test. As shown in the code below, and we select the **top 20** tests. Some tests are actually the same test but with different names, we regard them together as one test.

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import psycopg2
from IPython.display import display, HTML

plt.style.use('ggplot')

sqluser = 'postgres'
dbname = 'mimic'
schema_name = 'mimiciii_generate'
password = '012500'
con = psycopg2.connect(dbname=dbname, user=sqluser, password=password)
query_schema = 'set search_path to ' + schema_name + ';'

patient_sepsis_complete_query = '''
WITH   patient_sepsis as (
        select * from mimiciii_generate.patient_sepsis_complete
        )
        select * from patient_sepsis

'''

patient_sepsis_complete = pd.read_sql_query(patient_sepsis_complete_query, c
case_num = patient_sepsis_complete['icustay_id'].nunique()
test_counts = patient_sepsis_complete.groupby('test_label')['icustay_id'].nu
test_percent = (test_counts / case_num * 100).round(2).reset_index()
test_percent.columns = ['test_label', 'test_percent']
test_percent = test_percent.sort_values(by='test_percent', ascending=False).
print(test_percent)
```

```
                 test_label  test_percent
0        Glucose (70-105)         61.80
1     Potassium (3.5-5.3)         61.80
2        Sodium (135-148)         61.75
3              Hematocrit         61.64
4      Chloride (100-112)         61.59
...                   ...           ...
1250   Erythromycin Oint.          0.05
1251                 FIO2          0.05
1252    FIO2 Alarm [High]          0.05
1253     FIO2 Alarm [Low]          0.05
1254  KCL-10 MEQ-DIALYSIS          0.05

[1255 rows x 2 columns]
```

The final selected medical test lables are as shown below.

Measurement Type	Initial Label	Merged Label
Blood Sugar	Glouse	glouse
	Glucose (70-105)	
	Fingerstick Glucose	Fingerstick Glucose
Electrolyte	potassium	potassium
	Potassium (3.5-5.3)	
	Potassium (3.5-5.3)	
	Sodium (135-148)	Sodium
	Chloride (100-112)	Chloride
	Magnesium (1.6-2.6)	Magnesium
Blood Component	Hematocrit (35-51)	Hematocrit
	Hematocrit	
	Hemoglobin	Hemoglobin
	RBC(3.6-6.2)	RBC
	RBC	
	Platelets	Platelets
	WBC (4-11,000)	WBC
	WBC (4-11,000)	
	WBC 4.0-11.0	
Kidney Function	BUN (6-20)	BUN
	BUN	
	Creatinine (0-1.3)	Creatinine
Blood Gas	Carbon Dioxide	Carbon Dioxide
Vital Sign	Heart Rate	Heart Rate

	Heart Rhythm	Heart Rhythm
	Respiratory Rate	Respiratory Rate
	Temperature (C to F)	Temperature (C to F)

**Min, Max, and Avg Calculations for Each Test

Example for Glucose:

- Min: 70
- Max: 105
- Avg: Calculated from available data
- Alarm times: Times of the value below/beyong certern threshold. (Respiratory Rate)

Exception: for Heart Rhythm, which is in text value, we calculate the mode and the last value of the patient.

3. Pre-process

Last Case Handling

Since the same patient will have similar test values, which might affect the model, we select the last time one patient get admitted to ICU because of sepsis if the patient get admitted to ICU for multiple times.

Drop Duplicates

Since the mode of heart rhythm might not be a unique value if two or more values have the same amount. We decided to delete all of them completely. Only 10 patients (20 records) will be removed, which might not affect the data set too much. Also, only one of the duplicated records ends up died. If we are doing an outlier detection, this also does not remove too much of the positive samples.

Drop Nulls

Top 20 most common tests among sepsis patients.If a patient has more than 25% of tests not conducted, then remove that patient. End up in 922 cases in total.

Below is how we build the dataset and a sample of this dataset

--We process the initial data on PostgreSQL

In [ ]:
```
complete_test_query = '''
-- glucose
with  all_glu as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                MIN(CAST(test_value AS FLOAT)) AS min_,
                MAX(CAST(test_value AS FLOAT)) AS max_,
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Glucose'
and test_label != 'Fingerstick Glucose'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),

glucose as(
    select subject_id,
                icustay_id,
                MIN(cast(min_ as float)) as glu_min,
                MAX(cast(max_ as float)) as glu_max,
                round(cast(avg(avg_) as numeric), 2) as glu_avg
                from all_glu
                group by subject_id, icustay_id
),


-- finger glucose
figglucose as (
    SELECT DISTINCT subject_id,
                                icustay_id,
                MIN(CAST(test_value AS FLOAT)) AS figglu_min,
                MAX(CAST(test_value AS FLOAT)) AS figglu_max,
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label = 'Fingerstick Glucose'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id
),


-- potassium
all_potassium as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                MIN(CAST(test_value AS FLOAT)) AS min_,
                MAX(CAST(test_value AS FLOAT)) AS max_,
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Potassium'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),

potassium as (
    select subject_id,
                icustay_id,
                MIN(cast(min_ as float)) as pot_min,
                MAX(cast(max_ as float)) as pot_max,
                round(cast(avg(avg_) as numeric), 2) as pot_avg
                from all_potassium
                group by subject_id, icustay_id
),


-- sodium
all_sodium as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                MIN(CAST(test_value AS FLOAT)) AS min_,
                MAX(CAST(test_value AS FLOAT)) AS max_,
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'sodium'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

sodium as (
    select subject_id,
                icustay_id,
                MIN(cast(min_ as float)) as sod_min,
                MAX(cast(max_ as float)) as sod_max,
                round(cast(avg(avg_) as numeric), 2) as sod_avg
                from all_sodium
                group by subject_id, icustay_id
),


-- Hematocrit
all_hema as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Hematocrit'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

hema as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS hem_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS hem_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS hem_avg
                from all_hema
                group by subject_id, icustay_id
),


-- Chloride
all_chlo as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Chloride'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

chlo as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS chl_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS chl_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS chl_avg
                from all_chlo
                group by subject_id, icustay_id
  ),


-- BUN
all_bun as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'bun'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

bun as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS bun_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS bun_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS bun_avg
                from all_bun
                group by subject_id, icustay_id
  ),


-- Creatinine
all_crea as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Creatinine'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
```

```
GROUP BY subject_id, icustay_id, test_label),

crea as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS cre_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS cre_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS cre_avg
                from all_crea
                group by subject_id, icustay_id
  ),


-- Hemoglobin
hemo as (
    SELECT DISTINCT subject_id,
                                icustay_id,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC),2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Hemoglobin'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),

-- Carbon dioxide
carb as (
    SELECT DISTINCT subject_id,
                                icustay_id,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC),2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Carbon Dioxide'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),


-- RBC
all_rbc as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'RBC'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

rbc as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS rbc_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS rbc_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS rbc_avg
                from all_rbc
                group by subject_id, icustay_id
  ),


-- Platelets
all_plat as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Platelets'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

pla as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS pla_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS pla_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS pla_avg
                from all_plat
                group by subject_id, icustay_id
  ),


-- WBC
all_wbc as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'wbc'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

wbc as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS wbc_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS wbc_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS wbc_avg
                from all_wbc
                group by subject_id, icustay_id
  ),
-- heart rate
hea as (
    SELECT DISTINCT subject_id,
                                icustay_id,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC),2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'heart rate'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),


-- heart rhythm
frequency AS (
    SELECT
        subject_id,
        icustay_id,
        test_label,
        test_value,
        COUNT(*) AS freq
    FROM
        mimiciii_generate.patient_sepsis_complete
    WHERE
        test_label ~* 'heart rhythm' AND
        subject_id IS NOT NULL
    GROUP BY
        subject_id, icustay_id, test_label, test_value
),
max_frequency AS (
    SELECT
        subject_id,
        icustay_id,
        test_label,
        MAX(freq) AS max_freq
    FROM
        frequency
    GROUP BY
        subject_id, icustay_id, test_label
),
mode_value AS (
    SELECT
        f.subject_id,
        f.icustay_id,
        f.test_label,
        f.test_value AS mode_value
    FROM
        frequency f
    JOIN
        max_frequency mf ON f.subject_id = mf.subject_id
        AND f.icustay_id = mf.icustay_id
        AND f.test_label = mf.test_label
        AND f.freq = mf.max_freq
),
last_value AS (
    SELECT
        subject_id,
        icustay_id,
        test_label,
        test_value AS last_test_value,
        ROW_NUMBER() OVER (PARTITION BY subject_id, icustay_id, test_label O
    FROM
        mimiciii_generate.patient_sepsis_complete
    WHERE
        test_label ~* 'heart rhythm' AND
        subject_id IS NOT NULL
),

heaRhy as (SELECT
    lv.subject_id,
    lv.icustay_id,
    mv.mode_value as heaR_mode,
    lv.last_test_value as heaR_last
FROM
    last_value lv
JOIN
    mode_value mv ON lv.subject_id = mv.subject_id
    AND lv.icustay_id = mv.icustay_id
    AND lv.test_label = mv.test_label
WHERE
    lv.rn = 1
),


-- Magnesium
all_mag as (SELECT DISTINCT subject_id,
                                icustay_id,
                                test_label,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Magnesium'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label),

mag as (
    select subject_id,
                icustay_id,
                ROUND(CAST(MIN(CAST(min_ AS FLOAT)) AS NUMERIC), 2) AS mag_m
        ROUND(CAST(MAX(CAST(max_ AS FLOAT)) AS NUMERIC), 2) AS mag_max,
        ROUND(CAST(AVG(CAST(avg_ AS FLOAT)) AS NUMERIC), 2) AS mag_avg
                from all_mag
                group by subject_id, icustay_id
  ),


-- Respiratory Rate
res as (
    SELECT DISTINCT subject_id,
                                icustay_id,
                round(cast(MIN(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(cast(MAX(CAST(test_value AS FLOAT)) as numeric),2) AS
                round(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC),2) AS
FROM mimiciii_generate.patient_sepsis_complete
WHERE test_label ~* 'Respiratory Rate'
and not test_label ~* 'Respiratory Rate Set'
and test_value ~ '^\d+(\.\d+)?$'
and subject_id is not null
GROUP BY subject_id, icustay_id, test_label
),


--SpO2
spo as (
    SELECT DISTINCT
    subject_id,
    icustay_id,
    ROUND(CAST(MIN(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS spo_min,
    ROUND(CAST(MAX(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS spo_max,
    ROUND(CAST(AVG(CAST(test_value AS FLOAT)) AS NUMERIC), 2) AS spo_avg,
    COUNT(CASE WHEN CAST(test_value AS FLOAT) < 90 THEN 1 ELSE NULL END) AS
FROM
    mimiciii_generate.patient_sepsis_complete
WHERE
    test_label = 'SpO2'
    AND test_value ~ '^\d+(\.\d+)?$'
    AND subject_id IS NOT NULL
GROUP BY
    subject_id, icustay_id, test_label
),


-- temperature F
TemperatureData AS (
    SELECT
        subject_id,
        icustay_id,
        CASE
            WHEN test_label = 'Temperature C' THEN CAST(test_value AS FLOAT)
            ELSE CAST(test_value AS FLOAT)
        END AS temp_f
    FROM
        mimiciii_generate.patient_sepsis_complete
    WHERE
        (test_label = 'Temperature F' OR test_label = 'Temperature C')
        AND test_value ~ '^\d+(\.\d+)?$'  -- Ensures test_value is numeric
        AND subject_id IS NOT NULL
),

temp as (SELECT
    subject_id,
    icustay_id,
    ROUND(CAST(MIN(temp_f) AS NUMERIC), 2) AS temp_min,
    ROUND(CAST(MAX(temp_f) AS NUMERIC), 2) AS temp_max,
    ROUND(CAST(AVG(temp_f) AS NUMERIC), 2) AS temp_avg
FROM
    TemperatureData
```

```
GROUP BY
    subject_id, icustay_id
)


-- basic info

select distinct
bi.subject_id,
bi.icustay_id,
bi.age_inicu_year as age,
bi.gender,
bi.marital_status,
bi.ethnicity,
bi.icu_duration_hour,
bi.icu_times,
bi.icu_times_total,
bi.icu_times_sepsis,
bi.icu_times_total_sepsis,

glu.glu_min,
glu.glu_max,
glu.glu_avg,

fg.figglu_min,
fg.figglu_max,
fg.figglu_avg,

pot.pot_min,
pot.pot_max,
pot.pot_avg,

sod.sod_min,
sod.sod_max,
sod.sod_avg,

hem.hem_min,
hem.hem_max,
hem.hem_avg,

chl.chl_min,
chl.chl_max,
chl.chl_avg,

bun.bun_min,
bun.bun_max,
bun.bun_avg,

cre.cre_min,
cre.cre_max,
cre.cre_avg,

hemo.hemo_min,
hemo.hemo_max,
hemo.hemo_avg,

car.car_min,
car.car_max,
car.car_avg,

rbc.rbc_min,
rbc.rbc_max,
rbc.rbc_avg,

pla.pla_min,
pla.pla_max,
pla.pla_avg,

wbc.wbc_min,
wbc.wbc_max,
wbc.wbc_avg,

hea.hea_min,
hea.hea_max,
hea.hea_avg,

heaR.heaR_mode,
heaR.heaR_last,

mag.mag_min,
mag.mag_max,
mag.mag_avg,

res.res_min,
res.res_max,
res.res_avg,

spo.spo_min,
spo.spo_max,
spo.spo_avg,
spo.spo_alarms,

temp.temp_min,
temp.temp_max,
temp.temp_avg,


bi.HOSPITAL_EXPIRE_FLAG,
bi.expire_flag,
bi.died_immediately

from
mimiciii_generate.patient_sepsis_complete bi
left join
glucose glu on bi.icustay_id = glu.icustay_id
left join
figglucose fg on bi.icustay_id = fg.icustay_id
left join
potassium pot on bi.icustay_id = pot.icustay_id
left join
sodium sod on bi.icustay_id = sod.icustay_id
left join
hema hem on bi.icustay_id = hem.icustay_id
left join
```

```
chlo chl on bi.icustay_id = chl.icustay_id
left join
bun on bi.icustay_id = bun.icustay_id
left join
crea cre on bi.icustay_id = cre.icustay_id
left join
hemo on bi.icustay_id = hemo.icustay_id
left join
carb car on bi.icustay_id = car.icustay_id
left join
rbc on bi.icustay_id = rbc.icustay_id
left join
pla on bi.icustay_id = pla.icustay_id
left join
wbc on bi.icustay_id = wbc.icustay_id
left join
hea on bi.icustay_id = hea.icustay_id
left join
heaRhy heaR on bi.icustay_id = heaR.icustay_id
left join
mag on bi.icustay_id = mag.icustay_id
left join
res on bi.icustay_id = res.icustay_id
left join
spo on bi.icustay_id = spo.icustay_id
left join
temp on bi.icustay_id = temp.icustay_id
'''
```

The dataset we will use in this project

In [3]:
```
import pandas as pd
file_path = 'F:\STUDY\python_code\Pract_DA\prepocess\patient_sepsis_all_drop
data = pd.read_csv(file_path)
data.head(10)
```

Out[3]:

	Unnamed: 0	subject_id	icustay_id	age	gender	marital_status	ethnicity	icu_dur
0	0	21	216859	87.82	M	MARRIED	WHITE	
1	1	33	296681	82.39	M	MARRIED	UNKNOWN/NOT SPECIFIED	
2	2	38	248910	75.94	M	WIDOWED	WHITE	
3	3	62	216609	68.77	M	MARRIED	PATIENT DECLINED TO ANSWER	
4	4	94	229012	74.43	M	MARRIED	ASIAN	
5	5	157	264885	80.53	M	SINGLE	WHITE	
6	7	188	278679	51.80	M	MARRIED	WHITE	
7	8	202	228132	75.76	F	MARRIED	WHITE	
8	9	234	252814	52.59	M	WIDOWED	WHITE	
9	10	242	270389	76.98	F	WIDOWED	UNKNOWN/NOT SPECIFIED	

10 rows × 72 columns

4. Comorbidity Consideration

Currently not considered. Sepsis is mostly caused by other deseases, and most of the cases don't have this recorded. We will have to dig deep in the raw data set to build this feature. So we decided to move on first to see how this dataset performs.

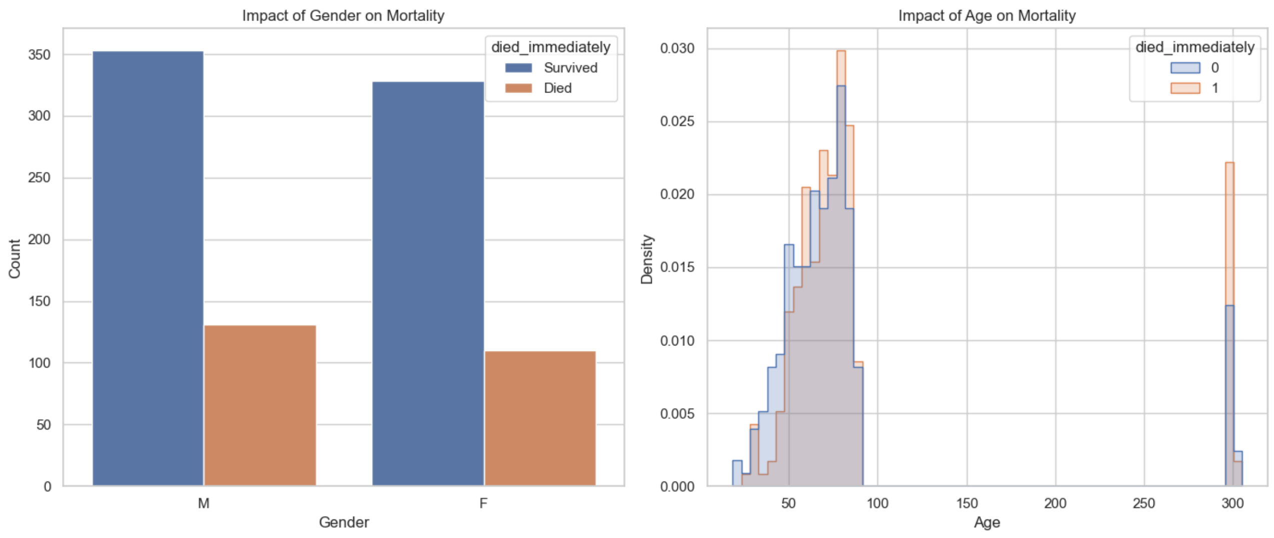
Data Description and Distribution

How death rate relate to the distribution of the values

In [22]:

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

file_path = r'F:\STUDY\python_code\Pract_DA\prepocess\patient_sepsis_all_drc
data = pd.read_csv(file_path)
```

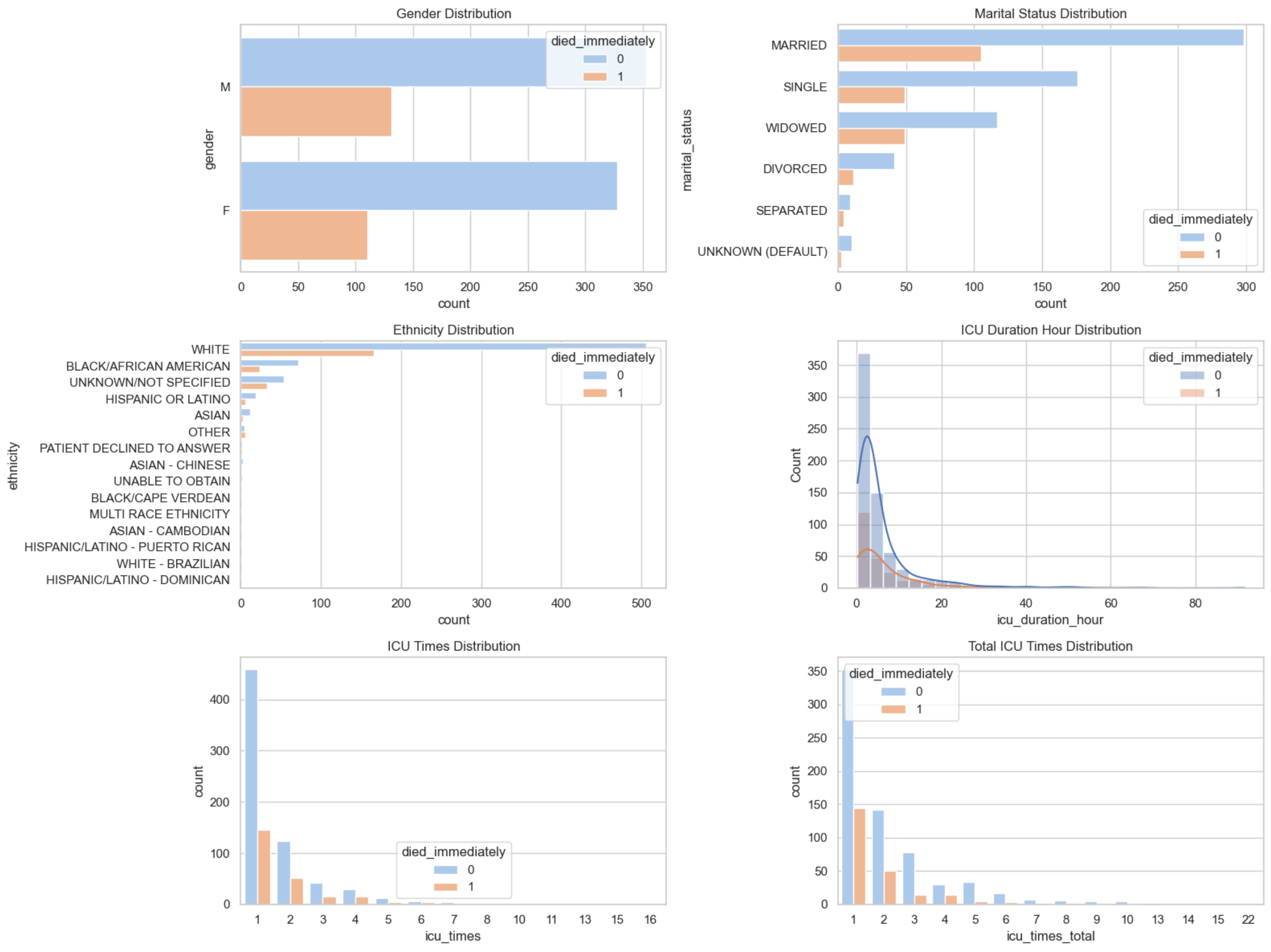
In [23]:
```python
sns.set(style="whitegrid")

fig, ax = plt.subplots(1, 2, figsize=(14, 6))
sns.countplot(x='gender', hue='died_immediately', data=data, ax=ax[0])
ax[0].set_title('Impact of Gender on Mortality')
ax[0].set_xlabel('Gender')
ax[0].set_ylabel('Count')
ax[0].legend(title='died_immediately', labels=['Survived', 'Died'])

sns.histplot(data=data, x='age', hue='died_immediately', element='step', sta
ax[1].set_title('Impact of Age on Mortality')
ax[1].set_xlabel('Age')
ax[1].set_ylabel('Density')

plt.tight_layout()
plt.show()
```



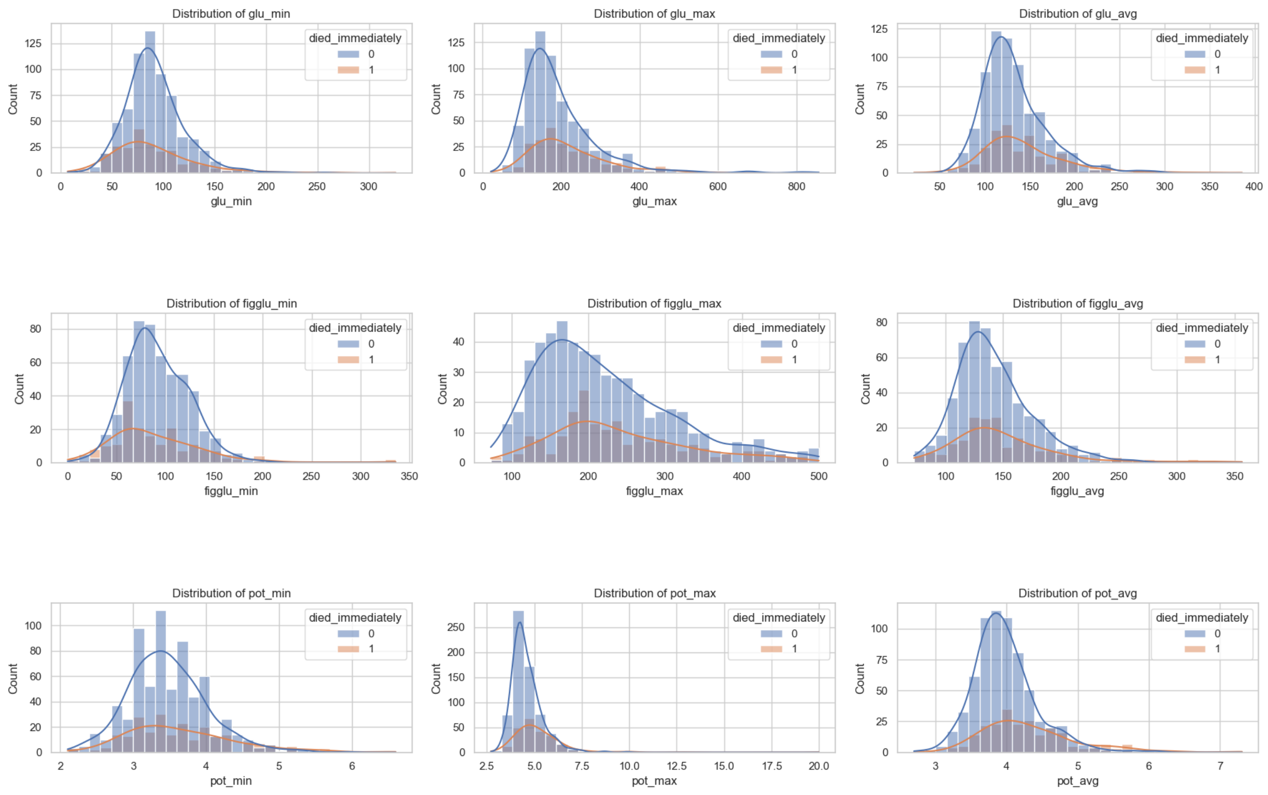
The gender is balanced, and the death rate within each gender also consistants with each other.

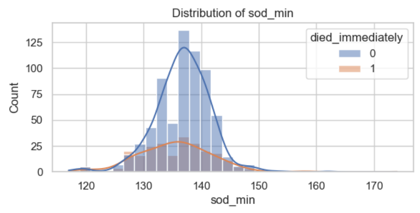
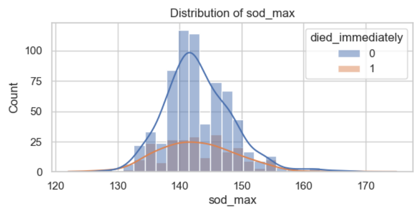
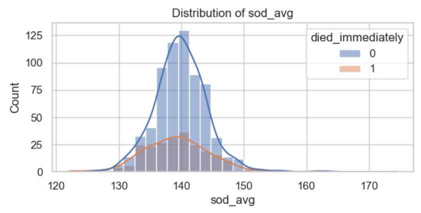
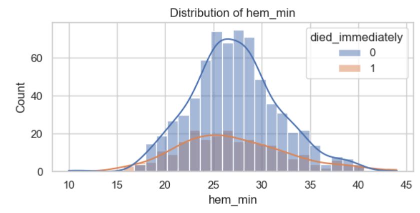
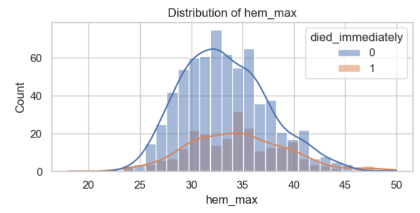
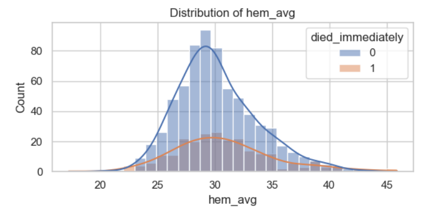
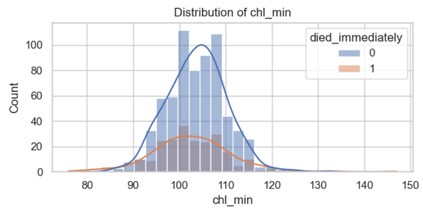
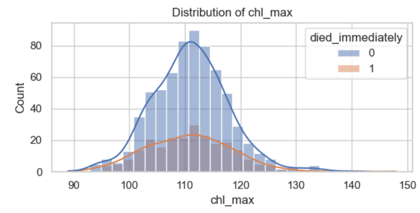
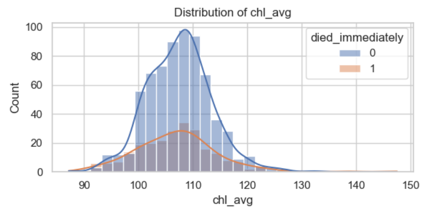
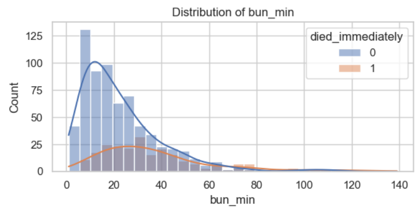
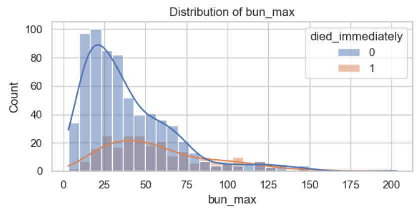
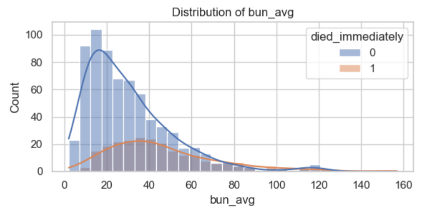
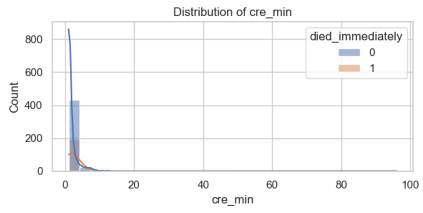
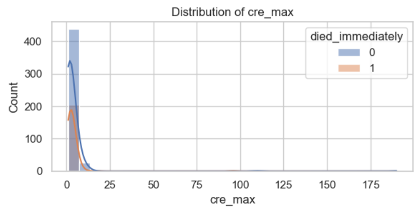
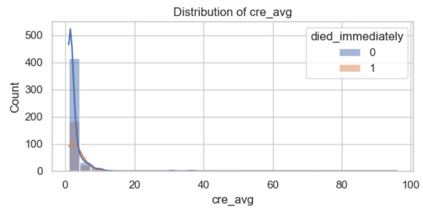
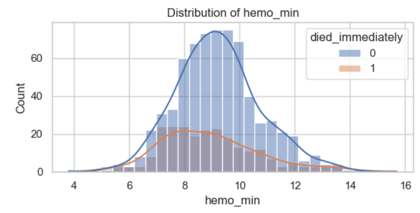
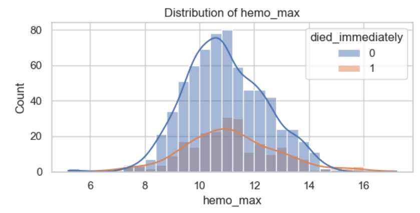
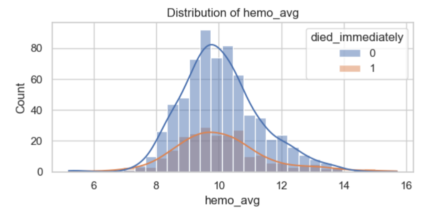
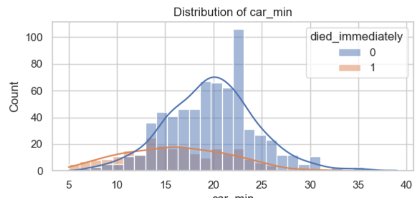
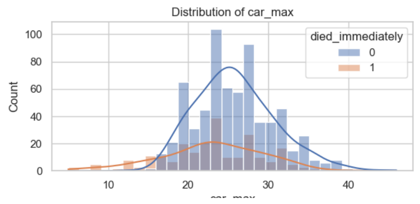
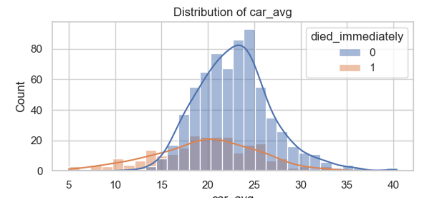
The death rate also increase as the age increases, which also shows a strong correlation between age and death.

```
In [24]: marital_status_order = data['marital_status'].value_counts().index
         ethnicity_order = data['ethnicity'].value_counts().index

         fig, axes = plt.subplots(3, 2, figsize=(16, 12))

         sns.countplot(y='gender', data=data, hue='died_immediately', ax=axes[0, 0],
         axes[0, 0].set_title('Gender Distribution')

         sns.countplot(y='marital_status', data=data, order=marital_status_order, hue
         axes[0, 1].set_title('Marital Status Distribution')

         sns.countplot(y='ethnicity', data=data, order=ethnicity_order, hue='died_imm
         axes[1, 0].set_title('Ethnicity Distribution')

         sns.histplot(data=data, x='icu_duration_hour', hue='died_immediately', bins=
         axes[1, 1].set_title('ICU Duration Hour Distribution')

         sns.countplot(x='icu_times', data=data, hue='died_immediately', ax=axes[2, 0
         axes[2, 0].set_title('ICU Times Distribution')

         sns.countplot(x='icu_times_total', data=data, hue='died_immediately', ax=axe
         axes[2, 1].set_title('Total ICU Times Distribution')

         plt.tight_layout()
         plt.show()
```

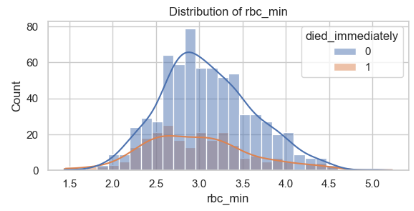


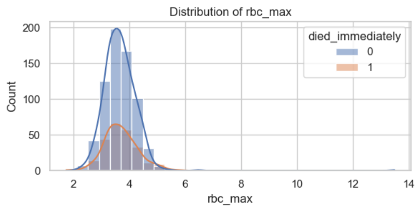
In [21]:
```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# 读取 csv 文件
file_path = 'F:/STUDY/python_code/Pract_DA/prepocess/patient_sepsis_all_drop
data = pd.read_csv(file_path)

test_columns = data.columns[12:-3]  # 获取要绘制的列

# 计算行数
rows = (len(test_columns) + 2) // 3

# 创建子图
fig, axes = plt.subplots(rows, 3, figsize=(18, rows * 4))
axes = axes.flatten()

# 遍历列并绘制图表
for i, col in enumerate(test_columns):
    sns.histplot(data=data, x=col, hue='died_immediately', ax=axes[i], bins=
    axes[i].set_title(f'Distribution of {col}')
    if col == 'hear_mode' or col == 'hear_last':
        axes[i].tick_params(axis='x', rotation=90)

# 隐藏多余的子图
for j in range(i + 1, len(axes)):
    axes[j].set_visible(False)

# 调整布局
plt.tight_layout()
plt.show()
```

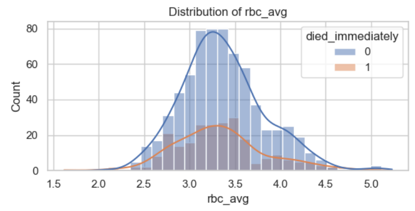


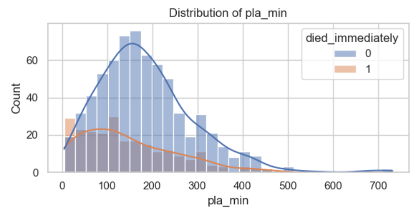

Distribution of sod_min
Count
sod_min
died_immediately
0
1

Distribution of sod_max
Count
sod_max
died_immediately
0
1

Distribution of sod_avg
Count
sod_avg
died_immediately
0
1

Distribution of hem_min
Count
hem_min
died_immediately
0
1

Distribution of hem_max
Count
hem_max
died_immediately
0
1

Distribution of hem_avg
Count
hem_avg
died_immediately
0
1

Distribution of chl_min
Count
chl_min
died_immediately
0
1

Distribution of chl_max
Count
chl_max
died_immediately
0
1

Distribution of chl_avg
Count
chl_avg
died_immediately
0
1

Distribution of bun_min
Count
bun_min
died_immediately
0
1

Distribution of bun_max
Count
bun_max
died_immediately
0
1

Distribution of bun_avg
Count
bun_avg
died_immediately
0
1

Distribution of cre_min
Count
cre_min
died_immediately
0
1

Distribution of cre_max
Count
cre_max
died_immediately
0
1

Distribution of cre_avg
Count
cre_avg
died_immediately
0
1

Distribution of hemo_min
Count
hemo_min
died_immediately
0
1

Distribution of hemo_max
Count
hemo_max
died_immediately
0
1

Distribution of hemo_avg
Count
hemo_avg
died_immediately
0
1

Distribution of car_min
Count
died_immediately
0
1

Distribution of car_max
Count
died_immediately
0
1

Distribution of car_avg
Count
died_immediately
0
1

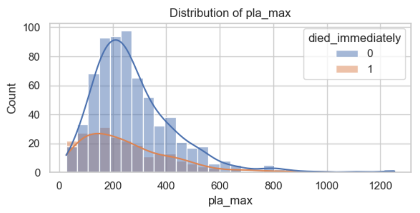
cal_min cal_max cal_avg

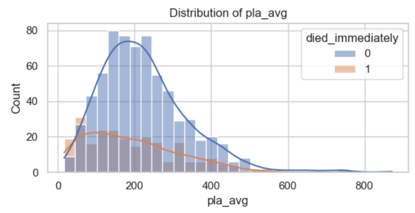


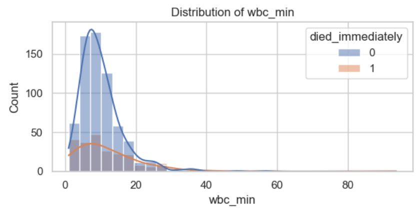


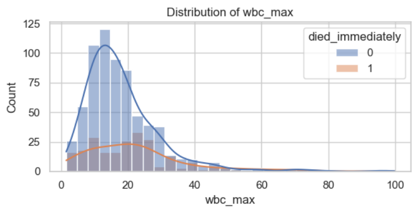


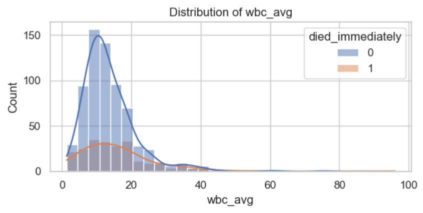


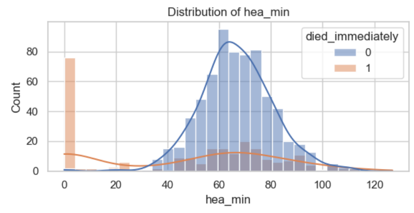


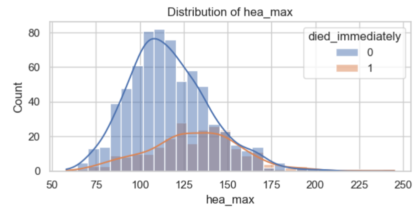


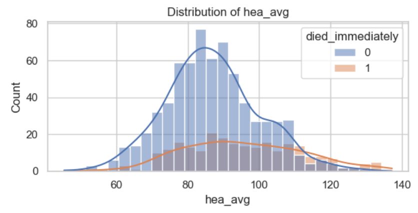


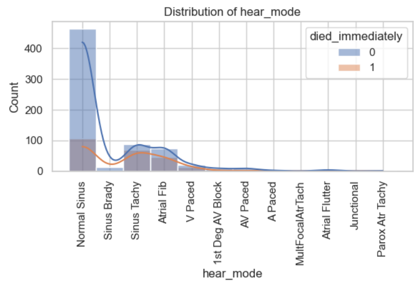


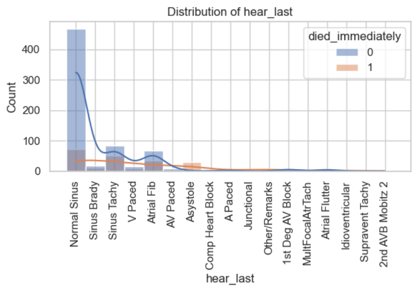


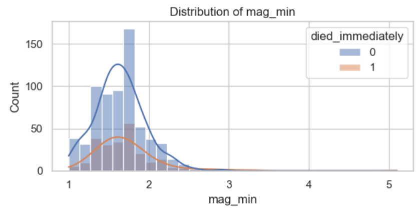


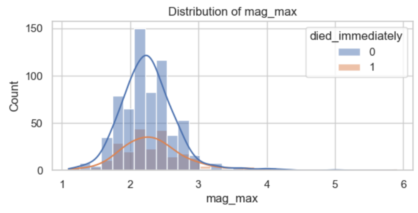


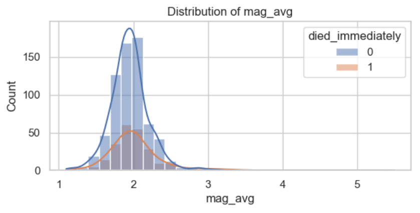


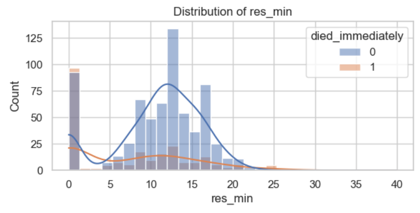


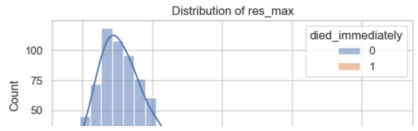


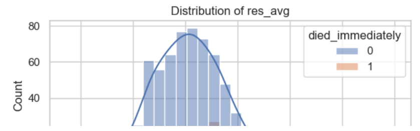


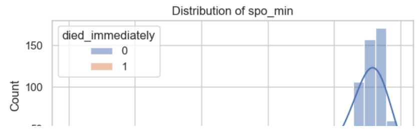


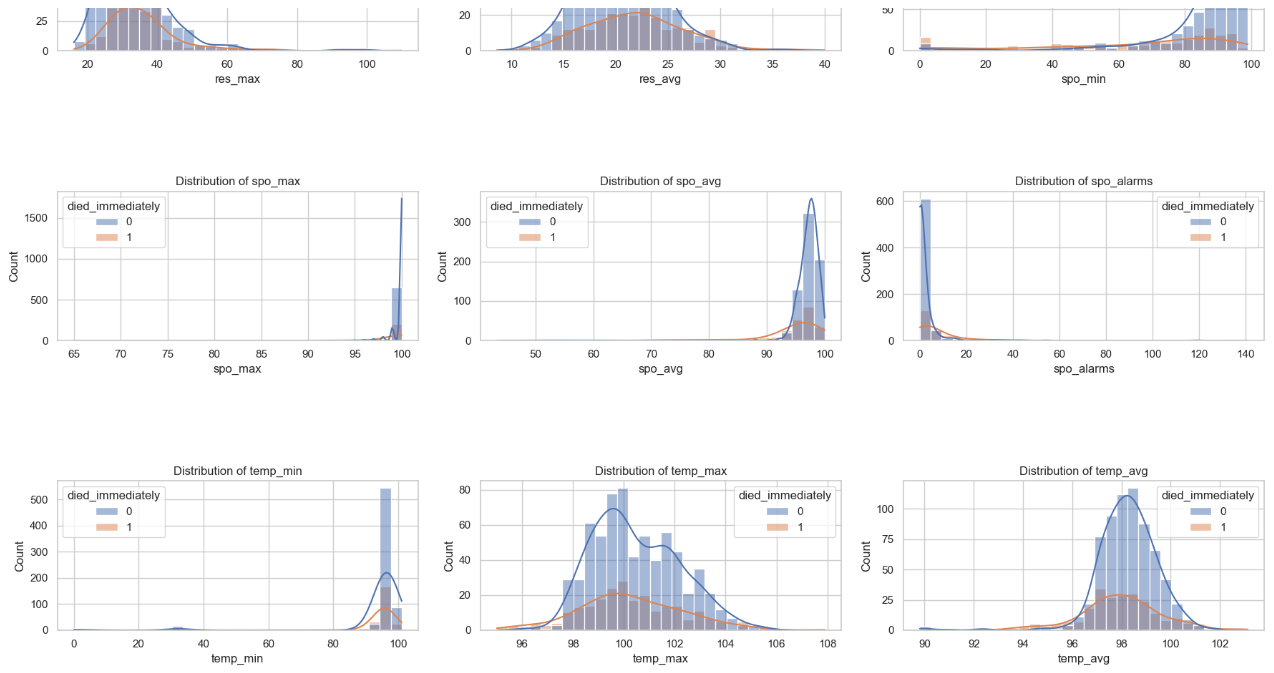












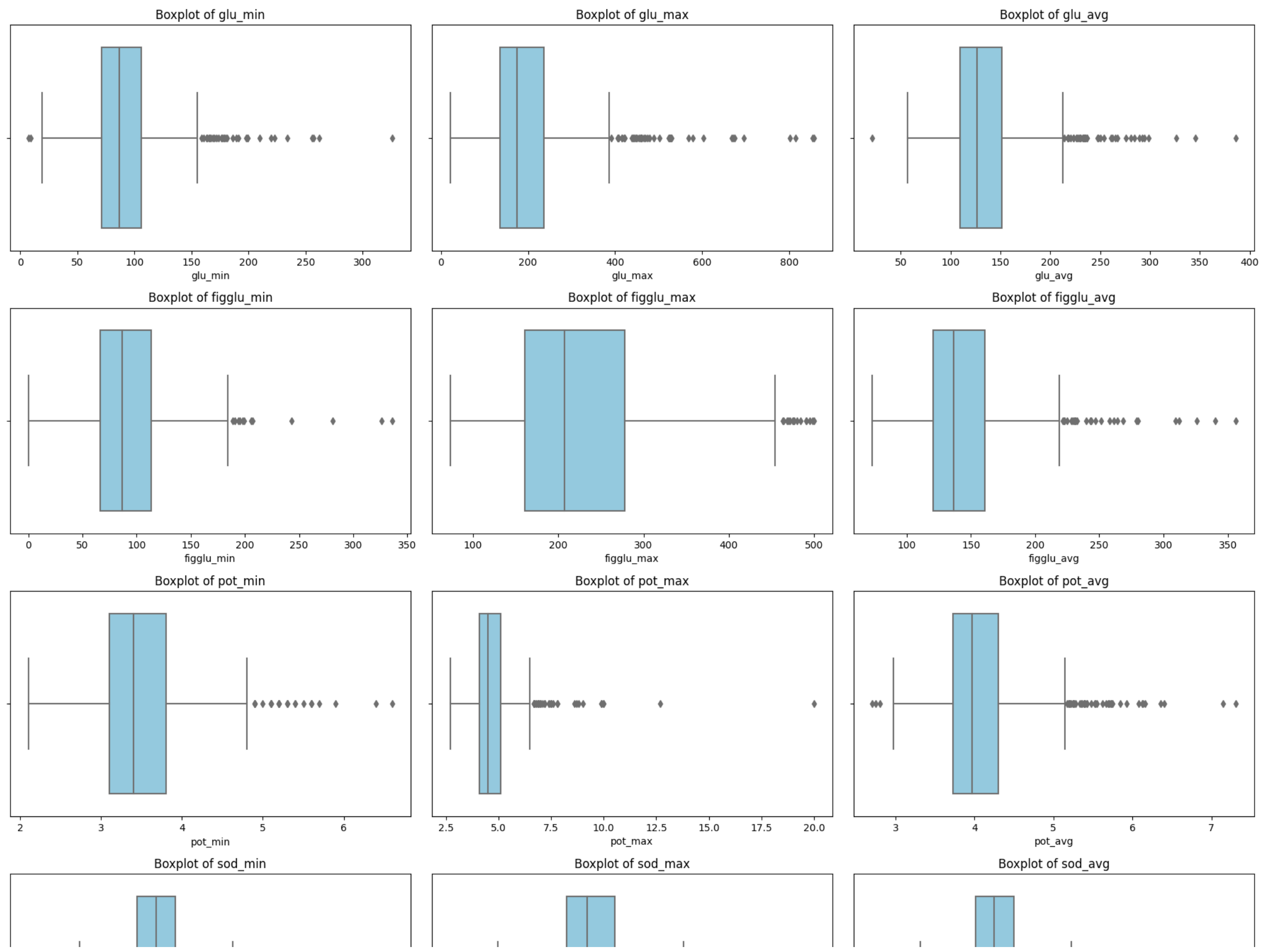
Most values follows normal distribution and shows a strong correlation with the distribution of the values.

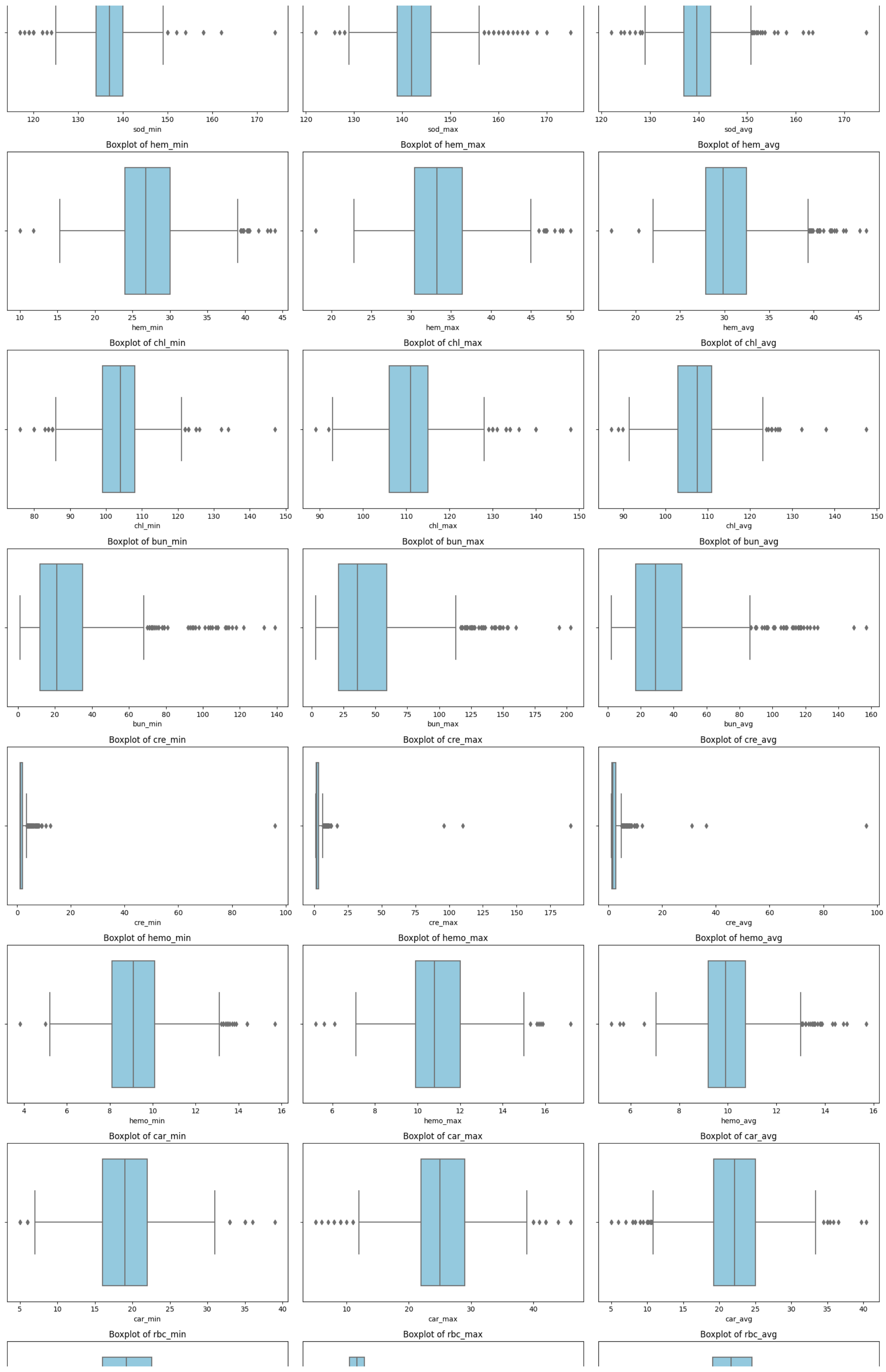
Hea_min, as an example, with extremely low values will clearly lead to death.

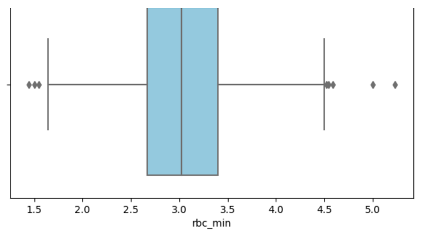
Box plots to identify ourliers

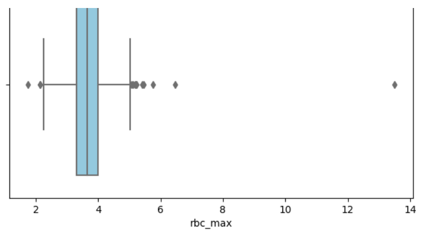
Further steps will need to deal with the outliers

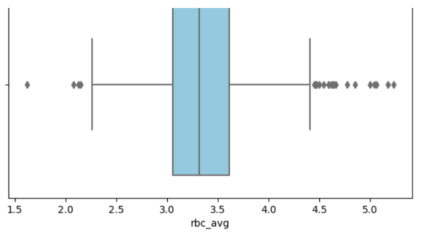
```
In [5]: import matplotlib.pyplot as plt
        import pandas as pd
        import seaborn as sns

        file_path = 'F:\STUDY\python_code\Pract_DA\prepocess\patient_sepsis_all_drop
        data = pd.read_csv(file_path)

        test_columns = data.columns[12:-3]

        rows = (len(test_columns) + 2) // 3

        fig, axes = plt.subplots(rows, 3, figsize=(18, rows * 4))
        axes = axes.flatten()

        for i, col in enumerate(test_columns):
            if data[col].dtype in ['float64', 'int64']:  # Numeric data: Vertical bo
                sns.boxplot(data=data, x=col, ax=axes[i], color='skyblue')
                axes[i].set_title(f'Boxplot of {col}')
            else:  # Categorical data can't be used in boxplots, so we skip those ca
                axes[i].text(0.5, 0.5, f"{col} is not numeric",
                             ha='center', va='center', fontsize=12)
                axes[i].set_axis_off()

        for j in range(i + 1, len(axes)):
            axes[j].set_visible(False)

        plt.tight_layout()
        plt.show()
```



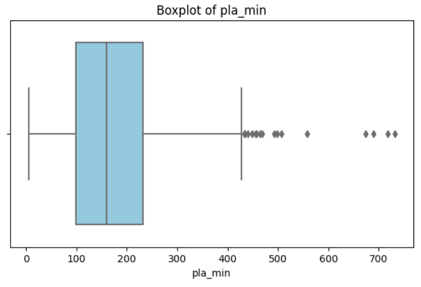

sod_min
sod_max
sod_avg
Boxplot of hem_min
Boxplot of hem_max
Boxplot of hem_avg
hem_min
hem_max
hem_avg
Boxplot of chl_min
Boxplot of chl_max
Boxplot of chl_avg
chl_min
chl_max
chl_avg
Boxplot of bun_min
Boxplot of bun_max
Boxplot of bun_avg
bun_min
bun_max
bun_avg
Boxplot of cre_min
Boxplot of cre_max
Boxplot of cre_avg
cre_min
cre_max
cre_avg
Boxplot of hemo_min
Boxplot of hemo_max
Boxplot of hemo_avg
hemo_min
hemo_max
hemo_avg
Boxplot of car_min
Boxplot of car_max
Boxplot of car_avg
car_min
car_max
car_avg
Boxplot of rbc_min
Boxplot of rbc_max
Boxplot of rbc_avg



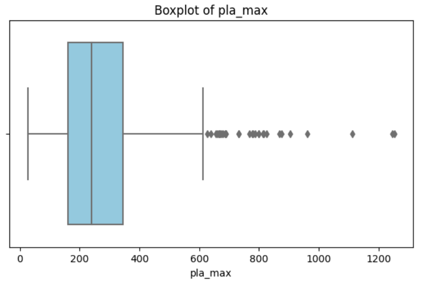




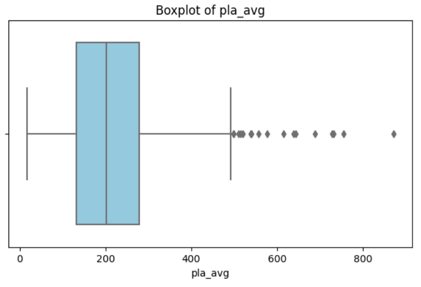
Boxplot of pla_min



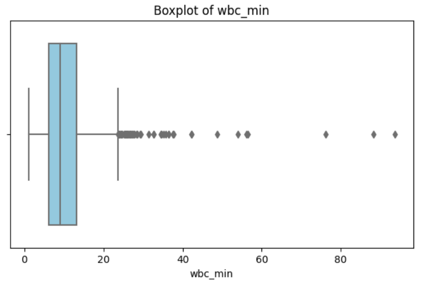
Boxplot of pla_max



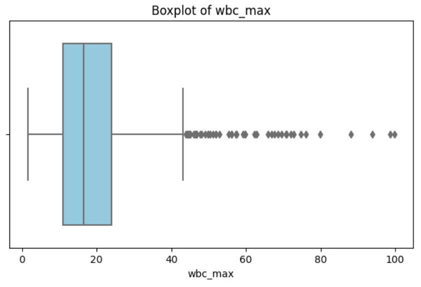
Boxplot of pla_avg



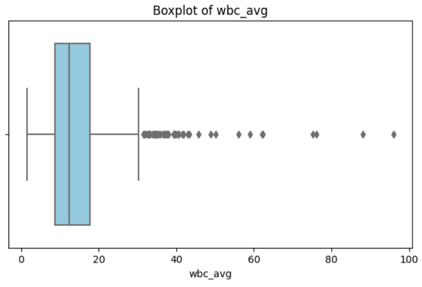
Boxplot of wbc_min



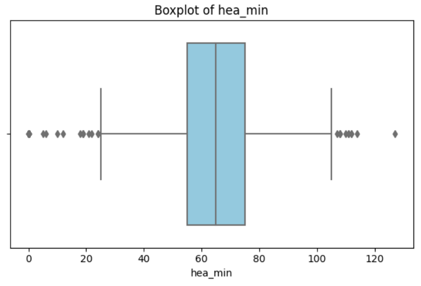
Boxplot of wbc_max



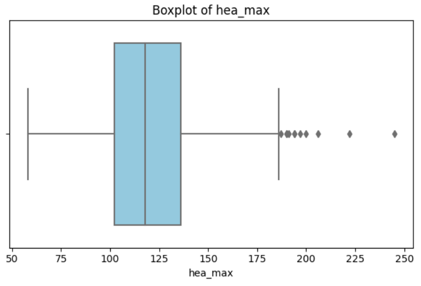
Boxplot of wbc_avg



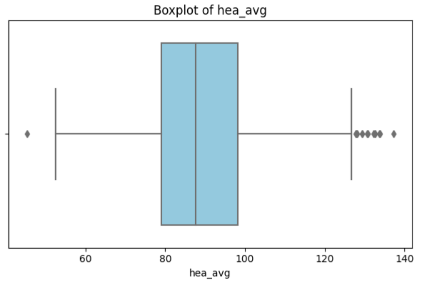
Boxplot of hea_min



Boxplot of hea_max



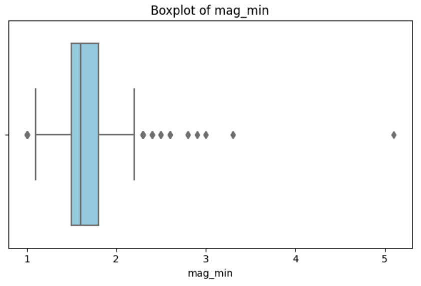
Boxplot of hea_avg



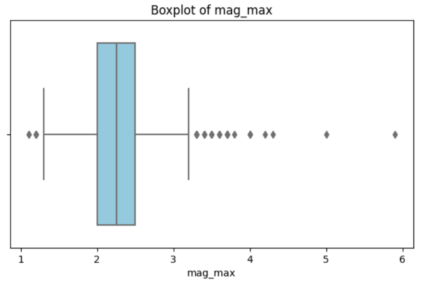
hear_mode is not numeric

hear_last is not numeric

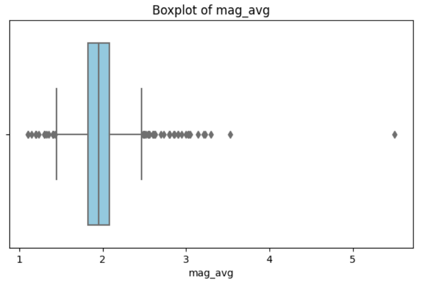
Boxplot of mag_min



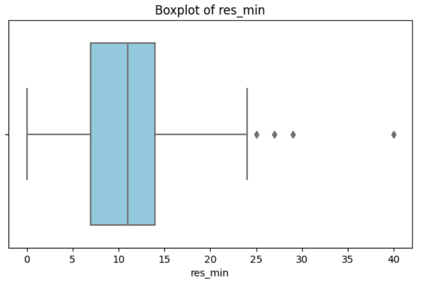
Boxplot of mag_max



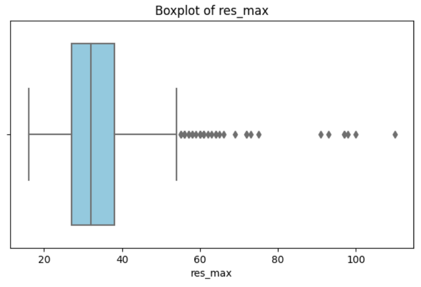
Boxplot of mag_avg



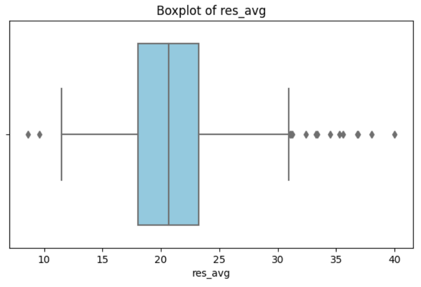
Boxplot of res_min



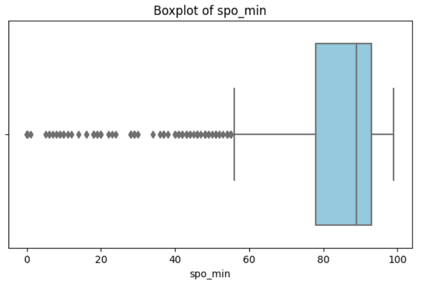
Boxplot of res_max



Boxplot of res_avg



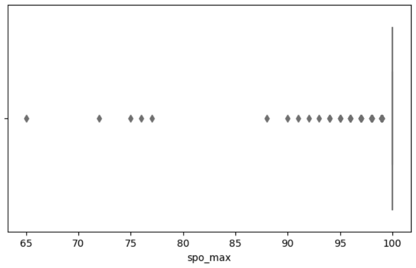
Boxplot of spo_min

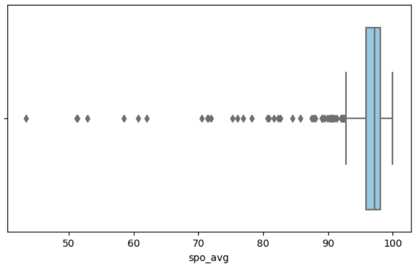


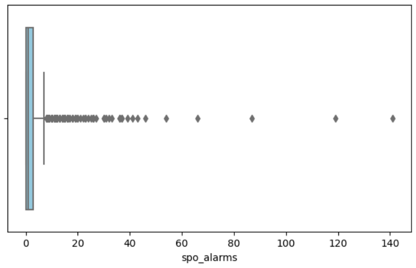
Boxplot of spo_max

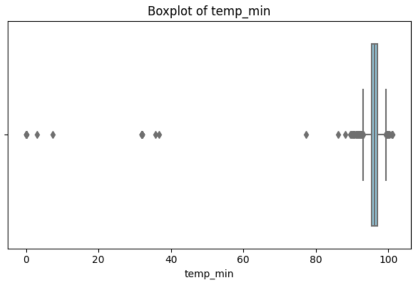
Boxplot of spo_avg

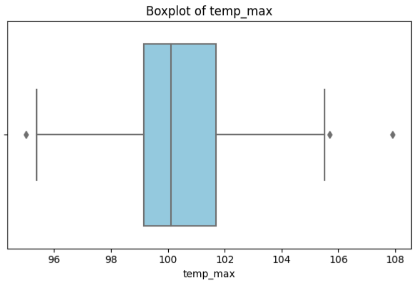
Boxplot of spo_alarms

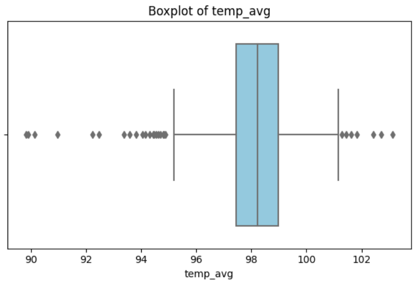

65
70
75
80
85
90
95
100
spo_max


50
60
70
80
90
100
spo_avg


0
20
40
60
80
100
120
140
spo_alarms


Boxplot of temp_min
0
20
40
60
80
100
temp_min


Boxplot of temp_max
96
98
100
102
104
106
108
temp_max


Boxplot of temp_avg
90
92
94
96
98
100
102
temp_avg

What's next?

1, Further preprocess the data (Tue-Thu this week)

Outliers and Null values

For outliers that are obviously impossible to happen, like age being over 200 years as shown in the histgraph of age, we will use average to subsitute them.

For outliers that are possible to happen, like extram values in medical test values, we tend to preserve them for now to keep as much information as possible about the patient.

For numeric null values, we will use the average to fill them.

For text null values, we will use the mode to fill them.

Another kind of outliers is text values being 'other' or 'unknown' like in **Marital stats**, we will use the mode to substitute them.

Text values

There are three features with text values : gender & marital status & ethnicity.

For gender, the distribution is balanced, so we can use 0/1 to represent male/female.

For marital status, since there are not too many different values, we tend to use one-hot encoding.

For ethnicity, the majority is white, and most death happened in white, and other ethnicities only have a very small portition comparing to white. Therefore, we tend to use 'white' and 'non-white' to substitute the current values. Then use 0/1 to represent them as well.

2, Model to be used (This weekend)

We plan to use a lighted_weighted CNN with several conv1D layers to see how it will behave on this unbalanced dataset, and then decide whether we need an outlier detection model. The details of the model will be describe below in the MLM.

Further, we will compare the CNN to other outlier detection models, like DB-Scan, One-class SVM etc. on time-consumption and recall/precision.

Structure of the MLM Workflow

The MLM workflow is represented as:

$$D \xrightarrow{\phi} D' \xrightarrow{\psi(M)} P \xrightarrow{\omega} R$$

- **D**: Raw ICU patient data
- **ϕ**: Data preprocessing morphism that transforms (D) into a clean feature set (D')
- **ψ(M)**: CNN training morphism that applies the model (M) on (D') to generate predictions (P)
- **ω**: Evaluation morphism that compares predictions (P) with ground truth to assess performance (R) using a confusion matrix

---

1. CNN Model Workflow Morphisms

1. Data Preprocessing Morphism

Steps:

1. **Data Cleaning**: Remove duplicates and fill missing values with the mean.
2. **Feature Scaling**: Apply **Z-score normalization** for consistent scaling.
3. **Encoding Categorical Features**: Convert `gender` and `ethnicity` using **Label Encoding**.
4. **Reshaping Data**: Reshape data for CNN input (adding a channel dimension).

$$\phi(D) = D'$$

---

2. CNN Training Morphism

A 1D Convolutional Neural Network is used to learn patterns from the ICU data and predict patient mortality.

Model Architecture:

- **Input Layer**: Reshaped data (samples, timesteps, 1).
- **Conv1D Layer**: Extracts feature patterns across input data.
- **MaxPooling1D Layer**: Reduces the dimensionality and retains essential information.
- **Fully Connected Layers**: Final dense layers for classification.
- **Output Layer**: Sigmoid activation for binary classification (0 = survived, 1 = died).

---

2. Evaluation Morphism

The confusion matrix helps us evaluate the model's performance by focusing on the recall. This is particularly important in medical applications, where false negatives (patients predicted to survive but who actually die) can have severe consequences.