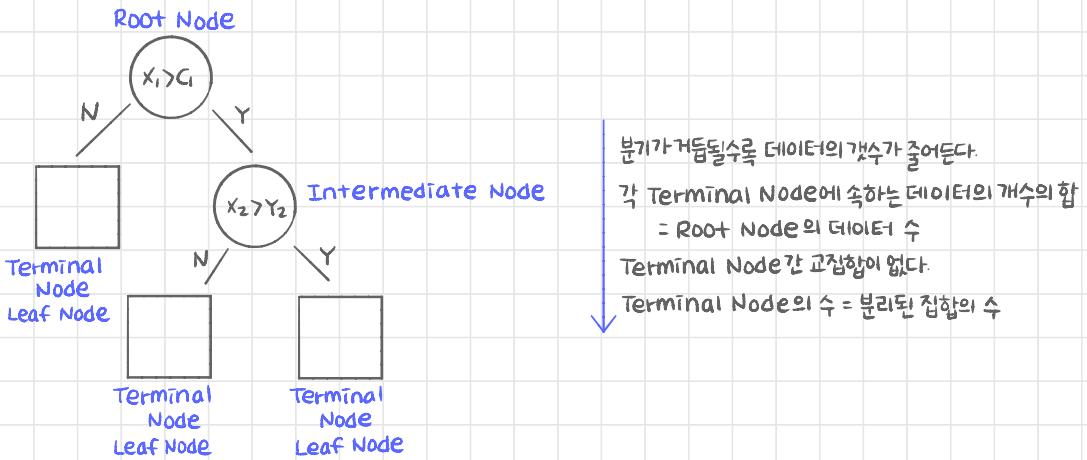


## Decision Tree

- 한번에 하나씩의 설명변수를 사용하여 예측 가능한 규칙들의 집합을 생성하는 알고리즘
  - 장점: 데이터 전처리가 거의 필요하지 않다.
  - Classification, regression 모두 가능
- ↳ 최종목표: 각 노드들의 엔트로피 지수를 0으로, 즉 한 노드(공간)에 같은 특성을 가진 데이터들만 모이게 하는 것이다.



### \* decision tree의 classification

- 새로운 데이터가 특정 Terminal Node에 속한다는 정보를 확인한 뒤 해당 Terminal Node에서 가장 빈도가 높은 범주에 새로운 데이터를 분류

### \* decision tree의 Regression

- 해당 Terminal Node의 종속변수의 평균을 예측값으로 반환
  - 예측값의 종류는 Terminal Node 갯수와 일치
- ex) Terminal Node의 수가 3개라면 새로운 데이터가 1000개가 주어져도 decision tree는 딱 3종류의 결과를 출력한다.



Root node A가 어떤 기준에 의해 B와 C로 분할,  
B가 D와 E로 분할된다면, terminal node는 C, D, E  
→ 전체 데이터 A가 세 개의 부분집합 C, D, E로 분할된 것.

D특성을 가지고 있는 새로운 데이터가 주어졌을 때 decision tree는 D집합을 대표할 수 있는 값 (classification = 최빈값, regression = 평균)을 반환하는 방식으로 예측

## \* Impurity / Uncertainty

- decision tree는 구분 뒤 각 영역의 순도 (Homogeneity)가 증가, 불순도 (Impurity) 혹은 불확실성 (Uncertainty)이 최대한 감소하도록 하는 방향으로 학습을 진행한다.
- information gain : homogeneity ↑  
impurity / uncertainty ↓

## \* 순도(homogeneity)를 측정하는 방법

- 엔트로피(entropy)

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

( $m$ 개의 레코드가 속하는 A영역에 대한 entropy)

$p_k = A$ 영역에 속하는 레코드 가운데  $k$ 번주에 속하는 레코드의 비율)

· entropy 감소 = uncertainty 감소 = homogeneity 증가 = impurity 감소 = information gain

·  $A$ 영역에 속한 모든 레코드가 동일한 범주에 속할 경우 (= uncertainty 최소 = homogeneity 최대) entropy = 0

· 범주가 둘뿐이고 해당 개체의 수가 동일하게 반반씩 섞여 있을 경우 (= uncertainty 최대 = homogeneity 최소)

entropy = 1



- 지니계수(Gini Index)

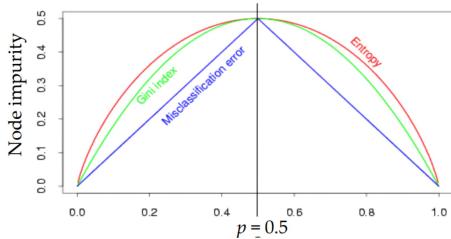
$$G.I(A) = \sum_{i=1}^d R_i \left( 1 - \sum_{k=1}^m p_{ik}^2 \right)$$

Sex <= 0.5
gini = 0.473
samples = 891
value = [549, 342]
class = Survived

$$\rightarrow 1 - \left( \frac{549^2}{891} + \frac{342^2}{891} \right)$$

$$= 0.379 - 0.147$$

$$= 0.473$$



$p=0.5$  (두 범주가 반반씩 섞여 있는 경우) 일 때  
impurity 최대

범주가 두 개일 때 한쪽 범주에 속한 비율 ( $p$ )에 따른 불순도의 변화량

# Random Forest



- decision tree를 여러 개 모아놓으면 숲이 되는데 이 숲을 구성하는 방법을 random으로 함
- 나무마다 독립 변수가 다르게 들어갈 수 있도록 그 수(독립 변수 사용 갯수)를 제한 시키는 것 → bagging 기법 사용
- Bagging : Bootstrap aggregating

원 표본에서 중복을 허용하여 데이터 셋을 만든다.

데이터 셋을 여러 개 만들 수 있고, 데이터가 각 셋마다 다르다.

bagging을 통해 random forest를 훈련시키는 방법

1. bootstrap을 통해  $N$ 개의 train data set을 생성한다.

2.  $N$ 개의 기초 분류기(tree)들을 훈련시킨다.

3. 기초 분류기(tree)들을 하나의 분류기(random forest)로 결합한다. (평균 또는 과반수 투표 방식 이용)

- 장점 : decision tree보다 정확도가 높아진다.

- 단점 : decision tree가 가지는 설명력을 상실한다. (변수의 가중치, 우선순위를 파악할 수 없으니 결과를 해석할 수 없음)

여러 트리들을 '다르게' 만든다.

