



# Classification: Decision Trees

These slides were assembled by Eric Eaton, with grateful acknowledgement of the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution. Please send comments and corrections to Eric.

# Function Approximation

## Problem Setting

- Set of possible instances  $\mathcal{X}$
- Set of possible labels  $\mathcal{Y}$
- Unknown target function  $f : \mathcal{X} \rightarrow \mathcal{Y}$
- Set of function hypotheses  $H = \{h \mid h : \mathcal{X} \rightarrow \mathcal{Y}\}$

**Input:** Training examples of unknown target function  $f$

$$\{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^n = \{\langle \mathbf{x}_1, y_1 \rangle, \dots, \langle \mathbf{x}_n, y_n \rangle\}$$

**Output:** Hypothesis  $h \in H$  that best approximates  $f$

# Sample Dataset (was Tennis Played?)

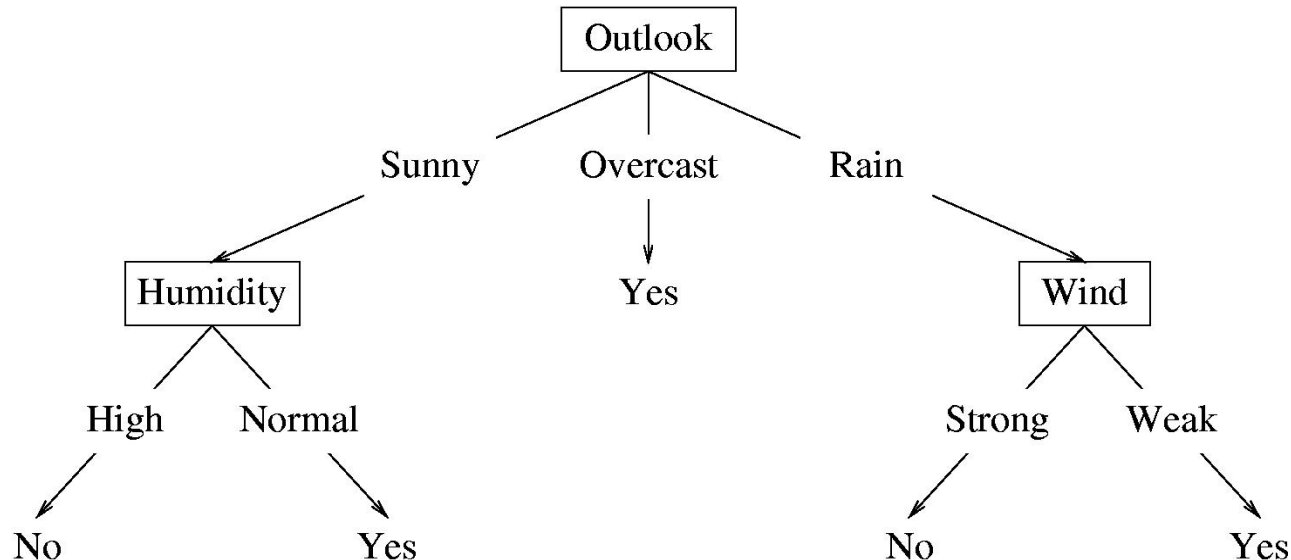
- Columns denote features  $X_i$
- Rows denote labeled instances  $\langle x_i, y_i \rangle$
- Class label denotes whether a tennis game was played

$\langle x_i, y_i \rangle$

Predictors				Response
Outlook	Temperature	Humidity	Wind	Class
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

# Decision Tree

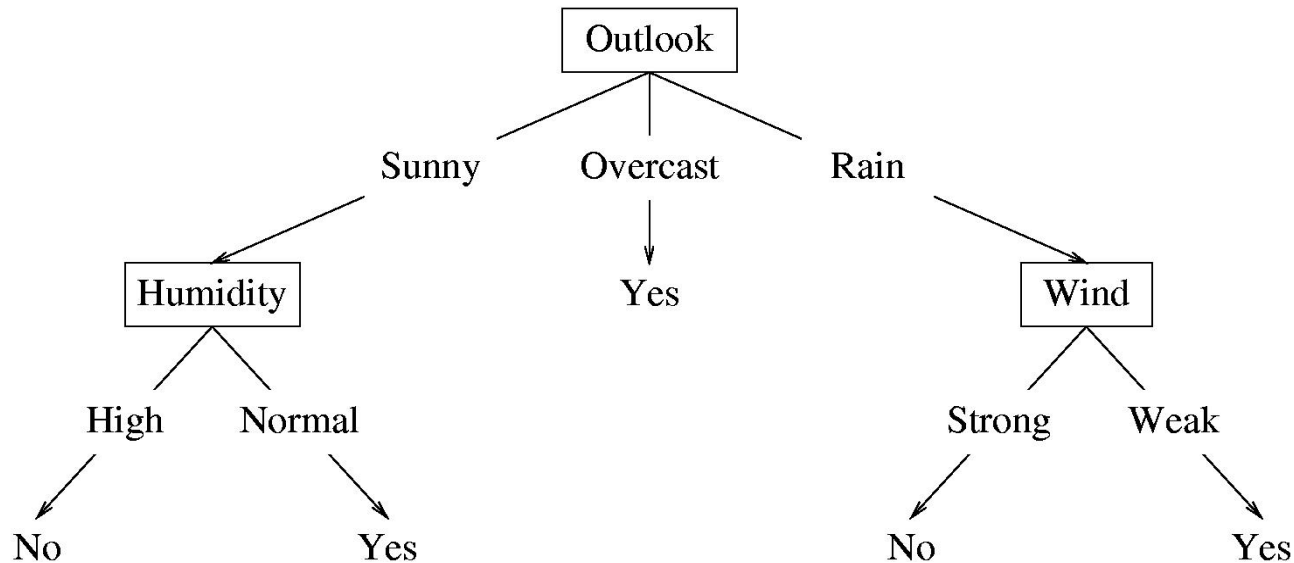
- A possible decision tree for the data:



- Each internal node: test one attribute  $X_i$
- Each branch from a node: selects one value for  $X_i$
- Each leaf node: predict  $Y$

# Decision Tree

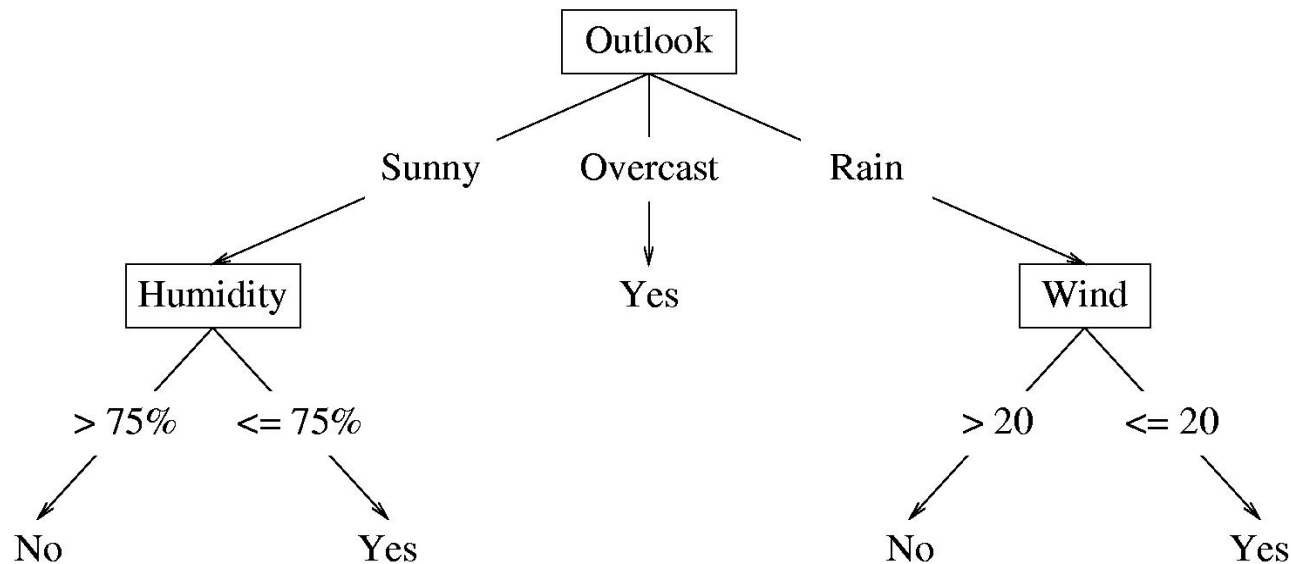
- A possible decision tree for the data:



- What prediction would we make for  
<outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

# Decision Tree

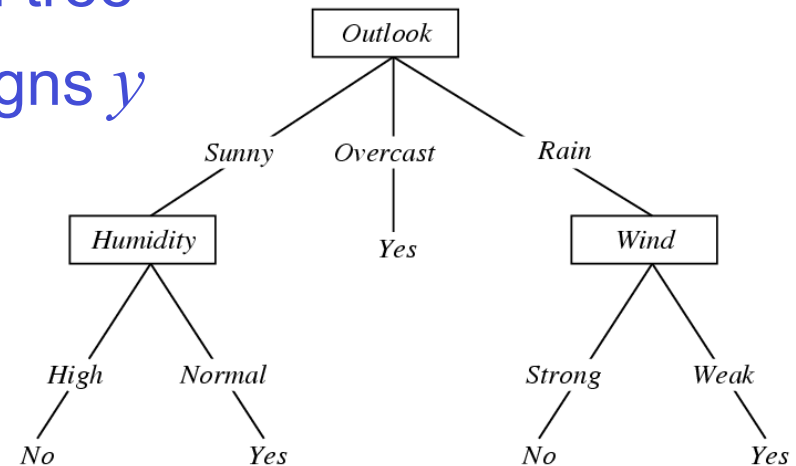
- If features are continuous, internal nodes can test the value of a feature against a **threshold**



# Decision Tree Learning

## Problem Setting:

- Set of possible instances  $X$ 
  - each instance  $x$  in  $X$  is a feature vector
  - e.g.,  $\langle \text{Humidity}=\text{low}, \text{Wind}=\text{weak}, \text{Outlook}=\text{rain}, \text{Temp}=\text{hot} \rangle$
- Unknown target function  $f: X \rightarrow Y$ 
  - $Y$  is discrete valued
- Set of function hypotheses  $H = \{ h \mid h: X \rightarrow Y \}$ 
  - each hypothesis  $h$  is a decision tree
  - trees sorts  $x$  to leaf, which assigns  $y$



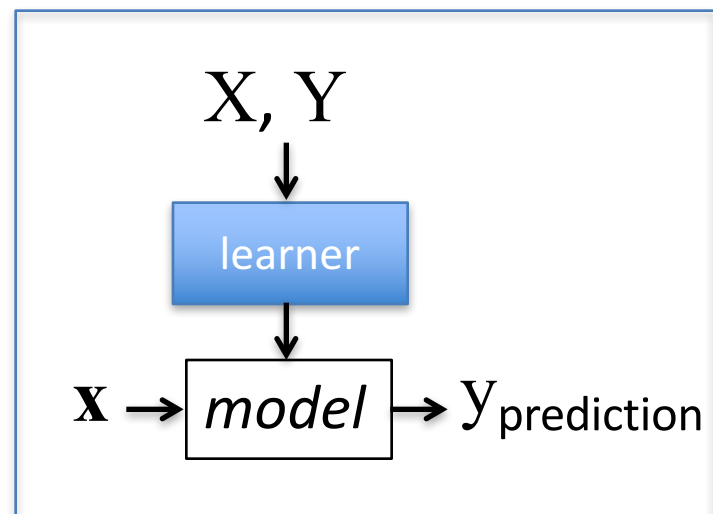
# Stages of (Batch) Machine Learning

**Given:** labeled training data  $X, Y = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^n$

- Assumes each  $\mathbf{x}_i \sim \mathcal{D}(\mathcal{X})$  with  $y_i = f_{target}(\mathbf{x}_i)$

**Train the model:**

$model \leftarrow classifier.train(X, Y)$



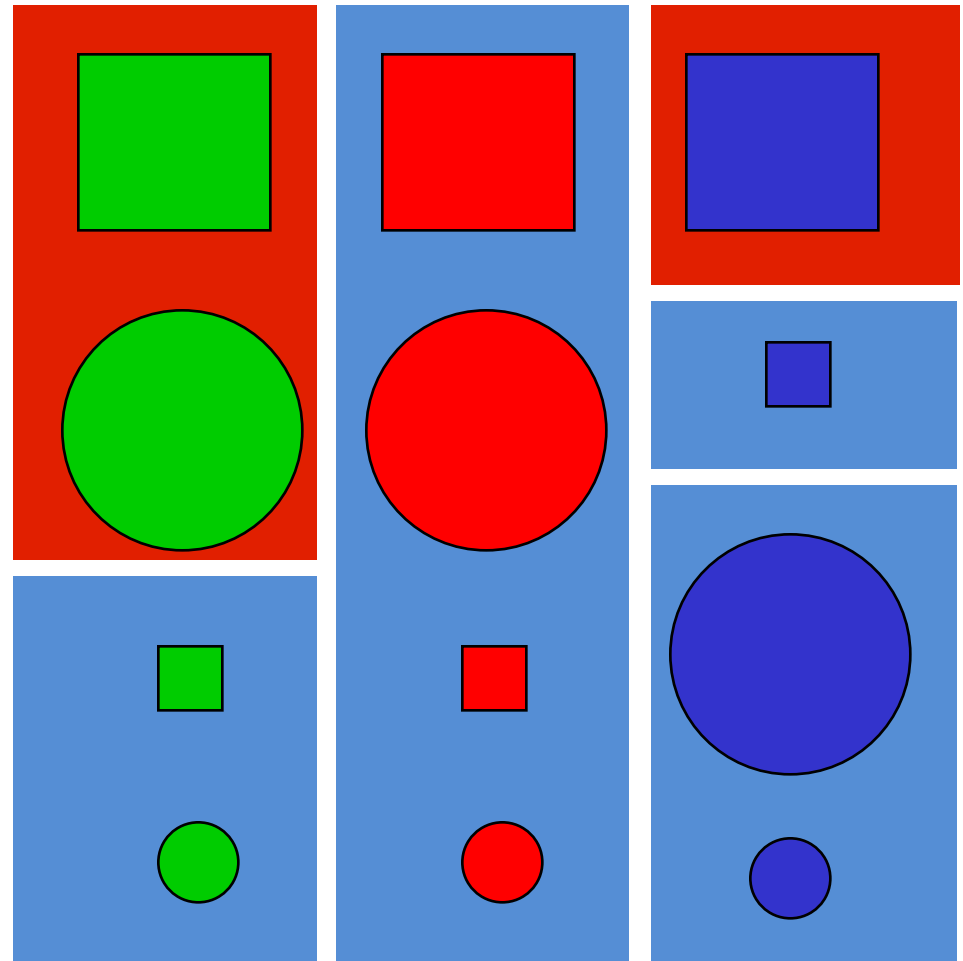
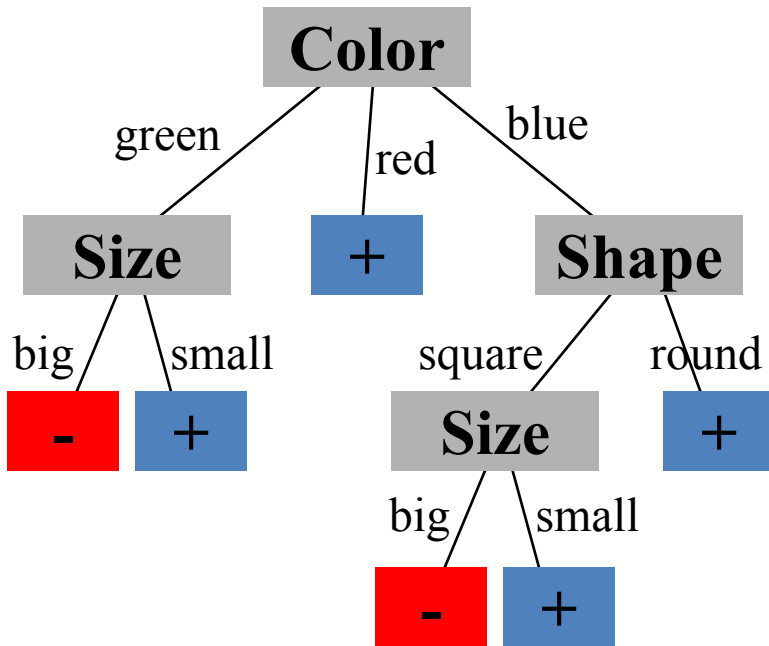
**Apply the model to new data:**

- Given: new unlabeled instance  $\mathbf{x} \sim \mathcal{D}(\mathcal{X})$

$y_{prediction} \leftarrow model.predict(\mathbf{x})$

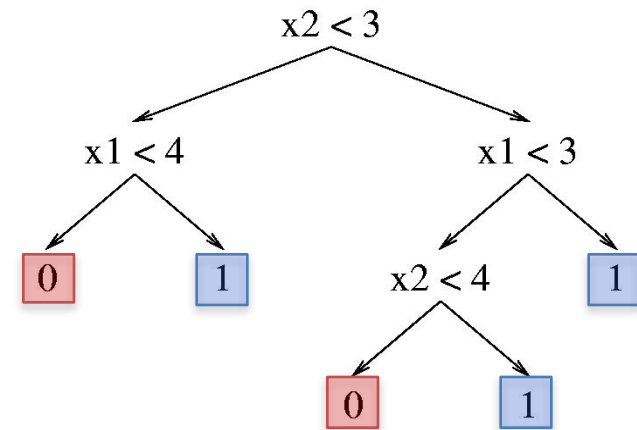
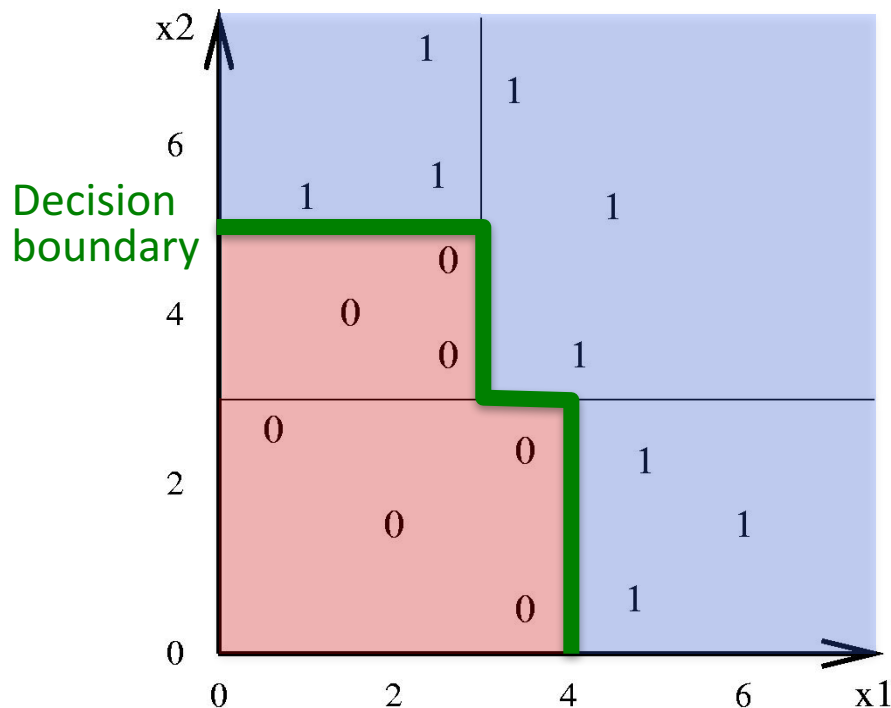


# Decision Tree Induced Partition



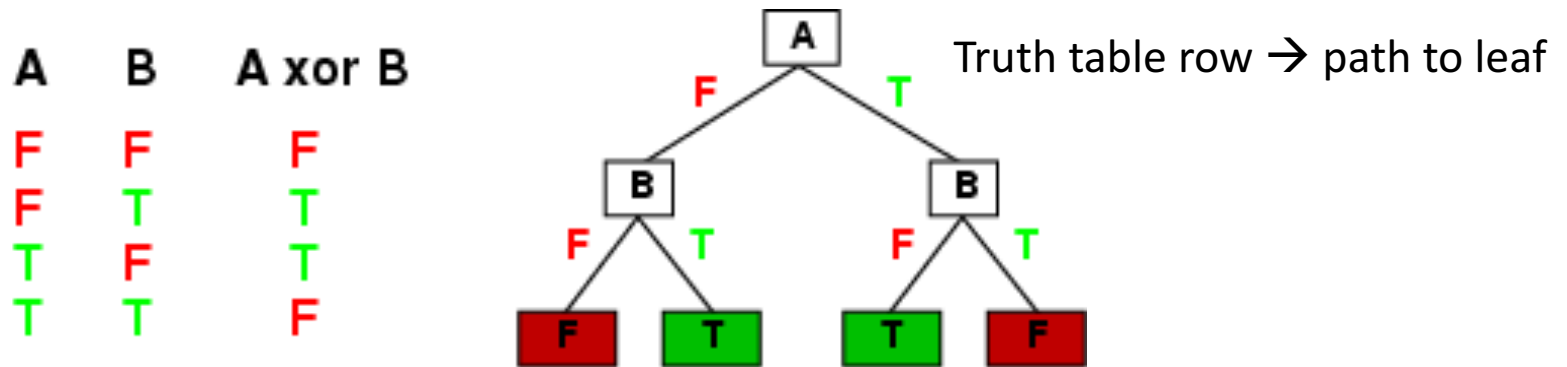
# Decision Tree – Decision Boundary

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label
  - or a probability distribution over labels



# Expressiveness

- Decision trees can represent any boolean function of the input attributes

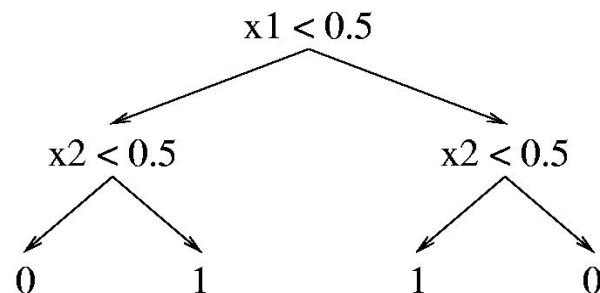
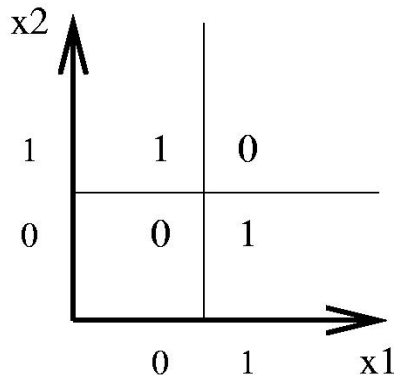


- In the worst case, the tree will require exponentially many nodes

# Expressiveness

Decision trees have a variable-sized hypothesis space

- As the #nodes (or depth) increases, the hypothesis space grows
  - Depth 1 (“decision stump”): can represent any boolean function of one feature
  - Depth 2: any boolean fn of two features; some involving three features (e.g.,  $(x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_3)$ )
  - etc.



# Another Example:

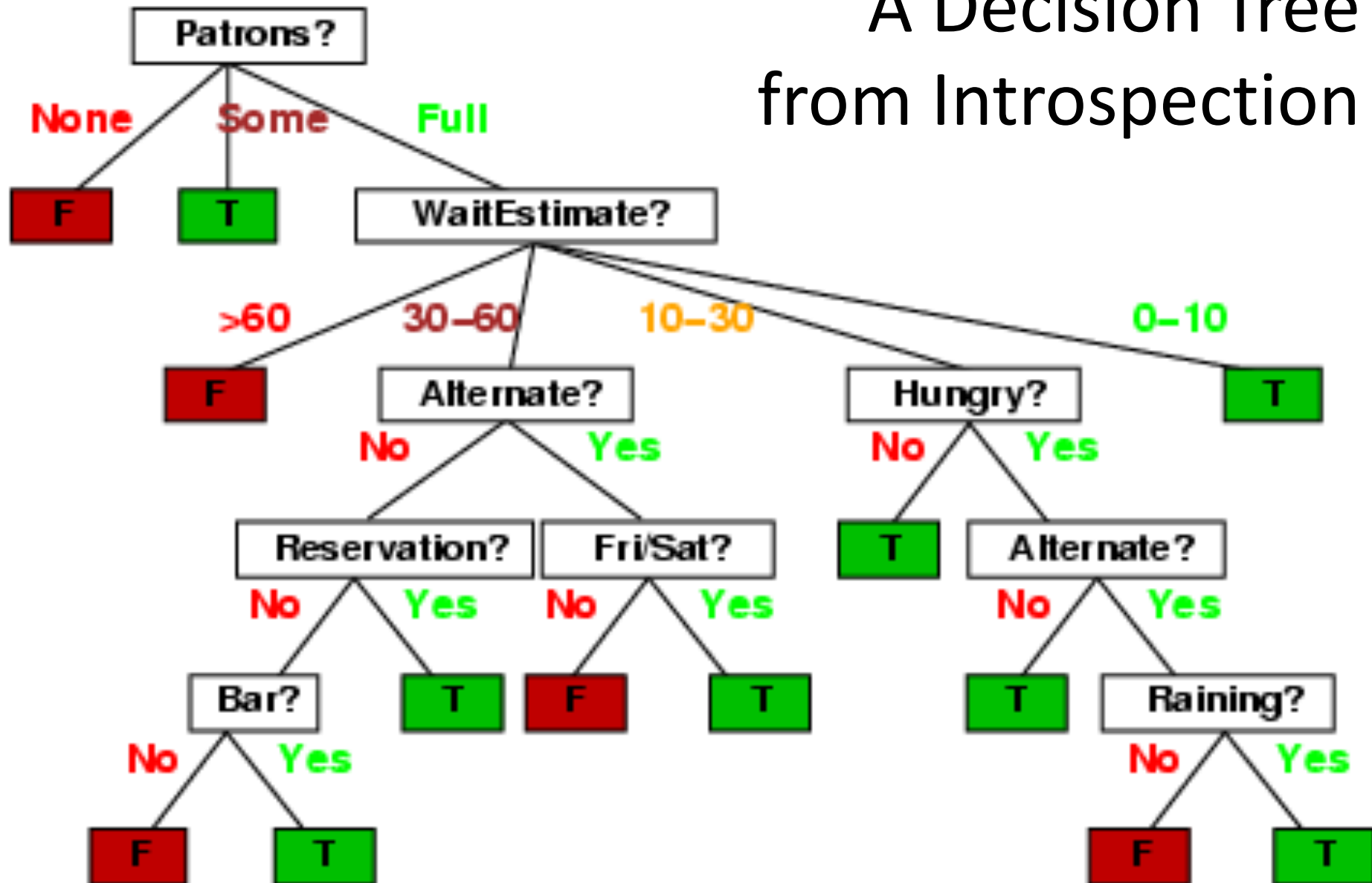
## Restaurant Domain (Russell & Norvig)

Model a patron's decision of whether to wait for a table at a restaurant

Example	Attributes										Target
	<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>Wait</i>
$X_1$	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

~7,000 possible cases

# A Decision Tree from Introspection



Is this the best decision tree?

# Preference bias: Ockham's Razor

- Principle stated by William of Ockham (1285-1347)
  - “*non sunt multiplicanda entia praeter necessitatem*”
  - entities are not to be multiplied beyond necessity
  - AKA Occam's Razor, Law of Economy, or Law of Parsimony

**Idea:** The simplest consistent explanation is the best

- Therefore, the smallest decision tree that correctly classifies all of the training examples is best
  - Finding the provably smallest decision tree is NP-hard
  - ...So instead of constructing the absolute smallest tree consistent with the training examples, construct one that is pretty small

# Basic Algorithm for Top-Down Induction of Decision Trees

[ID3, C4.5 by Quinlan]

*node* = root of decision tree

Main loop:

1.  $A \leftarrow$  the “best” decision attribute for the next node.
2. Assign  $A$  as decision attribute for *node*.
3. For each value of  $A$ , create a new descendant of *node*.
4. Sort training examples to leaf nodes.
5. If training examples are perfectly classified, stop. Else, recurse over new leaf nodes.

How do we choose which attribute is best?



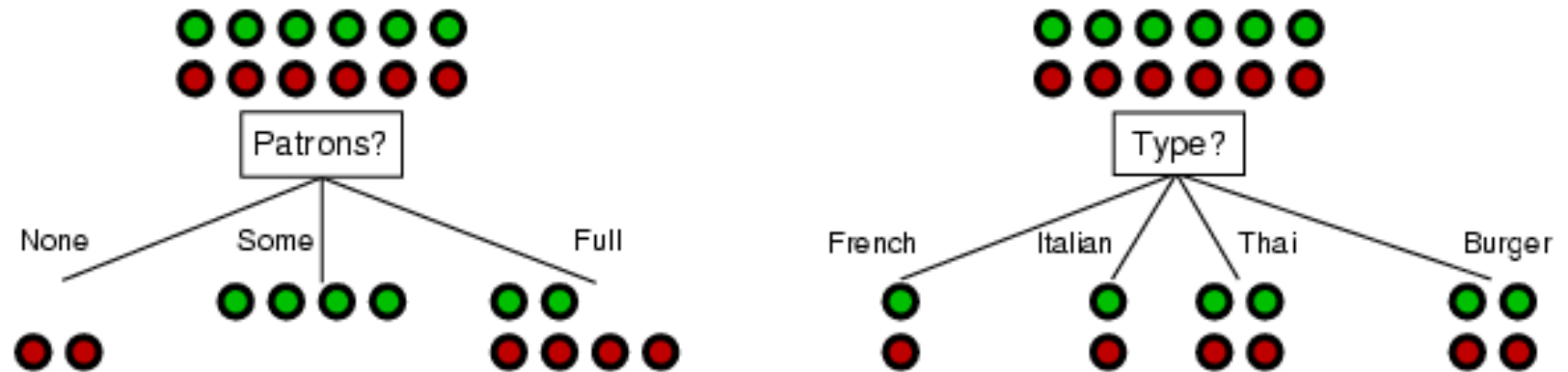
# Choosing the Best Attribute

**Key problem:** choosing which attribute to split a given set of examples

- Some possibilities are:
  - **Random:** Select any attribute at random
  - **Least-Values:** Choose the attribute with the smallest number of possible values
  - **Most-Values:** Choose the attribute with the largest number of possible values
  - **Max-Gain:** Choose the attribute that has the largest expected *information gain*
    - i.e., attribute that results in smallest expected size of subtrees rooted at its children
- The ID3 algorithm uses the Max-Gain method of selecting the best attribute

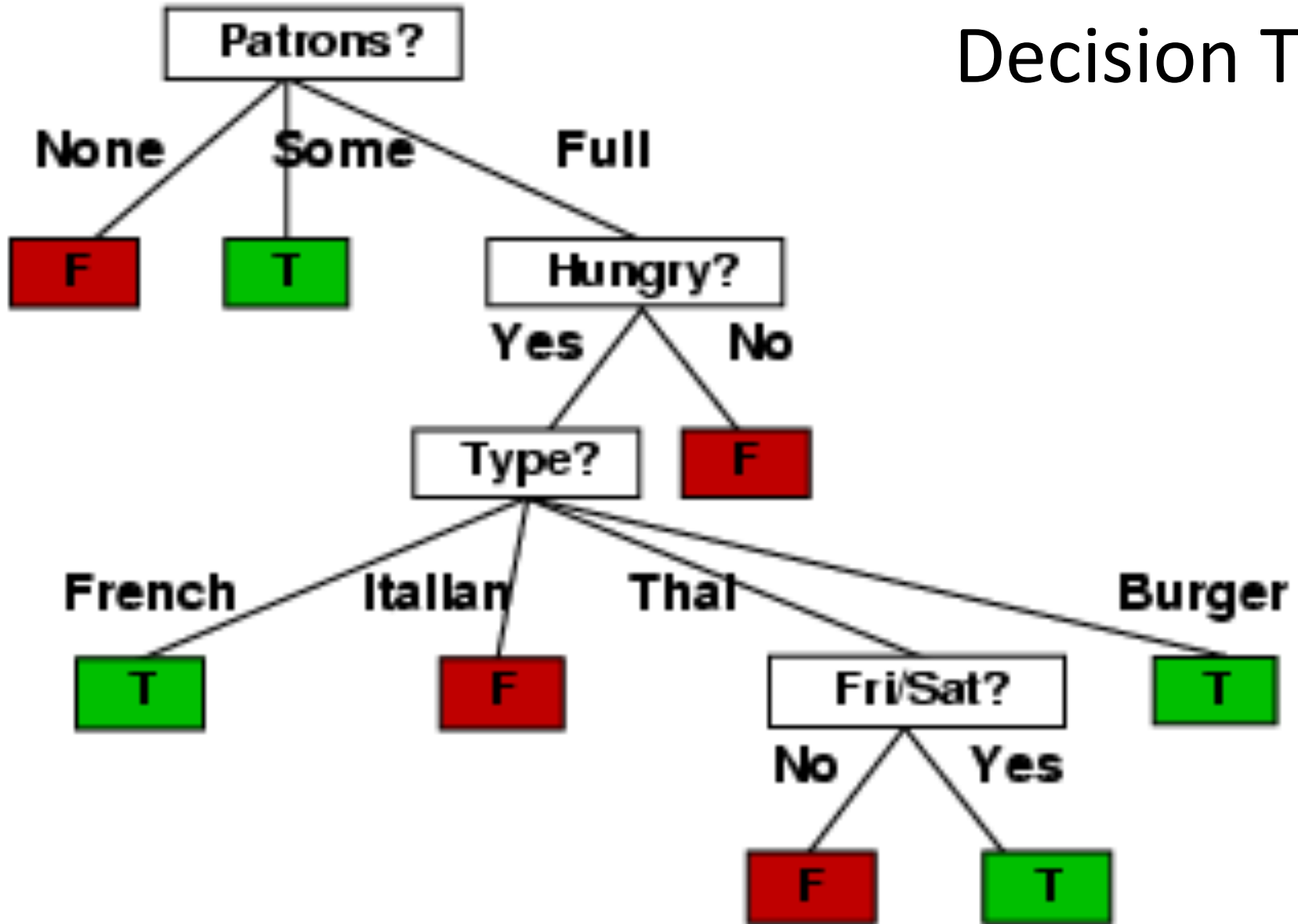
# Choosing an Attribute

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

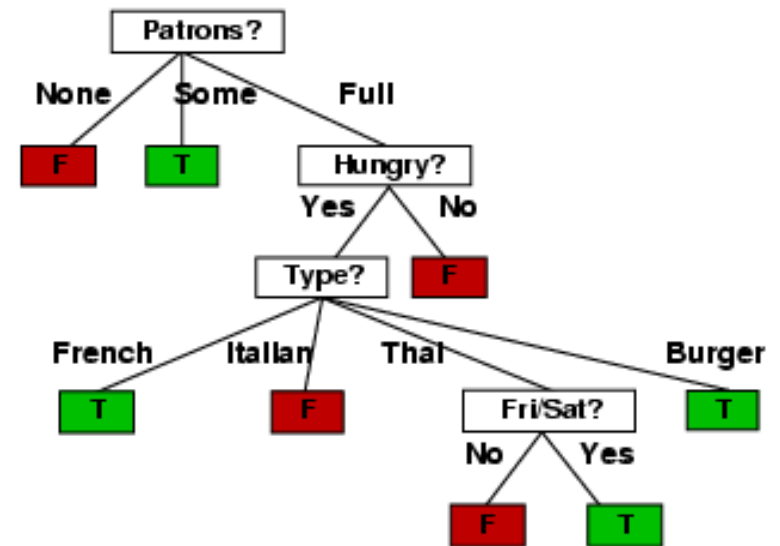
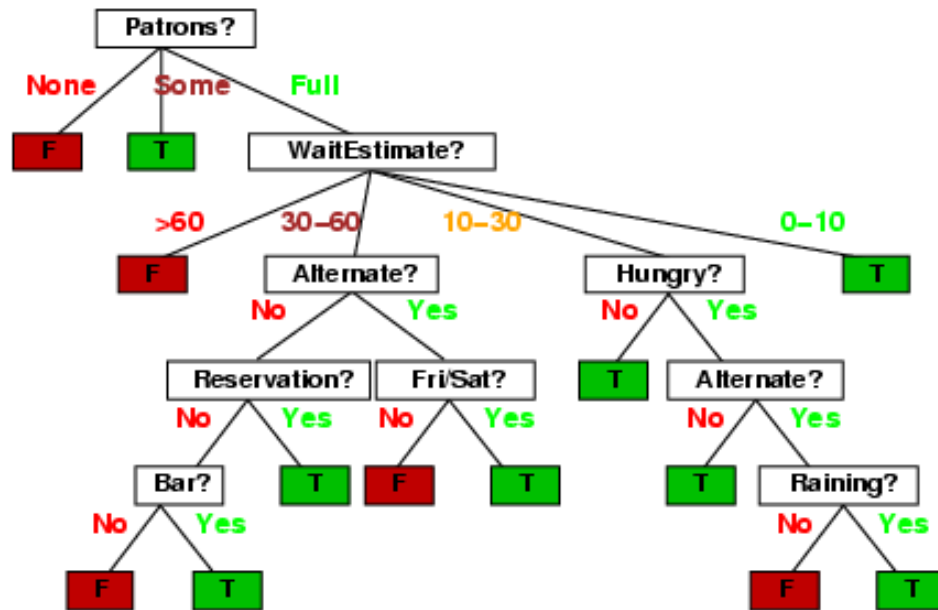


Which split is more informative: *Patrons?* or *Type?*

# ID3-induced Decision Tree



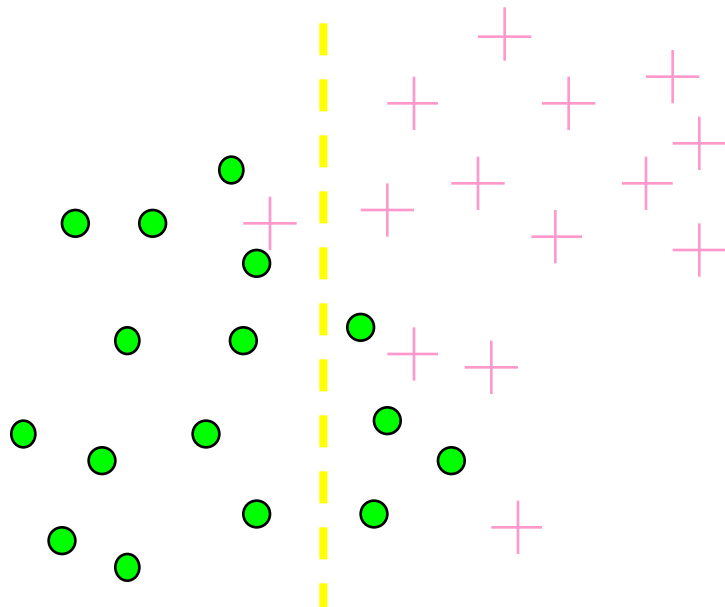
# Compare the Two Decision Trees



# Information Gain

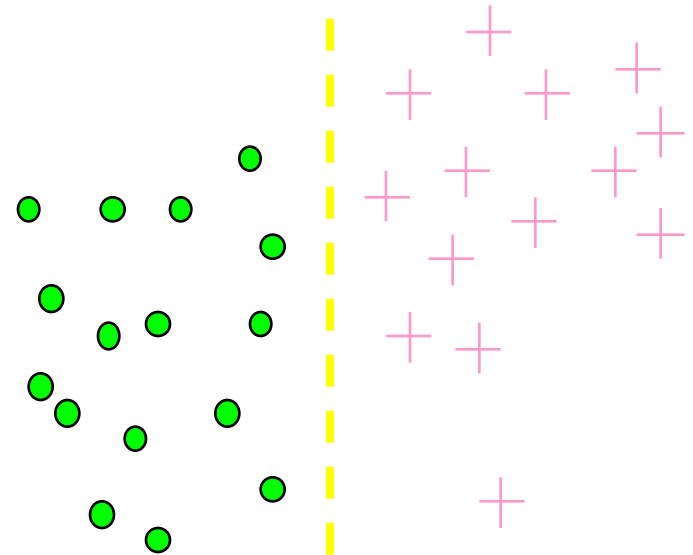
Which test is more informative?

**Split over whether  
Balance exceeds 50K**



Less or equal 50K      Over 50K

**Split over whether  
applicant is employed**

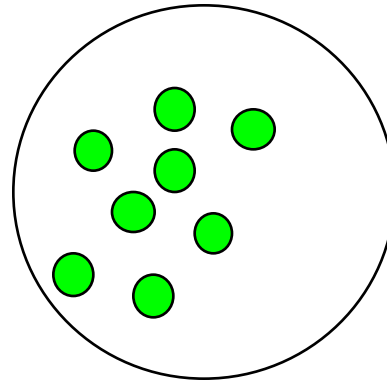
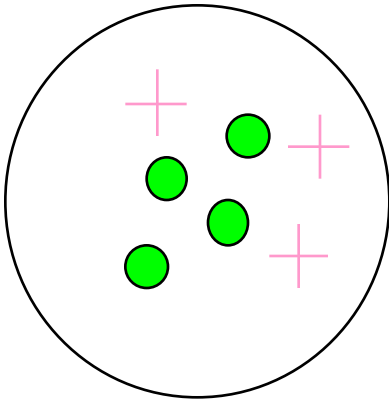


Unemployed      Employed

# Information Gain

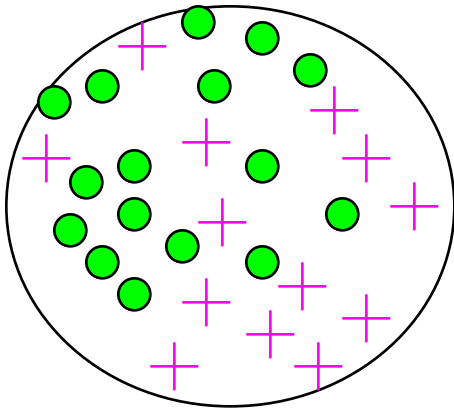
## Impurity/Entropy (informal)

- Measures the level of **impurity** in a group of examples

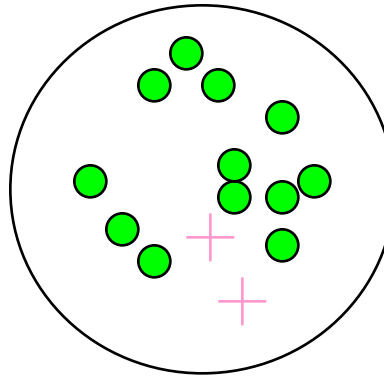


# Impurity

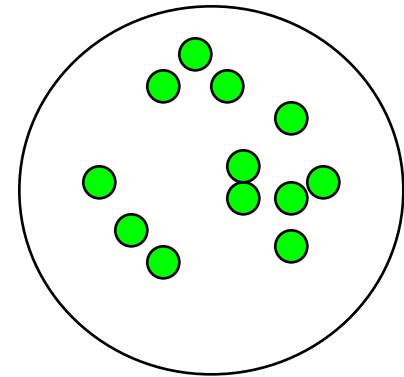
**Very impure group**



**Less impure**



**Minimum  
impurity**

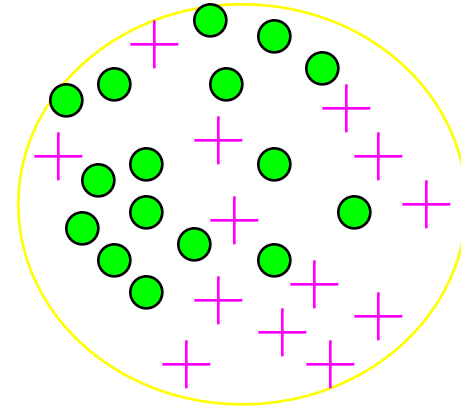


# Entropy: a common way to measure impurity

- Entropy = 
$$\sum_i -p_i \log_2 p_i$$

$p_i$  is the probability of class  $i$

Compute it as the proportion of class  $i$  in the set.



- Entropy comes from information theory. The higher the entropy the more the information content.

What does that mean for learning from examples?



# 2-Class Cases:

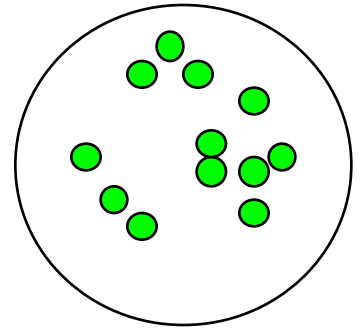
$$\text{Entropy } H(x) = - \sum_{i=1}^n P(x = i) \log_2 P(x = i)$$

- What is the entropy of a group in which all examples belong to the same class?

- entropy =  $-1 \log_2 1 = 0$

not a good training set for learning

**Minimum impurity**

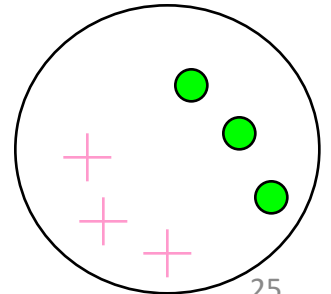


- What is the entropy of a group with 50% in either class?

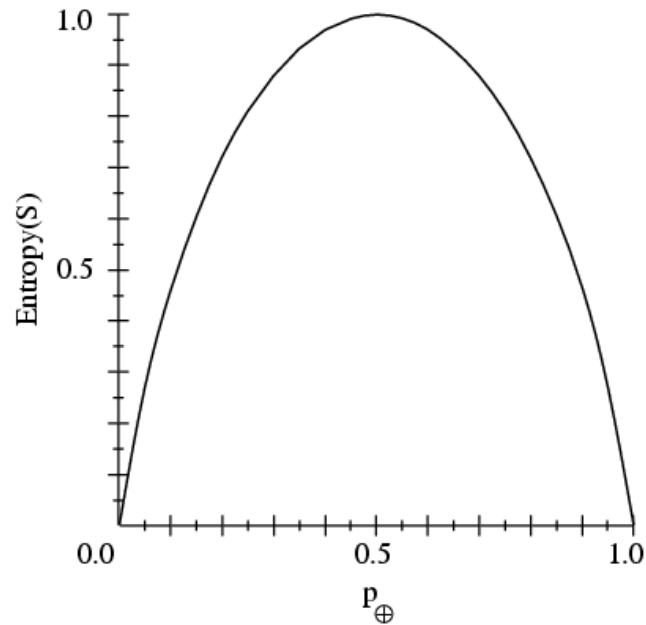
- entropy =  $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

good training set for learning

**Maximum impurity**



# Sample 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  $S$
- Entropy measures the impurity of  $S$

$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

# Information Gain

- We want to determine **which attribute** in a given set of training feature vectors is **most useful** for discriminating between the classes to be learned.
- **Information gain** tells us how important a given attribute of the feature vectors is.
- We will use it to decide the ordering of attributes in the nodes of a decision tree.

# From Entropy to Information Gain

Entropy  $H(X)$  of a random variable  $X$

$$H(X) = - \sum_{i=1}^n P(X = i) \log_2 P(X = i)$$

Specific conditional entropy  $H(X|Y=v)$  of  $X$  given  $Y=v$  :

$$H(X|Y = v) = - \sum_{i=1}^n P(X = i|Y = v) \log_2 P(X = i|Y = v)$$

Conditional entropy  $H(X|Y)$  of  $X$  given  $Y$  :

$$H(X|Y) = \sum_{v \in \text{values}(Y)} P(Y = v) H(X|Y = v)$$

Mutual information (aka Information Gain) of  $X$  and  $Y$  :

$$I(X, Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

# Information Gain

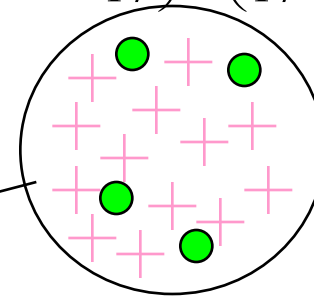
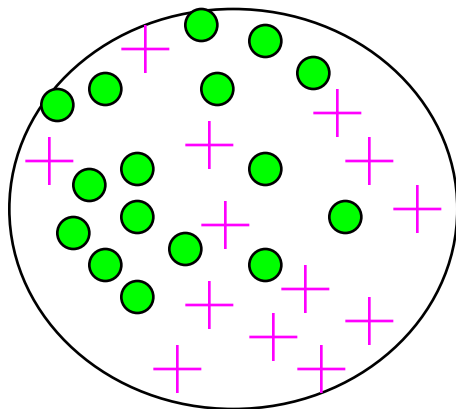
Information Gain is the expected reduction in entropy of target variable  $Y$  for data sample  $S$ , due to sorting

# Calculating Information Gain

**Information Gain** = entropy(parent) – [average entropy(children)]

**child entropy**  $-\left(\frac{13}{17} \cdot \log_2 \frac{13}{17}\right) - \left(\frac{4}{17} \cdot \log_2 \frac{4}{17}\right) = 0.787$

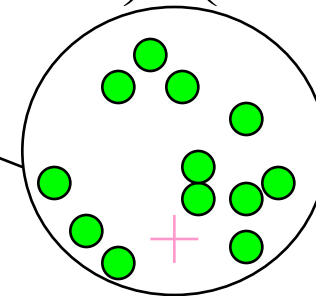
Entire population (30 instances)



17 instances

**child entropy**  $-\left(\frac{1}{13} \cdot \log_2 \frac{1}{13}\right) - \left(\frac{12}{13} \cdot \log_2 \frac{12}{13}\right) = 0.391$

**parent entropy**  $-\left(\frac{14}{30} \cdot \log_2 \frac{14}{30}\right) - \left(\frac{16}{30} \cdot \log_2 \frac{16}{30}\right) = 0.996$



13 instances

**(Weighted) Average Entropy of Children** =  $\left(\frac{17}{30} \cdot 0.787\right) + \left(\frac{13}{30} \cdot 0.391\right) = 0.615$

**Information Gain = 0.996 - 0.615 = 0.38**

# Entropy-Based Automatic Decision Tree Construction

Training Set X

$x_1 = (f_{11}, f_{12}, \dots, f_{1m})$

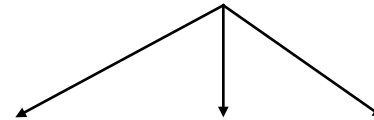
$x_2 = (f_{21}, f_{22}, \dots, f_{2m})$

$\vdots$

$\vdots$

$x_n = (f_{n1}, f_{n2}, \dots, f_{nm})$

Node 1  
What feature  
should be used?



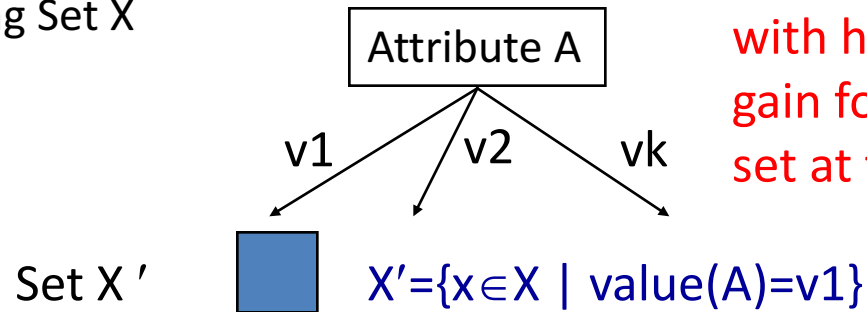
What values?

Quinlan suggested **information gain** in his ID3 system and later the **gain ratio**, both based on **entropy**.

# Using Information Gain to Construct a Decision Tree

Full Training Set X

Choose the attribute A with highest information gain for the full training set at the root of the tree.



repeat  
recursively  
till when?

Construct child nodes for each value of A. Each has an associated subset of vectors in which A has a particular value.



# Training Examples

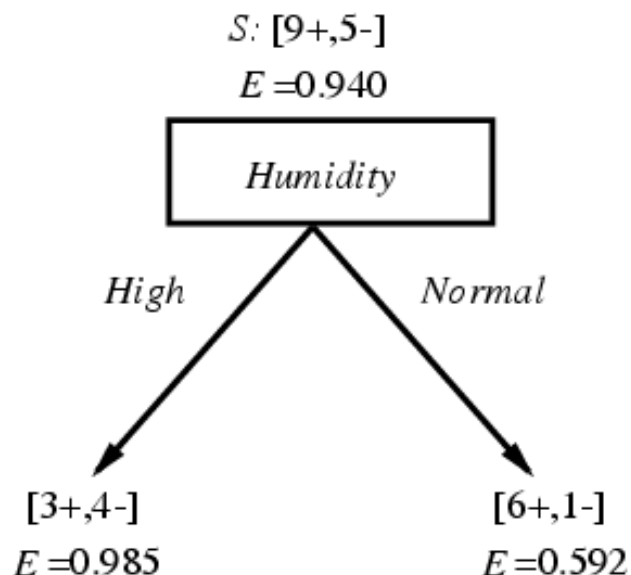
---

Day	Outlook	Temperature	Humidity	Wind	PlayTenn
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	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	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

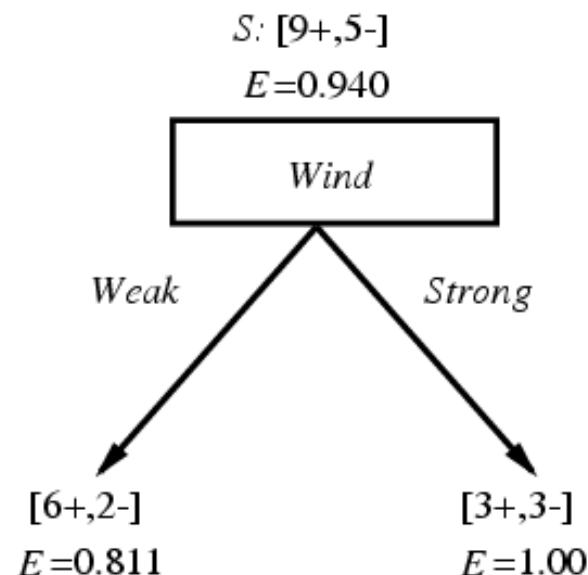
# Selecting the Next Attribute

---

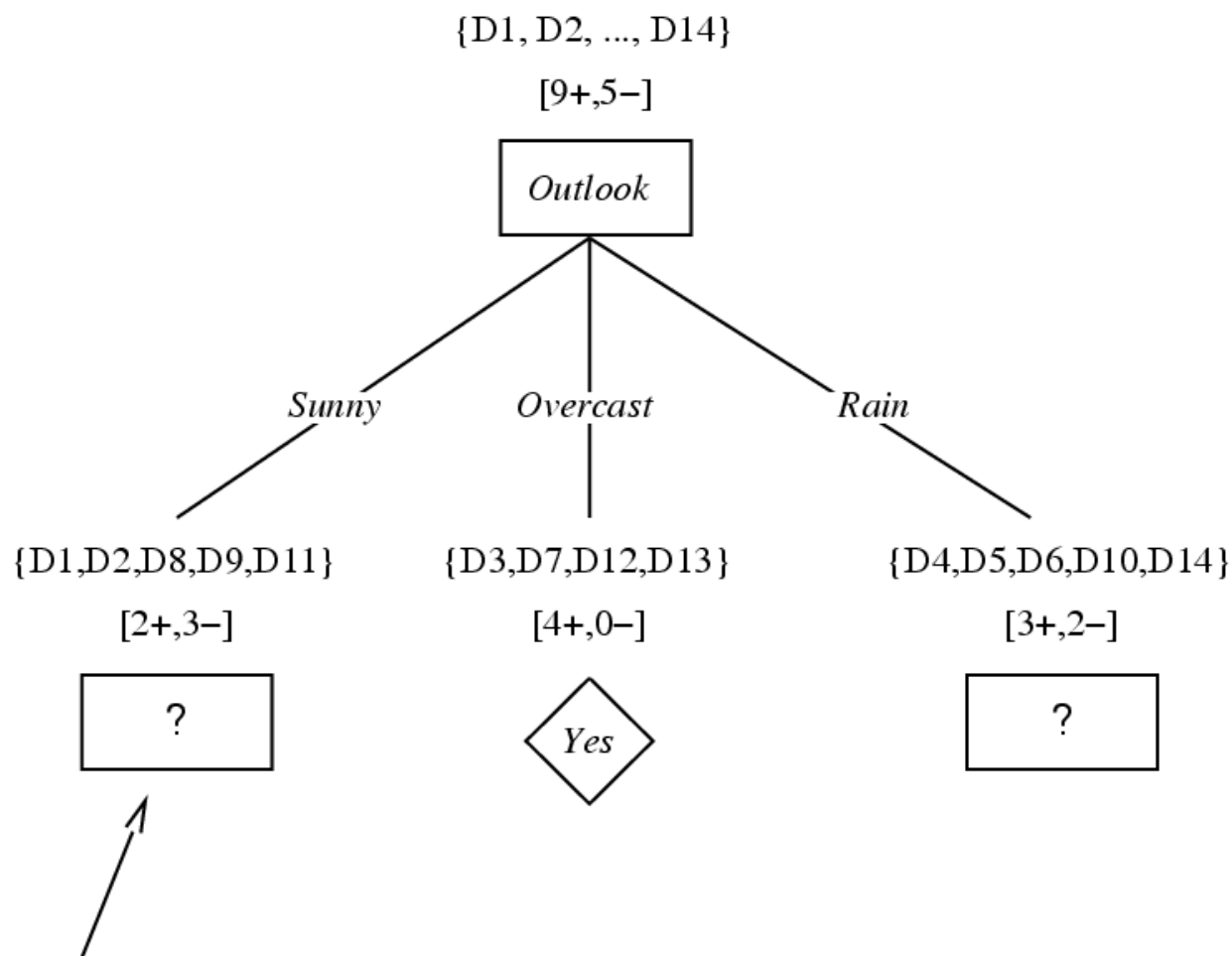
Which attribute is the best classifier?



$$\begin{aligned} \text{Gain}(S, \text{Humidity}) &= .940 - (7/14).985 - (7/14).592 \\ &= .151 \end{aligned}$$



$$\begin{aligned} \text{Gain}(S, \text{Wind}) &= .940 - (8/14).811 - (6/14)1.0 \\ &= .048 \end{aligned}$$



*Which attribute should be tested here?*

$$S_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

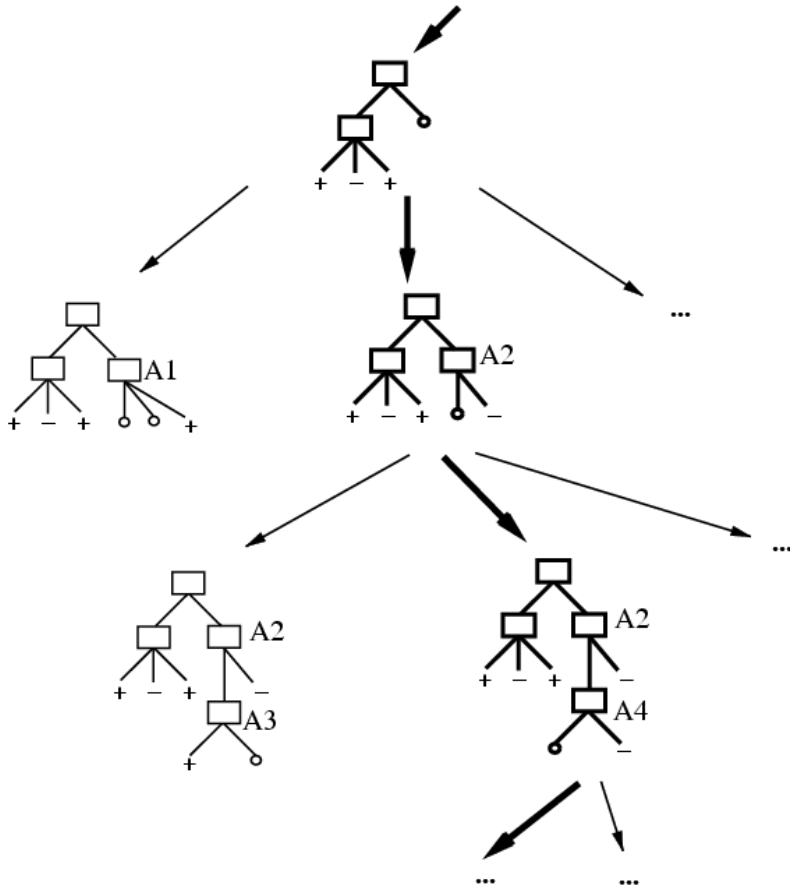
$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970$$

$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5) 1.0 - (3/5) .918 = .019$$

# Which Tree Should We Output?

- ID3 performs heuristic search through space of decision trees
- It stops at smallest acceptable tree. Why?



The ID3 algorithm builds a decision tree, given a set of non-categorical attributes  $C_1, C_2, \dots, C_n$ , the class attribute  $C$ , and a training set  $T$  of records

```
function ID3(R:input attributes, C:class attribute,  
S:training set) returns decision tree;
```

```
    If  $S$  is empty, return single node with value Failure;
```

```
    If every example in  $S$  has same value for  $C$ , return  
    single node with that value;
```

```
    If  $R$  is empty, then return a single node with most  
    frequent of the values of  $C$  found in examples  $S$ ;
```

```
    # causes errors -- improperly classified record
```

```
    Let  $D$  be attribute with largest  $\text{Gain}(D, S)$  among  $R$ ;
```

```
    Let  $\{d_j \mid j=1, 2, \dots, m\}$  be values of attribute  $D$ ;
```

```
    Let  $\{S_j \mid j=1, 2, \dots, m\}$  be subsets of  $S$  consisting of  
    records with value  $d_j$  for attribute  $D$ ;
```

```
    Return tree with root labeled  $D$  and arcs labeled  
     $d_1 \dots d_m$  going to the trees  $\text{ID3}(R - \{D\}, C, S_1) \dots$   
     $\text{ID3}(R - \{D\}, C, S_m)$ ;
```

# How well does it work?

Many case studies have shown that decision trees are at least as accurate as human experts.

- A study for diagnosing breast cancer had humans correctly classifying the examples 65% of the time; the decision tree classified 72% correct
- British Petroleum designed a decision tree for gas-oil separation for offshore oil platforms that replaced an earlier rule-based expert system
- Cessna designed an airplane flight controller using 90,000 examples and 20 attributes per example

# Extensions of ID3

- Using gain ratios
- Real-valued data
- Noisy data and overfitting
- Generation of rules
- Setting parameters
- Cross-validation for experimental validation of performance
- C4.5 is an extension of ID3 that accounts for unavailable values, continuous attribute value ranges, pruning of decision trees, rule derivation, and so on