6장
결정트리(Decision Tree)

결정트리(Decision Tree)

- 분류, 회귀문제에모두적용가능함.
- 복잡한데이터셋도학습할수있음.
- 데이터전처리가필요하지않음 : 스케일링 필요없음
- 사이킷런은 CART 알고리즘을 사용 : 이진트리만 생성 – 연속값 처리, 회귀가능

  (참고 : ID3는 다중 분기 가능. 범주형 자료에 사용)
- 랜덤포레스트와 그래디언트부스팅앙상블학습의 기본학습기
- 화이트박스(White box) <--> 블랙박스 like 랜덤포레스트, 신경망
- 비모수모델(nonparametric model) : 수학적 분포함수에 의존하지 않음

  (비모수모델은 데이터 분포에 제한이 없어 복잡도가 큼 ➔ 과대적합)

Simple Example

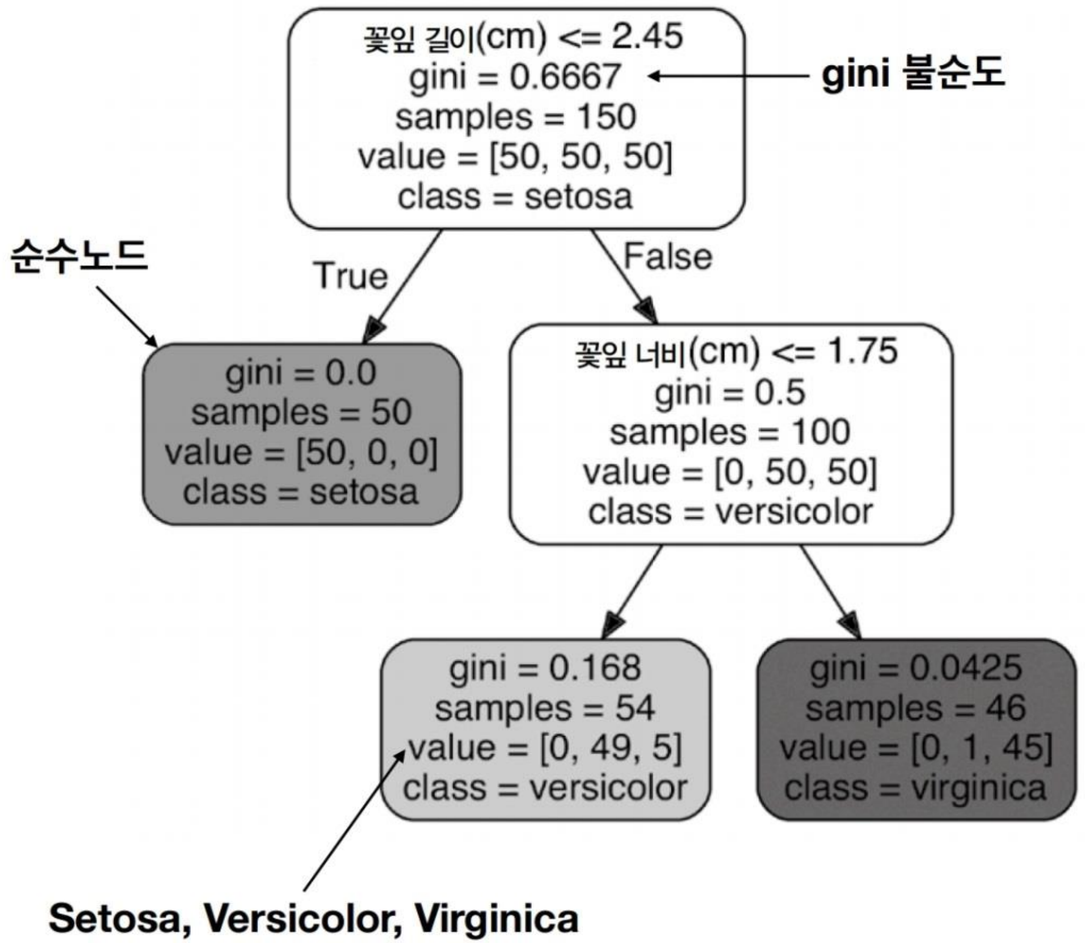
```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier

iris = load_iris()
X = iris.data[:, 2:]  # 꽃잎의 길이와 너비
y = iris.target

tree_clf = DecisionTreeClassifier(max_depth=2)
tree_clf.fit(X, y)

from sklearn.tree import export_graphviz

export_graphviz(
        tree_clf,
        out_file=image_path("iris_tree.dot"),
        feature_names=iris.feature_names[2:],
        class_names=iris.target_names,
        rounded=True,
        filled=True
    )
```

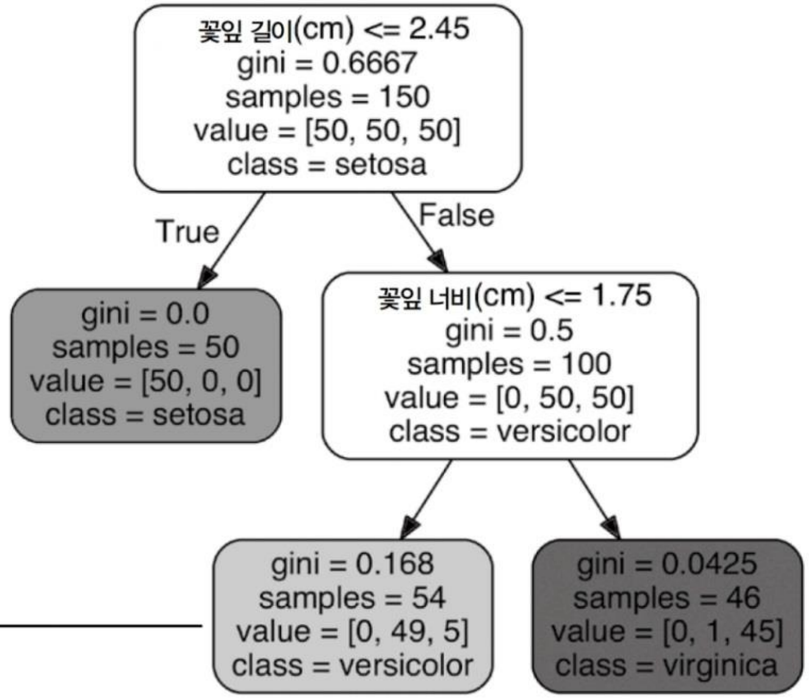


If 꽃잎길이<=2.45 and 꽃잎너비<=1.75 then virginica (확률=45/46)

지니 불순도(Gini impurity)

- 지니 불순도는 노드의 샘플 클래스가 얼마나 분산되어 있는지를 측정합니다.
- DecisionTreeClassifier(criterion='gini'), 기본값
- 최악 0.5 ~ 최상 0

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

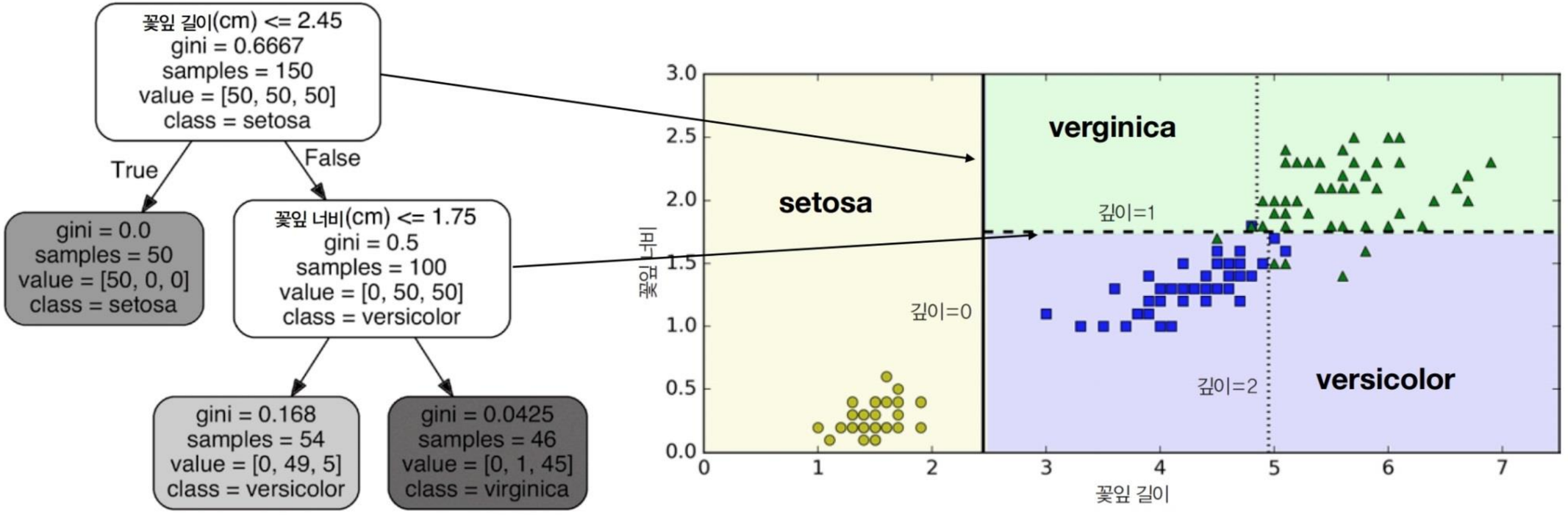


- $p_{i,k}$는 i 번째 노드에 있는 훈련 샘플 중 클래스 k에 속한 비율

$$1 - (0/54)^2 - (49/54)^2 - (5/54)^2 \approx 0.168$$

Criterion={"gini", "entropy"}, default="gini"

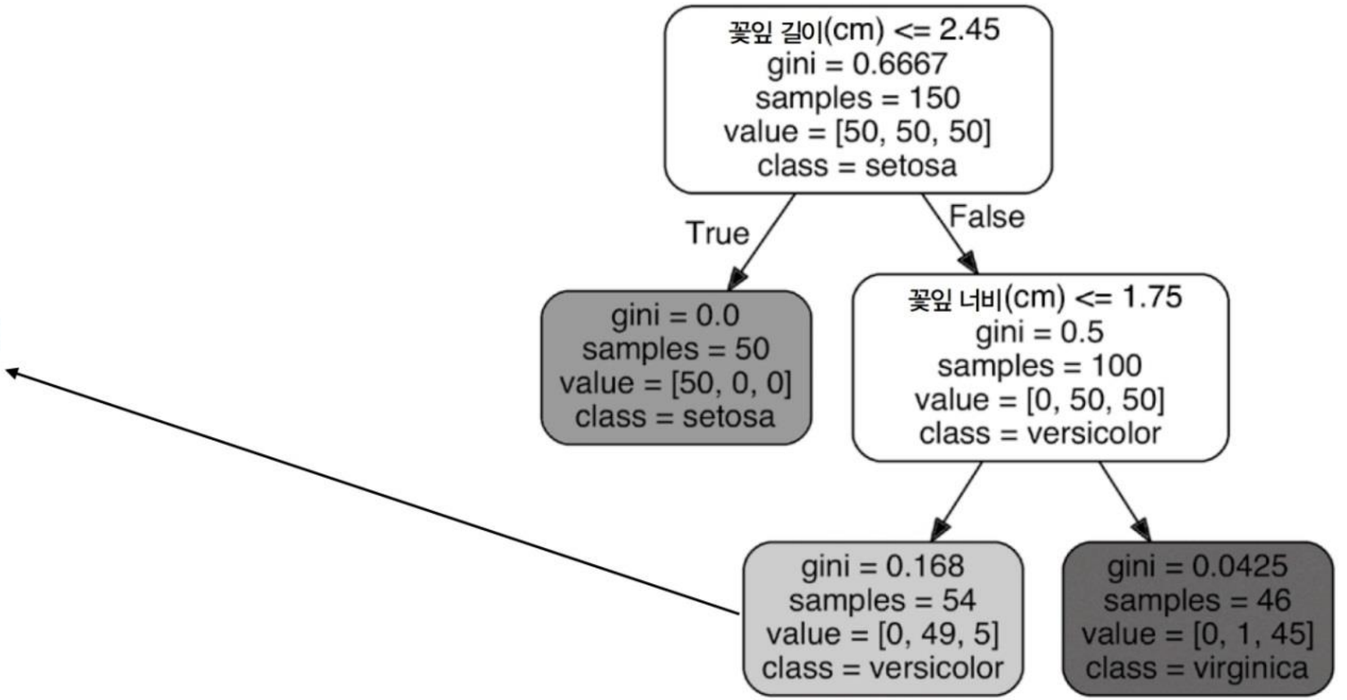
결정 경계(decision boundary)



클래스 확률 추정

- 리프 노드의 훈련 샘플의 클래스 비율
- Ex. 길이 5cm, 너비 1.5cm

```
>>> tree_clf.predict_proba([[5, 1.5]])
array([[ 0. ,  0.90740741,  0.09259259]])
>>> tree_clf.predict([[5, 1.5]])
array([1])
```



CART (Classification And Regression Tree) 알고리즘

노드분할 : 특징 k 에 임계값 t_k로 나눔. (아래 비용함수를 최소화하는 k, t_k 찾아서...)

$$J(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}$$

여기서 $G_{left/right}$ 는 왼쪽/오른쪽 서브셋의 불순도

$m_{left/right}$ 는 왼쪽/오른쪽 서브셋의 샘플 수

- 노드분할을 이용해서 root 노드(전체 훈련세트)를 둘로 나눔 : root노드의 깊이는 0
- 자식 노드들을 각각 둘로 나눔 : 깊이 1 증가
  (Gini 불순도값이 감소하지 않는 노드는 나누지 않음 ➔ 리프노드가 됨)
- 이 과정을 계속 반복 : 깊이가 max_depth가 되면 중지

계산 복잡도

깊이가 ***d*** 인 균형 이진 트리의 리프 노트 개수

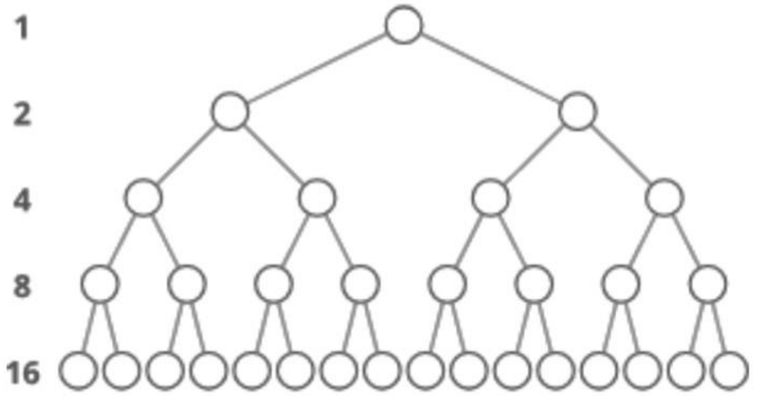
$$n = 2^d$$

리프 노트가 훈련 샘플 개수 ***m*** 개 만큼 있다면

$$m = 2^d$$

$$d = log_2 m$$

예측의 계산 복잡도 $O(log_2 m)$



노드 하나에서 특성 하나를 정렬하는 복잡도

$$mlog(m)$$

노드 하나에서 전체 특성을 정렬하는 복잡도

$$nmlog(m)$$

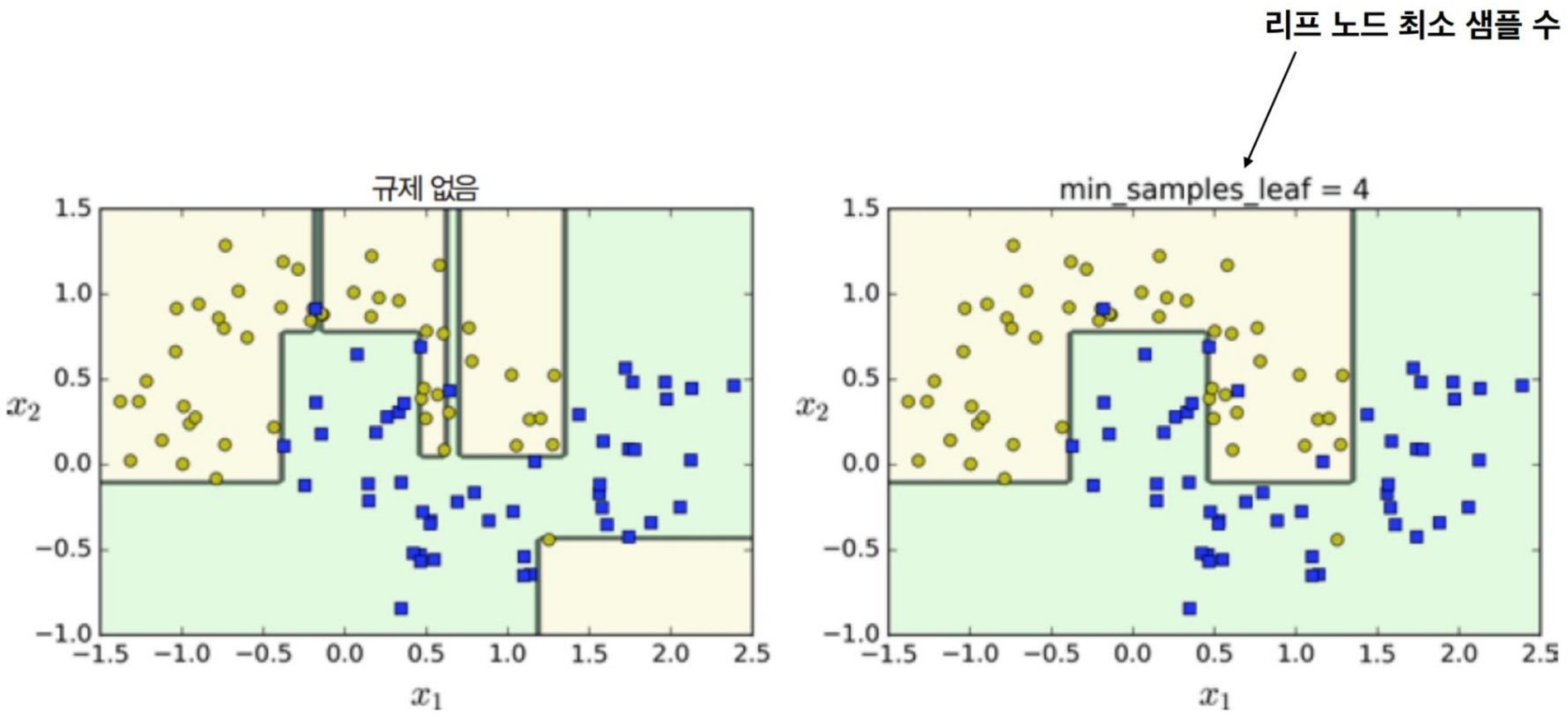
전체 노드에서 전체 특성을 정렬하는 복잡도

$$nm^2log(m)$$

규제 매개변수 (Regularization Hyperparameters)

- max_depth: 결정 트리 최대 깊이, 기본값 None
- min_samples_split: 분할되기 위해 노드가 가져야 하는 최소 샘플 수, 2
- min_samples_leaf: 리프 노드가 가지고 있어야 할 최소 샘플 수, 1
- min_weight_fraction_leaf: min_samples_leaf와 동일. 샘플 수 대신 전체 샘플 수에서의 비율, 0
- max_leaf_nodes: 리프 노드의 최대 수, None
- max_features: 각 노드에서 분할에 사용할 특성의 최대 수, None
- min_impurity_decrease: 분할로 얻어질 최소한의 불순도, 0

규제 사례

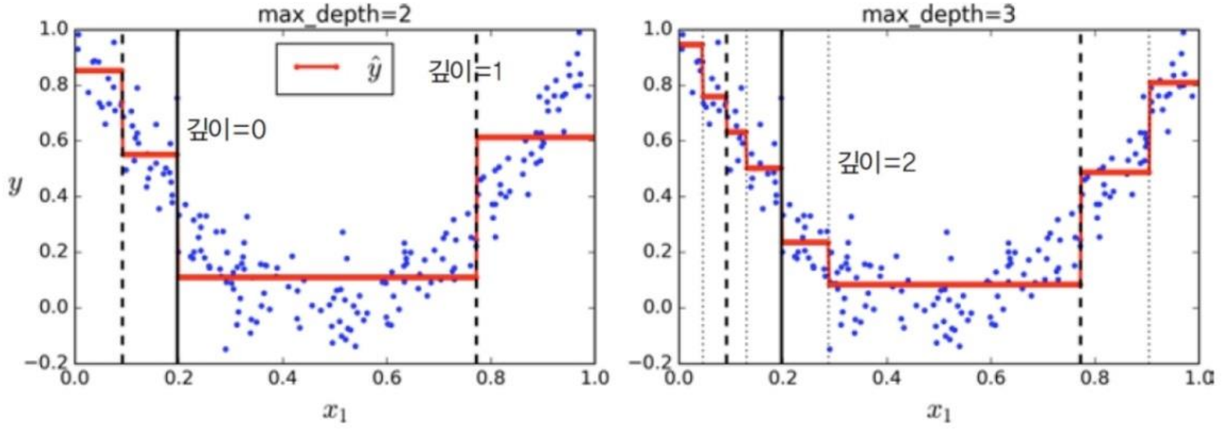

리프 노드 최소 샘플 수
규제 없음
min_samples_leaf = 4
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
x_1
x_2

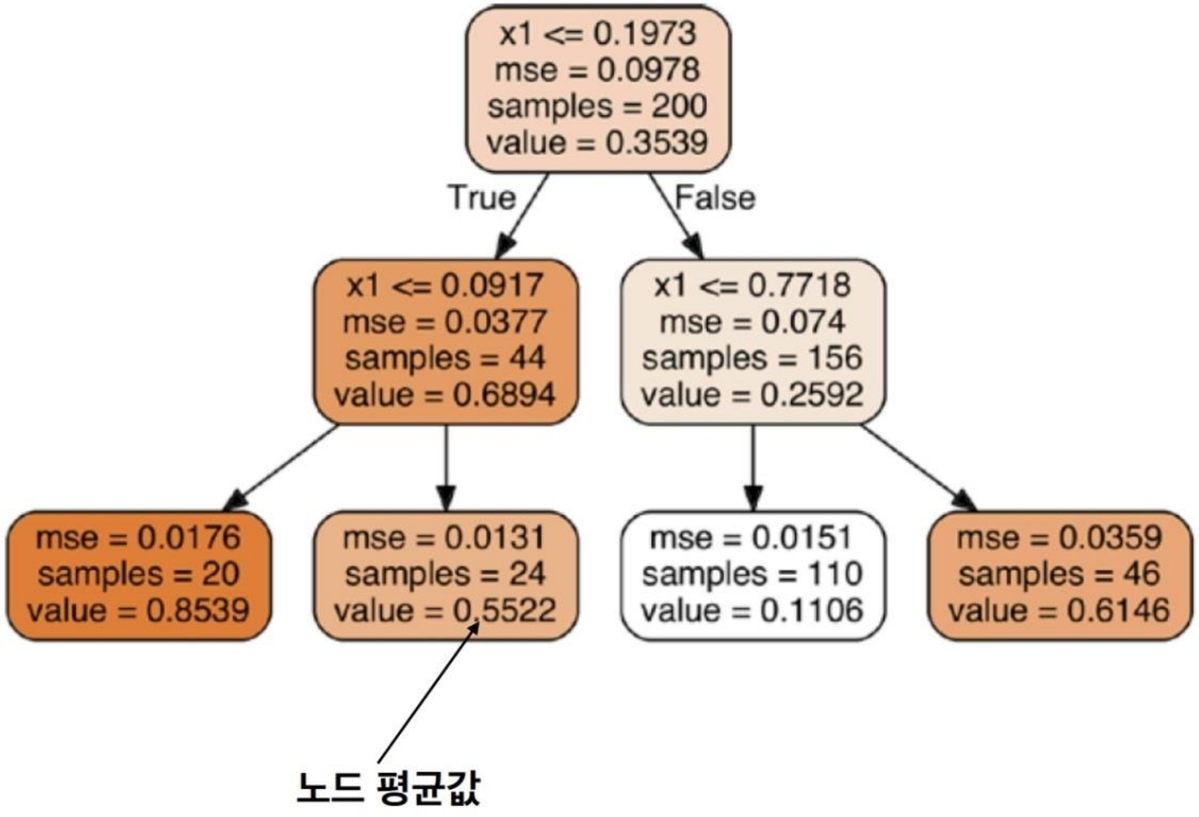
DecisionTreeRegressor

- DecisionTreeClassifier(criterion='mse'), 또는 'mae'

```
from sklearn.tree import DecisionTreeRegressor

tree_reg = DecisionTreeRegressor(max_depth=2)
tree_reg.fit(X, y)
```





회귀의 비용 함수

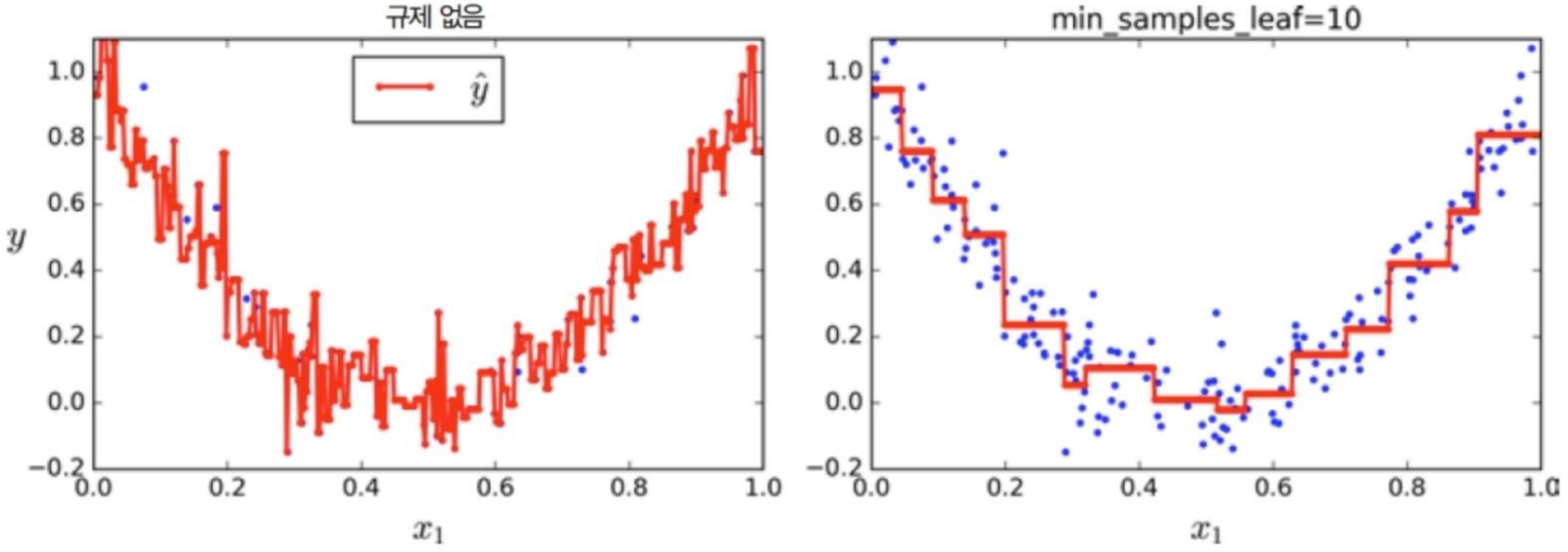
$$J(k, t_k) = \frac{m_{\text{left}}}{m} \text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m} \text{MSE}_{\text{right}}$$

여기서

$$\begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)} \right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

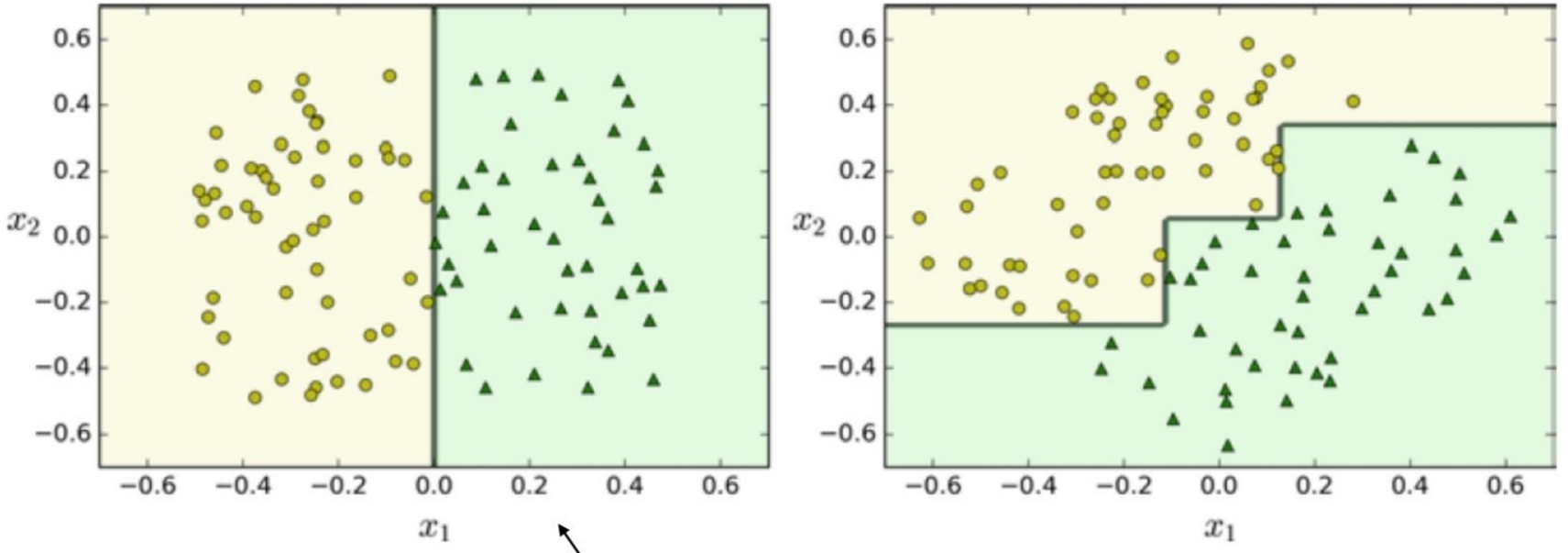
회귀 모델의 규제

리프 노드 최소 샘플 수

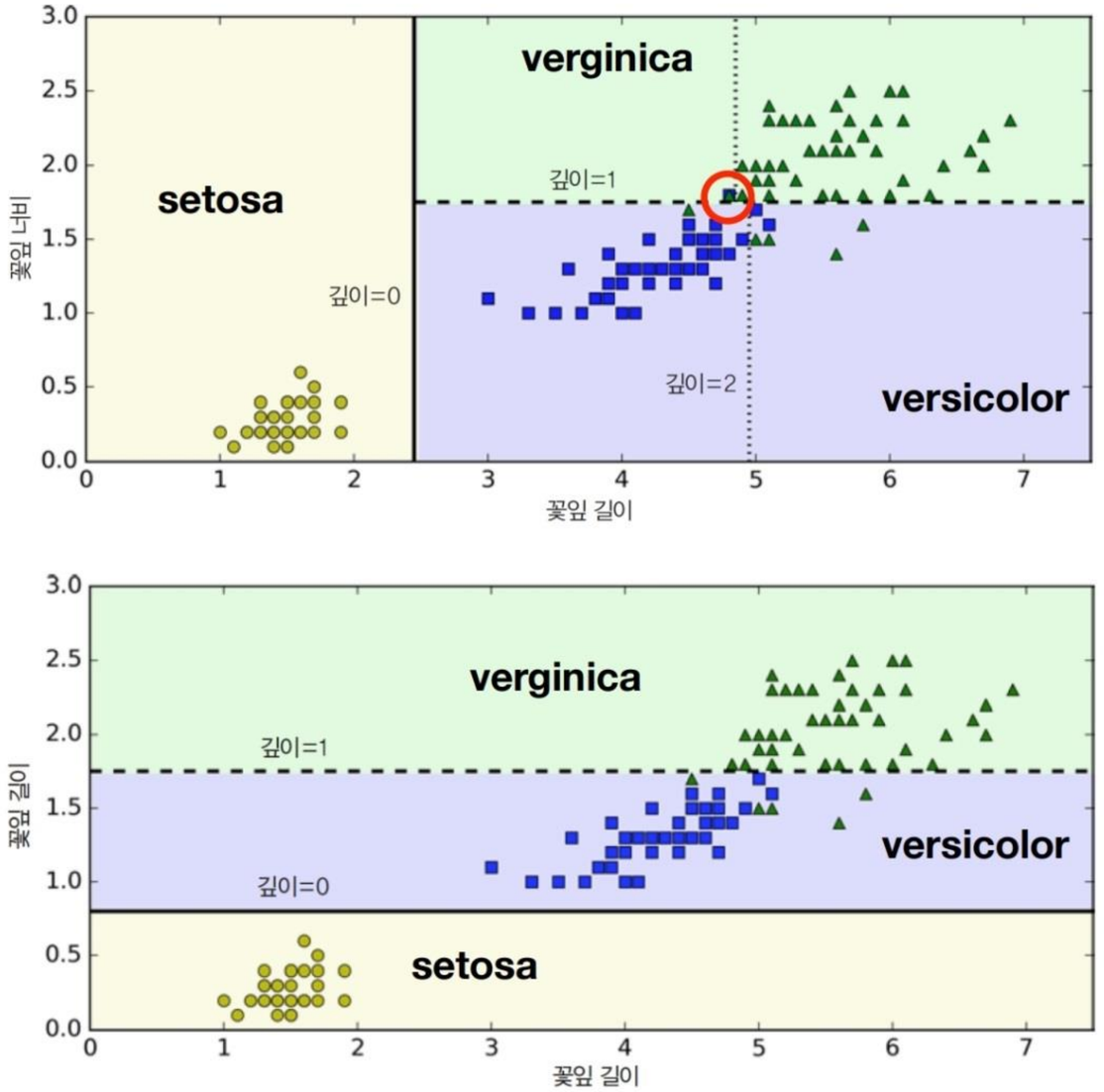


회전 불안정성

- PCA를 사용하여 직교하는 주성분으로 표현하는 것이 좋음



훈련 세트 분할에 민감



빨간 동그라미로 표시한 샘플이 없었으면 꽃잎 길이로 먼저 분할하여 아래 그림과 같은 결과가 나올 수도 있음 ➔ 훈련세트의 작은 변화에도 민감함

감사합니다