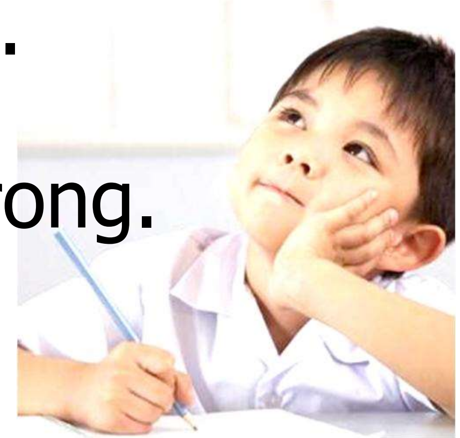
Ch19 Learning from Examples

Shun-Shii Lin
Department of Computer Science & Information Engineering
National Taiwan Normal University

Acknowledgements: This presentation is created by Shun-Shii Lin based on the lecture slides from *The Artificial Intelligence: A Modern Approach by Russell & Norvig,* and various materials from the web.

Ways humans learn things

- Talking, walking, running…
- Learning by reading or being told facts.
- Tutoring.
- Being informed when one is correct/wrong.
- Experience.
- Feedback from the environment.
- Analogy.
- Comparing certain features of existing knowledge to new problems.
- Self-reflection.
- Thinking things in ones own mind, deduction, discovery.



What's involved in Intelligence?

A) Ability to interact with the real world

- perceive, understand, and act.
- speech recognition and understanding.
- image understanding (computer vision).

B) Reasoning and Planning

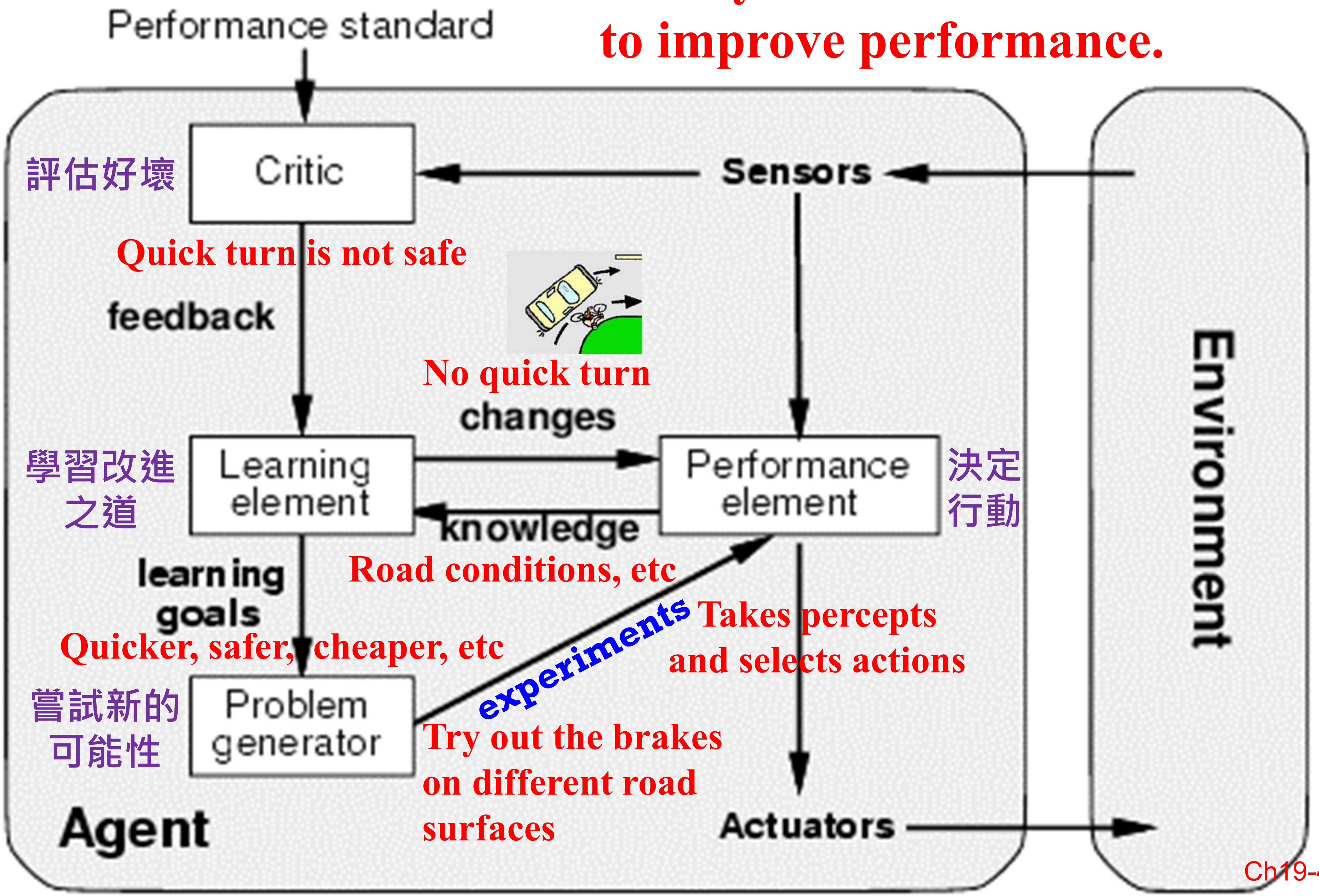
- modelling the external world.
- problem solving, planning, and decision making.
- *ability to deal with unexpected problems, uncertainties.*

C) Learning and Adaptation

- We are continuously learning and adapting.
- We want systems that adapt to us!

Learning agents

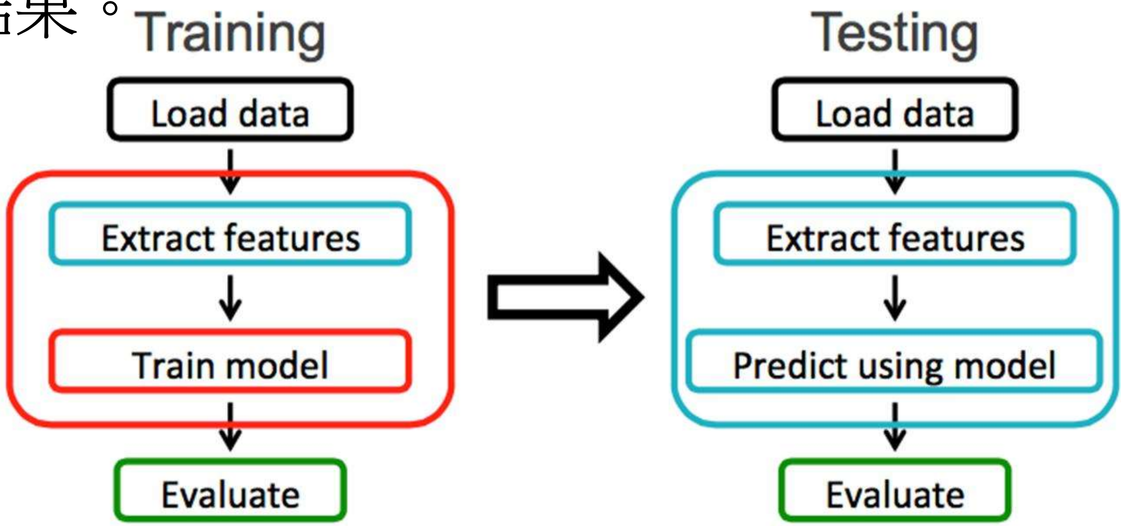
Learning enables an agent to modify its decision mechanisms to improve performance.



19.1 Forms of Learning

Three types of feedback:

一、監督學習 (Supervised learning)：從給定的訓練數據集合中，學習出一個模式（函數 / **learning model**），當新的數據到來時，可以根據這個模式預測結果。



→監督學習的訓練集合包括輸入特徵和輸出結果。訓練集合中的結果是由人標註的。輸出可以是一個連續的值 (也就是迴歸分析，**regression**)，或是一個非連續的分類(**classification**)。

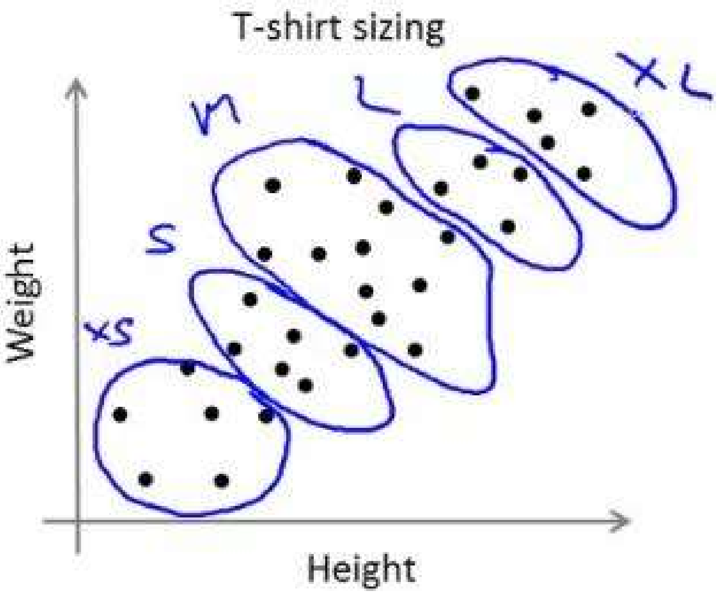
→**Example–an agent is presented with many camera images and is told which ones contain cats; the agent learns a function from images to a Boolean output(whether the image contains cats).**

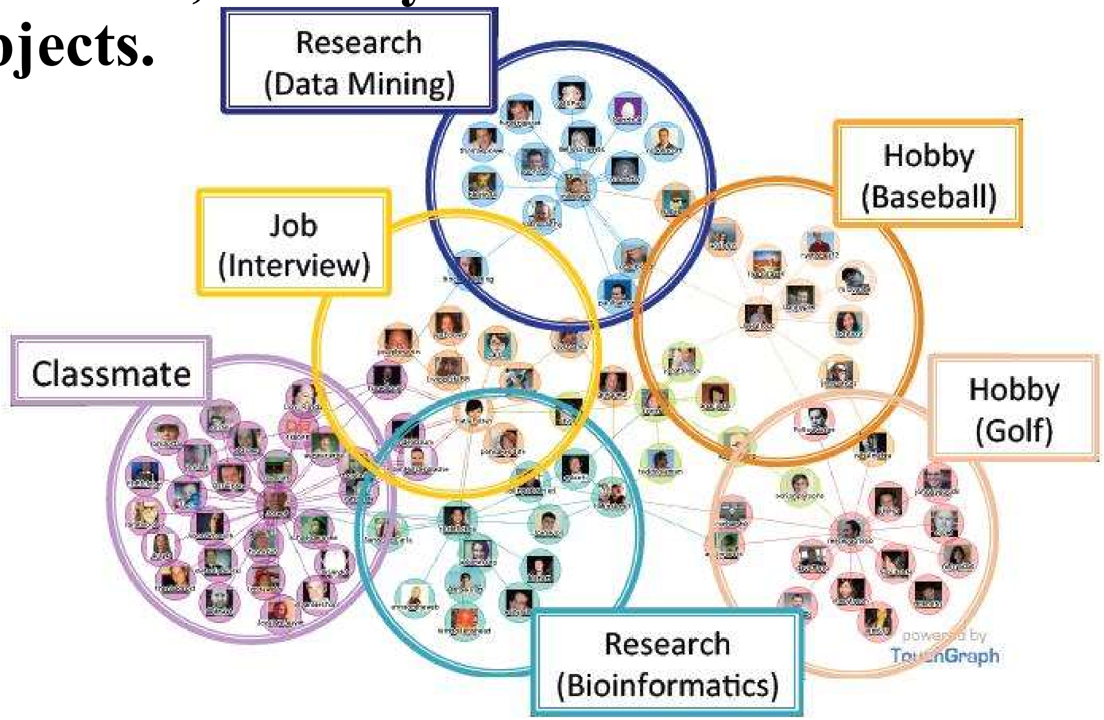
Three types of feedback:

二、無監督學習 (unsupervised learning)：與監督學習相比，訓練集合沒有人為標註的結果。常見的無監督學習如聚類(Clustering)。

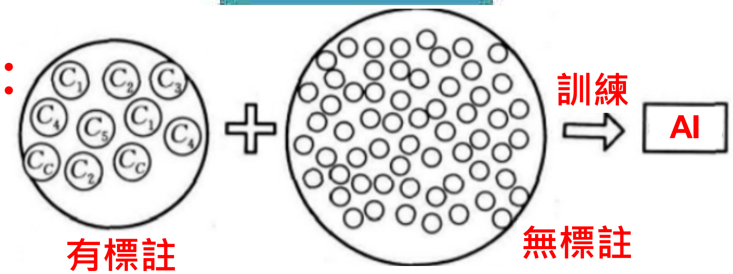
→**learn a patterns in the input when no specific output values are supplied.**

→**Example: identity T-shirt sizes, identify communities in the Internet, identify sky objects.**





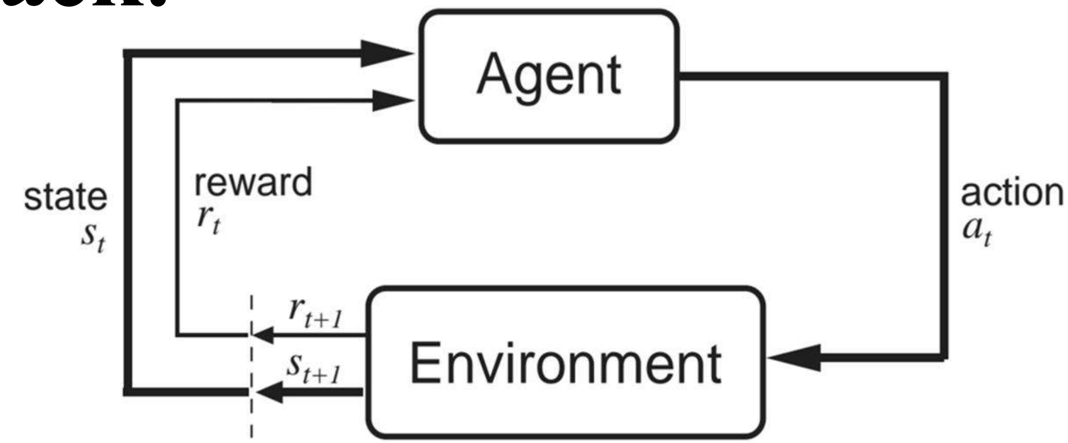
半監督學習 (Semi-supervised learning)：介於監督學習與無監督學習之間。訓練集合只有一部份有人為標註。



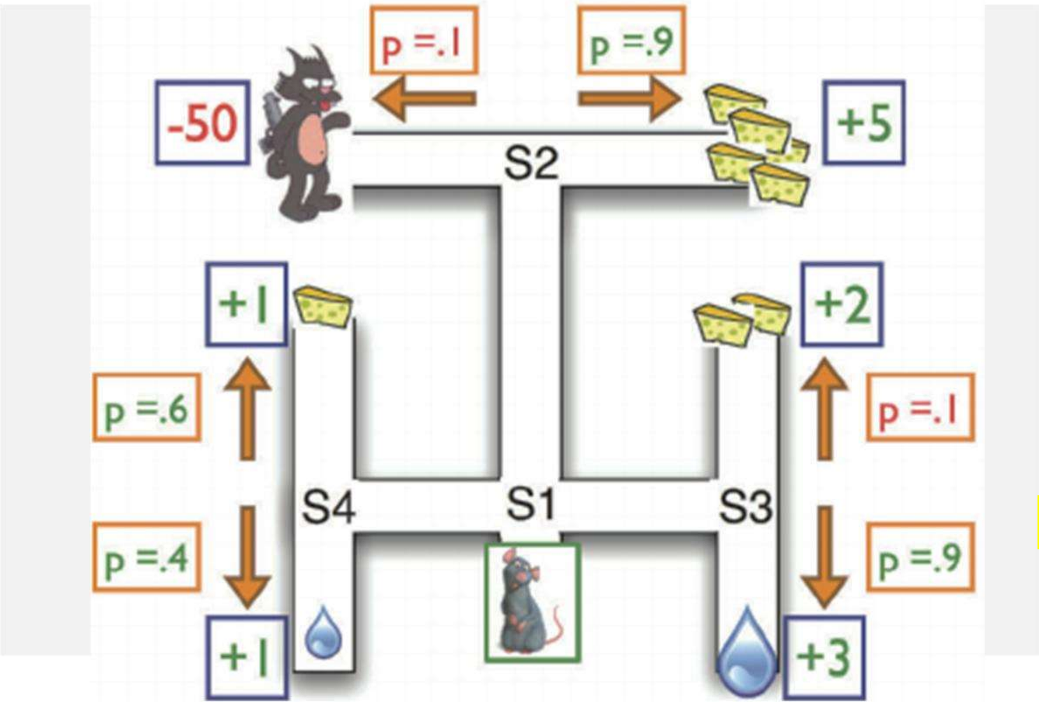
Three types of feedback:

三、增強學習 (reinforcement learning)：通過大量實驗來學習如何行動。根據反饋的結果期望值來做出決策。

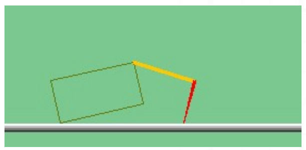
→learn from reinforcement or (occasional) rewards.

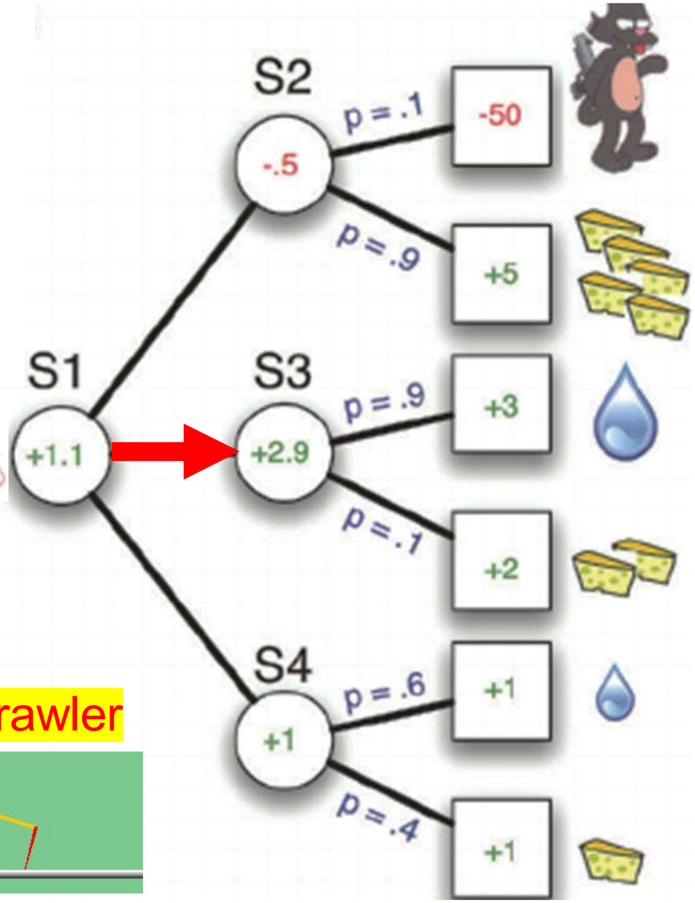


→Example: The rat at the bottom of a maze can navigate and obtain various rewards.



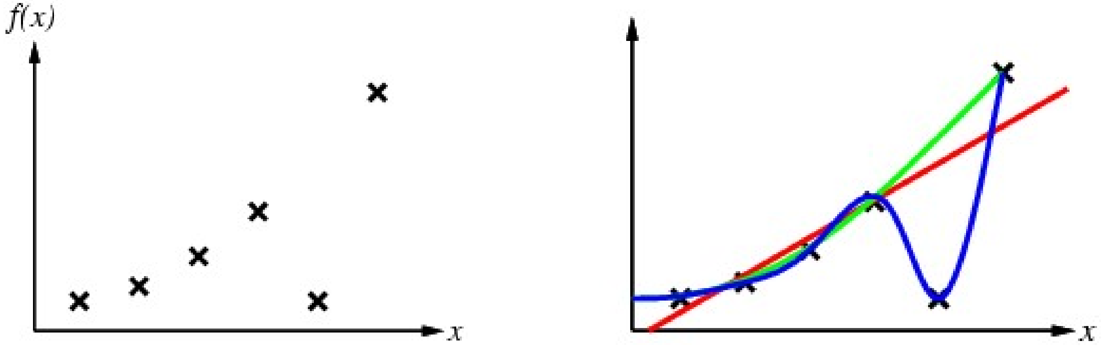
Demo Crawler





19.2 Supervised Learning

E.g., curve fitting:



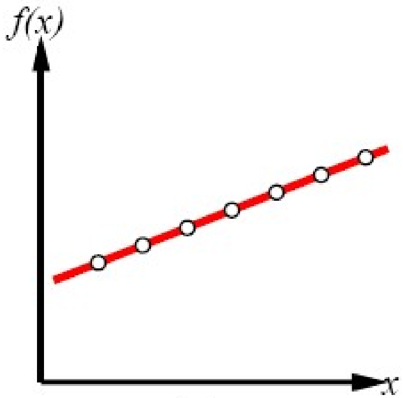
- In deterministic supervised learning the aim is to recover the unknown function f given *examples* $(\mathbf{x}, f(\mathbf{x}))$, where $\mathbf{x}$ is the input (vector).
- In *pure inductive inference* (or *induction*) the result is a *hypothesis* h, which is function that approximates f.
- A good hypothesis will *generalize* well – will predict unseen instances correctly.
- The hypothesis is chosen from a hypothesis space $\mathcal{H}$.
- For example, when both x and $f(x)$ are real numbers, then $\mathcal{H}$ can be, e.g., the set of polynomials of degree at most k:

$$3x^2 + 2, \qquad x^{17} - 4x^3, \ldots$$

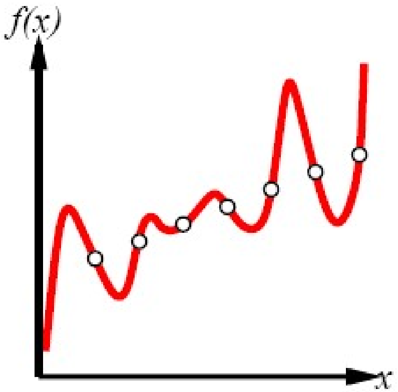
Multiple consistent hypotheses?

Polynomials of degree at most k

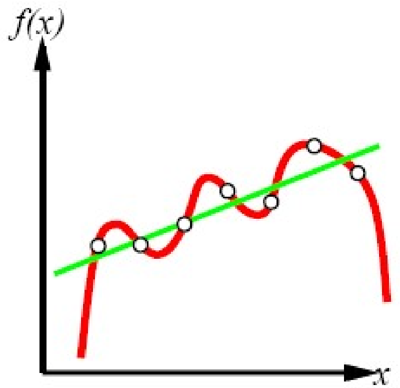
How to choose from among multiple consistent hypotheses?



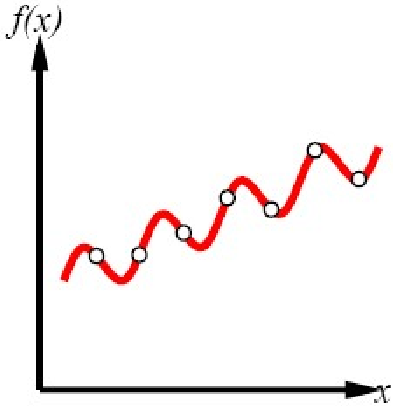
(a)
Linear hypothesis



(b)
Degree 7 polynomial hypothesis



(c)
Degree 6 polynomial and approximate linear fit



(d)
Sinusoidal hypothesis

Ockham's razor: prefer the simplest hypothesis that consistent with the data.

「奧坎的剃刀」：極簡之法則，14世紀邏輯學家William of Occam提出，他說「切勿浪費較多東西，去做『用較少的東西也可以做好的事情』」。

Example of supervised learning: classification

- We lend money to people.
- We have to predict whether they will pay us back or not.
- People have various (say, binary) features:
  - do we know their Address? do they have a Criminal record? high Income? Educated? Old? Unemployed?
- We see examples: (Y = paid back, N = not)

  +a, -c, +i, +e, +o, +u: Y
  -a, +c, -i, +e, -o, -u: N
  +a, -c, +i, -e, -o, -u: Y
  -a, -c, +i, +e, -o, -u: Y
  -a, +c, +i, -e, -o, -u: N
  -a, -c, +i, -e, -o, +u: Y
  +a, -c, -i, -e, +o, -u: N
  +a, +c, +i, -e, +o, -u: N

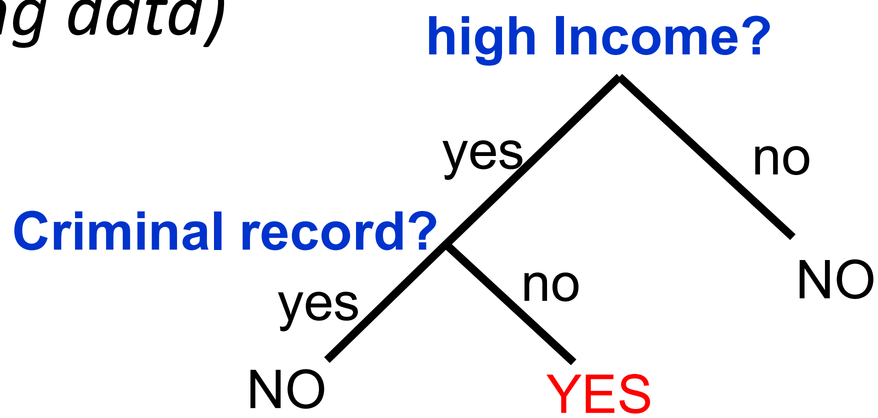
- Next person is +a, -c, +i, -e, +o, -u. Will we get paid back?

Classification…

- We want some hypothesis h that predicts whether we will be paid back.

  +a, -c, +i, +e, +o, +u: Y
  -a, +c, -i, +e, -o, -u: N
  +a, -c, +i, -e, -o, -u: Y
  -a, -c, +i, +e, -o, -u: Y
  -a, +c, +i, -e, -o, -u: N
  -a, -c, +i, -e, -o, +u: Y
  +a, -c, -i, -e, +o, -u: N
  +a, +c, +i, -e, +o, -u: N

  Next person is +a, -c, +i, -e, +o, -u. Will we get paid back?
- Lots of possible hypotheses: will be paid back if…
  - Income is high *(wrong on 2 occasions in training data)*
  - Income is high and no Criminal record *(always right in training data)*
  - (Address is known AND ((NOT Old) OR Unemployed)) OR ((NOT Address is known) AND (NOT Criminal Record)) *(always right in training data)*
- Which one seems best?
- Anything better?



19.3 Learning decision trees

Problem: decide whether to wait for a table at a restaurant, based on the following 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)

Attribute-based representations

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

訓練資料集合

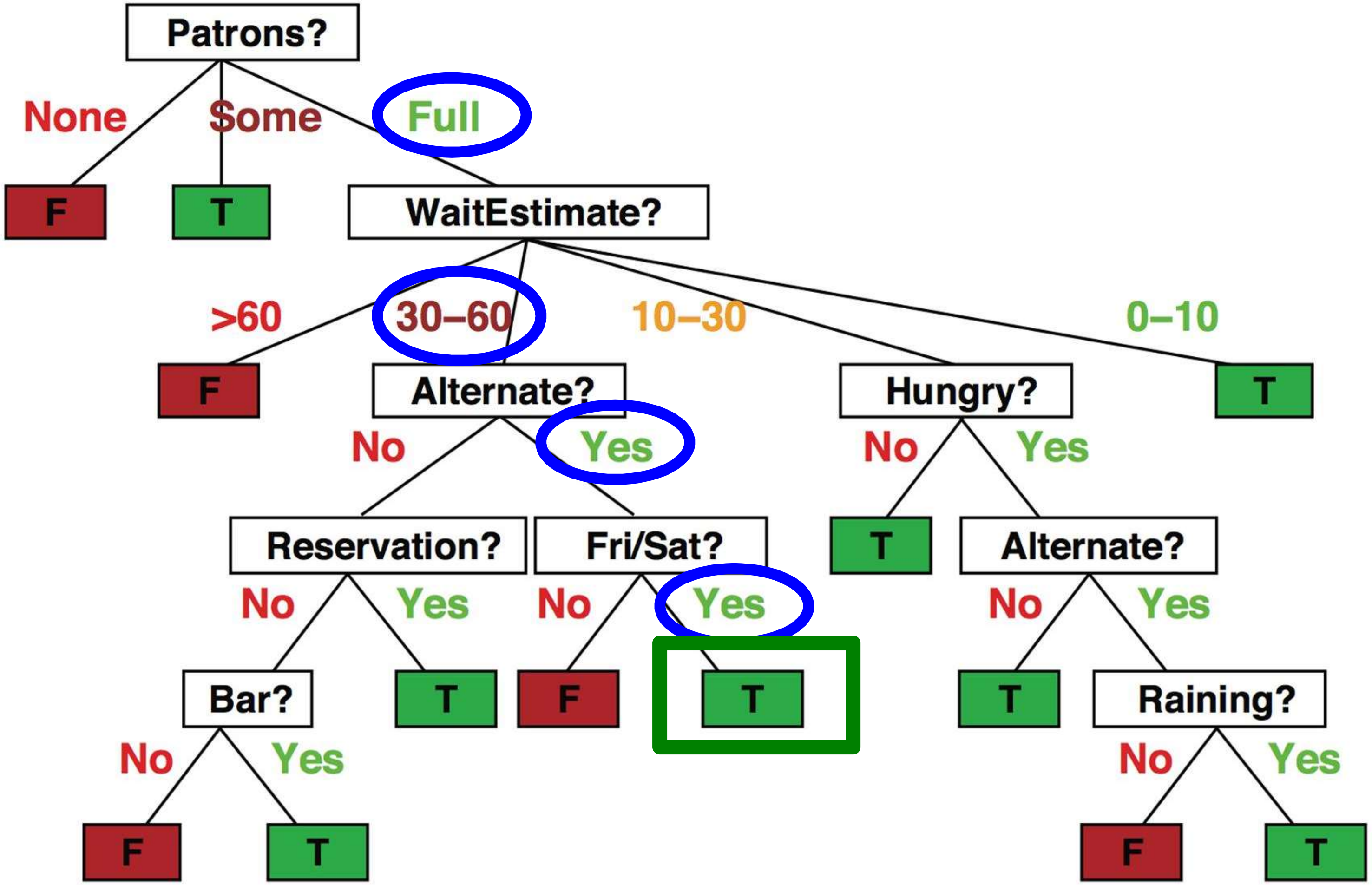
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F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

6+ 6-

Decision trees

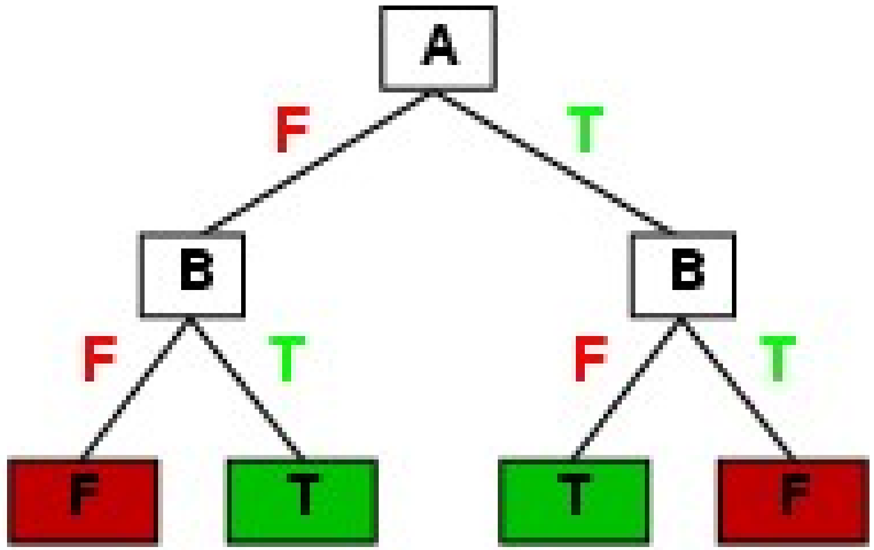
- One possible representation for hypotheses.
- E.g., here is the “true” tree for deciding whether to wait:



Expressiveness

- Decision trees can express any function of the input attributes.
- E.g., for Boolean functions, truth table row → path to leaf:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- Trivially, there is a consistent decision tree for any training set with one path to leaf for each example (unless *f*(x) nondeterministic in *x*), but it probably won't generalize to new examples.
- Prefer to find more compact decision trees.

Hypothesis spaces

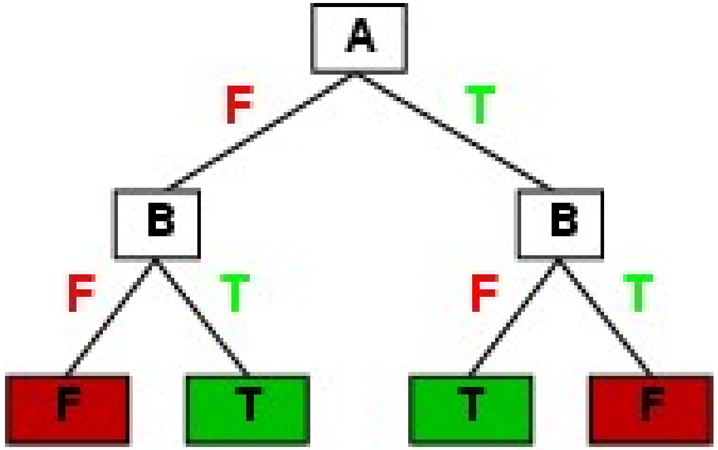
How many distinct decision trees with *n* Boolean attributes?

= number of Boolean functions

= number of distinct truth tables with 2^n rows = 2^{2^n}

- E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees.

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



How many purely conjunctive hypotheses (e.g., *Hungry* $\wedge$ $\neg$*Rain*)?

- Each attribute can be in (positive), in (negative), or out (不用)

  $\Rightarrow$ 3^n distinct conjunctive hypotheses.

- More expressive hypothesis space$\Rightarrow$ may get worse predictions.
  - increases chance that target function can be expressed.
  - increases number of hypotheses consistent with training set.

Decision Tree Learning (DTL)

- Aim: find a small tree consistent with the training examples.
- Idea: (recursively) choose "most significant" attribute as root of (sub)tree.

基本的演算法概念:

1.資料設定：將原始資料分成兩組，一部分為訓練資料，一部分為測試資料

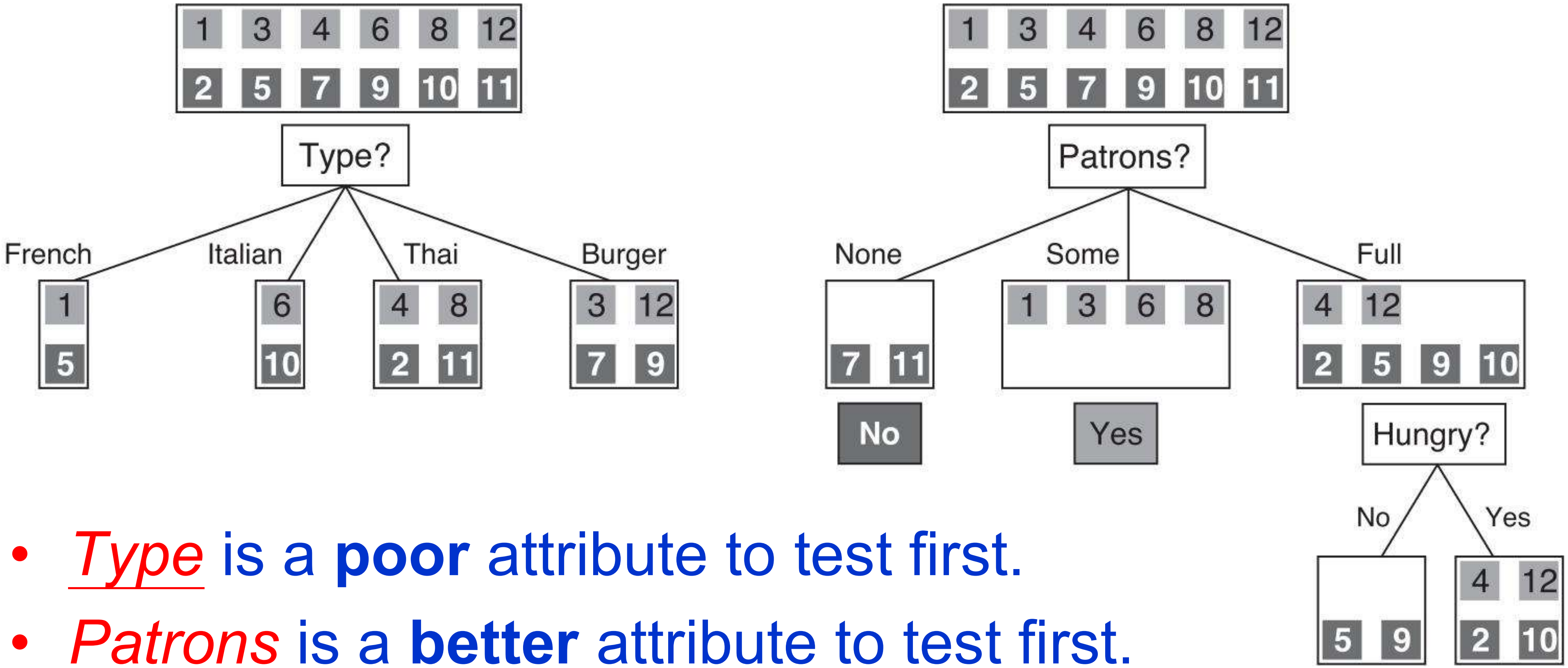
2.決策樹生成：使用訓練資料來建立決策樹，而在每一個內部節點，則依據屬性選擇指標 (如：資訊理論(Information Theory)...) 來評估選擇哪個屬性做分支的依據。又稱節點分割 (Splitting Node)

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
else if all *examples* have the same classification **then return** the classification
else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
else
  $A \leftarrow \mathrm{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*) //Choose-Best-Attribute
  tree ← a new decision tree with root test A
  for each value v_k of A **do**
    $exs \leftarrow \{e : e \in examples$ **and** $e.A = v_k\}$
    subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − A, *examples*)
    add a branch to *tree* with label $(A = v_k)$ and subtree *subtree*
  return *tree*

Choosing an attribute

- Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative“.



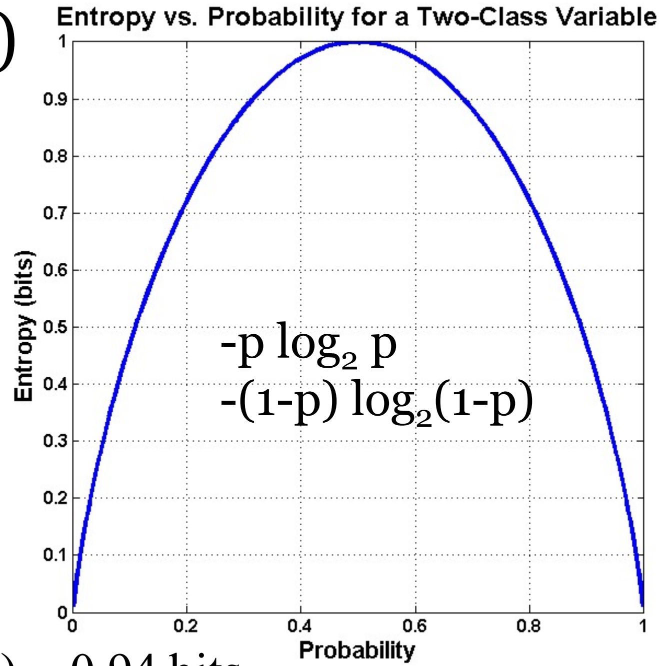
- *Type* is a **poor** attribute to test first.
- *Patrons* is a **better** attribute to test first.
  Then *Hungry* is a **fairly good** second test.
- How to choose the best attribute?

Using information theory

1948年Claude Shannon提出

- To implement **Choose-Best-Attribute** in the DTL algorithm, we can use Information Content (Entropy**熵**):

$$I(P(v_1), \ldots, P(v_n)) = \Sigma_{i=1} -P(v_i) \log_2 P(v_i)$$



熵 (亂度，Entropy)可衡量一個隨機變數的不確定性，熵愈大，則不確定性愈高。

【例】若硬幣公平，則擲出正、反機率一樣 (最亂)。若硬幣不公平，則愈不亂。

Ex: 若擲14次硬幣，出現了9個正面與5個反面，則熵=

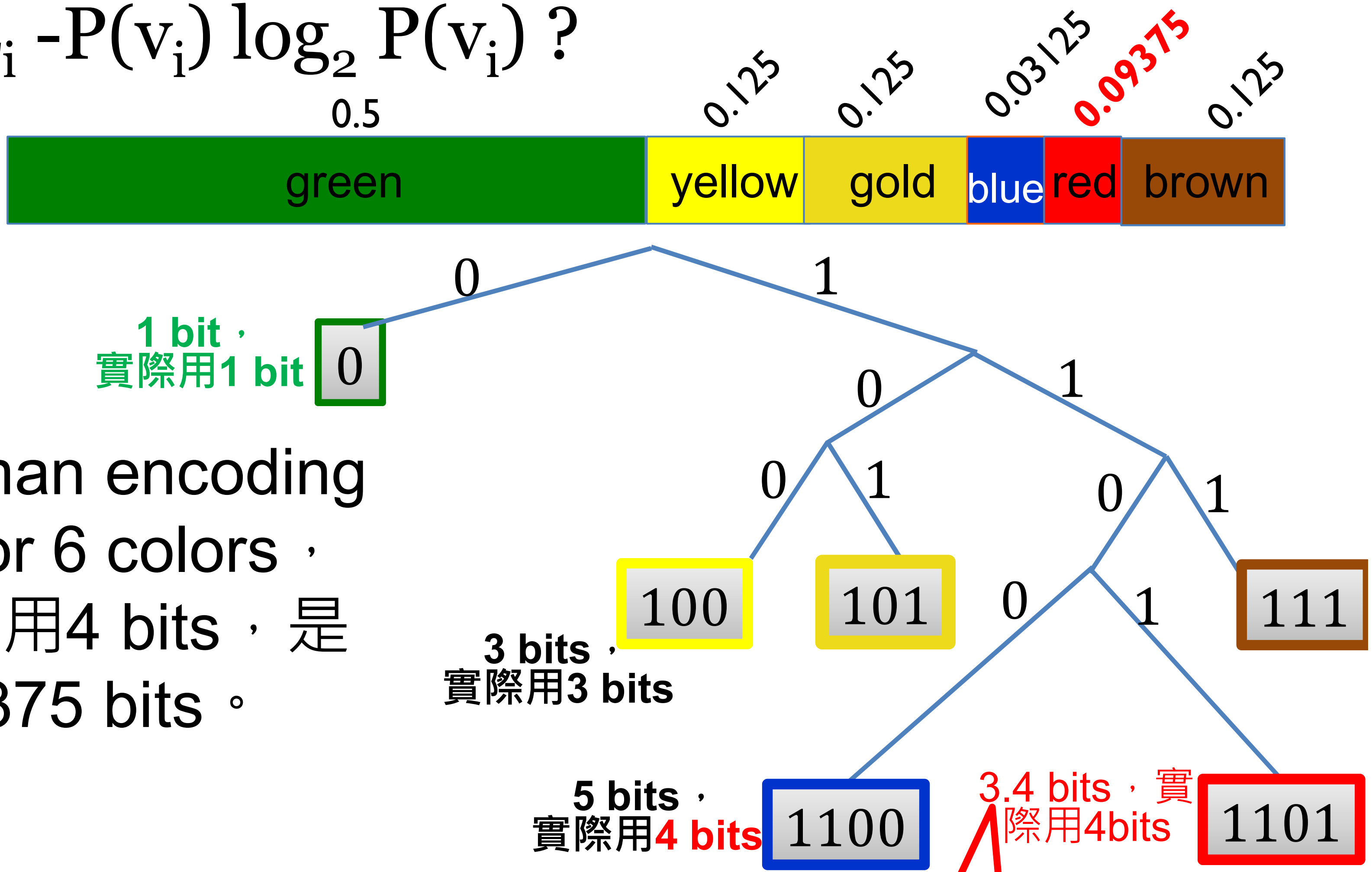
$$-\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right)-\left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) = -0.64*(-0.64) - 0.36*(-1.49) = 0.94 \text{ bits}$$

若擲出正面與反面的數量是一樣，則熵爲 1 bit(最亂)。

若怎麼擲都只出現正面，則熵爲 0 bit(最不亂)。

【例】預測明天天氣，機率：晴天25%，下雨25%，陰天25%，下雪25%，則熵=4*(-0.25)($\log_2$0.25)=2 bits ⇒平均位元長度=2。

Why $\Sigma_i -P(v_i)\log_2 P(v_i)$?



想像Huffman encoding scheme for 6 colors，但red不是用4 bits，是用3.4150375 bits。

$$\text{平均長度} =$$

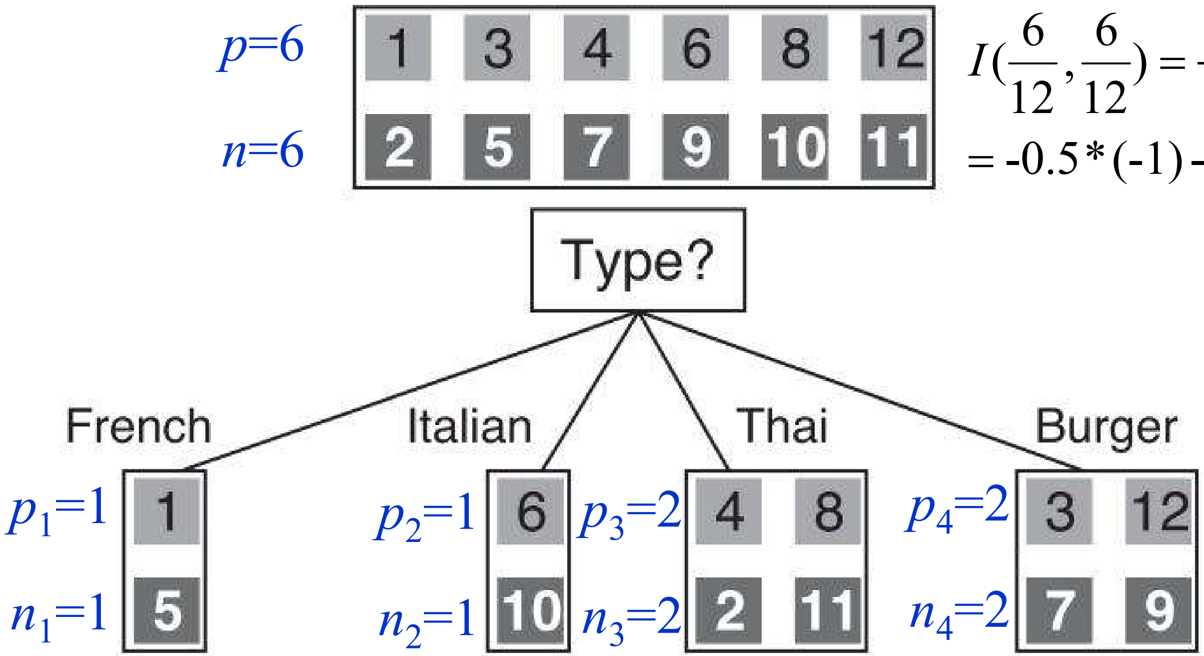
$$= 0.5 * \boxed{\log_2 \frac{1}{0.5}} + 0.125 * \boxed{\log_2 \frac{1}{0.125}} + 0.125 * \log_2 \frac{1}{0.125} + 0.03125 * \boxed{\log_2 \frac{1}{0.03125}} + 0.09375 * \boxed{\log_2 \frac{1}{0.09375}} + 0.125 * \log_2 \frac{1}{0.125}$$

$$= \frac{1}{2} * \boxed{1} + \frac{1}{8} * \boxed{3} + \frac{1}{8} * 3 + \frac{1}{32} * \boxed{5} + \frac{3}{32} * \boxed{3.4150375} + \frac{1}{8} * 3 = 2.10141$$

$$\text{故} Entropy = \sum_x p(x) \log_2 \frac{1}{p(x)} = -\sum_x p(x) \log_2 p(x)$$

Information gain(IG)：用Type來分割

$$IG(Attribute) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - \sum_{i=1}^{v} \frac{p_i+n_i}{p+n} I\left(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\right)$$



$$I\left(\frac{6}{12}, \frac{6}{12}\right) = -\left(\frac{6}{12}\right)\log_2\left(\frac{6}{12}\right) - \left(\frac{6}{12}\right)\log_2\left(\frac{6}{12}\right)$$

$$= -0.5*(-1) - 0.5*(-1) = 1$$

Gain值愈小，表示分割後資料的凌亂程度愈大，用來分類資料會愈不好。

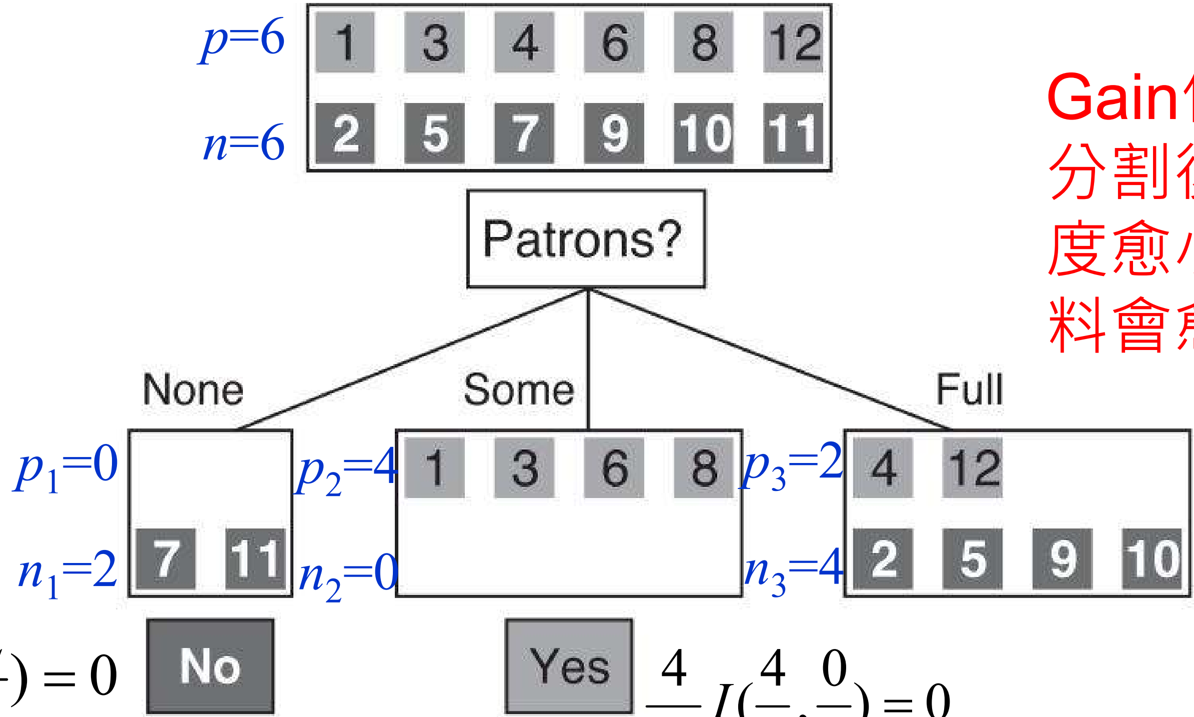
$$\frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) = \frac{2}{12} \quad \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) = \frac{2}{12} \quad \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) = \frac{4}{12} \quad \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) = \frac{4}{12}$$

$$IG(Type) = I\left(\frac{6}{12}, \frac{6}{12}\right) - \left[\frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right)\right] = 0 \text{ bits}$$

- *Type* is a **poor** attribute to test first.

Information gain(IG)：用Patrons來分割

$$I(\frac{6}{12},\frac{6}{12}) = -(\frac{6}{12})\log_2(\frac{6}{12}) - (\frac{6}{12})\log_2(\frac{6}{12})$$
$$= -0.5*(-1) - 0.5*(-1) = 1$$



Gain值愈大，表示分割後資料的凌亂程度愈小，用來分類資料會愈佳。

$$\frac{6}{12}I(\frac{2}{6},\frac{4}{6}) = 0.459$$

$$\frac{2}{12}I(\frac{0}{2},\frac{2}{2}) = 0$$

$$\frac{4}{12}I(\frac{4}{4},\frac{0}{4}) = 0$$

the highest

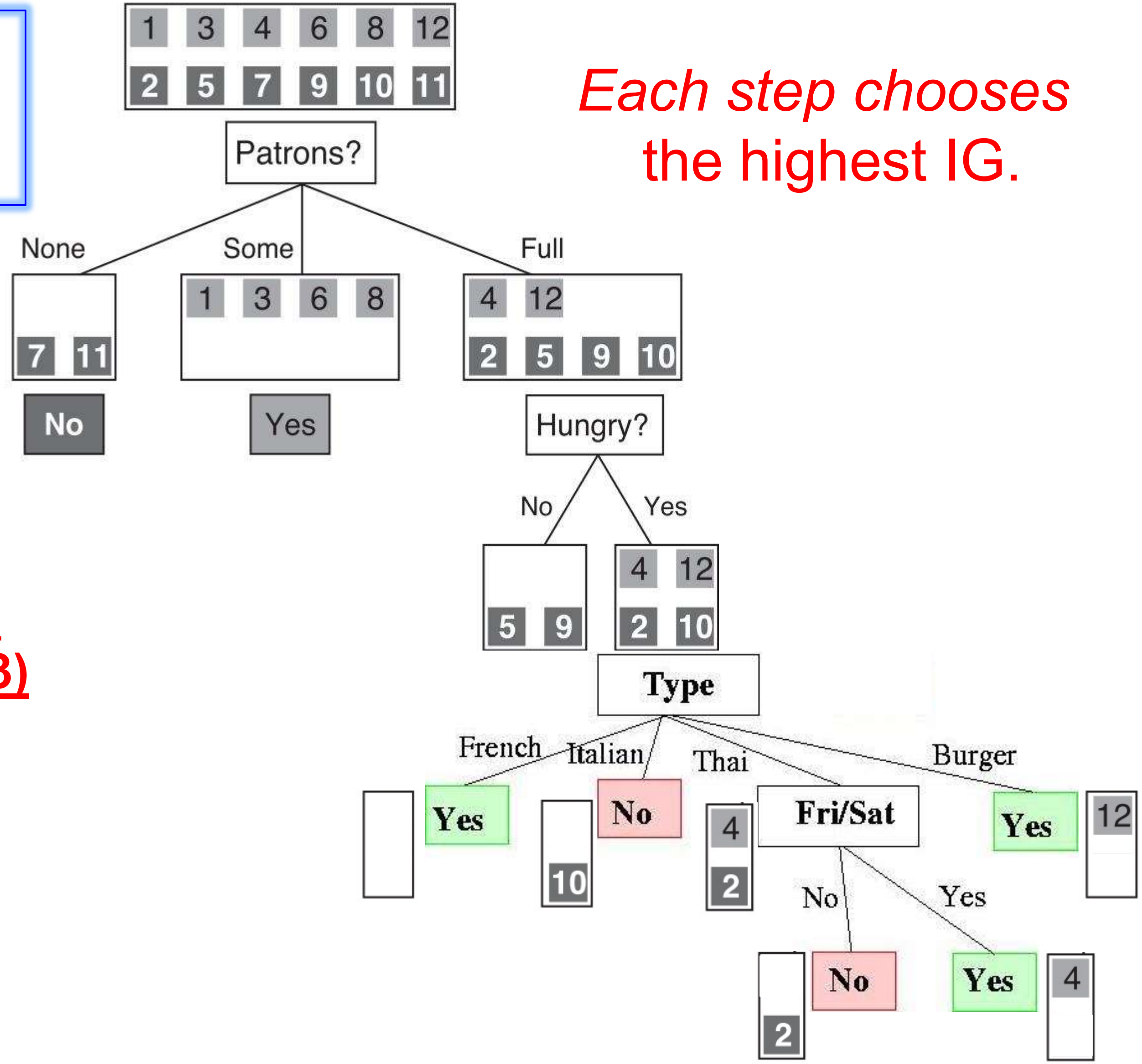
$$IG(Patrons) = I(\frac{6}{12},\frac{6}{12}) - [\frac{2}{12}I(\frac{0}{2},\frac{2}{2}) + \frac{4}{12}I(\frac{4}{4},\frac{0}{4}) + \frac{6}{12}I(\frac{2}{6},\frac{4}{6})] = 0.541 \text{ bits}$$

Patrons has the highest IG and is chosen as the root.

Decision tree learned from the 12 examples

The scheme is designed to **minimize the depth of the final tree**.

Each step chooses the highest IG.



實例：1975年Ross Quinlan 提出ID3 演算法 (Iterative Dichotomiser 3)

- Substantially simpler than "true" tree---a more complex hypothesis isn't justified by small amount of data.

另例：Decision Tree for Play Tennis

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

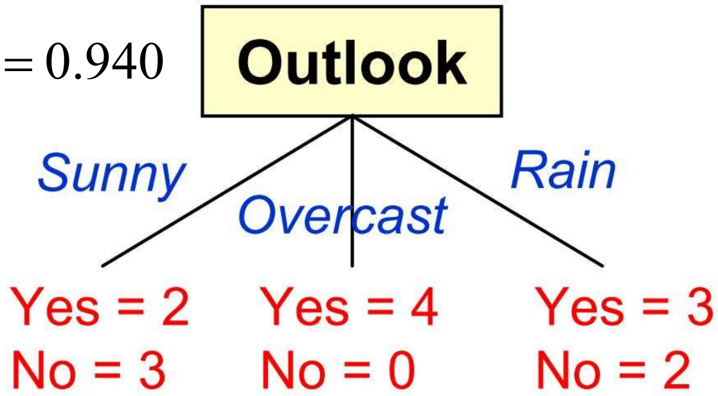
$$Entropy = I\left(\frac{9}{14}, \frac{5}{14}\right) = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) = 0.940$$

9+ 5-

Information Gain

$Entropy =$

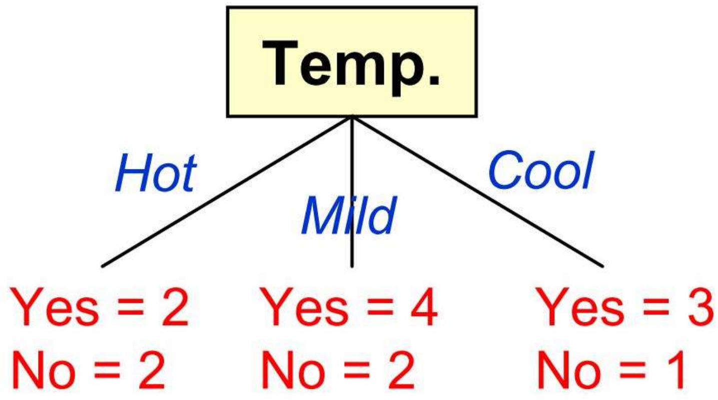
$I(\frac{9}{14}, \frac{5}{14}) = 0.940$



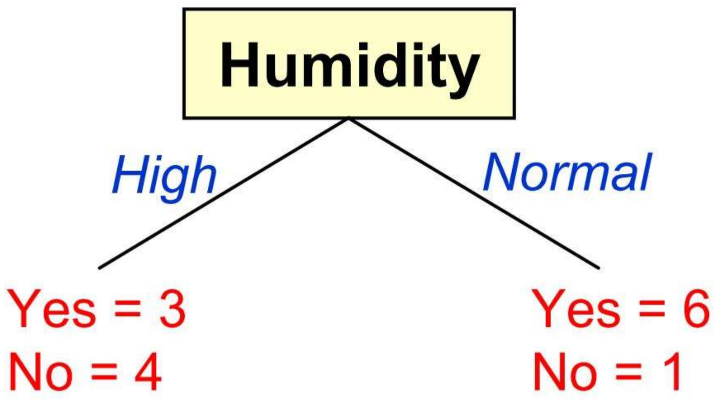
Gain(S, Outlook) = 0.246

the highest

=0.940-(5/14)*0.971 -(4/14)*0.0–(5/14)*0.0971

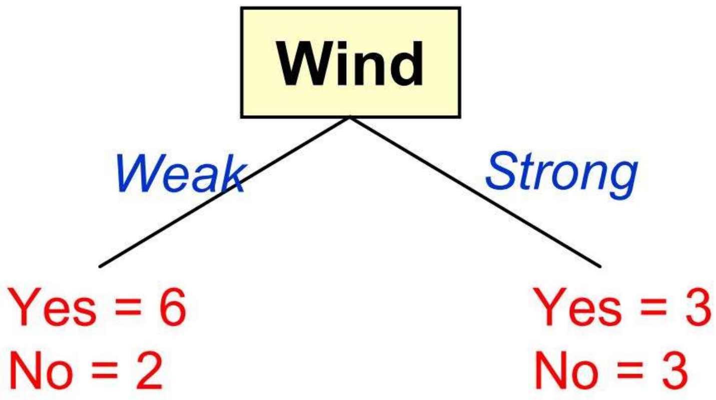


Gain(S, Temperature) = 0.029



Gain(S, Humidity) = 0.151

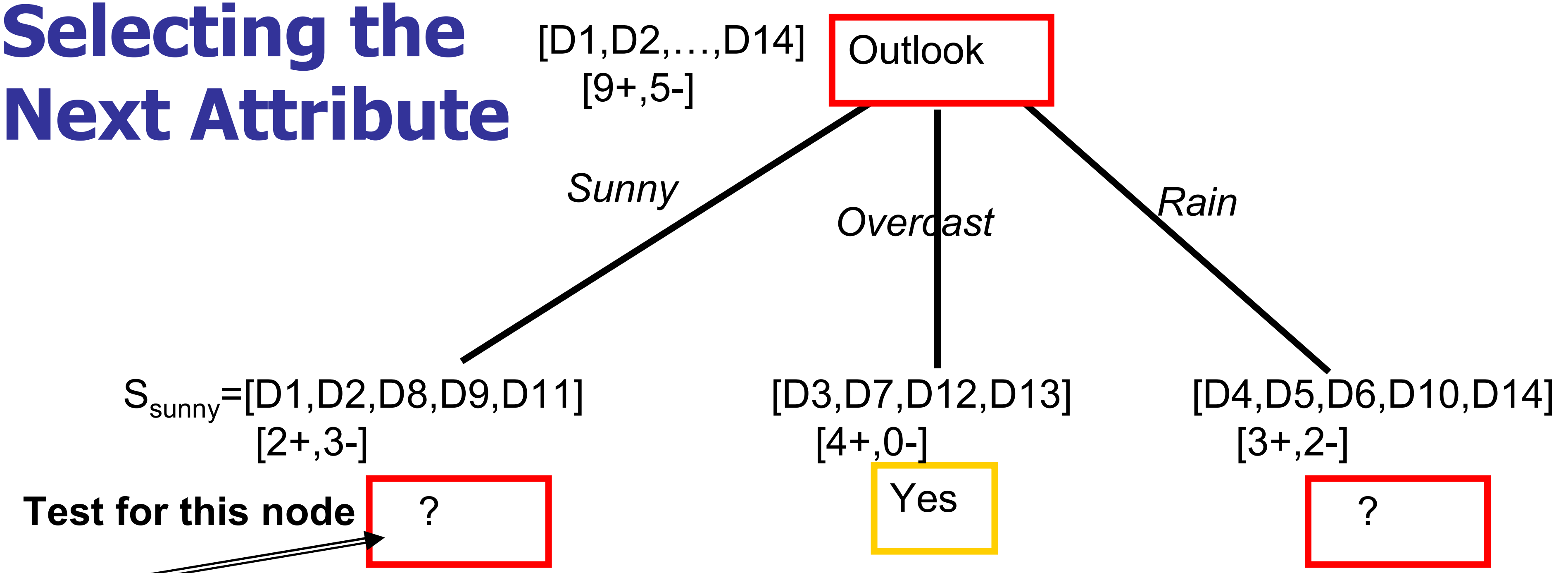
=0.940-(7/14)*0.985–(7/14)*0.592



Gain(S, Wind) = 0.048

=0.940-(8/14)*0.811–(6/14)*1.0

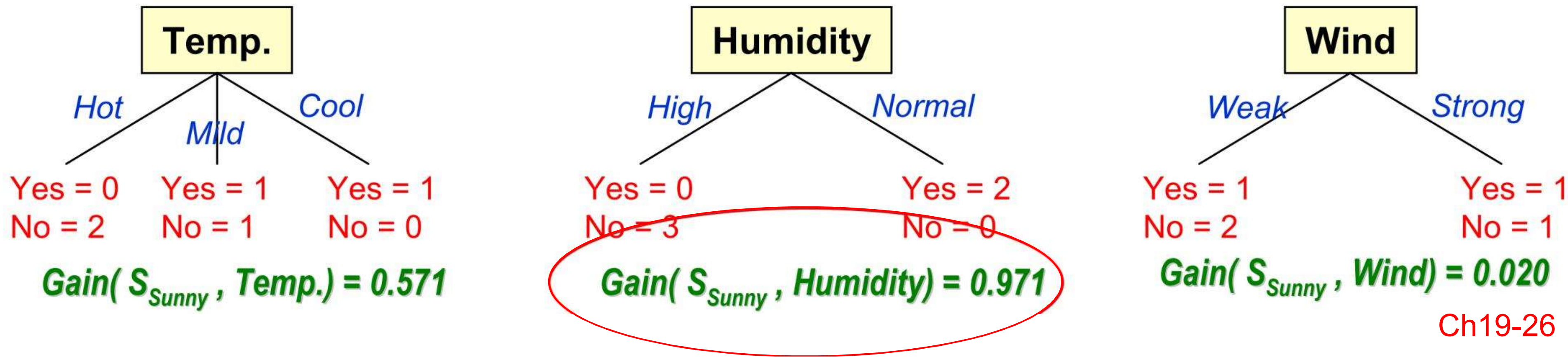
Selecting the Next Attribute



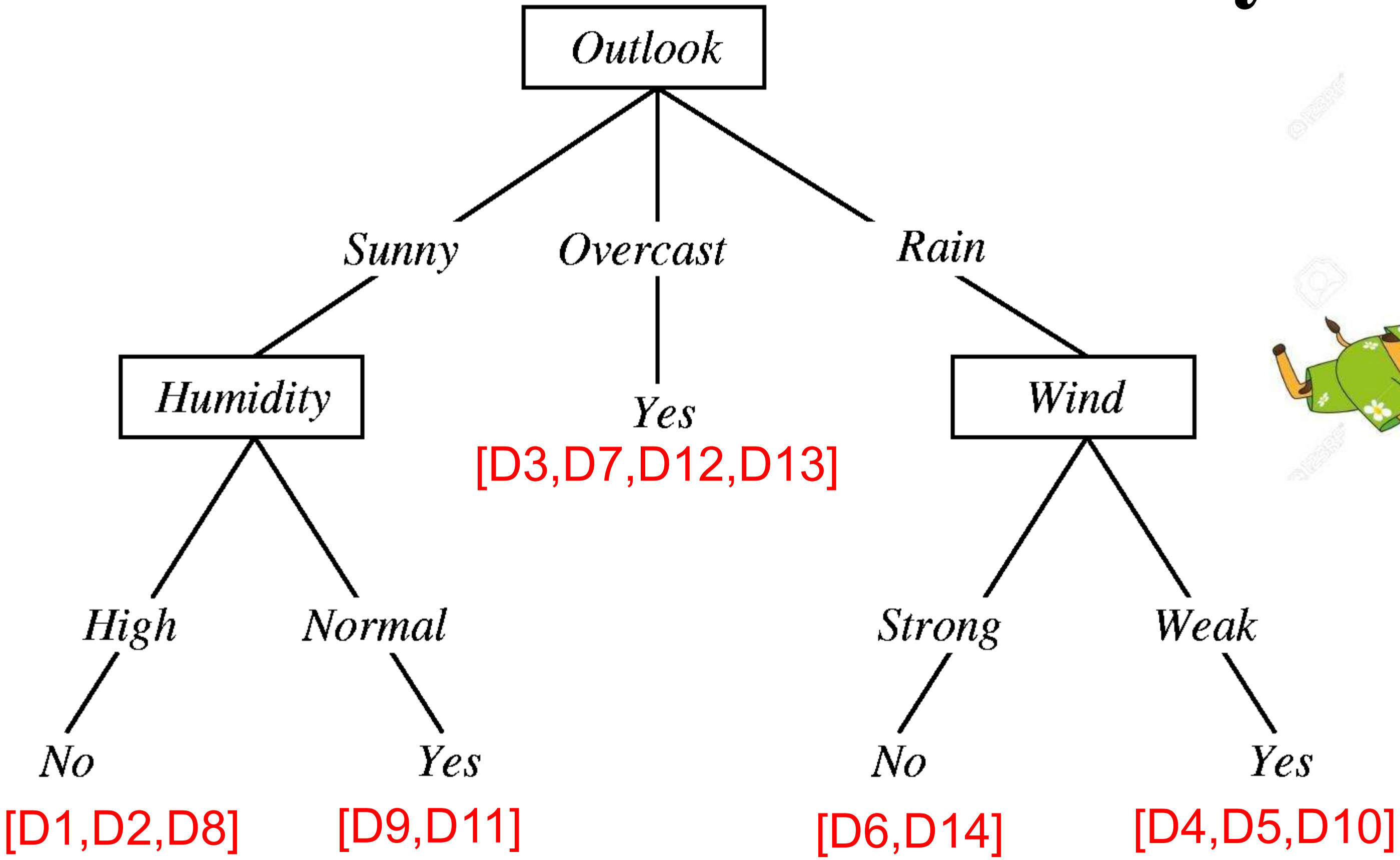
$\text{Gain}(S_{sunny}, \text{Temp.}) = 0.971 - (2/5)0.0 - 2/5(1.0) - (1/5)0.0 = 0.571$

$\text{Gain}(S_{sunny}, \text{Humidity}) = 0.971 - (3/5)0.0 - 2/5(0.0) = 0.971$ the highest

$\text{Gain}(S_{sunny}, \text{Wind}) = 0.971 - (2/5)1.0 - 3/5(0.918) = 0.020$



Final Decision Tree for Play Tennis 9+ 5-



Decision tree learning的總結：

1. 通常可快速對大量資料做出可行且效果良好的決策樹，且結果易於理解。
2. overfitting(過度適配)資料：指對範例的過度訓練，導致訓練結果不是針對一般特性，反而是訓練資料的局部特性。對測試樣本的結果將變得不精確。
3. 連續值屬性：透過定義新的離散值屬性來實現，即先把連續值屬性的值域分割爲離散的區間集合，或設定門檻值以進行二分法。
4. 屬性選擇的其他度量標準：尚有其他的度量標準，也都各有利弊。