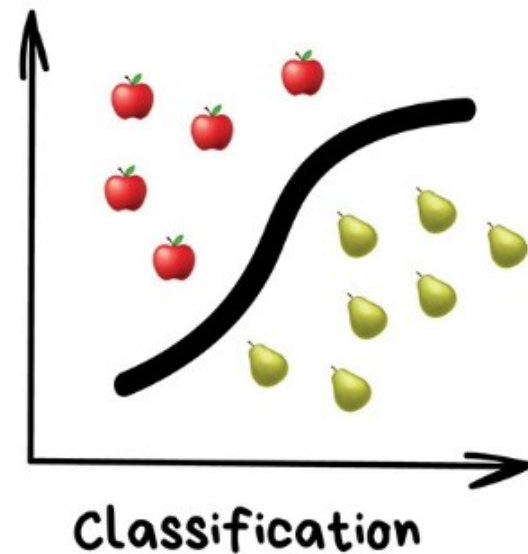


Machine Learning & Scikit-Learn

	주제	내용
1	Scikit-learn 소개와 데이터 전처리	-Scikit-learn 소개 -Data preprocessing -Data의 상관관계
2	지도학습:회귀	-선형회귀 -로지스틱 회귀
3	지도학습: 분류	-kNN - SVM
4	지도학습: 분류	-나이브 베이지안 분류기 -결정 트리
5	비지도학습: 군집 성능평가	-k-means -DBSCAN -성능평가, 정밀도, 재현율
6	파이프라인 인공신경망	- 안면인식 - Perceptron

Day4. 분류

- 분류(Classification)
 - 둘 이상의 이산적인 범주로 레이블을 예측하는 분야
 - 오늘날의 활용 분야
 - ✓ 스팸 필터링
 - ✓ 언어 분류
 - ✓ 문서 유사도 분석
 - ✓ 필기체 문자 인식
 - ✓ 사기 판단(fraud detection)
 - 방법론
 - ✓ Naive Bayes, Decision Tree, Logistic Regression, K-Nearest Neighbors, SVM(Support Vector Machine)



#4

나이브 베이지안 분류기 (Naive Bayes)

■ 나이브 베이지안 분류기

‘어리숙한’: 모든 사건을 독립적이라고 가정

- 조건부 확률 기반 방법
- (예) 날씨에 따라 필드에 나갈 지 말지를 결정하려고 하는 문제

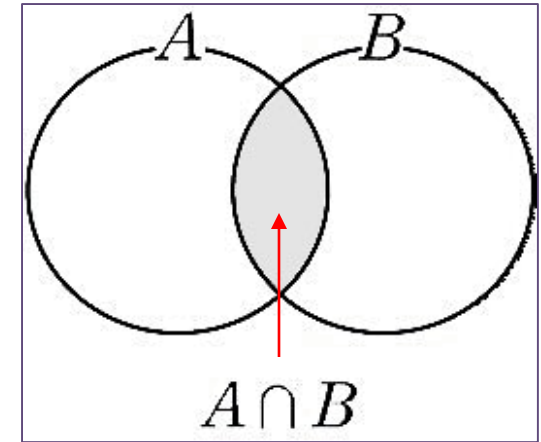
오늘 날씨: 비가 오고, 온도가 적당하고, 습도가 높고, 바람이 분다

	outlook	temperature	humidity	windy	play
0	rainy	hot	high	False	no
1	rainy	hot	high	True	no
2	overcast	hot	high	False	yes
3	sunny	mild	high	False	yes
4	sunny	cool	normal	False	yes
5	sunny	cool	normal	True	no
6	overcast	cool	normal	True	yes
7	rainy	mild	high	False	no
8	rainy	cool	normal	False	yes
9	sunny	mild	normal	False	yes
10	rainy	mild	normal	True	yes
11	overcast	mild	high	True	yes
12	overcast	hot	normal	False	yes
13	sunny	mild	high	True	no

- 조건부 확률과 베이지안 룰

- ✓ 사건 A와 B가 동시에 일어날 확률

$$P(A \cap B) = P(A) * P(B|A) = P(B) * P(A|B)$$



- 조건부 확률과 베이지안 룰

✓ 사건 A와 B가 동시에 일어날 확률

$$P(A \cap B) = P(A) * P(B|A) = P(B) * P(A|B)$$

- 검증






Number of occurrences	case B	case \bar{B}	sum
condition A	2  	3   	5
condition \bar{A}	6      	9       	15
sum	8	12	20

Diagram illustrating the calculation of $P(B, \text{given } A) \cdot P(A)$ to find $P(A \cap B)$.

Visual representation of the joint probability $P(A \cap B)$ using smiley faces: 2 green (A and B) and 3 orange (A and \bar{B}).

Visual representation of the marginal probability $P(A)$ using smiley faces: 2 green (A and B) and 6 blue (A and \bar{B}).

Visual representation of the marginal probability $P(B)$ using smiley faces: 2 green (A and B) and 9 gray (\bar{A} and B).

Visual representation of the joint probability $P(A \cap B)$ using smiley faces: 2 green (A and B) and 3 orange (A and \bar{B}).

Equation: $P(B, \text{given } A) \cdot P(A) = P(A \cap B)$

Blank boxes for calculation: $\square \cdot \square = \square$

Diagram illustrating the calculation of $P(A, \text{given } B) \cdot P(B)$ to find $P(A \cap B)$.

Visual representation of the joint probability $P(A \cap B)$ using smiley faces: 2 green (A and B) and 3 orange (A and \bar{B}).

Visual representation of the marginal probability $P(A)$ using smiley faces: 2 green (A and B) and 6 blue (A and \bar{B}).

Visual representation of the marginal probability $P(B)$ using smiley faces: 2 green (A and B) and 9 gray (\bar{A} and B).

Visual representation of the joint probability $P(A \cap B)$ using smiley faces: 2 green (A and B) and 3 orange (A and \bar{B}).

Equation: $P(A, \text{given } B) \cdot P(B) = P(A \cap B)$

Blank boxes for calculation: $\square \cdot \square = \square$

- 조건부 확률과 베이지안 룰

- ✓ 베이지안 룰

$$P(A) * P(B|A) = P(B) * P(A|B)$$

$$P(A|B) = \frac{P(A) * P(B|A)}{P(B)}$$

$P(A)$: 이미 일어난 사건의 확률(= 사전 확률, prior probability)

$P(B|A)$: 이미 일어난 상황 A에서 B가 일어날 확률(=가능도, likelihood)

$P(B)$: 모든 가설에 대한 증거가 발생할 확률(=에비던스 모델, evidence model)

$P(A|B)$: B가 일어났을 때, A가 일어날 확률(=사후 확률, posterior probability)

- Scikit learn에서 나이브 베이지안 분류 기준

$$P(A|B) = \frac{P(A) * P(B|A)}{P(B)}$$

✓ MAP(Maximum A Posteriori)

- 사후 확률을 계산하여 더 높은 확률을 가지는 것을 정답으로 분류
- (예) 오늘 날씨: 맑고, 온도가 적당하고, 습도가 높고, 바람이 분다
골프 치러 나갈까? **NO**

○ Outlook: sunny

○ Temperature: mild

○ Humidity: high

○ Windy: true

$$P(Yes|Sunny,Mild,High,True) = \frac{P(Sunny,Mild,High,True|Yes)P(Yes)}{P(Sunny,Mild,High,True)}$$

$$P(No|Sunny,Mild,High,True) = \frac{P(Sunny,Mild,High,True|No)P(No)}{P(Sunny,Mild,High,True)}$$

$$P(Sunny,Mild,High,True|Yes)P(Yes) = 0.33 * 0.44 * 0.33 * 0.33 * 9/14 = 0.02$$

$$P(Sunny,Mild,High,True|No)P(No) = 0.4 * 0.4 * 0.8 * 0.6 * 5/14 = 0.08$$

- Scikit learn에서 제공하는 나이브 베이지안 종류
(자료의 특성에 따른 분류기 제공)
 - ✓ 가우시안 나이브 베이즈 `GaussianNB()`
 - 연속적인 값에 사용
 - ✓ 다항분포 나이브 베이즈 `MultinomialNB()`
 - 이산적인 값에 사용
 - 하나의 특성이 여러 종류로 나뉘는 경우
 - ✓ 베르누이 나이브 베이즈(이항분포) `BernoulliNB()`
 - 이산적인 값에 사용
 - 모든 특성이 두 종류로만 나뉘는 경우

■ 날씨에 대한 예

(1) 자료 읽기

```
import pandas as pd  
data = pd.read_csv('c:/MLdata/weather_nominal.csv', sep=',')
```

data

	outlook	temperature	humidity	windy	play
0	rainy	hot	high	False	no
1	rainy	hot	high	True	no
2	overcast	hot	high	False	yes
3	sunny	mild	high	False	yes

■ 날씨에 대한 예

(2) Naive Bayes 모듈 중 다항분포 나이브 베이즈 가져와 인스턴스화 하기

```
from sklearn.naive_bayes import MultinomialNB
```

```
multinomial_model = MultinomialNB()
```

```
multinomial_model
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

■ 날씨에 대한 예

(3) Category 문자 데이터를 숫자 카테고리로 변환(Map 활용)

mapping dictionary 만들기

outlook_dic = {'overcast':0, 'rainy':1, 'sunny':2}

temperature_dic = {'cool':0, 'hot':1, 'mild':2}

humidity_dic = {'high':0, 'normal':1}

windy_dic = {False:0, True: 1}

딕셔너리를 활용해 데이터 매핑

data['outlook'] = data['outlook'].map(outlook_dic)

data['temperature'] = data['temperature'].map(temperature_dic)

data['humidity'] = data['humidity'].map(humidity_dic)

data['windy'] = data['windy'].map(windy_dic)

	outlook	temperature	humidity	windy	play
0	rainy	hot	high	False	no
1	rainy	hot	high	True	no
2	overcast	hot	high	False	yes
3	sunny	mild	high	False	yes
4	sunny	cool	normal	False	yes
5	sunny	cool	normal	True	no
6	overcast	cool	normal	True	yes



	outlook	temperature	humidity	windy	play
0	1	1	0	0	no
1	1	1	0	1	no
2	0	1	0	0	yes
3	2	2	0	0	yes
4	2	0	1	0	yes
5	2	0	1	1	no
6	0	0	1	1	yes
7	1	2	0	0	no
8	1	0	1	0	yes
9	2	2	1	0	yes
10	1	2	1	1	yes
11	0	2	0	1	yes
12	0	1	1	0	yes
13	2	2	0	1	no

■ 날씨에 대한 예

(4) Multinomial 모델 학습

```
multinomial_model.fit(data.iloc[:, :4], data['play'])
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

(5) 오늘의 날씨에 대한 예측

```
#오늘의 날씨에 대한 예측
```

```
multinomial_model.predict([[1, 2, 0, 1]])
```

비가 오고, 온도가 적당하고, 습도가 높고, 바람이 분다

```
outlook_dic = {'overcast':0, 'rainy':1, 'sunny':2}
temperature_dic = {'cool':0, 'hot':1, 'mild':2}
humidity_dic = {'high':0, 'normal':1}
windy_dic = {False:0, True: 1}
```

- 날씨에 대한 예
(6) 계산된 확률

계산된 확률

```
multinomial_model.predict_proba([[2, 2, 0, 1]])
```

```
array([[0.55477945, 0.44522055]])
```

NO 확률

Yes 확률

■ 연속적인 데이터(iris)에 대한 나이브 베이지안 분류기

(1) 자료의 준비

```
from sklearn.datasets import load_iris  
iris = load_iris()
```

```
iris_df = pd.DataFrame(iris.data, columns=iris.feature_names)  
iris_df['species'] = iris.target
```

```
iris_df.head()
```

Data frame
으로 변환

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

- 연속적인 데이터(iris)에 대한 나이브 베이지안 분류기

- (2) 가우시안 모델 인스턴스화

```
# 가우시안 모델 인스턴스화  
from sklearn.naive_bayes import GaussianNB  
gaussian_model = GaussianNB()
```

- 연속적인 데이터(iris)에 대한 나이브 베이지안 분류기

(3) 데이터를 학습데이터와 테스트 데이터로 분할 후 학습 데이터로 학습

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris_df.iloc[:, :4],
                                                    iris_df['species'],
                                                    test_size = 0.33)
```

```
gaussian_model.fit(X_train, y_train)
```

GaussianNB(priors=None, var_smoothing=1e-09)

■ 연속적인 데이터(iris)에 대한 나이브 베이지안 분류기

(4) 테스트 데이터로 성능 평가

성능 평가

```
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test, gaussian_model.predict(X_test)))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
1	0.94	0.94	0.94	18
2	0.93	0.93	0.93	14
accuracy			0.96	50
macro avg	0.96	0.96	0.96	50
weighted avg	0.96	0.96	0.96	50

```
print(confusion_matrix(y_test, gaussian_model.predict(X_test)))
```

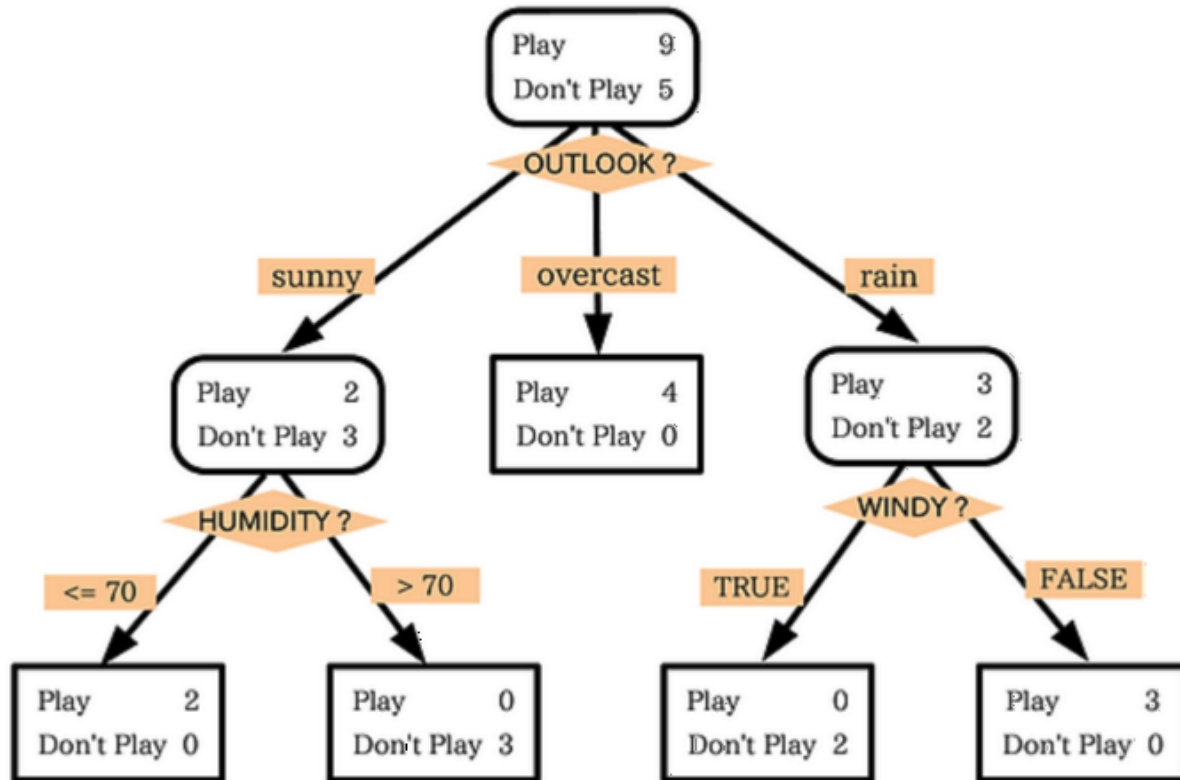
```
[[18  0  0]
 [ 0 17  1]
 [ 0  1 13]]
```

#5

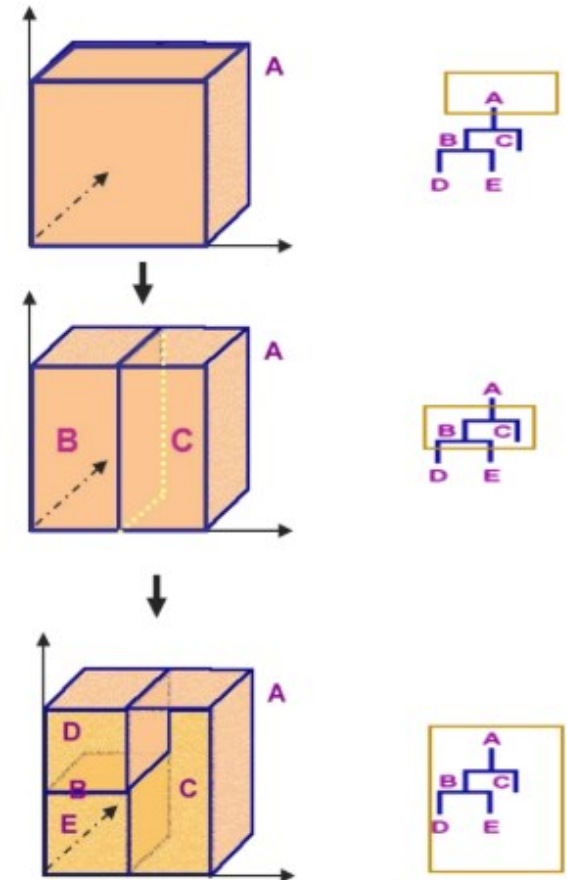
결정 트리 (Decision Tree)

결정 트리 모델

Dependent variable: PLAY



트리에 의한 영역 분할



■ 결정 트리 모델

- 아직 일어나지않은 일에 대하여 미리 판단해야 하는 종류의 문제에 적용
- 데이터들의 분포를 분류하여 결과를 예측하는 알고리즘
 - ✓ 어떤 모양의 (선형, 비선형 상관 없이) 데이터가 들어와도 풀 수 있음
- Classification과 regression 을 수행하는 분류 알고리즘

■ 장점

- ✓ 시각화 될 수 있어, 이해하기 쉬우며 해석이 간단하다.
- ✓ 데이터를 전처리하는 절차가 간단하다.
- ✓ 데이터가 늘어남에 따라 트리의 사용 횟수는 \log 로 증가한다.
- ✓ 여러 개의 결과 값이 나오는 데이터의 분석이 가능하다
- ✓ White-box 모델이다. 내부가 투명하여 결과가 나오는 이유를 설명할 수 있다.
- ✓ 통계학적인 추정을 통하며 모델의 신뢰성을 평가할 수 있다.

■ 단점

- ✓ 과하게 복잡한 트리를 생성해서 데이터를 일반화를 잘 시키지 못한다.
- ✓ 불안정하다. - 데이터의 약간의 변이는 다른 decision tree를 형성할 수 있다.
- ✓ 지역 최적화 방법이다.
- ✓ XOR과 같은 문제를 잘 표현하지 못한다.

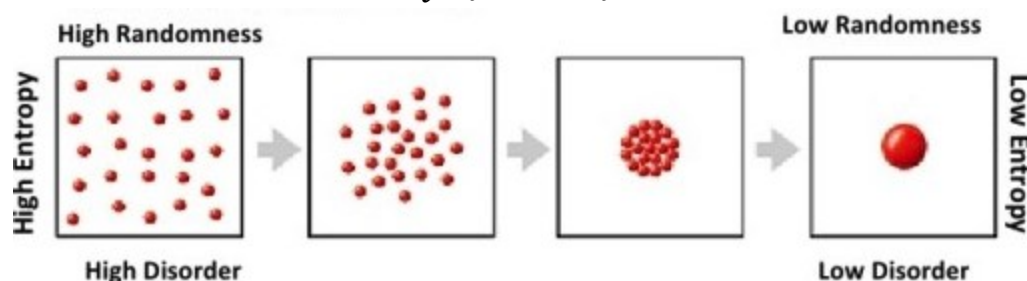
- 결정 트리의 영역 분할
 - 정보획득(Information gain)이 많은 방향
 - 영역의 순도(homogeneity)는 증가, 불순도(Impurity) 감소하는 방향으로 분할

■ 순도를 계산하는 방법

1) Entropy에 의한 방법

✓ 엔트로피 = 무질서도 = 정보력이 낮음

$$Entropy(p) = -\sum_i p_i \log_2 p_i$$



2) Gini index(불순도)에 의한 방법

3) MSE(Mean square error)에 의한 방법

- Regression 모델로 사용되는 경우

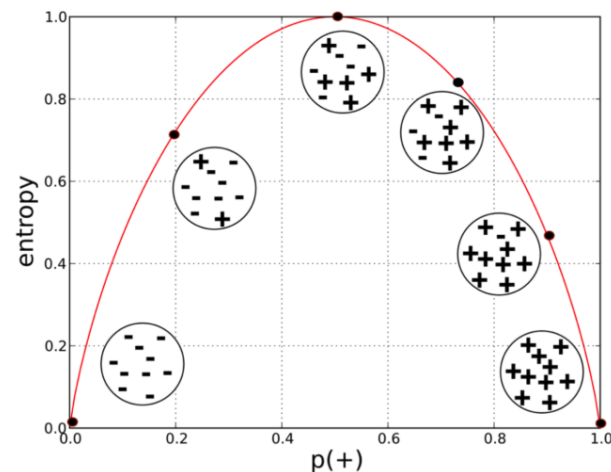
■ 결정 트리를 만드는 방법

1) Entropy에 의한 방법

- ✓ m개의 레코드가 속하는 A 영역에 대한 엔트로피

$$Entropy(A) = - \sum_{k=1}^m p_k \log_2(p_k)$$

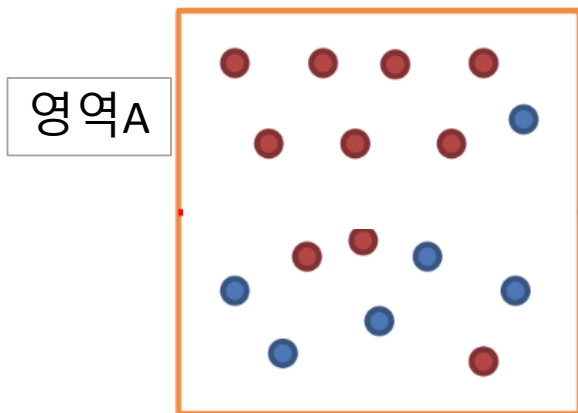
- ✓ 분할에 의해 엔트로피가 낮아짐



■ 결정 트리를 만드는 방법

1) Entropy에 의한 방법

- ✓ 예) 빨간 동그라미(범주 1) 10개, 파란동그라미(범주 2) 6개 일 때, 영역 A의 엔트로피



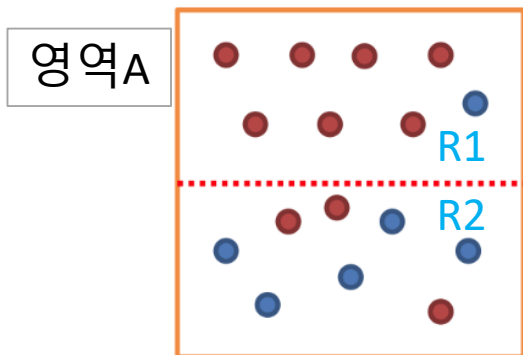
$$Entropy(A) = - \sum_{k=1}^m p_k \log_2(p_k)$$

$$Entropy(A) = -\frac{10}{16} \log_2 \left(\frac{10}{16} \right) - \frac{6}{16} \log_2 \left(\frac{6}{16} \right) \approx 0.95$$

■ 결정 트리를 만드는 방법

1) Entropy에 의한 방법

✓ 예) 분할 후의 엔트로피



$$Entropy(A) = \sum_{i=1}^d R_i \left(- \sum_{k=1}^m p_k \log_2(p_k) \right)$$

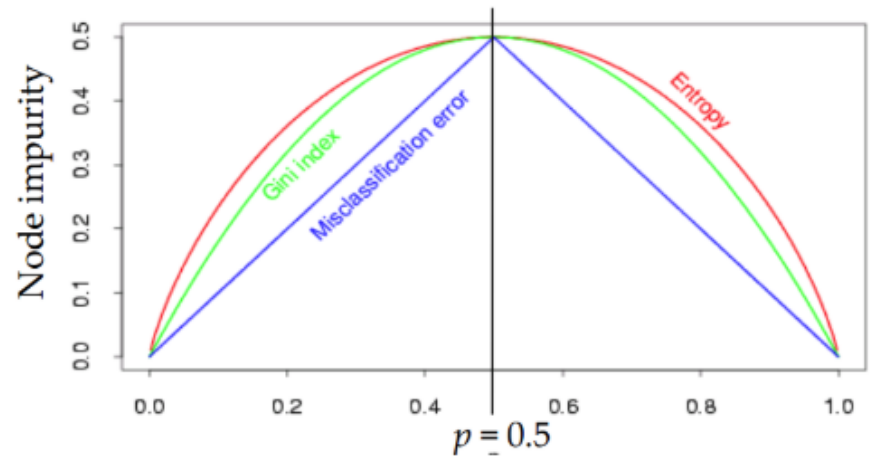
(R_i : 분할 전 자료 중에 분할 후 i 영역에 속하는 비율)

$$Entropy(A) = \underbrace{0.5 \times \left(-\frac{7}{8} \log_2 \left(\frac{7}{8} \right) - \frac{1}{8} \log_2 \left(\frac{1}{8} \right) \right)}_{8/16} + \underbrace{0.5 \times \left(-\frac{3}{8} \log_2 \left(\frac{3}{8} \right) - \frac{5}{8} \log_2 \left(\frac{5}{8} \right) \right)}_{8/16} \approx 0.75$$

- 결정 트리를 만드는 방법

- 2) 지니 인덱스에 의한 방법

$$G.I(A) = \sum_{i=1}^d \left(R_i \left(1 - \sum_{k=1}^m p_{ik}^2 \right) \right)$$



■ Weather 데이터에 적용하기

(1) 데이터 준비

```
import pandas as pd  
data = pd.read_csv('c:/MLdata/weather_nominal.csv', sep=',')
```

data

	outlook	temperature	humidity	windy	play
0	rainy	hot	high	False	no
1	rainy	hot	high	True	no
2	overcast	hot	high	False	yes
3	sunny	mild	high	False	yes
4	sunny	cool	normal	False	yes
5	sunny	cool	normal	True	no
6	overcast	cool	normal	True	yes
7	rainy	mild	high	False	no
8	rainy	cool	normal	False	yes
9	sunny	mild	normal	False	yes
10	rainy	mild	normal	True	yes
11	overcast	mild	high	True	yes
12	overcast	hot	normal	False	yes
13	sunny	mild	high	True	no

■ Weather 데이터에 적용하기

(2) 카테고리 문자열을 수치로 변환

```
# mapping dictionary 만들기
```

```
outlook_dic = {'overcast':0, 'rainy':1, 'sunny':2}
```

```
temperature_dic = {'cool':0, 'hot':1, 'mild':2}
```

```
humidity_dic = {'high':0, 'normal':1}
```

```
windy_dic = {False:0, True: 1}
```

```
# 딕셔너리를 활용해 데이터 매핑
```

```
data['outlook'] = data['outlook'].map(outlook_dic)
```

```
data['temperature'] = data['temperature'].map(temperature_dic)
```

```
data['humidity'] = data['humidity'].map(humidity_dic)
```

```
data['windy'] = data['windy'].map(windy_dic)
```

data

	outlook	temperature	humidity	windy	play
0	1	1	0	0	no
1	1	1	0	1	no
2	0	1	0	0	yes

■ Weather 데이터에 적용하기

(3) 결정 트리 분류기 인스턴스 생성

```
# Decision Tree classifier instance 생성
```

```
from sklearn import tree
```

```
clf = tree.DecisionTreeClassifier()
```

```
clf
```

'gini' 혹은 'entropy'

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,  
                        max_features=None, max_leaf_nodes=None,  
                        min_impurity_decrease=0.0, min_impurity_split=None,  
                        min_samples_leaf=1, min_samples_split=2,  
                        min_weight_fraction_leaf=0.0, presort=False,  
                        random_state=None, splitter='best')
```

- Weather 데이터에 적용하기

- (4) 결정 트리 분류기 학습

```
# 결정 트리 분류기 학습
```

```
clf.fit(data.iloc[:, :4], data['play'])
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,  
                        max_features=None, max_leaf_nodes=None,  
                        min_impurity_decrease=0.0, min_impurity_split=None,  
                        min_samples_leaf=1, min_samples_split=2,  
                        min_weight_fraction_leaf=0.0, presort=False,  
                        random_state=None, splitter='best')
```

- Weather 데이터에 적용하기

(5) 오늘같은 날씨에는 필드에 나갈까? 예측

맑고, 온도가 적당하고, 습도가 높고, 바람이 분다

```
clf.predict([[2,2,0,1]])
```

```
array(['no'], dtype=object)
```

비가 오고, 온도가 적당하고, 습도가 높고, 바람이 분다

```
clf.predict([[1,2,0,1]])
```

```
array(['no'], dtype=object)
```

```
outlook_dic = {'overcast':0, 'rainy':1, 'sunny':2}
temperature_dic = {'cool':0, 'hot':1, 'mild':2}
humidity_dic = {'high':0, 'normal':1}
windy_dic = {False:0, True: 1}
```

Q & A