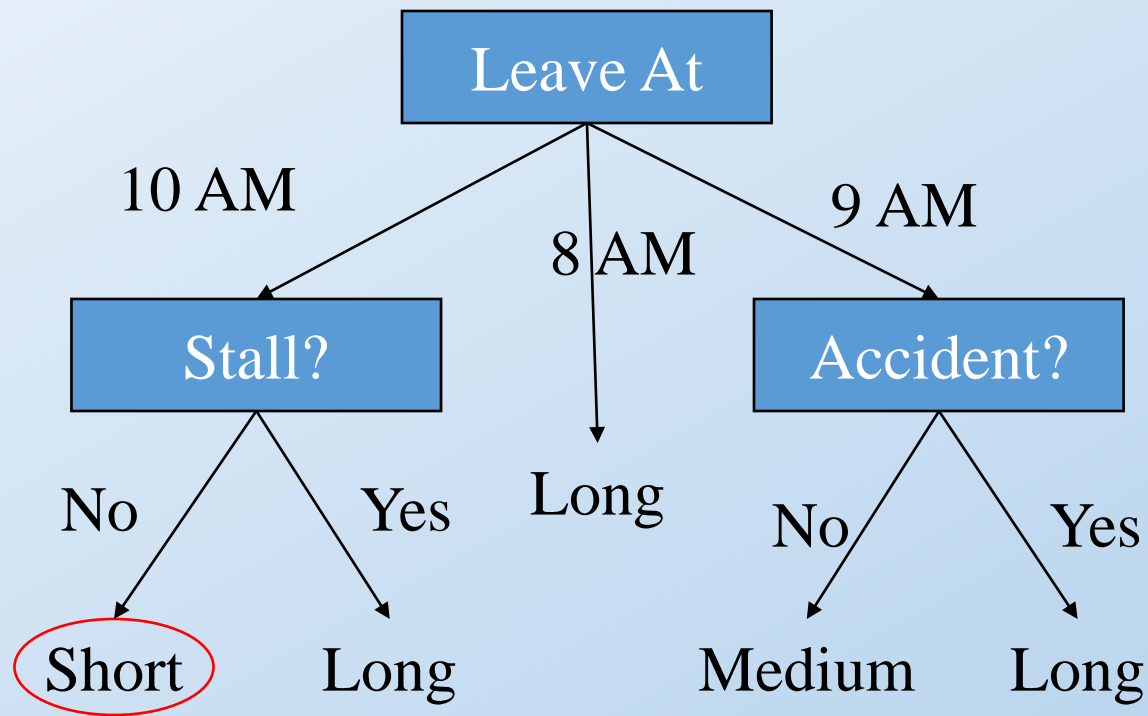


# Decision trees

# What is a Decision Tree?

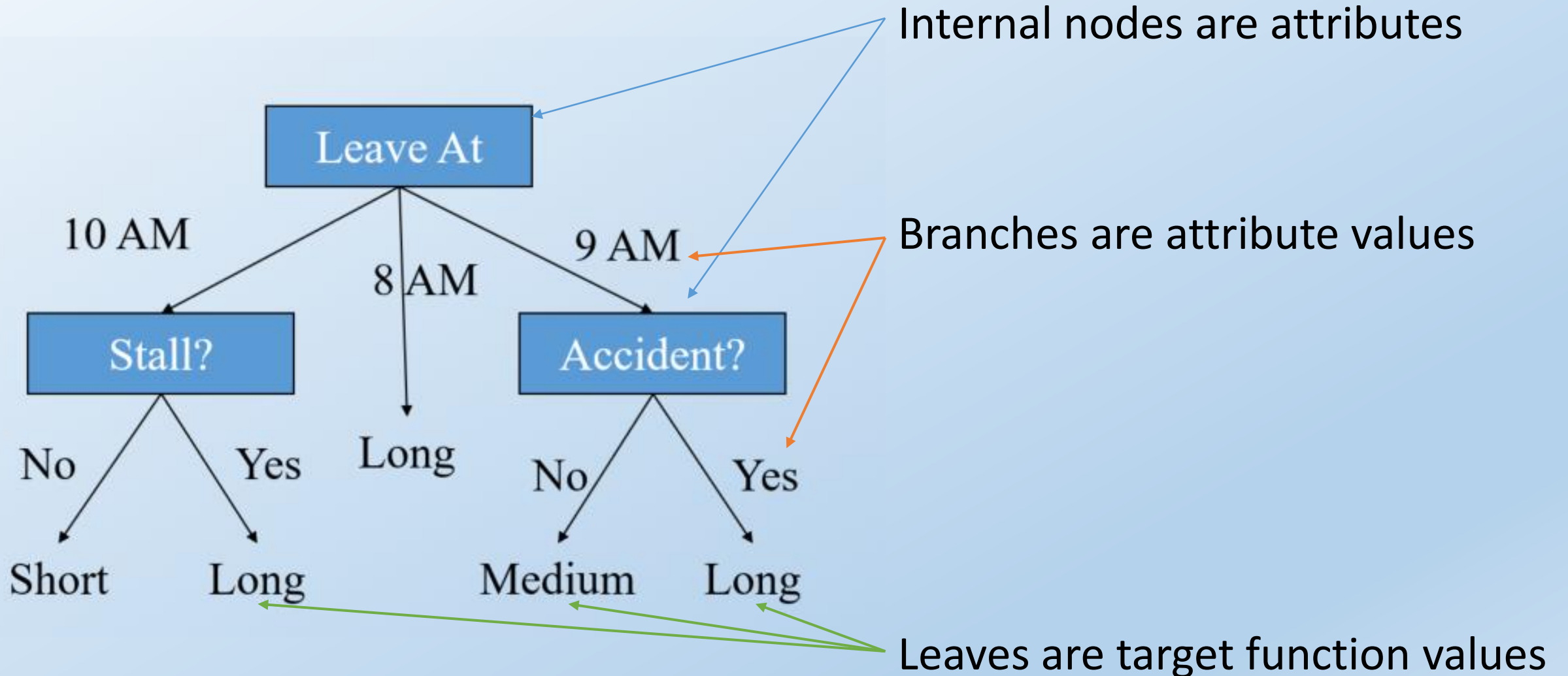
- An *inductive learning task*
  - Use particular facts to make more generalized conclusions
- A predictive model based on a branching series of Boolean tests
  - These smaller Boolean tests are less complex than a one-stage classifier
- Let's look at a sample decision tree...

# Predicting Commute Time



If we leave at 10 AM and there are no cars stalled on the road, what will our commute time be?

# Decision trees



# When to consider Decision Trees

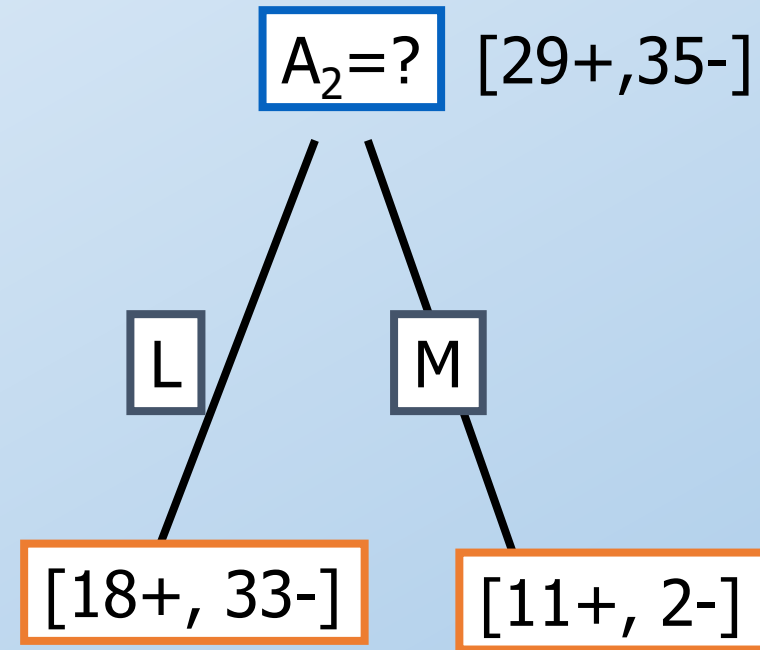
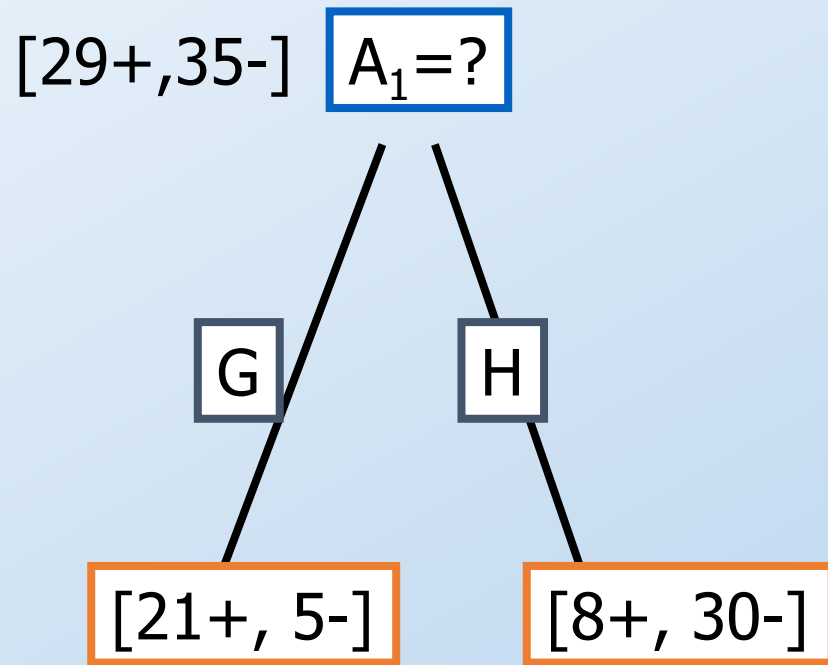
- Instances describable by attribute-value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data
- Missing attribute values
- Examples:
  - Medical diagnosis
  - Credit risk analysis
  - Object classification

# Top-Down Induction of Decision Trees - ID3

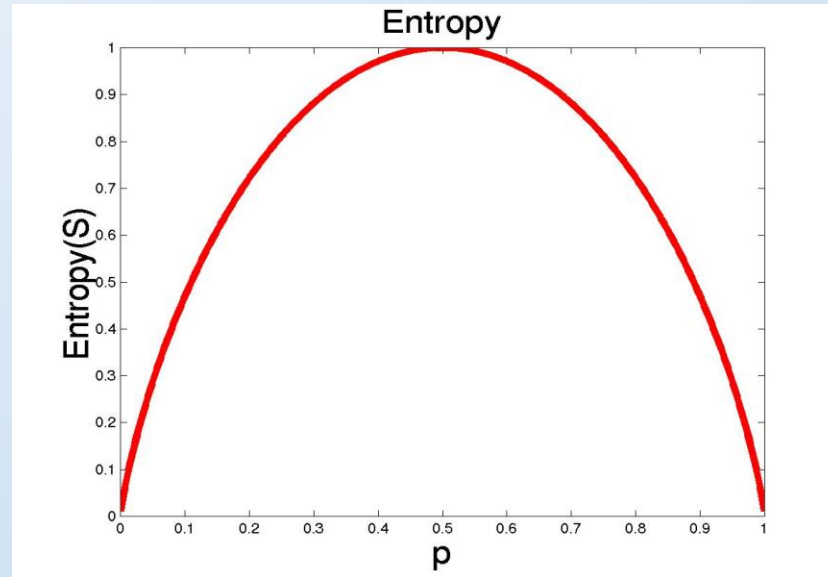
ID3( $X, y$ )

1. If  $X$  is empty, return leaf node with most common class in original training set as the class
2. If all values in  $y$  are equal, return leaf node with  $y[0]$  as the class
3. Let  $A$  be the **best** attribute in  $X$  to predict  $y$
4. Create a node **node** containing attribute  $A$
5. For each possible value  $v$  of  $A$ :
  - a. Let  $X_v$  be the subset of  $X$  for which  $A=v$
  - b. Let  $y_v$  be the target values of examples in  $X_v$
  - c. Remove  $A$  from  $X_v$
  - d.  $child = ID3(X_v, y_v)$
  - e. Create branch from **node** to **child** with value  $v$
6. Return **node**

# Which attribute is best?



# Entropy for binary classification



- $S$  is a sample of training examples
- $p_+$  is the proportion of positive examples
- $p_-$  is the proportion of negative examples
- Entropy measures the impurity of  $S$

$$\text{Entropy}(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$



# Entropy

- Entropy(S)= expected number of bits needed to encode class (+ or -) of randomly drawn members of S (under the optimal, shortest length-code)

Why?

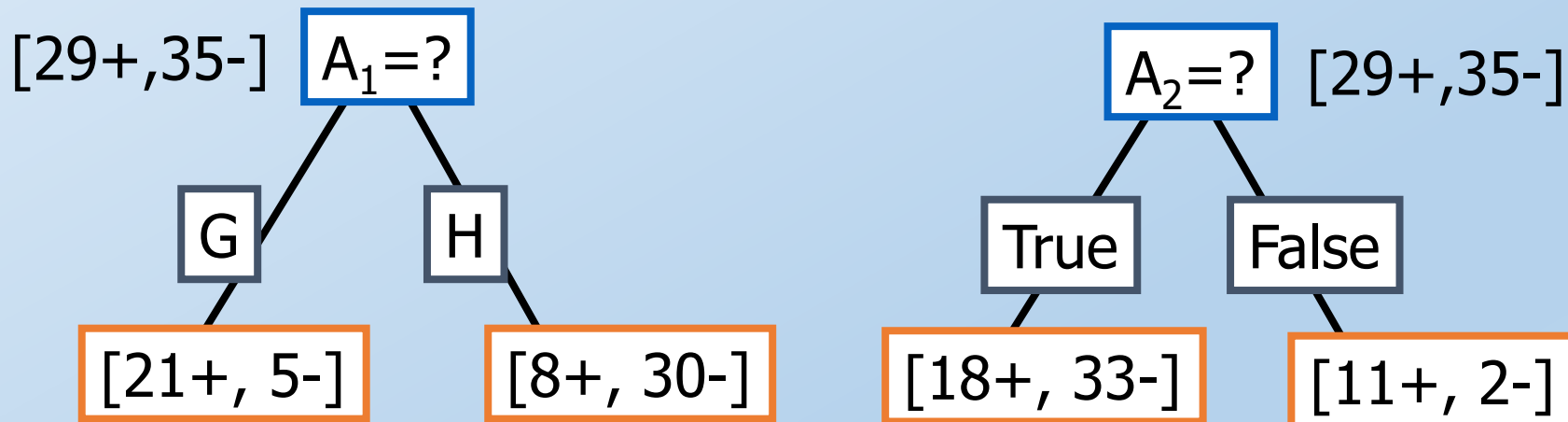
- Information theory optimal length code assign  $-\log_2 p$  bits to messages having probability  $p$ .
- So the expected number of bits to encode (+ or -) of random member of S:  
 $-p_+ \log_2 p_+ - p_- \log_2 p_-$

# Information Gain (S=E)

- Gain(S,A): expected reduction in entropy due to sorting S on attribute A

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

$$\begin{aligned} Entropy([29+, 35-]) &= -29/64 \log_2 29/64 - 35/64 \log_2 35/64 \\ &= 0.99 \end{aligned}$$

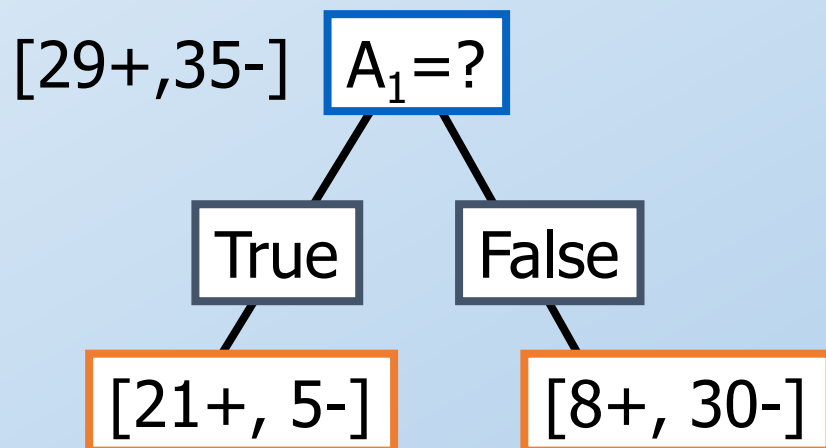


# Information Gain

$$\text{Entropy}([21+, 5-]) = 0.71$$

$$\text{Entropy}([8+, 30-]) = 0.74$$

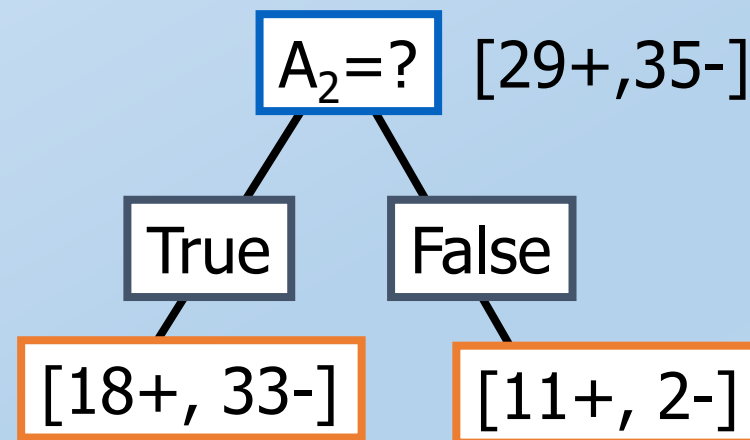
$$\begin{aligned}\text{Gain}(S, A_1) &= \text{Entropy}(S) \\ &\quad - 26/64 * \text{Entropy}([21+, 5-]) \\ &\quad - 38/64 * \text{Entropy}([8+, 30-]) \\ &= 0.27\end{aligned}$$



$$\text{Entropy}([18+, 33-]) = 0.94$$

$$\text{Entropy}([11+, 2-]) = 0.62$$

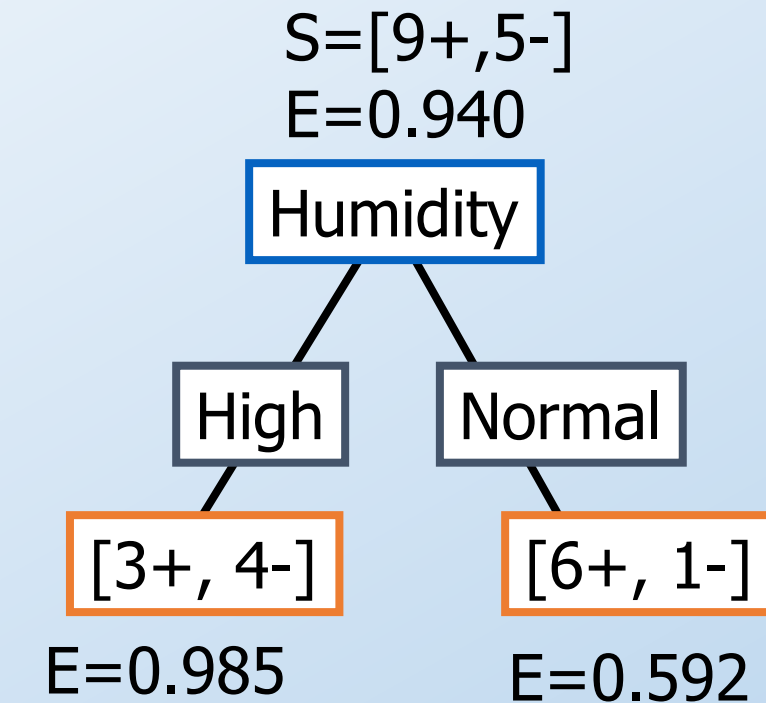
$$\begin{aligned}\text{Gain}(S, A_2) &= \text{Entropy}(S) \\ &\quad - 51/64 * \text{Entropy}([18+, 33-]) \\ &\quad - 13/64 * \text{Entropy}([11+, 2-]) \\ &= 0.12\end{aligned}$$



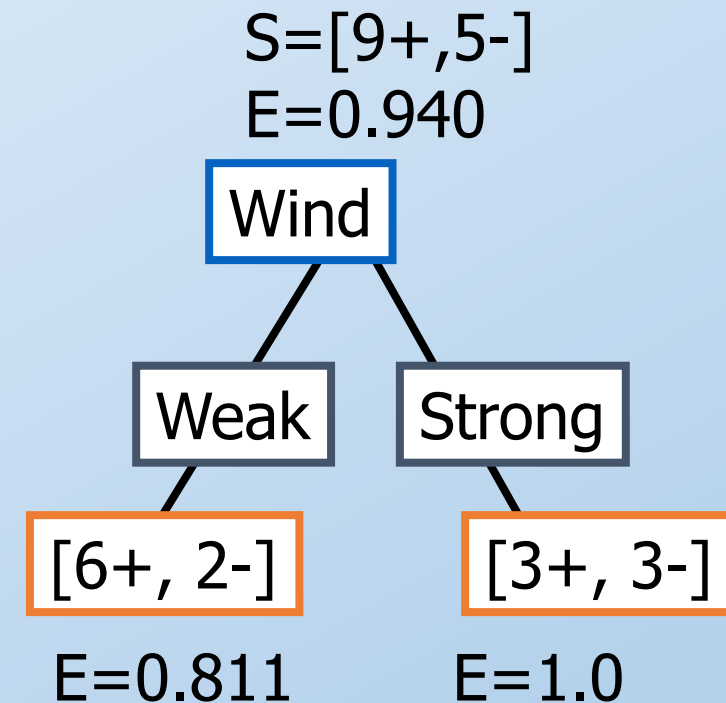
# Training Examples

| Day | Outlook  | Temp. | Humidity | Wind   | Play Tennis |
|-----|----------|-------|----------|--------|-------------|
| 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   | Weak   | Yes         |
| D8  | Sunny    | Mild  | High     | Weak   | No          |
| D9  | Sunny    | Cold  | Normal   | Weak   | Yes         |
| D10 | Rain     | Mild  | Normal   | Strong | 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



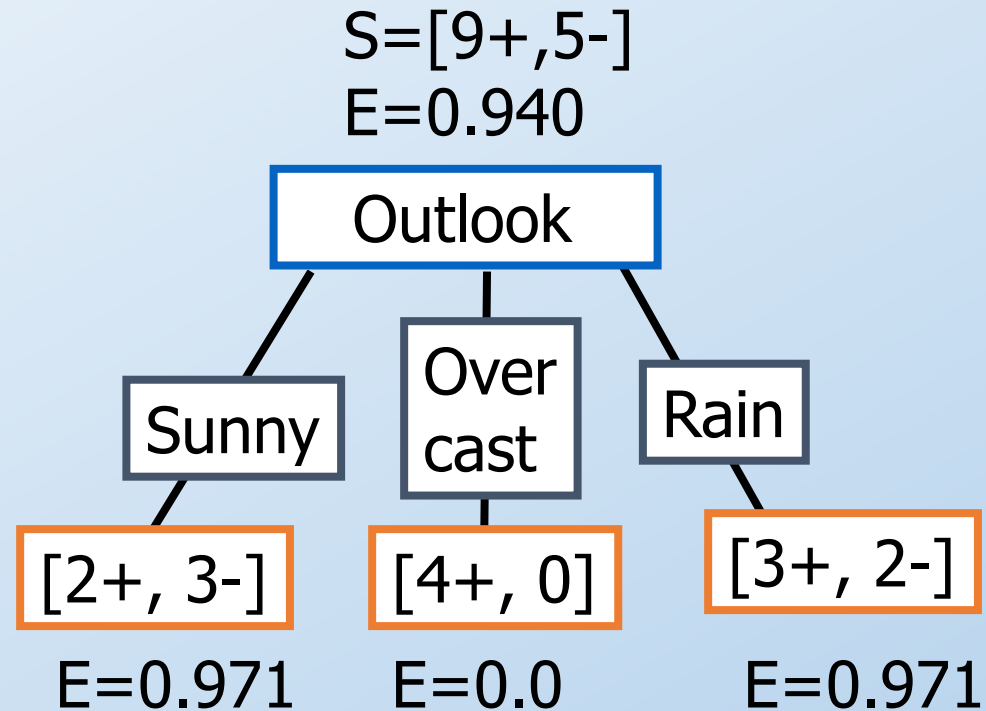
$$\begin{aligned}\text{Gain}(S, \text{Humidity}) &= 0.940 - (7/14) * 0.985 \\ &\quad - (7/14) * 0.592 \\ &= 0.151\end{aligned}$$



$$\begin{aligned}\text{Gain}(S, \text{Wind}) &= 0.940 - (8/14) * 0.811 \\ &\quad - (6/14) * 1.0 \\ &= 0.048\end{aligned}$$

Humidity provides greater info. gain than Wind, w.r.t target classification.

# Selecting the Next Attribute



$$\begin{aligned} \text{Gain}(S, \text{Outlook}) &= 0.940 - (5/14) * 0.971 \\ &\quad - (4/14) * 0.0 - (5/14) * 0.971 \\ &= 0.247 \end{aligned}$$

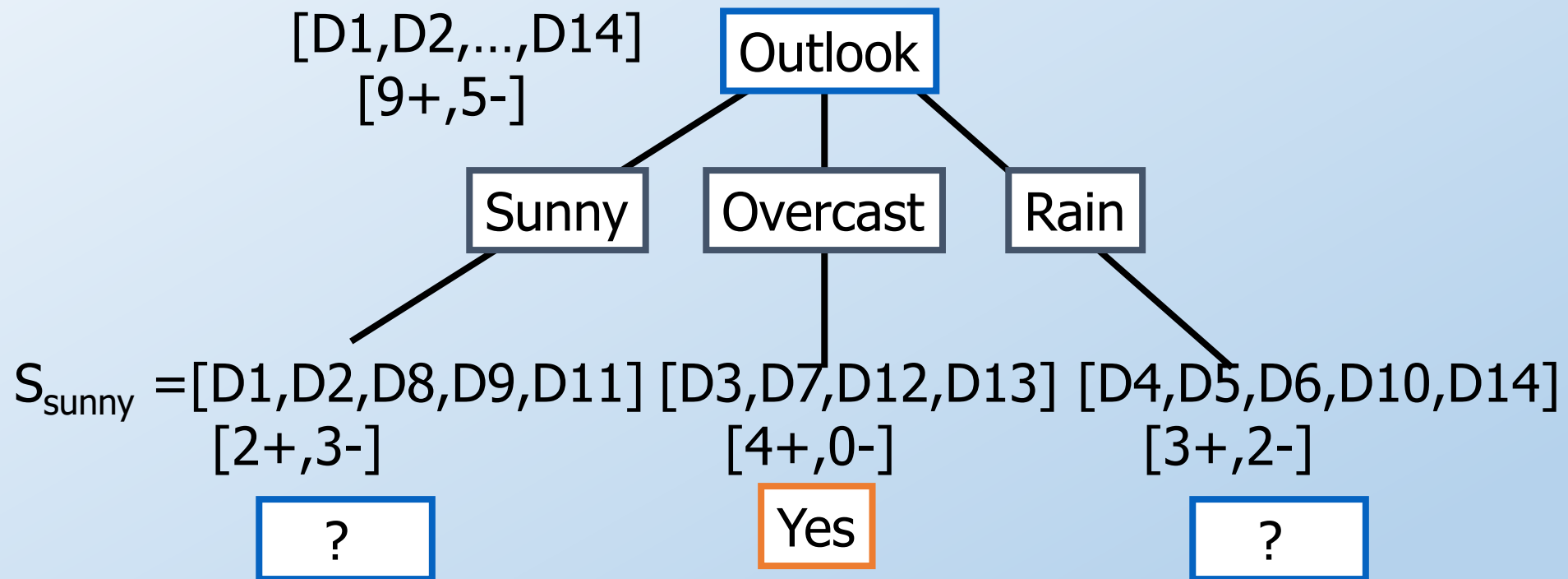
# Selecting the Next Attribute

The information gain values for the 4 attributes are:

- $\text{Gain}(S, \text{Outlook}) = 0.247$
- $\text{Gain}(S, \text{Humidity}) = 0.151$
- $\text{Gain}(S, \text{Wind}) = 0.048$
- $\text{Gain}(S, \text{Temperature}) = 0.029$

where  $S$  denotes the collection of training examples

# ID3 Algorithm



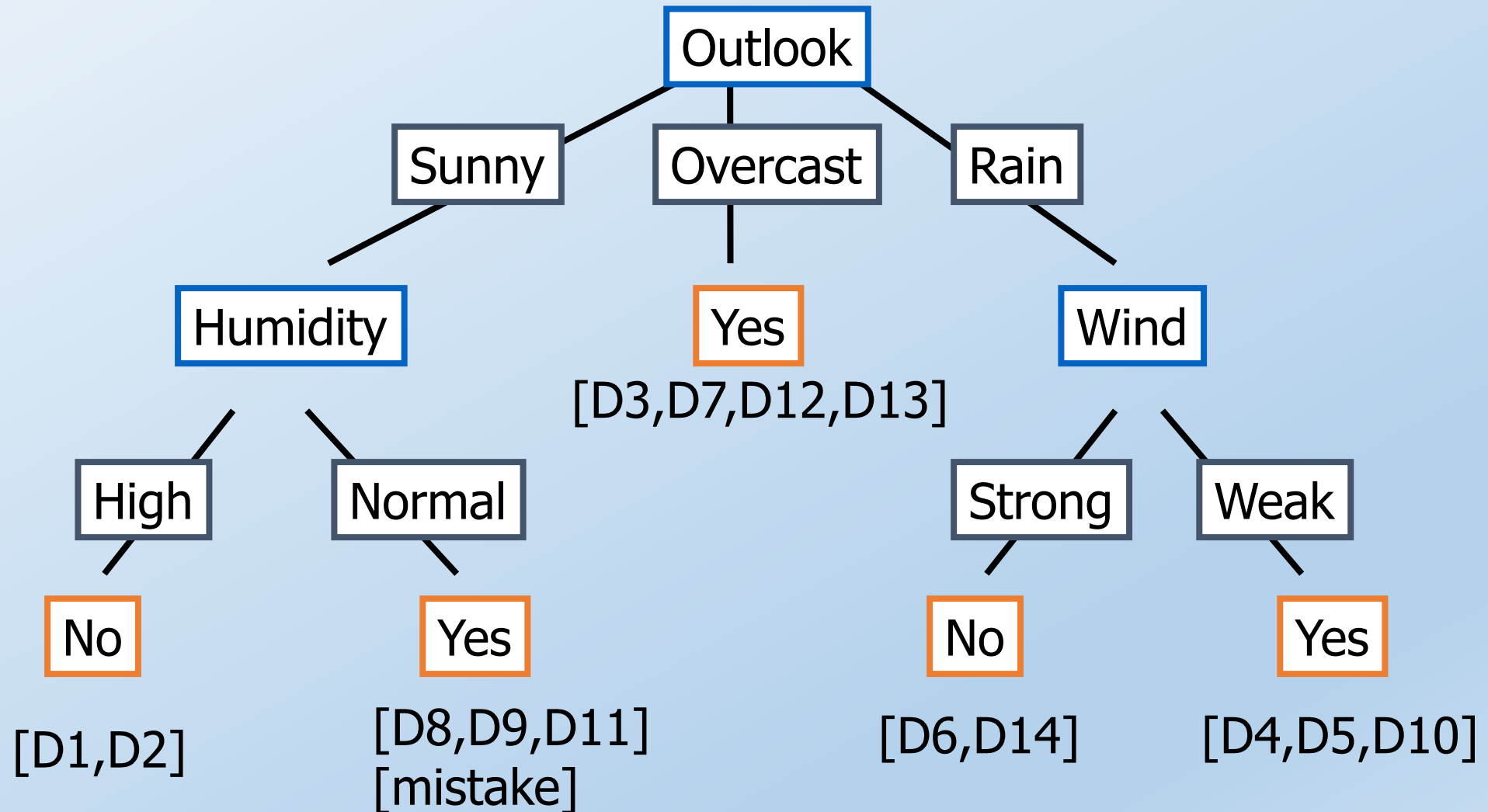
$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = 0.970 - (3/5)0.0 - 2/5(0.0) = 0.970$$

$$\text{Gain}(S_{\text{sunny}}, \text{Temp.}) = 0.970 - (2/5)0.0 - 2/5(1.0) - (1/5)0.0 = 0.570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = 0.970 - (2/5)1.0 - 3/5(0.918) = 0.019$$



# ID3 Algorithm



# Occam's Razor

"If two theories explain the facts equally well, then the simpler theory is to be preferred"

Arguments in favor:

- Fewer short hypotheses than long hypotheses
- A short hypothesis that fits the data is unlikely to be a coincidence
- A long hypothesis that fits the data might be a coincidence

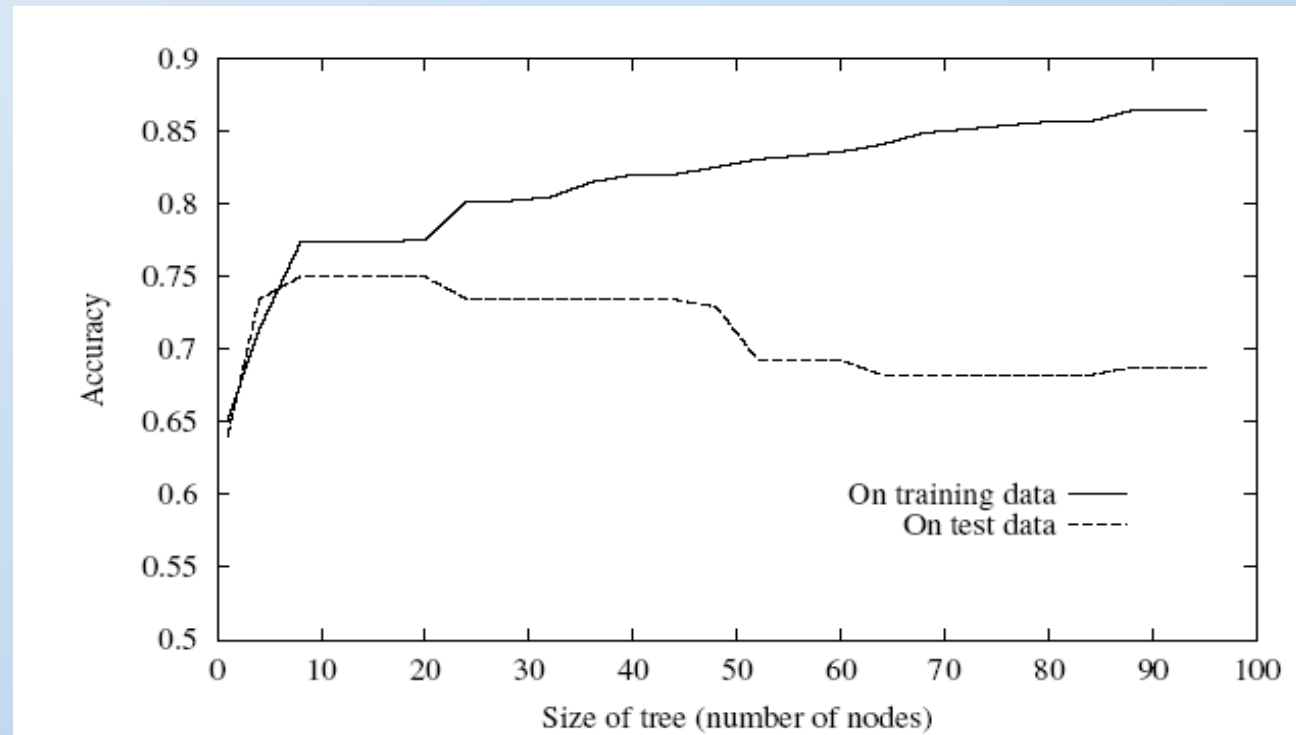
Arguments opposed:

- There are many ways to define small sets of hypotheses

# Overfitting

One of the biggest problems with decision trees is **Overfitting**

- You can build a decision tree to classify training data perfectly, but it will usually perform poorly on test data
- It takes  $\log(n)$  binary decisions to split a training set of size  $n$  such that each leaf classifies exactly one example – a binary tree of height 20 has over 1,000,000 leaves



# Avoid Overfitting

- Stop growing when most examples associated to the node belong to the same class
- Limit maximum depth
- Limit number of nodes
- Grow full tree, then post-prune

# Dealing with real-valued attributes

- For every real-valued attribute  $A(x)$ 
  - Generate a binary attribute  $A_t(x)$  which is true if  $x \geq t$  for a threshold  $t$
  - Choose threshold  $t$  that maximizes information gain

# Dealing with real-valued attributes

- For every real-valued attribute  $A(x)$ 
  - Generate a binary attribute  $A_t(x)$  which is true if  $x \geq t$  for a threshold  $t$
  - Choose threshold  $t$  that maximizes information gain

|             |      |      |      |      |      |      |
|-------------|------|------|------|------|------|------|
| Temperature | 15°C | 18°C | 19°C | 22°C | 24°C | 27°C |
| PlayTennis  | No   | No   | Yes  | Yes  | Yes  | No   |

# Dealing with real-valued attributes

- For every real-valued attribute  $A(x)$ 
  - Generate a binary attribute  $A_t(x)$  which is true if  $x \geq t$  for a threshold  $t$
  - Choose threshold  $t$  that maximizes information gain

|             |      |      |      |      |      |      |
|-------------|------|------|------|------|------|------|
| Temperature | 15°C | 18°C | 19°C | 22°C | 24°C | 27°C |
| PlayTennis  | No   | No   | Yes  | Yes  | Yes  | No   |

← Sort by value

# Dealing with real-valued attributes

- For every real-valued attribute  $A(x)$ 
  - Generate a binary attribute  $A_t(x)$  which is true if  $x \geq t$  for a threshold  $t$
  - Choose threshold  $t$  that maximizes information gain

|             |      |      |      |      |      |      |
|-------------|------|------|------|------|------|------|
| Temperature | 15°C | 18°C | 19°C | 22°C | 24°C | 27°C |
| PlayTennis  | No   | No   | Yes  | Yes  | Yes  | No   |

← Sort by value

Find neighbors that belong to different classes:

Candidate  
 $t = (18+19)/2$   
 $=18.5$

Candidate  
 $t = (24+27)/2$   
 $=25.5$



# Dealing with real-valued attributes

$$\begin{aligned}\text{InfoGain}(\text{Temp}_{18.5}) &= \text{entropy}(3+,3-) - \frac{2}{6} \text{entropy}(0+,2-) - \frac{4}{6} \text{entropy}(3+,1-) \\ &= 1 - \frac{2}{6} (0) - \frac{4}{6} (0.8113) = 0.4591\end{aligned}$$

$$\begin{aligned}\text{InfoGain}(\text{Temp}_{25.5}) &= \text{entropy}(3+,3-) - \frac{1}{6} \text{entropy}(0+,1-) - \frac{5}{6} \text{entropy}(3+,2-) \\ &= 1 - \frac{1}{6} (0) - \frac{5}{6} (0.971) = 0.1909\end{aligned}$$


| Temperature | 15°C | 18°C | 19°C | 22°C | 24°C | 27°C |
|-------------|------|------|------|------|------|------|
| PlayTennis  | No   | No   | Yes  | Yes  | Yes  | No   |

↑  
Candidate  
 $t = (18+19)/2 = 18.5$

↑  
Candidate  
 $t = (24+27)/2 = 25.5$

# Dealing with real-valued attributes

$$\begin{aligned}\text{InfoGain}(\text{Temp}_{18.5}) &= \text{entropy}(3+,3-) - \frac{2}{6} \text{entropy}(0+,2-) - \frac{4}{6} \text{entropy}(3+,1-) \\ &= 1 - \frac{2}{6} (0) - \frac{4}{6} (0.8113) = 0.4591\end{aligned}$$

 **Best split**

$$\begin{aligned}\text{InfoGain}(\text{Temp}_{25.5}) &= \text{entropy}(3+,3-) - \frac{1}{6} \text{entropy}(0+,1-) - \frac{5}{6} \text{entropy}(3+,2-) \\ &= 1 - \frac{1}{6} (0) - \frac{5}{6} (0.971) = 0.1909\end{aligned}$$

|             |      |      |      |      |      |      |
|-------------|------|------|------|------|------|------|
| Temperature | 15°C | 18°C | 19°C | 22°C | 24°C | 27°C |
| PlayTennis  | No   | No   | Yes  | Yes  | Yes  | No   |

↑  
Candidate  
 $t = (18+19)/2 = 18.5$

↑  
Candidate  
 $t = (24+27)/2 = 25.5$

# Dealing with real-valued target function

Called regression trees

Only change to ID3:

- Selection criterion for best attribute

- Choose the attribute that results in the largest MSE reduction  
assuming the prediction in each node is  $\text{mean}(y)$

Leaves are real values

# Top-Down Induction of Regression Trees - ID3

ID3( $X, y$ )

1. If  $X$  is empty, return leaf node with mean target value in original training set as the label
2. If all values in  $y$  are equal, return leaf node with  $y[0]$  as the class
3. Let  $A_t$  be the **best** attribute in  $X$  to predict  $y$
4. Create a node **node** containing attribute  $A$  and threshold  $t$
5. For  $v$  in {false, true}:
  - a. Let  $X_v$  be the subset of  $X$  for which  $A > t = v$
  - b. Let  $y_v$  be the target values of examples in  $X_v$
  - c.  $child = ID3(X_v, y_v)$
  - d. Create branch from **node** to **child** with value  $v$
6. Return **node**