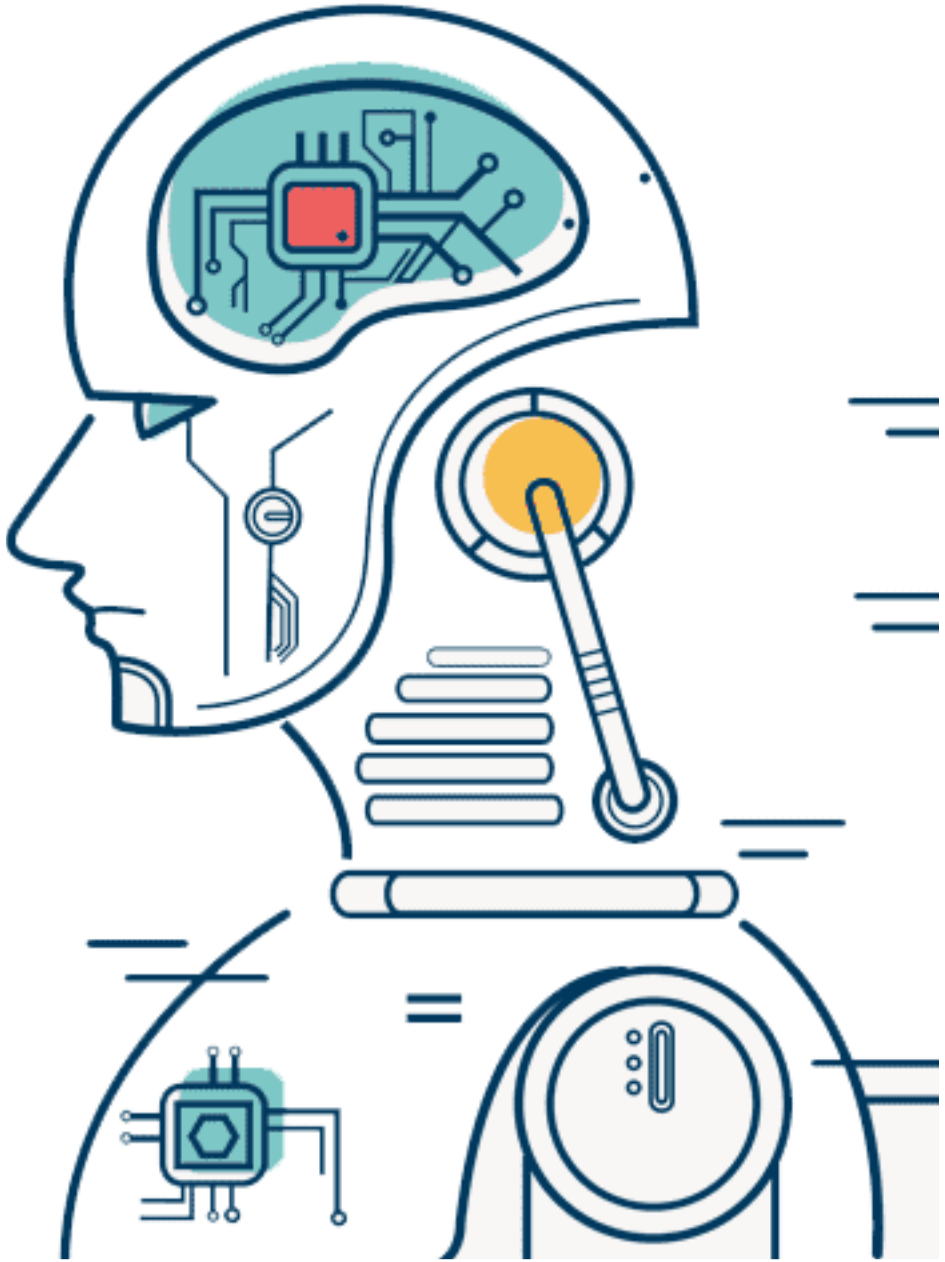
스마트인재개발원
Smart Human Resources Development

Machine Learning



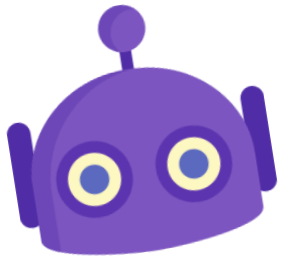
Chapter 3 지도 학습

(Decision Tree, Label Encoding, One-hot Encoding, Cross validation, 정규화)

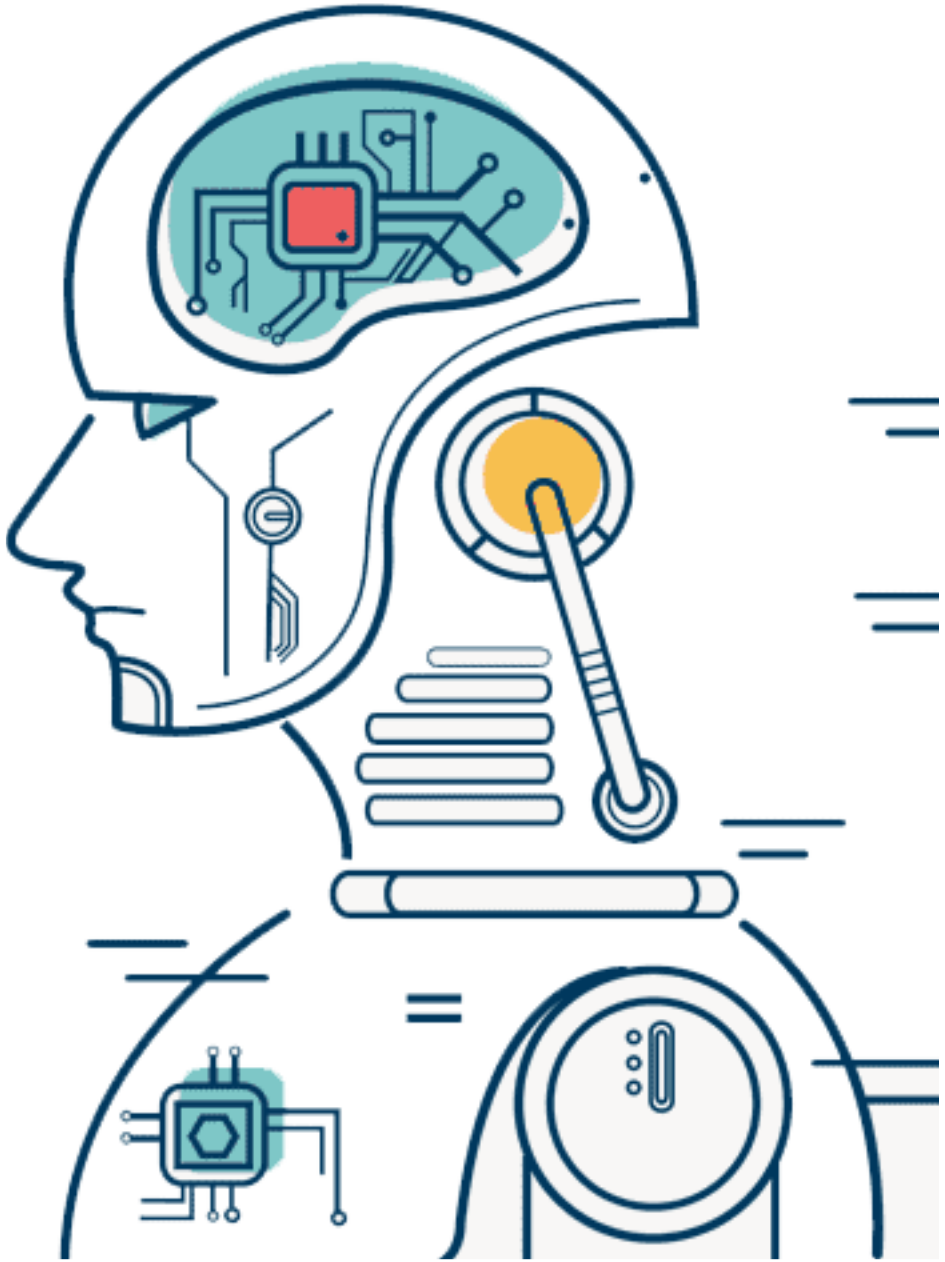
학습목표

- Decision Tree 알고리즘을 이해 할 수 있다.
- 결측치가 있는지 확인할 수 있다.
- Label 인코딩과 One-hot 인코딩을 이해 할 수 있다.
- 교차 검증 기법을 이해 할 수 있다.
- 정규화 (Normalization)을 이해할 수 있다.

스마트인재개발원
Smart Human Resources Development

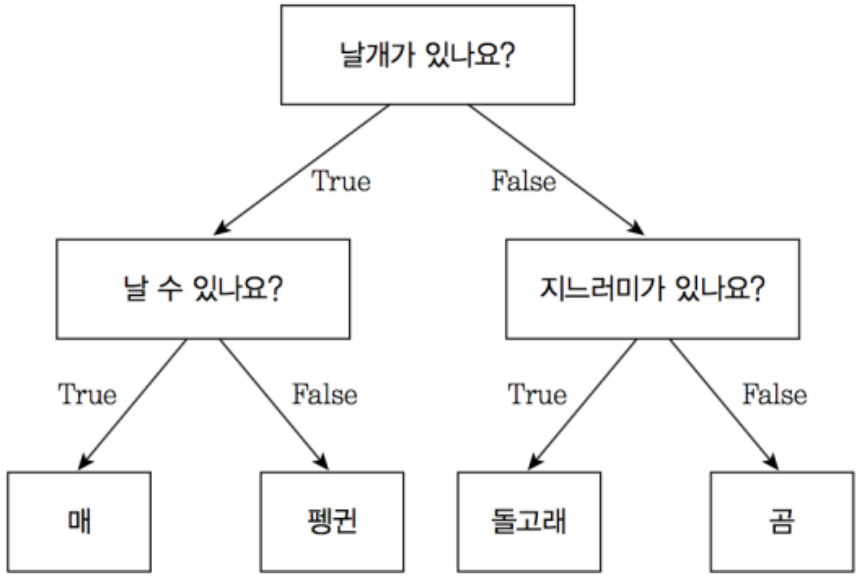


Decision Tree

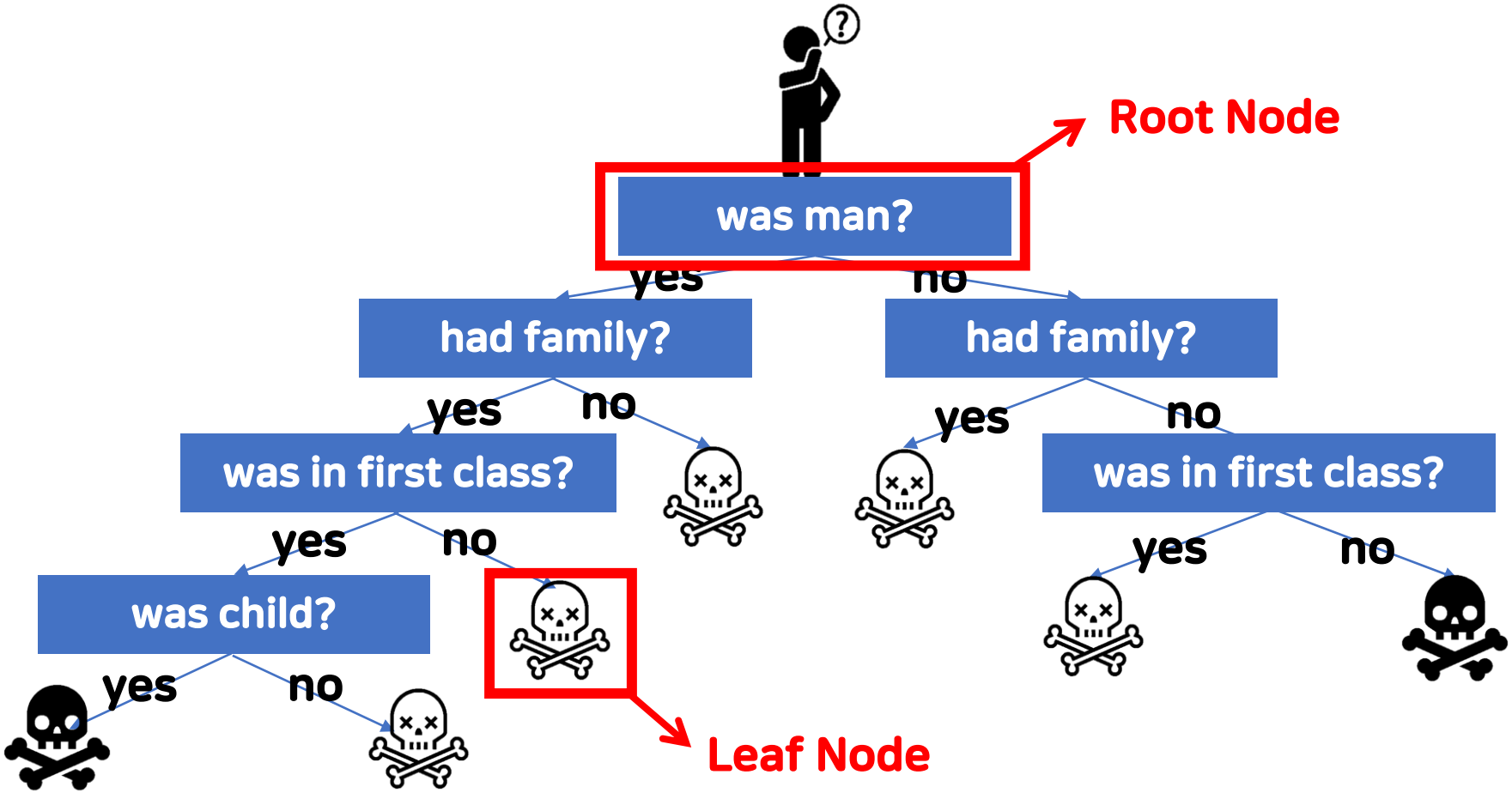


Decision Tree(결정트리)

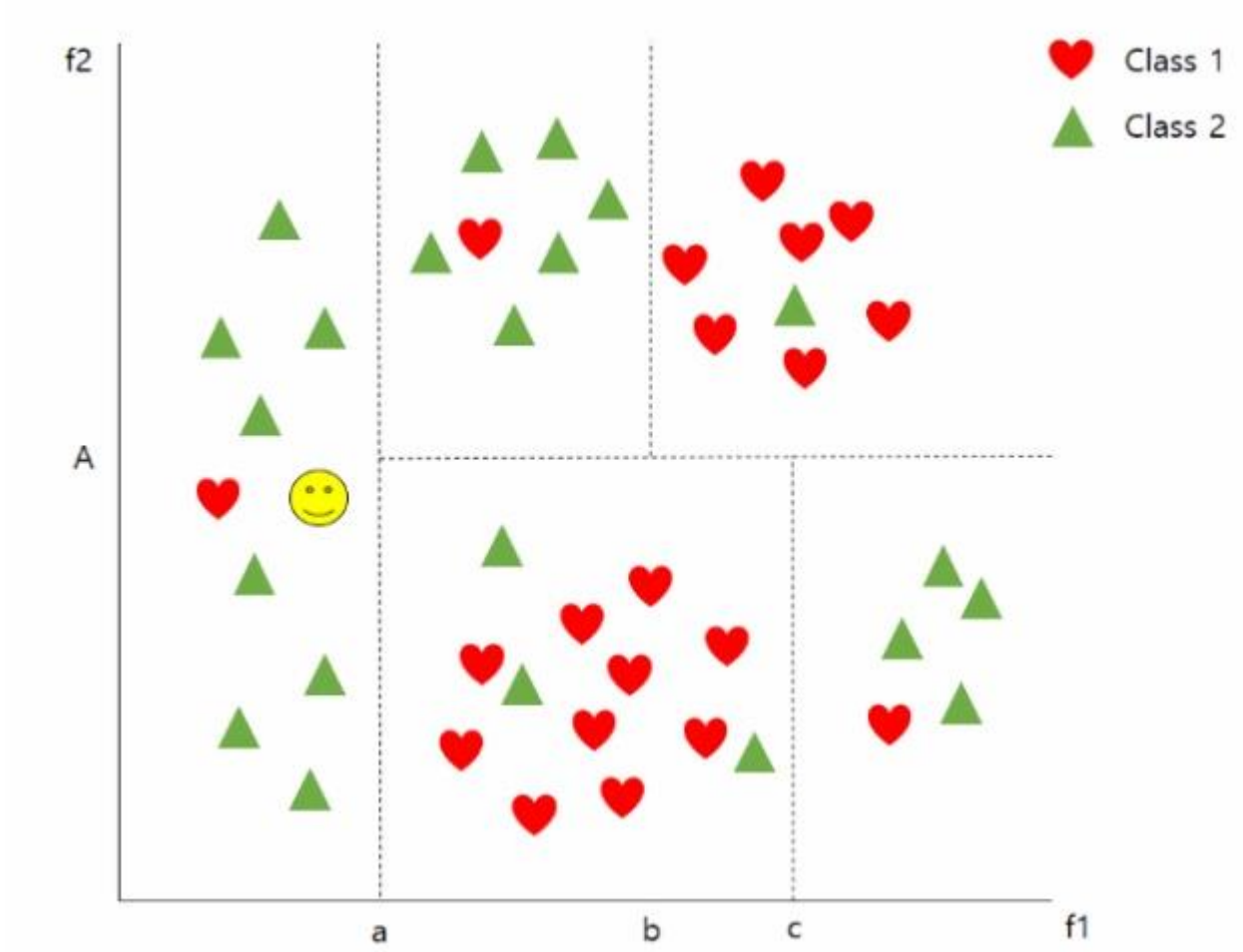
- Tree를 만들기 위해 예/아니오 질문을 반복하며 학습한다.
- 분류와 회귀에 모두 사용 가능

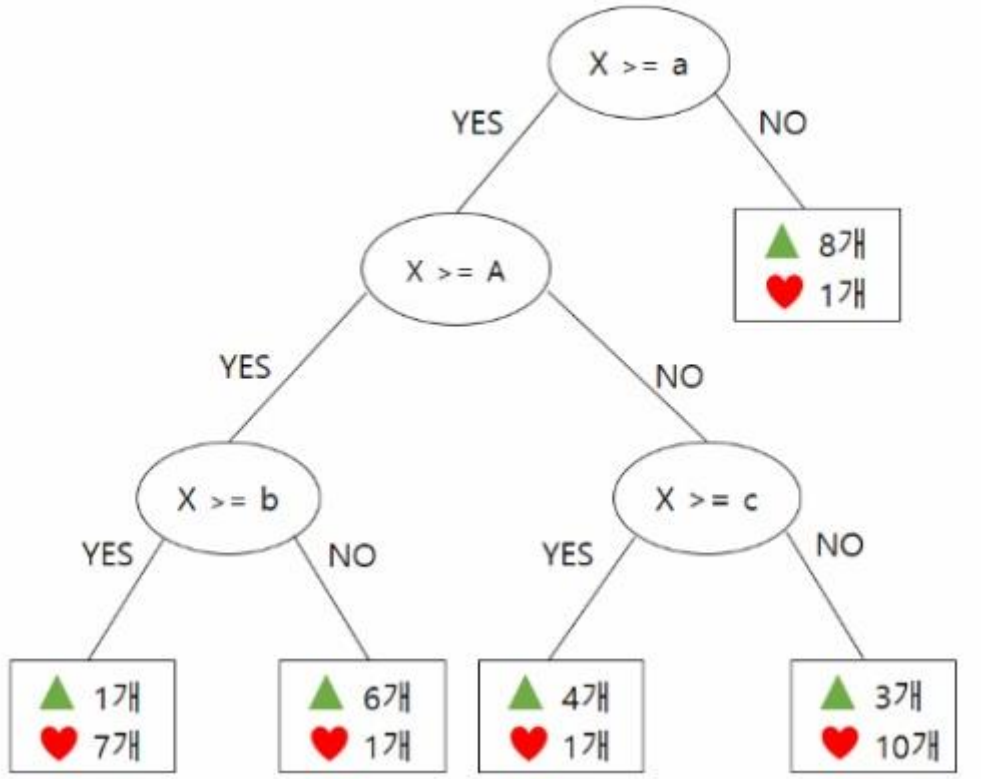


Decision Tree(결정트리)



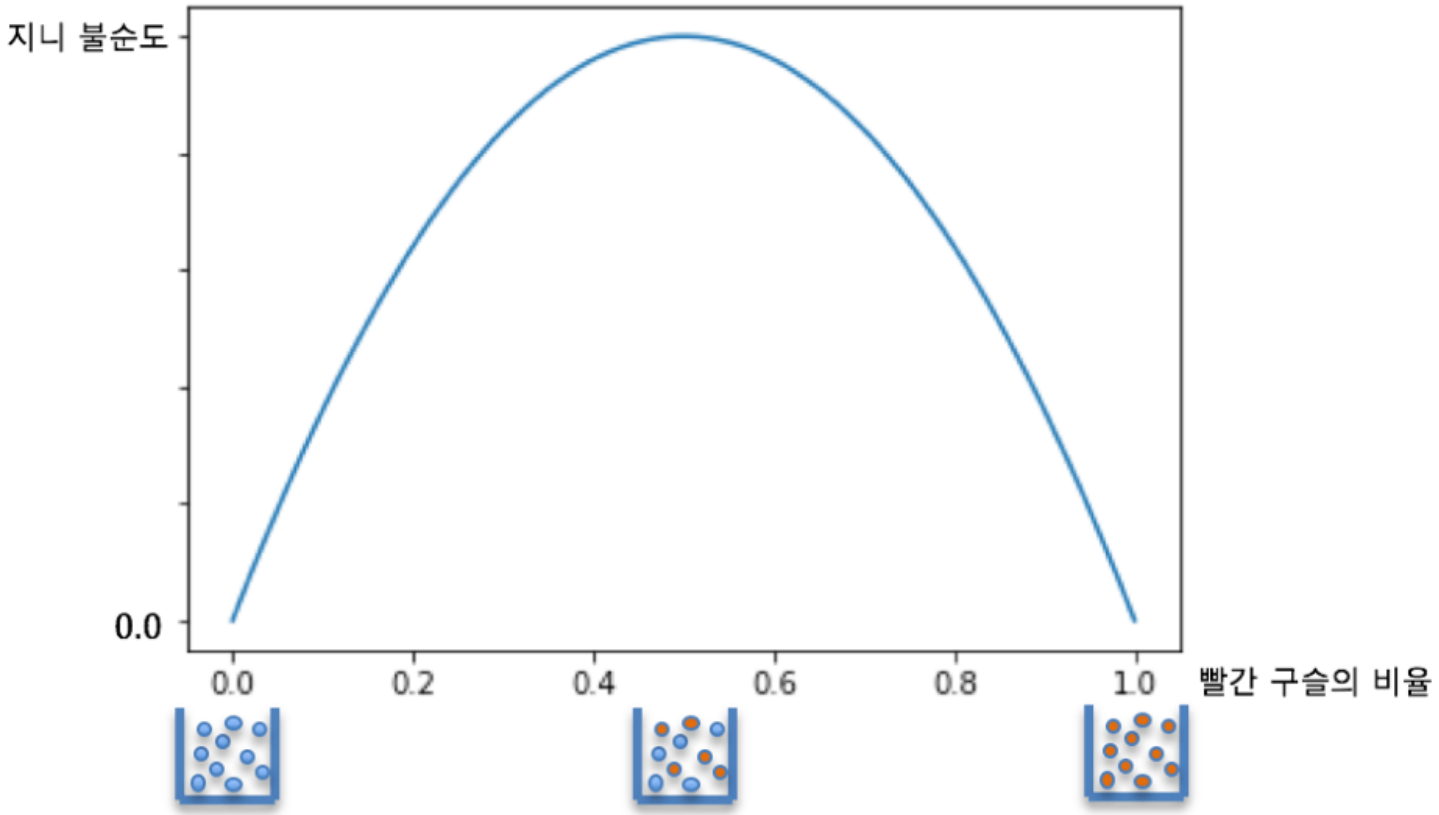
Decision Tree(결정트리)

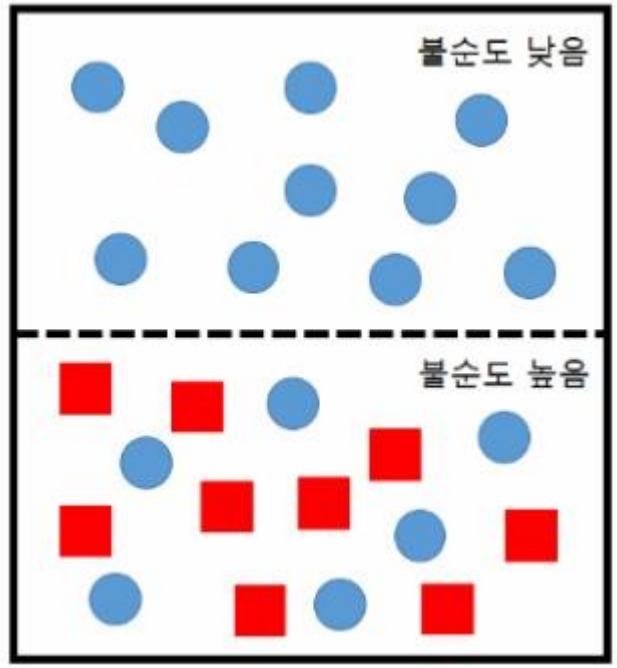




https://jeongmin-lee.tistory.com/85

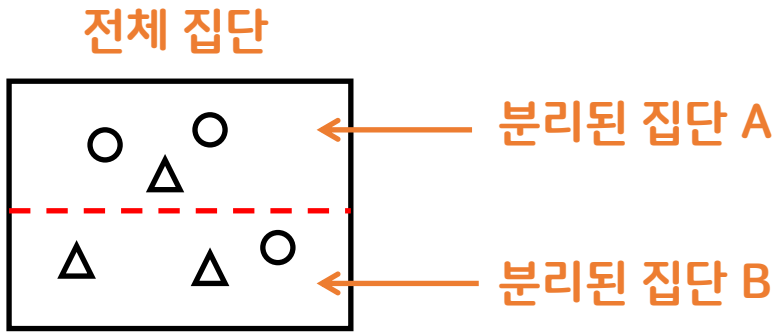
Gini Impurity(지니 불순도)





Gini Impurity(지니 불순도)

$$GI = \sum_{i=1}^{d}(R_i(1 - \sum_{k=1}^{m} p_{ik}^2))$$



- i : 분리된 집단의 수 (A, B)
- k : 전체집단에서 집단의 수 (○, △)

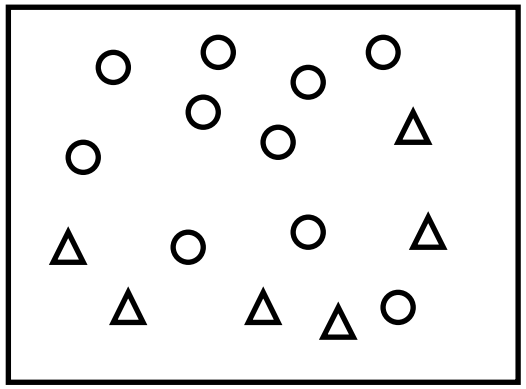
- p_k : 분리된 집단에서 k집단이 속한 비율

분리된 집단 A에서 ○ 집단이 속한 비율 : 2/3

- R_i : 전체 집단에서 분리된 집단의 비율

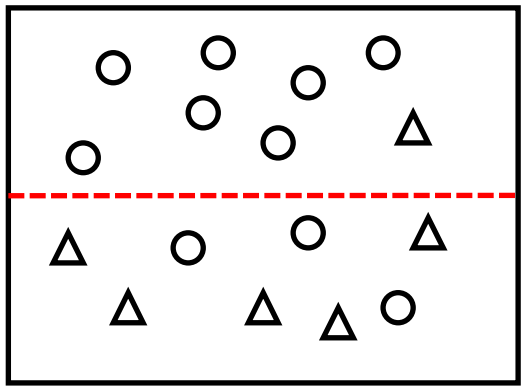
전체 집단에서 분리된 집단 A의 비율 : 3/6

Gini Impurity(지니 불순도)



○의 GI

$$GI = 1 - \left(\frac{10}{16}\right)^2 - \left(\frac{6}{16}\right)^2 \approx \mathbf{0.47}$$

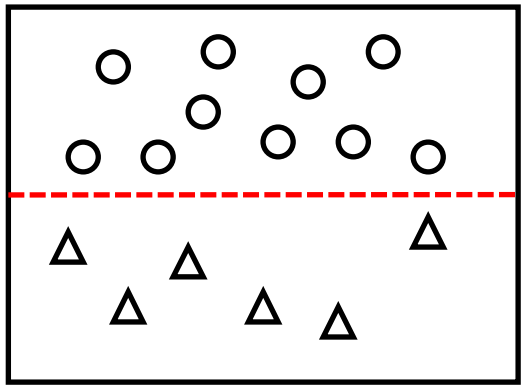


○의 GI

$$GI = 0.5 \times \left(1 - \left(\frac{7}{8}\right)^2 - \left(\frac{1}{8}\right)^2\right) +$$

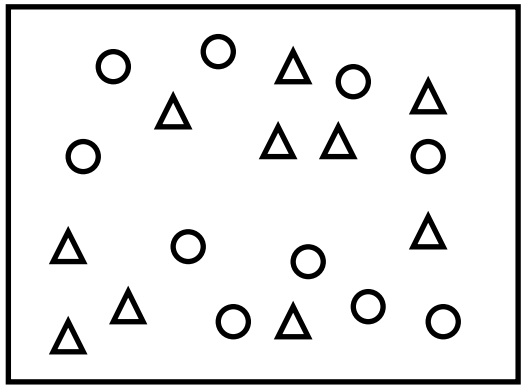
$$0.5 \times \left(1 - \left(\frac{5}{8}\right)^2 - \left(\frac{3}{8}\right)^2\right) \approx \mathbf{0.34}$$

Gini Impurity(지니 불순도)



잘 분류가 된 ○의 GI

$$GI = 0.5 \times (1 - \left(\frac{10}{10}\right)^2) + 0.5 \times (1 - \left(\frac{8}{8}\right)^2) = \mathbf{0}$$

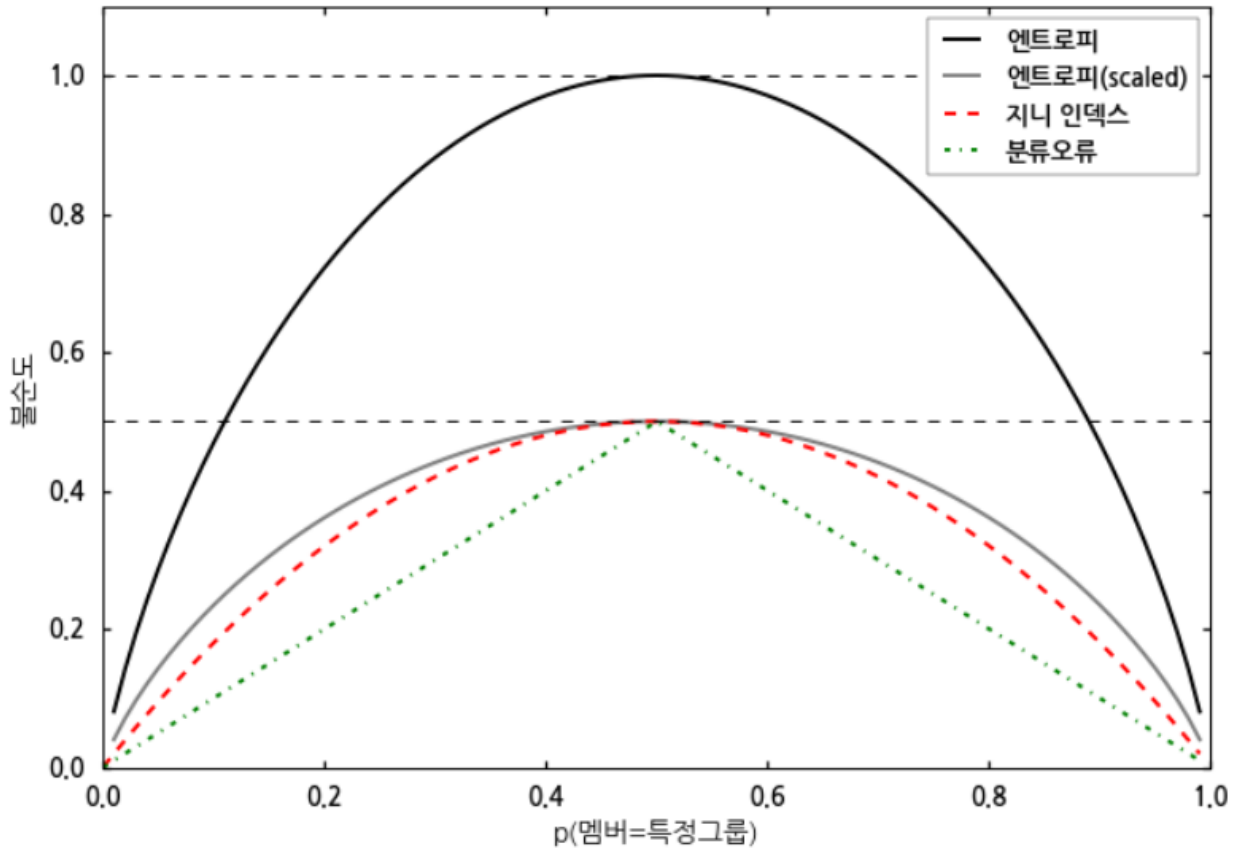


반반씩 섞인 데이터라면 ○의 GI

$$GI = 1 - \left(\frac{10}{20}\right)^2 - \left(\frac{10}{20}\right)^2 = \mathbf{0.5}$$

다른 불순도 계산방법 비교

Gini 불순도가 다른 방법에 비해 빠르게 불순도가 증가



Decision Tree(결정트리)

- 순수 노드 : 타깃 값이 한 개인 리프 노드
- 모든 노드가 순수 노드가 될 때 까지 학습 → 복잡, 과대적합
- 새로운 데이터 포인트가 들어오면 해당하는 노드를 찾아 분류라면 더 많은 클래스를 선택하고, 회귀라면 평균을 구함

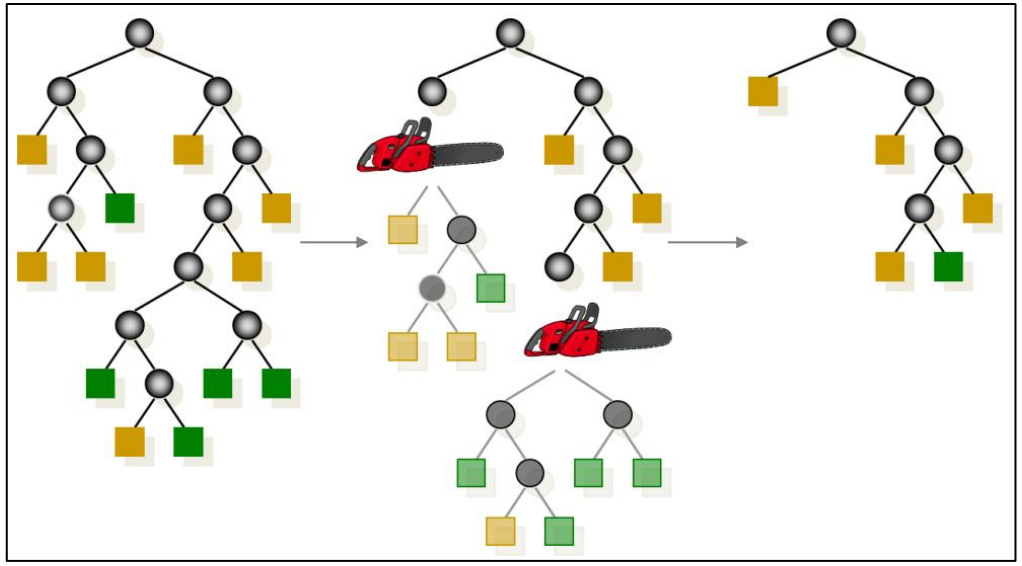
Decision Tree(결정트리) 과대적합 제어

- 노드 생성을 미리 중단하는 사전 가지치기(pre-pruning)와 트리를 만든 후에 크기가 작은 노드를 삭제하는 사후 가지치기(post-pruning)가 있다
  (sklearn은 사전 가지치기만 지원)

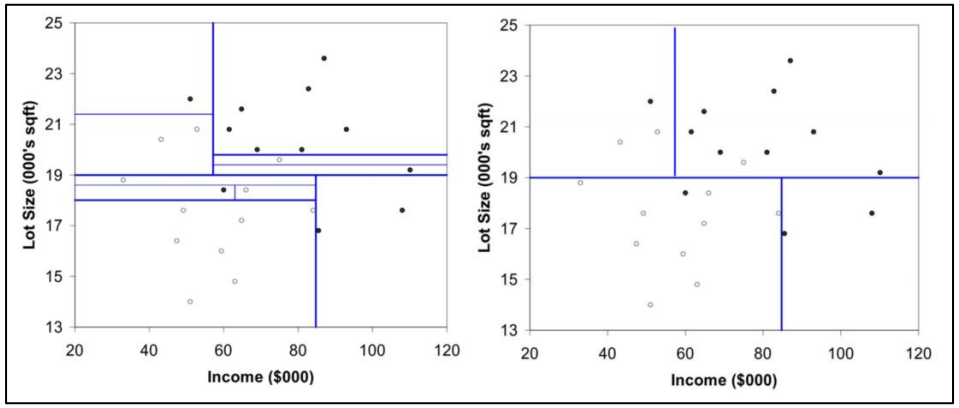
- 트리의 최대 깊이나 리프 노드의 최대 개수를 제어
  (max_depth, max_leaf_nodes)

- 노드가 분할 하기 위한 데이터 포인트의 최소 개수를 지정
  (min_sample_split)

Decision Tree(결정트리) 과대적합 제어



아래쪽 노드로 분할되지 않는다는 개념



주요 매개변수(Hyperparameter)

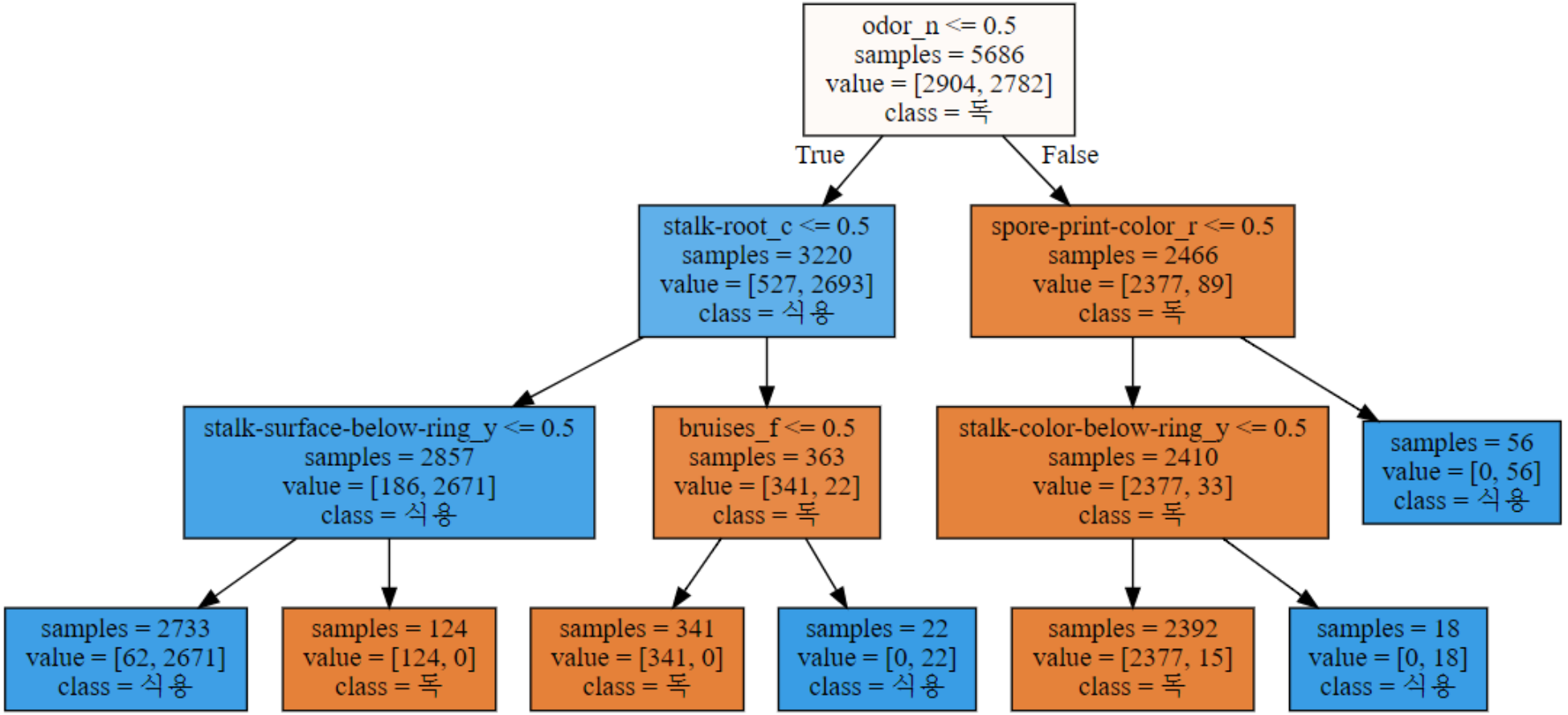
scikit-learn의 경우

DecisionTreeClassifier(max_depth, max_leaf_nodes, min_sample_split, min_sample_leaf, **criterion**)

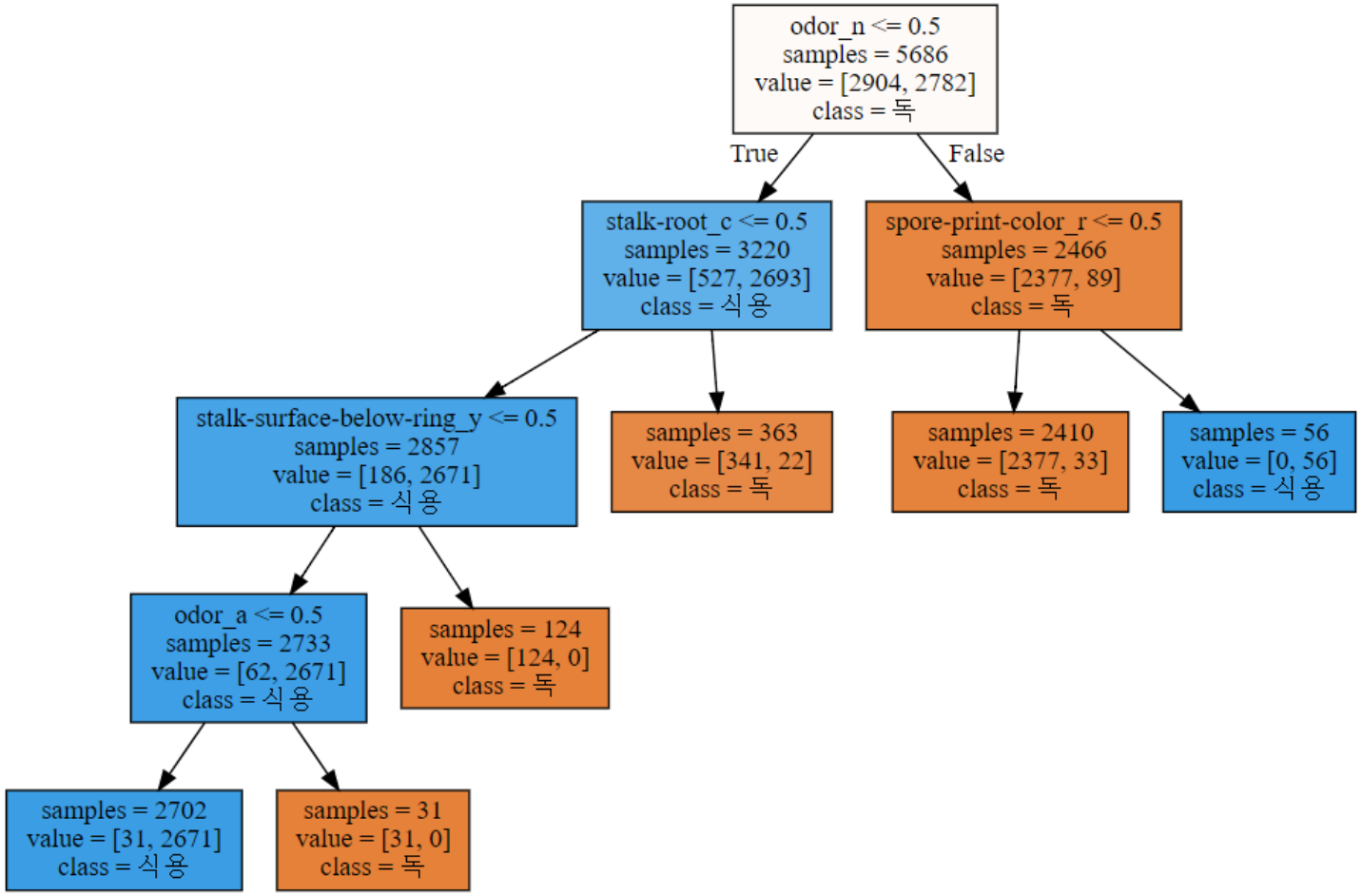
- 트리의 최대 깊이 : max_depth
  (값이 클수록 모델의 복잡도가 올라간다.)
- 리프 노드의 최대 개수 : max_leaf_nodes
- 리프 노드로 분리 가능한 최소 샘플의 개수 : min_sample_split
- 리프 노드를 구성하는 최소 샘플의 개수 : min_samples_leaf

- criterion : 불순도 측정 방법 (gini, entropy)

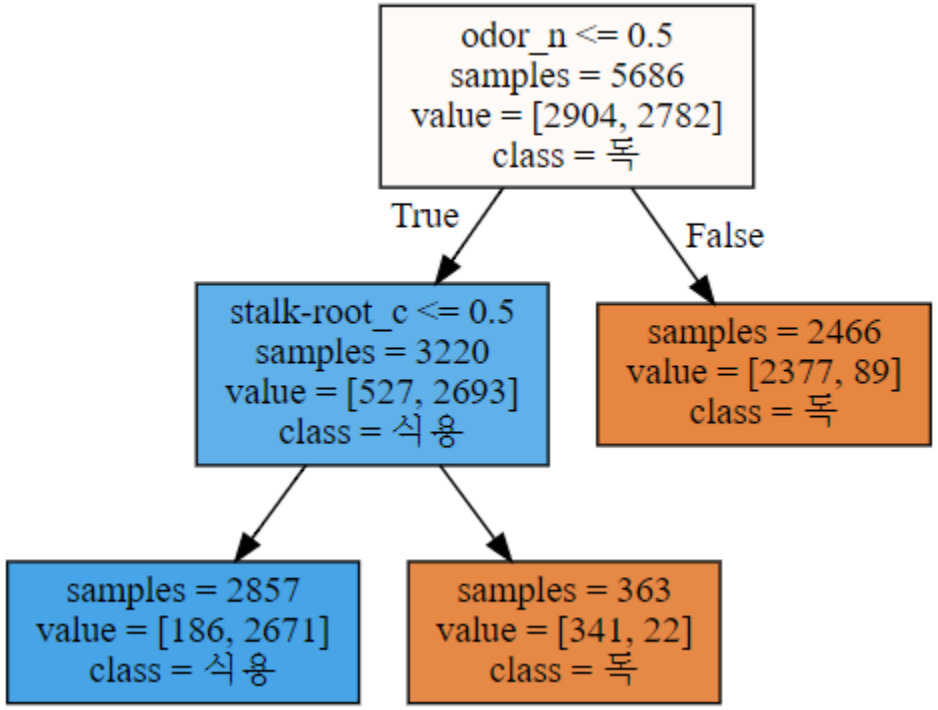
사전가지치기 (max_depth=3)



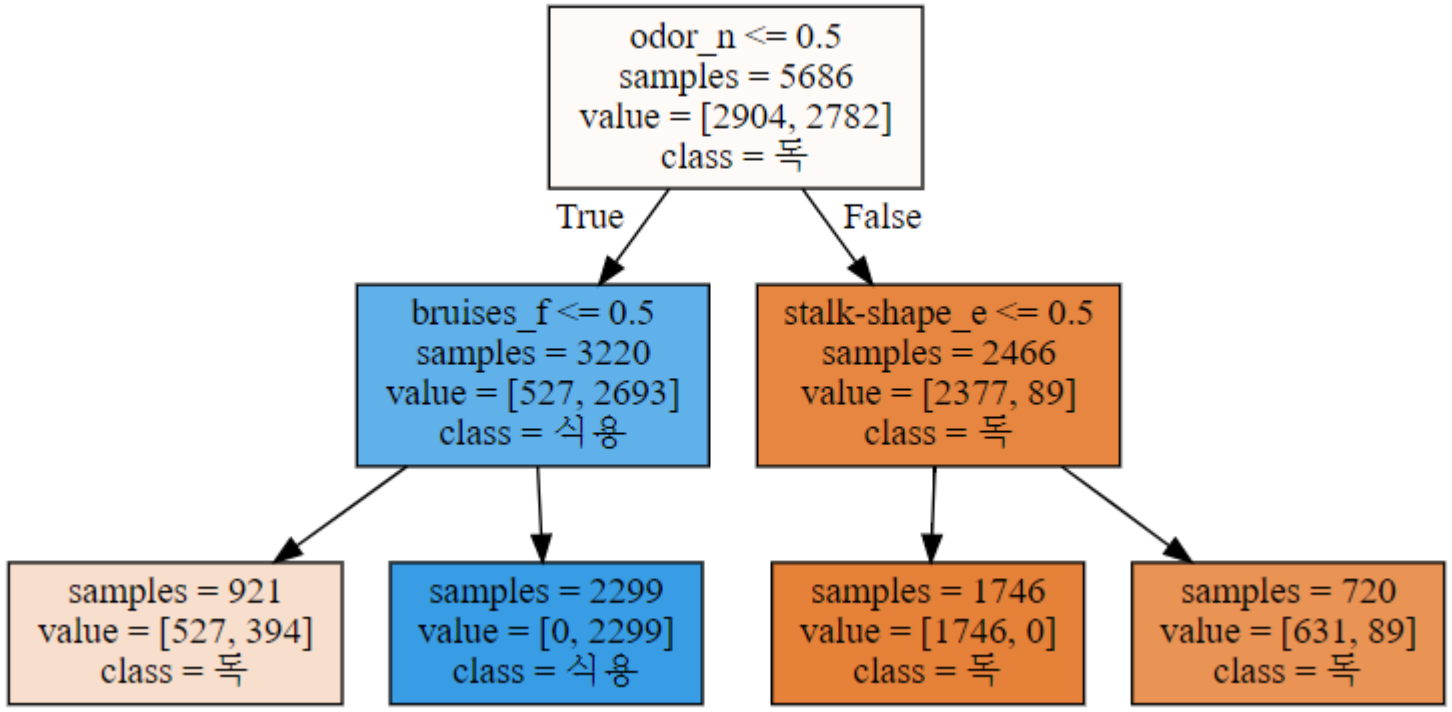
사전가지치기 (max_leaf_nodes=6)



사전가지치기 (min_sample_split=3000)



사전가지치기 (min_samples_leaf=500)



사후 가지치기 (post-pruning)

- 초기에 트리를 최대 크기로 만듬 → 마지막 노드의 불순도가 최소가 되도록 분할
  → 복잡도 최대

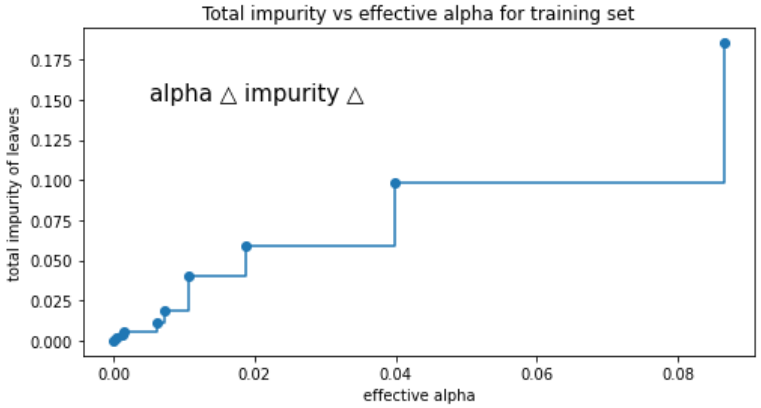
- 교차검증 점수가 최소가 되는 분할이 되도록 트리를 줄임

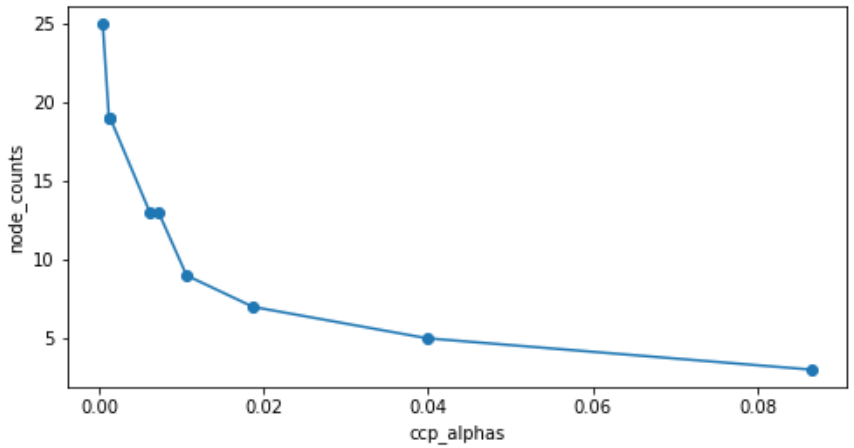
- 사전 가지치기가 위 층부터 노드를 분리하는 과정이라면 사후 가지치기는 아래 층 노드부터 합해가는 과정

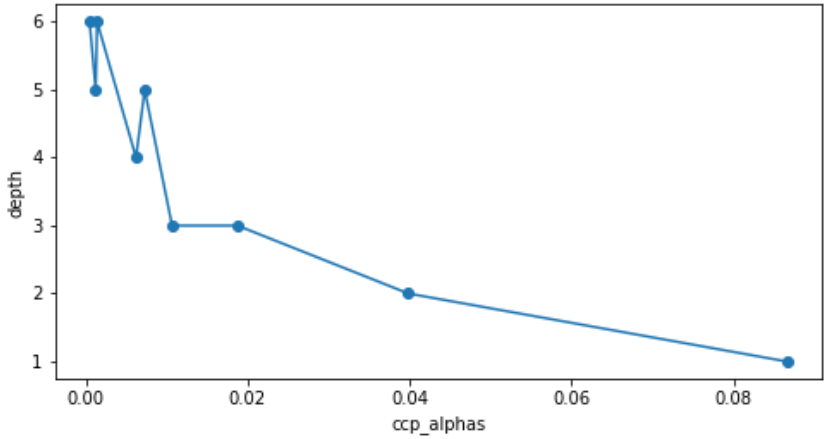
https://jeongmin-lee.tistory.com/85

사후 가지치기 (post-pruning)

- 링크강도 ccp_alpha 값을 제어
- ccp_alpha 값이 증가하면 불순도가 증가
- ccp_alpha 값이 증가하면 노드수 / depth 감소







https://jeongmin-lee.tistory.com/85

장단점

- 만들어진 모델을 쉽게 시각화할 수 있어 이해하기 쉽다.
  (white box model)

- 각 특성이 개별 처리되기 때문에 데이터 스케일에 영향을 받지 않아 특성의 정규화나 표준화가 필요 없다.

- 트리 구성시 각 특성의 중요도를 계산하기때문에 특성 선택 (Feature selection)에 활용될 수 있다.

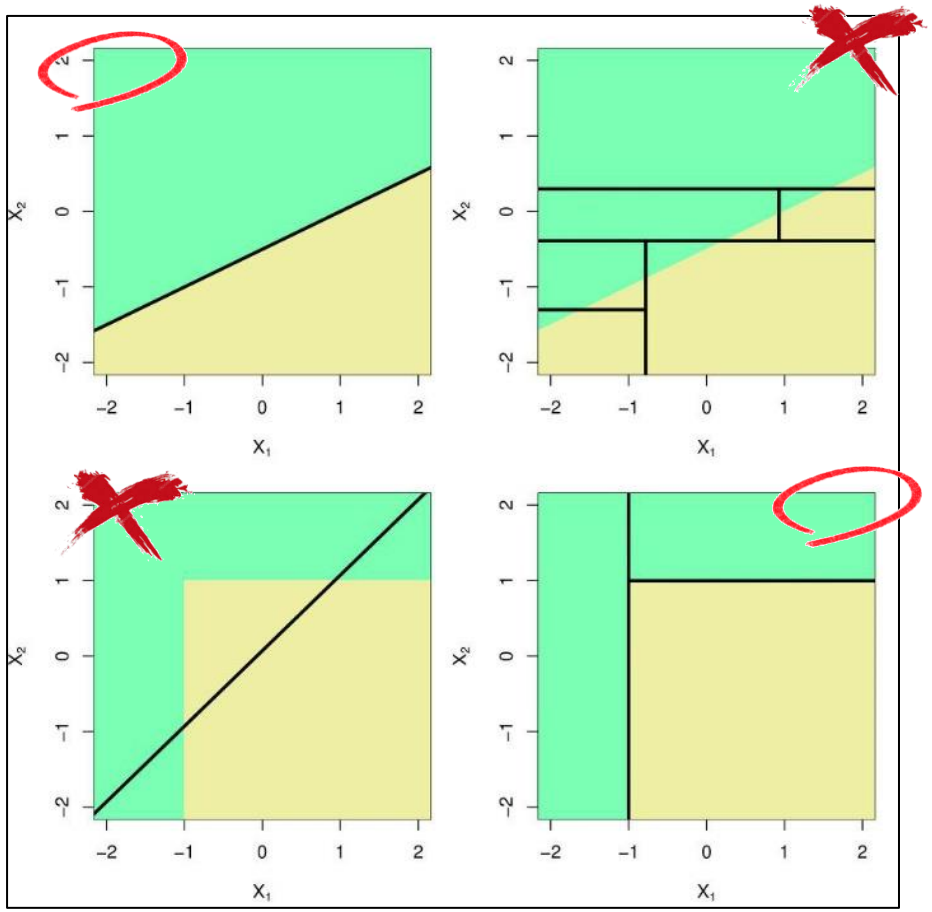
장단점

- 훈련데이터 범위 밖의 포인트는 예측 할 수 없다.
  (ex : 시계열 데이터)

- 가지치기를 사용함에도 불구하고 과대적합되는 경향이 있어 일반화 성능이 좋지 않다.

장단점

- 의사결정나무는 선형모형에는 적합하지 않음

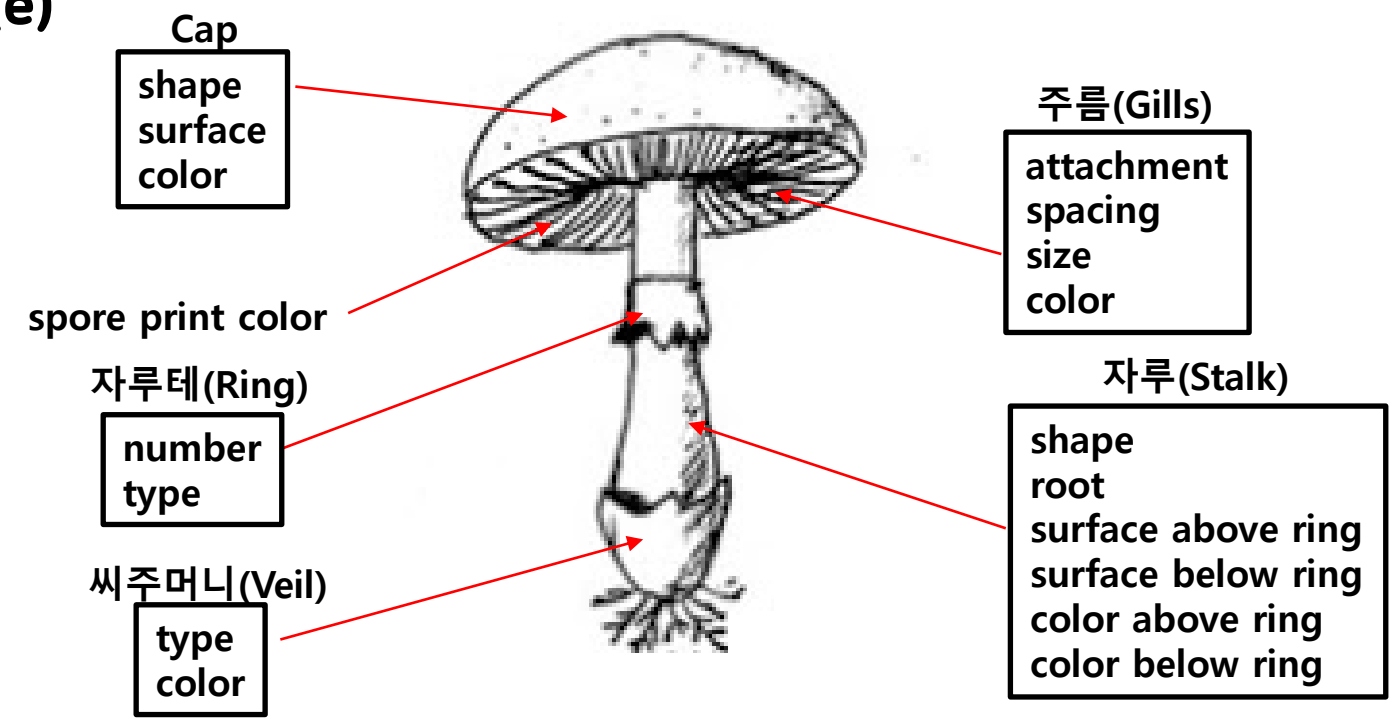


출처 : http://blog.hyeongeun.com/21

Mushroom 데이터 활용
Decision Tree 분류 실습

Mushroom 데이터셋

- 8124개의 버섯 종류 데이터
- 22개의 특징 (18개의 버섯 특징, 4개의 다른 특징 (Habitat (서식지), Population(분포 형태), Bruises(타박상), Odor(냄새)))
- 라벨 : 독성 (p), 식용(e)



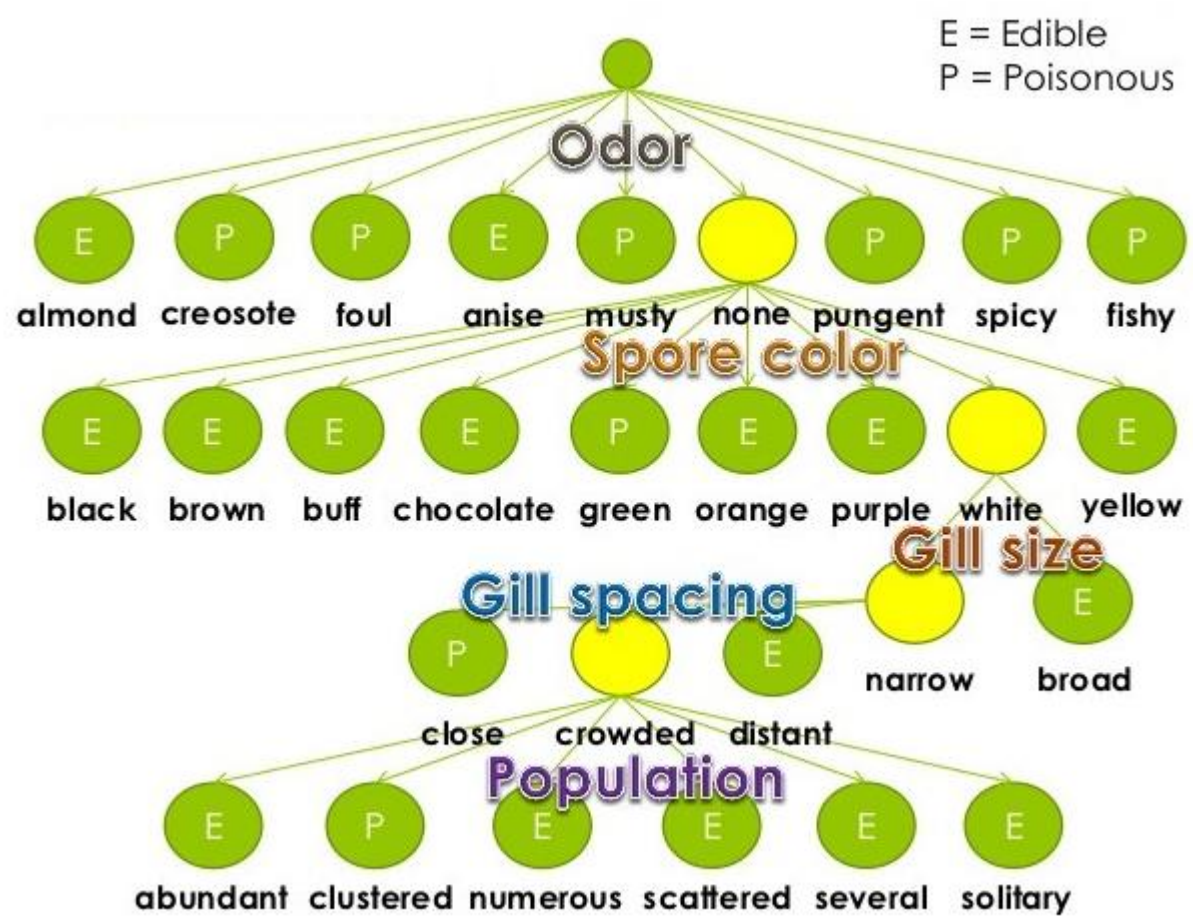
Decision Tree 실습

Mushroom 데이터셋

poisonous : 독버섯(poisonous), 식용버섯(edible)
cap-shape : 갓 모양(b,c,x,f,k,s) : 원뿔/평면/볼록 등
cap-surface : 갓 표면(f,g,y,s) : 섬유질/비늘모양/부드러움 등
cap-color : 갓 색(n,b,c,g,r,p,u,e,w,y) : 계피/회색/노란색 등
bruises : 타박상(t,f) : 예/아니오
odor : 냄새(a,l,c,y,f,m,n,p,s) : 아몬드,생선,매운 등

gill-attachment(자실층 위치), gill-spacing(자실층 간격), gill-size(자실층 크기), gill-color(자실층 색), stalk-shape(자루 모양), stalk-root(자루 뿌리), stalk-surface-above-ring(자루 표면 위 자루테), stalk-surface-below-ring(자루 표면 아래 자루테), stalk-color-above-ring(자루 색 위 자루테), stalk-color-below-ring(자루 색 아래 자루테), veil-type(베일 유형), veil-color(베일 색), ring-number(링 번호), ring-type(링 타입), spore-pring-color(포자 색), population(인구), habitat(서식지)

Mushroom 데이터셋



결측치 확인 – info() 함수

전체 데이터 개수와 일치하지 않는 특성(컬럼)이 있는지 확인

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8124 entries, 0 to 8123
Data columns (total 23 columns):
poisonous                 8124 non-null object
cap-shape                 8124 non-null object
cap-surface               8124 non-null object
cap-color                 8124 non-null object
bruises                   8124 non-null object
odor                      8124 non-null object
gill-attachment           8124 non-null object
gill-spacing              8124 non-null object
gill-size                 8124 non-null object
gill-color                8124 non-null object
stalk-shape               8124 non-null object
stalk-root                8124 non-null object
stalk-surface-above-ring  8124 non-null object
stalk-surface-below-ring  8124 non-null object
stalk-color-above-ring    8124 non-null object
stalk-color-below-ring    8124 non-null object
veil-type                 8124 non-null object
veil-color                8124 non-null object
ring-number               8124 non-null object
ring-type                 8124 non-null object
spore-print-color         8124 non-null object
population                8124 non-null object
habitat                   8124 non-null object
dtypes: object(23)
memory usage: 1.4+ MB
```

데이터 표현

- 숫자형(연속형) 특성 : 숫자로 이루어진 순서가 있는 데이터
- 범주형 특성 : 문자 형태로 된 값을 구분하기 위한 데이터
- Encoding : 범주형 데이터를 숫자형 데이터로 변환 (Label Encoding, One-hot Encoding, Word Embedding)
- Binning : 숫자형 데이터를 범주형 데이터로 변환

Decision Tree 실습

스마트인재개발원
Smart Human Resources Development

범주형 특성

연속형 특성

	A	B	C	D
1	Gender	Height	Weight	Label
2	Male	174	96	Obesity
3	Male	189	87	Normal
4	Female	185	110	Obesity
5	Female	195	104	Overweight
6	Male	149	61	Overweight
7	Male	189	104	Overweight
8	Male	147	92	Extreme Obesity
9	Male	154	111	Extreme Obesity
10	Male	174	90	Overweight
11	Female	169	103	Obesity
12	Male	195	81	Normal
13	Female	159	80	Obesity
14	Female	192	101	Overweight
15	Male	155	51	Normal
16	Male	191	79	Normal
17	Female	153	107	Extreme Obesity
18	Female	157	110	Extreme Obesity
19	Male	140	129	Extreme Obesity
20	Male	144	145	Extreme Obesity

Decision Tree 실습

스마트인재개발원
Smart Human Resources Development

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	poisonous	cap-shape	cap-surfac	cap-color	bruises	odor	gill-attach	gill-spacin	gill-size	gill-color	stalk-shap	stalk-root	stalk-surfa	stalk-surfa	stalk-color
2	p	x	s	n	t	p	f	c	n	k	e	e	s	s	w
3	e	x	s	y	t	a	f	c	b	k	e	c	s	s	w
4	e	b	s	w	t	l	f	c	b	n	e	c	s	s	w
5	p	x	y	w	t	p	f	c	n	n	e	e	s	s	w
6	e	x	s	g	f	n	f	w	b	k	t	e	s	s	w
7	e	x	y	y	t	a	f	c	b	n	e	c	s	s	w
8	e	b	s	w	t	a	f	c	b	g	e	c	s	s	w
9	e	b	y	w	t	l	f	c	b	n	e	c	s	s	w
10	p	x	y	w	t	p	f	c	n	p	e	e	s	s	w
11	e	b	s	y	t	a	f	c	b	g	e	c	s	s	w
12	e	x	y	y	t	l	f	c	b	g	e	c	s	s	w
13	e	x	y	y	t	a	f	c	b	n	e	c	s	s	w
14	e	b	s	y	t	a	f	c	b	w	e	c	s	s	w
15	p	x	y	w	t	p	f	c	n	k	e	e	s	s	w

범주형(이산형) 특성이기 때문에 인코딩 필요

categorical feature

Label 인코딩 or One-hot 인코딩 방식을 이용해 수치화한다.

특성의 클래스 종류 확인 – unique() 함수

해당 특성(클래스)이 범주형 데이터인 경우 값의 종류를 확인

X['cap-shape'].unique()

```
array(['x', 'b', 's', 'f', 'k', 'c'], dtype=object)
```

특성의 클래스 수 확인 – value_counts() 함수

해당 특성(클래스)이 범주형 데이터인 경우 값의 종류와 개수를 확인

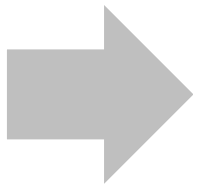
```
X['cap-shape'].value_counts()
```

```
x     3656
f     3152
k      828
b      452
s       32
c        4
Name: cap-shape, dtype: int64
```

Label Encoding : 수치 값으로 mapping하는 작업

```
X1["cap-shape"] = X1["cap-shape"]
        .map({"x":0, "f":1, "k":2, "b":3, "s":4, "c":5})
```

cap-shape
x
b
x
k
f
s
b
s
c

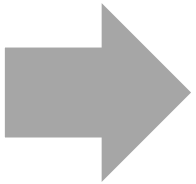


cap-shape
0
3
0
2
1
4
3
4
5

One-hot Encoding : 0 or 1의 값을 가진 여러 개의 새로운 특성으로 변경하는 작업

```
X_one_hot = pd.get_dummies(X2)
```

cap-shape
x
b
x
k
f
s
b
s
c



cap-shape_b	cap-shape_c	cap-shape_f	cap-shape_k	cap-shape_s	cap-shape_x
0	0	0	0	0	1
1	0	0	0	0	0
0	0	0	0	0	1
0	0	0	1	0	0
0	0	1	0	0	0
0	0	0	0	1	0
1	0	0	0	0	0
0	0	0	0	1	0
0	1	0	0	0	0

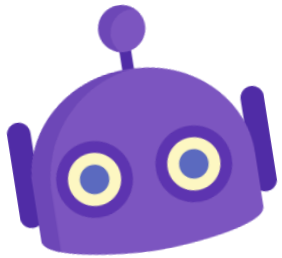
Feature Selection : 라벨과 연관관계가 높은 특성을 선택하는 작업

```
fi = tree_model.feature_importances_
df = pd.DataFrame(fi, index=X_one_hot.columns)
df.sort_values(by=0, ascending=False)
```

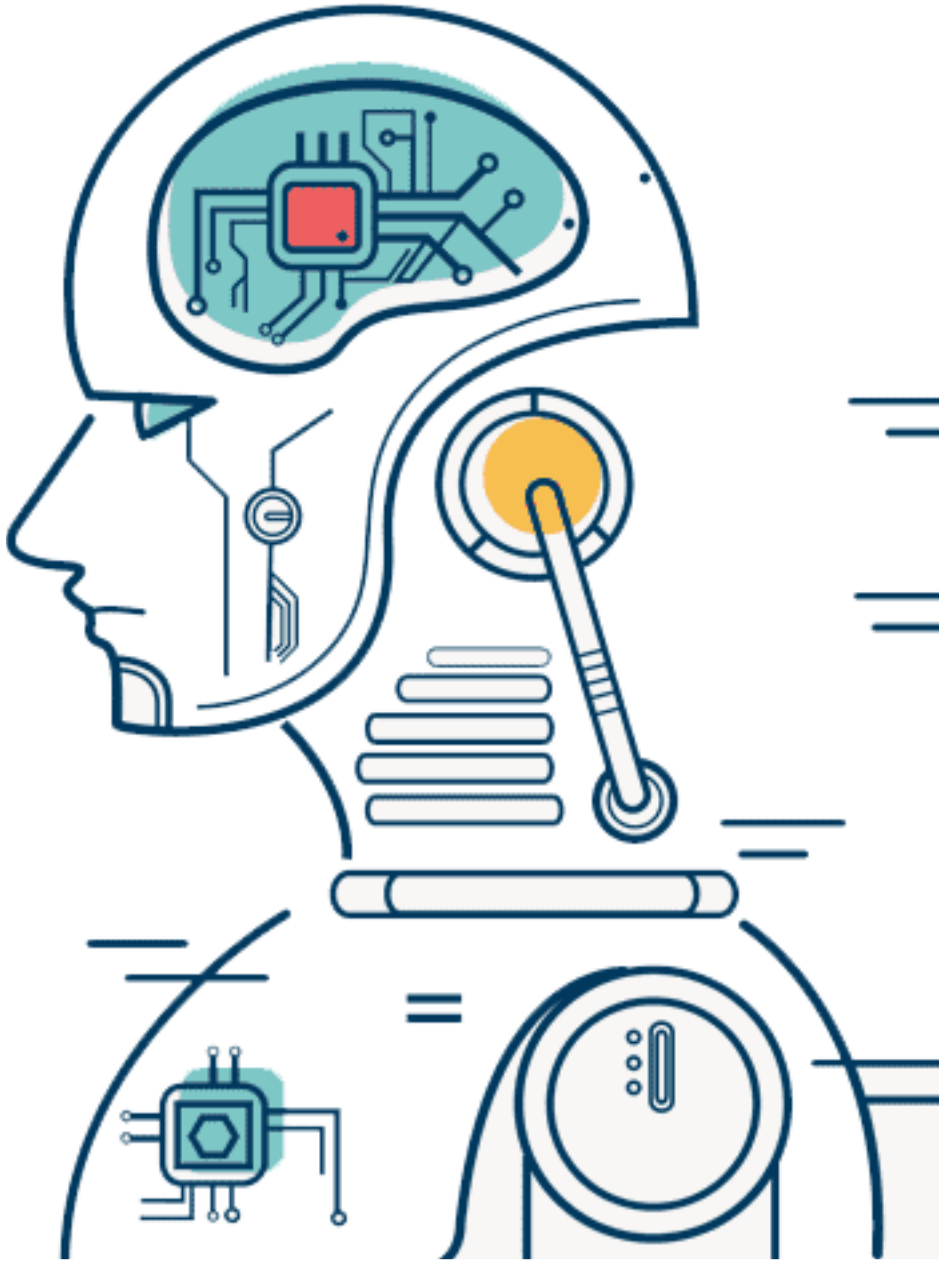
odor_n	0.623863
stalk-root_c	0.170163
stalk-root_r	0.082323
spore-print-color_r	0.036219
odor_a	0.023723
...	...

사전 가지치기 (pre-pruning)을 이용해서
모델의 성능을 향상시켜 보세요.

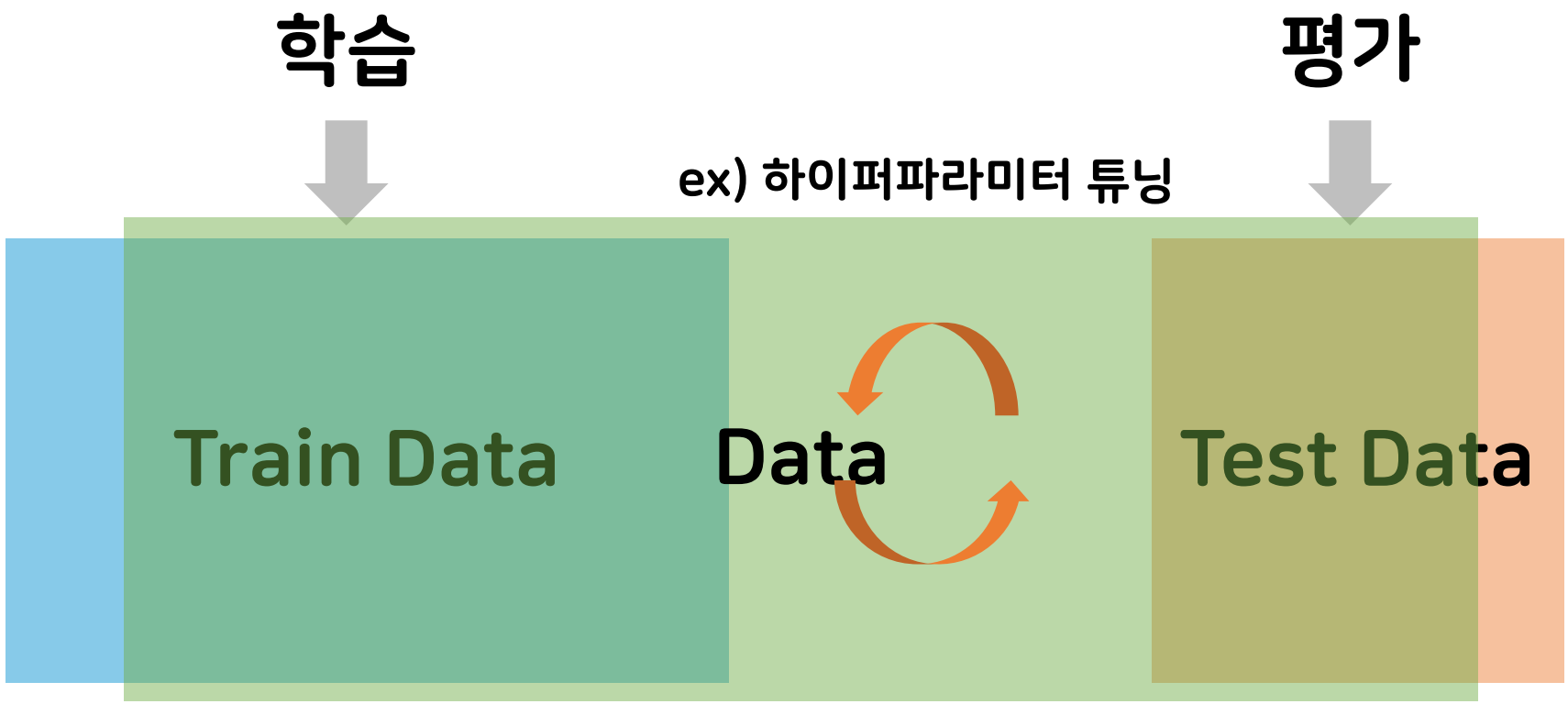
스마트인재개발원
Smart Human Resources Development



Cross validation
(교차검증)

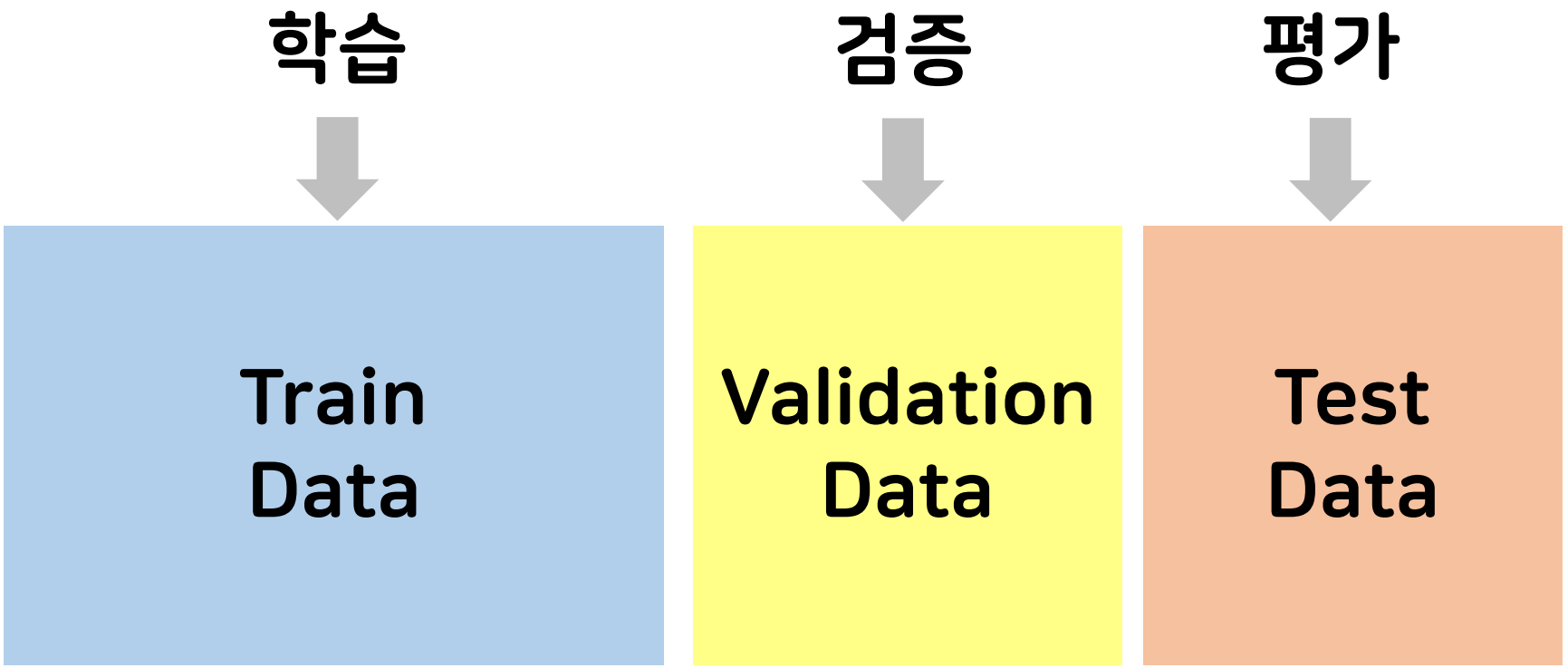


Cross validation(교차검증)



테스트 세트에 맞게 학습 될 수 있다.

Cross validation(교차검증)



Cross validation
(교차검증)

**학습-평가 데이터 나누기를 여러 번 반복하여
일반화 에러를 평가하는 방법**

K-fold cross-validation 동작 방법

1. 데이터 셋을 k개로 나눈다.

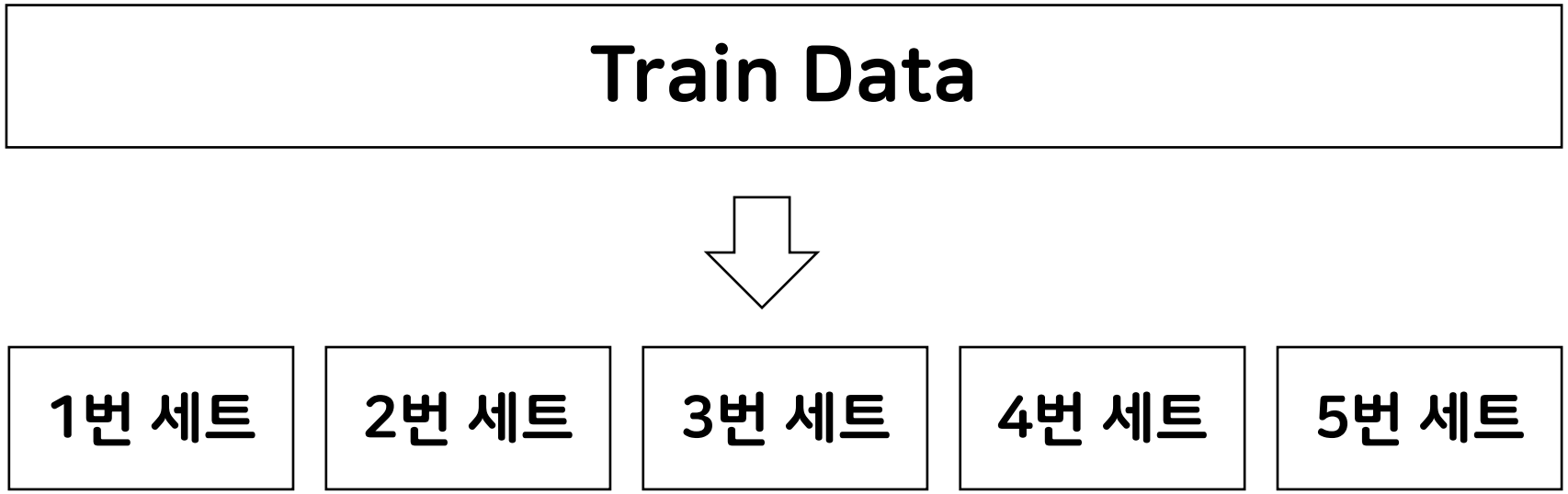
2. 첫 번째 세트를 제외하고 나머지에 대해 모델을 학습한다.
   그리고 첫 번째 세트를 이용해서 평가를 수행한다.

3. 2번 과정을 마지막 세트까지 반복한다.

4. 각 세트에 대해 구했던 평가 결과의 평균을 구한다.

Cross validation(교차검증)

K-fold cross-validation 동작 방법



Cross validation(교차검증)

K-fold cross-validation 동작 방법

cross-validation 장/단점

- 데이터의 여러 부분을 학습하고 평가해서 일반화 성능을 측정하기 때문에 안정적이고 정확하다. (샘플링 차이 최소화)
- 모델이 훈련 데이터에 대해 얼마나 민감한지 파악가능 (점수 대역 폭이 넓으면 민감)
- 데이터 세트 크기가 충분하지 않은 경우에도 유용하게 사용 가능하다.
- 여러 번 학습하고 평가하는 과정을 거치기 때문에 계산량이 많아진다

Cross validation(교차검증)

스마트인재개발원
Smart Human Resources Development

cross_val_score() 함수

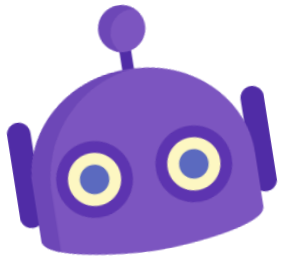
```
from sklearn.model_selection import cross_val_score

score = cross_val_score(모델, 데이터, 라벨, cv=나눌 개수)

score
```

```
[0.86680328 0.87704918 0.875      0.90554415 0.8788501 ]
```

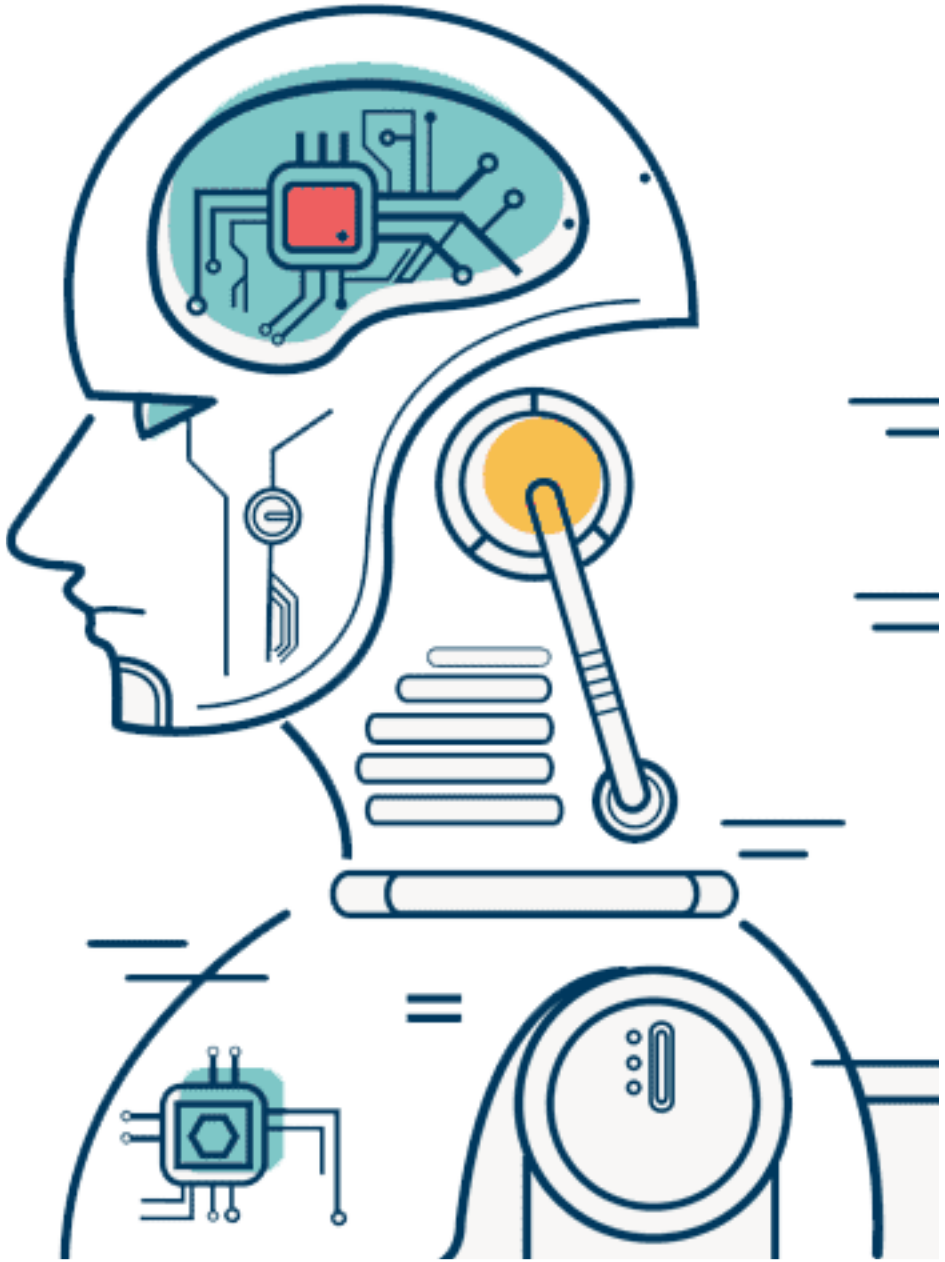
Cross validation(교차검증) 실습

Decision Tree를 활용해 Mushroom 데이터를 학습하고 교차검증을 적용해보자.

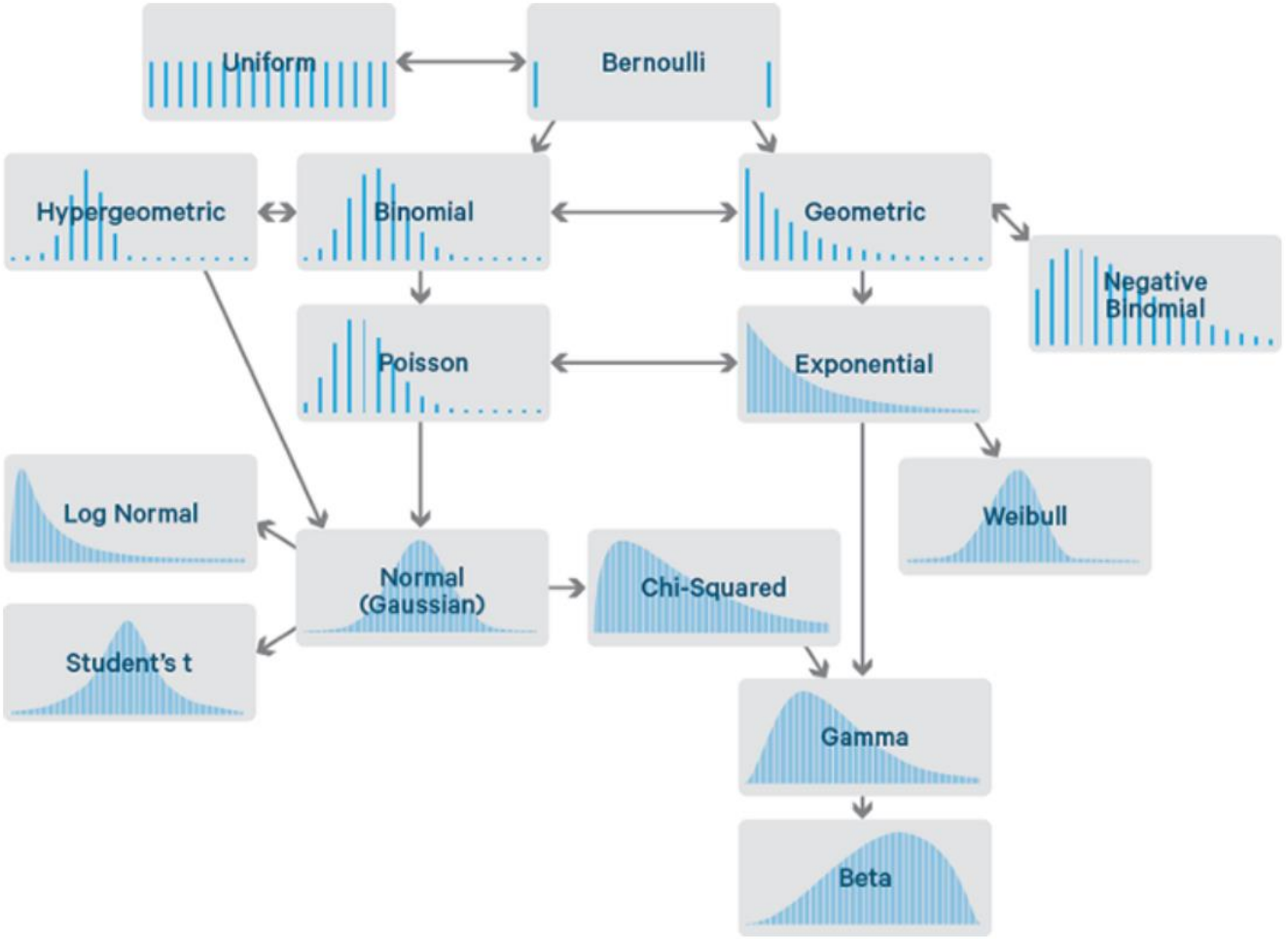
스마트인재개발원
Smart Human Resources Development



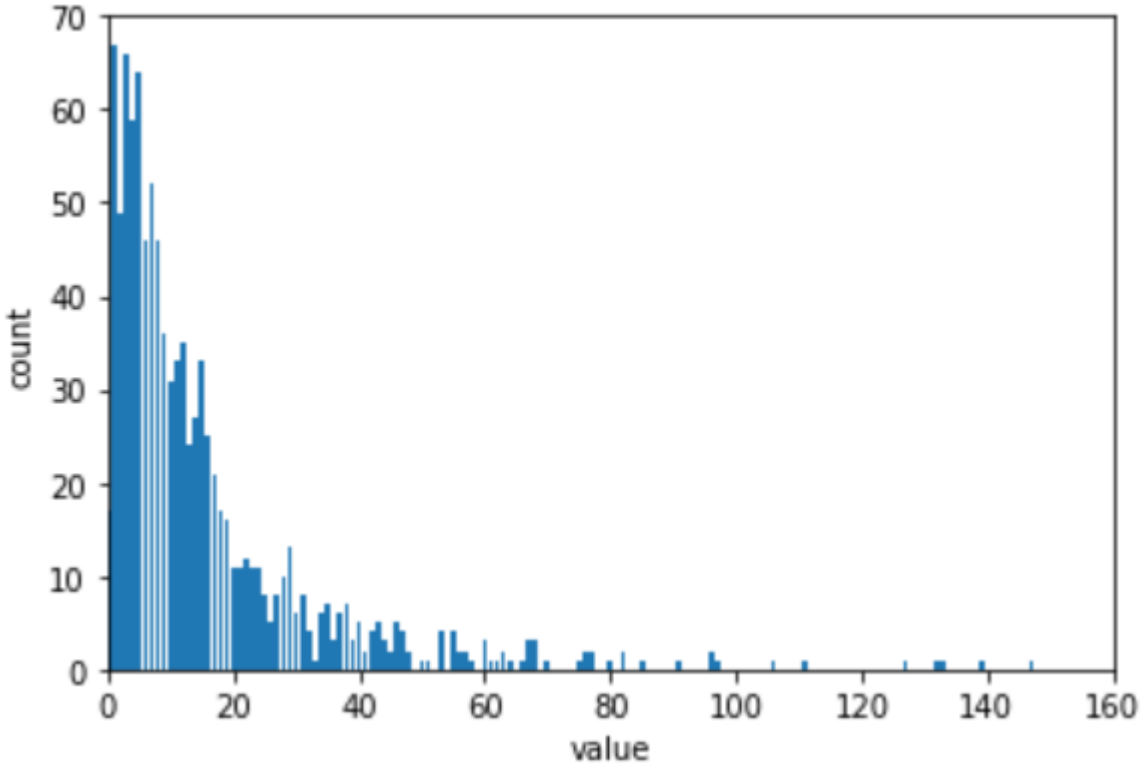
정규화 (Normalization)



Normalization : 컬럼 데이터의 분포를 정규분포로 만드는 작업

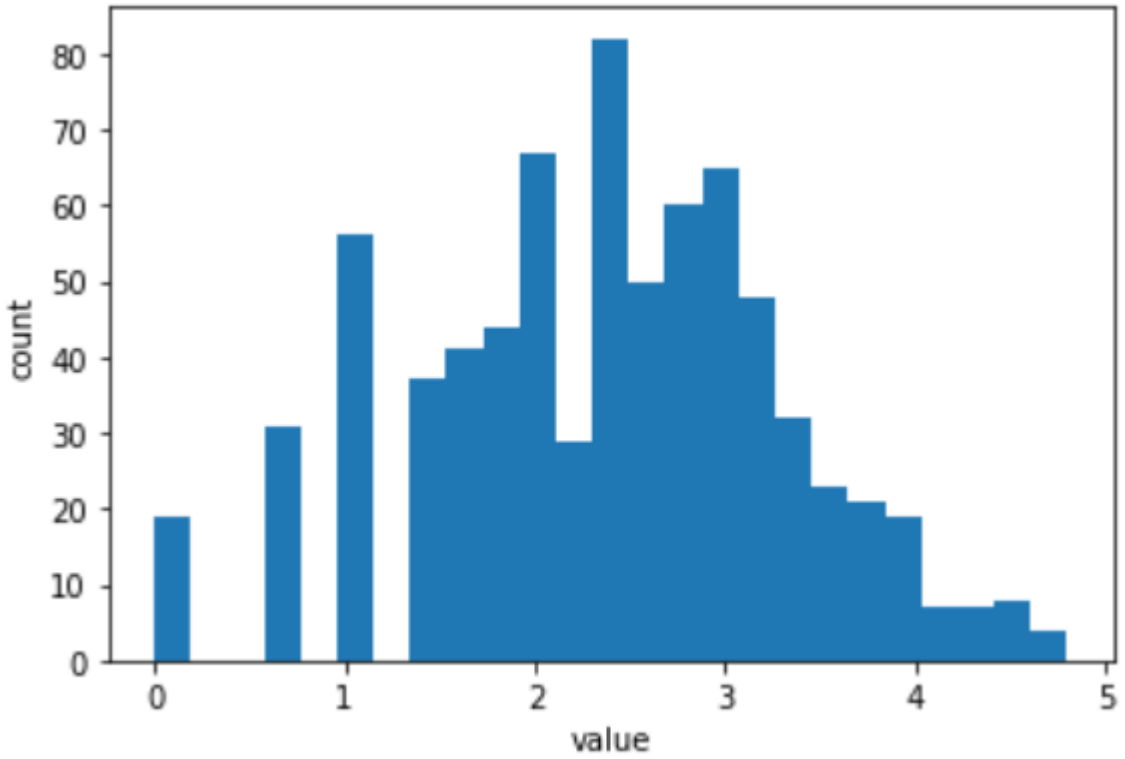


data_skew 데이터 : 포아송 분포의 회귀용 데이터



정규화

```
X_train_log = np.log(X_train + 1)
```



정규화를 하지 않는 경우와 정규화를 한 경우 KNeiborsRegressor를 이용하여 정확도가 개선되었는지 확인해보자