

7주차 정리노트

<https://di-bigdata-study.tistory.com/2>

결정 트리

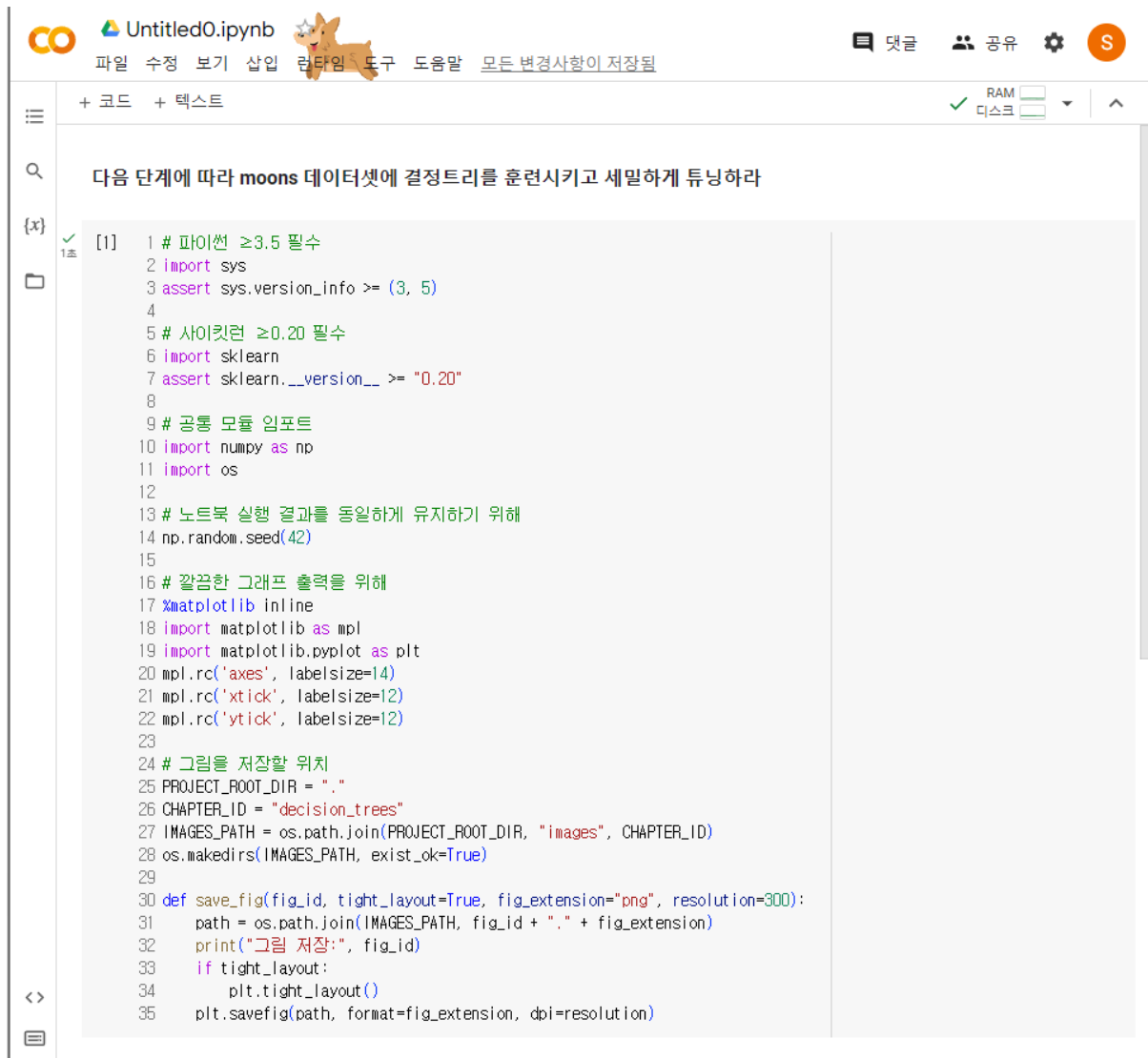
- 분류, 회귀, 다중출력 작업이 가능한 머신러닝 알고리즘
- 랜덤 포레스트의 기본 구성 요소

gini : 해당 노드의 불순도 측정값

- gini 값이 높을 수록 잘 분류되지 못한 것
- gini 값이 0이면 순수?

6.10 연습 문제(정답 확실 X)

다음 단계에 따라 moons 데이터셋에 결정트리를 훈련시키고 세밀하게 튜닝하라



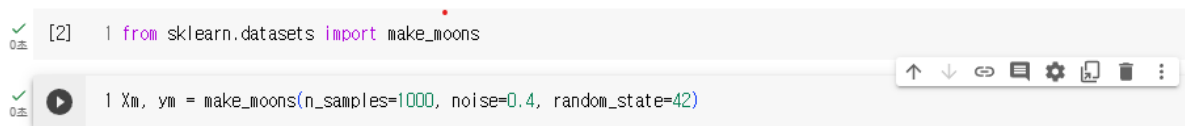
Untitled0.ipynb

다음 단계에 따라 moons 데이터셋에 결정트리를 훈련시키고 세밀하게 튜닝하라

```
[1] 1 # 파이썬 ≥3.5 필수
2 import sys
3 assert sys.version_info >= (3, 5)
4
5 # 사이킷런 ≥0.20 필수
6 import sklearn
7 assert sklearn.__version__ >= "0.20"
8
9 # 공통 모듈 임포트
10 import numpy as np
11 import os
12
13 # 노트북 실행 결과를 동일하게 유지하기 위해
14 np.random.seed(42)
15
16 # 깔끔한 그래프 출력을 위해
17 %matplotlib inline
18 import matplotlib as mpl
19 import matplotlib.pyplot as plt
20 mpl.rcParams['axes', labelsz=14)
21 mpl.rcParams['xtick', labelsz=12)
22 mpl.rcParams['ytick', labelsz=12)
23
24 # 그림을 저장할 위치
25 PROJECT_ROOT_DIR = "."
26 CHAPTER_ID = "decision_trees"
27 IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID)
28 os.makedirs(IMAGES_PATH, exist_ok=True)
29
30 def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
31     path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
32     print("그림 저장:", fig_id)
33     if tight_layout:
34         plt.tight_layout()
35     plt.savefig(path, format=fig_extension, dpi=resolution)
```

1. make_moons(n_sample=1000, noise=0.4)를 사용해 데이터셋을 생성한다.

1. make_moons(n_sample=1000, noise=0.4)를 사용해 데이터셋을 생성한다.



```
[2] 1 from sklearn.datasets import make_moons

1 X, y = make_moons(n_samples=1000, noise=0.4, random_state=42)
```

- make_moons 는 정확하게 뭔지?
 - 분류용 가상 데이터를 생성하는 함수 중 하나

- 초승달 모양 클러스터 두 개 형상의 데이터를 생성한다.
- `make_moons` 명령으로 만든 데이터는 직선을 사용하여 분류할 수 없다.
- 인수 :
 - `n_samples` : 표본 데이터의 수, 디폴트 100
 - `noise` : 잡음의 크기. 0이면 정확한 반원을 이룸

2. 이를 `train_test_split()`을 사용해 훈련 세트와 테스트 세트로 나눈다.

2. 이를 `train_test_split()`을 사용해 훈련 세트와 테스트 세트로 나눈다.

```
[5] 1 from sklearn.datasets import load_breast_cancer
    2 from sklearn.model_selection import train_test_split

1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- `train_test_split()` 함수
 - 데이터셋을 테스트 데이터와 훈련 데이터로 분류해준다.
 - `test_size` 인수는? : 훈련, 테스트 데이터의 비율을 결정함 (0.2 ?)
 - `random_state`에 값을 입력하지 않으면 랜덤한 선택으로 데이터 값을 분류한다.
특정 수(위에서는 42)로 결정하면 Seed 값이 동일해서 동일한 방식으로 데이터를 분류해준다.

3. `DecisionTreeClassifier`의 최적의 매개변수를 찾기 위해 교차 검증과 함께 그리드 탐색을 수행한다. (`GridSearchCV`를 사용하면 됨. 여러가지 `max_leaf_nodes` 값을 시도)

3. **DecisionTreeClassifier**의 최적의 매개변수를 찾기 위해 교차 검증과 함께 그리드 탐색을 수행한다. (**GridSearchCV**를 사용하면 됨. 여러가지 **max_leaf_nodes** 값을 시도)

0초

[9]

1 from sklearn.model_selection import GridSearchCV

0초

[10]

1 from sklearn.tree import DecisionTreeClassifier

0초

[11]

1 params = {'max_leaf_nodes':list(range(2, 100)), 'min_samples_split':[2,3,4]}
2 gcv = GridSearchCV(DecisionTreeClassifier(random_state=42), params, cv=3, scoring='f1', verbose=1)

4초

[12]

1 gcv.fit(Xm_train, ym_train)

Fitting 3 folds for each of 294 candidates, totalling 882 fits

GridSearchCV

GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
param_grid={'max_leaf_nodes': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23, 24, 25, 26, 27, 28, 29, 30,
31, ...],
'min_samples_split': [2, 3, 4]},
scoring='f1', verbose=1)

estimator: DecisionTreeClassifier

DecisionTreeClassifier(random_state=42)

DecisionTreeClassifier

DecisionTreeClassifier(random_state=42)

0초

▶

1 gcv.best_estimator_

DecisionTreeClassifier

DecisionTreeClassifier(max_leaf_nodes=4, random_state=42)

4. 찾은 매개 변수를 사용해 전체 훈련 세트에 대해 모델을 훈련시키고 테스트 세트에서 성능을 측정한다.

4. 찾은 매개 변수를 사용해 전체 훈련 세트에 대해 모델을 훈련시키고 테스트 세트에서 성능을 측정한다.

0초

[14]

1 from sklearn.metrics import accuracy_score
2
3 ym_pred = gcv.predict(Xm_test)
4 accuracy_score(ym_test, ym_pred)

0.855

하..