

430.457

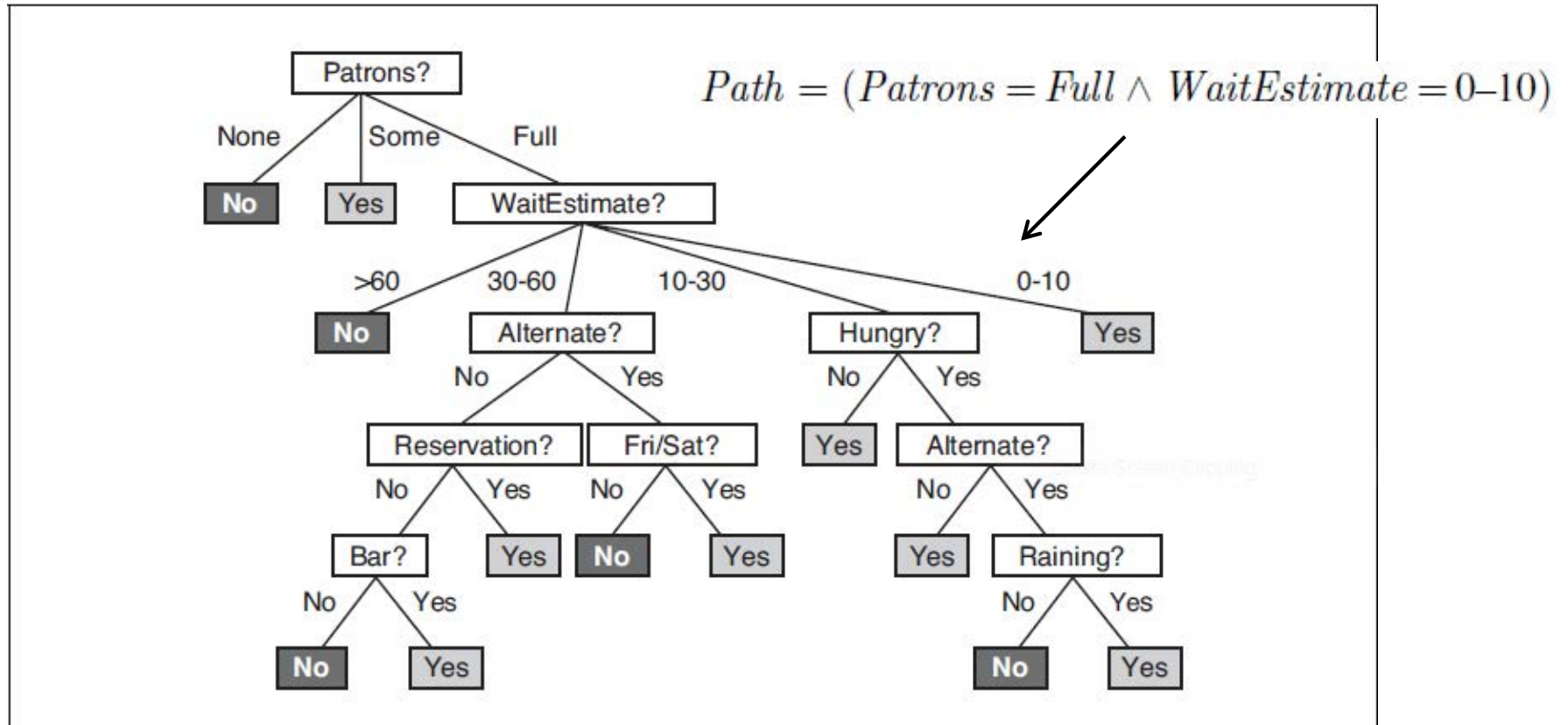
Introduction to Intelligent Systems

Prof. Songhwai Oh
ECE, SNU

DECISION TREES

Decision Trees

Decision tree to decide whether to wait for a table at a restaurant or not

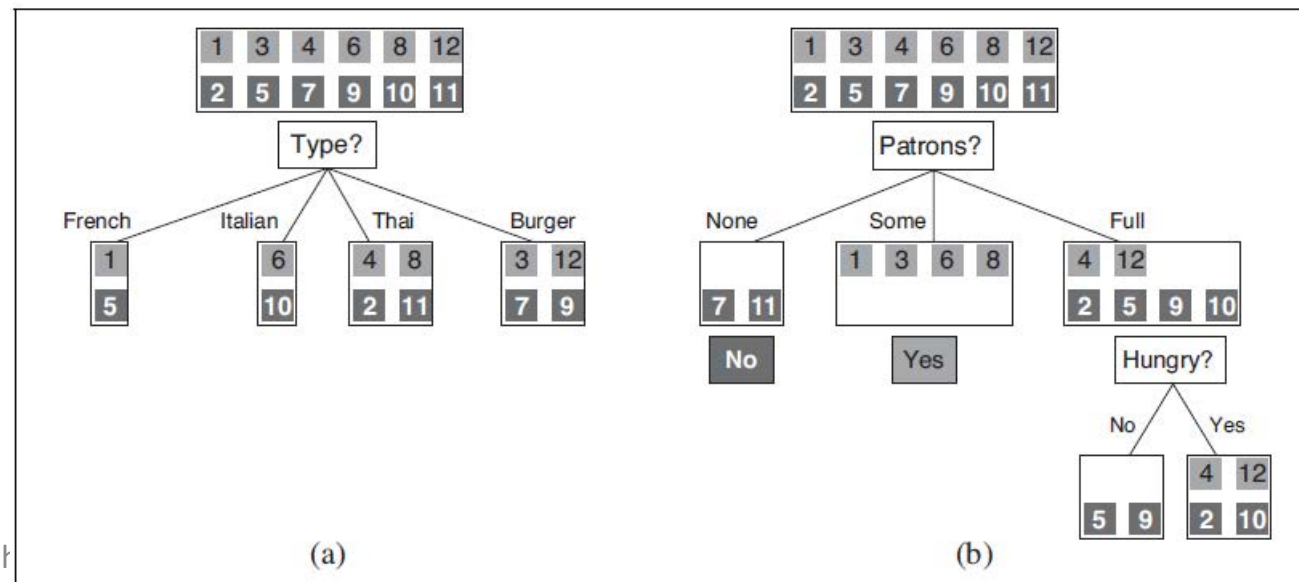


$$Goal \Leftrightarrow (Path_1 \vee Path_2 \vee \dots)$$

- Any function in propositional logic can be expressed as a decision tree.
- The number of n variable Boolean functions is $2^{(2^n)}$.

Learning a Decision Tree from Data

Example	Input Attributes										Goal
	<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>WillWait</i>
x_1	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	$y_1 = \text{Yes}$
x_2	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	$y_2 = \text{No}$
x_3	No	Yes	No	No	Some	\$	No	No	Burger	0-10	$y_3 = \text{Yes}$
x_4	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	$y_4 = \text{Yes}$
x_5	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	$y_5 = \text{No}$
x_6	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	$y_6 = \text{Yes}$
x_7	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	$y_7 = \text{No}$
x_8	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	$y_8 = \text{Yes}$
x_9	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	$y_9 = \text{No}$
x_{10}	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	$y_{10} = \text{No}$
x_{11}	No	No	No	No	None	\$	No	No	Thai	0-10	$y_{11} = \text{No}$
x_{12}	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	$y_{12} = \text{Yes}$

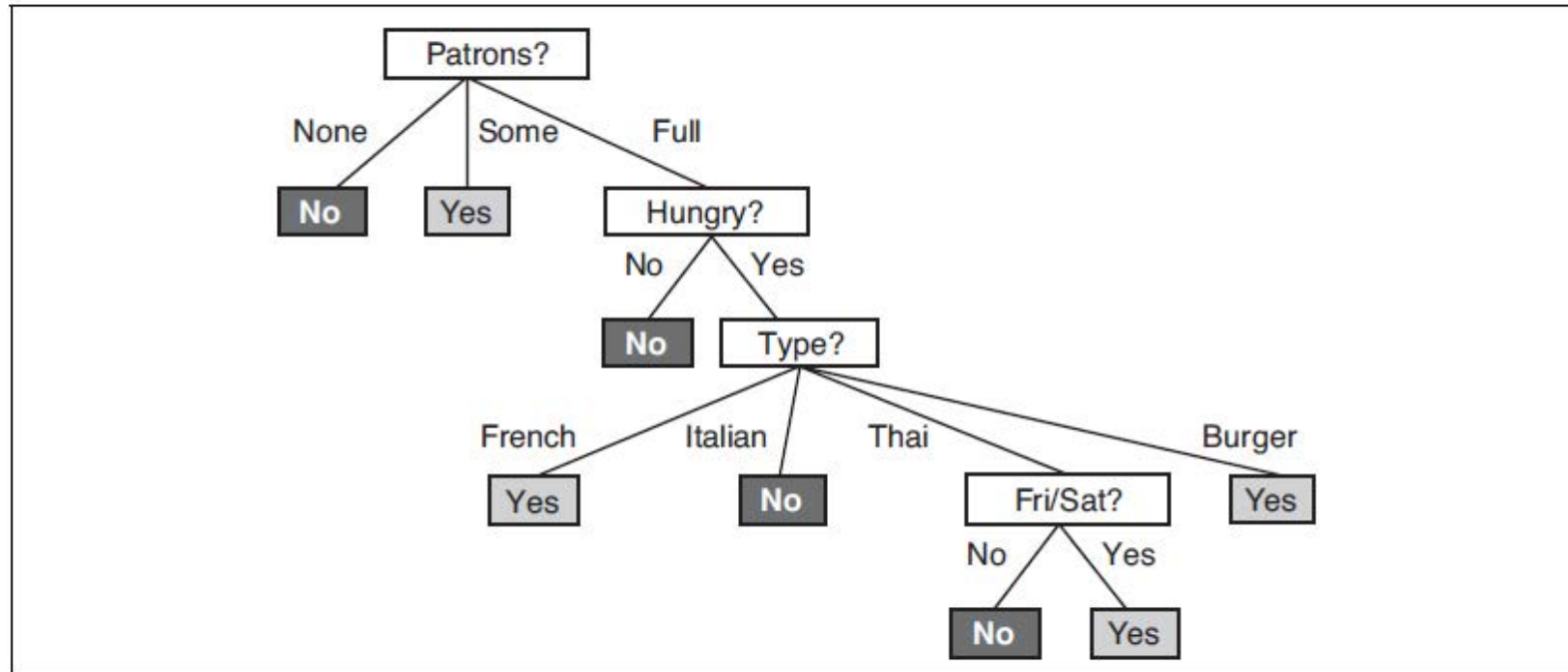


Decision Tree Learning Algorithm

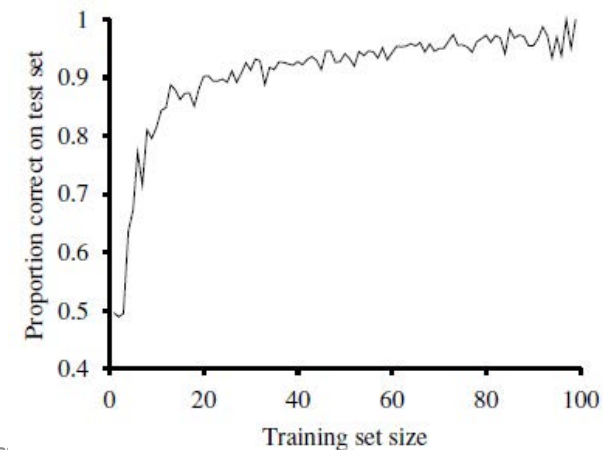
```
function DECISION-TREE-LEARNING(examples, attributes, parent_examples) returns  
a tree  
  
  if examples is empty then return PLURALITY-VALUE(parent_examples)  
  else if all examples have the same classification then return the classification  
  else if attributes is empty then return PLURALITY-VALUE(examples)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

- The problem space is too big so we cannot perform exhaustive search.
- Find a good approximate solution with a small consistent tree.
- Decision-Tree-Learning is a greedy algorithm based on divide-and-conquer.

Decision tree learned by the algorithm using the training set



Learning curve



Information Gain

- Used to implement the function $Importance(a, examples)$.
- **Entropy:** a measure of the uncertainty of a random variable. For a random variable V with PMF $P(v_k)$, the entropy of V is

$$H(V) = - \sum_k P(v_k) \log_2 P(v_k).$$

E.g., the entropy of a fair coin flip is $H(Fair) = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$; the entropy of a biased coin with 99% heads is $H(Loaded) = -(0.99 \log_2 0.99 + 0.01 \log_2 0.01) \approx 0.08$ bits.

- $B(q)$, entropy of a binary random variable (true with probability q):

$$B(q) = -(q \log_2 q + (1 - q) \log_2 (1 - q)).$$

- With p positive examples and n negative examples, the entropy of the goal attribute in a decision network is

$$H(Goal) = B\left(\frac{p}{p+n}\right).$$

-
- Suppose attribute A has d distinct values, divide the training set E into subsets $E_1, \dots, E_d$.
 - Each subset E_k has p_k positive examples and n_k negative examples.
 - If we follow the k -th branch, we will need an additional $B(p_k/(p_k + n_k))$ bits of information to answer the question.
 - A randomly chosen example from the training set has the k -th value for the attribute with probability $(p_k + n_k)/(p + n)$.
 - Hence, the expected entropy remaining after testing attribute A is

$$Remainder(A) = \sum_{k=1}^d \frac{p_k + n_k}{p + n} B\left(\frac{p_k}{p_k + n_k}\right).$$

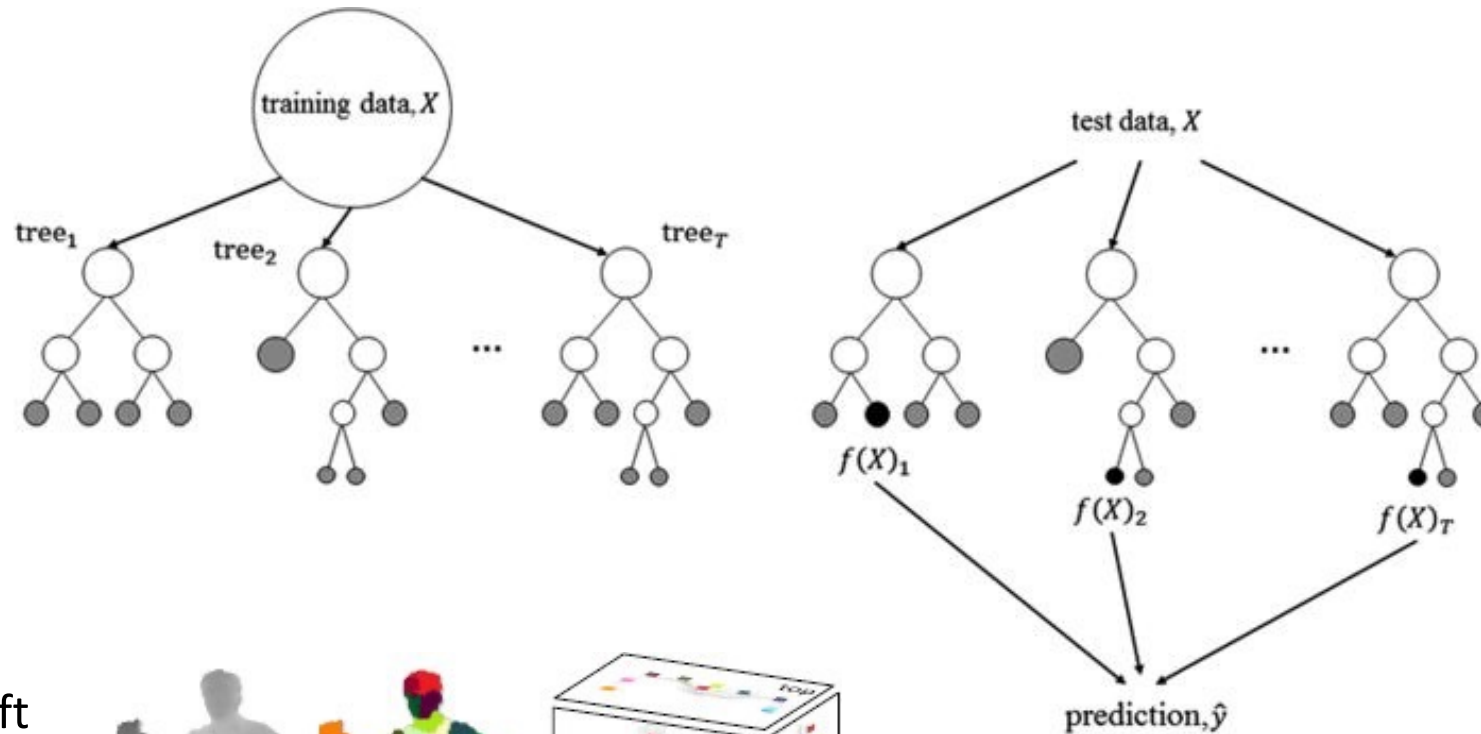
- **Information gain** from the attribute test on A is

$$Gain(A) = B\left(\frac{p}{p + n}\right) - Remainder(A).$$

- For our example:

$Gain(Patrons) = 1 - \left[\frac{2}{12}B\left(\frac{0}{2}\right) + \frac{4}{12}B\left(\frac{4}{4}\right) + \frac{6}{12}B\left(\frac{2}{6}\right)\right] \approx 0.541 \text{ bits},$
 $Gain(Type) = 1 - \left[\frac{2}{12}B\left(\frac{1}{2}\right) + \frac{2}{12}B\left(\frac{1}{2}\right) + \frac{4}{12}B\left(\frac{2}{4}\right) + \frac{4}{12}B\left(\frac{2}{4}\right)\right] = 0 \text{ bits},$

Random Forest



Microsoft
Kinect

