

# Machine Learning

# 机器学习

## Lecture 4: 决策树

李洁

nijanice@163.com

# 监督学习

## Supervised Learning

- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1,2,\dots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx f_{\theta}(x^{(i)})$$

- Function set  $\{f_{\theta}(x^{(i)})\}$  is called hypothesis space
- Learning is referred to as updating the parameter  $\theta$  to make the prediction closed to the corresponding label

# 监督学习

## Supervised Learning

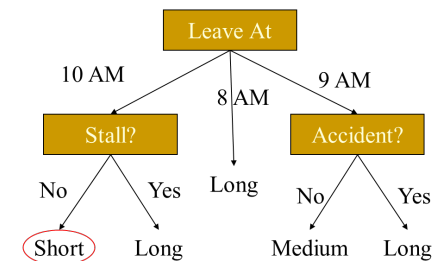
- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1,2,\dots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx f_{\theta}(x^{(i)}) \Rightarrow$$

a decision tree



- Function set  $\{f_{\theta}(x^{(i)})\}$  is called hypothesis
- Learning is referred to as updating the parameters  $\theta$  so that the prediction is close to the corresponding label

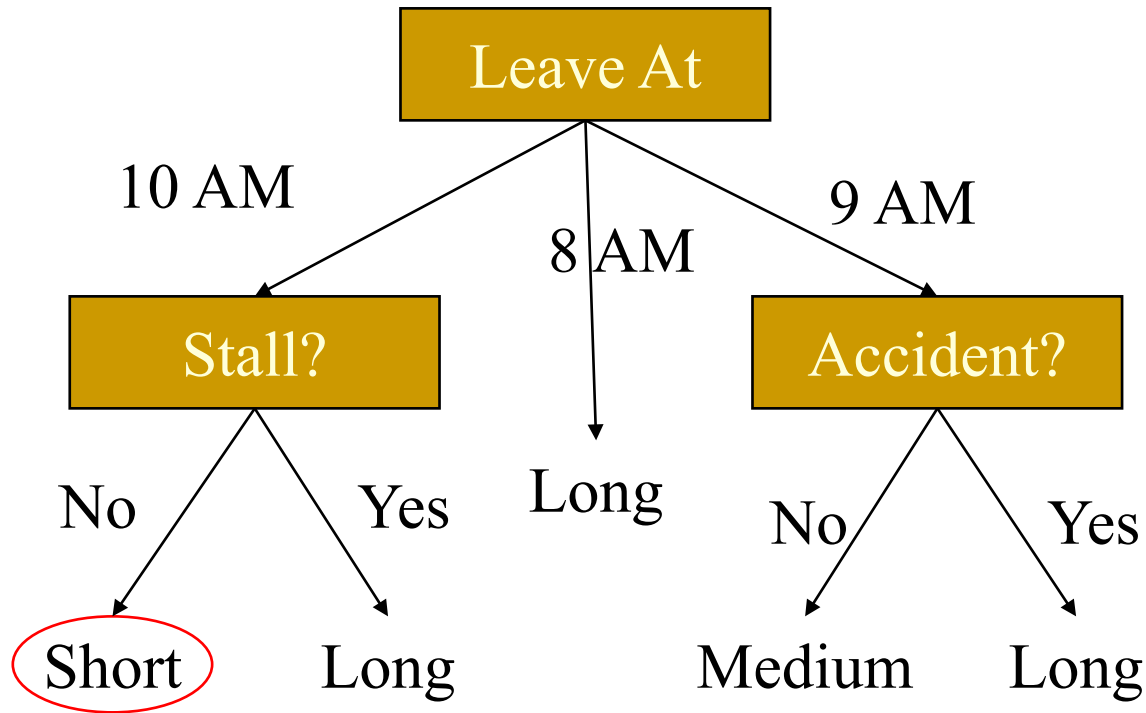
# 什么是决策树

## What is a Decision Tree?

- Let's look at a sample decision tree...

# 决策树举例

## Predicting Commute Time

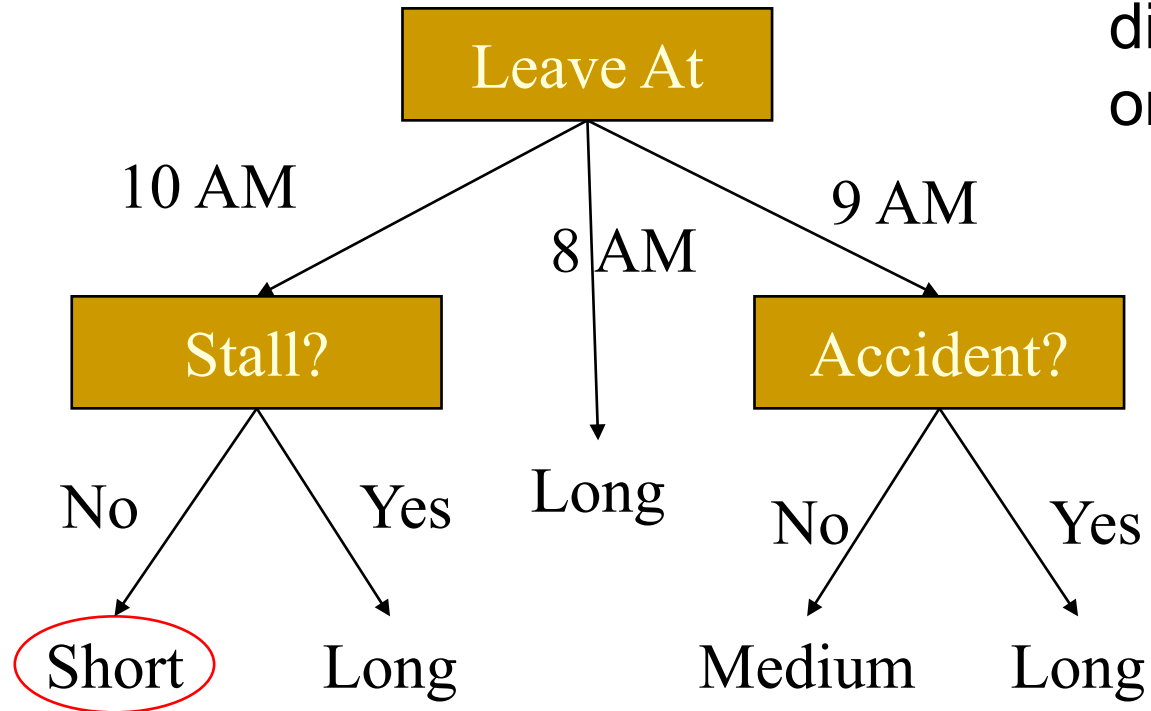


- *Decision tree representation:*
  - Each internal node tests an attribute
  - Each branch corresponds to attribute value
  - Each leaf node assigns a classification

If we leave at 10 AM and there are no cars stalled on the road, what will our commute time be?

# 决策树与规则集合

## Representation as a Rule Set



If we leave at 10 AM and there are no cars stalled on the road, what will our commute time be?

*Re-representation as if-then rules:*  
disjunction of conjunctions of constraints on the attribute value instances

```
if hour == 8am
    commute time = long
else if hour == 9am
    if accident == yes
        commute time = long
    else
        commute time = medium
else if hour == 10am
    if stall == yes
        commute time = long
    else
        commute time = short
```

# 决策树学习举例

- Problem: decide whether to wait for a table at a restaurant.  
What **attributes** would you use?

Goal predicate: Will wait?

- Attributes
  1. **Alternate**: is there an alternative restaurant nearby?
  2. **Bar**: is there a comfortable bar area to wait in?
  3. **Fri/Sat**: is today Friday or Saturday?
  4. **Hungry**: are we hungry?
  5. **Patrons**: number of people in the restaurant (None, Some, Full)
  6. **Price**: price range (\$, \$\$, \$\$\$)
  7. **Raining**: is it raining outside?
  8. **Reservation**: have we made a reservation?
  9. **Type**: kind of restaurant (French, Italian, Thai, Burger)
  10. **WaitEstimate**: estimated waiting time (0-10, 10-30, 30-60, >60)

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels (also called targets or outcomes).
- E.g., situations where I will/won't wait for a table:

12 examples

6 +

6 -

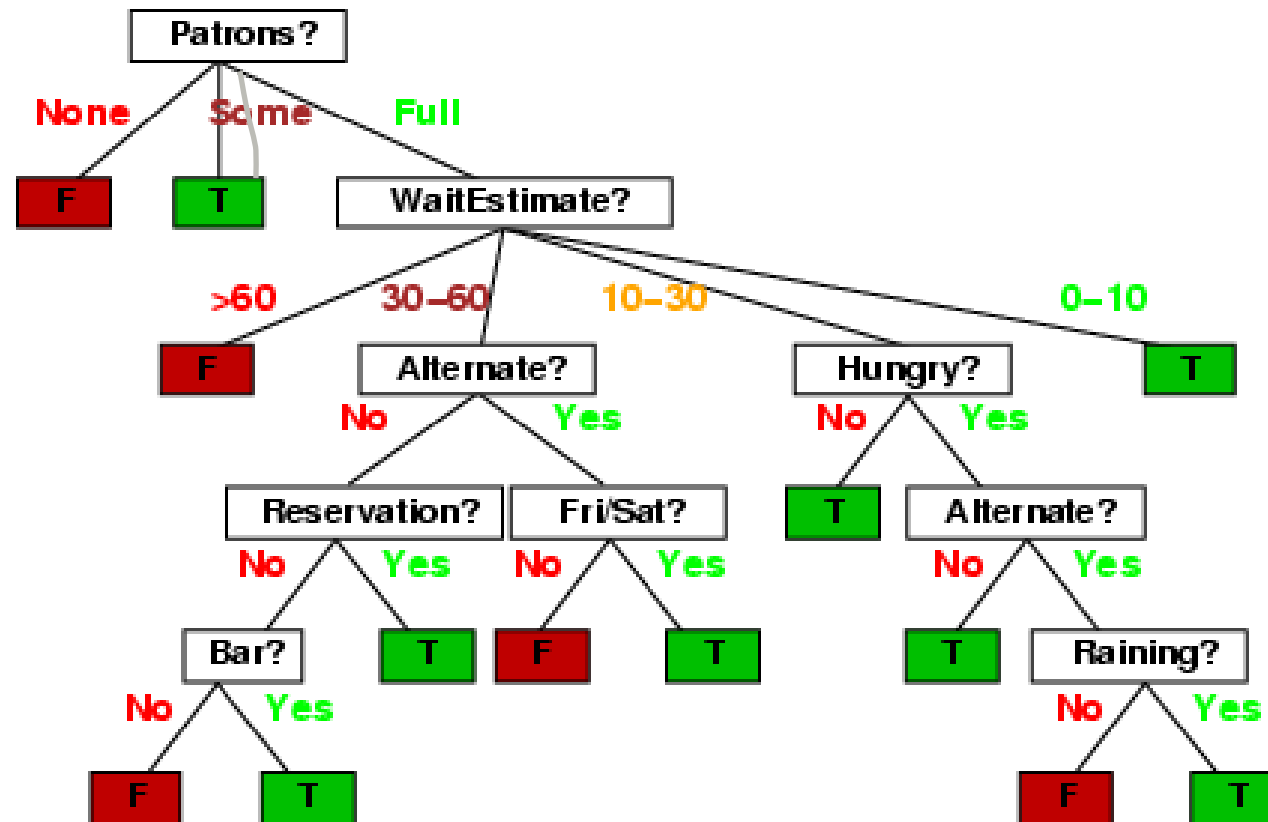
Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- Classification of examples is positive (T) or negative (F)



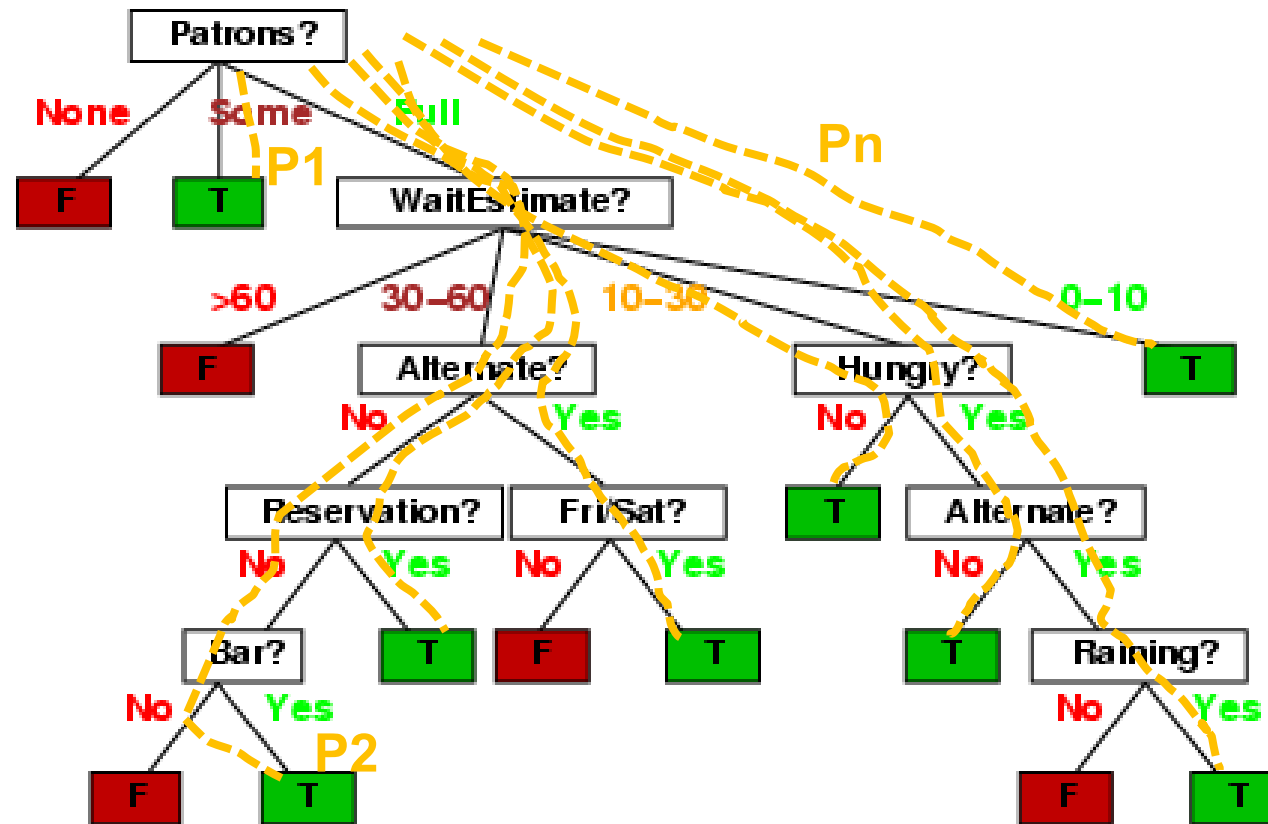
# 决策树学习举例

- One possible representation for hypotheses
- E.g., here is a tree for deciding whether to wait:



# 决策树学习举例

- One possible representation for hypotheses
- E.g., here is a tree for deciding whether to wait:



$$\text{WillWait}(s) \leftrightarrow (P1(s) \vee P2(s) \vee \dots \vee Pn(s))$$

# 决策树的表达能力

Any particular decision tree hypothesis for WillWait goal predicate can be seen as

a disjunction of a conjunction of tests,

i.e., an assertion of the form:

$$\forall s \text{ WillWait}(s) \leftrightarrow (P1(s) \vee P2(s) \vee \dots \vee Pn(s))$$

Where each condition  $P_i(s)$  is a conjunction of tests corresponding to the path from the root of the tree to a leaf with a positive outcome.

# 决策树的表达能力

For 10 Boolean attributes, there are  $2^{10}$  possible combinations of input values (features).

For each input combination, there are 2 possible outputs (0/1).

Input features		Output	
0 0 0 0 0 0 0 0 0 0	How many entries does this table have?  $2^{10}$	0/1	Decision trees can express any Boolean function
0 0 0 0 0 0 0 0 0 1		0/1	
0 0 0 0 0 0 0 0 1 0		0/1	
0 0 0 0 0 0 0 1 0 0		0/1	
...			
1 1 1 1 1 1 1 1 1 1		0/1	

# 决策树的表达能力

For 10 Boolean attributes, there are  $2^{10}$  possible combinations of input values (features).

For each input combination, there are 2 possible outputs (0/1).

Input features		Output	
0 0 0 0 0 0 0 0 0 0	How many entries does this table have?  $2^{10}$	0/1	Decision trees can express any Boolean function
0 0 0 0 0 0 0 0 0 1		0/1	
0 0 0 0 0 0 0 0 1 0		0/1	
0 0 0 0 0 0 0 1 0 0		0/1	
...			
1 1 1 1 1 1 1 1 1 1		0/1	

**Total number of distinct Boolean functions (and corresponding decision trees):** Since there are  $2^{10}$  possible input combinations, and each combination can yield one of two possible outputs (0 or 1), the total number of distinct Boolean functions is:

$$= 2^{2^{10}}$$

# 监督学习

## Supervised Learning

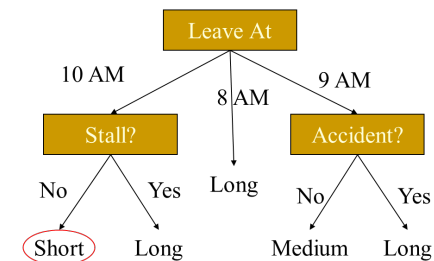
- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1,2,\dots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx f_{\theta}(x^{(i)})$$

a decision tree



- Function set  $\{f_{\theta}(x^{(i)})\}$  is called hypothesis
- Learning is referred to as updating the parameters  $\theta$  so that the prediction is close to the corresponding label

# 决策树学习算法

## Decision tree learning Algorithm

- Goal: Finding a decision tree that agrees with training set.

# 决策树学习算法

## Decision tree learning Algorithm

- Goal: Finding a decision tree that agrees with training set.

Could we construct a decision tree that has one path to a leaf for each instance, where the path tests sets each attribute value to the value of the instance?

What is the problem with this from a learning point of view?



# 决策树学习算法

## Decision tree learning Algorithm

- Goal: Finding a decision tree that agrees with training set.

Could we construct a decision tree that has one path to a leaf for each instance, where the path tests sets each attribute value to the value of the instance?

What is the problem with this from a learning point of view?

**Problem:** This approach would just memorize training instances. How to deal with new instances?

**It doesn't generalize!**

# 决策树学习算法

## Decision tree learning Algorithm

- The basic idea behind any decision tree algorithm is as follows:
  - Choose the **best attribute(s)** to split the remaining instances and make that attribute a decision node
  - Repeat this process recursively for each child
  - Stop when:
    - All the instances have the same target value
    - There are no more attributes
    - There are no more instances
    - Minimum sample size for a node (early stopping)
    - Maximum tree depth...

# ID3 启发式算法

## ID3 Heuristic algorithm

- How to determine the **best attribute**?

– ID3 splits attributes based on their entropy.  
Entropy is the measure of randomness.



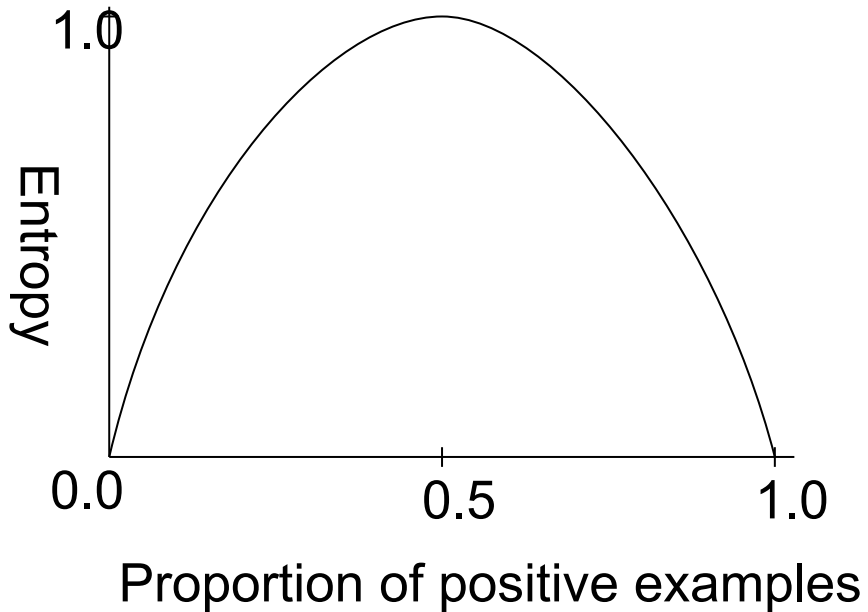
$p(\text{head})=0.5$   
 $p(\text{tail})=0.5$   
 $H=1$



$p(\text{head})=0.51$   
 $p(\text{tail})=0.49$   
 $H=0.9997$

# 熵函数

## Entropy Function



For a collection  $D$  having positive and negative examples, entropy is given as:

$$Entropy(D) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

$p$  - # positive examples  
 $n$  - # negative examples

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels (also called targets or outcomes).
- E.g., sit **What's the entropy of this collection of examples?**

12 examples

6 +

6 -

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- Classification of examples is positive (T) or negative (F)

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels (also called targets or outcomes).
- E.g., sit **What's the entropy of this collection of examples?**

12 examples

$$\begin{aligned}
 &6 + \\
 &6 - \\
 &p = n = 6; \\
 &\frac{P}{P + n} = \frac{n}{P + n} \\
 &= 0.5
 \end{aligned}$$

$$\begin{aligned}
 Entropy(D) &= -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n} \\
 &= -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = 1
 \end{aligned}$$

$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- Classification of examples is positive (T) or negative (F)

# 熵的通用计算公式

## General Formula for Entropy

- Calculation of entropy

$$Entropy(D) = - \sum_{k=1}^{|y|} p_k \log_2 p_k$$

- $D$  = the set of examples
- $p_k$  = the portion of examples in  $D$  that belong to class  $k$ .
- $|y|$  = the number of distinct classes (i.e. the size of the range of the target value) .

It is applicable to both binary and multiclass classification problems.

# ID3基于信息增益来选择属性

## Choosing an attribute: Information Gain

- Intuition: the **best attribute** is the attribute that **reduces the entropy (the uncertainty) the most**.
- the information gain of a given attribute  $a$  relative to a collection of examples  $D$  is defined as:

$$Gain(D, a) = Entropy(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} Entropy(D^v)$$

$D^v$  is the subset of  $D$  where the attribute  $a$  takes on value  $V$ ,  
And  $V$  is the number of distinct values that the attribute  $a$  can take.



# ID3算法构建决策树

## Decision Tree Building: ID3 Algorithm

Start from the root node containing all data

- For each node, calculate **the information gain** of all available features
- Choose the feature with the highest information gain
- Split the data of at the node based on the selected feature (the feature is not reused in subsequent splits)
- Repeat the above steps recursively for each resulting node, until one of the following stopping conditions is met:
  - All instances at the node have the same target value.
  - No remaining features to split.
  - Maximum tree depth or minimum sample size for a node is reached.

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels (also called targets or outcomes).
- E.g., situations where I will/won't wait for a table:

12 examples

6 +

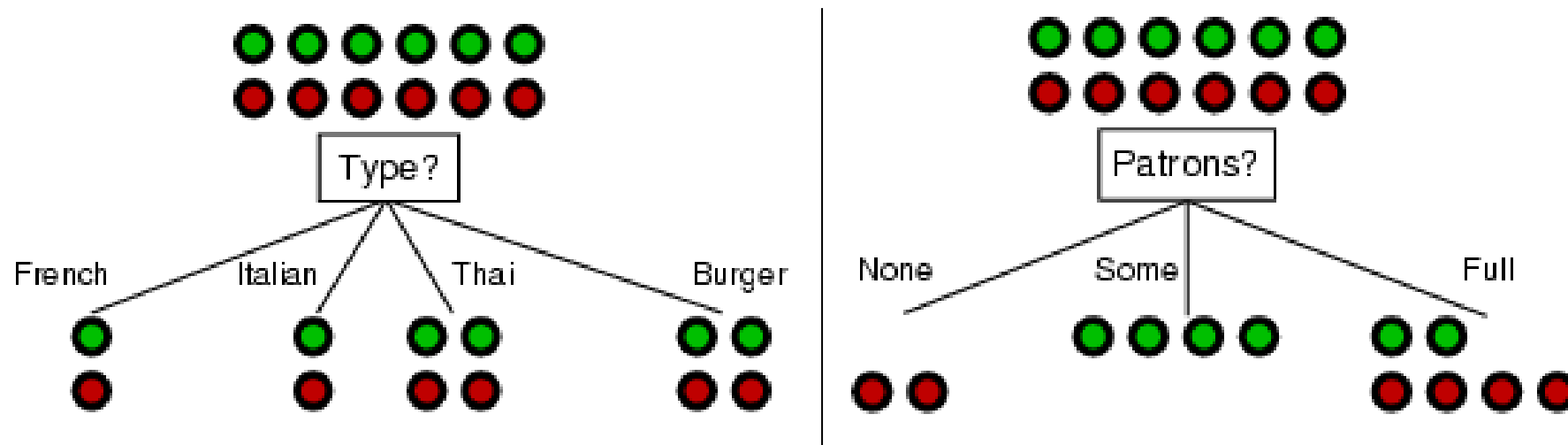
6 -

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- Classification of examples is positive (T) or negative (F)

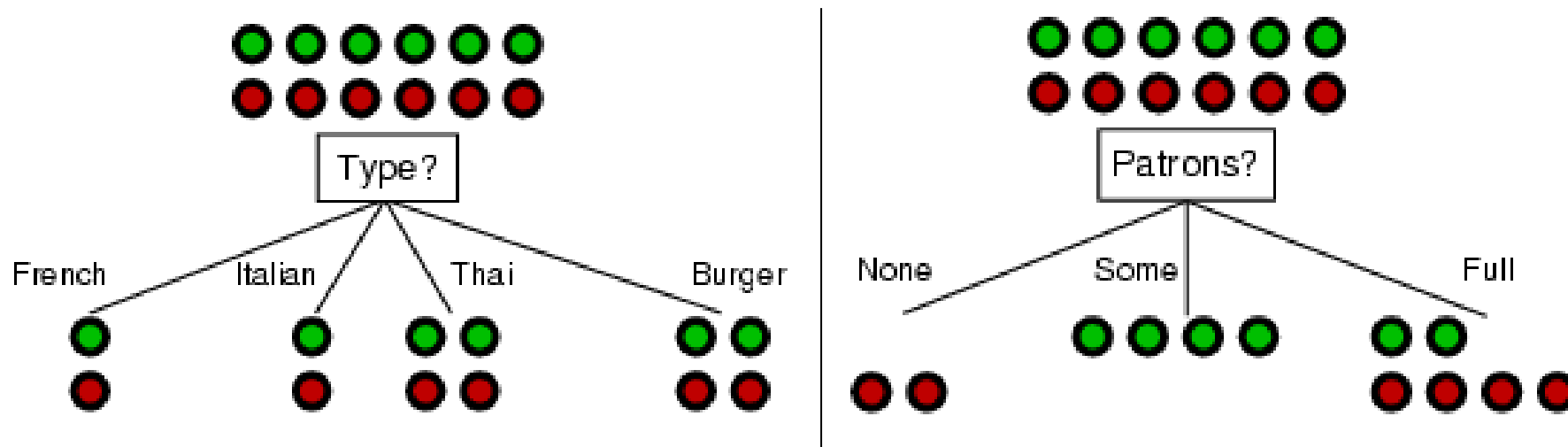
# 决策树学习举例

Which one should we pick?



# 决策树学习举例

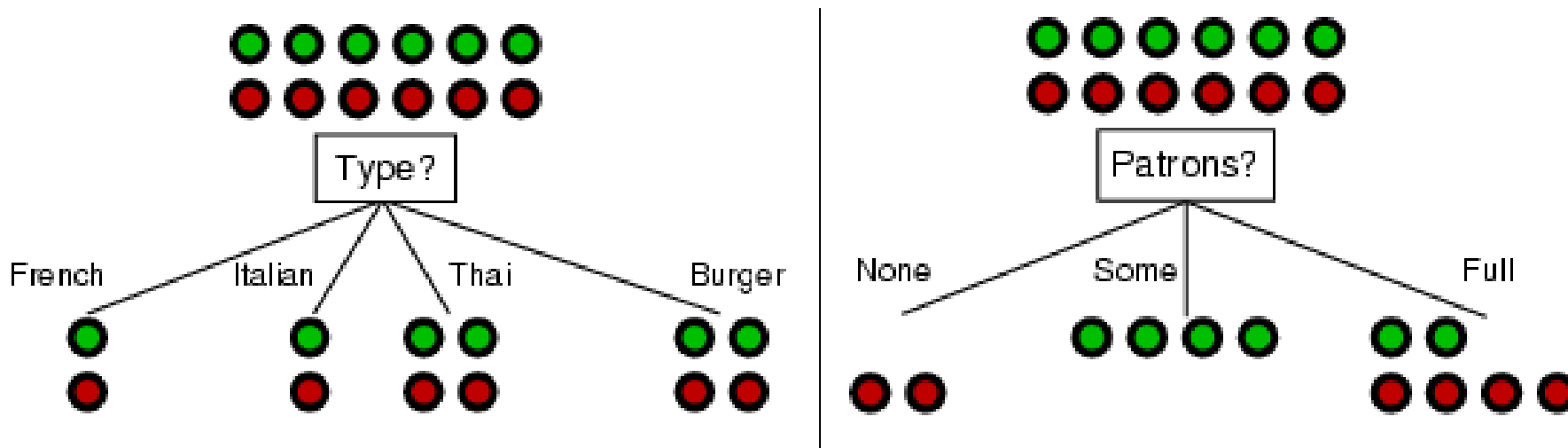
Goal: Choose the feature with the highest information gain



A **perfect attribute** would ideally divide the examples into sub-sets that are **all positive or all negative**...  
i.e. maximum information gain.

# 决策树学习举例

$$Gain(D, a) = Entropy(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} Entropy(D)$$

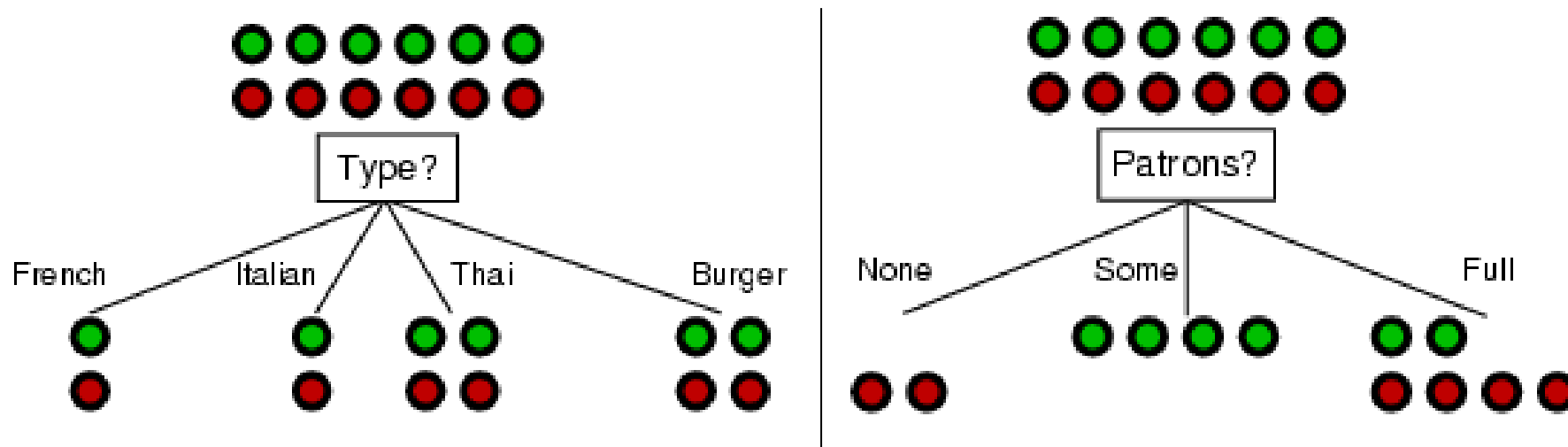


$$Gain(D, Type) = 1 - \left[ \frac{2}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) \right] = 0$$

$$Gain(D, Patrons) = 1 - \left[ \frac{2}{12} Entropy(0, 1) + \frac{4}{12} Entropy(1, 0) + \frac{6}{12} Entropy\left(\frac{2}{6}, \frac{4}{6}\right) \right] = 0.541$$

# 决策树学习举例

**Patrons** has the highest 'information gain' of all attributes and so is chosen as the root.



$$Gain(D, Type) = 1 - \left[ \frac{2}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} Entropy\left(\frac{1}{2}, \frac{1}{2}\right) \right] = 0$$

$$Gain(D, Patrons) = 1 - \left[ \frac{2}{12} Entropy(0, 1) + \frac{4}{12} Entropy(1, 0) + \frac{6}{12} Entropy\left(\frac{2}{6}, \frac{4}{6}\right) \right] = 0.541$$

# 奥卡姆剃刀原则

## Principle of Occam's razor

Among competing hypotheses, the one with the fewest assumptions should be selected.

Recall the function set  $\{f_{\theta}(x^{(i)})\}$  is called hypothesis space

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N L(y^{(i)}, f_{\theta}(x^{(i)})) + \lambda \Omega(\theta)$$

Original loss      Penalty on assumptions

Inductive Bias  
prefer/search bias

Structural Risk Minimization, SRM

# ID3的归纳偏置

## Inductive Bias in ID3

ID3 algorithm **prefers shorter trees** by selecting attributes with higher information gain near the root. This helps in reducing the overall complexity of the tree.

- Occam's razor: prefer the shortest hypothesis that fits the data
- The bias in ID3 is a preference for some hypotheses(**prefer/search bias**), rather than a restriction of hypothesis space  $H$ . The algorithm prioritizes certain types of models (e.g., shorter trees), but other hypotheses are still possible.



# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels (also called targets or outcomes).
- E.g., situations where I will/won't wait for a table:

12 examples

6 +

6 -

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-60	F
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10	T
$X_4$	T	F	T	T	Full	\$	F	F	Thai	10-30	T
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10	F
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60	F
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10	F
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60	T

- Classification of examples is positive (T) or negative (F)

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels called targets or outcomes).

- E.g., situations where I will/won't wait for a table:

12 examples

6 +

6 -

Example	Attributes									
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-50
$X_3$	F	T	F	F	Some	\$	F	F	Burger	>60
$X_4$	T	F	T	T	Full	\$	F	F	Thai	>60
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30
$X_{11}$	F	F	F	F	None	\$	F	F	Thai	0-10
$X_{12}$	T	T	T	T	Full	\$	F	F	Burger	30-60

Number

1

2

3

4

5

6

7

8

9

10

11

12

- Classification of examples is positive (T) or negative (F)

If add an attribute—'number'

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels called targets or outcomes).
- E.g., situations where I will/won't wait for a table:

12 examples

6 +

6 -

Example	Attributes									
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>
$X_1$	T	F	F	T	Some	\$\$\$	F	T	French	0-10
$X_2$	T	F	F	T	Full	\$	F	F	Thai	30-50
$X_3$	F	T	F	F	Some	\$	F	F	Burger	0-10
$X_4$	T	F	T	T	Full	\$	F	F	Thai	0-10
$X_5$	T	F	T	F	Full	\$\$\$	F	T	French	>60
$X_6$	F	T	F	T	Some	\$\$	T	T	Italian	0-10
$X_7$	F	T	F	F	None	\$	T	F	Burger	0-10
$X_8$	F	F	F	T	Some	\$\$	T	T	Thai	0-10
$X_9$	F	T	T	F	Full	\$	T	F	Burger	>60
$X_{10}$	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30

Number

1

2

3

4

5

6

7

8

9

10

Number has the highest 'information gain' of all attributes and so is chosen as the root.

- Classification

If add an attribute—'number'

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels called targets or outcomes).

- E.g., situations where I will/won't wait for a table:

12 exa

6

6

$$\begin{aligned}
 & \text{Gain}(D, \text{number}) \\
 &= \text{Entropy}(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Entropy}(D) \\
 &= 1 - \left[ \frac{6}{12} \text{Entropy}(1,0) + \frac{6}{12} \text{Entropy}(0,1) \right] \\
 &= 1 - 0 = 1
 \end{aligned}$$

Number

1

2

3

4

5

6

7

8

9

10

es

- Classification: 'Number' has the highest 'information gain' of all attributes and so is chosen as the root.

If add an attribute—'number'

# ID3基于信息增益来选择属性

## Choosing an attribute: Information Gain

- Intuition: the **best attribute** is the attribute that **reduces the entropy (the uncertainty) the most**.
- the information gain of a given attribute  $a$  relative to a collection of examples  $D$  is defined as:

$$Gain(D, a) = Entropy(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} Entropy(D^v)$$

$D^v$  is the subset of  $D$  where the attribute  $a$  takes on value  $V$ ,  
And  $V$  is the number of distinct values that the attribute  $a$  can take.

# 信息增益率和C4.5算法

## Information Gain Ratio and C4.5

$$Gain\_ratio(D, a) = \frac{Gain(D, a)}{IV(a)}$$
$$a_* = \operatorname{argmax}_{a \in A} Gain\_ratio(D, a)$$

$IV(a)$  = intrinsic value, the greater the possible number of attributes is, the greater the value is.

$$IV(a) = - \sum_{v=1}^V \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}$$

$D^v$  is the subset of  $D$  where the attribute  $a$  takes on value  $V$ , and  $V$  is the number of distinct values that the attribute  $a$  can take.

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels called targets or outcomes).

- E.g., situations where I will/won't wait for a table:

12 exa  
6  
6

$$\begin{aligned}
 & \text{Gain}(D, \text{number}) \\
 &= \text{Entropy}(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Entropy}(D) \\
 &= 1 - \left[ \frac{6}{12} \text{Entropy}(1,0) + \frac{6}{12} \text{Entropy}(0,1) \right] \\
 &= 1 - 0 = 1
 \end{aligned}$$

Number  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10

- Classification  
Number has the highest 'information gain' of all attributes and so is chosen as the root.

If add an attribute—'number'

# 决策树学习举例

- Examples (also called instances, observations, or samples) are described by features (also called attributes, or variables) and labels called targets or outcomes).

- E.g.,  $Gain\_ratio(D, number) = \frac{Gain(D, a)}{IV(a)}$

12 exar

6 H

6 -

$$\begin{aligned}
 &= \frac{Gain(D, a)}{-\sum_{v=1}^V \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}} \\
 &= \frac{1}{-12 \frac{1}{12} \log_2 \frac{1}{12}} \\
 &= \frac{1}{\log_2 12}
 \end{aligned}$$

Est
0-10
30-50
> 60
0-10
0-10
0-10
> 60
10-30

Number

1

2

3

4

5

6

7

8

9

10

es

Number has the highest 'information gain' of all

- Classification attributes and so is chosen as the root.

If add an attribute—'number'



# 基尼指数和CART算法

## Gini index and CART

$$Gini\_index(D, a = v) = \frac{|D^l|}{|D|} Gini(D^l) + \frac{|D^r|}{|D|} Gini(D^r)$$

$$a_*, v_* = \underset{a \in A}{argmin} Gini_{index}(D, a = v)$$

$D^l$  and  $D^r$  are the left and right subsets of  $D$  after splitting by  $a = v$ .

$Gini(D)$  reflects the probability that two samples randomly selected from  $D$  belong to different classes.

$$Gini(D) = \sum_{k=1}^{|y|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|y|} p_k^2$$

CART

Breiman et al., 1984

Classification and Regression tree

Where  $p_k$  is the proportion of samples in class  $k$  in the dataset  $D$

# 基尼指数和CART算法

## Gini index and CART

Binary  
Tree

$$Gini\_index(D, a = v) = \frac{|D^l|}{|D|} Gini(D^l) + \frac{|D^r|}{|D|} Gini(D^r)$$

$$a_*, v_* = \underset{a \in A}{argmin} Gini_{index}(D, a = v)$$

$D^l$  and  $D^r$  are the left and right subsets of  $D$  after splitting by  $a = v$ .

$Gini(D)$  reflects the probability that two samples randomly selected from  $D$  belong to different classes.

$$Gini(D) = \sum_{k=1}^{|y|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|y|} p_k^2$$

CART

Breiman et al., 1984

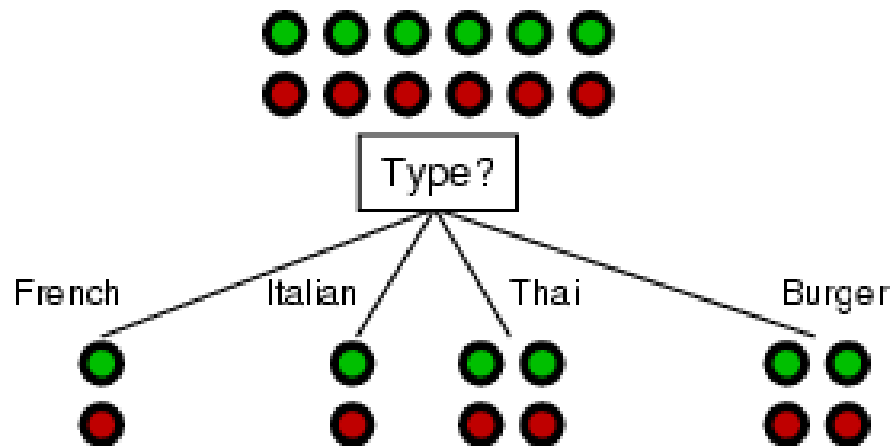
Classification and Regression tree

Where  $p_k$  is the proportion of samples in class  $k$  in the dataset  $D$

# 决策树学习举例

## Learning decision trees: An example

$$Gini\_index(D, a = v) = \frac{|D^l|}{|D|} Gini(D^l) + \frac{|D^r|}{|D|} Gini(D^r)$$

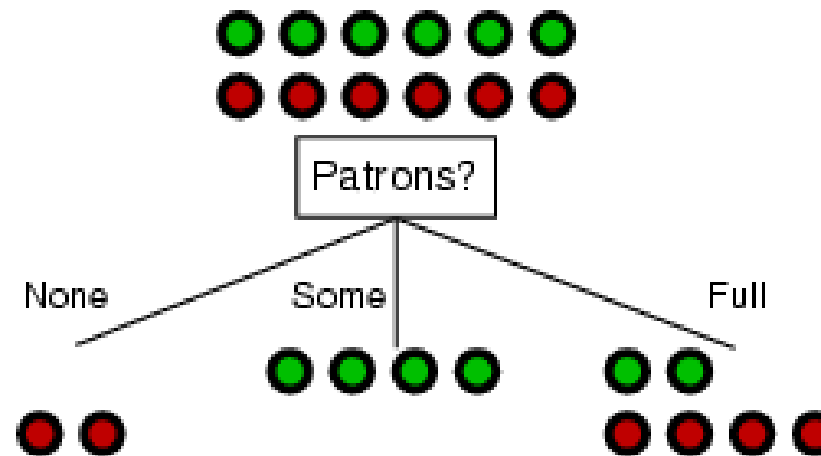


$Gini\_index(D, Type = French) =$

$Gini\_index(D, Type = Italian) =$

$Gini\_index(D, Type = Thai) =$

$Gini\_index(D, Type = Burger) =$



$Gini\_index(D, Patrons = None) =$

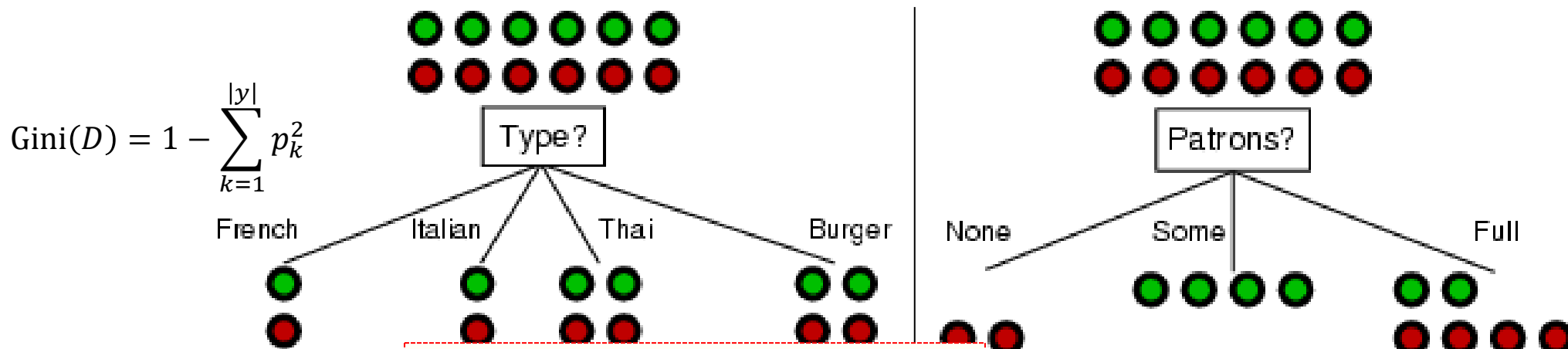
$Gini\_index(D, Patrons = some) =$

$Gini\_index(D, Patrons = Full) =$

# 决策树学习举例

## Learning decision trees: An example

$$Gini\_index(D, a = v) = \frac{|D^l|}{|D|} Gini(D^l) + \frac{|D^r|}{|D|} Gini(D^r)$$

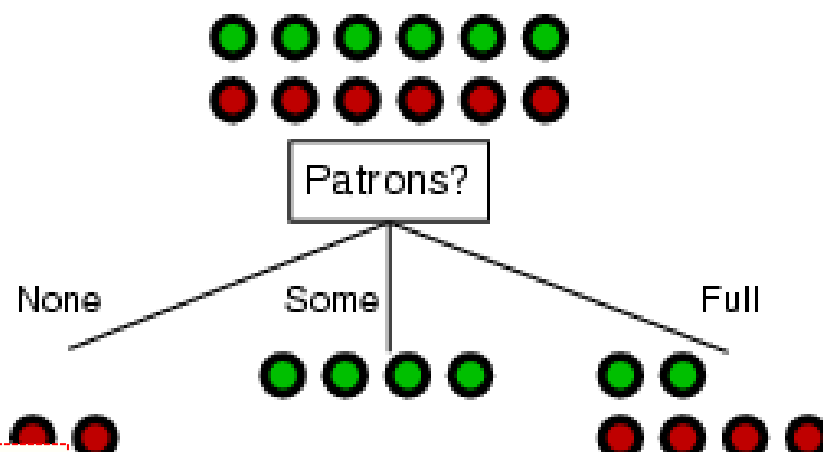


$$Gini\_index(D, Type = French) = \frac{2}{12} \left( 1 - \frac{1^2}{2} - \frac{1^2}{2} \right) + \frac{10}{12} \left( 1 - \frac{5^2}{10} - \frac{5^2}{10} \right) = 0.5$$

$$Gini\_index(D, Type = Italian) = \frac{2}{12} \left( 1 - \frac{1^2}{2} - \frac{1^2}{2} \right) + \frac{10}{12} \left( 1 - \frac{5^2}{10} - \frac{5^2}{10} \right) = 0.5$$

$$Gini\_index(D, Type = Thai) = \frac{4}{12} \left( 1 - \frac{2^2}{4} - \frac{2^2}{4} \right) + \frac{8}{12} \left( 1 - \frac{4^2}{8} - \frac{4^2}{8} \right) = 0.5$$

$$Gini\_index(D, Type = Burger) =$$



$$Gini\_index(D, Patrons = None) = \frac{2}{12} 0 + \frac{10}{12} \left( 1 - \frac{5^2}{10} - \frac{5^2}{10} \right) = 0.42$$

$$Gini\_index(D, Patrons = some) = \frac{4}{12} 0 + \frac{8}{12} \left( 1 - \frac{2^2}{8} - \frac{6^2}{8} \right)$$

$$Gini\_index(D, Patrons = Full) = 0.25$$

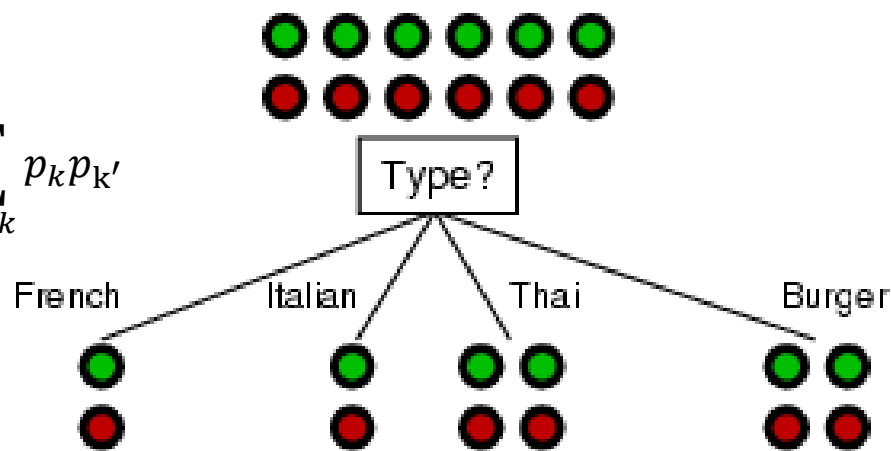
$$\frac{6}{12} \left( 1 - \frac{2^2}{6} - \frac{4^2}{6} \right) + \frac{6}{12} \left( 1 - \frac{2^2}{8} - \frac{6^2}{8} \right) = 0.44$$

# 决策树学习举例

## Learning decision trees: An example

$$Gini\_index(D, a = v) = \frac{|D^l|}{|D|} Gini(D^l) + \frac{|D^r|}{|D|} Gini(D^r)$$

$$Gini(D) = \sum_{k=1}^{|y|} \sum_{k' \neq k} p_k p_{k'}$$

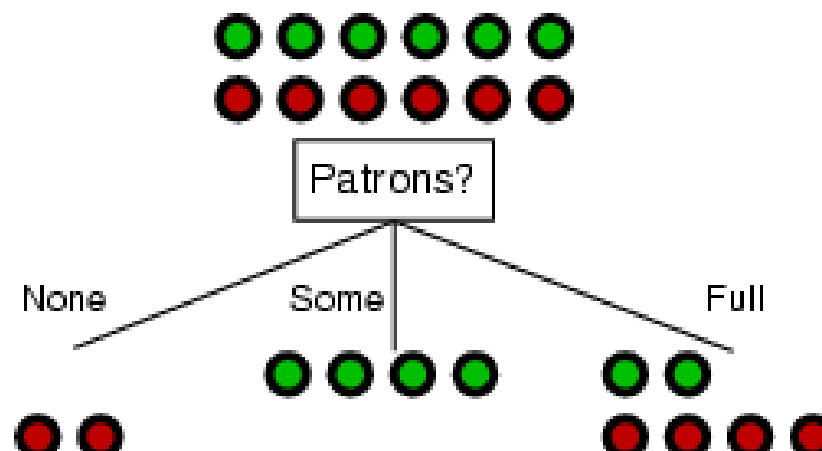


$$Gini\_index(D, Type = French) = \frac{2}{12} \left( 2 \frac{1}{2} \frac{1}{2} \right) + \frac{10}{12} \left( 2 \frac{5}{10} \frac{5}{10} \right)$$

$$Gini\_index(D, Type = Italian) = 0.5$$

$$Gini\_index(D, Type = Thai) = \frac{4}{12} \left( 2 \frac{2}{4} \frac{2}{4} \right)$$

$$Gini\_index(D, Type = Burger) = \frac{8}{12} \left( 2 \frac{4}{8} \frac{4}{8} \right) = 0.5$$



$$Gini\_index(D, Patrons = None) = \frac{2}{12} 0 + \frac{10}{12} \left( 2 \frac{5}{10} \frac{5}{10} \right) = 0.42$$

$$Gini\_index(D, Patrons = some) =$$

$$Gini\_index(D, Patrons = Full) = \frac{4}{12} 0 + \frac{8}{12} \left( 2 \frac{2}{8} \frac{6}{8} \right) = 0.25$$

$$\frac{6}{12} \left( 2 \frac{2}{6} \frac{4}{6} \right) + \frac{6}{12} \left( 2 \frac{2}{6} \frac{4}{6} \right) = 0.44$$

# 均方误差和CART算法

## Mean Squared Error (MSE) and CART

$$a_*, v_* = \underset{a \in A}{\operatorname{argmin}} \left[ \min_{c^l} \sum_{x^i \in D^l} (y^i - c^l)^2 + \min_{c^r} \sum_{x^i \in D^r} (y^i - c^r)^2 \right]$$

$a_*, v_*$  are the optimal attribute and split point that **minimize the total Mean Squared Error (MSE)**.

$D^l$  and  $D^r$  are the left and right subsets of  $D$  after splitting on attribute  $a$  with a threshold  $v$ .

$c^l$  and  $c^r$  are predicted values for the left and right subsets, respectively.

$y^i$  represents the actual target value for the  $i$ -th sample.

The CART can be applied to regression task!

# 均方误差和CART算法

## Mean Squared Error (MSE) and CART

$$a_*, v_* = \underset{a \in A}{\operatorname{argmin}} \left[ \min_{c^l} \sum_{x^i \in D^l} (y^i - c^l)^2 + \min_{c^r} \sum_{x^i \in D^r} (y^i - c^r)^2 \right]$$

$c^l$  and  $c^r$  are predicted values for the left and right subsets, respectively.  
 $y^i$  represents the actual target value for the  $i$ -th sample.

To minimize the Mean Squared Error (MSE), the predicted value is typically the mean of the target values in the subset.

$$\frac{\partial \sum_{x^i \in D^l} (y^i - c^l)^2}{\partial c^l} = -2 \sum_{x^i \in D^l} (y^i - c^l) = 0$$



$$c_l = \frac{1}{N^l} \sum_{x^i \in D^l} y^i$$

$$\frac{\partial \sum_{x^i \in D^r} (y^i - c^r)^2}{\partial c^r} = -2 \sum_{x^i \in D^r} (y^i - c^r) = 0$$



$$c_r = \frac{1}{N^r} \sum_{x^i \in D^r} y^i$$

# 均方误差和CART算法

## MSE and CART

$$a_*, v_* = \underset{a \in A}{\operatorname{argmin}} \left[ \min_{c^l} \sum_{x^i \in D^l} (y^i - c^l)^2 + \min_{c^r} \sum_{x^i \in D^r} (y^i - c^r)^2 \right]$$

$$c_l = \frac{1}{N^l} \sum_{x^i \in D^l} y^i, \quad c_r = \frac{1}{N^r} \sum_{x^i \in D^r} y^i$$

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
mse	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74

$$s = 1.5, R_1 = \{1\}, R_2 = \{2, 3, 4, 5, 6, 7, 8, 9, 10\},$$

$$c_1 = 5.56, c_2 = 7.5$$





# 均方误差和CART算法

## MSE and CART

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05


v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
$c^l$	5.56								
$c^r$	7.5								
MSE	15.72								

$$=(5.7+5.91+6.4+6.8+7.05+8.9+8.7+9+9.05)/9$$

$$\sum_{x^i \in D^l} (y^i - c^l)^2 + \sum_{x^i \in D^r} (y^i - c^r)^2$$


# 均方误差和CART算法

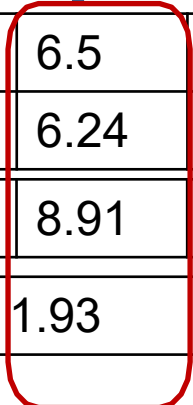
## MSE and CART

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
										
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24	6.62	6.88	7.11	
$c^r$	7.5	7.73	7.99	8.25	8.54	8.91	8.92	9.03	9.05	
MSE	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74	

# 均方误差和CART算法


## MSE and CART

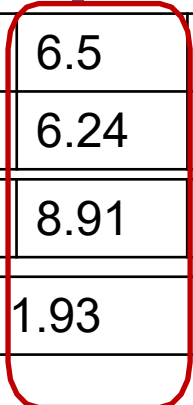
x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
										
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24	6.62	6.88	7.11	
$c^r$	7.5	7.73	7.99	8.25	8.54	8.91	8.92	9.03	9.05	
MSE	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74	



# 均方误差和CART算法

## MSE and CART

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
										
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24				
$c^r$	6.37	6.54	6.75	6.93	7.05	8.91				
MSE	1.31	0.75	0.28	0.44	1.01	1.93				



# 均方误差和CART算法


## MSE and CART

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24				
$c^r$	6.37									
MSE	1.31	0.11	0.09	0.11	0.11	0.11				

$$=(5.7+5.91+6.4+6.8+7.05)/5$$

# 均方误差和CART算法

## MSE and CART

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
										
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24				
$c^r$	6.37	6.54	6.75	6.93	7.05	8.91				
MSE	1.31	0.75	0.28	0.44	1.01	1.93				

# 均方误差和CART算法

## MSE and CART

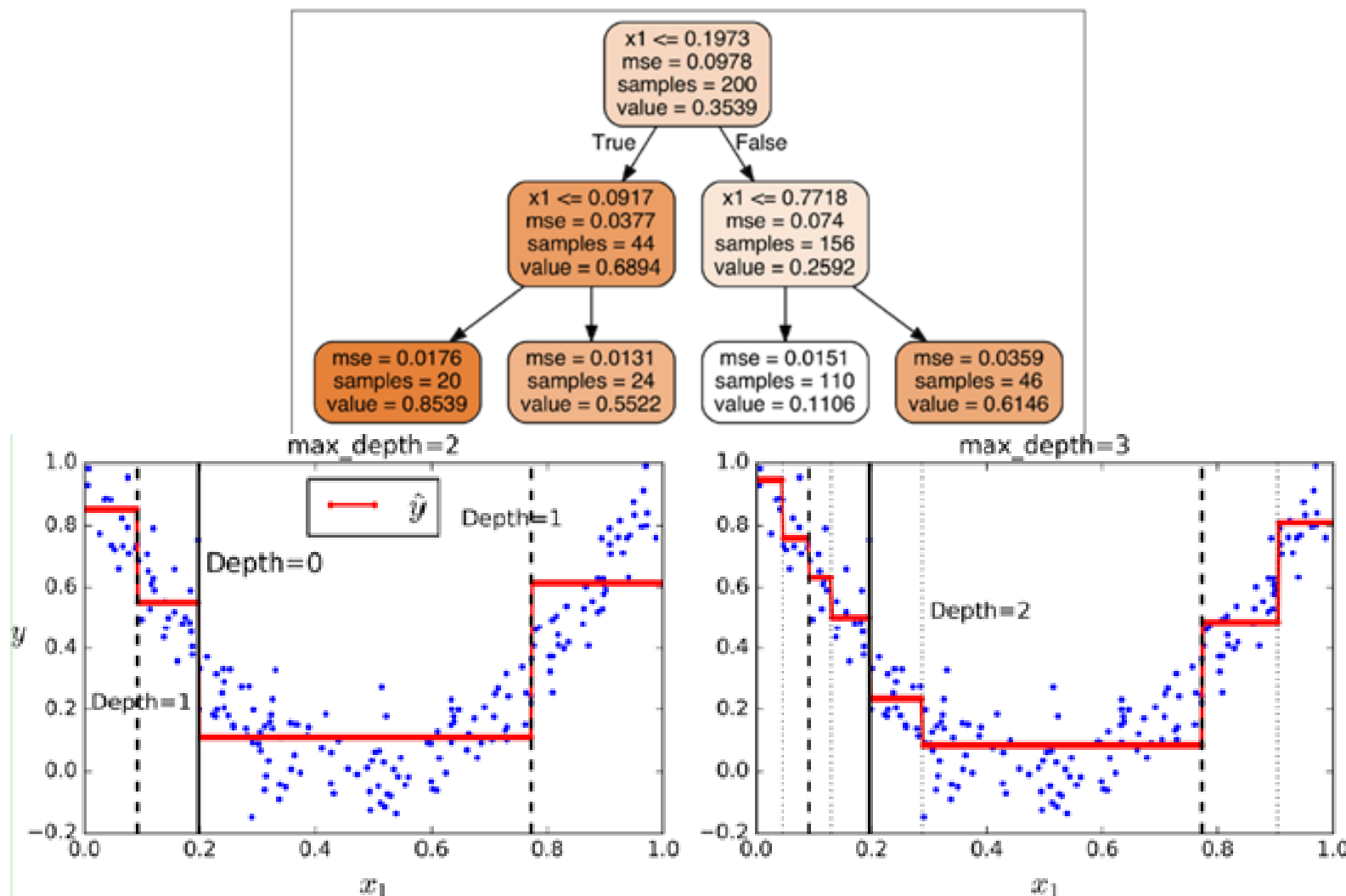
x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05
v	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5	
$c^l$	5.56	5.63	5.72	5.89	6.07	6.24				
$c^r$	6.37	6.54	6.75	6.93	7.05	8.91				
MSE	1.31	0.75	0.28	0.44	1.01	1.93				

$$T = \begin{cases} 5.72 & x \leq 3.5 \\ 6.75 & 3.5 < x \leq 6.5 \\ 8.91 & x > 6.5 \end{cases}$$



# CART算法

## Classification and Regression Tree



The CART can be applied to regression task!

# 决策树经典算法小结

## Summary of DT algorithms

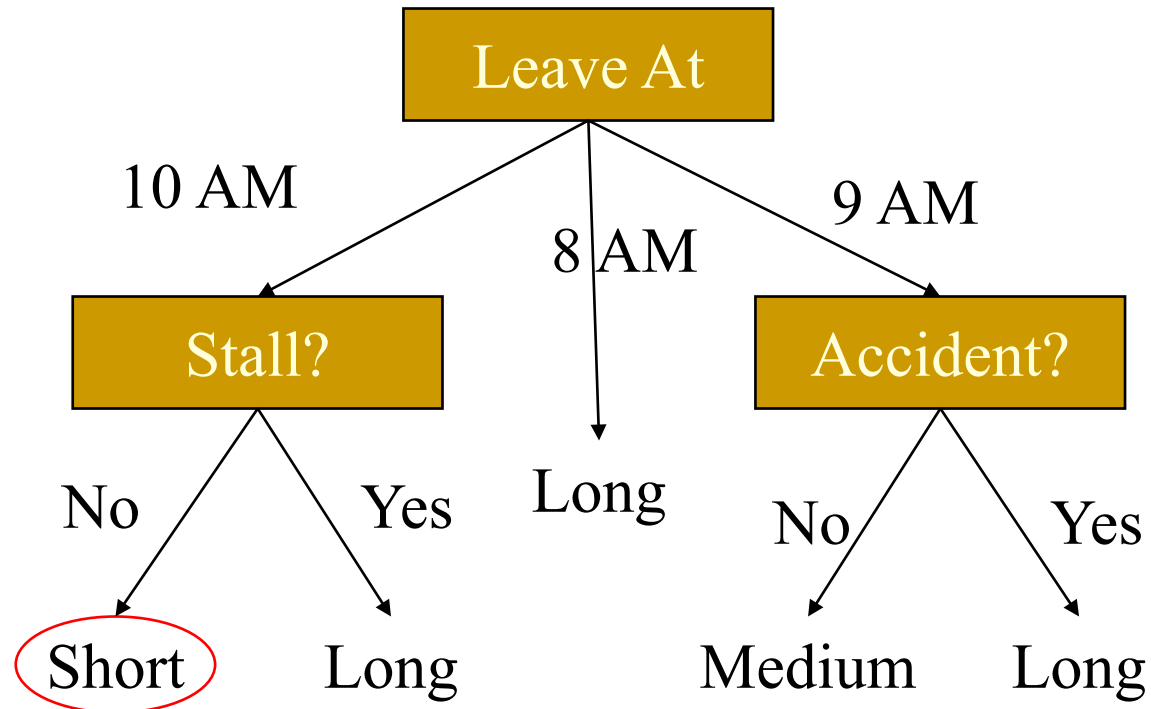
	support task	criterion structure	tree structure	continuous attribute value	missing attribute value	pruning	attribute multiple use
<b>ID3</b>	classification	Gain information	multiple Tree	No support	No support	No support	No support
<b>C4.5</b>	classification	Gain information rate	multiple Tree	support	support	support	No support
<b>CART</b>	Classification; regression	Gini index; mean square error	binary tree	support	support	support	support

# 连续值问题

## Continuous attribute Problem

- Consider the attribute commute time, If we broke down leave time to the minute, we might get this:

8:00 (L), 8:02 (L), 8:07 (M), 9:00 (S), 9:20 (S), 9:25 (S), 10:00 (S), 10:02 (M)



# 连续值问题

## Continuous attribute Problem

- Consider the attribute commute time, If we broke down leave time to the minute, we might get this:

8:00 (L), 8:02 (L), | 8:07 (M), | 9:00 (S), 9:20 (S), 9:25 (S), 10:00 (S), | 10:02 (M)

cut points

cut points

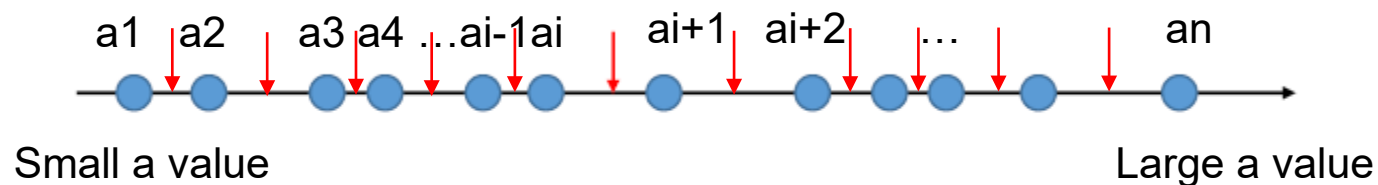
cut points

discretization

Since entropy is very low for each branch, we have  $n$  branches with  $n$  leaves. This would not be helpful for predictive modeling.

# 连续值离散化

## Continuous attribute discretization



Calculate candidate cut points

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leq i \leq n - 1 \right\}$$

the best partition point(s) are selected according to

$$T_a^* = \underset{T_a}{\operatorname{argmin}} \operatorname{Gini\_index} \quad (\text{CART})$$

$$T_a^* = \underset{T_a}{\operatorname{argmax}} \operatorname{Gain\_ratio} \quad (\text{C4.5})$$

# 缺失值处理

## Strategies for missing attribute values

$\tilde{D}$  — the subset of  $D$  in which samples that have no missing values in the attribute  $a$   
 $\tilde{D}^v$  — the subset of  $\tilde{D}$  in which samples that have  $v$  values in the attribute  $a$   
 $\tilde{D}_k$  — the subset of  $\tilde{D}$  in which samples belonging to a class  $k$   
 $w_x$  — the weight to each sample  $x$

Q1: 如何在属性缺失的情况下进行划分属性选择?

Calculate Gain according to

$$\begin{aligned}\text{Gain}(D, a) &= \rho \times \text{Gain}(\tilde{D}, a) \\ &= \rho \times \left( \text{Ent}(\tilde{D}) - \sum_{v=1}^V \tilde{r}_v \text{Ent}(\tilde{D}^v) \right)\end{aligned}$$

$$\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x}$$

$$\tilde{p}_k = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq k \leq |\mathcal{Y}|)$$

$$\tilde{r}_v = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq v \leq V)$$

$$\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k = 1, \sum_{v=1}^V \tilde{r}_v = 1.$$

$$\text{Ent}(\tilde{D}) = - \sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k$$

Q2: 如何在属性缺失的情况下进行划分属性选择?

C4.5:  
Probability weights

Assign  $x$ (with missing attribute  $a$ ) to each of the possible value  
With a probability  $\tilde{r}_v \cdot w_x$

# 缺失值处理

## Strategies for missing attribute values

$\tilde{D}$  — the subset of  $D$  in which samples that have no missing values in the attribute  $a$   
 $\tilde{D}^v$  — the subset of  $\tilde{D}$  in which samples that have  $v$  values in the attribute  $a$   
 $\tilde{D}_k$  — the subset of  $\tilde{D}$  in which samples belonging to a class  $k$   
 $w_x$  — the weight to each sample  $x$

A1: 根据缺失率的大小  
对信息增益率进行打折

$$\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x}$$

考虑样本权重后的  
具有完整属性值的  
样本比例

$$\tilde{p}_k = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq k \leq |\mathcal{Y}|)$$

具有完整属性值的类  
别  $k$  的加权样本比例

$$\tilde{r}_v = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq v \leq V)$$

具有特定属性值  $v$  的  
加权样本比例

$$\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k = 1, \sum_{v=1}^V \tilde{r}_v = 1.$$

Calculate Gain according to

$$\begin{aligned} \text{Gain}(D, a) &= \rho \times \text{Gain}(\tilde{D}, a) \\ &= \rho \times \left( \text{Ent}(\tilde{D}) - \sum_{v=1}^V \tilde{r}_v \text{Ent}(\tilde{D}^v) \right) \end{aligned}$$

$$\text{Ent}(\tilde{D}) = - \sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k$$

Assign  $x$ (with missing attribute  $a$ ) to  
each of the possible value  
With a probability  $\tilde{r}_v \cdot w_x$

A2: 将缺失属性的样本按  
属性值的分布进行分配

C4.5:  
Probability weights

# 缺失值处理

## Strategies for missing attribute values

ID	A	B	C	
1	T	NaN	F	Y
2	T	F	NaN	N
3	F	F	F	Y
4	F	T	T	N

ID	A	C
1	T	F
3	F	F
4	F	T

If A=T : C=F  
else : C=T

error:30%

ID	B	C
3	F	F
4	T	T

If B=F : C=F  
else : C=T

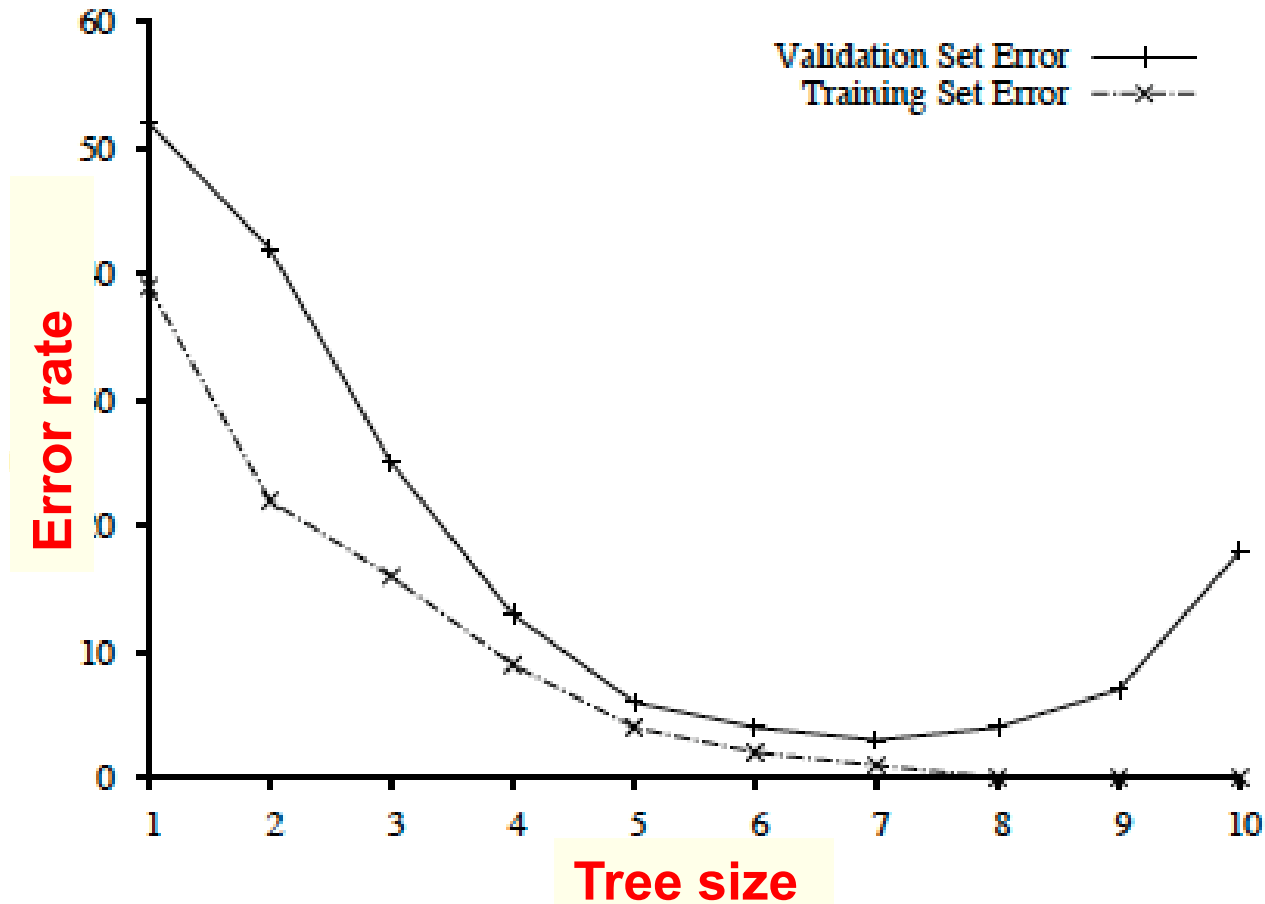
error:0%

Surrogate Variables order: B<A

CART:  
surrogate splits

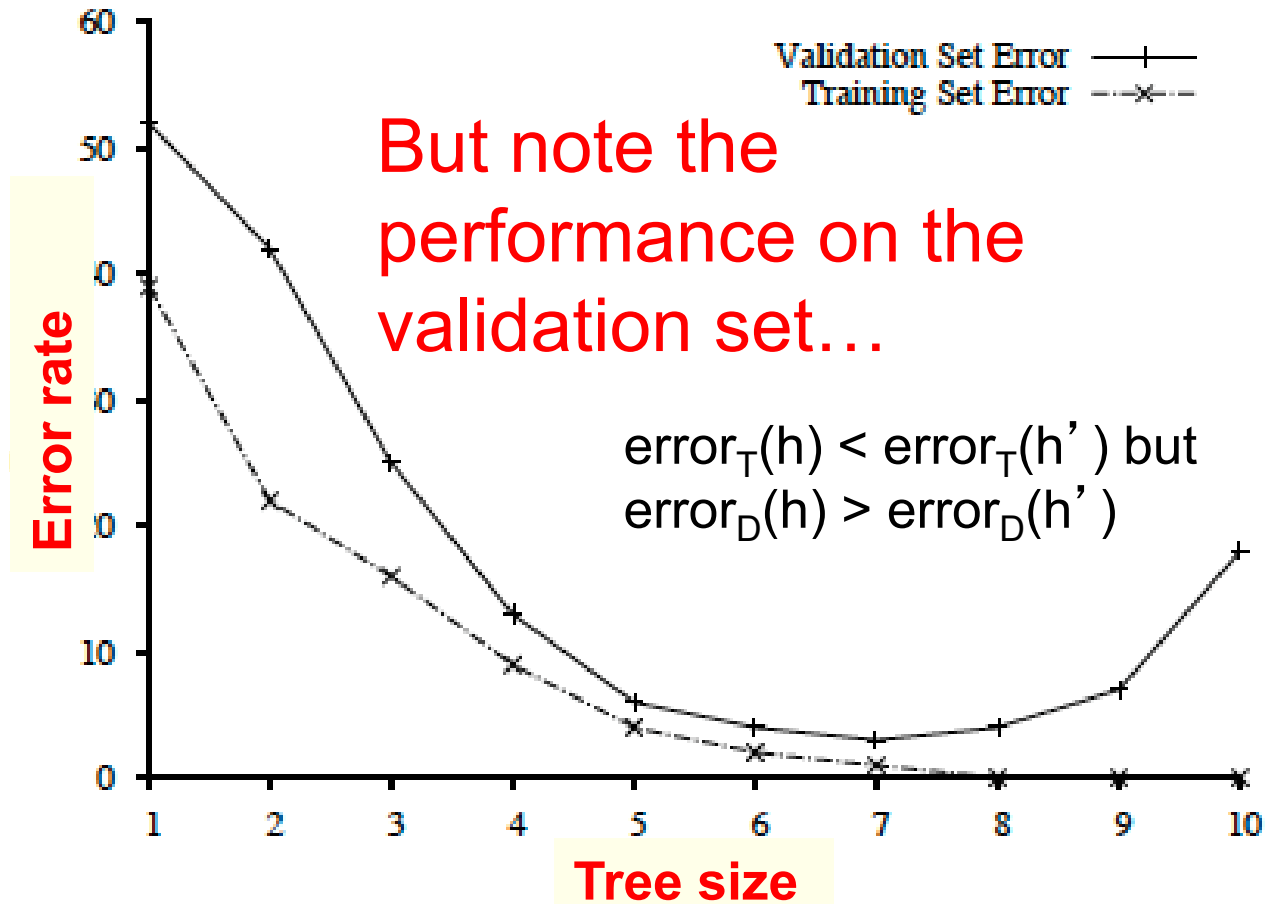


# 树的大小 Tree size



We set tree size as a parameter in our DT learning algorithm  
Note: with larger and larger trees,  
we just do better and better on the training set!

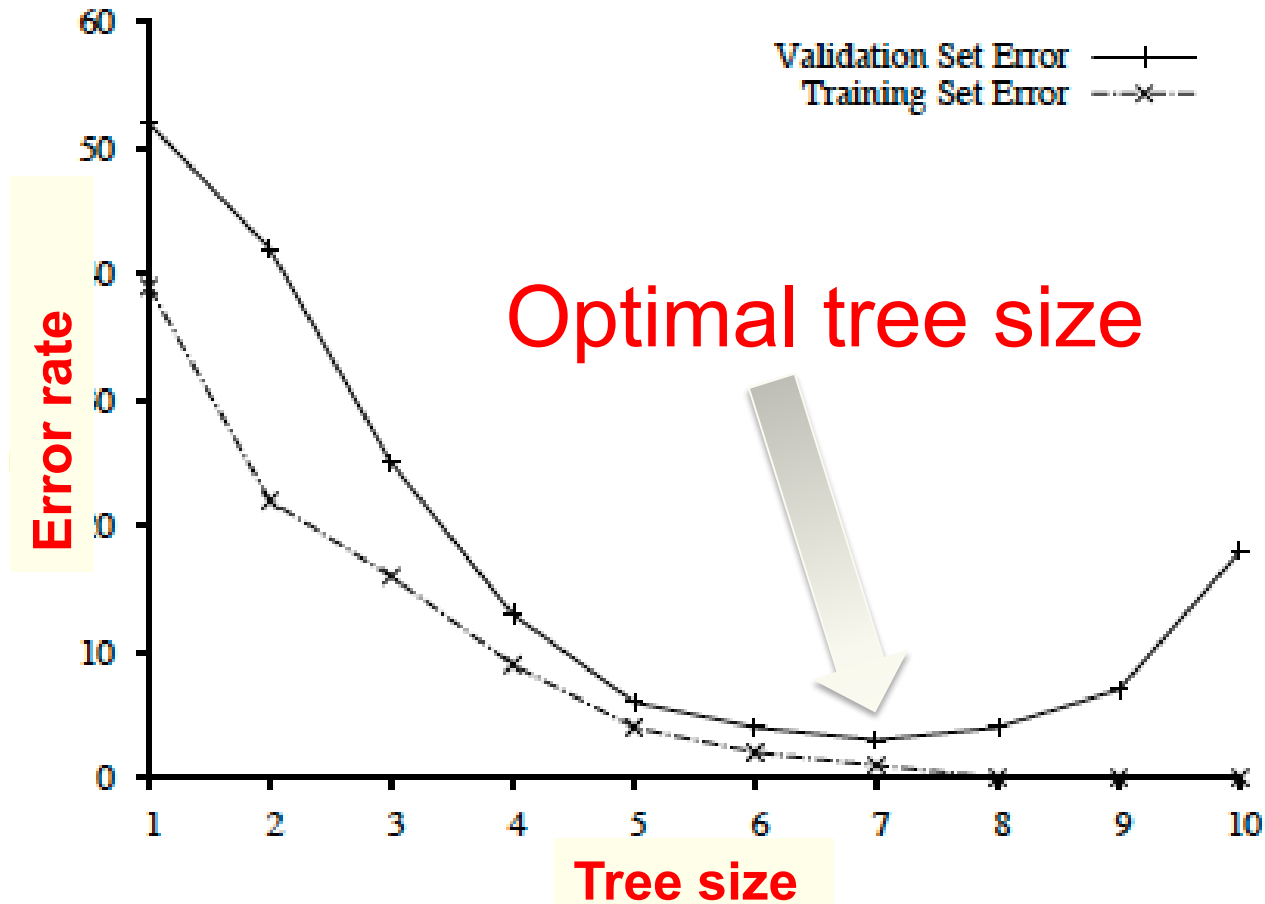
# 树的大小 Tree size



We set tree size as a parameter in our DT learning algorithm  
Note: with larger and larger trees,  
we just do better and better on the training set!

# 树的大小

## Tree size



We set tree size as a parameter in our DT learning algorithm  
Note: with larger and larger trees,  
we just do better and better on the training set!

# 确定树的最优大小

## Find optimal tree size

- Procedure for finding the optimal tree size is called ‘model selection’.
  - To determine validation error for each tree size, use k-fold cross-validation.
  - Uses “all data - test set” --- k times splits that set into a training set and a validation set.
  - After right decision tree size is found from the error rate curve on validation data, train on all training data to get final decision tree (of the right size).
  - Finally, evaluate tree on the test data (not used before) to get true generalization error (to unseen examples).

# 树剪枝

## Pruning Trees

- technique for reducing the number of attributes used in a tree – **pruning**
  - Remove subtrees for better generalization (requires a separate pruning set)
  - Two types of pruning:
    - Pre-pruning (forward pruning)
    - Post-pruning (backward pruning)

# 预剪枝

## Prepruning

- In prepruning, we decide during the building process when to stop adding attributes
  - e.g., all attributes have been used; the number of instances in a node has less than a certain threshold; the accuracy can not be improved
  - reduce the risk of overfitting
- However, this may be problematic – Why?
  - underfitting
  - Sometimes the current division of some branches can not improve the generalization performance, but the subsequent division on the basis of it may lead to a significant performance improvement

# 后剪枝

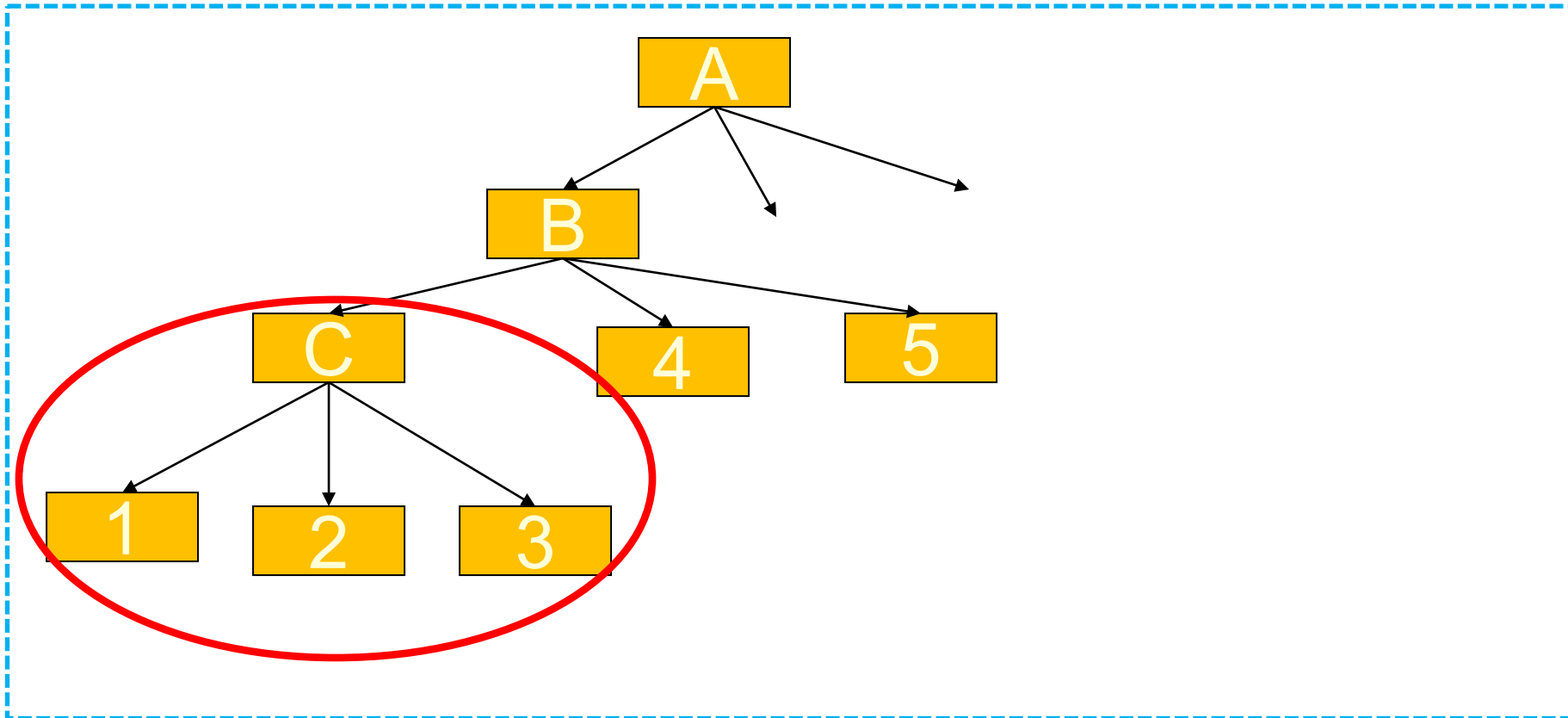
## Postpruning

- Postpruning is a technique that waits until a full decision tree is built and then prunes it to remove unnecessary branches, thereby reducing overfitting.
  - Reduced-Error Pruning(REP) : Evaluates the pruning effect using a validation set and removes branches that do not improve validation set performance.
  - Pesimistic-Error Pruning(PEP) : Commonly used in C4.5 algorithm, this method prunes based on a pessimistic estimation of error.
  - Cost-Complexity Pruning(CCP): Commonly used in the CART algorithm, this method prunes the tree by minimizing a cost function that balances model complexity and error rate.
- Postpruning techniques:
  - Direct Leaf Pruning, Node Merging, Sibling Merging, Level-Based Pruning
  - Subtree Replacement, Subtree Raising
- Advantages and Disadvantages:
  - Low risk of underfitting
  - Better generalization
  - Longer training time

# 子树代替

## Subtree Replacement

- Entire subtree is replaced by a single leaf node

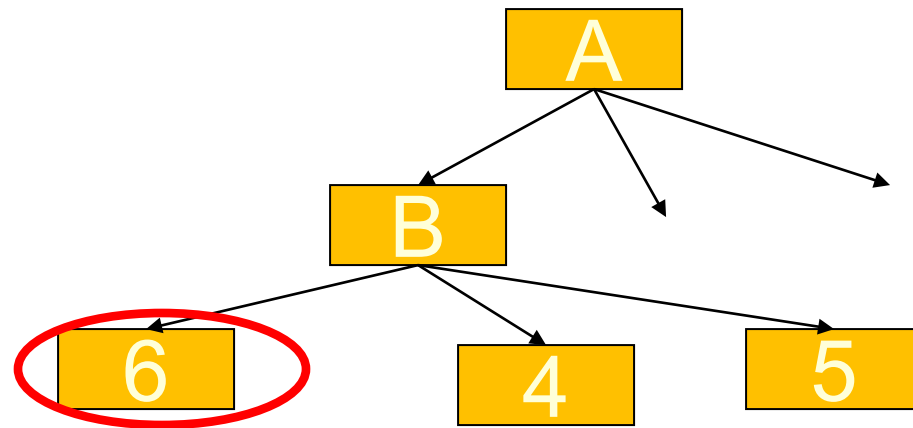




# 子树代替

## Subtree Replacement

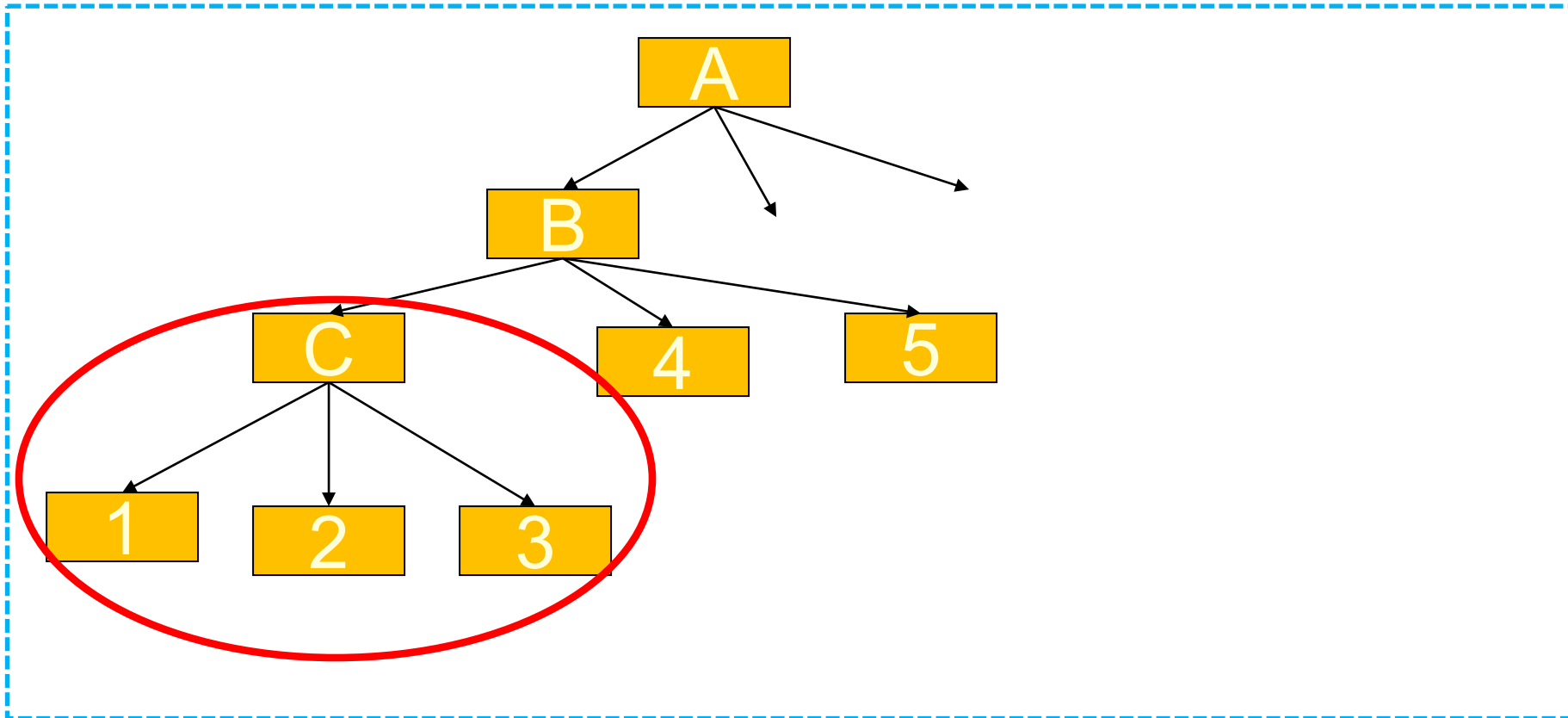
- Entire subtree is replaced by a single leaf node



Generalizes tree a little more,  
but may increase accuracy!

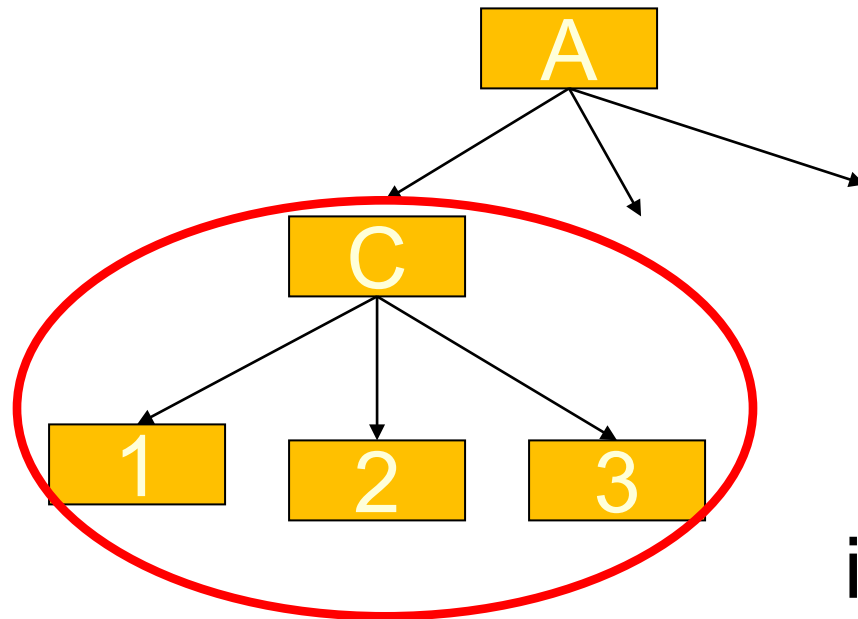
# 子树提升 Subtree Raising

- Entire subtree is raised onto another node



# 子树提升 Subtree Raising

- Entire subtree is raised onto another node



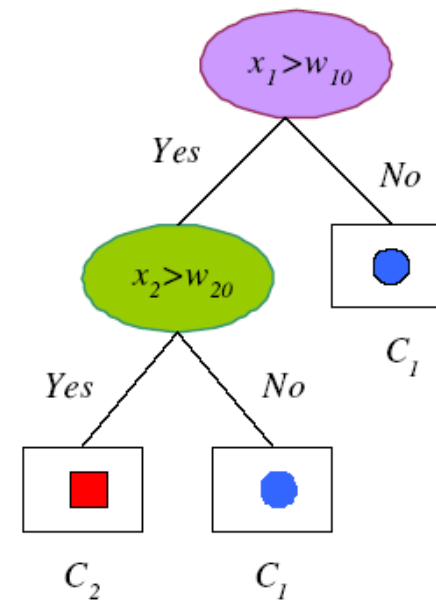
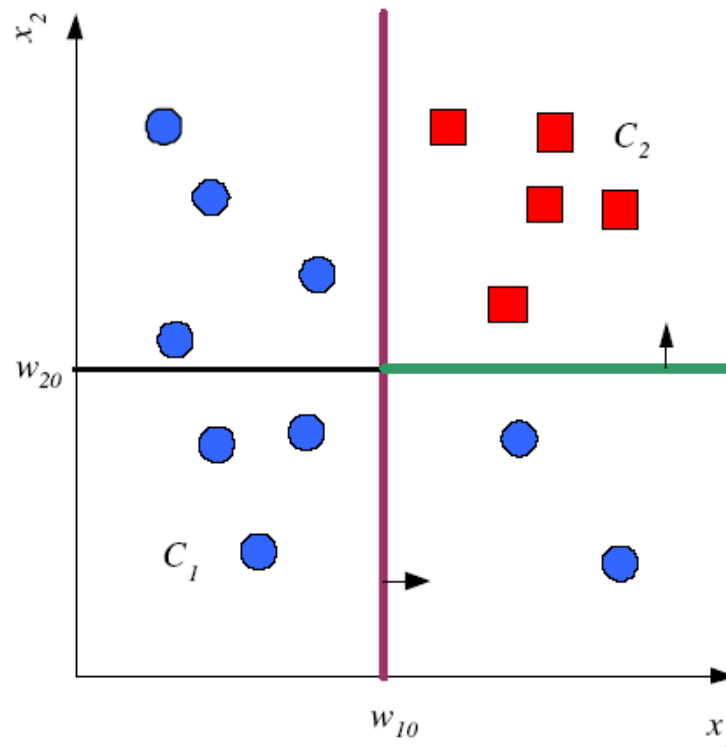
If the subtree provides a good generalization of the ancestor's decisions, the intermediate nodes are pruned away.

it is very time consuming !

# 决策树分类界面

## Decision Boundary

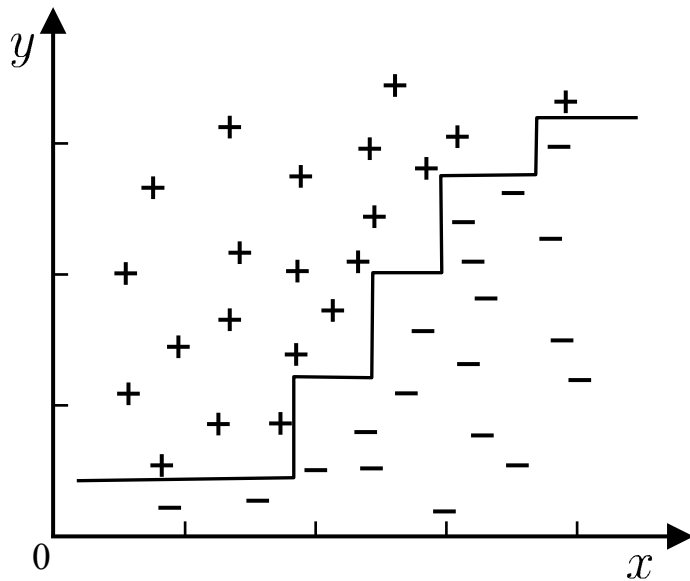
- Decision trees divide the feature space into axis-parallel(hyper-)rectangles
- Each rectangular region is labeled with one label



# 决策树分类界面

## Decision Boundary

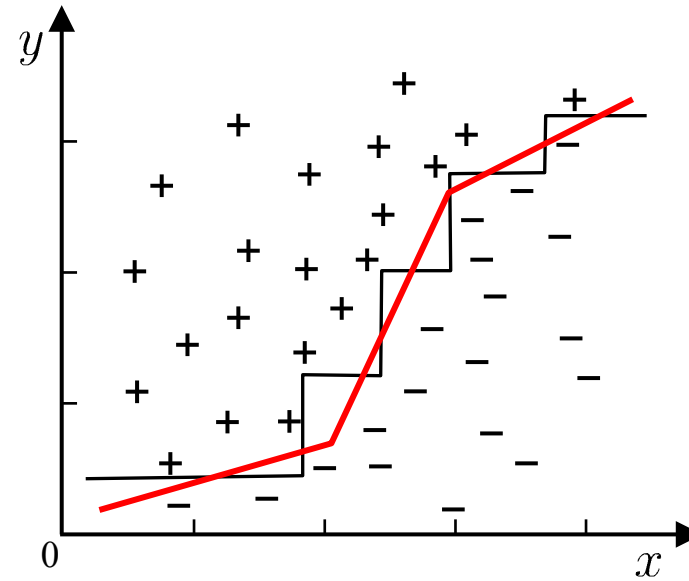
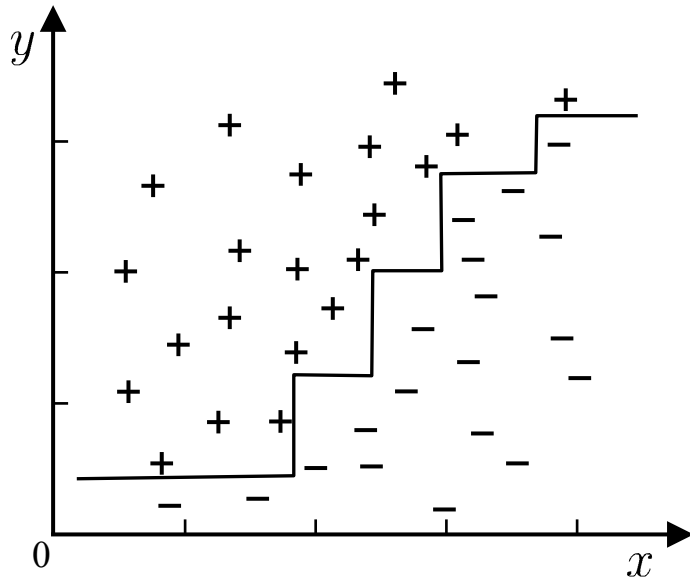
- Many segments must be used to get better approximations when the learning tasks are complex



# 决策树分类界面

## Decision Boundary

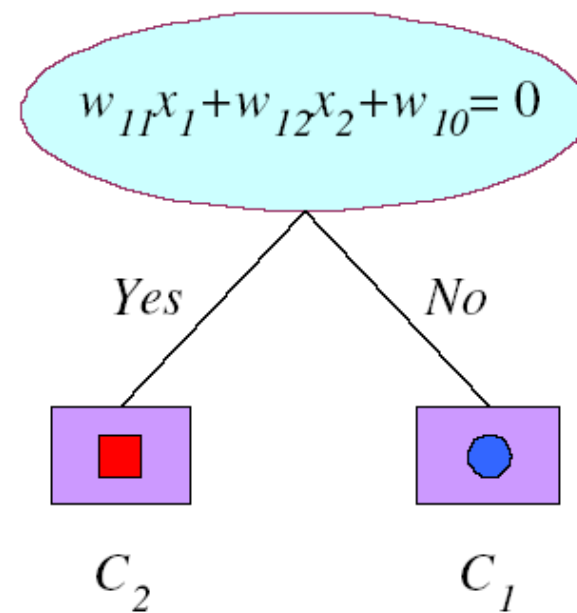
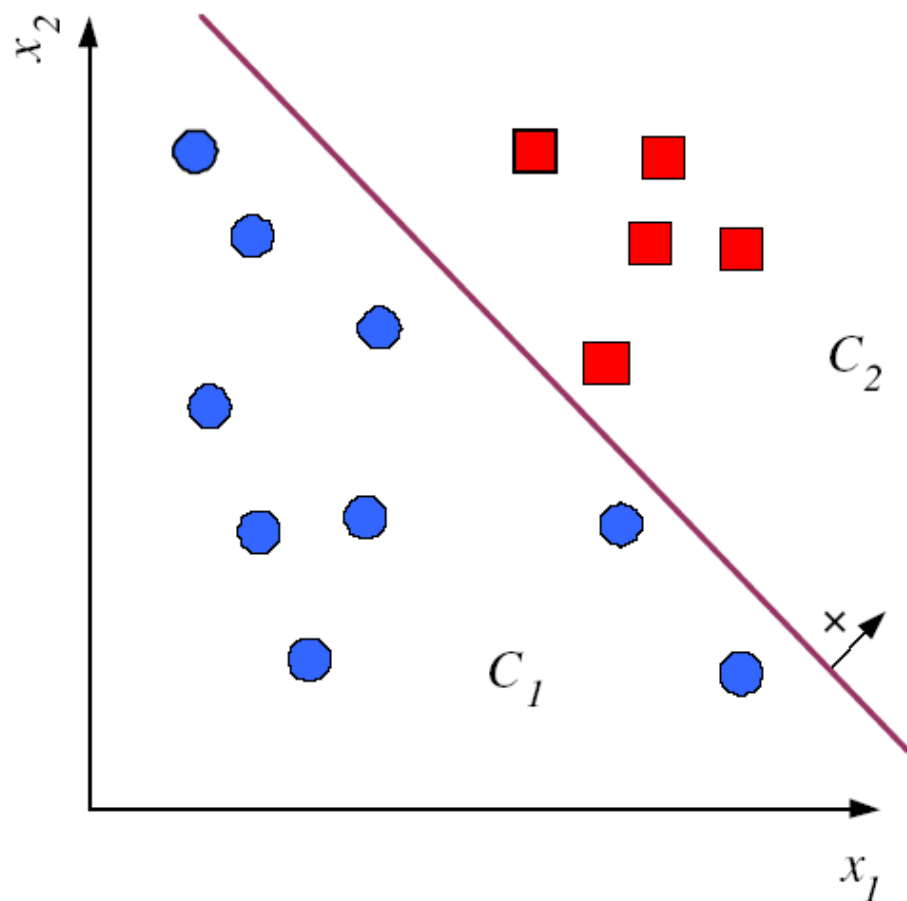
- many segments must be used to get better approximations when the learning tasks are complex



# 多变量决策树

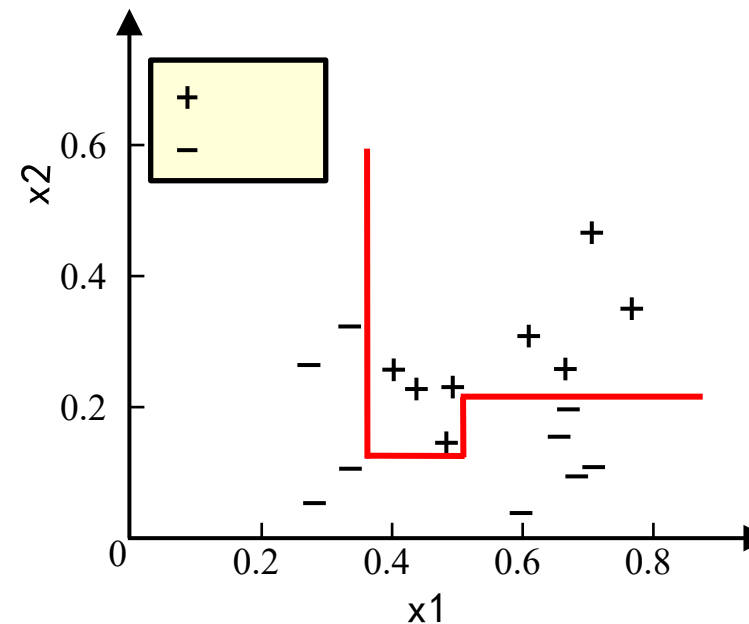
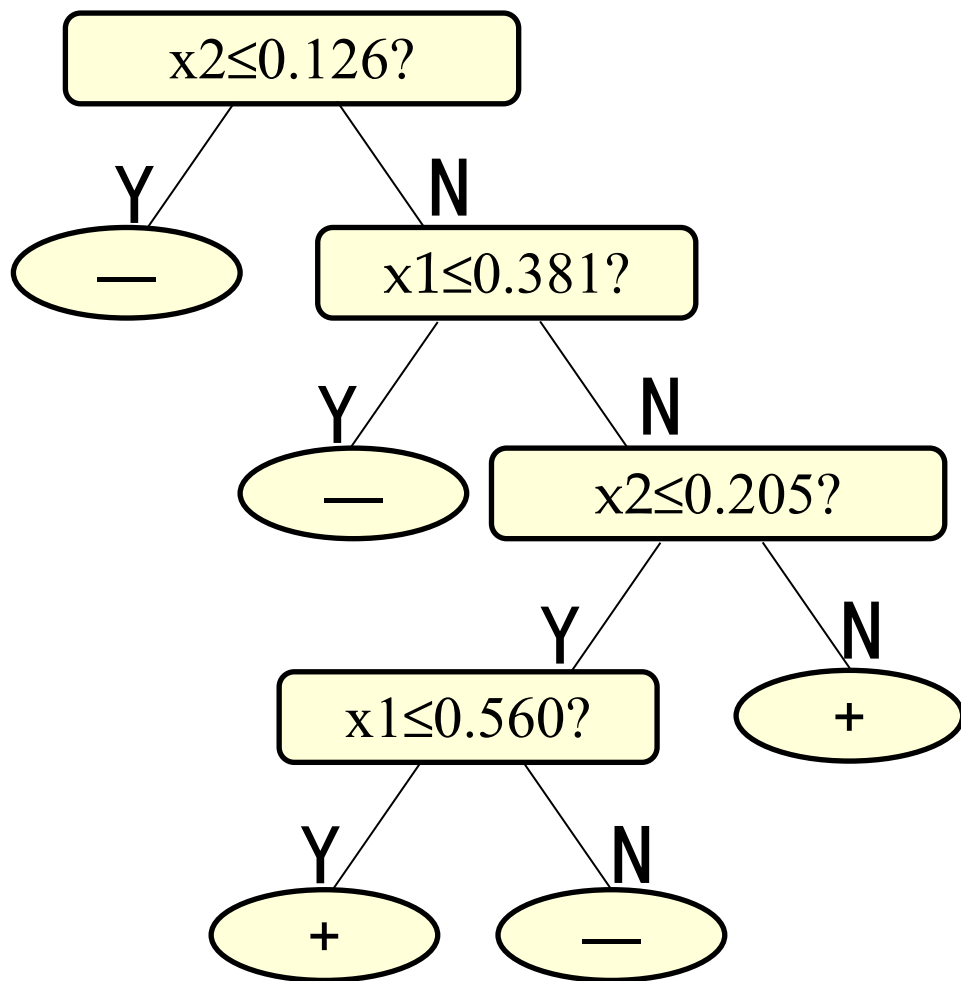
## Multivariate Trees

- Internal nodes can test linear combination of attributes



# 单变量决策树

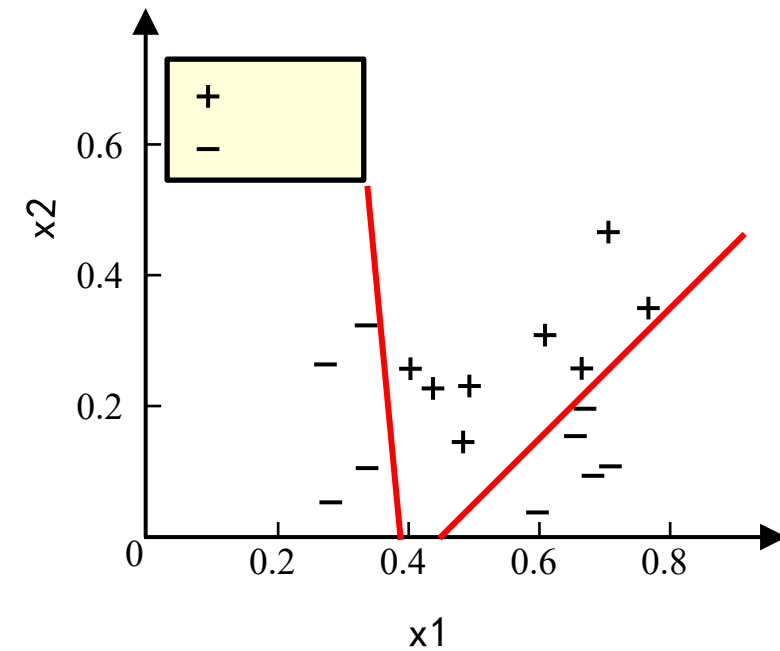
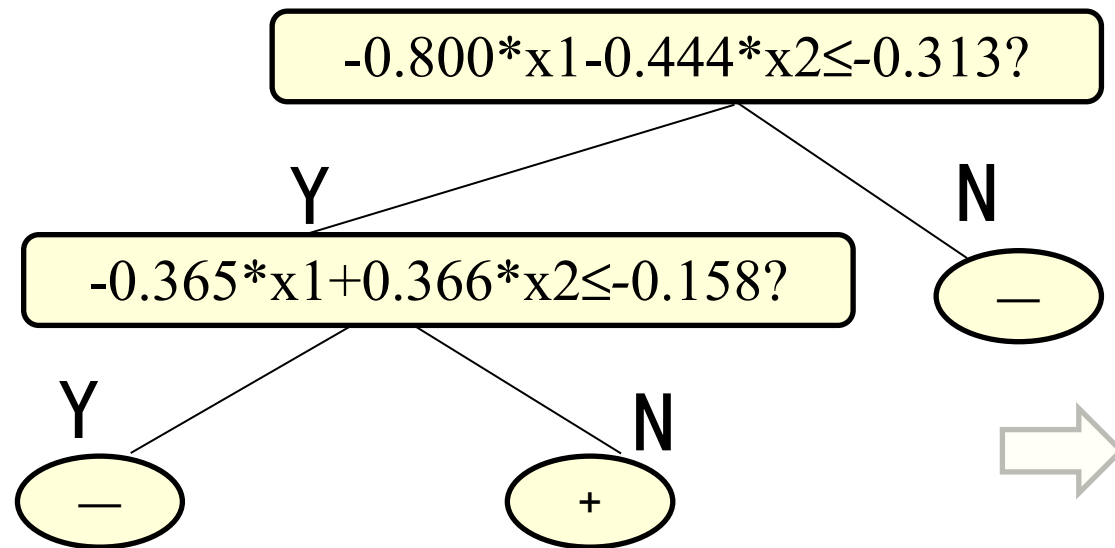
## Univariate Trees





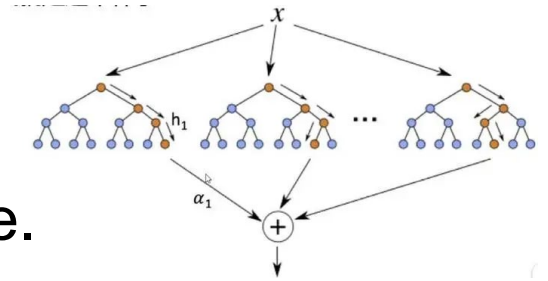
# 多变量决策树

## Multivariate Trees



# 集成模型

## Ensemble model



Multiple decision trees are combined to improve the overall performance.

- Random Forest

Each tree is built using a random subset of the data and a random subset of the feature. Each tree is constructed independently and trained in parallel, with no dependence between them.

- Boosted Tree

Each new tree tries to correct the errors of the previous ones (trained on the residuals). Trees are built sequentially, with each tree dependent on the previous ones.

- **XGBoost** : An efficient and fast gradient boosting algorithm known for its high performance.
- **LightGBM**: A lightweight gradient boosting framework optimized for speed and memory efficiency, especially suitable for large datasets.
- **CatBoost** : A gradient boosting library designed to handle categorical features efficiently, with automatic handling of categorical data and robust performance even with default parameters.

# 决策树经典算法小结

## Summary of DT algorithms

	support task	criterion structure	tree structure	continuous attribute value	missing attribute value	pruning	attribute multiple use
<b>ID3</b>	classification	Gain information	multiple Tree	No support	No support	No support	No support
<b>C4.5</b>	classification	Gain information rate	multiple Tree	support	support	support	No support
<b>CART</b>	Classification; regression	Gini index; mean square error	binary tree	support	support	support	support

