레드 와인과 화이트 와인 구별해보기!

이번엔 여러 데이터(특징)으로
레드 와인과 화이트 와인을 구별해 보자!

1

먼저 로지스틱 회귀로 모델 만들어보자!

2

데이터 준비

```
import pandas as pd

wine = pd.read_csv('https://raw.githubusercontent.com/Gonteer/2024DongALINC/main/001ML/00107_DecisionTree/wine.csv')
```

✓ 1.5s

```
wine.head()
```

✓ 0.0s

	alcohol	sugar	pH	class
0	9.4	1.9	3.51	0.0
1	9.8	2.6	3.20	0.0
2	9.8	2.3	3.26	0.0
3	9.8	1.9	3.16	0.0
4	9.4	1.9	3.51	0.0

class – 0 : 레드와인(음성)
1 : 화이트 와인(양성)

3

데이터 준비

```
wine.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6497 entries, 0 to 6496
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   alcohol  6497 non-null   float64
 1   sugar    6497 non-null   float64
 2   pH       6497 non-null   float64
 3   class    6497 non-null   float64
dtypes: float64(4)
memory usage: 203.2 KB
```

```
wine.describe()
```

✓ 0.0s

	alcohol	sugar	pH	class
count	6497.000000	6497.000000	6497.000000	6497.000000
mean	10.491801	5.443235	3.218501	0.753886
std	1.192712	4.757804	0.160787	0.430779
min	8.000000	0.600000	2.720000	0.000000
25%	9.500000	1.800000	3.110000	1.000000
50%	10.300000	3.000000	3.210000	1.000000
75%	11.300000	8.100000	3.320000	1.000000
max	14.900000	65.800000	4.010000	1.000000

4

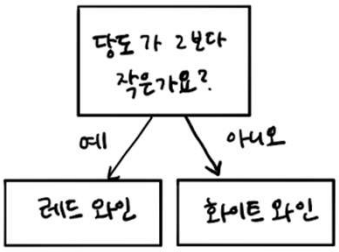
여기까진 알려드릴께! 나머지는 스스로!

그럼 설명이 쉬운 모델이 없을까?

8

결정 트리

- 설명하기 쉬운 모델 → '스무고개'와 같음
  - 질문을 하나씩 던져서 정답을 맞춰감
- 데이터를 잘 나눌 수 있는 질문을 찾는다면 계속 질문을 추가해 분류 정확도를 높임



- 사이킷런에서 제공 → DecisionTreeClassfier 클래스

9

결정트리 : 모델 학습

- 결정트리는 **표준화 전처리 할 필요가 없다**는 장점이 있음
  - 특성 값의 스케일은 결정 트리 알고리즘에겐 영향을 미치지 않음

```
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier(random_state=42)
dt.fit(train_input, train_target)

print(dt.score(train_input, train_target))
print(dt.score(test_input, test_target))
```

✓ 0.0s Python

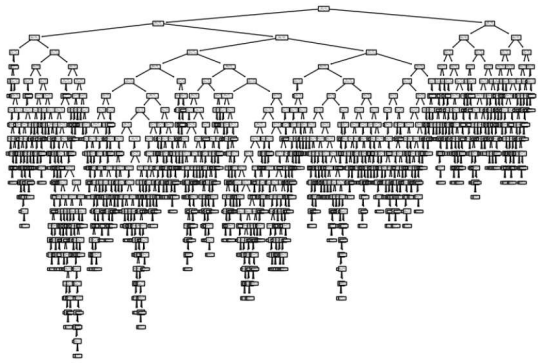
```
0.9973316912972086
0.8516923076923076
```

과대적합된 모델!

10

결정트리 : 출력해보기!

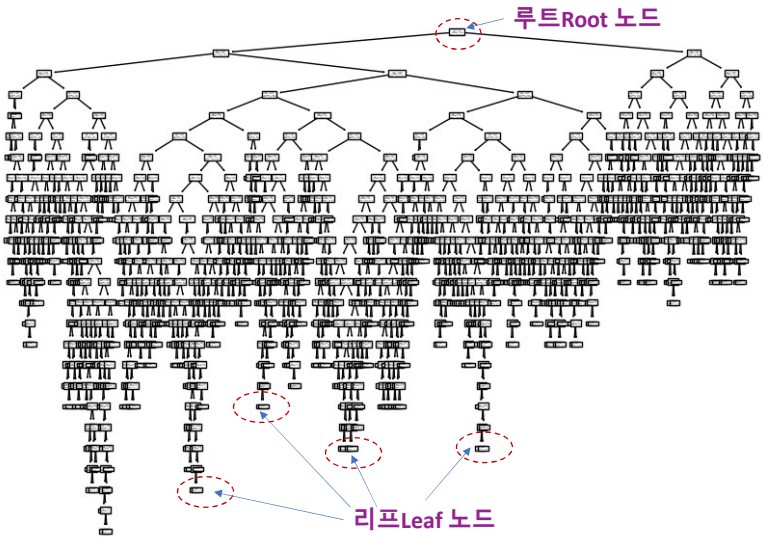
```
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

plt.figure(figsize=(10,7))
plot_tree(dt)
plt.show()
```



- 사이킷런에서 plot_tree() 함수를 사용해 결정트리를 이해하기 쉬운 트리 그림을 출력

11

결정트리 : 출력해보기!

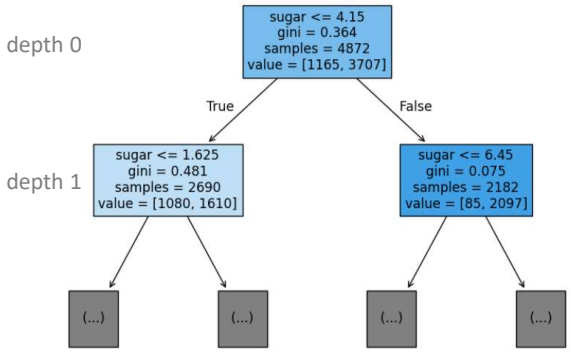


12

결정트리 분석

```
plt.figure(figsize=(10,7))
plot_tree(dt, max_depth=1, filled=True, feature_names=['alcohol', 'sugar', 'pH'])
plt.show()
```

0.0s

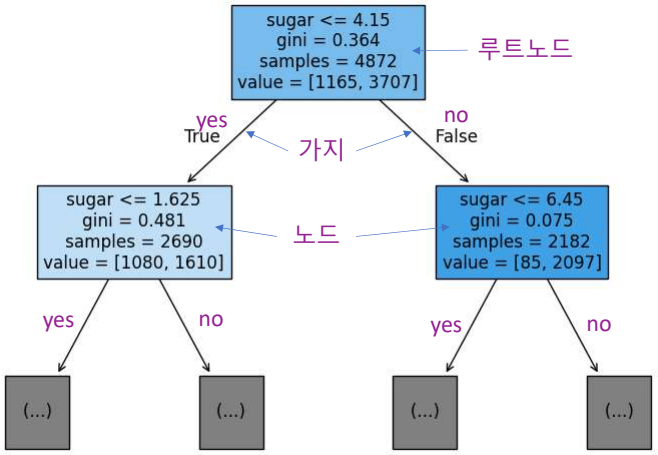


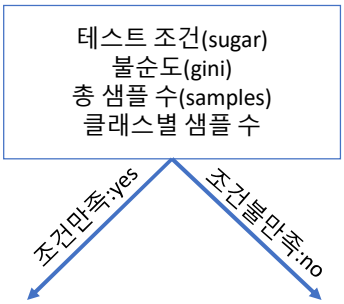
- plot_tree 매개변수
  - max_depth : 출력 노드 깊이
  - filled : 클래스에 맞게 노드 색칠함
  - feature_names : 특성의 이름 전달

13

class – 0 : 레드와인(음성)
1 : 화이트 와인(양성)

결정트리 분석





filled = True 경우
어떤 클래스의 비율이 높아지면
점점 진한색으로 표시됨

14

결정트리 분석

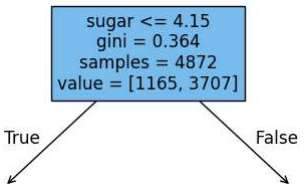
- 결정트리 예측은 리프노드에서 가장 많은 클래스가 예측 클래스가 됨(K-최근접 이웃과 비슷함)
- 만약 이 결정 트리의 성장을 멈춘다면 왼쪽 노드에 도달한 샘플과 오른쪽 노드에 도달한 샘플은 모두 양성 클래스로 예측
- 두 노드 모두 양성 클래스 개수가 많기 때문

- 근데 노드 상자 안 gini가 무엇일가?

15

지니 불순도

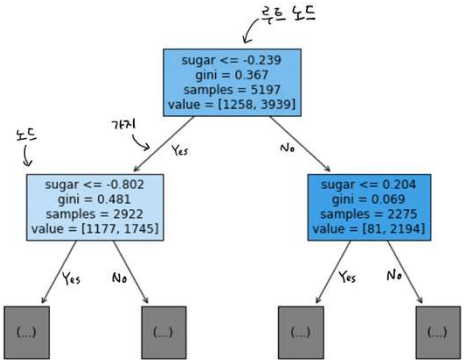
- 사이킷런 DecisionTreeClassfier 클래스의 criterion 매개변수 기본값
- criterion 매개변수 용도는 노드에서 데이터를 분할할 기준을 정하는 것
- 앞의 결과 트리에서 루트노드는 어떻게 당도 4.15를 기준으로 왼쪽과 오른쪽 노드로 나누었을까?
  - criterion 매개변수에 지정한 지니 불순도를 사용



16

class – 0 : 레드와인(음성)
1 : 화이트 와인(양성)

지니 불순도



지니불순도 = 1 − (음성클래스 비율^2 + 양성클래스 비율^2)

루트노드 $1-((\frac{1258}{5197})^2+(\frac{3939}{5197})^2)=0.367$

$1-((\frac{50}{100})^2+(\frac{50}{100})^2)=0.5$

만약에 100개의 샘플이 있는 어떤 노드 클래스 비율 ½ 경우 (최악)

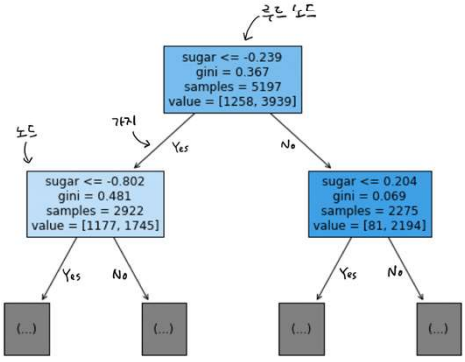
하나의 클래스만 있어 지니 불순도 0 → 순수노드 $1-((\frac{0}{100})^2+(\frac{100}{100})^2)=0$

17

지니 불순도

0.367 – (2922 / 5197) × 0.481 – (2275 / 5197) × 0.069 = 0.066

부모의 불순도 − $\frac{\text{왼쪽 노드의 샘플수}}{\text{부모의 샘플수}}$ × 왼쪽 노드의 불순도 − $\frac{\text{오른쪽 노드의 샘플수}}{\text{부모의 샘플수}}$ × 오른쪽 노드의 불순도



- 결정 트리 모델은 부모 노드와 자식 노드의 불순도 차이가 가능한 크도록 트리를 성장 시킴
- 부모와 자식 노드 사이의 불순도 차이 → **정보이득**
- 결정 트리는 정보이득이 최대가 되도록 나눔 → 이때 지니 불순도 사용함

18

엔드로피 불순도

- DicisionTreeClassifier 클래스의 criterion='entropy' 지정해 엔트로피 불순도 사용
- 노드의 클래스 비율을 사용하지만 지니 불순도와 달리 밑이 2인 로그를 사용해 곱함

-음성클래스 비율 × $\log_2$(음성 클래스 비율) – 양성 클래스 비율 × $\log_2$(양성 클래스 비율)

$$-(1258/5197) \times \log_2(1258/5197) - (3939/5197) \times \log_2(3939/5197) = 0.798$$

- 지니 불순도와 별 차이 없음(여기선 그대로 지니 불순도 사용)

19

결정 트리에서 불순도 정리!

- 결정 트리에서는 불순도 기준을 사용해 정보 이득이 최대가 되도록 노드 분할
- 노드를 순수하게 나눌수록 정보 이득 커짐
- 새로운 샘플에 대해 예측할 때에는 노드의 질문에 따라 트리 이동
- 마지막에 도달한 노드이 클래스 비율을 보고 예측을 만듬

20

가지치기

- 지금까지 트리는 제한 없이 자라남
- 현재 훈련 세트보다 테스트 세트에서 점수가 크게 낮음
  → 과대적합 상태
- 이를 위해 결정 트리도 가지치기 필요함!
- 그렇지 않으면 끝까지 자라나는 트리가 생성됨
  - 계속 이렇게 되면 훈련 세트에는 잘 맞지만, 테스트 점수는 그에 못 미침
  - 즉, 일반화가 잘 안 됨!

- 해결책 : 자라날 수 있는 트리의 최대 깊이 지정!

21

가지치기

- 해결책 : **자라날 수 있는 트리의 최대 깊이 지정**!
- 사이킷런 DecisionTreeClassifier 클래스의 max_depth 매개변수를 지정하면 됨

```
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier(max_depth=3, random_state=42)
dt.fit(train_input, train_target)

print(dt.score(train_input, train_target))
print(dt.score(test_input, test_target))
```

0.0s

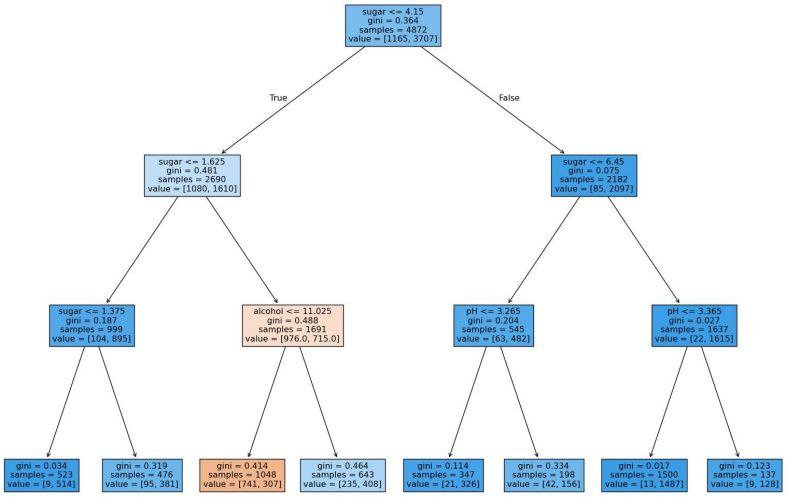
```
0.8499589490968801
0.8363076923076923
```

22

가지치기

```
plt.figure(figsize=(20,15))
plot_tree(dt, filled=True, feature_names=['alcohol', 'sugar', 'pH'])
plt.show()
```
✓ 0.2s

- 그래프 그려봄!



23

가지치기

- 결정 트리는 어떤 특성이 가장 유용한지 나타내는 특성 중요도 계산해줌
- 이 트리의 루트 노드와 깊이 1에서 당도를 사용했기 때문에 당도(sugar)가 가장 유용한 특성 중 하나
- 특성 중요도 확인
  - 모두 더하면 1이 됨
  - 각 노드의 정보 이득과 전체 샘플에 대한 비율을 곱한 후 특성별로 더하여 계산함

```
print(dt.feature_importances_)
```
✓ 0.0s
```
['alcohol', 'sugar', 'pH']
[0.12871631 0.86213285 0.00915084]
```

- 특성 중요도를 활용하면 결정 트리 모델의 특성 선택에 활용 할 수 있음

24

아쉬운 결정트리 모델... 업그레이드 할땐!

- 결정 트리 성능을 높이기 위해...
- 훈련 데이터 중 일부 검증 세트 마련, 이후 교차 검증 적용
- 다양한 매개변수, 즉, 하이퍼파라미터를 자동으로 찾는 방법 적용
  - 모델이 학습 할 수 없어 사용자가 지정해야 하는 파라미터
  - 사이킷런에서는 머신러닝 라이브러리 사용할 때 이런 하이퍼파라미터는 모두 클래스나 메소드의 매개변수로 표현함

25

감사합니다

내용 출처 정보 : https://www.hanbit.co.kr/store/books/look.php?p_code=B2002963743 https://gooopy.tistory.com/123

26