



# 제조 데이터 사이언스

## 분류 모형

2025 Spring

Industrial Data Science Lab & Unique AI

# 1. 지도학습과 비지도학습

## ➤ Machine Learning 모형 구분

### 지도학습 (Supervised Learning)

종속 및 독립변수를 이용하여 주어진 독립(설명)변수를 바탕으로 종속(반응)변수 예측 모형 제시

예: 회귀/분류 모형

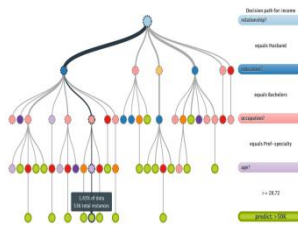


### 비지도학습 (Unsupervised Learning)

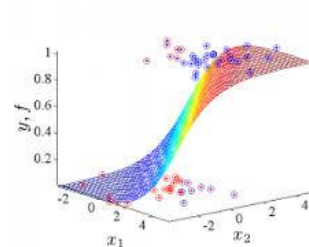
Target(종속변수/반응변수)이 없으며, 독립(설명)변수 간의 관계나 이를 바탕으로 개체들을 구분하여 의미 있는 결과를 제시

예: 군집 분석, 연관성 분석, 주성분 / 요인 분석

[decision tree]



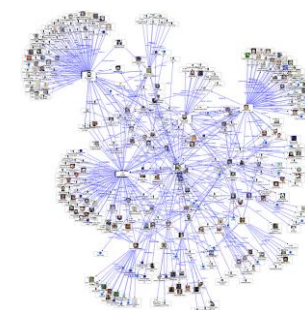
[logistic regression]



[clustering analysis]

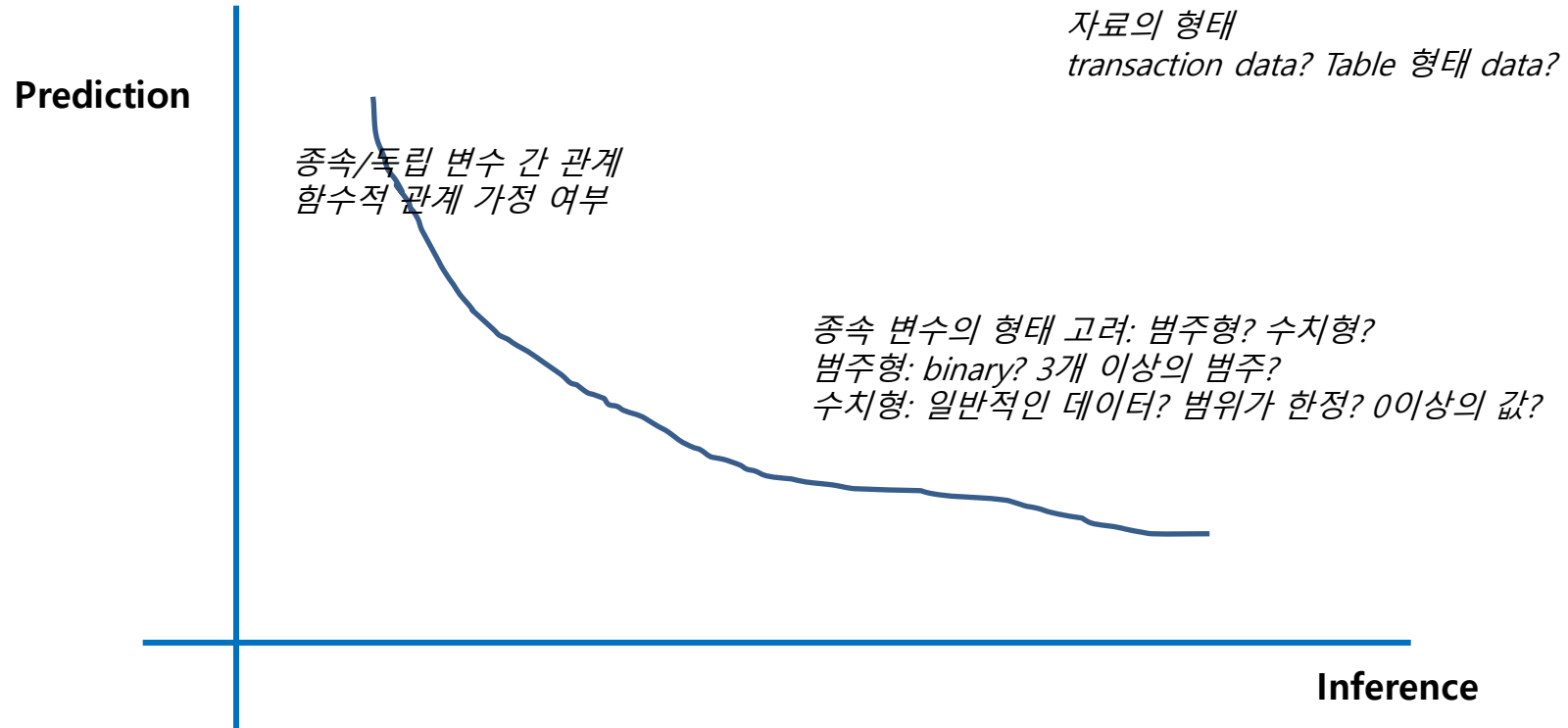


[link analysis]



# 1. 지도학습과 비지도학습

## ➤ 데이터 분석의 목적



## 2. 분류모형이란

- **분류(Classification):**
  - 지도 학습 중에서, 주어진 데이터를 기반으로 범주형 Target값에 따라 분류하고 예측하는 모형
- **정분류율(Accuracy):**
  - 분류모형에서 실제 Target을 정확하게 예측한 비율
- **교차검증(Cross Validation):**
  - 훈련용 데이터로 모델링을 하고 테스트 데이터로 모형의 성능을 파악

## 2. 분류모형이란

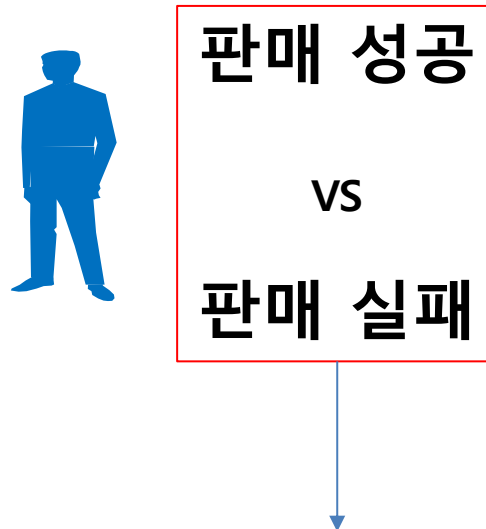
X1	X2	X3	Y
Yes	12	Blue	O
Yes	87	Green	O
No	44	Blue	X
Yes	19	Red	X
No	32	Green	O
No	14	Blue	O

## 2. 분류모형이란

- 분류(Classification) 모형

- 주어진 변수를 사용해서 **Target 변수를** 예측하는 지도 학습의 기법으로,
- 주어진 데이터를 바탕으로 **범주형 Target값에 따라** 분류하고 예측하는 모형

예: 마케팅의 성공 여부?



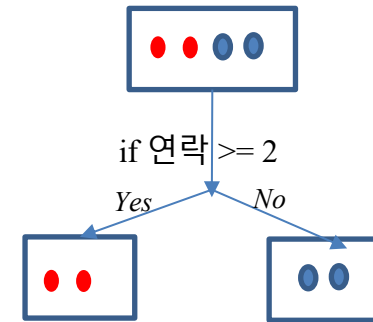
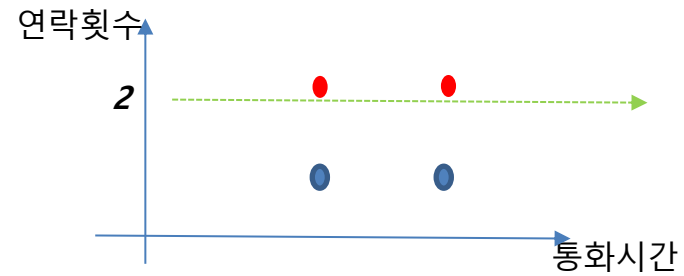
범주(Category): 같은 특성을 지닌 부류나 범위!

## 2. 분류모형이란

고객	target	input	
	판매여부	연락횟수	통화시간
AAA	N	1	5.5
BBB	N	1	4.5
CCC	Y	2	4.5
DDD	Y	2	5.5



● 성공  
● 불가



## 2. 분류모형이란

- 판매 예측 모형은 얼마나 정확할까? Cross Validation

마케팅 결과

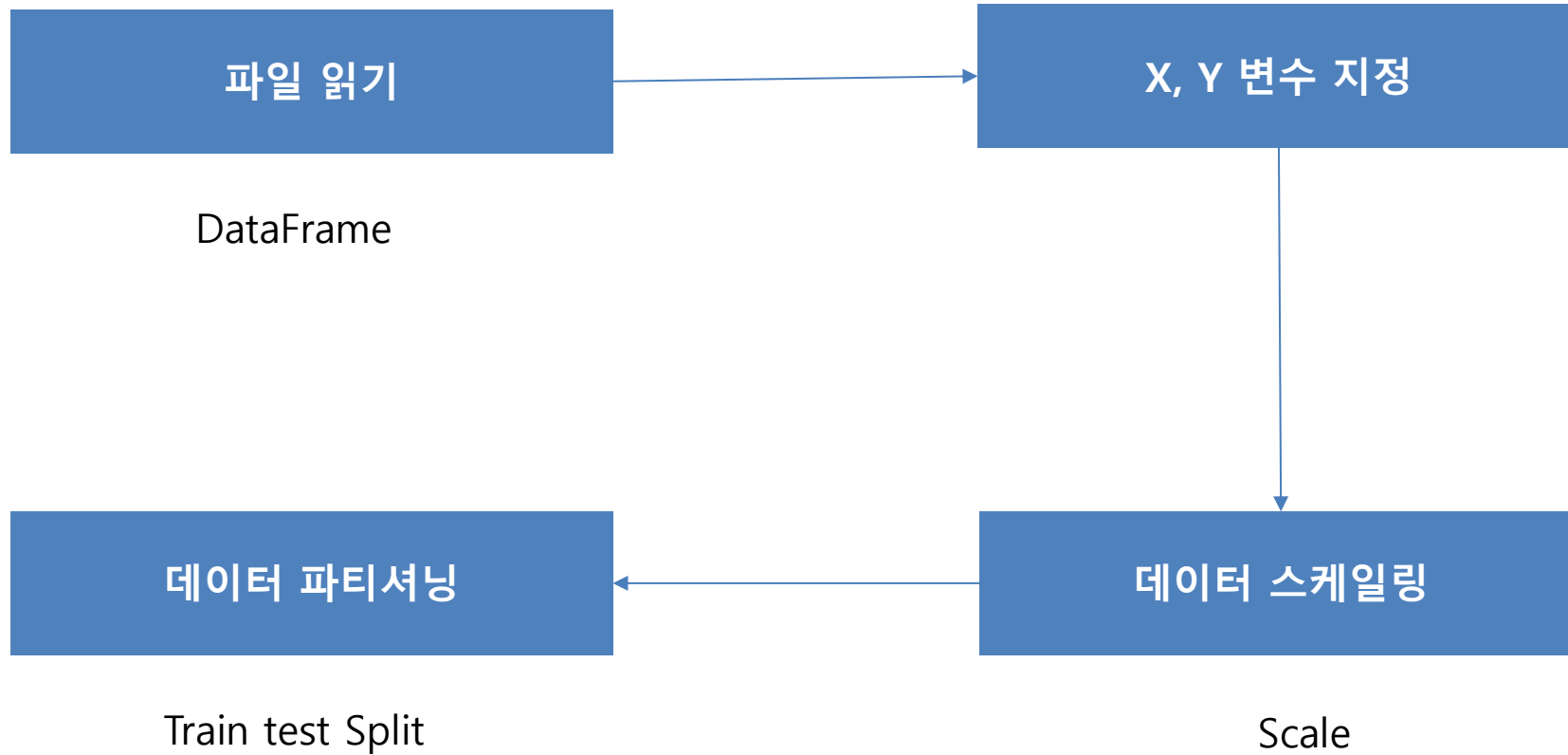
고객	판매 여부 (예측)	실제 결과
AAA	Y	N
BBB	N	N
CCC	Y	Y
DDD	N	Y

정분류율 (Accuracy): 50%



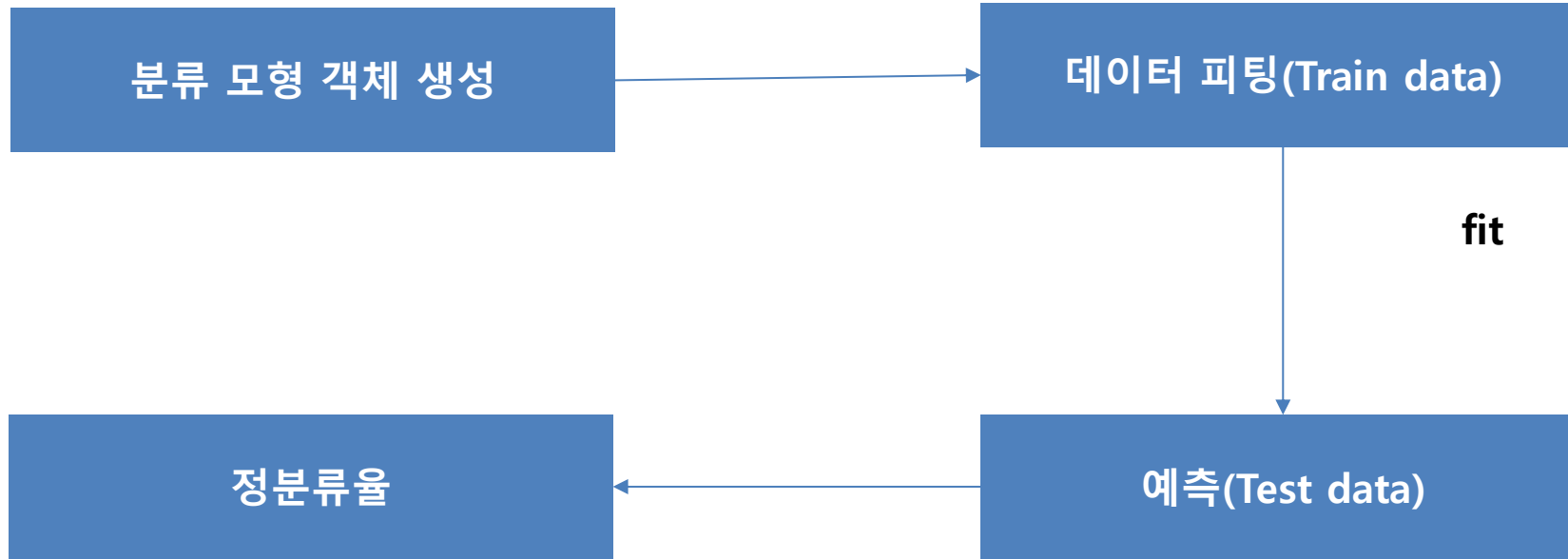
## 2. 분류모형이란

### 분류를 위한 데이터 준비



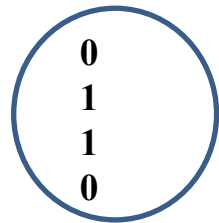
## 2. 분류모형이란

### 분류 모델링



불리언 결과에 평균!

- Y와 N이 같은지? False
- N과 N이 같은지? True
- Y와 Y가 같은지? True
- N과 Y가 같은지? False



평균!

predict

## 2. 분류모형이란

- 분류모형의 평가
  - Confusion Matrix

	실제 Y	실제 N
예측 Y	True Positive(TP)	False Positive(FP)
예측 N	False Negative(FN)	True Negative(TN)

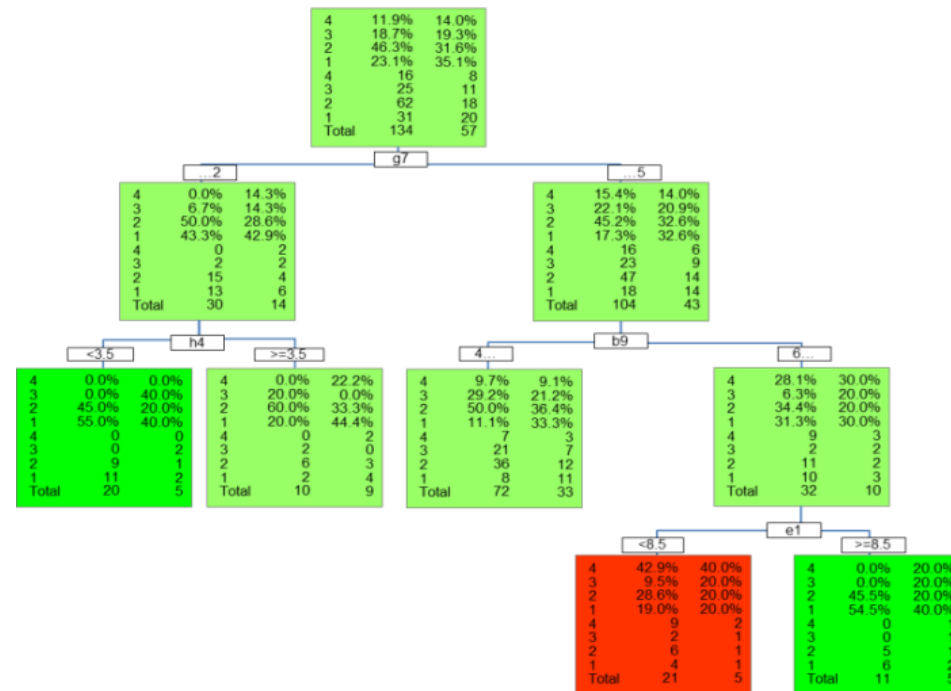
- $N = TP + FP + FN + TN$
- 예측 결과에 따라 True, False 구분
- 예측 값에 따라 Positive, Negative 구분

Metric	Formula	설명
정분류율 or Accuracy	$(TP + TN) / N$	전체 결과 중 맞게 분류한 비율
오분류율	$(FP + FN) / N$	전체 결과 중 잘못 분류한 비율
Precision	$TP / (TP + FP)$	Y로 예측된 것 중 실제로도 Y인 비율

### 3. Decision Tree

- **Decision Tree**

- An empirical tree represents a segmentation of the data that is created by applying a series of simple rules
- 장점:
  - 해석의 용이성 / 상호작용 효과의 해석: / 비모수적 모형(선형성, 정규성, 등분산성의 가정 불필요)

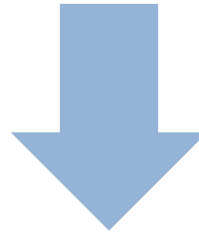


### 3. Decision Tree

- Tree 구조: 그래프의 일종, 여러 노드로 구성이 되며, 서로 다른 두 노드를 잇는 길이 하나 뿐인 구조, 최상위 노드인 루트노드로 부터 분기되어 형성
- 의사결정나무(Decision Tree): 일련의 간단한 규칙으로 만들어진, Target을 나눠가는 Tree 구조를 사용하는 분류모형
- 가지치기(Pruning): Decision Tree에서 자료에 과적합되는 것을 방지하기 위해 오분류율이 커지는 분기를 제거하는 것을 의미

### 3. Decision Tree

분류(Classification) 모형은 범주형 Target 변수를 예측하는 지도 학습의 기법!



#### Decision tree

- 분류 기법 중 하나
- 1980년대 부터 사용!
  - C4.5 / C5.0 : information theory, entropy, Quinlan (1983)
  - CART(Classification and regression Tree): Gini index, Breiman et al. (1984)
  - CHAID(Chi-squared Automatic Interaction Detector): Chi-square test 이용, Kass(1980)
  - .....

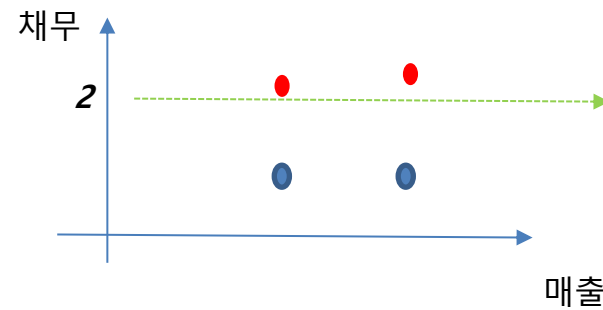
### 3. Decision Tree

투자 대상 기업 4곳!

회사	target	input	
	부도여부	채무	매출
AAA	N	1	5.5
BBB	N	1	4.5
CCC	Y	2	4.5
DDD	Y	2.1	5.5

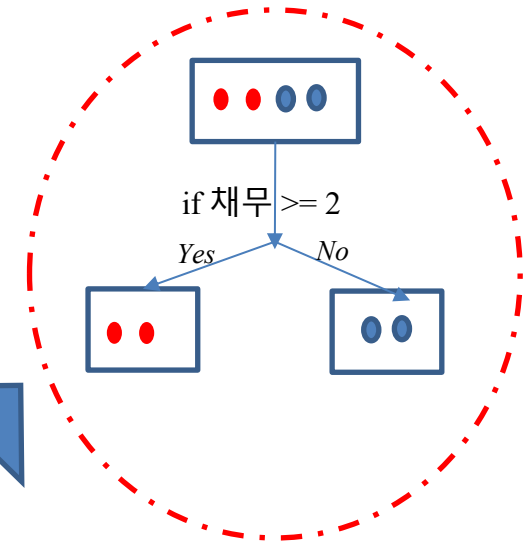
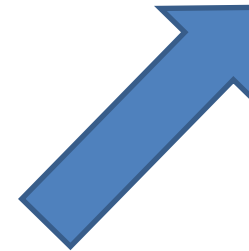


● 부도  
● 정상



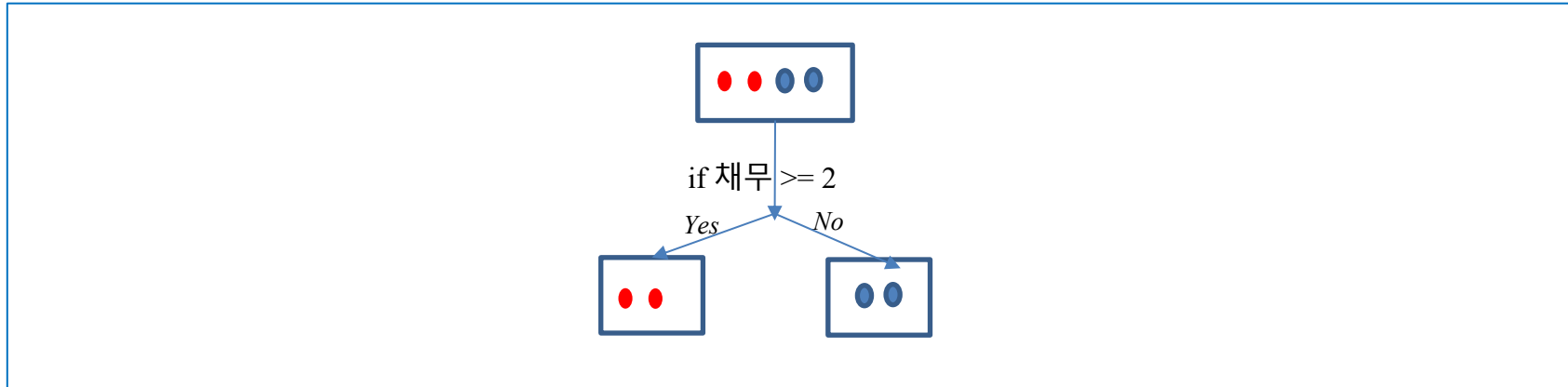
15

15



결정을 해주는 나무?

### 3. Decision Tree



- Tree 구조: 그래프의 일종, Root로 부터 시작해서, 부모 노드에서 자식노드로 분기해나가며 구성된 자료 구조



Decision Tree(의사결정나무)

***“일련의 간단한 규칙으로 만들어진, Target을 나뉘는 Tree 구조”***



### 3. Decision Tree

회사	부도여부	채무	매출
AAA	N	1	5.5
BBB	N	1	4.5
CCC	Y	2	4.5
DDD	Y	2.1	5.5

채무 변수의 값들: 1, 2, 2.1

1을 기준으로

1 이상



VS

1 미만

2를 기준으로

2 이상



VS

2 미만



2.1을 기준으로

2.1 이상

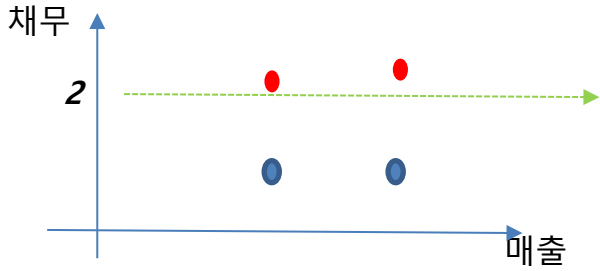


VS

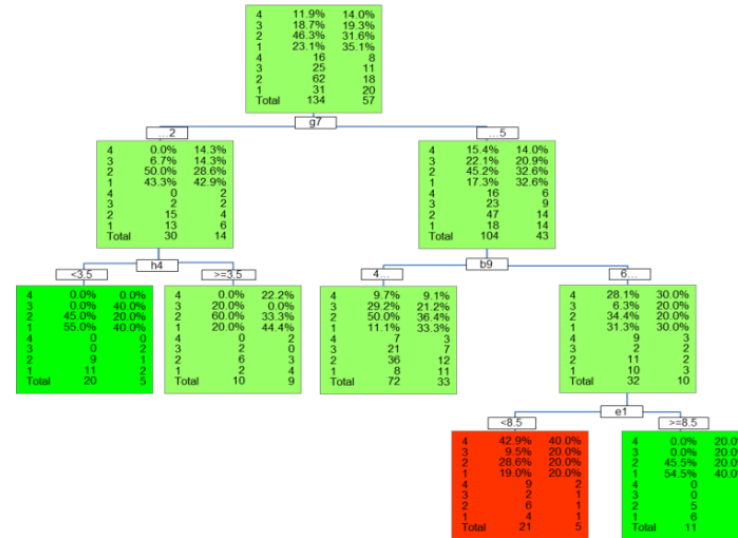
2.1 미만



● 부도  
● 정상



### 3. Decision Tree



- 해석의 용이성
- 상호작용 효과의 해석
- 복잡한 가정 불필요!

VS

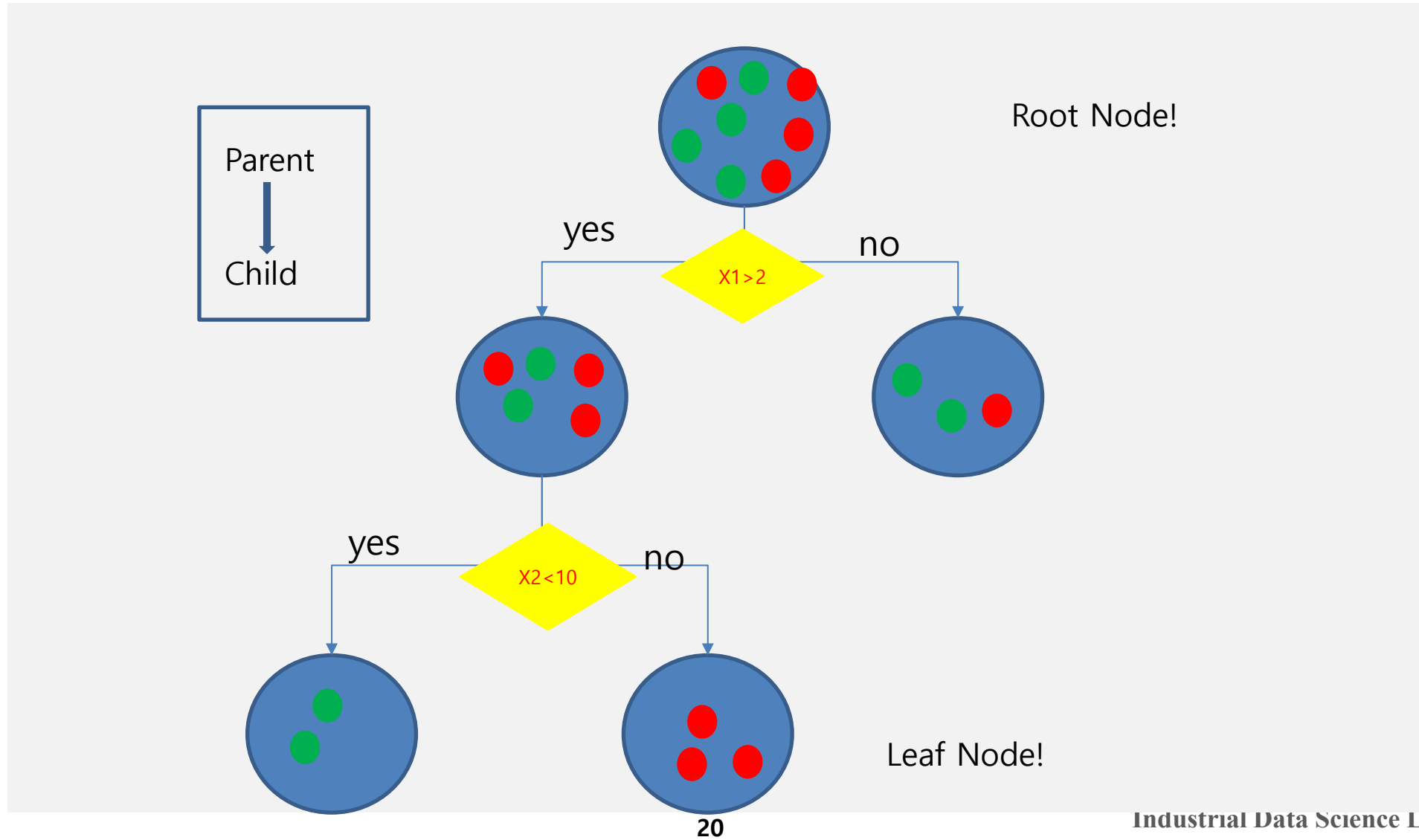
- 비연속성
- 선형성 또는 주효과 결여
- 안정성 부족

### 3. Decision Tree

- **Decision Tree 주요 구성요소**
  - Decision Tree는 Root부터 Leaf node 사이 여러 Node로 구성되며, 이때 Node의 분기는 Rule에 의해 이뤄짐.
  - Root부터 각 Leaf Node까지를 각각 Branch라고 하며, Branch까지의 노드의 수를 Depth라고 부름.
- Decision Tree의 시각화: Text로 출력된 Rule들을 효과적으로 보기 위해 Tree구조를 시각화

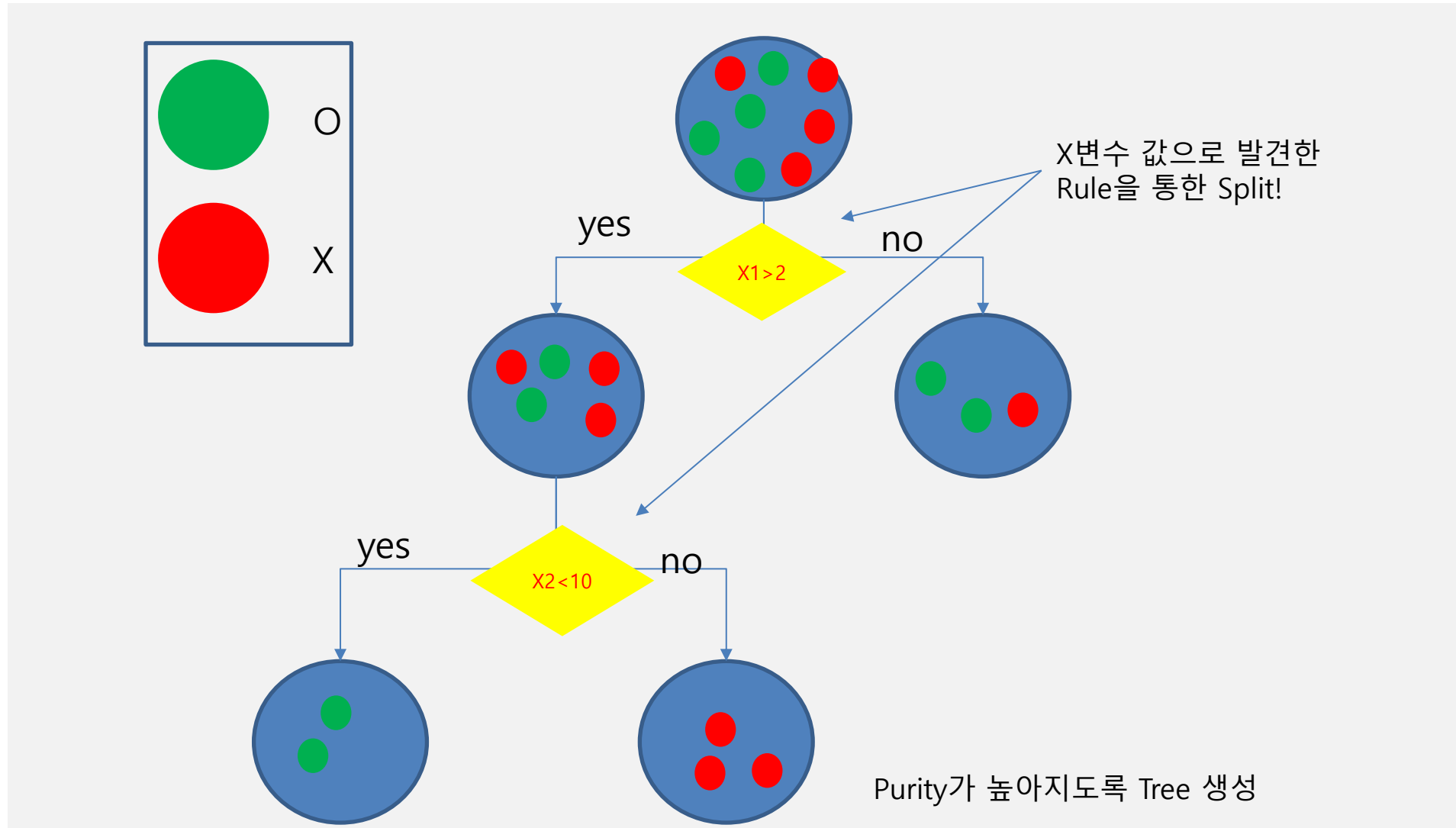
### 3. Decision Tree

Decision Tree의 구성요소: Root, Leaf Node



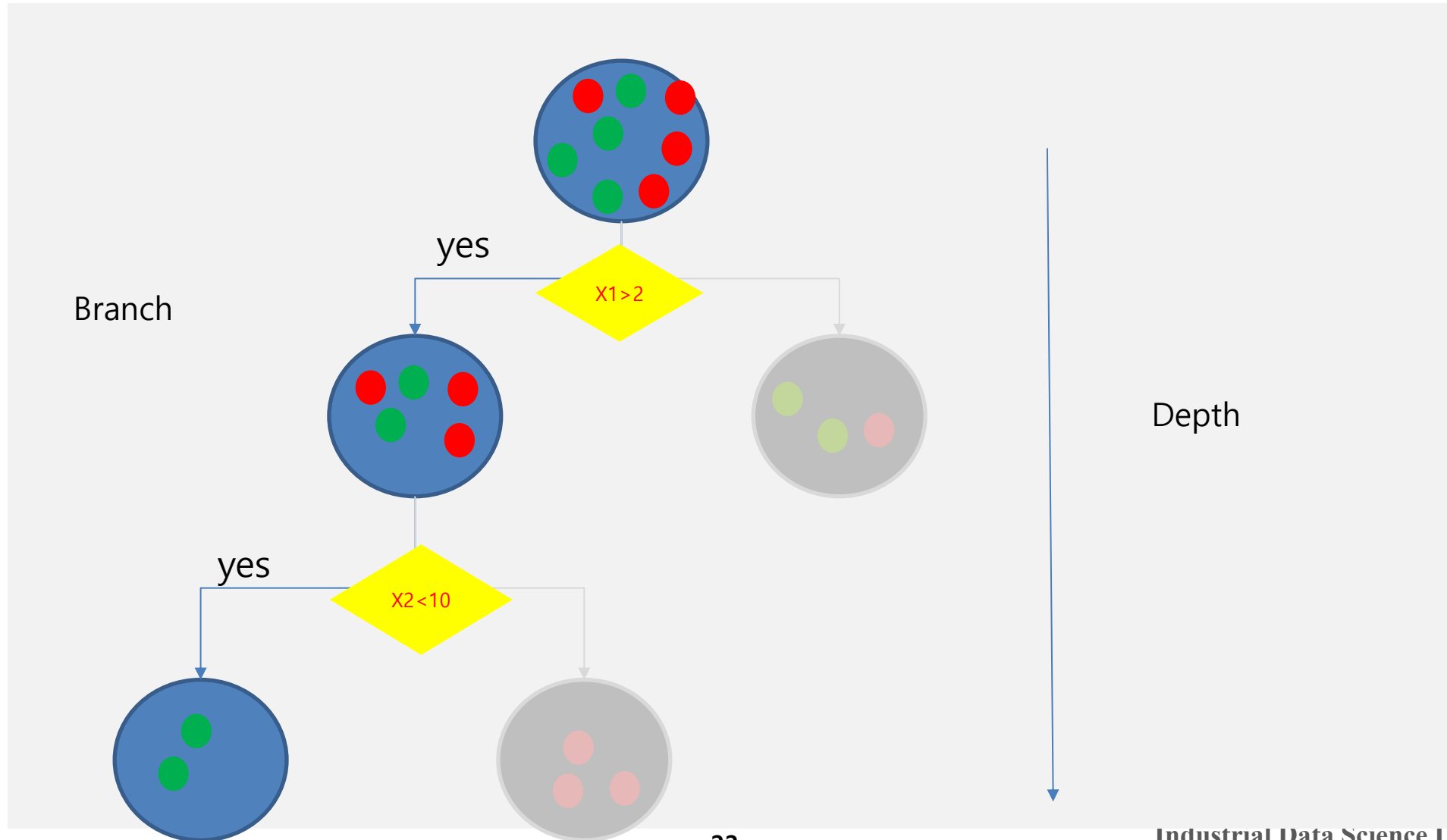
### 3. Decision Tree

Decision Tree의 구성요소: Rule



### 3. Decision Tree

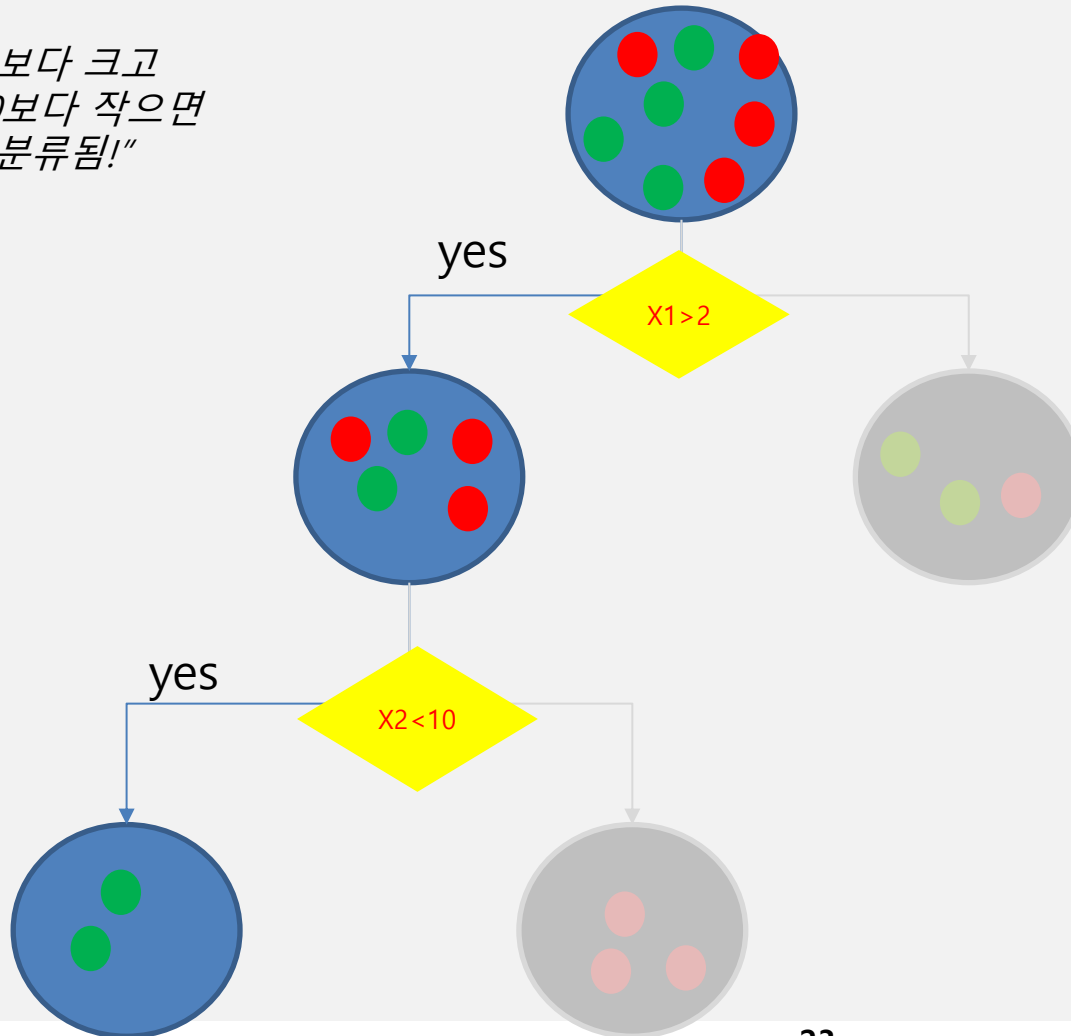
Decision Tree의 구성요소: Branch & Depth



### 3. Decision Tree

#### Decision Tree의 이해

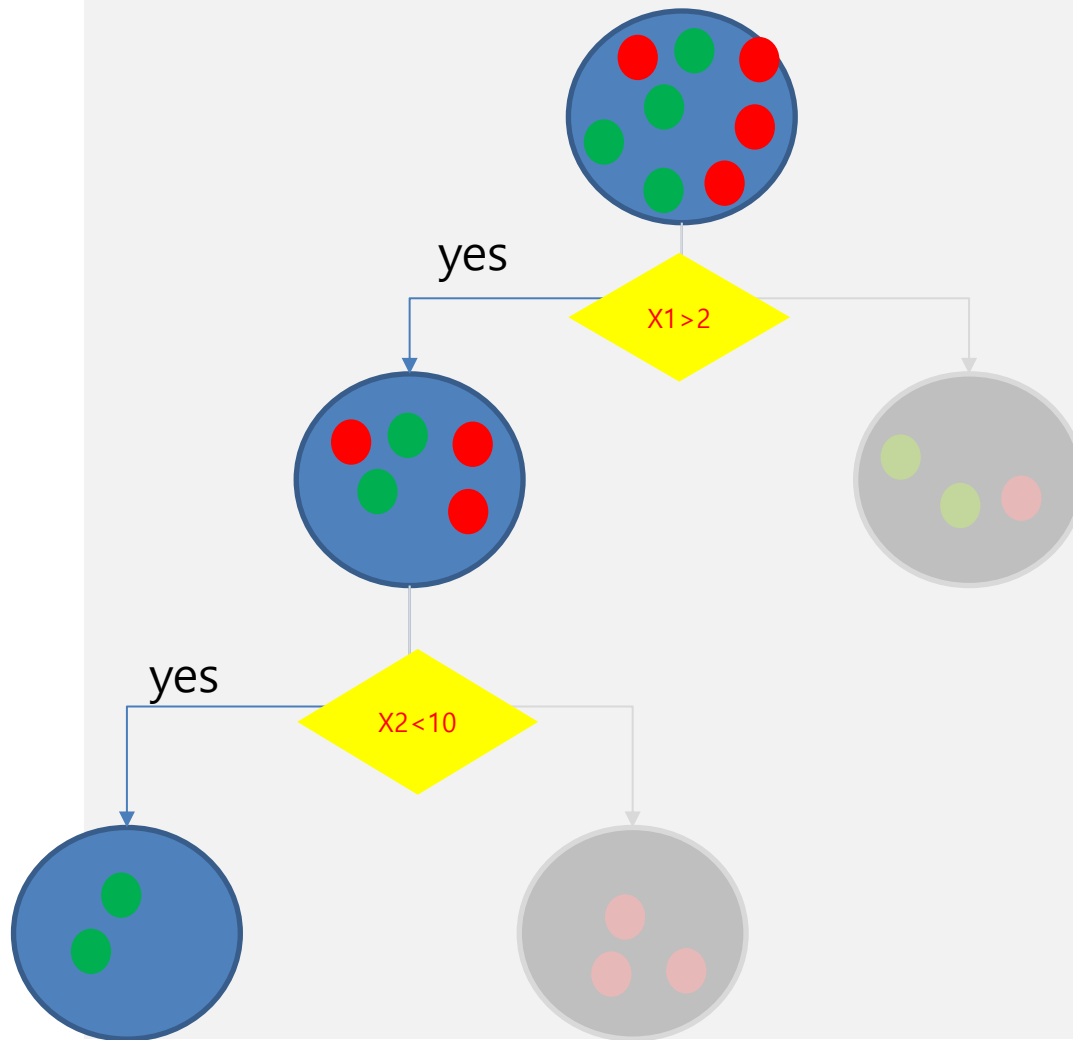
"X1이 2보다 크고  
X2가 10보다 작으면  
●으로 분류됨!"



- Rule을 통한 Split의 이해
- 여러 Rule들을 통한 변수들의 interaction 이해
- Decision Tree의 출력!

### 3. Decision Tree

#### Decision Tree의 이해



- Root
  - $X1 > 2$ 
    - $X2 < 10$ 
      - ●으로 분류
    - $X2 \geq 10$ 
      - ●으로 분류
  - $X1 \leq 2$ 
    - .....



## 4. k-NN과 Naïve Bayes

### ➤ 베이즈 정리

- 확률변수의 조건부(conditional) 확률분포와 주변부(marginal) 확률분포를 연관 짓는 정리. 즉, 새로운 자료에서 나온 확률에 기반하여 과거의 확률을 향상(update).

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

### ➤ 베이즈 정리 이용 예

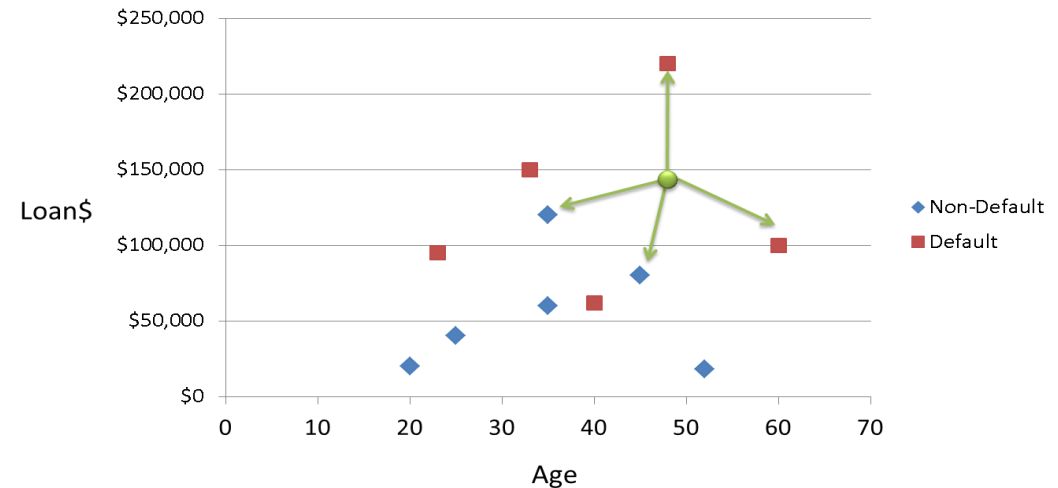
- 어떤 공장에서 일반적으로 공장이 원활한 경우 95%의 양품을 생산하지만, 공장이 원활하게 운영이 되지 않는 경우 70%의 양품을 생산
- 공장 관리자는 생산되는 제품의 품질을 바탕으로 공장 운영이 원활한지를 모니터링 하고, 이를 공장 운영에 반영
- 정리
  - O: 공장이 원활하게 운영
  - OC: 공장이 원활하게 운영되지 않음
  - S: 양품 생산
  - SC:불량 생산
  - $P(S|O)$ 와  $P(S|OC)$ 를 알고 있음
- $P(O|S)=?$ 
  - 이 확률을 바로 구할 수 없으므로 베이즈 정리 이용
  - $P(O|S) = P(S|O)P(O) / (P(S|O)P(O)+P(S|OC)P(OC))$

## 4. k-NN과 Naïve Bayes

### ➤ KNN

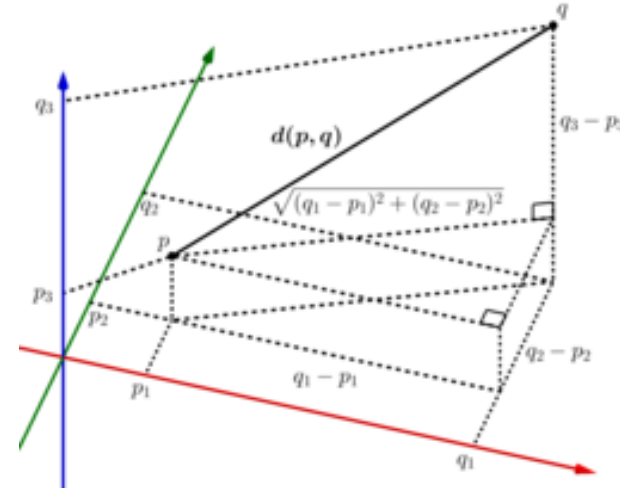
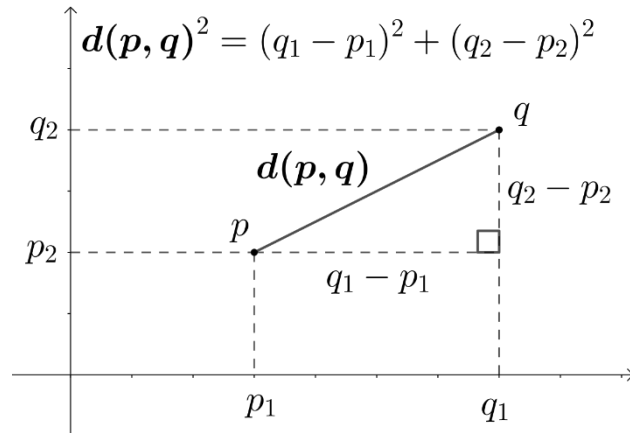
- 1970년대 시작되었으며, 비모수적 기법
- 모든 가능한 케이스를 저장하고, 새로운 케이스를 유사도 기반하여 분류
- 모든 케이스는  $n$ 차원의 공간에서 점과 대응되며, 유클리드 혹은 맨해튼 거리 관점에서 인접한 이웃이 정의됨
- 이산형 및 연속형 가능

- 여러 명칭들
  - K-Nearest Neighbors
  - Memory-Based Reasoning
  - Example-Based Reasoning
  - Instance-Based Learning
  - Case-Based Reasoning
  - Lazy Learning

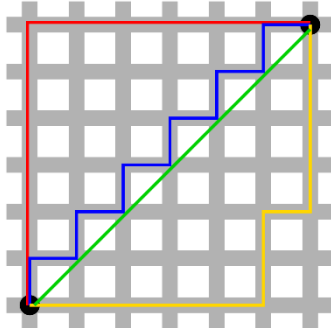


## 4. k-NN과 Naïve Bayes

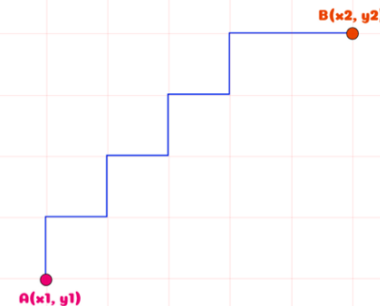
- 유클리드 거리와 맨해튼 거리 비교
  - 유클리드 거리(Euclidean Distance)



- 맨해튼 거리(Manhattan Distance)
  - 격자 모양의 경로에서 측정된 거리



$$\text{Manhattan}(A, B) = |x_1 - x_2| + |y_1 - y_2|$$



## 4. k-NN과 Naïve Bayes

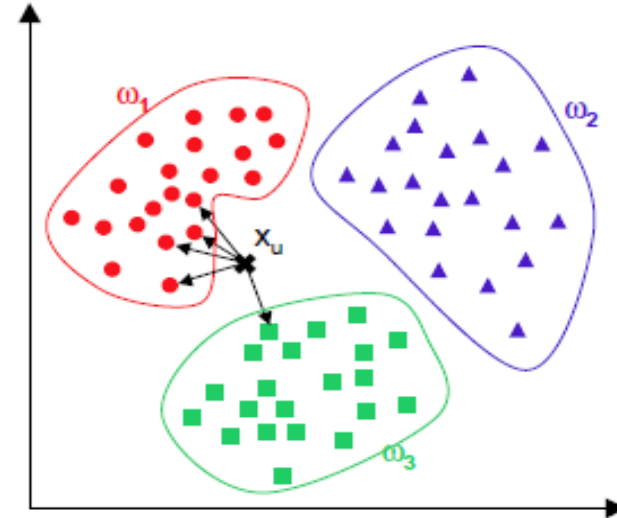
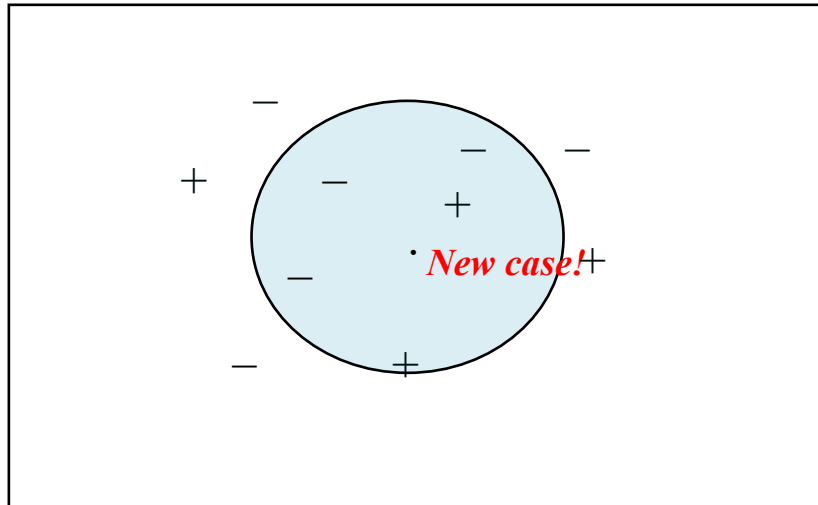
### ➤ KNN

- Bayes Classifier의 한 종류
- K-Nearest Neighbor
  - Bayes Rule? 주어진  $x$ 에 대한  $y$ 의 확률
    - *Classify observation to the class with largest probability*
  - 간단한 방법 & 우수한 성능
  - 최적의  $K$ 를 구하기
- **K?**
  - If  $K=1$ , select the nearest neighbor
  - If  $K>1$ ,
    - For classification select the most frequent neighbor.
    - For regression calculate the average of  $K$  neighbors.
  - $K$ 는 주로 홀수로 선택

## 4. k-NN과 Naïve Bayes

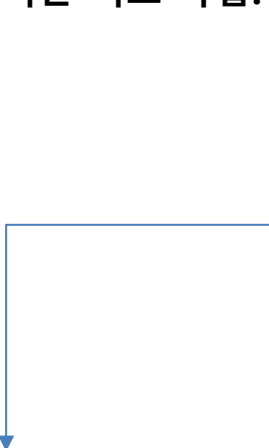
### ➤ KNN

- 연속형 값에 대한 k-NN은 k개의 인접 이웃의 평균 값을 반환
  - Distance-weighted nearest neighbor algorithm
  - 각 k개의 이웃이 새로운 케이스에 얼마나 기여하는지를 거리를 기반으로 산출
- Robust to noisy data by averaging k-nearest neighbors



## 4. k-NN과 Naïve Bayes

- Naïve Bayes 분류기
  - **베이즈 정리**
  - **conditional independence** assumption: Feature끼리는 서로 독립!
  - 쉽고 빠른 학습이 가능
  - 이미 계산된 조건부 확률에 의해 예측이 이뤄짐
- Naïve Bayes: popular **generative** model
  - 비교적 성능이 우수
  - 앙상블 학습에서 Base 학습기로 잘 활용됨



두 사건  $A, B$ 에 대해  $P(A \text{ and } B) = P(A)P(B)$ 이면  $A$ 와  $B$ 는 서로 독립

## 4. k-NN과 Naïve Bayes

### ➤ Bayes classification

- Difficulty: learning the joint probability

$$P(C | \mathbf{X}) \propto P(\mathbf{X} | C)P(C) = P(X_1, \dots, X_n | C)P(C)$$

### ➤ Naïve Bayes classification

- 가정: **all input features are conditionally independent!** (이런 가정이 Naïve)
- MAP classification rule: for

$$P(X_1, \dots, X_n | C)$$

$$\begin{aligned} P(X_1, X_2, \dots, X_n | C) &= P(X_1 | X_2, \dots, X_n, C)P(X_2, \dots, X_n | C) \\ &= P(X_1 | C)P(X_2, \dots, X_n | C) \\ &= P(X_1 | C)P(X_2 | C) \cdots P(X_n | C) \end{aligned}$$

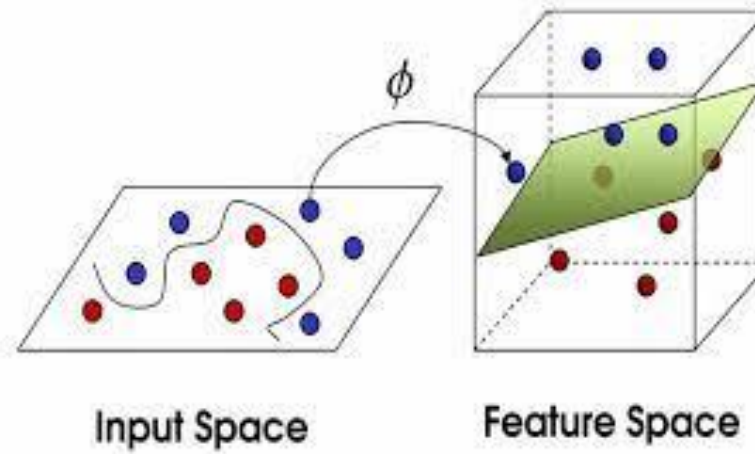
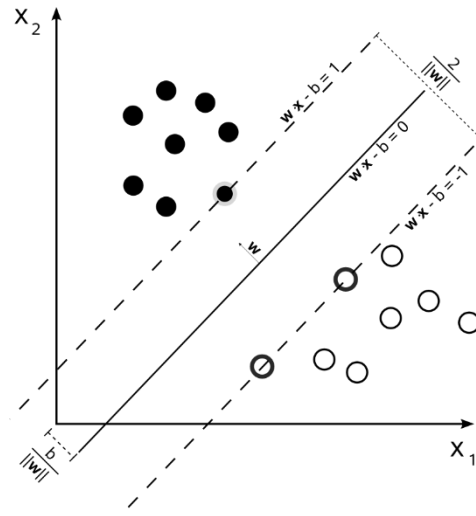
$$\mathbf{x} = (x_1, x_2, \dots, x_n)$$

$$[P(x_1 | c^*) \cdots P(x_n | c^*)]P(c^*) > [P(x_1 | c) \cdots P(x_n | c)]P(c), \quad c \neq c^*, c = c_1, \dots, c_L$$

## 5. SVM

### ➤ Support Vector Machine

- 1990년대 개발
- *One of the best "out of the box" classifiers*
- Maximal Margin Classifier의 Generalization 모형





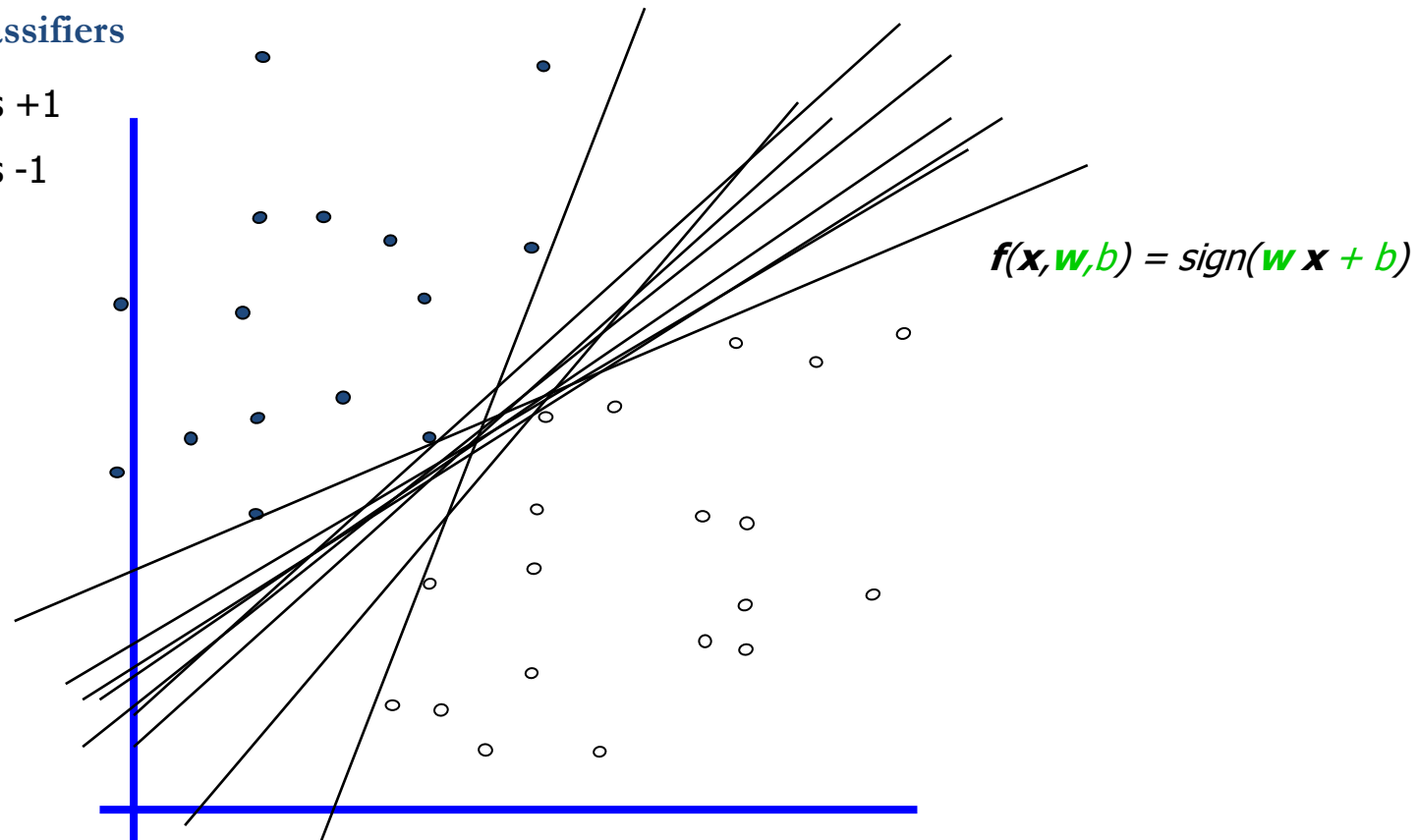
## 5. SVM

### ➤ Support Vector Machine

- 두 종류 점을 구분하는 방법을 고민
- 아래의 직선들로 가능하지만 다양한 가능성이 존재
- 이 중에서 최선의 직선을 찾는다면?

### Linear Classifiers

- denotes +1
- denotes -1

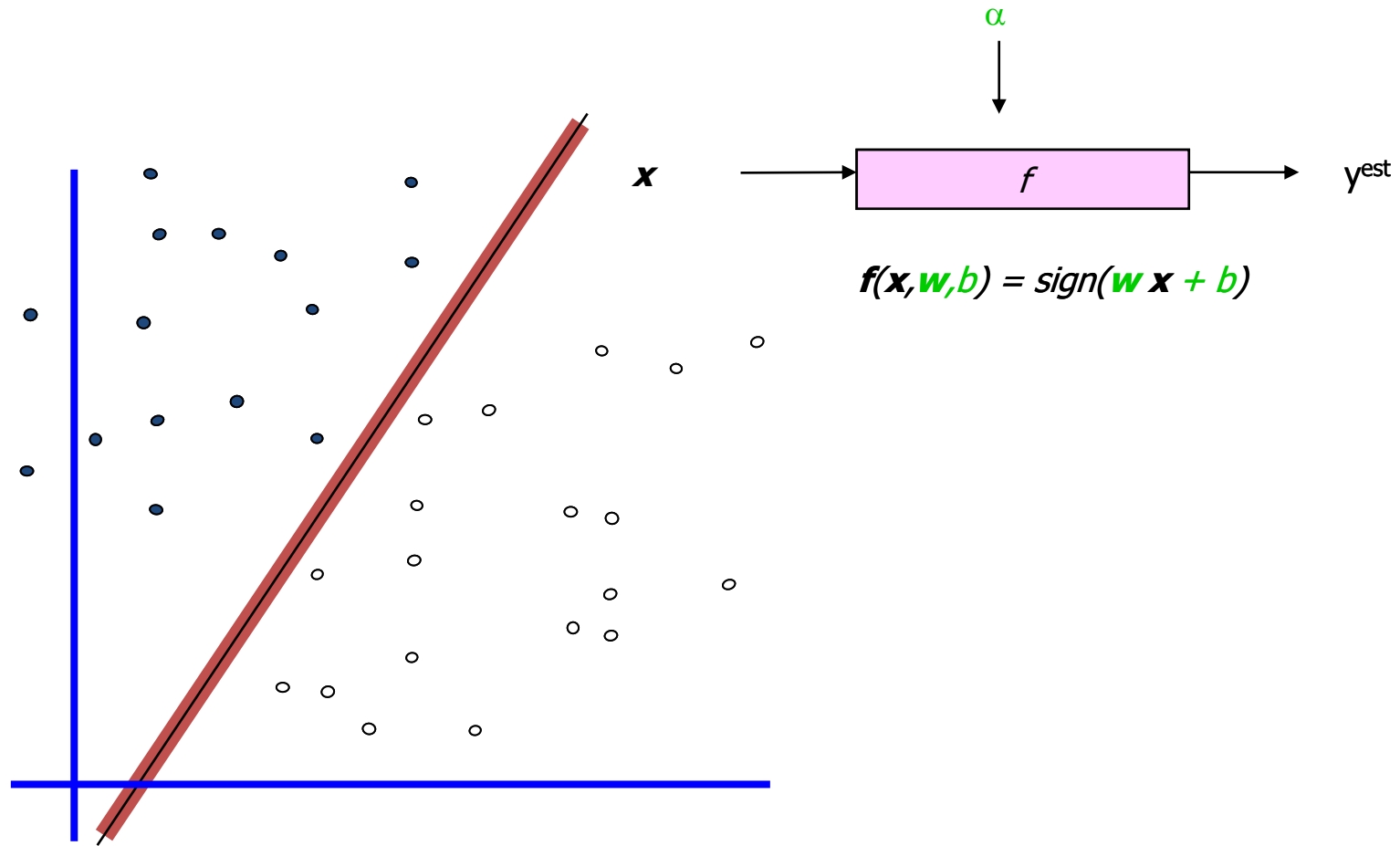


## 5. SVM

- **Margin:** 점에 닿기 전까지의 선형 분류기의 width

### Classifier Margin

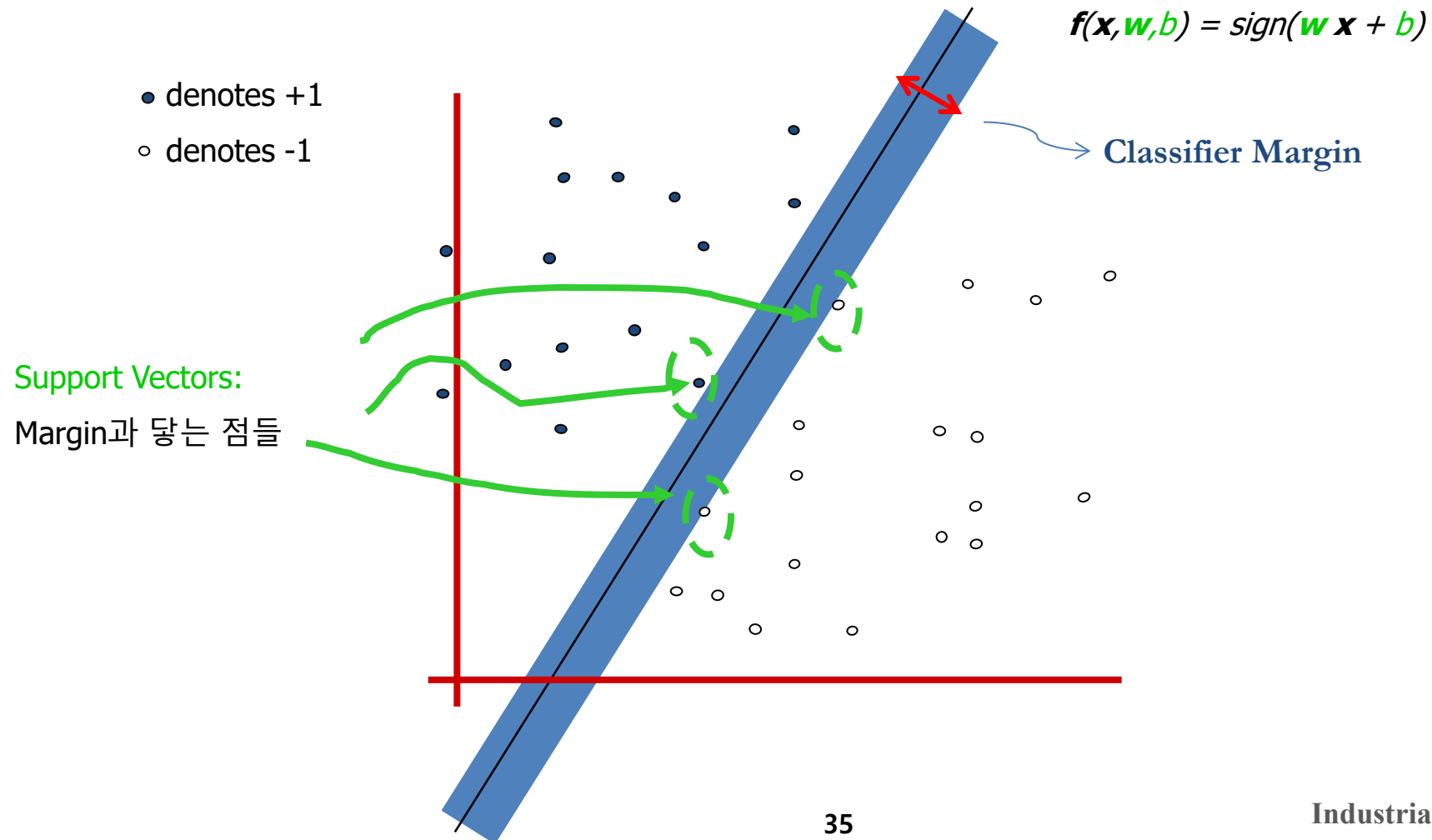
- denotes +1
- denotes -1



## 5. SVM

### ➤ Support Vector Machine

- ✓ Margin: 점에 닿기 전까지의 선형 분류기의 width
- ✓ Maximum margin linear classifier : 최대 margin을 갖는 선형 분류기, 특히 이것은 가장 단순한 형태의 SVM으로 Linear SVM이라 함

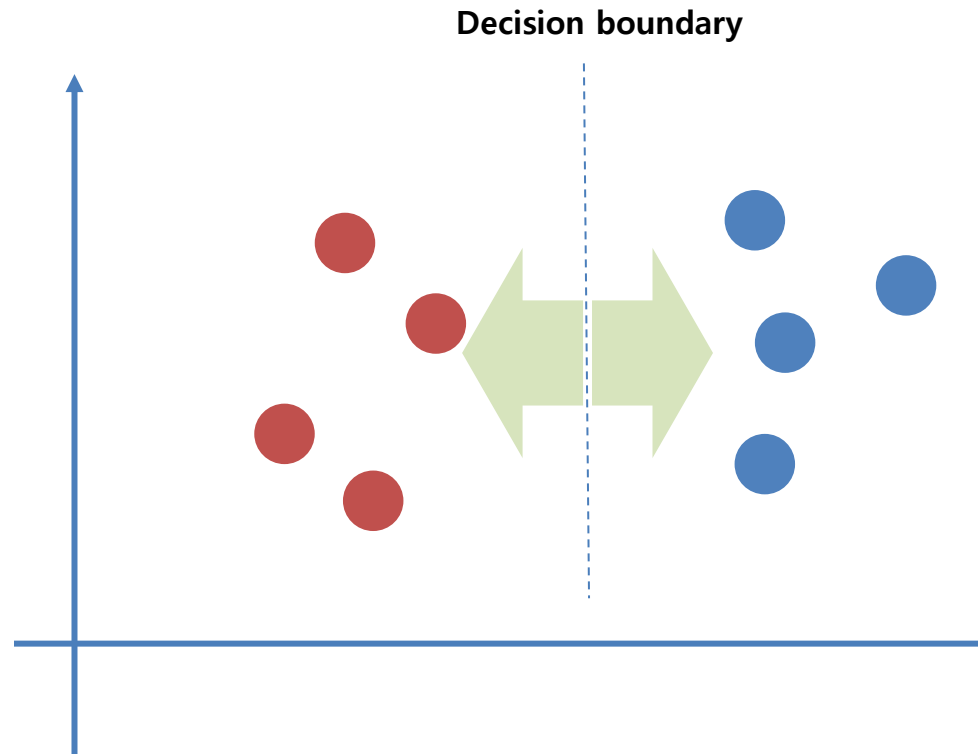


## 5. SVM

### ➤ Support Vector Machine

- 특징

- Margin의 최대화
- Robustness
- 성능 개선
- 이상치의 처리



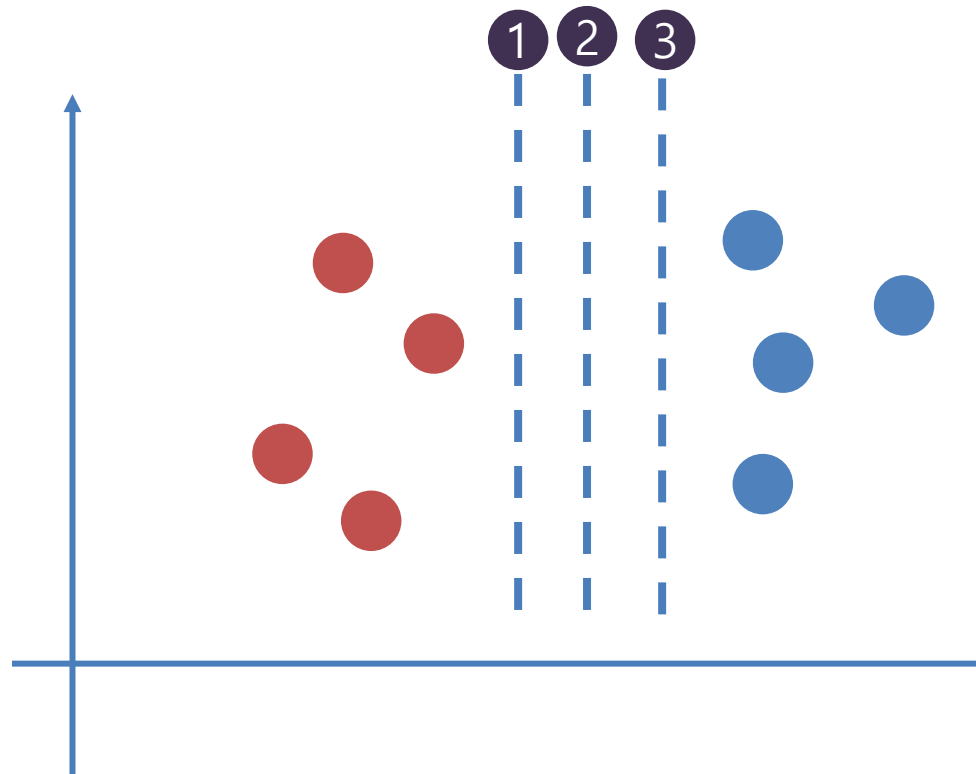
## 5. SVM

### ➤ Support Vector Machine

- 특징

- Margin의 최대화
- Robustness
- 성능 개선
- 이상치의 처리

- Decision boundary가 2인 경우 모형이 Robust
- Maximum Margin은 Robustness를 최대화



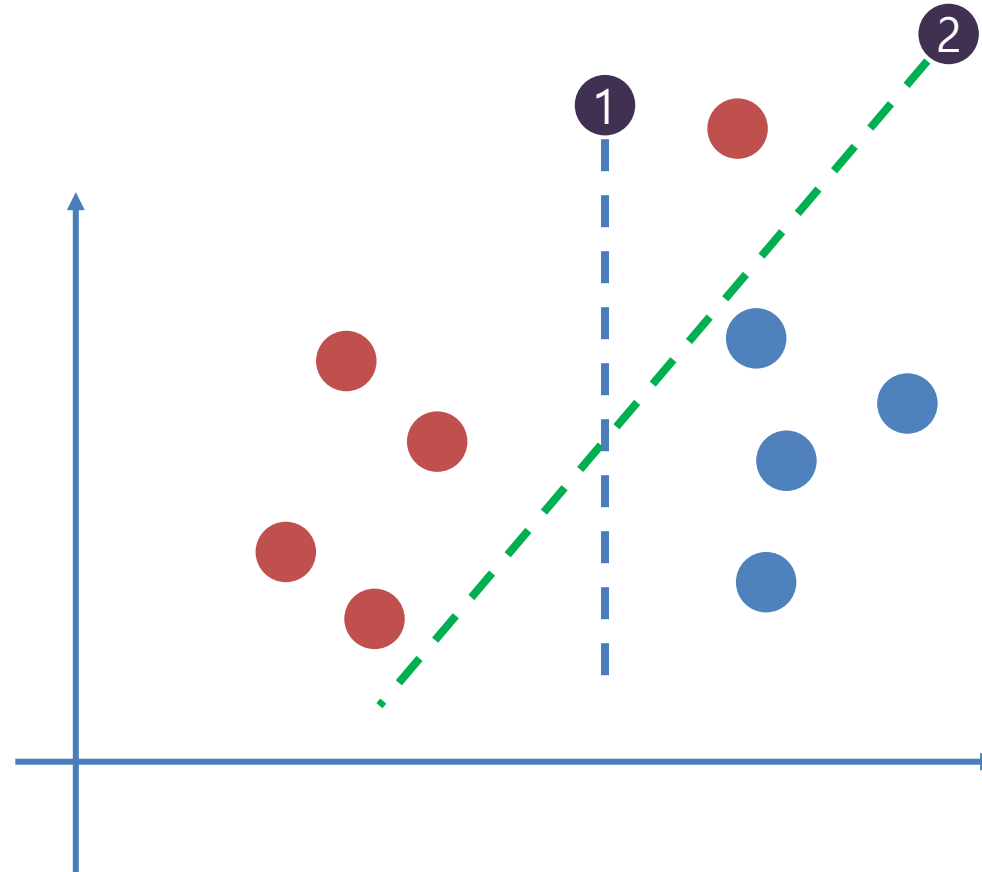
## 5. SVM

### ➤ Support Vector Machine

- 특징

- Margin의 최대화
- Robustness
- 성능 개선-보다 정확한 분류
- 이상치의 처리

- Maximum Margin 보다는 정확한 분류가 우선

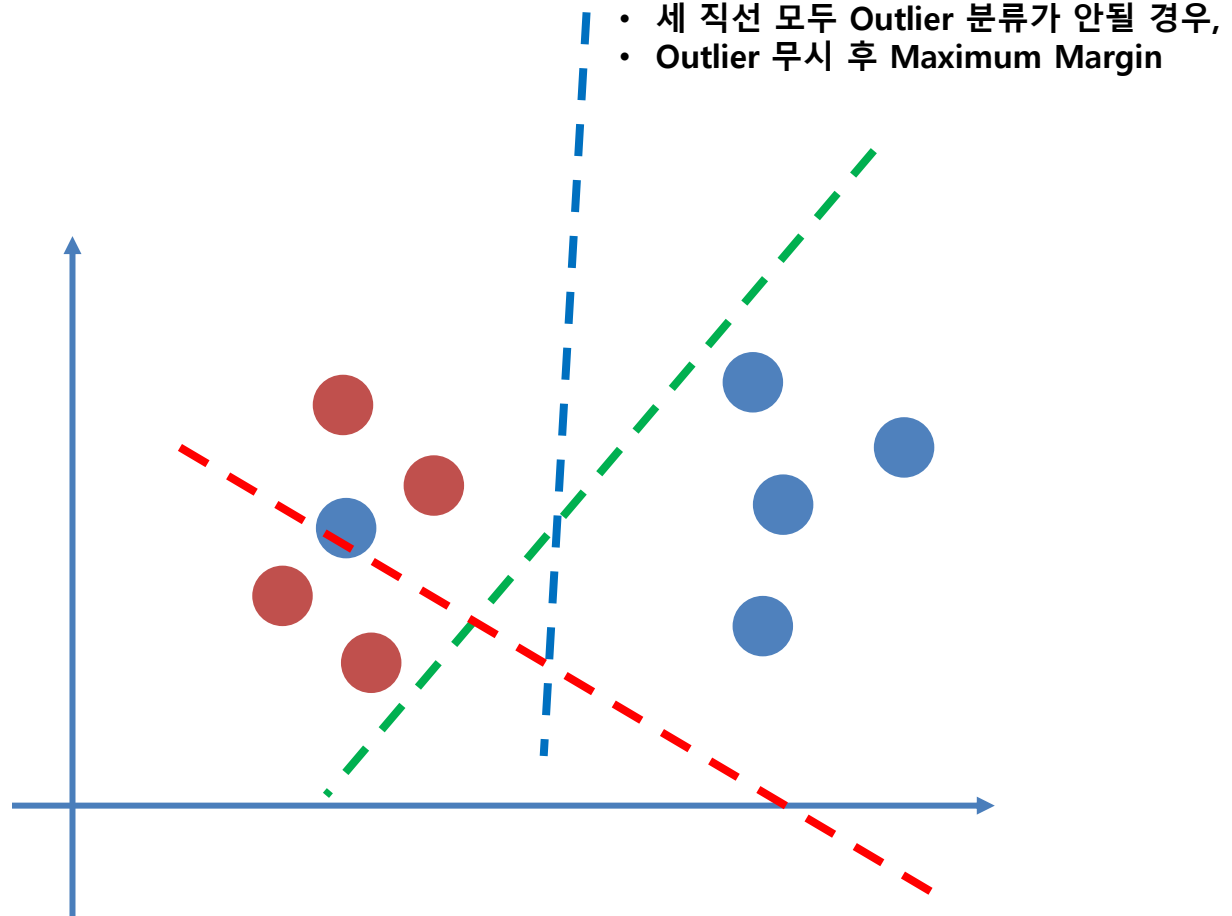


## 5. SVM

### ➤ Support Vector Machine

#### ▪ 특징

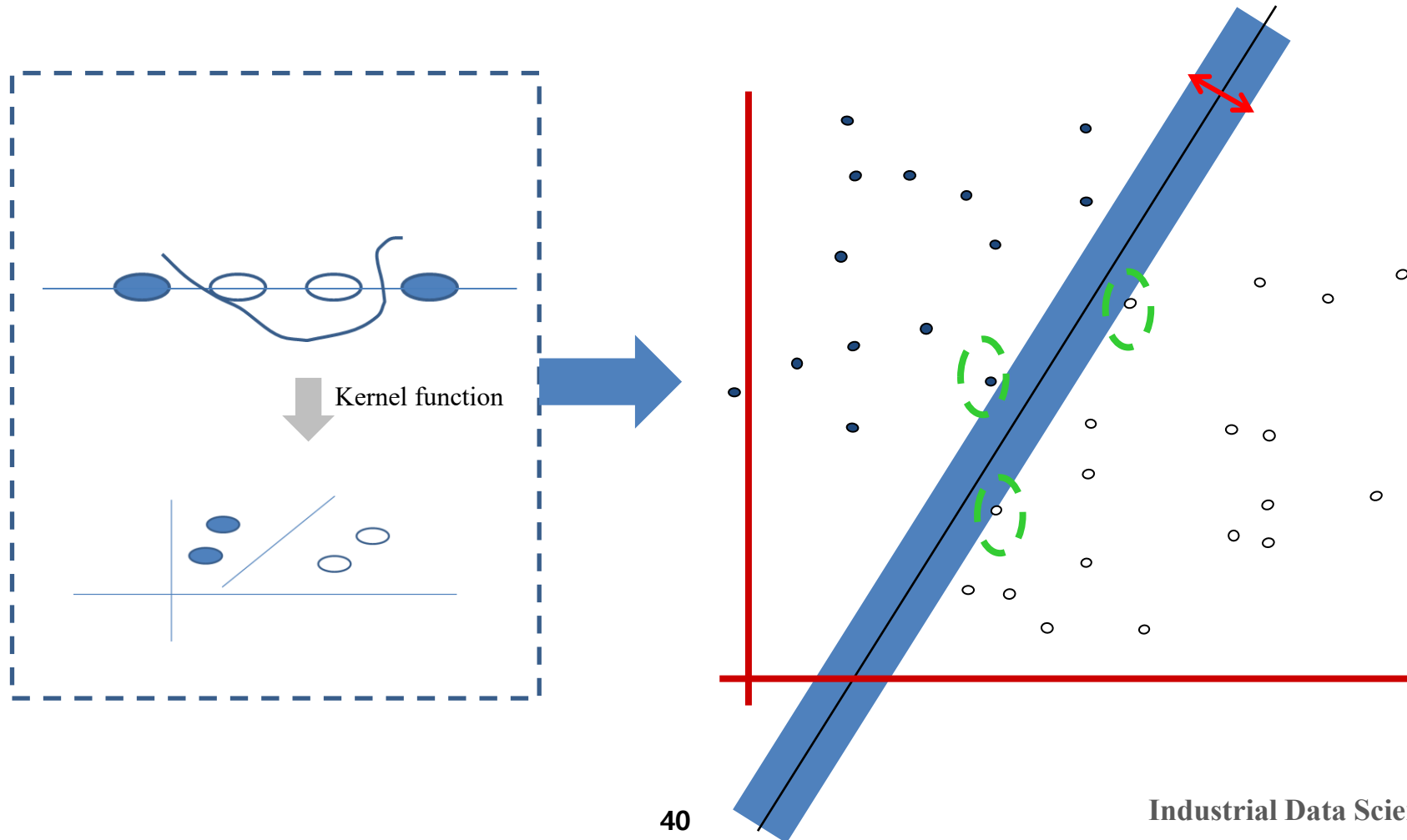
- Margin의 최대화
- Robustness
- 성능 개선
- 이상치의 처리



## 5. SVM

### ➤ Support Vector Machine

- ✓ **Kernel Trick:** 모든 데이터를 항상 초평면 또는 선으로 나눌 수 없으며, 이 경우 주어진 자료를 평면으로 표현할 수 있는 고 차원을 변환





## 5. SVM

### ➤ Support Vector Machine

- ✓ 저차원 공간(Low dimensional space)을 고차원 공간(High dimensional space)로 매핑

① X,Y 변수가 Not Separable!

② X,Y에 커널 트릭을 통해 변수 생성: Z, Q, R, ....

- Can be linearly separable!

③ 고차원 공간의 Linearly Separable Line

④ 저차원 공간으로 다시 매핑

- Non linear separable Line

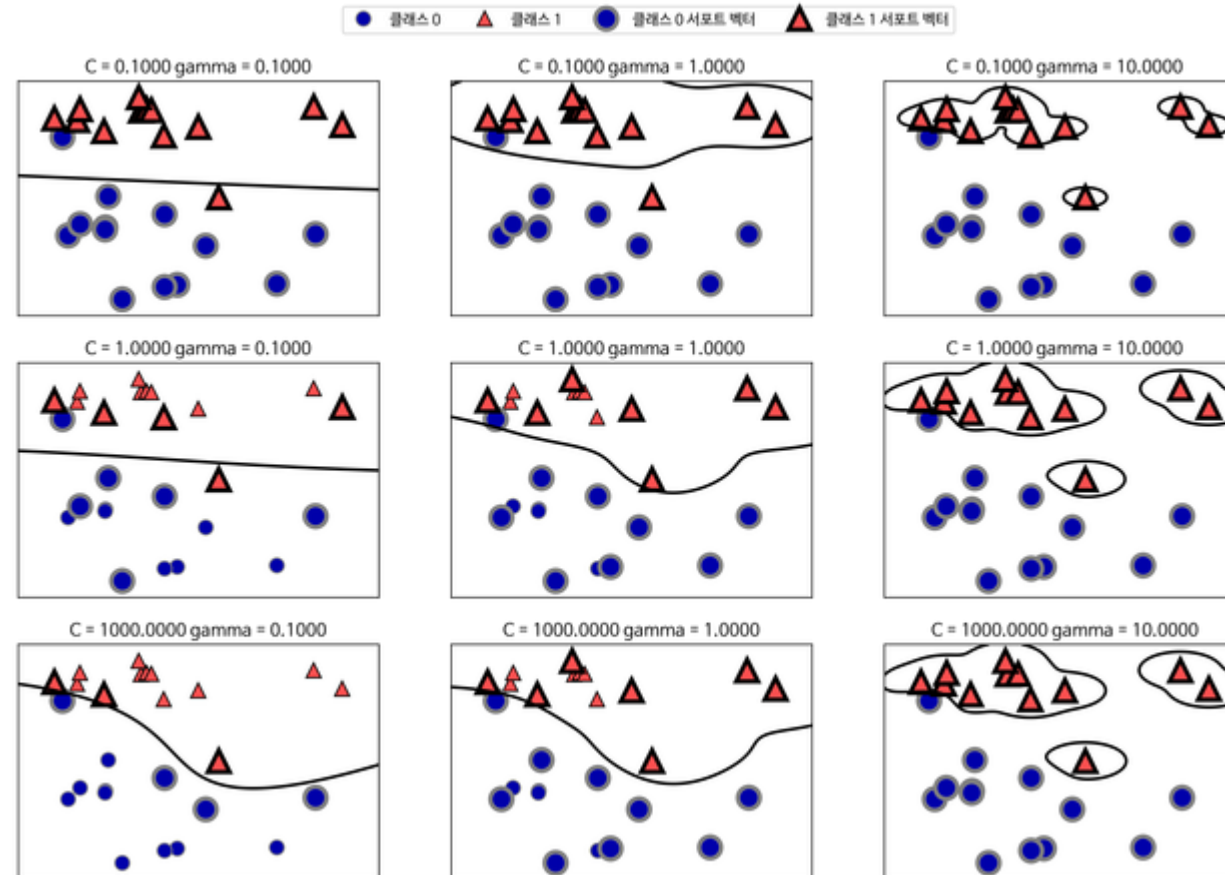
## 5. SVM

### ➤ Support Vector Machine in sklearn

- Kernel (Decision boundary의 모양)
  - Linear, polynomial, sigmoid, rbf
- C: Decision boundary의 Training points와 Smoothing 정도
  - C가 크면 더 많은 Training points : 곡선, 굴곡
  - C가 작으면 더 Smoothing : 직선
- Gamma: Decision Boundary의 Training point에 영향을 주는 데이터의 범위, 즉 reach를 의미
  - Gamma가 크다!
    - Reach가 작다 -> Decision Boundary 인근의 Training point가 영향, 개별 Training point의 영향->굴곡
  - Gamma가 작다!
    - Reach가 크다 -> 더 많은 데이터가 Training point로 사용 -> 직선 모양

## 5. SVM

### ➤ Support Vector Machine in sklearn



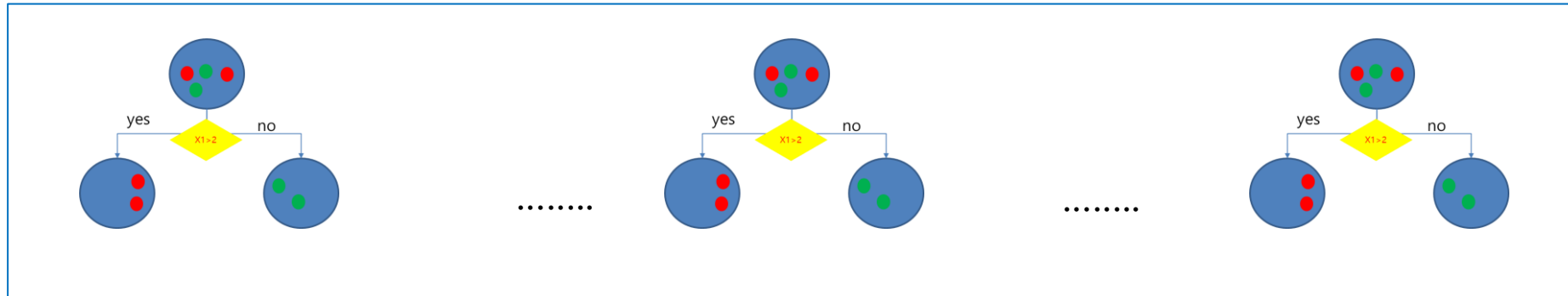
## 6. Random Forest

- Ensemble 기법: 여러 분류 모형의 결과를 결합하는 기법
- Random Forest: 앙상블 학습 방법의 일종으로, 훈련 과정에서 구성한 다수의 Decision Tree로부터 Voting을 통해 결과 예측
- Bagging: 주어진 데이터에서 랜덤하게 여러 개의 같은 크기의 부분집합을 생성
- Out of Bag과 Voting: Out of Bag(OOB)는 Bagging에서 제외되는 데이터들을 의미하며, Voting은 Random Forest내 여러 Decision Tree의 결과 중 다수의 결과를 선택하는 방법

## 6. Random Forest

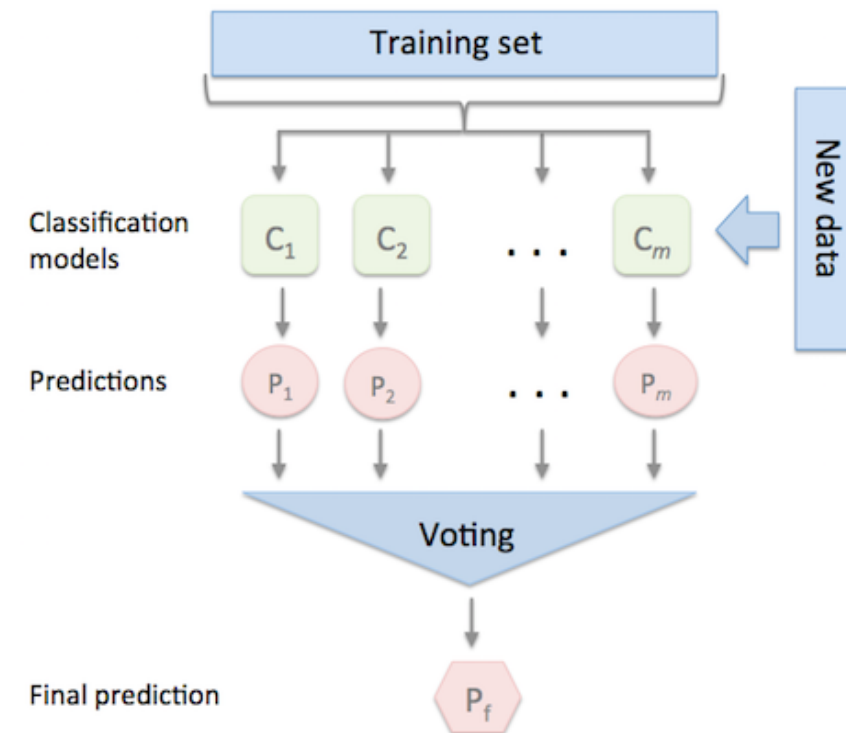
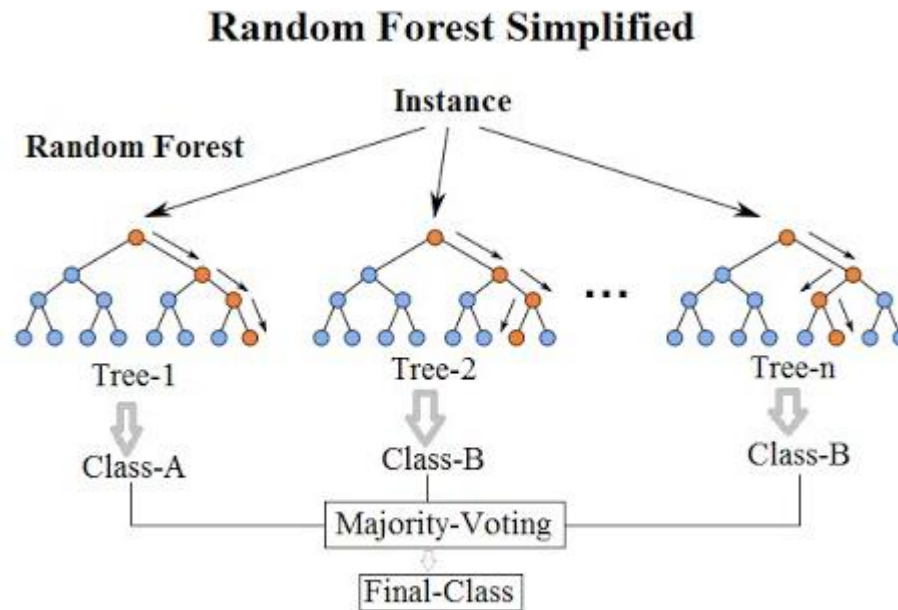
### Random Forest

- Breiman의 " bagging " 과 변수 랜덤 선택 아이디어 기반
- 처음에는 random decision forests로 시작하여 발전
- 데이터의 다양한 경우를 반영할 수 있도록 보완
- 다양한 경우에 대한 Decision Tree를 통해 성능과 안정성을 제고



## 6. Random Forest

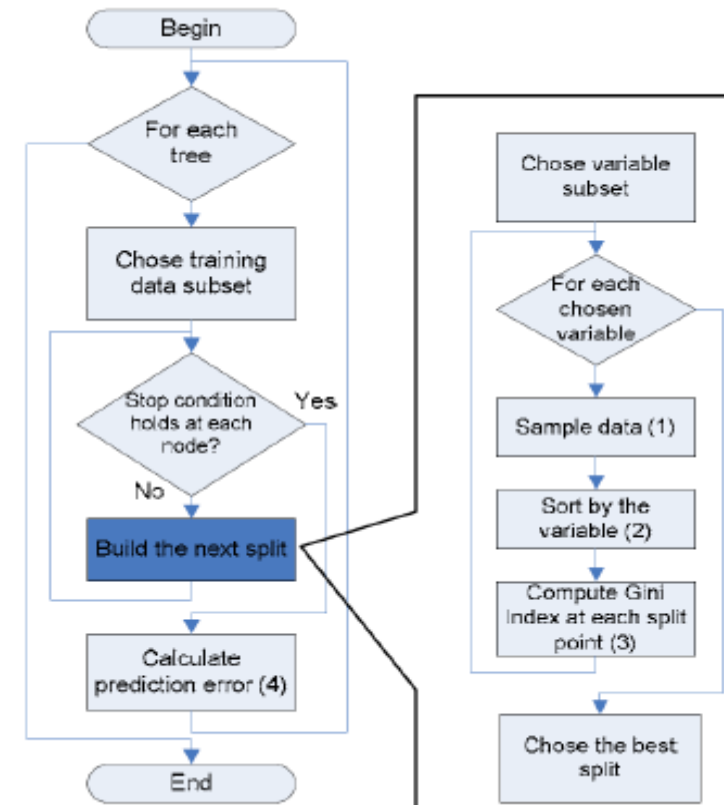
- **Random forest (or random forests)**
  - Ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees



## 6. Random Forest

- **Algorithm**

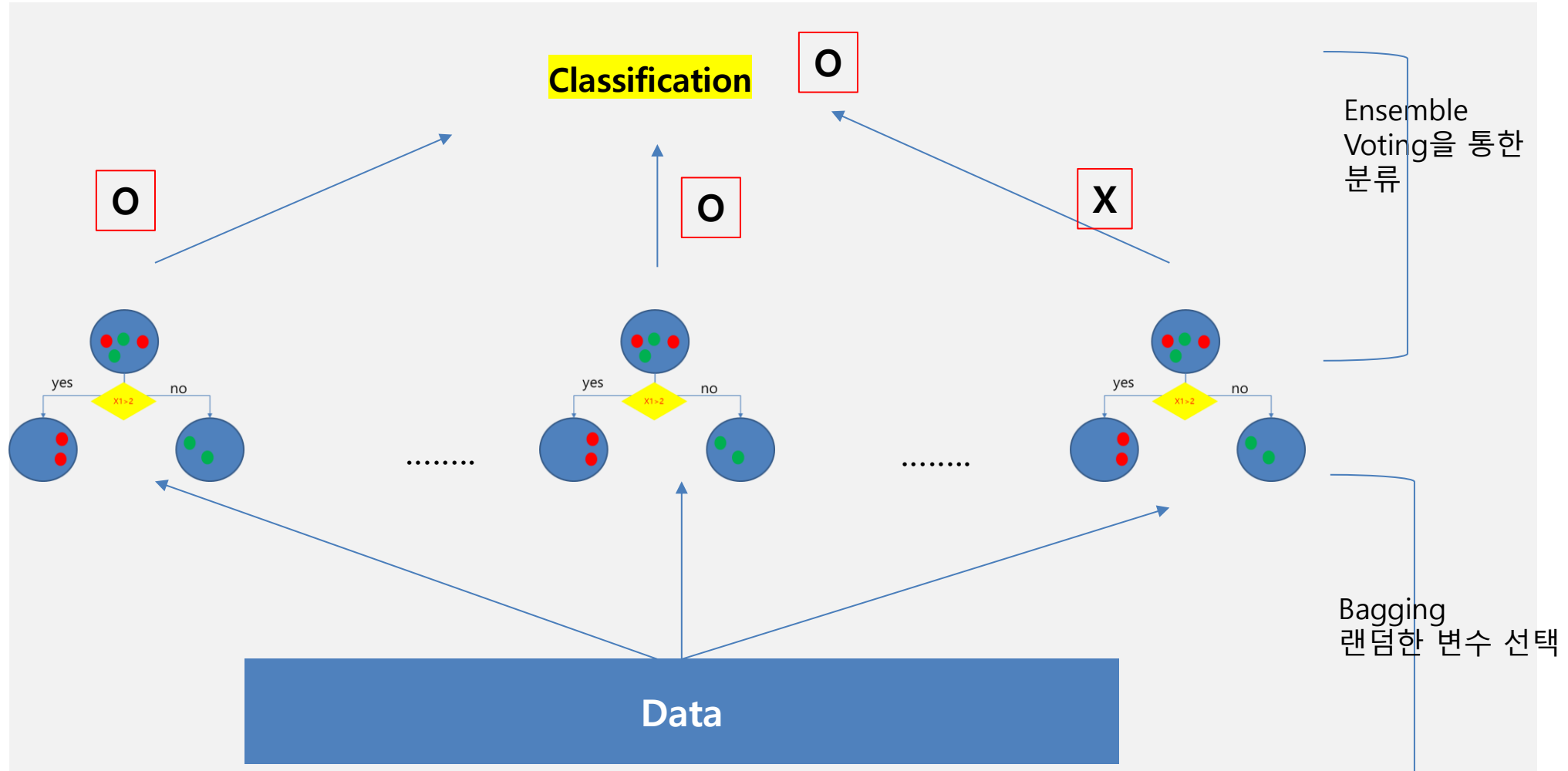
- ① N: # of training cases / M: 분류기의 변수
- ② M개 중 m개의 변수가 Tree의 각 노드에서 분류에 사용
- ③ N개의 training case 중에서 각 tree에 사용되는 n개의 case를 선택 (예: bootstrap sample). 선택되지 않은 Case는 error 추정에 사용
- ④ 각 tree의 각 노드에서, m개의 변수를 무작위 선택하여 분류에 사용. 이후 m개의 변수로 가장 분류를 잘하도록 계산
- ⑤ 각 Tree **fully grown and not pruned**



## 6. Random Forest

### *Random Forest*

- 데이터의 다양한 경우를 반영할 수 있도록 보완
- 안정성을 제고

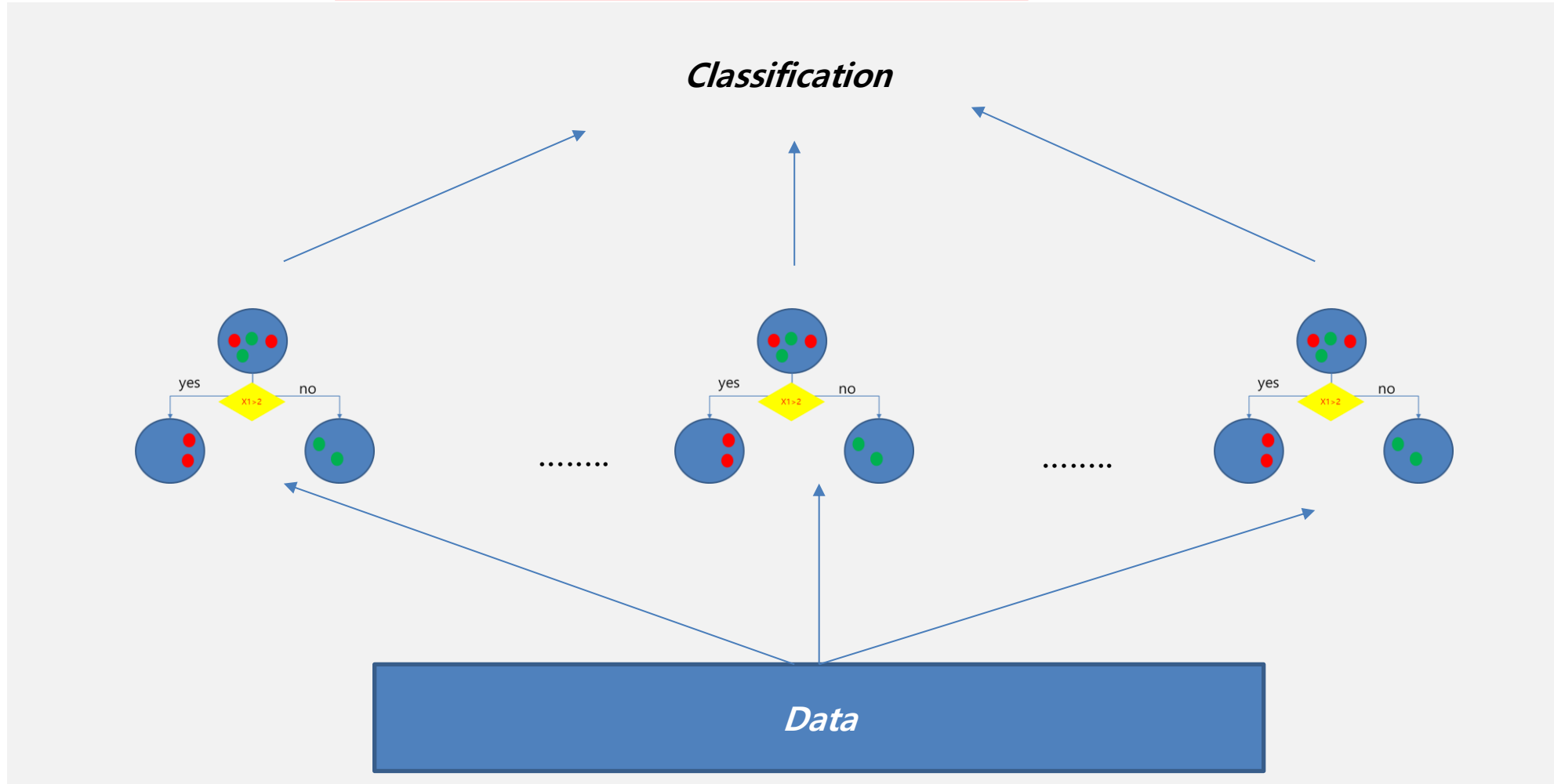




## 6. Random Forest

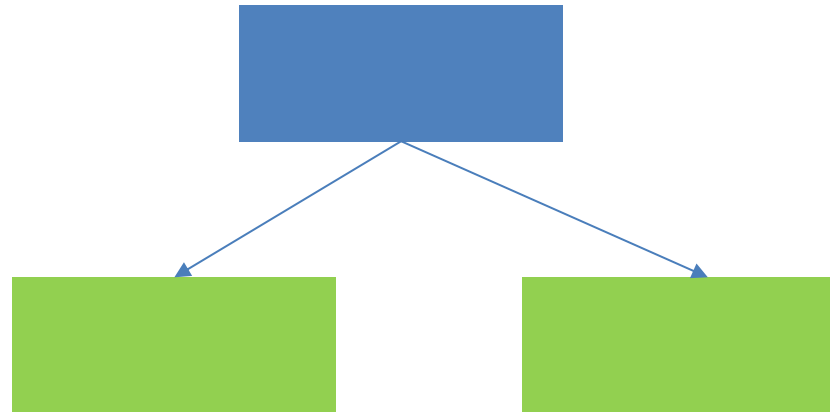
### Random Forest

- 몇 개의 Decision Tree를 만들 것인지?
- 몇 개의 X변수를 Random하게 선택할 것인지?



## 7. Adaboost

- Adaboost는 Ensemble 기법의 Boosting을 DT에 적용
- Stump로 부터 학습을 시작
  - Stump: 단순한 형태의 Tree, Weak learner

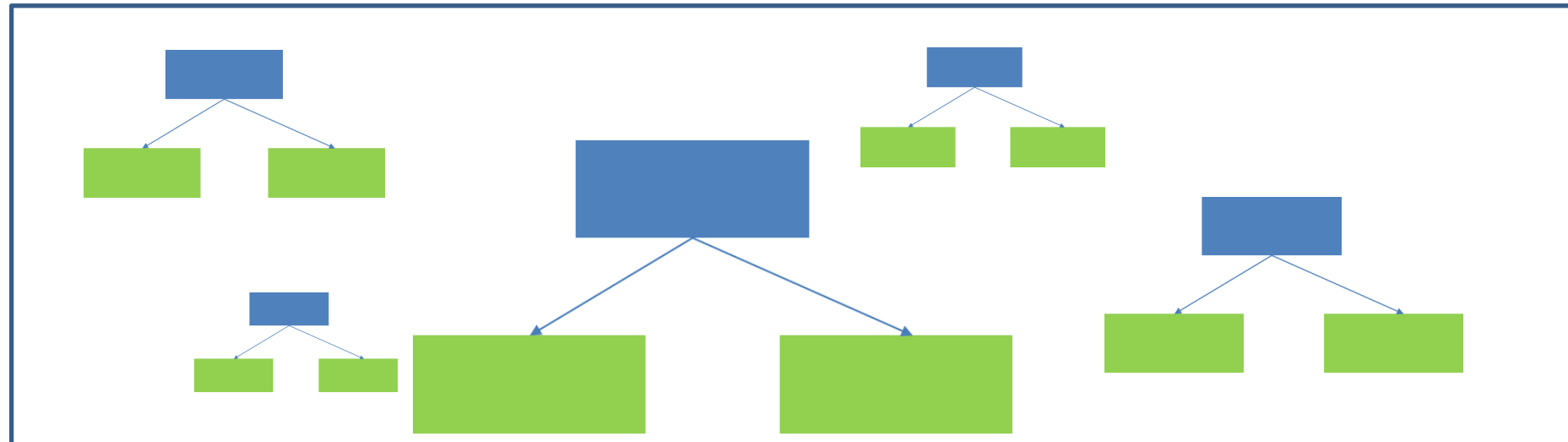


## 7. Adaboost

- Forest of stumps를 활용
  - Random Forest: 모든 tree는 같은 weight를 가짐
  - Adaboost: Stump마다 중요도의 차이가 존재
- Random Forest에서는 Tree가 같은 중요도를 지님

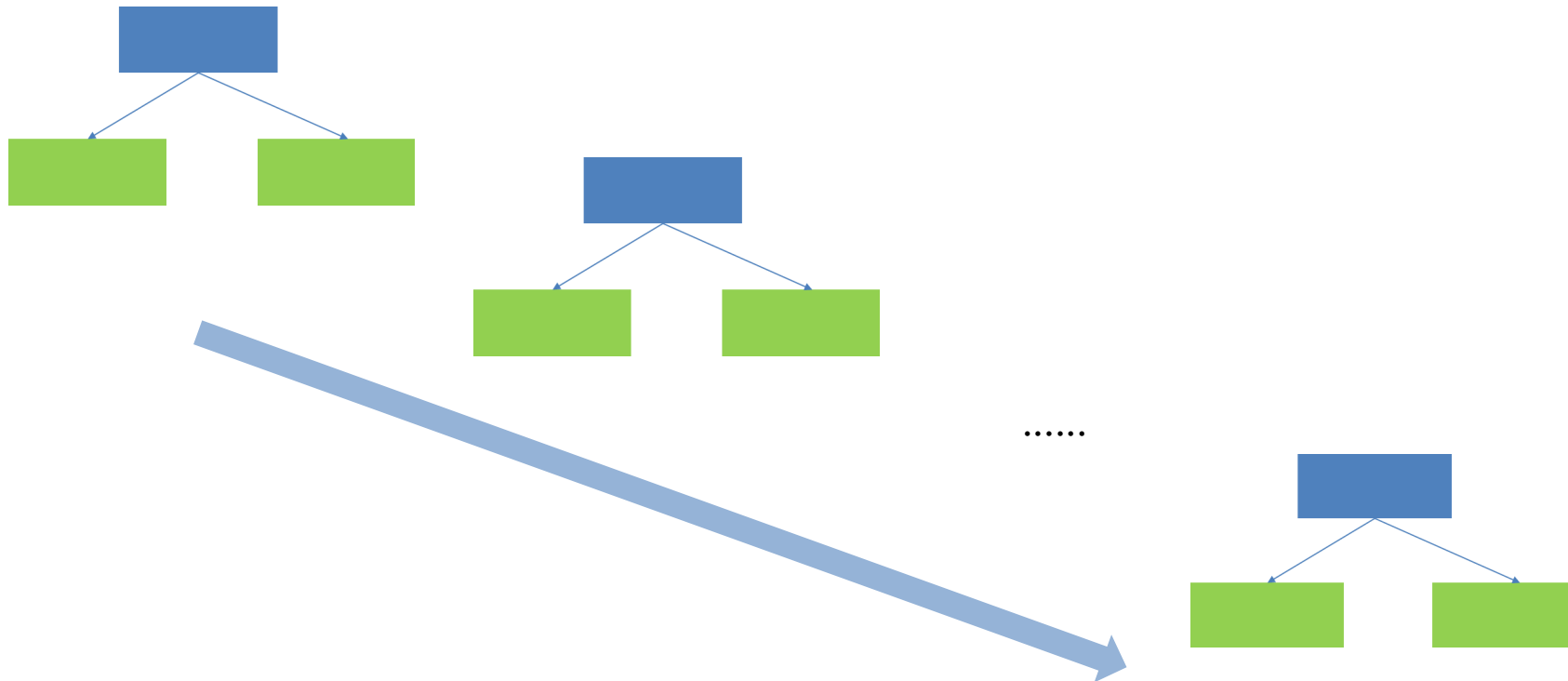


- Adaboost에서의 stump의 중요도: Amount of say로 표현, 클 수록 결과에 큰 영향을 미침



## 7. Adaboost

- Forest of stumps
  - 첫 stump는 다음 stump에 영향, 순차적으로 다음 stump에 영향을 주는 방식



## 7. Adaboost

- Example

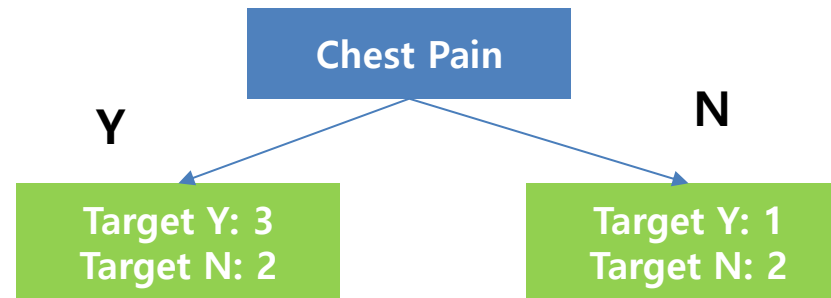
Target

Sample Weight의 합은 1

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

## 7. Adaboost

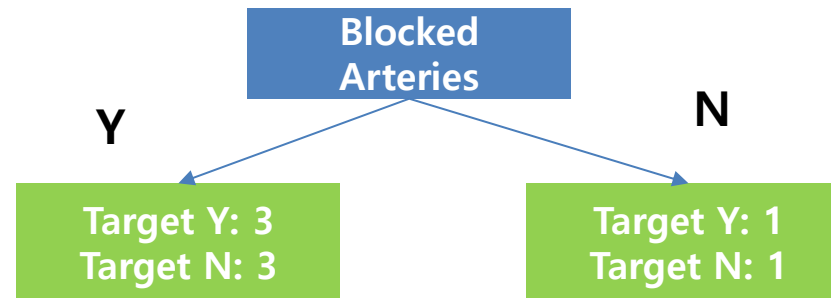
- 각 변수별 Target과의 관계



Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

## 7. Adaboost

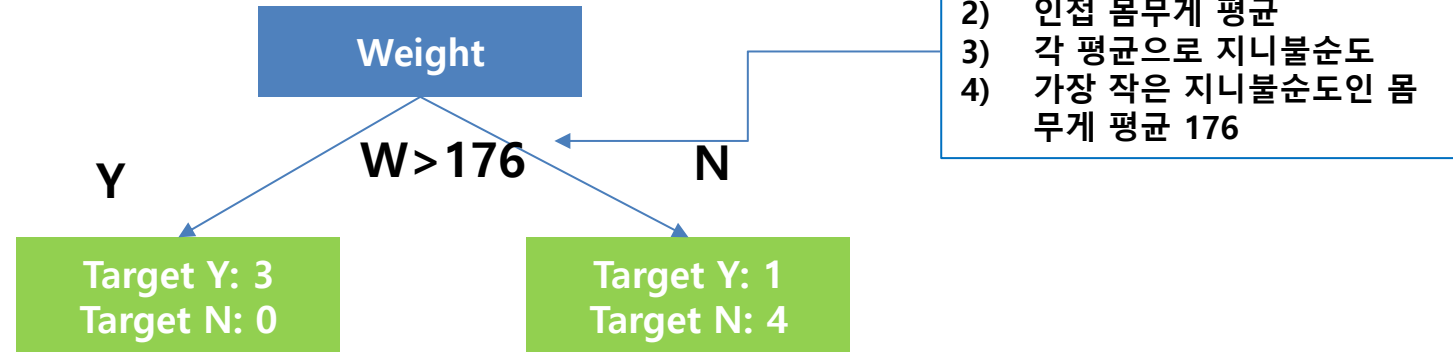
- 각 변수별 Target과의 관계



Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

## 7. Adaboost

- 각 변수별 Target과의 관계

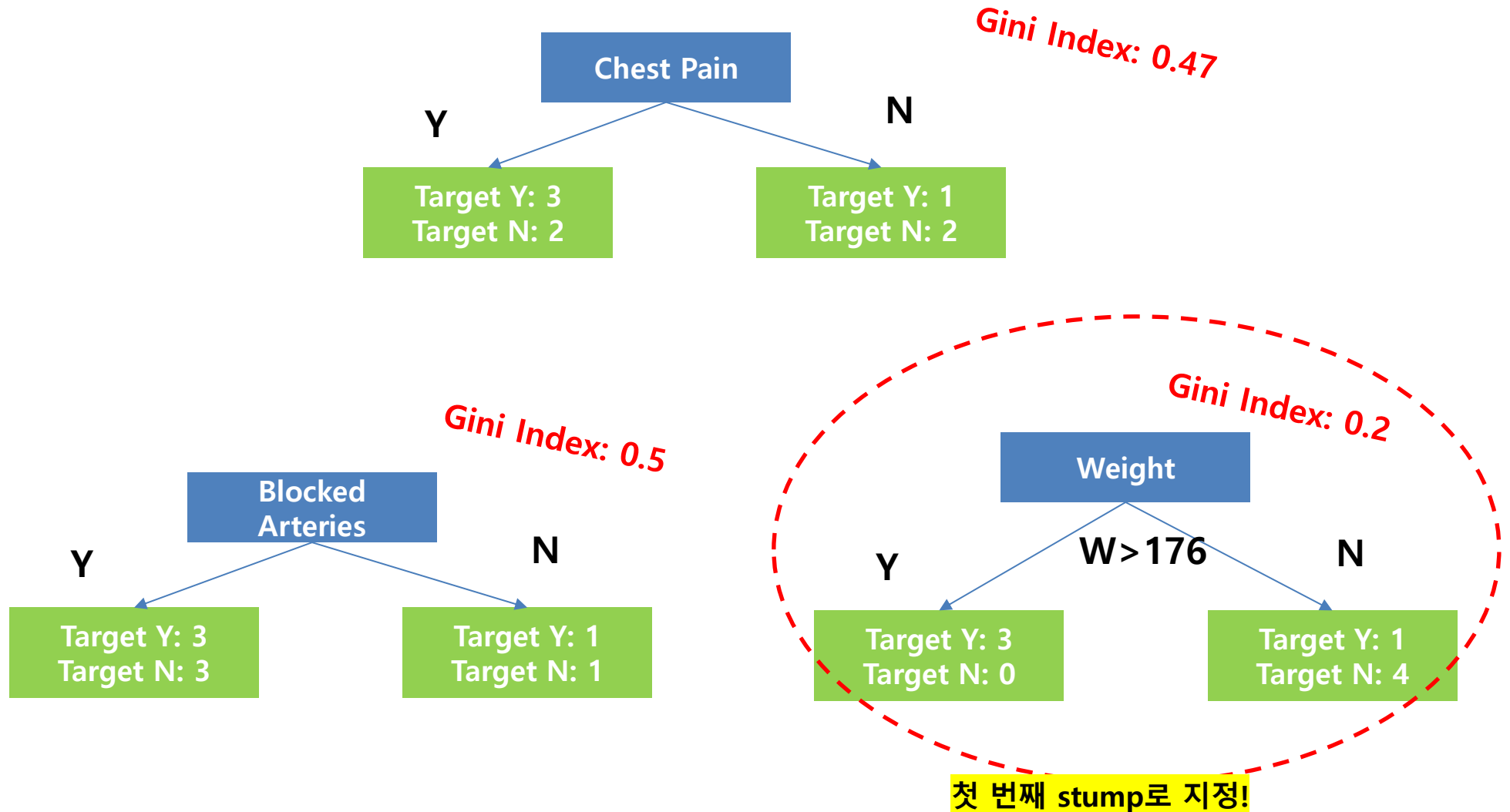


Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8



## 7. Adaboost

- 각 stump별 지니 계수



# 7. Adaboost

- 첫 stump의 Total Error



Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

Amount of Say = 0.97

$$\text{Amount of Say} = \frac{1}{2} \log \left( \frac{1 - \text{Total Error}}{\text{Total Error}} \right)$$

Amount of Say 계산

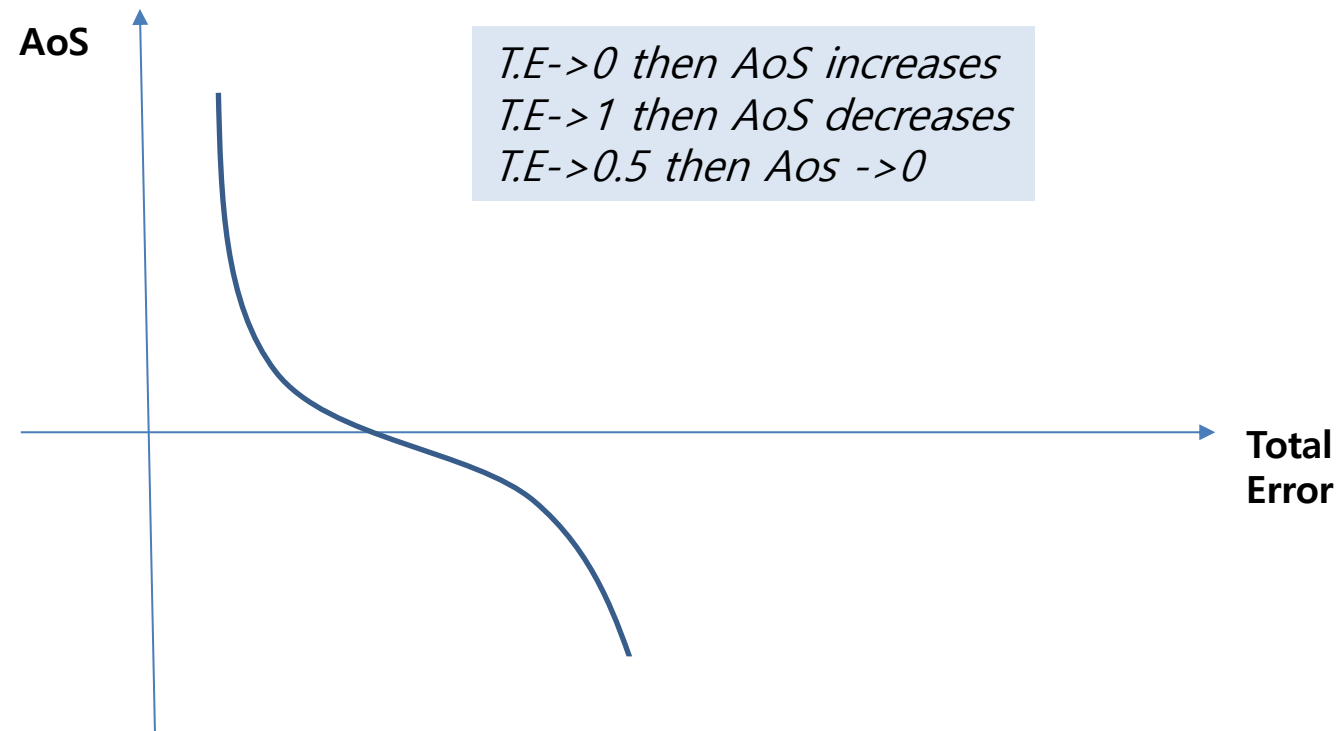
Total Error = 1/8  
(0~1사이 값)

첫 stump에서  
틀린 Case

## 7. Adaboost

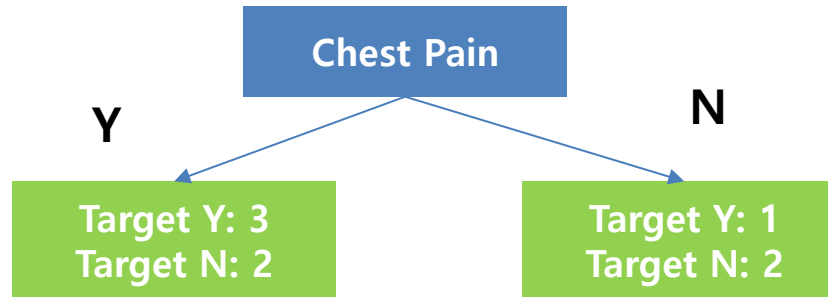
- 첫 stump의 Total Error

$$\text{Amount of Say} = \frac{1}{2} \log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$



## 7. Adaboost

- AoS 계산 예: Chest Pain (실제 stump는 아님)



Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

Amount of Say = 0.42

$$\text{Amount of Say} = \frac{1}{2} \log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

Amount of Say 계산

Total Error = 3/8  
(0~1사이 값)

*Chest Pain  
stump에서  
틀린 Case*

## 7. Adaboost

- 두 번째 Stump 계산

- 첫 Stump가 잘못 분류한 Sample의 Weight를 높여줌
  - 이후 원래 데이터에서 샘플링을 통해 새롭게 데이터 구성
  - Weight를 활용한 샘플링
- 첫 Stump에서 오분류된 Sample이 높은 Weight으로 더 많이 Sampling됨
  - 다음 Stump에서 이전 단계에 오분류된 Obs.에 집중

$$\text{New Sample Weight} = \text{Sample Weight} \times e^{\text{amount of say}}$$

AoS가 크면 Weight도 증가

- 첫 stump에서의 4번째 행

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

$$\text{New Sample Weight} = \frac{1}{8} \times e^{0.97} = 0.125 \times 2.64 = 0.33$$

## 7. Adaboost

- 두 번째 Stump 계산

- 첫 Stump에서 정분류된 Sample은 낮은 Weight으로 덜 Sampling
- 다음 Stump에서 이전 단계에 정분류된 Obs.는 덜 고려함

$$\text{New Sample Weight} = \text{Sample Weight} \times e^{-\text{amount of say}}$$

AoS가 크면 Weight는 감소

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Y	Y	205	Y	1/8
N	Y	180	Y	1/8
Y	N	210	Y	1/8
Y	Y	167	Y	1/8
N	Y	156	N	1/8
N	Y	125	N	1/8
Y	N	168	N	1/8
Y	Y	172	N	1/8

$$\text{New Sample Weight} = \frac{1}{8} \times e^{-0.97} = 0.125 \times 0.38 = 0.05$$

- Weight의 합이 1이 되도록 정규화

## 7. Adaboost

- 두 번째 Stump를 위한 Sampling

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight	New Weight	Sampling을 위한 값
Y	Y	205	Y	1/8	0.07	0~0.07
N	Y	180	Y	1/8	0.07	0.07~0.14
Y	N	210	Y	1/8	0.07	0.14~0.21
Y	Y	167	Y	1/8	0.49	0.21~0.7
N	Y	156	N	1/8	0.07	0.7~0.77
N	Y	125	N	1/8	0.07	0.77~0.84
Y	N	168	N	1/8	0.07	0.84~0.91
Y	Y	172	N	1/8	0.07	0.91~1

- 0~1사이 난수 생성
- 해당 난수 값이 속하는 행을 선택
- 중복해서 선택 가능
- Weight가 높은 행이 Sampling될 확률이 높음

# 7. Adaboost

- 다음 Stump를 위한 Sampling 및 학습 반복

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight	New Weight	Sampling을 위한 값
Y	Y	205	Y	1/8	0.07	0~0.07
N	Y	180	Y	1/8	0.07	0.07~0.14
Y	N	210	Y	1/8	0.07	0.14~0.21
Y	Y	167	Y	1/8	0.49	0.21~0.7
N	Y	156	N	1/8	0.07	0.7~0.77
N	Y	125	N	1/8	0.07	0.77~0.84
Y	N	168	N	1/8	0.07	0.84~0.91
Y	Y	172	N	1/8	0.07	0.91~0.98

- 다음 계산을 위해 가중치 초기화

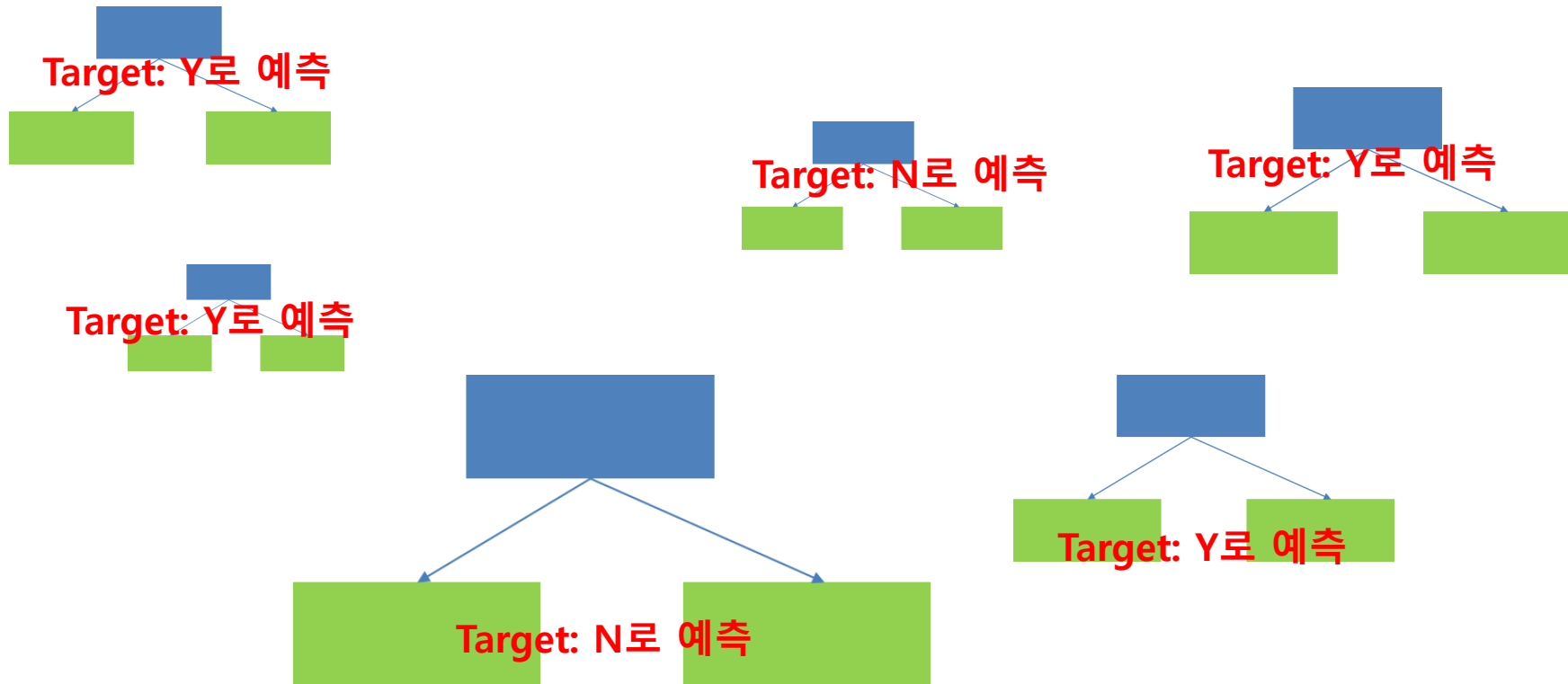
Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
N	Y	156	N	1/8
Y	Y	167	Y	1/8
N	Y	125	N	1/8
Y	Y	167	Y	1/8
Y	Y	167	Y	1/8
Y	Y	172	N	1/8
Y	Y	205	Y	1/8
Y	Y	167	Y	1/8



## 7. Adaboost

- Adaboost의 예측 방식

- 주어진 X값에 대해 Forest of Stump 내의 Stump에 적용

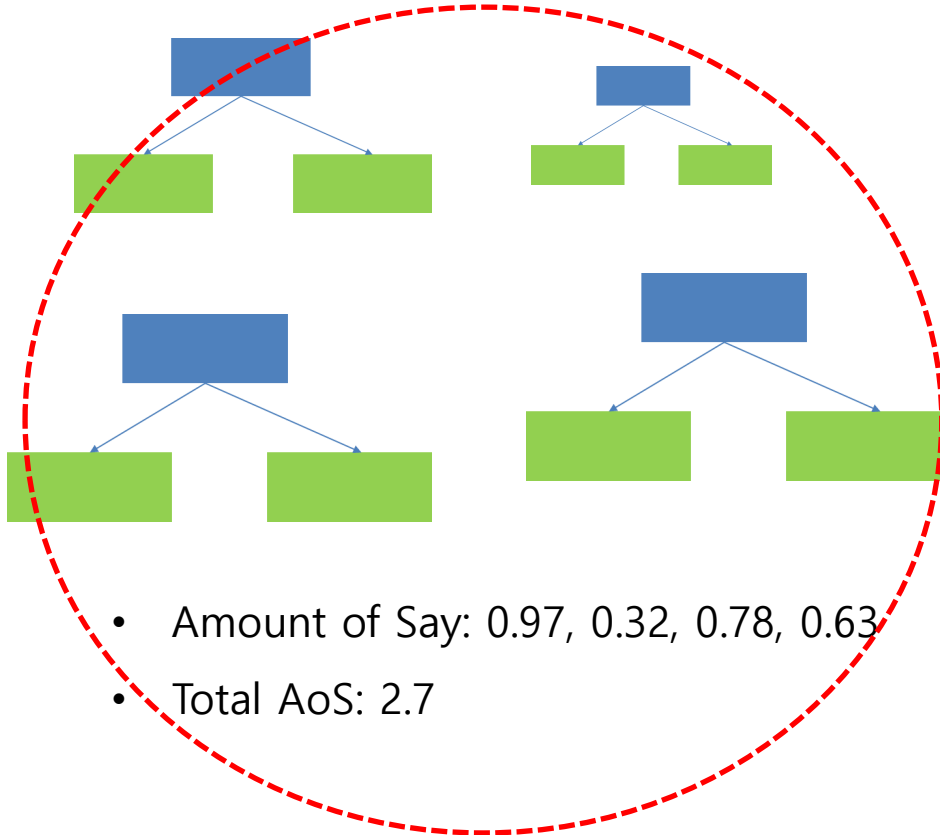


## 7. Adaboost

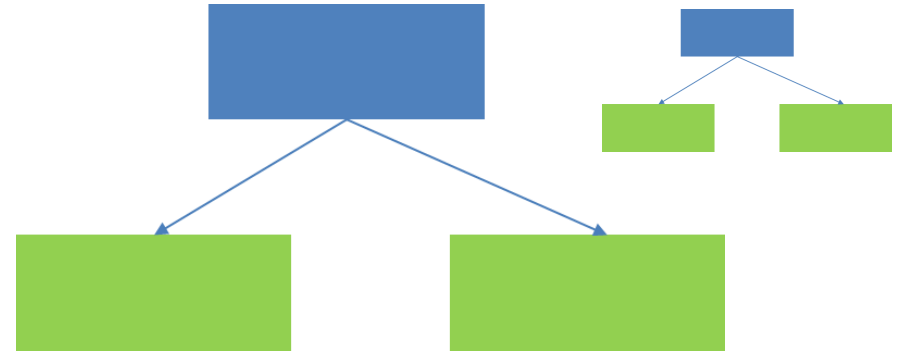
- Adaboost의 예측 방식

- 주어진 X값에 대해 Forest of Stump 내의 Stump에 적용

Target: Y로 예측



Target: N로 예측



## 8. Gradient Boost

- **Adaboost VS Gradient Boost**

- Adaboost: 여러 Stump의 순차적 계산
- GB: leaf로 부터 시작
  - Leaf: Target에 대한 초기 추정값(예: 평균,  $\log(\text{odds ratio})$  등)
  - Stump가 아닌 Tree를 생성: 각 tree는 leaf가 8~32개 크기 수준으로 생성

X1	X2	X3	Target
Y	12	Blue	O
Y	87	Green	O
N	44	Blue	X
Y	19	Red	X
N	32	Green	O
N	14	Blue	O

## 8. Gradient Boost

- **Gradient Boost for Classification, Step 1**

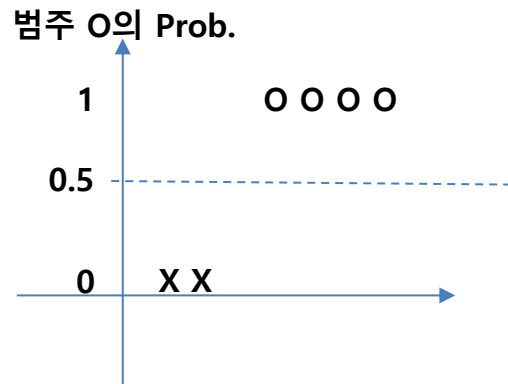
- Leaf의 계산
  - X 범주 2개 대비 O범주는 4개, Odds = 4/2, leaf는  $\log(\text{odds}) = 0.7$

X1	X2	X3	Target
Y	12	Blue	O
Y	87	Green	O
N	44	Blue	X
Y	19	Red	X
N	32	Green	O
N	14	Blue	O

## 8. Gradient Boost

### • Gradient Boost for Classification, Step 1

- Leaf의 계산: X 범주 2개 대비 O 범주는 4개, Odds = 4/2, leaf는  $\log(\text{odds}) = 0.7$
- Leaf를 통한 O 범주의 확률?
  - $\text{Exponential}(\log(\text{odds})) / (1 + \text{exponential}(\log(\text{odds}))) = 0.7$
  - 이 값이 기준인 0.5와 비교하여 O, X 분류
- Residual을 계산: 예를 들어 O는 확률 1이고, leaf 는 0.7이어서 Residual은 0.3



같은 X변수들로  
Residual에 대한 Tree

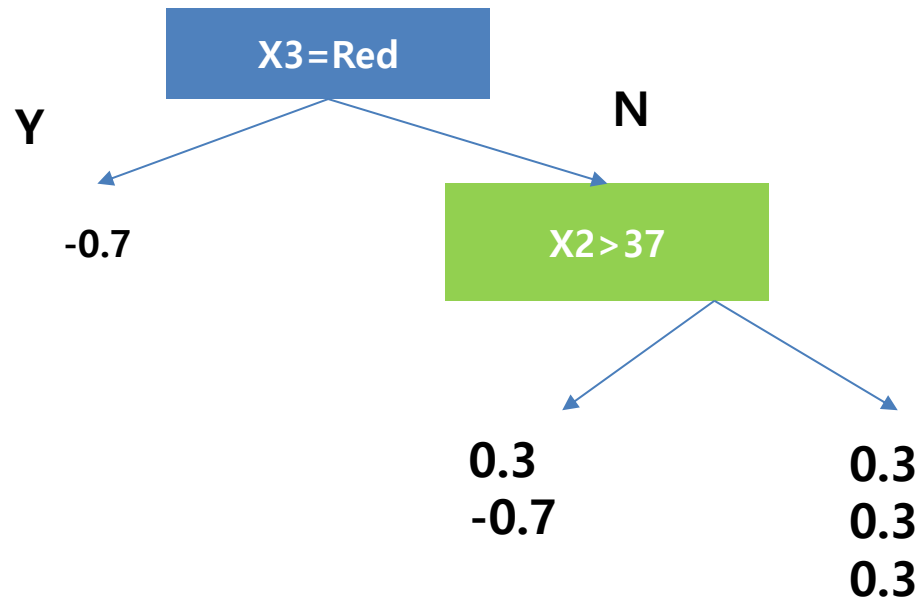
X1	X2	X3	Target	Residual
Y	12	Blue	O	0.3
Y	87	Green	O	0.3
N	44	Blue	X	-0.7
Y	19	Red	X	-0.7
N	32	Green	O	0.3
N	14	Blue	O	0.3

## 8. Gradient Boost

- Gradient Boost, Step 1

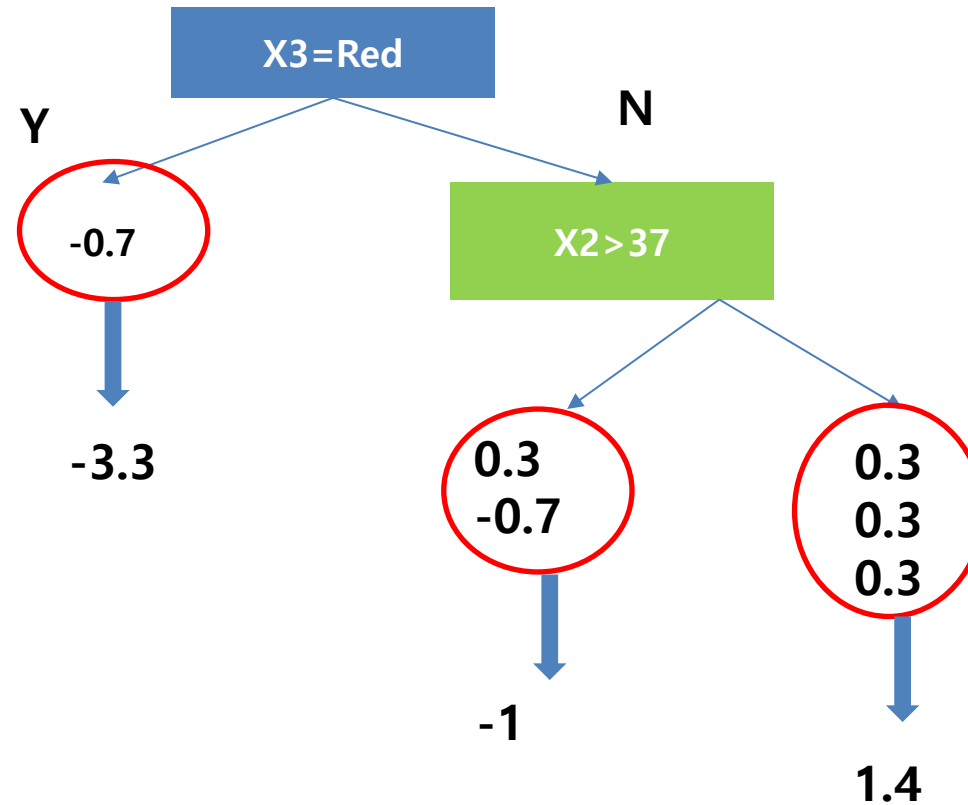
- **1st Tree**

- leaf의 수를 8~32로 제한하며 그 범위내에서 tree 생성



## 8. Gradient Boost

- Gradient Boost, Step 1
  - **1st Tree**

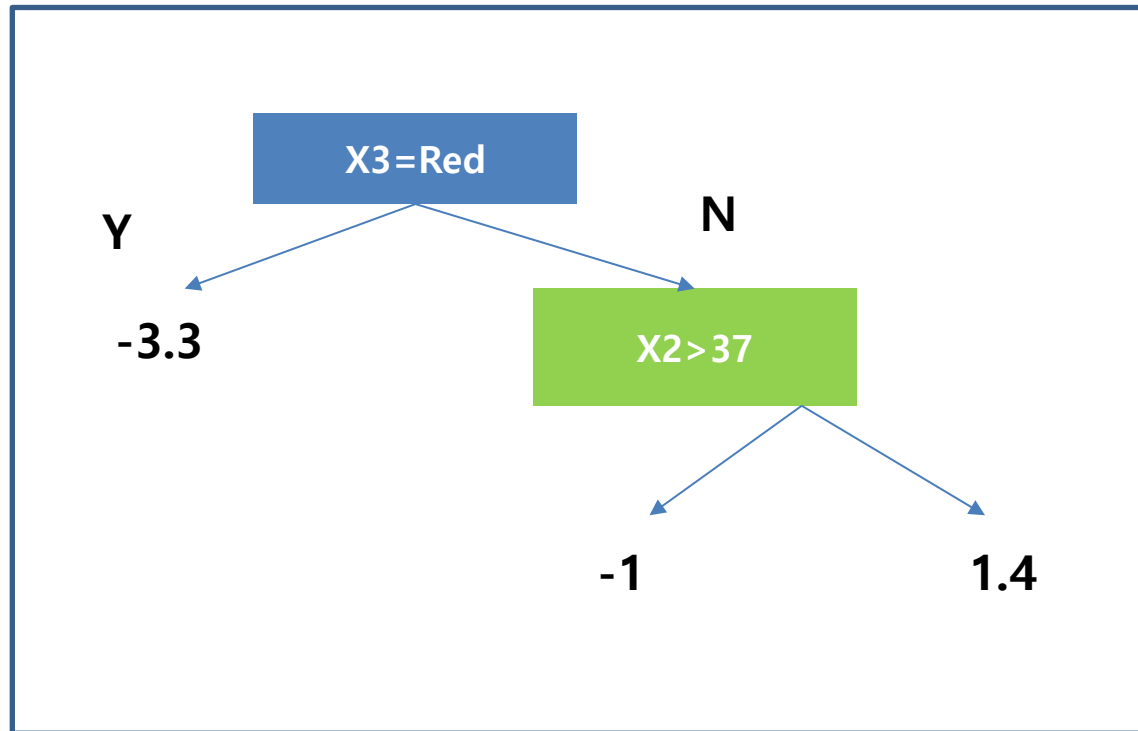


$$\frac{\sum \text{Residuals}}{\sum (\text{Previous Prob} \times (1 - \text{Previous Prob}))}$$

## 8. Gradient Boost

- Gradient Boost, Step 2
  - Leaf 의 initial prediction에 tree에 학습을 반영하여 계산
  - Leaf + 1st Tree

$$0.7 + 0.8 X$$





## 8. Gradient Boost

- **Gradient Boost for Classification, Step 3**

- 각 범주에 대한 발생 확률 계산
  - 1<sup>st</sup> Obs의 업데이트된  $\log(\text{odds})$ 는 1.8
    - Leaf  $0.7 + 1.4(\text{from tree}) \times 0.8 = 1.8$
  - 1<sup>st</sup> Obs의 확률:  $\frac{e^{1.8}}{1+e^{1.8}}$

X1	X2	X3	Target	Residual	Prob.
Y	12	Blue	O	0.3	0.9
Y	87	Green	O	0.3	0.5
N	44	Blue	X	-0.7	0.5
Y	19	Red	X	-0.7	0.1
N	32	Green	O	0.3	0.9
N	14	Blue	O	0.3	0.9

## 8. Gradient Boost

- **Gradient Boost for Classification, Step 3**

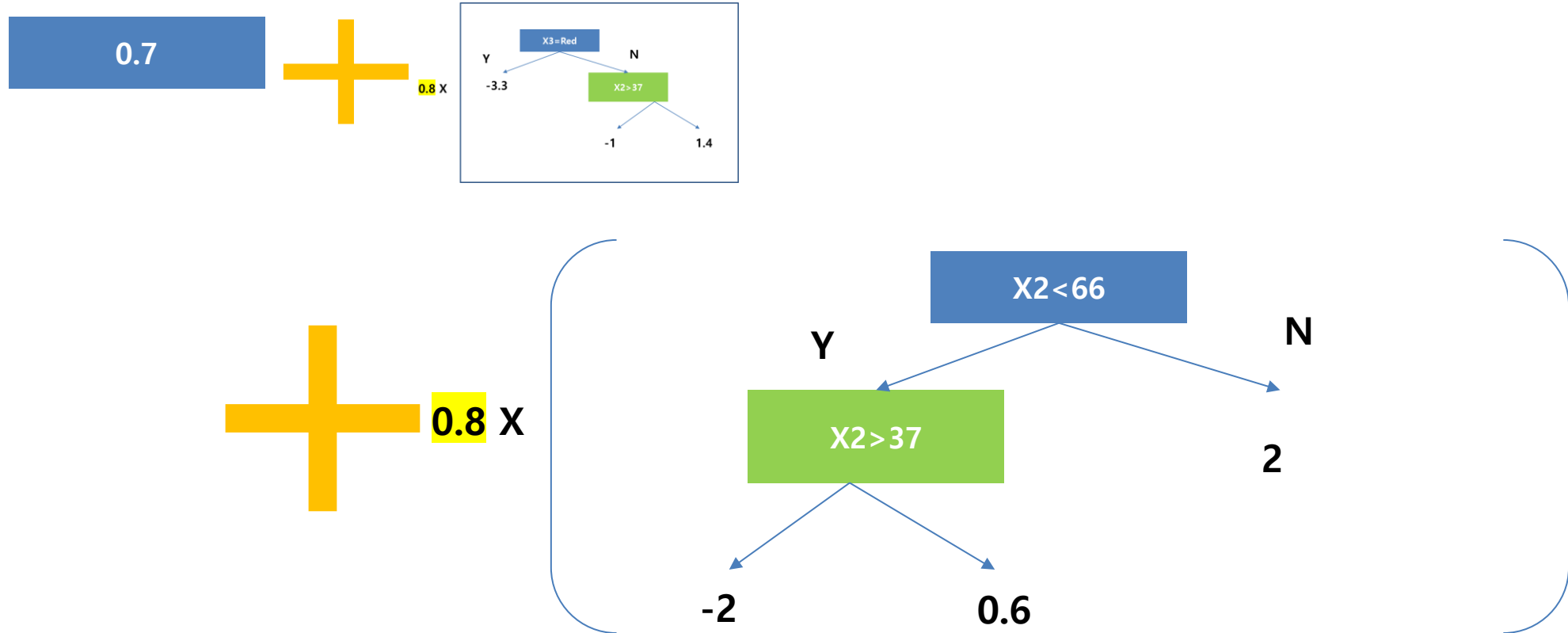
- Residual 다시 계산, 다음 tree 생성

X1	X2	X3	Target	Residual	Prob.	New Residual
Y	12	Blue	O	0.3	0.9	1-0.9
Y	87	Green	O	0.3	0.5	1-0.5
N	44	Blue	X	-0.7	0.5	0-0.5
Y	19	Red	X	-0.7	0.1	0-0.1
N	32	Green	O	0.3	0.9	1-0.9
N	14	Blue	O	0.3	0.9	1-0.9

## 8. Gradient Boost

- **Gradient Boost, Step 3**

- Learning Rate: 0~1사이, 이 예에서는 0.1 사용



## 8. Gradient Boost

- **Gradient Boost for Regression**

- GB: leaf로 부터 시작
  - Leaf: Target에 대한 초기 추정값(예: 평균,  $\log(\text{odds ratio})$  등)
  - Stump가 아닌 Tree를 생성: 각 tree는 leaf가 8~32개 크기 수준으로 생성

Target			
Height	Color	Gender	Weight
1.6	B	M	88
1.6	G	F	76
1.5	B	F	56
1.8	R	M	73
1.5	G	M	77
1.4	B	F	57

## 8. Gradient Boost

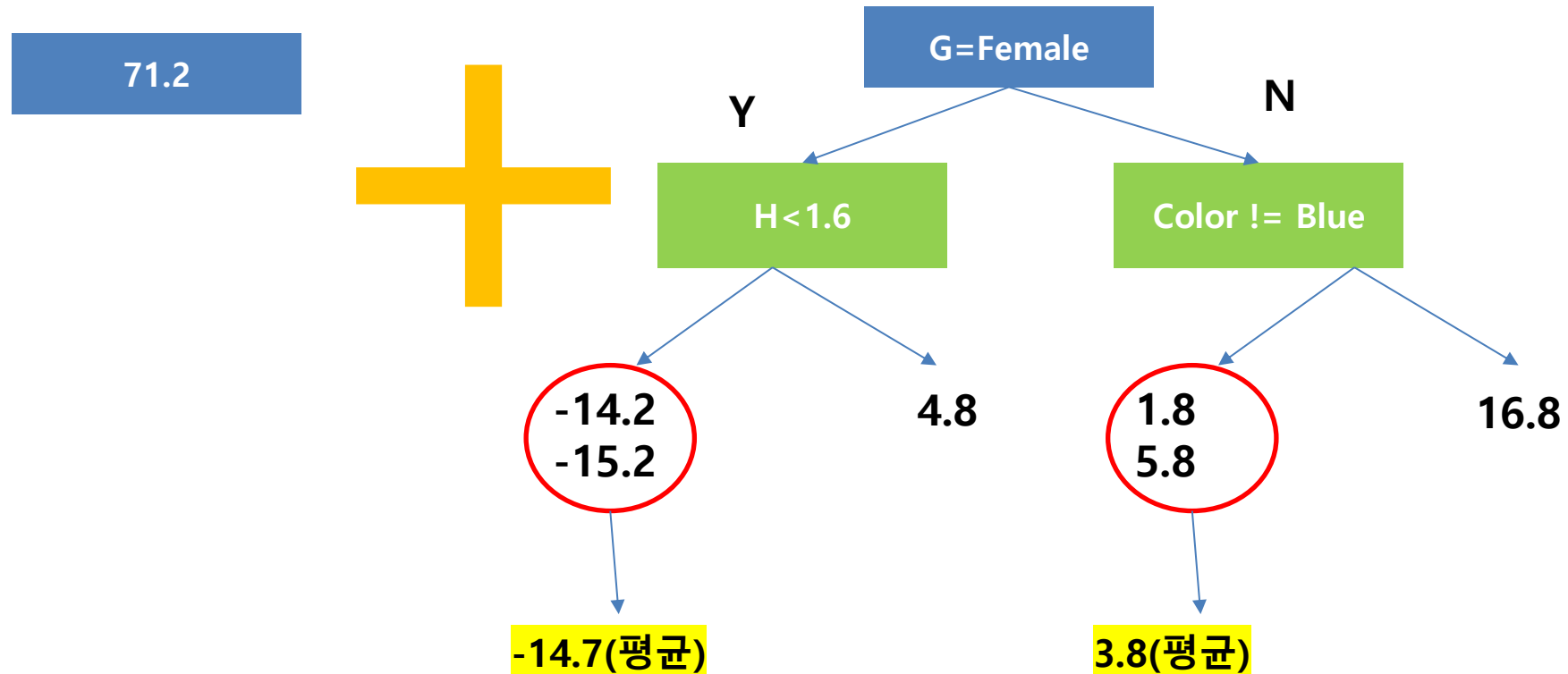
- **Gradient Boost, Step 1**
  - Leaf의 계산
  - Target인 Weight의 평균: 71.2
  - Residual을 계산: 실제값과 예측값의 차이(error)

같은 X변수들로  
Residual에 대한 Tree

Height	Color	Gender	Weight	Residual
1.6	B	M	88	16.8
1.6	G	F	76	4.8
1.5	B	F	56	-15.2
1.8	R	M	73	1.8
1.5	G	M	77	5.8
1.4	B	F	57	-14.2

## 8. Gradient Boost

- Gradient Boost, Step 1
  - Leaf + **1st Tree**

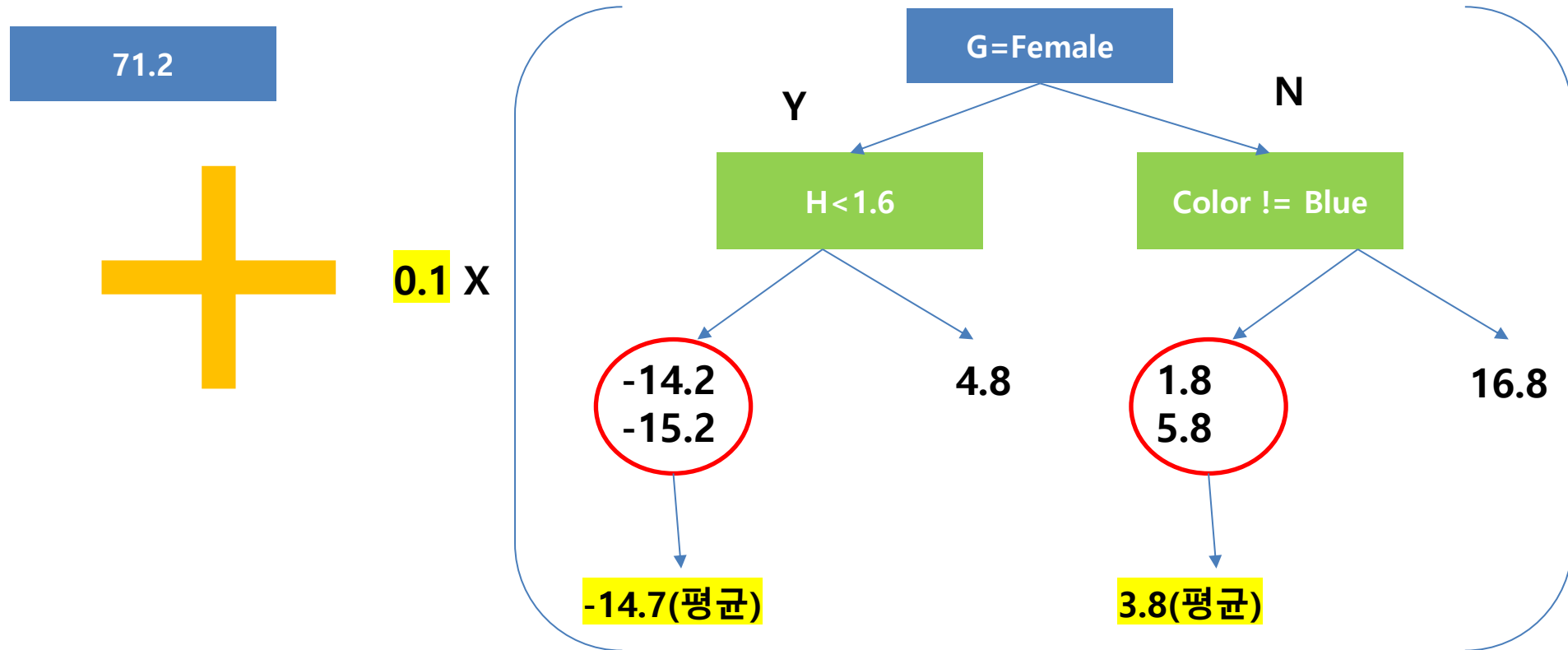


- Male, Blue인 경우 예측 예시:**
  - $71.2 + 16.8 = 88$  (관측치와 동일하지만 과적합)
  - Bias는 작지만 Variance 큰 상태

## 8. Gradient Boost

- Gradient Boost, Step 2

- 과적합 방지, 학습속도 조절을 위한 학습율 도입
- Learning Rate: 0~1사이, 이 예에서는 0.1 사용

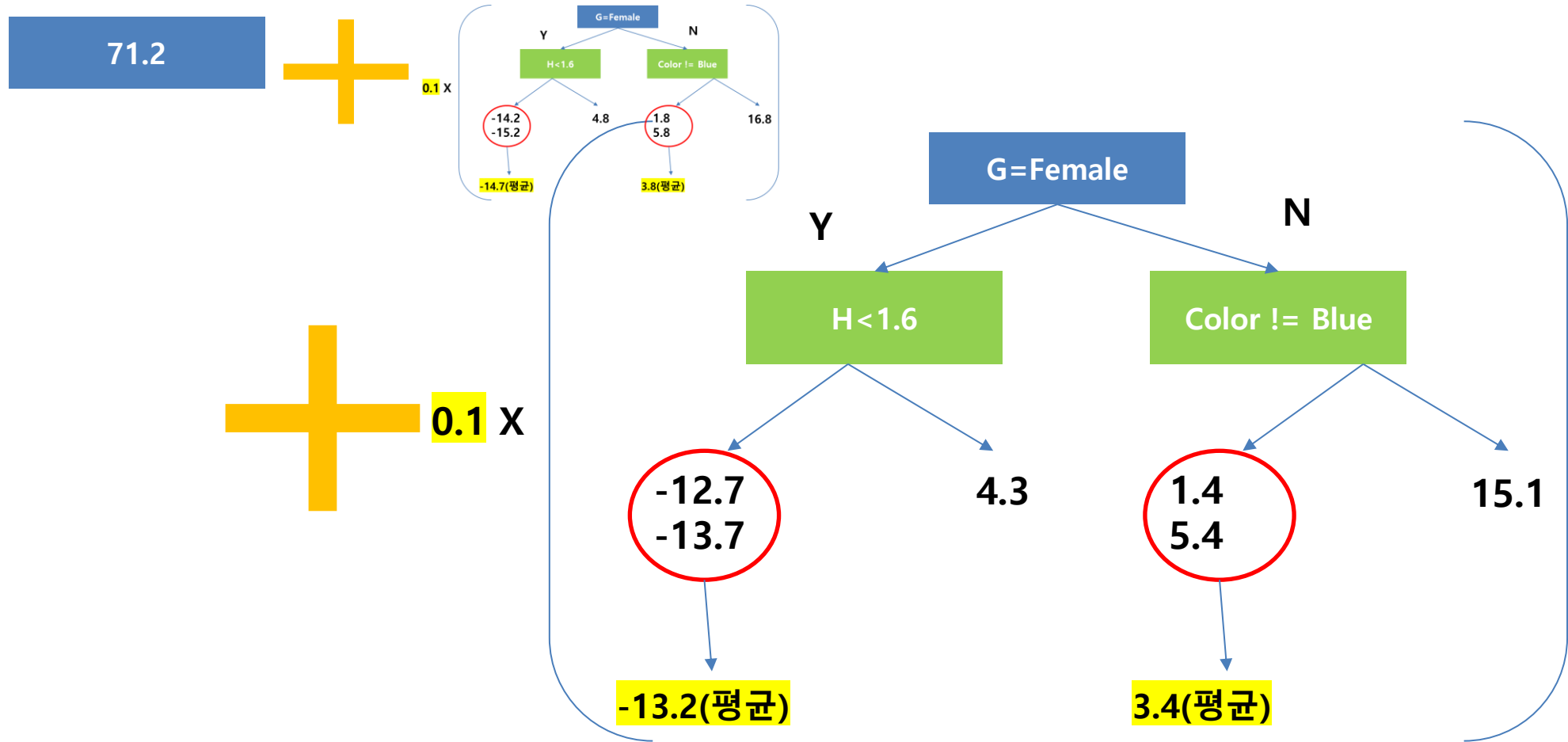


- **Male, Blue인 경우 예측 예시:**  $71.2 + 0.1 \times 16.8 = 72.9$ 
  - 실제값에 가까워지지만, 그 정도가 조절됨 (Gradient의 개념)
  - Variance를 낮게 유지할 수 있음

## 8. Gradient Boost

- Gradient Boost, Step 3

- Learning Rate: 0~1사이, 이 예에서는 0.1 사용



- H=1.6, Male, Blue인 경우 예측 예시:  $71.2 + 0.1 \times 16.8 + 0.1 \times 15.1 = 74.4$



## 8. Gradient Boost

- Gradient Boost, Step 3

- 학습을 반영 예측값을 통한 두 번째 Residual 계산

같은 X변수들로 New  
Residual에 대한 Tree

Height	Color	Gender	Weight	Residual	Residual(new)
1.6	B	M	88	16.8	15.1
1.6	G	F	76	4.8	4.3
1.5	B	F	56	-15.2	-13.7
1.8	R	M	73	1.8	1.4
1.5	G	M	77	5.8	5.4
1.4	B	F	57	-14.2	-12.7



Residual 크기 감소

## 8. Gradient Boost

- **Gradient Boost**

- 위의 과정을 계속 반복
  - 정해진 iteration한도 까지 반복
  - 또는 이전 단계와 이후 단계의 Residual 차이가 없을 때까지 반복
- 매 iteration에서의 Tree의 leaf는 8~32개 사이에서 생성
- 매 iteration마다 다르게 생성
  - 1<sup>st</sup> tree: leaf 8개
  - 2<sup>nd</sup> tree: leaf 32개
  - 3<sup>rd</sup> tree: leaf 16개
  - ...



## Industrial Data Science Lab

Contact:

won.sang.l@gwnu.ac.kr

<https://sites.google.com/view/idslab>