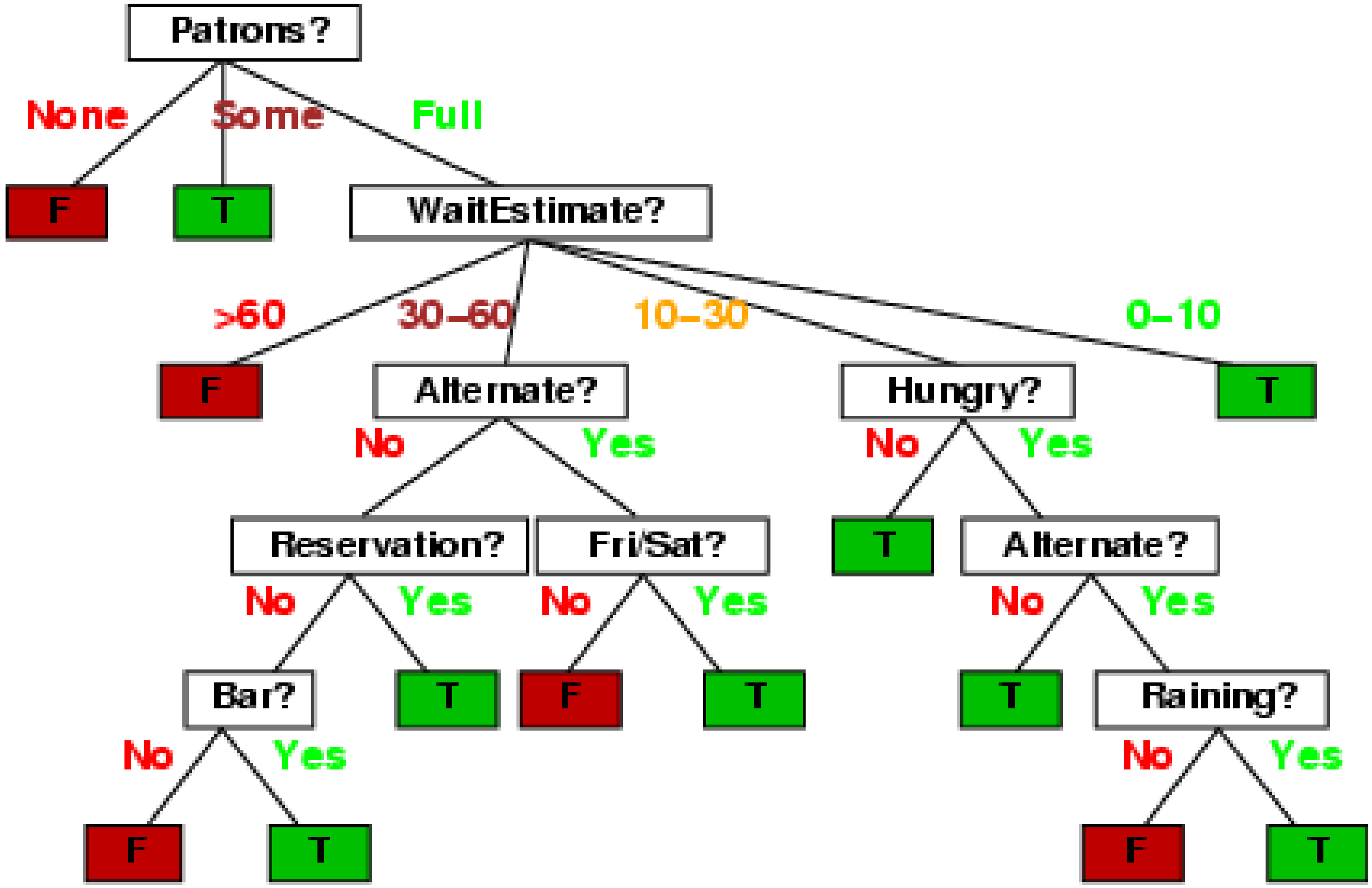
CSE 473

Lecture 25
(Chapter 18)

Learning Decision Trees

A "personal" decision tree for deciding whether to wait at a restaurant

- A decision tree for *Wait?* based on personal "rules of thumb":



Input Data for Learning

- Past examples when I did/did not 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

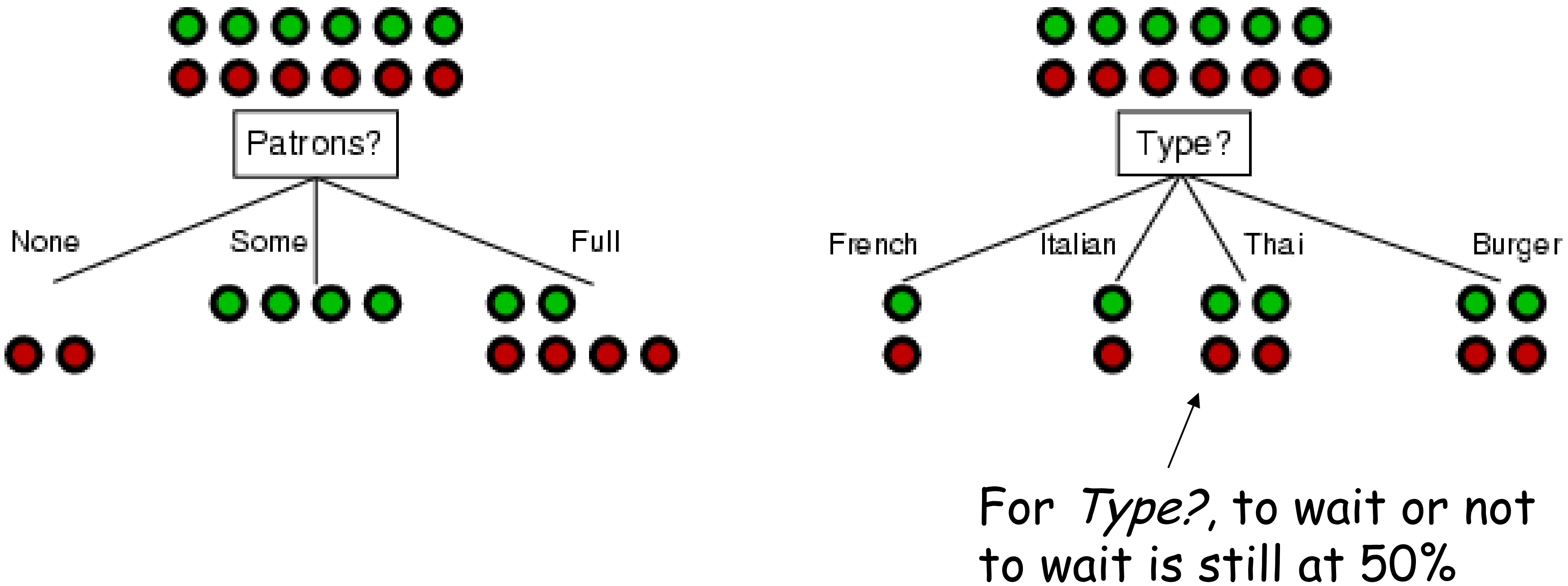
Decision Tree Learning

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

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** MODE(*examples*)
    else
        $\boxed{best \leftarrow \text{CHOOSE-ATTRIBUTE}(attributes, examples)}$
        tree ← a new decision tree with root test *best*
        for each value v_i of *best* **do**
            $examples_i \leftarrow$ {elements of *examples* with $best = v_i$}
            $\boxed{subtree \leftarrow \text{DTL}(examples_i, attributes - best, \text{MODE}(examples))}$
            add a branch to *tree* with label v_i and subtree *subtree*
        return *tree*

Choosing an attribute to split on



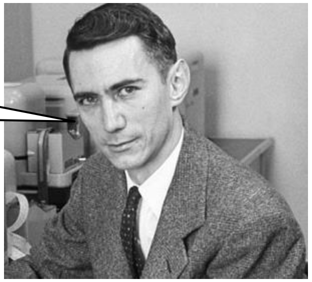
- Idea: a good attribute should reduce uncertainty
  - E.g., splits the examples into subsets that are (ideally) "all positive" (T) or "all negative“ (F)
- *Patrons?* is a better choice

Reduce uncertainty?

How do you quantify uncertainty?

http://a.espncdn.com/media/ten/2006/0306/photo/g_mcenroe_195.jpg

Use information theory!



- Entropy measures the amount of uncertainty in a probability distribution

- Entropy (or information content in bits) of an answer to a question with *n* possible answers $v_1, \ldots, v_n$:

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

Using information theory

- Suppose we have p examples with Wait = True (positive) and n examples with Wait = False (negative).

- Our best estimate of the probabilities of Wait = true or false is given by:

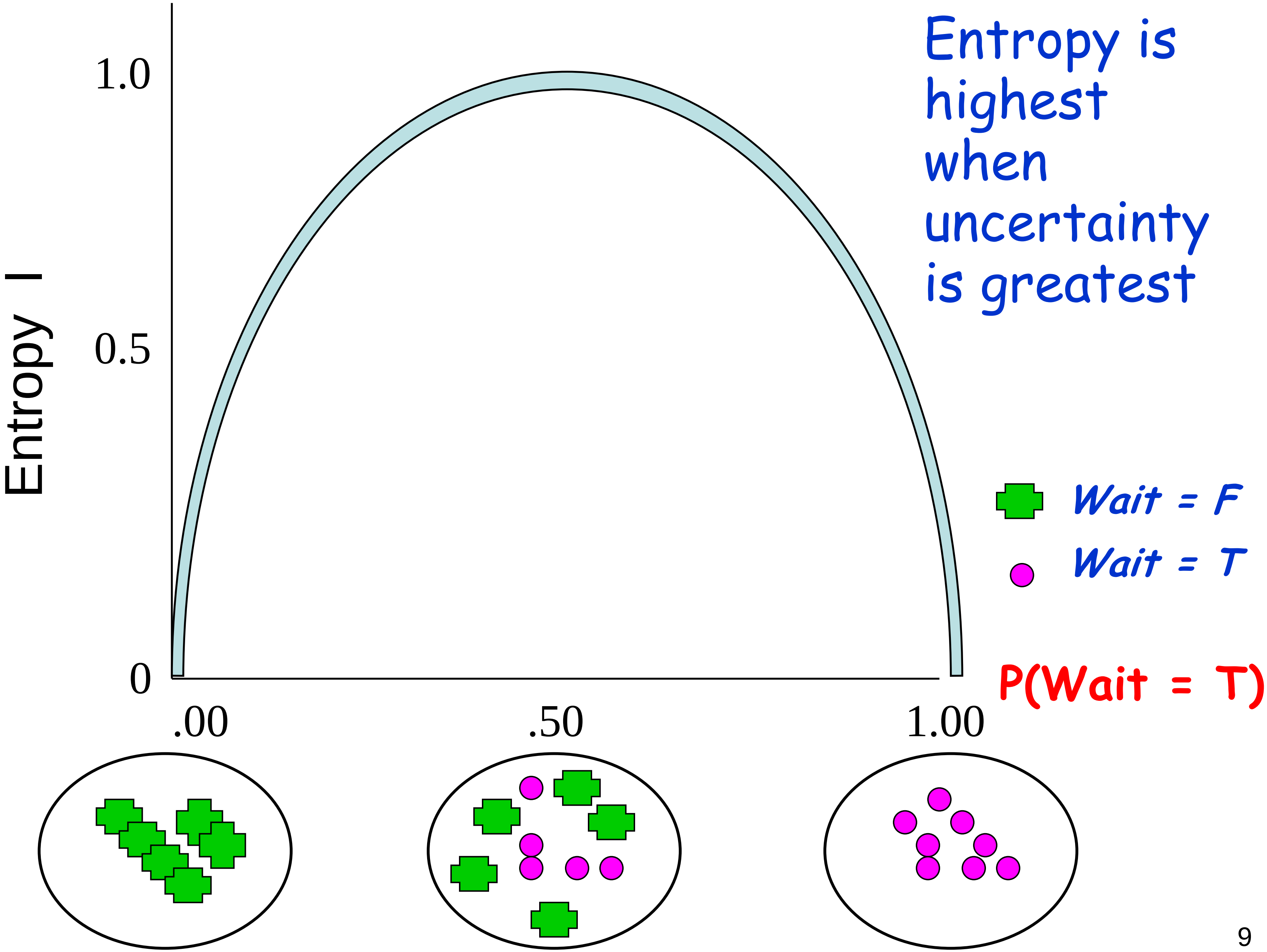
$$P(true) \approx p / p + n$$

$$p(false) \approx n / p + n$$

- Hence the entropy (in bits) is given by:

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Entropy is highest when uncertainty is greatest



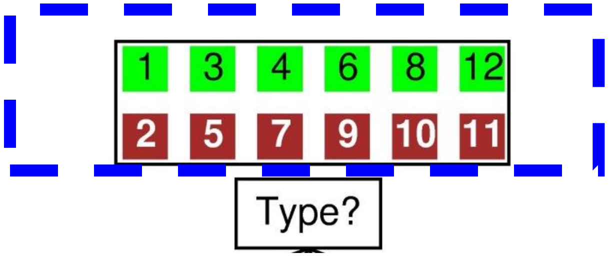
Choosing an attribute to split on

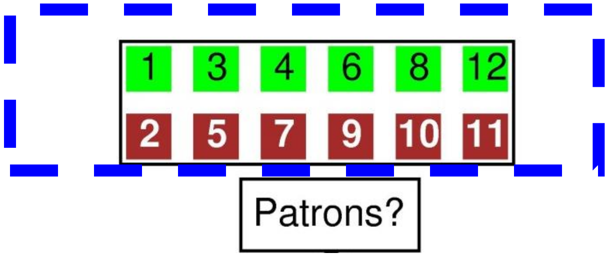
- Idea: a good attribute should *reduce uncertainty* and result in “gain in information”
- How much information do we gain if we disclose the value of some attribute?

- Answer:

  uncertainty before – uncertainty after

Back at the Restaurant





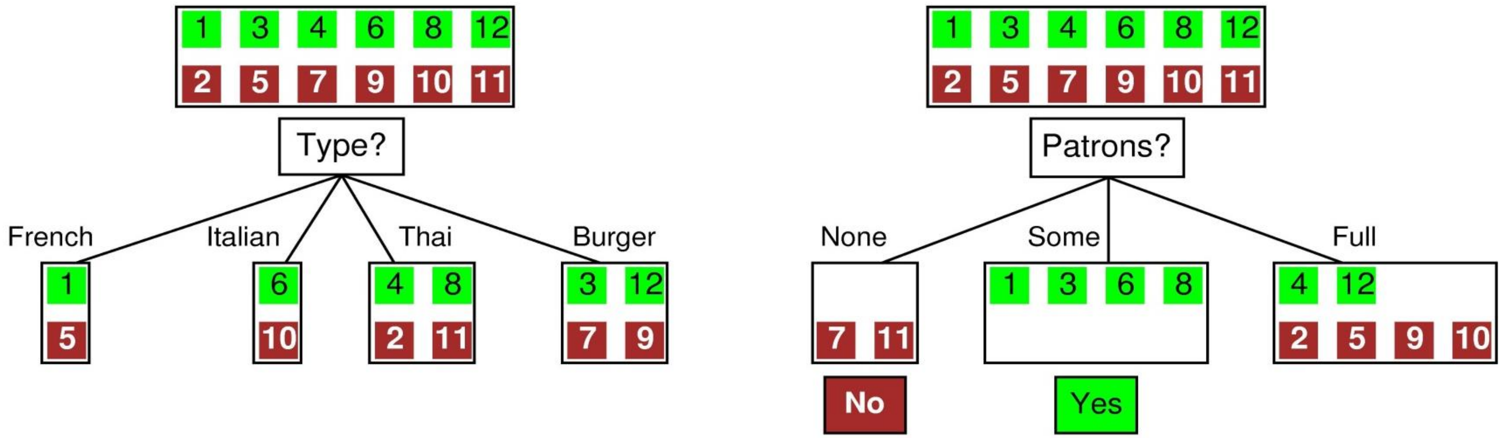
Before choosing an attribute: 6 True and 6 False

Entropy = - 6/12 log(6/12) – 6/12 log(6/12)

= - log(1/2) = log(2) = 1 bit

There is "1 bit of information to be discovered"

Choosing an Attribute



If we choose Type: Along "French": entropy = 1 bit.

Information gain = 1-1 = 0. (same for other branches)

If we choose Patrons:

In branches "None" and "Some", entropy = 0

For "Full", entropy = -2/6 log(2/6)-4/6 log(4/6) = 0.92

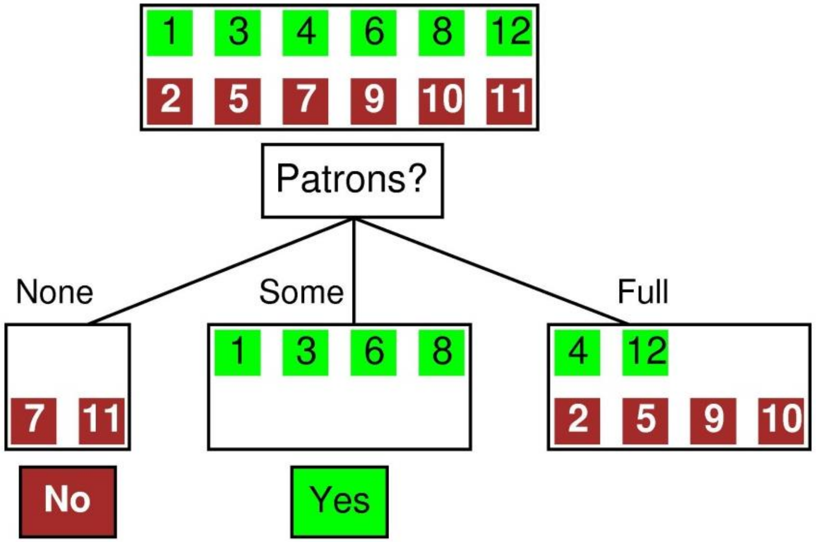
So info gain = (1-0) or (1-0.92) bits > 0 in all cases

Choosing Patrons gains more information!

Combining entropy across branches

- Computing average entropy
- Weight entropies according to probability of branches

  2/12 times we entered "None" so weight for "None" = 1/6
  "Some" has weight: 4/12 = 1/3
  "Full" has weight: 6/12 = ½



2/12 4/12 6/12

$$AvgEntropy = \sum_{i=1}^{N} \frac{p_i + n_i}{p + n} Entropy\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

Sum over all *N* branches

weight for each branch

entropy for each branch

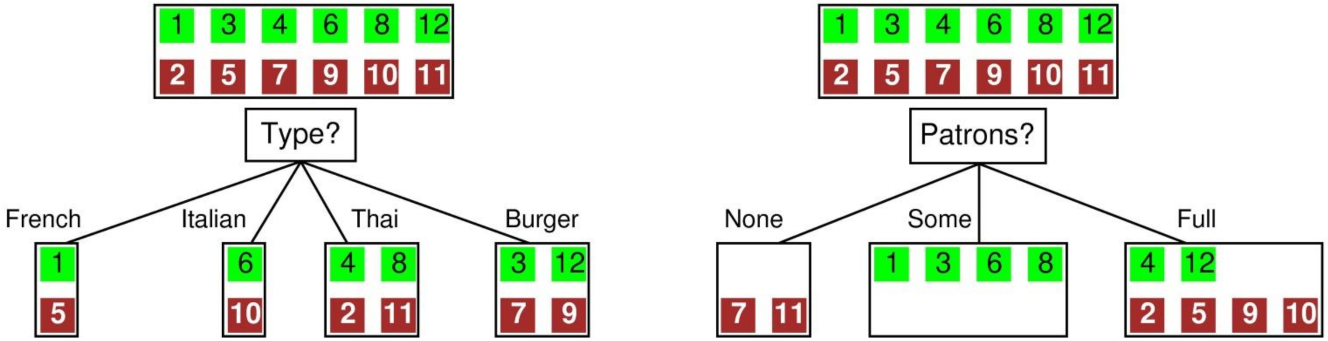
Information gain

- Information Gain (IG) (= reduction in entropy) when choosing attribute A:

  IG(A) = Entropy before choosing
  – AvgEntropy after choosing A

- When constructing each level of decision tree, choose attribute with largest IG

Information gain in our example



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

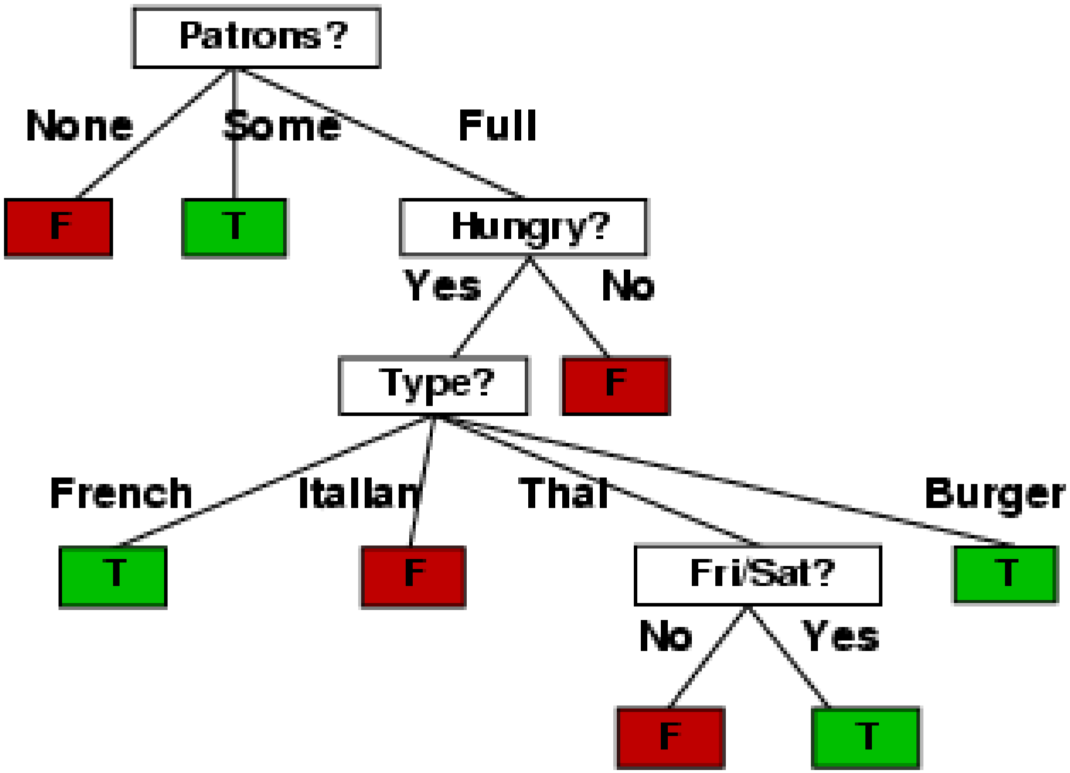
$$IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .541 \text{ bits}$$

Patrons has highest IG of all attributes

⇒ DTL algorithm chooses *Patrons* as the root

Learned Decision Tree for "Wait?"

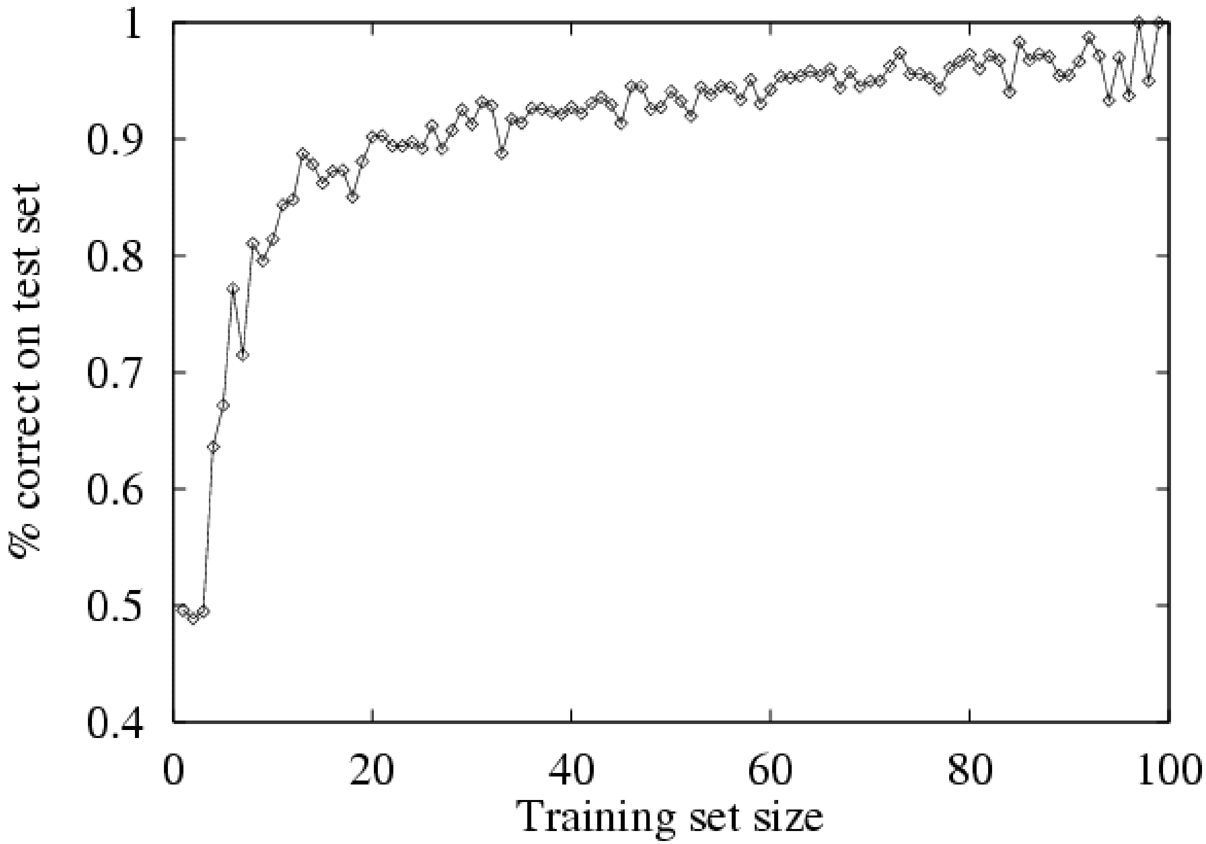
- Decision tree learned from the 12 examples:



- Substantially simpler than "rules-of-thumb" tree
  - more complex hypothesis not justified by small amount of data

Performance Evaluation

- How do we know that the learned tree $h \approx true\ f$?
- Answer: Try *h* on a new test set of examples
- Learning curve = % correct on test set as a function of training set size



Generalization

- How do we know the classifier function we have learned is good?
  - Look at generalization error on test data
    - Method 1: Split data into separate training and test sets (the "hold out" method)
      - What if the split you chose was bad?
    - Method 2: Cross-Validation

Cross-validation

- K-fold cross-validation:
  - Divide data into K subsets of equal size
  - Train learning algorithm K times, each time leaving out one of the subsets, and compute error on left-out subset
  - Report average error over all subsets
- Leave-1-out cross-validation:
  - Train on all but 1 data point, test on that data point; repeat for each point
  - Report average error over all points

Next Time

- Other classification methods
  - Linear Classification
  - Support Vector Machines
- To Do:
  - Project 4
  - Read Chapter 18