



DATABASE-10조

고혈압인 암 환자 데이터 분석

김도훈 21016004

장태중 19015075

변민석 21016114



목차

- 1 주제 선정 이유
- 2 데이터 분석
- 3 전처리
- 4 모델링

주제 선정 이유

고혈압인 암환자들의
사망률 연관관계 분석




고혈압인 암 환자 데이터 분석

사용 데이터
mimic-iv-2.2\hosp

d_icd_diagnoses
diagnoses_icd
patients
admissions
omr

병합한 데이터

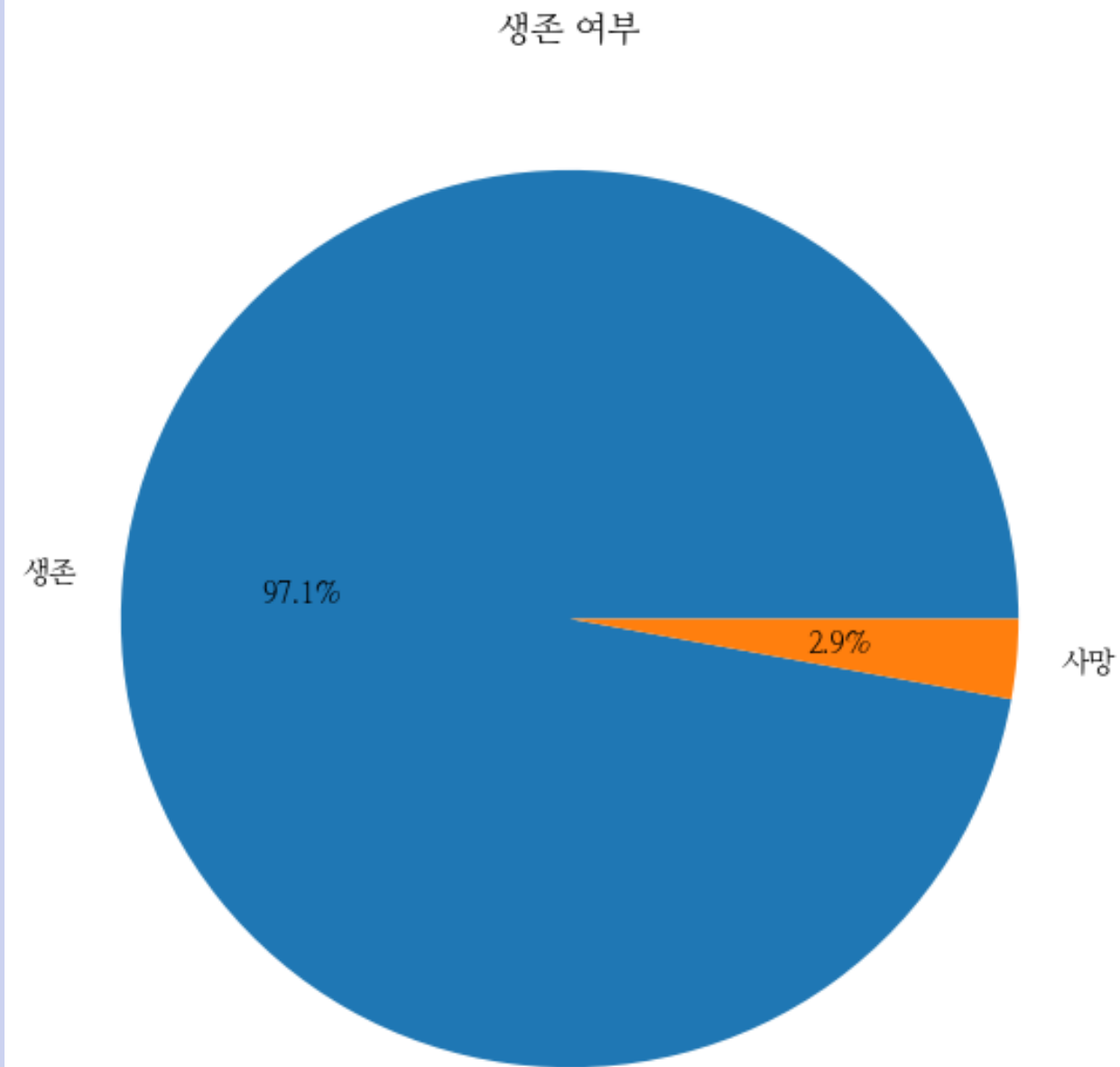
df_grouped과 df_data
d_labitems, labevents
first_records와 df_data



데이터 분석

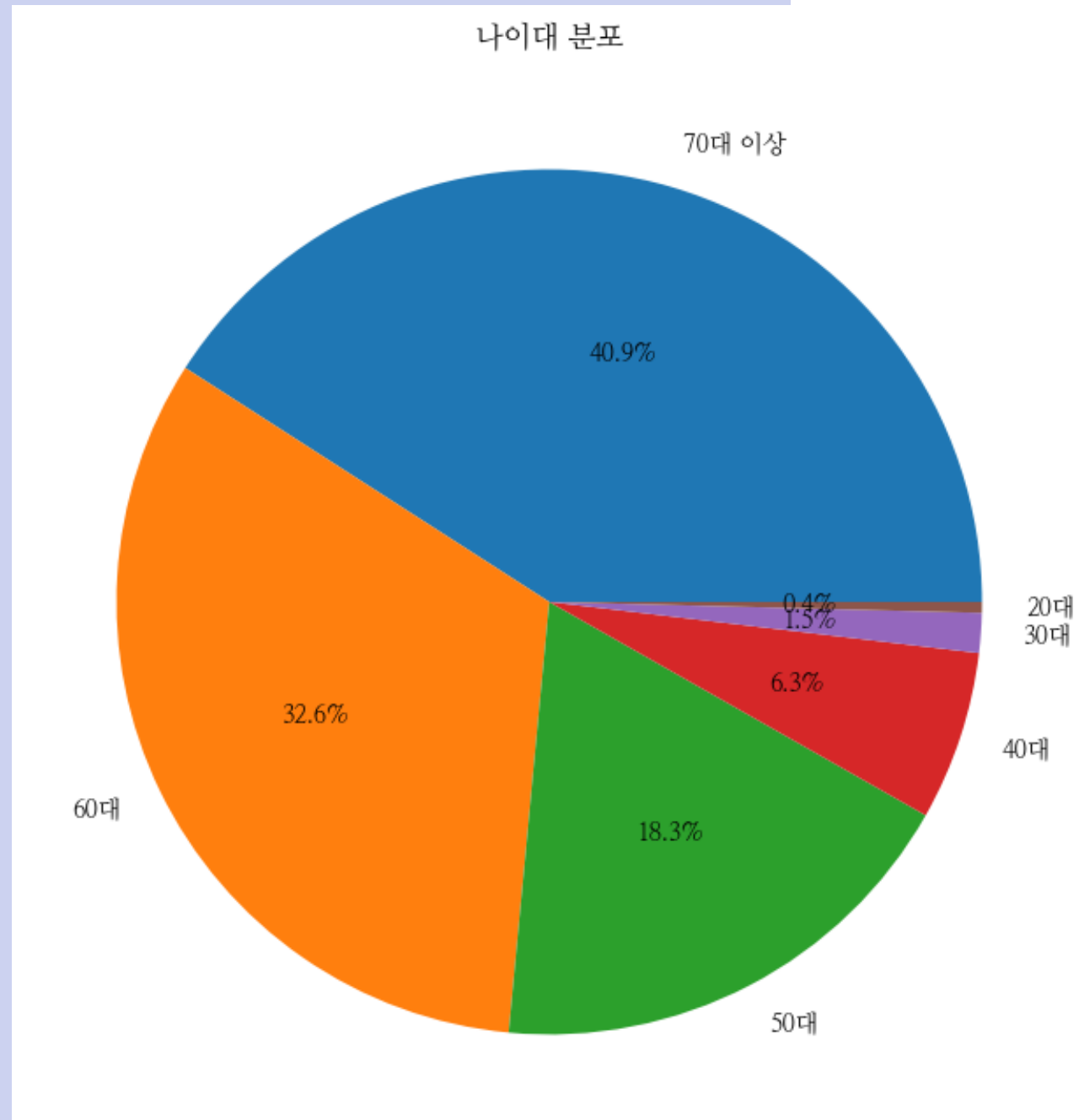
- 1 생존, 사망자 분포
- 2 나이대 별 데이터 분석
- 3 성별 데이터 분석
- 4 BMI 데이터
- 5 체내 칼륨과 나트륨 농도
- 6 혈압에 따른 사망, 생존 분포
- 7 변수간 상관관계
- 8 빈도 상위10개 약물

1. 생존 및 사망자 분포

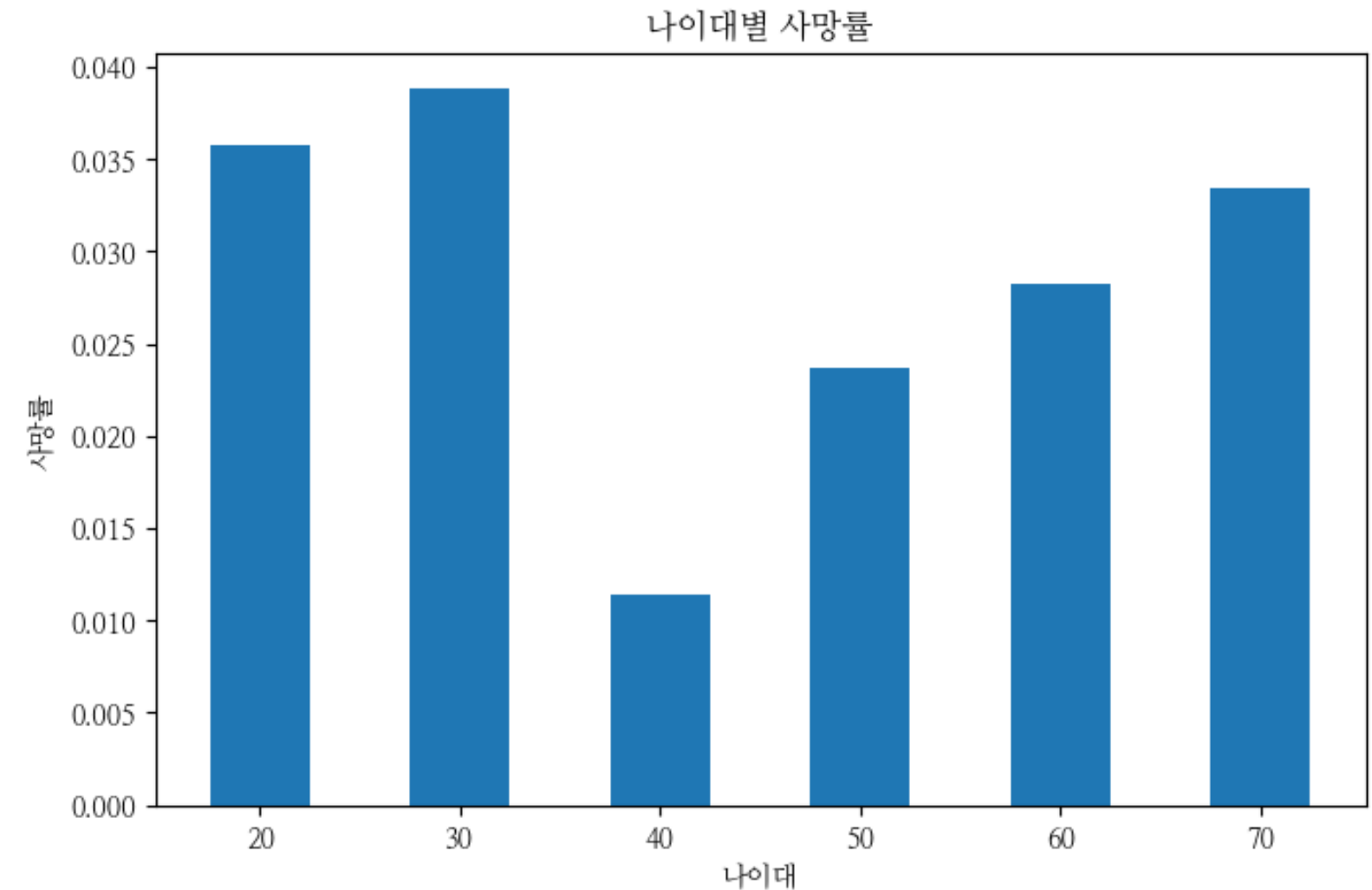


대다수의 환자가 생존한 것으로 나타납니다.
생존자가 전체 데이터셋의 약 97.1%를 차지하고 있습니다.
반면 사망자는 전체 데이터셋의
약 2.9%를 차지하고 있습니다.

2. 나이대 별 데이터 분석



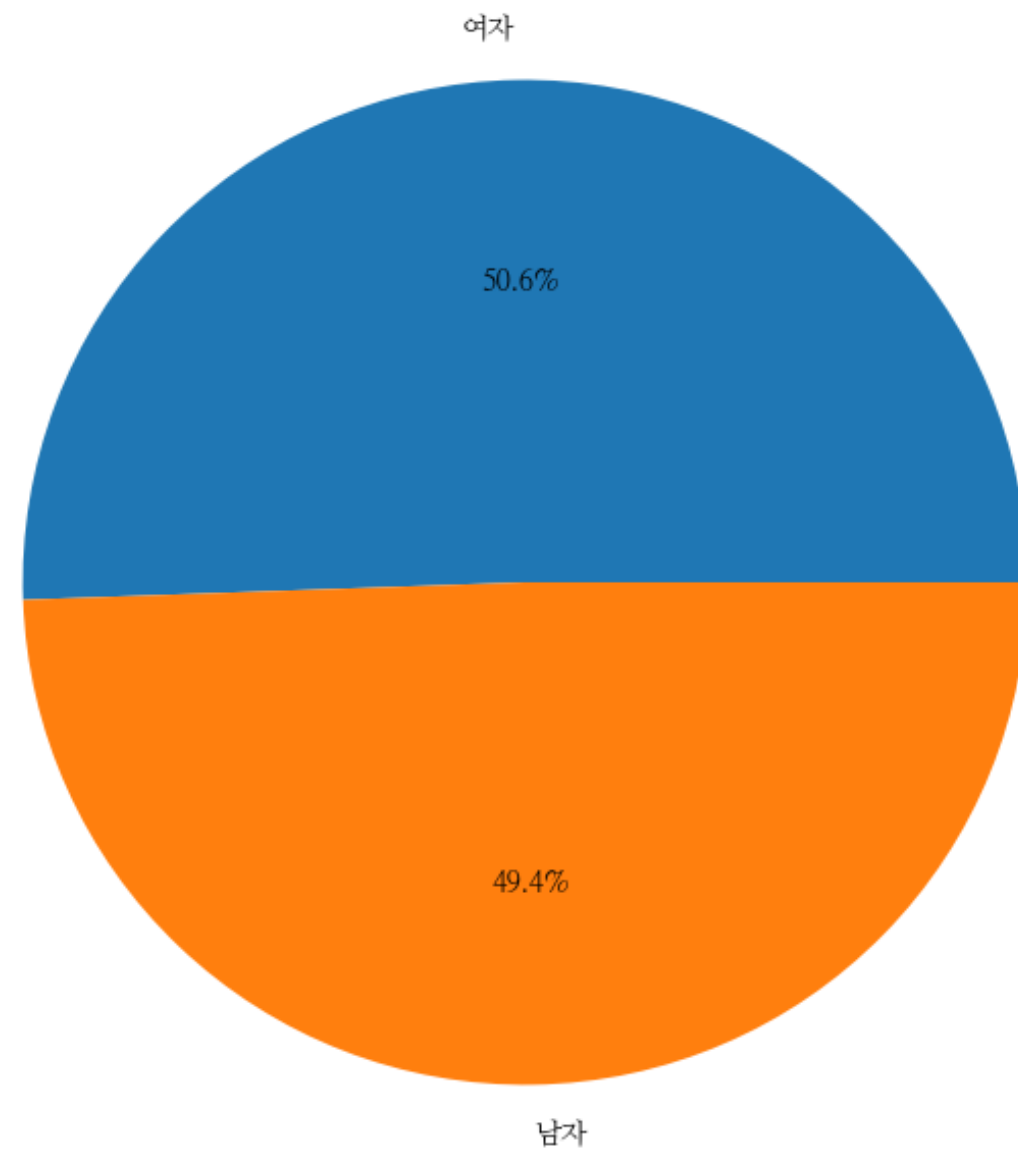
가장 많은 환자들이 '70대 이상' 나이 그룹에 속함을 확인할 수 있습니다. 젊은 나이대일수록 데이터셋의 비중이 낮음.



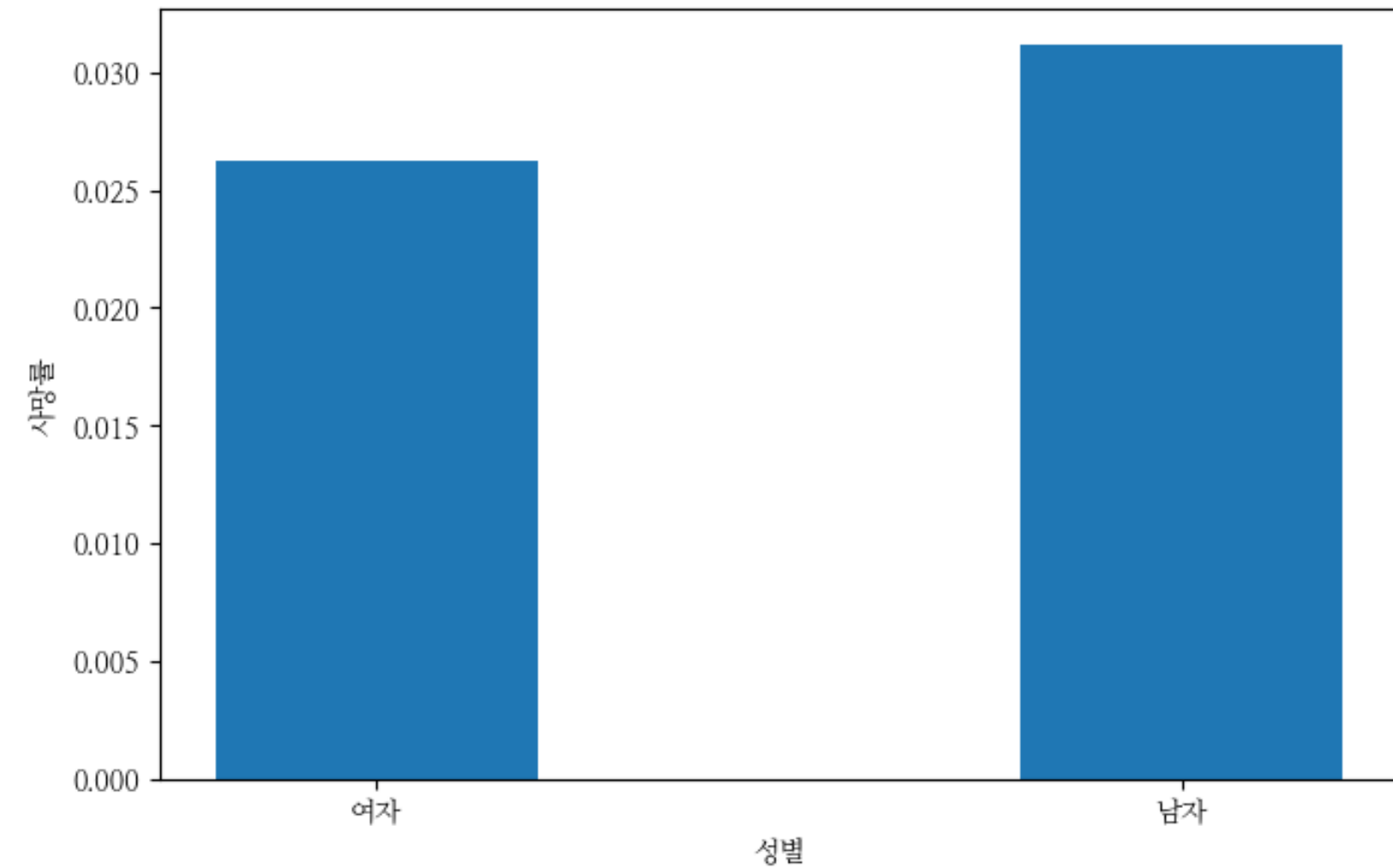
40대에서 낮아졌다가 사망률이 다시 증가함

3.성별 데이터 분석

성별 분포



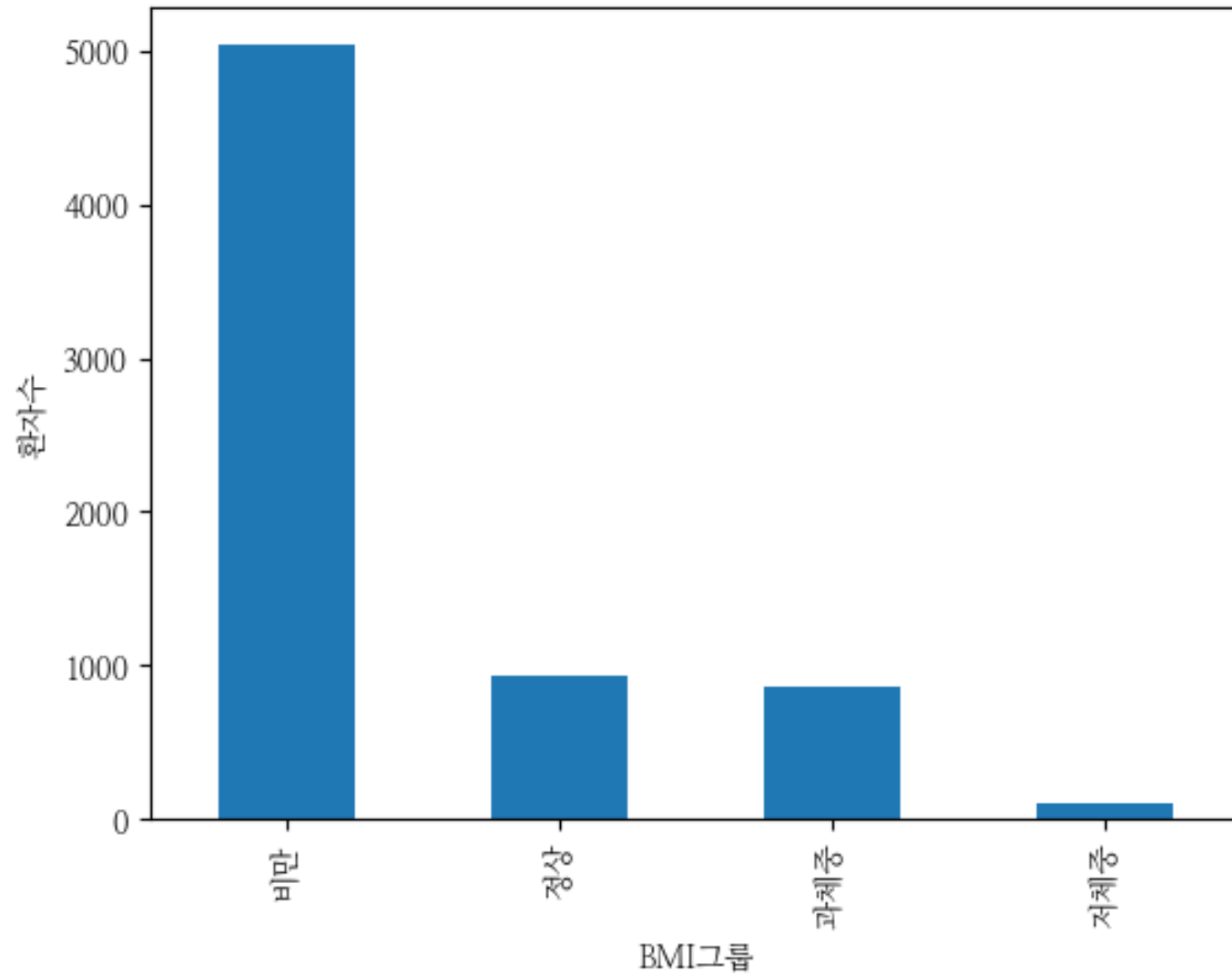
성별 사망률



여성의 사망률이 남성의 사망률보다 낮음을 확인할 수 있습니다.

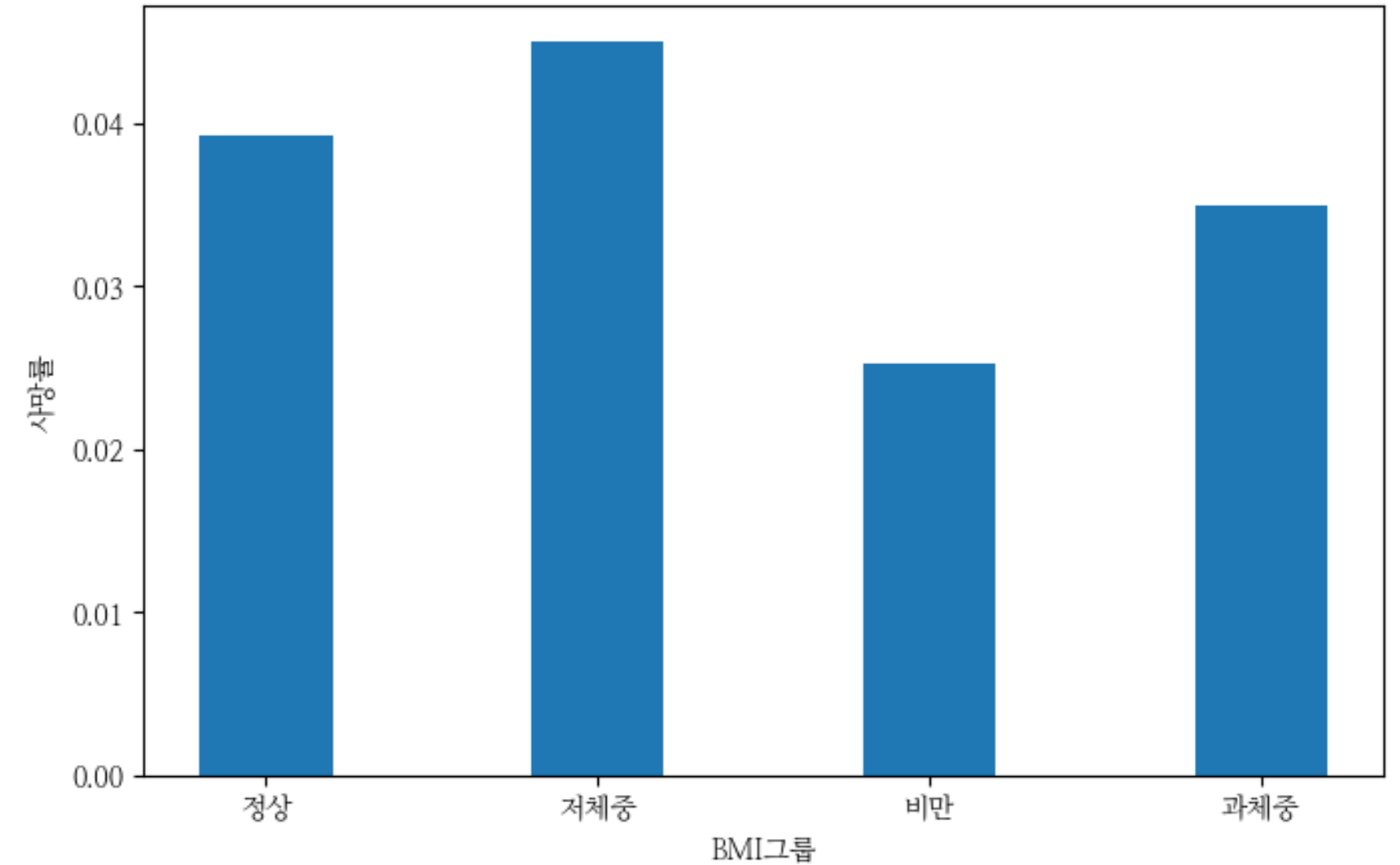
4.BMI 데이터

BMI그룹별 환자수



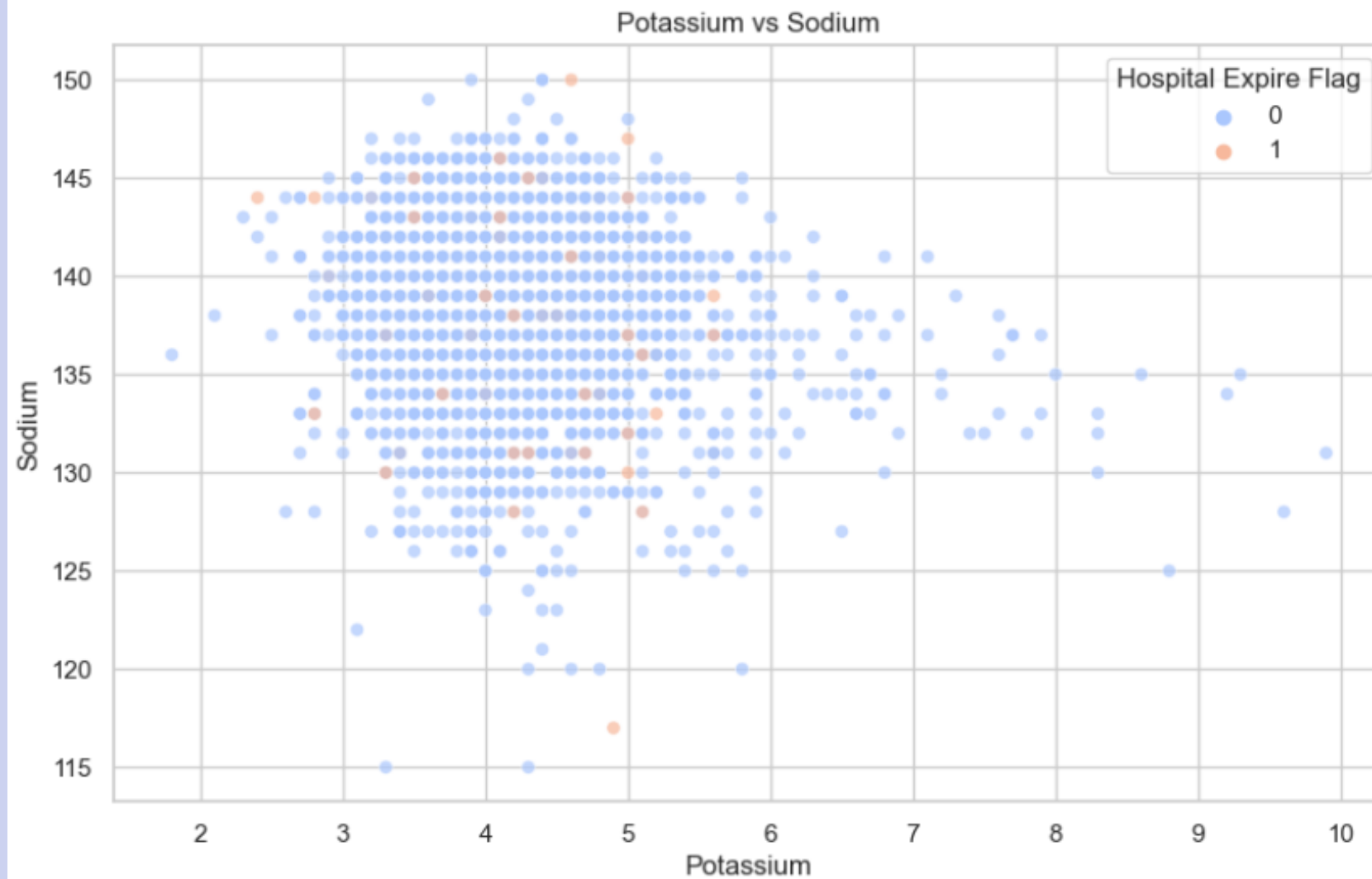
가장 많은 환자가 '비만' BMI 그룹에 속하고 있음을 확인할 수 있습니다.
'저체중' 그룹의 환자 수가 가장 적음을 알 수 있습니다.

BMI그룹별 사망률



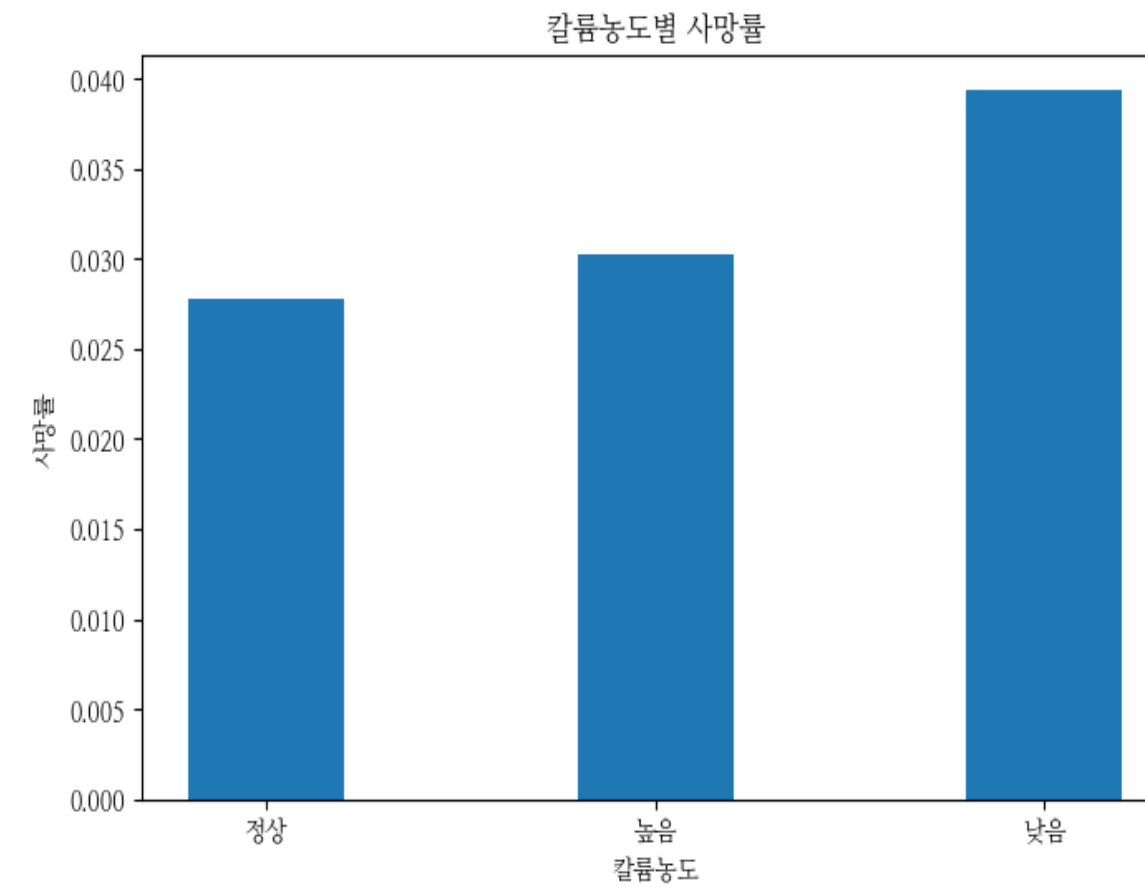
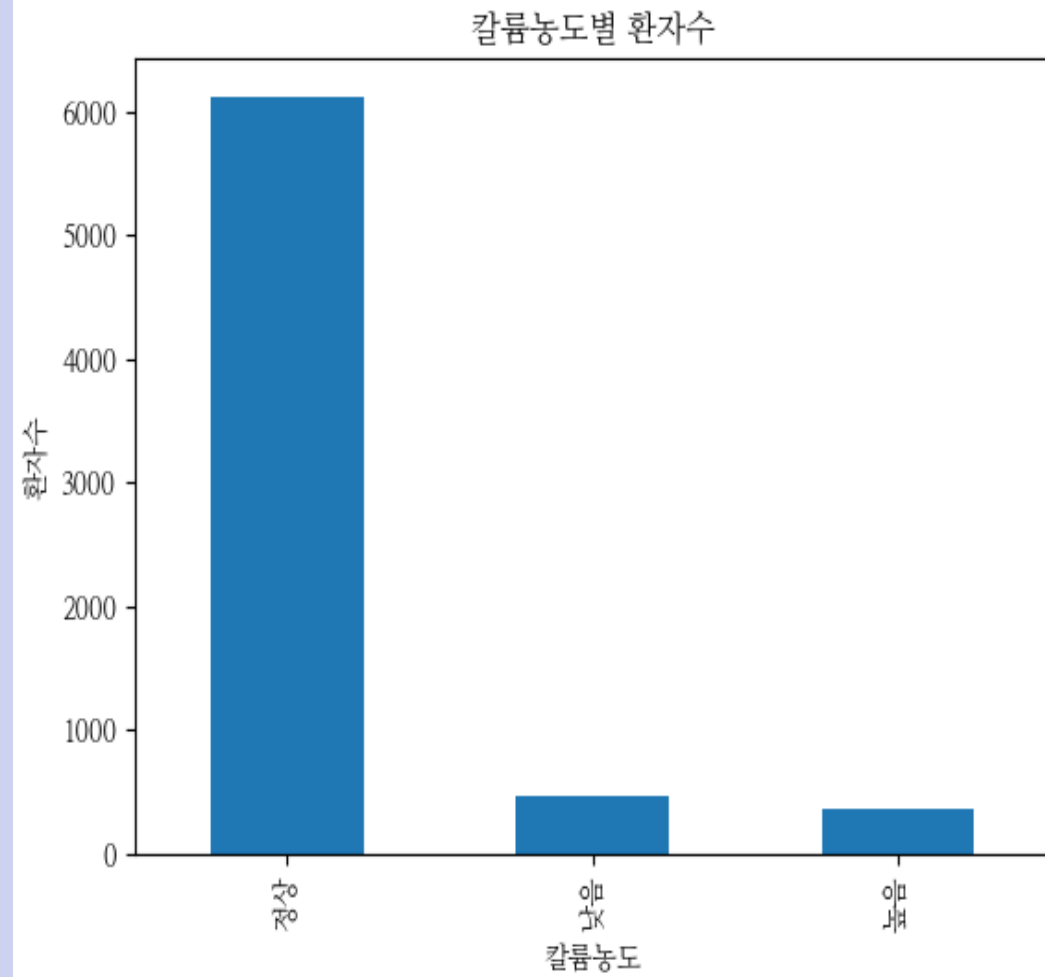
저체중에 고혈압을 가진 암환자가 가장 사망률이 높다

5. 체내 칼륨과 나트륨 농도

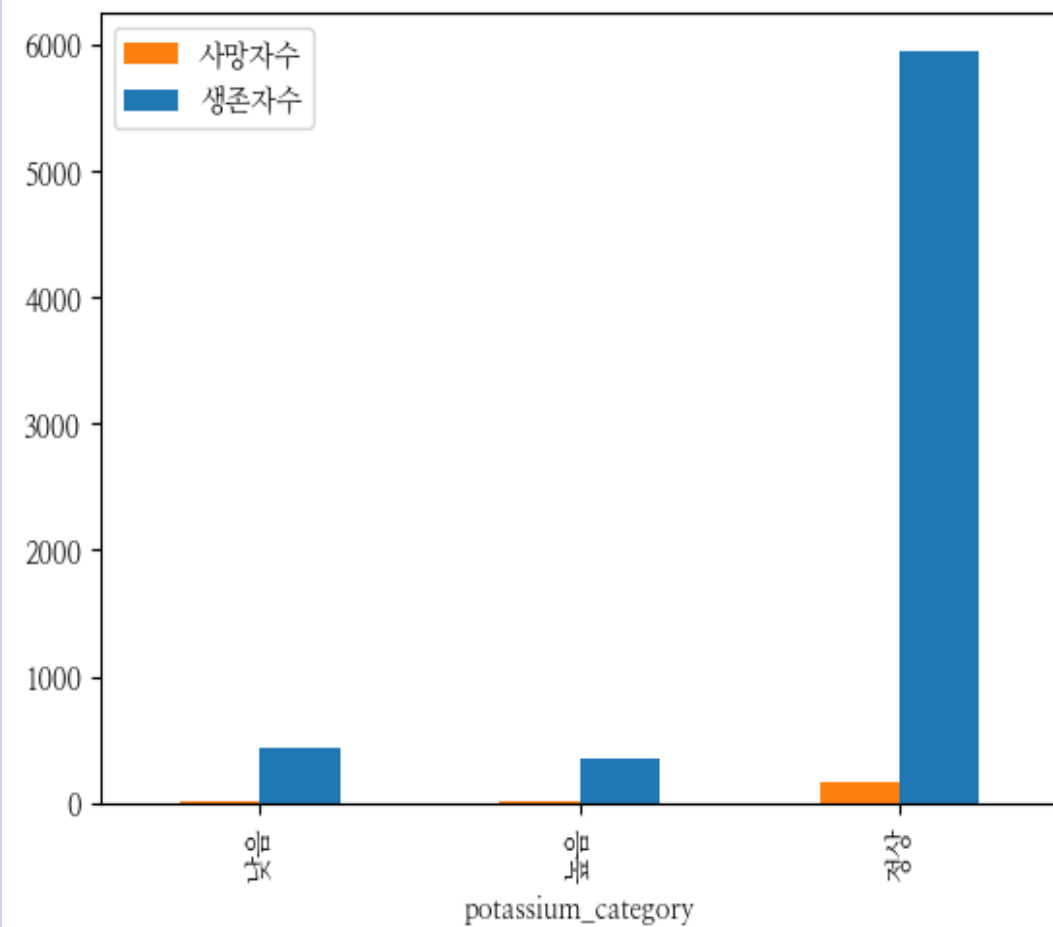


x축은 체내 칼륨 농도를,
y축은 체내 나트륨 농도를 나타냅니다.
데이터 포인트는 각 환자를 나타내며,
생존 여부에 따라 색상이 달라집니다.
(생존자 파란색, 사망자 붉은색)

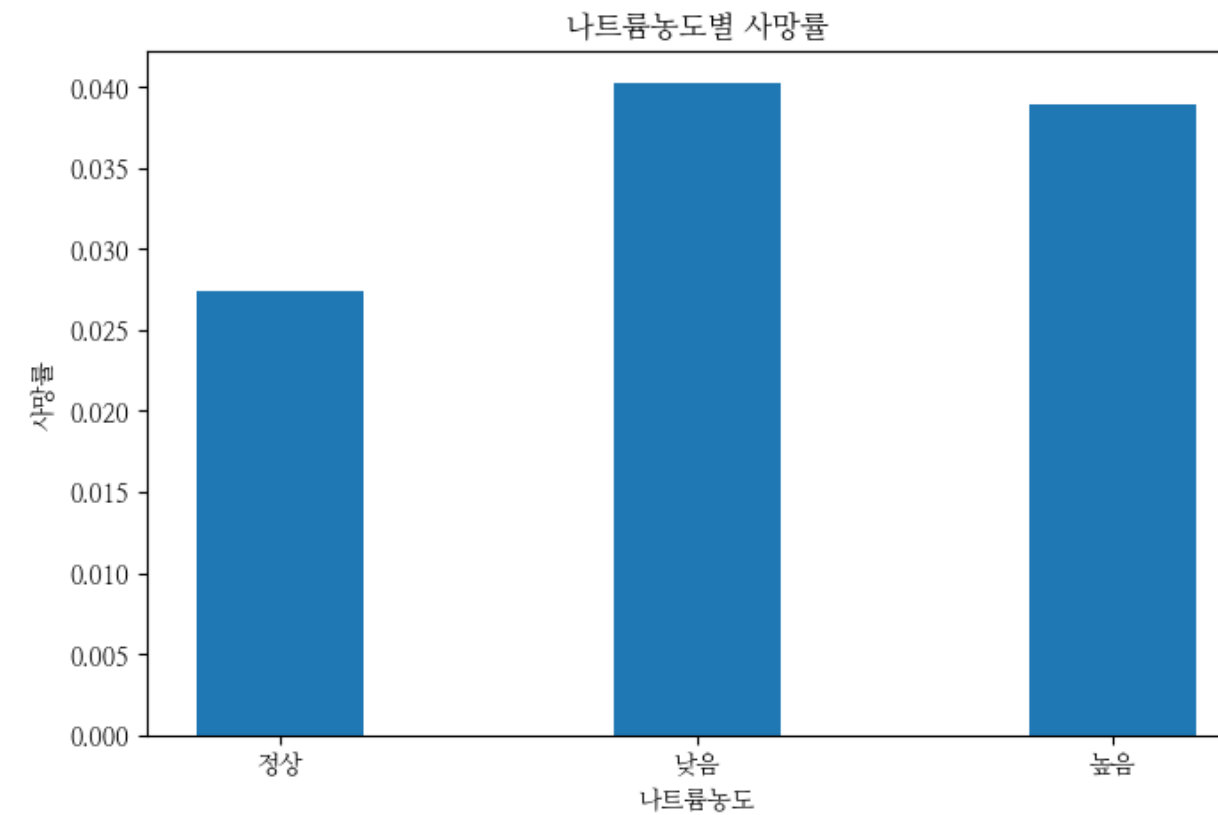
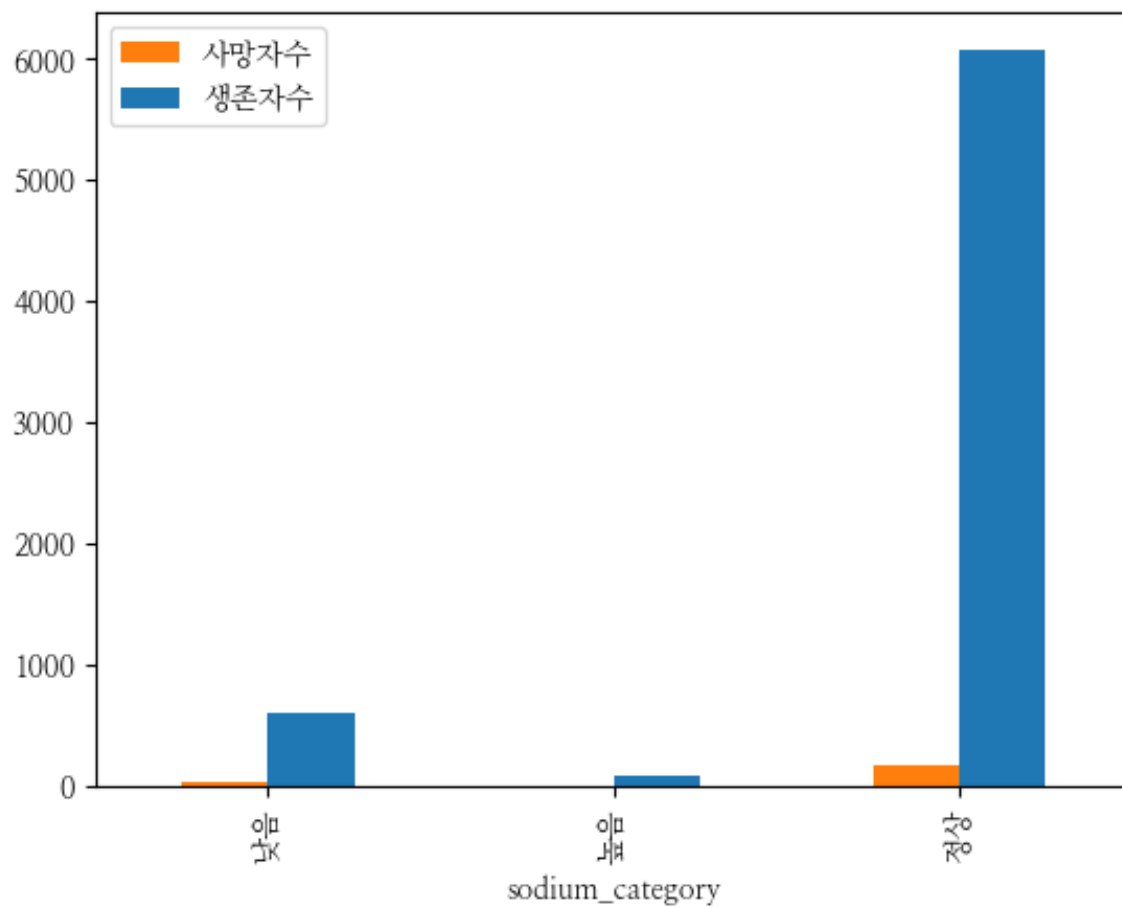
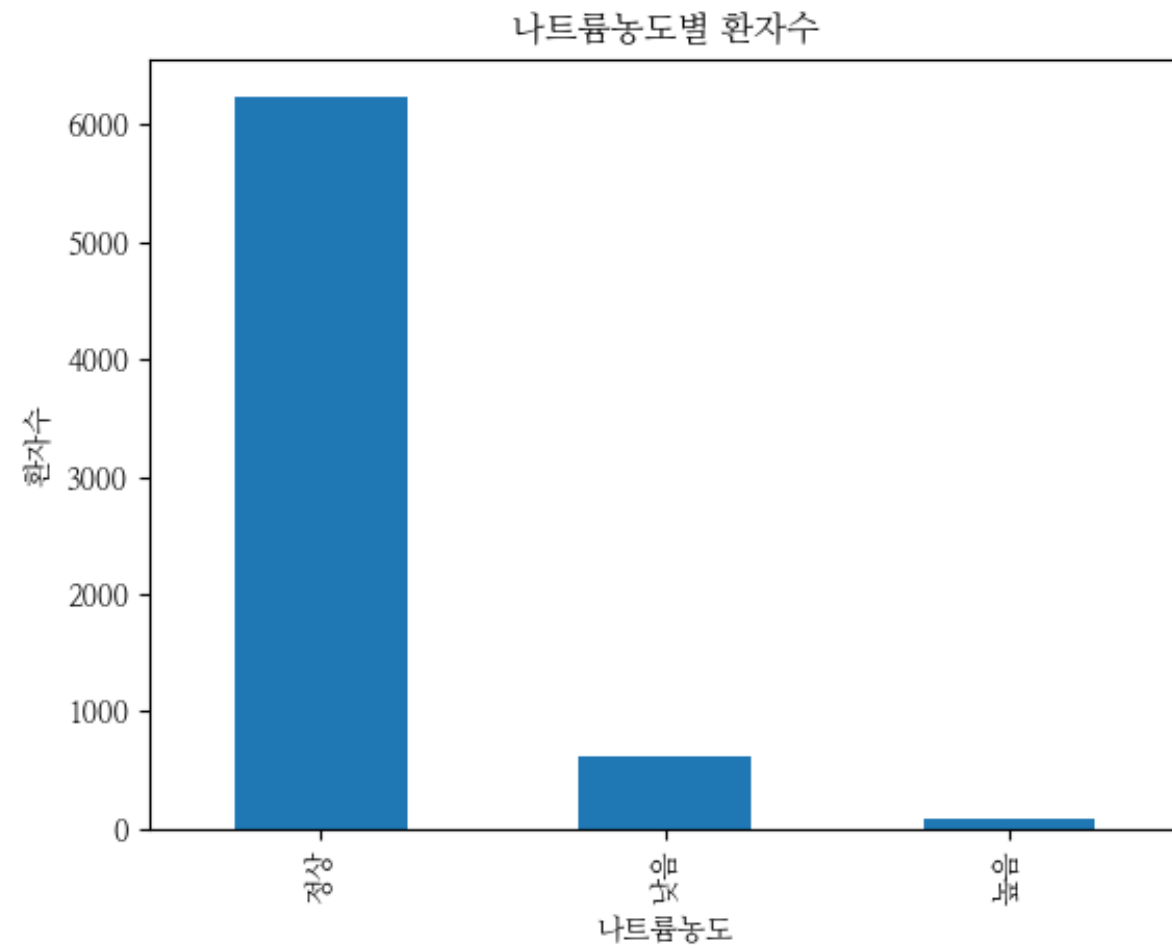
5. 체내 칼륨 농도



칼륨 농도에 따른 사망률을 막대 그래프로 나타낸 것입니다.
칼륨 농도가 '낮음'인 그룹에서 상대적으로 높은 사망률을 보입니다.

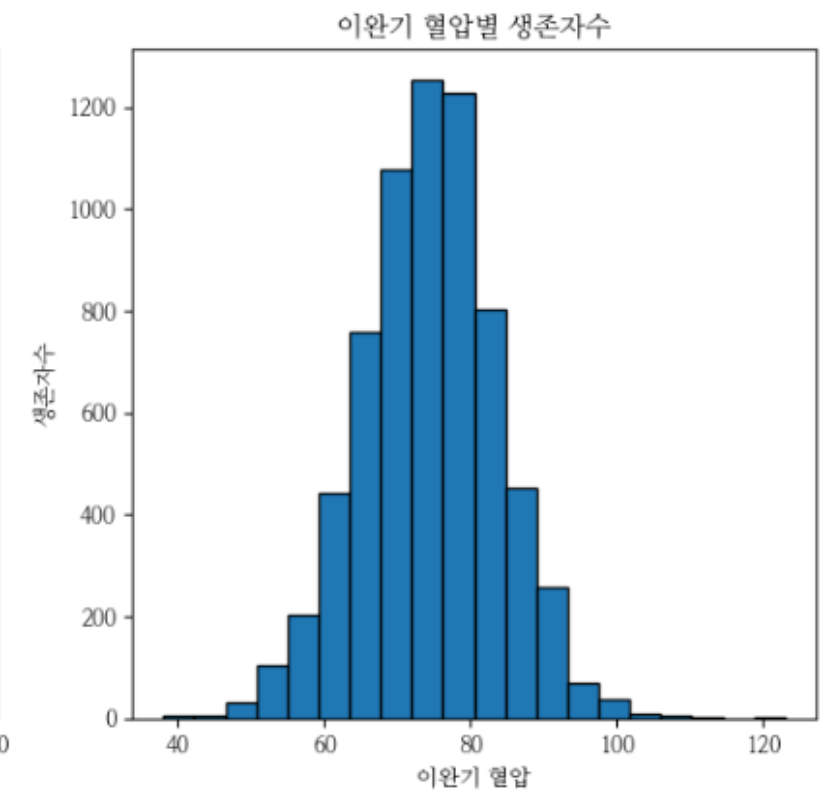
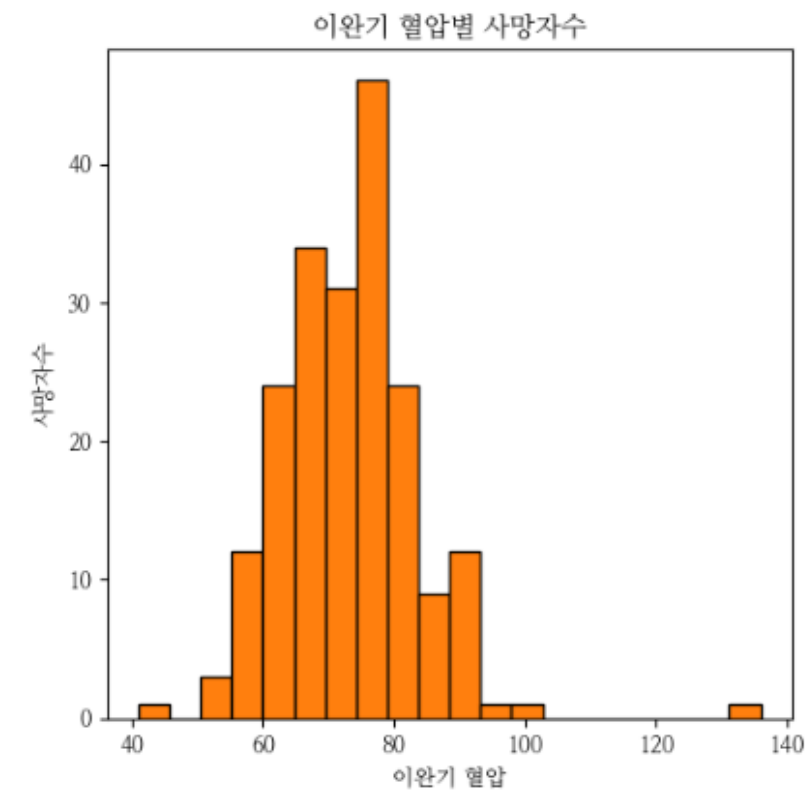
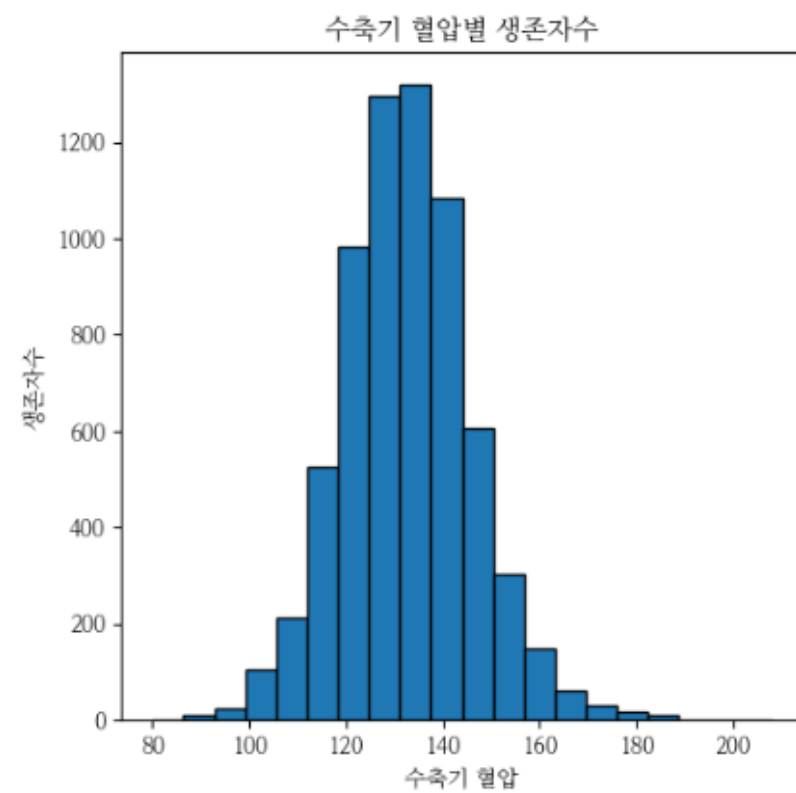
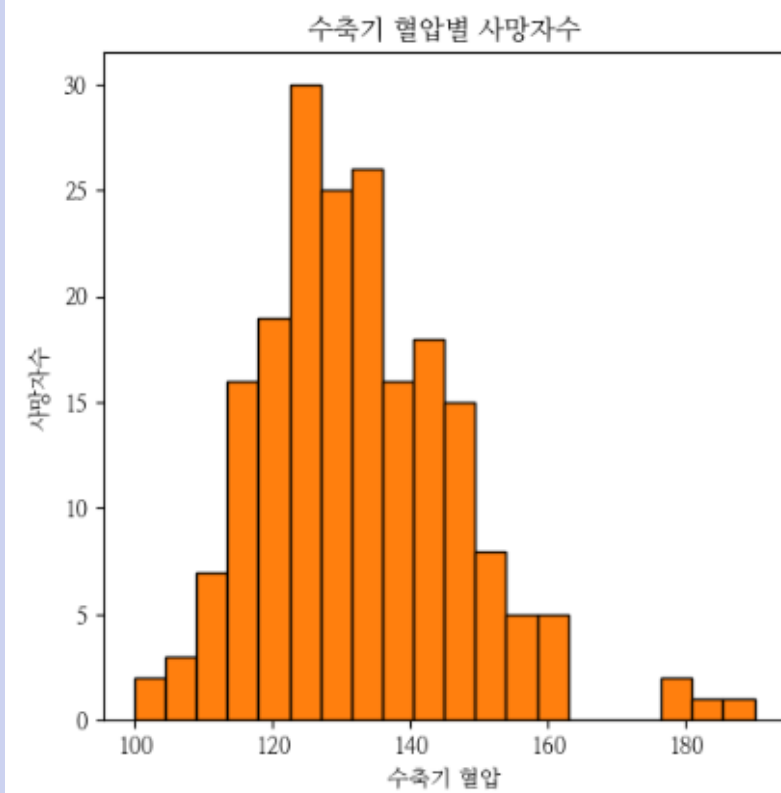


5. 체내 나트륨 농도



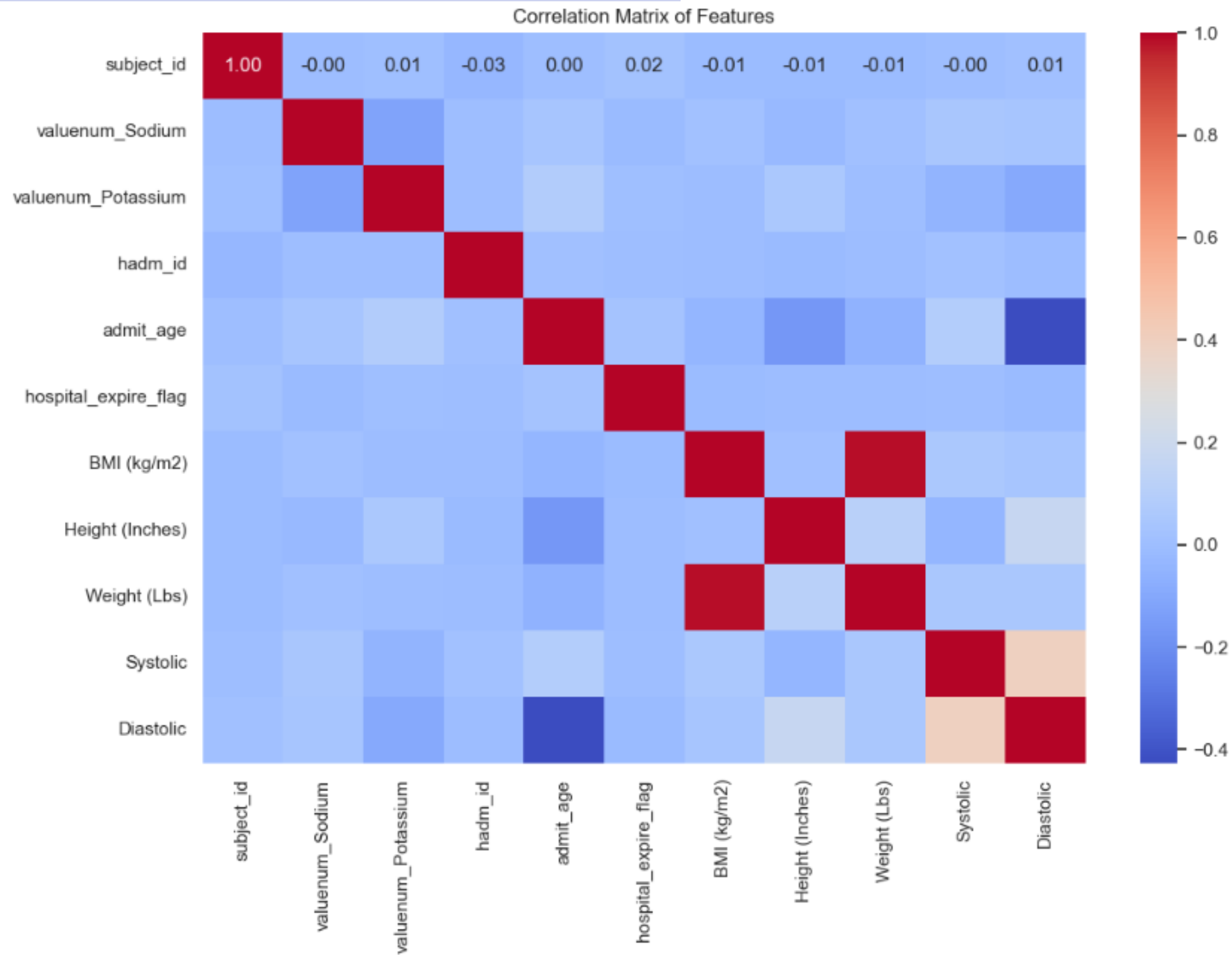
나트륨 농도에 따른 사망률을 막대 그래프로 나타낸 것입니다.
나트륨 농도가 '낮음'인 그룹에서 상대적으로 높은 사망률을 보입니다.

6. 혈압에 따른 사망, 생존 분포



비정상 혈압일 때 사망자 수가 늘어남을 보임

7. 변수간 상관관계

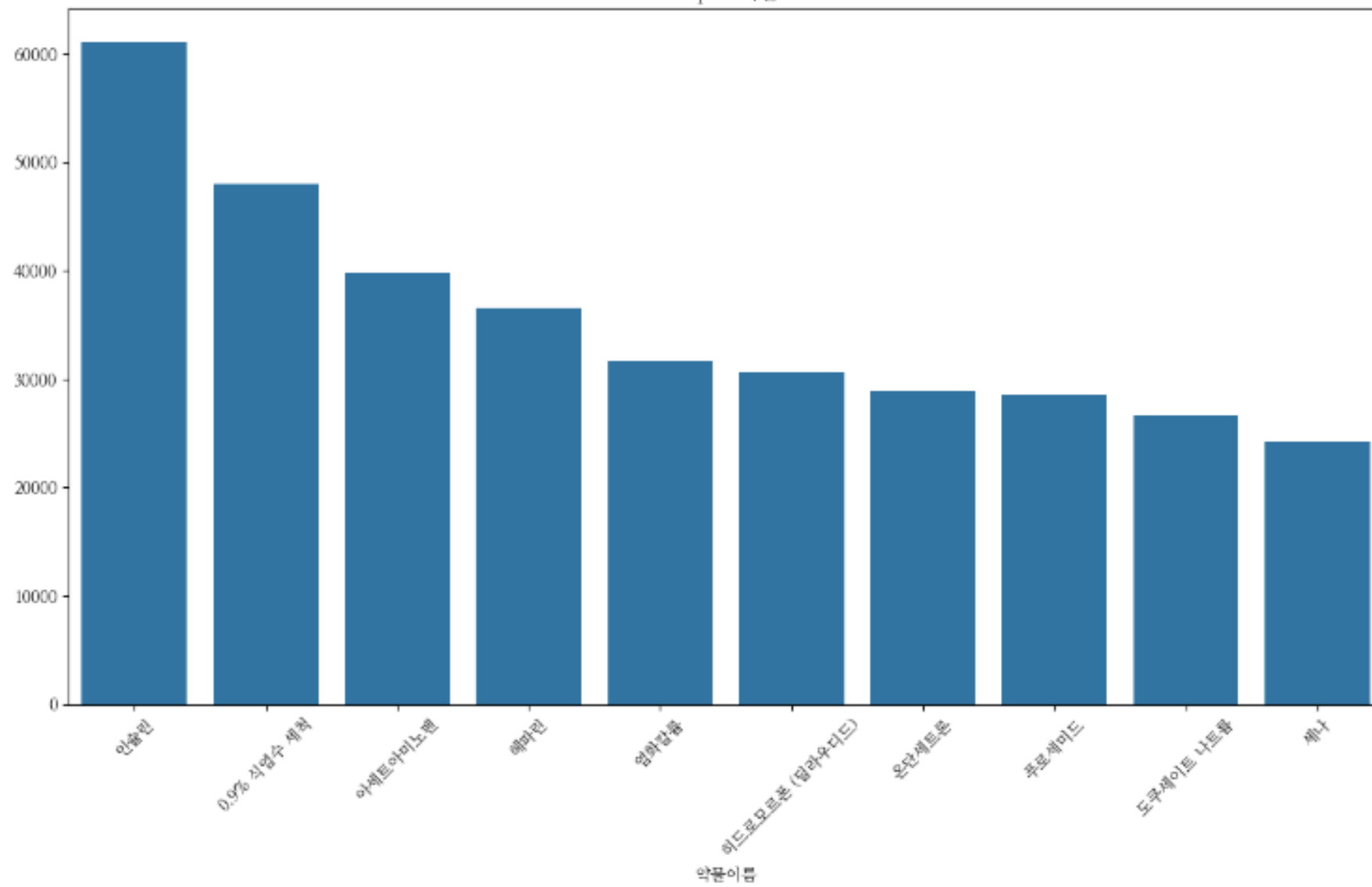


신장이 일정할 때, 체중의 변화가 BMI에 직접적으로 반영되기 때문에 두 변수 사이의 상관관계는 매우 높게 나타납니다.

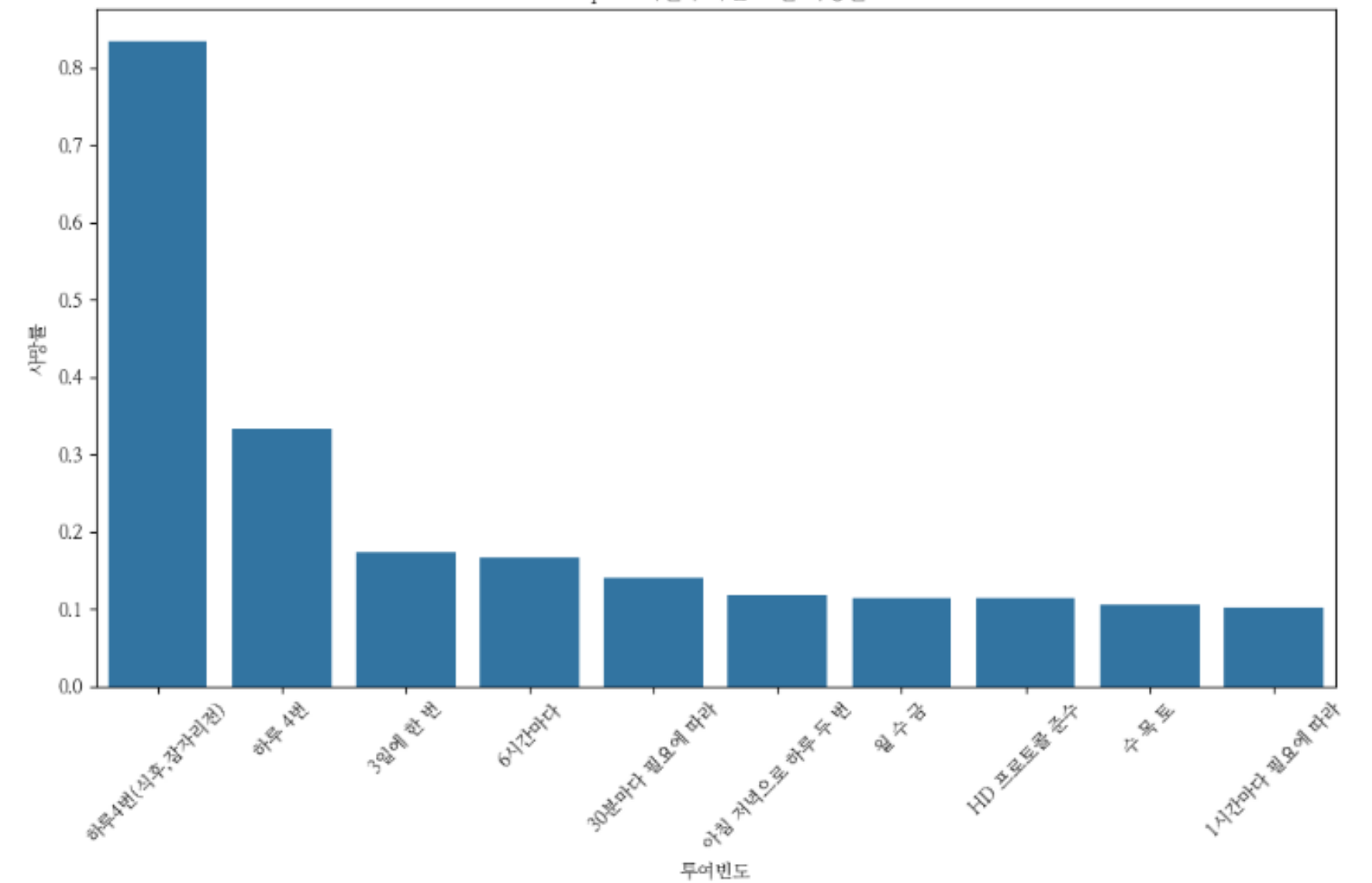
상대적으로 "admit_age"와 "diastolic"의 상관관계가 높게 나타나고 있습니다

8. 빈도 상위10개 약물

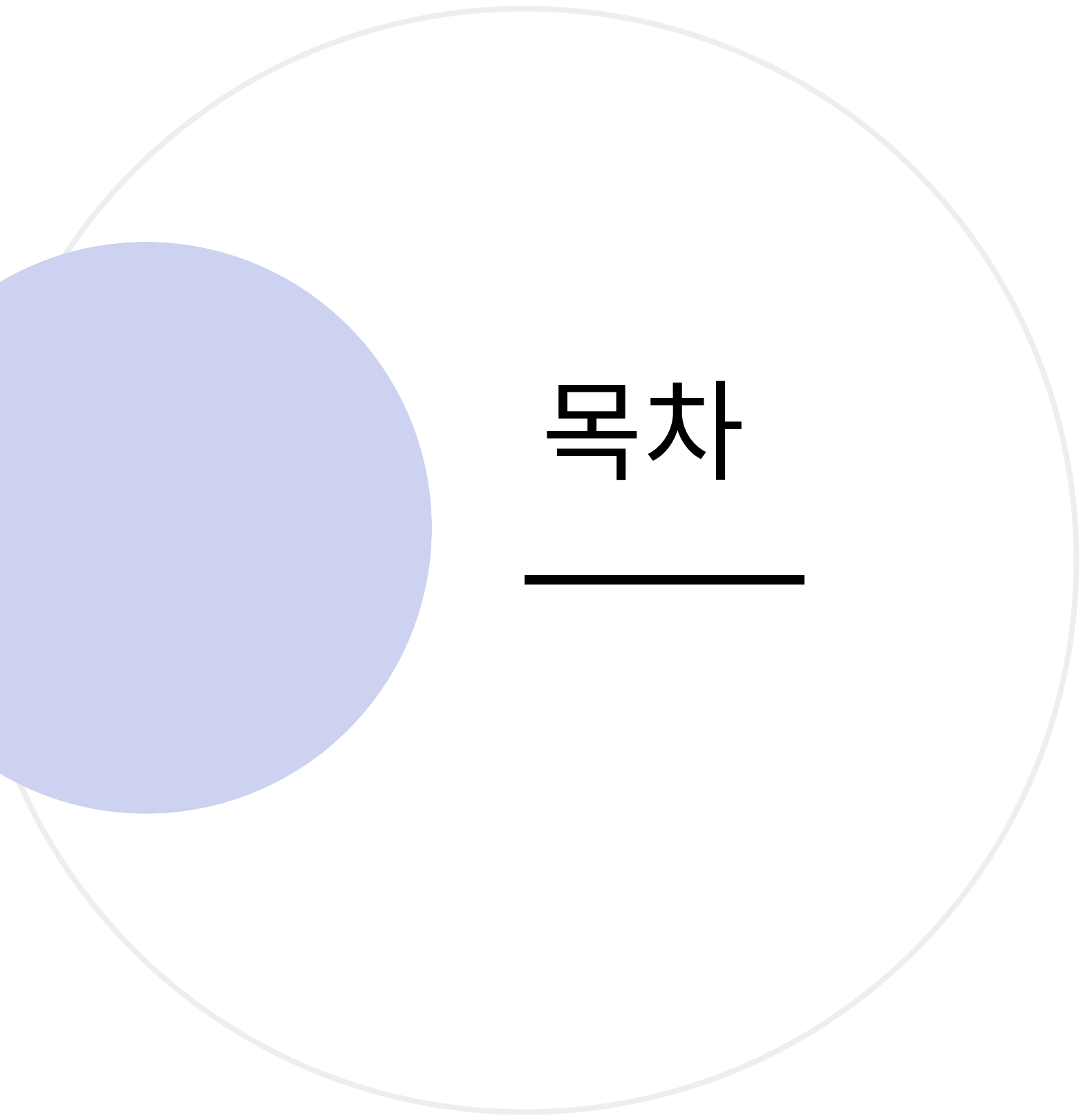
Top 10 약물



Top 10 약물투여빈도 별 사망률



고혈압인 암환자 사망 예측 머신러닝



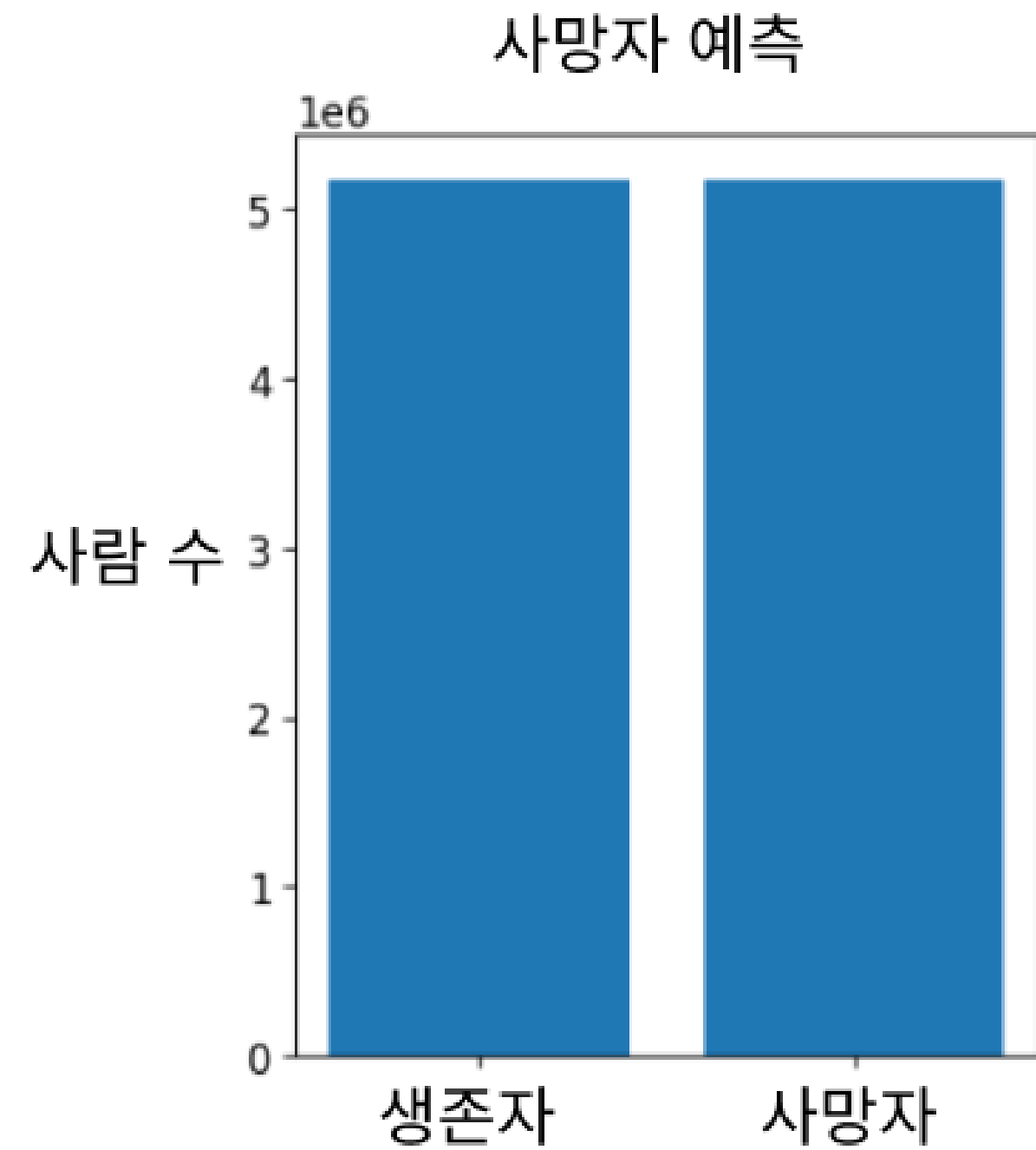
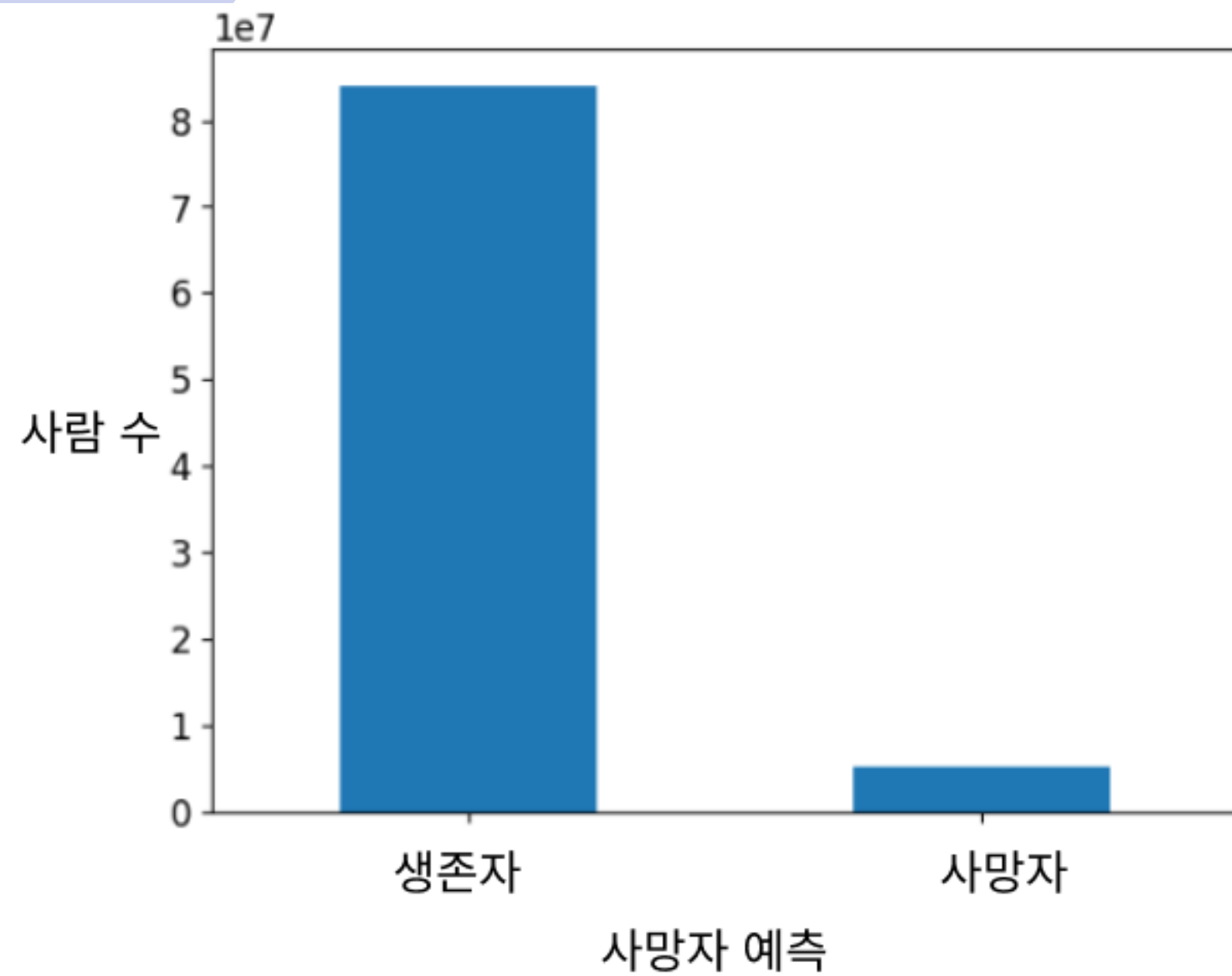
목차

- 1 전처리
- 2 모델링
- 3 Hard Voting

전처리

- 1 nan값 전처리(nan값을 Unkown로 처리)
- 2 under sampling(불균형한 라벨 수를 맞춰주기)
- 3 데이터 스케일링(데이터를 정규분포로 만들기)

under sampling



데이터 스케일링

Robust Scaler 사용

이상치에 대한 민감도가 낮기 때문에
이상치가 있는 데이터를 스케일링 하는데 유용합니다

	admit_age	gender	insurance	language	marital_status	race	hospital_expire_flag	medication	frequency	drug
0	-0.928571	1	2	1	1	28	0	50	95	34
1	-0.071429	0	2	1	1	28	0	806	16	860
2	-2.071429	0	2	1	1	28	0	50	95	72
3	-0.571429	1	2	1	2	28	0	15	16	1786
4	-1.000000	1	2	1	0	28	0	458	91	503
...
10350291	-0.785714	1	2	1	2	28	1	1552	121	1659
10350292	-0.785714	1	2	1	2	28	1	1552	121	1659
10350293	-0.785714	1	2	1	2	28	1	1552	121	1659
10350294	-0.785714	1	2	1	2	28	1	1552	121	1659
10350295	-0.785714	1	2	1	2	28	1	1552	121	1659

행 10350295
열 10개

1. **admit_age**: 환자의 입원 시 나이
2. **gender**: 환자의 성별
3. **insurance**: 환자의 보험 종류
4. **language**: 환자가 사용하는 언어
5. **marital_status**: 환자의 결혼 여부
6. **race**: 환자의 인종
7. **hospital_expire_flag**: 환자가 입원 중 사망했는지 여부를 나타내는 플래그
8. **medication**: 환자가 투약 중인 약물의 종류
9. **frequency**: 약물 투여의 빈도
10. **drug**: 특정 약물의 이름이나 코드

데이터

- 데이터 준비 `'y = df_dh['hospital_expire_flag']':`
타겟 변수로 활용할 'hospital_expire_flag' 열을 설정합니다.
`'X = df_dh.drop('hospital_expire_flag', axis=1)':`
타겟 변수를 제외한 나머지 열을 설명 변수로 설정합니다.
- 데이터 분할 `train_test_split(X, y, train_size=0.8, random_state=42):`
전체 데이터를 80%의 훈련 데이터와 20%의 테스트 데이터로 분할합니다.
- 데이터 변환 `'to_cupy()', 'astype('float32')':` cuDF 데이터를 CuPy 배열로 변환하고,
데이터 타입을 float32로 변환하여 GPU 가속을 활용합니다.

모델링

사용한 라이브러리

sklearn

cuml

GPU 가속: cuml은 NVIDIA GPU를 활용하여
머신러닝 모델을 학습하고 예측할 수 있습니다.

(cuml이 sklearn보다 실행시간이 약10배 정도 빨랐음)

모델링

동일하게 전처리된 데이터 사용

GridSearchCV를 활용하여 최적의 하이퍼파라미터를 탐색.

평가지표

f1스코어 사용

accuracy 사용



모델링

randomforest

lightgbm

AdaBoost

xgboost

catboost

randomforest

Accuracy: 0.8335115890360666
F1 Score: 0.8430269869458261

lightgbm

Accuracy: 0.9559462044578418
F1 Score: 0.9568415801942824

AdaBoost

Accuracy: 0.8790247625672686
F1 Score: 0.8838944365060305

xgboost

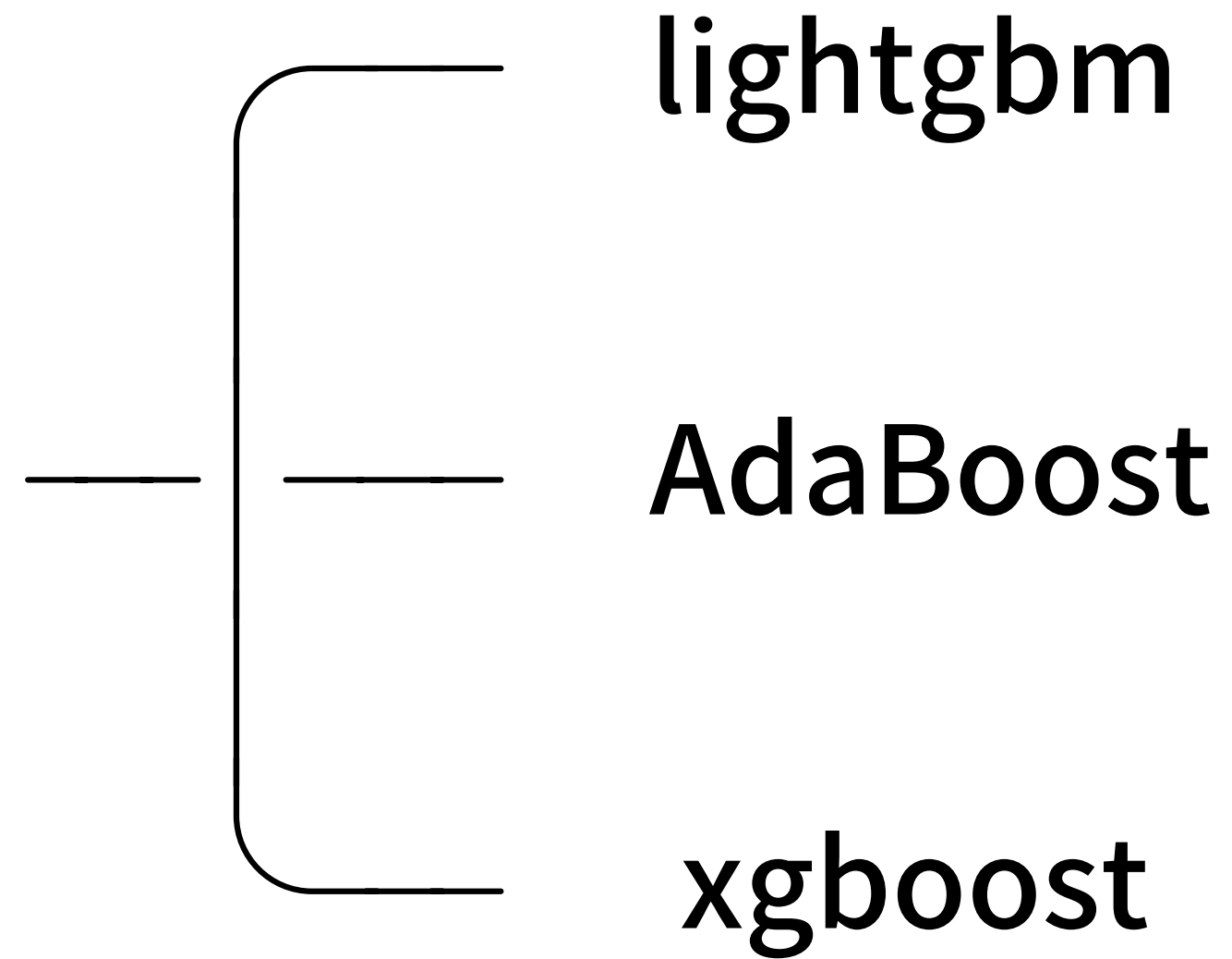
Accuracy: 0.9360583751195618
F1 Score: 0.9375277581251956

catboost

Accuracy: 0.8688207105108064
F1 Score: 0.873493905742713

Hard Voting

각 모델 만든 예측값들을 다수결 투표해,
가장 많은 표를 얻은 예측값으로 결정



Accuracy: 0.9271161222380028
F1 Score: 0.927467840457517



THANK YOU!

