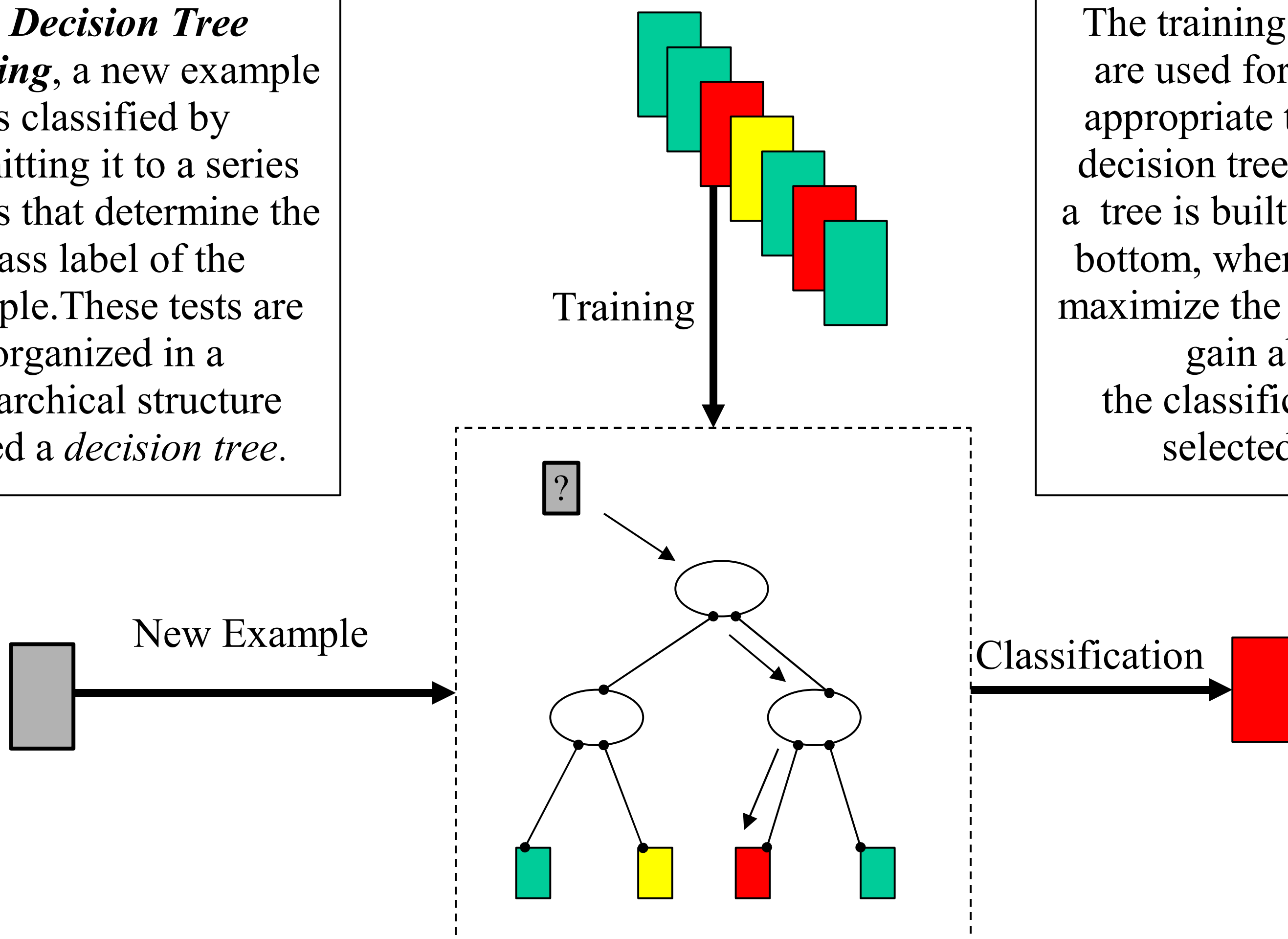
Decision Tree Learning

In ***Decision Tree Learning***, a new example is classified by submitting it to a series of tests that determine the class label of the example.These tests are organized in a hierarchical structure called a *decision tree*.

The training examples are used for choosing appropriate tests in the decision tree. Typically, a tree is built from top to bottom, where tests that maximize the information gain about the classification are selected first.

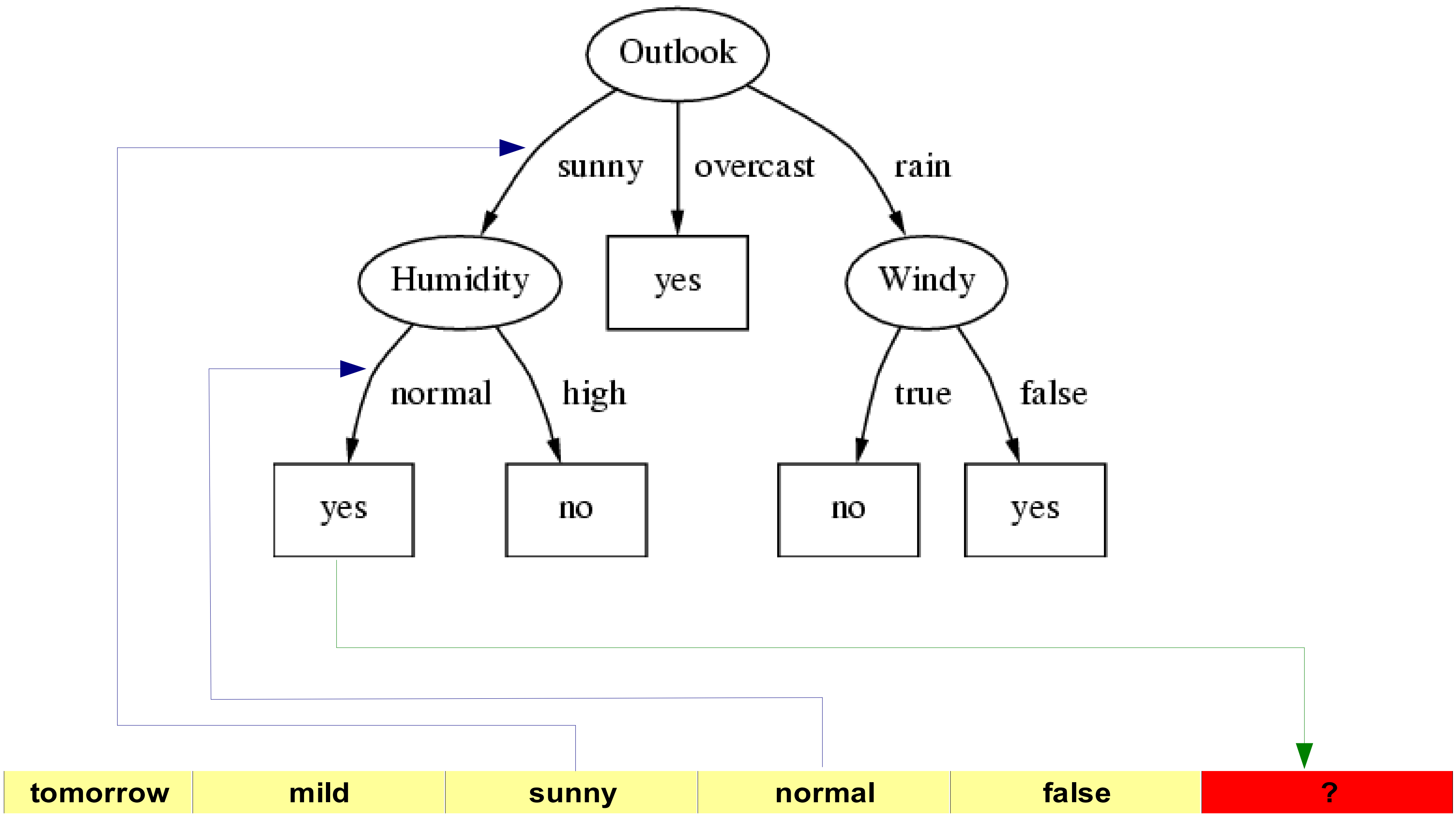


A Sample Task

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

today	cool	sunny	normal	false	?
tomorrow	mild	sunny	normal	false	?

Decision Tree Learning



Divide-And-Conquer Algorithms

- Family of decision tree learning algorithms
  - TDIDT: Top-Down Induction of Decision Trees
- Learn trees in a Top-Down fashion:
  - divide the problem in subproblems
  - solve each problem

Basic Divide-And-Conquer Algorithm:

1. select a test for root node
   Create branch for each possible outcome of the test
2. split instances into subsets
   One for each branch extending from the node
3. repeat recursively for each branch, using only instances that reach the branch
4. stop recursion for a branch if all its instances have the same class

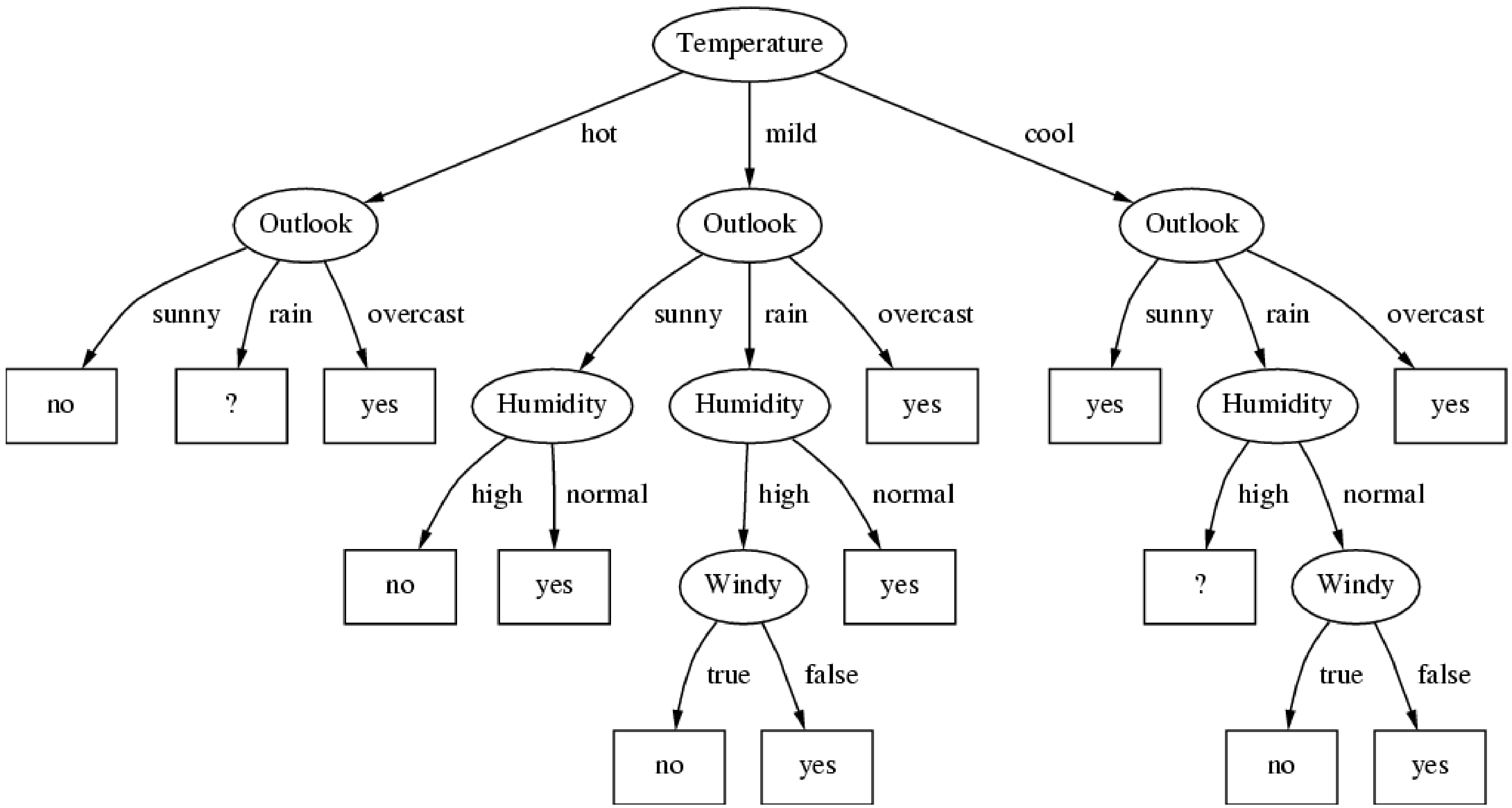
ID3 Algorithm

Function ID3

- **Input:** Example set S
- **Output:** Decision Tree DT

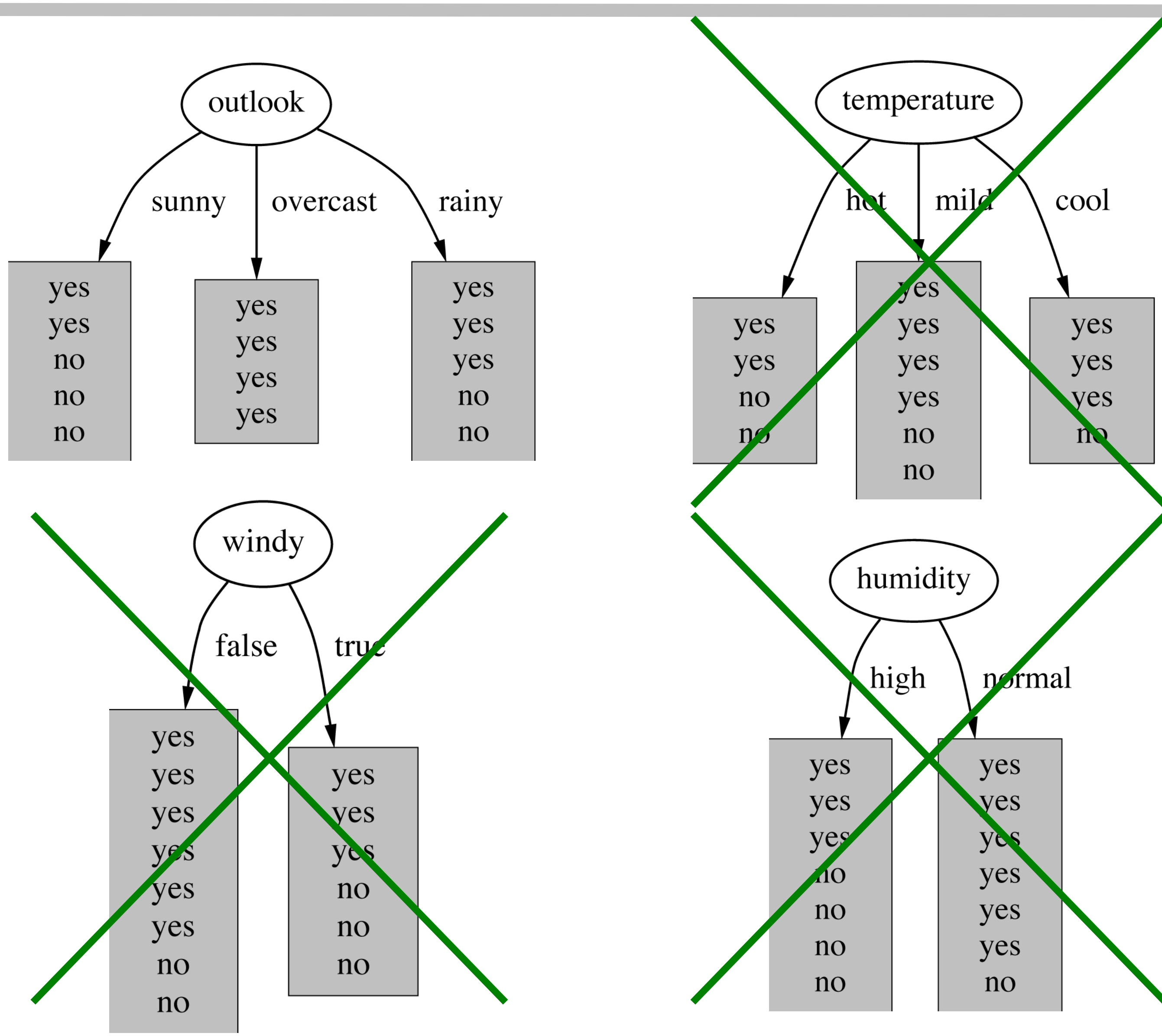
- If all examples in S belong to the same class c
  - return a new leaf and label it with c
- Else
  i. Select an attribute A according to some heuristic function
  ii. Generate a new node DT with A as test
  iii. For each Value v_i of A
    (a) Let S_i = all examples in S with $A = v_i$
    (b) Use ID3 to construct a decision tree DT_i for example set S_i
    (c) Generate an edge that connects DT and DT_i

A Different Decision Tree



- also explains all of the training data
- will it generalize well to new data?

Which attribute to select as the root?



What is a good Attribute?

- We want to grow a simple tree

  → a good attribute prefers attributes that split the data so that each successor node is as *pure* as posssible

  - i.e., the distribution of examples in each node is so that it mostly contains examples of a single class

- In other words:
  - We want a measure that prefers attributes that have a high degree of „order“:
    - Maximum order: All examples are of the same class
    - Minimum order: All classes are equally likely

  → Entropy is a measure for (un-)orderedness

  - Another interpretation:
    - Entropy is the amount of information that is contained
    - all examples of the same class → no information

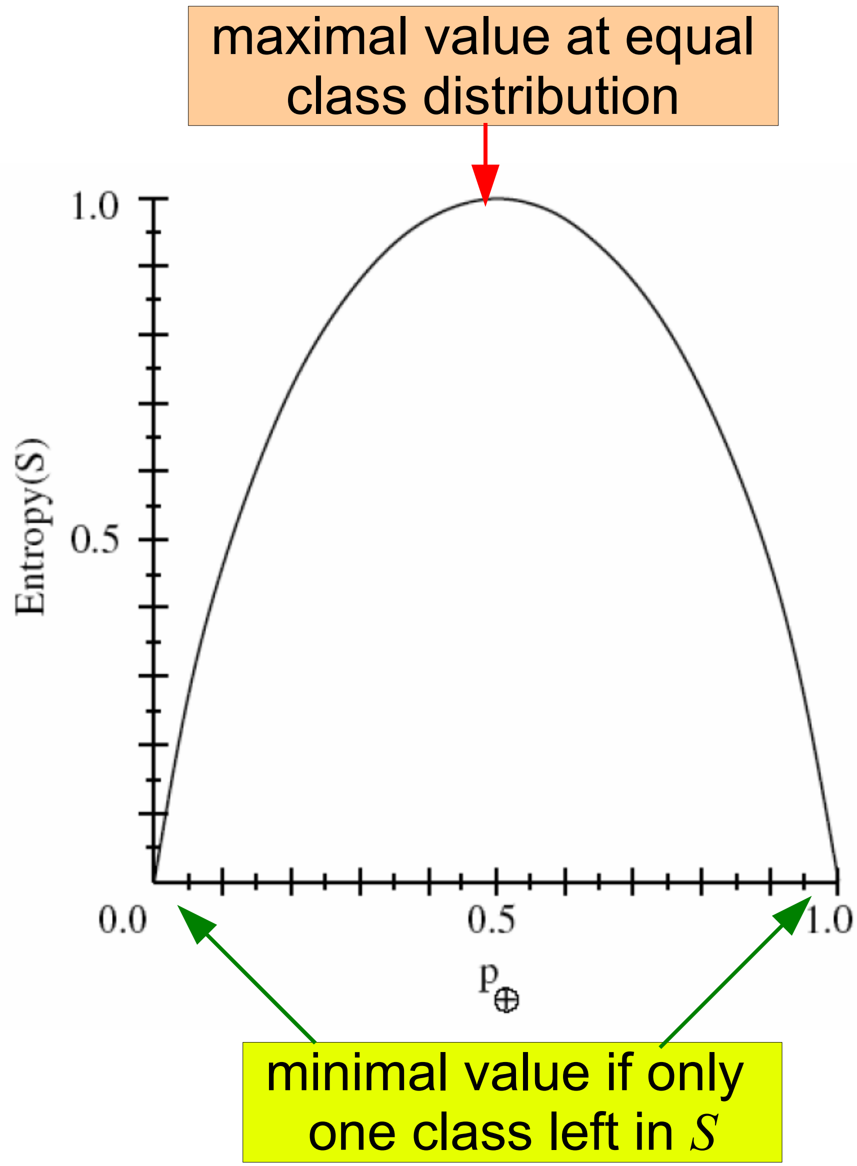
Entropy (for two classes)

- S is a set of examples
- $p_\oplus$ is the proportion of examples in class $\oplus$
- $p_\ominus = 1 - p_\oplus$ is the proportion of examples in class $\ominus$

Entropy:

$$E(S) = -p_\oplus \cdot \log_2 p_\oplus - p_\ominus \cdot \log_2 p_\ominus$$

- Interpretation:
  - amount of unorderedness in the class distribution of S



Example: Attribute Outlook

- Outlook = sunny: 3 examples yes, 2 examples no

$$E(\text{Outlook}=\text{sunny}) = -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) = 0.971$$

- Outlook = overcast: 4 examples yes, 0 examples no

$$E(\text{Outlook}=\text{overcast}) = -1\log(1) - 0\log(0) = 0$$

Note: this is normally undefined. Here: = 0

- Outlook = rainy : 2 examples yes, 3 examples no

$$E(\text{Outlook}=\text{rainy}) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log\left(\frac{2}{5}\right) = 0.971$$

Entropy (for more classes)

- Entropy can be easily generalized for $n > 2$ classes
  - p_i is the proportion of examples in S that belong to the i-th class

$$E(S) = -p_1 \log p_1 - p_2 \log p_2 \ldots - p_n \log p_n = -\sum_{i=1}^{n} p_i \log p_i$$

Average Entropy / Information

- **Problem:**
  - Entropy only computes the quality of a single (sub-)set of examples
    - corresponds to a single value
  - How can we compute the quality of the entire split?
    - corresponds to an entire attribute
- **Solution:**
  - Compute the weighted average over all sets resulting from the split
    - weighted by their size

$$I(S,A)=\sum_i \frac{|S_i|}{|S|}\cdot E(S_i)$$

- **Example:**
  - Average entropy for attribute *Outlook*:

$$I(\text{Outlook})=\tfrac{5}{14}\cdot 0.971+\tfrac{4}{14}\cdot 0+\tfrac{5}{14}\cdot 0.971=0.693$$

Information Gain

- When an attribute A splits the set S into subsets S_i
  - we compute the average entropy
  - and compare the sum to the entropy of the original set S

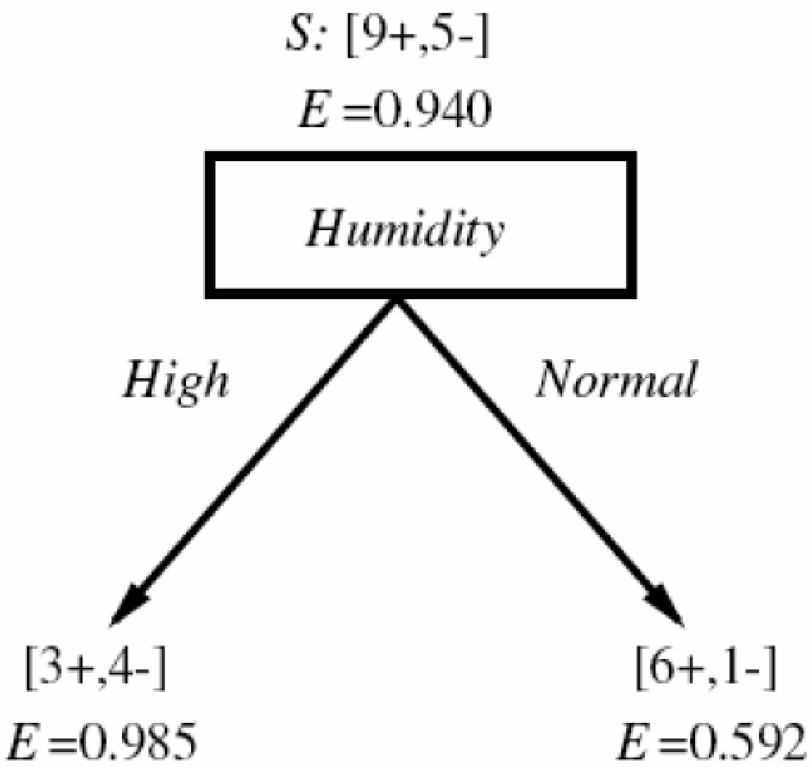
Information Gain for Attribute A

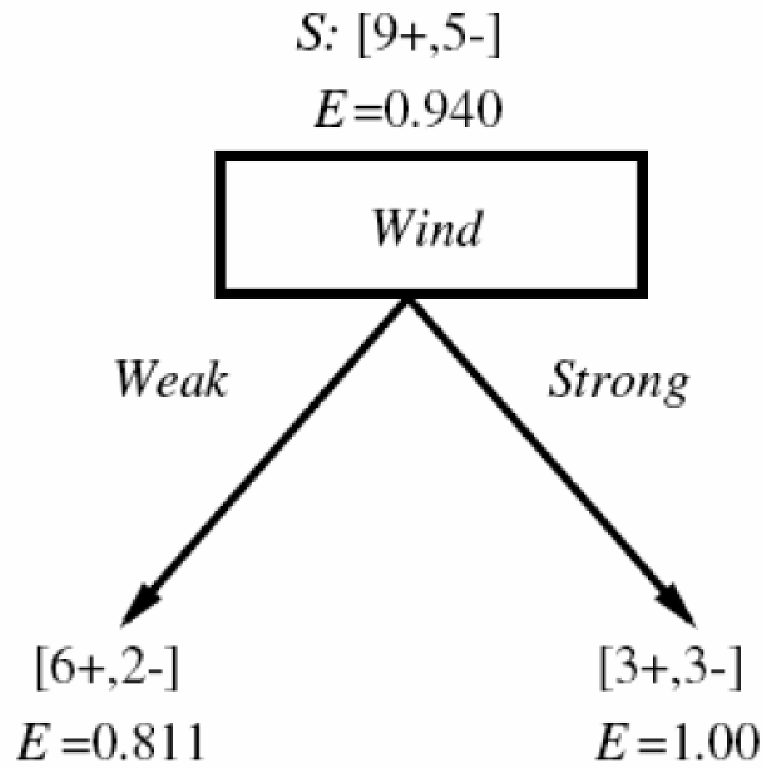
$$Gain(S,A) = E(S) - I(S,A) = E(S) - \sum_i \frac{|S_i|}{|S|} \cdot E(S_i)$$

- The attribute that maximizes the difference is selected
  - i.e., the attribute that reduces the unorderedness most!

- **Note**:
  - maximizing information gain is equivalent to minimizing average entropy, because $E(S)$ is constant for all attributes A

Example





Gain (S, Humidity)

$= .940 - (7/14).985 - (7/14).592$
$= .151$

Gain (S, Wind)

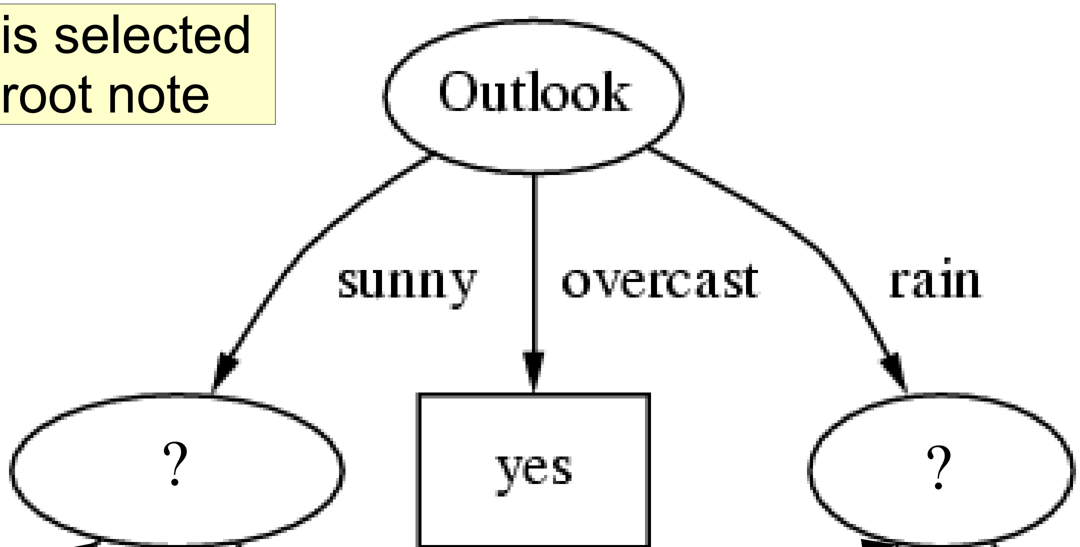
$= .940 - (8/14).811 - (6/14)1.0$
$= .048$

$Gain(S, Outlook) = 0.246$

$Gain(S, Temperature) = 0.029$

Example (Ctd.)

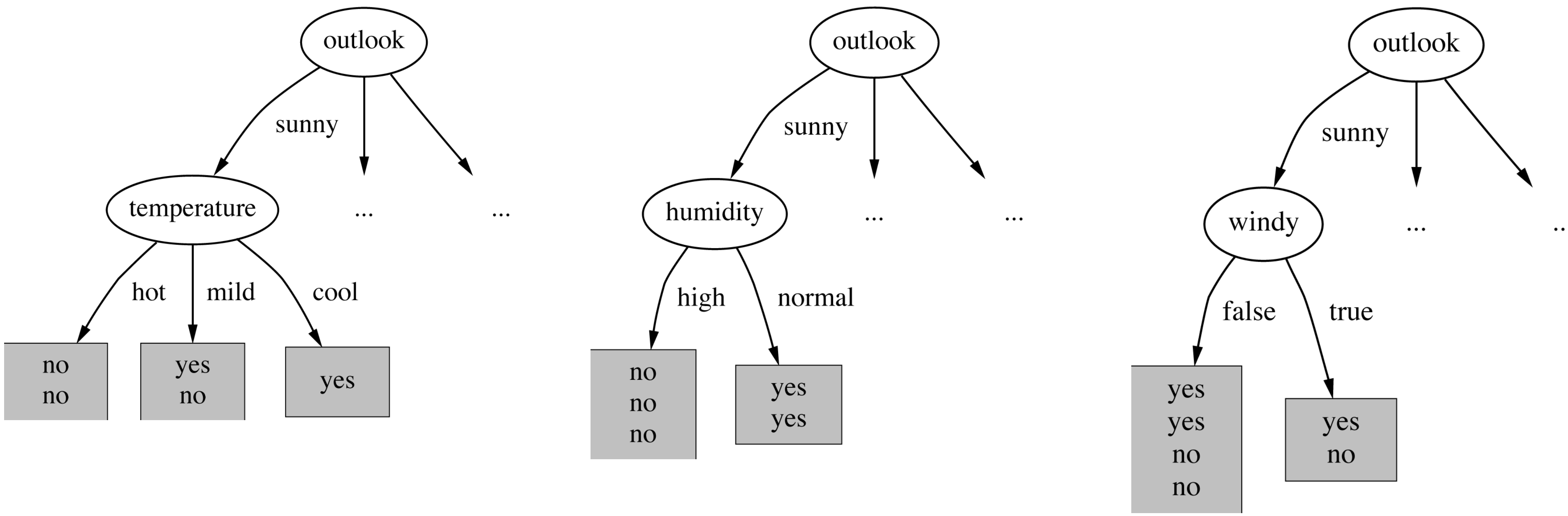
Outlook is selected as the root note



further splitting necessary

Outlook = overcast contains only examples of class **yes**

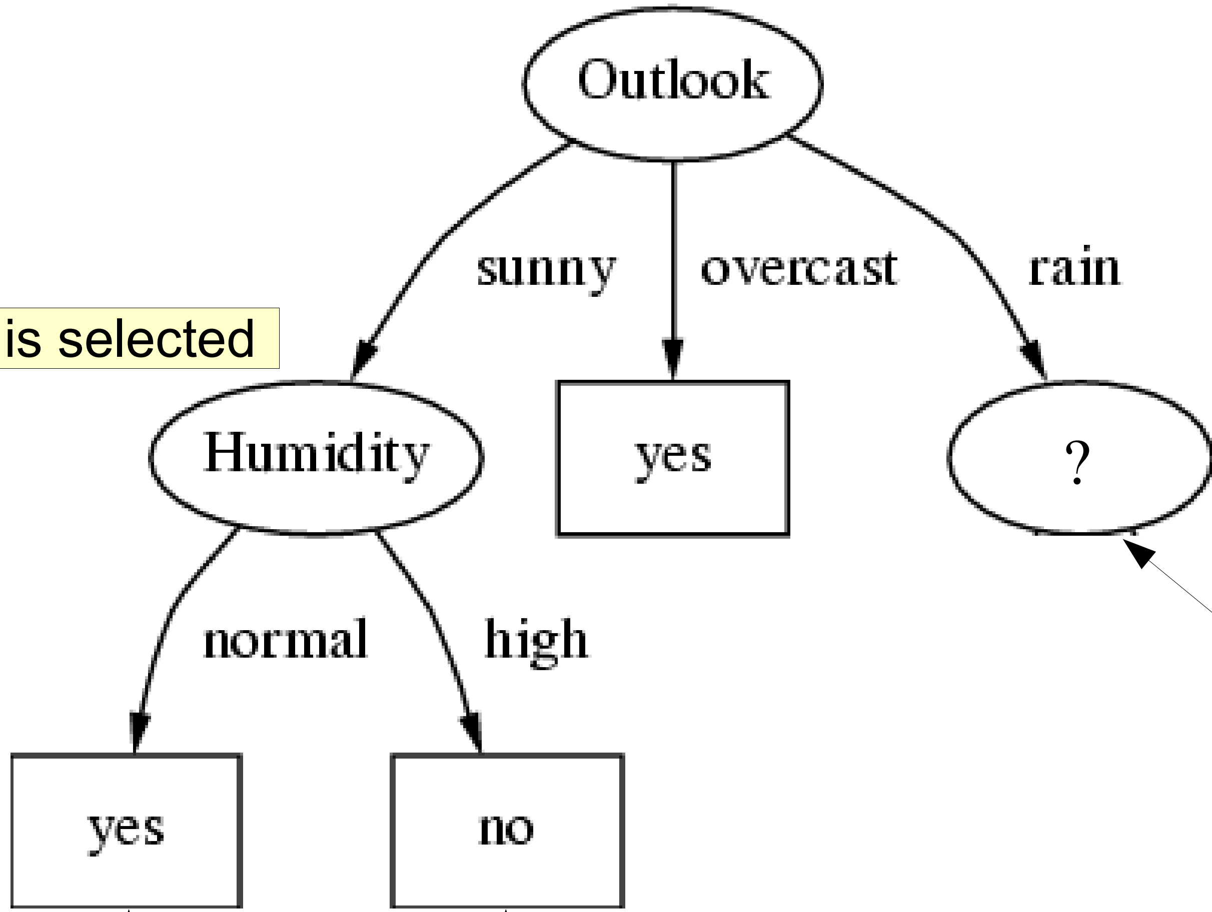
Example (Ctd.)



$$\left.\begin{aligned}\text{Gain}(\textit{Temperature}) &= 0.571 \text{ bits}\\ \text{Gain}(\textit{Humidity}) &= 0.971 \text{ bits}\\ \text{Gain}(\textit{Windy}) &= 0.020 \text{ bits}\end{aligned}\right\}$$

Humidity is selected

Example (Ctd.)

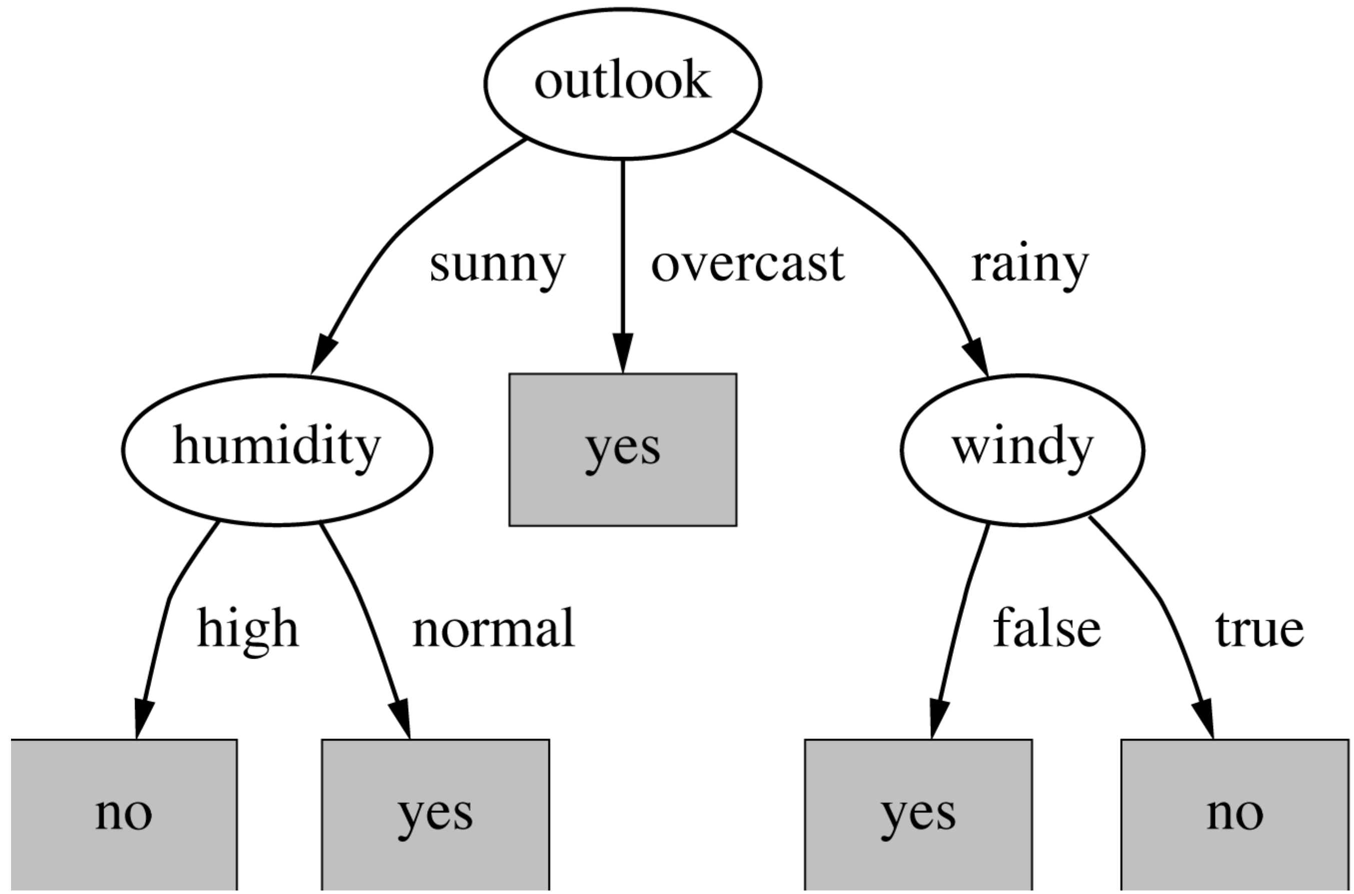


Humidity is selected

further splitting necessary

Pure leaves
→ No further expansion necessary

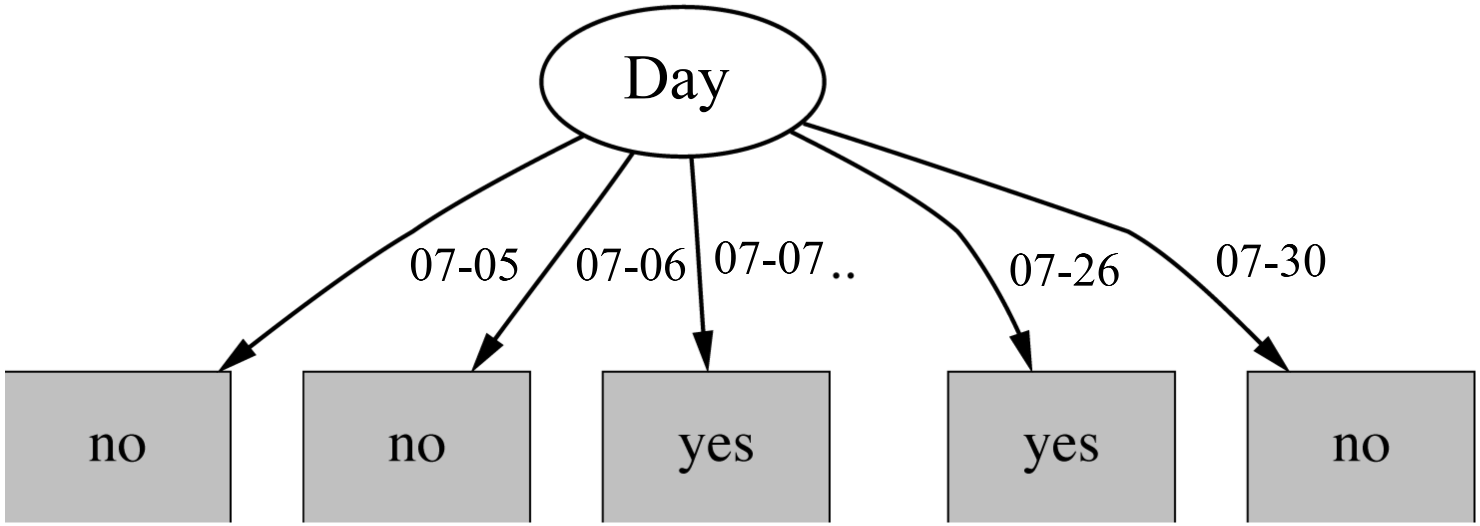
Final decision tree



Highly-branching attributes

- Problematic: attributes with a large number of values
  - extreme case: each example has its own value
    - e.g. example ID; Day attribute in weather data
- Subsets are more likely to be pure if there is a large number of different attribute values
  - Information gain is biased towards choosing attributes with a large number of values
- This may cause several problems:
  - *Overfitting*
    - selection of an attribute that is non-optimal for prediction
  - *Fragmentation*
    - data are fragmented into (too) many small sets

Decision Tree for Day attribute



- Entropy of split:

$$I(\text{Day}) = \frac{1}{14}\left(E([0,1]) + E([0,1]) + \ldots + E([0,1])\right) = 0$$

  - Information gain is maximal for Day (0.940 bits)

Alternative Measures

- Gain ratio: penalize attributes like income by incorporating split information
  - $SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$
    - Split information is sensitive to how broadly and uniformly the attribute splits the data
  - $GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$
- Gain ratio can be undefined or very large
  - Only test attributes with above average Gain

Gain ratios for weather data

Outlook		Temperature	
Info:	0.693	Info:	0.911
Gain: 0.940-0.693	0.247	Gain: 0.940-0.911	0.029
Split info: info([5,4,5])	1.577	Split info: info([4,6,4])	1.557
Gain ratio: 0.247/1.577	0.157	Gain ratio: 0.029/1.557	0.019
Humidity		Windy	
Info:	0.788	Info:	0.892
Gain: 0.940-0.788	0.152	Gain: 0.940-0.892	0.048
Split info: info([7,7])	1.000	Split info: info([8,6])	0.985
Gain ratio: 0.152/1	0.152	Gain ratio: 0.048/0.985	0.049

- Day attribute would still win...
  - one has to be careful which attributes to add...
- Nevertheless: Gain ratio is more reliable than Information Gain

Gini Index

- Many alternative measures to Information Gain
- Most popular altermative: Gini index
  - used in e.g., in CART (Classification And Regression Trees)
  - impurity measure (instead of entropy)

$$Gini(S) = 1 - \sum_i p_i^2$$

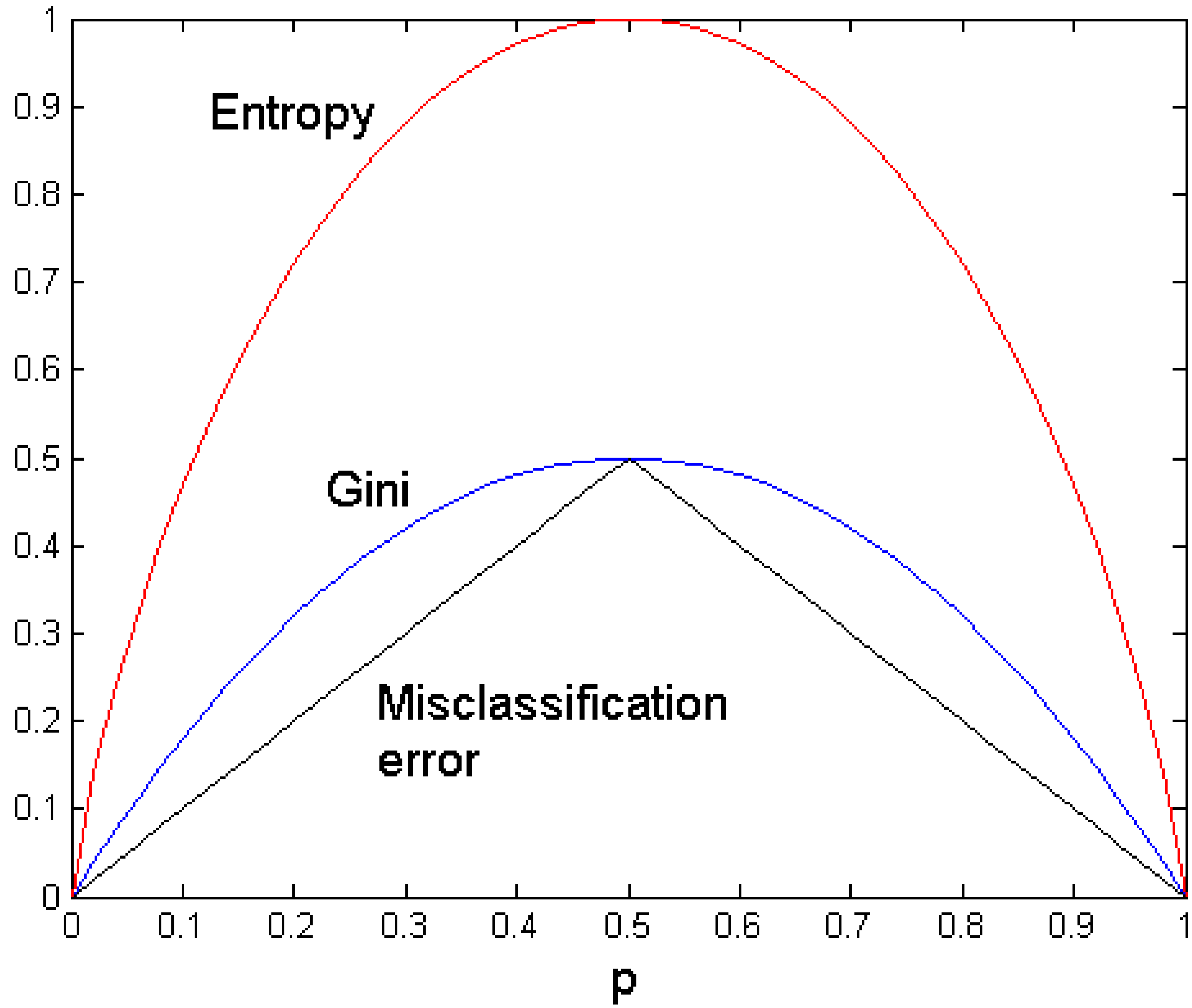
  - average Gini index (instead of average entropy / information)

$$Gini(S, A) = \sum_i \frac{|S_i|}{|S|} \cdot Gini(S_i)$$

  - Gini Gain
    - could be defined analogously to information gain
    - but typically avg. Gini index is minimized instead of maximizing Gini gain

Comparison among Splitting Criteria

For a 2-class problem:



ACKNOWLEDGMENT

- *Slides borrowed from Johannes Fürnkranz.*