

5주차 예습

☰ 다중 선택	예습
📅 마감일	@2025년 10월 6일

Kaggle 1. 의약품 분류 예측 모델 분석

환자의 다양한 생체 데이터(나이, 성별, 혈압, 콜레스테롤, Na-K 비율)를 기반으로 가장 적합한 의약품 (Drug)을 예측하는 **분류 모델(Classification Model)** 구축 과정을 단계별로 정리

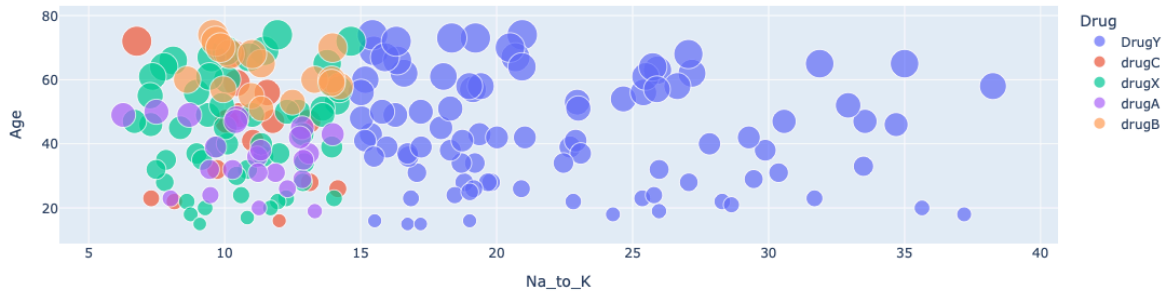
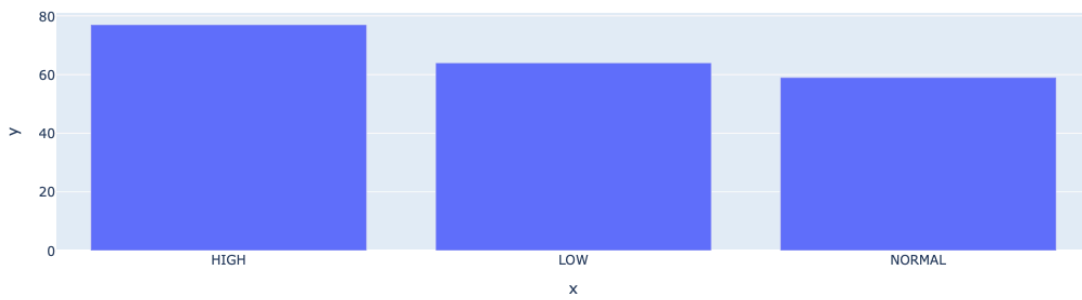
본 분석은 drug200.csv 데이터셋을 사용하며, 주요 목표는 주어진 환자 특성을 독립 변수 (Feature)로 활용하여 어떤 약물이 처방될지 예측하는 **지도 학습(Supervised Learning)** 기반의 분류 모델을 개발하고 평가하는 것이다.

핵심 분석 단계

1. **데이터 탐색 및 시각화**: 데이터의 구조와 특성 파악 및 시각화를 통한 데이터 이해
2. **데이터 전처리**: 분류 모델 학습을 위해 범주형 데이터를 수치형으로 변환
3. **모델 구축 및 평가**: **Decision Tree Classifier** 와 **Random Forest Classifier** 를 구축하고, 정확도 (Accuracy)와 혼동 행렬(Confusion Matrix)을 통해 성능을 평가

데이터 탐색 및 시각화

- **컬럼**: **Age** , **Sex** , **BP** , **Cholesterol** , **Na_to_K** (Na-K 비율), **Drug**
- **특이사항**: 초기 데이터에는 성별, 혈압, 콜레스테롤, 약물 4개의 범주형 컬럼이 존재 → 모델 학습을 위해 **수치형으로 변환**



위와 같이 산점도, 히스토그램, 막대그래프, 파이 차트 등의 시각화가 가능하다.

데이터 전처리

분류 모델 학습을 위해 모든 범주형 데이터를 수치형으로 **레이블 인코딩**방식으로 변환했다.

```
# 예시 1: Change Cholesterol type
# HIGH = 1
# NORMAL = 0
```

```
dataclass.Cholesterol = [1 if i == "HIGH" else 0 for i in dataclass.Cholesterol]
```

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	1	0	1	25.355	4
1	47	0	2	1	13.093	1
2	47	0	2	1	10.114	1
3	28	1	1	1	7.798	3
4	61	1	2	1	18.043	4
...
195	56	1	2	1	11.567	1
196	16	0	2	1	12.006	1
197	52	0	1	1	9.894	3
198	23	0	1	0	14.020	3
199	40	1	2	0	11.349	3

전부 수치형 값이 됨

분류 모델 구축 및 평가

+) 변환된 데이터를 모델 학습과 평가를 위해 훈련 세트(x_train, y_train)와 테스트 세트(x_test, y_test)로 분할했음.

분류 모델

1. 결정 트리 분류기: 노드의 불순도를 낮추는 방향으로 데이터를 분할하여 예측을 수행하는 모델

- 기본 모델

```
dtc = DecisionTreeClassifier()
dtc.fit(x_train, y_train)
predict = dtc.predict(x_test)
## 정확도: 0.9667
```

- 지니 불순도 기준: 데이터가 얼마나 균일하지 않은지를 측정하며, 0에 가까울수록 순수함

```
DTC_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=0)
DTC_gini.fit(x_train, y_train)
y_pred_gini = DTC_gini.predict(x_test)
# 테스트 세트 정확도: 0.9000
```

```
y_pred_train_gini = DTC_gini.predict(x_train)
```

```
# 훈련 세트 정확도: 0.9143
```

```
array([3, 3, 2, 4, 2, 3, 3, 3, 4, 2, 0, 3, 4, 3, 4, 3, 4, 3, 4, 4, 4, 4,
       3, 4, 2, 4, 3, 3, 2, 3, 4, 2, 0, 0, 3, 3, 3, 3, 3, 4, 4, 4, 4, 3,
       3, 0, 0, 2, 4, 3, 4, 3, 4, 4, 3, 3, 4, 4, 2, 4, 4, 4, 2, 3, 4, 4,
       4, 3, 4, 3, 3, 2, 3, 2, 4, 4, 4, 4, 3, 4, 0, 3, 3, 4, 0, 4, 0, 4,
       4, 0, 4, 4, 2, 3, 4, 3, 2, 4, 3, 4, 4, 3, 4, 4, 4, 4, 4, 4, 3, 4,
       4, 2, 3, 3, 3, 4, 3, 4, 4, 4, 0, 4, 2, 3, 3, 2, 4, 4, 4, 4, 4, 3,
       2, 4, 3, 4, 2, 3, 2, 3])
```

y_pred_train_gini 실행 시 출력되는 혼동 행렬

- 엔트로피 기준: 엔트로피는 데이터 세트 내의 불확실성을 측정하며, 값이 높을수록 불순함

```
DTC_en = DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=0)
DTC_en.fit(x_train, y_train)
y_pred_en = DTC_en.predict(x_test)
# 정확도: 0.9000
```

혼동 행렬: 지니 모델과 동일한 결과 보임. 훈련 세트 정확도 역시 0.9143으로 동일.

2. 랜덤 포레스트 분류기: 다수의 결정 트리를 생성하고 그들의 예측을 종합하여 최종 결정을 내리는 앙상블 학습 모델

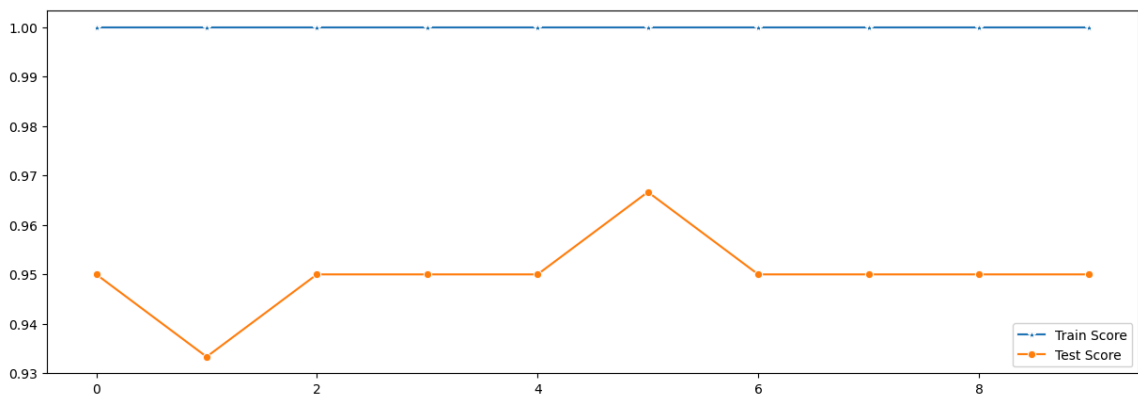
- 최적 파라미터 탐색

- **random_state** : 0부터 9까지의 값 중 **5**가 훈련/테스트 점수의 차이가 적고 테스트 점수가 높게 유지되는 **최적의 값**으로 확인됨

```
# 최적의 random_state 값 찾기
test_score_list = []
train_score_list = []

for i in range(0,10):
    rfc2 = RandomForestClassifier(random_state=i)
    rfc2.fit(x_train, y_train)
    test_score_list.append(rfc2.score(x_test, y_test))
    train_score_list.append(rfc2.score(x_train, y_train))

plt.figure(figsize=(15,5))
p = sns.lineplot(x=range(0,10), y=train_score_list, marker='*', label='Train Score') # 버전에 따른 수정
p = sns.lineplot(x=range(0,10), y=test_score_list, marker='o', label='Test Score')
```



이렇게 시각화됨 - 5가 최적!

- **n_estimators (트리 개수)**: 10부터 100까지의 값 중 100개 이상에서 테스트 점수가 가장 높고 안정적인 **최적의 개수**로 확인됨
- 최종 모델 (**n_estimators=100** , **random_state=5**)
 - 훈련 세트 정확도: 1.0000
 - 테스트 세트 정확도: 0.9667

모델별 최종 성능 비교

모델	훈련 세트 정확도	테스트 세트 정확도	주요 평가 지표
결정 트리 (기본)	0.9143	0.9000	높은 정확도
랜덤 포레스트 (최적)	1.0000	0.9667	높은 정확도, 과적합 완화

혼동 행렬: 이진 분류기의 예측 결과를 표 형식으로 나타낸 것으로, 참값이 알려져 있을 때 테스트 데이터 세트에서 분류 모델의 성능을 설명 (특히 다중 클래스 분류 문제에서 각 클래스 별로 모델이 얼마나 정확하게 예측했는지, 그리고 어떤 클래스를 다른 클래스로 오분류했는지를 상세하게 보여줌)

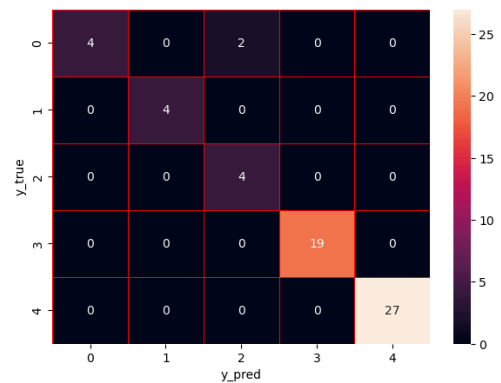
*기본 모델 분석 (예시)

```
# For Decision Tree
from sklearn.tree import DecisionTree
Classifier
from sklearn.metrics import confusion
_matrix

cm_des = DecisionTreeClassifier()

cm_des.fit(x_train,y_train)
y_pred_cm = cm_des.predict(x_test)
y_true = y_test

cm_des1 = confusion_matrix( y_true, y
_pred_cm)
cm_des1
```



- **정분류:** 모델은 총 $4+4+4+19+27=58$ 건을 정확하게 예측 (대각선의 합)
- **오분류:**
 - **drugB** (클래스 0)로 분류되어야 할 2건이 **drugA** (클래스 2)로 오분류 (실제 0, 예측 2)

- **drugC** (클래스 1)로 분류되어야 할 0건이 **drugX** (클래스 3)로 오분류 (실제 1, 예측 3)
- 정확도: 대각선 합 / 60(테스트 세트 개수) = 약 0.9667 (96.67%)

정리

- **레이블 인코딩**: 문자열 형태의 범주형 데이터를 0, 1, 2 등의 정수형 숫자로 변환하여 모델이 학습할 수 있도록 준비하는 과정.
- **결정 트리**: 데이터를 특정 기준에 따라 분할하며 예측하는 모델. 노드를 나눌 때 지니 불순도(Gini Impurity)나 엔트로피를 사용하여 불순도를 최소화하는 특성을 선택함.
- **랜덤 포레스트**: 여러 개의 독립적인 결정 트리를 만들고 이들의 결과를 취합하여 최종 결과를 결정하는 **앙상블** 기법으로, 단일 트리의 과적합을 방지하고 예측 성능을 높임.
- **정확도**: 전체 예측 건수 중 모델이 정답을 맞힌 건수의 비율로, 분류 모델의 가장 기본적인 성능 지표.
- **혼동 행렬(Confusion Matrix)**: 모델의 예측과 실제 값 사이의 관계를 보여주는 표로, 각 클래스별 정분류/오분류 상태를 직관적으로 파악 가능.

Kaggle 2. 심장 질환 예측 모델 분석

핵심 분석 단계

1. **데이터 탐색 및 시각화**: 데이터의 구조와 특성 파악 및 시각화를 통한 데이터 이해.
2. **데이터 전처리**: 분류 모델 학습을 위해 범주형 데이터를 수치형으로 변환 (CatBoost의 경우 별도 전처리 없이 범주형 변수를 직접 처리).
3. **모델 구축 및 평가**: 다양한 모델(앙상블, 부스팅 등)을 구축하고, 정확도(Accuracy)를 통해 성능을 평가.

데이터 탐색 및 시각화

- 컬럼: Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, ST_Slope, HeartDisease.
- 특이사항:
 - 초기 데이터에는 **중복값이나 결측치가 없었음**.
 - 타겟 변수 **HeartDisease**: 1(질환)이 55.34%, 0(정상)이 44.66%로 **거의 균형 잡힌 데이터**

분류 모델 구축 및 평가

- **모델 비교**: Dummy Classifier를 시작으로 Logistic Regression, SVM, KNN, 앙상블 모델(Random Forest 등), 부스팅 모델(XGBoost, LightGBM, CatBoost)을 비교
- **최적 모델**: 하이퍼파라미터 튜닝을 적용한 **CatBoost** 모델이 가장 높은 정확도를 달성!

CatBoost

gradient boosting 라이브러리로, LightGBM이나 XGBoost와 마찬가지로 **트리 기반 앙상블 모델**임.

다른 라이브러리와 차별되는 장점:

- 범주형 변수 자동 인코딩 지원: One-hot encoding 불필요
- 빠른 학습 속도 + GPU 지원
- 오버피팅 방지에 강한 **Ordered boosting** 기법
- 결측치 자동 처리

즉, 데이터 전처리 부담이 적고 실무에서도 자주 쓰이는 모델.

CatBoost 모델 (Hyperparameter Tuning 적용)

- **최적화 목표**: Logloss 또는 CrossEntropy.
- **사용된 파라미터**: objective, colsample_bylevel, depth, boosting_type, bootstrap_type 등.
- **최적 모델 정확도 (Accuracy)**: **0.9058** (튜닝 전 0.8804 대비 향상).