Chapter 6: Classification

Dong-Kyu Chae

PI of the Data Intelligence Lab @HYU
Department of Computer Science & Data Science
Hanyang University

Data
Intelligence
LAB

Topics

- What is classification?
- Issues regarding classification
- Classification by decision tree induction
- Random Forest

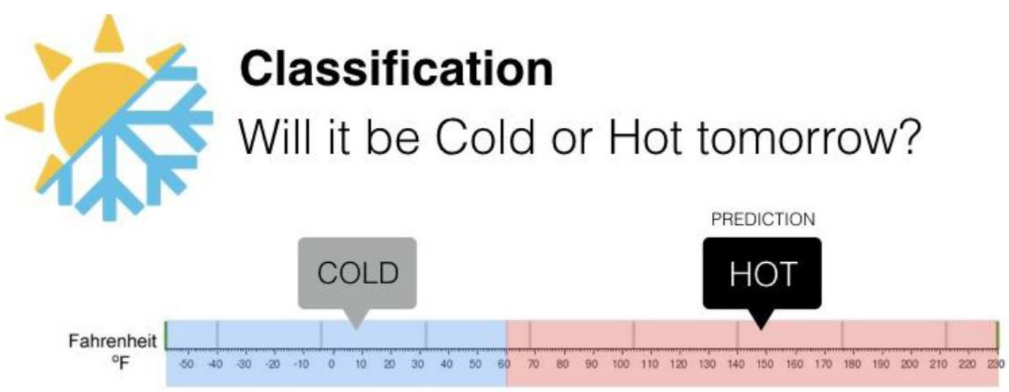
Extended ver.

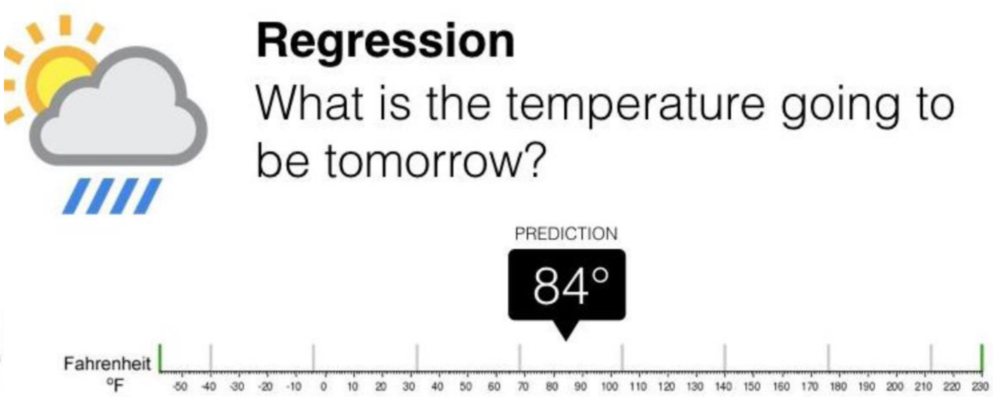
Composed of multiple D.T

- Rule-based classification
- Associative classification
- Lazy learners (or learning from your neighbors)
- Accuracy and error measures
- Ensemble methods
- Summary

What is Classification?

- **Classification**
  - predicts categorical class labels (discrete or nominal)
  - constructs a model by learning the training set (having the class labels) and classifies new data by using the model
- **VS. Regression**
  - It models a continuous-valued functions and predicts unknown or missing values by using the model





Classification

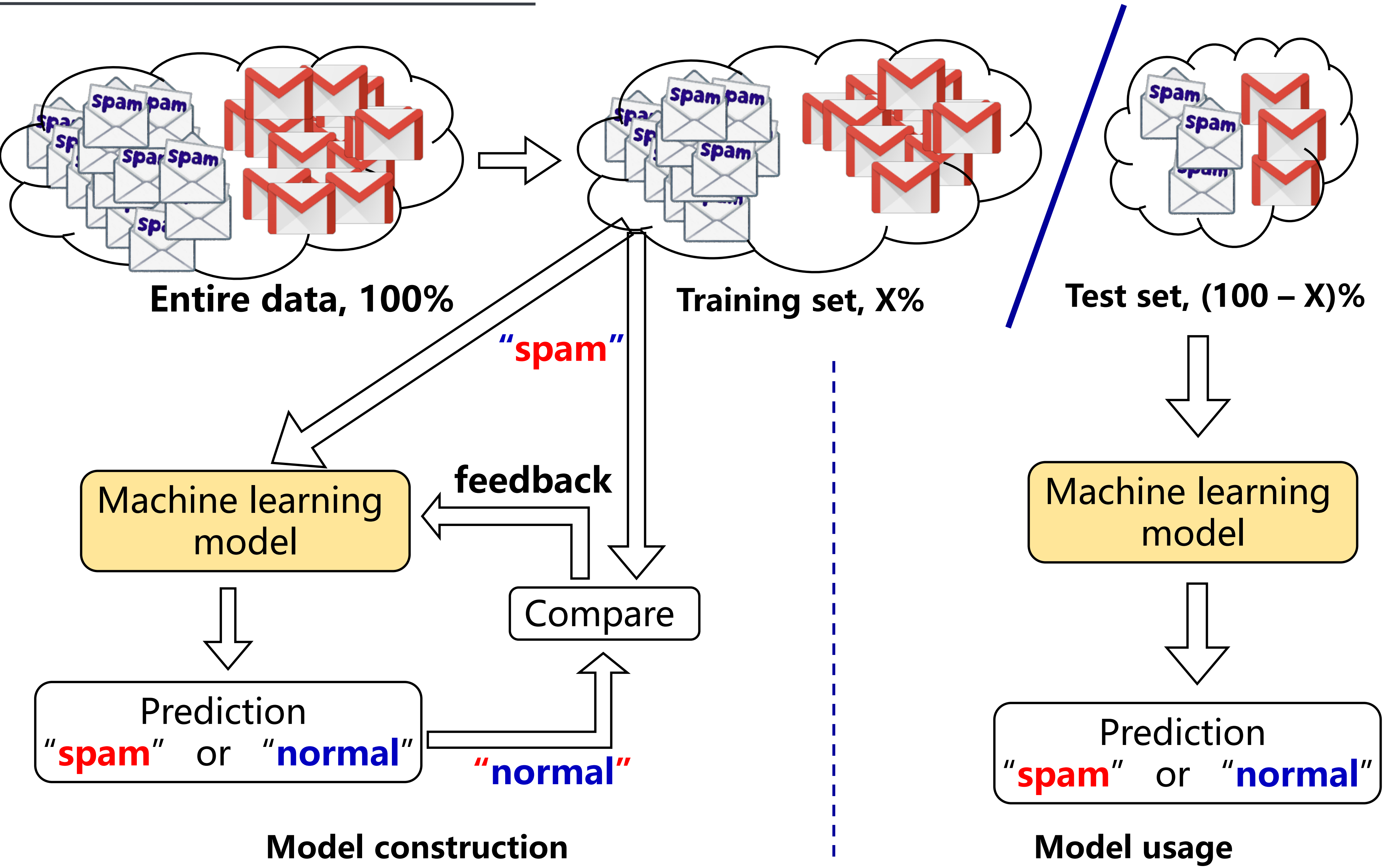
Model construction

- Goal: to describe a set of predetermined classes by using a training data
- Training data
  - A set of data points/tuples/samples used for model construction
  - Each data: <feat-1, feat-2, …., feat-n, class label> (feature / attribute)
  - Each data is assumed to belong to one of all possible classese
- Model
  - Maps each data <feat-1, feat-2, …., feat-n> to a specific class label
  - Represented as classification rules, decision trees, networks, mathematical formula, etc

Model usage

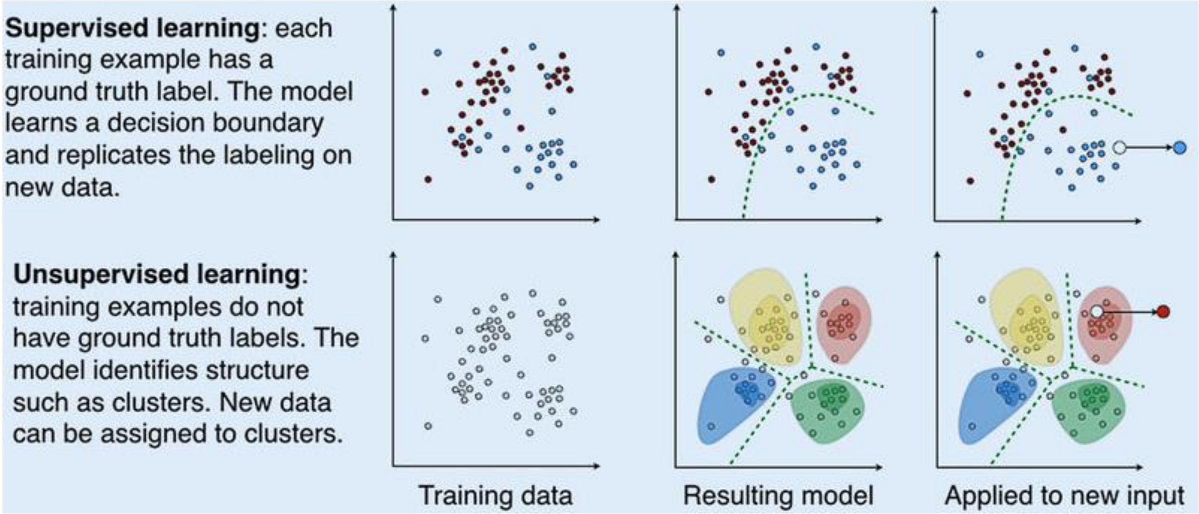
- Goal: to classify the future or unknown samples by using the model

Example: Spam Mail Detection



Supervised vs. Unsupervised Learning

- **Supervised learning (classification)**
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set
- **Unsupervised learning (clustering)**
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of analyzing clusters or distributions in the data



Issues in Classification: Data Preparation

- **Data cleaning**
  - Preprocess data in order to reduce noise and handle missing values
- **Relevance analysis (feature selection)**
  - Remove the irrelevant (index, ID, etc...) or redundant attributes (year-salary and monthly salary, etc...)
- **Data transformation**
  - Generalize and/or normalize data

Issues in Classification: Evaluation Points

- **Accuracy**
  - # of correctly classified data / # of entire test data
- **Speed**
  - time to construct the model (training time)
  - time to use the model (testing time)
- **Robustness:** handling noise, error, outliers and missing values
- **Scalability:** handling a growing size of data

  data 양 늘어나도 모델은 여전히 학습하는데 합리적인 시간 걸려야 함

  data size: n, training complexity: $O(n)$ ⇒ 'Scalable' → natural, affordable

  $O(n^2)$ ⇒ Not scalable → 이런 종류의 모델은 빅데이터 environment에 적용할수없음
- **Interpretability**

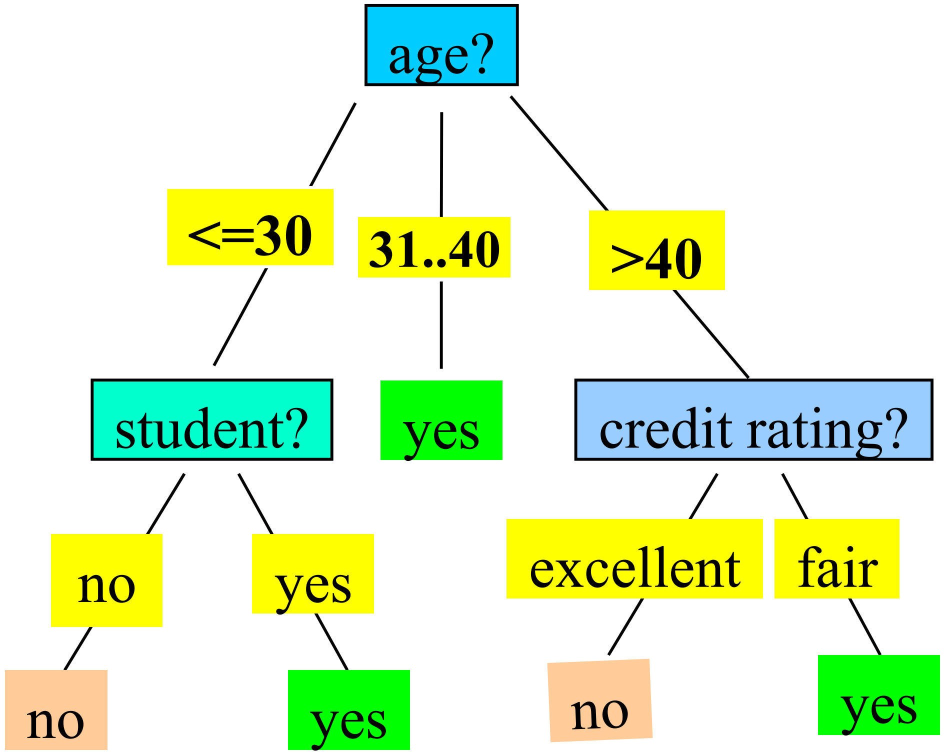
  : can explain how each classification result has been made
  - understanding and insight provided by the model
- …

Decision Tree

What is Decision tree?

- A decision tree is a graphical representation of all the possible solutions to a decision based on certain conditions
- Each **branch node** represents a **choice** between alternatives, and each **leaf node represents a decision**

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no



Overview of Decision Tree Induction : Build DT

Algorithm overview

- A greedy algorithm that constructs a decision tree in a top-down, recursive, divide-and-conquer manner
- At start, all the training examples are at the root
- Examples are partitioned recursively based on the selected feature
- Features are selected on the basis of a heuristic or statistical measure (e.g., information gain)

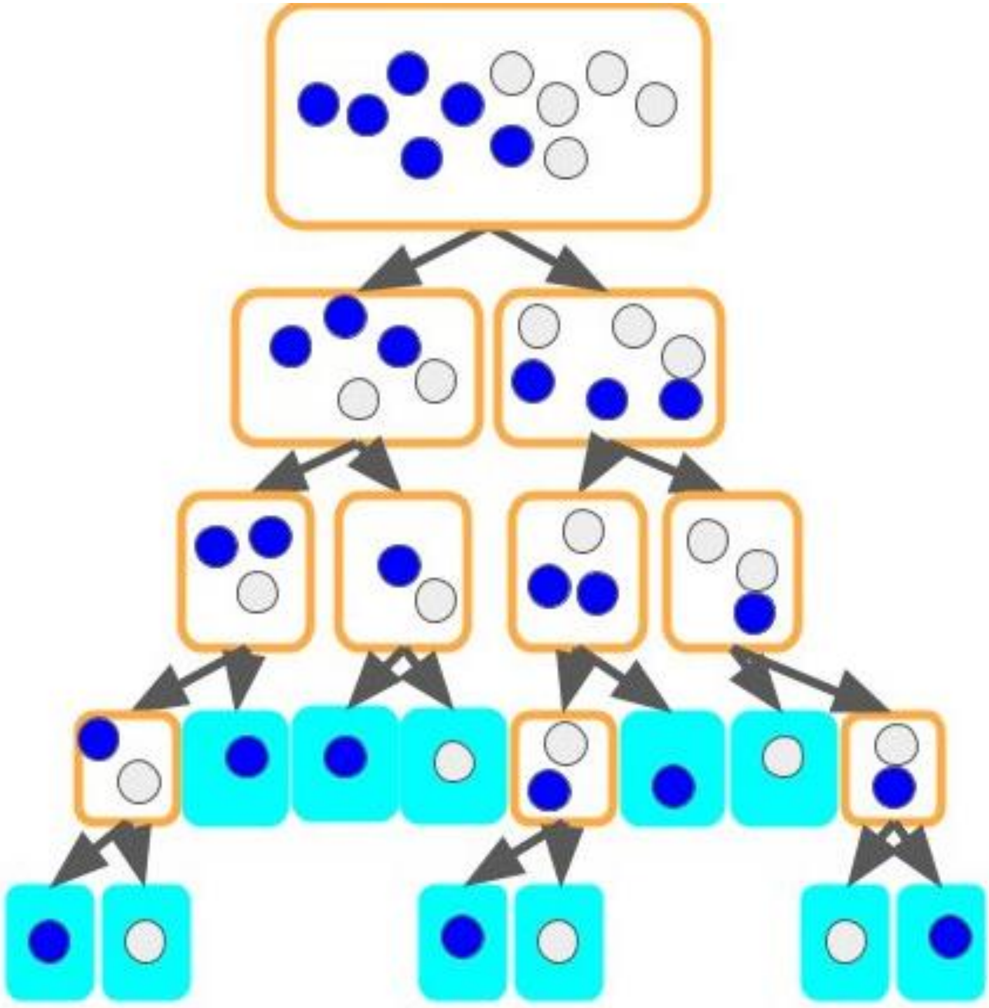
Conditions for stopping the partitioning process

- The training examples for a node belong to the same class – perfectly classified on that branch
- There are no remaining features for further partitioning – majority voting is employed for classifying the leaf

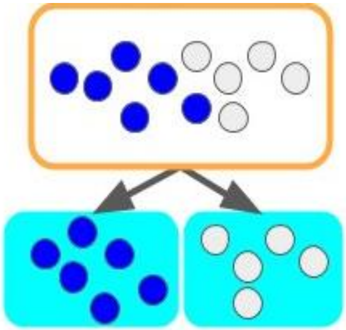
Decision Tree Induction

☐ Basic idea: greedy & recursive & divide and conquer

- Step 1 : Start with an empty tree
- Step 2 : Select a feature to split data
- For each split of the tree:
  - Step 3 : If nothing more to, make predictions
  - Step 4 : Otherwise, go to Step 2 & continue (recurse) on this split



Ideal case



homogeneous (= Not heterogeneous)

Decision Tree Induction

Algorithms

1) **ID3** : **entropy**
2) **C4.5** : **Gain Ratio**
3) **CART** : **Gini index**

all based on same concept: greedy approach, top-down manner, recursive, divide-and-conquer strategy

⇒ only diffrences: How to select the features

Common idea

- For each feature A: **how heterogeneous** (이질적인) the resulting separation is?
  - It measures **how much different classes** of data are mixed after separating them according to a given test feature **A**
  - The lower, the better   The degree of heterogeneous is low, it is better.

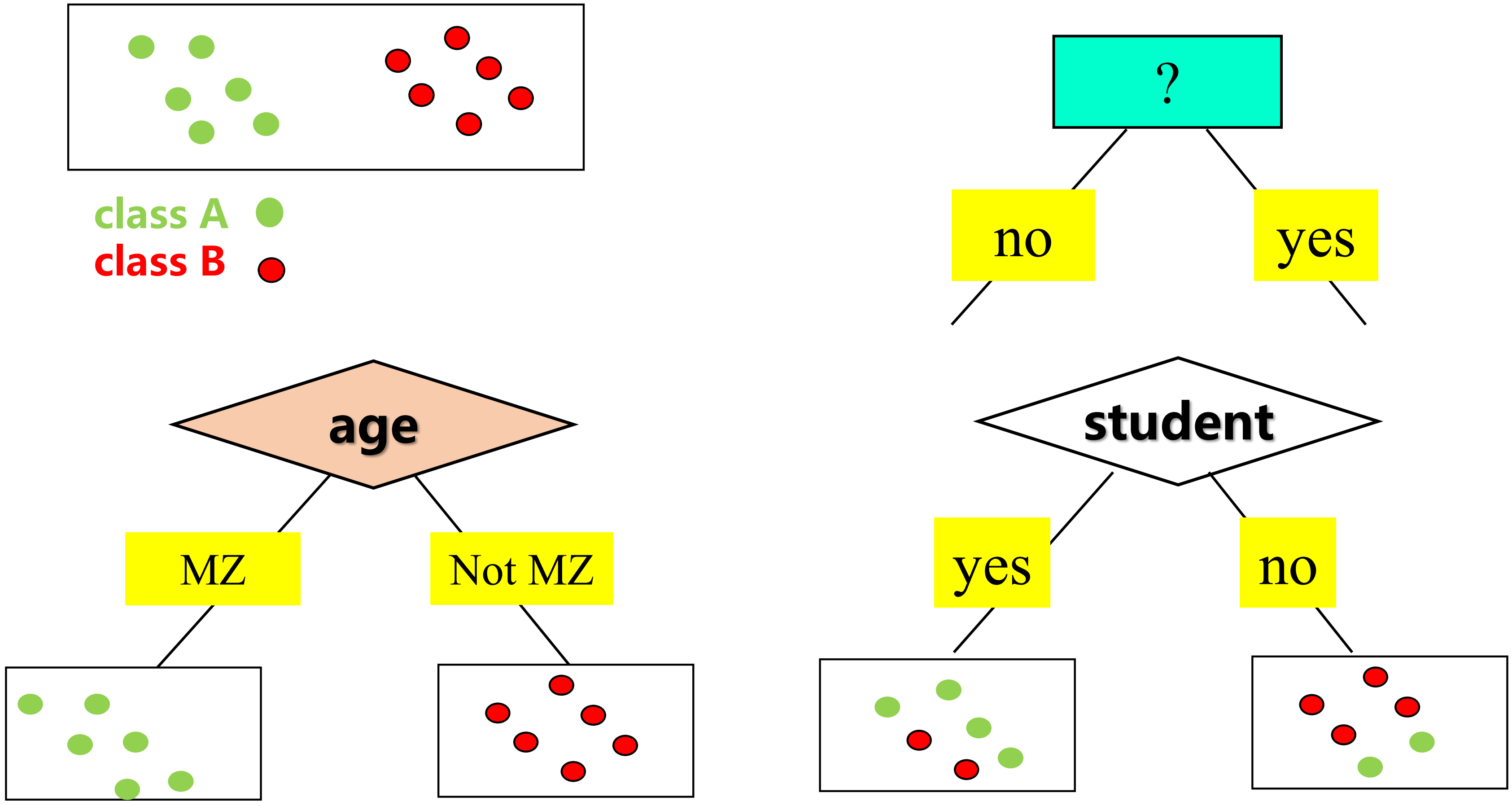
Feature types

- Features are assumed to be categorical (for simplicity)
- If continuous-valued, they are discretized in advance

Feature Selection

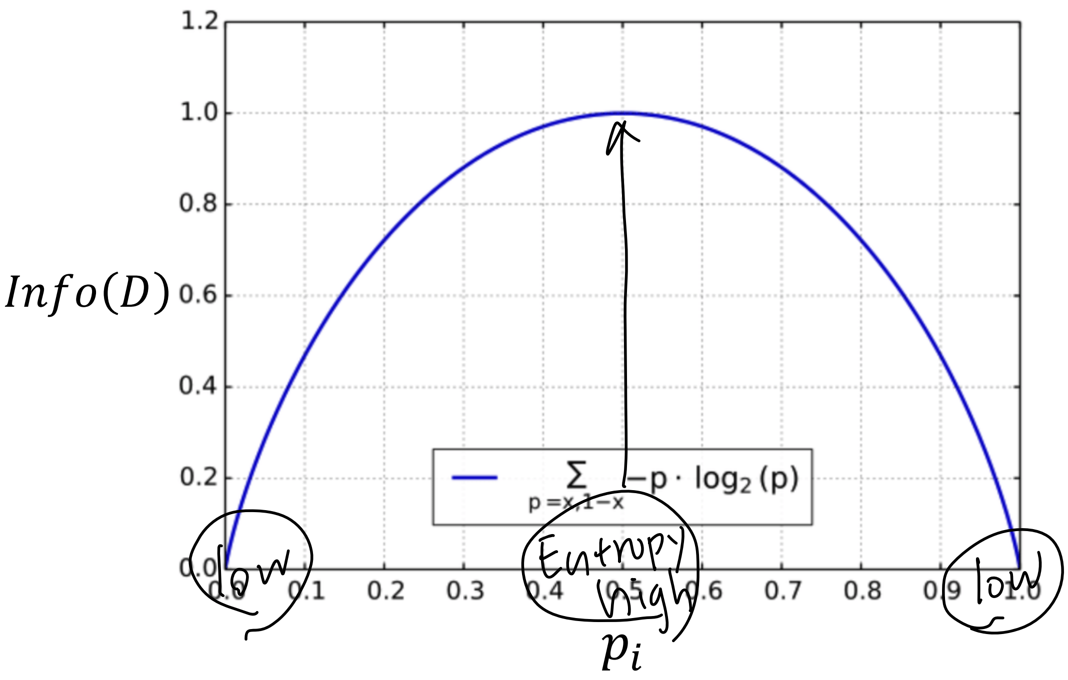
❑ Which feature is the best?

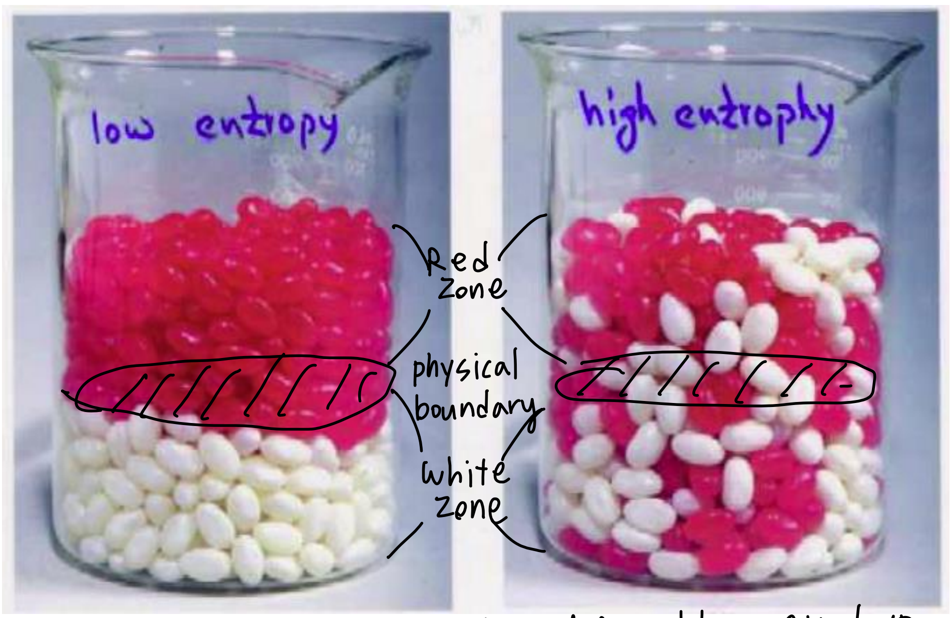
- ❑ Partitions data into **more homogeneous (less heterogeneous)** groups
  - Similar keywords: entropy, impurity, heterogeneity, …



ID3 (Iterative Dichotomiser 3)

- **Information Gain is used, which is based on Entropy.**
  - **Entropy** is a numerical expression of the amount of information in a probability distribution. Entropy can measure how the data mix or well partitioned
  - $Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$
    - p_i is the probability that an arbitrary example in our data D belongs to class i
  - The more **heterogeneous** the data distribution, the **greater** the entropy.





Stable, well seperated, well ordered
Entropy is low

Not stable, messed up
Entropy is high

ID3 (Iterative Dichotomiser 3)

Information gain (GAIN) of a given feature "A"

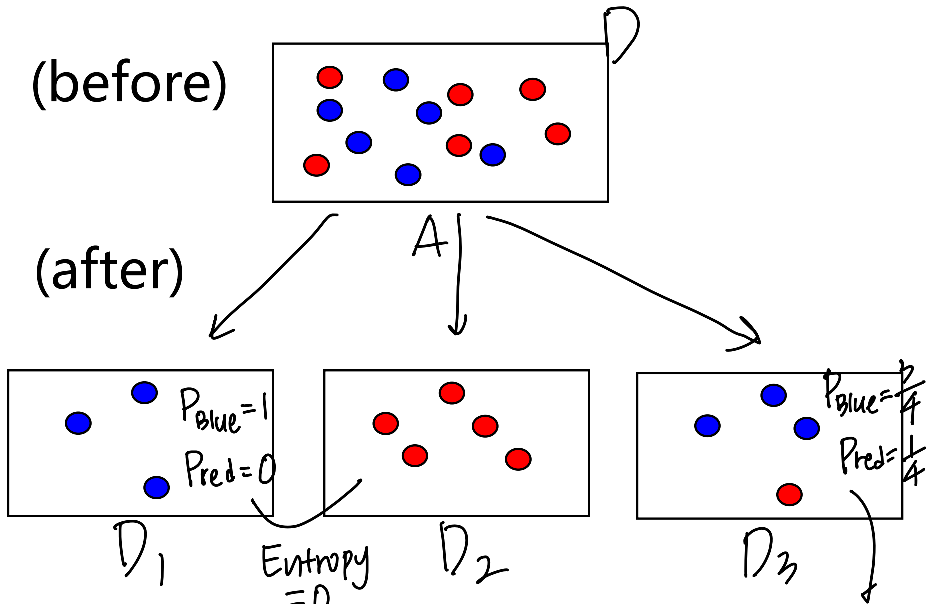
- The **difference of entropy before/after** separating with A
- The feature with the most entropy reduction is the best choice!

$$Gain(A) = Info(D) - Info_A(D)$$

$Info(D) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \qquad \text{(before)}$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j) \qquad \text{(after)}$$



V: # of seperation → (V=3)

- Select one feature that has the **highest information gain**

Entropy

$Info(D_3)$

$= -\frac{3}{4}\log_2(\frac{3}{4}) - \frac{1}{4}\log_2(\frac{1}{4})$

$\Rightarrow Info_A(D) = \frac{4}{12} Info(D_3)$

Working Example

credit	term	income	age	loan
Fair	3years	High	<=30	risky
Fair	3years	High	<=30	risky
Excellent	3years	High	>50	safe
Poor	5years	High	31...50	safe
Poor	5years	Low	31...50	safe
Poor	5years	Low	>50	risky
Excellent	5years	Low	<=30	safe
Fair	3years	High	31...50	risky
Fair	5years	Low	>50	safe
Poor	3years	Low	>50	safe
Fair	3years	Low	31...50	safe
Excellent	3years	High	31...50	safe
Excellent	5years	Low	<=30	safe
Poor	5years	high	>50	risky

credit	s_i	r_i	Info(p_i, n_i)
Excellent	4	0	0
Fair	2	3	0.971
Poor	3	2	0.971

$$Info(D) = \boldsymbol{I(9,5)} = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

$$Info_{credit}(D) = \frac{4}{14}I(4,0) + \frac{5}{14}I(2,3) + \frac{5}{14}I(3,2) = 0.694$$

$$Gain(\boldsymbol{credit}) = Info(D) - Info_{credit}(D) = 0.246$$

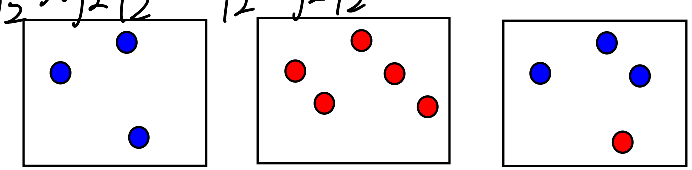
Similarly,

- $Gain(term) = 0.016$
- $Gain(income) = 0.152$
- $Gain(age) = 0.050$

C4.5, an Evolution of ID3

- **Information gain measure is biased towards features with a large number of values**
  - If a feature "A" has 5 values and "B" has 2 values,
  - A tends to have higher information gain than B
- **C4.5 uses Gain Ratio (normalization to information gain)**
  - **Gain Ratio** takes the number and size of branches into account when choosing a feature

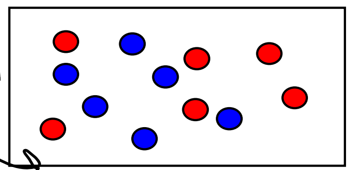
$- \frac{3}{12}\log_2\left(\frac{3}{12}\right) - \frac{5}{12}\log_2\frac{5}{12} - \frac{4}{12}\log_2\frac{4}{12}$



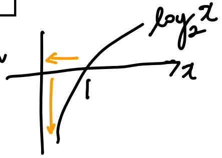
$$SplitInfo_A(D) = -\sum_{j=1}^{3} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

$$Gain(A) = Info(D) - Info_A(D)$$

Mixture of class label is totally ignored



분자값 작아지면



Seperation 더 많이 될수록 SplitInfo 값 ↑

→ log → − 큰값 → Entropy 식에 더해지면 → + ↑↑

$$\boldsymbol{GainRatio}(A) = Gain(A) \,/\, SplitInfo_A(D)$$

Panalty (: how many partitions are created by feature A)

C4.5, an Evolution of ID3

Example

credit	term	income	age	loan
Fair	3years	High	<=30	risky
Fair	3years	High	<=30	risky
Excellent	3years	High	>50	safe
Poor	5years	High	31…50	safe
Poor	5years	Low	31…50	safe
Poor	5years	Low	>50	risky
Excellent	5years	Low	<=30	safe
Fair	3years	High	31…50	risky
Fair	5years	Low	>50	safe
Poor	3years	Low	>50	safe
Fair	3years	Low	31…50	safe
Excellent	3years	High	31…50	safe
Excellent	5years	Low	<=30	safe
Poor	5years	high	>50	risky

credit	s_i	r_i	$Info(p_i, n_i)$
Excellent	4	0	0
Fair	2	3	0.971
Poor	3	2	0.971

$$\text{Gain}(credit) = Info(D) - Info_{credit}(D) = 0.246$$

$$\text{SplitInfo}(credit) = -\frac{5}{14}\log_2\frac{5}{14} - \frac{4}{14}\log_2\frac{4}{14} - \frac{5}{14}\log_2\frac{5}{14}$$
$$= 1.557$$

$$\textbf{GainRatio}(credit) = 0.246/1.577 = 0.1559$$

CART (Classification and Regression Trees)

- **Gini index**: shares same idea with the entropy
  - $gini(D) = 1 - \sum_{i=1}^{v} p_i^2$, where j indicates the class index
  - For a feature “A”, $gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$
  - Its idea is very similar with ID3: ($Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$)

ID3와 비슷하므로 Skip

- **Reduction in impurity:**
  - which is very similar with Information Gain
  - The feature providing the largest reduction in impurity (using information gain) is chosen to as the test feature to split the node

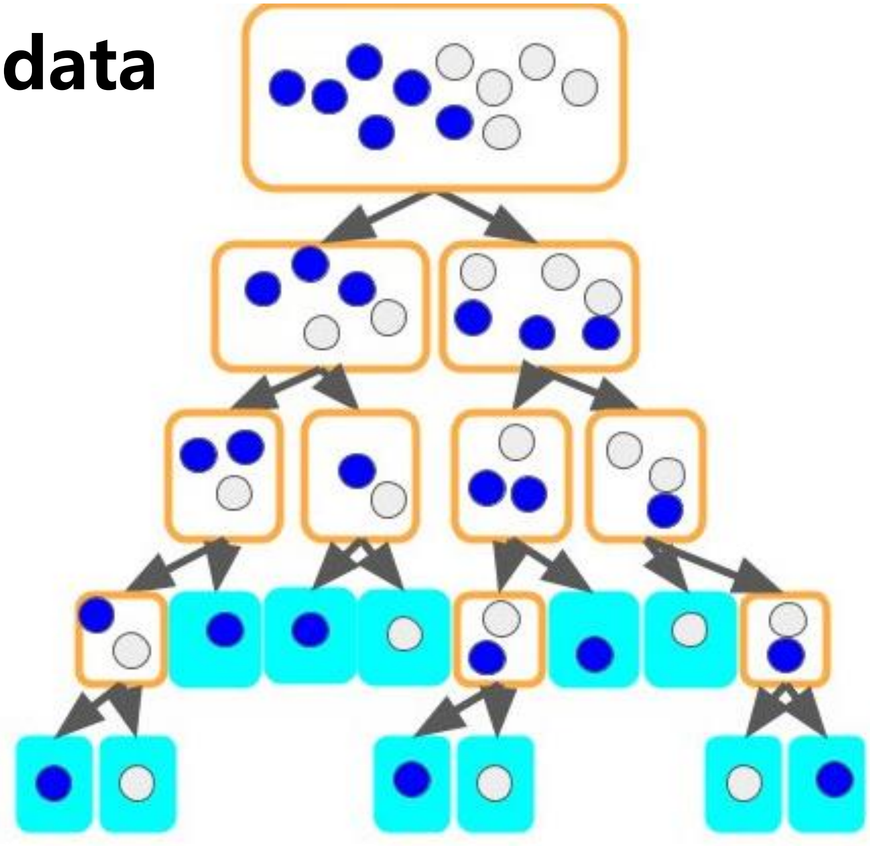
$$\Delta gini(A) = gini(D) - gini_A(D)$$

Overfitting

- **When to stop?**
  - One way: stop when all samples for a given node **belong to the same class**

- **100% accuracy for the training data**
  - It is good? No, overfitting!

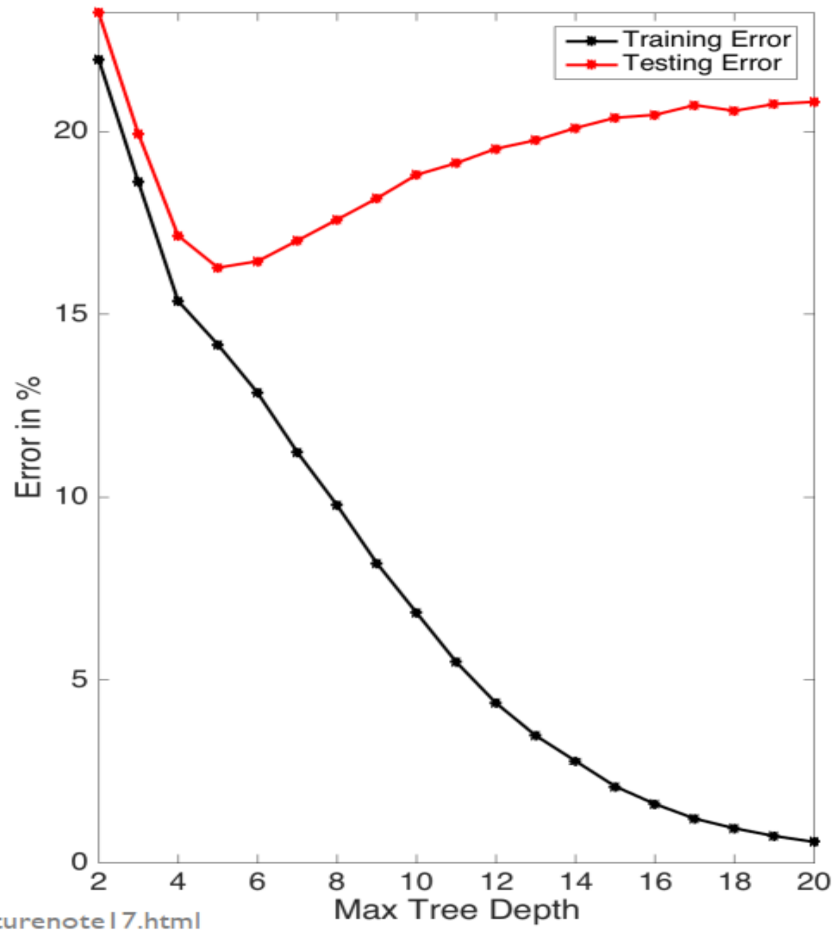
	# of words	# of attached files	# of links	# of malicious words	.....	spam
mail #1	256	0	3	7	.....	1 (Yes)
mail #2	56	1	0	3	.....	0 (No)
mail #3	24	1	0	1	.....	0
mail #4	672	0	0	0	.....	0
mail #5	67	2	4	3	.....	1
mail #6	48	0	2	6	.....	0
mail #7	79	1	3	8	.....	1
			.....			



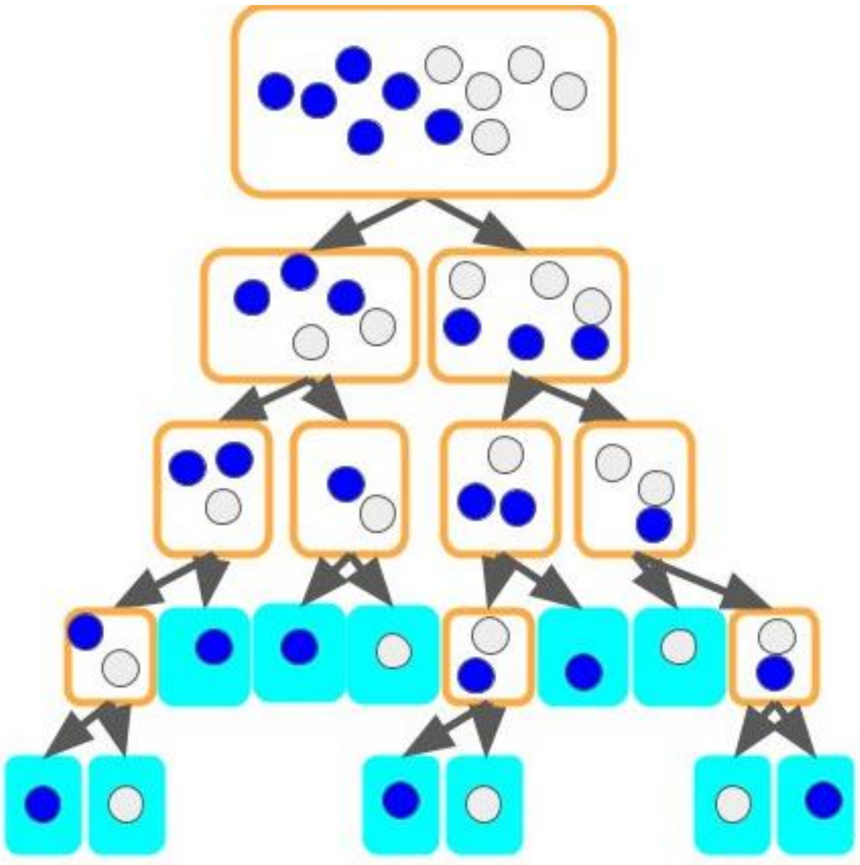
Overfitting

❑ Overfitting of decision tree models

- ❑ Too many branches, some may reflect anomalies due to noise or outliers
- ❑ Poor accuracy for unseen samples

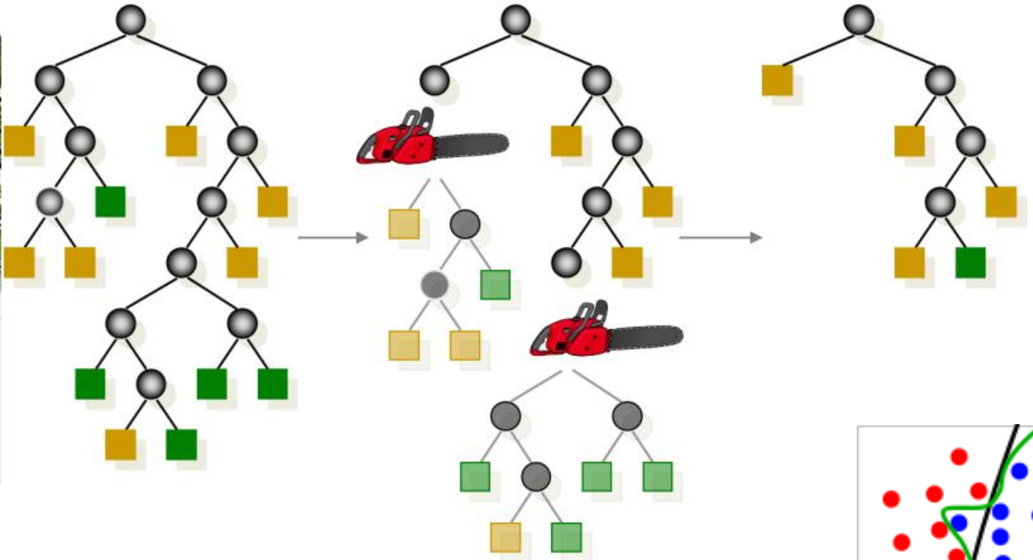


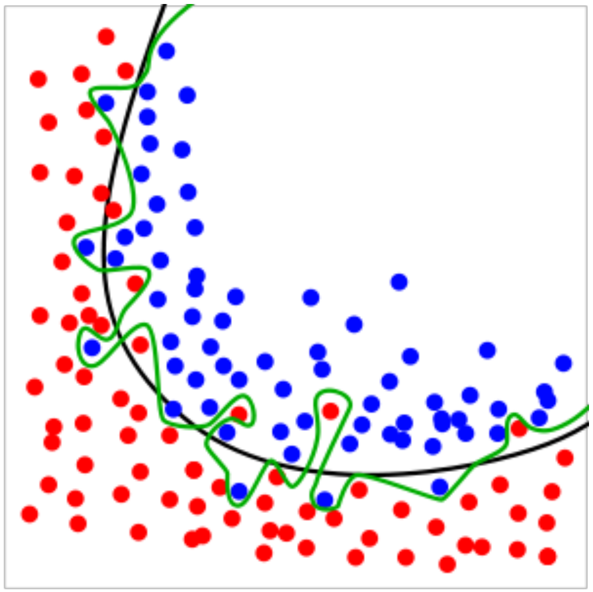
turenote17.html



Tree Pruning

- ❑ **Pruning can be seen as smoothing the decision boundary**





Tree Pruning

Two approaches to avoid overfitting

Pre-pruning: Halt tree construction early

→ while construct tree

- Do not split a node if this would result in the goodness measure falling below a threshold → validation set으로 구할 수 있음
- **Difficult to choose an appropriate threshold**

: Minimum samples split, Maximum tree depth, Minimum gain ….

Post-pruning: Remove branches from a "fully grown" tree

- Get a sequence of progressively pruned trees
- Use a set of data **(validation set)** different from the training data to decide which is the "best pruned tree"
- Nodes are removed only if the resulting pruned tree performs no worse than the original over **the validation set.** pruning 후 accuracy가 더 좋으면 pruning 하기로 결정
- Pruning of nodes continues until further pruning is harmful (i.e., decreases accuracy of the tree over the validation set)

Random Forest

overfitting 피할수있는 더 널리 사용되는 방법

❑ Ensemble

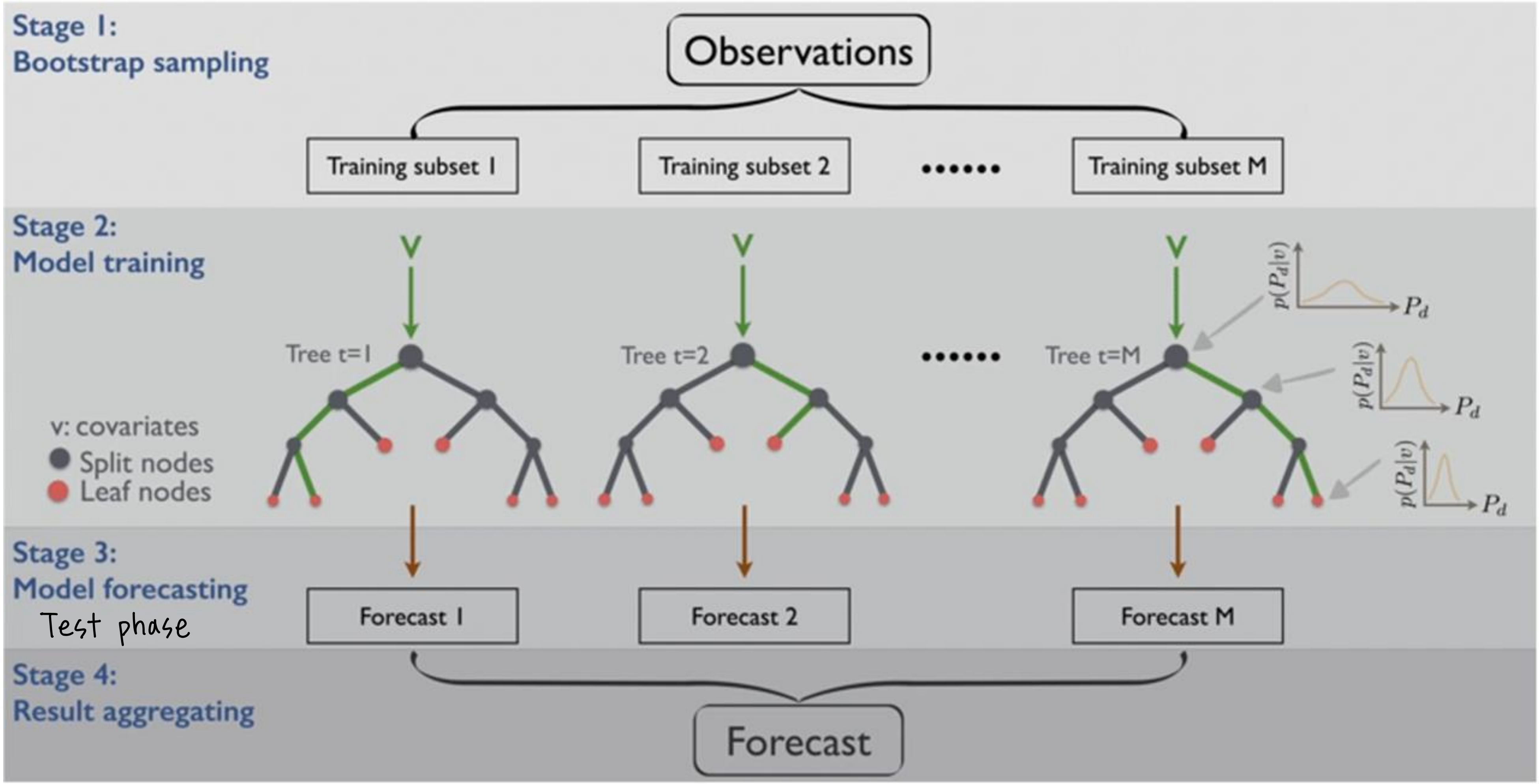
- ❑ It relies on multiple models to increase the predictive power

❑ Random Forest

- ❑ It is a decision tree version of ensemble
- ❑ Forest: it builds 500 (or less) ~ 10,000 (or more) decision trees

Random Forest

❑ Graphical overview



Random Forest

❑ Drawing a bootstrap sample

- ❑ Randomly sample data one-by-one **with replacement**
- ❑ Repeat N times (N is the number of examples in our training data)

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y^{10}
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

duplicate 가능

Bootstrap 2

x^7	y^7
x^1	y^1
x^{10}	y^{10}
x^1	y^1
x^8	y^8
x^6	y^6
x^2	y^2
x^6	y^6
x^4	y^4
x^9	y^9

• • •

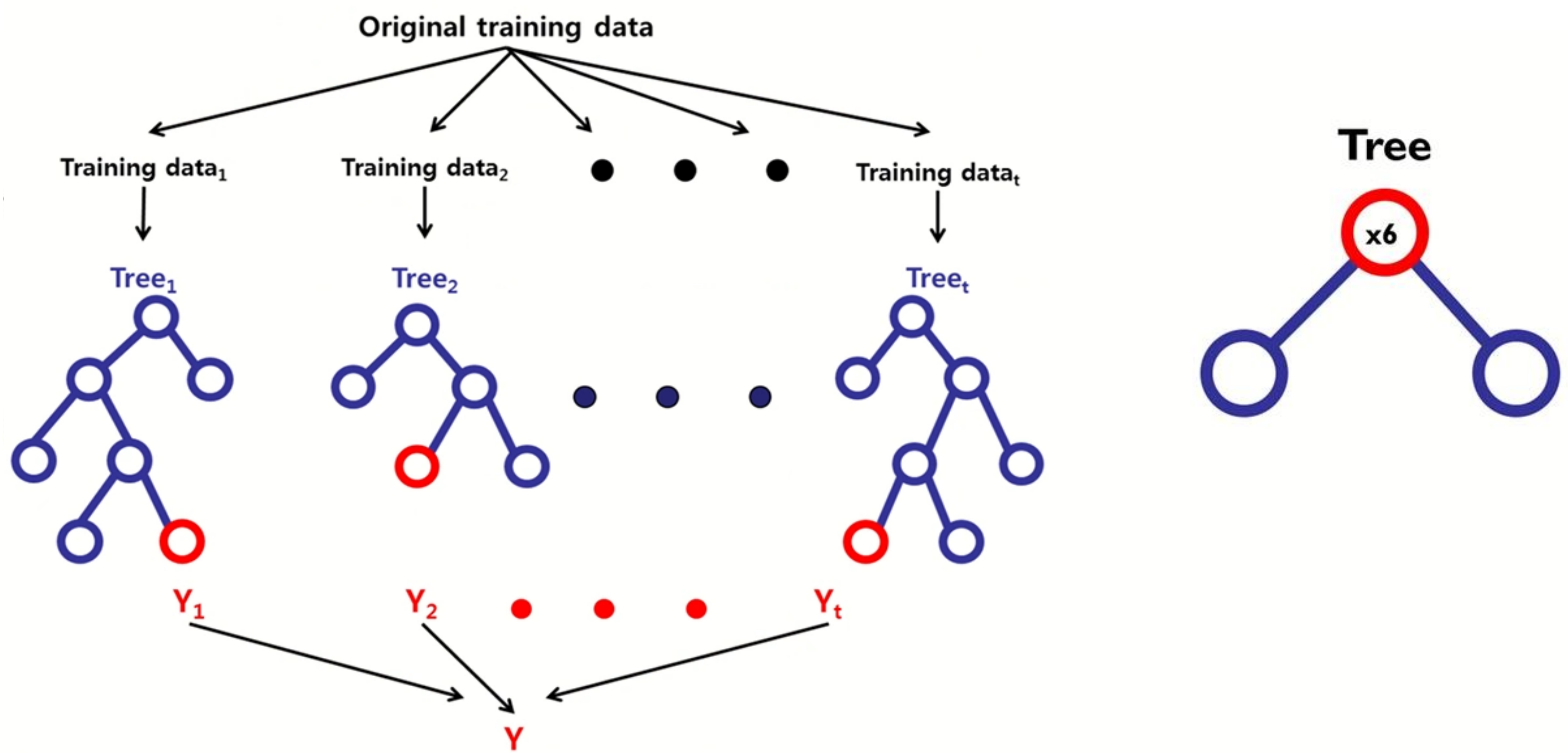
Bootstrap B

x^9	y^9
x^5	y^5
x^2	y^2
x^4	y^4
x^7	y^7
x^2	y^2
x^5	y^5
x^{10}	y^{10}
x^8	y^8
x^2	y^2

original dataset과 같은 수의 data points를 가지도록 구성.

Random Forest

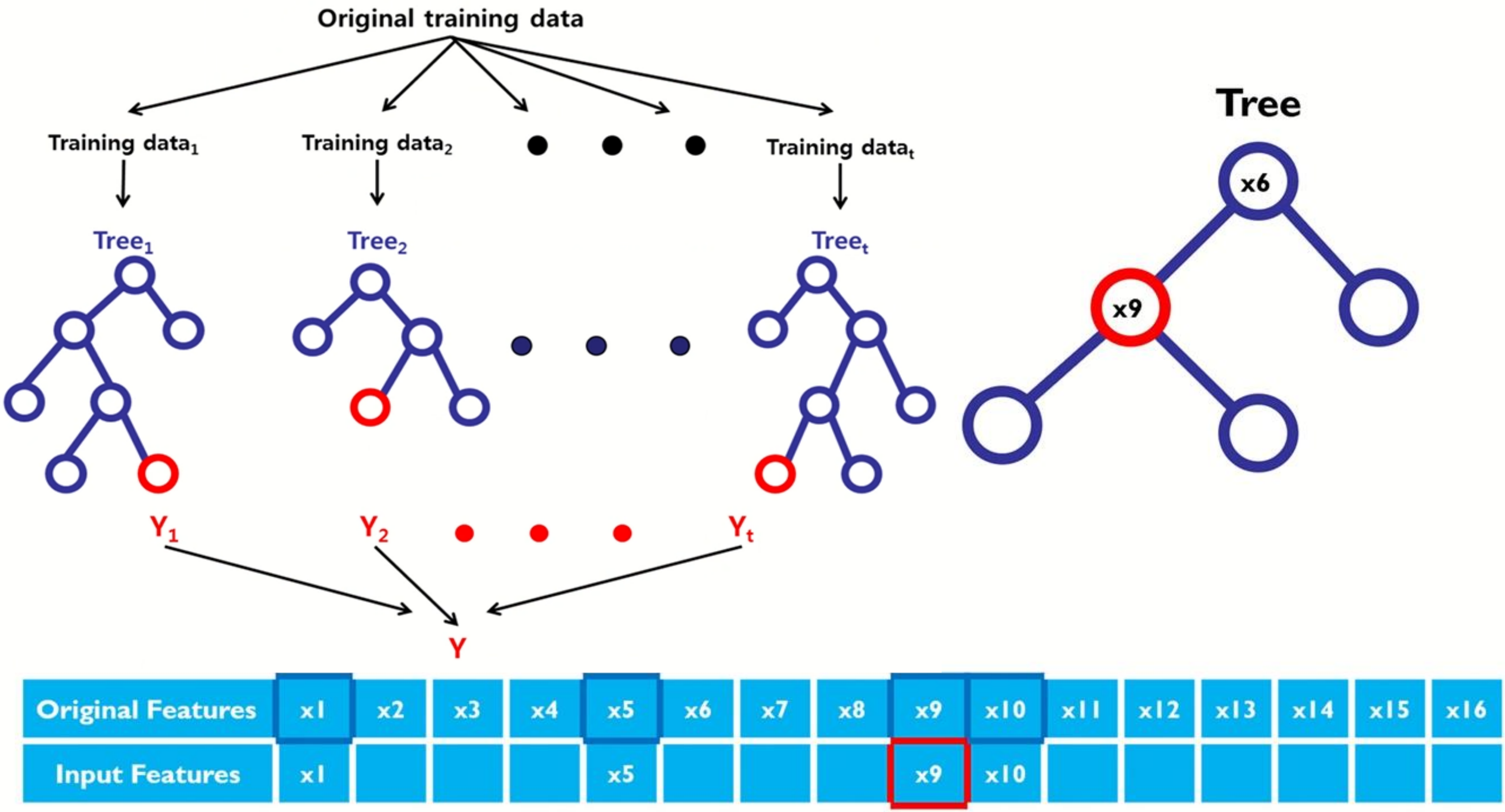
- **Randomly select several features from the entire feature set**



Original Features	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16
Input Features			x3		x5	x6				x10						

Random Forest

- **Randomly select** several features from the entire feature set



Random Forest

❑ Aggregation: majority voting

$$\hat{y}_{Ensemble} = arg\max_i \left(\sum_{j=1}^{n} \delta(\hat{y}_j = i), \quad i \in \{0, 1\} \right)$$

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

$$\sum_{j=1}^{n} \delta(\hat{y}_j = 0) = 4$$

$$\sum_{j=1}^{n} \delta(\hat{y}_j = 1) = 6$$

$$\hat{y}_{Ensemble} = 1$$

Random Forest

❑ Aggregation: weighted voting

If you want to apply more opinion
performing well decision tree,
train accuracy를 weigh로 적용

$$\hat{y}_{Ensemble} = arg\max_i \left(\frac{\sum_{j=1}^{n}(TrnAcc_j) \cdot \delta(\hat{y}_j = i)}{\sum_{j=1}^{n}(TrnAcc_j)}, \quad i \in \{0, 1\} \right)$$

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

$$\frac{\sum_{j=1}^{n}(TrnAcc_j) \cdot \delta(\hat{y}_j = 0)}{\sum_{j=1}^{n}(TrnAcc_j)} = 0.424$$

$$\frac{\sum_{j=1}^{n}(TrnAcc_j) \cdot \delta(\hat{y}_j = 1)}{\sum_{j=1}^{n}(TrnAcc_j)} = 0.576$$

$$\hat{y}_{Ensemble} = 1$$

Summary

Decision tree model

- Recursively select the best feature/attribute in a greedy manner
- Information gain, entropy, gini index, ….

Overfitting

- Tree pruning
- Random forest

Random forest

- Ensemble of 500~10,000 decision trees
- Each tree is trained on a bootstrapped sample
- When constructing each tree, we randomly select a small number of candidate features in every recursion

Thank You