Machine Learning 10-401, Spring 2018

Decision Tree Learning

01/22/ 2018

Maria-Florina (Nina) Balcan

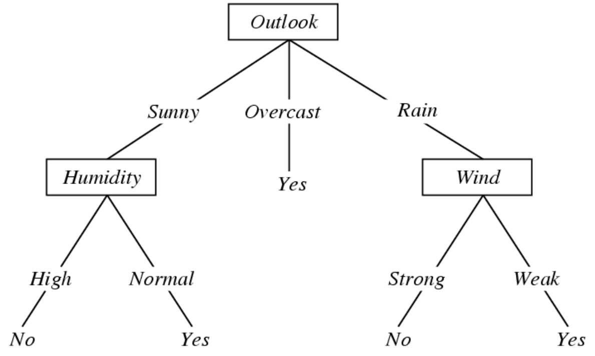
Learning Decision Trees.

Supervised Classification.

Useful Readings:

- Mitchell, Chapter 3
- Bishop, Chapter 14.4

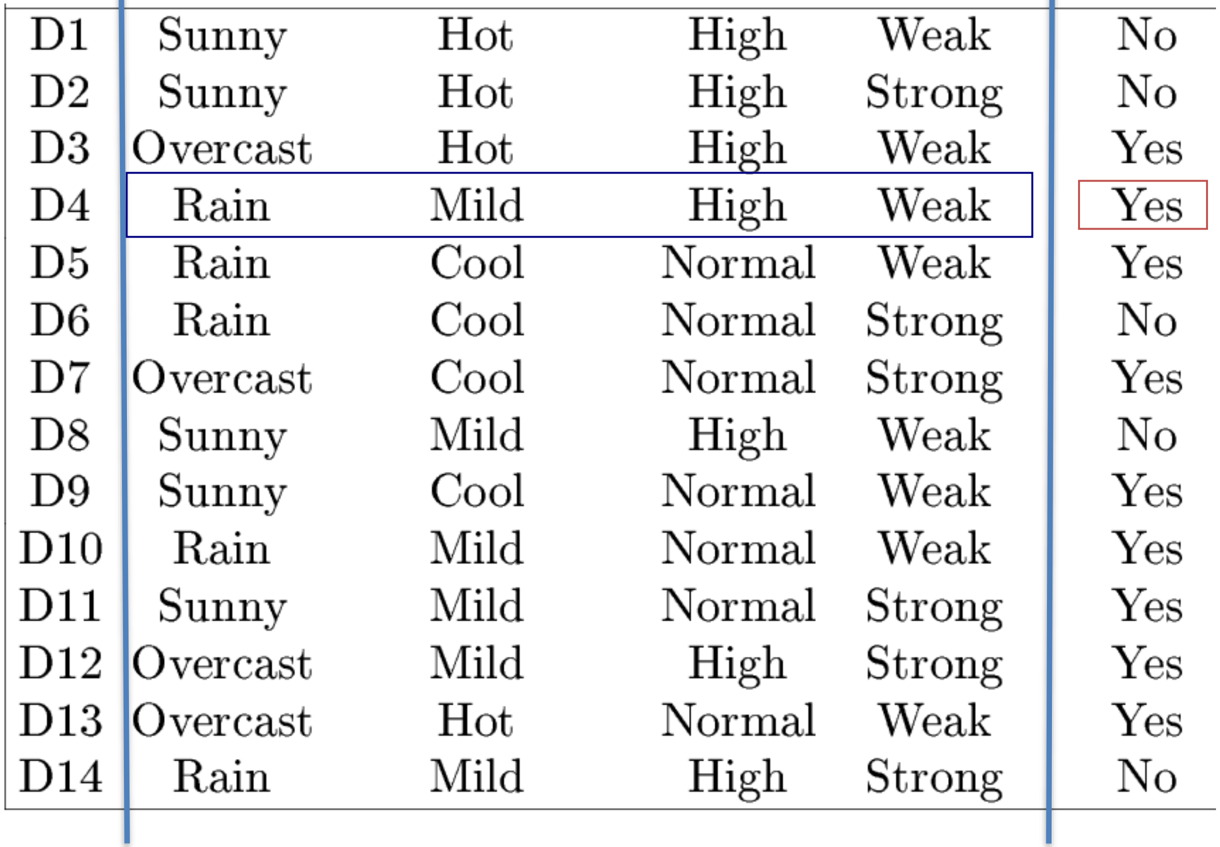
DT learning: Method for learning discrete-valued target functions in which the function to be learned is represented by a decision tree.



Supervised Classification: Decision Tree Learning

Example: learn concept **PlayTennis** (i.e., decide whether our friend will play tennis or not in a given day)

Simple Training Data Set



Day	Outlook	Temperature	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	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

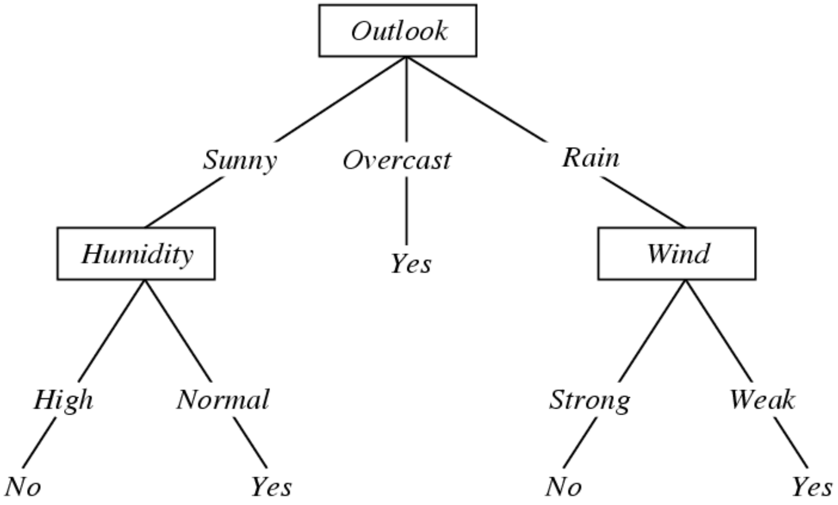
example

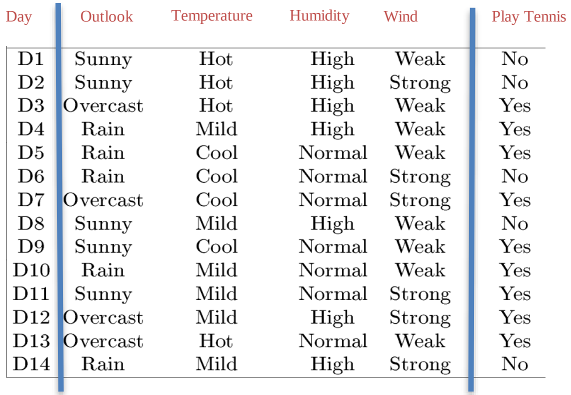
label

Supervised Classification: Decision Tree Learning

- Each internal node: test one (discrete-valued) attribute X_i
- Each branch from a node: corresponds to one possible values for X_i
- Each leaf node: predict Y (or $P(Y=1|x \in \text{leaf})$)

Example: A Decision tree for
f: <Outlook, Temperature, Humidity, Wind> → PlayTennis?



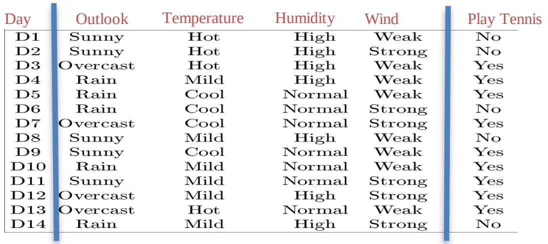


Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

E.g., x=(Outlook=sunny, Temperature-Hot, Humidity=Normal,Wind=High), f(x)=Yes.

Supervised Classification: Problem Setting

Input: Training labeled examples $\{(x^{(i)},y^{(i)})\}$ of unknown target function f



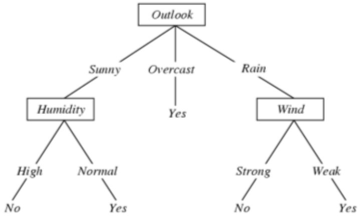
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

- Examples described by their values on some set of features or attributes
  - E.g. 4 attributes: *Humidity*, *Wind*, *Outlook*, *Temp*
    - e.g., <*Humidity=High*, *Wind=weak*, *Outlook=rain*, *Temp=Mild*>
  - Set of possible instances X (a.k.a instance space)

- Unknown target function $f : X \rightarrow Y$
  - e.g., $Y=\{0,1\}$ label space
  - e.g., 1 if we play tennis on this day, else 0

Output: Hypothesis $h \in H$ that (best) approximates target function f

- Set of function hypotheses $H=\{\ h \mid h : X \rightarrow Y\ \}$
  - each hypothesis h is a decision tree

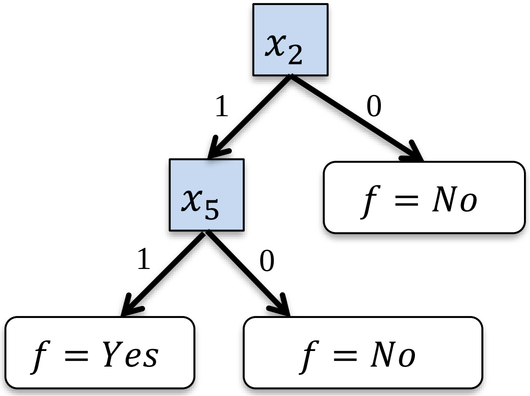


Supervised Classification: Decision Trees

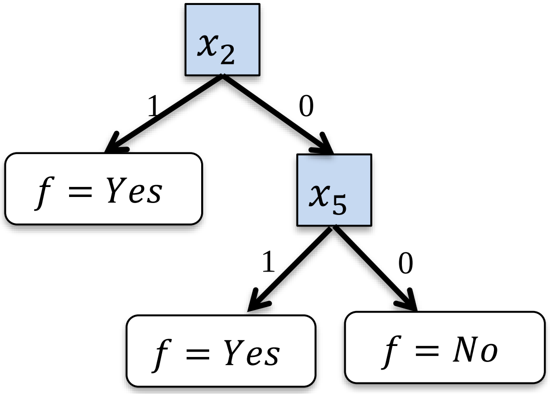
Suppose $X = <x_1, \ldots x_n>$
where x_i are boolean-valued variables

How would you represent the following as DTs?

$f(x) = x_2 \; AND \; x_5$?



$f(x) = x_2 \; OR \; x_5$

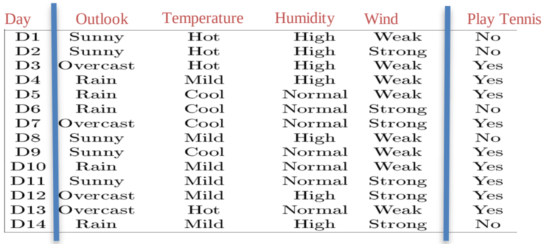


Hwk: How would you represent $X_2 X_5 \vee X_3 X_4 (\neg X_1)$?

Supervised Classification: Problem Setting

Input: Training labeled examples $\{(x^{(i)},y^{(i)})\}$ of unknown target function f

- Examples described by their values on some set of features or attributes
    - E.g. 4 attributes: *Humidity*, *Wind*, *Outlook*, *Temp*
        – e.g., <*Humidity*=*High*, *Wind*=*weak*, *Outlook*=*rain*, *Temp*=*Mild*>
    - Set of possible instances X (a.k.a instance space)

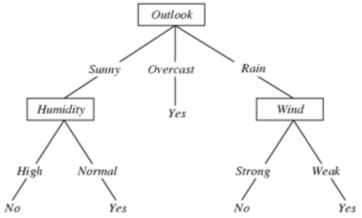


Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

- Unknown target function $f : X \rightarrow Y$
    - e.g., Y={0,1} label space
    - e.g., 1 if we play tennis on this day, else 0

Output: Hypothesis $h \in H$ that (best) approximates target function f

- Set of function hypotheses $H=\{\ h \mid h : X \rightarrow Y\ \}$
    – each hypothesis h is a decision tree



Core Aspects in Decision Tree & Supervised Learning

How to automatically find a good hypothesis for training data?

- This is an **algorithmic** question, the main topic of computer science

When do we generalize and do well on unseen data?

- **Learning theory** quantifies ability to ***generalize*** as a function of the amount of training data and the hypothesis space
- **Occam's razor:** use the *simplest* hypothesis consistent with data!

Fewer short hypotheses than long ones

- a short hypothesis that fits the data is less likely to be a statistical coincidence
- highly probable that a sufficiently complex hypothesis will fit the data

Core Aspects in Decision Tree & Supervised Learning

How to automatically find a good hypothesis for training data?

- This is an **algorithmic** question, the main topic of computer science

When do we generalize and do well on unseen data?

- **Occam’s razor:** use the *simplest* hypothesis consistent with data!

- Decision trees: if we were able to find a small decision tree that explains data well, then good generalization guarantees.
  - NP-hard [Hyafil-Rivest’76]: unlikely to have a poly time algorithm

- Very nice practical heuristics; top down algorithms, e.g, ID3

Top-Down Induction of Decision Trees

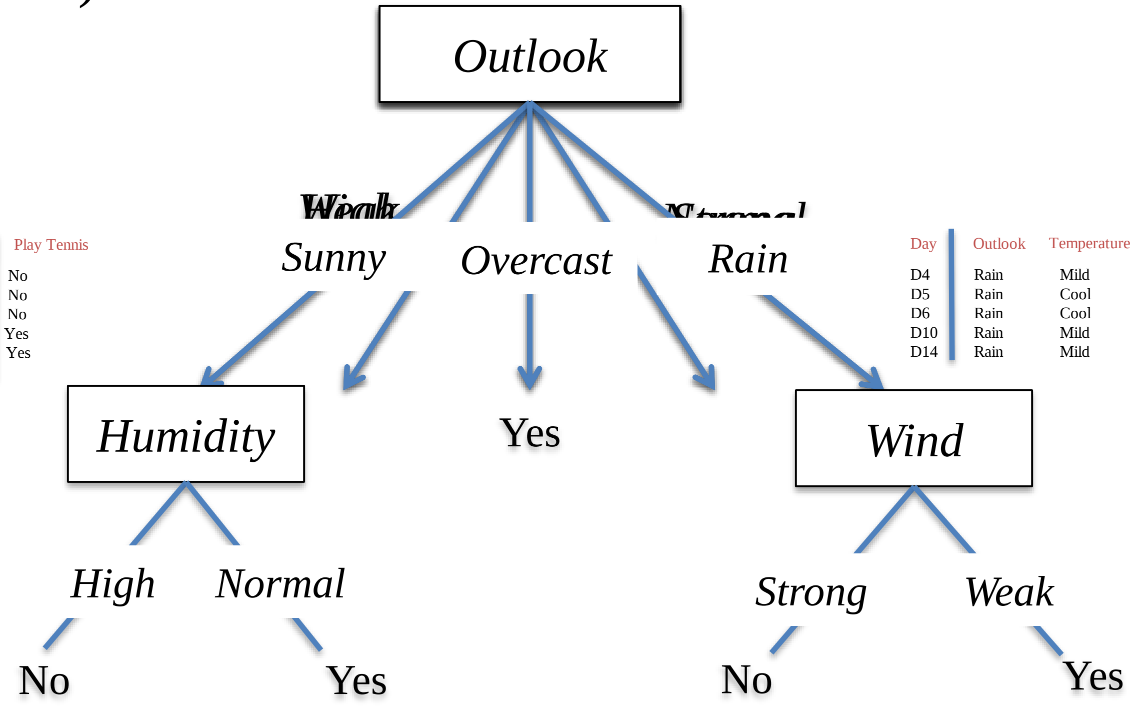
[ID3, C4.5, Quinlan]

ID3: Natural greedy approach to growing a decision tree top-down (from the root to the leaves by repeatedly replacing an existing leaf with an internal node.).

Algorithm:

- Pick "best" attribute to split at the root based on training data.
- Recurse on children that are impure (e.g, have both Yes and No).

Day	Outlook	Temperature	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	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



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]

ID3: Natural greedy approaches where we grow the tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node.

```
node = Root
Main loop:
    1. A ← the "best" decision attribute for next node
    2. Assign A as decision attribute for node
    3. For each value of A, create new descendent of node
    4. Sort training examples to leaf nodes
    5. If training examples perfectly classified, Then STOP,
    Else iterate over new leaf nodes.
```

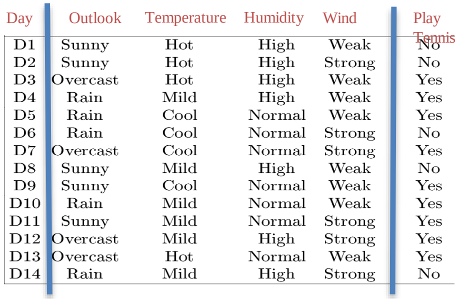
Key question: Which attribute is best?

Top-Down Induction of Decision Trees [ID3, C4.5, Quinlan]

ID3: Natural greedy approach to growing a decision tree top-down.

Algorithm:

- Pick "best" attribute to split at the root based on training data.
- Recurse on children that are impure (e.., have both Yes and No).



Day	Outlook	Temperature	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	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

Key question: Which attribute is best?

ID3 uses a statistical measure called information gain (how well a given attribute separates the training examples according to the target classification)

Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]

Which attribute to select?

ID3: The attribute with highest information gain.

a statistical measure of how well a given attribute separates the training examples according to the target classification

Information Gain of A is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

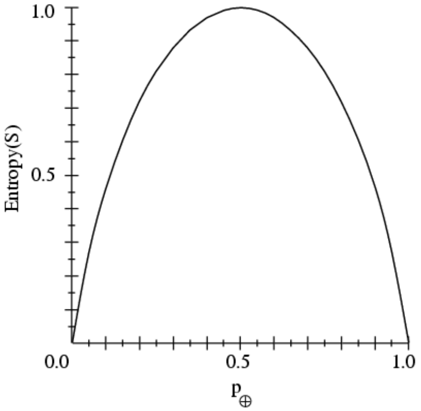
$$Gain(S,A) = H_S(Y) - H_S(Y|A)$$

Entropy information theoretic measure that characterizes the impurity of a labeled set S.

Sample Entropy of a Labeled Dataset

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S.
- $p_{\ominus}$ is the proportion of negative examples in S.
- Entropy measures the impurity of S.

$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

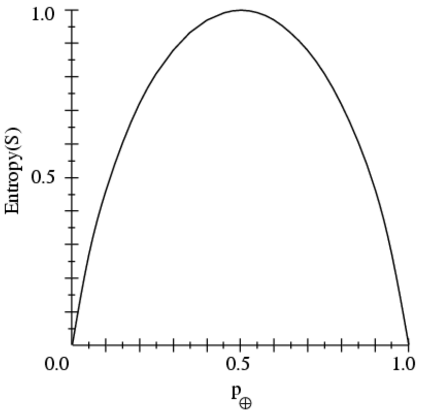


- E.g., if all negative, then entropy=0. If all positive, then entropy=0.
- If 50/50 positive and negative then entropy=1.
- If 14 examples with 9 positive and 5 negative, then entropy=.940

Sample Entropy of a Labeled Dataset

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S.
- $p_{\ominus}$ is the proportion of negative examples in S.
- Entropy measures the impurity of S.

$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

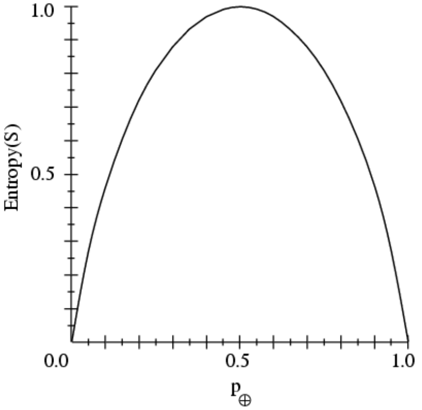


Interpretation from information theory: expected number of bits needed to encode label of a randomly drawn example in S.

- If S is all positive, receiver knows label will be positive, don’t need any bits.
- If S is 50/50 then need 1 bit.
- If S is 80/20, then in a long sequence of messages, can code with less than 1 bit on average (assigning shorter codes to positive examples and longer codes to negative examples).

Sample Entropy of a Labeled Dataset

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S.
- $p_{\ominus}$ is the proportion of negative examples in S.
- Entropy measures the impurity of S.



$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} \; - p_{\ominus} \log_2 p_{\ominus}$$

If labels not Boolean, then $H(S) = \sum_{i \in Y} -p_i \log_2 p_i$

E.g., if c classes, all equally likely, then $H(S) = \log_2 c$

Information Gain

Given the definition of entropy, can define a measure of effectiveness of attribute in classifying training data:

Information Gain of A is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

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

entropy of original collection

Expected entropy after S is partitioned using attribute A

sum of entropies of subsets S_v weighted by the fraction of examples that belong to S_v.

Information Gain

Given the definition of entropy, can define a measure of effectiveness of attribute in classifying training data:

Information Gain of A is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

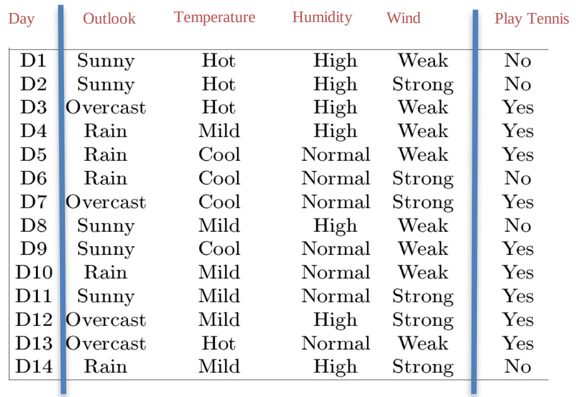
$$Gain(S,A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

entropy of original collection

Expected entropy after S is partitioned using attribute A

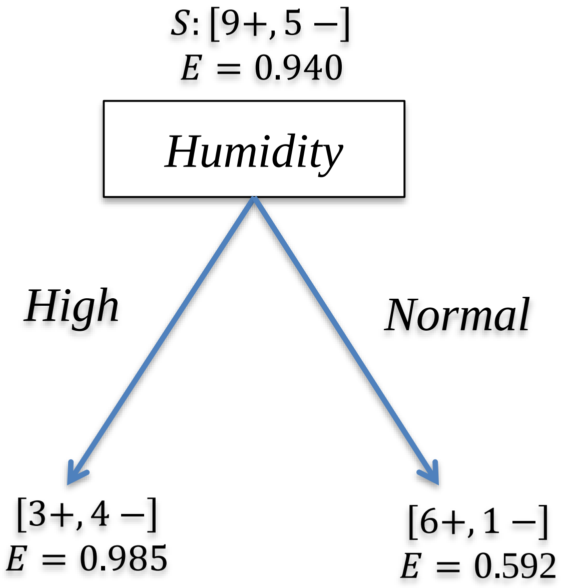
Gain(S,A) information provided about the target function, given the value of some other attribute A.

Selecting the Next Attribute



Day	Outlook	Temperature	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	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

Which attribute is the best classifier?



S: $[9+, 5-]$

$E = 0.940$

Humidity

High

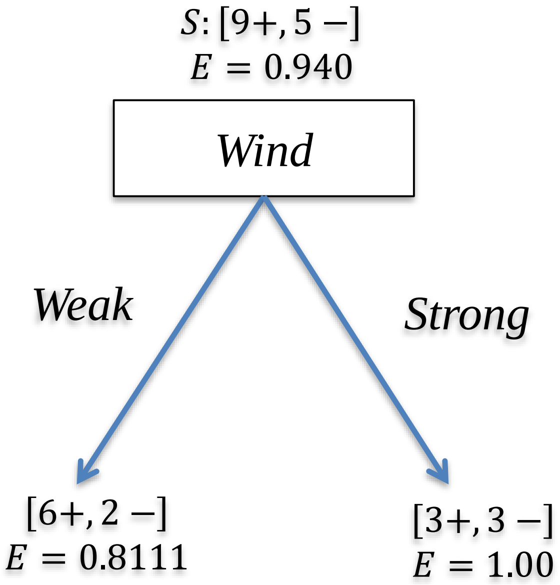
Normal

$[3+, 4-]$

$E = 0.985$

$[6+, 1-]$

$E = 0.592$



S: $[9+, 5-]$

$E = 0.940$

Wind

Weak

Strong

$[6+, 2-]$

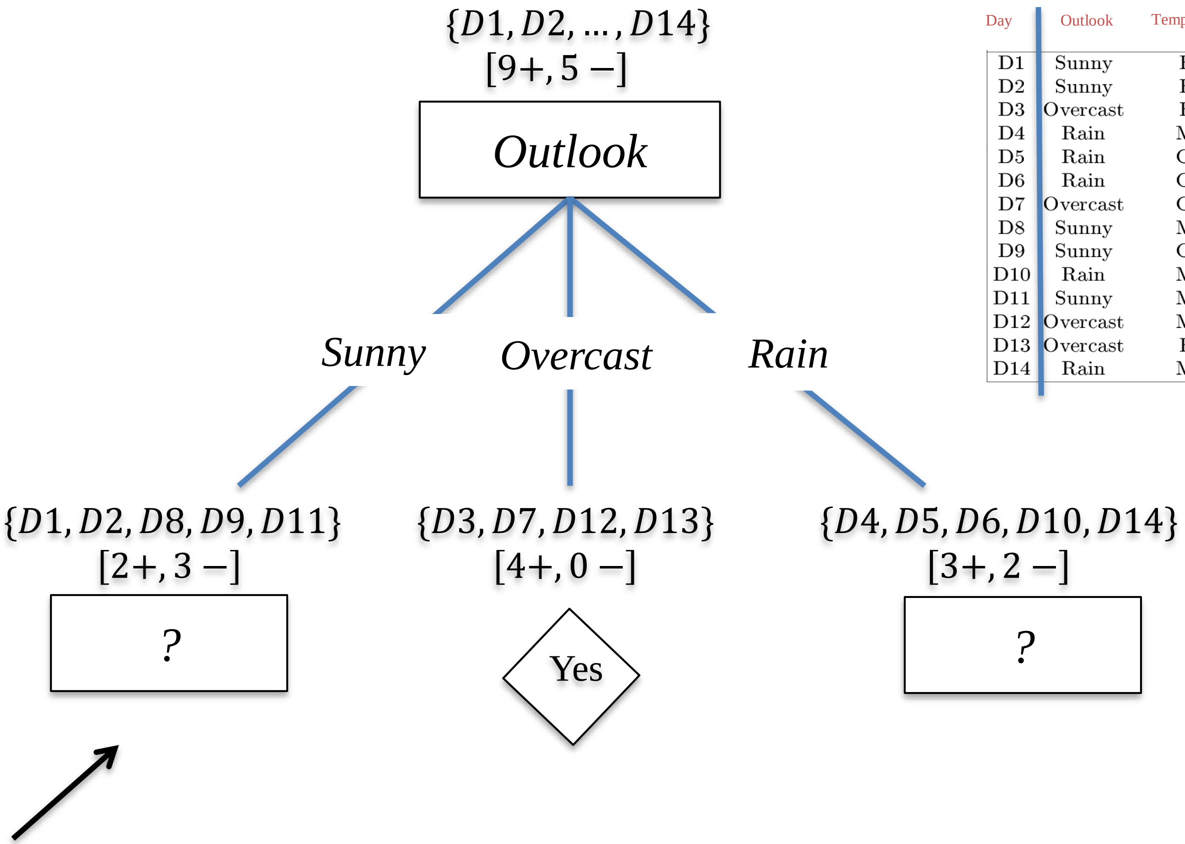
$E = 0.8111$

$[3+, 3-]$

$E = 1.00$

$$Gain(S, Humidity)$$
$$= .940 - \left(\frac{7}{14}\right).985 - \left(\frac{7}{14}\right).592$$
$$= .151$$

$$Gain(S, Wind)$$
$$= .940 - \left(\frac{8}{14}\right).811 - \left(\frac{6}{14}\right)1.0$$
$$= .048$$



Day	Outlook	Temperature	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	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

Which attribute should be tested here?

$$s_{sunny} = \{D1, D2, D8, D9, D11\}$$

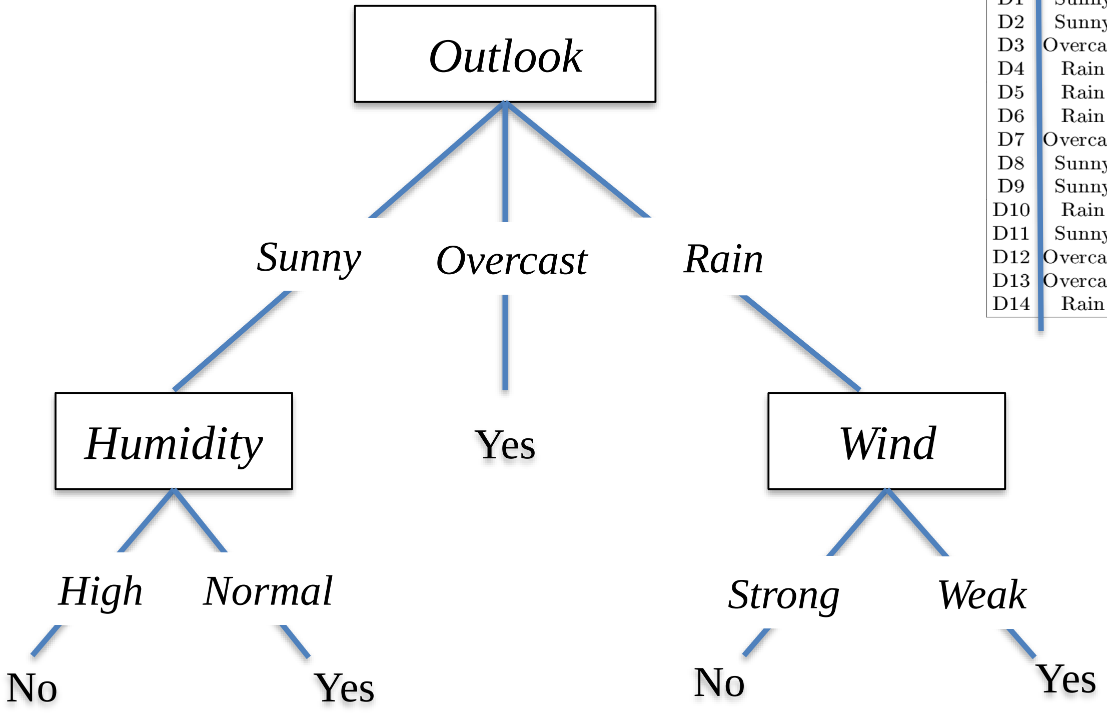
$$Gain(s_{sunny}, Humidity) = .970 - \left(\frac{3}{5}\right)0.0 - \left(\frac{2}{5}\right)0.0 = .970$$

$$Gain(s_{sunny}, Temperature) = .970 - \left(\frac{2}{5}\right)0.0 - \left(\frac{2}{5}\right)1.0 - \left(\frac{1}{5}\right)0.0 = .570$$

$$Gain(s_{sunny}, Wind) = .970 - \left(\frac{2}{5}\right)1.0 - \left(\frac{3}{5}\right).918 = .019$$

Final Decision Tree for

f: <Outlook, Temperature, Humidity, Wind> → PlayTennis?



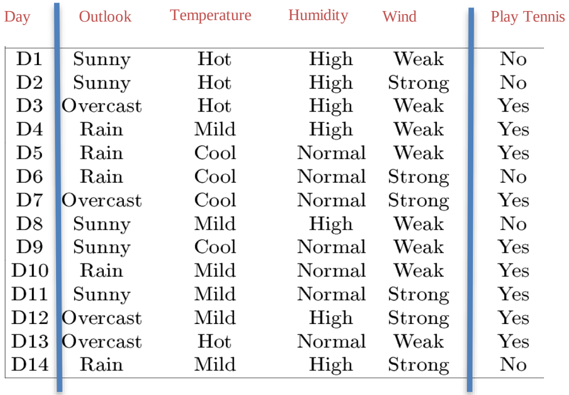
Outlook

Sunny *Overcast* *Rain*

Humidity Yes *Wind*

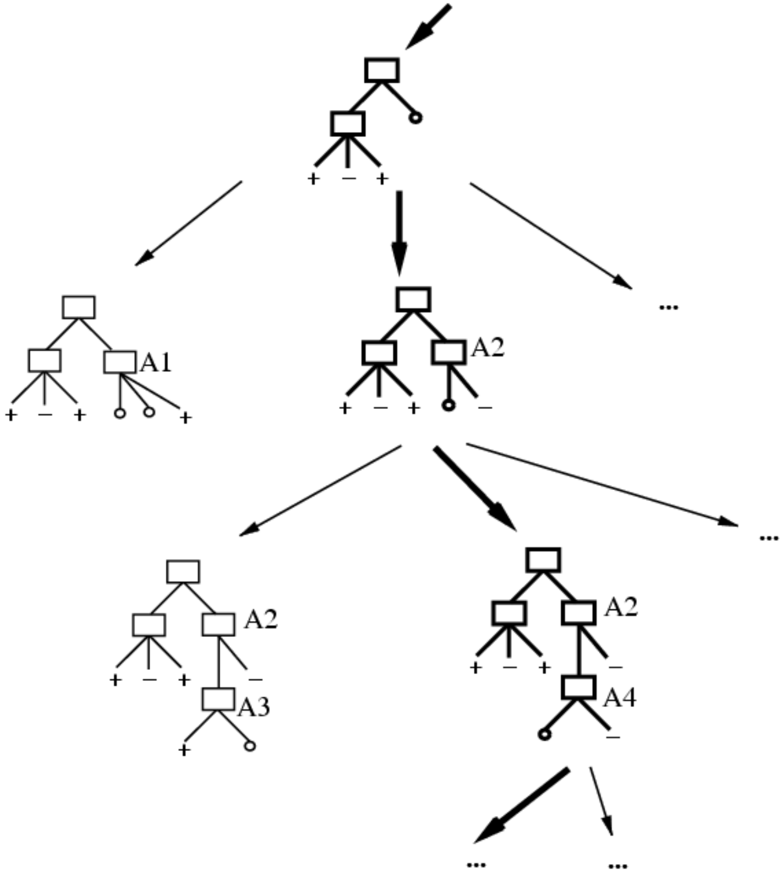
High *Normal* *Strong* *Weak*

No Yes No Yes



Day	Outlook	Temperature	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	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

Properties of ID3

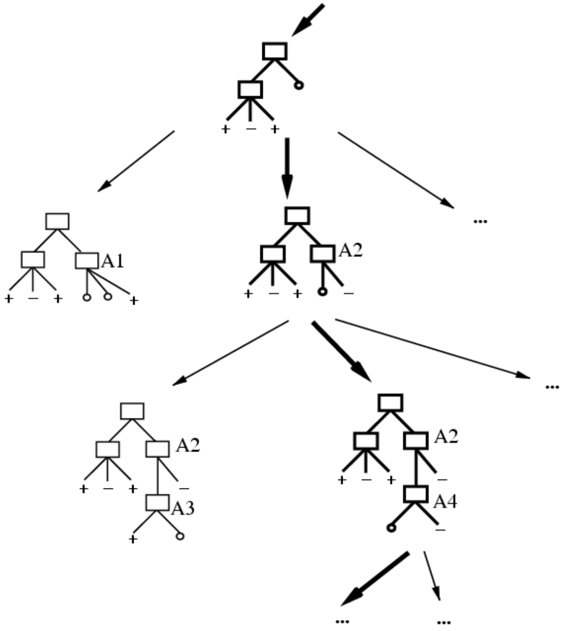


- ID3 performs heuristic search through space of decision trees

- It stops at smallest acceptable tree. Why?

Occam's razor: prefer the simplest hypothesis that fits the data

Properties of ID3



- ID3 performs heuristic search through space of decision trees

- It tends to have the right bias (output short decision trees), but it can still overfit.

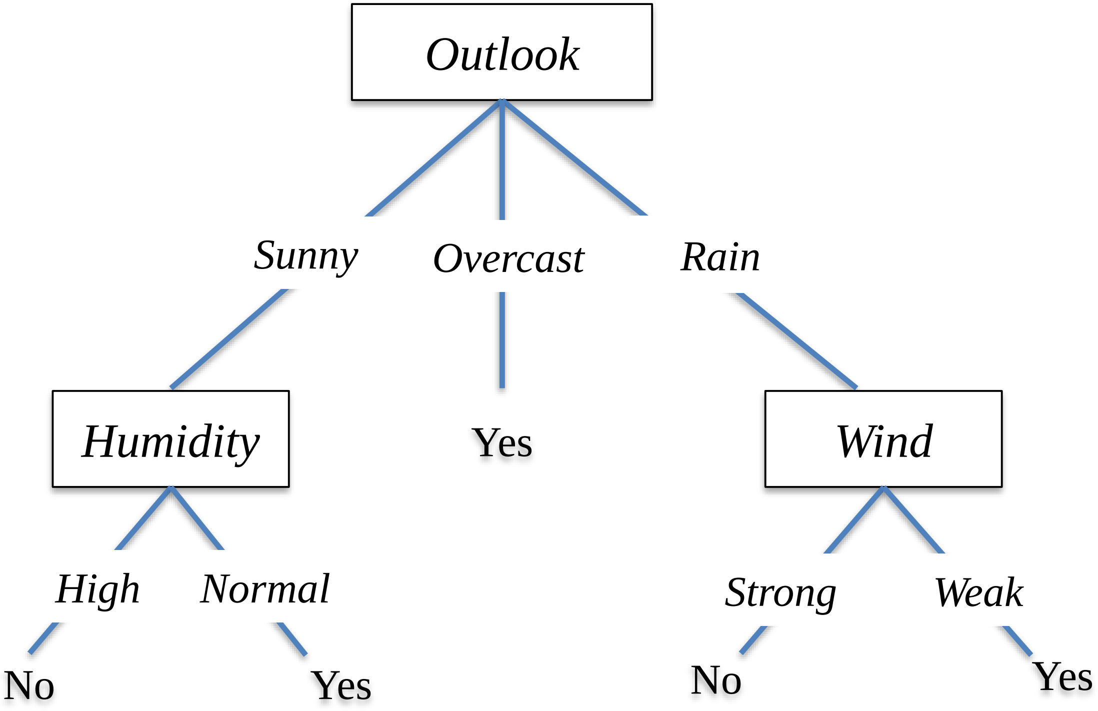
- It might be beneficial to prune the tree by using a validation dataset.

Overfitting in Decision Trees

Consider adding noisy training example #15:

Sunny, Hot, Normal, Strong, PlayTennis = No

What effect on earlier tree?



Properties of ID3

Overfitting could occur because of noisy data and because ID3 is not guaranteed to output a small hypothesis even if one exists.

Consider a hypothesis *h* and its

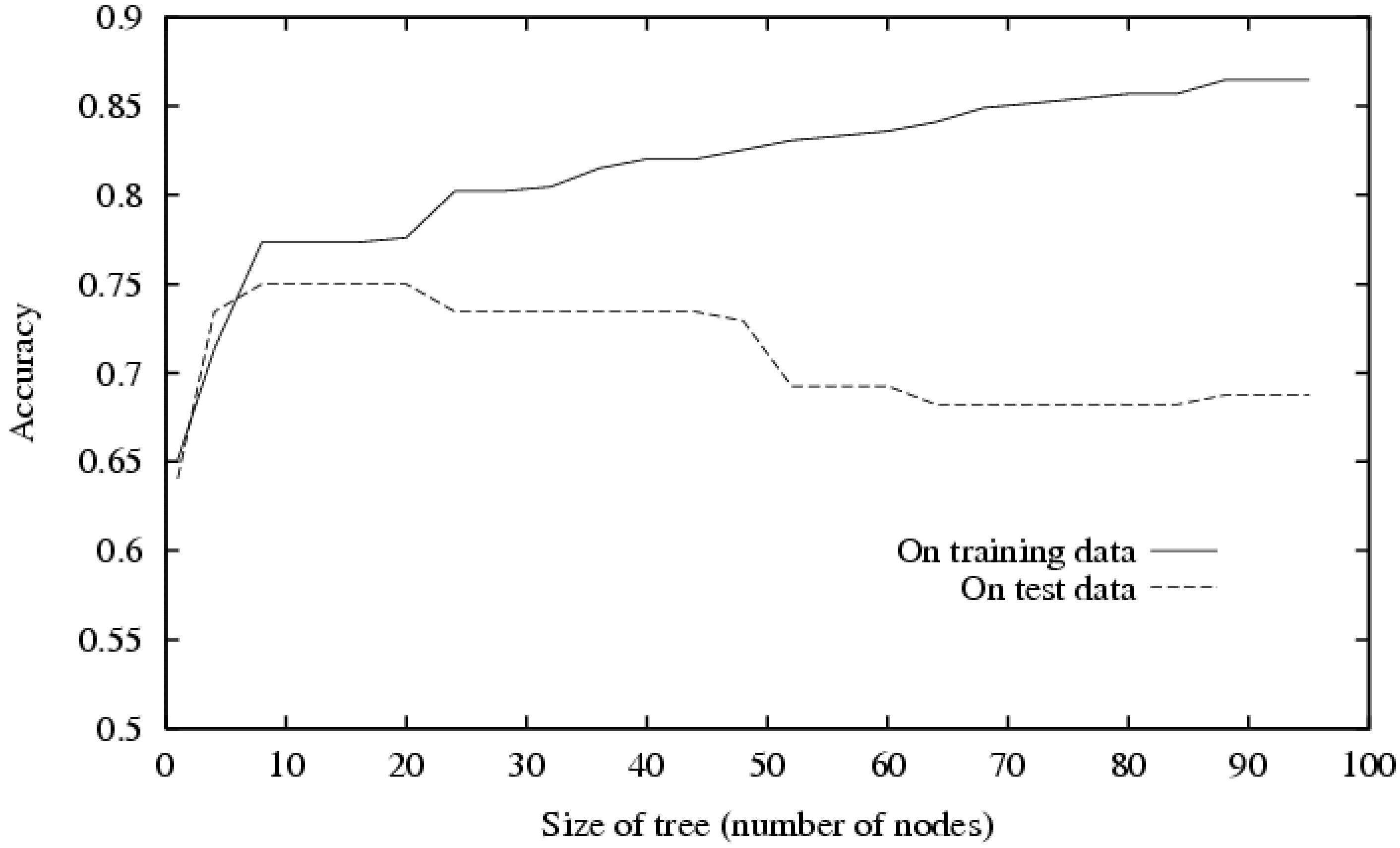
- Error rate over training data: $error_{train}(h)$
- True error rate over all data: $error_{true}(h)$

We say *h* overfits the training data if

$$error_{true}(h) > error_{train}(h)$$

Amount of overfitting = $error_{true}(h) - error_{train}(h)$

Overfitting in Decision Tree Learning



Task: learning which medical patients have a form of diabetes.

Avoiding Overfitting

How can we avoid overfitting?

- Stop growing when data split not statistically significant
- Grow full tree, then post-prune

Key Issues in Machine Learning

- How can we gauge the accuracy of a hypothesis on unseen data?
  - **Occam's razor:** use the *simplest* hypothesis consistent with data! This will help us avoid overfitting.
  - ***Learning theory*** will help us quantify our ability to ***generalize*** as a function of the amount of training data and the hypothesis space
- How do we find the best hypothesis?
  - This is an **algorithmic** question, the main topic of computer science

- How do we choose a hypothesis space?
  - Often we use **prior knowledge** to guide this choice
- How to model applications as machine learning problems? (engineering challenge)