

# Classification Outline

- Classification Problem Overview
- Algorithm performance measures
- Classification Techniques
  - Decision Trees
    - ID3, MDL Pruning, CART, Gini
  - Neural networks
    - Backpropagation, radial basis functions
  - Rule inducers
    - 1R
  - PRISM algorithm
- Ensemble techniques (Bagging, Boosting, Stacking)

# Classification Problem

- Given a database  $D = \{t_1, t_2, \dots, t_n\}$  and a set of classes  $C = \{C_1, \dots, C_m\}$ , the ***Classification Problem*** is to define a mapping  $f: D \rightarrow C$  where each  $t_i$  is assigned to one class.
- Actually divides  $D$  into ***equivalence classes***.
- Our aim is to find this function that minimizes generalization error

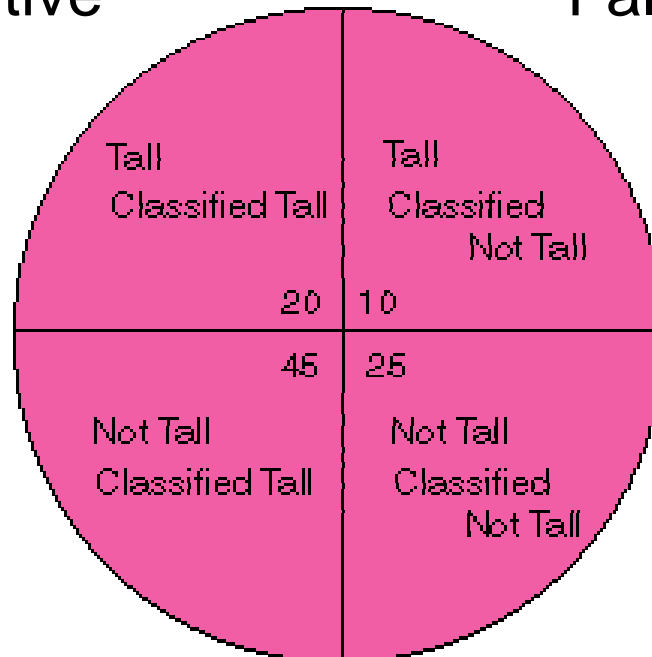
# Classification Examples

- Teachers classify students' by grades.
- Is a session/transaction malignant or benign
- Identify individuals with credit risks.
- Allocate a web session as being a buyer or browser
- Is a group of transactions fraudulent

# Classification Performance

True Positive

False Negative



False Positive

True Negative

# Confusion Matrix Example

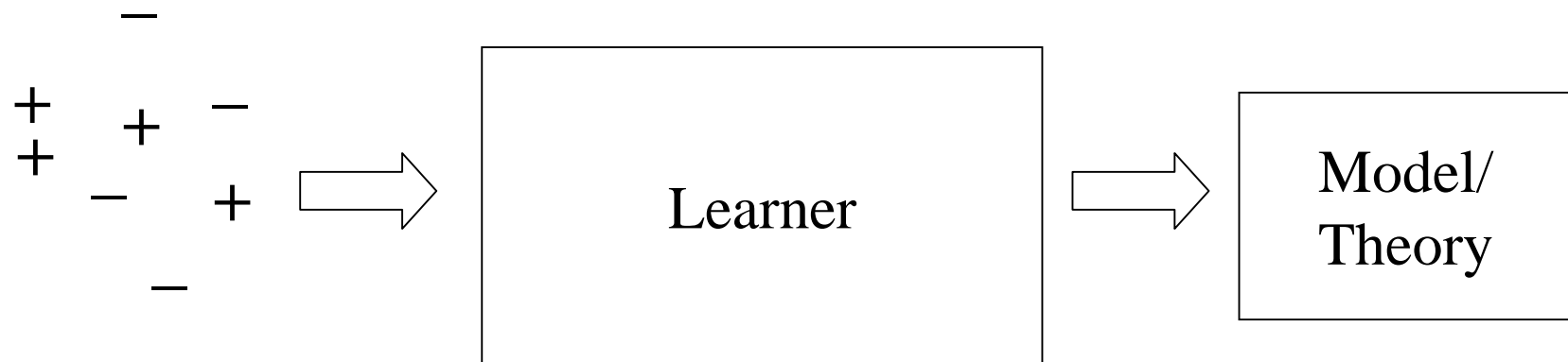
Using height data example with Output1 correct and Output2 actual assignment

<b>Actual Membership</b>	<b>Assignment</b>		
	<b>Short</b>	<b>Medium</b>	<b>Tall</b>
Short	0	4	0
Medium	0	5	3
Tall	0	1	2

# Classification

## Functional Overview – Learning Stage

Examples



Age, ... income, {**Default** | **NoDefault**}

Case A. 30, ..., \$110K, Default

Case B. 50, ..., \$110K, NoDefault

Case C. 45, ..., \$90K, NoDefault

Case A. 32, ..., \$105K, Default

Case B. 49, ..., \$82K, NoDefault

Case C. 29, ..., \$50K, NoDefault

# Classification

## Functional Overview – Application Stage

Unlabeled Examples

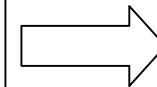
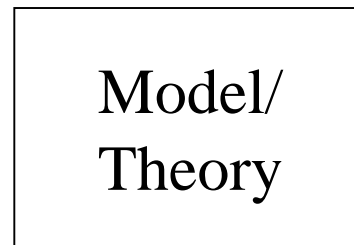
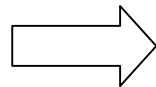
Predictions

Age, ... income, {**Default** | **NoDefault**}

Case zx. 29, ..., \$113K, ?

Case zy. 42, ..., \$81K, ?

Case zz. 41, ..., \$92K, ?



zx, Default

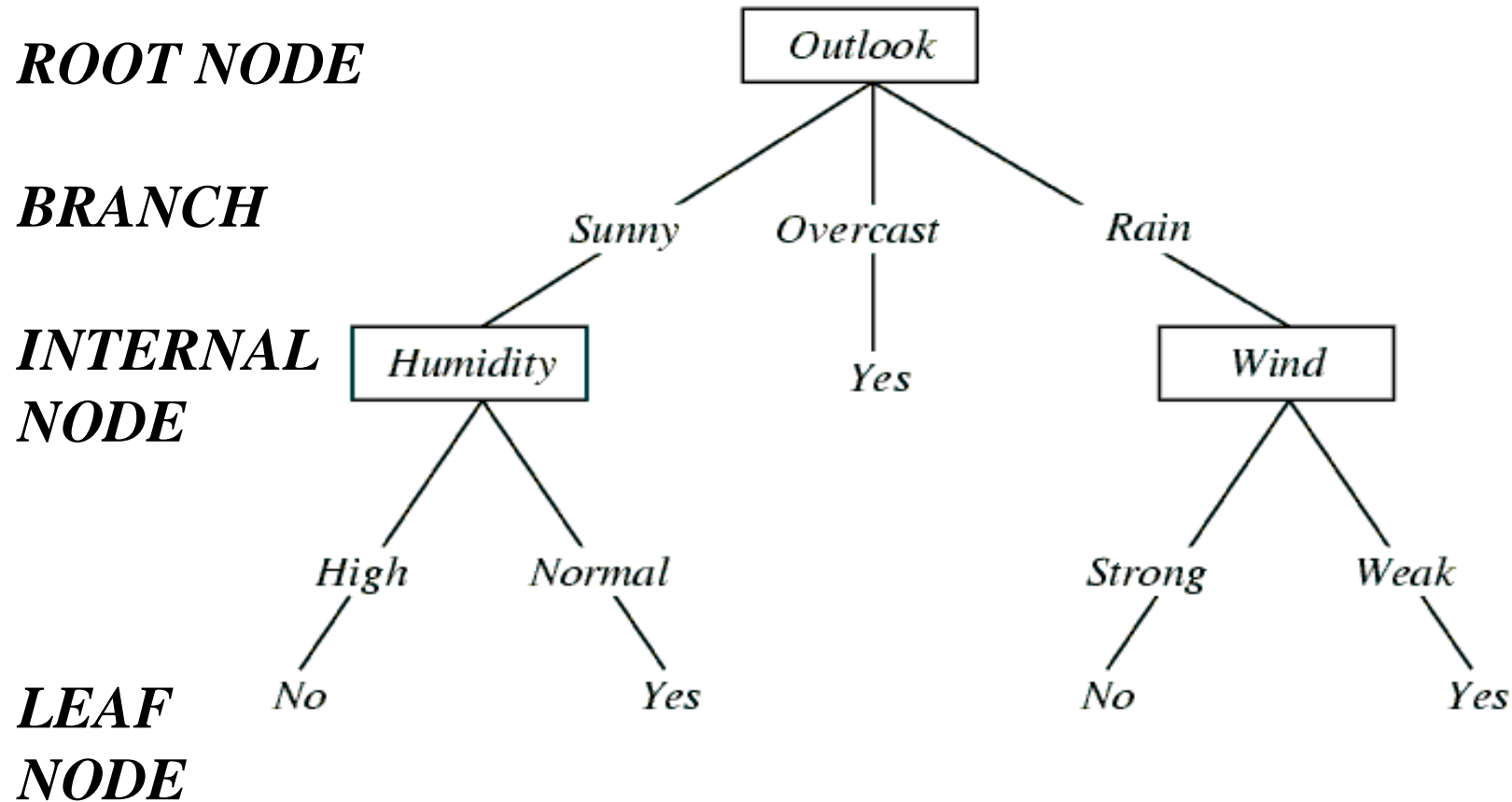
zy. NoDefault

zz. NoDefault

# Classification Terminology

- Attribute, instances
- Training set, test set
- Training set accuracy
- Test set accuracy
- X-fold validation
- Confusion Matrix
- False positive
- False negative
- Cost sensitive classification
- Independent and dependent attributes

# Decision Tree For Playing Tennis



Disjunction of conjunctions

# How and When To Use Decision Trees

- How
  - Classification
    - Gain insights into why customers default on loans
  - Prediction
    - Screen customer loan applications
  - Feature/column/attribute selection
  - To explain other learning techniques
- When
  - Instances are attribute value pairs
  - Discrete target function
  - Noisy training data or missing values

# Classification Using Decision Trees

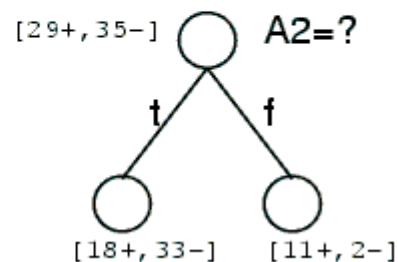
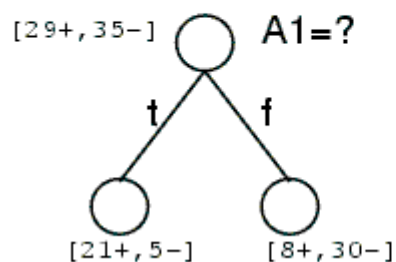
- ***Partitioning based:*** Divide search space into rectangular regions.
- Tuple placed into class based on the region within which it falls.
- DT approaches differ in how the tree is built: ***DT Induction***
- Internal nodes associated with attribute and arcs with values for that attribute.
- Algorithms: ID3, C4.5, CART

# Top-Down Tree Induction

Main loop:

1.  $A \leftarrow$  the “best” decision attribute for next *node*
2. Assign  $A$  as decision attribute for *node*
3. For each value of  $A$ , create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?



# Decision Tree

Given:

- $D = \{t_1, \dots, t_n\}$  where  $t_i = \langle t_{i1}, \dots, t_{ih} \rangle$
- Database schema contains  $\{A_1, A_2, \dots, A_h\}$
- Classes  $C = \{C_1, \dots, C_m\}$

***Decision or Classification Tree*** is a tree associated with  $D$  such that

- Each internal node is labeled with attribute,  $A_i$
- Each arc is labeled with predicate which can be applied to attribute at parent
- Each leaf node is labeled with a class,  $C_j$

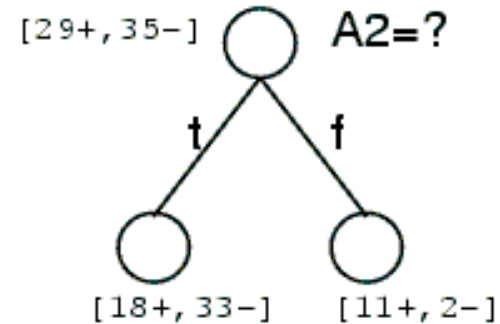
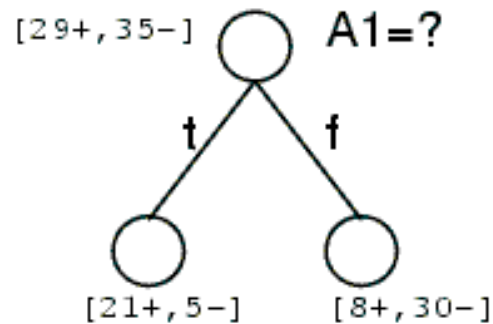
# Which Column and Split Point?

- Multitude of techniques:
  - Entropy/Information gain
  - Chi square test (CHAID)
    - Test of independence
  - GINI index

# Information Gain

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

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



# Entropy

*Entropy*( $S$ ) = expected number of bits needed to encode class ( $\oplus$  or  $\ominus$ ) of randomly drawn member of  $S$  (under the optimal, shortest-length code)

Why?

Information theory: optimal length code assigns  $-\log_2 p$  bits to message having probability  $p$ .

So, expected number of bits to encode  $\oplus$  or  $\ominus$  of random member of  $S$ :

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

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

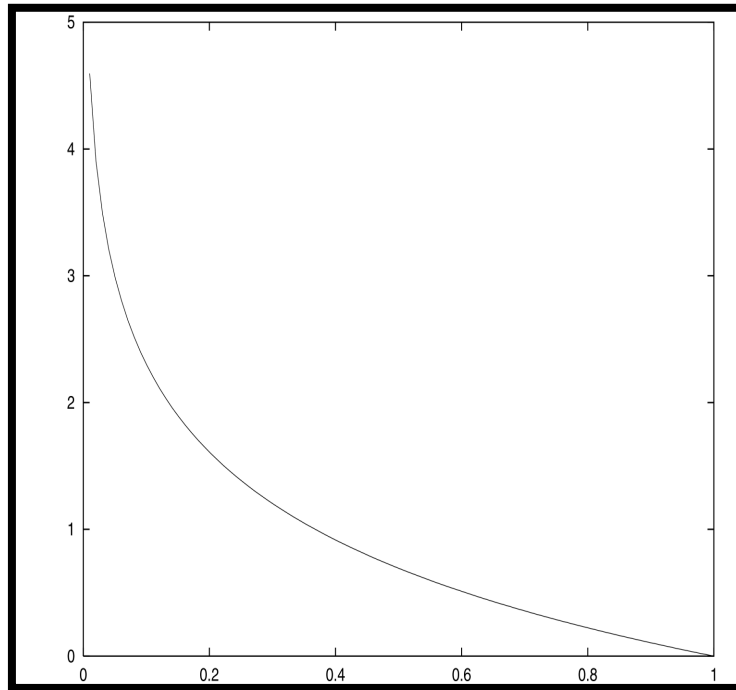
# Information/Entropy

- Given probabilities  $p_1, p_2, \dots, p_s$  whose sum is 1, *Entropy* is defined as:

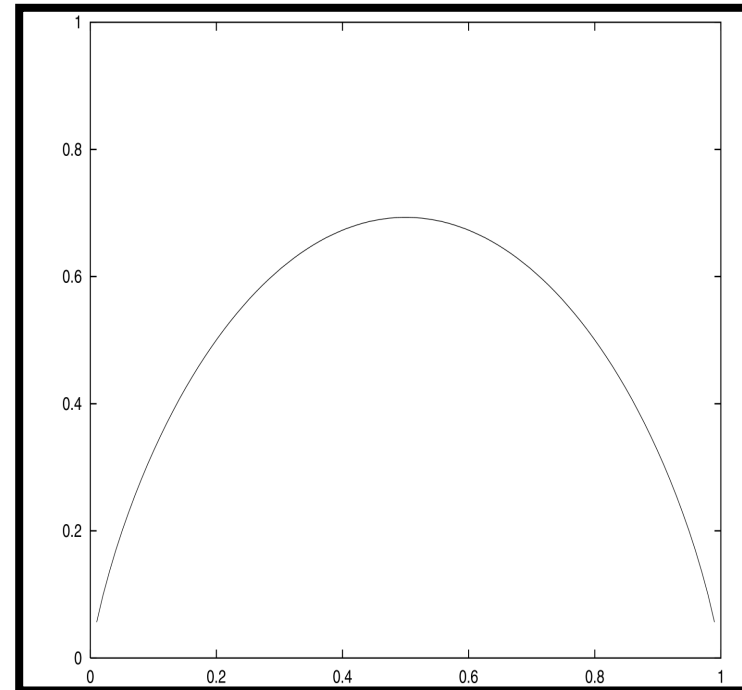
$$H(p_1, p_2, \dots, p_s) = \sum_{i=1}^s (p_i \log(1/p_i))$$

- Entropy measures the amount of randomness or surprise or uncertainty.
- Goal in classification
  - no surprise
  - entropy = 0

# Entropy



$\log(1/p)$



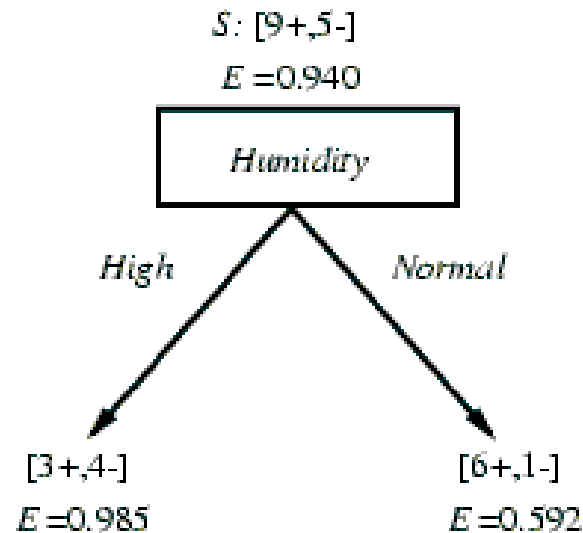
$H(p, 1-p)$

# Data Set

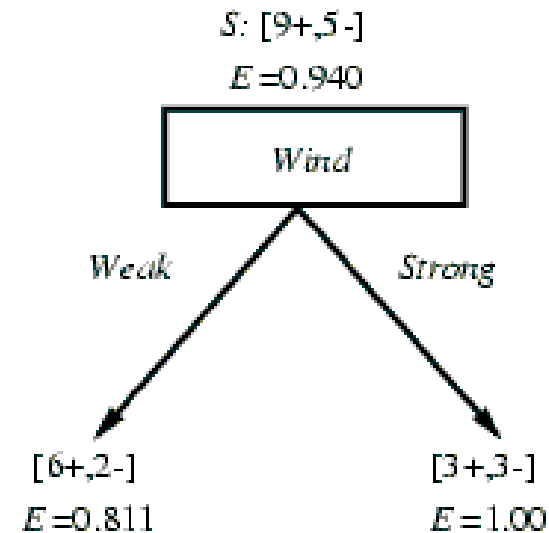
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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

# Choosing the Next Attribute - 1

Which attribute is the best classifier?

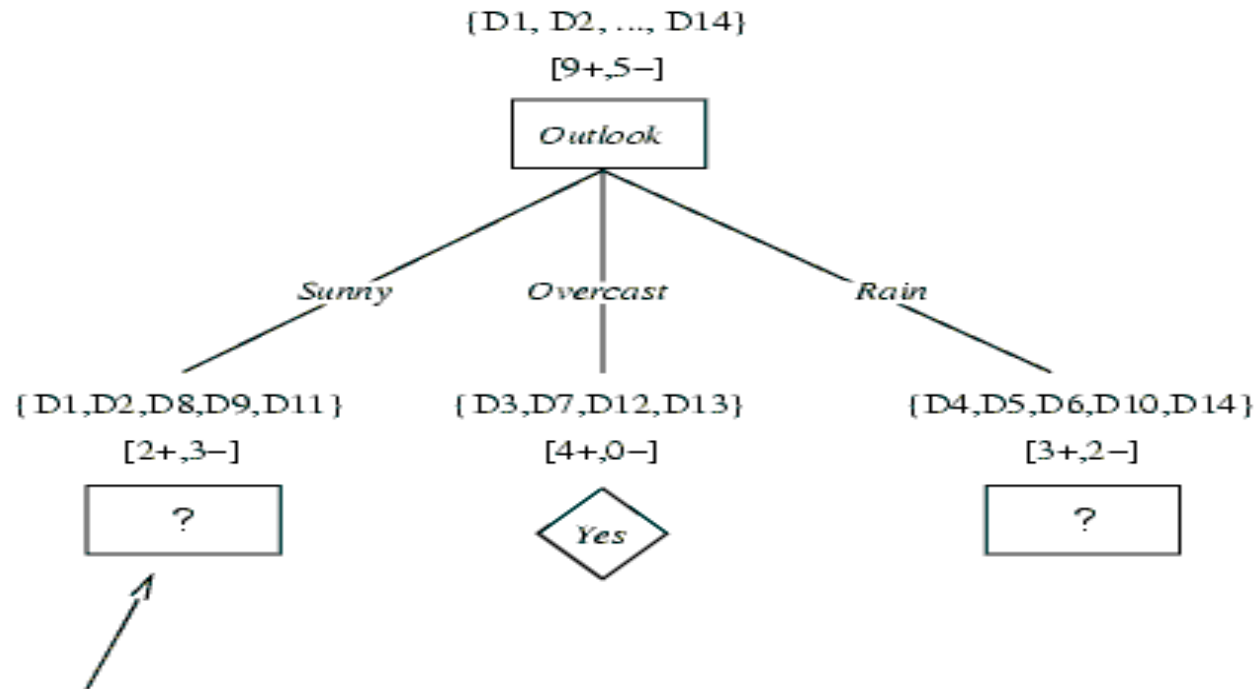


$$\begin{aligned} \text{Gain}(S, \text{Humidity}) &= .940 - (7/14) \cdot 0.985 - (7/14) \cdot 0.592 \\ &= .151 \end{aligned}$$



$$\begin{aligned} \text{Gain}(S, \text{Wind}) &= .940 - (8/14) \cdot 0.811 - (6/14) \cdot 1.0 \\ &= .048 \end{aligned}$$

# Choosing the Next Attribute - 2



Which attribute should be tested here?

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

$$\text{Gain}(S_{\text{sunny}}, \text{Humidity}) = .970 - (3/5)0.0 - (2/5)0.0 = .970$$

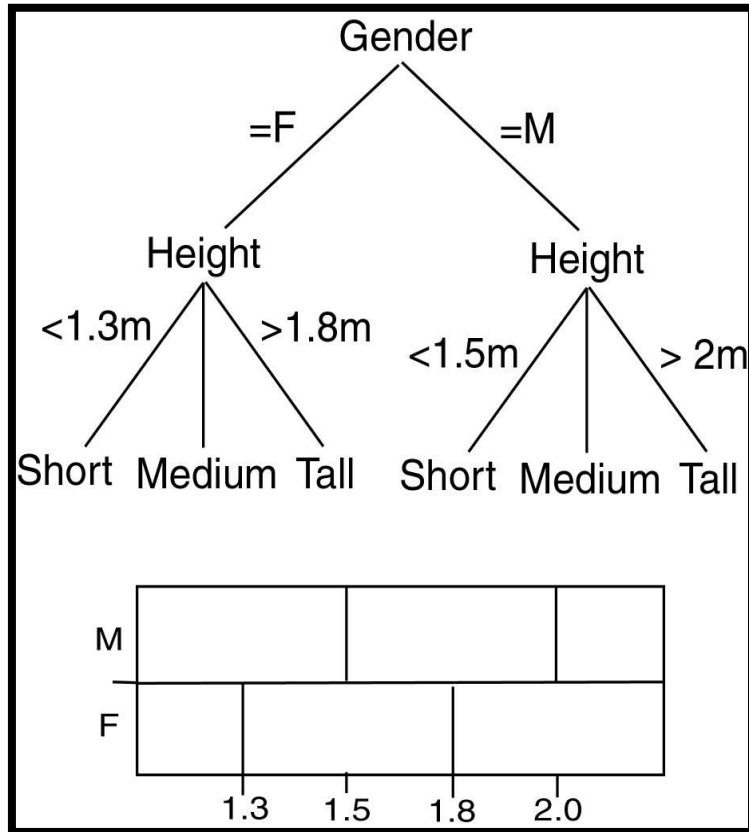
$$\text{Gain}(S_{\text{sunny}}, \text{Temperature}) = .970 - (2/5)0.0 - (2/5)1.0 - (1/5)0.0 = .570$$

$$\text{Gain}(S_{\text{sunny}}, \text{Wind}) = .970 - (2/5)1.0 - (3/5).918 = .019$$

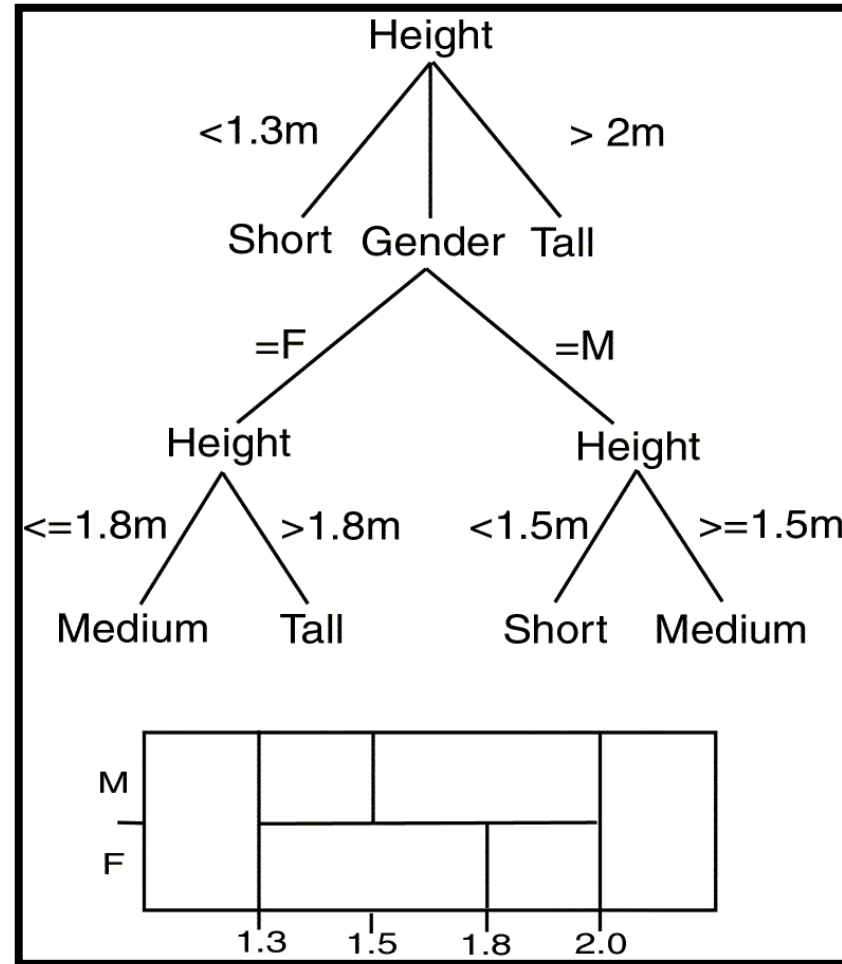
# Occam's Razor

- 14<sup>th</sup> Century Franciscan friar; William of Occam.
- The principle states that "Entities should not be multiplied unnecessarily."
- People often reinvented Occam's Razor
  - Newton - "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."
- To most scientist the razor is:
  - "when you have two competing theories which make exactly the same predictions, the one that is simpler is the better."

# Comparing DTs



Balanced



Deep

# DT Issues

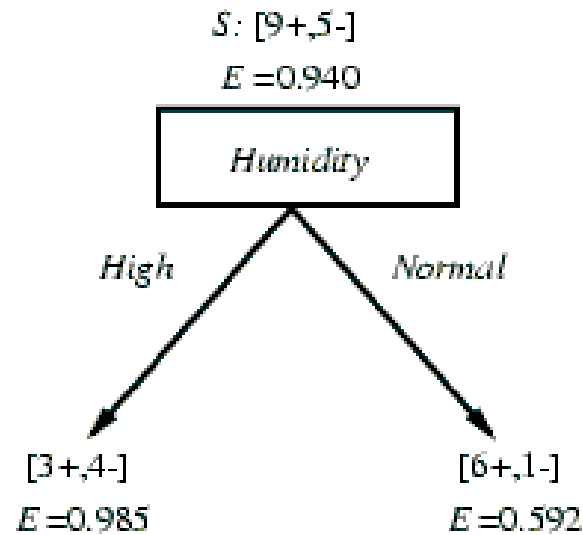
- Choosing Splitting Attributes
- Ordering of Splitting Attributes
- Splits
- Tree Structure
- Stopping Criteria
- Pruning

# Data Set

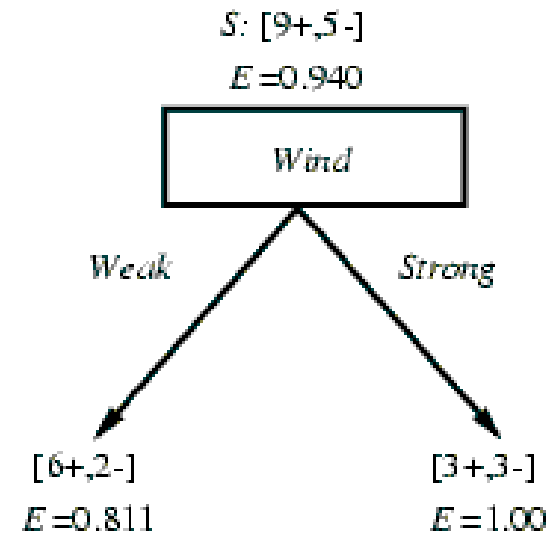
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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

# Choosing the Next Attribute - 1

Which attribute is the best classifier?



$$\begin{aligned} \text{Gain}(S, \text{Humidity}) &= .940 - (7/14) \cdot 0.985 - (7/14) \cdot 0.592 \\ &= .151 \end{aligned}$$



$$\begin{aligned} \text{Gain}(S, \text{Wind}) &= .940 - (8/14) \cdot 0.811 - (6/14) \cdot 1.0 \\ &= .048 \end{aligned}$$

# Confidences

```
outlook = sunny
|  humidity = high: no (3.0)
|  humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
|  windy = TRUE: no (2.0)
|  windy = FALSE: yes (3.0)
```

Number of Leaves : 5

Size of the tree : 8

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

How many possible combos

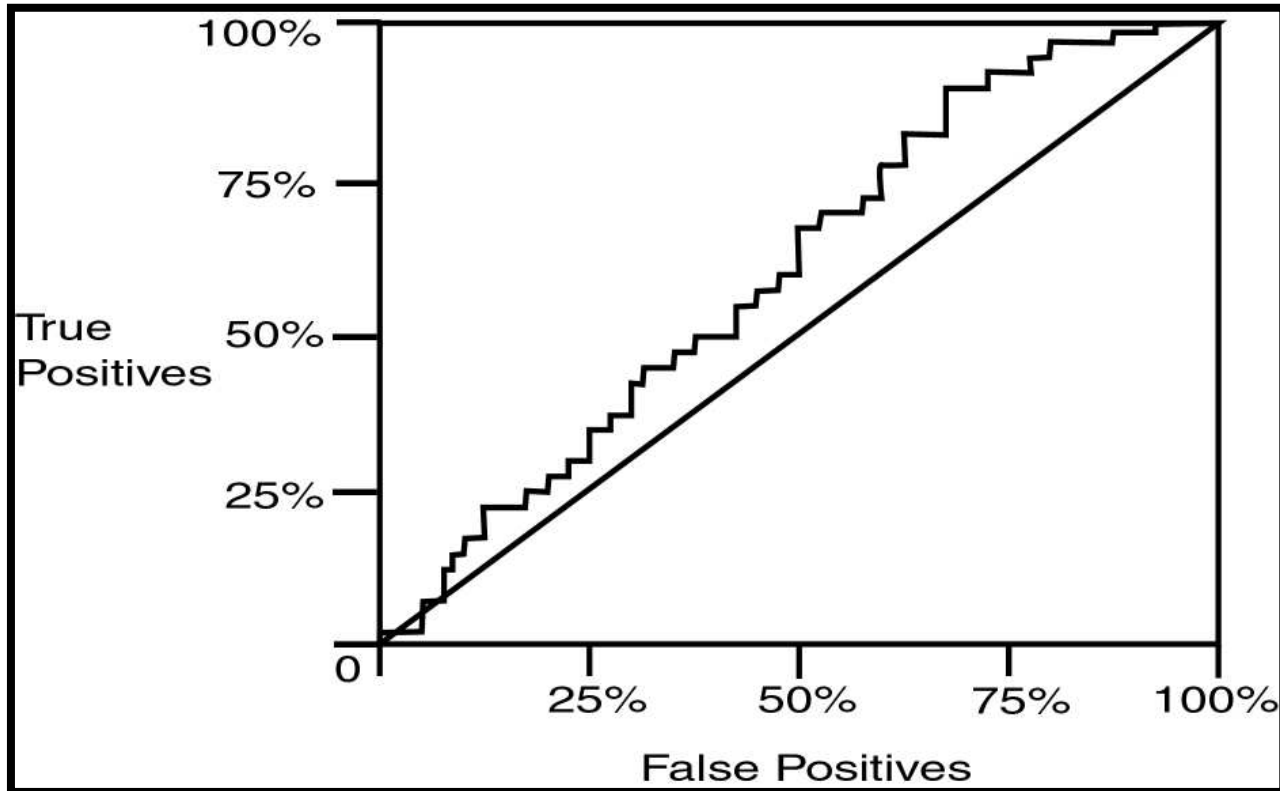
Test on ...

D15, Sunny, Mild, Low, Weak, ?

D16, Sunny, Mild, Low, Strong, ?

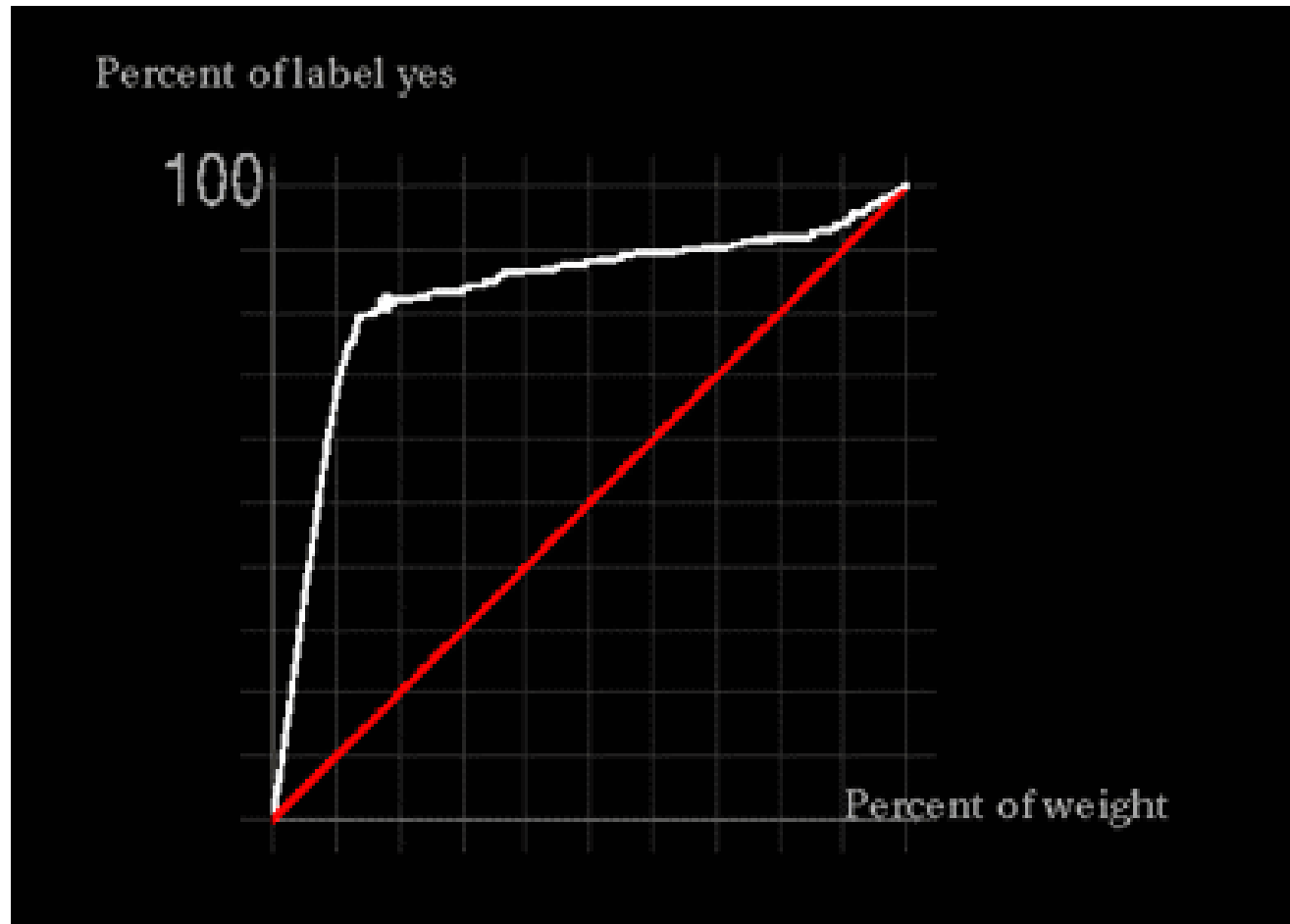
D17, Overcast, Hot, Normal, Strong, ?

# Operating Characteristic Curve



Lift, gains and response curves,

# Lift Curve



# Stopping Criteria

- What type of tree will perfectly classify the training data (ie. 100% training set accuracy)?
- Is this a bad thing?, Why?
- What does this tell you about the relationship between the dependent and independent attributes?
- Apriori stopping criteria, when:
  - A certain tree depth is reached
  - Number of records at a node goes below some threshold.
  - All potential splits are insignificant
- Want the tree to be simple, but not too simple

# How Do We Know When We've Overfitted The Training Data?

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

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

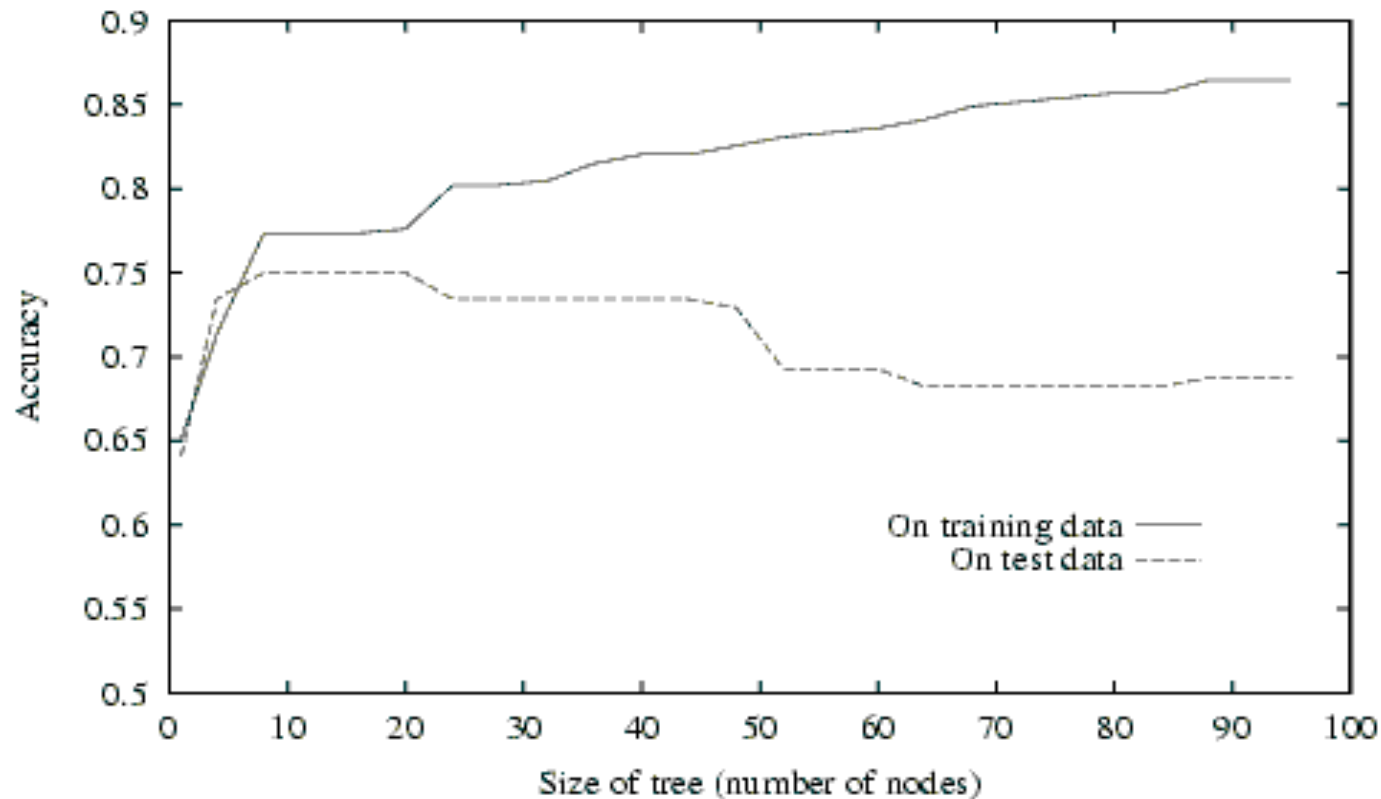
and

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

**Is there any other way?**

# Training Set Error Should Approximately Equal Test Set Error

Optimum model complexity?



# Trimming/Pruning Trees

- Stopping criterion can be some what arbitrary.
- Automatic pruning of trees
  - Ask the data, “How far should we split the data”.
  - Two general approaches:
    - Use part of the training set as a validation set
    - Use entire training set (usually an MDL approach).

# Using Pruning To Prevent Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

How to select “best” tree:

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

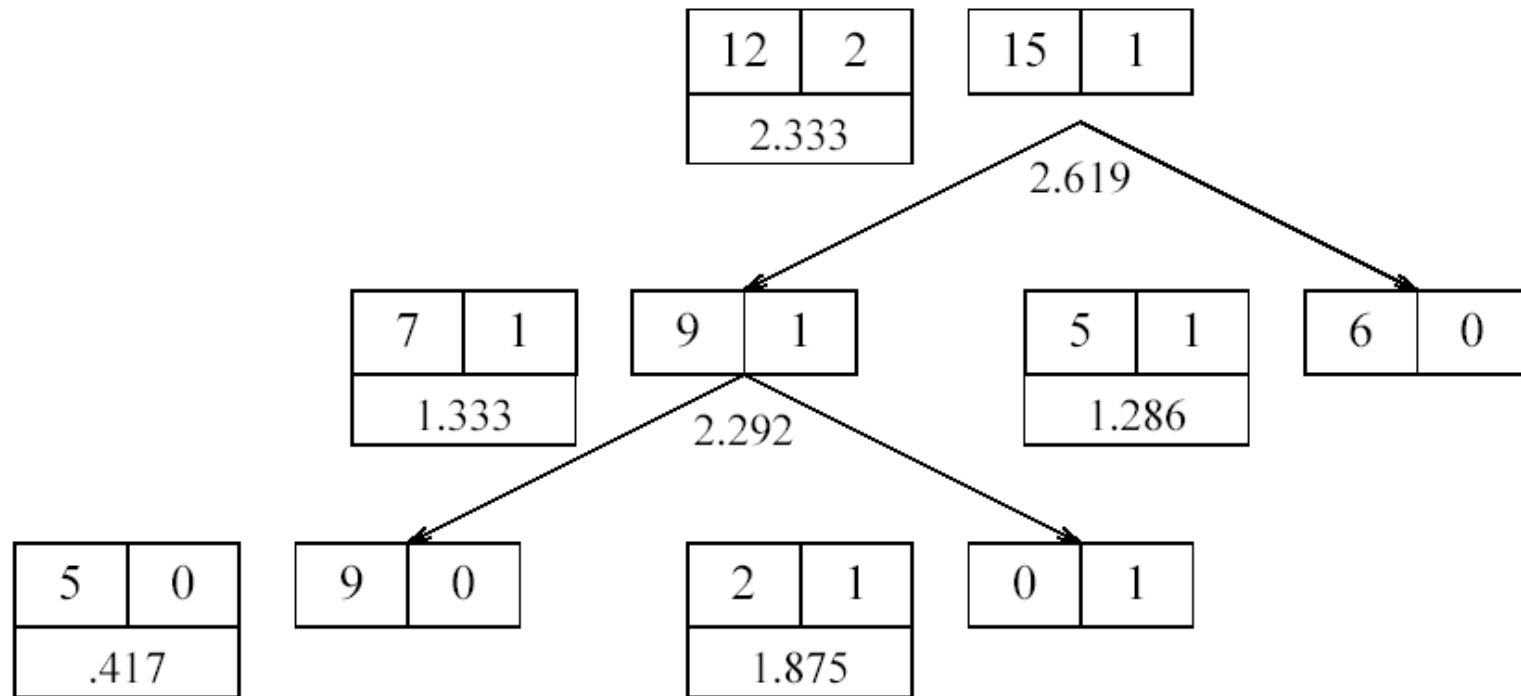
# Reduced Error Pruning

Split data into *training* and *validation* set

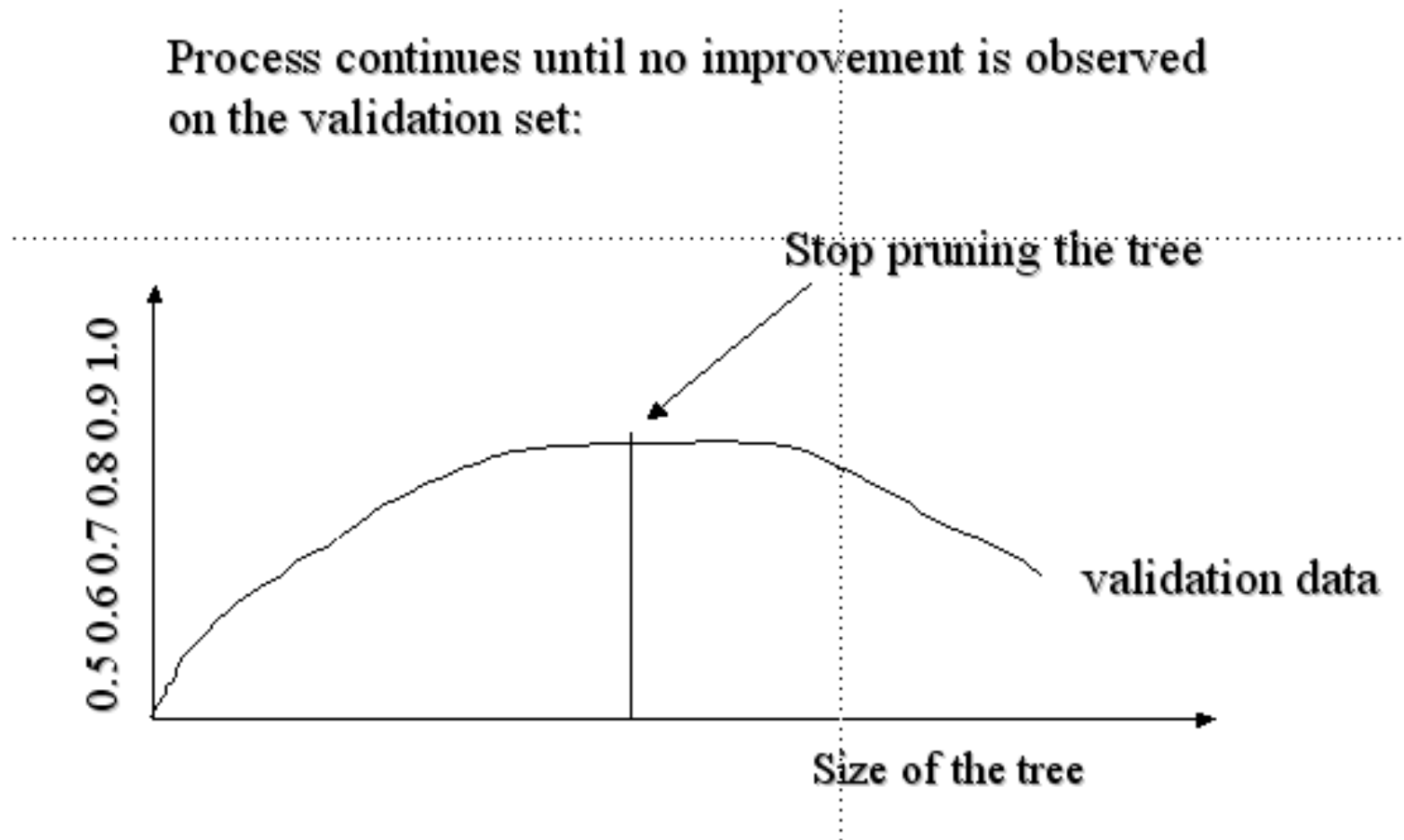
Do until further pruning is harmful:

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?

# Reduced Error Pruning



# Results of Reduced Error Pruning

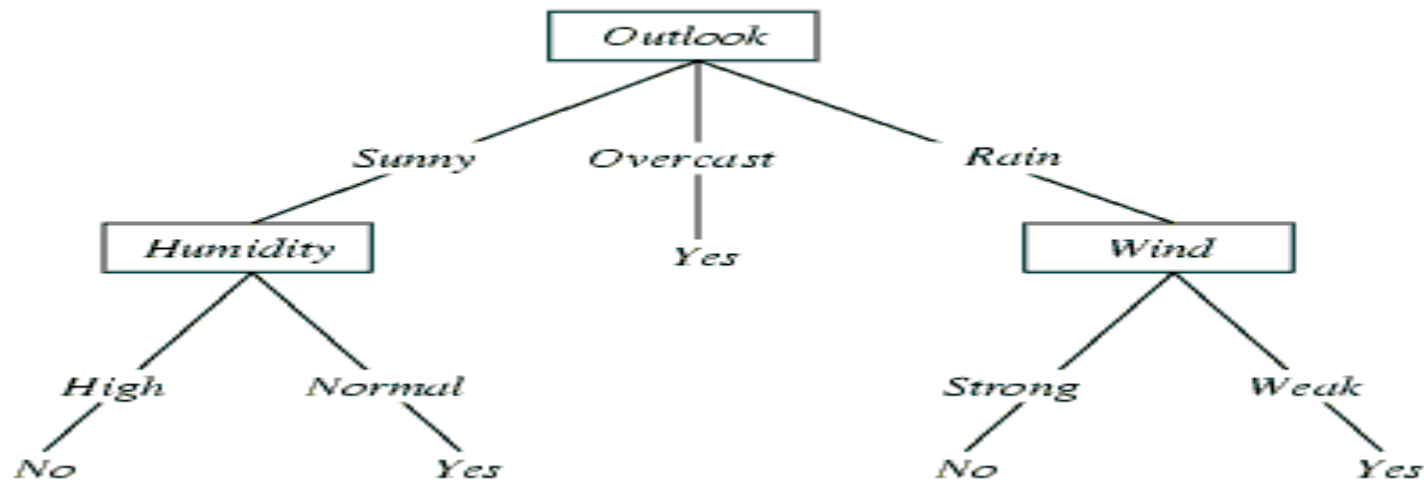


Consider the use of learning a tree is to make prediction  
What is the fundamental assumption that this learning algorithm is making

# Rule Post-Pruning

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

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



# Data Set

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