Machine Learning
机器学习

Lecture4：决策树

李洁
nijanice@163.com

监督学习
Supervised Learning

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

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

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 θ to make the prediction closed to the corresponding label

线性回归模型
Linear Regression Model

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

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\dots,N}$$

$x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \dots x_n^{(i)})^T$

$y^{(i)} =$ output data(label) of i^{th} training example

let the machine learn a function from data to label

$$y \approx \boxed{f_\theta(x)} \Rightarrow \boxed{f_\theta(x) = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n + \theta_0}$$

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

逻辑斯蒂回归
Logistic Regression

- 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

$$y^{(i)} \approx \boxed{f_\theta(x^{(i)})} \Rightarrow \boxed{f_\theta(x) = \frac{1}{1+e^{-\theta^T x}}}$$

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

监督学习
Supervised Learning

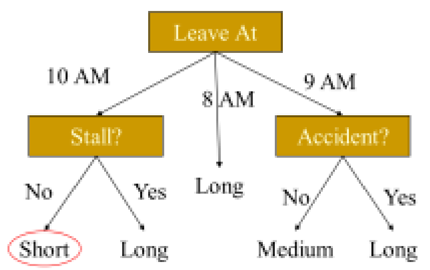
- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

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

a decision tree



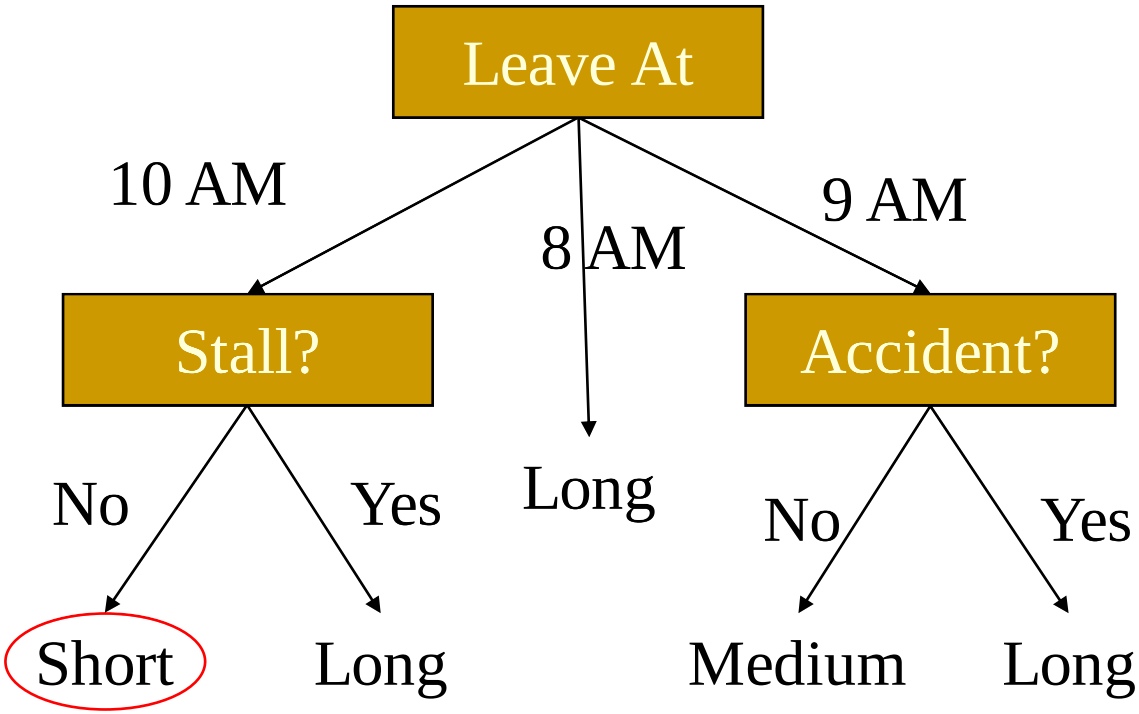
- Function set $\{f_\theta(x^{(i)})\}$ is called hypoth
- Learning is referred to as updating the the prediction closed to the corresponding label

什么是决策树
What is a Decision Tree?

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

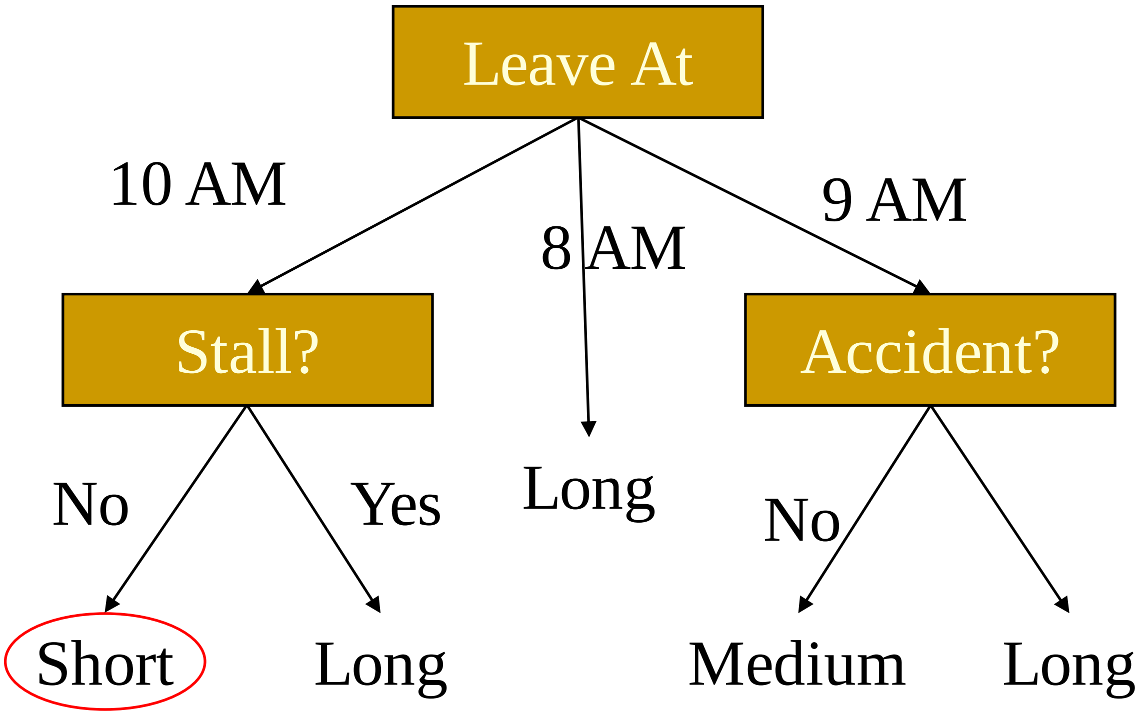
- *Re-representation as if-then rules: disjunction of conjunctions of* constraints on the attribute value instances

决策树举例
Predicting Commute Time



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
Decision Tree as a Rule Set



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
Learning Decision Tree Example

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

Goal predicate: Will wait?

决策树学习举例
Learning Decision Tree Example

- 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
Learning Decision Tree Example

- Examples described by attribute/feature values
- E.g., situations where I will/won't wait for a table:

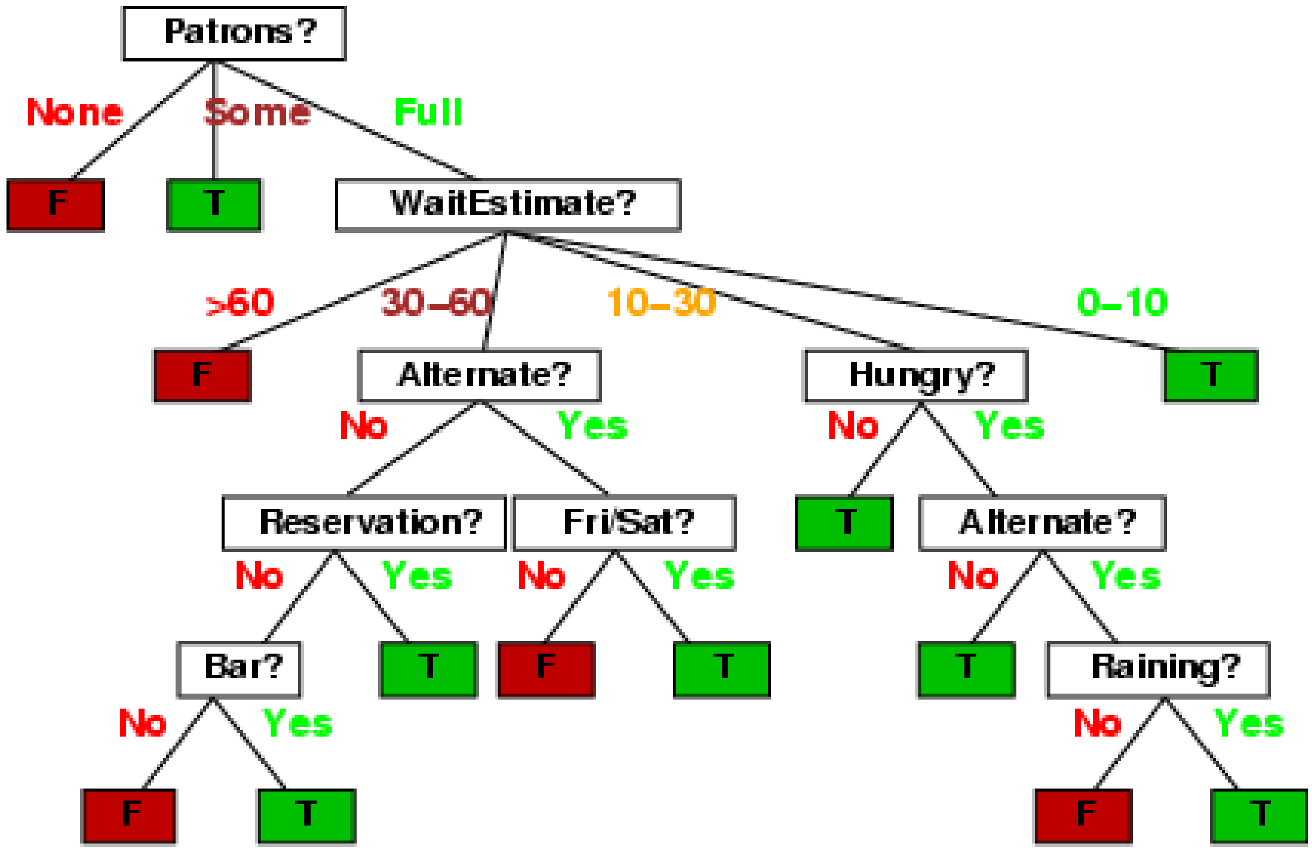
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	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

12 examples
6 +
6 -

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

决策树学习举例
Learning Decision Tree Example

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



决策树的表达能力
Expressiveness of Decision Tree

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:

$$\forall s \;\; WillWait(s) \leftrightarrow (P1(s) \vee P2(s) \vee \ldots \vee Pn(s))$$

Where each condition Pi(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
Expressiveness of Decision Tree

- How many distinct decision trees with 10 Boolean attributes?

Input features	Output
0 0 0 0 0 0 0 0 0 0	0/1
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
Expressiveness of Decision Tree

- How many distinct decision trees with 10 Boolean attributes?

Input features		Output
0 0 0 0 0 0 0 0 0 0		0/1
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

How many entries does this table have?

2^{10}

决策树的表达能力
Expressiveness of Decision Tree

- How many distinct decision trees with 10 Boolean attributes?

Input features	Output
0 0 0 0 0 0 0 0 0 0	0/1
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

How many entries does this table have?

$$2^{10}$$

So how many Boolean functions with 10 Boolean attributes are there, given that each entry can be 0/1?

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

决策树的表达能力
Expressiveness of Decision Tree

- How many distinct decision trees with n Boolean attributes?

  = number of distinct truth tables with 2^n rows

  $= 2^{2^n}$

E.g. how many Boolean functions on 6 attributes? A lot…

- With 6 Boolean attributes, there are 18,446,744,073,709,551,616 possible trees!

Googles calculator could not handle 10 attributes ☺!

监督学习
Supervised Learning

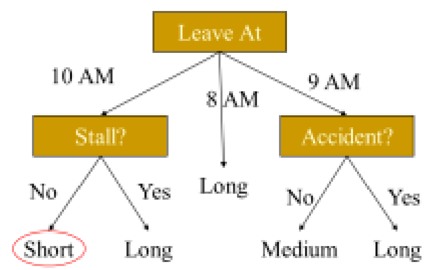
- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

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

a decision tree



- Function set $\{f_\theta(x^{(i)})\}$ is called hypoth
- Learning is referred to as updating the
  the prediction closed 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 sample, where the path tests sets each attribute value to the value of the sample?

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 sample, where the path tests sets each attribute value to the value of the sample?

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

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 samples and make that attribute a decision node
  - Repeat this process for recursively for each subtree
  - Stop (create a leaf node), when:
    - All the samples belong to the same class
    - There are no more attributes, or all the samples have the same attribute values
    - The node contains fewer than a minimum number of samples.
    - …
  - assign the leaf node the class label that is most common among the samples in the node (majority voting, for classification task)

ID3 启发式算法
ID3 Heuristic Algorithm

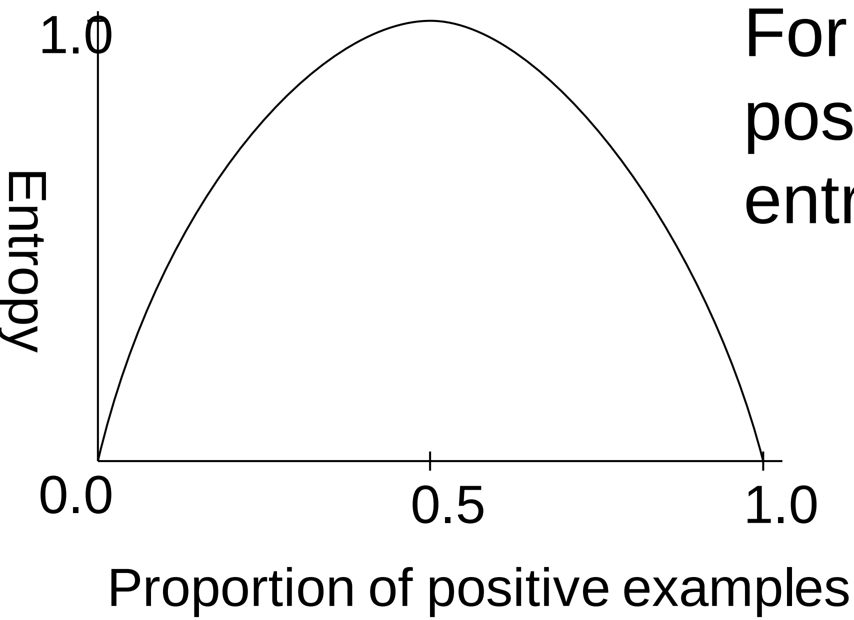
- How to determine the best attribute?

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

p(head)=0.5
p(tail)=0.5
H=1

p(head)=0.51
p(tail)=0.49
H=0.9997

熵
Entropy



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

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

p - # positive examples

n - # negative examples

Example taken from
Tom Mitchell's
Machine Learning

决策树学习举例
Learning Decision Tree Example

- Examples described by attribute/feature values
- E.g., situations where I will/won't wait for a table:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
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

12 examples
6 +
6 -

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

决策树学习举例
Learning Decision Tree Example

What' s the entropy of this collection of examples?

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
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
Learning Decision Tree Example

What’s the entropy of this collection of examples?

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
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

12 examples
6 +
6 -

$p = n = 6;$

$$\frac{P}{P+n} = \frac{n}{P+n}$$

$$= 0.5$$

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

决策树学习举例
Learning Decision Tree Example

What’s the entropy of this collection of examples?

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

$$= -0.5\, log2(0.5) - 0.5\, log2(0.5) = 1$$

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

12 examples
6 +
6 -

$p = n = 6;$

$$\frac{P}{P+n} = \frac{n}{P+n}$$

$$= 0.5$$

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

熵
Entropy

- Calculation of entropy

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

- D = set of examples
- p_k = the portion of D belong to class k.
- $|y|$ = size of the range of the target attribute

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:

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

D^v is the subset of D for which attribute a has value v

ID3
J. Ross Quinlan,1975

ID3算法构建决策树
Decision Tree Building: ID3 Algorithm

Start from the root node with all data

- For each node, calculate the information gain of all possible attributes
- Choose the attribute with the highest information gain
- Split the samples of the node according to the attribute (can not be used again)
- Do the above recursively for each leaf node, until
  - All of examples have been correctly classified
  - Or there is no attribute to select

决策树学习举例
Learning Decision Tree Example

- Examples described by attribute values
- E.g., situations where I will/won't wait for a table:

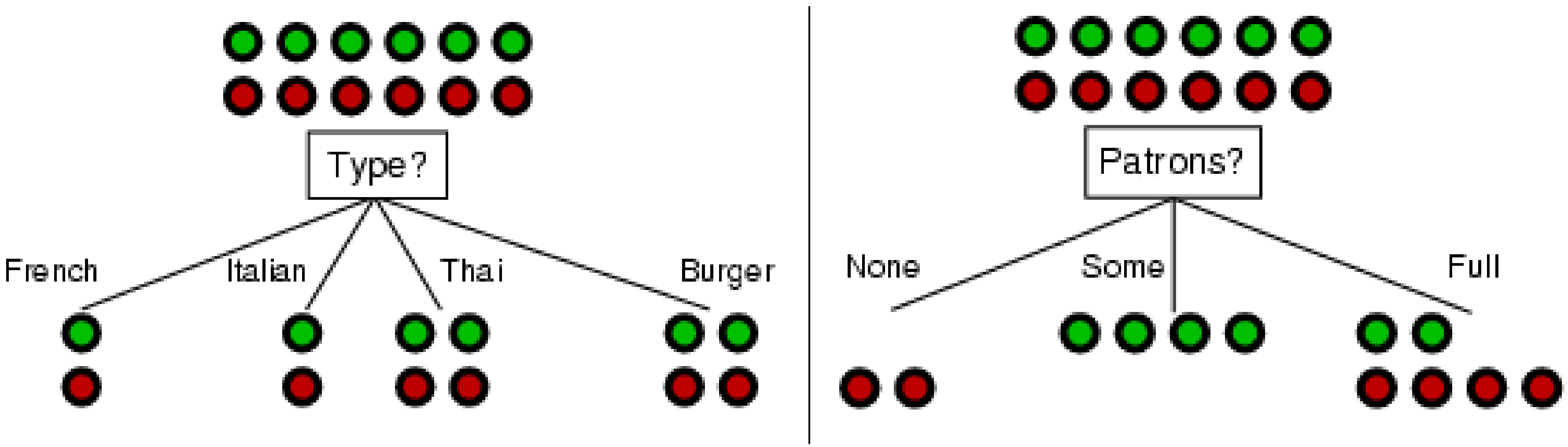
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	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

12 examples
6 +
6 -

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

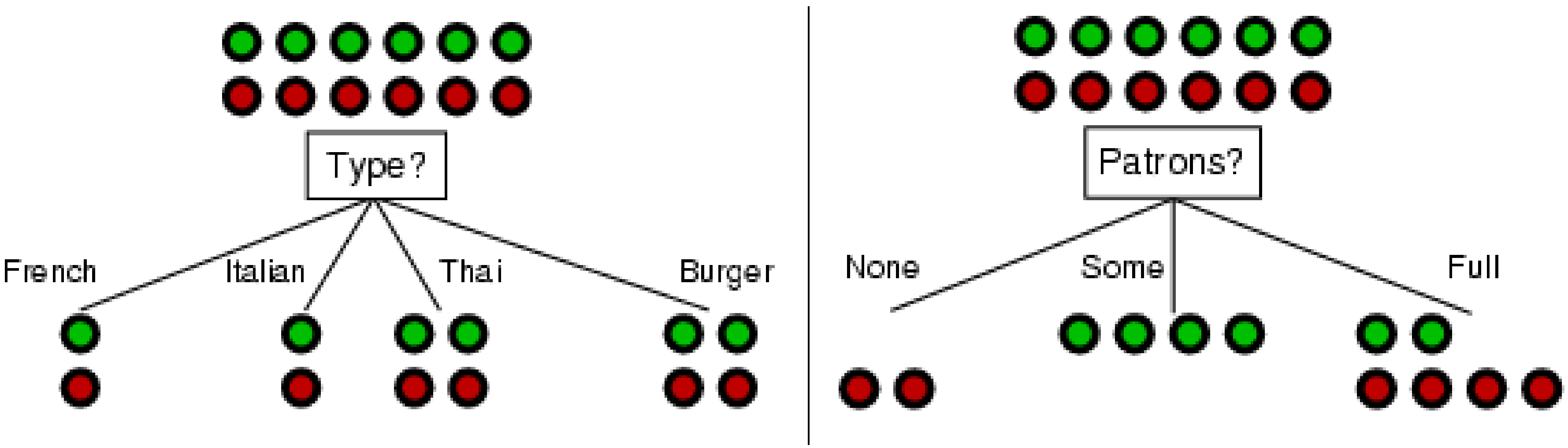
决策树学习举例
Learning Decision Tree Example

Which one should we pick?



决策树学习举例
Learning Decision Tree Example

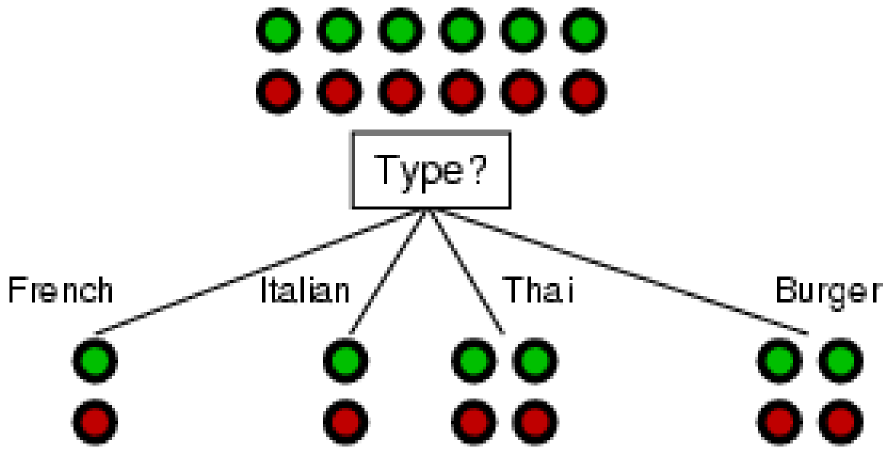
Goal: trees with short paths to leaf nodes

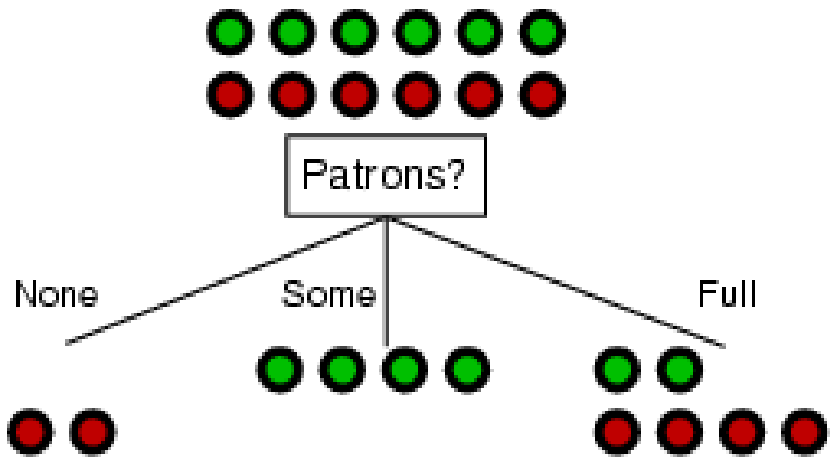


A perfect attribute would ideally divide the examples into sub-sets that are all positive or all negative… i.e. maximum information gain.

决策树学习举例
Learning Decision Tree Example

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



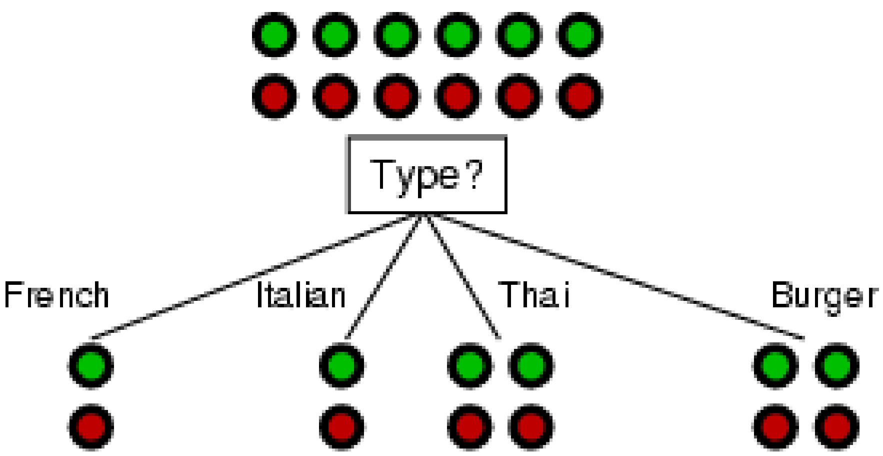


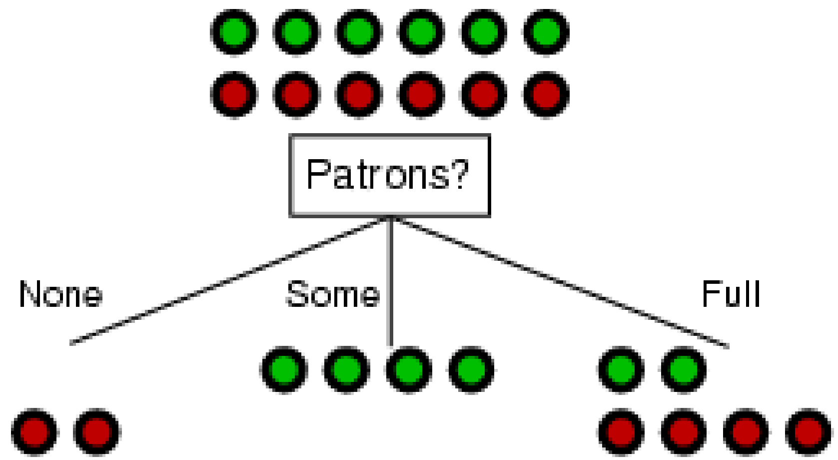
$Gain(D, Type)$

$Gain(D, Patrons)$

决策树学习举例
Learning Decision Tree Example

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



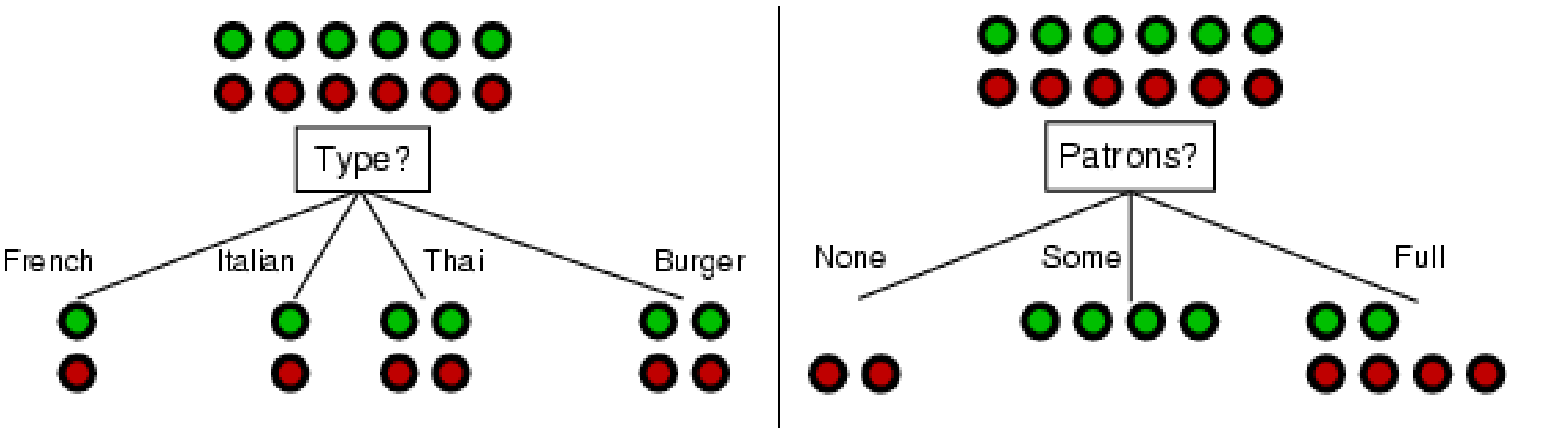


$$Gain(D,Type)$$
$$= 1 - \left[\frac{2}{12} Entropy\left(\frac{1}{2},\frac{1}{2}\right) + \frac{2}{12} Entropy\left(\frac{1}{2},\frac{1}{2}\right) + \frac{2}{12} Entropy\left(\frac{1}{2},\frac{1}{2}\right) + \frac{2}{12} Entropy\left(\frac{1}{2},\frac{1}{2}\right)\right] = 0$$

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

决策树学习举例
Learning Decision Tree Example

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



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

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

决策树学习举例
Learning Decision Tree Example

- Examples described by attribute values
- E.g., situations where I will/won't wait for a table:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
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

12 examples
6 +
6 -

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

决策树学习举例
Learning Decision Tree Example

- If add an attribute—’number’

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	[illegible]
X_5	T	F	T	F	Full	$$$	F	T	French	>60	[illegible]
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	[illegible]
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	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

Number
1
2
3
4
5
6
7
8
9
10
11
12

es

-

决策树学习举例
Learning Decision Tree Example

- If add an attribute—'number'

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
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	
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	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

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.

决策树学习举例
Learning Decision Tree Example

- If add an attribute—'number'

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	
X_5	T	F	T	F	Full	$$$	F	T	French	>60	
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	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

Number
1
2
3
4
5
6
7
8
9
10

es

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

决策树学习举例
Learning Decision Tree Example

- If add an attribute—'number'

Number
1
2
3
4
5
6
7
8
9
10

$$Gain(D, \text{number})$$

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

$$= 1 - \left[\frac{6}{12} Entropy(1,0) + \frac{6}{12} Entropy(0,1)\right]$$

$$= 1 - 0 \quad = 1$$

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

信息增益和ID3算法
Information Gain and ID3

- 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:

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

$$a_* = \underset{a \in A}{\operatorname{argmax}}\, Gain\,(D,a)$$

D^v is the subset of D for which attribute a has value v

ID3
J. Ross Quinlan,1975

信息增益率和C4.5算法
Information Gain Ratio and C4.5

$$Gain\_\text{ratio}(D,a) = \frac{Gain(D,a)}{IV(a)}$$

$$a_* = \underset{a \in A}{\text{argmax}}\, Gain\_\text{ratio}\,(D,a)$$

$IV(a)$ = intrinstic 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 for which attribute a has value v

C4.5 is an extension of ID3
J. Ross Quinlan, The Morgan Kaufmann Series
in Machine Learning, Pat Langley.
Gain Ratio for Attribute Selection

决策树学习举例
Learning decision trees: An example

- If add an attribute—'number'

$$Gain\_\text{ratio}(D, number) = \frac{Gain(D,a)}{IV(a)}$$

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

$$= \frac{1}{-12 \frac{1}{12} \log_2 \frac{1}{12}}$$

$$= \frac{1}{\log_2 12}$$

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.

基尼指数和CART算法
Gini index and CART

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

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

D^l and D^r are the subsets of D splitted by a =v.

$Gini(D)$ reflects the probability that two samples randomly selected from D are inconsistent

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

CART
Breiman et al., 1984
Classification and Regression tree

基尼指数和CART算法
Gini index and CART

Binary Tree

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

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

D^l and D^r are the subsets of D splitted by a $=v$.

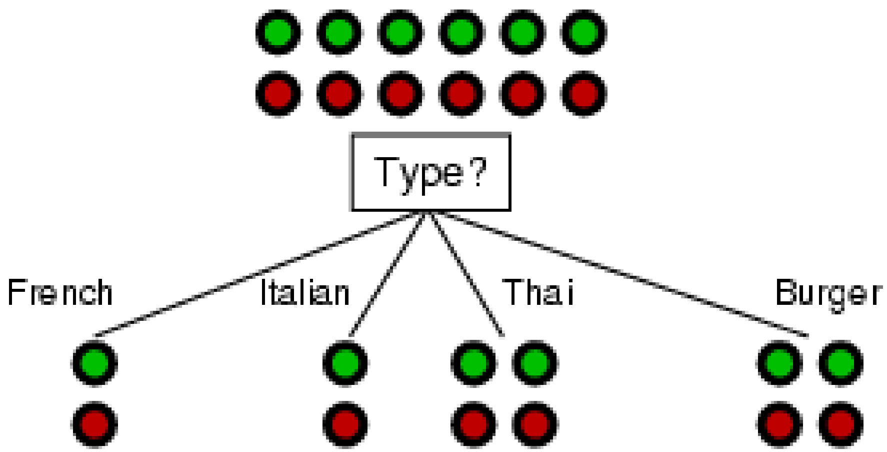
$Gini(D)$ reflects the probability that two samples randomly selected from D are inconsistent

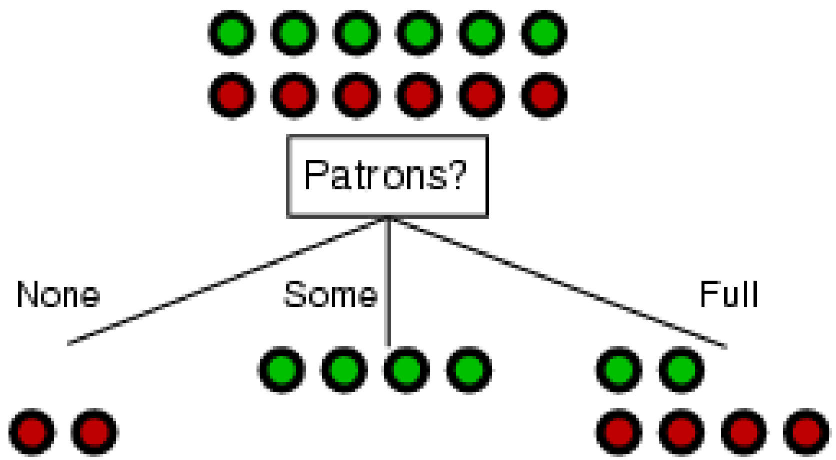
$$\text{Gini}(D) = \sum_{k=1}^{|y|} \sum_{\mathrm{k}' \neq k} p_k p_{\mathrm{k}'} = 1 - \sum_{k=1}^{|y|} p_k^2$$

CART
Breiman et al., 1984
Classification and Regression tree

决策树学习举例
Learning decision trees: An example

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





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

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

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

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

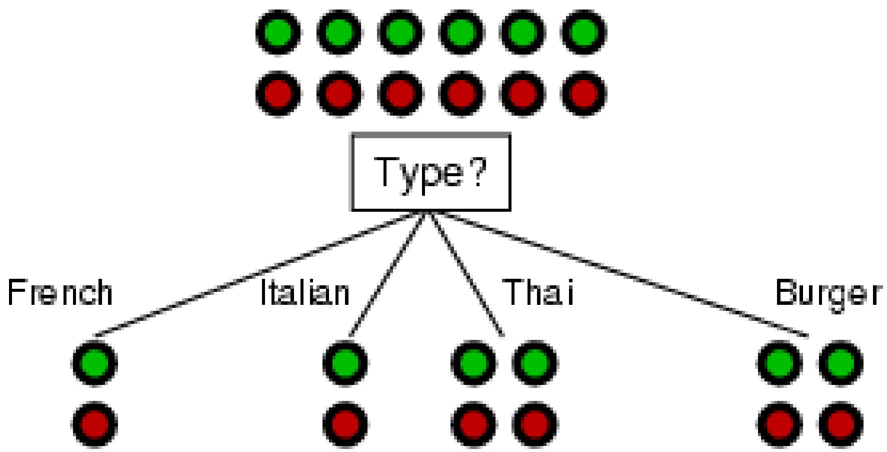
$Gini\_index(D, Patrons = None) =$

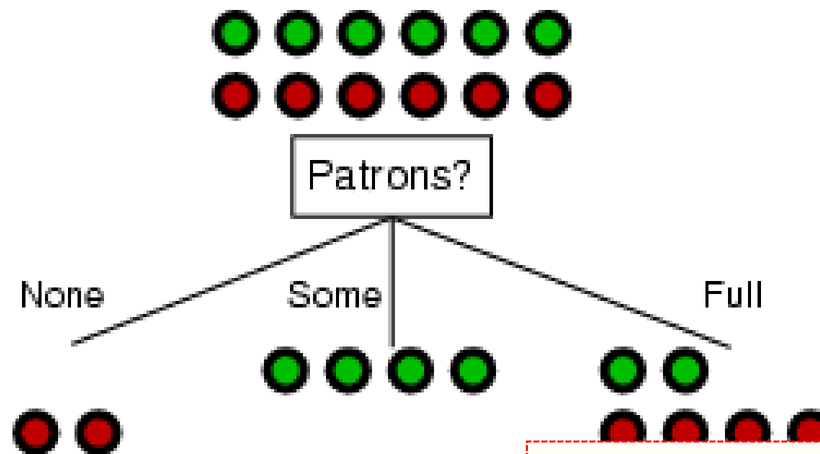
$Gini\_index(D, Patrons = some) =$

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

决策树学习举例
Learning decision trees: An example

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





$Gini\_index(D, Type = French) =$ $\frac{2}{12} + \frac{10}{12} \times 2 \times \frac{5}{10} \times \frac{5}{10} = \frac{7}{12}$

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

$Gini\_index(D, Type = Thai) =$ $\frac{4}{12} \times 2 \times \frac{2}{4} \times \frac{2}{4} + \frac{8}{12} \times 2 \times \frac{4}{8} \times \frac{4}{8} = \frac{1}{2}$

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

$Gini\_index(D, Patrons = No$ $\frac{2}{12} \times 0 + \frac{10}{12} \times 2 \times \frac{6}{10} \times \frac{4}{10} = \frac{2}{5}$

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

$Gini\_$ $\frac{6}{12} \times 2 \times \frac{2}{6} \times \frac{4}{6} + \frac{6}{12} \times 2 \times \frac{2}{6} \times \frac{4}{6} = \frac{4}{9}$ $l)$

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 samples and make that attribute a decision node
  - Repeat this process for recursively for each subtree
  - Stop (create a leaf node), when:
    - All the samples belong to the same class
    - There are no more attributes, or all the samples have the same attribute values
    - The node contains fewer than a minimum number of samples.
    - …
  - assign the leaf node the class label that is most common among the samples in the node (majority voting, for classification task)

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 samples and make that attribute a decision node
  - Repeat this process for recursively for each subtree
  - Stop (create a leaf node), when:
    - All the samples belong to the same class
    - There are no more attributes, or all the samples have the same attribute values
    - The node contains fewer than a minimum number of samples.
    - …
  - assign the leaf node with the average of the target values of the samples in the node (for regression task)

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 samples and make that attribute a decision node
  - Repeat this process for recursively for each subtree
  - Stop (create a leaf node), when:
    - All the samples belong to the same class
    - There are no more attributes, or all the samples have the same attribute values
    - The node contains fewer than a minimum number of samples.
    - …
  - assign the leaf node the class label that is most common among the samples in the node (majority voting, for classification task)

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 samples … node
  - Repeat th… subtree
  - Stop (cre…
    - All the…
    - There … amples have the same attribute values
    - The node contains fewer than a minimum number of samples.
    - …
  - assign the leaf node the class label that is most common among the samples in the node (majority voting, for classification task)

For classification task

ID3: Information Gain

C4.5：Information Gain Ratio

CART：Gini index

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 sample
  - Repeat
  - Stop (c
    - All tl
    - Ther have the s
    - The er of sam
    - …
  - assign the leaf node with the average of the target values of the samples in the node (for regression task)

For classification task

ID3: Information Gain

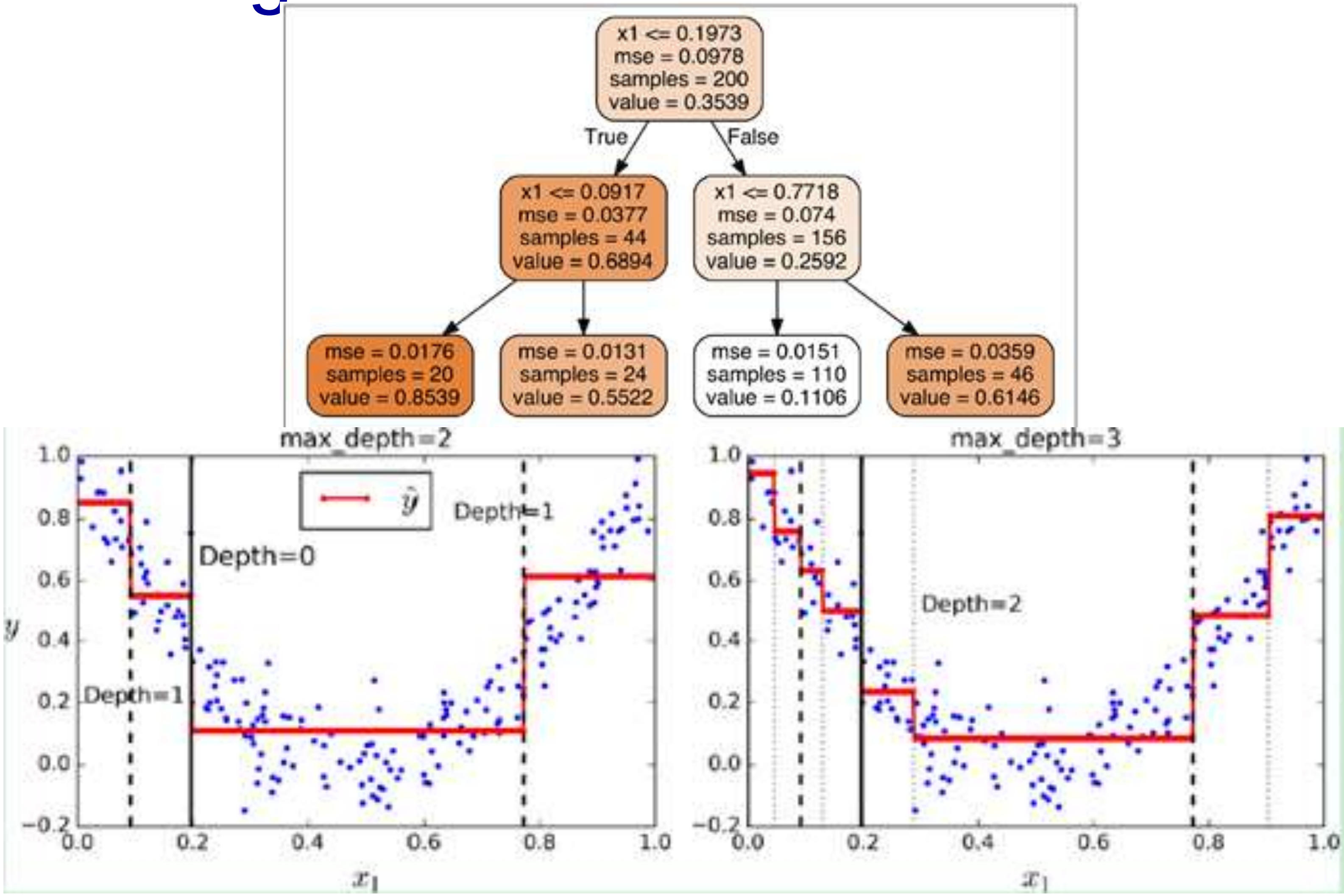
C4.5：Information Gain Ratio

CART：Gini index

For regression task

Mean Squared Error (MSE)

决策树推广到回归
Regression with Decision Trees



The CART can be applied to regression task!

基尼指数和CART算法
Gini index and CART

Binary Tree

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

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

D^l and D^r are the subsets of D splitted by a $= v$.

$Gini(D)$ reflects the probability that two samples randomly selected from D are inconsistent

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

CART
Breiman et al., 1984
Classification and Regression tree

均方误差和CART算法
MSE and CART

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

D^l and D^r are the subsets of D splitted by a $=v$.

The CART can be applied to regression task!

均方误差和CART算法
MSE and CART

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

D^l and D^r are the subsets of D splitted by a $=v$.

$$\frac{d \sum_{x^i \in D^l} \left(y^i - c^l\right)^2}{dc^l} = 0 \quad \Rightarrow \quad c^l = \frac{1}{N^l} \sum_{x^i \in D^l} y^i,$$

$$\frac{d \sum_{x^i \in D^r} \left(y^i - c^r\right)^2}{dc^r} = 0 \quad \Rightarrow \quad c^r = \frac{1}{N^r} \sum_{x^i \in D^r} y^i$$

The CART can be applied to regression task!

均方误差和CART算法
MSE and CART

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

D^l and D^r are the subsets of D splitted by a =v.

$$\frac{d \sum_{x^i \in D^l} \left(y^i - c^l\right)^2}{dc^l} = 0 \quad \Rightarrow \quad c^l = \frac{1}{N^l} \sum_{x^i \in D^l} y^i,$$

$$\frac{d \sum_{x^i \in D^r} \left(y^i - c^r\right)^2}{dc^r} = 0 \quad \Rightarrow \quad c^r = \frac{1}{N^r} \sum_{x^i \in D^r} y^i$$

The CART can be applied to regression task!

均方误差和CART算法
MSE and CART

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

D^l and D^r are the subsets of D splitted by a =v.

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

The CART can be applied to regression task!

均方误差和CART算法
MSE and CART

a	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

a	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
c^l									

c^r									

MSE									

均方误差和CART算法
MSE and CART

a	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

a	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

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

均方误差和CART算法
MSE and CART

a	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

a	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

a	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

a	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

a	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

a	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

a	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

a	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

a	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

a	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			

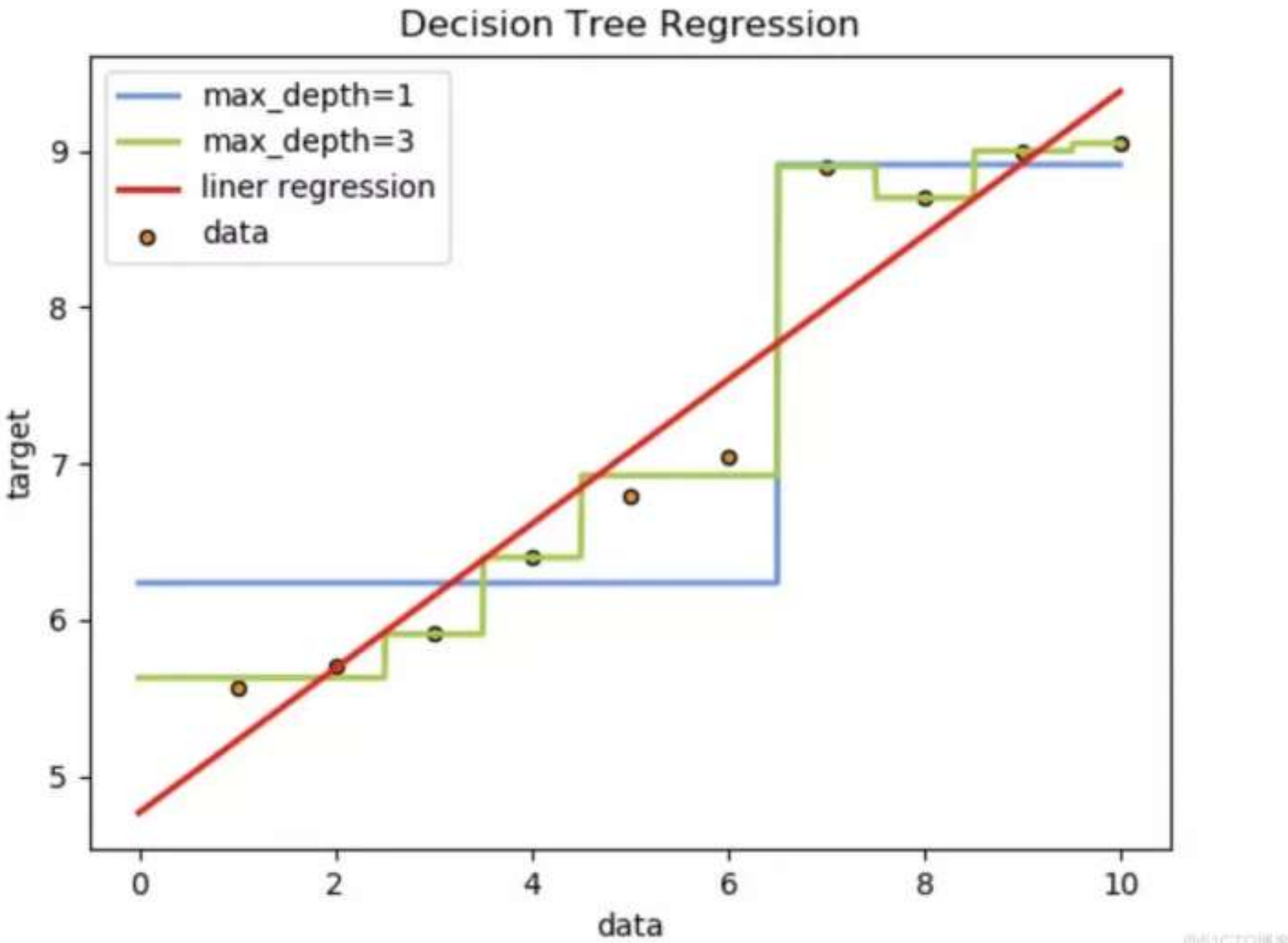
$$
T = \begin{cases} 5.72 & x \leq 3.5 \\ 6.75 & 3.5 < x \leq 6.5 \\ 8.91 & x > 6.5 \end{cases}
$$

决策树和线性回归
Decision Tree & Linear Regression

决策树和线性回归
Decision Tree & Linear Regression

- Decision tree regression and linear regression are two different regression methods with some notable differences:
  - Model Structure
    - a tree-like structure: the path from the root node to a leaf node
    - a linear equation: the linear combination of input features
  - Model Complexity:
    - adapt to more complex, non-linear relationships
    - more suitable for modeling linear relationships
  - Predictive Interpretability:
    - provide an intuitive decision path
    - provide coefficients for feature weights

决策树和线性回归
Decision Tree & Linear Regression



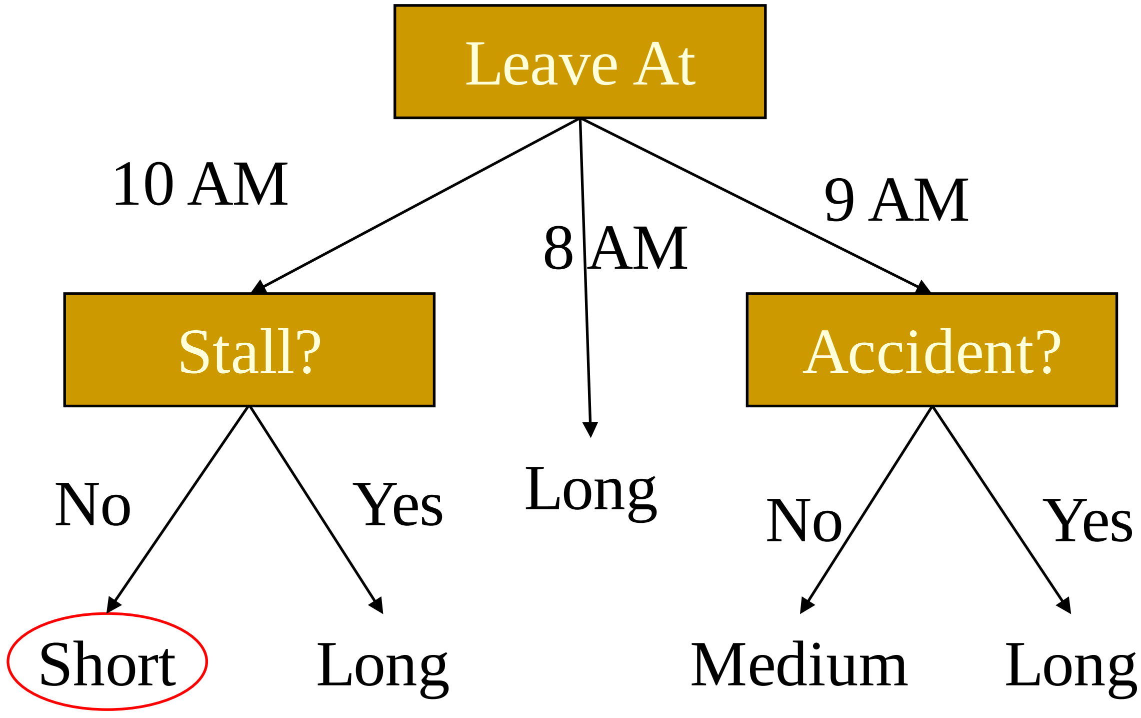
决策树经典算法小结
Summary of DT algorithms

	support task	**criterion structure**	**tree structure**	**continous attribute value**	**missing attribute value**	**pruning**	**attribute multiple use**
ID3	classification	Gain information	multiple Tree	No support	No support	No support	No support
C4.5	classification	Gain information rate	multiple Tree	support	support	support	No support
CART	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:

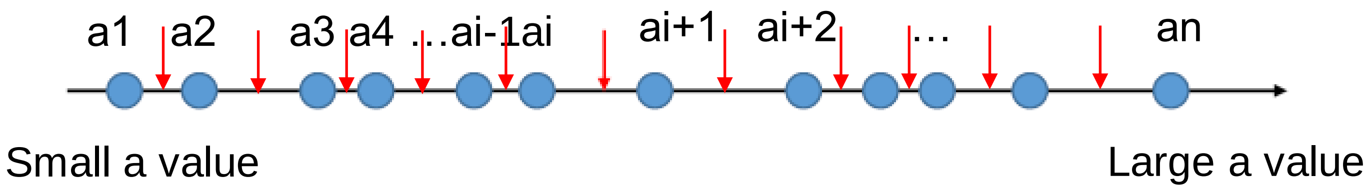
8:00 (L), 8:02 (L), | 8:07 (M), | 9:00 (S), 9:20 (S), 9:25 (S), 10:00 (S), | 10:02 (M)

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

$$T_a^* = \underset{T_a}{argmin}\, Gini\_\text{index} \quad \text{（CART）}$$

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

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 a^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

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

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

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

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

Calculate Gain according to

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

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

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

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

ID	B	C
3	F	F
4	T	T

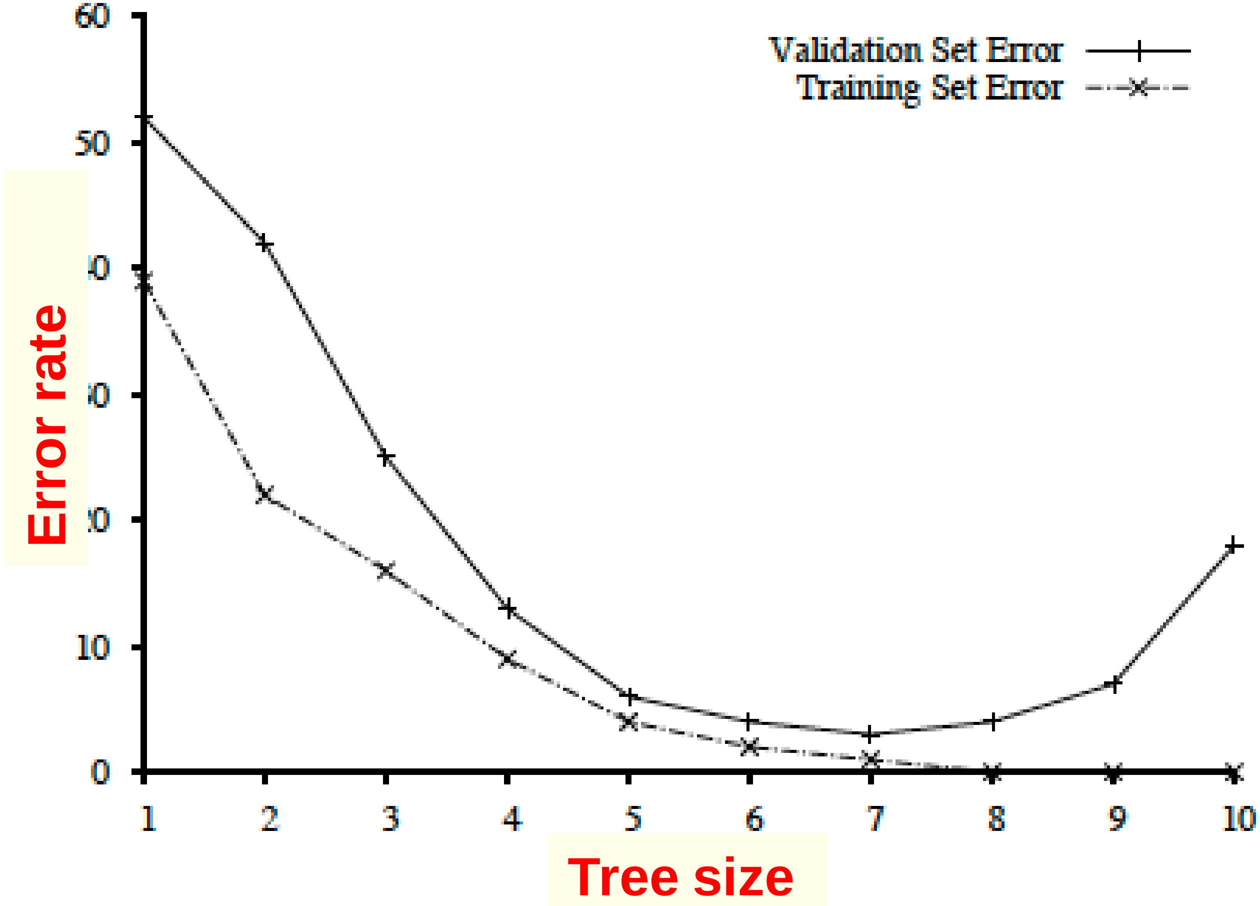
If A=T : C=F
else : C=T
error:30%

If B=F : C=F
else : C=T
error:0%

CART:
surrogate splits

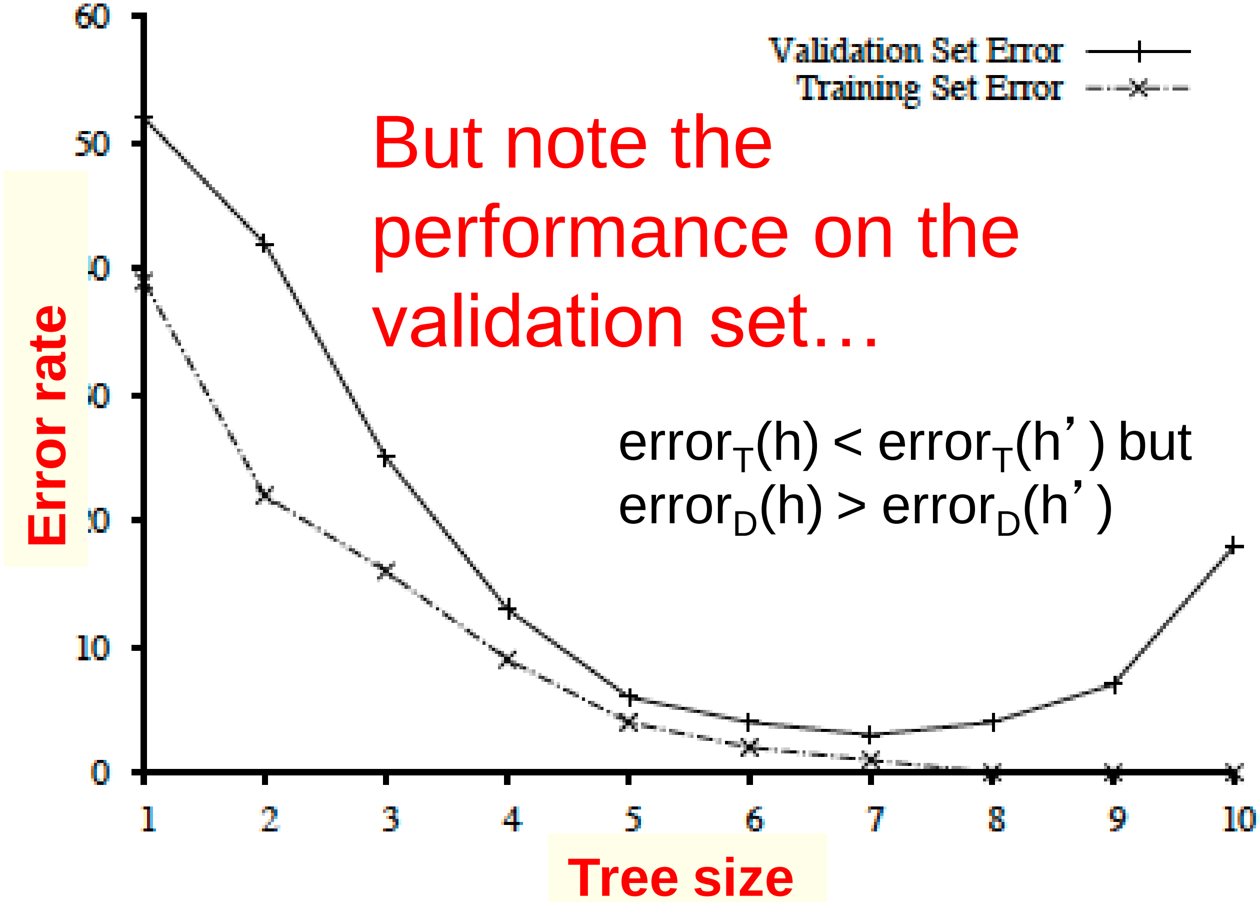
Surrogate Variables order: B<A

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



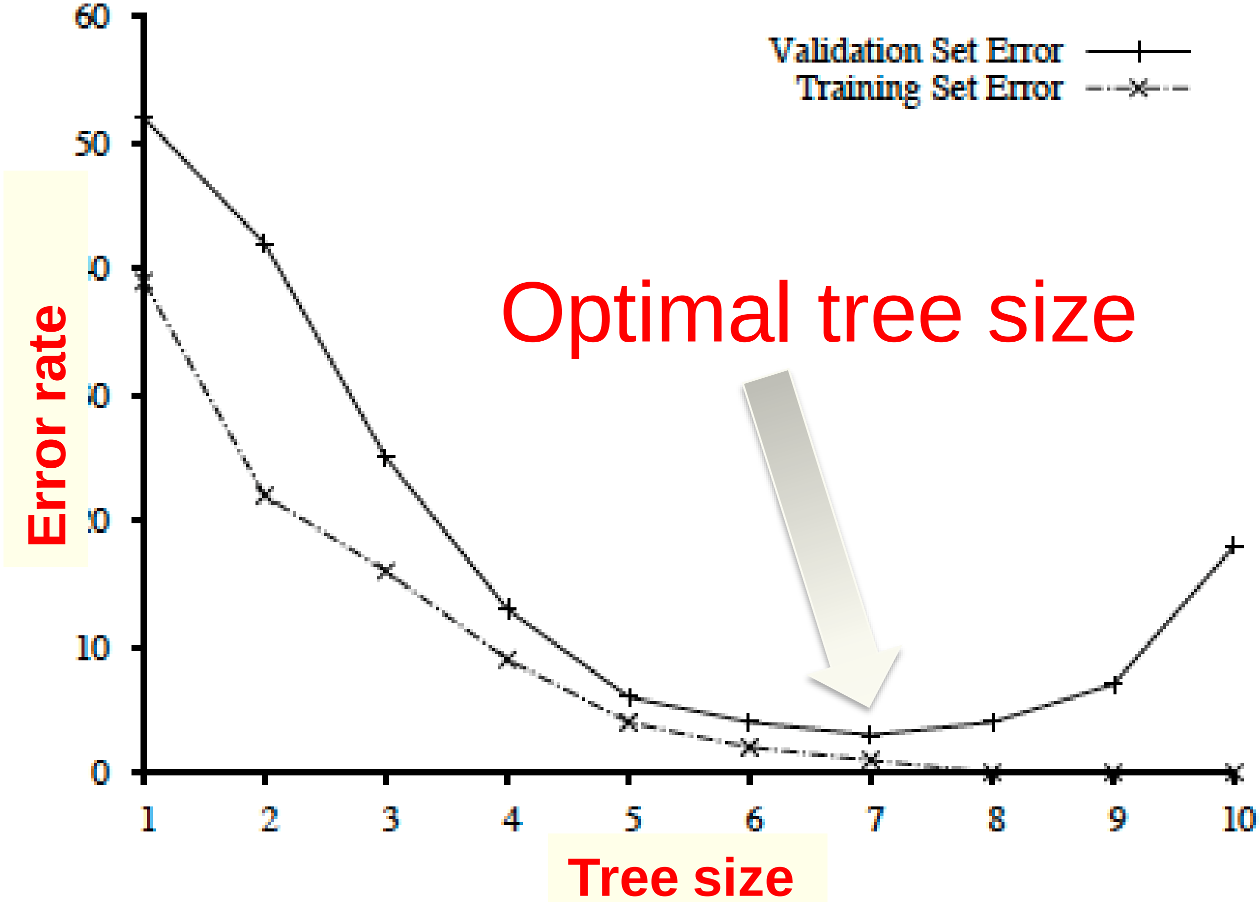
But note the performance on the validation set…

$\text{error}_T(h) < \text{error}_T(h')$ but
$\text{error}_D(h) > \text{error}_D(h')$

Overfitting!

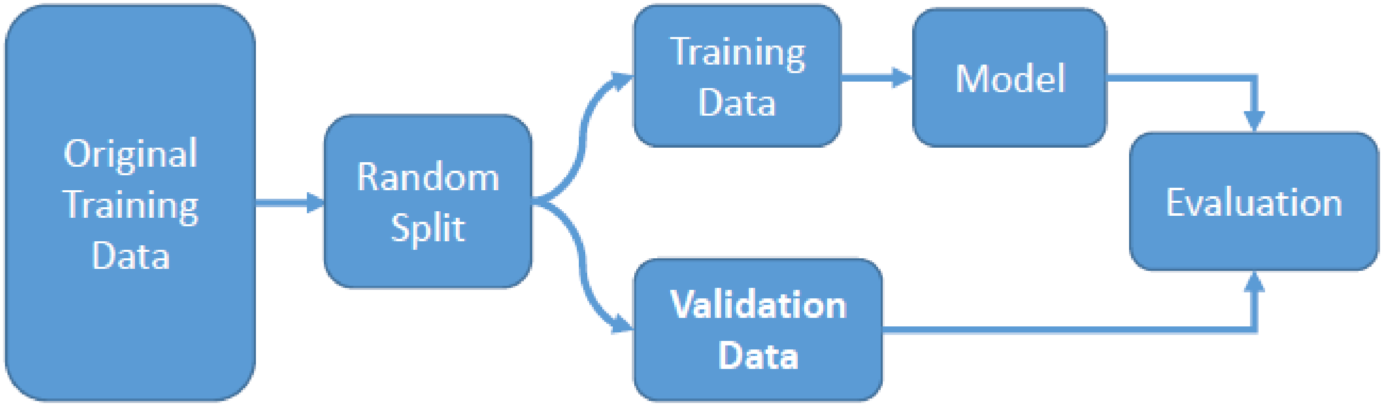
We set tree size as a parameter in our DT learning algorithm
Note: with larger and larger trees,
we just do better and better on the training set!

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!

交叉验证
Cross validation for model selection



K-fold Cross Validation

1. Set hyperparameters
2. For *K* times repeat:
   - Randomly split the original training data into training and validation datasets
   - Train the model on training data and evaluate it on validation data, leading to an evaluation score
3. Average the *K* evaluation scores as the model performance

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 been improved
  - reduce the risk of overfitting

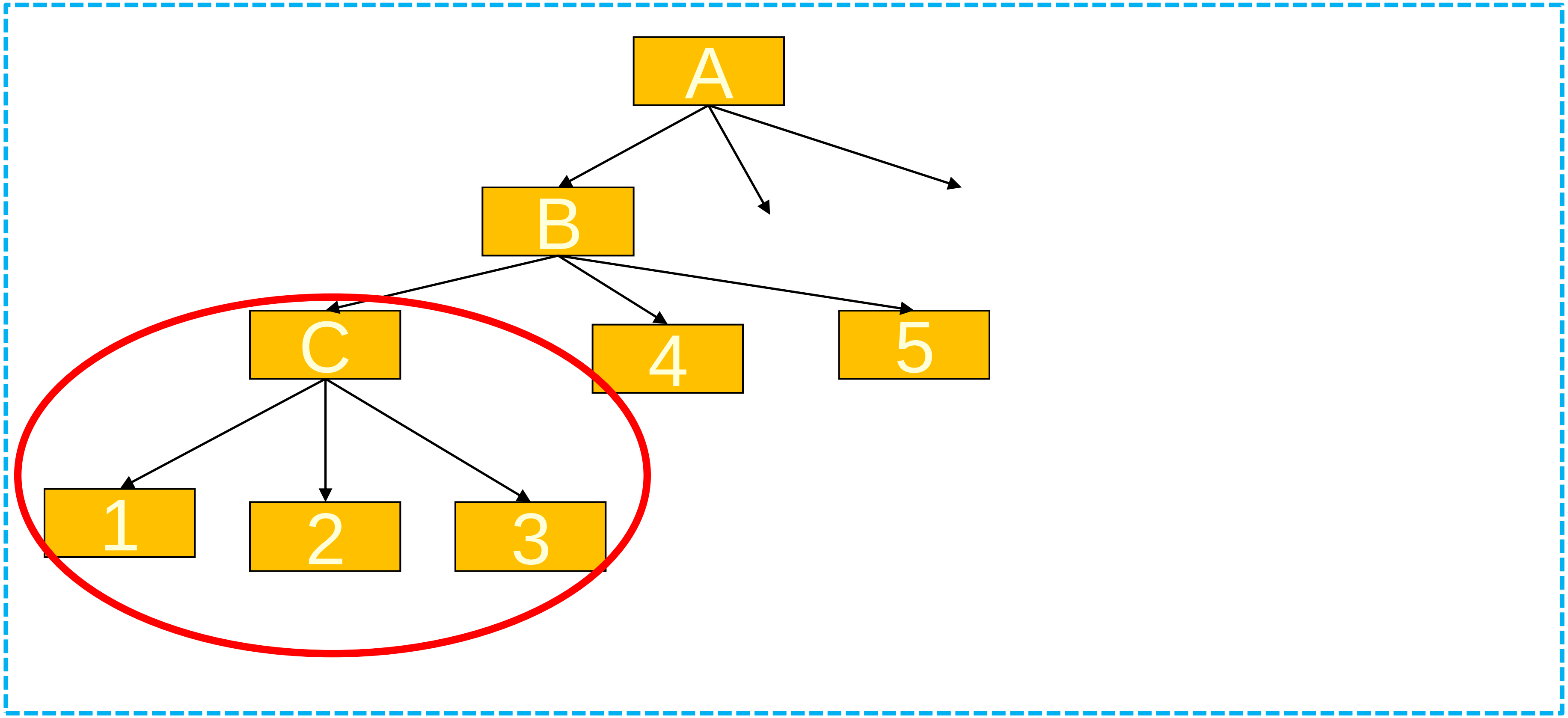
- However, this may be problematic – Why?
  - underfitting
  - Sometimes the current division of some branches can not improve the generalization performance, but the subsequent division on the basis of it may lead to a significant performance improvement

后剪枝
Postpruning

- Postpruning waits until the full decision tree has built and then prunes the attributes
  - Reduced-Error pruning(REP)
  - Pesimistic-Error pruining(PEP)
  - Cost-Complexity pruning(CCP)
  - Pessimistic Pruning
- Two techniques:
  - Subtree Replacement
  - Subtree Raising
- the risk of under fitting is small
- Generalization performance is often better than pre pruning
- The training time is large

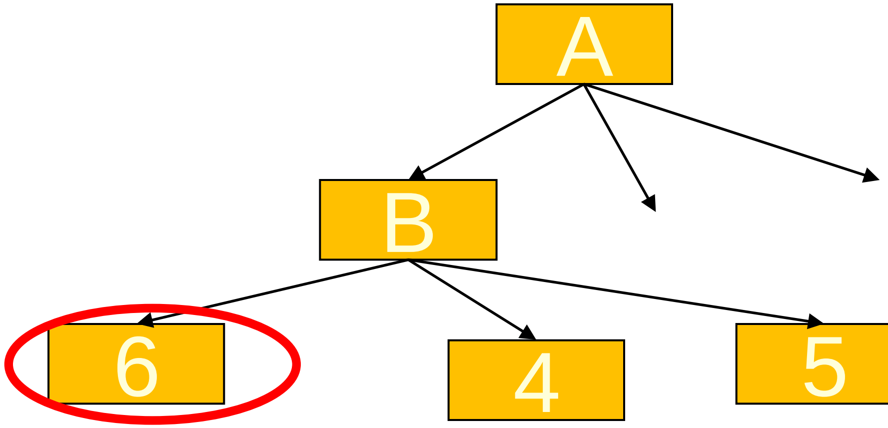
子树代替
Subtree Replacement

- Entire subtree is replaced by a single leaf node



子树代替
Subtree Replacement

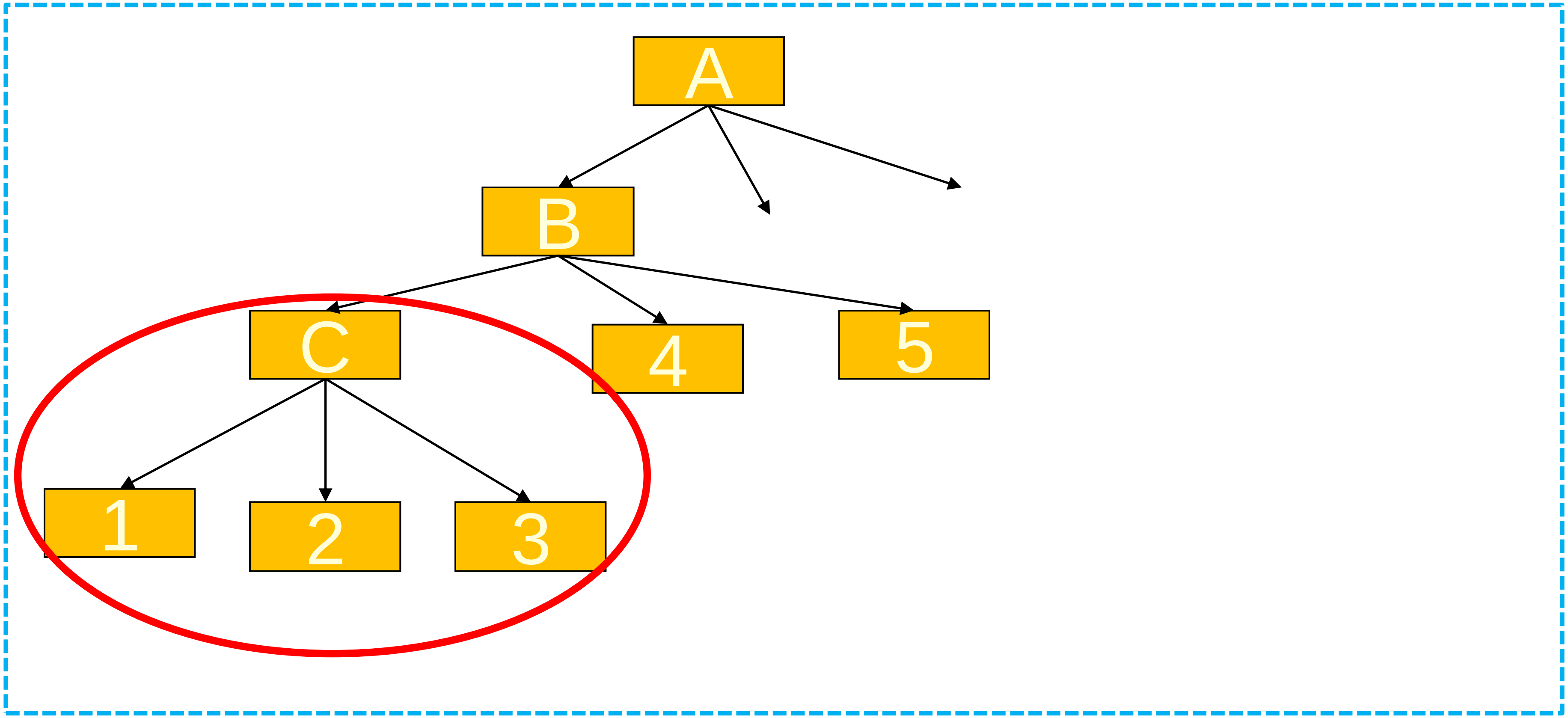
- Entire subtree is replaced by a single leaf node



Generalizes tree a little more,
but may increase accuracy!

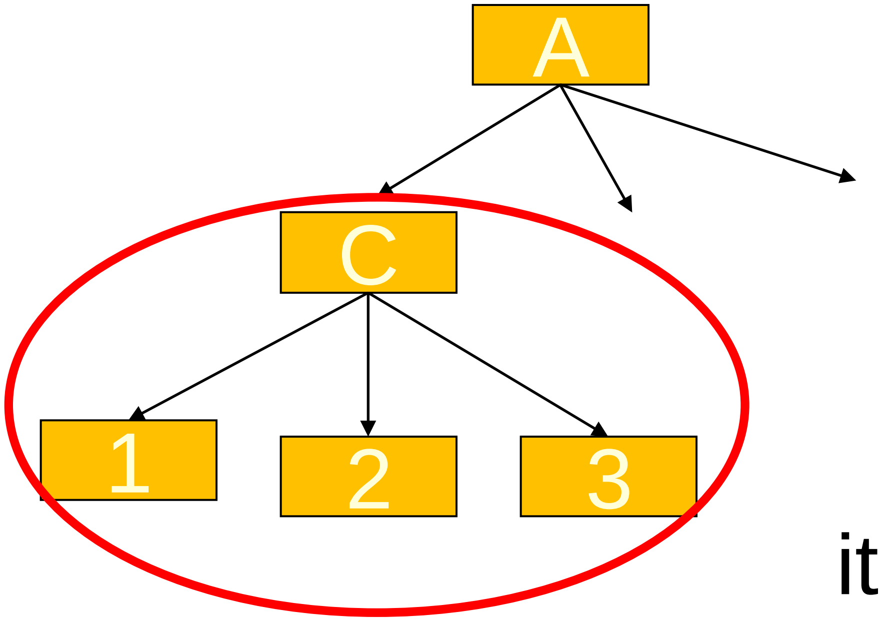
子树提升
Subtree Raising

- Entire subtree is raised onto another node



子树提升
Subtree Raising

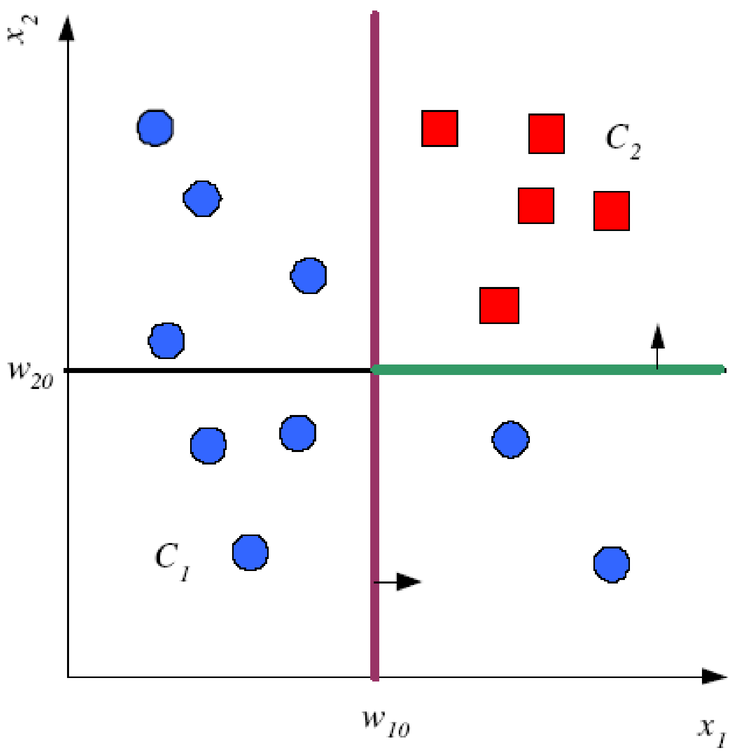
- Entire subtree is raised onto another node

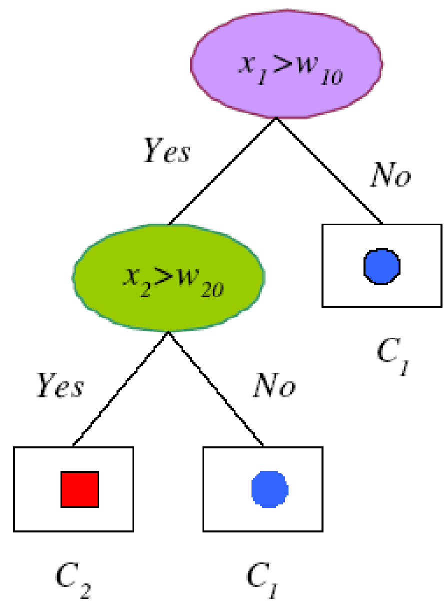


it is very time consuming !

决策树分类界面
Decision Boundary

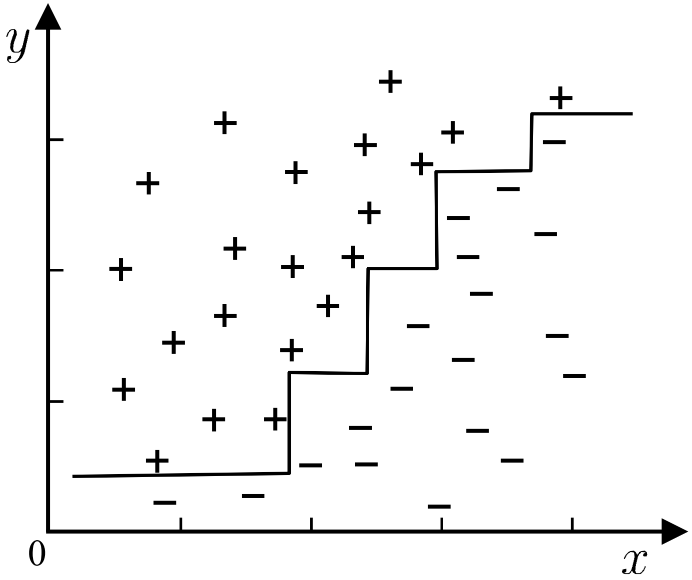
- Decision trees divide the feature space into axis-parallel(hyper-)rectangles
- Each rectangular region is labeled with one label





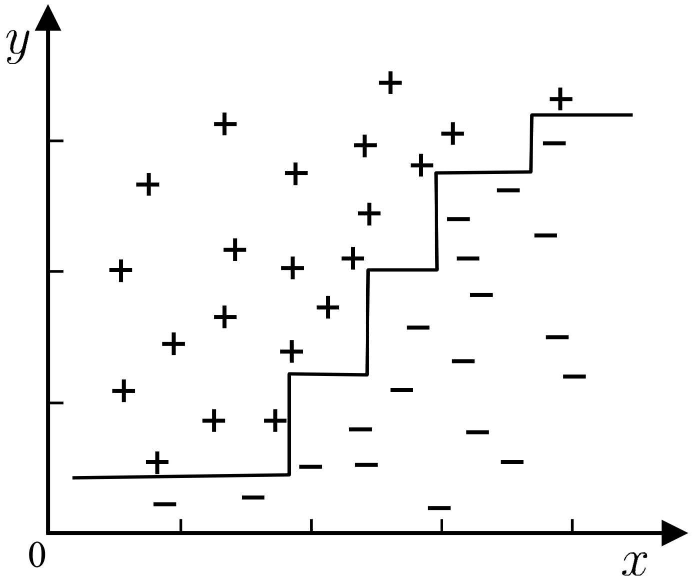
决策树分类界面
Decision Boundary

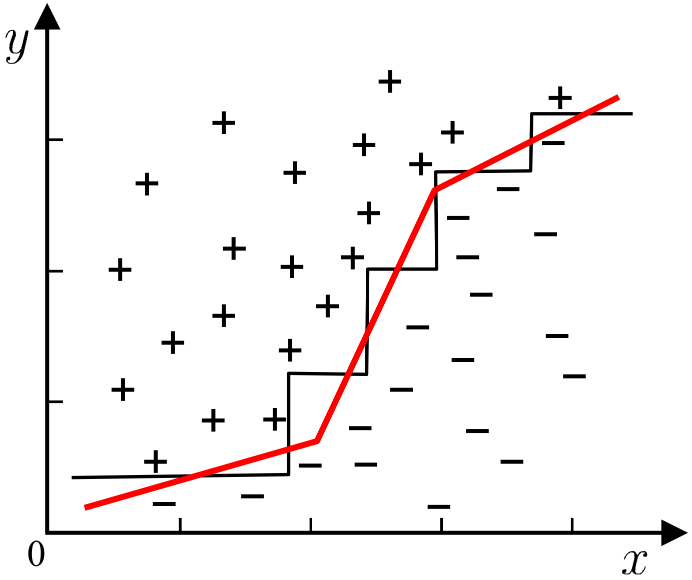
- Many segments must be used to get better approximations when the learning tasks are complex



决策树分类界面
Decision Boundary

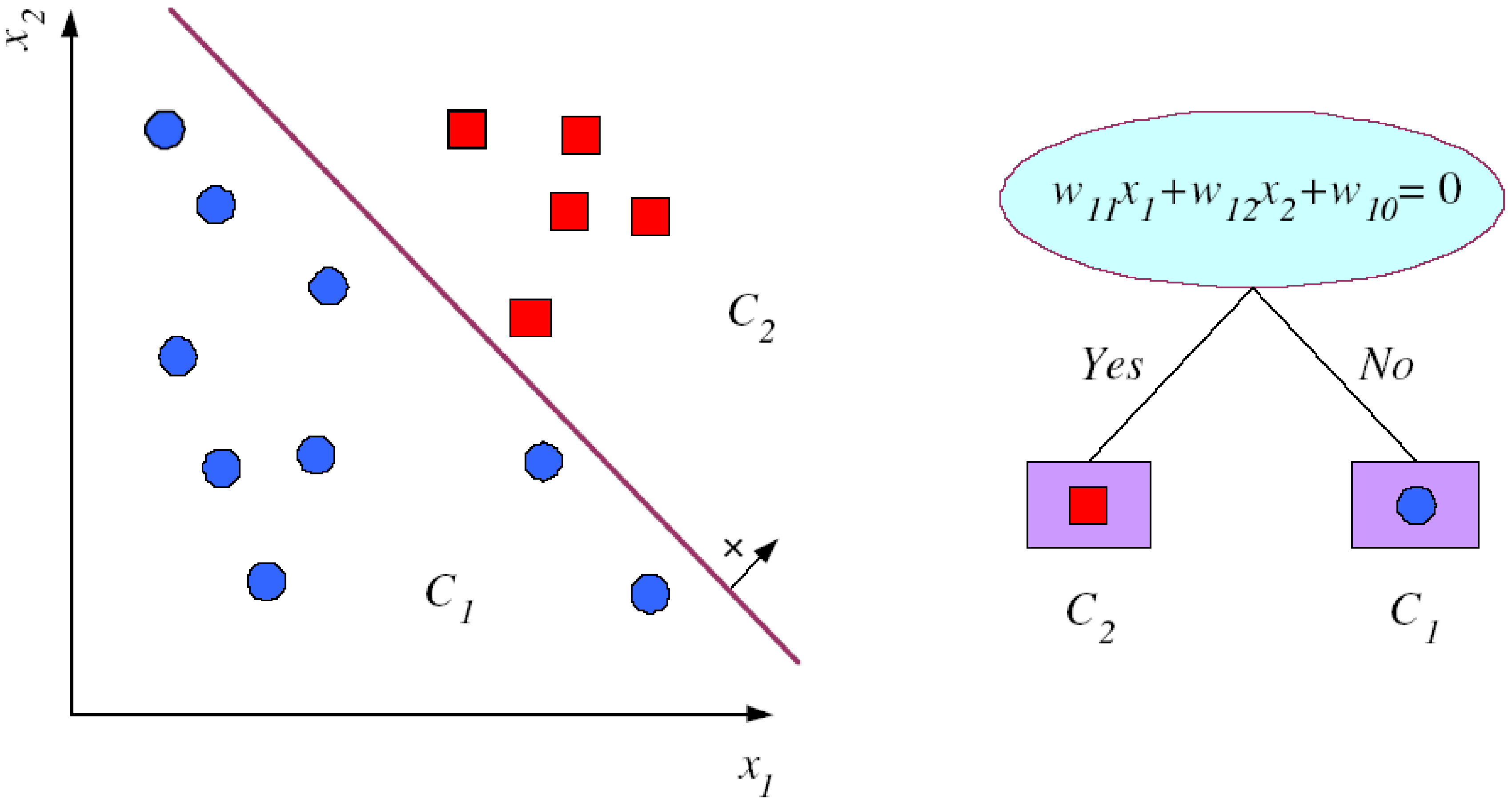
- many segments must be used to get better approximations when the learning tasks are complex



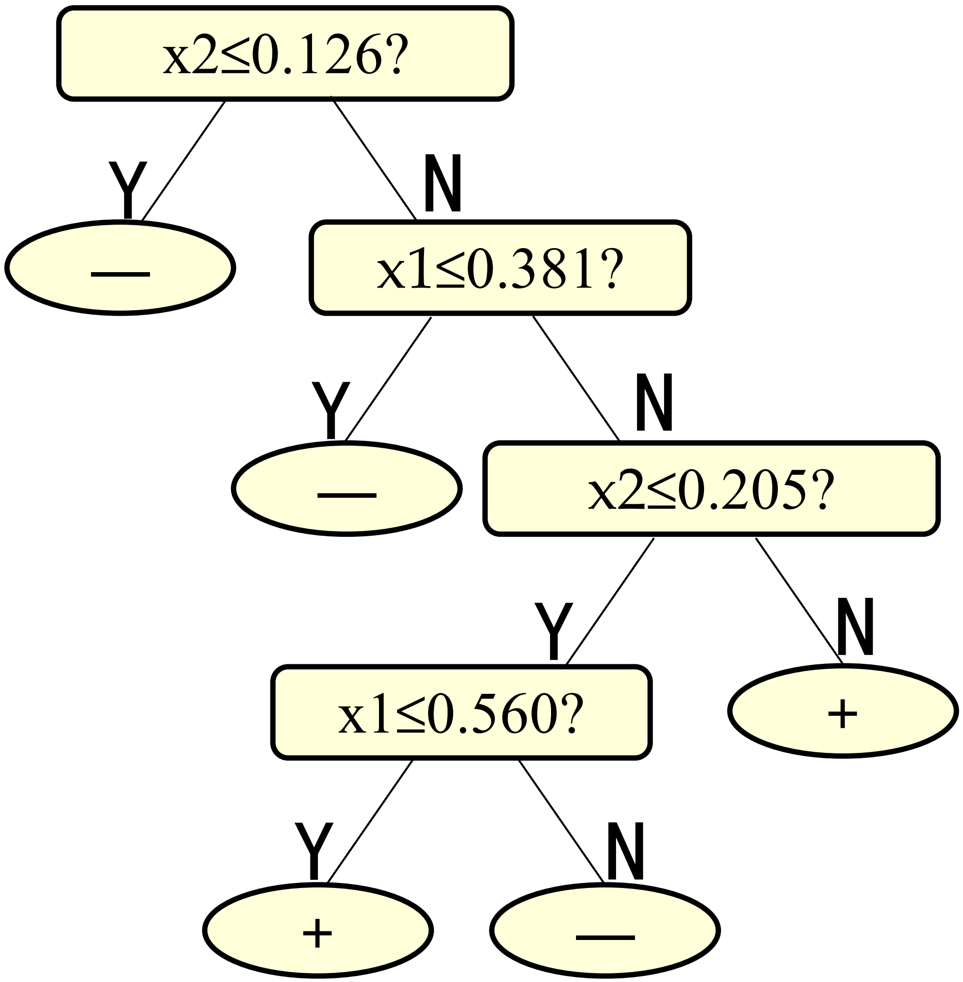


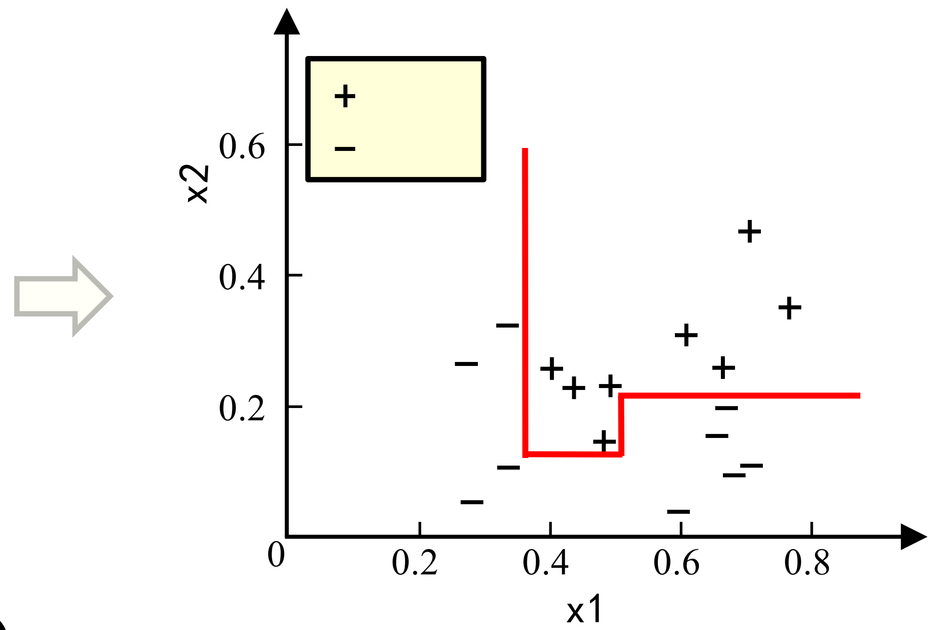
多变量决策树
Multivariate Trees

- Internal nodes can test linear combination of attributes

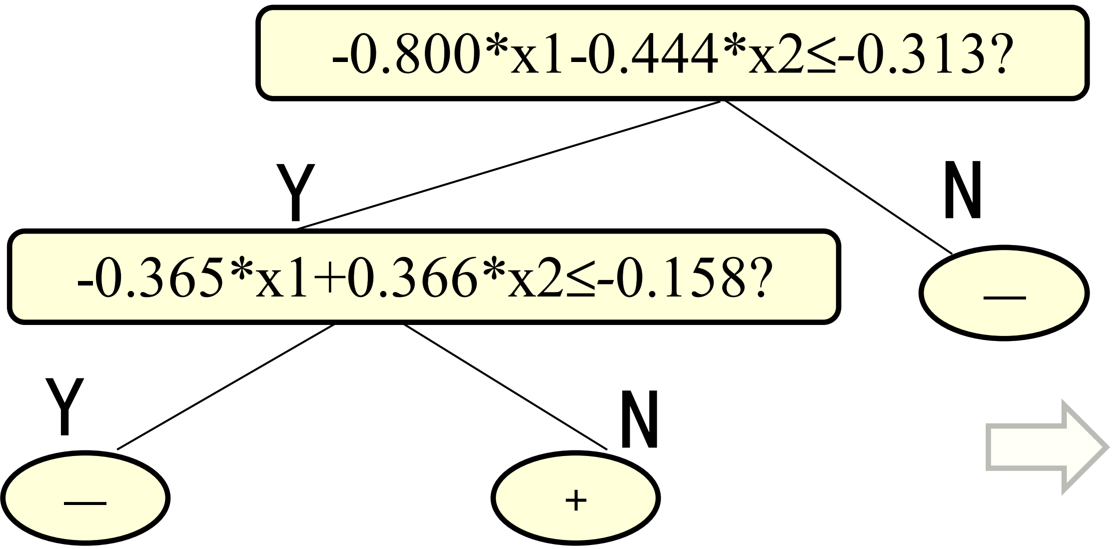


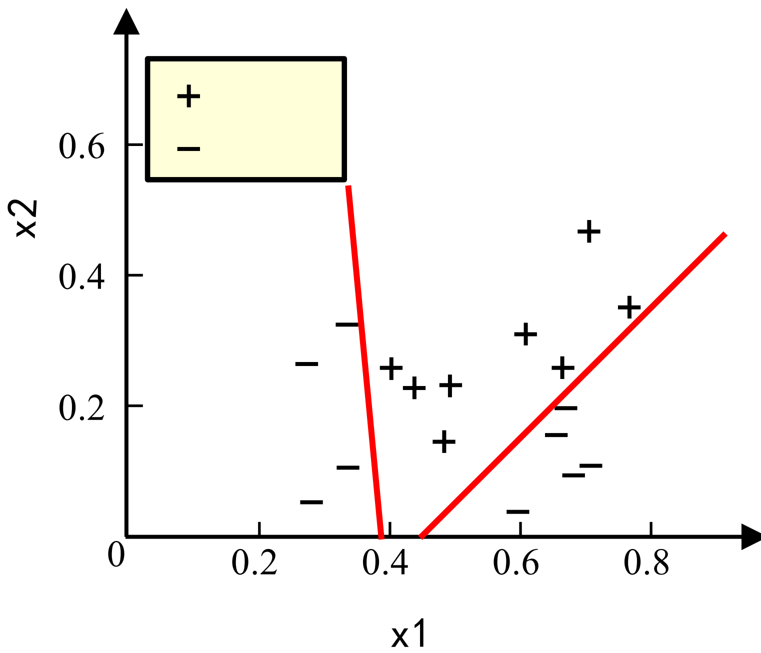
单变量决策树
Univariate Trees





多变量决策树
Multivariate Trees





OC1 [Murthy et al., 1994] [Brodley and Utgoff,1995]

集成模型
Ensemble model

Multiple decision trees are combined to improve the overall performance.

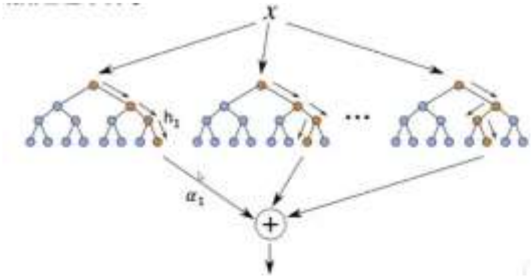
- Random Forest

  Each tree is built using a random subset of the data and a random subset of the feature.

- Boosted Tree

  Each new tree tries to correct the errors of the previous ones (trained on the residuals)

  - XGBoost
  - LightGBM
  - …



监督学习
Supervised Learning

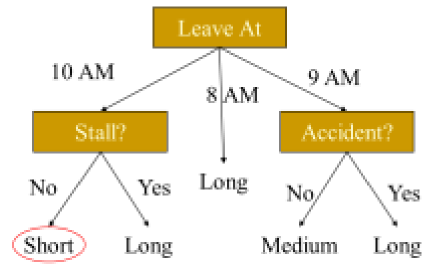
- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

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

a decision tree



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

思考题

- 决策树算法的损失函数是什么？是否要做特征缩放？为什么？