SMAI-S25-L15: Decision Trees

C. V. Jawahar

IIIT Hyderabad

March 4, 2025

Decision Tree Learning

- Decision tree representation
- A specific (ID3) learning algorithm
- Entropy, Information gain
- Overfitting

Good references (Chapter 3 of Mitchel's ML Book. [1]) Also [2]

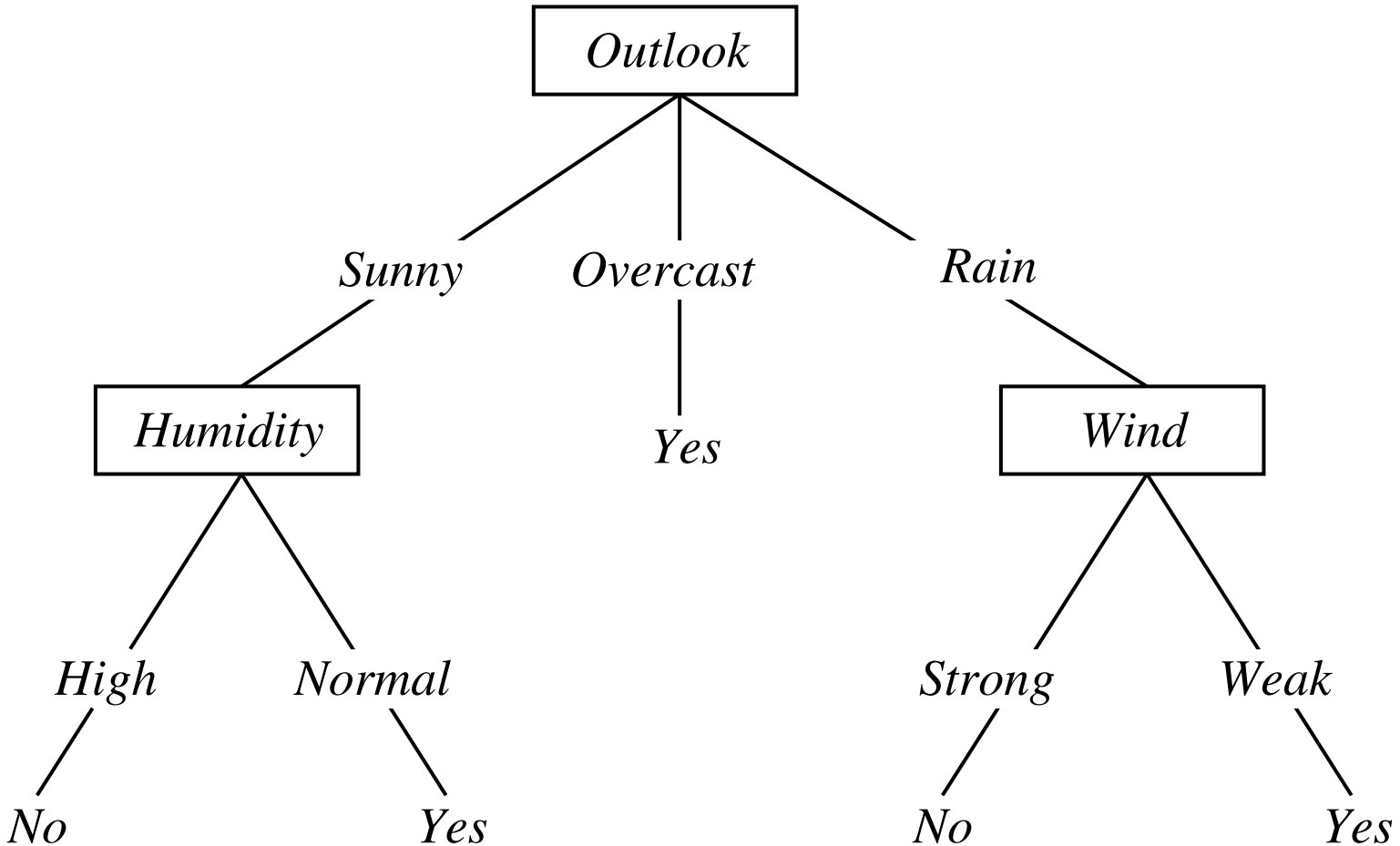
[1] https://www.cs.cmu.edu/ tom/files/MachineLearningTomMitchell.pdf

[2] `https://www.cs.toronto.edu/~axgao/cs486686_f21/lecture_notes/Lecture_07_on_Decision_Trees.pdf`

Training Examples

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

Decision Tree for PlayTennis



Decision Trees

Decision tree representation:

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

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

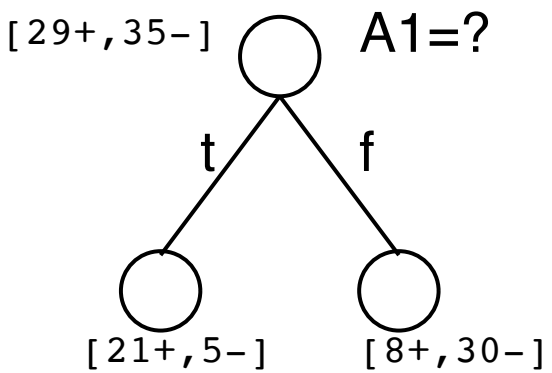
Examples:

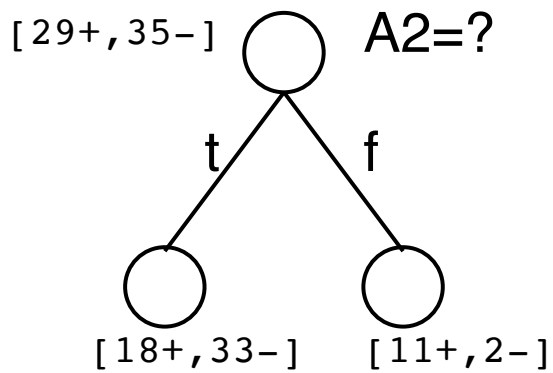
- Equipment or medical diagnosis
- Credit risk analysis
- Modeling calendar scheduling preferences

Top-Down Induction of Decision Trees

Main loop:

1. Select the "best" decision attribute for next node
2. Assign attribute as decision attribute for node
3. For each value of the attribute, create new descendant
4. Sort training examples to leaf nodes
5. If training examples are perfectly classified, then STOP; otherwise, iterate over new leaf nodes





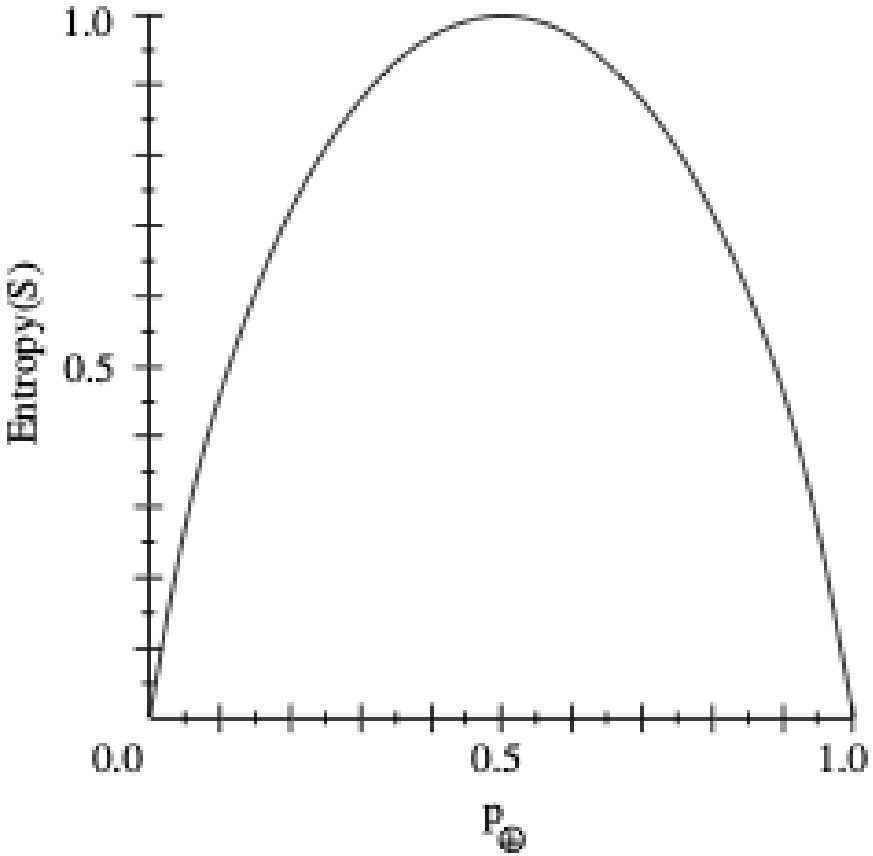
Entropy

We use entropy in choosing which attribute to pick for splitting.

- 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

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

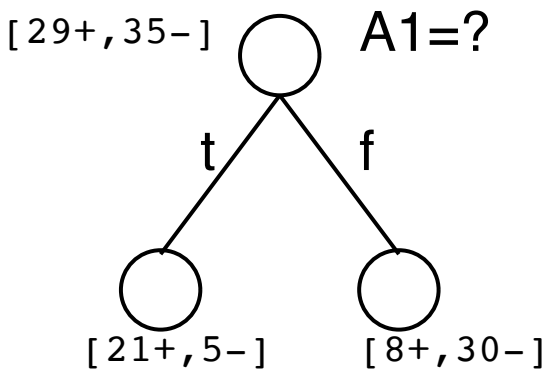
Entropy

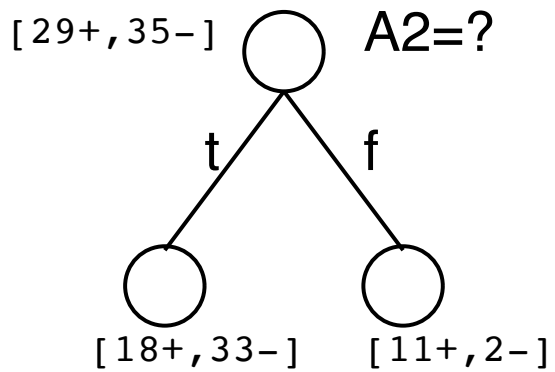


Information Gain

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

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



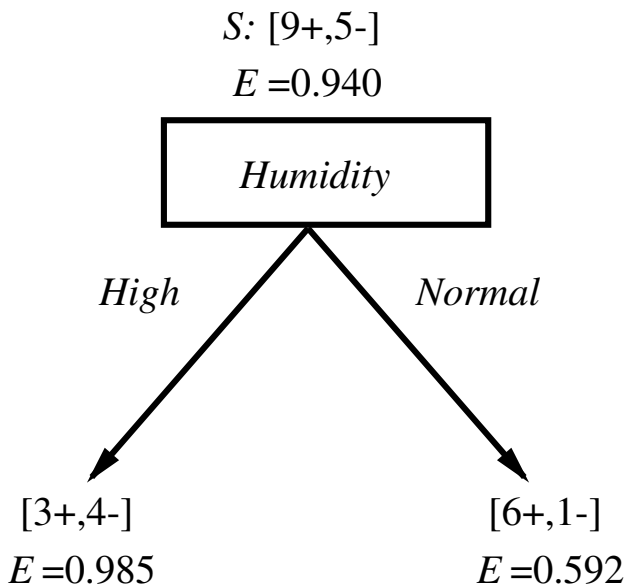


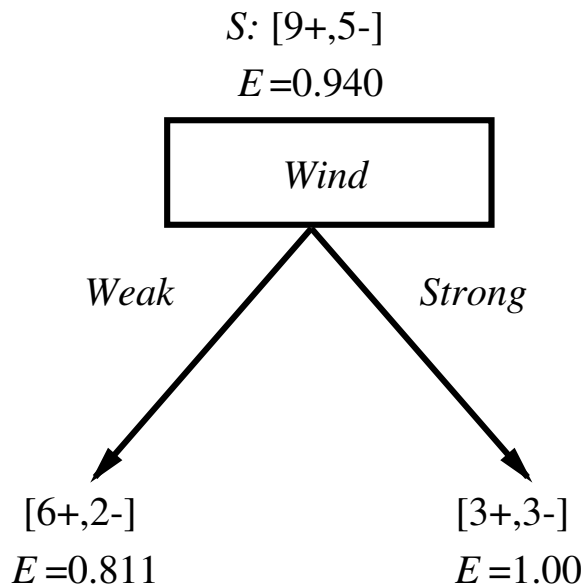
Training Examples

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

Selecting the Next Attribute

Which attribute is the best classifier?





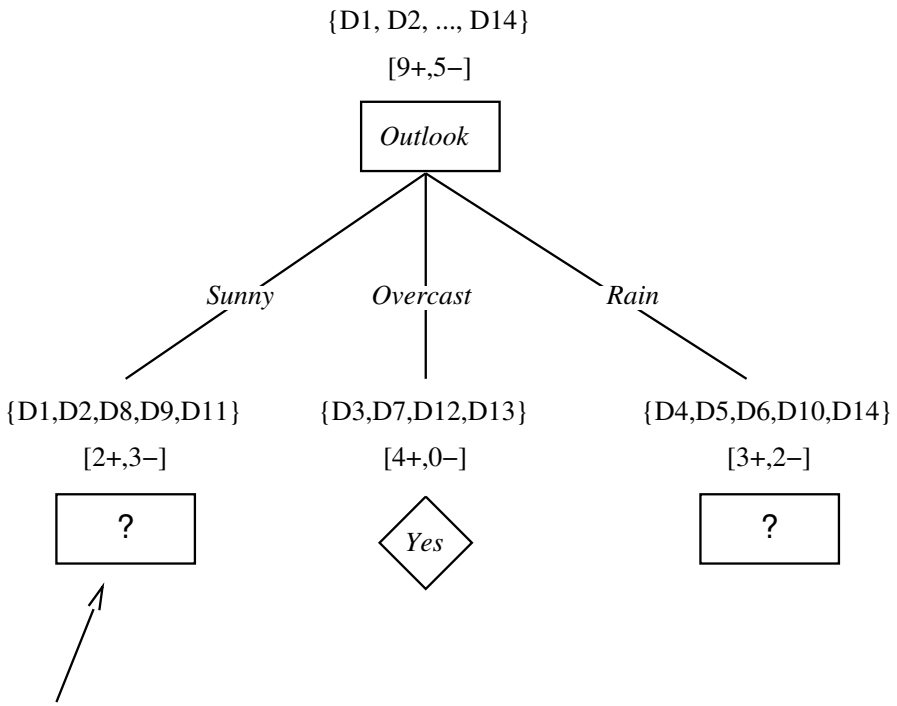
Gain (S, Humidity)

= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)

= .940 - (8/14).811 - (6/14)1.0
= .048

Selecting the Next Attribute



Which attribute should be tested here?

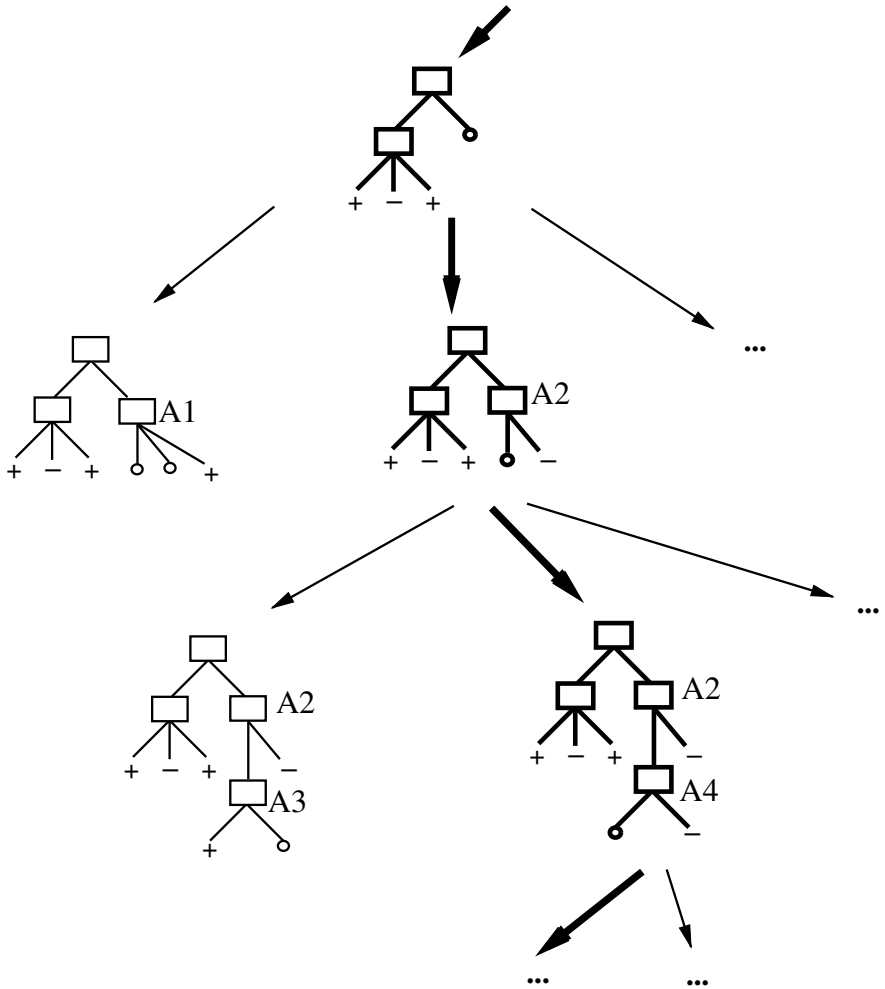
S_{sunny} = {D1,D2,D8,D9,D11}

Gain (S_{sunny} , *Humidity*) = .970 − (3/5) 0.0 − (2/5) 0.0 = .970

Gain (S_{sunny} , *Temperature*) = .970 − (2/5) 0.0 − (2/5) 1.0 − (1/5) 0.0 = .570

Gain (S_{sunny} , *Wind*) = .970 − (2/5) 1.0 − (3/5) .918 = .019

Selecting the Next Attribute



Hypothesis Space Search by ID3

- Hypothesis space is complete!
  - Target function surely in there...
- Outputs a single hypothesis (which one?)
  - Can't play 20 questions...
- No back tracking
  - Local minima...
- Statisically-based search choices
  - Robust to noisy data...
- Inductive bias: approx "prefer shortest tree"

Inductive Bias in ID3

Unbiased?

Not really...

- Preference for short trees, and for those with high information gain attributes near the root
- Bias is a preference for some hypotheses, rather than a restriction of hypothesis space H
- Occam's razor: prefer the shortest hypothesis that fits the data

Occam's Razor

Why prefer short hypotheses?

Argument in favor:

- Fewer short hyps. than long hyps.

  a short hyp that fits data unlikely to be coincidence

  a long hyp that fits data might be coincidence

Argument opposed:

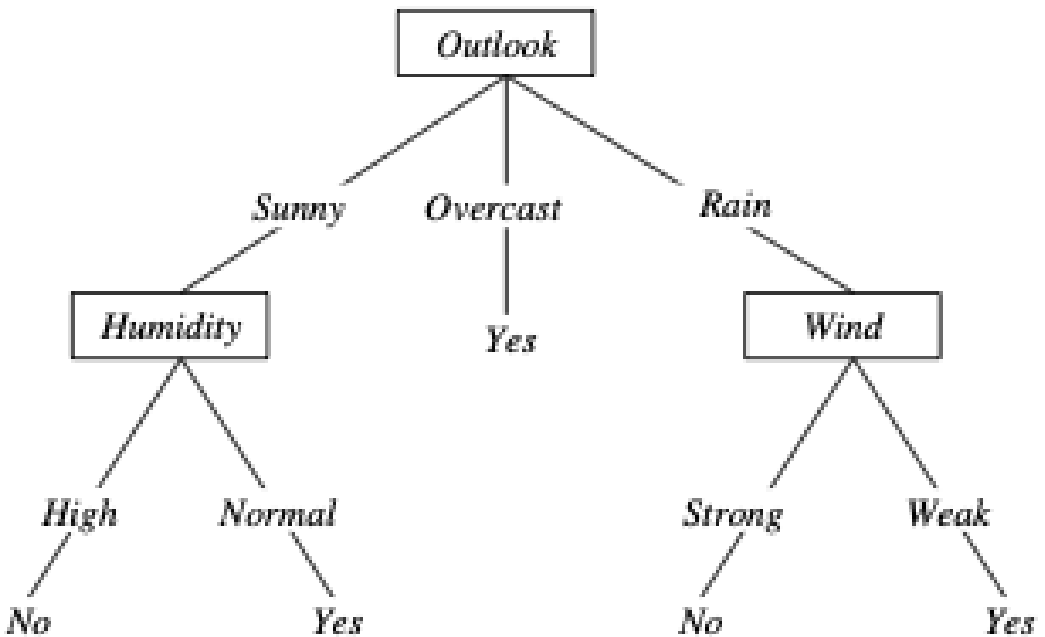
- There are many ways to define small sets of hyps
- e.g., all trees with a prime number of nodes that use attributes beginning with "Z"
- What's so special about small sets based on size of hypothesis??

Overfitting in Decision Trees

Consider adding noisy training example #15:

$$\textit{Sunny},\ \textit{Hot},\ \textit{Normal},\ \textit{Strong},\ \textit{PlayTennis} = \textit{No}$$

What effect on earlier tree?



Overfitting

Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

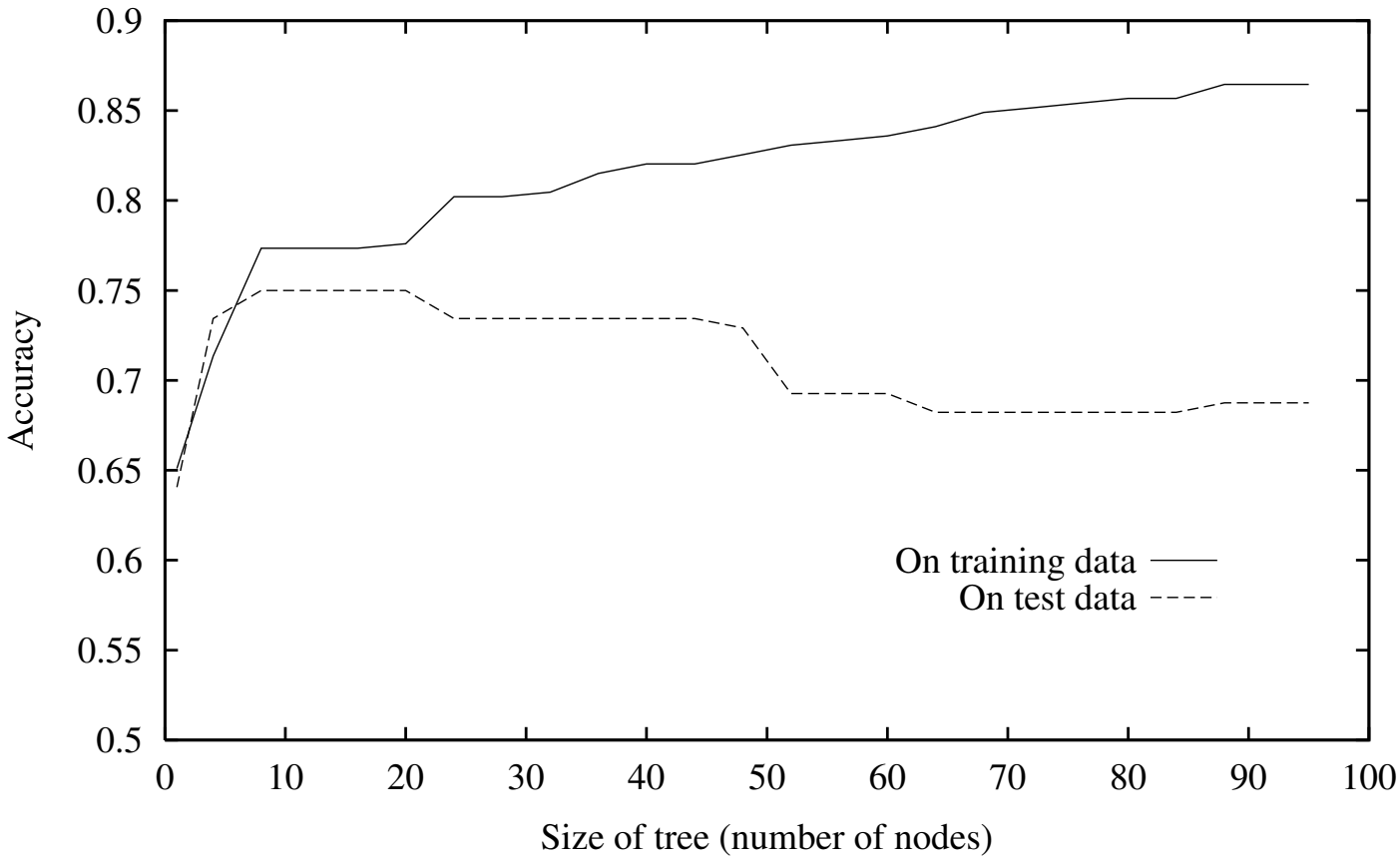
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

Overfitting in Decision Trees



Avoiding Overfitting

- Stop growing when data split is not statistically significant
- Grow full tree, then post-prune
  How can we avoid overfitting?

How to select "best" tree:

- Measure performance over training data
- Measure performance over separate validation data set
- MDL: minimize $size(tree) + size(misclassifications(tree))$

Reduced-Error Pruning

Split data into *training* and *validation* set

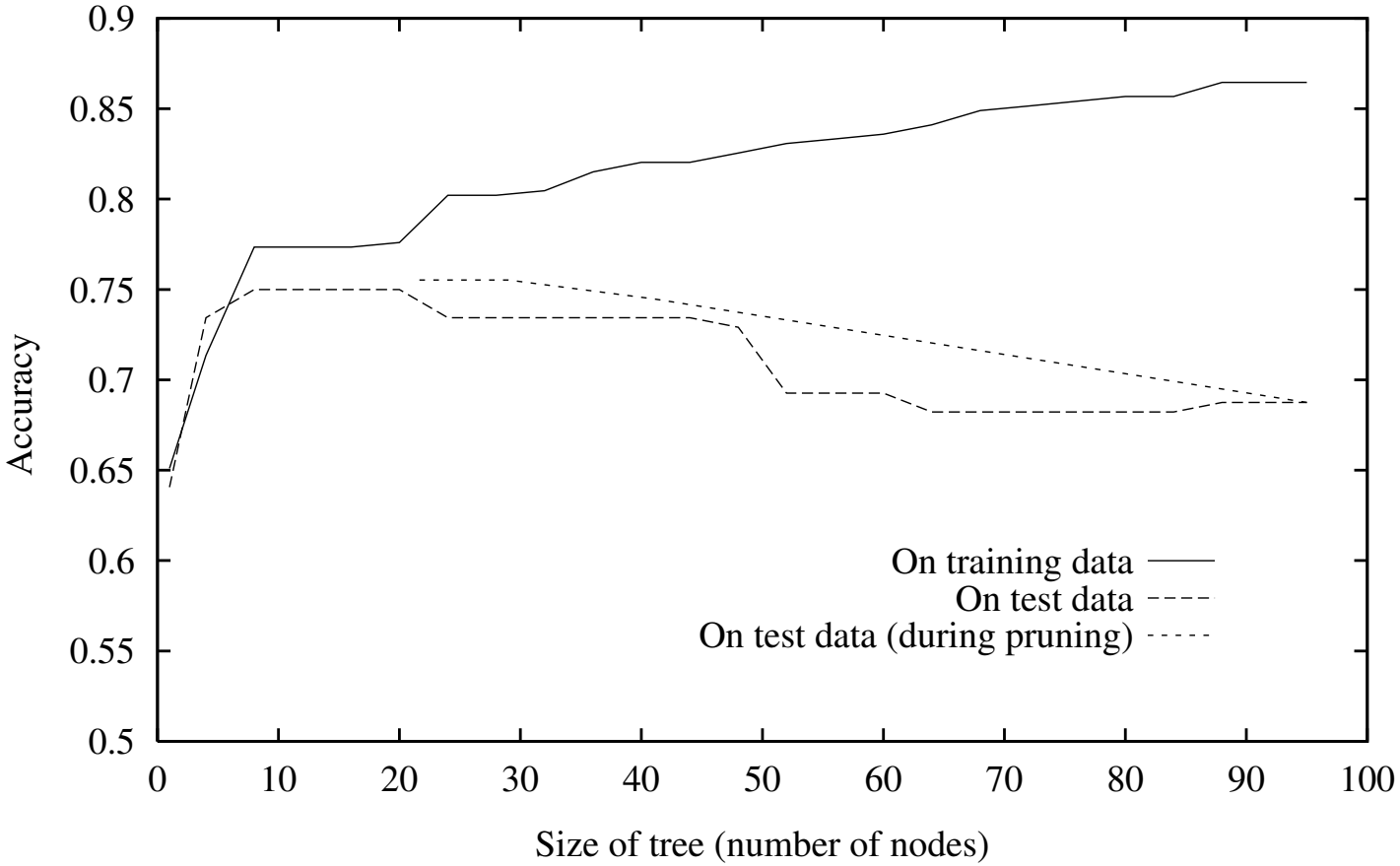
1. Evaluate impact on validation set of pruning each possible node
2. Greedily remove the one that most improves validation set accuracy

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree
- What if data is limited?

Reduce Error Pruning

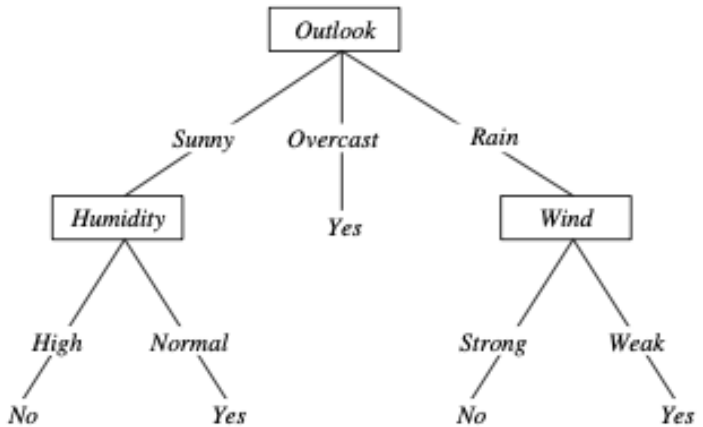


Rule Post-Pruning

1. Convert tree to equivalent set of rules
2. Prune each rule independently
3. Sort final rules into desired sequence

Perhaps most frequently used method (e.g., C4.5)

Converting a Tree to Rules



IF $(Outlook = Sunny)(Humidity = High)$
THEN $PlayTennis = No$
IF $(Outlook = Sunny)(Humidity = Normal)$
THEN $PlayTennis = Yes$
...

Continuous Valued Attributes

Create a discrete attribute to test continuous values

- Example: `(Temperature > 72.3) = true, false`

Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

Attributes with Many Values

Problem:

- If attribute has many values, *Gain* will select it
- Imagine using $Date = Jun\_3\_1996$ as attribute

One approach: use *GainRatio* instead

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

where S_i is subset of S for which A has value v_i

Problem 1 and 2

For the given problem construct the tree (show all entropy calculations!)

Problem 3

Draw decision tree for (i) A EXOR B (ii) (A AND B) OR (C AND D)

Problem 4 5

Consider a problem with two real attributes.

$$((0,0),+),((1,1),-),((2,2),+),((3,3),+)$$

Draw a decision tree that can classify the training data correctly (no need of formally finding; you should be able to guess)
On a 2 D plane, draw the decision boundary and the traing samples.
Should the positive region of a DT be always connected. Argue with an example.