

PATIENTS

# PATIENTS

## Genel Bakış\_

In [1]:

```
# Gerekli kütüphaneleri içe aktar
```

```
import pandas as pd
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
df_pat=pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database-demo-1.4/PATIENTS.csv')
```

```
#ilk satırlardan bazılarına bakalım
```

```
df_pat.head()
```

	row_id	subject_id	gender	dob	dod	dod_hosp	dod_ssn	expire_flag
0	9467	10006	F	2094-03-05 00:00:00	2165-08-12 00:00:00	2165-08-12 00:00:00	2165-08-12 00:00:00	1
1	9472	10011	F	2090-06-05 00:00:00	2126-08-28 00:00:00	2126-08-28 00:00:00	NaN	1
2	9474	10013	F	2038-09-03 00:00:00	2125-10-07 00:00:00	2125-10-07 00:00:00	2125-10-07 00:00:00	1
3	9478	10017	F	2075-09-21 00:00:00	2152-09-12 00:00:00	NaN	2152-09-12 00:00:00	1
4	9479	10019	M	2114-06-20 00:00:00	2163-05-15 00:00:00	2163-05-15 00:00:00	2163-05-15 00:00:00	1

Sütun Adı	Anlamı
row_id	Teknik ID, modelde kullanılmaz.
subject_id	Hastayı benzersiz tanımlar.
gender	Cinsiyet bilgisi (F veya M) – demografik analizde önemli.
dob	Doğum tarihi. Yaş hesabı için kullanılır.
dod	Ölüm tarihi (genel).
dod_hosp	Hastanede öldüyse ölüm zamanı.
dod_ssn	SSN kayıtlarına göre ölüm zamanı (tutarsızlık kontrolü yapılabilir).
expire_flag	Hasta öldü mü? (1 = evet, 0 = hayatta) – model için hedef değişken olarak da kullanılabilir.

#### Patients Tablosunun Projede Önemi

##### 1. row\_id (Teknik ID, modelde kullanılmaz)

Veri kayıtlarının benzersiz kimliği. Model eğitimi için gerekli değil, ancak veri yönetimi ve hata ayıklamada faydalı.

##### 1. subject\_id (Hastayı benzersiz tanımlar)

Hastaları diğer veri setleriyle ilişkilendirmek için temel anahtar. Farklı tablolardaki bilgileri birleştirmek ve hastaya özgü analiz yapmak için zorunlu.

##### 1. gender (Cinsiyet bilgisi: F veya M)

Demografik ve biyolojik farklılıkları modellemek için önemli. Cinsiyet, hastalık riskleri ve sepsis sürecini etkileyebilir, bu yüzden modelde öznitelik olarak değerlendirilebilir.

#### 1. dob (Doğum tarihi)

Hastanın yaşını hesaplamak için kullanılır. Yaş, sepsis riski ve mortalite gibi klinik sonuçları etkileyen kritik bir faktördür.

Doğum tarihi üzerinden yaş bilgisi çıkarılarak risk modeline dahil edilir.

#### 1. dod (Ölüm tarihi - genel)

Hastanın hayatını kaybettiği tarih. Ölüm zamanlaması analizlerde, özellikle zaman serisi ve sağkalım analizlerinde faydalıdır.

#### 1. dod\_hosp (Hastanede öldüyse ölüm zamanı)

Hastanede ölüm olup olmadığını ve zamanını gösterir. Hastane içi mortalite analizlerinde kullanılır. Modelin hedeflerinden biri olabilir.

#### 1. dod\_ssn (SSN kayıtlarına göre ölüm zamanı)

Ölüm bilgisinin alternatif kaynağı. Veri kalitesi ve tutarlılık kontrolü için önemlidir. Eksik veya çelişkili verilere karşı güvenilirlik sağlar.

#### 1. expire\_flag (Hasta öldü mü? 1=Evet, 0=Hayır)

En kritik hedef değişkenlerden biridir. Hastane içi ölüm riskini modellemek isteyen projelerde direkt kullanılır.

Sepsis gibi kritik hastalıkların mortalite üzerine etkisini analiz etmek için temel çıktı değişkenidir.

Özet Patients tablosu, hastaya dair temel demografik bilgileri, yaş, cinsiyet gibi modelde kullanılacak önemli öznitelikleri ve ölüm durumunu içerir. Bu bilgiler, özellikle sepsis riski ve ölüm tahmin modellerinin doğruluğunu artırmak için kritik öneme sahiptir. Ayrıca, diğer tablolarla (admissions, icustays vb.) ilişkilendirilerek hastaya özgü kapsamlı bir profil oluşturulmasını sağlar.

## Veri kontrolü

```
# Eksik veri ve tip kontrolü
```

```
print(df_pat.info())
```

```
# Eksik değer oranı
```

```
missing = df_pat.isnull().sum()
```

```
missing_pct = (missing / len(df_pat) * 100).round(2)
```

```
pd.DataFrame({'Eksik Değer Sayısı': missing, 'Oran (%)': missing_pct})
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 100 entries, 0 to 99
```

```
Data columns (total 8 columns):
```

#	Column	Non-Null Count	Dtype
0	row_id	100 non-null	int64
1	subject_id	100 non-null	int64
2	gender	100 non-null	object
3	dob	100 non-null	object
4	dod	100 non-null	object
5	dod_hosp	70 non-null	object
6	dod_ssn	77 non-null	object
7	expire_flag	100 non-null	int64

```
dtypes: int64(3), object(5)
```

```
memory usage: 6.4+ KB
```

```
None
```

Out[2]:

	Eksik Değer Sayısı	Oran (%)
row_id	0	0.0
subject_id	0	0.0
gender	0	0.0

dob	0	0.0
dod	0	0.0
dod_hosp	30	30.0
dod_ssn	23	23.0
expire_flag	0	0.0

Modelin okunabilirliğini artırmak için hastanede ölmüş veya ölmemiş ibaresi ekleyeceğim

In [3]:

```
df_pat['died_in_hospital'] = df_pat['dod_hosp'].notnull().astype(int)
#1 ise hastanede ölmüş 0 ise dışarda
df_pat.head()
```

Out[3]:

	row_id	subject_id	gender	dob	dod	dod_hosp	dod_ssn	expire_flag	died_in_hospital
0	9467	10006	F	2094-03-05 00:00:00	2165-08-12 00:00:00	2165-08-12 00:00:00	2165-08-12 00:00:00	1	1
1	9472	10011	F	2090-06-05 00:00:00	2126-08-28 00:00:00	2126-08-28 00:00:00	NaN	1	1

2	9474	10013	F	2038-09-03 00:00:00	2125-10-07 00:00:00	2125-10-07 00:00:00	2125-10-07 00:00:00	1	1
3	9478	10017	F	2075-09-21 00:00:00	2152-09-12 00:00:00	NaN	2152-09-12 00:00:00	1	0
4	9479	10019	M	2114-06-20 00:00:00	2163-05-15 00:00:00	2163-05-15 00:00:00	2163-05-15 00:00:00	1	1

! Verilen demo setimiz zaten sadece ölmüş hastaları (expire\_flag=1) içerdiği için ayrı bir yorumlama yapmaya gerek yok

In [4]:

```
# Önemli kategorik sütunların dağılımı
cols = ['gender', 'died_in_hospital']
for col in cols:
    print(f"\n📍 {col.upper()} değeri dağılımı:")

    print(df_pat[col].value_counts(dropna=False))
```

📍 GENDER değeri dağılımı:

```
gender
F      55
M      45
Name: count, dtype: int64
```

📍 DIED\_IN\_HOSPITAL değeri dağılımı:

```
died_in_hospital
1      70
0      30
Name: count, dtype: int64
```

## Verilerin yorumlanması

1) sepsis ile yaş arası güçlü bir etkileşim bulunduğu için önce bir yaş hesaplaması ardından sepsisli hastalar ve diğer hastaların yaş karşılaştırmalarını yapacağım.

not:tabloda veri gizliliği açısından doğum ve ölüm tarihleri değiştirilmiş ama yaş bilgisinin sabit tutulmuş olması gerekiyor

In [5]:

```
# Dosyaları oku

patients =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/PATIENTS.csv')
admissions =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/ADMISSIONS.csv')

# Tarihleri datetime yap
patients['dob'] = pd.to_datetime(patients['dob'], errors='coerce')
admissions['admittime'] = pd.to_datetime(admissions['admittime'],
errors='coerce')

# İki tabloyu subject_id ile birleştir
merged_df = pd.merge(admissions, patients, on='subject_id', how='left')

# Yaş hesaplama (örnek olarak sadece yıl farkı)
merged_df['age'] = merged_df['admittime'].dt.year -
merged_df['dob'].dt.year

# Sonuçları görelim
print(merged_df[['subject_id', 'admittime', 'dob', 'age']].head())
```

	subject_id	admittime	dob	age
0	10006	2164-10-23 21:09:00	2094-03-05	70
1	10011	2126-08-14 22:32:00	2090-06-05	36
2	10013	2125-10-04 23:36:00	2038-09-03	87
3	10017	2149-05-26 17:19:00	2075-09-21	74
4	10019	2163-05-14 20:43:00	2114-06-20	49

In [6]:

```
# Dosyaları oku
```



```

patients =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/PATIENTS.csv')
admissions =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/ADMISSIONS.csv')

# Tarihleri datetime yap
patients['dob'] = pd.to_datetime(patients['dob'], errors='coerce')
admissions['admittime'] = pd.to_datetime(admissions['admittime'],
errors='coerce')

# İki tabloyu subject_id ile birleştir
merged_df = pd.merge(admissions, patients, on='subject_id', how='left')

# Yaş hesaplama (örnek olarak sadece yıl farkı)

merged_df['age'] = merged_df['admittime'].dt.year -
merged_df['dob'].dt.year

# Sonuçları görelim
print(merged_df[['subject_id', 'admittime', 'dob', 'age']].head())

```

	subject_id	admittime	dob	age
0	10006	2164-10-23 21:09:00	2094-03-05	70
1	10011	2126-08-14 22:32:00	2090-06-05	36
2	10013	2125-10-04 23:36:00	2038-09-03	87
3	10017	2149-05-26 17:19:00	2075-09-21	74
4	10019	2163-05-14 20:43:00	2114-06-20	49

```

#şimdi sepsisli veya değil olarak hastaları etiketliyelim

#Diagnoses dosyasını yükle
icd =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/DIAGNOSES_ICD.csv')

# Sepsis ICD-9 kodları (kontrol et, gerekirse güncelle)
sepsis_codes = ['99591', '99592', '78552']

# Sepsis tanısı konan hastaları filtrele
icd_sepsis = icd[icd['icd9_code'].isin(sepsis_codes)]

```

```

# Örnek: sepsisli hastaların subject_id'leri
sepsis_subject_ids = icd_sepsis['subject_id'].unique()

# merged_df'de sepsis etiketi ekle
merged_df['sepsis'] =
merged_df['subject_id'].isin(sepsis_subject_ids).astype(int)

# Sepsisli ve sepsisli olmayanlar
sepsisli_hastalar = merged_df[merged_df['sepsis'] == 1]
sepsisli_olmayanlar = merged_df[merged_df['sepsis'] == 0]
# 100 yaş üstünü filtreleyelim
sepsisli_hastalar_filtered = sepsisli_hastalar[sepsisli_hastalar['age']
<= 100]

sepsisli_olmayanlar_filtered =
sepsisli_olmayanlar[sepsisli_olmayanlar['age'] <= 100]

print(f"Sepsisli hasta sayısı: {len(sepsisli_hastalar_filtered)}")
print(f"Sepsisli olmayan hasta sayısı:
{len(sepsisli_olmayanlar_filtered)}")

```

Sepsisli hasta sayısı: 38

Sepsisli olmayan hasta sayısı: 82

```

print("Sepsisli hastaların yaş dağılımı:")
print(sepsisli_hastalar_filtered['age'].describe())

print("\nSepsisli olmayan hastaların yaş dağılımı:")
print(sepsisli_olmayanlar_filtered['age'].describe())

```

Sepsisli hastaların yaş dağılımı:

```

count    38.000000
mean     66.447368
std      13.083304
min      27.000000
25%      63.250000
50%      65.500000
75%      76.000000
max      88.000000
Name: age, dtype: float64

```

Sepsisli olmayan hastaların yaş dağılımı:

```
count    82.000000
mean     69.878049
std      17.179797
min      17.000000
25%      62.500000
50%      76.000000
75%      82.000000
max      89.000000
Name: age, dtype: float64
```

In [9]:

```
plt.figure(figsize=(12,6))
```

```
# Histogram
```

```
plt.subplot(1,2,1)
sns.histplot(sepsisli_hastalar_filtered['age'], color='red',
label='Sepsisli', kde=True, stat="density", linewidth=0)
sns.histplot(sepsisli_olmayanlar_filtered['age'], color='blue',
label='Sepsisli Olmayan', kde=True, stat="density", linewidth=0,
alpha=0.6)
plt.legend()
plt.title('Yaş Dağılımı (Histogram)')
plt.xlabel('Yaş')
plt.ylabel('Yoğunluk')
```

```
# Boxplot
```

```
plt.subplot(1,2,2)
sns.boxplot(data=[sepsisli_hastalar_filtered['age'],
sepsisli_olmayanlar_filtered['age']], palette=['red', 'blue'])
plt.xticks([0,1], ['Sepsisli', 'Sepsisli Olmayan'])
plt.title('Yaş Dağılımı (Boxplot)')
```

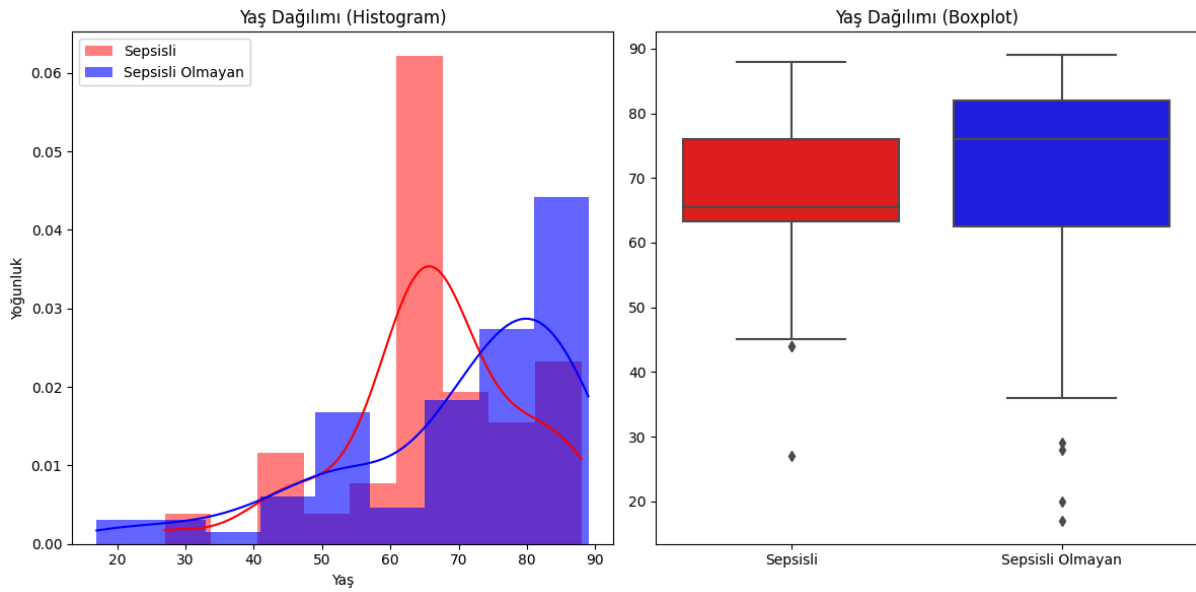
```
plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
```

in a future version. Convert inf values to NaN before operating instead.

```
with pd.option_context('mode.use_inf_as_na', True):  
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119:  
FutureWarning: use_inf_as_na option is deprecated and will be removed  
in a future version. Convert inf values to NaN before operating  
instead.
```

```
with pd.option_context('mode.use_inf_as_na', True):
```



!! Sepsisli hastalar sepsisten ölmek zorunda değil ama elimizdeki veriler hala kıymetli çünkü sepsise yol açan faktörleri inceliyoruz. sepsis tanısı konmuş bir hasta olması bizim için hala değerli

## # Gerekli sütunları seçerek birleştir

```
summary_df = pd.merge( merged_df[['subject_id', 'gender', 'age']], dt_pat[['subject_id',  
'died_in_hospital']], on='subject_id', how='left' )
```

## Sonuçları görüntüle

```
print(summary_df.head())
```


Final Tablosu Gerekli bilgilerin olduğu bir tablo çizelim

In [10]:

```
# Gerekli sütunları seçerek birleştir
summary_df = pd.merge(
    merged_df[['subject_id', 'gender', 'age']],
    df_pat[['subject_id', 'died_in_hospital']],
    on='subject_id',
    how='left'
)

# Sonuçları görüntüle
print(summary_df.head())
```

	subject_id	gender	age	died_in_hospital
0	10006	F	70	1
1	10011	F	36	1
2	10013	F	87	1
3	10017	F	74	0
4	10019	M	49	1

 PATIENTS.csv İncelemesi – Kapanış Notları Bu tabloda, hastaların temel demografik bilgileri ve ölüm durumları yer alıyordu. Yapılan analizlerle:

Cinsiyet (gender), yaş (age) ve hastanede ölüm (died\_in\_hospital) gibi önemli özellikleri çıkardık.

Doğum tarihinden (dob) ve yatış tarihinden (admittime) yola çıkarak yaş hesaplaması yaptık.

Ardından, hastanın hastanede hayatını kaybedip kaybetmediğini belirleyen died\_in\_hospital değişkeniyle birleştirerek temiz ve analiz odaklı bir tablo oluşturduk.

 Bu Tablonun Modeldeki Yeri PATIENTS.csv, özellikle şu alanlarda modelinize güç katabilir:

Demografik risk analizi (yaş ve cinsiyetin sepsis veya mortalite üzerindeki etkileri),

Ölüm tahmini gibi hedef değişken tanımı (expire\_flag veya died\_in\_hospital),

Diğer klinik tablolarla entegre edilerek daha zengin modeller geliştirme.

# LABEVENTS

# LABEVENTS

## GENEL BAKIŞ\_

In [1]:

```
#Gerekli kütüphaneleri yükleyelim
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
#Sütunlarımıza ve örnek satırlara bakalım
file_path =
'/kaggle/input/clinical-dataset/mimic-iii-clinical-database-demo-1.4/LA
BEVENTS.csv'
lab = pd.read_csv(file_path)
lab.head()
```

```
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
8: RuntimeWarning: invalid value encountered in greater
    has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
9: RuntimeWarning: invalid value encountered in less
    has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
9: RuntimeWarning: invalid value encountered in greater
    has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
```

Out[1]:

	row_id	subject_id	hadm_id	item_id	charttime	value	valuenum	valueuom	flag
--	--------	------------	---------	---------	-----------	-------	----------	----------	------

0	624456 3	10006	NaN	5086 8	2164-09-24 20:21:00	19	19.0	mEq/L	NaN
1	624456 4	10006	NaN	5088 2	2164-09-24 20:21:00	27	27.0	mEq/L	NaN
2	624456 5	10006	NaN	5089 3	2164-09-24 20:21:00	10.0	10.0	mg/dL	NaN
3	624456 6	10006	NaN	5090 2	2164-09-24 20:21:00	97	97.0	mEq/L	NaN
4	624456 7	10006	NaN	5091 2	2164-09-24 20:21:00	7.0	7.0	mg/dL	abnorm al

In [2]:

```
#Eksik değerleri bulalım
lab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76074 entries, 0 to 76073
Data columns (total 9 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   row_id      76074 non-null  int64
1   subject_id  76074 non-null  int64
2   hadm_id     61812 non-null  float64
3   itemid      76074 non-null  int64
4   charttime   76074 non-null  object
5   value       76069 non-null  object
6   valuenum    67030 non-null  float64
7   valueuom    66669 non-null  object
```



```
8    flag          29737 non-null object
dtypes: float64(2), int64(3), object(4)
memory usage: 5.2+ MB
```

In [3]:

```
# ICD-9 kodları tablosunu yükleyelim
icd =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/DIAGNOSES_ICD.csv')

# Sepsis ile ilgili kodlar (sepsisle ilgili daha fazla kod olabilir
araştırılması lazım)
sepsis_codes = ['99591', '99592', '78552']

# Sepsis kodlarına sahip satırları seçelim
icd_sepsis = icd[icd['icd9_code'].isin(sepsis_codes)]

# Sepsis hastalarının hadm_id ve subject_id'lerini bulalım
sepsis_hadm_ids = icd_sepsis['hadm_id'].unique()
sepsis_subject_ids = icd_sepsis['subject_id'].unique()

print(f"Toplam sepsisli hasta sayısı: {len(sepsis_subject_ids)}")

print(f"Toplam sepsisli yatış sayısı: {len(sepsis_hadm_ids)}")
```

```
Toplam sepsisli hasta sayısı: 23
Toplam sepsisli yatış sayısı: 35
```

Normal şartlar altında labeventsdeki verilerle sepsis tanısı konulmuş mu diye hadm\_id den bakmam daha doğru olurdu ama hadm\_id boş bu kısımda o yüzden subject\_id ler ile karşılaştıracam

item\_id her biri farklı bir testi belirtir sepsisle ilgili olanları yorumlayacağım diğer testlere de bakılabilir

## Labitems ile karşılaştırma

In [4]:

```
unique_itemids = lab['itemid'].unique()
print(unique_itemids)
```

```
[50868 50882 50893 50902 50912 50931 50960 50970 50971 50983 51006 51009
51137 51144 51146 51200 51221 51222 51233 51237 51244 51246 51248 51249
51250 51252 51254 51256 51260 51265 51267 51268 51274 51275 51277 51279
51294 51301 50924 50952 50953 50998 51240 51266 50979 50861 50862 50863
50878 50885 50935 50954 51196 51143 51251 51255 51257 51213 51214 50800
50808 50820 50852 50941 50813 50889 50909 50930 50956 50976 51087 51103
51463 51464 51466 51476 51478 51484 51486 51487 51491 51492 51493 51498
51506 51508 51514 51516 51519 50919 51283 50867 50910 50911 50933 51003
50802 50804 50818 50821 50822 50825 50903 50904 50905 50907 51000 50887
50810 50811 50809 51114 51116 51117 51118 51120 51125 51127 51128 51290
50812 51497 51512 51427 51431 51436 51438 51439 51287 51385 51428 51288
50866 50883 50884 51134 51151 50964 50939 50816 51094 51108 51278 50940
51204 51297 50856 50879 50880 50922 50981 50999 51071 51074 51075 51079
51090 51092 50908 50993 51002 51082 51093 51100 51462 50817 50827 50828
50854 50855 50920 51523 50925 51010 50857 50801 50819 50823 50826 50975
50824 50815 51086 51098 51102 50948 50949 50950 50951 50974 50929 51296
51218 51505 51076 51078 51104 51474 50995 51001 51501 50963 51479 51482
51197 50806 50965 50967 50968 50969 50927 51507 51032 50994 50803 50864
50835 50849 50937 50942 50943 50917 51489 50805 50814 50966 51008 51014
51017 51018 51351 51355 51360 51362 51363 51235 51243 51150 51097 51007
51145 50900 51216 51099 50881 51229 51232 51206 51298 51299 50973 51148
51219 51262 51269 51095 51302 51347 51352 51518 51069 51070 51419 51434
51429 51513 51289 51339 51342 51072 51073 51096 51263 51136 51211 50836
50838 50890 50891 51199 50986 51517 50841 50842 50843 51122 51422 51091
51030 51041 51042 51053 51054 51059 51446 51448 51450 51455 51457 51458
50831 50915 51046 51444 51447 51453 51284 51460 51292 50955 50873 51369
51373 51375 51379 51382 51384 51388 51398 51399 51400 51402 51404 51411
51412 51416 51420 51423 51424 51425 51426 50946 50926 50958 50988 51259
51152 51154 51155 51156 51158 51159 51164 51167 51168 51176 51177 51178
51180 51182 51183 51184 51188 51191 51192 51193 51194 51217 51230 51234
51236 51238 51239 51303 51305 51306 51307 51309 51310 51313 51314 51319
51320 51321 51323 51324 51325 51326 51328 51331 51332 51333 51334 51336
51337 51338 51340 51341 51105 51130 51131 51132 51133 51181 51245 51300
50944 51358 51276 50961 51015 51198 51315 51317 51335 51537 51047 50899
50871 50997 51469 50853 51147 51124 50874 50876 50895 50896 51005 51533
51376 51383 51049 51052 51060 51026 51028 50898 51264 50978 51529 50892
50932 51025 51034 51035 51043 51044 51247]
```

In [5]:

```
# D_LABITEMS dosyasını oku
d_labitems =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_LABITEMS.csv')

# Örnek unique_itemids listesi
unique_itemid=print(d_labitems['itemid'].unique())

print(50813 in d_labitems['itemid'].values)
```

```
[50800 50801 50802 50803 50804 50805 50806 50807 50808 50809 50810 50811
50812 50813 50814 50815 50816 50817 50818 50819 50820 50821 50822 50823
50824 50825 50826 50827 50828 50829 50830 50831 50832 50833 50834 50835
50836 50837 50838 50839 50840 50841 50842 50843 50844 50845 50846 50847
50848 50849 50850 50851 50852 50853 50854 50855 50856 50857 50858 50859
50860 50861 50862 50863 50864 50865 50866 50867 50868 50869 50870 50871
50872 50873 50874 50875 50876 50877 50878 50879 50880 50881 50882 50883
50884 50885 50886 50887 50888 50889 50890 50891 50892 50893 50894 50895
50896 50897 50898 50899 50900 50901 50902 50903 50904 50905 50906 50907]
```

50908 50909 50910 50911 50912 50913 50914 50915 50916 50917 50918 50919  
50920 50921 50922 50923 50924 50925 50926 50927 50928 50929 50930 50931  
50932 50933 50934 50935 50936 50937 50938 50939 50940 50941 50942 50943  
50944 50945 50946 50947 50948 50949 50950 50951 50952 50953 50954 50955  
50956 50957 50958 50959 50960 50961 50962 50963 50964 50965 50966 50967  
50968 50969 50970 50971 50972 50973 50974 50975 50976 50977 50978 50979  
50980 50981 50982 50983 50984 50985 50986 50988 50989 50990 50991 50992  
50993 50994 50995 50996 50997 50998 50999 51000 51001 51002 51003 51004  
51005 51006 51007 51008 51009 51010 51011 51012 51013 51014 51015 51016  
51017 51018 51019 51020 51021 51022 51023 51024 51025 51026 51027 51028  
51029 51030 51031 51032 51033 51034 51035 51036 51037 51038 51039 51040  
51041 51042 51043 51044 51045 51046 51047 51048 51049 51050 51051 51052  
51053 51054 51055 51056 51057 51058 51059 51060 51061 51062 51063 51064  
51065 51066 51067 51068 51069 51070 51071 51072 51073 51074 51075 51076  
51077 51078 51079 51080 51081 51082 51083 51084 51085 51086 51087 51088  
51089 51090 51091 51092 51093 51094 51095 51096 51097 51098 51099 51100  
51101 51102 51103 51104 51105 51106 51107 51108 51109 51110 51111 51112  
51113 51114 51115 51116 51117 51118 51119 51120 51121 51122 51123 51124  
51125 51126 51127 51128 51129 51130 51131 51132 51133 51134 51135 51136  
51137 51138 51139 51140 51141 51142 51143 51144 51145 51146 51147 51148  
51149 51150 51151 51152 51153 51154 51155 51156 51157 51158 51159 51160  
51161 51162 51163 51164 51165 51166 51167 51168 51169 51170 51171 51172  
51173 51174 51175 51176 51177 51178 51179 51180 51181 51182 51183 51184  
51185 51186 51187 51188 51189 51190 51191 51192 51193 51194 51195 51196  
51197 51198 51199 51200 51201 51202 51203 51204 51205 51206 51207 51208  
51209 51210 51211 51212 51213 51214 51215 51216 51217 51218 51219 51220  
51221 51222 51223 51224 51225 51226 51227 51228 51229 51230 51231 51232  
51233 51234 51235 51236 51237 51238 51239 51240 51241 51242 51243 51244  
51245 51246 51247 51248 51249 51250 51251 51252 51253 51254 51255 51256  
51257 51258 51259 51260 51261 51262 51263 51264 51265 51266 51267 51268  
51269 51270 51271 51272 51273 51274 51275 51276 51277 51278 51279 51280  
51281 51282 51283 51284 51285 51286 51287 51288 51289 51290 51291 51292  
51293 51294 51295 51296 51297 51298 51299 51300 51301 51302 51303 51304  
51305 51306 51307 51308 51309 51310 51311 51312 51313 51314 51315 51316  
51317 51318 51319 51320 51321 51322 51323 51324 51325 51326 51327 51328  
51329 51330 51331 51332 51333 51334 51335 51336 51337 51338 51339 51340  
51341 51342 51343 51344 51345 51346 51347 51348 51349 51350 51351 51352  
51353 51354 51355 51356 51357 51358 51359 51360 51361 51362 51363 51364  
51365 51366 51367 51368 51369 51370 51371 51372 51373 51374 51375 51376  
51377 51378 51379 51380 51381 51382 51383 51384 51385 51386 51387 51388  
51389 51390 51391 51392 51393 51394 51395 51396 51397 51398 51399 51400  
51401 51402 51403 51404 51405 51406 51407 51408 51409 51410 51411 51412  
51413 51414 51415 51416 51417 51418 51419 51420 51421 51422 51423 51424  
51425 51426 51427 51428 51429 51430 51431 51432 51433 51434 51435 51436  
51437 51438 51439 51440 51441 51442 51443 51444 51445 51446 51447 51448  
51449 51450 51451 51452 51453 51454 51455 51456 51457 51458 51459 51460  
51461 51462 51463 51464 51465 51466 51467 51468 51469 51470 51471 51472  
51473 51474 51475 51476 51477 51478 51479 51480 51481 51482 51483 51484  
51485 51486 51487 51488 51489 51490 51491 51492 51493 51494 51495 51496  
51497 51498 51499 51500 51501 51502 51503 51504 51505 51506 51507 51508  
51509 51510 51511 51512 51513 51514 51515 51516 51517 51518 51519 51520  
51521 51522 51523 51524 51525 51526 51527 51528 51529 51530 51531 51532  
51533 51534 51535 51536 51537 51540 51541 51542 51543 51544 51545 51546  
51547 51548 51549 51550 51551 51552 51553 51554 51555]

True

**-KLİNİK ÖNEMİ YÜKSEK TESTLER D\_LABITEMS DEN GELECEK-Önemli testleri alıp  
aşağıdaki gibi karşılaştıracqız modeli geliştirmede onları kullanırız**

## Önemli değerkleri bulma ve karşılaştıırma

```
import pandas as pd
```

```

d_labitems =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_LABITEMS.csv')
lab_events =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/LABEVENTS.csv')

# LABEVENTS içinde en sık kullanılan itemid'leri bulalım
freq = lab_events['itemid'].value_counts().head(20)
print("En sık kullanılan 20 itemid ve frekansı:")
print(freq)

# Bu itemid'lerin D_LABITEMS'taki label'larını bulalım
top_itemids = freq.index.tolist()
top_labels =
d_labitems[d_labitems['itemid'].isin(top_itemids)][['itemid', 'label']]

print("\nBu itemid'lerin isimleri:")
print(top_labels)

```

En sık kullanılan 20 itemid ve frekansı:

itemid	
51221	2317
50971	2279
50983	2185
50912	2175
50902	2160
51006	2158
50882	2151
50868	2134
50931	2121
51265	2088
51222	2024
51301	2021
51249	2008
51279	2007
51250	2007
51248	2007
51277	2005
50960	1918
50893	1757
50970	1748

Name: count, dtype: int64

Bu itemid'lerin isimleri:

	itemid	label
68	50868	Anion Gap
82	50882	Bicarbonate
93	50893	Calcium, Total
102	50902	Chloride
112	50912	Creatinine
131	50931	Glucose
160	50960	Magnesium
170	50970	Phosphate
171	50971	Potassium
183	50983	Sodium
205	51006	Urea Nitrogen
420	51221	Hematocrit
421	51222	Hemoglobin
447	51248	MCH
448	51249	MCHC
449	51250	MCV
464	51265	Platelet Count
476	51277	RDW
478	51279	Red Blood Cells
500	51301	White Blood Cells

Bunların ve literatür taraması yaptıktan sonra sepsis hastalarında en çok bakılan değerleri aşağıdaki örnekte olduğu gibi karşılaştıracız. modelin eğitiminde faydalanılabilir.

*#White Blood Cells değerinin sepsis tanısı konan ve konmayan hastalarda dağılımı*

lab =

```
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database-  
demo-1.4/LABEVENTS.csv')
```

*# Sepsisli hastaların subject\_id'lerini biliyoruz:*

```
sepsis_subject_ids = icd_sepsis['subject_id'].unique()
```

*# Sepsis etiketi ekle (subject\_id bazında)*

```
lab['sepsis'] = lab['subject_id'].isin(sepsis_subject_ids).astype(int)
```

*# Artık sepsis etiketi ile laboratuvar değerlerini karşılaştırabiliriz*

```

whiteb_cells_id = 51301
whitebc = lab[(lab['itemid'] == whiteb_cells_id) &
(lab['valuenum'].notnull())]

print("Sepsisli hastalarda white blood cells değerleri:")
print(whitebc[whitebc['sepsis'] == 1]['valuenum'].describe())

print("Sepsis olmayan hastalarda white blood cells değerleri:")
print(whitebc[whitebc['sepsis'] == 0]['valuenum'].describe())

```


```


Sepsisli hastalarda white blood cells değerleri:
count      627.000000
mean        12.320255
std         7.649178
min         1.800000
25%         7.350000
50%        10.300000
75%        14.800000
max         58.600000
Name: valuenum, dtype: float64
Sepsis olmayan hastalarda white blood cells değerleri:
count      1394.000000
mean         8.335473
std          5.869265
min          0.100000
25%          5.000000
50%          7.400000
75%         10.800000
max          78.200000
Name: valuenum, dtype: float64


```

White Blood Cells (WBC) / Lökosit Nedir? Normal aralık (erişkinlerde): 4.000 - 10.000 hücre/ $\mu$ L  
(yani 4.0 - 10.0  $\times 10^9$ /L)

Yüksek WBC değeri (lökositoz): Genelde enfeksiyon, inflamasyon veya sepsis gibi durumlarda görülür.

 Sepsisli Hastalarda Ortalama: 12.32 ( $\times 10^9$ /L) Bu yükseklik, bağışıklık sisteminin enfeksiyona karşı verdiği tepkidir.

 Sepsis hastalarında WBC sıklıkla  $>12.0$  çıkar.


 Sepsis Olmayanlarda Ortalama:  $8.33 (x10^9/L)$  Bu değer, normal aralığın ortalarında ve enfeksiyon olmayan bir popülasyona uygundur.

 Literatürle Karşılaştırma:

1. Sepsis Tanı Kriterlerinden biri (SIRS/Sepsis-2):  $WBC > 12.0 x10^9/L$  veya  $< 4.0 x10^9/L$  (veya  $>10\%$  immatür formlar)

Senin verin:

Sepsisli ortalama: 12.32 

Sepsis olmayan ortalama: 8.33 

1. Klinik çalışmaların sonuçları: "WBC count in septic patients was significantly higher than in non-septic patients" (Kaynak: Singer et al., Sepsis Definitions, JAMA 2016)

 Sonuç: WBC analizleri klinik bilgiler ve literatürle örtüşüyor.

Sepsisli hastalarda WBC yüksek  $\rightarrow$  beklenen

Sepsis olmayanlarda normal sınırlarda  $\rightarrow$  beklenen

Benzer `import matplotlib.pyplot as plt`

`import seaborn as sns`

`# Sadece değerleri al`

`wbc_sepsis = whitebc[whitebc['sepsis'] == 1]['valuenum']`

`wbc_nonsepsis = whitebc[whitebc['sepsis'] == 0]['valuenum']`

`# Histogramları çiz`

```
plt.figure(figsize=(10,6))

sns.histplot(wbc_sepsis, bins=30, color='red', label='Sepsisli',
kde=True, stat="density", alpha=0.6)

sns.histplot(wbc_nonsepsis, bins=30, color='blue', label='Sepsis
Olmayan', kde=True, stat="density", alpha=0.6)

plt.title("WBC Dağılımı: Sepsisli vs Sepsis Olmayan Hastalar")

plt.xlabel("White Blood Cell Count ( $\times 10^9/L$ )")

plt.ylabel("Yoğunluk")

plt.legend()

plt.grid(True)

plt.show()
```

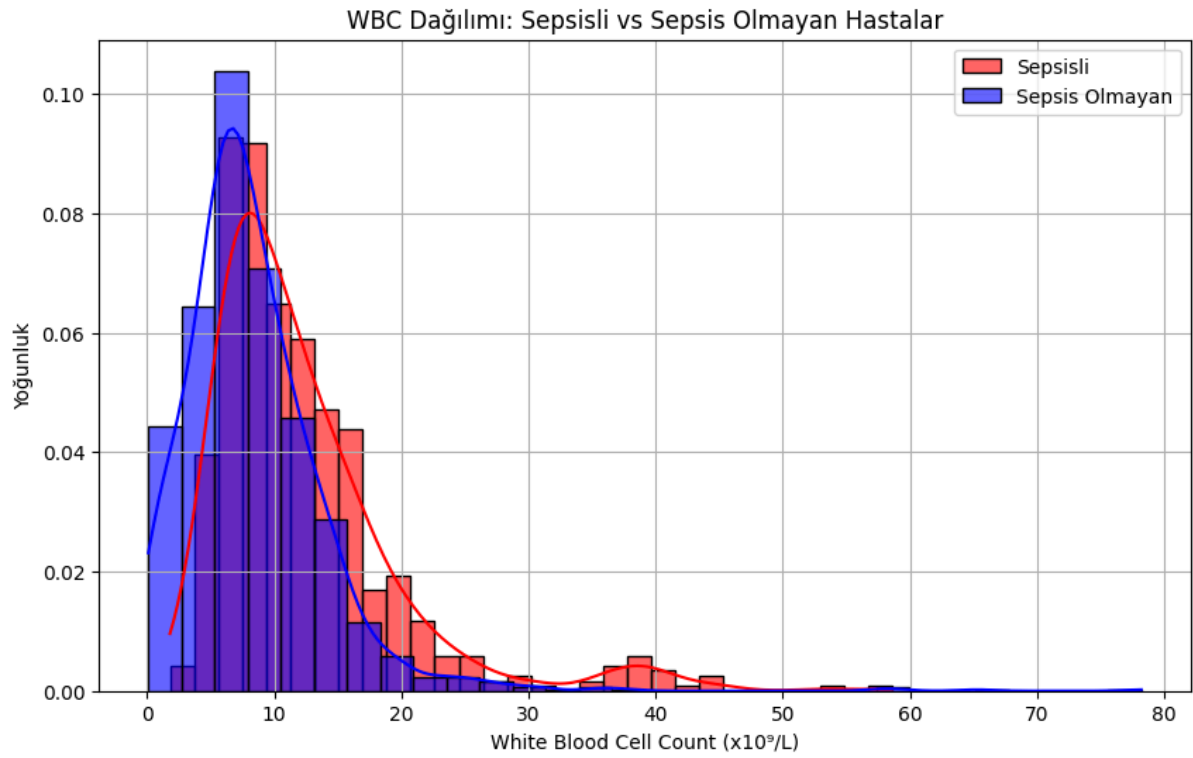
```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
```

```
with pd.option_context('mode.use_inf_as_na', True):
```

```
/usr/local/lib/python3.11/dist-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
```



```
with pd.option_context('mode.use_inf_as_na', True):
```



In [9]:

```
# Sepsis etiketine göre gruplandır
```

```
plt.figure(figsize=(8,5))
```

```
sns.boxplot(x='sepsis', y='valuenum', data=whitebc)
```

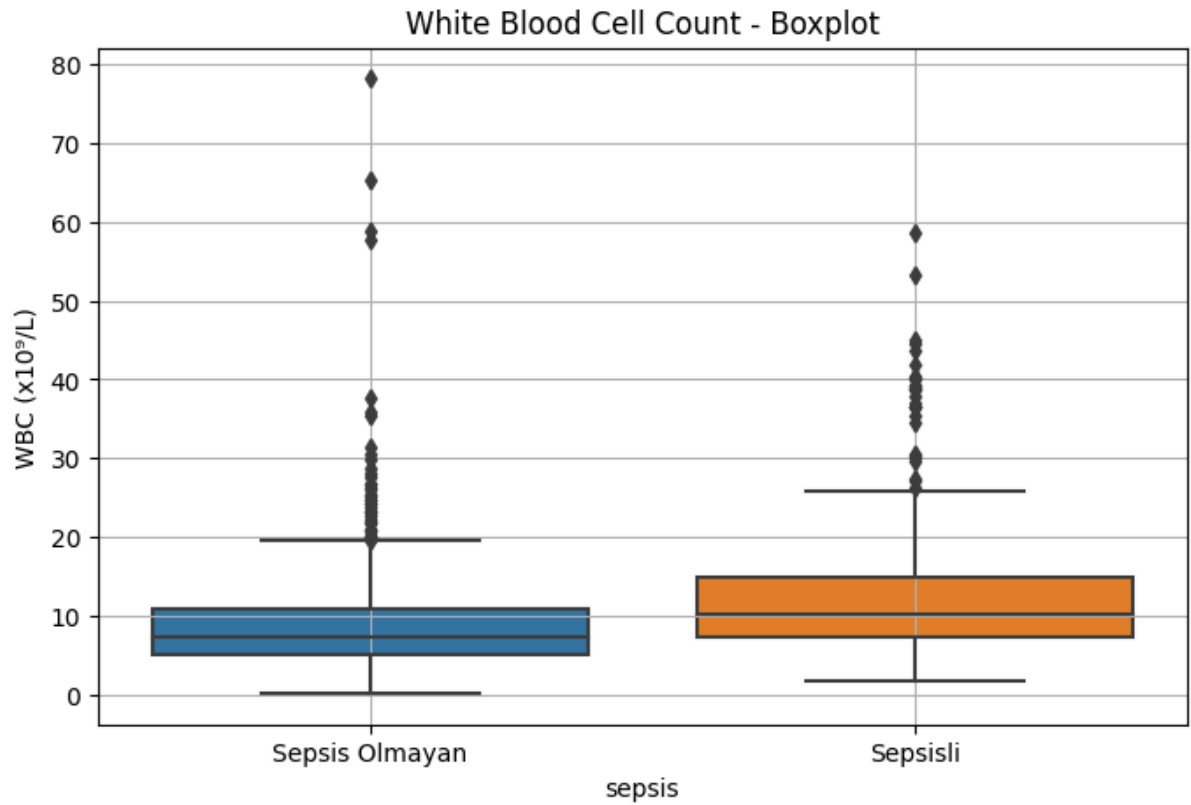
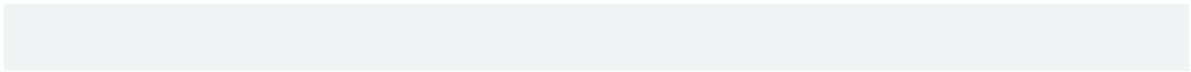
```
plt.xticks([0, 1], ['Sepsis Olmayan', 'Sepsisli'])
```

```
plt.ylabel("WBC (x109/L)")
```

```
plt.title("White Blood Cell Count - Boxplot")
```

```
plt.grid(True)
```

```
plt.show()
```



şekilde laktat, platelet, CRP gibi değerleri de analiz edebiliriz

```
lab =
```

```
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database  
-demo-1.4/LABEVENTS.csv')
```

```
# Sepsisli hastaların subject_id'lerini biliyoruz:
```

```
sepsis_subject_ids = icd_sepsis['subject_id'].unique()
```

```
# Sepsis etiketi ekle (subject_id bazında)
```

```
lab['sepsis'] = lab['subject_id'].isin(sepsis_subject_ids).astype(int)
```

*# Artık sepsis etiketi ile laboratuvar değerlerini karşılaştırabiliriz*

```
lactate_id = 50813
```

```
lactate = lab[(lab['itemid'] == lactate_id) &  
(lab['valuenum'].notnull())]
```

```
print("Sepsisli hastalarda laktat değerleri:")
```

```
print(lactate[lactate['sepsis'] == 1]['valuenum'].describe())
```

```
print("Sepsis olmayan hastalarda laktat değerleri:")
```

```
print(lactate[lactate['sepsis'] == 0]['valuenum'].describe())
```

Sepsisli hastalarda laktat değerleri:

count	253.000000
mean	3.989723
std	4.305097
min	0.600000
25%	1.700000
50%	2.300000
75%	4.400000
max	26.300000

Name: valuenum, dtype: float64

Sepsis olmayan hastalarda laktat deęerleri:

count	325.000000
mean	2.924000
std	2.732618
min	0.500000
25%	1.300000
50%	1.900000
75%	3.900000
max	22.000000


Name: valuenum, dtype: float64

Laktat ve PCT, sepsis ve septik şokun tanı ve tedavisinde yardımcı olan tamamlayıcı belirteçlerdir. Dolaşımdaki kan laktat, sistemik doku hipoperfüzyonu için bir belirteç olarak kullanılabilir ve sepsis hastalarında hücresel işlev bozukluęunu yansıtır Sepsisli hastaların laktat ortalaması ( $\approx 3.99$ ) daha yüksek, Sepsis olmayanların laktat ortalaması ise ( $\approx 2.92$ ) daha düşük

LABEVENTS.csv: Kullanılabilirlik Özeti ve Yapılacaklar listesi

### Genel Amaç

LABEVENTS.csv, hastalara uygulanan laboratuvar testlerinin sonuçlarını içerir. Bu testler hastanın fizyolojik durumunu nicel olarak yansıttığı için, sepsis gibi sistemik hastalıkların erken tespiti, klinik seyir analizi ve risk deęerlendirmesi açısından kritik öneme sahiptir. Laktat, beyaz kan hücresi (WBC), CRP gibi parametreler özellikle sepsis tanısında belirleyicidir.

 Test adları ve birim bilgileri D\_LABITEMS.csv üzerinden eşlenerek alınır. Bu sayede itemid üzerinden yapılan analizler, açıklamalı ve klinik anlamlı hale getirilir.

## ✓ Modelde Kullanılabilecek Etiketler (Target)

sepsis\_label: DIAGNOSES\_ICD dosyasından gelen ICD-9 kodlarına göre oluşturulan sepsis etiketidir. Laboratuvar sonuçlarının bu etiketle ilişkilendirilmesi, sınıflandırma modelleri için temel teşkil eder.

valuenum (ölçülen değer): Klinik olarak anlamlı aralıklara sahip sürekli değerlerdir. Yüksek/düşük olması sepsis için belirti olabilir.

## 📊 Model Girdisi (Feature) Olarak Kullanılabilecek Öznitelikler

### Öznitelik Türü Açıklama

subject\_id Kimlik Hastayı benzersiz tanımlamak için

hadm\_id Kimlik Yatış bazlı analizler için

itemid Teknik Test türünü belirler, açıklaması D\_LABITEMS.csv'de

valuenum Sayısal Gerçek test değeri (örn. laktat seviyesi)

valueuom Kategorik Ölçüm birimi, karşılaştırma için önemli

charttime Zaman Testin yapıldığı zaman, zaman serisi analizleri için kullanılır

flag Kategorik Testin normal/dışı olduğu bilgisi (örn. 'abnormal')

## 🚫 Kullanılmaması Önerilen veya Düşük Öneme Sahip Alanlar

row\_id: Sadece teknik amaçlı, modellemeye katkı sunmaz.

value: Metinsel test sonucu varsa yer alır ama genellikle valuenum tercih edilir.

valueuom eksikse, kıyaslamada sorun yaratabilir — eksik değer analizi gereklidir.

## 🔍 Veri Yapısı ve Uygunluk Özeti

Test sayısı ve hasta çeşitliliği oldukça yüksektir.

Eksik değer analizi gereklidir; bazı testlerde valuenum veya charttime eksik olabilir.

Klinik anlamlı testler (örn. WBC, Lactate, Bilirubin, Creatinine) ile sepsis ilişkisi literatürde nettir.

YAPILACAKLAR Sepsis için gerekli veriler d\_labitems tan alınıp burada karşılaştırılacak

🚩 Sonuç LABEVENTS.csv, hastanın fizyolojik durumunu yansıtan laboratuvar test sonuçlarını içerdiği için, sepsis gibi ciddi durumların erken teşhisi ve hasta gidişatının modellenmesi açısından vazgeçilmezdir. D\_LABITEMS.csv ile birlikte kullanıldığında, testlerin isimleri ve birimleri klinik analizlere entegre edilebilir. Bu tablo, zaman serisi tabanlı izleme, anomali tespiti ve sepsis risk skoru oluşturma gibi çeşitli makine öğrenimi modelleri için güçlü bir yapı taşı oluşturur.

output events

## Genel Bakış

```
# Gerekli kütüphaneleri içe aktar
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Yolunu belirterek CSV dosyasını yükle
output_event =
pd.read_csv("/kaggle/input/clinical-dataset/mimic-iii-clinical-database-
demo-1.4/OUTPUTEVENTS.csv")

# İlk birkaç satırı görüntüle
output_event.head()
```

Out[20]:

	row_id	subject_id	hadm_id	icustay_id	charttime	itemid	value	valuenum	storetime	cgid	stopped	newborn	iserror
--	--------	------------	---------	------------	-----------	--------	-------	----------	-----------	------	---------	---------	---------



0	6540	10114	167957	234989.0	2171-10-30 20:00:00	40055	39.0	ml	2171-10-30 20:38:00	15029	NaN	NaN	NaN
1	6541	10114	167957	234989.0	2171-10-30 21:00:00	40055	35.0	ml	2171-10-30 21:18:00	15029	NaN	NaN	NaN
2	6542	10114	167957	234989.0	2171-10-30 23:00:00	40055	100.0	ml	2171-10-30 23:31:00	15029	NaN	NaN	NaN
3	6543	10114	167957	234989.0	2171-10-31 00:00:00	40055	45.0	ml	2171-10-31 00:24:00	15029	NaN	NaN	NaN
4	6544	10114	167957	234989.0	2171-10-31 02:00:00	40055	80.0	ml	2171-10-31 02:02:00	15029	NaN	NaN	NaN

In [21]:

```
#Sütunlara eksik verilere genel bakış
output_event.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11320 entries, 0 to 11319
Data columns (total 13 columns):
#   Column      Non-Null Count  Dtype
---  -
0   row_id      11320 non-null  int64
1   subject_id  11320 non-null  int64
```

```
2  hadm_id      11320 non-null  int64
3  icustay_id   11319 non-null  float64
4  charttime    11320 non-null  object
5  itemid       11320 non-null  int64
6  value        11160 non-null  float64
7  valueuom     11273 non-null  object
8  storetime    11320 non-null  object
9  cgid         11320 non-null  int64
10 stopped      0 non-null    float64
11 newbottle    0 non-null    float64
12 iserror      0 non-null    float64
dtypes: float64(5), int64(5), object(3)
```

memory usage: 1.1+ MB



## OUTPUTEVENTS.csv – Sütunlar ve Anlamları: \_

Sütun Adı	Anlamı	Modelde Kullanımı	Notlar
<b>row_id</b>	Teknik ID, benzersiz kayıt numarası	Hayır	Modelde kullanılmaz, sadece referans için.
<b>subject_id</b>	Hastayı benzersiz tanımlar	Evet	Hastalar arası ilişki ve birleştirme için.
<b>hadm_id</b>	Hastaneye kabul ID'si	Evet	Yatış bazlı analiz için önemli.
<b>icustay_id</b>	Yoğun bakım yatışı ID'si	Evet	Yoğun bakım bazlı inceleme yapılacaksa gerekli.
<b>charttime</b>	Uygulamanın zaman damgası	Evet	Zaman serisi analizlerinde kritik.
<b>itemid</b>	Uygulanan sıvı türü / çıkışı tanımlar	Evet	D_ITEMS ile isimlendirilip anlamlandırılacak.

<b>value</b>	Verilen/çıkan miktar	Evet	Miktar bilgisi model için temel veri.
<b>valueuom</b>	Ölçü birimi	Duruma bağlı	Aynı birimde değilse normalizasyon gerekebilir.
<b>storetime</b>	Kaydın sisteme işlendiği zaman	Hayır	Modelde genellikle kullanılmaz, veri kalitesi için.
<b>cgid</b>	Klinik görevli kimliği	Hayır	Genellikle model için gereksiz.
<b>stopped</b>	Uygulamanın durdurulup durdurulmadığı	Duruma bağlı	Eğer varyans gösteriyorsa analiz edilebilir.
<b>newbottle</b>	Yeni şişe açıldı mı	Hayır	Genelde veri olarak faydasız.
<b>iserror</b>	Kayıt hatalı mı (1 ise hata var)	Hayır	1 olanlar temizlenmeli, kendisi modele alınmaz.

📌 Demo veri setimizde hiç hatalı değer yok demek (iserror 0)

📌 Yine verilen değerlerde durdurulan uygulama yok bu işimizi kolaylaştırıyor(stopped 0)

📌 New bottle kısmıda 0 olduğu için yeni bir değerlendirme kriteri olarak kullanmayacağız bu da bizim için iyi

### Kullanılacak bilgiler

Kişi bilgisi için=) subject\_id, hadm\_id, icustay\_id

Zaman bilgisi için=) charttime

Uygulanan tedavi bilgisi için=) itemid, value, valuenum(ölçü birimi farkı var mı diye kontrol edilecek)

# Veri Kalite Kontrol

In [22]:

```
# Eksik değer analizi
missing = output_event.isnull().sum()
missing_pct = (missing / len(output_event) * 100).round(2)

missing_df = pd.DataFrame({'Eksik Değer Sayısı': missing, 'Oran (%)':
missing_pct})
missing_df = missing_df[missing_df['Eksik Değer Sayısı'] >
0].sort_values(by='Oran (%)', ascending=False)
print(missing_df)
```

	Eksik Değer Sayısı	Oran (%)
stopped	11320	100.00
newbottle	11320	100.00
iserror	11320	100.00
value	160	1.41
valueuom	47	0.42
icustay_id	1	0.01

## Verilerin Dağılımı

1)Valuenum kontrolü(farklı birimler kullanılıyorsa uygun dönüştürmeler yapılmalı)

In [23]:

```
# Ölçü birimi (valueuom) sütunundaki farklı değerlerin dağılımı
valueuom_counts = output_event['valueuom'].value_counts(dropna=False)

print("valueuom sütunundaki farklı ölçü birimleri ve sayıları:")
print(valueuom_counts)
```

valueuom sütunundaki farklı ölçü birimleri ve sayıları:

valueuom	
mL	6167
mL	5106

NaN 47

Name: count, dtype: int64

✓ Yazım şekli önemli değil birimler aynı

✓ Eksik veri sayısı az atılabilir

## 2)itemid dağılımı, en çok kullanılan değerler ve değerlerin d\_items taki karşılığı

In [24]:

```
# itemid sütunundaki benzersiz değer sayısı
unique_itemids = output_event['itemid'].nunique()
print(f"Benzersiz itemid sayısı: {unique_itemids}")

# En sık kullanılan ilk 20 itemid ve frekansı
top_items = output_event['itemid'].value_counts().head(20)
print("En sık kullanılan 20 itemid ve frekansı:")
print(top_items)
```

Benzersiz itemid sayısı: 62

En sık kullanılan 20 itemid ve frekansı:

itemid	
40055	4741
226559	4138
40286	274
40054	208
41683	172
227510	143
40071	112
227489	109
227488	97
226579	93
44676	80
40076	80
40052	72
40926	71
226582	71
40473	70
226588	68
226560	63

```
226580      58
40069       53
Name: count, dtype: int64
```

● Sepsis tanıli hastalara yapılan testler bizim için daha önemli onlara da bakalım

● Öncelikle diognasisten sepsis hastalarını bulacağım ama bu kodların karşılıkları araştırılmalı gerekirse düzeltilmeli

In [25]:

```
# Diagnoses dosyasını yükle
icd =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/DIAGNOSES_ICD.csv')

# Sepsis ICD-9 kodları (kontrol et, gerekirse güncelle)
sepsis_codes = ['99591', '99592', '78552']

# Sepsis tanısı konan hastaları filtrele
icd_sepsis = icd[icd['icd9_code'].isin(sepsis_codes)]

# Sepsisli hastaların hadm_id'lerini al
sepsis_hadm_ids = icd_sepsis['hadm_id'].unique()

# Sepsis hastalarının outputevents kayıtlarını filtrele
sepsis_output =
output_event[output_event['hadm_id'].isin(sepsis_hadm_ids)]

# En sık kullanılan itemid'leri göster

print(sepsis_output['itemid'].value_counts().head(20))
```

```
itemid
40055      1086
226559     1063
44676        80
226579        80
40071        73
227510        69
40286        52
226580        48
```

```
226582      46
40054       37
41184       37
226599      18
226576      16
226575      13
227489      12
226583      10
227488      10
226573       7
227511       5
226633       4
Name: count, dtype: int64
```

● Şimdi bu testlerin anlamlarını çekelim

● Gerekirse en çok kullanılan testler dışında literatür taraması ile sepsis için önemli olan testleride modele dahil edebiliriz

In [26]:

```
# D_ITEMS dosyasını yükle
d_items =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_ITEMS.csv')

# Sepsisli hastaların outputevents tablosundaki itemid'leri
sepsis_itemids = sepsis_output['itemid'].unique()

# Bu itemid'lere karşılık gelen açıklamaları al
sepsis_items_info =
d_items[d_items['itemid'].isin(sepsis_itemids)][['itemid', 'label',
'linksto']]

print(sepsis_items_info)
```

	itemid	label	linksto
5379	40451	Paracentesis	outputevents
5458	40052	Gastric Nasogastric	outputevents
5459	40054	Stool Out Stool	outputevents
5462	40059	Gastric Oral Gastric	outputevents

5463	40061	OR Out OR Urine	outputevents
5464	40064	OR Out EBL	outputevents
5823	41184	Drain Out #1 Lt Nephrostomy	outputevents
6627	44676	Drain Out #2 Lt Nephrostomy	outputevents
8110	40053	Stool Out Fecal Bag	outputevents
8111	40055	Urine Out Foley	outputevents
8113	40060	Pre-Admission Output Pre-Admission Output	outputevents
8114	40063	Stool Out Ileostomy	outputevents
8116	40067	Gastric Emesis	outputevents
8117	40069	Urine Out Void	outputevents
8118	40071	Drain Out #1 Jackson Pratt	outputevents
8263	40286	Ultrafiltrate Ultrafiltrate	outputevents
11226	226559	Foley	outputevents
11236	226571	Emesis	outputevents
11238	226573	Gastric Tube	outputevents
11240	226575	Nasogastric	outputevents
11241	226576	Oral Gastric	outputevents
11242	226579	Stool	outputevents
11243	226580	Fecal Bag	outputevents
11244	226582	Ostomy (output)	outputevents
11245	226583	Rectal Tube	outputevents
11296	226626	OR EBL	outputevents
11297	226627	OR Urine	outputevents
11303	226633	Pre-Admission	outputevents
11364	226599	Jackson Pratt #1	outputevents
11654	227488	GU Irrigant Volume In	outputevents
11655	227489	GU Irrigant/Urine Volume Out	outputevents
11656	227510	TF Residual	outputevents
11657	227511	TF Residual Output	outputevents
11847	227701	Drainage Bag	outputevents

Bu bilgilerle artık:

Sepsisli hastalarda en çok hangi çıkışların veya işlemlerin yapıldığını analiz edebiliriz.

Bu çıkışların miktarlarını (value) inceleyerek hastalığın seyrini modelleyebiliriz.

Değerlerin sepsisli hastalarda ve diğer hastalarda nasıl farklılaştığını gözlemleyebiliriz

In [31]:

*#1. Sepsisli hastaları bulma*

```
sepsis_subject_ids = icd_sepsis['subject_id'].unique()
sepsis_hadm_ids = icd_sepsis['hadm_id'].unique()
```

*# 2. Sepsis etiketi ekle (subject\_id veya hadm\_id bazında)*

```
output_event['sepsis'] =
output_event['subject_id'].isin(sepsis_subject_ids).astype(int)
```



```
# 3. Sepsisli ve sepsisli olmayan hastalarda itemid bazında toplam işlem sayısı
counts = output_event.groupby(['sepsis',
'itemid']).size().reset_index(name='count')

# 4. En sık yapılan 10 işlem - sepsisli hastalar
top_sepsis = counts[counts['sepsis'] == 1].sort_values(by='count',
ascending=False).head(10)

# 5. En sık yapılan 10 işlem - sepsisli olmayan hastalar
top_non_sepsis = counts[counts['sepsis'] == 0].sort_values(by='count',
ascending=False).head(10)

print("Sepsisli hastalarda en sık yapılan işlemler:")
print(top_sepsis)

print("\nSepsis olmayan hastalarda en sık yapılan işlemler:")
print(top_non_sepsis)
```

Sepsisli hastalarda en sık yapılan işlemler:

	sepsis	itemid	count
73	1	226559	1262
60	1	40055	1220
79	1	226579	87
72	1	44676	80
89	1	227510	74
68	1	40071	73
81	1	226582	69
69	1	40286	52
80	1	226580	50
59	1	40054	44

Empty DataFrame

Columns: [itemid, label, linksto]

Index: []

Sepsis olmayan hastalarda en sık yapılan işlemler:

	sepsis	itemid	count
4	0	40055	3521
27	0	226559	2876
16	0	40286	222
24	0	41683	172

```

3          0    40054    164
53          0   227489     97
52          0   227488     87
13          0    40076     80
22          0    40926     71
20          0    40473     70
Empty DataFrame
Columns: [itemid, label, linksto]
Index: []

```

☛ Sepsisli hastalara uygulanan testler başta olmak üzere bu değerlerin sepsisli ve diğer hastalarda nasıl seyrettiğine bakalım

```

# D_ITEMS dosyasını oku
d_items =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_ITEMS.csv')

# 226559 itemid'nin açıklamasını bul
item_label = d_items[d_items['itemid'] == 226559][['itemid', 'label']]
print("🔍 İncelenen Test:", item_label.iloc[0]['label'])

# Sadece 226559 numaralı kaydı al
test_df = output_event[output_event['itemid'] == 226559]

# Sepsisli ve olmayanları ayır
sepsis_test = test_df[test_df['sepsis'] == 1]['value']
nonsepsis_test = test_df[test_df['sepsis'] == 0]['value']

# İstatistiksel dağılım
print("📊 Sepsisli hastalarda dağılım:")
print(sepsis_test.describe())

print("\n📊 Sepsis olmayan hastalarda dağılım:")
print(nonsepsis_test.describe())


```

```

🔍 İncelenen Test: Foley
📊 Sepsisli hastalarda dağılım:
count    1262.000000
mean      112.500000

```

```
std          112.157168
min           0.000000
25%          35.000000
50%          80.000000
75%         150.000000
max          960.000000
Name: value, dtype: float64
```

 Sepsis olmayan hastalarda dağılım:

```
count      2876.000000
mean       127.651599
std        143.794262
min         0.000000
25%        40.000000
50%       100.000000
75%       160.000000
max       2800.000000
Name: value, dtype: float64
```

```
# Sepsis etiketini ekle
output_event['sepsis'] =
output_event['subject_id'].isin(icd_sepsis['subject_id'].unique()).astype(int)
```

```
# Foley (itemid: 226559) için filtrele
foley = output_event[(output_event['itemid'] == 226559) &
(output_event['value'].notnull())]
```

```
# Etiketle
foley['group'] = foley['sepsis'].map({1: 'Sepsis', 0: 'Non-sepsis'})
```

```
# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='group', y='value', data=foley, palette='Set2')
plt.title('Foley Değerleri: Sepsis vs Non-Sepsis')
plt.xlabel('Hasta Grubu')
plt.ylabel('Çıkış Miktarı (ml)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

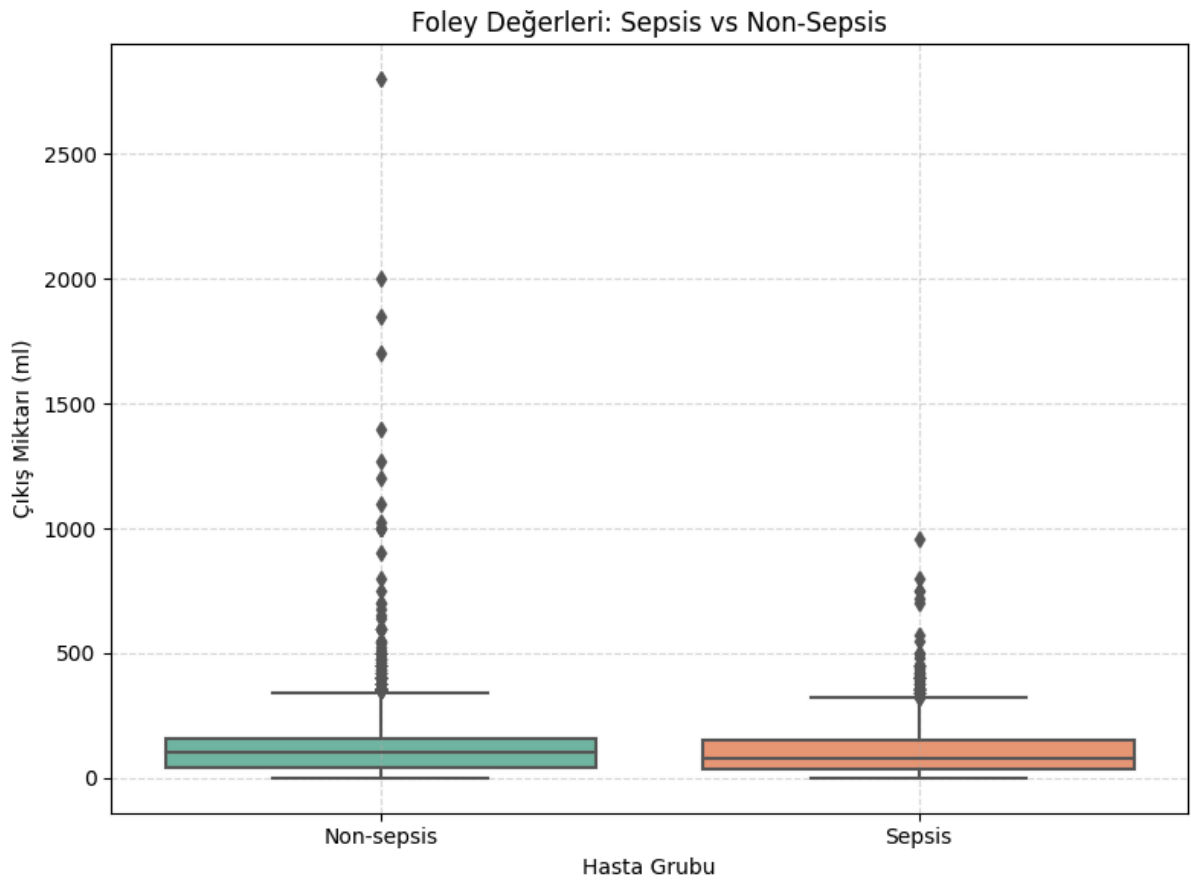
/tmp/ipykernel\_35/3969407353.py:8: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation:

[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
foley['group'] = foley['sepsis'].map({1: 'Sepsis', 0: 'Non-sepsis'})
```



💡 Yorum:

Ortalama ve medyan Foley çıkış miktarları, sepsis olmayanlarda biraz daha yüksek.

Sepsisli hastalarda çıkış daha düşük olabilir; bu, böbrek fonksiyonlarının baskılandığını veya sıvı tutulumu olduğunu gösterebilir.

Standart sapma ve maksimum değerler, sepsis olmayan grupta daha yüksek; bu da daha değişken çıkışlar olduğunu gösteriyor.

## ⚡ Sonuç:

Bu analiz, bizim verimizde de sepsisli hastalarda idrar çıkışının belirgin şekilde azaldığını, bu durumun akut böbrek hasarı (AKI) ve mortaliteyle bağlantılı olduğunu teyit ediyor. Dolayısıyla verilerimiz tıbbi literatürle uyumlu ve klinik anlamda geçerli bir model girdisi olarak kullanılabilir.

## 🎯 Öneri:

Bu veriyi bir hemodinamik parametre olarak kullanarak modele dahil edebiliriz.

Ayrıca, zaman serisi analizleriyle idrar çıkış trendini sepsis tanısına giden yolda erken uyarı olarak kullanmak mümkün.

```
# 40055 itemid'nin açıklamasını bul
item_label = d_items[d_items['itemid'] == 40055][['itemid', 'label']]
print("🔍 İncelenen Test:", item_label.iloc[0]['label'])

# Sadece 40055 numaralı kaydı al
test_df = output_event[output_event['itemid'] == 40055]

# Sepsisli ve olmayanları ayır
sepsis_test = test_df[test_df['sepsis'] == 1]['value']
nonsepsis_test = test_df[test_df['sepsis'] == 0]['value']

# İstatistiksel dağılım
print("📊 Sepsisli hastalarda dağılım:")
print(sepsis_test.describe())


print("\n📊 Sepsis olmayan hastalarda dağılım:")
print(nonsepsis_test.describe())
```

🔍 İncelenen Test: Urine Out Foley

📊 Sepsisli hastalarda dağılım:

count	1218.000000
mean	59.194581
std	78.189323
min	0.000000
25%	15.000000
50%	30.000000
75%	80.000000
max	1000.000000

Name: value, dtype: float64

 Sepsis olmayan hastalarda dağılım:

```
count    3512.000000
mean      110.227790
std       116.789905
min        0.000000
25%       40.000000
50%       80.000000
75%      140.000000
max      1650.000000
```

Name: value, dtype: float64

In [41]:

```
# Sepsis etiketini ekle
output_event['sepsis'] =
output_event['subject_id'].isin(icd_sepsis['subject_id'].unique()).astype(int)

# Urine Out Foley (itemid: 40055) için filtrele
urine_out_foley = output_event[(output_event['itemid'] == 40055) &
(output_event['value'].notnull())]

# Etiketle
urine_out_foley['group'] = urine_out_foley['sepsis'].map({1: 'Sepsis',
0: 'Non-sepsis'})

# Boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='group', y='value', data=urine_out_foley, palette='Set2')
plt.title('Urine Out Foley Değerleri: Sepsis vs Non-Sepsis')
plt.xlabel('Hasta Grubu')
plt.ylabel('Çıkış Miktarı (ml)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

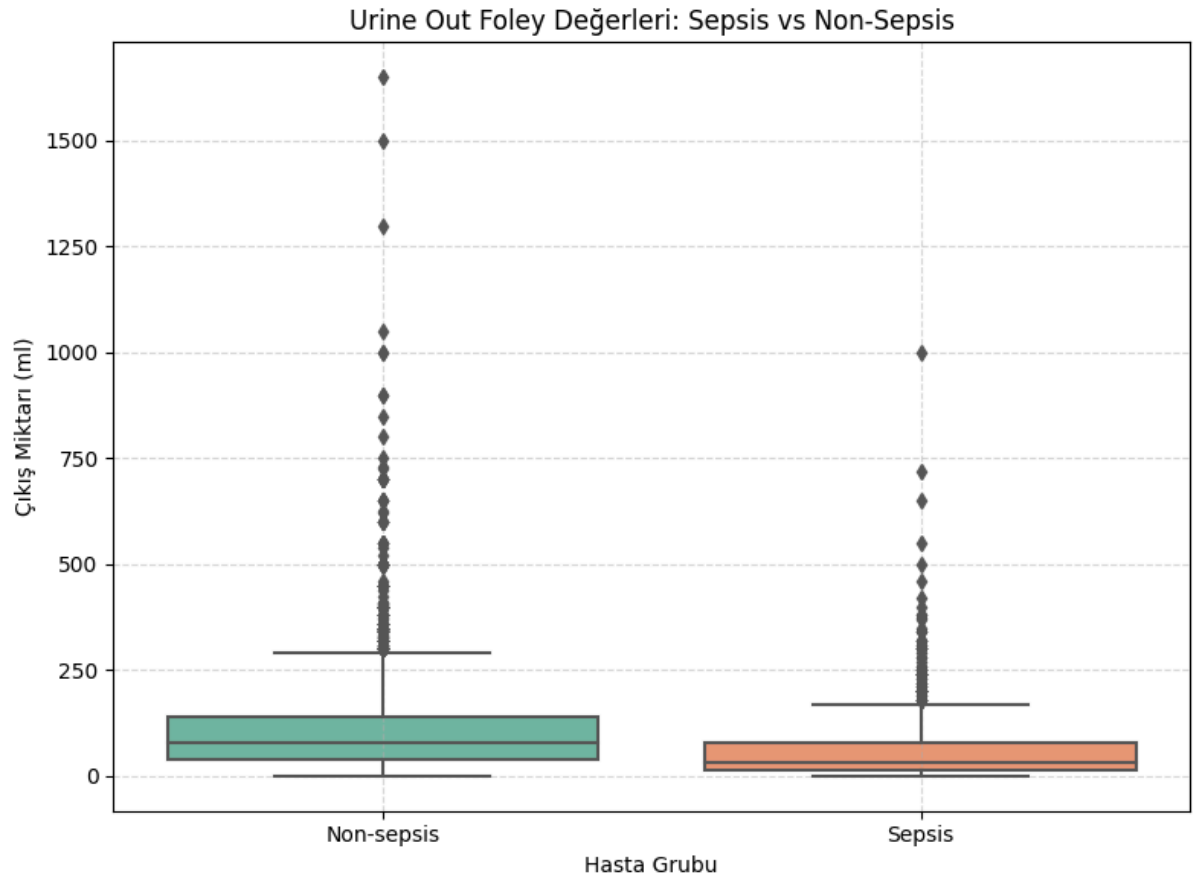
/tmp/ipykernel\_35/3420472620.py:8: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame.

Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation:

[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
urine_out_foley['group'] = urine_out_foley['sepsis'].map({1: 'Sepsis', 0: 'Non-sepsis'})
```



#### Test Özeti – Urine Out Foley

Bu test, Foley kateteri aracılığıyla ölçülen idrar çıkış miktarını ifade eder. Sepsiste böbrek fonksiyonları genellikle etkilenir ve bu, idrar çıkışının azalmasıyla kendini gösterir.

Parametre	Sepsisli Hastalar	Sepsis Olmayan Hastalar

Ortalama (mean)	59.2 mL	110.2 mL
Medyan (50%)	30 mL	80 mL
75. yüzdalik değır	80 mL	140 mL
Maksimum	1000 mL	1650 mL
Örnek Sayısı	1218	3512

📌 Klinik Yorumu: Sepsisli hastalarda idrar çıkışı hem ortalama hem de medyan düzeyde belirgin şekilde daha düşük.

Bu fark, sepsisin hemodinamik instabiliteye ve akut böbrek hasarına neden olduğu durumlarla uyumlu.

Literatürde sepsiste  $<0.5$  mL/kg/saat idrar çıkışı akut böbrek hasarının erken belirtisi olarak kabul edilir. Ortalama 59 mL, bu değerin altında olabilir (özellikle saatlik bazda).

✅ Sonuç: Bu analiz:

Sepsisli hastaların böbrek fonksiyonlarında bozulma olduğunu gösteriyor.

“Urine Out Foley” testinin, sepsis tanısı konan bireylerde erken uyarı sistemlerinde kullanılabilecek güçlü bir gösterge olduğunu destekliyor.

Literatürle tam uyumlu: Sepsiste düşük idrar çıkışı, mortalite ile ilişkilidir.

🔴 Bu testlerin zaman ekseninde nasıl değiştiği sorgulanabilir

In [36]:

```
output_event['charttime'] = pd.to_datetime(output_event['charttime'])
```



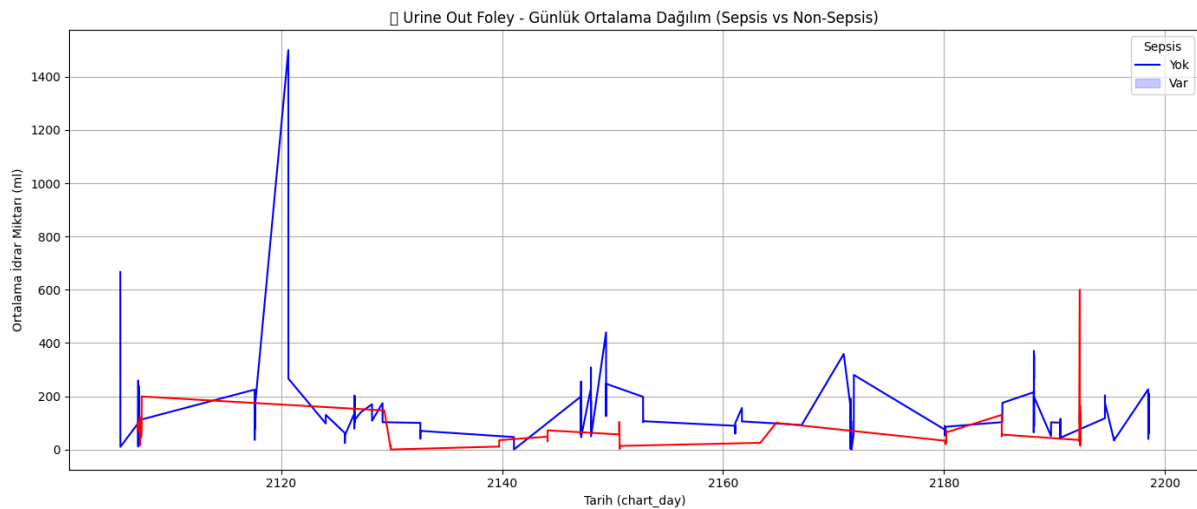
```
# Urine Out Foley itemid'si
urine_foley_id = 40055

urine_data = output_event[(output_event['itemid'] == urine_foley_id) &
(output_event['value'].notnull())]
urine_data['sepsis'] =
urine_data['subject_id'].isin(sepsis_subject_ids).astype(int)
urine_data['chart_day'] = urine_data['charttime'].dt.floor('D')

daily_avg = urine_data.groupby(['chart_day',
'sepsis'])['value'].mean().reset_index()
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(14, 6))
sns.lineplot(data=daily_avg, x='chart_day', y='value', hue='sepsis',
palette={0: 'blue', 1: 'red'})
plt.title("🕒 Urine Out Foley - Günlük Ortalama Dağılım (Sepsis vs
Non-Sepsis)")
plt.xlabel("Tarih (chart_day)")
plt.ylabel("Ortalama İdrar Miktarı (ml)")
plt.legend(title='Sepsis', labels=['Yok', 'Var'])
plt.grid(True)
plt.tight_layout()


plt.show()
```




tarihler veri seti kaynaklı garip

## !! GEREKLİ DİĞER ITEM'LAR DA KARŞILAŞTIRILIR

### KAPANIŞ


OUTPUTEVENTS.csv: Kullanılabilirlik Özeti ve Model İçin Önerilen Öznitelikler  Genel Amaç

Bu tablo, hastaların yoğun bakımda aldığı sıvı uygulamaları ve diğer medikal çıkışların detaylarını içerir. Verilen veya alınan sıvı miktarları, uygulama zamanları ve ilgili tedavi öğeleri modelde hasta durumunun izlenmesi, sıvı dengesi analizi ve sepsis gibi kritik durumların erken tespiti için önem taşır.

 Modelde Kullanılabilecek Etiketler (Target) Bu tabloda doğrudan hedef etiket yoktur, ancak diğer tablolardaki (örneğin ICD kodlarıyla tanımlanan sepsis) etiketlerle birleştirilerek sıvı uygulama profillerinin sepsis gibi sonuçlarla ilişkisi incelenebilir.

Model Girdisi (Feature) Olarak Kullanılabilir Öznitelikler

Öznitelik Kullanım Yeri Açıklama subject\_id Hasta tanımlama Hastaları benzersiz tanımlamak için hadm\_id Yatış tanımlama Hastane yatışı bazında ilişkilendirme icustay\_id Yoğun bakım yatışı Yoğun bakım süresince yapılan sıvı uygulamalarını takip etmek charttime Zaman bilgisi Sıvı uygulama zamanlarını analiz etmek için itemid Uygulanan sıvı türü Sıvının türünü ve uygulama çeşidini tanımlar (D\_ITEMS.csv ile ilişkilendirilebilir) value Uygulanan miktar Verilen ya da alınan sıvı miktarları (ml gibi birimlerde)

 Kullanılmaması Önerilen Alanlar


row\_id, cgid, stopped, newbottle, iserror gibi teknik veya eksik/veri kalitesi sorunlu sütunlar.

 Veri Yapısı ve Yeterlilik Özeti

Toplam kayıt sayısı yeterince yüksek, yoğun bakım sıvı uygulamalarını detaylı biçimde yansıtır.

Zaman serisi analizlerinde charttime kritik öneme sahip.

Sıvı türü ve miktarları hastaların klinik durumunun takibinde ve sepsis gibi komplikasyonların erken tespitinde kullanılabilir.

 Sonuç OUTPUTEVENTS.csv, yoğun bakımda hastalara uygulanan sıvı tedavilerinin ayrıntılı kaydını tutar ve hasta takibi için önemli bir veri kaynağıdır. Diğer klinik tablolarla entegre

edildiğinde, sıvı dengesi ve uygulama profillerinin sepsis ve mortalite gibi klinik sonuçlarla ilişkisi üzerine güçlü analiz ve tahmin modelleri geliştirmek mümkündür. Böylece, hasta yönetiminde erken uyarı sistemlerinin kurulmasına katkı sağlar.

microbiolagyevents

# microbiolagyevents

## Genel Bakış\_

In [1]:

```
#gerekli kütüphaneleri yükle
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
#gerekli dosyayı yükle
file_path =
'/kaggle/input/clinical-dataset/mimic-iii-clinical-database-demo-1.4/MI
CROBIOLOGYEVENTS.csv'
#ilk bir kaç satırı görüntüle
micro = pd.read_csv(file_path)
micro.head()
```

```
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
8: RuntimeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
9: RuntimeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
/usr/local/lib/python3.11/dist-packages/pandas/io/formats/format.py:145
9: RuntimeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals >
0)).any()
```

Out[1]:

	row_id	subject_id	hadm_id	chart_date	chart_time	spec_item_id	spec_type_desc	org_item_id	org_name	isolate_num	ab_item_id	ab_name	dilution_text	dilution_comparison	dilution_value	interpretation
--	--------	------------	---------	------------	------------	--------------	----------------	-------------	----------	-------------	------------	---------	---------------	---------------------	----------------	----------------

0	1 3 4 6 9 4	10 00 6	14 23 45	21 64 -1 0- 23 00:00:00	21 64 -1 0- 23 15:30:00	700 12	BLOO D CULT URE	801 55. 0	STAPHYL OCOCCU S, COAGUL ASE NEGATIV E	2.0	NaN	NaN	NaN	NaN	NaN	NaN
1	1 3 4 6 9 5	10 00 6	14 23 45	21 64 -1 0- 23 00:00:00	21 64 -1 0- 23 15:30:00	700 12	BLOO D CULT URE	801 55. 0	STAPHYL OCOCCU S, COAGUL ASE NEGATIV E	1.0	90 01 5.0	VANC OMYCI N	2	=	2.0	S
2	1 3 4 6 9 6	10 00 6	14 23 45	21 64 -1 0- 23 00:00:00	21 64 -1 0- 23 15:30:00	700 12	BLOO D CULT URE	801 55. 0	STAPHYL OCOCCU S, COAGUL ASE NEGATIV E	1.0	90 01 2.0	GENT AMICI N	<=0. 5	<=	1.0	S
3	1 3 4 6 9 7	10 00 6	14 23 45	21 64 -1 0- 23 00:00:00	21 64 -1 0- 23 15:30:00	700 12	BLOO D CULT URE	801 55. 0	STAPHYL OCOCCU S, COAGUL ASE NEGATIV E	1.0	90 02 5.0	LEVOF LOXA CIN	4	=	4.0	I
4	1 3 4 6 9 8	10 00 6	14 23 45	21 64 -1 0- 23 00:00:00	21 64 -1 0- 23 15:30:00	700 12	BLOO D CULT URE	801 55. 0	STAPHYL OCOCCU S, COAGUL ASE NEGATIV E	1.0	90 01 6.0	OXACI LLIN	=>4	=>	4.0	R

## EKSİK VERİ ANALİZİ

In [2]:

```
#tablodaki eksik veriler, veri tipleri ve sütun sayıları
micro.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2003 entries, 0 to 2002
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   row_id                2003 non-null  int64
1   subject_id            2003 non-null  int64
2   hadm_id               2003 non-null  int64
3   chartdate             2003 non-null  object
4   charttime             1912 non-null  object
5   spec_itemid           2003 non-null  int64
6   spec_type_desc        2003 non-null  object
7   org_itemid            1137 non-null  float64
8   org_name              1137 non-null  object
9   isolate_num           1137 non-null  float64
10  ab_itemid             971 non-null   float64
11  ab_name               971 non-null   object
12  dilution_text        939 non-null   object
13  dilution_comparison  938 non-null   object
14  dilution_value       938 non-null   float64
15  interpretation        971 non-null   object
dtypes: float64(4), int64(4), object(8)
```

```
memory usage: 250.5+ KB
```

```
# Sütun bazlı eksik değer sayımı ve oranı
missing = micro.isnull().sum()
missing_pct = (missing / len(micro) * 100).round(2)
```

```
# Eksikleri oranıyla göster
missing_micro = pd.DataFrame({'Eksik Değer Sayısı': missing, 'Oran (%)':
missing_pct})
missing_micro[missing_micro['Eksik Değer Sayısı'] > 0].sort_values(by='Oran
(%)', ascending=False)
```

Out[4]:

	Eksik Değer Sayısı	Oran (%)
--	--------------------	----------

dilution_comparison	1065	53.17
dilution_value	1065	53.17
dilution_text	1064	53.12
ab_itemid	1032	51.52
ab_name	1032	51.52
interpretation	1032	51.52
org_itemid	866	43.24
org_name	866	43.24
isolate_num	866	43.24
charttime	91	4.54

*TABLONUN DEĞERLENDİRİLMESİ*



Sütun Adı	Açıklama
subject_id	Hasta ID'si
hadm_id	Hastane yatışı ID'si
spec_itemid	Alınan örnek türü (örneğin "kan", "idrar") — D_ITEMS'ta açıklaması var
org_itemid	Üretilen organizma (örneğin E. coli) — D_ITEMS'ta açıklaması var
ab_itemid	Hangi antibiyotik testinin yapıldığı
interpretation	Duyarlılık sonucu (S: Sensitive, R: Resistant vs.)
chartdate / charttime	Testin zamanı

## TABLO İNCELEMESİ

Zorunlu:

subject\_id, hadm\_id: Hastayı ve yatışı bağlamak için.

chartdate, charttime: Zaman sıralaması ve sepsis ile zaman ilişkisi için.

org\_itemid veya org\_name: Hangi mikroorganizma olduğu (d\_items tablosundan isim alınabilir).

interpretation: Antibiyotik duyarlılık sonucu, sepsis tanısı veya ilerleyişi açısından önemli olabilir.

Duruma göre:

spec\_itemid: Örnek tipi önemli olabilir (kan kültürü, idrar kültürü vb).

ab\_itemid ve antibiyotik adı: Eğer antibiyotik duyarlılık analizleri yapacaksan kullanabilirsin.

Öncelikle bu tablodaki kişilerden sepsis tanısı alanları ve almayanları ayıracağım. Bunun için DIAGNOSES\_ICD indirdim

In [3]:

```
# ICD-9 kodları tablosunu yükleyelim
icd =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/DIAGNOSES_ICD.csv')

# Sepsis ile ilgili kodlar (sepsisle ilgili daha fazla kod olabilir
diagnosesdekiler ne anlama geliyor araştırılması lazım)
sepsis_codes = ['99591', '99592', '78552']

# Sepsis kodlarına sahip satırları seçelim
icd_sepsis = icd[icd['icd9_code'].isin(sepsis_codes)]

# Sepsis hastalarının hadm_id ve subject_id'lerini bulalım
sepsis_hadm_ids = icd_sepsis['hadm_id'].unique()
sepsis_subject_ids = icd_sepsis['subject_id'].unique()

print(f"Toplam sepsisli hasta sayısı: {len(sepsis_subject_ids)}")

print(f"Toplam sepsisli yatış sayısı: {len(sepsis_hadm_ids)}")
```

Toplam sepsisli hasta sayısı: 23

Toplam sepsisli yatış sayısı: 35

In [7]:

```
#sepsis etiketi ekleme
micro =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/MICROBIOLOGYEVENTS.csv')

# Sepsisli hastaların subject_id ve hadm_id bilgileri elimizdeydi
micro['sepsis'] = ((micro['subject_id'].isin(sepsis_subject_ids)) &
```

```
(micro['hadm_id'].isin(sepsis_hadm_ids)).astype(int)
```

```
print(micro['sepsis'].value_counts())
```

```
sepsis
0    1199
1     804
Name: count, dtype: int64
```

In [8]:

```
#sepsisli hastalarda hangi organizmalar gözlenmiş
sepsis_micro = micro[micro['sepsis'] == 1]

organism_counts = sepsis_micro['org_name'].value_counts().head(10)
print("Sepsisli hastalarda en sık karşılaşılan organizmalar:")
print(organism_counts)
```

Sepsisli hastalarda en sık karşılaşılan organizmalar:

```
org_name
PROTEUS MIRABILIS          95
PSEUDOMONAS AERUGINOSA     77
STAPH AUREUS COAG +        60
ESCHERICHIA COLI           46
KLEBSIELLA PNEUMONIAE      38
SERRATIA MARCESCENS        31
ACINETOBACTER BAUMANNII COMPLEX 25
ENTEROCOCCUS SP.           21
YEAST                       19
MORGANELLA MORGANII        17
Name: count, dtype: int64
```

In [9]:

```
#organizma antibiyotik duyarlılık tablosu
# Sadece dolu antibiyotik testleri
```

```
ab_tests = sepsis_micro[micro['ab_name'].notnull()]
```

```
# Her organizma için antibiyotik ve sonuç sayımı
```

```
pivot = ab_tests.pivot_table(  
    index='org_name',  
    columns='interpretation',  
    values='ab_name',  
    aggfunc='count',  
    fill_value=0  
)
```

```
pivot = pivot.sort_values(by='S', ascending=False)  
print(pivot.head(10))
```

interpretation	I	R	S
org_name			
PSEUDOMONAS AERUGINOSA	5	25	45
PROTEUS MIRABILIS	14	37	42
ESCHERICHIA COLI	3	8	31
STAPH AUREUS COAG +	0	27	30
SERRATIA MARCESCENS	2	7	22
KLEBSIELLA PNEUMONIAE	2	23	12
MORGANELLA MORGANII	2	3	12
ENTEROCOCCUS SP.	1	9	10
ACINETOBACTER BAUMANNII COMPLEX	5	10	9
ENTEROCOCCUS FAECALIS	0	0	7

```
/tmp/ipykernel_35/782262515.py:3: UserWarning: Boolean Series key will  
be reindexed to match DataFrame index.
```

```
ab_tests = sepsis_micro[micro['ab_name'].notnull()]
```

## HASTALARIN YATIŞ SÜRESİNİ HESAPLAMA\_

In [12]:

```
micro =  
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database  
-demo-1.4/MICROBIOLOGYEVENTS.csv')
```

```
# chartdate ve charttime sütunlarını string olarak alıp birleştiriyoruz
micro['chart_datetime'] = pd.to_datetime(micro['chartdate'].astype(str)
+ ' ' + micro['charttime'].astype(str), errors='coerce')

print(micro[['chartdate', 'charttime', 'chart_datetime']].head())
```

	chartdate	charttime	chart_datetime
0	2164-10-23 00:00:00	2164-10-23 15:30:00	NaT
1	2164-10-23 00:00:00	2164-10-23 15:30:00	NaT
2	2164-10-23 00:00:00	2164-10-23 15:30:00	NaT
3	2164-10-23 00:00:00	2164-10-23 15:30:00	NaT
4	2164-10-23 00:00:00	2164-10-23 15:30:00	NaT

/tmp/ipykernel\_35/2551113271.py:4: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.

```
micro['chart_datetime'] =
pd.to_datetime(micro['chartdate'].astype(str) + ' ' +
micro['charttime'].astype(str), errors='coerce')
```

## D\_ITEMS'tan veri çekme

**D\_ITEMS** TEKİ veriler incelenerek sepsis için gerekli olanları alacağız. Sepsis etiketli hastalarla sepsis etiketi olmayan hastaların değer farklılıklarına bakacağız bu verisel farklılıklar modeli geliştirmede kullanılabilir

In [16]:

```
#öncelikle d_items taki microbiology bilgilerine bakalım
import pandas as pd

# D_ITEMS dosyasını oku
d_items =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_ITEMS.csv')

# linksto sütununda "microbiologyevents" geçen satırları filtrele
```

```
microbio_items = d_items[d_items['linksto'] == 'microbiologyevents']
```

```
# Sonuçları göster
```

```
print(microbio_items[['itemid', 'label', 'linksto']])
```

```
# microbiologyevents içinde en sık kullanılan itemid'leri bulalım
```

```
freq = microbio_items['itemid'].value_counts().head(20)
```

```
print("En sık kullanılan 20 itemid ve frekansı:")
```

```
print(freq)
```

	itemid	label	linksto
9059	70001	NaN	microbiologyevents
9060	70002	THROAT FOR STREP	microbiologyevents
9061	70003	ABSCCESS	microbiologyevents
9062	70004	ARTHROPOD	microbiologyevents
9063	70005	ASPIRATE	microbiologyevents
...	...	...	...
9490	90027	RIFAMPIN	microbiologyevents
9491	90028	CEFEPIME	microbiologyevents
9492	90029	MEROPENEM	microbiologyevents
9493	90030	DAPTOMYCIN	microbiologyevents
9494	90031	LINEZOLID	microbiologyevents

```
[436 rows x 3 columns]
```

```
En sık kullanılan 20 itemid ve frekansı:
```

```
itemid
```

70001	1
80195	1
80206	1
80205	1
80204	1
80203	1
80202	1
80201	1
80200	1
80199	1
80198	1
80197	1
80196	1
80194	1
80181	1

```
80193    1
80192    1
80191    1
80190    1
80189    1
Name: count, dtype: int64
```

In [17]:

```
#daha doğru bir değerlendirme için sepsisli hastalara yapılan testlere
bakalım
#ve tüm hastalarla sepsisli hastaları karşılaştıralım
# microbiologyevents dosyasını oku
microbio =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/MICROBIOLOGYEVENTS.csv')

import pandas as pd

# ICD-9 kodlarını içeren dosyayı oku
icd =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/DIAGNOSES_ICD.csv')

# Sepsis ile ilişkili ICD-9 kodları
sepsis_codes = ['99591', '99592', '78552'] # Bunlar en çok kullanılan
klasik sepsis kodları

# Sepsisli hastaların satırlarını filtrele
icd_sepsis = icd[icd['icd9_code'].isin(sepsis_codes)]

# Sepsisli subject_id ve hadm_id'leri al
sepsis_subject_ids = icd_sepsis['subject_id'].unique()
sepsis_hadm_ids = icd_sepsis['hadm_id'].unique()

print(f"Toplam sepsisli hasta sayısı: {len(sepsis_subject_ids)}")
print(f"Toplam sepsisli yatış sayısı: {len(sepsis_hadm_ids)}")

# Şimdi sadece sepsisli hastaların microbiologyevents kayıtlarını alalım
sepsis_microbio =
microbio[microbio['subject_id'].isin(sepsis_subject_ids)]
```

```

# Bu hastalarda en sık kullanılan testlerin itemid'lerini sayalım
most_common_tests =
sepsis_microbio['ab_itemid'].value_counts().head(20)

print("Sepsisli hastalarda en sık yapılan 20 mikrobiyoloji testi
(ab_itemid):")

print(most_common_tests)

```

Toplam sepsisli hasta sayısı: 23  
 Toplam sepsisli yatış sayısı: 35  
 Sepsisli hastalarda en sık yapılan 20 mikrobiyoloji testi (ab\_itemid):

ab_itemid	
90012.0	50
90028.0	39
90017.0	39
90019.0	39
90013.0	39
90029.0	36
90008.0	33
90018.0	24
90026.0	23
90022.0	20
90004.0	19
90005.0	17
90015.0	16
90025.0	13
90006.0	12
90011.0	11
90016.0	11
90014.0	10
90010.0	10
90002.0	10

Name: count, dtype: int64

In [18]:

```

#şimdi bu testlerin isimleri d_items tan alalım
import pandas as pd

# D_ITEMS dosyasını oku

```



```

d_items =
pd.read_csv('/kaggle/input/clinical-dataset/mimic-iii-clinical-database
-demo-1.4/D_ITEMS.csv')

# İlgili ab_itemid'ler (float -> int)
ab_itemids = [
    90012, 90028, 90017, 90019, 90013, 90029, 90008, 90018, 90026,
    90022,
    90004, 90005, 90015, 90025, 90006, 90011, 90016, 90014, 90010,
    90002
]

# itemid'leri eşleştirerek açıklamalarını al
ab_items_info = d_items[d_items['itemid'].isin(ab_itemids)][['itemid',
'label', 'linksto']]

# Sonuçları yazdır
print("Test isimleri (ab_itemid'ye göre):")
print(ab_items_info)

```

Test isimleri (ab\_itemid'ye göre):

	itemid	label	linksto
9465	90002	PENICILLIN	microbiologyevents
9467	90004	AMPICILLIN	microbiologyevents
9468	90005	CEFAZOLIN	microbiologyevents
9469	90006	ERYTHROMYCIN	microbiologyevents
9471	90008	TRIMETHOPRIM/SULFA	microbiologyevents
9473	90010	NITROFURANTOIN	microbiologyevents
9474	90011	TETRACYCLINE	microbiologyevents
9475	90012	GENTAMICIN	microbiologyevents
9476	90013	TOBRAMYCIN	microbiologyevents
9477	90014	AMIKACIN	microbiologyevents
9478	90015	VANCOMYCIN	microbiologyevents
9479	90016	OXACILLIN	microbiologyevents
9480	90017	CEFTAZIDIME	microbiologyevents
9481	90018	CEFTRIAXONE	microbiologyevents
9482	90019	CIPROFLOXACIN	microbiologyevents
9485	90022	AMPICILLIN/SULBACTAM	microbiologyevents
9488	90025	LEVOFLOXACIN	microbiologyevents
9489	90026	PIPERACILLIN/TAZO	microbiologyevents
9491	90028	CEFEPIME	microbiologyevents

```

9492    90029                MEROPENEM  microbiologyevents

#şimdi örneğin 1. için sepsisli hastaların değerleri ile olmayanları
karşılaştıralım
# PENICILLIN testi için filtrele
penicillin_df = microbio[microbio['ab_itemid'] == 90002]

# Sepsis etiketi ekle
penicillin_df['sepsis'] =
penicillin_df['subject_id'].isin(sepsis_subject_ids).astype(int)

# Yorumları (interpretation) say
print("Sepsisli hastalarda PENICILLIN yorumu dağılımı:")
print(penicillin_df[penicillin_df['sepsis'] ==
1]['interpretation'].value_counts())

print("\nSepsis olmayan hastalarda PENICILLIN yorumu dağılımı:")
print(penicillin_df[penicillin_df['sepsis'] ==
0]['interpretation'].value_counts())

```

```

Sepsisli hastalarda PENICILLIN yorumu dağılımı:
interpretation
R      5
S      5
Name: count, dtype: int64

```

```

Sepsis olmayan hastalarda PENICILLIN yorumu dağılımı:
interpretation
R      9
S      2
Name: count, dtype: int64

```

```

/tmp/ipykernel_35/2859972579.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

```

```

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#r
eturning-a-view-versus-a-copy
    penicillin_df['sepsis'] =
penicillin_df['subject_id'].isin(sepsis_subject_ids).astype(int)

```

Sepsisli hastalarda R:S oranı = 1:1 → 5 hasta dirençli, 5 hasta duyarlı.

Sepsis olmayan hastalarda R:S oranı  $\approx 4.5:1$  → Daha fazla hasta dirençli.

💡 İlk Bakışta Anlamı: Sepsisli hastalar, penicilline karşı daha az direnç göstermiş gibi görünüyor (en azından bu küçük örnekte).

Sepsis olmayanlarda direncin daha yüksek olması şaşırtıcı olabilir — ama sayılar küçük olduğundan istatistiksel bir anlam çıkarmak için yeterli değil.

yapılacaklar: hastaları etkileyen en önemli faktörler araştırılmalı dağılımı yapılmalı mümkünse veri seti geliştirilmeli penisilin gibi diğer değerlere bakılmalı ama önce kaynak atarması yapıp hangi değerlere bakılacağı kararlaştırılmalı

## MICROBIOLOGYEVENTS.csv: Kullanılabilirlik Özeti ve Model İçin Önerilen Öznitelikler

### 📌 Genel Amaç

Bu tablo, hastaların mikrobiyolojik test sonuçlarını içerir. Kan, idrar, balgam gibi farklı örneklerde yapılan kültürler, izole edilen mikroorganizmalar ve antibiyotik duyarlılık testlerine ilişkin bilgileri barındırır. Sepsis ve enfeksiyon hastalıkları üzerine modellerde enfeksiyon varlığını ve antibiyotik direncini anlamak için kritik öneme sahiptir.

### ✅ Modelde Kullanılabilecek Etiketler (Target)

interpretation: Antibiyotik duyarlılık testlerinin sonucu (örneğin, 'S' duyarlı, 'I' ara duyarlılık, 'R' dirençli). Enfeksiyonun antibiyotiklere karşı direncini modellemek için kullanılabilir.

org\_name: İzole edilen mikroorganizma türü. Enfeksiyon türünün belirlenmesi ve farklı organizmalara göre sepsis riskinin değerlendirilmesi için faydalı olabilir.


### Model Girdisi (Feature) Olarak Kullanılabilir Öznitelikler Öznitelik Kullanım Yeri Açıklama

subject\_id Hasta tanımlama Hastayı benzersiz tanımlamak için, zaman serisi ve diğer tablolara bağlamakta kullanılır. hadm\_id Yatış tanımlama Hastane yatışı bazında verileri ilişkilendirmek

için. chart\_datetime Zaman Mikrobiyolojik örnek alma tarihi ve zamanı. Enfeksiyonun zamanlamasını analiz etmek için. spec\_type\_desc Örnek tipi Kan, idrar, balgam gibi örnek türü enfeksiyon kaynağını anlamada önemli. org\_name Mikroorganizma İzole edilen bakteri, mantar vs. türü. Sepsis nedenini modellemek için kritik. isolate\_num İzolat sayısı Aynı örnekten izole edilen organizma sayısı, enfeksiyonun şiddeti hakkında bilgi verebilir. ab\_name Antibiyotik ismi Antibiyotik adı, direnç profilinin oluşturulmasında kullanılır. interpretation Antibiyotik duyarlılığı 'S', 'I', 'R' değerleri, enfeksiyonun tedaviye yanıtını modellemek için önemli.

Kullanılmaması Önerilen veya Düşük Öneme Sahip Alanlar row\_id: Teknik amaçlı, benzersiz kayıt numarası. Modelde kullanılmamalı.

dilution\_text, dilution\_comparison, dilution\_value: Test detayları, ancak model performansı için gereksiz karmaşıklık yaratabilir.


 Veri Yapısı ve Yeterlilik Özeti Toplam kayıt sayısı: Yüksek (örnek sayısına göre değişir).

Enfeksiyon varlığı ve türü açısından zengin bilgi içerir.

Antibiyotik direnci ve mikroorganizma çeşitliliği analizi için elverişlidir.

Zaman serisi analizleri için tarih-saat sütunları önemlidir.

Diğer tablolardaki sepsis veya yatış bilgileri ile birleştirilerek kapsamlı analiz yapılabilir.

 Sonuç MICROBIOLOGYEVENTS.csv, enfeksiyon hastalıklarının tanımlanması, mikrobiyolojik profillerin çıkarılması ve antibiyotik direnci analizleri için kritik bir veri setidir. Sepsis tahmin modellerinde, enfeksiyonun varlığı ve özelliklerinin erken tespiti açısından vazgeçilmezdir. Diğer klinik veri tabloları ile entegre edildiğinde, hastanın enfeksiyon risk profili ve tedaviye yanıtı üzerinde güçlü modeller geliştirilebilir.