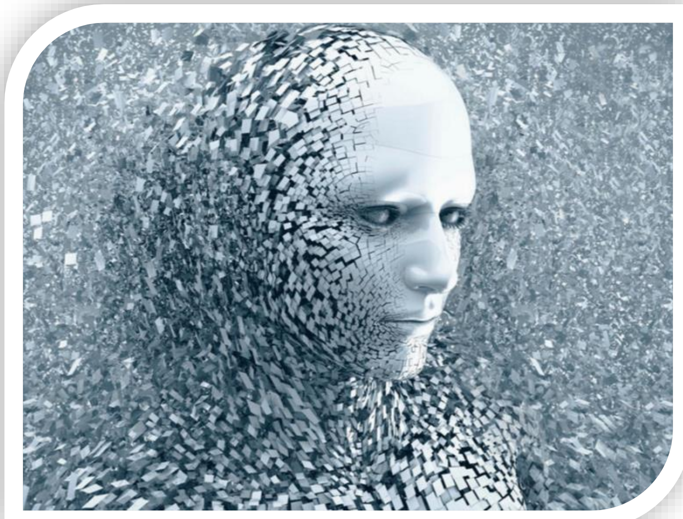


Artificial Intelligence

Session 7:
Decision Trees

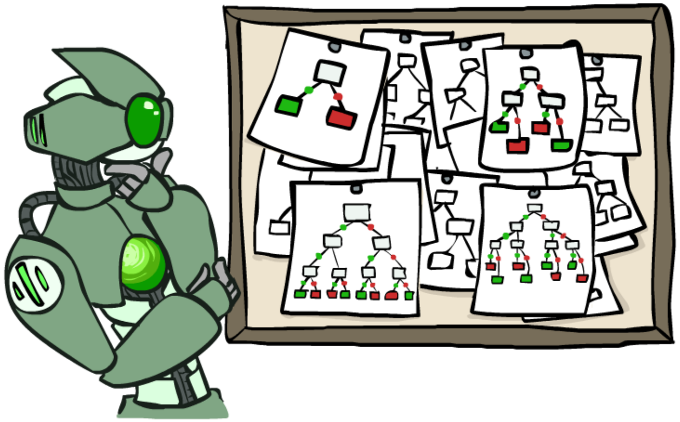
School of Computing and Engineering
University of West London, UK

Dr Massoud Zolgharni

Decision Trees

Decision Trees

- A hierarchical data structure that represents data by implementing a divide and conquer strategy
- Given a collection of examples, learn a decision tree that represents it
- Use this representation to classify new examples



A decision tree of a pair $(x; y)$ represents a function that takes the input attribute x (Boolean, discrete, continuous) and outputs a simple Boolean y

Decision Trees

E.g., situations where customers will/won't wait for a table.

Attributes:

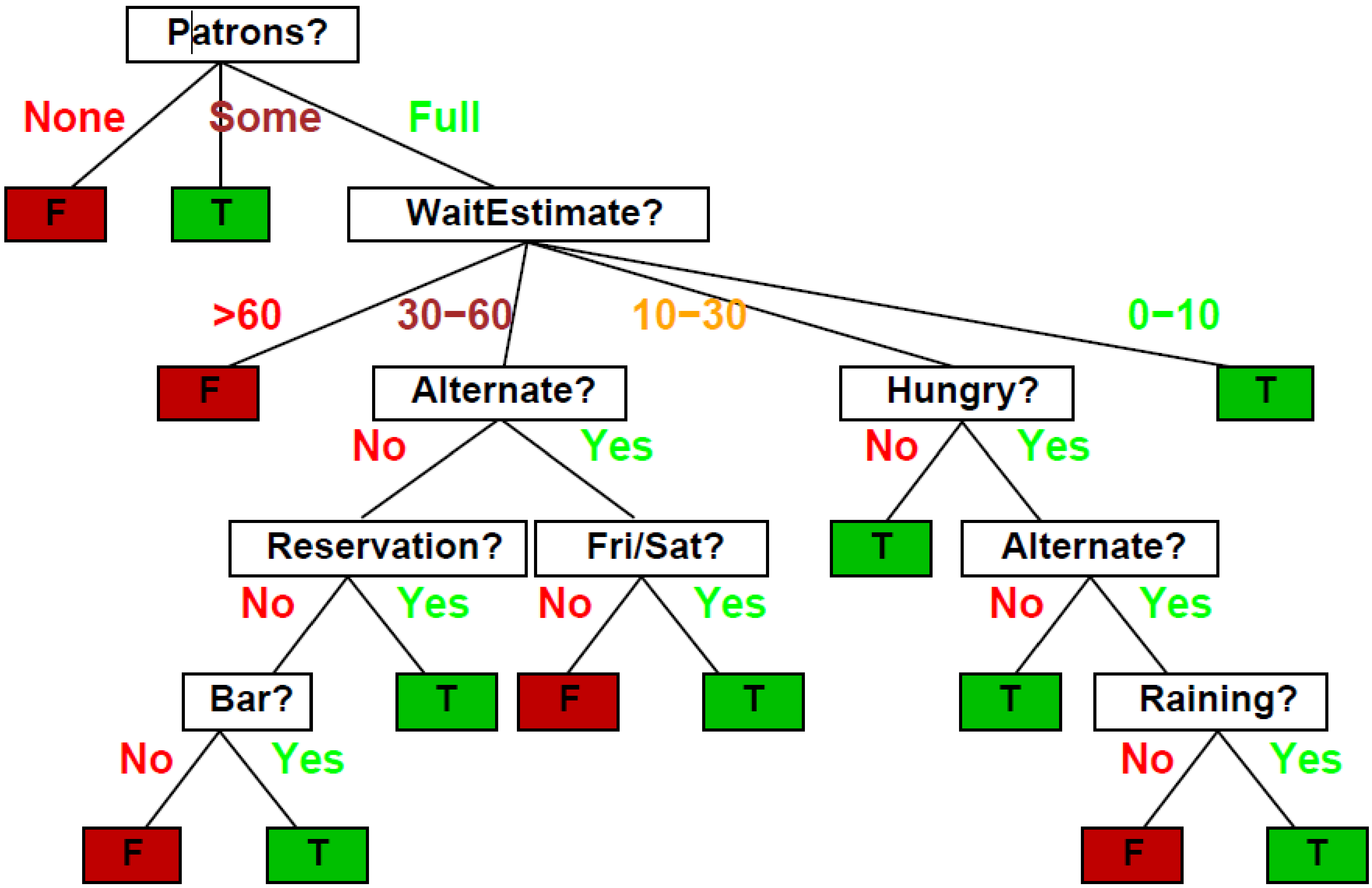
- Alternate: whether there is a suitable alternative restaurant nearby.
- Bar : whether the restaurant has a comfortable bar area to wait in.
- Fri/Sat: true on Fridays and Saturdays.
- Hungry: whether we are hungry.
- Patrons: how many people are in the restaurant (values are None, Some, and Full).
- Price: the restaurant's price range ($, $$, $$$).
- Raining: whether it is raining outside.
- Reservation: whether we made a reservation.
- Type: the kind of restaurant (French, Italian, Thai, or burger).
- WaitEstimate: the wait estimated by the host (0–10 minutes, 10–30, 30–60, or >60).

Training Set: (Classification of examples positive (T) or negative (F))

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

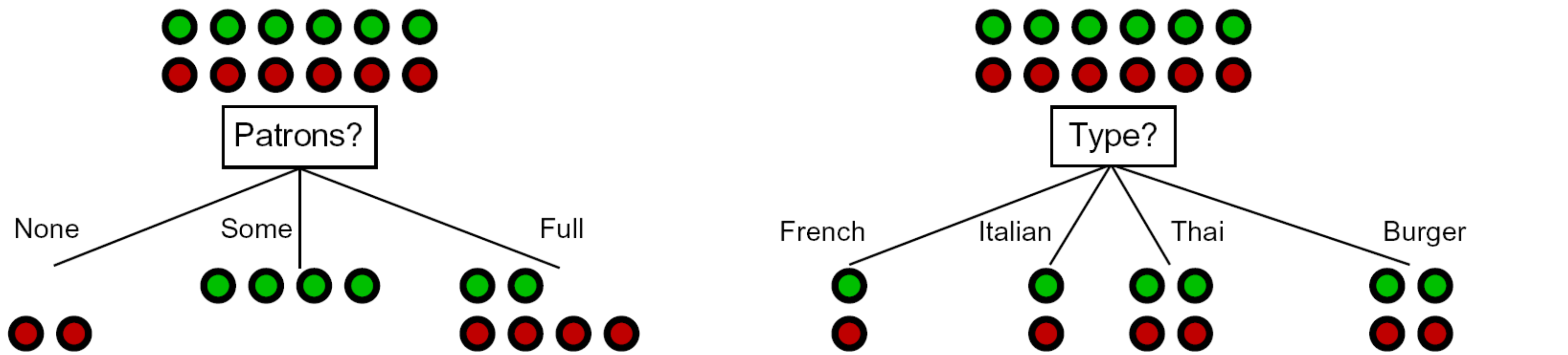
Decision Trees

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



Choosing an Attribute

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



- Patrons is a better choice—gives information about the classification
- need a measure of how “good” a split is, even if results aren’t perfectly separated out

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

Greedy divide-and-conquer

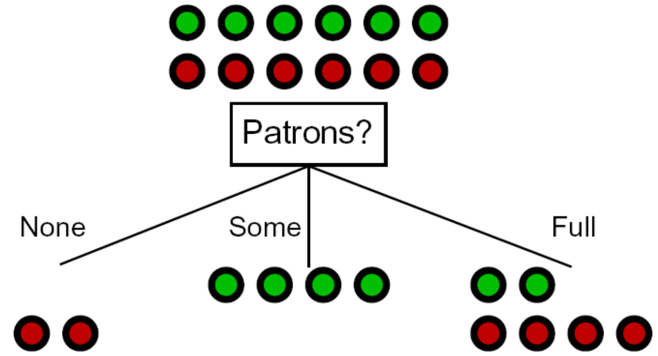
- Aim: find a small tree consistent with the training examples
- Idea: choose "most significant" attribute as root of (sub)tree and divide the problem up into smaller sub-problems that can be solved recursively

function DTL(*examples, attributes, default*) **returns** a decision tree

  if *examples* is empty **then return** *default*
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** Plurality_Value(*examples*)
  else
    best $\leftarrow$ Choose-Attribute(*attributes, examples*)
    tree $\leftarrow$ a new decision tree with root test *best*
    for each value v_i of *best* **do**
      $examples_i \leftarrow$ {elements of *examples* with $best = v_i$}
      subtree $\leftarrow$ DTL($examples_i$, *attributes* − *best*, Mode(*examples*))
      add a branch to *tree* with label v_i and subtree *subtree*
    return *tree*

Next Step: Recurse

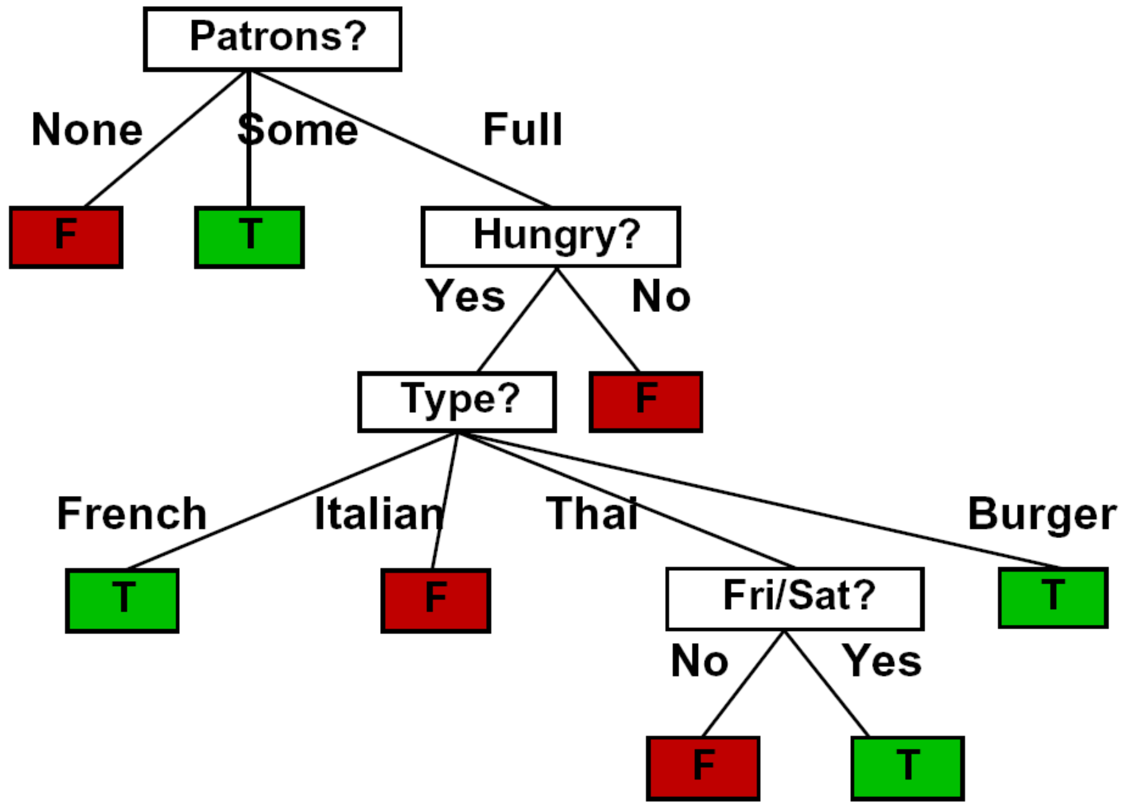
- Now we need to keep growing the tree!
- Two branches are done (why?)
- What to do under “full”?
  - See what examples are there…



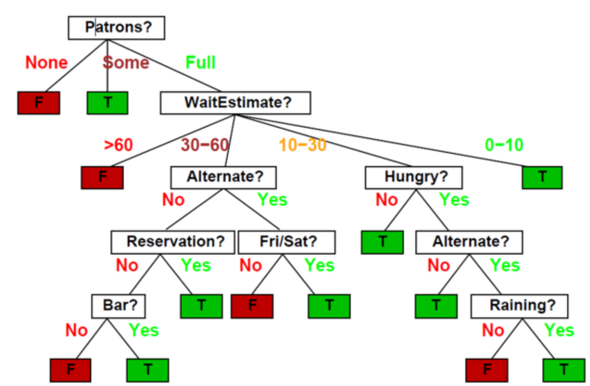
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*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

Learned Tree

- Decision tree learned from these 12 examples:



- Substantially simpler than “true” tree



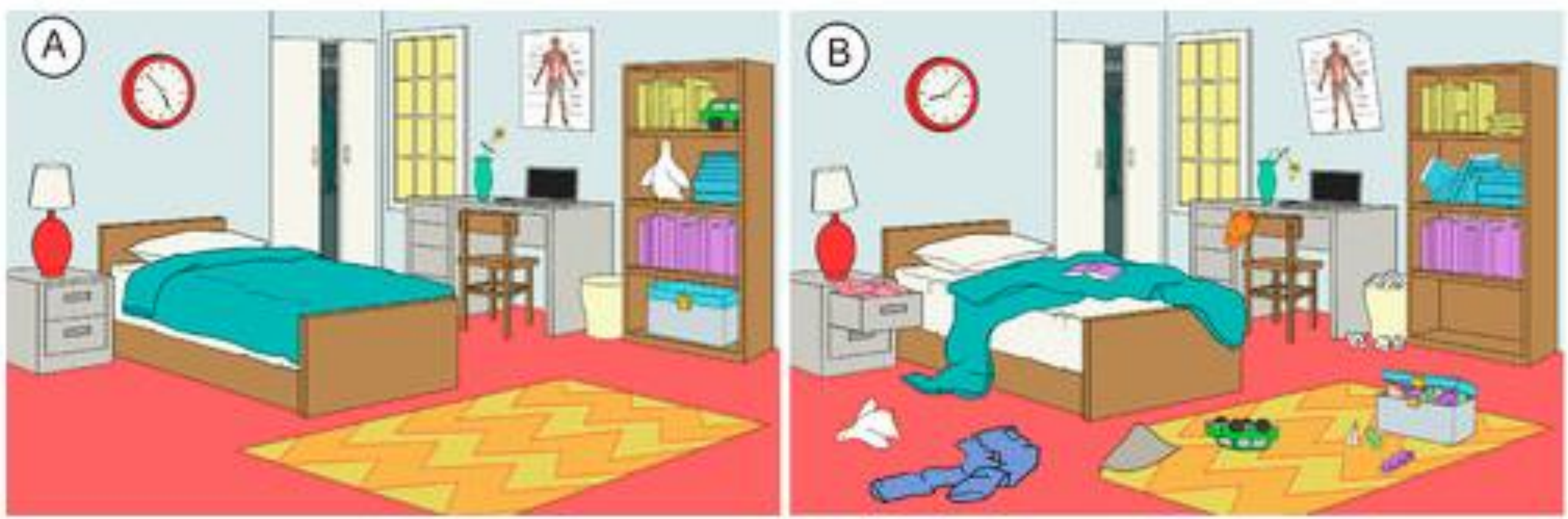
Choosing an Attribute

Which attribute is the best classifier?

- **Information Gain** - A statistical property that measures how well a given attribute separates the training examples according to their target classification.
- This measure is used to select among the candidate attributes at each step while growing the tree.

Entropy

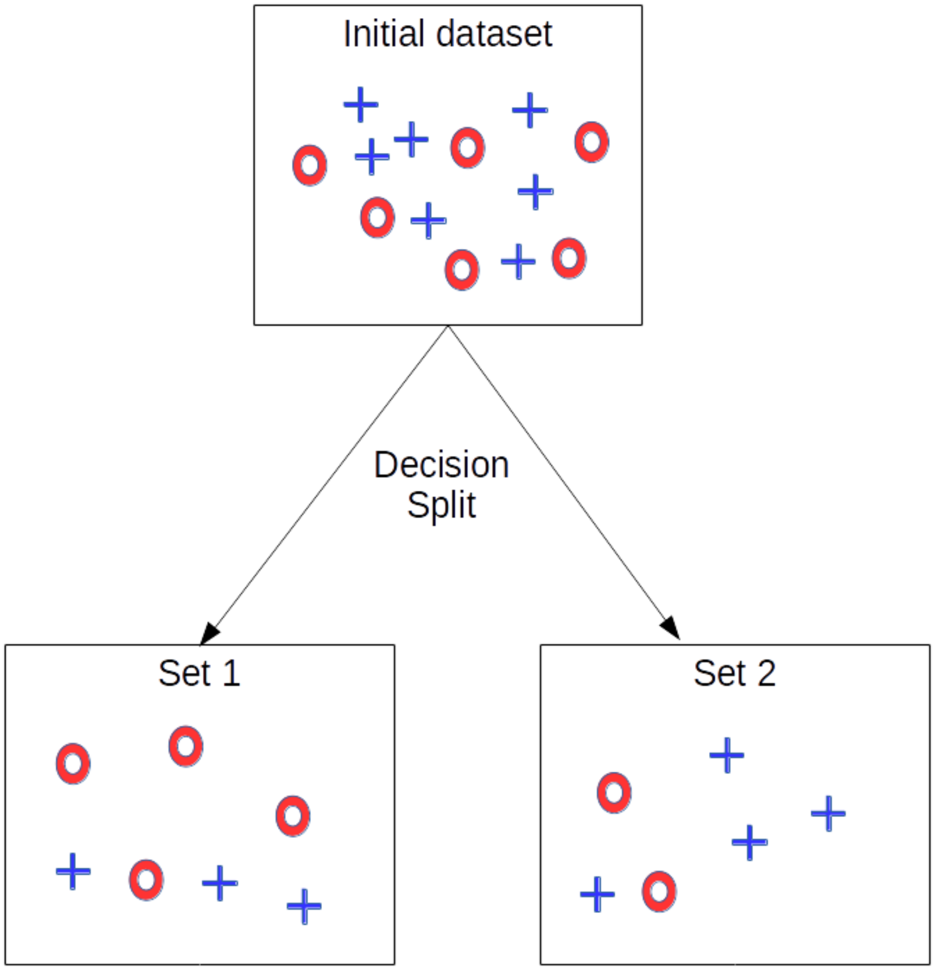
Analogy: measure of how messy the room is….



Low Entropy

High Entropy

Entropy



Entropy

S is a sample of training examples

$p_{\oplus}$ is the proportion of positive examples in S

$p_{\ominus}$ is the proportion of negative examples in S

Then the entropy measures the impurity of S:

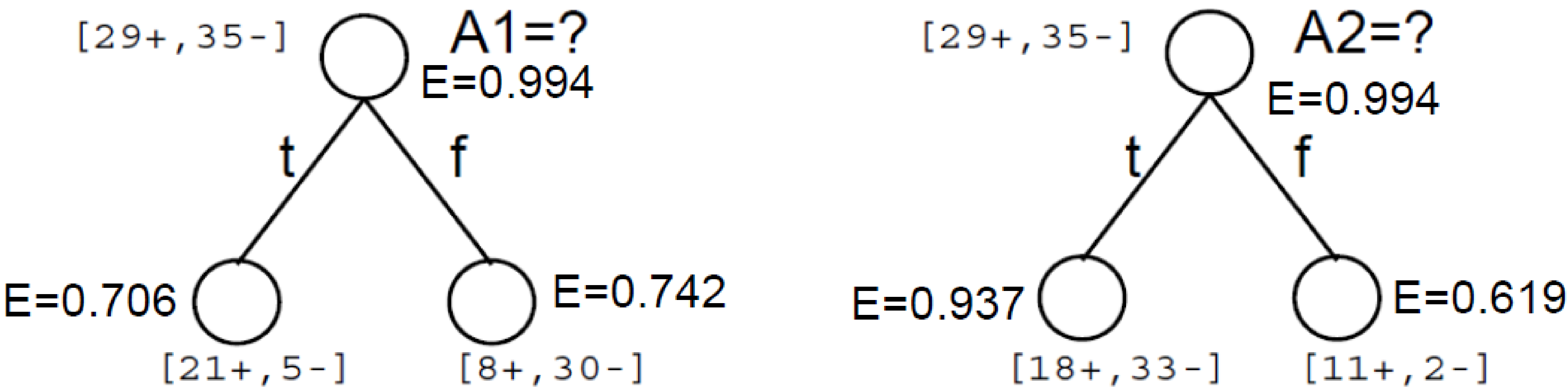
$$Entropy(S) = -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

If the target attribute can take C different values:

$$Entropy(S) = \sum_{i=1}^{C} -p_i \log_2 p_i$$

Entropy

$$Entropy(S) = -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$



Entropy([29+, 35-]) = - (29/64) $\log_2$(29/64) - (35/64) $\log_2$(35/64) = 0.994

Information Gain

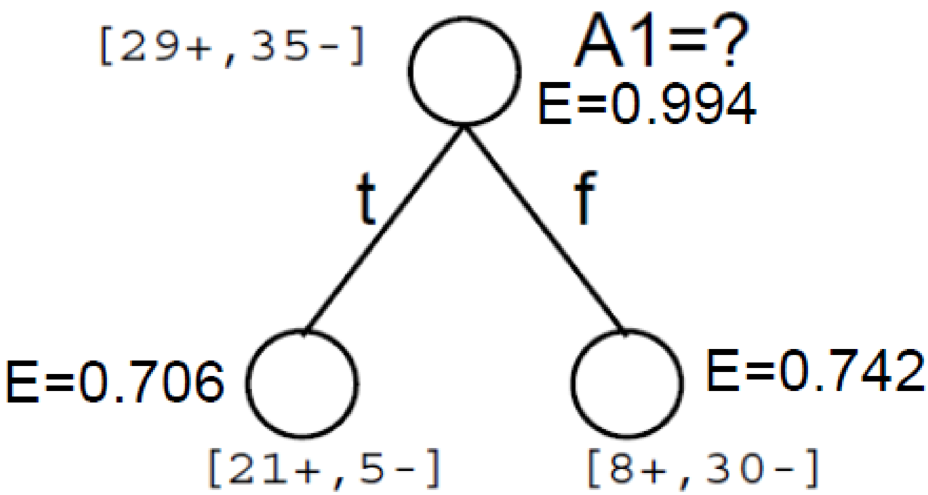
Gain(*S*,*A*) = expected reduction in entropy due to sorting on A

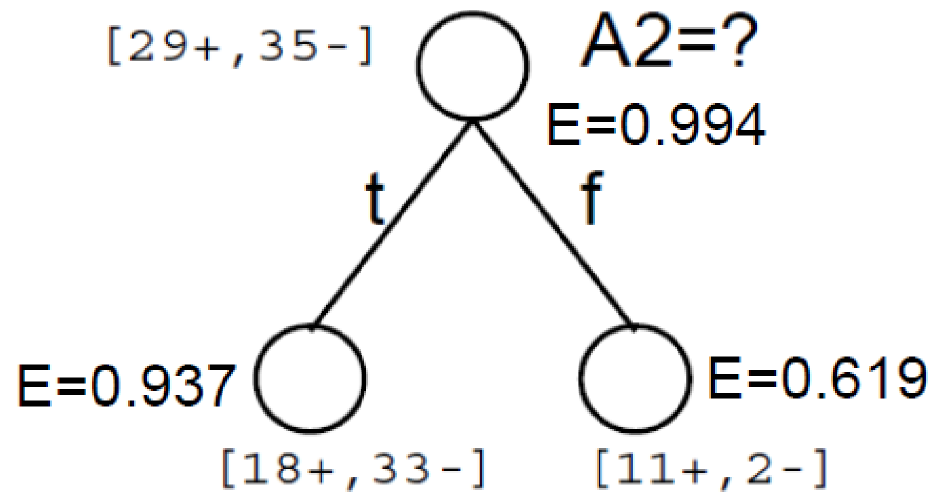
$$Gain(S,A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S_v is sum of entropies of each subset, weighted by fraction of examples $|S_v/S|$ that belong to S_v

Information Gain

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$





Gain(S,A1) = 0.994 –

(26/64)×0.706 – (38/64)×0.742

= 0.266

Information gained by partitioning

along attribute A1 is 0.266

Gain(S,A2) = 0.994 –

(51/64)×0.937 – (13/64)×0.619

= 0.121

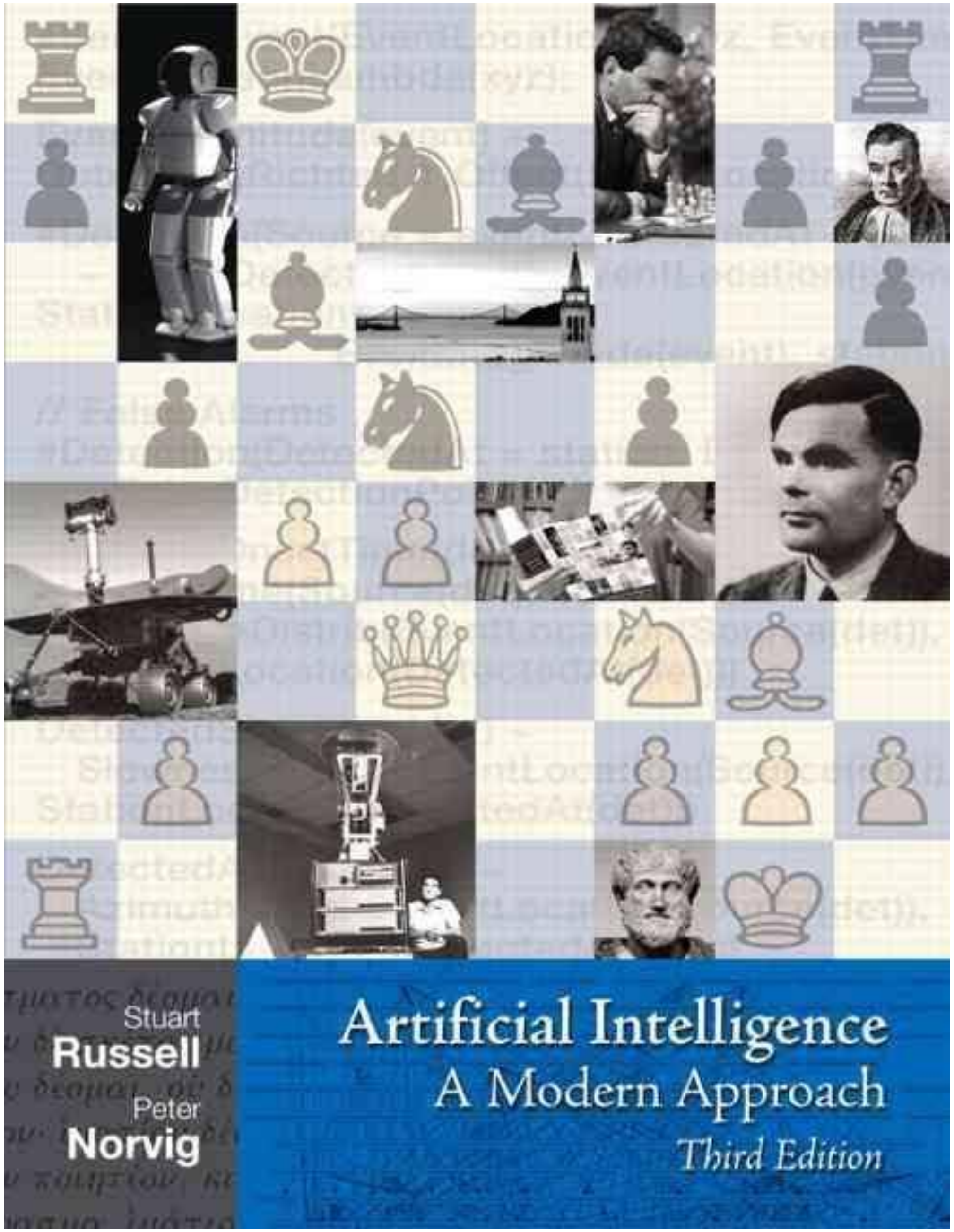
Information gained by partitioning

along attribute A2 is 0.121

Recommended reading

Stuart Russell, Peter Norvig: ***Artificial Intelligence A Modern Approach***

Chapter 18



ANY Questions?