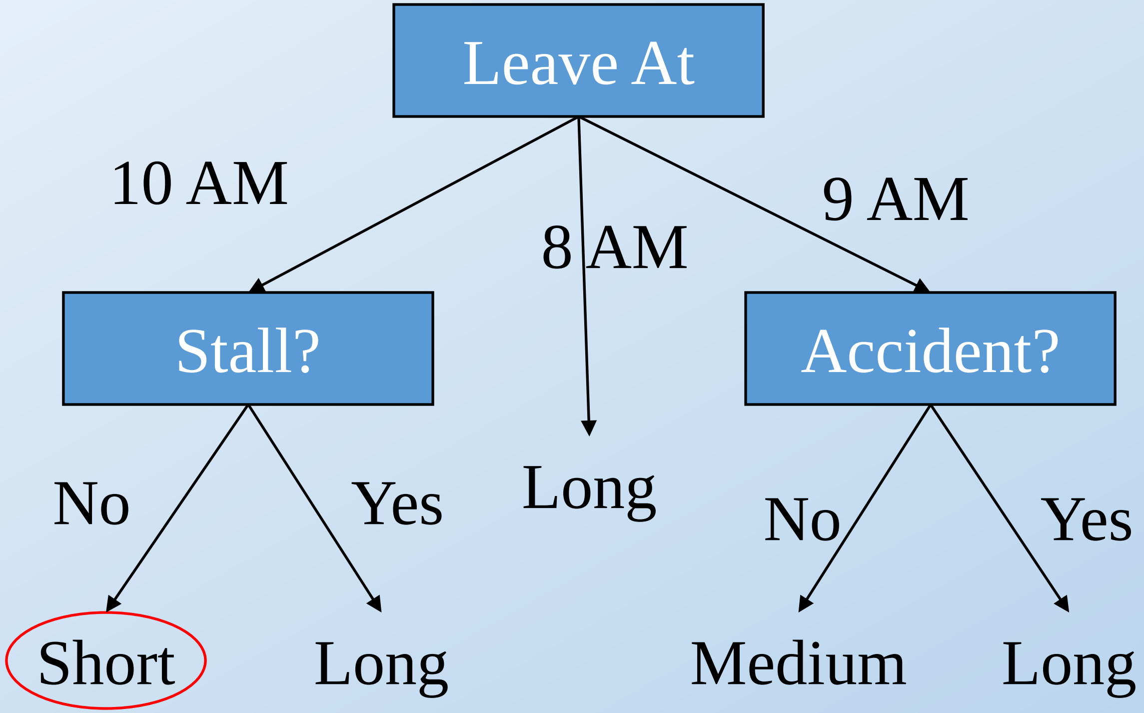
Decision and Regression Trees

What is a Decision Tree?

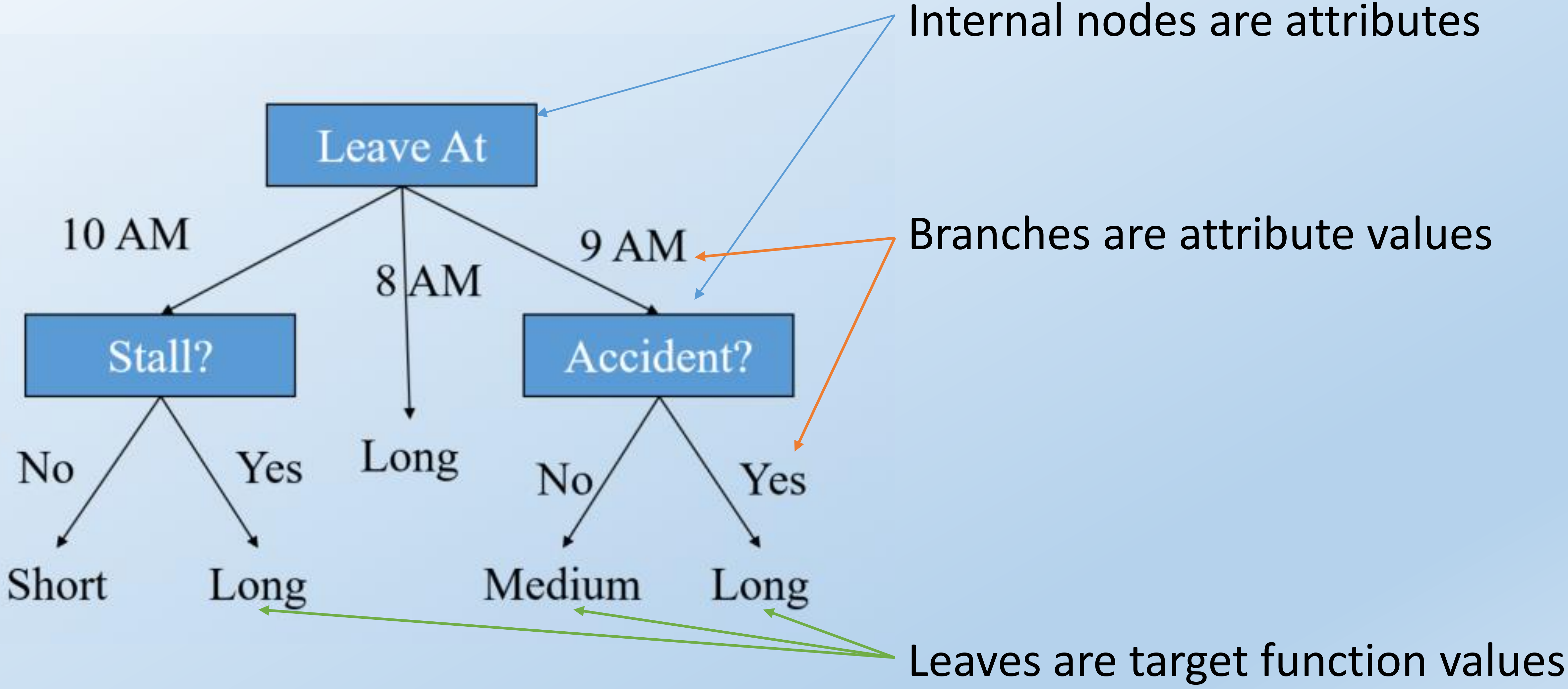
- An *inductive learning task*
  - Use particular facts to make more generalized conclusions
- A predictive model based on a branching series of Boolean tests
  - These smaller Boolean tests are less complex than a one-stage classifier
- Let’s look at a sample decision tree…

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

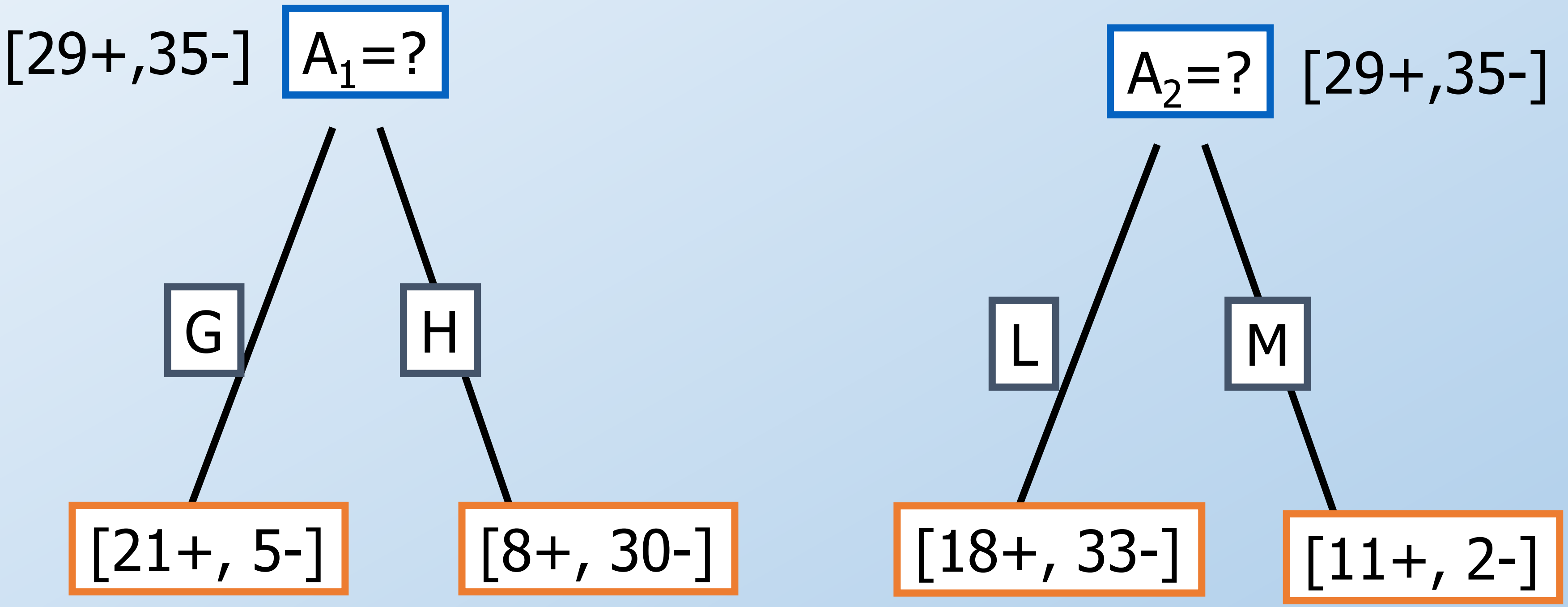
- Instances describable by attribute-value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data
- Missing attribute values
- Examples:
  - Medical diagnosis
  - Credit risk analysis
  - Object classification

Top-Down Induction of Decision Trees - ID3

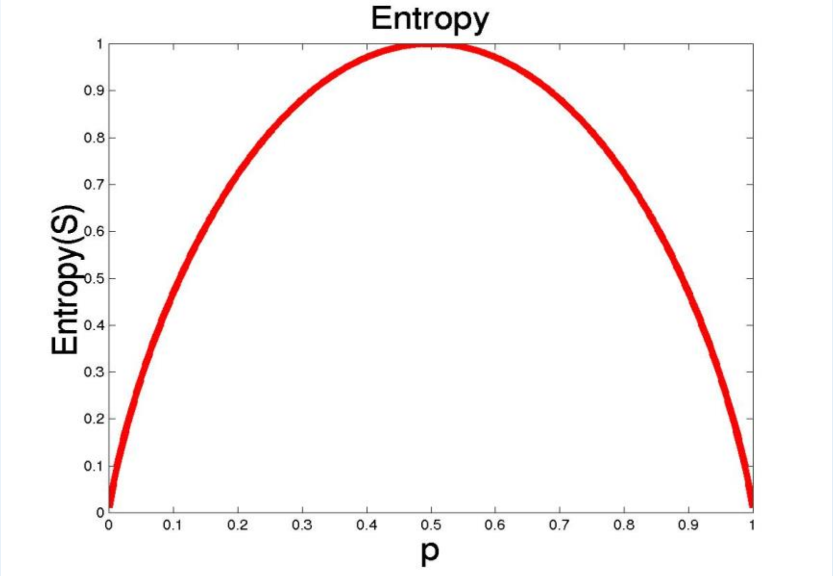
ID3(X,y)

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 $\mathbf{X_v}$ be the subset of X for which **A**=**v**
   b. Let $\mathbf{y_v}$ be the target values of examples in $\mathbf{X_v}$
   c. Remove **A** from $\mathbf{X_v}$
   d. child = ID3($\mathbf{X_v}$,$\mathbf{y_v}$)
   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

  $Entropy(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$

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:

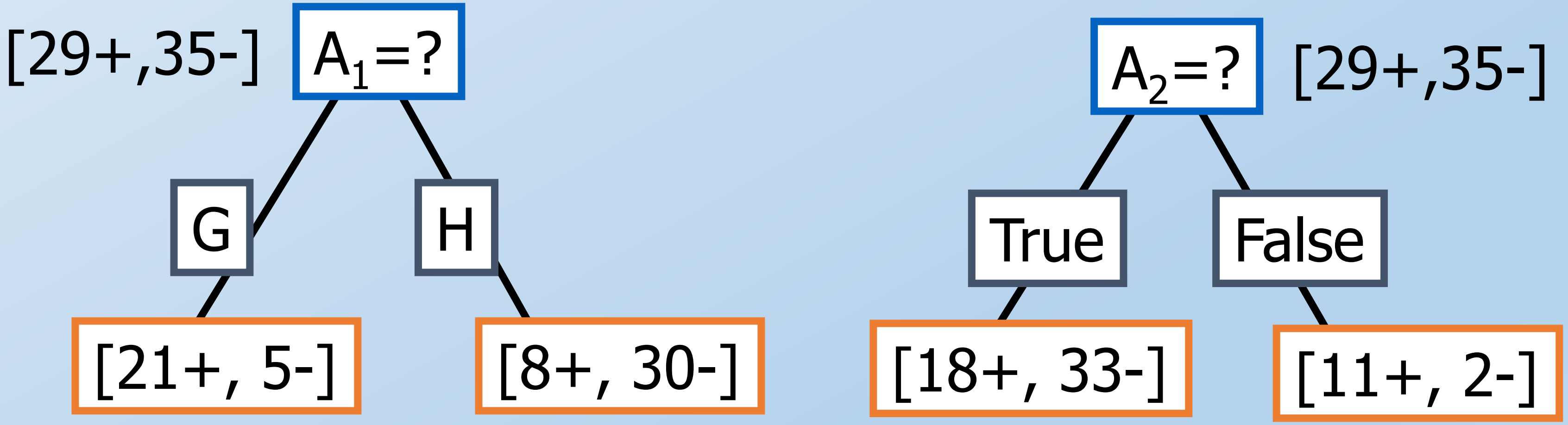
$$-p_+ \log_2 p_+ - p_- \log_2 p_-$$

Information Gain (S=E)

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

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

Entropy([29+,35-]) = -29/64 log2 29/64 – 35/64 log2 35/64
= 0.99



Information Gain

Entropy([21+,5-]) = 0.71

Entropy([8+,30-]) = 0.74

$\mathrm{Gain}(S,A_1)$=Entropy(S)

-26/64*Entropy([21+,5-])

-38/64*Entropy([8+,30-])

=0.27

Entropy([18+,33-]) = 0.94

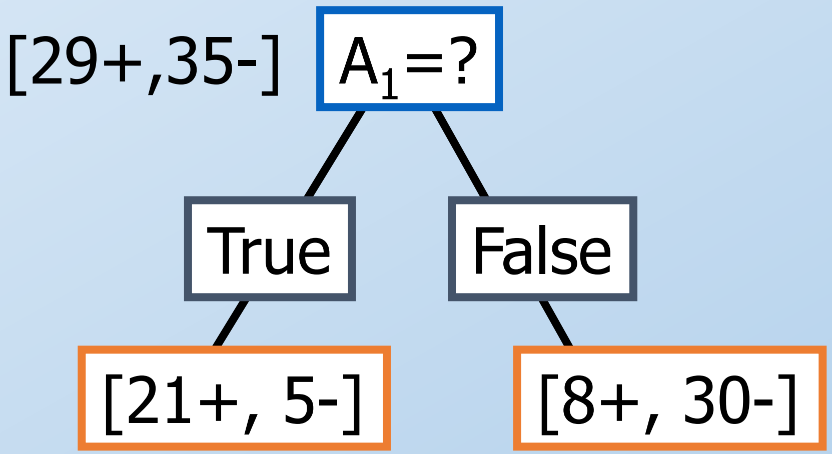
Entropy([11+,2-]) = 0.62

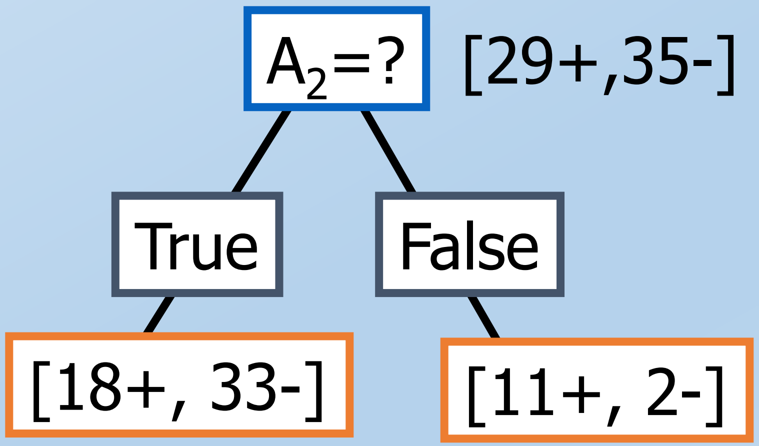
Gain(S,A2)=Entropy(S)

-51/64*Entropy([18+,33-])

-13/64*Entropy([11+,2-])

=0.12

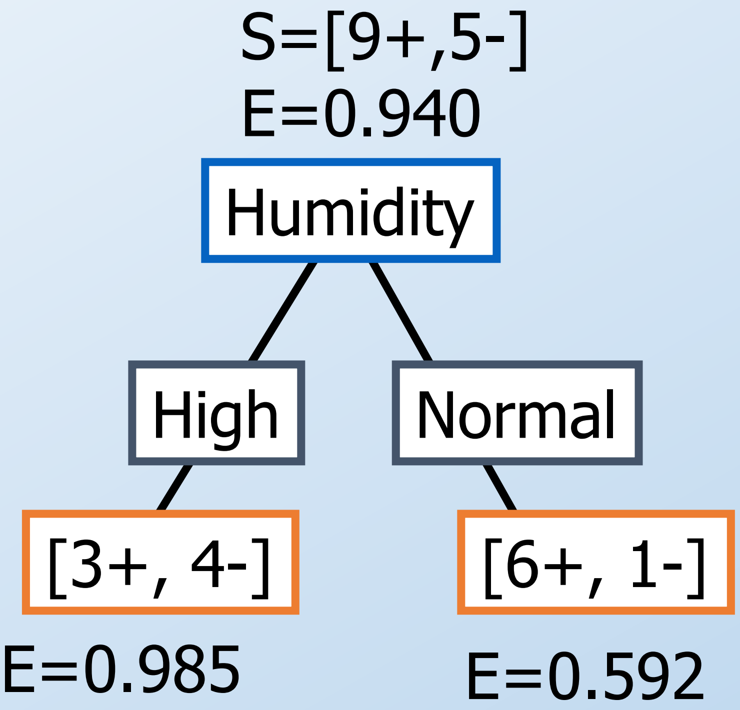


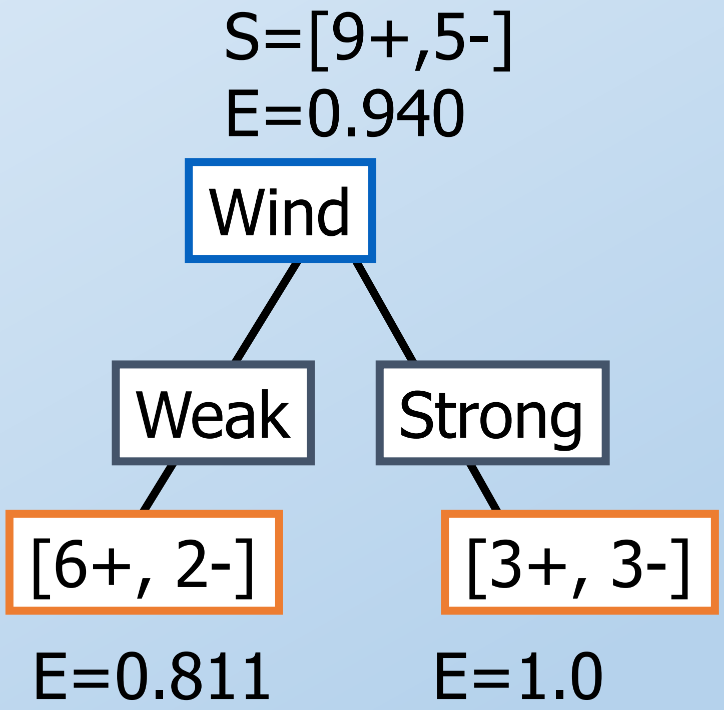


Training Examples

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



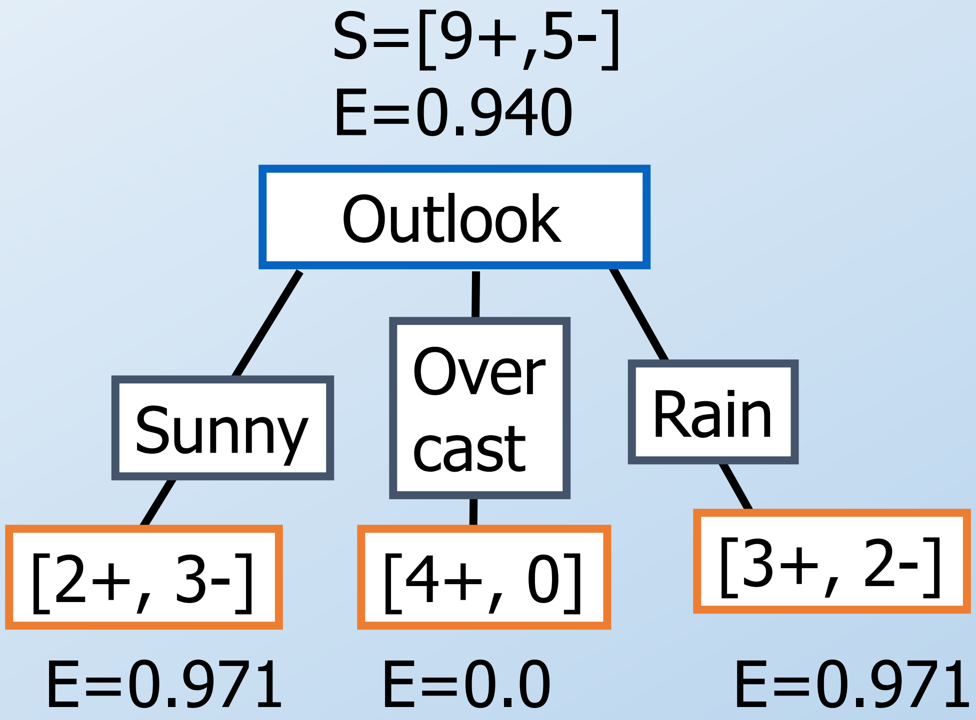


Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

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

Selecting the Next Attribute



Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

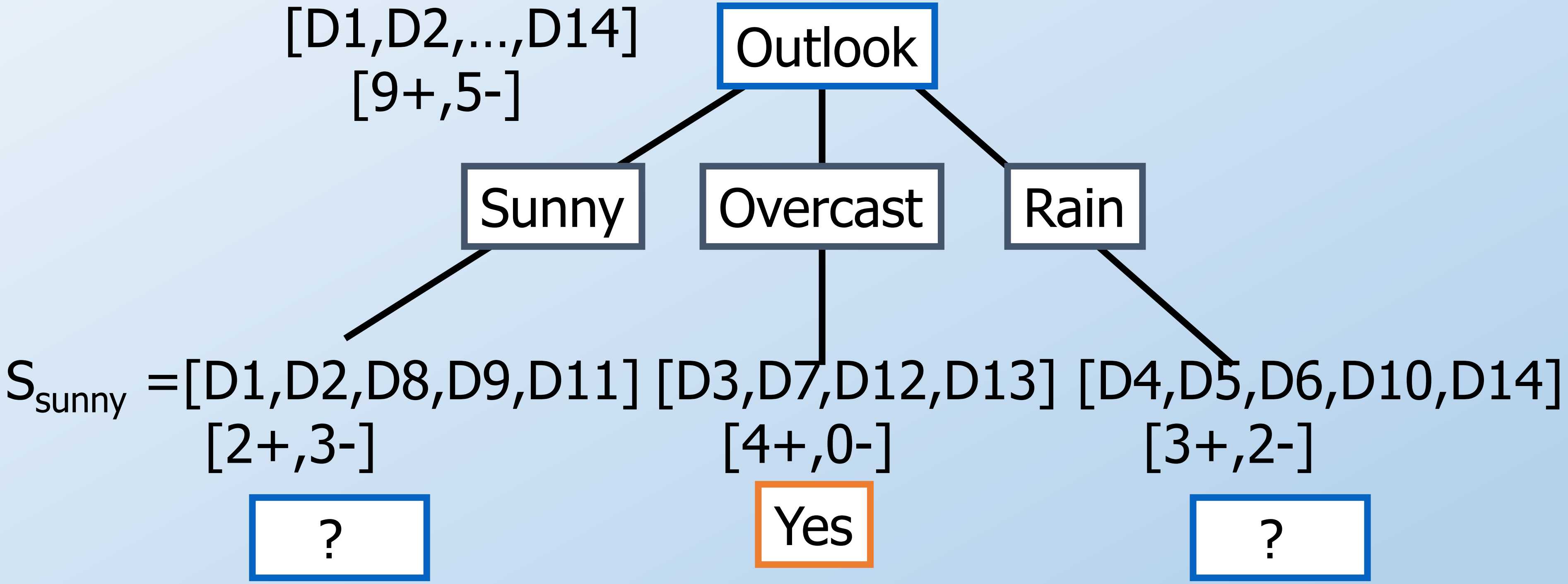
Selecting the Next Attribute

The information gain values for the 4 attributes are:

- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

where S denotes the collection of training examples

ID3 Algorithm

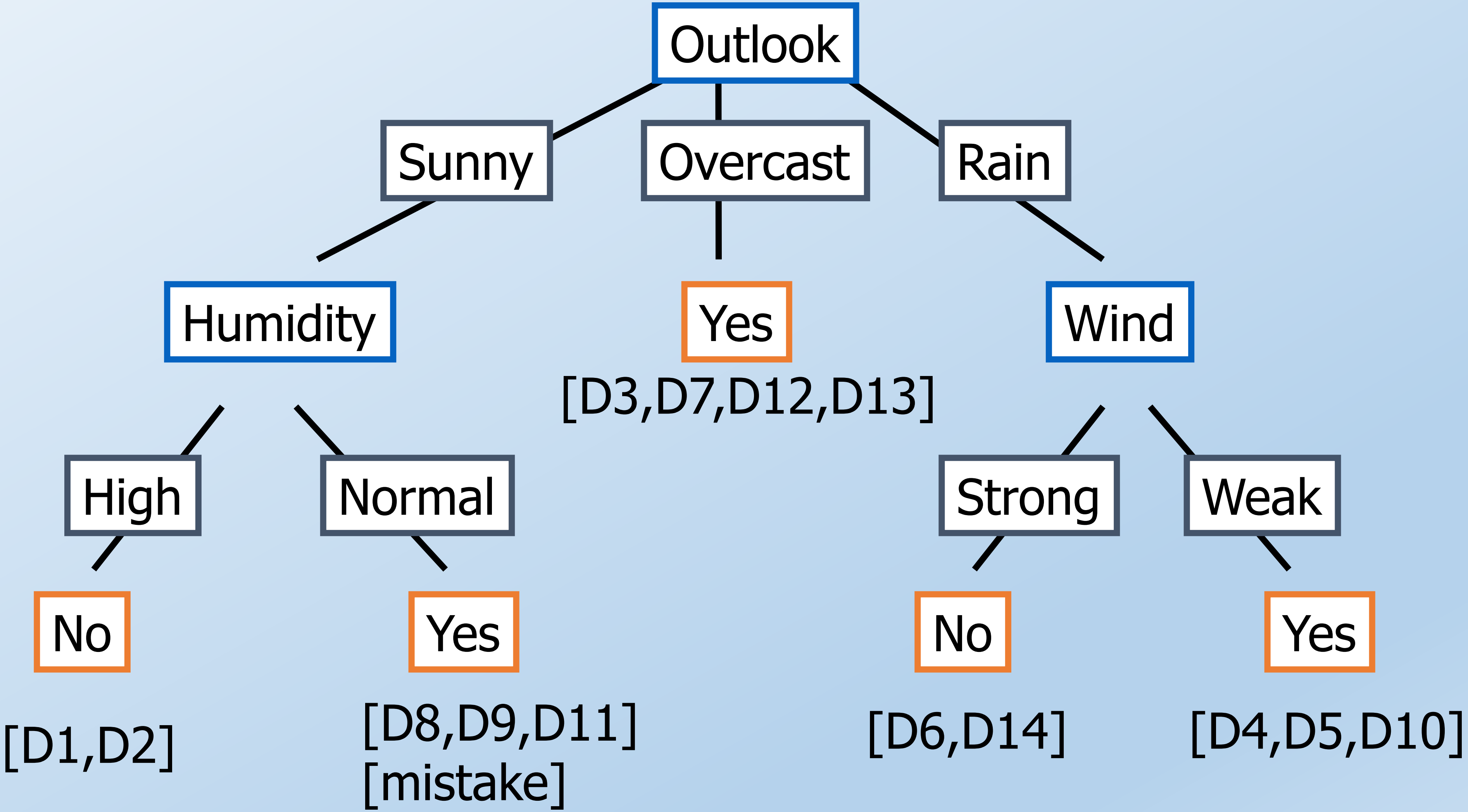


Gain(S_{sunny}, Humidity)=0.970-(3/5)0.0 – 2/5(0.0) = 0.970

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

Gain(S_{sunny}, 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 weel, 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

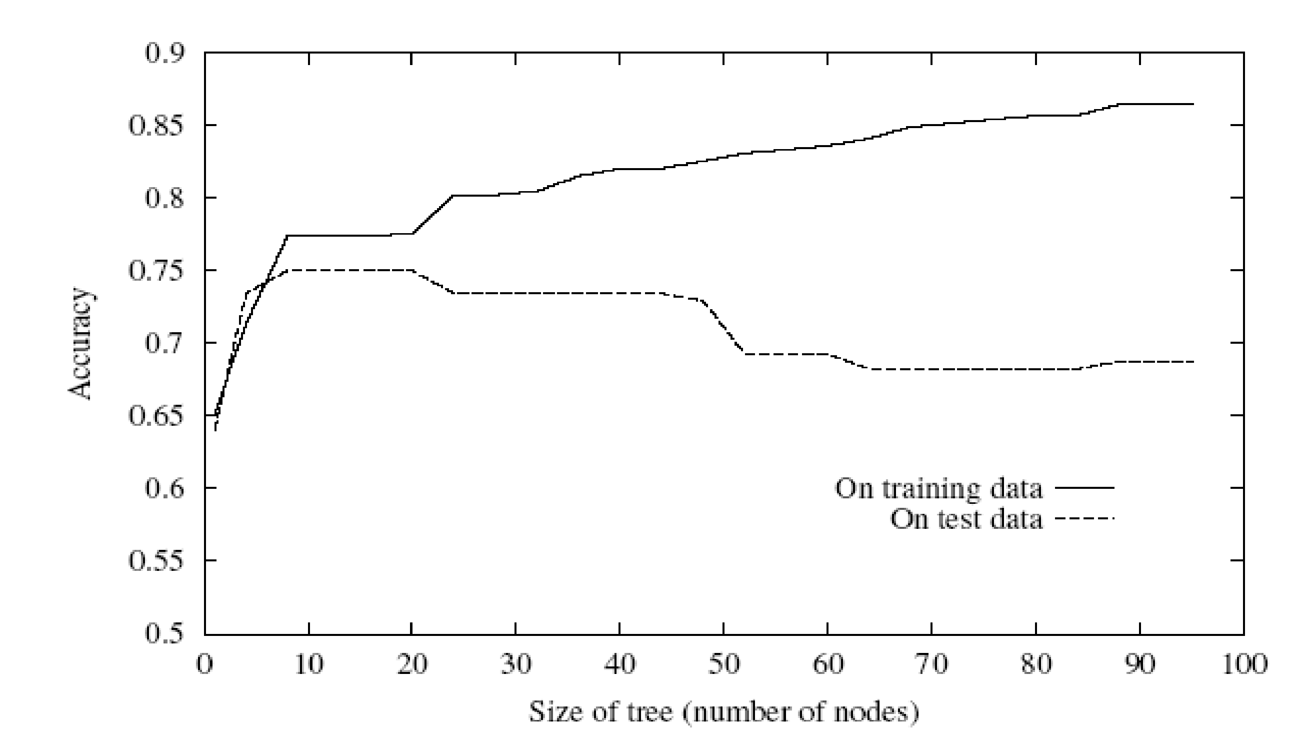
Arguments opposed:

- 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 attibute **A(x)**
  - Generate a binary attribute $\mathbf{A_t(x)}$ which is true if $\mathbf{x>=t}$ for a threshold **t**
  - Choose threshold **t** that maximizes information gain

Dealing with real-valued attributes

- For every real-valued attibute **A(x)**
  - Generate a binary attribute $\mathbf{A_t(x)}$ which is true if **x>=t** for a threshold **t**
  - Choose threshold **t** that maximizes information gain

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
PlayTennis	No	No	Yes	Yes	Yes	No

Dealing with real-valued attributes

- For every real-valued attibute **A(x)**
  - Generate a binary attribute $\mathbf{A_t(x)}$ which is true if **x>=t** for a threshold **t**
  - Choose threshold **t** that maximizes information gain

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
PlayTennis	No	No	Yes	Yes	Yes	No

← **Sort by value**

Dealing with real-valued attributes

- For every real-valued attibute **A(x)**
  - Generate a binary attribute $\mathbf{A_t(x)}$ which is true if $\mathbf{x>=t}$ for a threshold $\mathbf{t}$
  - Choose threshold **t** that maximizes information gain

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
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

$$\text{InfoGain}(\text{Temp}_{18.5}) = \text{entropy}(3+,3-) - \frac{2}{6}\text{entropy}(0+,2-) - \frac{4}{6}\text{entropy}(3+,1-)$$

$$= 1 - \frac{2}{6}(0) - \frac{4}{6}(0.8113) = 0.4591$$

$$\text{InfoGain}(\text{Temp}_{25.5}) = \text{entropy}(3+,3-) - \frac{1}{6}\text{entropy}(0+,1-) - \frac{5}{6}\text{entropy}(3+,2-)$$

$$= 1 - \frac{1}{6}(0) - \frac{5}{6}(0.971) = 0.1909$$

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
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

InfoGain($\mathbf{Temp_{18.5}}$) = entropy(3+,3-) - $\frac{2}{6}$ entropy(0+,2-) - $\frac{4}{6}$ entropy(3+,1-)

= 1 - $\frac{2}{6}$ (0) - $\frac{4}{6}$ (0.8113) = 0.4591 Best split

InfoGain($Temp_{25.5}$) = entropy(3+,3-) - $\frac{1}{6}$ entropy(0+,1-) - $\frac{5}{6}$ entropy(3+,2-)

= 1 - $\frac{1}{6}$ (0) - $\frac{5}{6}$ (0.971) = 0.1909

Temperature	15^0C	18^0C	19^0C	22^0C	24^0C	27^0C
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 mean(y)

Leaves are real values – the mean y of the examples assigned to the leaf during training

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 $\mathbf{A_t}$ 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 $\mathbf{X_v}$ be the subset of X for which **A > t = v**
   b. Let $\mathbf{y_v}$ be the target values of examples in $\mathbf{X_v}$
   c. child = ID3($\mathbf{X_v}$,$\mathbf{y_v}$)
   d. Create branch from **node** to **child** with value **v**
6. Return **node**

Regression Trees – Choosing the best attribute

Best_Attribute(X,y)
for i in range(X.shape[1]):
        find threshold $\mathbf{t_i}$ that results in minimum MSE for (X[:,i], y)
        Create binary attribute t_i < X[:,i] and store mse
Return **(i, $\mathbf{t_i}$)** that corresponds to minimum mse

Regression Trees – Choosing the best attribute

A	0	1	2	3	4	5	6	7	8	9	← Attribute
y	0.8	0.3	0.5	0.2	0.7	0.1	1.1	0.4	0.9	0.6	← Target function

Original data
Mean = 0.5600, Variance = 0.0924

Thresholds = [0.5 1.5 2.5 3.5 4.5 5.5 6.5 7.5 8.5]

Threshold= 0.5000
Left: [0.8] Mean = 0.8000, Variance = 0.0000
Right [0.3 0.5 0.2 0.7 0.1 1.1 0.4 0.9 0.6] Mean = 0.5333, Variance = 0.0956
mse = (1/10)*0.0000 + (9/10)*0.0956 = 0.0860

Threshold = 1.5000
Left: [0.8 0.3] Mean = 0.5500, Variance = 0.0625
Right [0.5 0.2 0.7 0.1 1.1 0.4 0.9 0.6] Mean = 0.5625, Variance = 0.0998
mse = (2/10)*0.0625 + (8/10)*0.0998 = 0.0924

Regression Trees – Choosing the best attribute

A	0	1	2	3	4	5	6	7	8	9
y	0.8	0.3	0.5	0.2	0.7	0.1	1.1	0.4	0.9	0.6

Threshold = 2.5000
Left: [0.8 0.3 0.5] Mean = 0.5333, Variance = 0.0422
Right [0.2 0.7 0.1 1.1 0.4 0.9 0.6] Mean = 0.5714, Variance = 0.1135
mse = (3/10)*0.0422 + (7/10)*0.1135 = 0.0921

Threshold = 3.5000
Left: [0.8 0.3 0.5 0.2] Mean = 0.4500, Variance = 0.0525
Right [0.7 0.1 1.1 0.4 0.9 0.6] Mean = 0.6333, Variance = 0.1056
mse = (4/10)*0.0525 + (6/10)*0.1056 = 0.0843

Threshold = 4.5000
Left: [0.8 0.3 0.5 0.2 0.7] Mean = 0.5000, Variance = 0.0520
Right [0.1 1.1 0.4 0.9 0.6] Mean = 0.6200, Variance = 0.1256
mse = (5/10)*0.0520 + (5/10)*0.1256 = 0.0888

Regression Trees – Choosing the best attribute

A	0	1	2	3	4	5	6	7	8	9
y	0.8	0.3	0.5	0.2	0.7	0.1	1.1	0.4	0.9	0.6

Threshold = 5.5000
Left: [0.8 0.3 0.5 0.2 0.7 0.1] Mean = 0.4333, Variance = 0.0656
Right [1.1 0.4 0.9 0.6] Mean = 0.7500, Variance = 0.0725
mse = (6/10)*0.0656 + (4/10)*0.0725 = 0.0683

Threshold = 6.5000
Left: [0.8 0.3 0.5 0.2 0.7 0.1 1.1] Mean = 0.5286, Variance = 0.1106
Right [0.4 0.9 0.6] Mean = 0.6333, Variance = 0.0422
mse = (7/10)*0.1106 + (3/10)*0.0422 = 0.0901

Threshold = 7.5000
Left: [0.8 0.3 0.5 0.2 0.7 0.1 1.1 0.4] Mean = 0.5125, Variance = 0.0986
Right: [0.9 0.6] Mean = 0.7500, Variance = 0.0225
mse = (8/10)*0.0986 + (2/10)*0.0225 = 0.0834

Regression Trees – Choosing the best attribute

A	0	1	2	3	4	5	6	7	8	9
y	0.8	0.3	0.5	0.2	0.7	0.1	1.1	0.4	0.9	0.6

Threshold = 8.5000
Left: [0.8 0.3 0.5 0.2 0.7 0.1 1.1 0.4 0.9] Mean = 0.5556, Variance = 0.1025
Right: [0.6] Mean = 0.6000, Variance = 0.0000
mse = (9/10)*0.1025 + (1/10)*0.0000 = 0.0922

Lowest mse is 0.0683, corresponding to threshold = 5.5