Decision Trees

Pasquale Rullo

rullo@mat.unical.it

History

Concept learning systems

1966: Hunt, conceptual learning

1977: Friedman/Breiman, CART (classification and regression trees)

1979: Quinlan, ID3 (interactive dichotomizer 3)

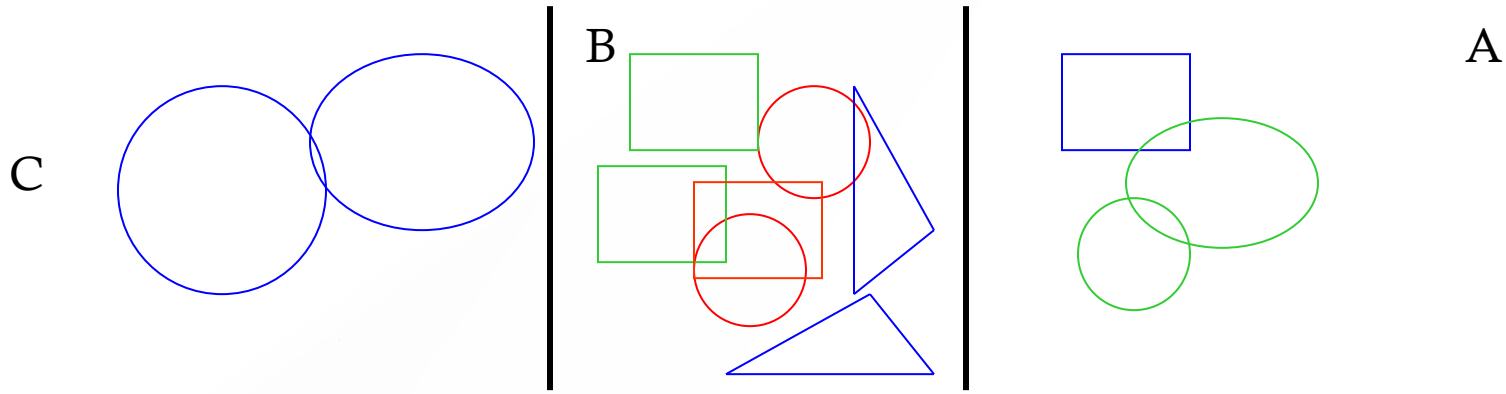
today's most popular algorithm: C4.5 (Quinlan)

→ J.R.Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, 1993

→ http://www.cse.unsw.edu.au/~quinlan/

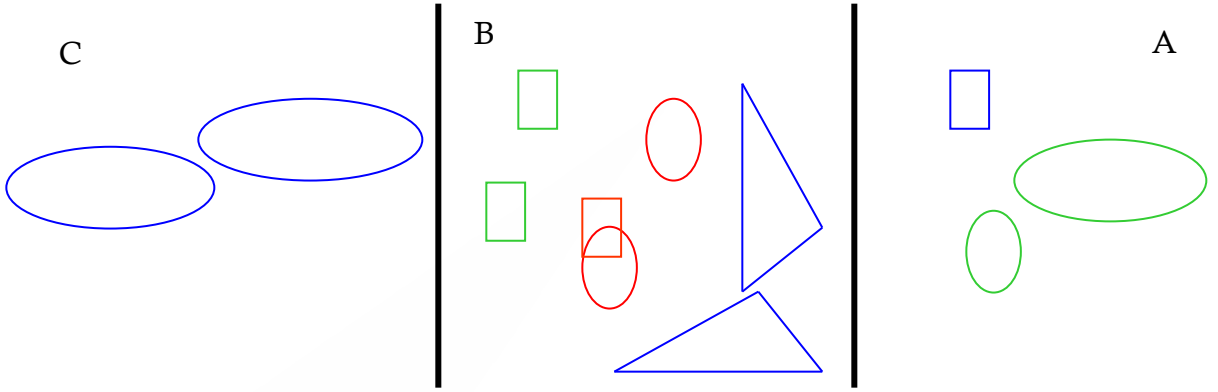
Decision Trees – an example

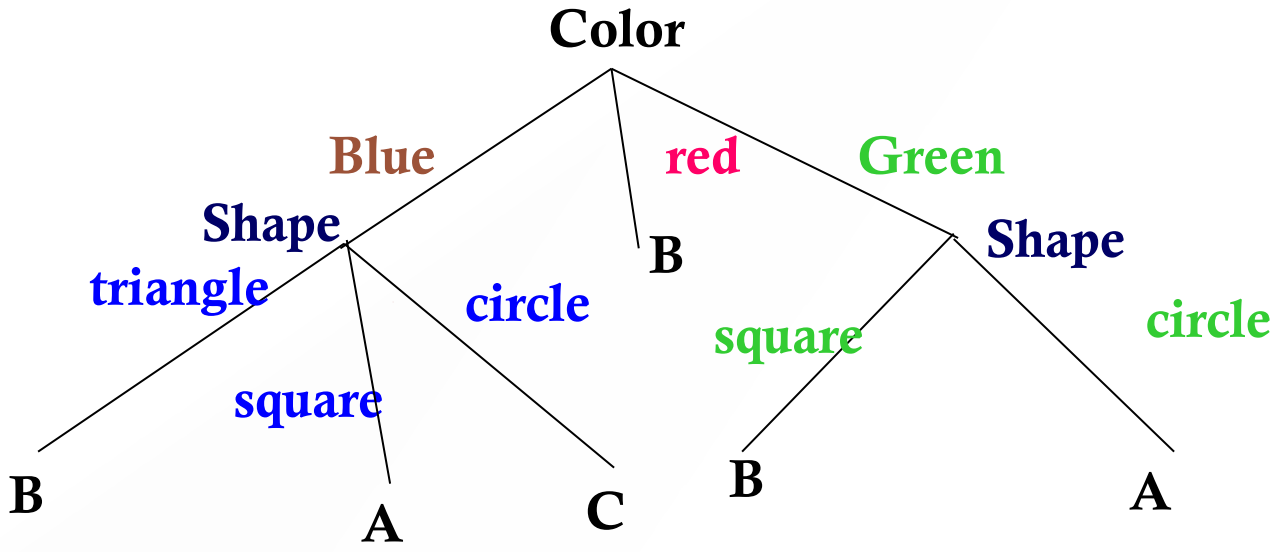
- What is the DT for the training data below?
- Two attributes: color and shape



Decision Trees – an example

Color	Shape	Class
red	circle	B
blue	square	A
…	…	…



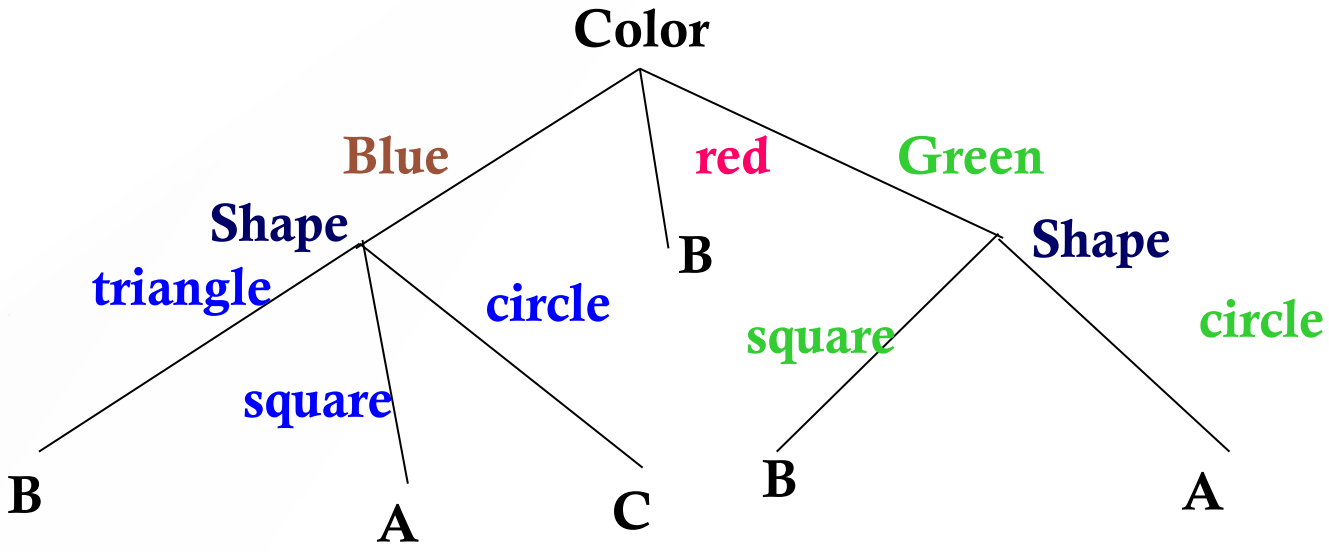


Decision Tree - Definition

- Decision tree:
  - interior nodes are labeled by attribute names a_i
  - arcs leaving an interior node labeled a_i are labelled by the values of a_i
  - leaf nodes labeled by class names
- A decision tree can be transformed into an equivalent set of symbolic rules, disjunction of conjunctions.

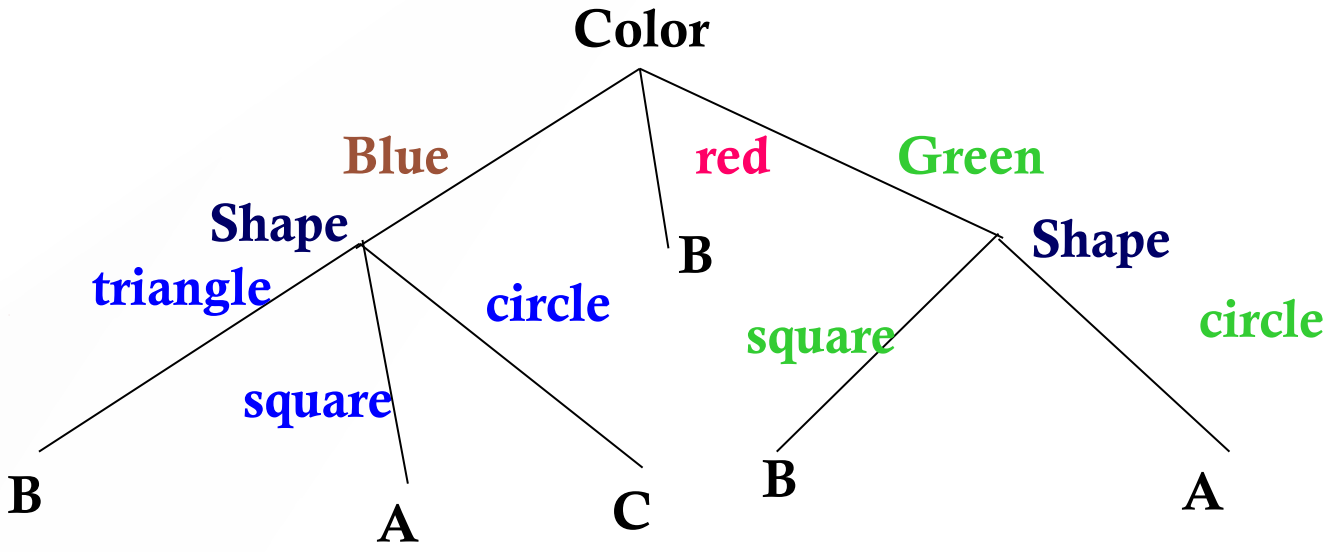
Decision Tree as Rules

- Color = blue and Shape = square → A
- Color= green and Shape = circle → A
- Color = red → B
- Color = green and Shape = square → B
- Color= blue and Shape = triangle → B
- Color= blue and Shape = circle→ C



Decision Trees as DNFs

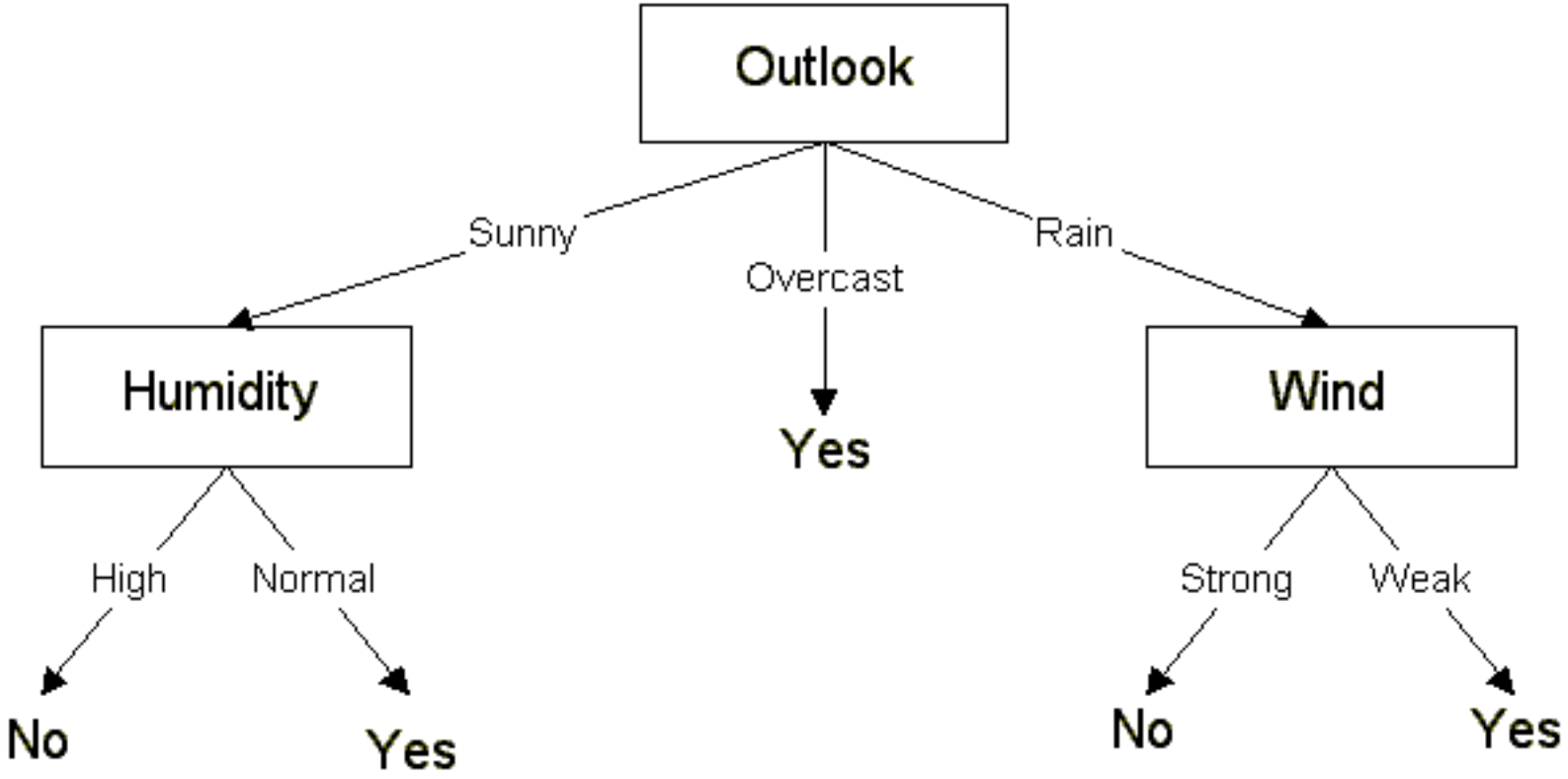
- **Class A**: (Color = blue and Shape = square) OR (Color= green and Shape = circle)
- **Class B**: (Color= blue and Shape = triangle) OR (Color = red) OR (Color = green and Shape = square)
- **Class C**: Color= blue and Shape = circle



A DT for PlayTennis (Mitchell)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

A DT for PlayTennis



The vertebrate example

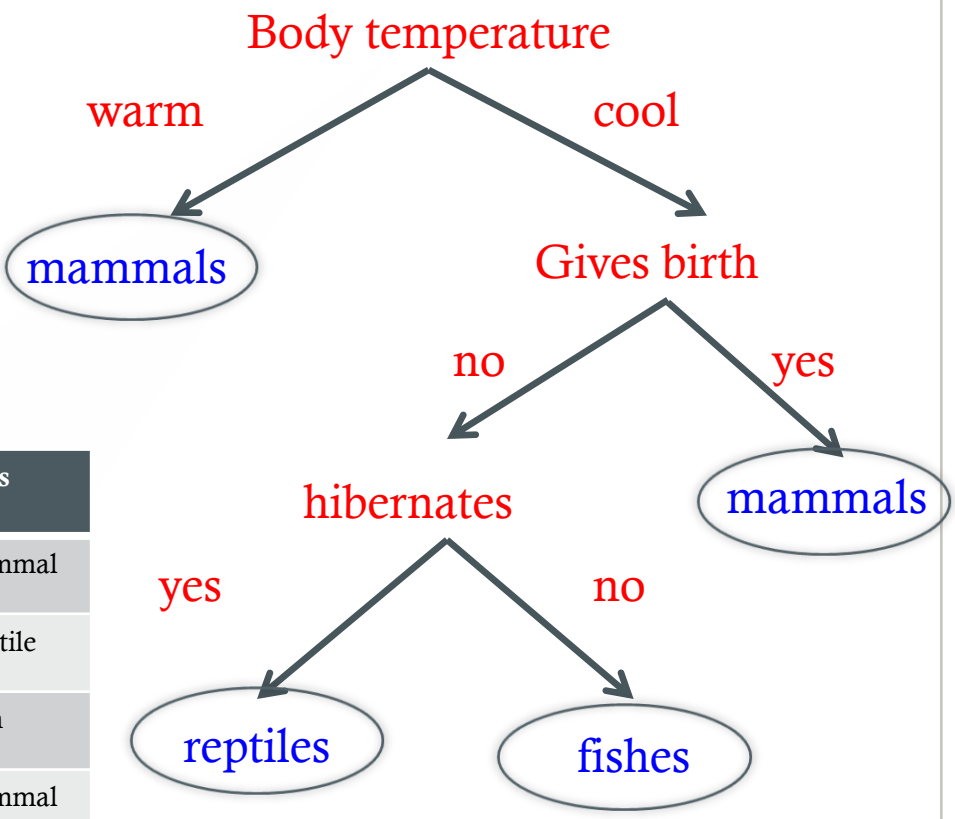
The vertebrate data set

Name	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	class
human	Warm	No	No	Yes	No	Yes	mammals
Python	Cool	No	No	No	Yes	No	Reptiles
salmon	Cool	Yes	No	No	No	No	Fishes
Whale	Cool	Yes	No	No	No	Yes	mammals

More decision trees may represent the training data

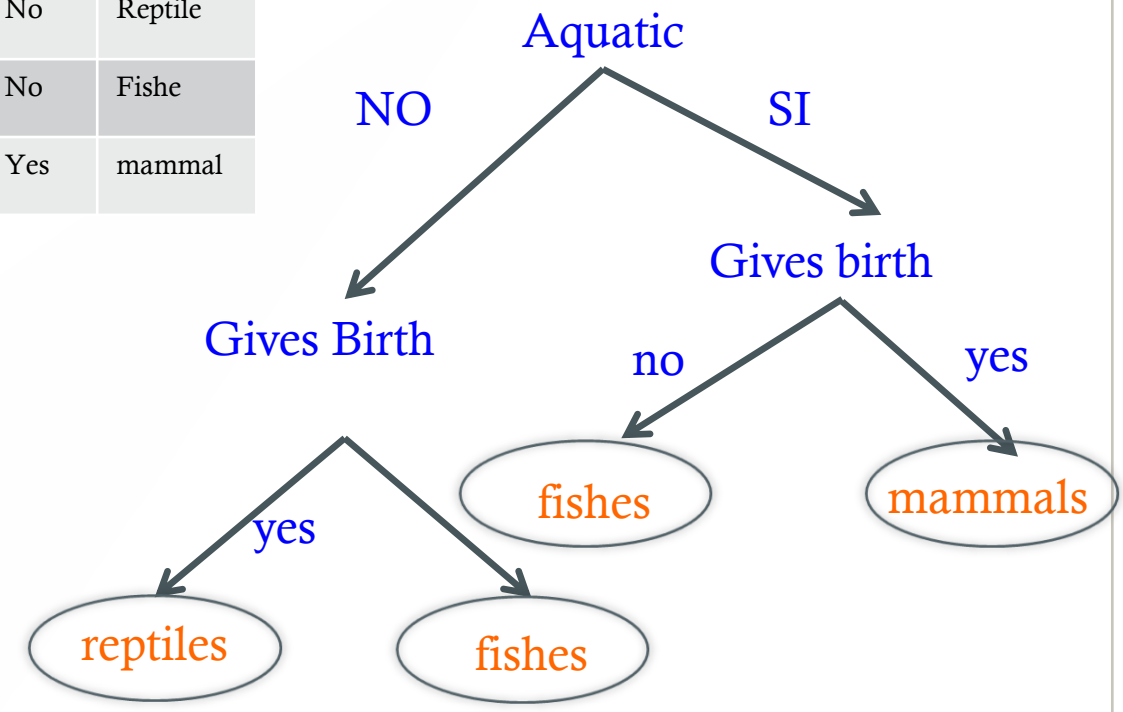
DTs for the vertebrate example

Name	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	Reptile
salmon	Cool	Yes	No	No	No	No	Fish
Whale	Cool	Yes	No	No	No	Yes	mammal



DTs for the vertebrate example

Name	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	Reptile
salmon	Cool	Yes	No	No	No	No	Fishe
Whale	Cool	Yes	No	No	No	Yes	mammal



Properties of DT's

- The space of decision trees is complete, i.e., there is at least one decision tree compatible with the training data (DTs are DNFs!)
- We might build a DT simply by representing each example – useless like DNF generated by CE!!

The Concept Learning System (**CLS**) Algorithm

CLS(Examples, Attributes):

{ create a Root node

If S contains only one class **C** **/* EXIT CONDITION 1**

return the single-node tree Root labeled **C**

If A empty **/* EXIT CONDITION 2 – the generated DT may not be a model**

return the single-node tree Root with label= most common class in Examples

Select an attribute **a** ∈ **A**; label Root by **a**; /* **a** is non-deterministically chosen

for each value **v** of **a** for which there is an example in **S**:

1- **S**:= set of examples in **Examples** with value **v** of **a**; **A := Attributes** \ {**a**};

2 - create an arc labeled by **v** and

- if S is empty attach a leaf node with label= most common class in Examples

- else attach subtree given by CLS **(S,A)**;

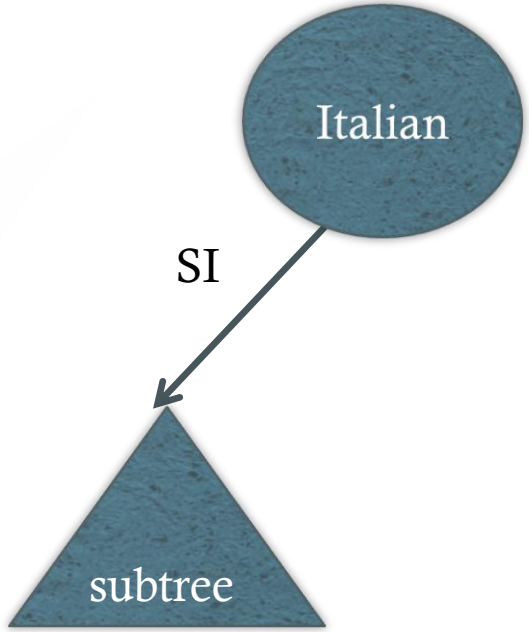
Student example

Id	Avg grade	Age	Italian	Sex	Passed DM Exam
1	A	D	SI	F	SI
2	B	D	SI	M	SI
3	A	E	NO	F	SI
4	C	E	SI	M	SI
5	C	E	NO	M	NO
6	C	E	NO	F	NO

- A: Avg grade >27; B: 22 ≤ Avg grade ≤ 27; C: Avg grade < 22
- D: age ≤ 22; E: age > 22
- Target function: the student has passed the data mining exam

CLS - A DT for the student example

Create a root node
…. Exit conditions fail
Select an attribute **a** ∈ **A**; label Root by **a**;
/* Select a= italian
for each value **v** of **a** for which there is an example in **S**
/* Select v= SI
1- **S**:= set of examples in **S** with **v** = **a**;
2- A := A \ {a};
/* A={Avg grade, age, sex}
3- create arc labeled **v** and subtree CLS **(S,A)**;



Id	Avg grade	Age	Italian	Sex	Passed
1	A	D	SI	F	SI
2	B	D	SI	M	SI
4	C	E	SI	M	SI

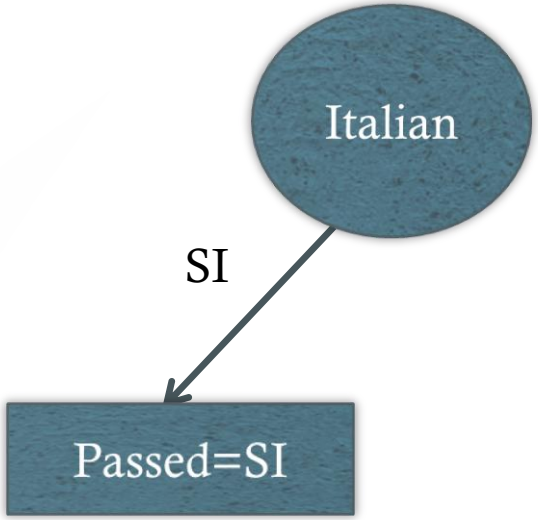
Set of examples with Italian = SI

CLS - A DT for the student example

create a Root node;

If S contains only one class /* Exit condition 1 occurs

return the single-node tree Root labeled **C**

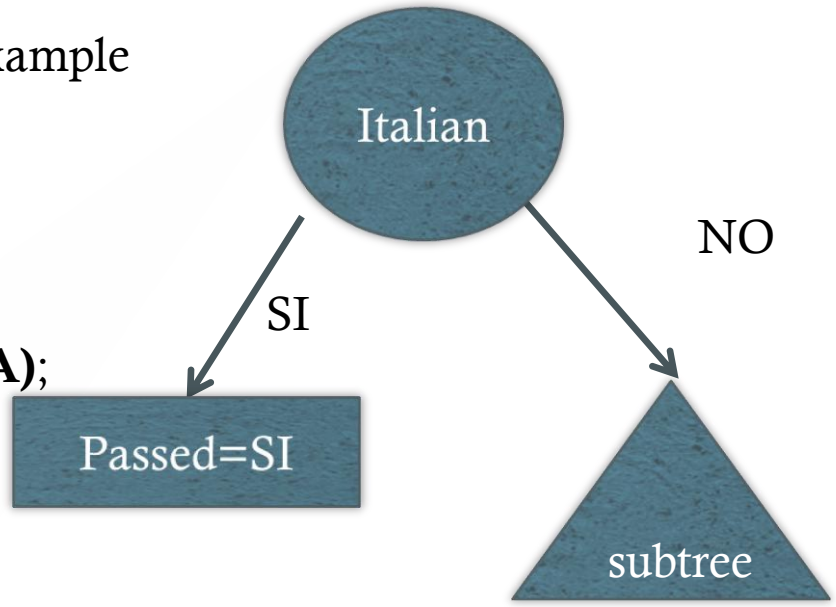


Id	Avg grade	Age	Italian	Sex	Passed
1	A	D	SI	F	SI
2	B	D	SI	M	SI
4	C	E	SI	M	SI

Set of examples with Italian = SI

CLS - A DT for the student example

Select an attribute **a** ∈ **A**; label Root by **a**;
/* a= Italian selected
for each value **v** of **a** for which there is an example in **S**:
/* Select v= NO
1- **S**:= set of examples in **S** with **v** = **a**;
2- A := A \ {a};
/* A={Avg grade, age, sex}
3- create arc labeled **v** and subtree CLS **(S,A)**;

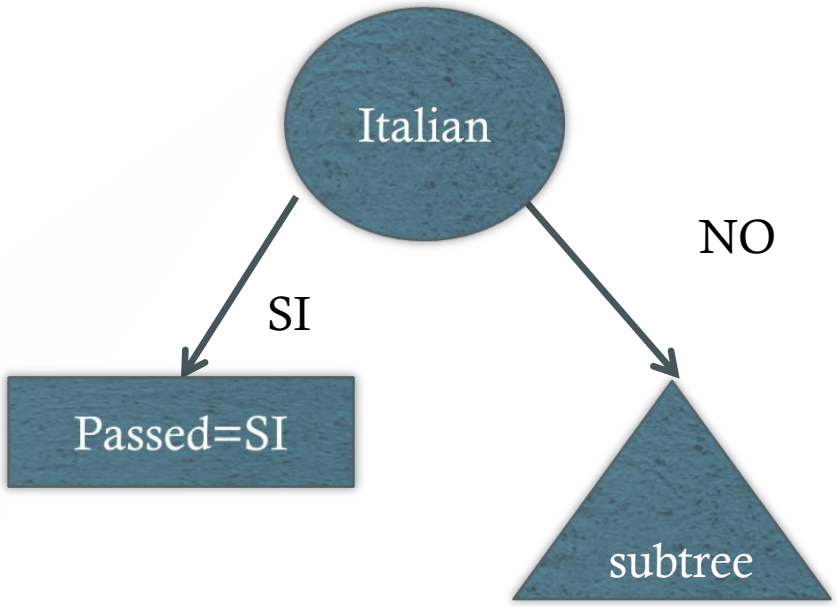


Id	Avg grade	Age	Italian	Sex	Passed
3	A	E	NO	F	SI
5	C	E	NO	M	NO
6	C	E	NO	F	NO

Set of examples with Italian = NO

CLS - A DT for the student example

Exit conditions fail



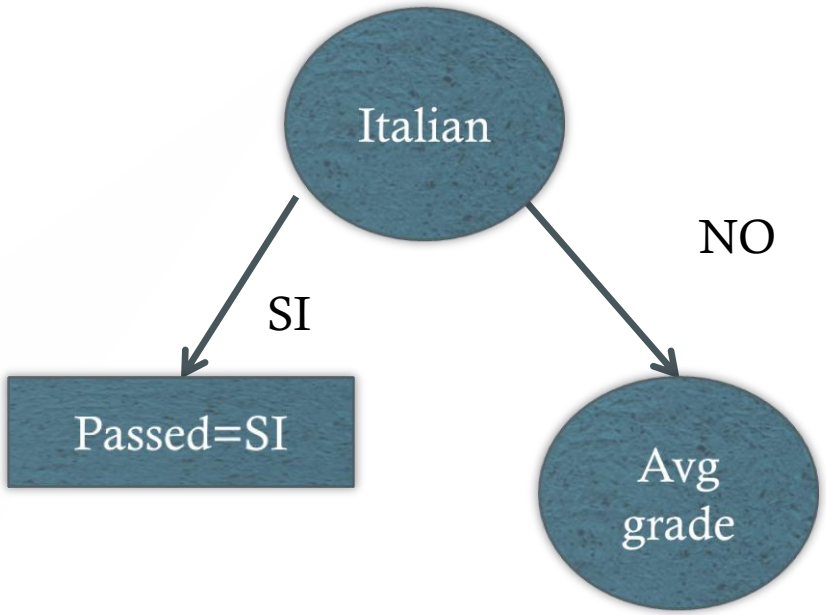
Id	Avg grade	Age	Italian	Sex	Passed
3	A	E	NO	F	SI
5	C	E	NO	M	NO
6	C	E	NO	F	NO

Set of examples with Italian = NO

CLS - A DT for the student example

Exit conditions fail

Select an attribute **a** ∈ **A**; label Root by **a**;
/* select a= avg grade



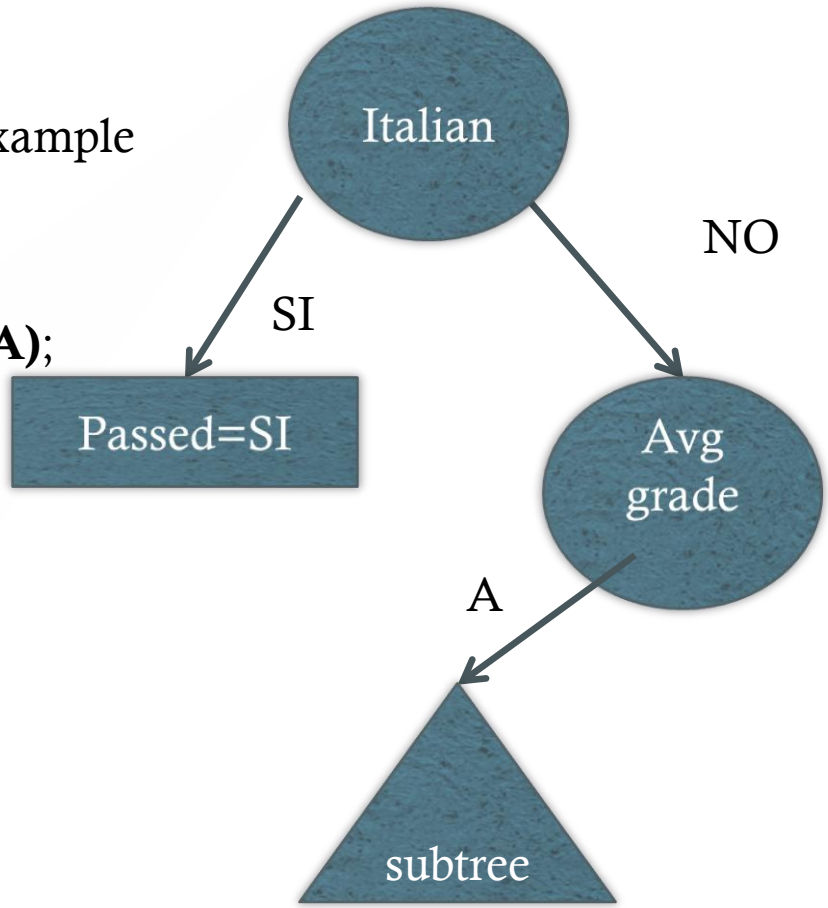
Id	Avg grade	Age	Italian	Sex	Passed
3	A	E	NO	F	SI
5	C	E	NO	M	NO
6	C	E	NO	F	NO

Set of examples with Italian = NO

CLS - A DT for the student example

Exit conditions fail

Select an attribute **a** ∈ **A**; label Root by **a**;
/* select a= avg grade
for each value **v** of **a** for which there is an example in **S**: /* v= A
1- **S**:= set of examples in **S** with **v** = **a**;
2- A := A \ {a}; /* A={age, sex}
3- create arc labeled **v** and subtree CLS **(S,A)**;



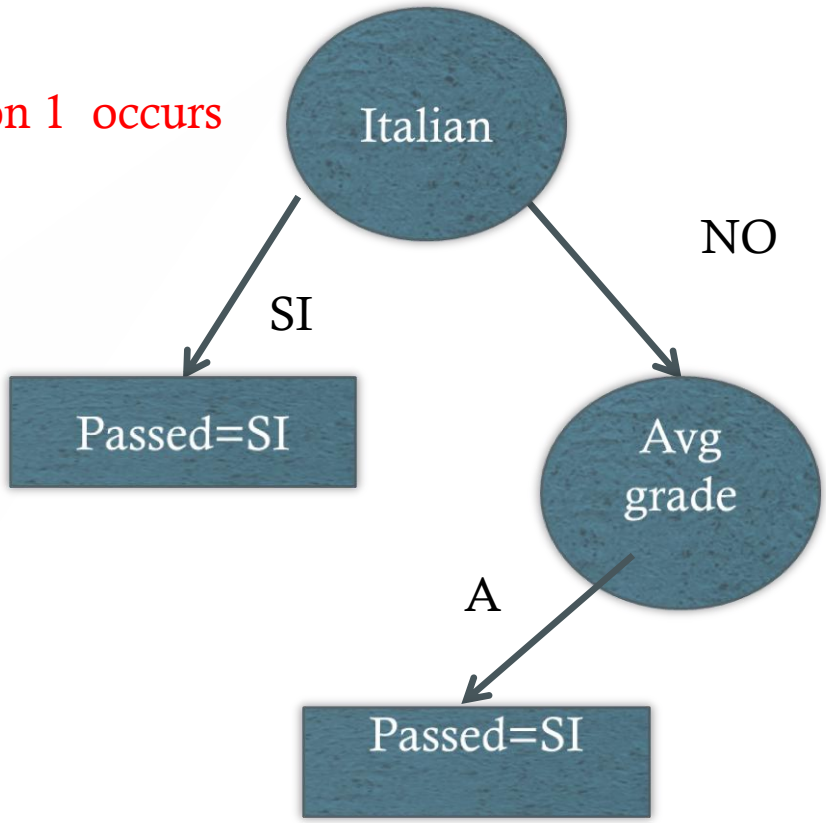
Id	Avg grade	Age	Italian	Sex	Passed
3	A	E	NO	F	SI

CLS - A DT for the student example

create a Root node;

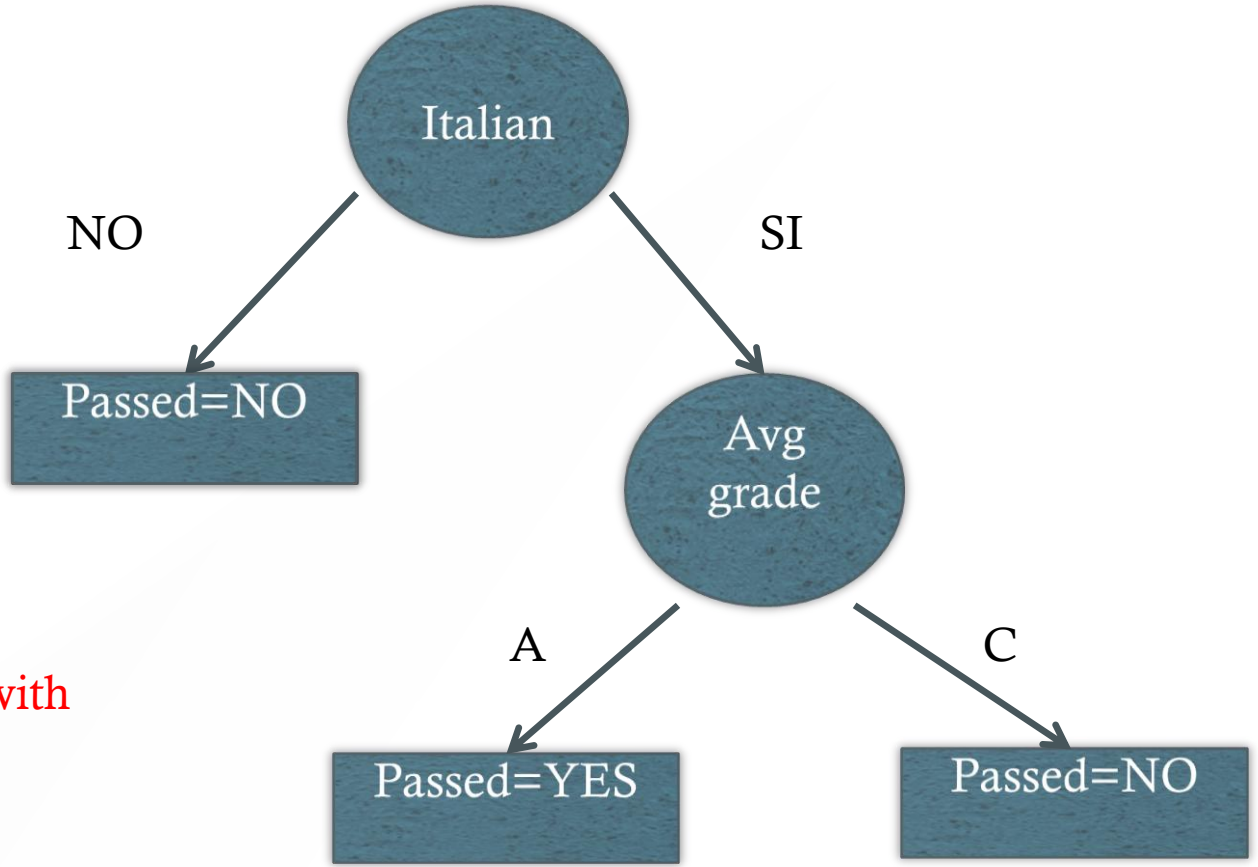
If S contains only one class /* Exit condition 1 occurs

return the single-node tree Root labeled **C**



Id	Avg grade	Age	Italian	Sex	Passed
3	A	E	NO	F	SI

CLS - A DT for the student example



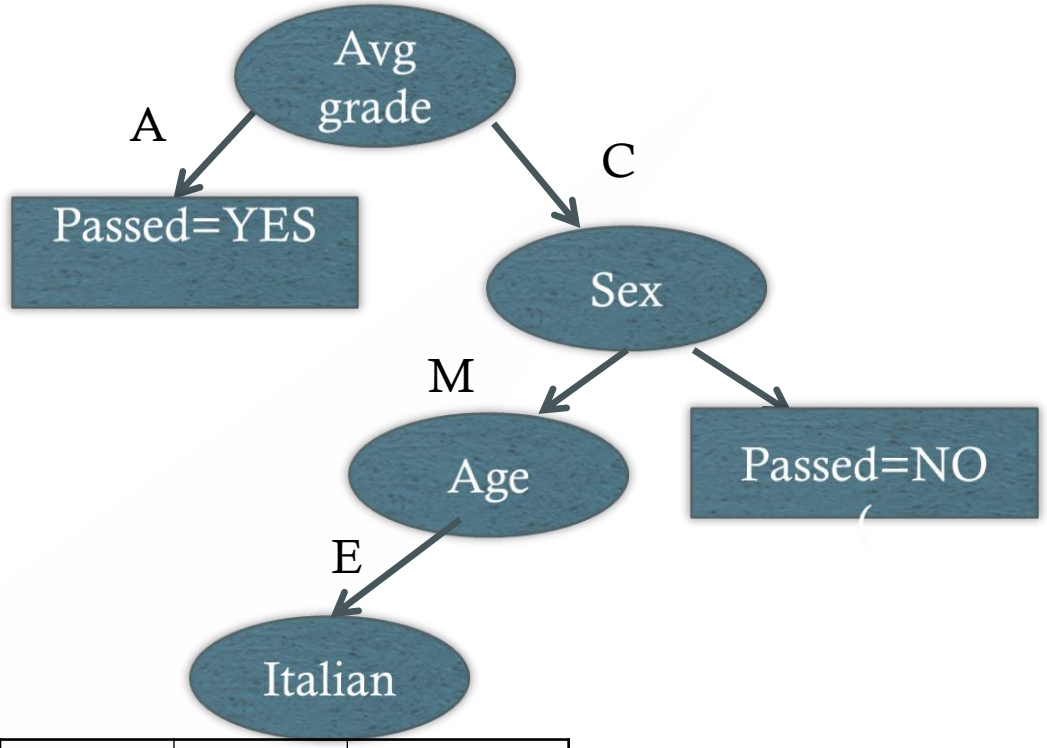
DT compatible with the training data

Student example

Id	Avg grade	Age	Italian	Sex	Passed DM Exam
1	A	D	SI	F	SI
2	B	D	SI	M	SI
3	A	E	NO	F	SI
4	C	E	SI	M	SI
5	C	E	SI	M	NO
6	C	E	NO	F	NO

- **A**: Avg grade >27; **B**: 22 ≤ Avg grade ≤ 27; **C**: Avg grade < 22
- **D**: age ≤ 22; **E**: age > 22
- **Target function**: the student has passed the data mining exam

CLS - Another DT for the student example

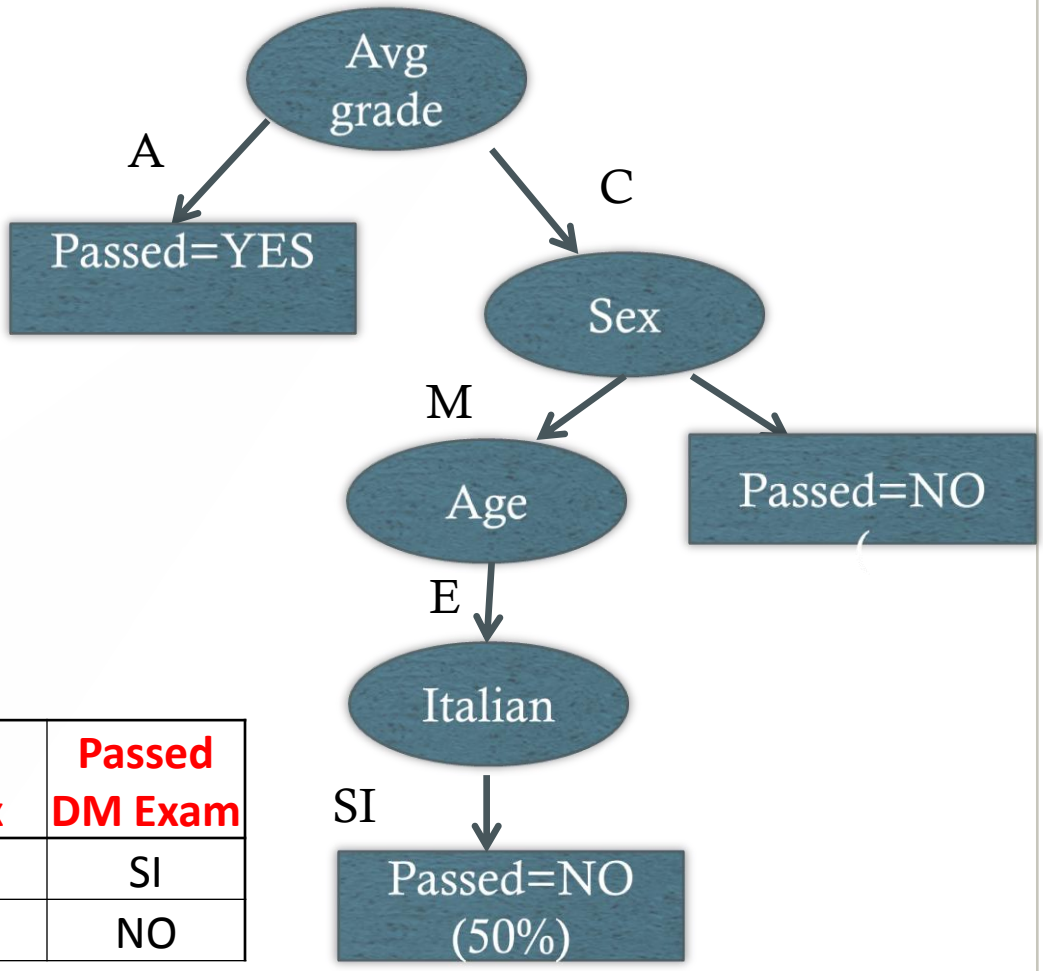


Id	Avg grade	Age	Italian	Sex	Passed DM Exam
4	C	E	SI	M	SI
5	C	E	SI	M	NO

CLS - Another DT for the student example

EXIT condition 2 occurs

- Not compatible with the samples
- Different choices of attributes generate different DTs



Id	Avg grade	Age	Italian	Sex	Passed DM Exam
4	C	E	SI	M	SI
5	C	E	SI	M	NO

The Concept Learning System (**CLS**) Algorithm

CLS(Examples, Attributes):

{ create a Root node

If S contains only one class **C** **/* EXIT CONDITION 1**

return the single-node tree Root labeled **C**

If A empty **/* EXIT CONDITION 2 – the generated DT may not be a model**

return the single-node tree Root with label= most common class in Examples

Select an attribute **a** ∈ **A**; label Root by **a**; /* **a** is non-deterministically chosen

for each value **v** of **a** for which there is an example in **S**:

1- **S**:= set of examples in **Examples** with value **v** of **a**; **A := Attributes** \ {**a**};

2 - create an arc labeled by **v** and

- if S is empty attach a leaf node with label= most common class in Examples

- else attach subtree given by CLS **(S,A)**;

Stopping Conditions

- Stopping condition for a leaf node N:
  1. The examples associated with N have all the same target class
  2. There is no more attribute whereby splitting the samples
- **NOTE**: if condition 2 happens, then the DT may not be a model of the training data

Algorithm CLS

- CLS searches through the attributes of the training instances and non-deterministically extracts one attribute.
- The algorithm never looks back to reconsider earlier choices (no backtracking).

The ID3 Algorithm

ID3(Examples, Attributes):

{ create a Root node

If S contains only one class **C** **/* EXIT CONDITION 1**

return the single-node tree Root labeled **C**

If A empty **/* EXIT CONDITION 2 – the generated DT may not be a model**

return the single-node tree Root with label= most common class in Examples

Select the best **a** ∈ **A**; label Root by **a** **/* what is the best attribute? Use Information Gain**

for each value **v** of **a** for which there is an example in **S**:

1- **S**:= set of examples in **Examples** with value **v** of **a**; **A := Attributes** \ {**a**};

2 - create an arc labeled by **v** and

- if S is empty attach a leaf node with label= most common class in Examples

- else attach subtree given by ID3**(S,A)**;

Algorithm ID3

- ID3 searches through the attributes of the training instances and extracts the attribute that best separates the given examples.
- Greedy, general-to-specific, recursive partitioning strategy which tends to provide the "smallest" DT; the output is not necessarily consistent with the training data
- We are interested in "small" DT's (why? Overfitting – Occam's razor) that well fit the training data

Some properties of ID3

- Efficient: worst case $O(dn^2)$ for training, $\approx O(\log n)$ for classification (d=number of attributes, n=number of examples)
- However, learned tree is rarely complete (number of leaves is $\leq n$). In practice, complexity is linear in both number of features (m) and number of training examples (n).

How does ID3 choose the best attribute?

- Entropy
  - A concept from thermodynamics and information theory
  - Initially used to compress signals
  - But more recently used for Data Mining

Entropy

- It can be used as a measure of the degree of disorder of the training data
- Thus, if all examples are in the same class (maximum order), the entropy is zero
- On the contrary, if the examples are evenly distributed across the classes, the entropy is maximum

Entropy

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

Entropy

- S consists of 14 examples, 9 positive and 5 negative (S = [9+,5-])
  - $E(S) = - p_+ \log p_+ - p_- \log p_- = -9/14 \log 9/14 – 5/14 \log 5/14 = 0.94$
- If S=[7+,7-]
  - $E(S) = ½ \log ½ - ½ \log ½ = 1$
- If S=[14+,0-]
  - $E(S) = 1 \log 1 – 0 \log 0 = 0$ (0 log 0 is set to 0)

Entropy

- More generally, if there are more c classes

$$E(S) = \Sigma_{i=1,c} - p_i \log p_i$$

- where pi is the proportion of S belonging to class i

Information Gain

- Information Gain IG(S,A) is the expected reduction in entropy caused by partitioning the examples of S according to attribute A
- $$IG(S,A) = E(S) - \Sigma_{v \text{ in values}(A)} \frac{|S_v|}{|S|} E(S_v)$$
- where S_v is the subset of S where A=v

Information Gain - Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

- S = [9+,5-]
- E(S) = - 9/14 log 9/14 - 5/14 log 5/14 = 0.94

A = wind, Values(A) = {weak, strong}

				wind	
D1	Sunny	Hot	High	Weak	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D2	Sunny	Hot	High	Strong	No
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D14	Rain	Mild	High	Strong	No

Information Gain - Example

- A = wind, Values(A) = {weak, strong}
- S_{weak} = [6+,2-], S_{strong} = [3+,3-]
- $E(S_{weak}) = - 6/8 \log 6/8 – 2/8 \log 2/8 = 0.81$
- $E(S_{strong}) = -3/6 \log 3/6 – 3/6 \log 3/6 = 1.00$

$$IG(S,wind) = E(S) – 8/14\ E(S_{weak}) – 6/14\ E(S_{strong})$$

$$IG(S,wind) = 0.94 - 8/14 * 0.81 – 6/14 * 1 = 0.048$$

- Information Gain IG(S,A) is the expected reduction in entropy caused by partitioning the examples of S according to attribute A
- The more discriminating A the higher IG

A = wind, Values(A) = {weak, strong}

				Wind	
D1	Sunny	Hot	High	Weak	Yes
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	Yes
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D2	Sunny	Hot	High	Strong	No
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	No
D11	Sunny	Mild	Normal	Strong	No
D12	Overcast	Mild	High	Strong	No
D14	Rain	Mild	High	Strong	No

Information Gain - Example

- A = wind, Values(A) = {weak, strong}
- S_{weak} = [8+,0-], S_{strong} = [0+,6-]
- $E(S_{weak}) = 0$
- $E(S_{strong}) = 0$

$$IG(S,wind) = E(S) - 0 - 0$$

$$IG(S,wind) = 0.94$$

- IG is a measure of the discriminating power of an attribute
- The higher its discriminating power the higher IG

Information Gain - Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

- S = [9+,5-]
- E(S) = - 9/14 log 9/14 - 5/14 log 5/14 = 0.94

Information Gain - Example

- A=Humidity, Values(A) = {high, normal}
- S_{high}= [3+,4-], S_{normal}=[6+,1-]
- $E(S_{high}) = 0.985$, $E(S_{normal}) = 0.592$

$$IG(S,Hum) = 0.94 - 7/14\ E(S_{high}) - 7/14\ E(S_{normal})$$
$$=0.151$$

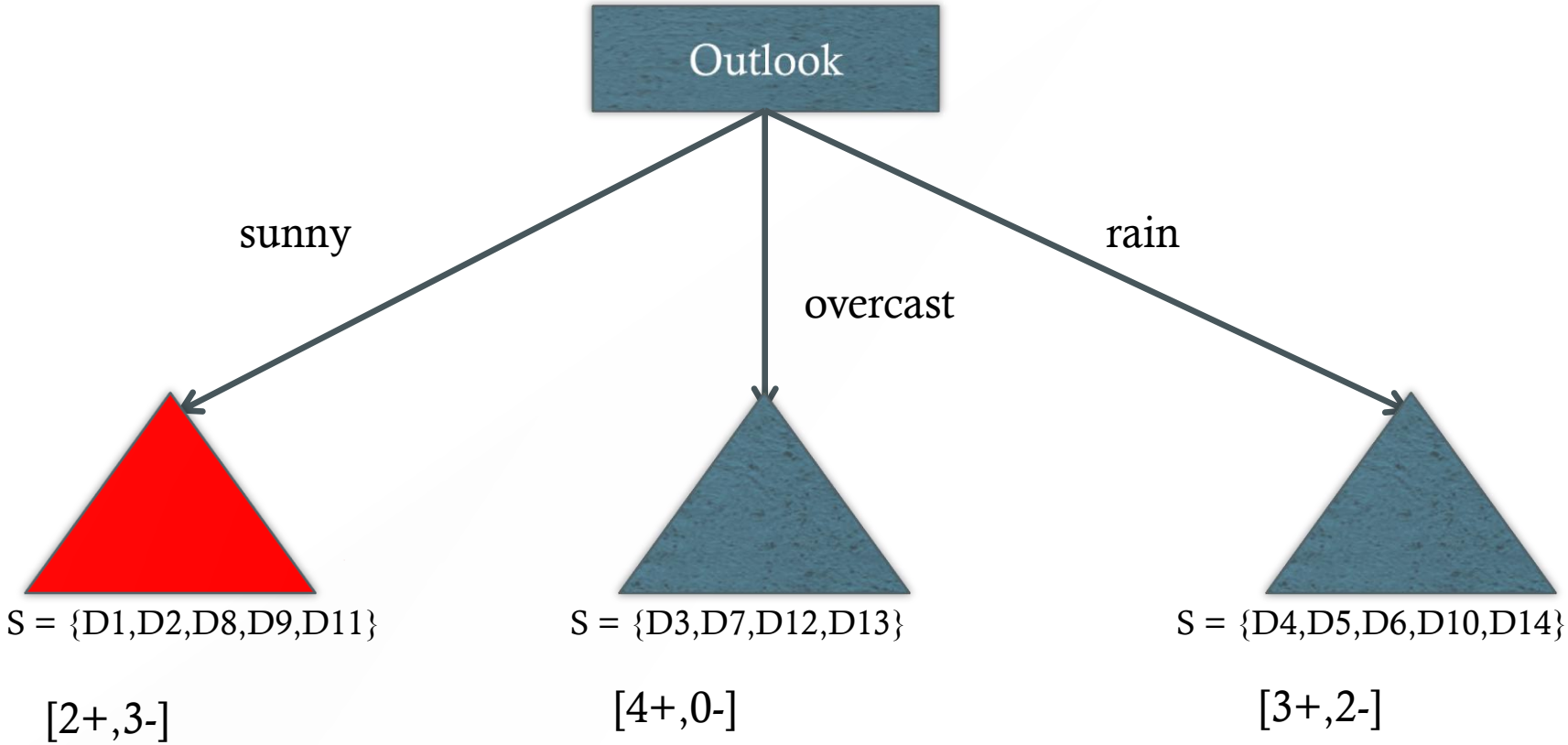
Information Gain - example

- IG(S,humidity) > IG(S,wind)
- The attribute Humidity induces a larger reduction in entropy than the attribute Wind
- The IG values for all attributes of the PlayTennis training set id
  - IG(PlayTennis,Outlook) = 0.246
  - IG(PlayTennis, Humidity) = 0.151
  - IG(PlayTennis, Wind) = 0.048
  - IG(PlayTennis,Temperature) = 0.029

Information Gain - example

- Algorithm ID3 selects as the best node the one having the highest IG
- *Outlook* is chosen by ID3 as the best node

ID3 and IG


Outlook
sunny
overcast
rain
S = {D1,D2,D8,D9,D11}
[2+,3-]
S = {D3,D7,D12,D13}
[4+,0-]
S = {D4,D5,D6,D10,D14}
[3+,2-]

Information Gain - Example

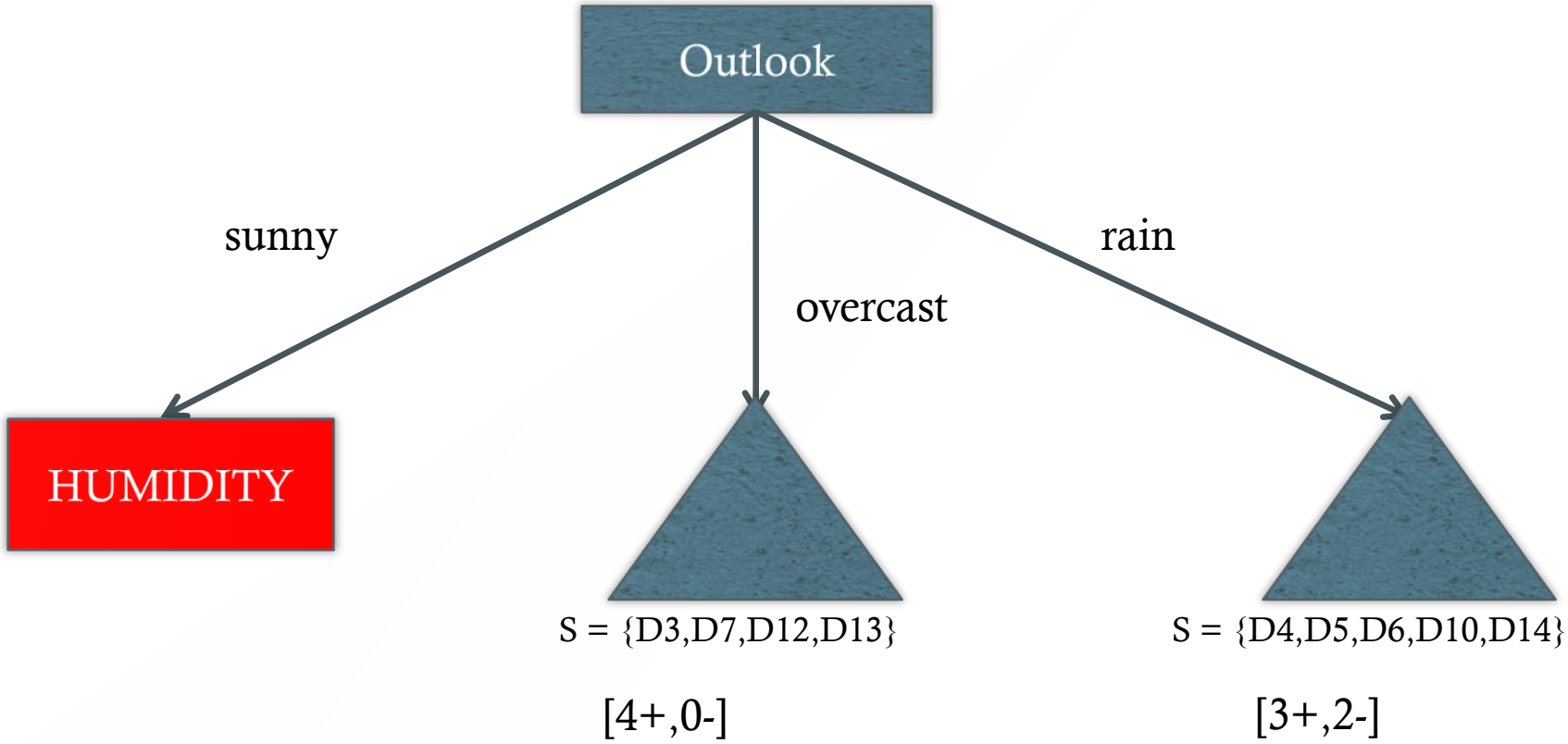
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes

- $S_{sunny} = \{D1, D2, D8, D9, D11\}$
- $S = [2+,3-]$
- $E(S_{sunny}) = - 2/5 \log 2/5 – 3/5 \log 3/5 = 0.97$

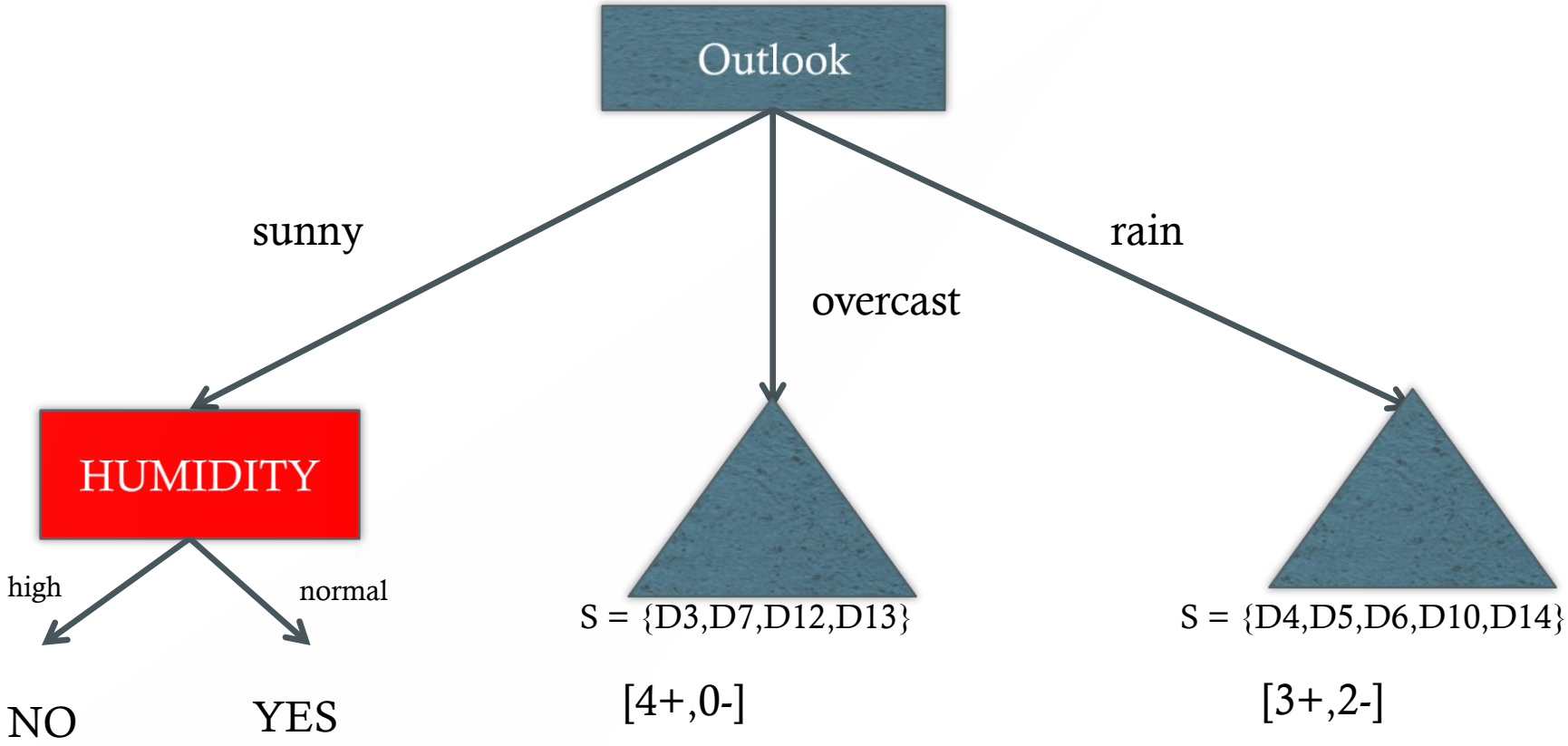
ID3 and IG

- The next question is "what attribute should be tested at the Sunny branch node?" Since we have used Outlook at the root, we only decide on the remaining three attributes: Humidity, Temperature, or Wind.
- S_{sunny} = {D1, D2, D8, D9, D11}
- IG(S_{sunny}, Humidity) = 0.970
- IG(S_{sunny}, Temperature) = 0.570
- IG(S_{sunny}, Wind) = 0.019

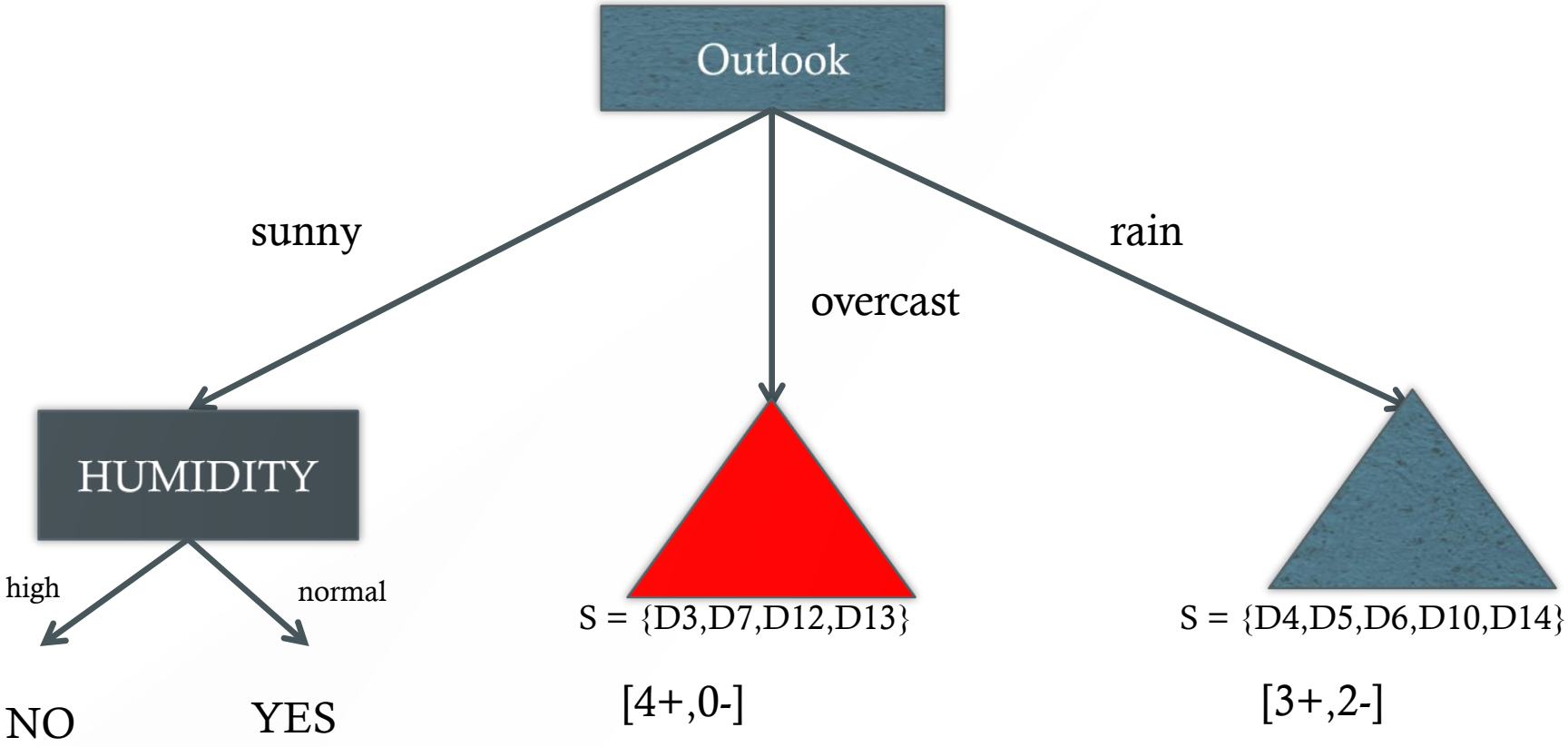
ID3 and IG


Outlook
sunny
overcast
rain
HUMIDITY
S = {D3,D7,D12,D13}
[4+,0-]
S = {D4,D5,D6,D10,D14}
[3+,2-]

ID3 and IG


Outlook
sunny
overcast
rain
HUMIDITY
high
normal
NO
YES
S = {D3,D7,D12,D13}
[4+,0-]
S = {D4,D5,D6,D10,D14}
[3+,2-]

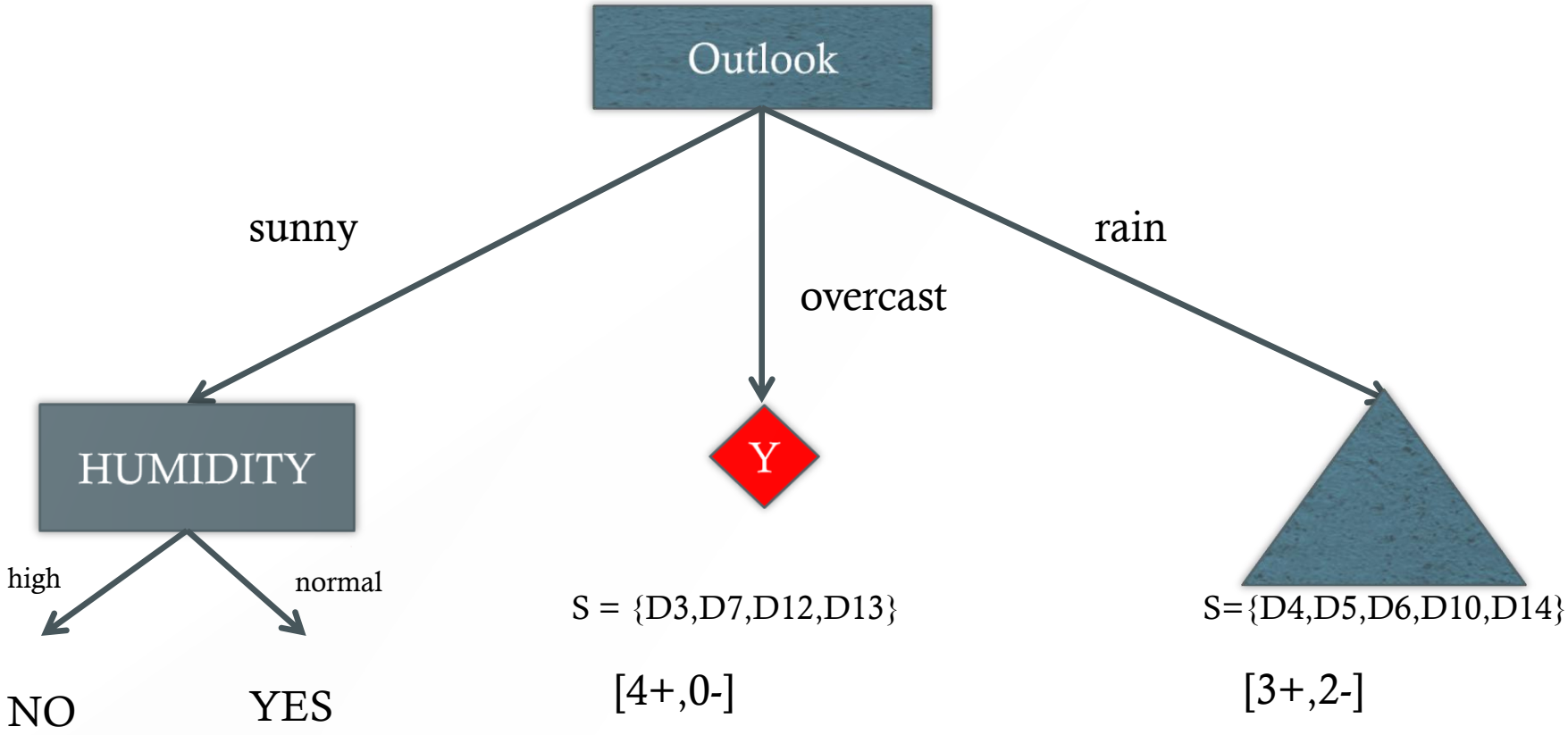
ID3 and IG


Outlook
sunny
overcast
rain
HUMIDITY
high
normal
NO
YES
S = {D3,D7,D12,D13}
[4+,0-]
S = {D4,D5,D6,D10,D14}
[3+,2-]

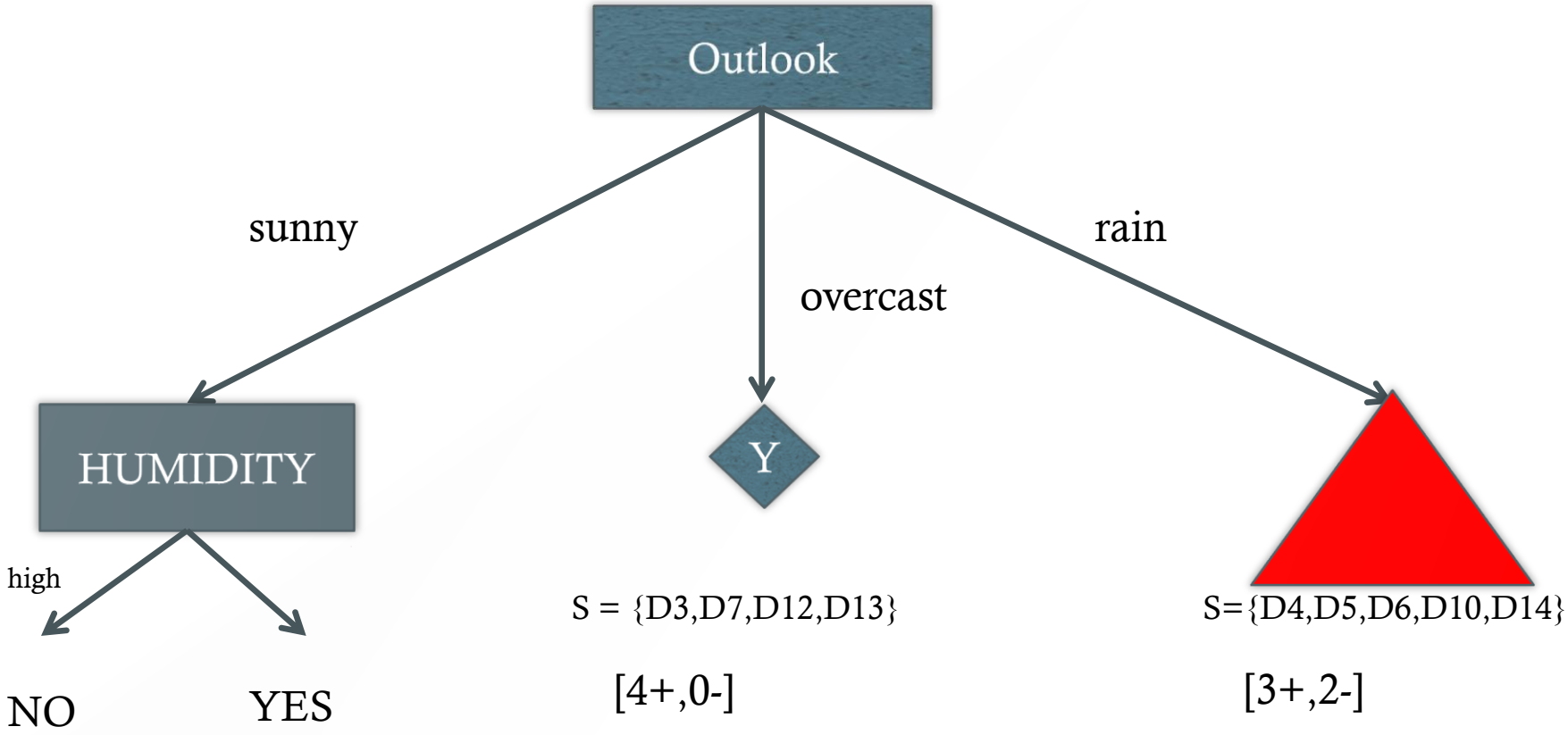
Information Gain - Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D3	Overcast	Hot	High	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes

ID3 and IG


Outlook
sunny
overcast
rain
HUMIDITY
Y
high
normal
NO
YES
S = {D3,D7,D12,D13}
[4+,0-]
S={D4,D5,D6,D10,D14}
[3+,2-]

ID3 and IG


Outlook
sunny
overcast
rain
HUMIDITY
Y
high
NO
YES
S = {D3,D7,D12,D13}
[4+,0-]
S={D4,D5,D6,D10,D14}
[3+,2-]

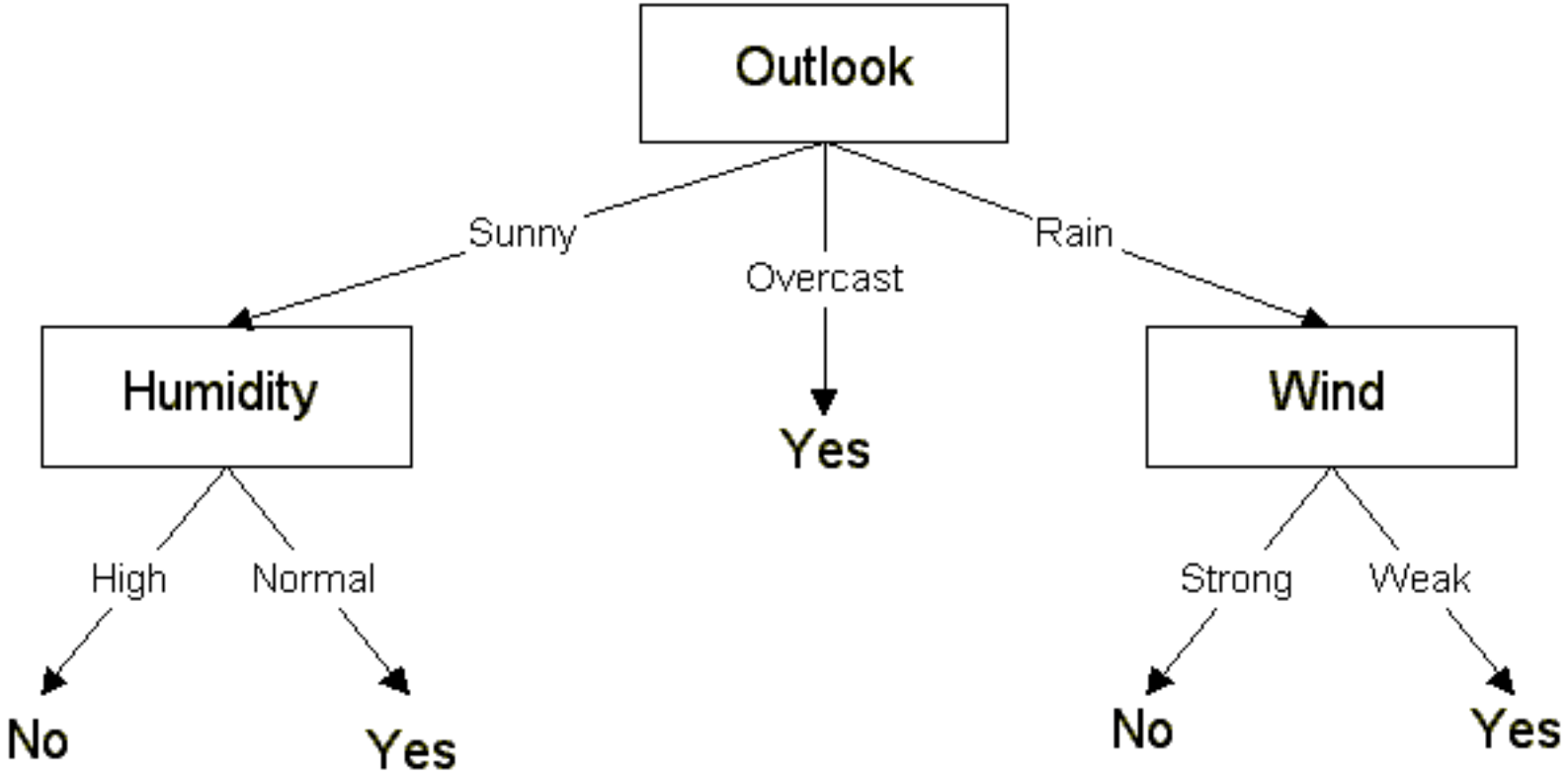
Information Gain - Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

- S = [3+,2-]
- $E(S_{rain}) = - 3/5 \log 3/5 – 2/5 \log 2/5 = 0.97$

- Compute the IG for S_{rain} w.r.t. Temperature, Humidity and Wind
- Select Wind as the best attribute

A DT for PlayTennis



The ID3 Algorithm

ID3(Examples, Attributes):

{ create a Root node

If S contains only one class **C** **/* EXIT CONDITION 1**

return the single-node tree Root labeled **C**

If A empty **/* EXIT CONDITION 2 – the generated DT may not be a model**

return the single-node tree Root with label= most common class in Examples

Select **a** ∈ **A** **which maximizes IG**; label Root by **a**

for each value **v** of **a** for which there is an example in **S**:

1- **S**:= set of examples in **Examples** with value **v** of **a**; **A := Attributes** \ {**a**};

2 - create an arc labeled by **v** and

- if S is empty attach a leaf node with label= most common class in Examples

- else attach subtree given by ID3**(S,A)**;

More on ID3

- The DT generated by ID3 is not in general compatible with the training data
- ID3 *incompletely* searches a *complete* HS
- CE *completely* searches an *incomplete* HS (conjunctions)

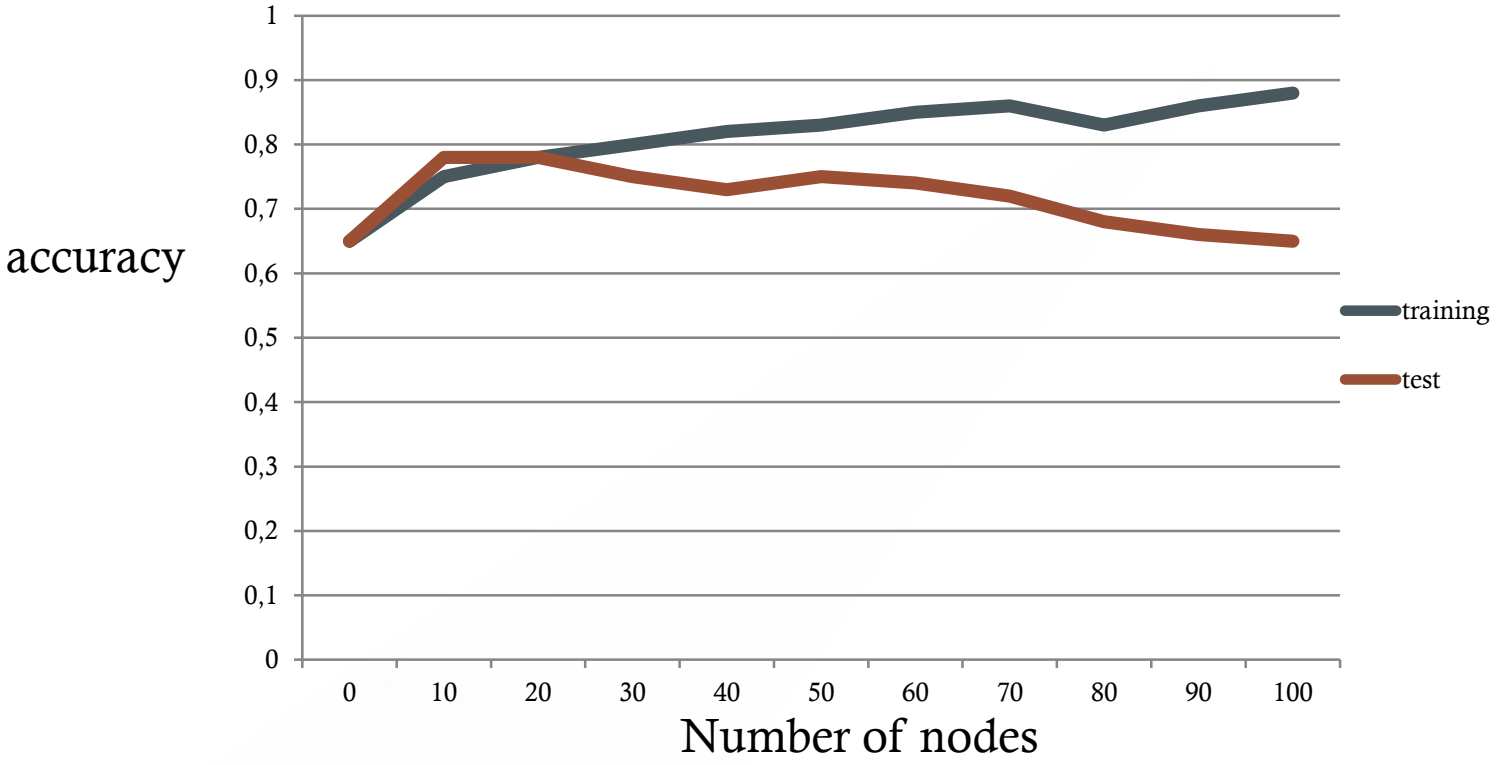
ID3 vs CE

- Inductive Bias:
  - Smaller DT's are preferred as a consequence of the IG-based strategy (at each step the attribute which best separates the examples is chosen). The IG metric provides support for balanced splitting. Smaller DT's encode more general DNF
  - DTs with attributes with highest IG closest to the root are generated
- Why smaller DT's should be preferable to larger ones? Overfitting

Overfitting (sovradattamento)

- When a model is to "hardwired" over the training data it tends to generalize poorly
- That is, learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization to unseen data.
- **Definition** (Mitchell): a hypothesis **h** is said to overfit the training data if there is another hypothesis **h'** such that **h** has a smaller error over the training examples, but **h'** has a smaller error over the test examples

DT and overfitting



- That is why smaller DT's aree preferable to larger ones

ID3 is robust to noise

- ID3 uses all training examples at each step to make statistical-based decisions (IG) on how to refine the current hypothesis
- This contrasts with Find-S or Candidate-Elimination which makes decisions based on single instances
- As a result, the search of ID3 is much less sensitive to errors in individual examples (noise)

Reducing the complexity of DT's

- **Pre-pruning**: the tree-growing algorithm is stopped before generating the full tree. To this end, new stopping conditions are used, e.g., stop if the IG of a leaf node is smaller than a given threshold
- **Post-pruning**: Grow the full tree and then remove nodes that seem not to have sufficient evidence

Post-pruning
Reduced-error Pruning

- Partition training data in "grow" and "validation" sets.
- Build a complete tree DT from the "grow" data.
- For each non-leaf node N in the tree do:
  - Temporarily prune the subtree below N and replace it with a leaf labeled with the current majority class at that node.
  - Measure the accuracy of the pruned tree DT' on the validation set
    - If acc(DT) <= acc(DT') then DT=DT'

Dealing with continuous-valued attributes

- Initial definition of ID3 is restricted in dealing with discrete sets of values. It handles symbolic attribute effectively.
- However, we have to extend its sphere to continuous-valued attributes (numeric attribute) to fit the real world scenario.
- One way to do that is to define new discrete valued attributes that partition the continuous-valued attribute into multiple intervals, each treated as a symbolic attribute.

Conclusions

- ID3 induces "small" DT's (that may not be fully compatible with the training data) by using a greedy, top-down recursive strategy based on the IG for selecting the best attribute at each step
- The tree –growing procedure can be seen as the process of partitioning the attribute space into disjoint regions until each region contains examples of the same class
- Small DT's generalize better and are less prone to overfitting
- DT's are a easy to interpret model representation
- ID3 is robust to the presence of noise in the training set

Exercise

Instance	A1	A2	class
1	T	T	+
2	T	T	+
3	T	F	-
4	F	F	+
5	F	T	-
6	F	T	-

- Entropy
- IG of A2
- DT