Decision trees

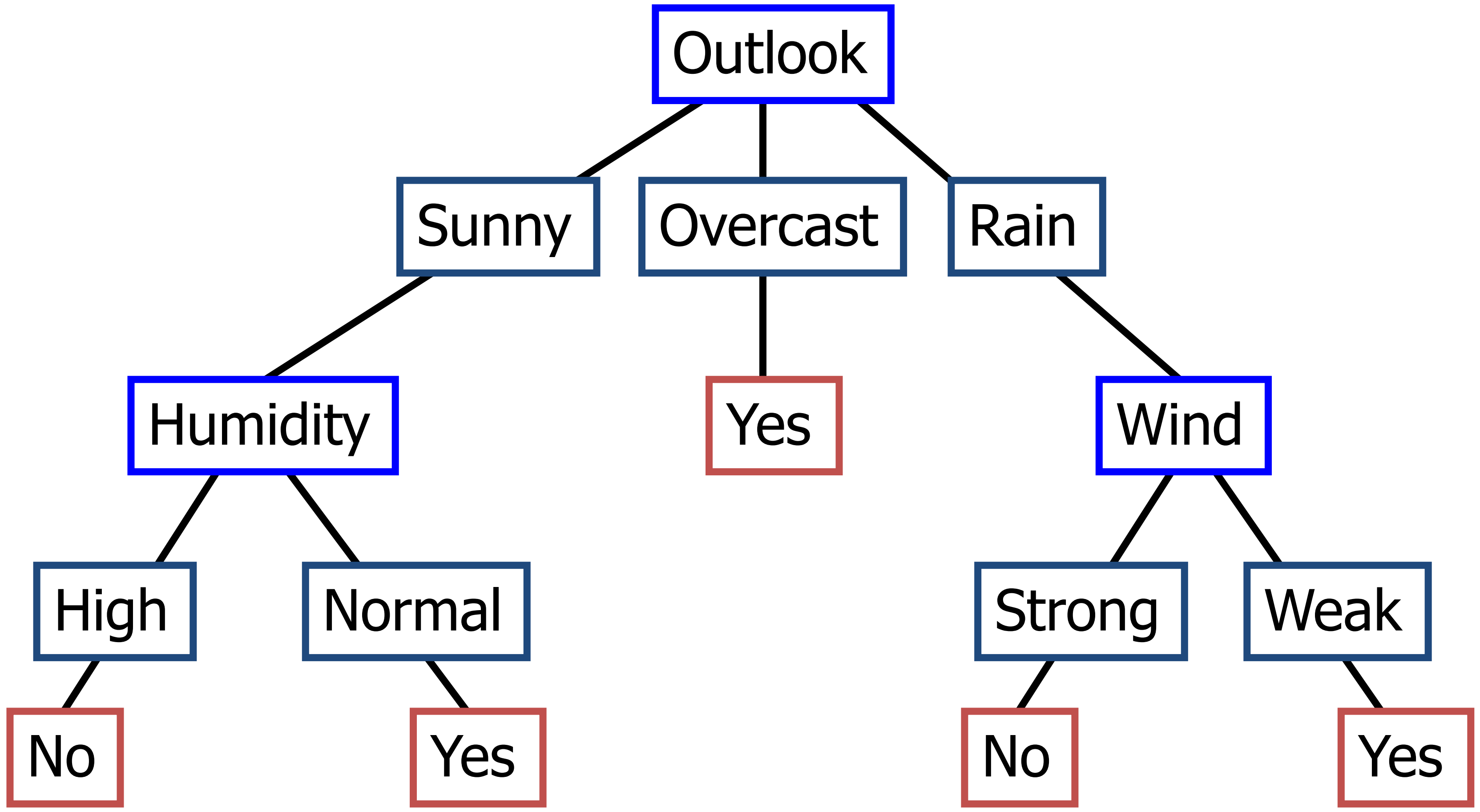
Machine Learning

Decision trees: table of contents

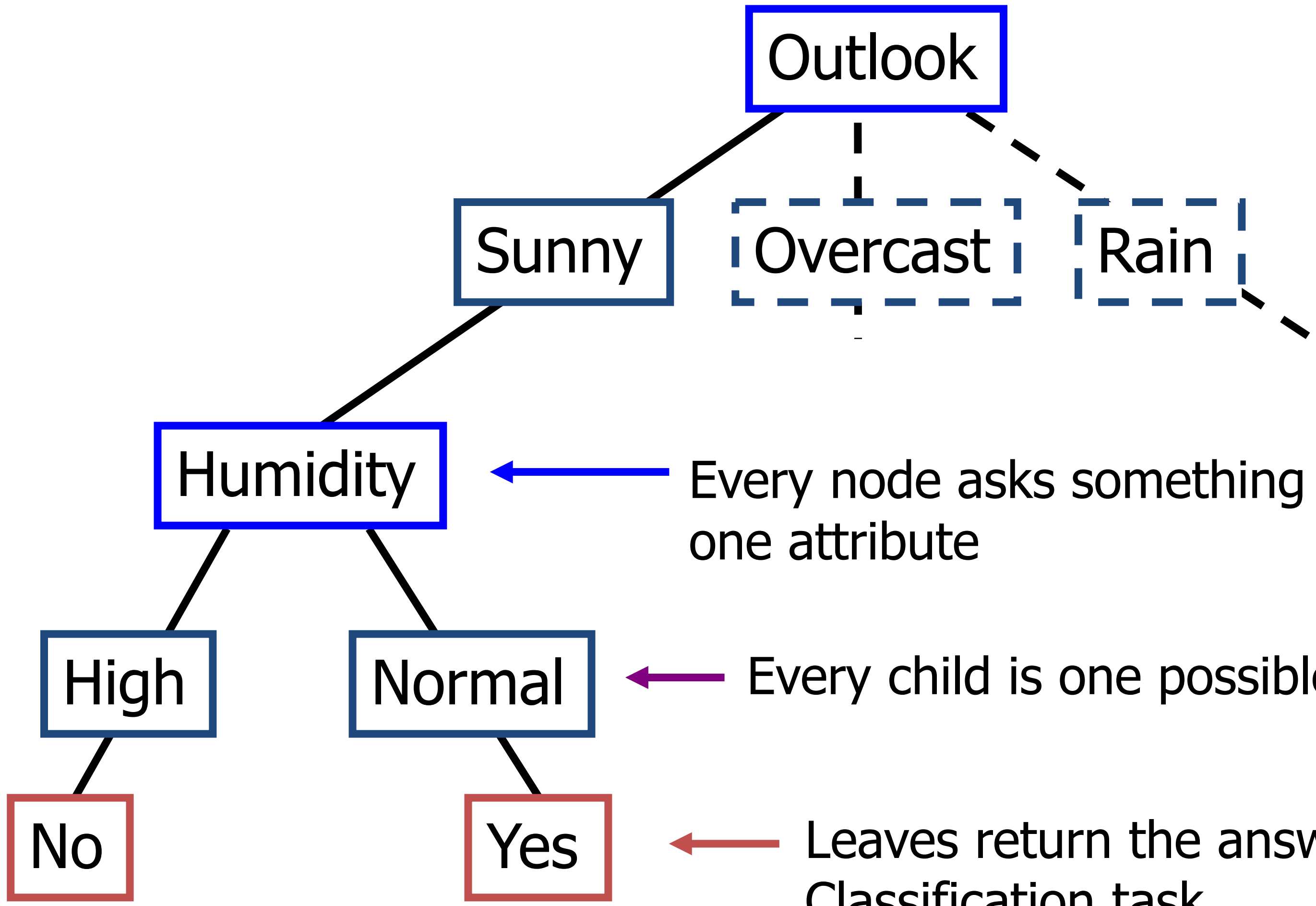
Toy example: “PlayTennis” dataset

Day	*Outlook*	*Temp.*	*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	Cold	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

Decision tree for "PlayTennis"



Decision tree for "PlayTennis"



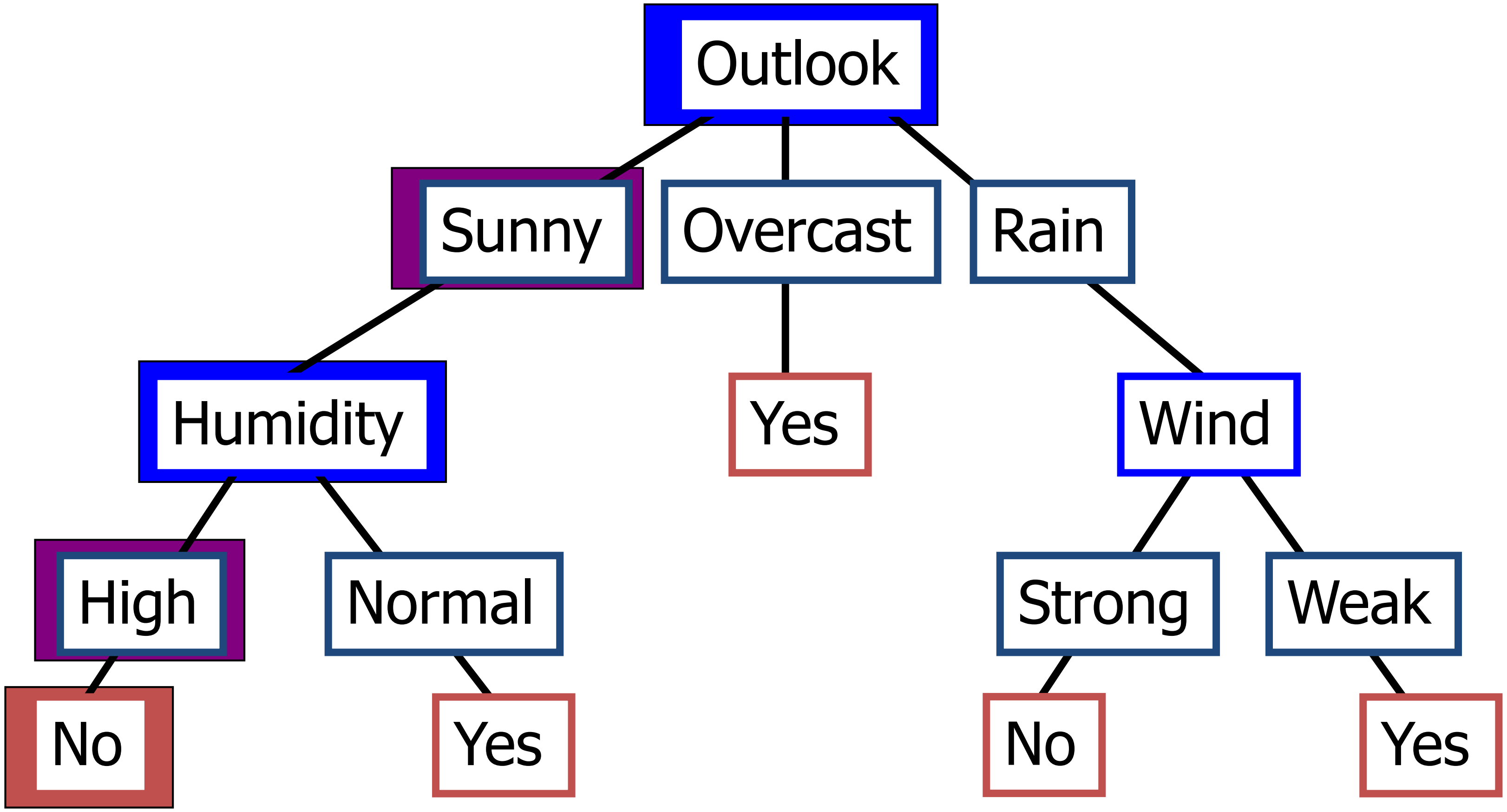
Every node asks something about one attribute

Every child is one possible answer

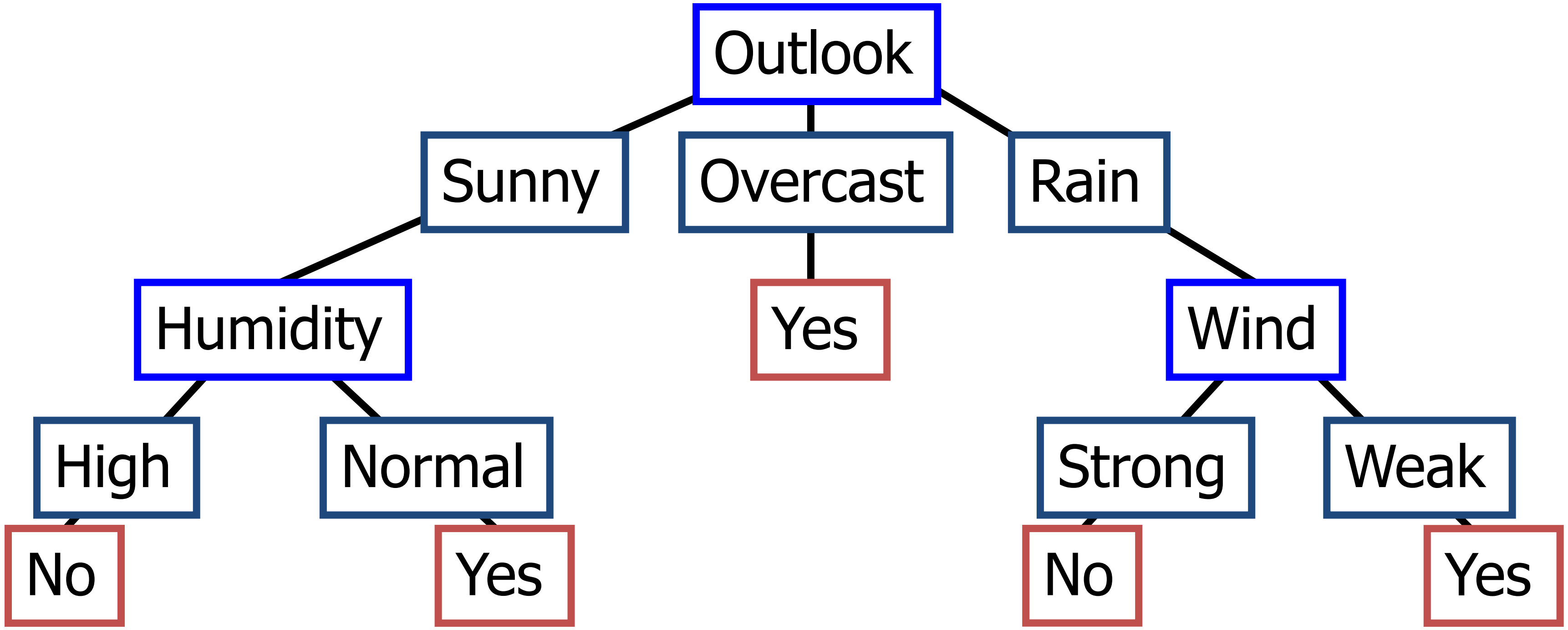
Leaves return the answer for the Classification task

Decision tree for “PlayTennis”

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	No



DT can represent complex rules



$$
\begin{aligned}
&(\text{Outlook=Sunny} \wedge \text{Humidity=Normal}) \\
\vee\ &(\text{Outlook=Overcast}) \\
\vee\ &(\text{Outlook=Rain} \wedge \text{Wind=Weak})
\end{aligned}
$$

Learning Decision Tres

- 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).

“Top-Down” induction of decision trees: ID3

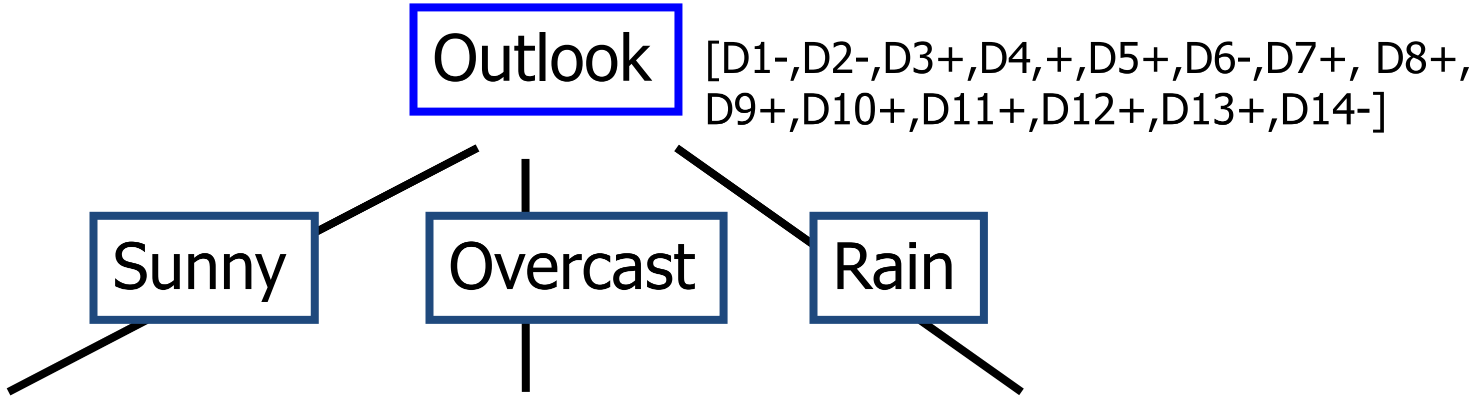
1. Set *node* to the training examples and put it as the root of the tree
2. A $\leftarrow$ the "best" test attribute for the current node
3. For each value of attribute A, create a new descendant node in the decision tree
4. Separate examples in *node* among descendants depending of the attribute value 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

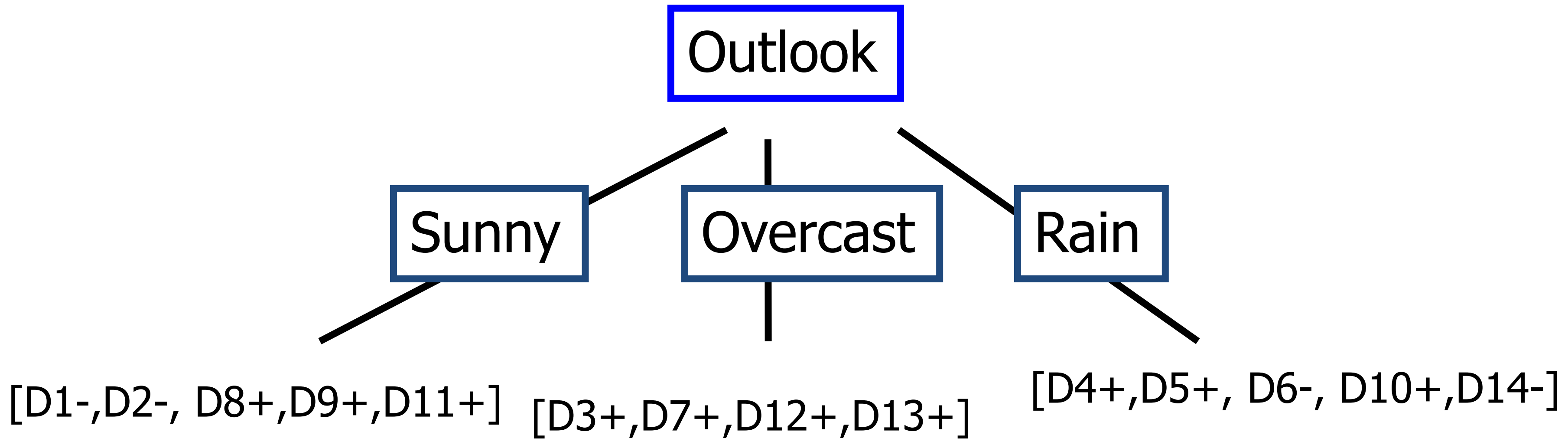
Outlook

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

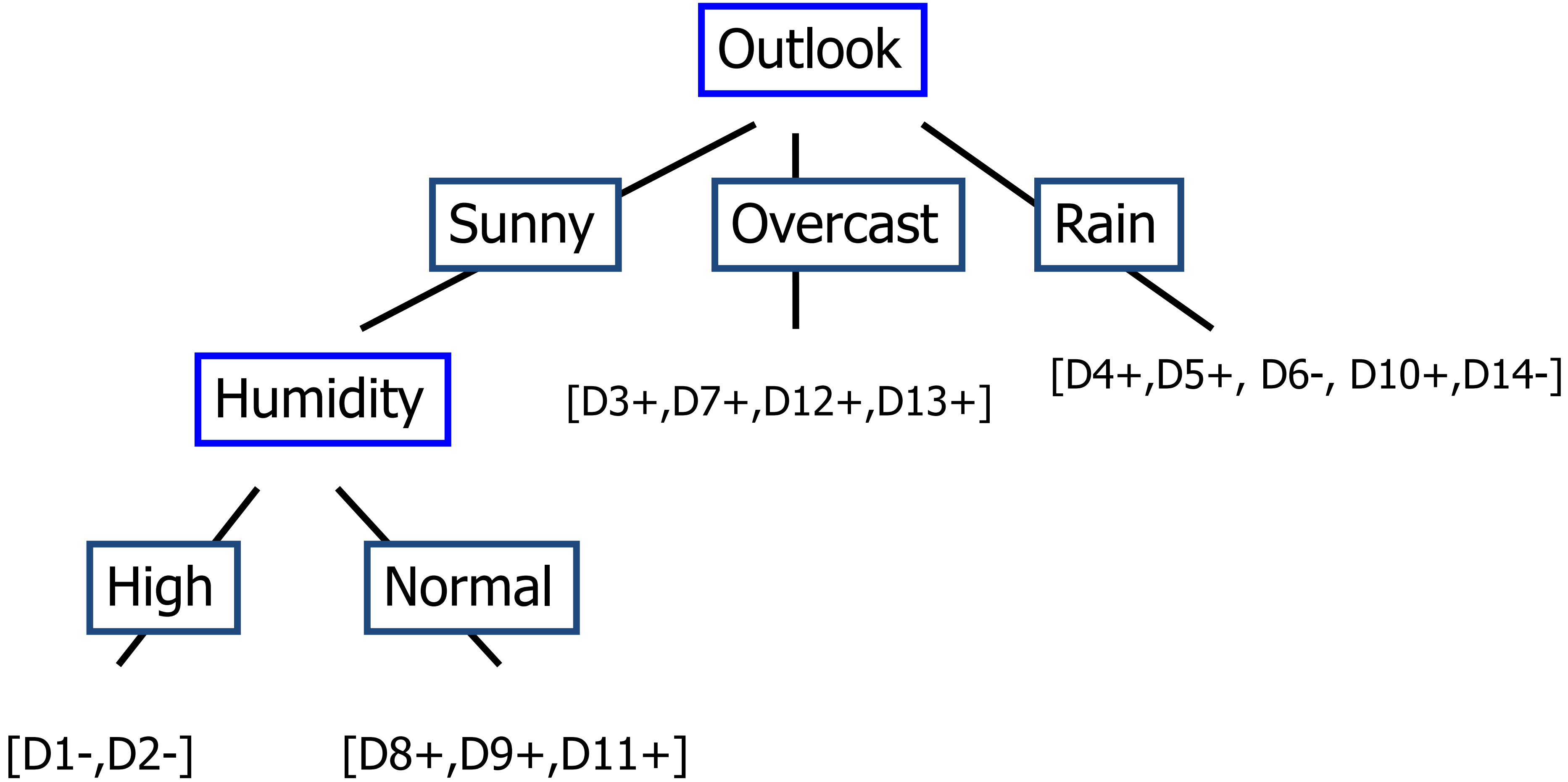
ID3


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

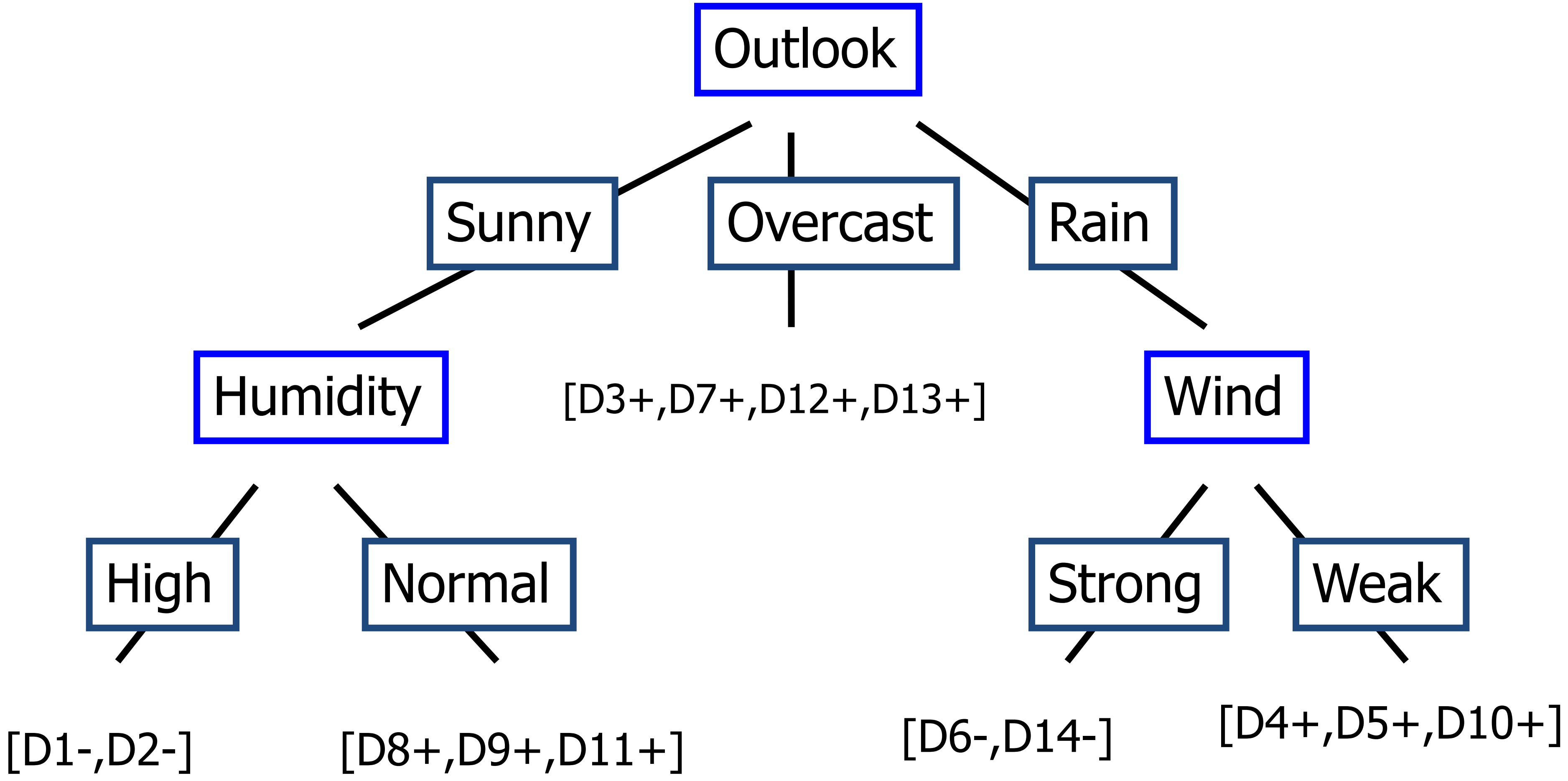
ID3


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

ID3

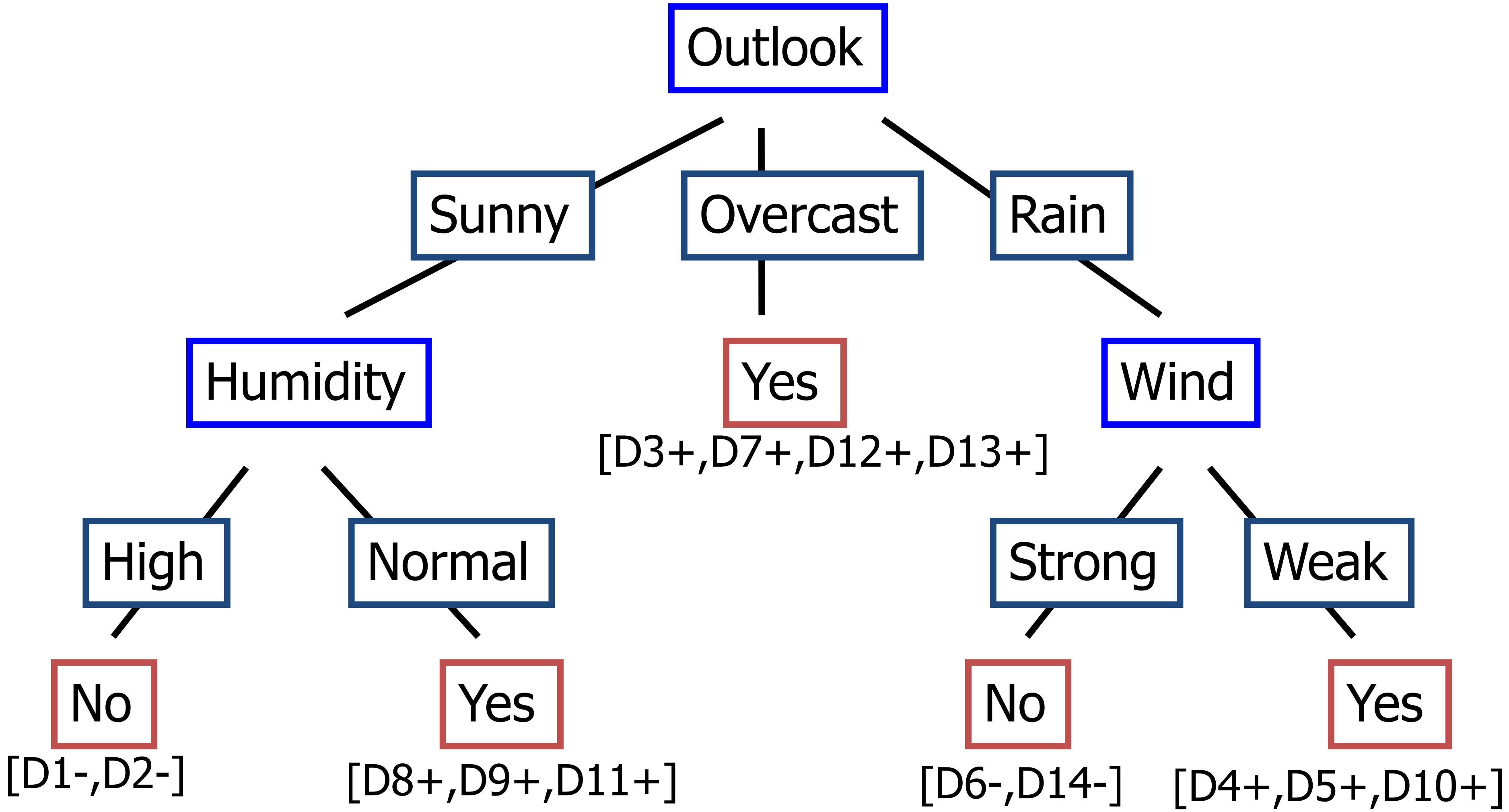

Outlook
Sunny
Overcast
Rain
Humidity
[D3+,D7+,D12+,D13+]
[D4+,D5+, D6-, D10+,D14-]
High
Normal
[D1-,D2-]
[D8+,D9+,D11+]

ID3



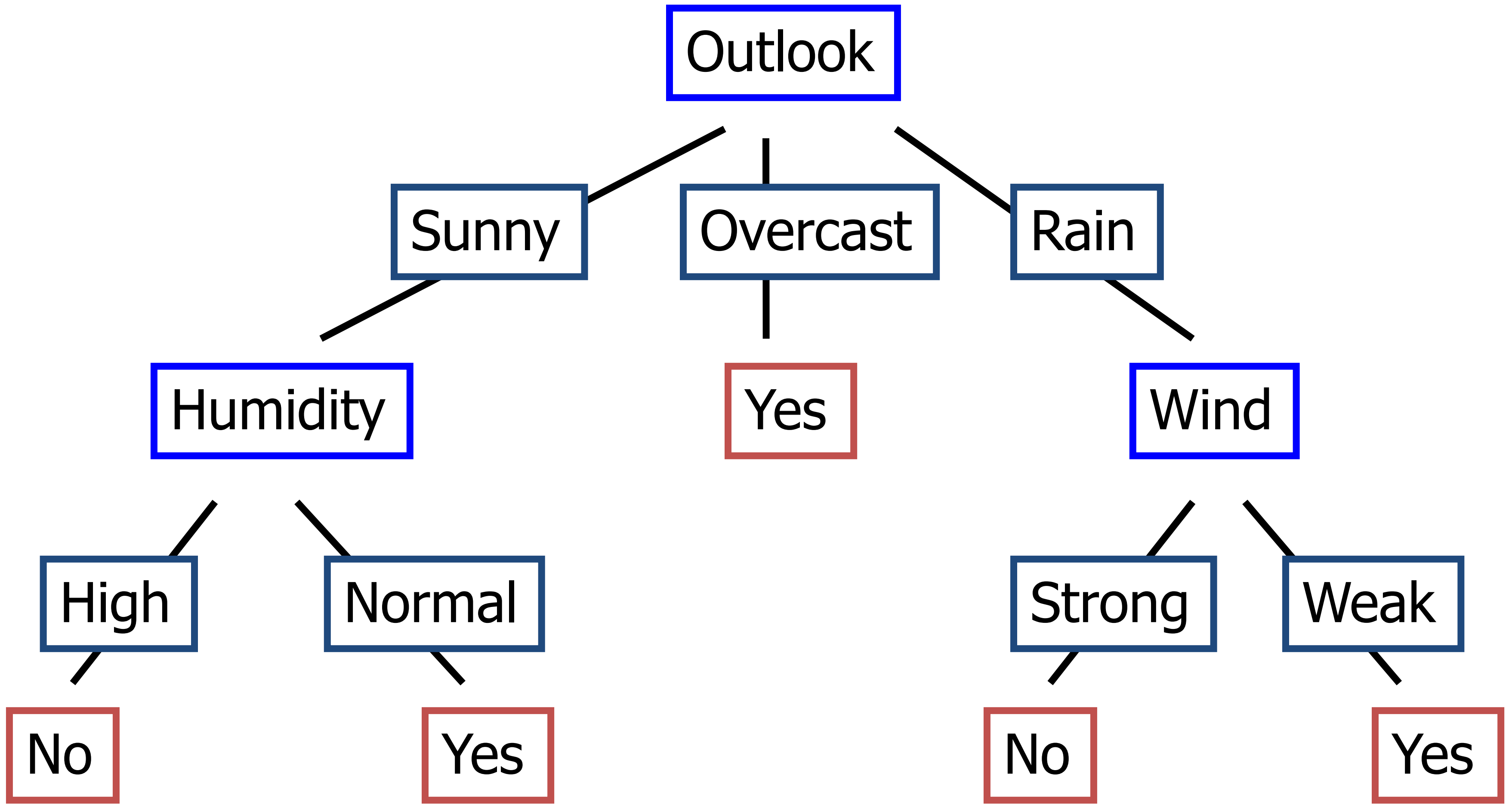
Outlook
Sunny
Overcast
Rain
Humidity
[D3+,D7+,D12+,D13+]
Wind
High
Normal
Strong
Weak
[D1-,D2-]
[D8+,D9+,D11+]
[D6-,D14-]
[D4+,D5+,D10+]

ID3



Outlook
Sunny
Overcast
Rain
Humidity
Yes
[D3+,D7+,D12+,D13+]
Wind
High
Normal
Strong
Weak
No
[D1-,D2-]
Yes
[D8+,D9+,D11+]
No
[D6-,D14-]
Yes
[D4+,D5+,D10+]

ID3


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

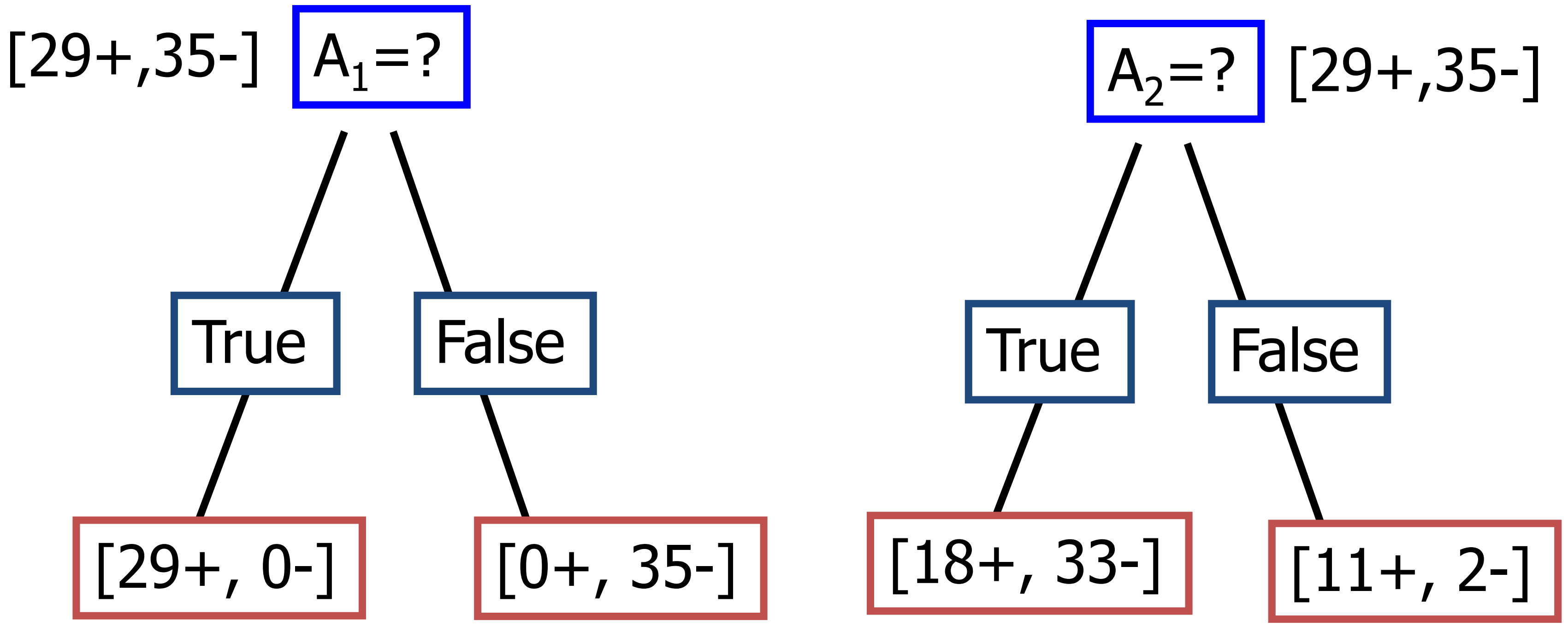
How to choose the best attribute?

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

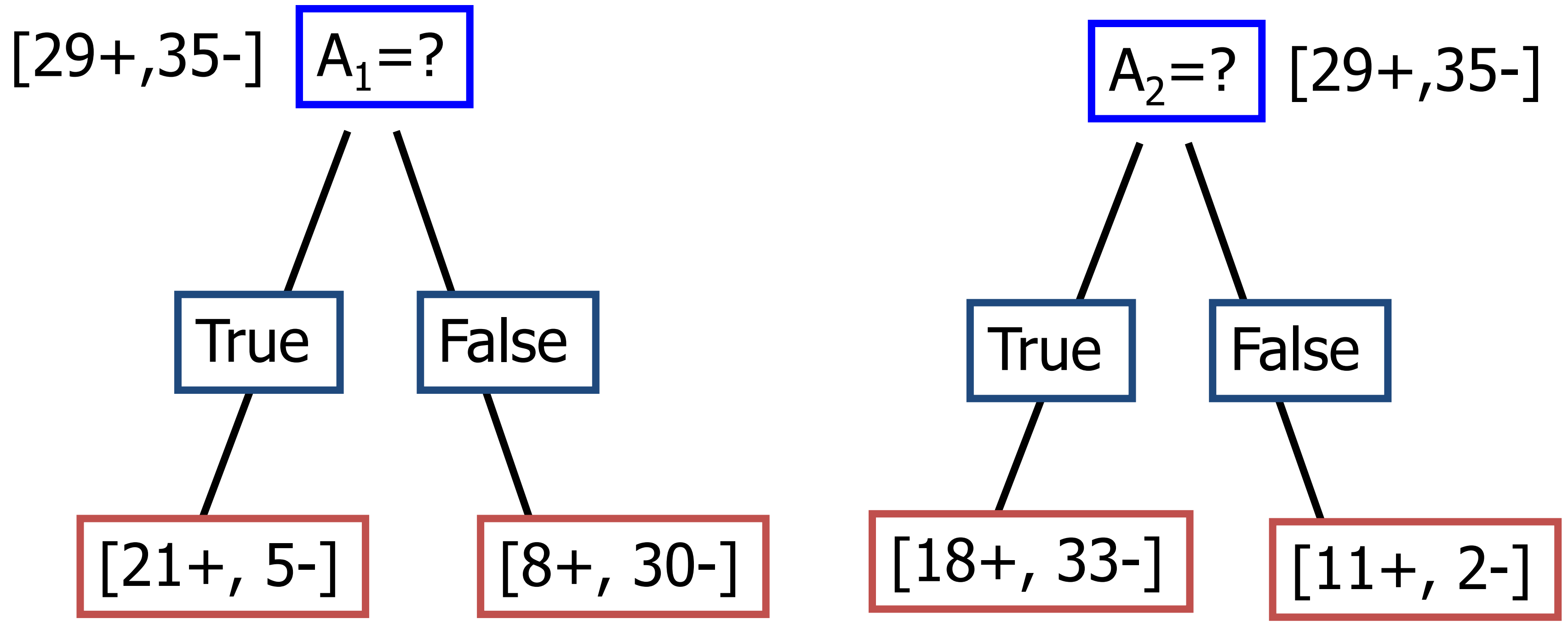
How to choose the best attribute?

- 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?



Information theory

- One answer gives us an information amount proportional to the unpredictability of the answer
- Example: Coin heads and tails
- Formally:

$$E(X) = -\sum_{i}^{n} p_i \log(p_i)$$

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

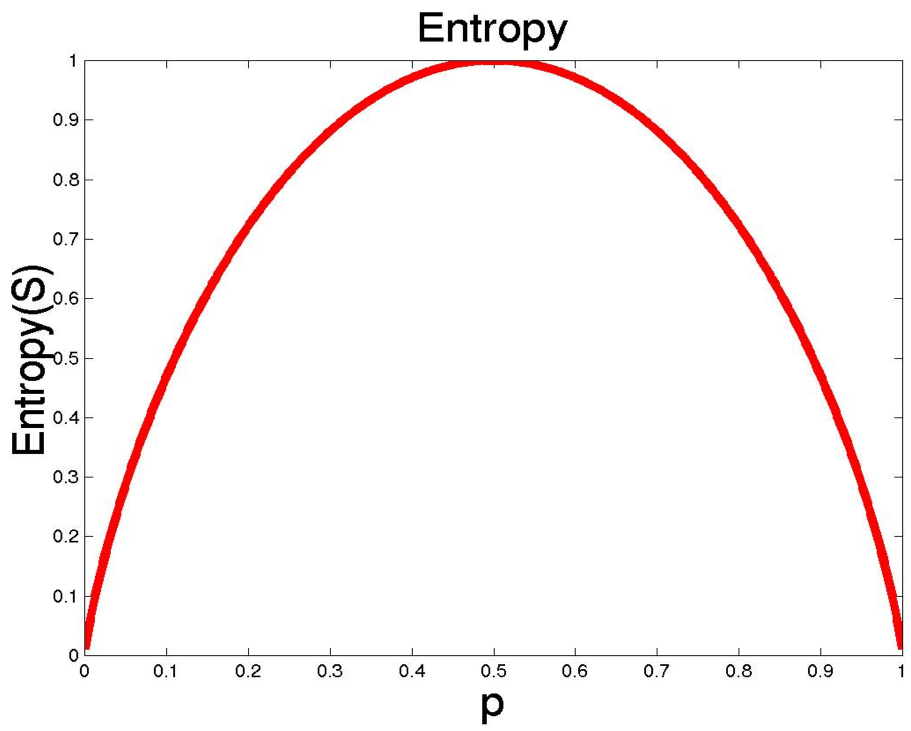
- Information Gain from an answer A is computed as:

$$IG(\mathcal{X}, A) = E(\mathcal{X}) - E(\mathcal{X}|A)$$

Binary answer case

$$E(A) = -p_+ \log_2 p_+ - p_- \log_2 p_- =$$

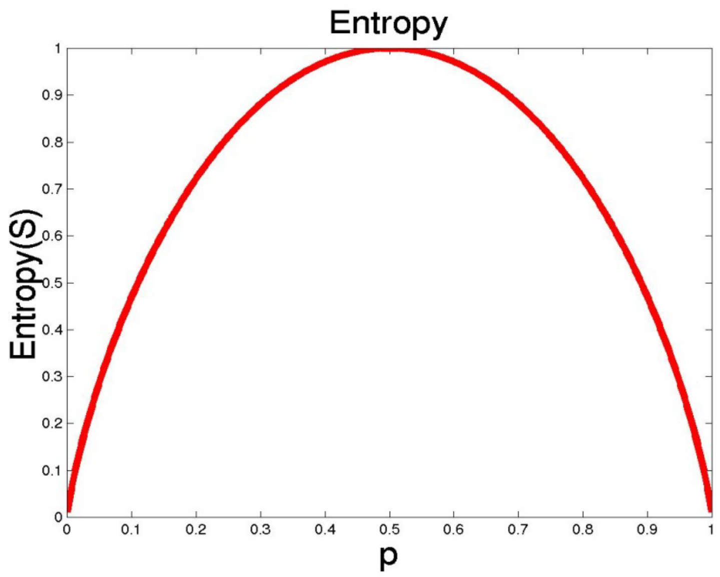
$$-p_+ \log_2 p_+ - (1-p_+)\log_2 (1-p_+)$$



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

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



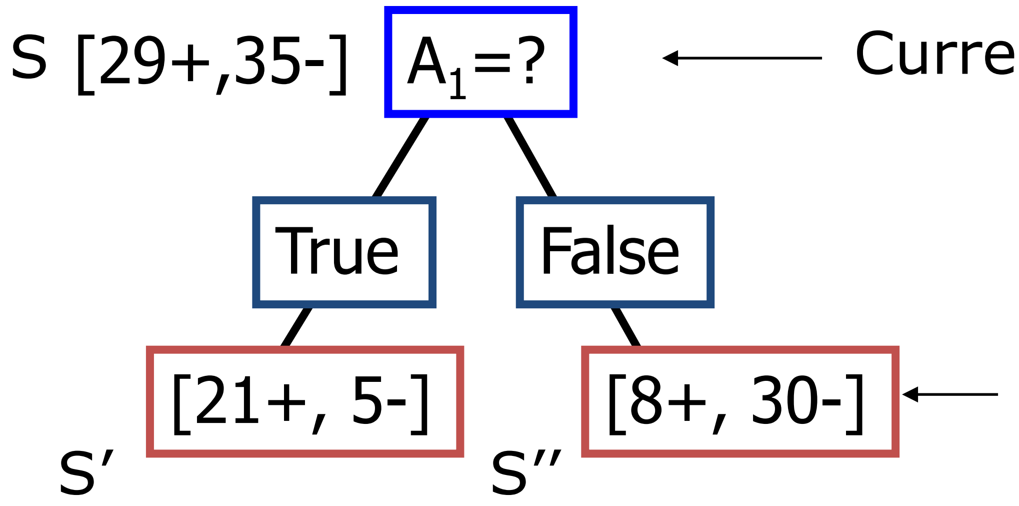
Entropy

- 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 after answering A

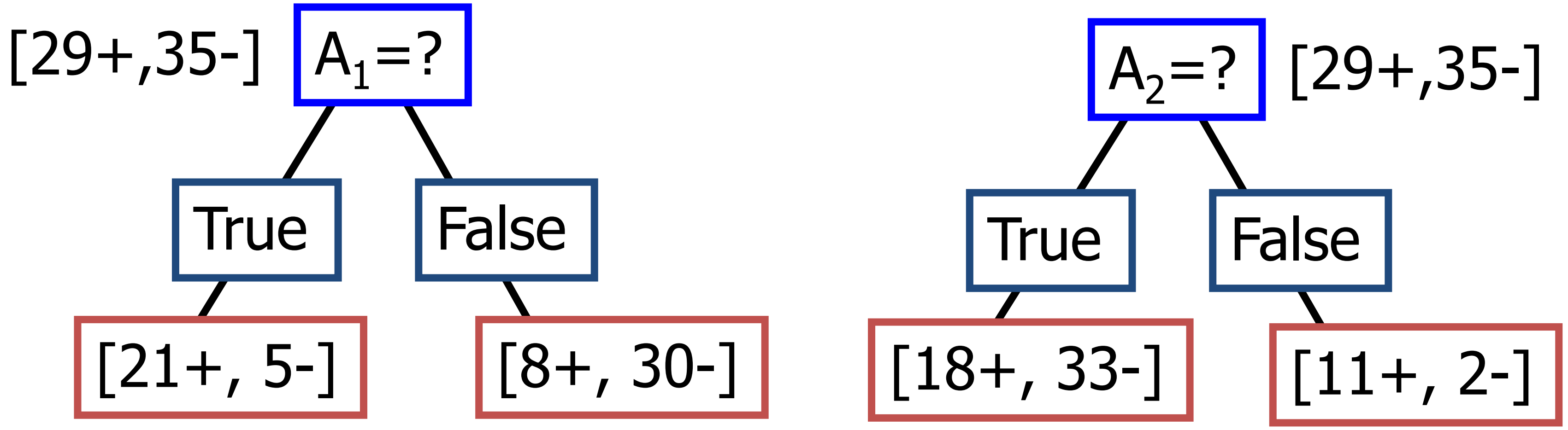
$$E(S|A) = \sum_{v_a \in A} p(A = v_a) E(S_{v_a})$$

$$E(S|A) = \frac{|S'|}{|S|} E(S') + \frac{|S''|}{|S|} E(S'')$$

Information gain

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

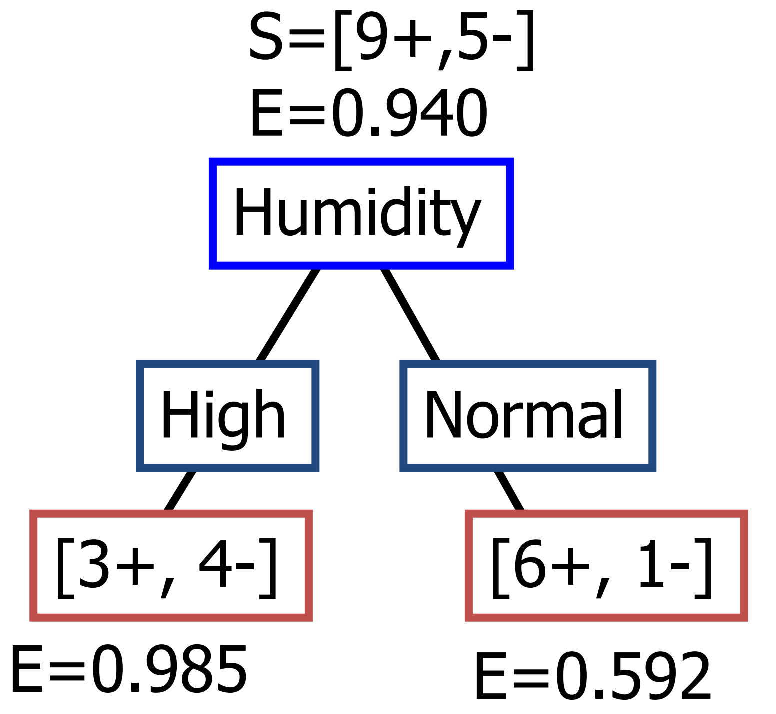
$$IG(S,A)=E(S) - \sum_{v \in values(A)} |S_v|/|S|\ E(S_v)$$

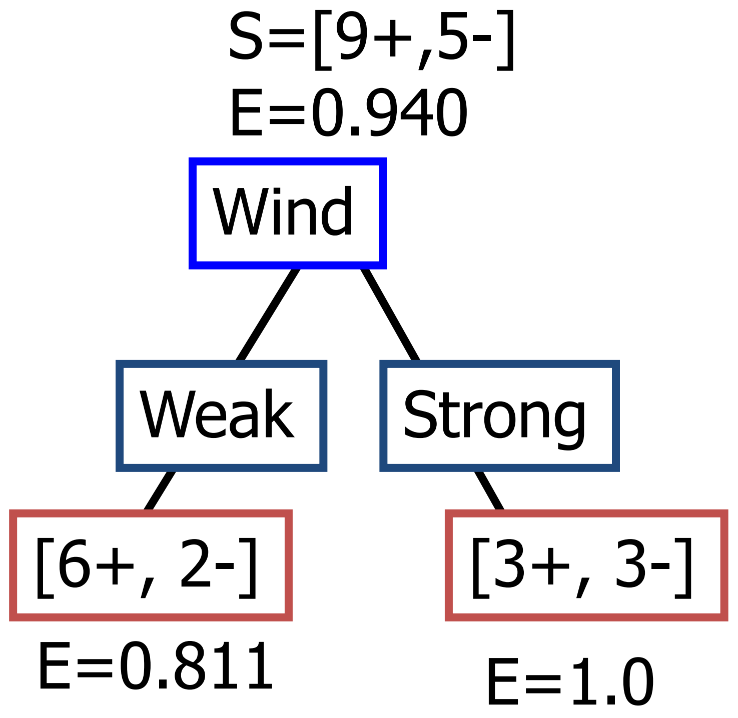


Which is the best attribute to choose?

- The one with highest information gain!!!
- Play tennis example:

Choosing the attribute

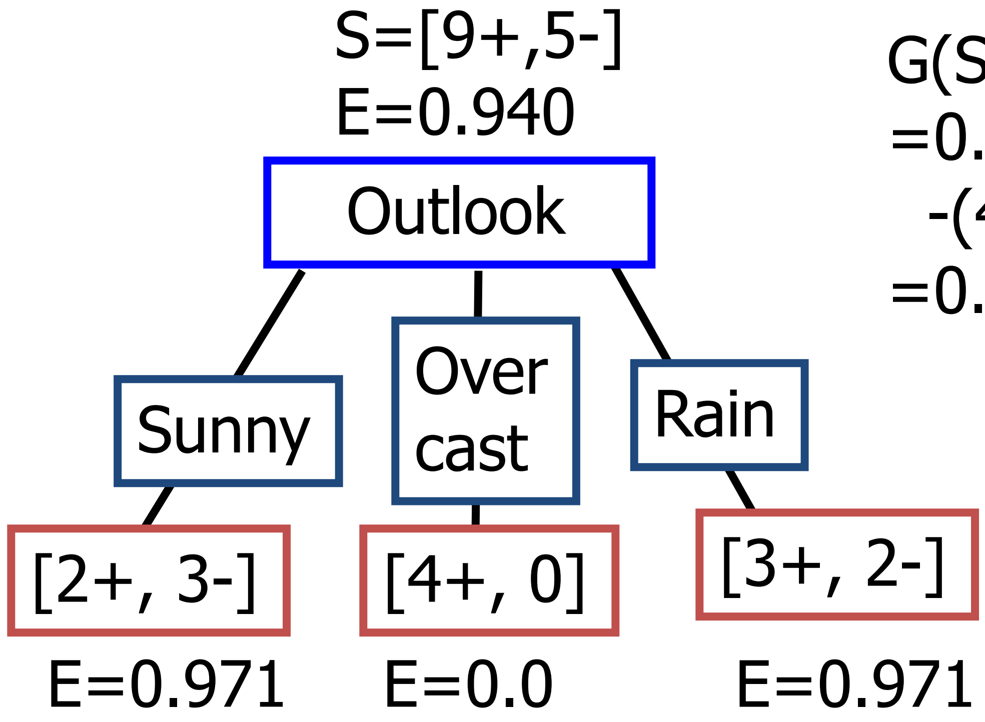




G(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

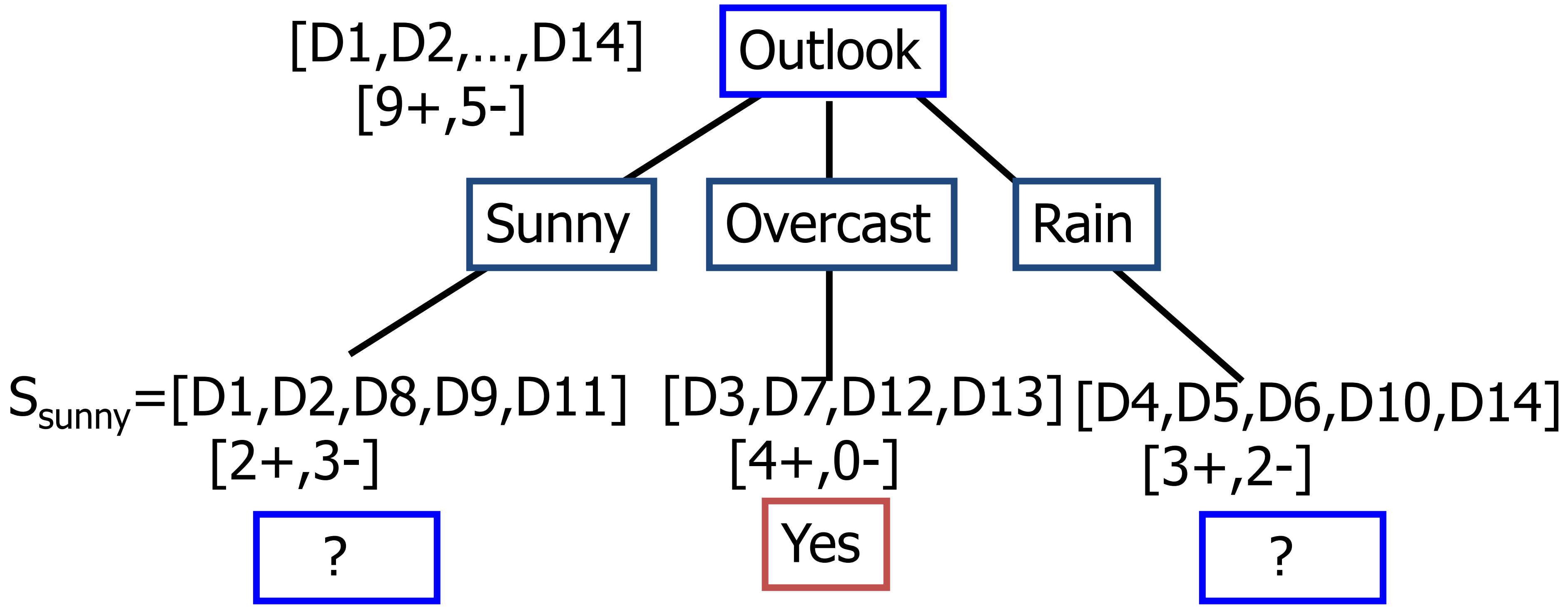
G(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

Choosing the attribute



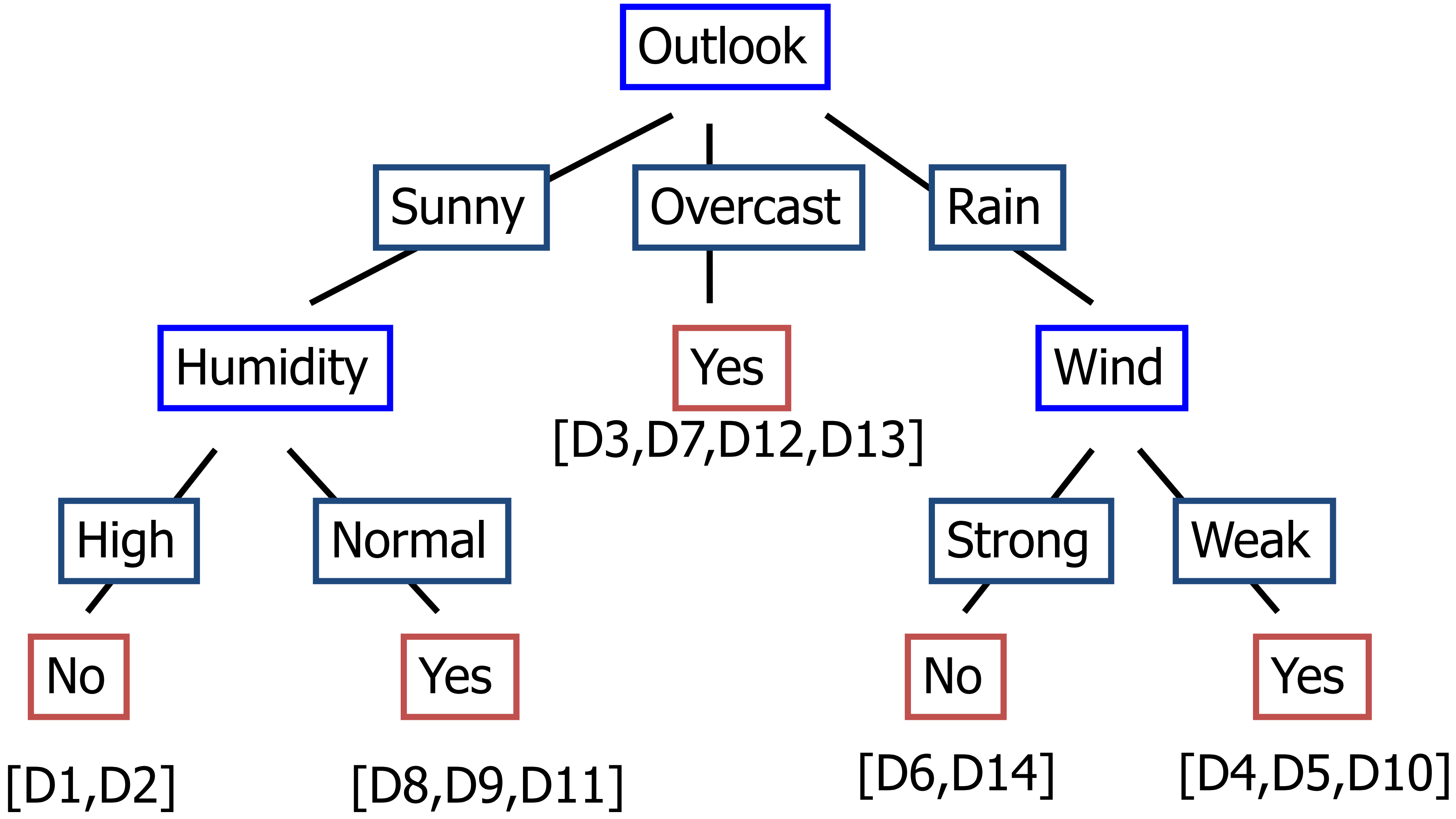
G(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Choosing next attribute



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



ID3 Algorithm

```
Algorithm: Decison Tree ($\mathcal{X}$: Examples, $\mathcal{C}$: Classification, $\mathcal{A}$: Attributes)
if all examples are from the same class
then
    return a leave with the class name
else
    Compute the Entropy of the examples ($\mathrm{E}(\mathcal{X})$)
    foreach a in $\mathcal{A}$ do
        Compute the conditional entropy ($\mathrm{E}(\mathcal{X}|a)$) and the information gain ($IG$)
    Pick the attribute that maximizes $IG$ ( a)
    Delete a from the list of attributes ($\mathcal{A}$)
    Generate a root node for the attribute a
    foreach partition generated by the values of the attribute a do
        Tree$_i$=Decision Tree(examples from $\mathcal{X}$ with a=v$_i$, examples classification, rest of attributes)
        generate a new branch with a=v$_i$ and Tree$_i$
    return the root node for a
```

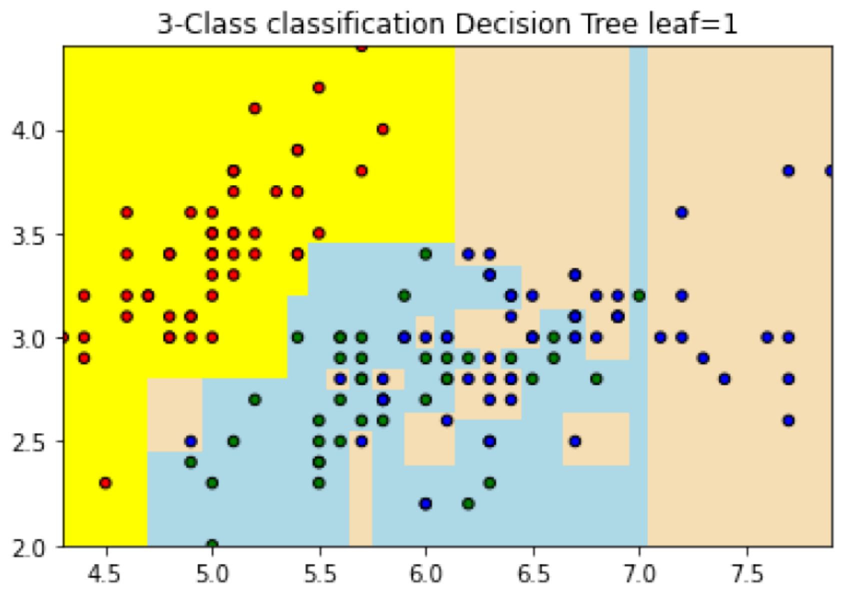
[Possible issues while building the tree]

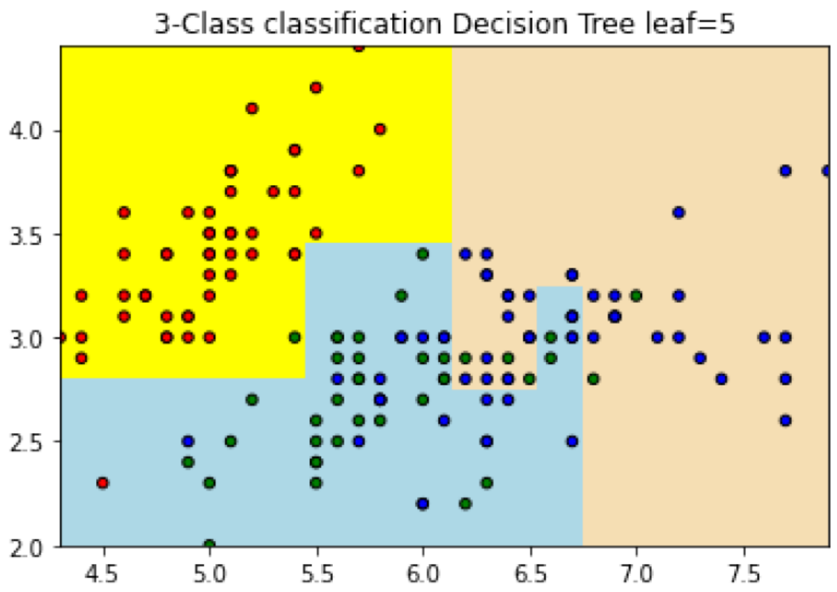
- What if there are not examples for one value when expanding a node? Do not generate branch. When testing, go to the most popular branch
- When we cannot expand the node but still there are + and – examples mixed in the node, label the node with the majority class.
- **Overfitting**

Overfitting in DTs

- Decision trees tend to overfit the data given that we can partition the examples until we reach a leave where all the predictions are correct
- As we descend on the tree the decisions are taken with less and less data
- We can regularize the learning by controlling the parameters that define the complexity of the tree
  - Minimum number of examples in a leave
  - Number of nodes/leaves in the tree
  - Maximum depth of the branches of the tree
  - Fixing a threshold for the heuristic

Examples of trees

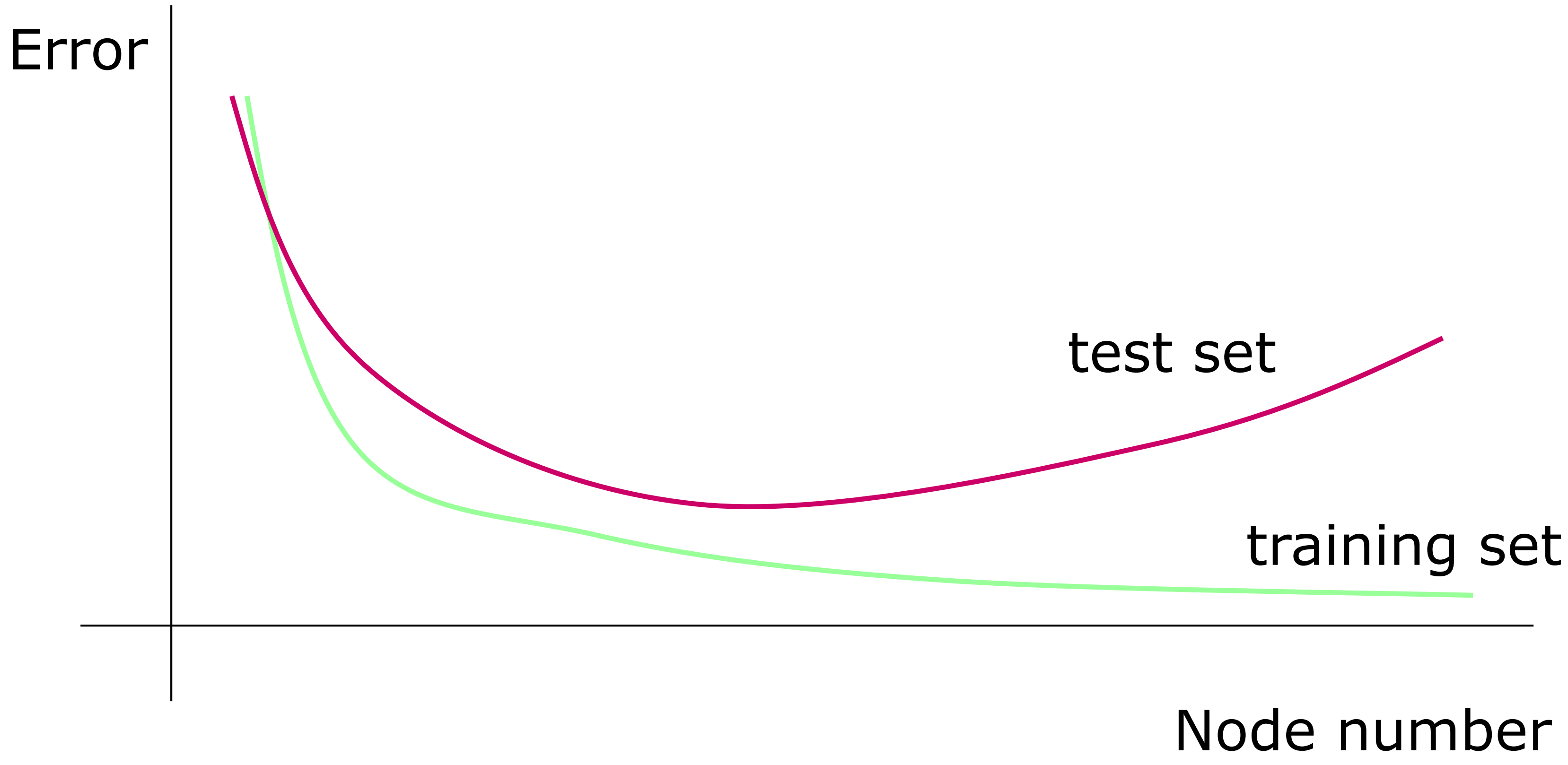




Overfitting in DTs

- An alternative is to overfit the tree and prune the result by estimating the generalization error of the leaves, deleting the nodes with an expected error larger than no having them (Post prunning)

Overfitting in DTs



Atributes

- Undesired effect: Bias towards attributes with a large number of different values in categorical variables
- Attributes with numeric values?
  - Discretize... But how to define thresholds?

Categorical Variables

- Information Gain has preference towards attributes with larger number of values
- This bias is due to having more possible answers, sort more the data (in fact, an answer gives you more IG, but also needed more information to answer the question)

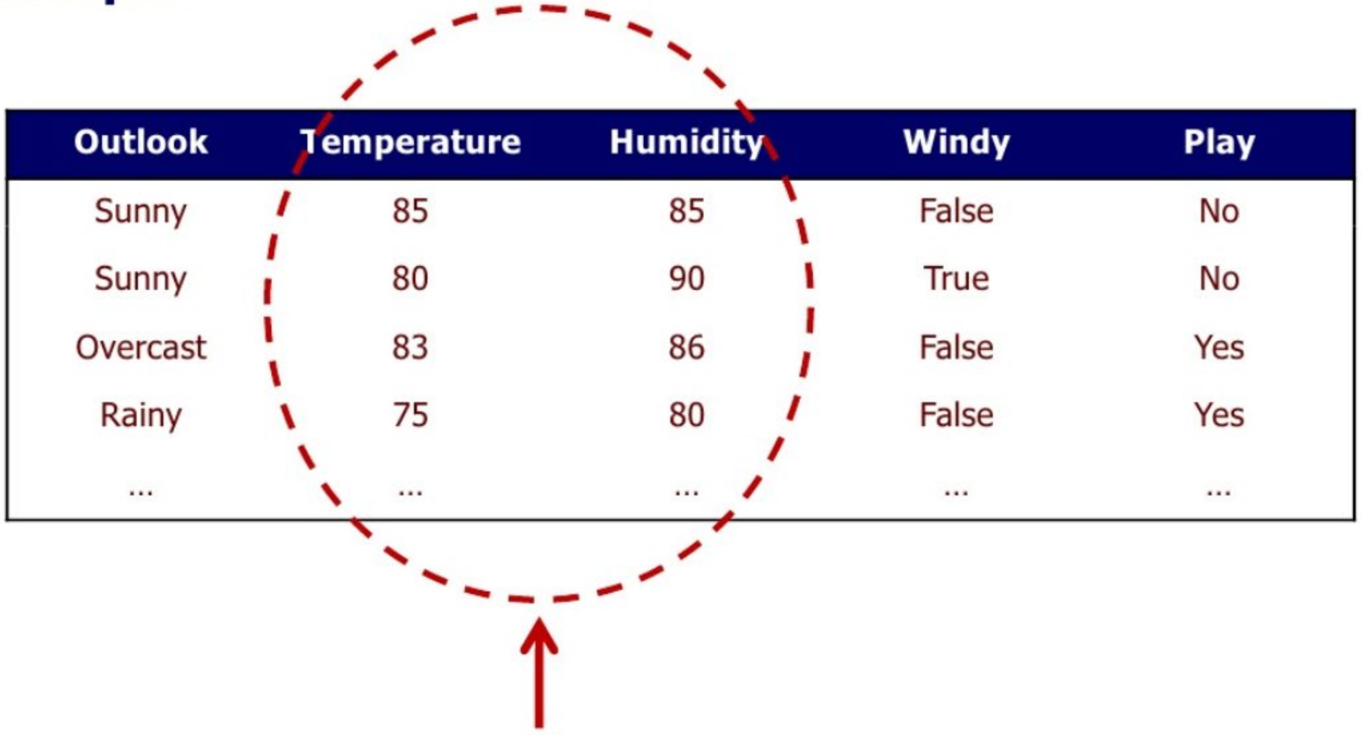
- In practical implementations categorical attributes are assumed to be binary
- Attributes with more than two values are transformed using one-hot encoding: This makes decision trees binary trees in practice

Atributes

- Undesired effect: Bias towards attributes with a large number of different values in categorical variables
- **Attributes with numeric values?**
  - **Discretize... But how to define thresholds?**

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] >= v_i]:
  - Start from sorted sequence of values A= {v_1, v_2, ..., 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] >= 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

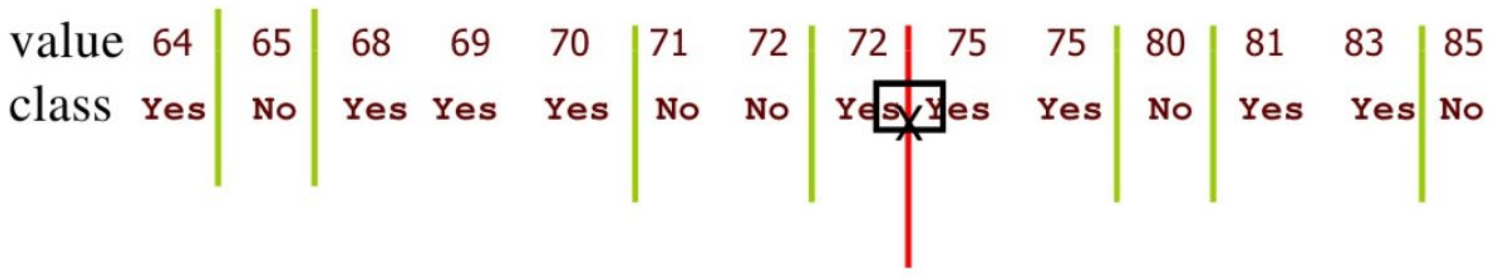
  - E.g.: temperature < 71.5: yes/4, no/2
    temperature ≥ 71.5: yes/5, no/3

  - Info([4,2],[5,3]) = 6/14 info([4,2]) + 8/14 info([5,3]) = 0.939 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
X

Potential optimal breakpoints

Breakpoints between values of the same class cannot be optimal

Decision Trees for regression

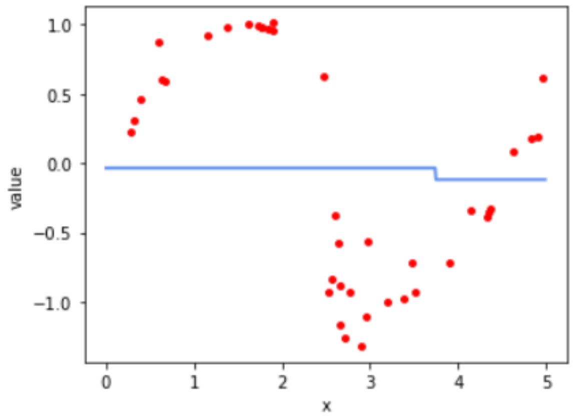
- Similar to classification, but minimize the sum of squared errors relative to the average target value in each induced partition;

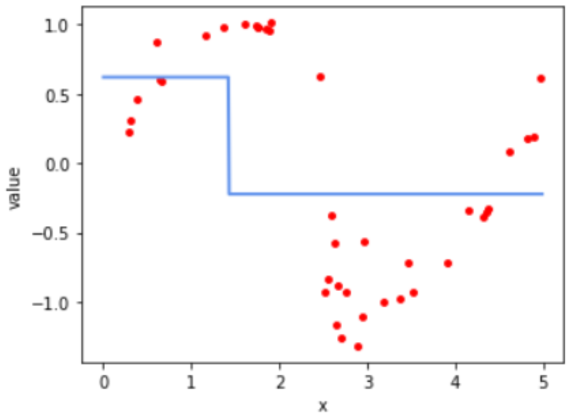
Find pair (v^*, s^*) where v^* is an input feature, and $s^* \in R$ s.t.

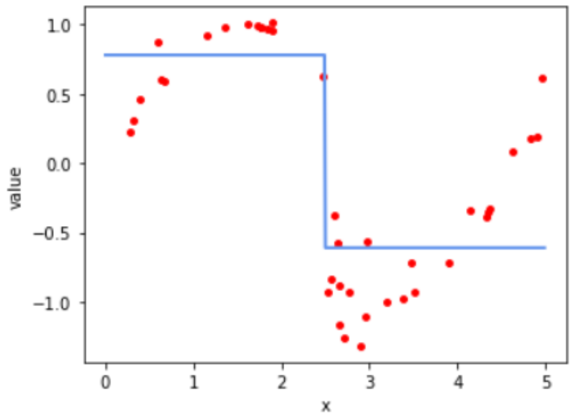
$$v^*, s^* := \arg\min_{v,s} \left[SSE\left(S_{v \leq s}\right) + SSE\left(S_{v > s}\right)\right]$$

where $\mathrm{SSE}(S) := \sum_{i:(\mathbf{x}_i, y_i) \in S} \left(y_i - \mathrm{avg}(S)\right)^2$

Decision Trees for regression







Advantages/Problems with DTs

- Advantages:
  - Their computational complexity is low
  - They perform automatic feature selection, only the attributes relevant for the task are used
  - They can be interpreted more easily than other models
- Drawbacks:
  - They have large variance, slight changes on the sample can result in completely different model
  - They are sensitive to noise and miss-classifications (regularization helps)
  - They approximate the concepts by partitions that are parallel to the attributes, so they can return complex models if classification borders are oblique