

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning

# CSCE 478/878 Lecture 3: Learning Decision Trees

Stephen Scott

(Adapted from Ethem Alpaydin and Tom Mitchell)

[sscott@cse.unl.edu](mailto:sscott@cse.unl.edu)

# Introduction

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree Representation  
Learning Trees  
Inductive Bias  
Overfitting  
Tree Pruning

*Decision trees form a simple, easily-interpretable, hypothesis*

- Interpretability useful in independent validation and explanation

Quick to train

Quick to evaluate new instances

Effective “off-the-shelf” learning method

Can be combined with boosting, including using “stumps”

# Outline

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning

- Decision tree representation
- Learning trees (ID3/C4.5/CART)
  - Entropy as a splitting criterion
  - Example run of algorithm
  - Regression trees
  - Variations
- Inductive bias
- Overfitting and pruning
- Deriving rules from tree

Decision Tree for *PlayTennis* (Mitchell)CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

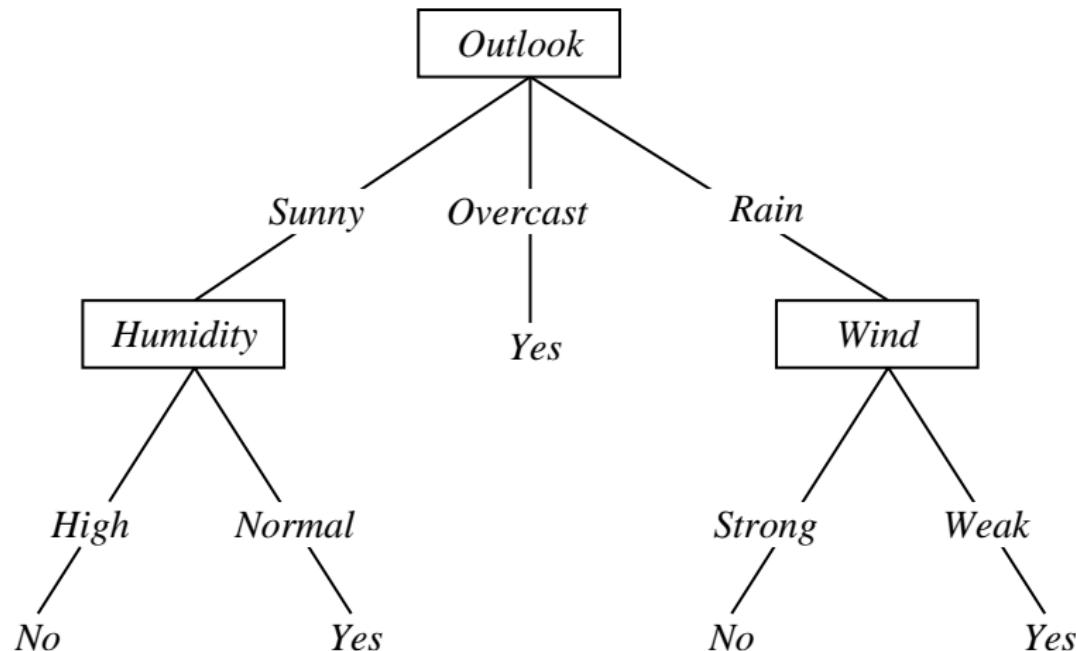
Outline

Tree  
RepresentationLearning  
Trees

Inductive Bias

Overfitting

Tree Pruning



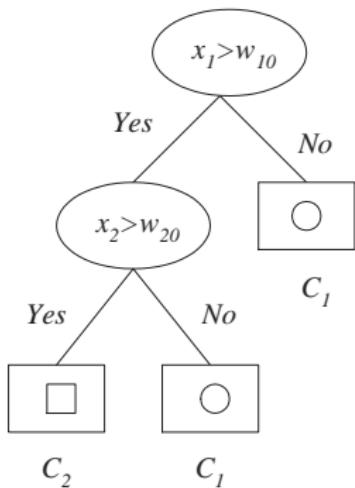
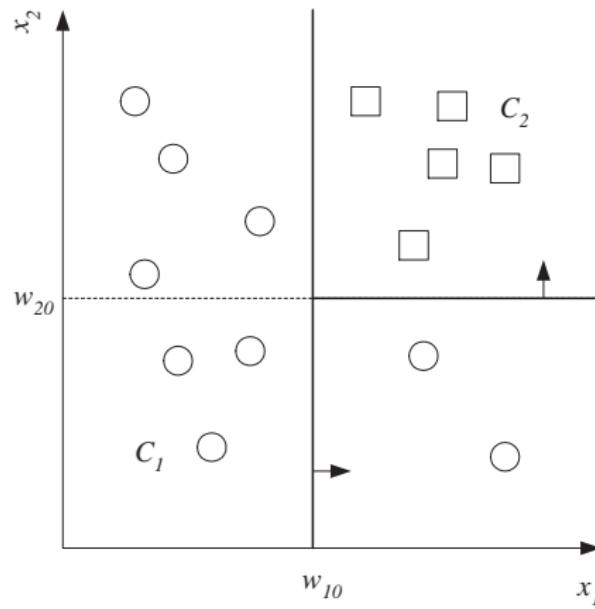
## With Numeric Attributes

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
OutlineTree  
RepresentationLearning  
TreesInductive Bias  
Overfitting

Tree Pruning



# Decision Tree Representation

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline

Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning

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

How would we represent:

- $\wedge, \vee, \text{XOR}$
- $(A \wedge B) \vee (C \wedge \neg D \wedge E)$

# High-Level Learning Algorithm

(ID3, C4.5, CART)

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm

Entropy  
Learning Algorithm  
Example Run  
Regression Trees  
Variations

Inductive Bias

Overfitting

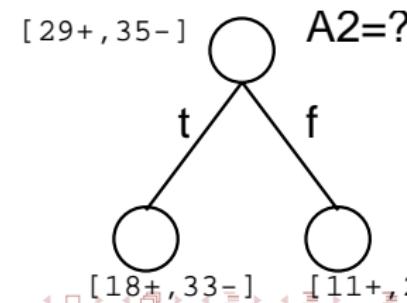
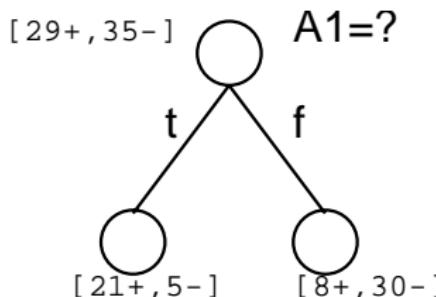
Tree Pruning

7/26

Main loop:

- 1  $A \leftarrow$  the “best” decision attribute for next node  $m$
- 2 Assign  $A$  as decision attribute for  $m$
- 3 For each value of  $A$ , create new descendant of  $m$
- 4 Sort (*partition*) training examples over children based on  $A$ ’s value
- 5 If training examples perfectly classified, Then STOP,  
Else recursively iterate over new child nodes

*Which attribute is best?*



# Entropy

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm

Entropy

Learning Algorithm

Example Run

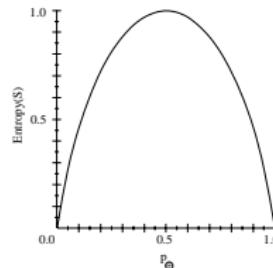
Regression Trees

Variations

Inductive Bias

Overfitting

Tree Pruning



- $\mathcal{X}_m$  is a sample of training examples reaching node  $m$
- $p_m^{\oplus}$  is the proportion of positive examples in  $\mathcal{X}_m$
- $p_m^{\ominus}$  is the proportion of negative examples in  $\mathcal{X}_m$
- *Entropy  $\mathcal{I}_m$  measures the impurity of  $\mathcal{X}_m$*

$$\mathcal{I}_m \equiv -p_m^{\oplus} \log_2 p_m^{\oplus} - p_m^{\ominus} \log_2 p_m^{\ominus}$$

or for  $K$  classes,

$$(9.3) \quad \mathcal{I}_m \equiv - \sum_{i=1}^K p_m^i \log_2 p_m^i$$

# Total Impurity

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm

Entropy

Learning Algorithm

Example Run

Regression Trees

Variations

Inductive Bias

Overfitting

Tree Pruning

- Now can look for an attribute  $A$ , when used to partition  $\mathcal{X}_m$  by value, produces the most pure (lowest-entropy) subsets
  - Weight each subset by relative size
  - E.g., size-3 subsets should carry less influence than size-300 ones
- Let  $N_m = |\mathcal{X}_m|$  = number of instances reaching node  $m$
- Let  $N_{mj}$  = number of these instances with value  $j \in \{1, \dots, n\}$  for attribute  $A$
- Let  $N_{mj}^i$  = number of these instances with label  $i \in \{1, \dots, K\}$
- Let  $p_{mj}^i = N_{mj}^i / N_{mj}$
- Then the *total impurity* is

$$(9.8) \quad \mathcal{I}'_m(A) \equiv - \sum_{j=1}^n \frac{N_{mj}}{N_m} \sum_{i=1}^K p_{mj}^i \log_2 p_{mj}^i$$

# Learning Algorithm

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm  
Entropy

Learning Algorithm

Example Run

Regression Trees

Variations

Inductive Bias

Overfitting

Tree Pruning

```
GenerateTree( $\mathcal{X}$ )
  If NodeEntropy( $\mathcal{X}$ )  $< \theta_I$  /* equation 9.3 */
    Create leaf labelled by majority class in  $\mathcal{X}$ 
    Return
   $i \leftarrow \text{SplitAttribute}(\mathcal{X})$ 
  For each branch of  $x_i$ 
    Find  $\mathcal{X}_i$  falling in branch
    GenerateTree( $\mathcal{X}_i$ )

SplitAttribute( $\mathcal{X}$ )
  MinEnt  $\leftarrow \text{MAX}$ 
  For all attributes  $i = 1, \dots, d$ 
    If  $x_i$  is discrete with  $n$  values
      Split  $\mathcal{X}$  into  $\mathcal{X}_1, \dots, \mathcal{X}_n$  by  $x_i$ 
       $e \leftarrow \text{SplitEntropy}(\mathcal{X}_1, \dots, \mathcal{X}_n)$  /* equation 9.8 */
      If  $e < \text{MinEnt}$   $\text{MinEnt} \leftarrow e$ ;  $\text{bestf} \leftarrow i$ 
    Else /*  $x_i$  is numeric */
      For all possible splits
        Split  $\mathcal{X}$  into  $\mathcal{X}_1, \mathcal{X}_2$  on  $x_i$ 
         $e \leftarrow \text{SplitEntropy}(\mathcal{X}_1, \mathcal{X}_2)$ 
        If  $e < \text{MinEnt}$   $\text{MinEnt} \leftarrow e$ ;  $\text{bestf} \leftarrow i$ 
  Return  $\text{bestf}$ 
```

# Example Run

## Training Examples

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm

Entropy

Learning Algorithm

Example Run

Regression Trees

Variations

Inductive Bias

Overfitting

Tree Pruning

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

# Example Run

## Selecting the First Attribute

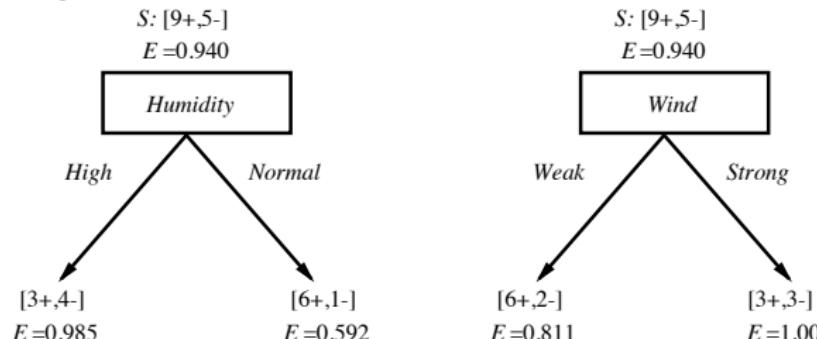
CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
RepresentationLearning  
Trees  
High-Level Algorithm  
Entropy  
Learning AlgorithmExample Run  
Regression Trees  
VariationsInductive Bias  
Overfitting

Tree Pruning

### Comparing *Humidity* to *Wind*:



$$\mathcal{I}'_m(\text{Humidity}) = (7/14)0.985 + (7/14)0.592 = 0.789$$

$$\mathcal{I}'_m(\text{Wind}) = (8/14)0.811 + (6/14)1.000 = 0.892$$

$$\mathcal{I}'_m(\text{Outlook}) = (5/14)0.971 + (4/14)0.0 + (5/14)0.971 = 0.694$$

$$\mathcal{I}'_m(\text{Temp}) = (4/14)1.000 + (6/14)0.918 + (4/14)0.811 = 0.911$$

# Example Run

## Selecting the Next Attribute

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

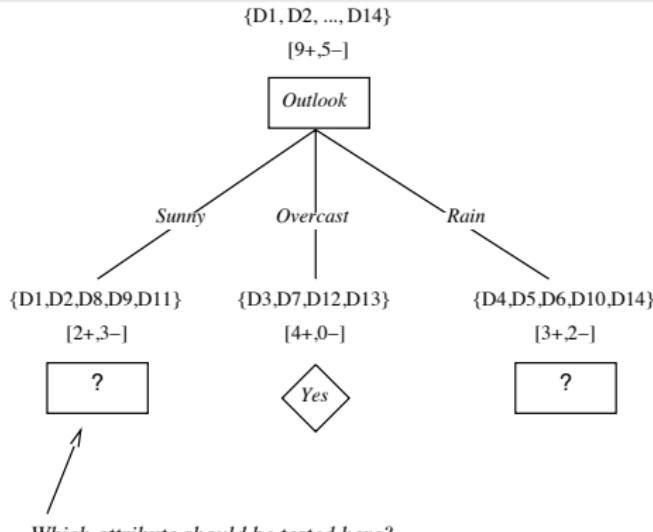
Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees  
High-Level Algorithm  
Entropy  
Learning Algorithm

Example Run  
Regression Trees  
Variations

Inductive Bias  
Overfitting  
Tree Pruning



$$\mathcal{X}_m = \{D_1, D_2, D_8, D_9, D_{11}\}$$

$$\mathcal{I}'_m(\text{Humidity}) = (3/5)0.0 + (2/14)0.0 = 0.0$$

$$\mathcal{I}'_m(\text{Wind}) = (2/5)1.0 + (3/5)0.918 = 0.951$$

$$\mathcal{I}'_m(\text{Temp}) = (2/5)0.0 + (2/5)1.0 + (1/5)0.0 = 0.400$$

# Regression Trees

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees  
High-Level Algorithm  
Entropy  
Learning Algorithm  
Example Run  
Regression Trees  
Variations

Inductive Bias  
Overfitting  
Tree Pruning

- A *regression tree* is similar to a decision tree, but with real-valued labels at the leaves
- To measure impurity at a node  $m$ , replace entropy with variance of labels:

$$E_m \equiv \frac{1}{N_m} \sum_{(\mathbf{x}^t, r^t) \in \mathcal{X}_m} (r^t - g_m)^2 ,$$

where  $g_m$  is the mean (or median) label in  $\mathcal{X}_m$

# Regression Trees (cont'd)

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees  
High-Level Algorithm  
Entropy  
Learning Algorithm  
Example Run  
Regression Trees

Variations

Inductive Bias  
Overfitting  
Tree Pruning

- Now can adapt Eq. (9.8) from classification to regression:

$$E'_m(A) \equiv \sum_{j=1}^n \frac{N_{mj}}{N_m} \left( \frac{1}{N_{mj}} \sum_{(\mathbf{x}^t, r^t) \in \mathcal{X}_{mj}} (r^t - g_{mj})^2 \right)$$

$$(9.14) \quad = \frac{1}{N_m} \sum_{j=1}^n \sum_{(\mathbf{x}^t, r^t) \in \mathcal{X}_{mj}} (r^t - g_{mj})^2 ,$$

where  $j$  iterates over the values of attribute  $A$

- When variance of a subset is sufficiently low, insert leaf with mean or median label as constant value

# Continuous-Valued Attributes

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction

Outline

Tree  
Representation

Learning  
Trees

High-Level Algorithm

Entropy

Learning Algorithm

Example Run

Regression Trees

Variations

Inductive Bias

Overfitting

Tree Pruning

Use threshold to map continuous to boolean, e.g.

$(Temperature > 72.3) \in \{t, f\}$

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

- Can show that threshold minimizing impurity must lie between two adjacent attribute values in  $\mathcal{X}$  such that label changed, so try all such values, e.g.,  $(48 + 60)/2 = 54$  and  $(80 + 90)/2 = 85$
- Now (dynamically) replace continuous attribute with boolean attributes  $Temperature_{>54}$  and  $Temperature_{>85}$  and run algorithm normally
- Other options: Split into multiple intervals rather than two; use thresholded linear combinations of continuous attributes (Sec 9.6)

# Attributes with Many Values

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees  
High-Level Algorithm  
Entropy  
Learning Algorithm  
Example Run

Regression Trees  
Variations

Inductive Bias

Overfitting

Tree Pruning

## Problem:

- If attribute  $A$  has many values, it might artificially minimize  $\mathcal{I}'_m(A)$
- E.g., if  $Date$  is attribute,  $\mathcal{I}'_m(A)$  will be low because several very small subsets will be created

One approach: penalize  $A$  with a measure of *split information*, which measures how broadly and uniformly attribute  $A$  splits data:

$$\mathcal{S}(A) \equiv - \sum_{j=1}^n \frac{N_{mj}}{N_m} \log_2 \frac{N_{mj}}{N_m} \in [0, \log_2 n]$$

# Unknown Attribute Values

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees  
High-Level Algorithm  
Entropy  
Learning Algorithm  
Example Run  
Regression Trees  
Variations

Inductive Bias  
Overfitting  
Tree Pruning

What if a training example is missing a value of  $A$ ?

Use it anyway (sift it through tree)

- If node  $m$  tests  $A$ , assign most common value of  $A$  among other training examples sifted to  $m$
- Assign most common value of  $A$  among other examples with same target value (either overall or at  $m$ )
- Assign probability  $p_j$  to each possible value  $v_j$  of  $A$ 
  - Assign fraction  $p_j$  of example to each descendant in tree

Classify new examples in same fashion

# Inductive Bias of Learning Algorithm

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning

- Hypothesis space  $\mathcal{H}$  is complete, in that *any* function can be represented
- Thus inductive bias does not come from restricting  $\mathcal{H}$ , but from *preferring* some trees over others
  - Tends to prefer shorter trees
  - Computationally intractable to find a guaranteed shortest tree, so heuristically apply greedy approach to locally minimize impurity

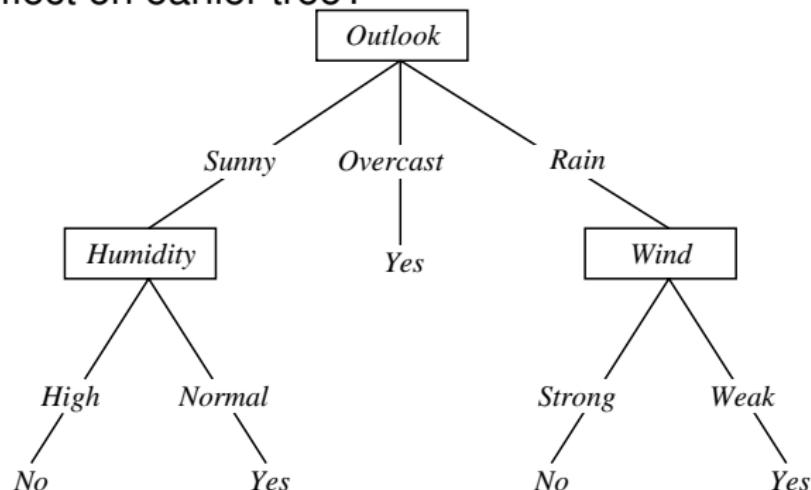
# Overfitting

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation  
Learning  
Trees  
Inductive Bias  
Overfitting  
Tree Pruning

- Consider adding noisy training example #15:  
*Sunny, Hot, Normal, Strong, PlayTennis = No*
- What effect on earlier tree?



- Expect old tree to generalize better since new one fits noisy example

# Overfitting (cont'd)

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning

- Consider error of hypothesis  $h$  over
  - training data (empirical error):  $error_{train}(h)$
  - entire distribution  $\mathcal{D}$  of data (generalization error):  $error_{\mathcal{D}}(h)$
- Hypothesis  $h \in \mathcal{H}$  *overfits* training data if there is an alternative hypothesis  $h' \in \mathcal{H}$  such that

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

and

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

# Overfitting (cont'd)

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

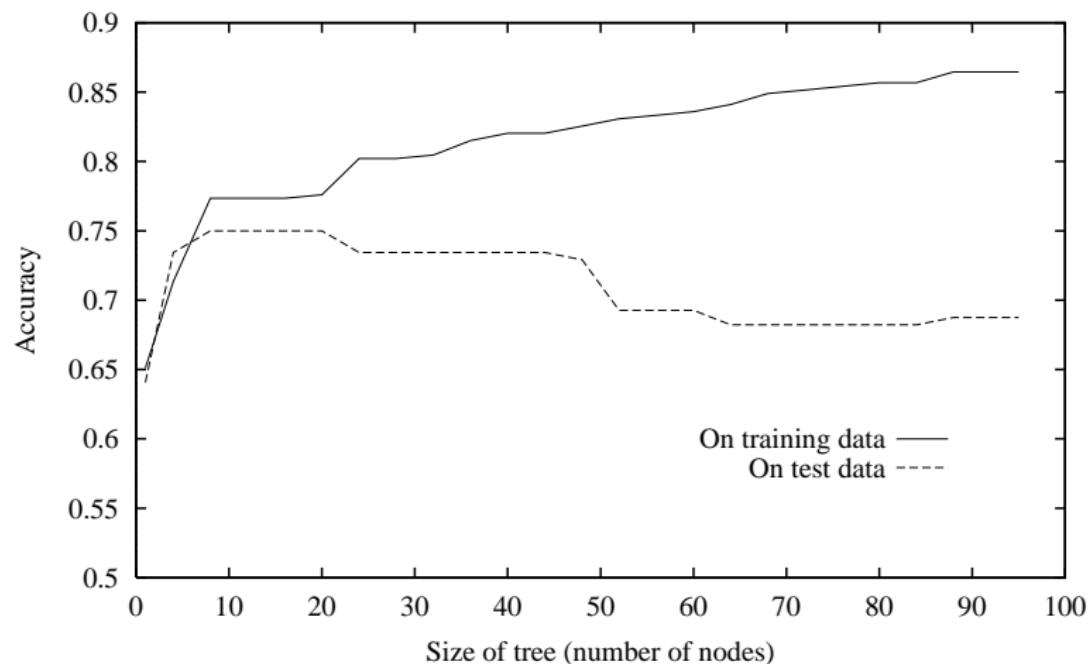
Introduction  
Outline  
Tree  
Representation

Learning  
Trees

Inductive Bias

Overfitting

Tree Pruning



# Pruning to Avoid Overfitting

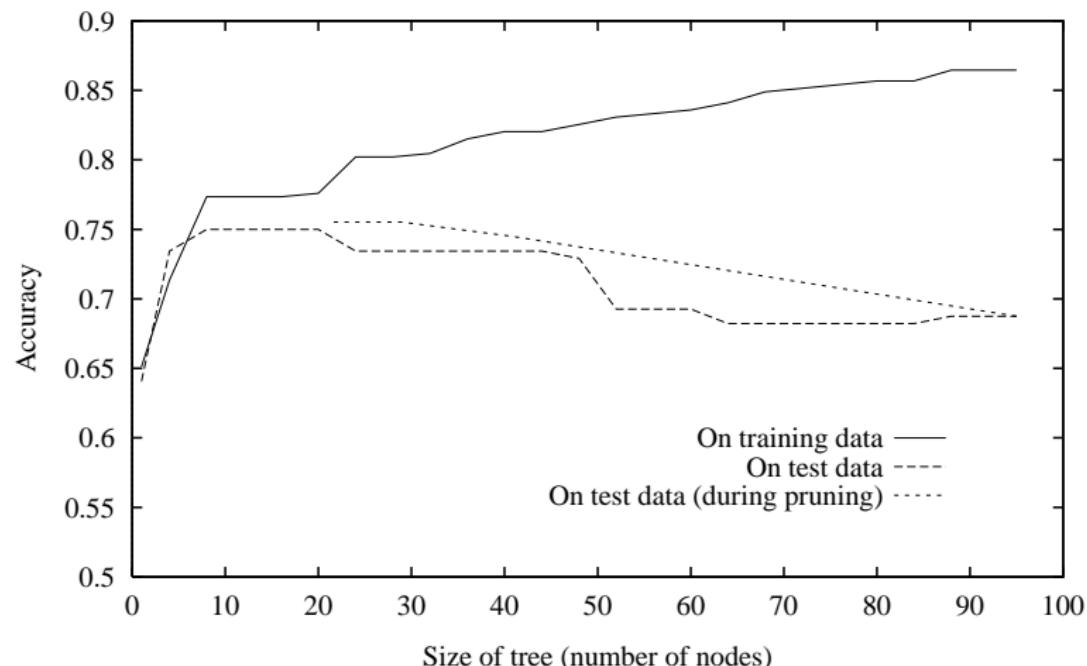
- To prevent trees from growing too much and overfitting the data, we can *prune* them
  - In spirit of Occam's Razor, minimum description length
- In *prepruning*, we allow skipping a recursive call on set  $\mathcal{X}_m$  and instead insert a leaf, even if  $\mathcal{X}_m$  is not pure
  - Can do this when entropy (or variance) is below a threshold ( $\theta_I$  in pseudocode)
  - Can do this when  $|\mathcal{X}_m|$  is below a threshold, e.g., 5
- In *postpruning*, we grow the tree until it has zero error on training set and then prune it back afterwards
  - First, set aside a *pruning set* not used in initial training
  - Then repeat until 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

# Pruning Example

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree Representation  
Learning Trees  
Inductive Bias  
Overfitting  
Tree Pruning  
Rule Postpruning



# Rule Postpruning

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

Introduction  
Outline  
Tree  
Representation

Learning  
Trees

Inductive Bias  
Overfitting

Tree Pruning  
Rule Postpruning

- Convert tree to equivalent set of rules
- Prune each rule independently of others by removing selected preconditions (the ones that improve accuracy the most)
- Sort final rules into desired sequence for use

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

# Converting A Tree to Rules

CSCE  
478/878  
Lecture 3:  
Learning  
Decision  
Trees

Stephen Scott

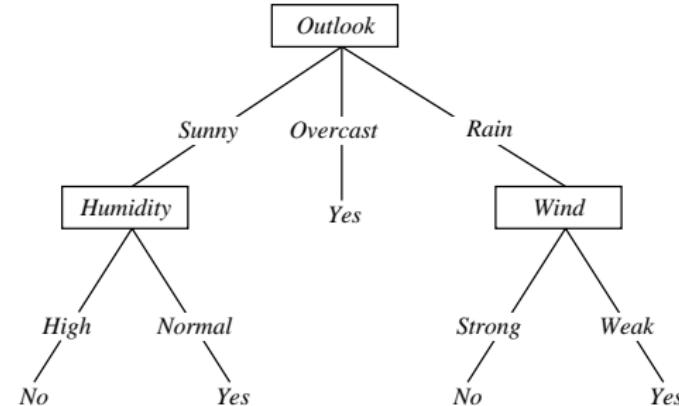
Introduction  
Outline  
Tree  
Representation

Learning  
Trees

Inductive Bias  
Overfitting

Tree Pruning

Rule Postpruning



**IF**  $(Outlook = \text{Sunny}) \wedge (Humidity = \text{High})$   
**THEN**  $\text{PlayTennis} = \text{No}$

**IF**  $(Outlook = \text{Sunny}) \wedge (Humidity = \text{Normal})$   
**THEN**  $\text{PlayTennis} = \text{Yes}$

...