

# Data Mining 3

---

Decision trees

# Decision trees: table of contents

---

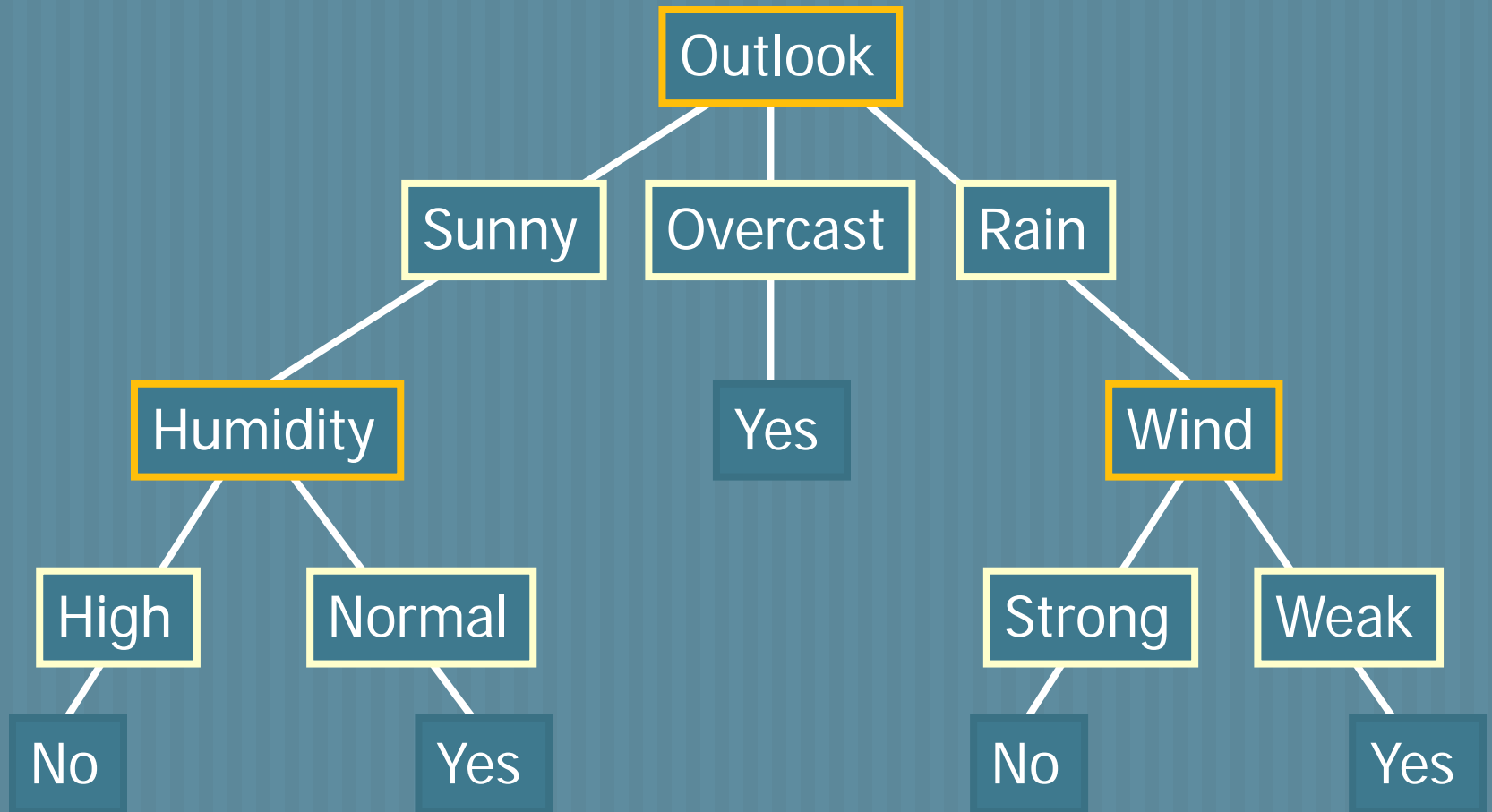
- Representational capacity of decision trees
- Entropy and information gain
- ID3 Learning algorithm
- Improvements to avoid overfitting

# Training examples for "PlayTennis"

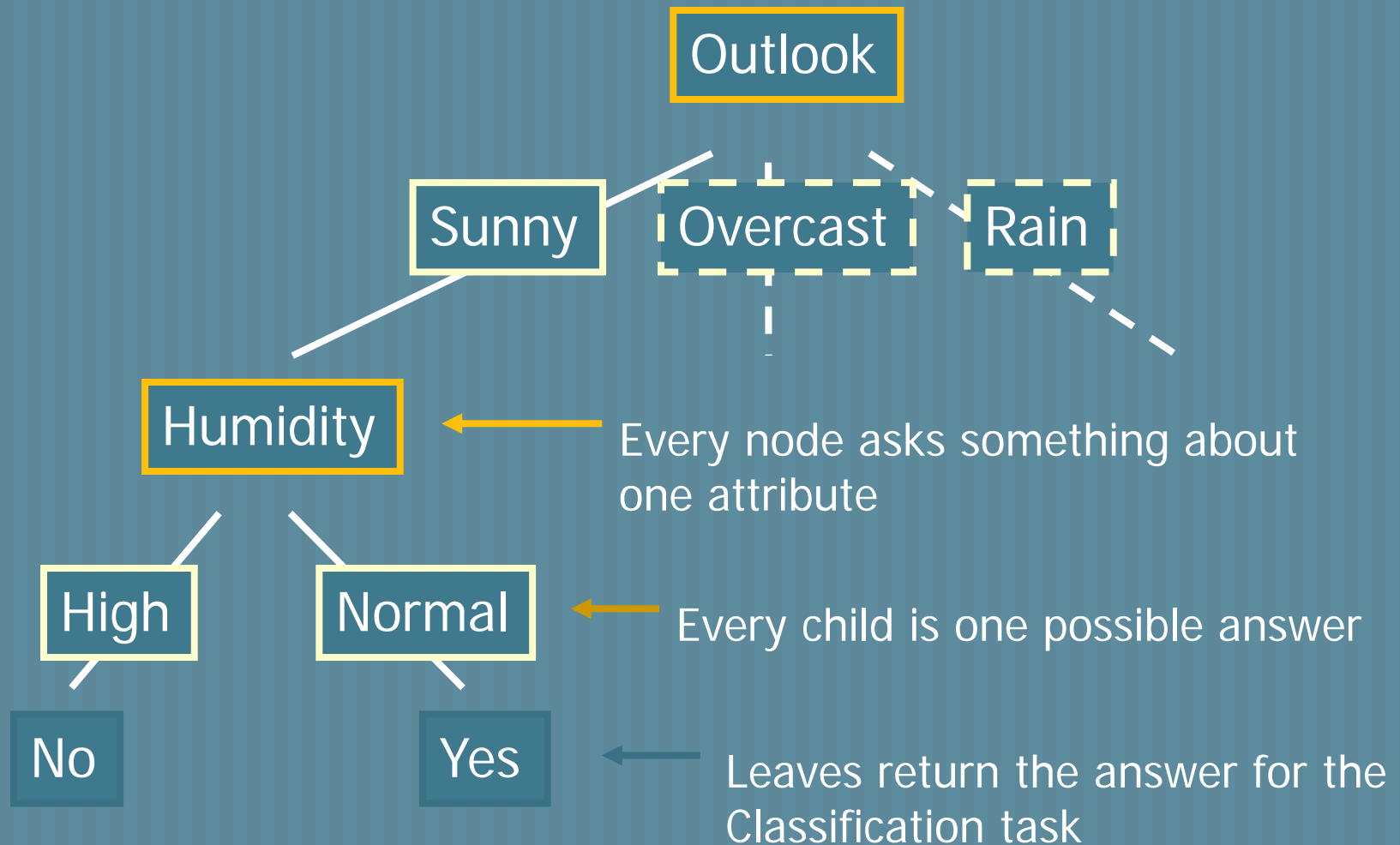
<i>Day</i>	<i>Outlook</i>	<i>Temp.</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
<b>D1</b>	Sunny	Hot	High	Weak	No
<b>D2</b>	Sunny	Hot	High	Strong	No
<b>D3</b>	Overcast	Hot	High	Weak	Yes
<b>D4</b>	Rain	Mild	High	Weak	Yes
<b>D5</b>	Rain	Cool	Normal	Weak	Yes
<b>D6</b>	Rain	Cool	Normal	Strong	No
<b>D7</b>	Overcast	Cool	Normal	Weak	Yes
<b>D8</b>	Sunny	Mild	High	Weak	No
<b>D9</b>	Sunny	Cold	Normal	Weak	Yes
<b>D10</b>	Rain	Mild	Normal	Strong	Yes
<b>D11</b>	Sunny	Mild	Normal	Strong	Yes
<b>D12</b>	Overcast	Mild	High	Strong	Yes
<b>D13</b>	Overcast	Hot	Normal	Weak	Yes
<b>D14</b>	Rain	Mild	High	Strong	No

# Decision tree for "PlayTennis"

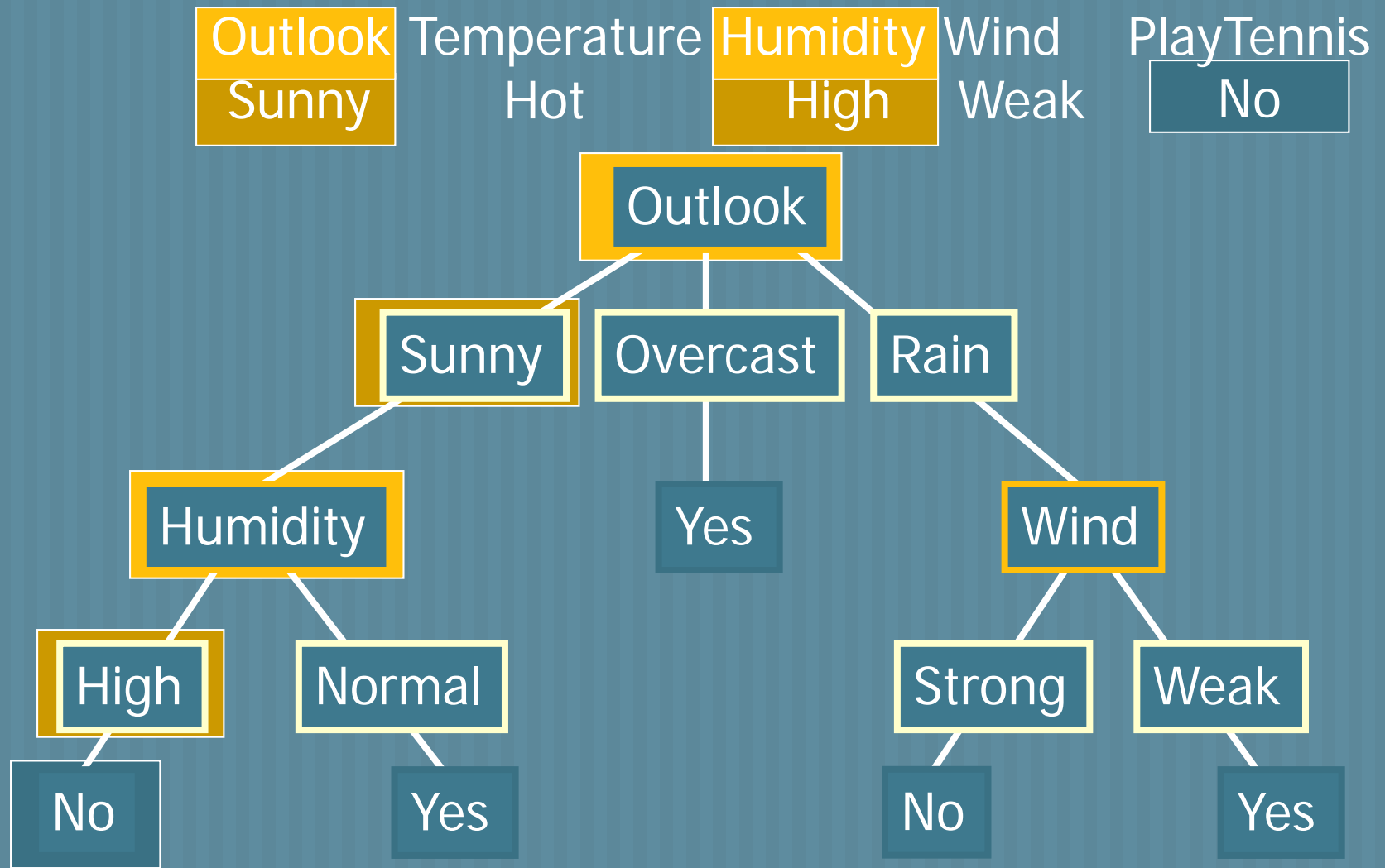
---



# Decision tree for "PlayTennis"

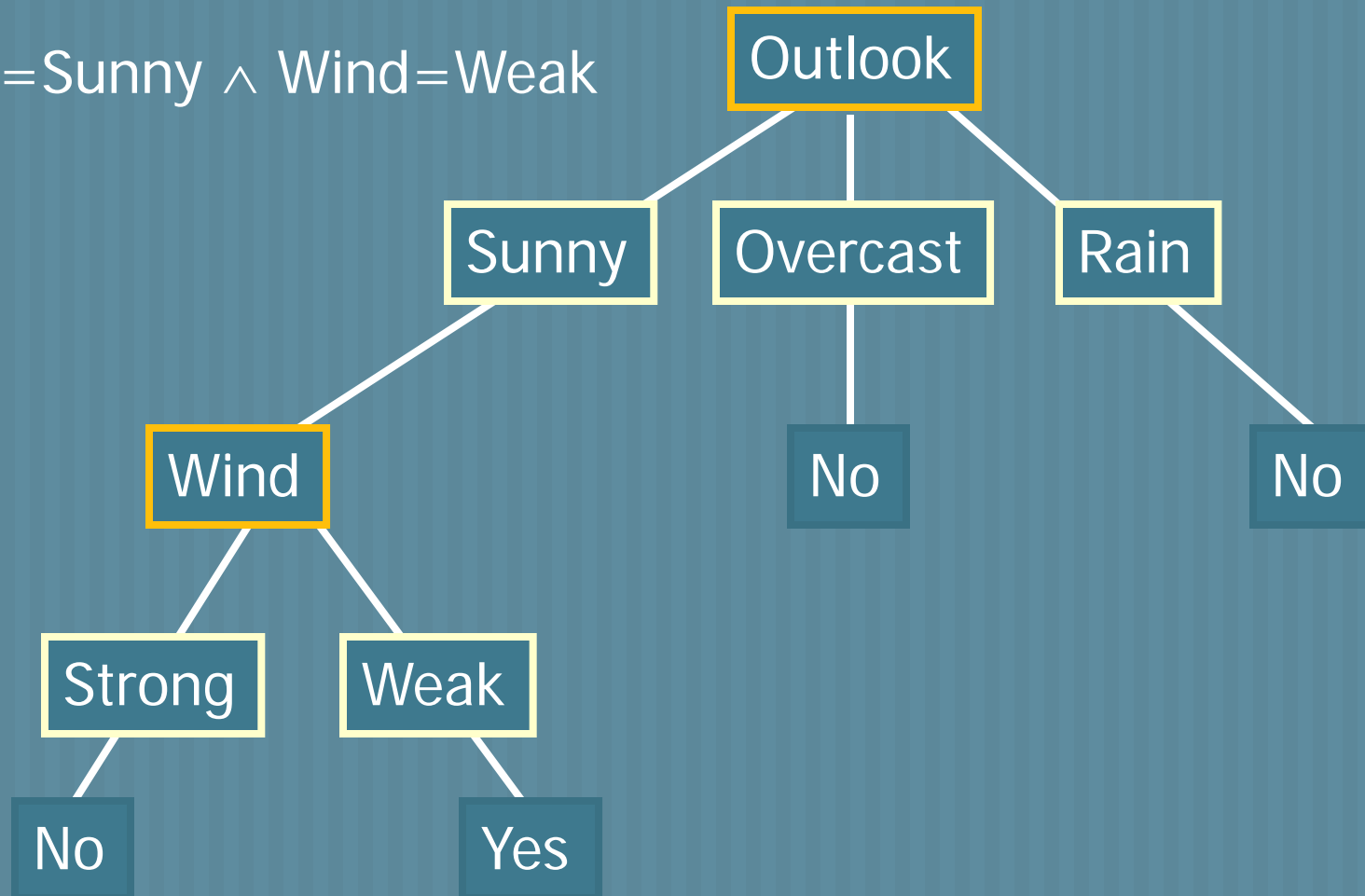


# Decision tree for "PlayTennis"

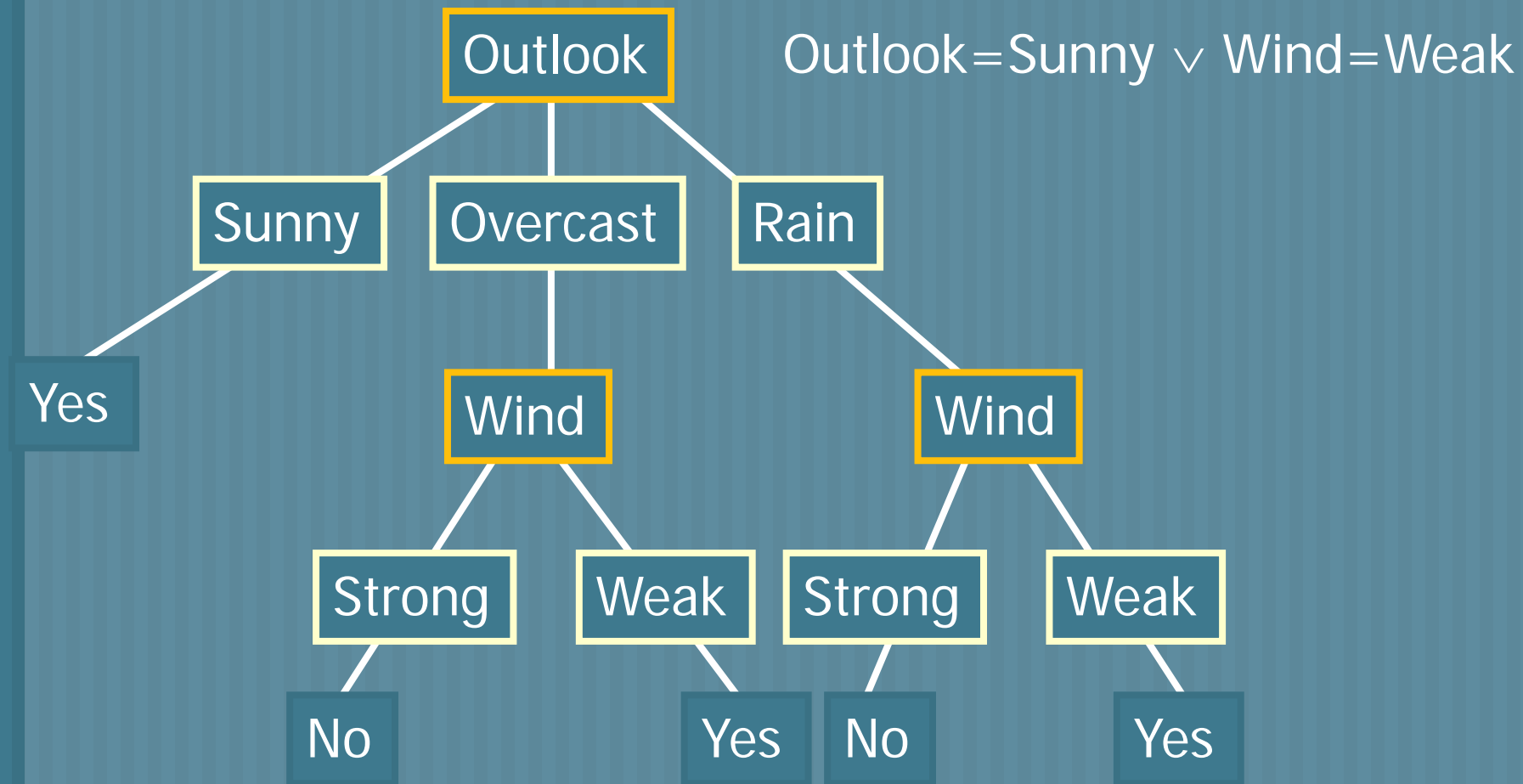


# DT can represent conjunctions

Outlook=Sunny  $\wedge$  Wind=Weak

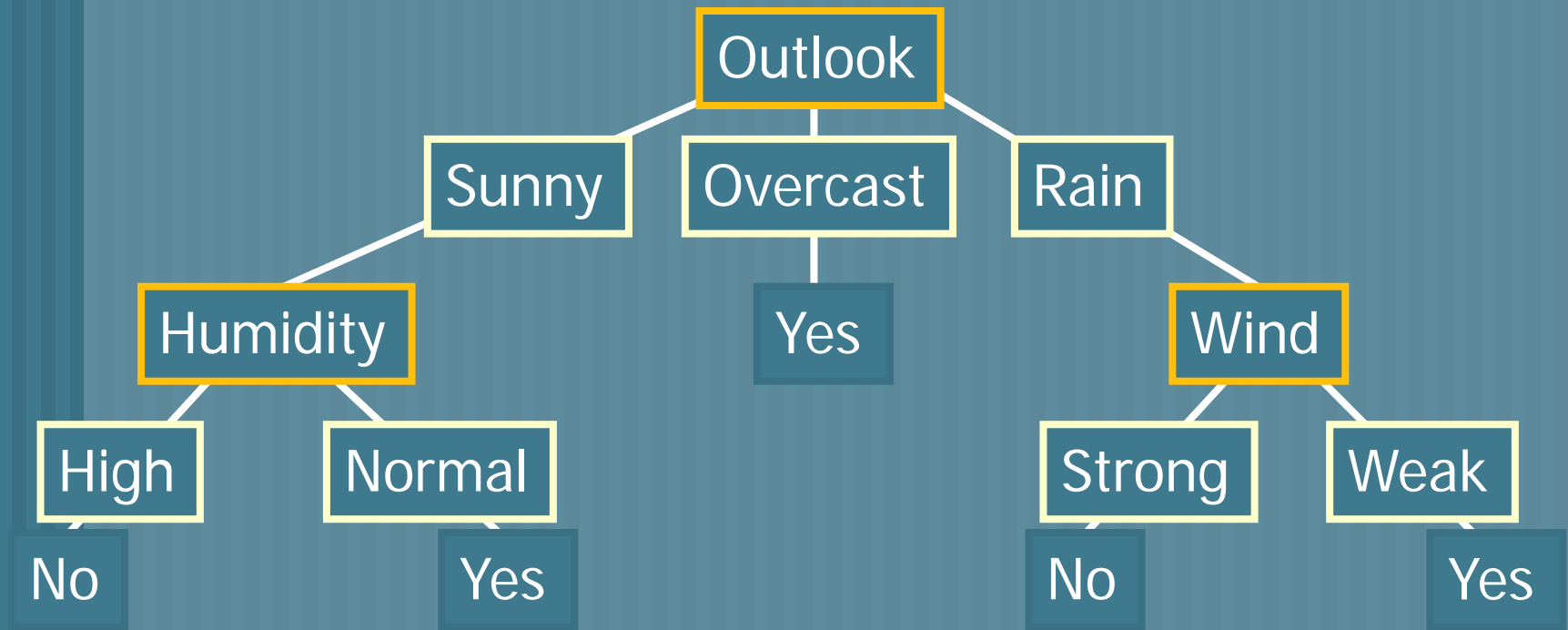


# DT can represent disjunctions





# DT can represent complex rules



(Outlook=Sunny  $\wedge$  Humidity=Normal)

✓ (Outlook=Overcast)

✓ (Outlook=Rain  $\wedge$  Wind=Weak)

# Learning Decision Trees

---

- From a training set, many possible trees
- Do not learn any tree but the simplest one! (simplicity for generalization)
- The simplest tree is the shortest one.
- One approach: Creating all the trees and keep the shortest one
- ... But this approach requires a too much computation time
- ID3/C4.5 builds short decision trees more efficiently (although does not ensure to get the shortest one).

# Initial requirements for ID3

---

- Examples described as pairs  $\langle \text{attribute}, \text{value} \rangle$
- The attributes that describe the examples are mainly discrete variables
- Concept to be learnt can be described by logical rules
- Examples:
  - Medical diagnosis
  - Analysis of risk in credit
  - Classification of objects for manipulation
  - ...

# “Top-Down” induction of decision trees: ID3

---

1. Set *node* to the training examples and put it as the root of the tree
2.  $A \leftarrow$  the "best" test attribute for the current node
3. For each value of attribute *A*, create a new descendant node in the decision tree
4. Separate examples in *node* among descendants depending of the attribute value for *A*
5. If there remain leafs nodes in the tree with mixed positive and negative examples, set this leaf to *node* and return to step 2.
6. In other case, label each leaf of the tree with the sign of the examples in the leaf, and return it

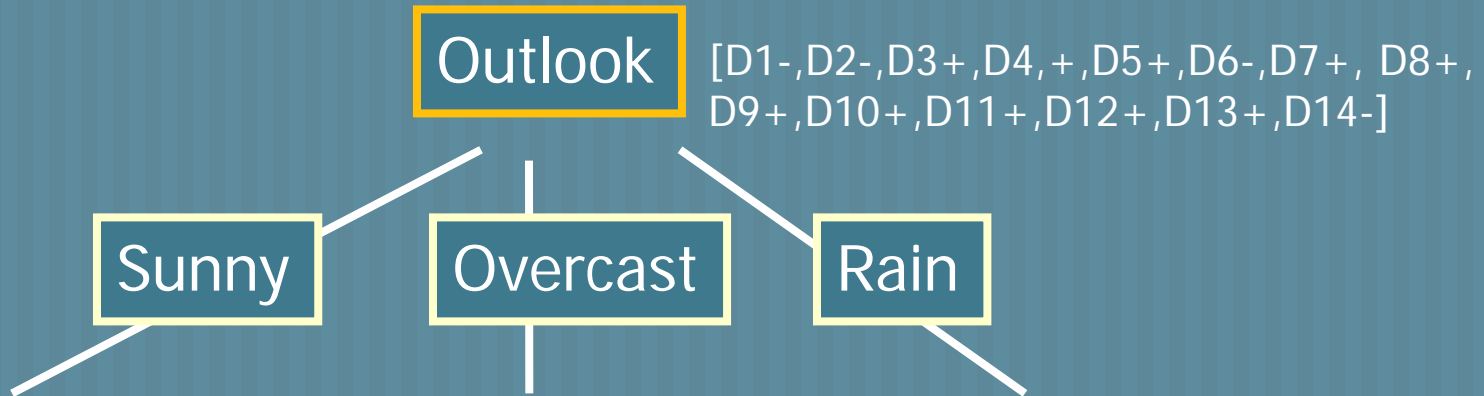
# ID3

---

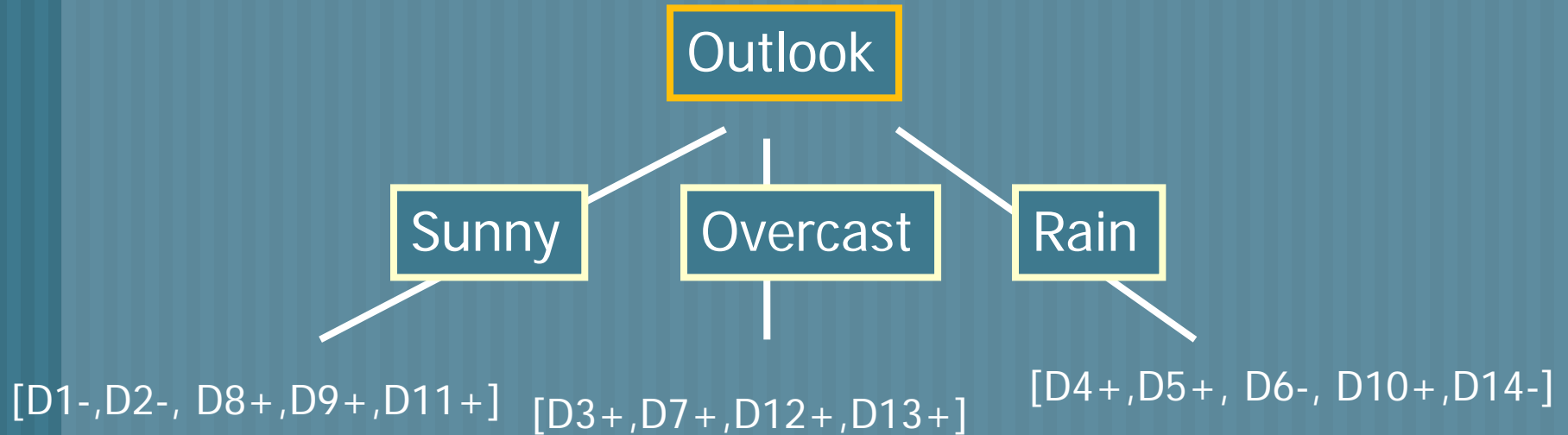
## Outlook

[D1-,D2-,D3+,D4,+,D5+,D6-,D7+, D8+,  
D9+,D10+,D11+,D12+,D13+,D14-]

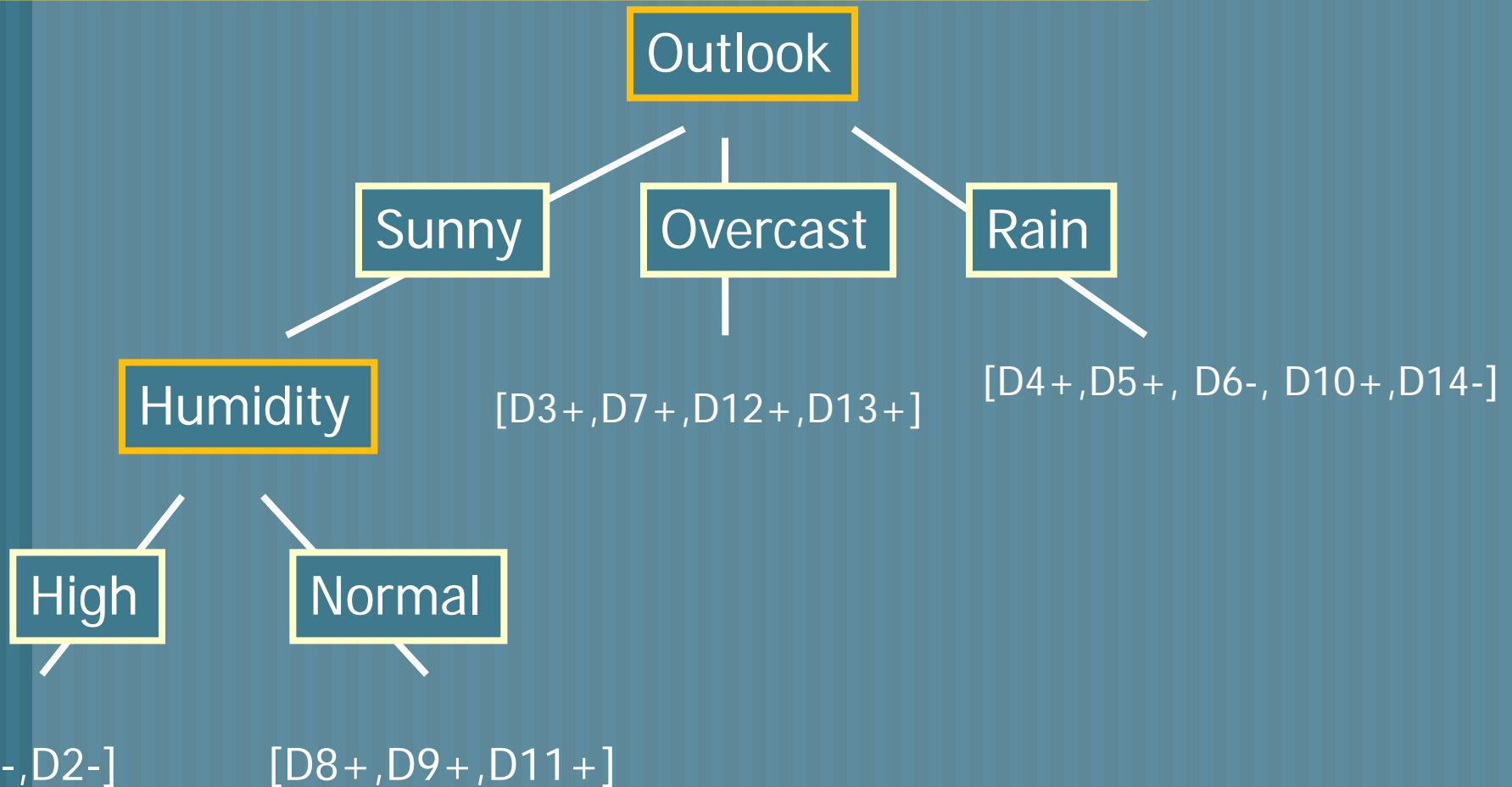
# ID3



# ID3

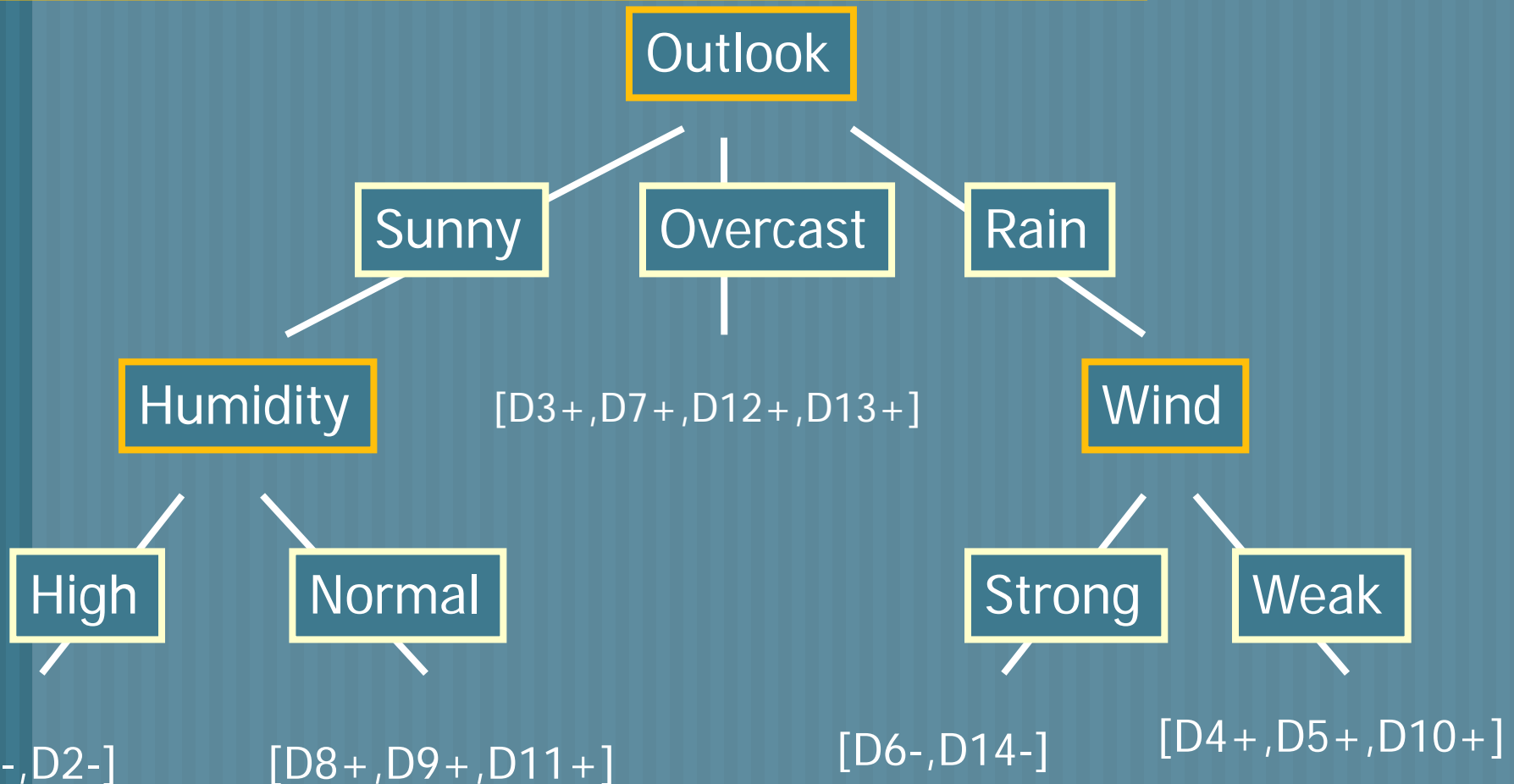


# ID3

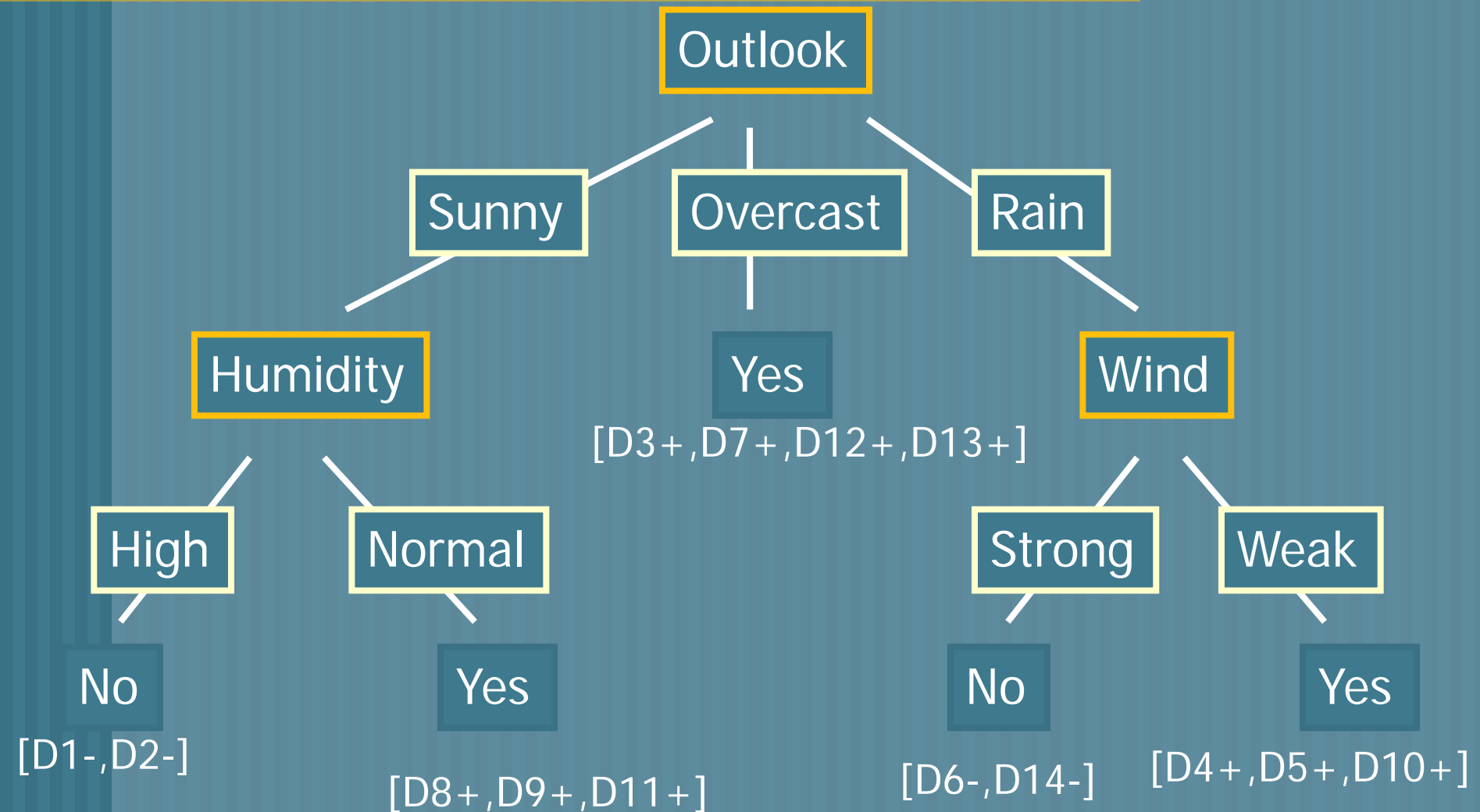




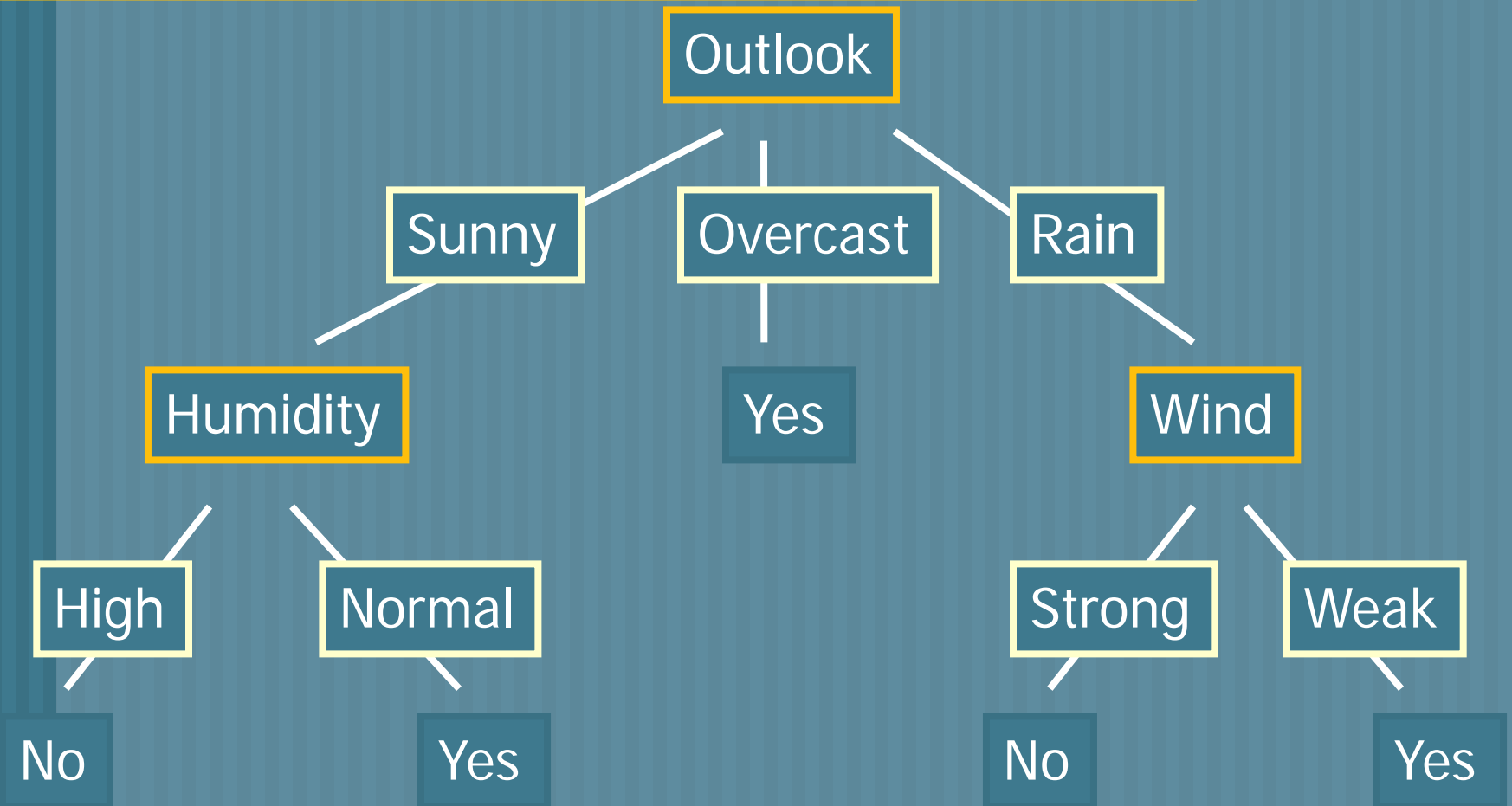
# ID3



# ID3



# ID3



# How to choose the best attribute?

---

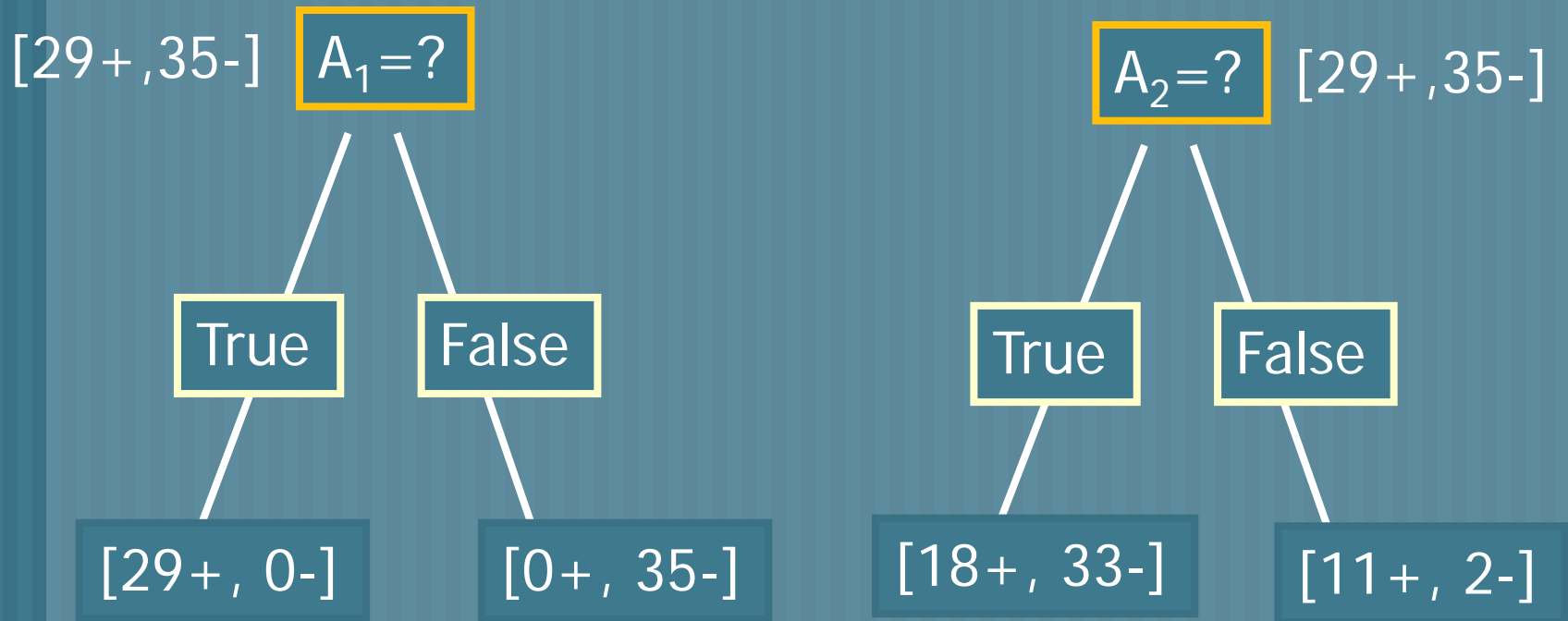
- Goal: To obtain the shortest tree
- ... equivalent to separate as quickly as possible + examples from - examples
- Therefore questions should be the maximum discriminative

# How to choose the best attribute?

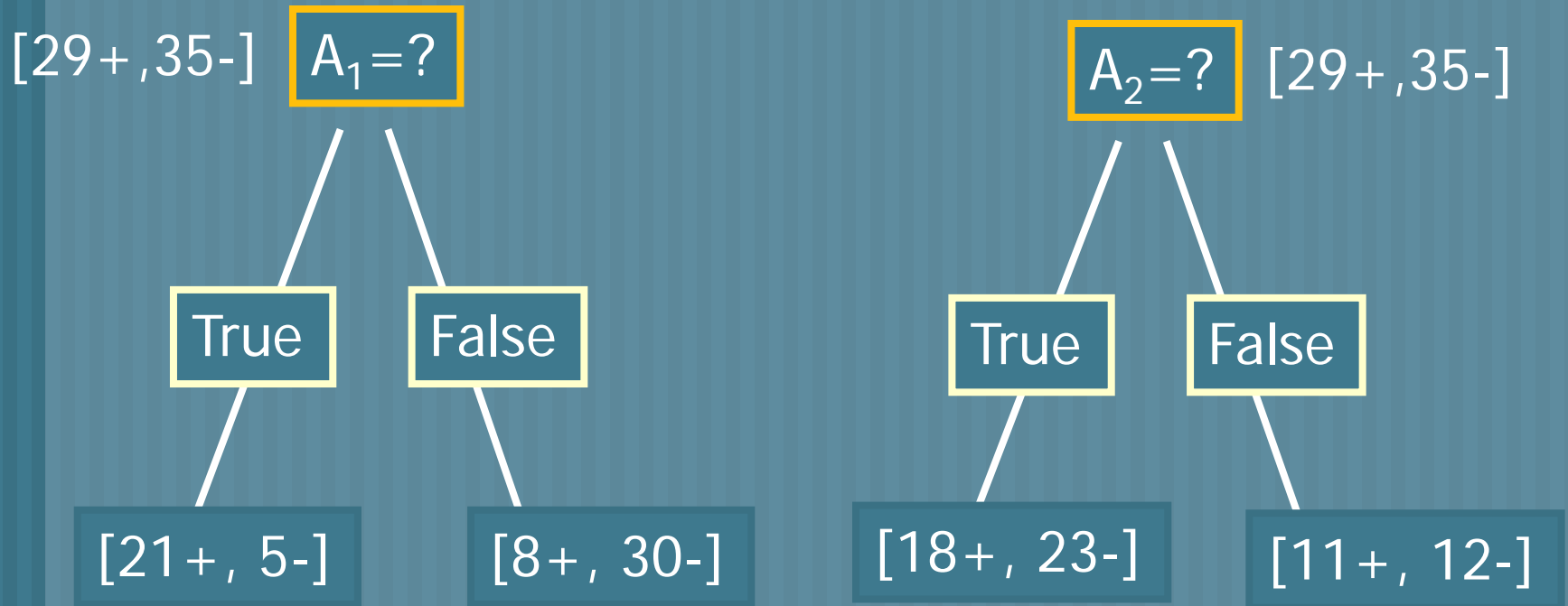
---

- Goal: To obtain the shortest tree
- ... equivalent to separate as quickly as possible + examples from - examples
- Therefore questions should be the maximum discriminative
- Information Gain...

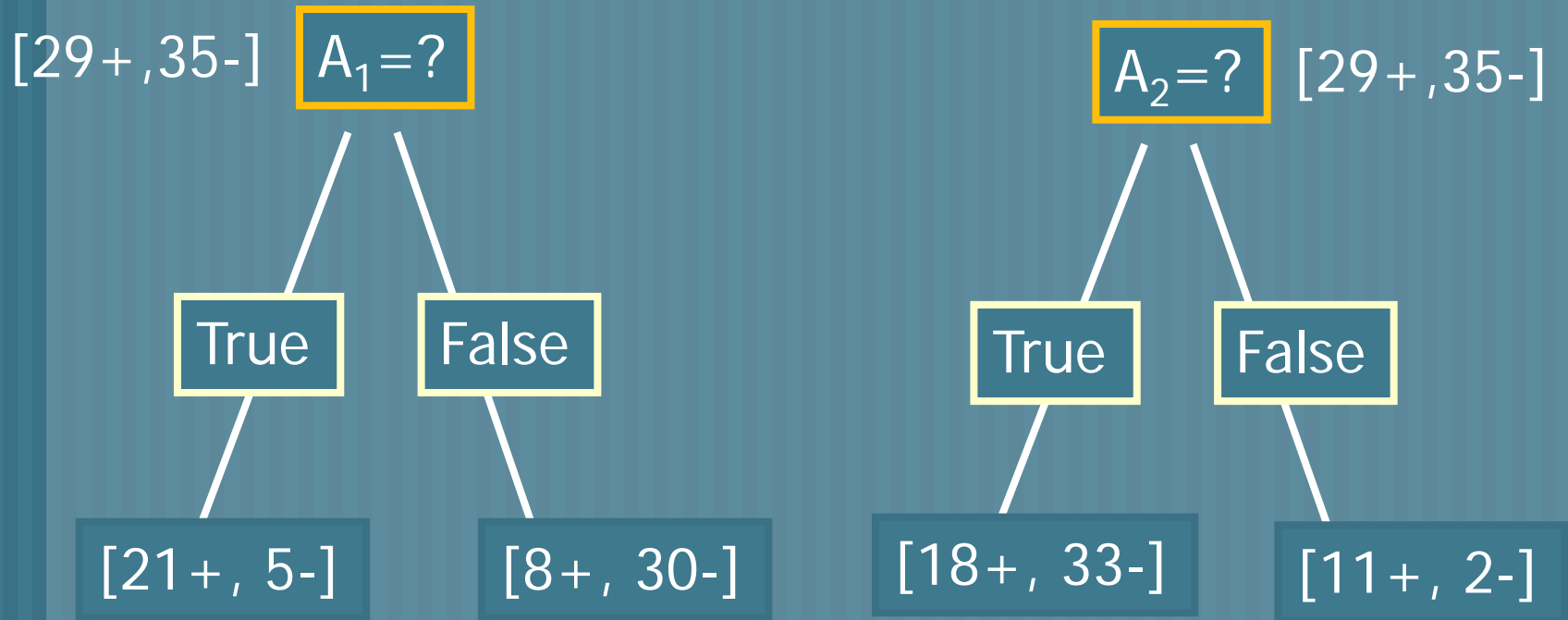
# What attribute is best?



# What attribute is best?



# What attribute is best?





# Information theory

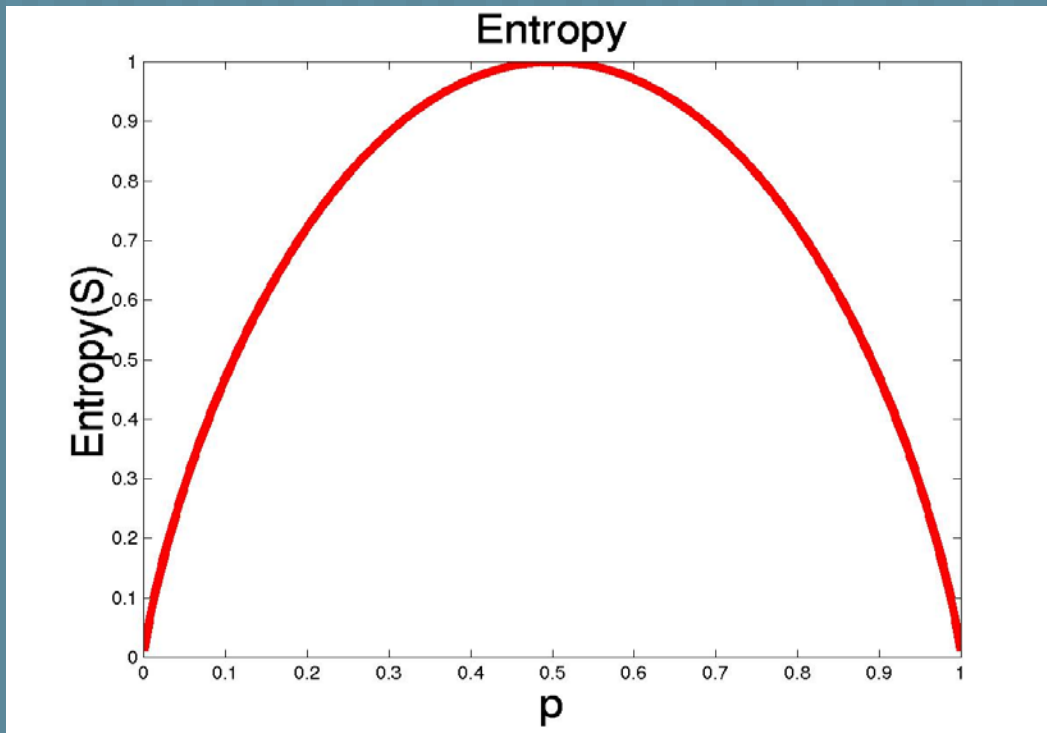
---

- One answer gives us an information amount proportional to the unpredictability of the answer
- Example: Diabetic Test
- Formally:

$$E(A) = -\sum_i^n p_i \log(p_i)$$

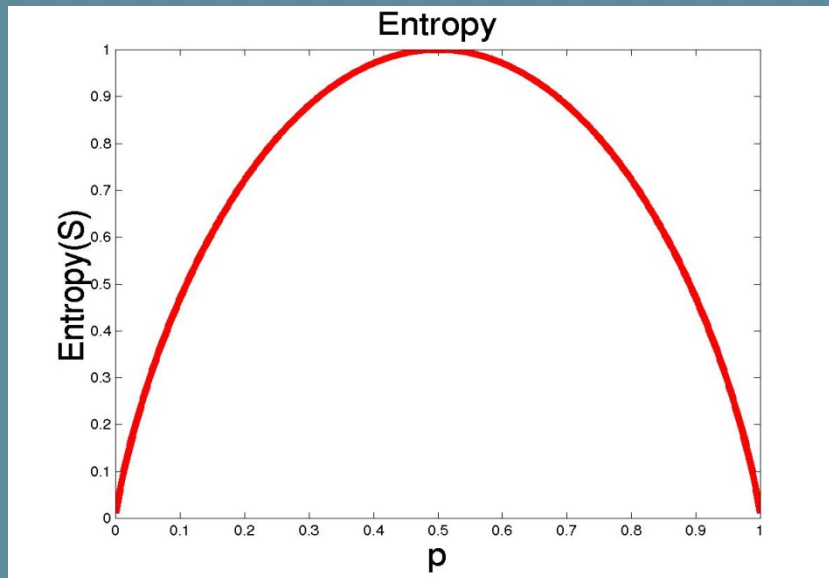
Where  $p_i$  is probability of answer  $i$ , and  $n$  is the number of possible answers

# Binary answer case



$$E(A) = -p_{SI} \log_2 p_{SI} - p_{NO} \log_2 p_{NO} =$$
$$-p_{SI} \log_2 p_{SI} - (1-p_{SI}) \log_2 (1-p_{SI})$$

# Binary answer case



- For DTs, we assume:
  - $p_+$  the proportion of + examples in  $S$
  - $p_-$  the proportion of - examples in  $S$
- $E(S)$  measures the "mixing" of examples in  $S$   
$$E(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

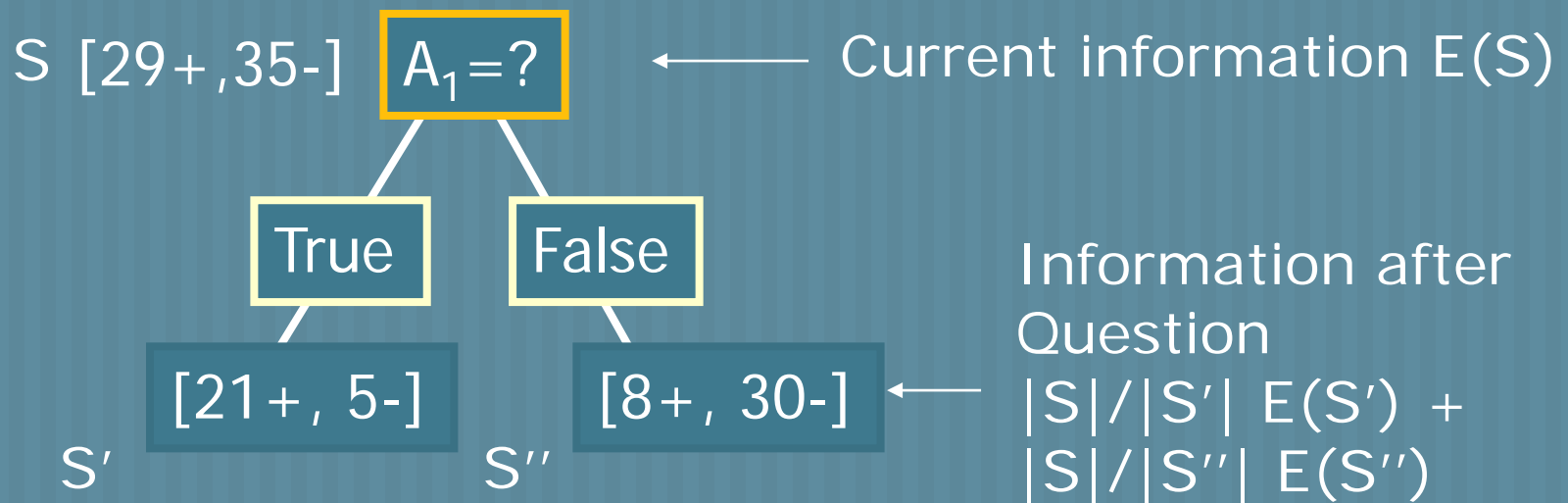
# Entropy

---

- $E(S)$  –entropy of  $S$ - measures necessary bits to compress information about current distribution of  $+$  and  $-$  examples in  $S$  considering it a random variable. It is a measure of information content taken from the Information Theory field.
- It also can be seen as a measure of disorder... That's why it is useful in ID3.

# Information gain

- Information Gain (IG) obtained from an answer is the difference of information before and after the answer.



# Information gain

$$G(S,A) = E(S) - \sum_{v \in \text{values}(A)} |S_v|/|S| E(S_v)$$

$$E([29+, 35-]) = -29/64 \log_2 29/64 - 35/64 \log_2 35/64 = 0.99$$



# Information gain

$$E([21+, 5-]) = 0.71$$

$$E([8+, 30-]) = 0.74$$

$$G(S, A_1) = E(S)$$

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

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

$$= 0.27$$

$$E([18+, 33-]) = 0.94$$

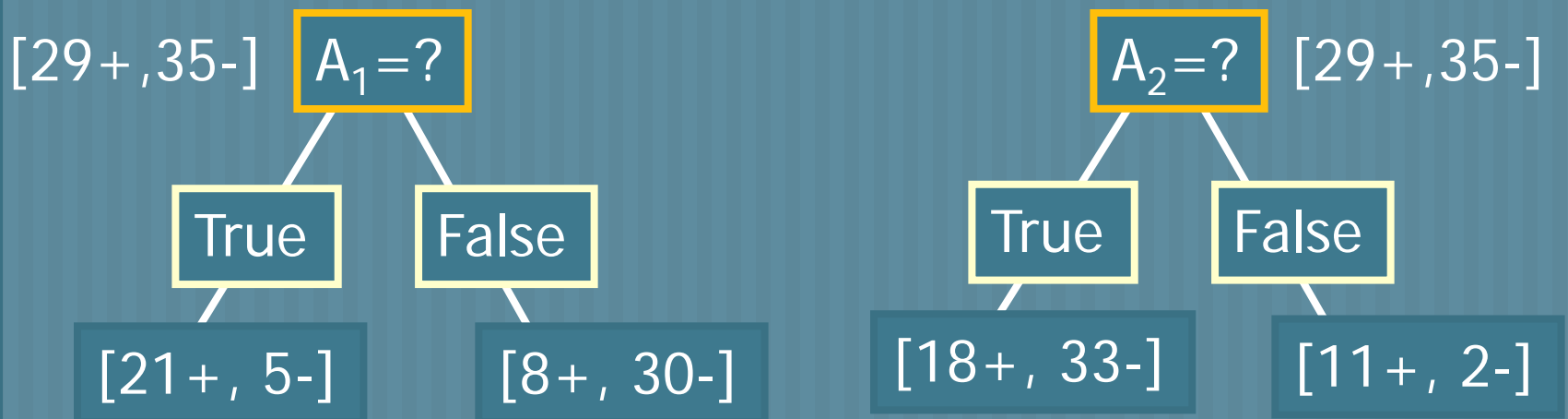
$$E([8+, 30-]) = 0.62$$

$$G(S, A_2) = E(S)$$

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

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

$$= 0.12$$



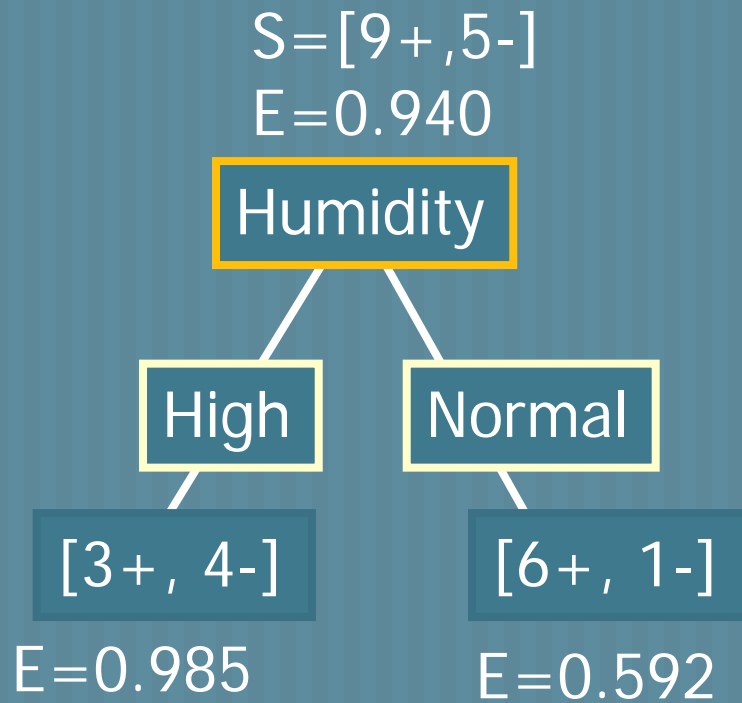
# Which is the best attribute to choose?

---

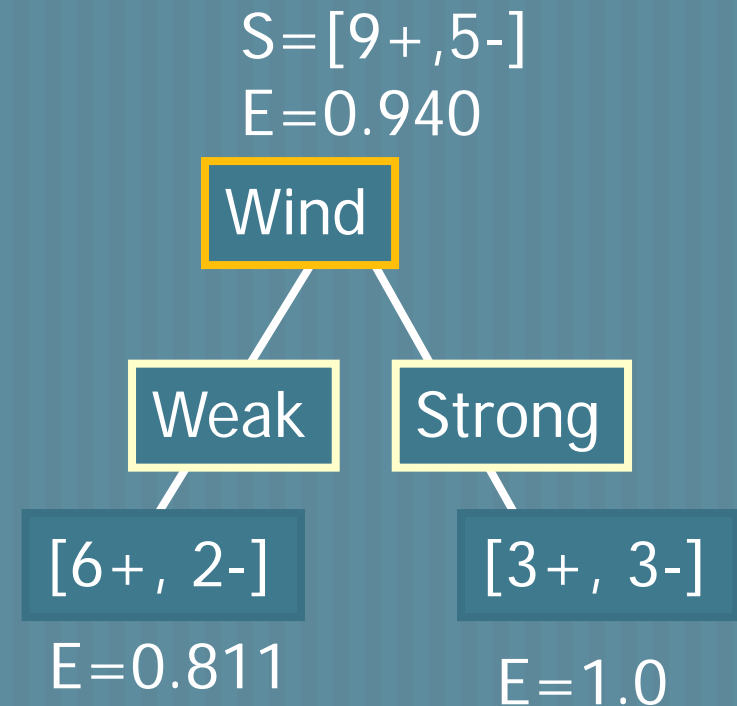
- The one with highest information gain!!!



# Choosing the attribute

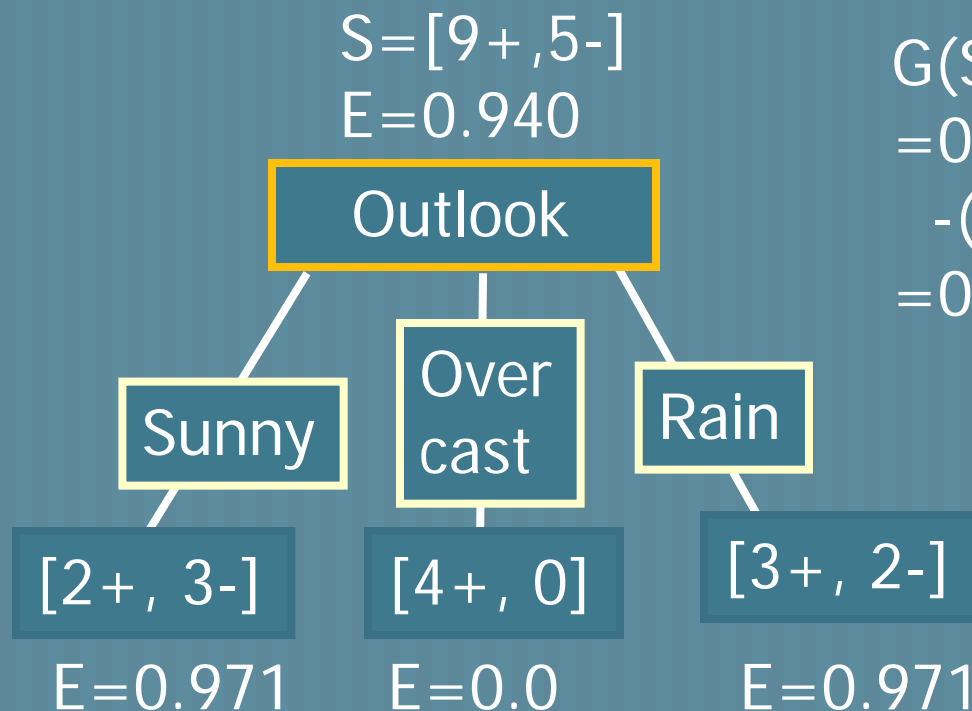


$$\begin{aligned} G(S, \text{Humidity}) &= 0.940 - (7/14) * 0.985 \\ &\quad - (7/14) * 0.592 \\ &= 0.151 \end{aligned}$$



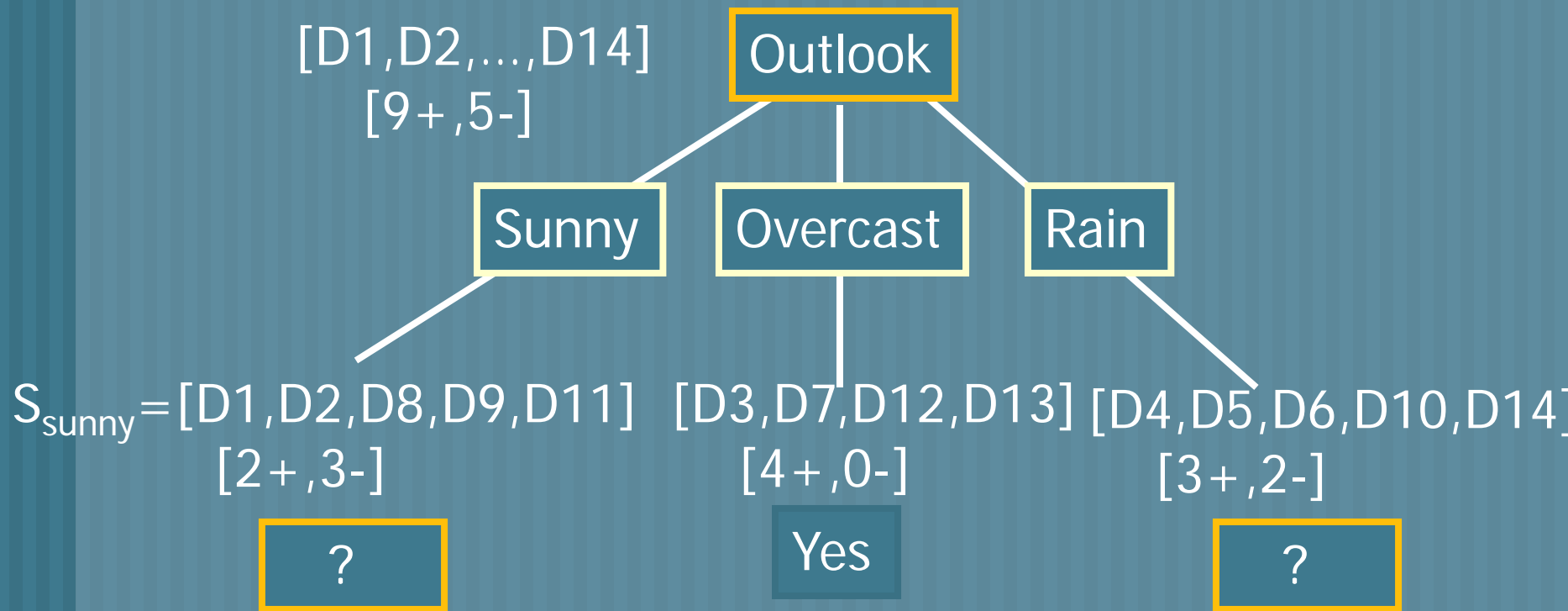
$$\begin{aligned} G(S, \text{Wind}) &= 0.940 - (8/14) * 0.811 \\ &\quad - (6/14) * 1.0 \\ &= 0.048 \end{aligned}$$

# Choosing the attribute



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

# Choosing next attribute

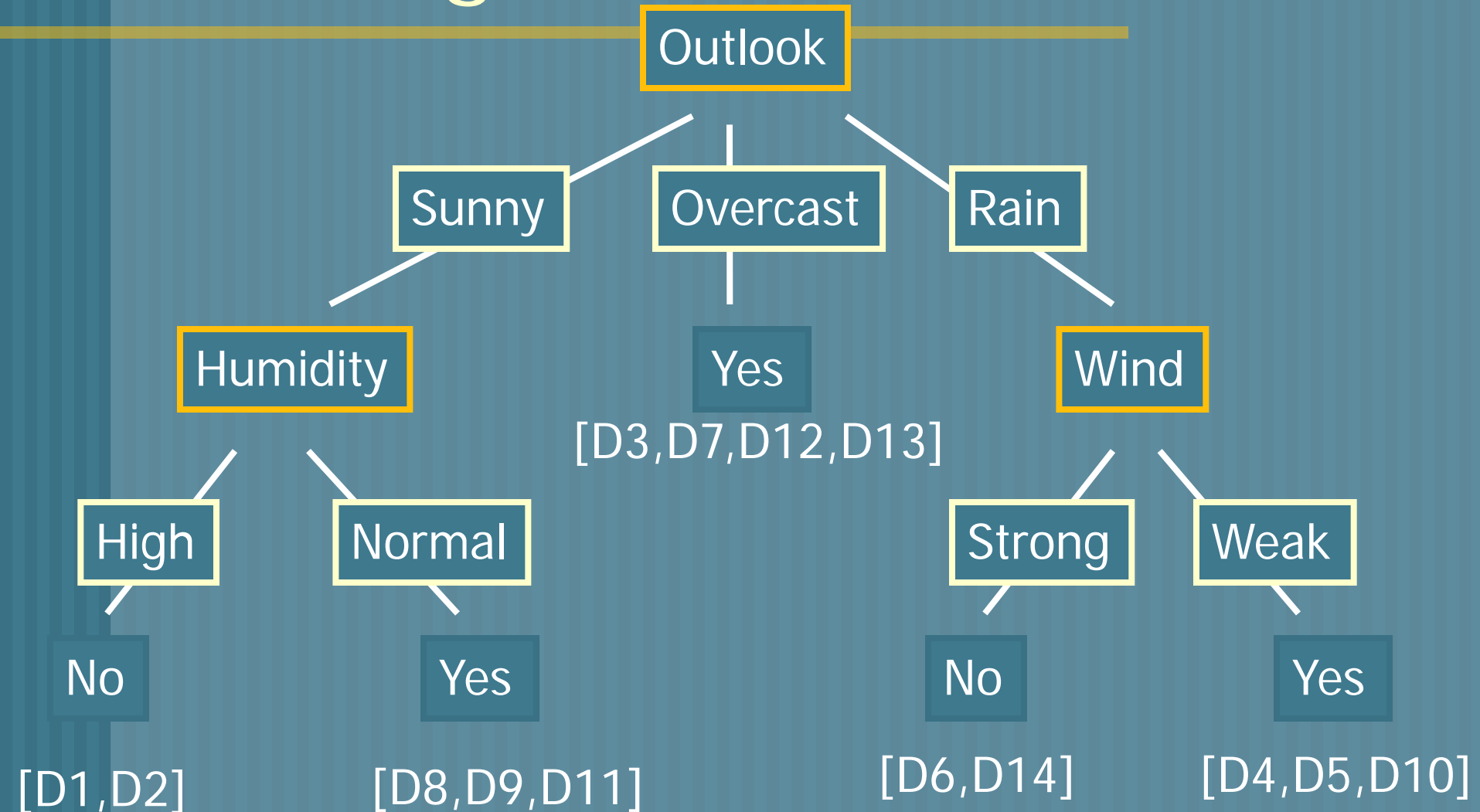


$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = 0.970 - (3/5)0.0 - 2/5(0.0) = 0.970$$

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

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

# ID3 Algorithm



# ID3 Algorithm

**Algorithm** ID3 (X: Examples, C: Classes, A: Attributes)

**IF** all examples are of the same class **return** leave with label of the class

**ELSE**

**FOR ALL** attribute  $a$  in  $A$

        Compute entropy ( $E$ ) and information gain  $IG(a)$

**END**

$best =$  attribute with highest  $IG$

$A = A - \{best\}$  % Remove *best* from the attribute list ( $A$ )

**FOR EACH** value  $i$  of *best*

$Tree\_i = ID3(\{X \mid best=v\_i\}, \{C(X) \mid best=v\_i\}, A)$

        Root  $Tree\_i$  to current node with branch  $v\_i$

**END**

**return** Tree

**EndAlgorithm**

# [Possible issues while building the tree]

---

- What if there are not examples for one value when expanding a node? Do not generate branch. When testing, go to the most popular branch
- When we cannot expand the node but still there are + and – examples mixed in the node, label the node with the majority class.
- **Overfitting**

# Overfitting in DTs

---



# How to avoid overfitting?

---

- Do not split examples when Information Gain above a threshold (pre-bounding)
- Prune some branches of the tree (post-pruning)
  - post-pruning in the tree
  - post-pruning of rules (C4.5)



# Pre-pruning

- If the information gain is very small means that the attributes are not worth to be used.
  - Grow the tree in these circumstances is equivalent to take arbitrary decisions
  - So, stop branching and assign label to node with the majority class.
  - This case can be detected by using  $\chi^2_{(c-1)(v-1)}$   
Split is made when rejecting the null hyp.

$$\sum_{c_i \in \mathcal{C}} \sum_{v_i \in A} \frac{(P(c_i | A(x) = v_i) - P(c_i))^2}{P(c_i)}$$

# Post pruning of the tree

---

Split data in training and validation subsets

Repeat until no improvement in validation subset error:

1. For each node compute the error on the validation set when removing the node (and so the subtree below the node)
2. Do the pruning that decrease the most the error on the validation set

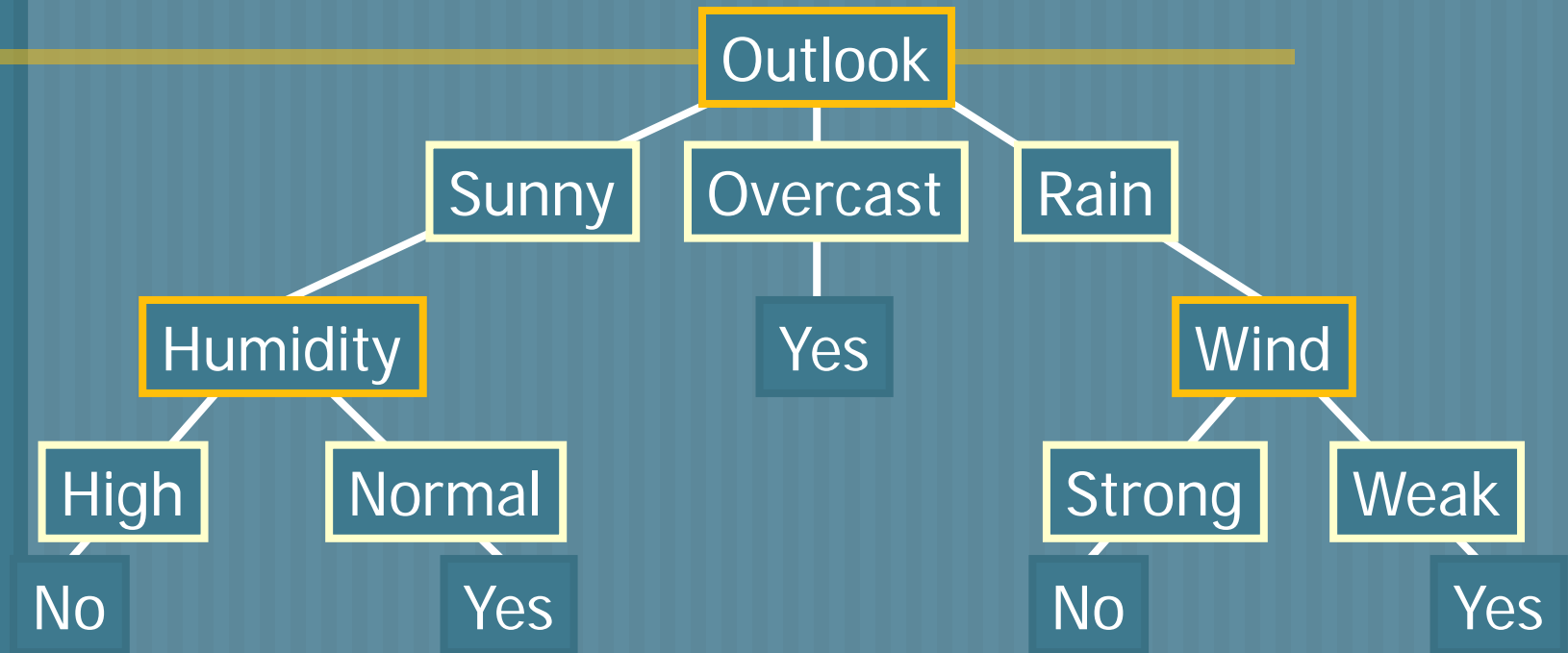
# Post pruning of rules (C4.5)

---

## ■ C4.5 Algorithm

1. Build the decision tree with ID3
2. Transform the tree into rules
3. Prune each rule independently by removing conditions that decrease the error on validation set
4. Sort the final set of rules to solve conflicts
5. Remove rules when it decrease error on validation set

# Translating a tree into a set of rules



$R_1$ : If (Outlook=Sunny)  $\wedge$  (Humidity=High) Then PlayTennis=No

$R_2$ : If (Outlook=Sunny)  $\wedge$  (Humidity=Normal) Then PlayTennis=Yes

$R_3$ : If (Outlook=Overcast) Then PlayTennis=Yes

$R_4$ : If (Outlook=Rain)  $\wedge$  (Wind=Strong) Then PlayTennis=No

$R_5$ : If (Outlook=Rain)  $\wedge$  (Wind=Weak) Then PlayTennis=Yes

# Extensions

---

- Attributes with numeric values?
  - Discretize... But how to define thresholds?
- Attributes with missing values?
  - Different problems in different steps
- Undesired effect: Bias towards attributes with a large number of different values
- Different costs for different questions

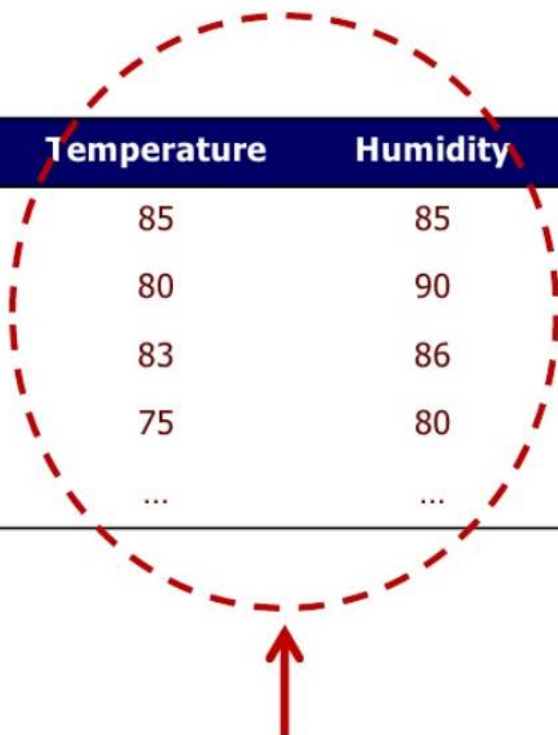
# Extensions

---

- **Attributes with numeric values?**
  - Discretize... But how to define thresholds?
- Attributes with missing values?
  - Different problems in different steps
- Undesired effect: Bias towards attributes with a large number of different values
- Different costs for different questions

# Attributes with numeric values

## □ Example



Outlook	Temperature	Humidity	Windy	Play
Sunny	85	85	False	No
Sunny	80	90	True	No
Overcast	83	86	False	Yes
Rainy	75	80	False	Yes
...	...	...	...	...

**Continuous attributes**

# Attributes with numeric values

---

- Given continuous attribute  $A$ , consider all different splits  $[A[x] < v_i]$  versus  $[A[x] \geq v_i]$ :
  - Start from sorted sequence of values  $A = \{v_1, v_2, \dots, v_n\}$
  - Consider each value as the threshold for the split. For each different split, compute the information gain. (Note: There are at most  $n-1$  possible splits)
  - Turn the split with highest IG into the boolean test for the node  
Data will be split into  $[A[x] < v_i]$  and  $[A[x] \geq v_i]$
- Attribute  $A$  cannot be removed for future splits
- Of course, this increase a lot the computation time.



# Attributes with numeric values

## □ Split on temperature attribute:

64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

- E.g.: temperature < 71.5: yes/4, no/2  
temperature ≥ 71.5: yes/5, no/3

- $\text{Info}([4,2],[5,3]) = 6/14 \text{ info}([4,2]) + 8/14 \text{ info}([5,3]) = 0.939 \text{ bits}$

- Place split points halfway between values
- Can evaluate all split points in one pass!

# Attributes with numeric values

## □ To speed up

- Entropy only needs to be evaluated between points of different classes

value	64	65	68	69	70	71	72	72	75	75	80	81	83	85
class	Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

Potential optimal breakpoints

Breakpoints between values of the same class cannot be optimal

# Extensions

---

- Attributes with numeric values?
  - Discretize... But how to define thresholds?
- **Attributes with missing values?**
  - **Different problems in different steps**
- Undesired effect: Bias towards attributes with a large number of different values
- Different costs for different questions

# Attributes with missing values?

---

- The problem can appear in three different steps:
  1. When choosing the attribute while building the DT
  2. When splitting the examples while building the DT
  3. When testing one example
- Possible solutions:
  - Solve the problem previously
  - Remove the examples (in 1 and 2)
  - Assume missing values as the average or mode of those of the majority class (1 or 3), or split examples with missing values proportionally (2)
  - Build a new category for the value (1 and 2)

# Attributes with missing values?

---

- Similar problem when building the tree and we don't have examples for each category.
- Or an example appear with a new category for one attribute.

# Extensions

---

- Attributes with numeric values?
  - Discretize... But how to define thresholds?
- Attributes with missing values?
  - Different problems in different steps
- **Undesired effect: Bias towards attributes with a large number of different values**
- Different costs for different questions

# Bias towards attributes with a large number of values

- Tendency to Overfitting (f.i.: DNI number)
- Example:

A <sub>1</sub>	n. ejem	A <sub>2</sub>	n. ejem
a	20+	a	40+
b	25+	b	40-
c	20-	c	10+ 10-
d	15-		
e	5+ 15-		

$$G(A_1) = 1 - \left( \overbrace{\frac{1}{5} \cdot 0}^a + \overbrace{\frac{1}{4} \cdot 0}^b + \overbrace{\frac{1}{5} \cdot 0}^c + \overbrace{\frac{3}{20} \cdot 0}^d + \overbrace{\frac{1}{5} \cdot \left( -\frac{1}{4} \cdot \log \frac{1}{4} \right) - \frac{3}{4} \cdot \log \frac{3}{4}}^e \right) = 0,84$$

$$G(A_2) = 1 - \left( \overbrace{\frac{2}{5} \cdot 0}^a + \overbrace{\frac{2}{5} \cdot 0}^b + \overbrace{\frac{1}{5} \cdot \left( -\frac{1}{2} \cdot \log \frac{1}{2} \right) - \frac{1}{2} \cdot \log \frac{1}{2}}^c \right) = 0,8$$

# Bias towards attributes with a large number of values

- One solution is to normalize IG by the number of categories

$$SI(\mathcal{X}, A) = - \sum_{\forall v_i \in A} \frac{\#[A(x) = v_i]}{\#\mathcal{X}} \cdot \log\left(\frac{\#[A(x) = v_i]}{\#\mathcal{X}}\right)$$

- In the example:

$$\frac{G(A_1)}{SI(A_1)} = \frac{0,86}{2,3} = 0,36 \quad \frac{G(A_2)}{SI(A_2)} = \frac{0,8}{1,52} = 0,52$$



# Extensions

---

- Attributes with numeric values?
  - Discretize... But how to define thresholds?
- Attributes with missing values?
  - Different problems in different steps
- Undesired effect: Bias towards attributes with a large number of different values
- **Different costs for different questions**

# Extensions

---

- Attributes with numeric values?
  - Discretize... But how to define thresholds?
- Attributes with missing values?
  - Different problems in different steps
- Undesired effect: Bias towards attributes with a large number of different values
- Different costs for different questions
  - Use other IG measures, f.i.  $G^2(S,A)/\text{Cost}(A)$

# Problems with DTs

---

- In general, very good approach to data mining. Efficient, reliable, but...
- Large variance in results (unstable: small changes in training set leads to very different DTs)
- Solution: Grow a forest of trees and let them vote. (We will see that later)