SOFA

✅

- **배경 설명**
  - SOFA(Sequential Organ Failure Assessment) 점수는 중환자실(ICU)에서 환자의 장기 기능을 평가하고 예후를 예측하는 데 사용되는 표준화된 도구. 각 장기 시스템(호흡기, 순환기, 신경계 등)의 기능 장애 수준을 0~4점으로 평가하고 이를 합산하여 전체 점수를 계산
- **활용 데이터 및 테이블**
  - **Respiratory**: PaO_2, FiO_2 (`labevents`, `chartevents`)
  - **Coagulation**: Platelets (`labevents`)
  - **Liver**: Bilirubin (`labevents`)
  - **Cardiovascular**: MAP, Vasopressors (`chartevents`)
  - **CNS**: GCS (`chartevents`)
  - **Renal**: Creatinine, Urine Output (`labevents`, `outputevents`)
  - **LABEVENTS**: 감염 지표(예: CRP, WBC) 변화.
- **주요 사용 사례**
  1. **환자 상태 추적**

     SOFA 점수는 시간 경과에 따른 장기 기능의 변화를 평가할 수 있어, 환자 상태가 악화되거나 개선되는지를 모니터링하는 데 유용
  2. **예후 예측**

     높은 SOFA 점수는 높은 사망률과 연관되어 있어, 의료진이 환자의 중증도를 이해하고 치료 계획을 세우는 데 도움을 줌
  3. **치료 효과 평가**

     특정 치료법이 장기 기능에 미치는 영향을 평가하는 데 사용
  4. **연구 목적**

     SOFA 점수는 ICU 데이터 분석 연구에서 중요한 지표로 사용. 예를 들어, MIMIC 데이터로 SOFA 점수를 계산하면 특정 치료가 환자 생존에 미치는 영향을 분석할 수 있음
- **모델링 방향**
  - **입력**: 계산된 SOFA 점수
  - **출력**: 환자 생존 (0=사망, 1=생존).
  - **알고리즘**: 로지스틱 회귀, XGBoost.

▼ SOFA 점수

SOFA 점수 기준: 각 장기 시스템에 대해 0~4점 범위로 평가됨

System	Parameter	0	1	2	3	4
Respiratory	PaO_2/FiO_2 (mmHg)	≥400	<400	<300	<200	<100
Coagulation	Platelets ($\times 10^3/\mu L$)	≥150	<150	<100	<50	<20
Liver	Bilirubin (mg/dL)	<1.2	1.2–1.9	2.0–5.9	6.0–11.9	≥12
Cardiovascular	MAP or vasopressors required	MAP ≥70	MAP <70	Dop ≤5	Dop >5 or Epi ≤0.1	Dop >15 or Epi >0.1
CNS	GCS Score	15	13–14	10–12	6–9	<6
Renal	Creatinine (mg/dL) or UO (mL/day)	<1.2	1.2–1.9	2.0–3.4	3.5–4.9 or UO <500	≥5.0 or UO <200

▼ MIMIC-III에서 SOFA 점수 계산

기본 준비

```
import pandas as pd
import numpy as np

# 필요한 데이터 불러오기
chartevents = pd.read_csv('CHARTEVENTS.csv')  # 생체 신호 데이터
labevents = pd.read_csv('LABEVENTS.csv')      # 실험실 검사 결과
outputevents = pd.read_csv('OUTPUTEVENTS.csv')  # 소변량 데이터
icustays = pd.read_csv('ICUSTAYS.csv')        # ICU 체류 정보
```

1. Respiratory: PaO_2/FiO_2

```
# PaO2와 FiO2 데이터 추출
paO2_data = labevents[labevents['itemid'] == 50821]  # itemid50821 = PaO2
```

```
fiO2_data = chartevents[chartevents['itemid'] == 223835]  # itemid223835 = FiO2

# PaO2/FiO2 계산
respiratory_data = paO2_data.merge(fiO2_data, on=['ICUSTAY_ID', 'CHARTTIME'], suffixes=('_paO2', '_fiO2'))
respiratory_data['PFRatio'] = respiratory_data['VALUENUM_paO2'] / (respiratory_data['VALUENUM_fiO2'] / 100)

# SOFA 점수 매핑
def map_respiratory_score(pf_ratio):
    if pf_ratio >= 400:
        return 0
    elif pf_ratio >= 300:
        return 1
    elif pf_ratio >= 200:
        return 2
    elif pf_ratio >= 100:
        return 3
    else:
        return 4

respiratory_data['SOFA_Respiratory'] = respiratory_data['PFRatio'].apply(map_respiratory_score)
```

2. Coagulation: Platelets

```
# Platelets 데이터 추출
platelets_data = labevents[labevents['ITEMID'] == 51265]  # ITEMID 51265 = Platelets

# SOFA 점수 매핑
def map_coagulation_score(platelet_count):
    if platelet_count >= 150:
        return 0
    elif platelet_count >= 100:
        return 1
    elif platelet_count >= 50:
        return 2
    elif platelet_count >= 20:
        return 3
    else:
        return 4

platelets_data['SOFA_Coagulation'] = platelets_data['VALUENUM'].apply(map_coagulation_score)
```

3. Liver: Bilirubin

```
# Bilirubin 데이터 추출
bilirubin_data = labevents[labevents['ITEMID'] == 50885]  # ITEMID 50885 = Bilirubin

# SOFA 점수 매핑
def map_liver_score(bilirubin):
    if bilirubin < 1.2:
        return 0
    elif bilirubin < 2.0:
        return 1
    elif bilirubin < 6.0:
        return 2
    elif bilirubin < 12.0:
        return 3
    else:
        return 4

bilirubin_data['SOFA_Liver'] = bilirubin_data['VALUENUM'].apply(map_liver_score)
```

4. MAP (Mean Arterial Pressure)

```
# MAP 데이터 추출 (ITEMID = 220045 for arterial MAP)
map_data = chartevents[chartevents['ITEMID'] == 220045]

# MAP 값 처리
map_data = map_data[['ICUSTAY_ID', 'CHARTTIME', 'VALUENUM']].dropna()
map_data.rename(columns={'VALUENUM': 'MAP'}, inplace=True)

# SOFA 점수 매핑
```

```
def map_cardiovascular_score(map_value):
    if map_value >= 70:
        return 0
    else:
        return 1

map_data['SOFA_Cardiovascular'] = map_data['MAP'].apply(map_cardiovascular_score)
```

5. GCS (Glasgow Coma Scale)

```
# GCS 관련 ITEMID 리스트
gcs_items = [223900, 223901, 223902]  # Eye, Motor, Verbal responses

# GCS 데이터 추출
gcs_data = chartevents[chartevents['ITEMID'].isin(gcs_items)]
gcs_data = gcs_data[['ICUSTAY_ID', 'CHARTTIME', 'ITEMID', 'VALUENUM']].dropna()

# GCS 점수 합산 (각 응답 점수를 시간별로 합산)
gcs_total = gcs_data.groupby(['ICUSTAY_ID', 'CHARTTIME'])['VALUENUM'].sum().reset_index()
gcs_total.rename(columns={'VALUENUM': 'GCS_Total'}, inplace=True)

# SOFA 점수 매핑
def map_cns_score(gcs_total):
    if gcs_total == 15:
        return 0
    elif gcs_total >= 13:
        return 1
    elif gcs_total >= 10:
        return 2
    elif gcs_total >= 6:
        return 3
    else:
        return 4

gcs_total['SOFA_CNS'] = gcs_total['GCS_Total'].apply(map_cns_score)
```

6-1. Creatinine

```
# Creatinine 데이터 추출 (ITEMID = 50912)
creatinine_data = labevents[labevents['ITEMID'] == 50912]
creatinine_data = creatinine_data[['ICUSTAY_ID', 'CHARTTIME', 'VALUENUM']].dropna()
creatinine_data.rename(columns={'VALUENUM': 'Creatinine'}, inplace=True)

# SOFA 점수 매핑
def map_renal_score(creatinine_value):
    if creatinine_value < 1.2:
        return 0
    elif creatinine_value < 2.0:
        return 1
    elif creatinine_value < 3.5:
        return 2
    elif creatinine_value < 5.0:
        return 3
    else:
        return 4

creatinine_data['SOFA_Renal'] = creatinine_data['Creatinine'].apply(map_renal_score)
```

6-2. Urine Output

```
# Urine Output 데이터 추출
urine_data = outputevents[outputevents['ITEMID'].isin([40055, 43175, 40069])]  # Urine output ITEMIDs
urine_data = urine_data[['ICUSTAY_ID', 'CHARTTIME', 'VALUE']].dropna()
urine_data.rename(columns={'VALUE': 'Urine_Output'}, inplace=True)

# 시간 단위 데이터를 일 단위로 합산
urine_daily = urine_data.groupby(['ICUSTAY_ID', urine_data['CHARTTIME'].dt.date])['Urine_Output'].sum().reset_index
()
urine_daily.rename(columns={'Urine_Output': 'Daily_Urine_Output'}, inplace=True)

# SOFA 점수 매핑
```

```
def map_urine_output_score(urine_output):
    if urine_output >= 500:
        return 0
    elif urine_output < 500 and urine_output >= 200:
        return 3
    else:
        return 4

urine_daily['SOFA_Urine_Output'] = urine_daily['Daily_Urine_Output'].apply(map_urine_output_score)
```

병합 및 최종 SOFA 점수 계산

이제 각 장기별 SOFA 점수를 병합하여 최종 SOFA 점수를 구함.

```
# SOFA 점수 병합: Respiratory, Coagulation, Liver, Cardiovascular (MAP + GCS), Renal (Creatinine or Urine Output)
sofa_scores = respiratory_data[['icustay_id', 'SOFA_Respiratory']].merge(
    platelets_data[['icustay_id', 'SOFA_Coagulation']],
    on='icustay_id', how='outer'
).merge(
    bilirubin_data[['icustay_id', 'SOFA_Liver']],
    on='icustay_id', how='outer'
).merge(
    map_data[['icustay_id', 'SOFA_Cardiovascular']].merge(
        gcs_total[['icustay_id', 'SOFA_CNS']],
        on='icustay_id', how='outer'
    ),
    on='icustay_id', how='outer'
).merge(
    creatinine_data[['icustay_id', 'SOFA_Renal']].merge(
        urine_daily[['icustay_id', 'SOFA_Urine_Output']],
        on='icustay_id', how='outer'
    ),
    on='icustay_id', how='outer'
)

# Renal 항목: Creatinine 또는 Urine Output 중 하나만 사용
# Renal 점수 계산 (Creatinine 우선, 없다면 Urine Output)
sofa_scores['SOFA_Renal_Final'] = sofa_scores[['SOFA_Renal', 'SOFA_Urine_Output']].min(axis=1)

# 최종 SOFA 점수 합산
sofa_scores['Total_SOFA'] = sofa_scores[
    ['SOFA_Respiratory', 'SOFA_Coagulation', 'SOFA_Liver', 'SOFA_Cardiovascular', 'SOFA_CNS', 'SOFA_Renal_Final']
].sum(axis=1, numeric_only=True)

# 결과 확인
print(sofa_scores[['icustay_id', 'Total_SOFA']].head())
```

결과 분석

- SOFA 점수는 각 장기 시스템별로 나누어 분석하거나, 총점을 계산해 환자 상태를 평가할 수 있음
- ICU 입실 첫 24시간 동안의 데이터를 사용하는 것이 일반적