Machine Learning, Decision Trees, Overfitting

Reading: Mitchell, Chapter 3

Bishop Section 1.6

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

September 12, 2006

Machine Learning 10-701/15-781

Instructors
- Tom Mitchell
- Eric Xing

TA's
- Fan Guo
- Yifen Huang
- Indra Rustandi

Course assistant
- Sharon Cavlovich

See webpage for
- Office hours
- Grading policy
- Final exam date
- Late homework policy
- Syllabus details
- ...

www.cs.cmu.edu/~epxing/Class/10701/

Machine Learning:

Study of algorithms that

- improve their performance P
- at some task T
- with experience E

well-defined learning task: <P,T,E>

Learning to Predict Emergency C-Sections

[Sims et al., 2000]

Data:

Patient103 time=1 → *Patient103* time=2 ... → *Patient103* time=n

Patient103 time=1	*Patient103* time=2	*Patient103* time=n
Age: 23	Age: 23	Age: 23
FirstPregnancy: no	FirstPregnancy: no	FirstPregnancy: no
Anemia: no	Anemia: no	Anemia: no
Diabetes: no	Diabetes: YES	Diabetes: no
PreviousPrematureBirth: no	PreviousPrematureBirth: no	PreviousPrematureBirth: no
Ultrasound: ?	Ultrasound: abnormal	Ultrasound: ?
Elective C–Section: ?	Elective C–Section: no	Elective C–Section: no
Emergency C–Section: ?	Emergency C–Section: ?	**Emergency C–Section: Yes**
...	...	...

9714 patient records, each with 215 features

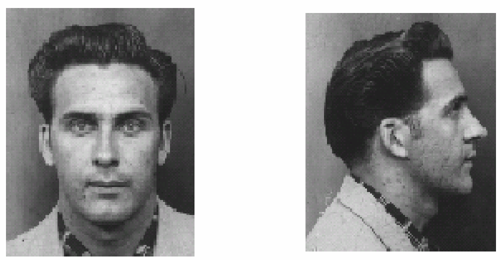
One of 18 learned rules:

```
If   No previous vaginal delivery, and
     Abnormal 2nd Trimester Ultrasound, and
     Malpresentation at admission
Then Probability of Emergency C-Section is 0.6

 Over training data: 26/41 = .63,
 Over test data: 12/20 = .60
```

Learning to detect objects in images

(Prof. H. Schneiderman)



Example training images
for each orientation

Learning to classify text documents

→ Company home page

vs

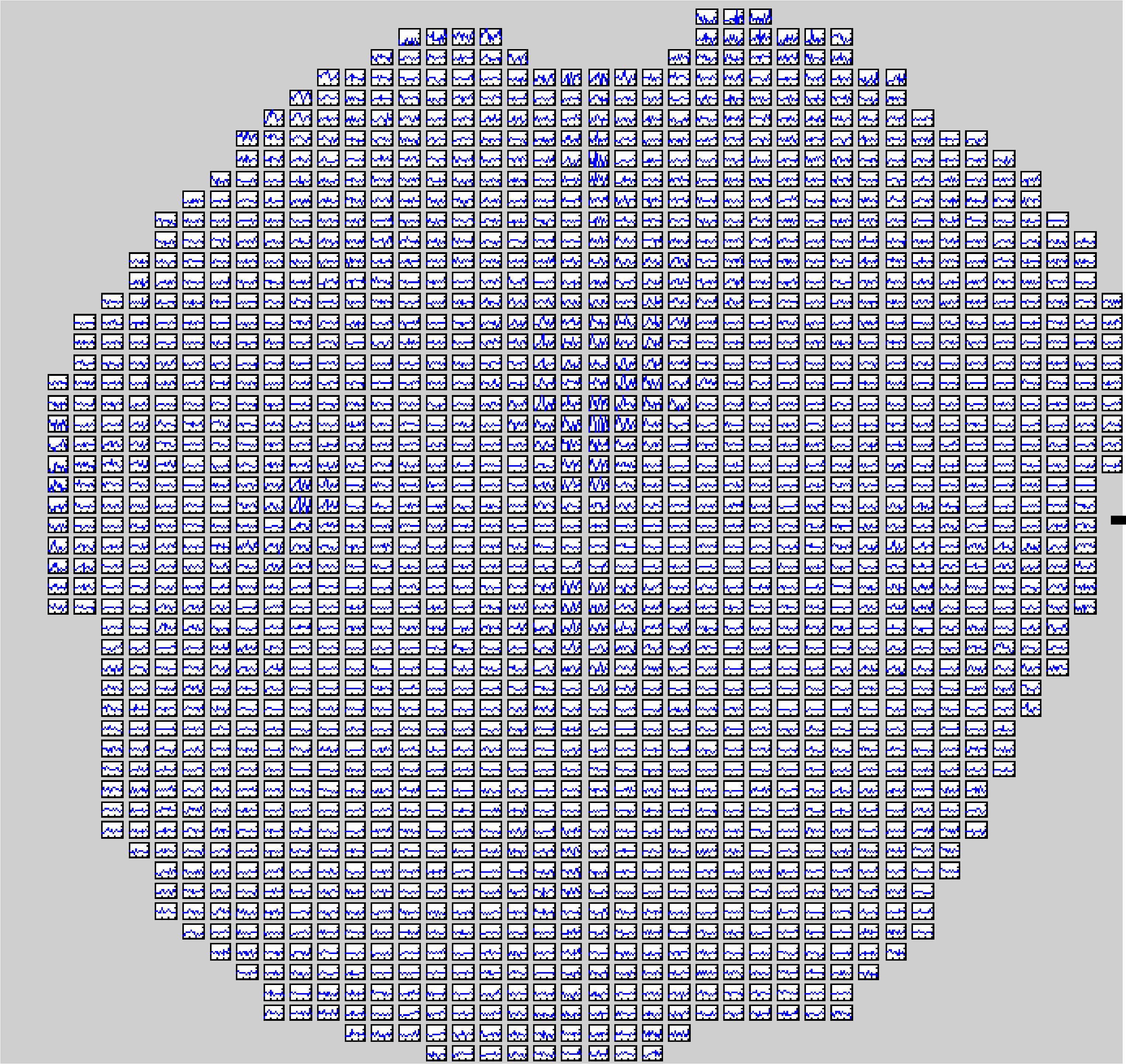
Personal home page

vs

University home page

vs

…

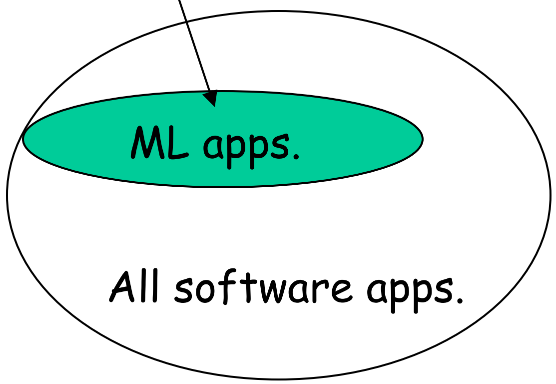


Reading
a noun
(vs verb)

[Rustandi et al.,
2005]

Growth of Machine Learning

- Machine learning is preferred approach to
  - Speech recognition, Natural language processing
  - Computer vision
  - Medical outcomes analysis
  - Robot control
  - …
- This ML niche is growing
  - Improved machine learning algorithms
  - Increased data capture, networking
  - Software too complex to write by hand
  - New sensors / IO devices
  - Demand for self-customization to user, environment



Function Approximation and Decision tree learning

Function approximation

Setting:

- Set of possible instances X
- Unknown target function f: $X \rightarrow Y$
- Set of function hypotheses $H=\{ h \mid h: X \rightarrow Y \}$

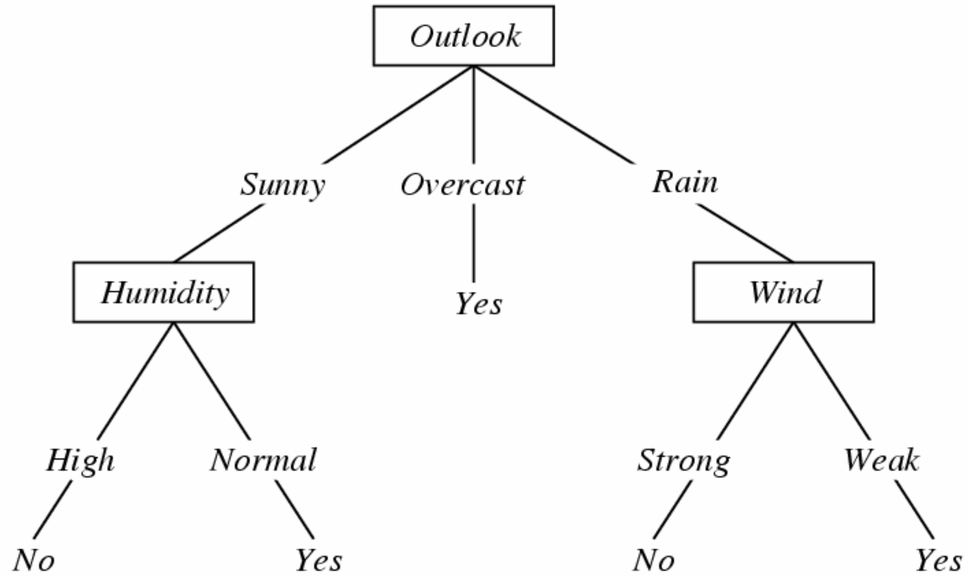
Given:

- Training examples $\{<x_i,y_i>\}$ of unknown target function f

Determine:

- Hypothesis $h \in H$ that best approximates f

Decision Tree for *PlayTennis*



How would you represent

$AB \vee CD(\neg E)$?

Each internal node: test one attribute X_i

Each branch from a node: selects one value for X_i

Each leaf node: predict Y (or $P(Y|X \in \text{leaf})$)

A Tree to Predict C-Section Risk

Learned from medical records of 1000 women

Negative examples are C-sections

```
[833+,167-] .83+ .17-
Fetal_Presentation = 1: [822+,116-] .88+ .12-
| Previous_Csection = 0: [767+,81-] .90+ .10-
| | Primiparous = 0: [399+,13-] .97+ .03-
| | Primiparous = 1: [368+,68-] .84+ .16-
| | | Fetal_Distress = 0: [334+,47-] .88+ .12-
| | | | Birth_Weight < 3349: [201+,10.6-] .95+ .
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+
| | | Fetal_Distress = 1: [34+,21-] .62+ .38-
| Previous_Csection = 1: [55+,35-] .61+ .39-
Fetal_Presentation = 2: [3+,29-] .11+ .89-
Fetal_Presentation = 3: [8+,22-] .27+ .73-
```

Top-Down Induction of Decision Trees

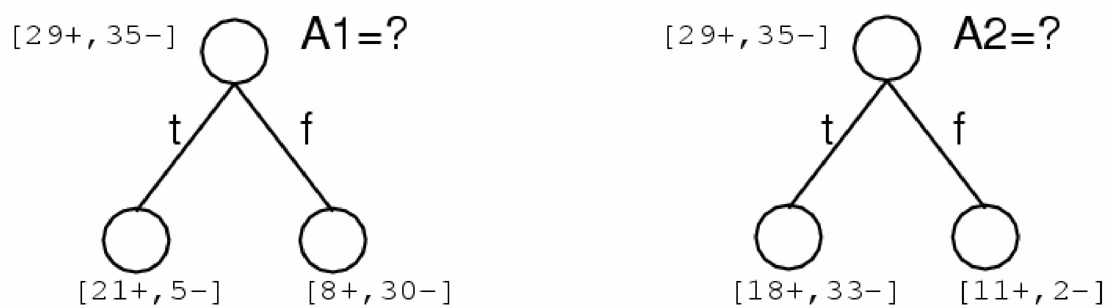
[ID3, C4.5, …]

node = Root

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?



Entropy

Entropy $H(X)$ of a random variable X

$$\boxed{H(X) = -\sum_{i=1}^{n} P(X=i)\log_2 P(X=i)}$$

$H(X)$ is the expected number of bits needed to encode a randomly drawn value of X (under most efficient code)

Why? Information theory:

- Most efficient code assigns $-\log_2 P(X=i)$ bits to encode the message $X=i$
- So, expected number of bits to code one random X is:

\# of possible values for X

$$\sum_{i=1}^{n} P(X=i)(-\log_2 P(X=i))$$

Entropy

Entropy *H(X)* of a random variable *X*

$$\boxed{H(X) = -\sum_{i=1}^{n} P(X=i)\log_2 P(X=i)}$$

Specific conditional entropy *H(X|Y=v)* of *X* given *Y=v* :

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log_2 P(X=i|Y=v)$$

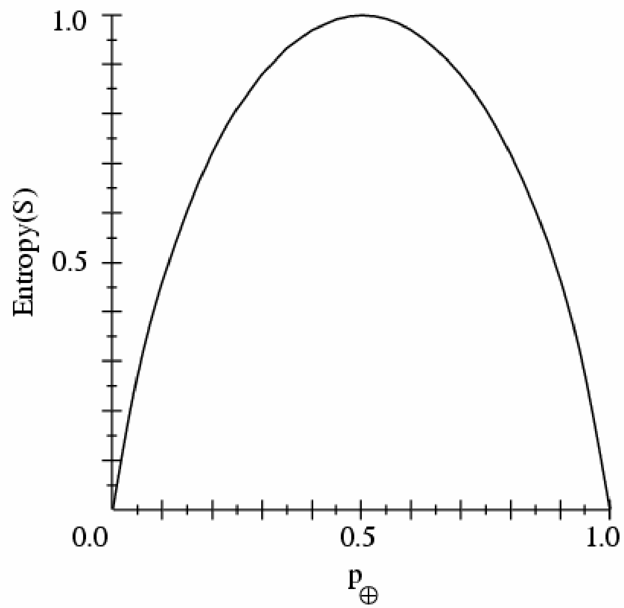
Conditional entropy *H(X|Y)* of *X* given *Y* :

$$H(X|Y) = \sum_{v \in values(Y)} P(Y=v)H(X|Y=v)$$

Mututal information (aka information gain) of *X* and *Y* :

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

Sample Entropy



- 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

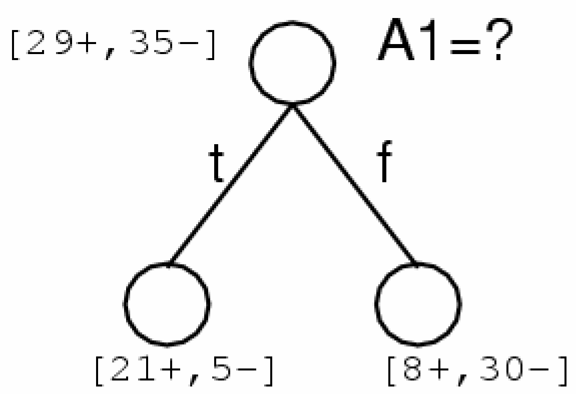
$$H(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

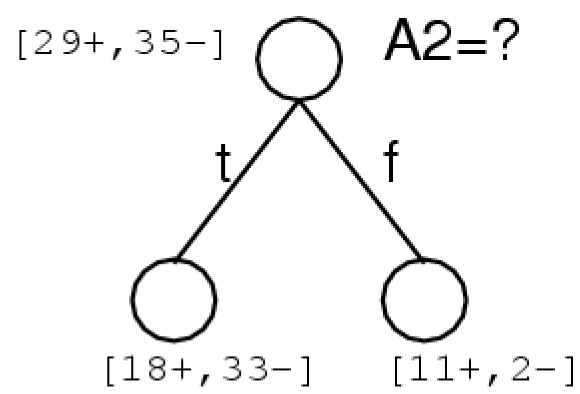
Information Gain

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

Subset of S for which $A=v$

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





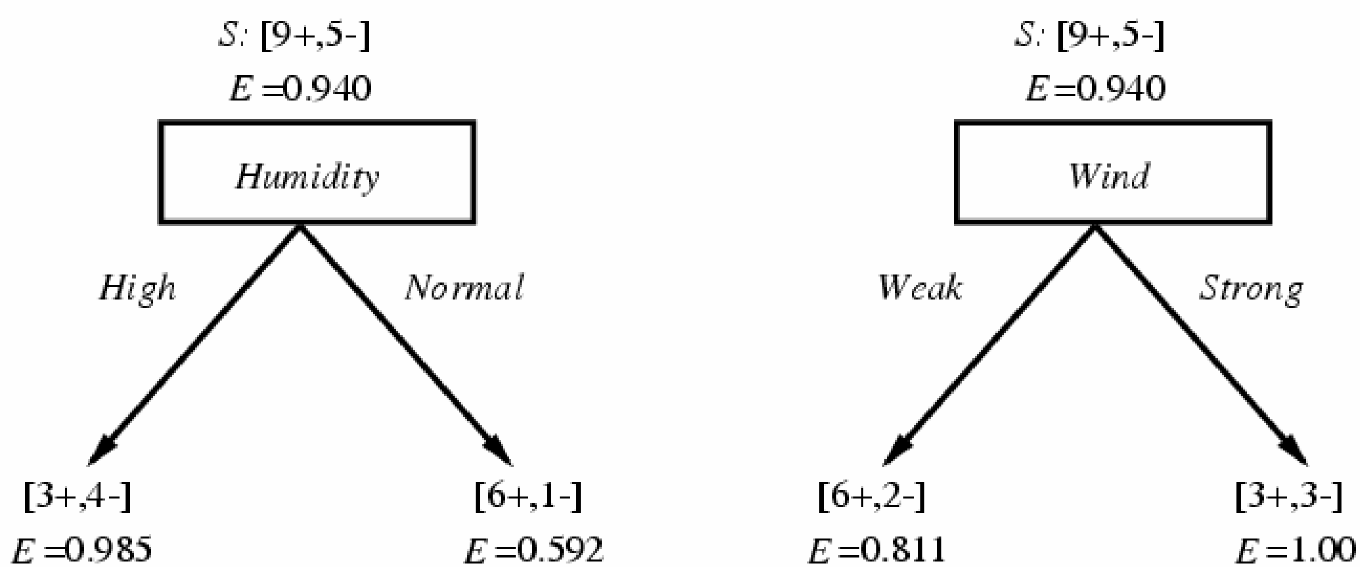
Gain(S,A) = mutual information between A and target class variable over sample S

Training Examples

Day	Outlook	Temperature	Humidity	Wind	PlayTenn
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?

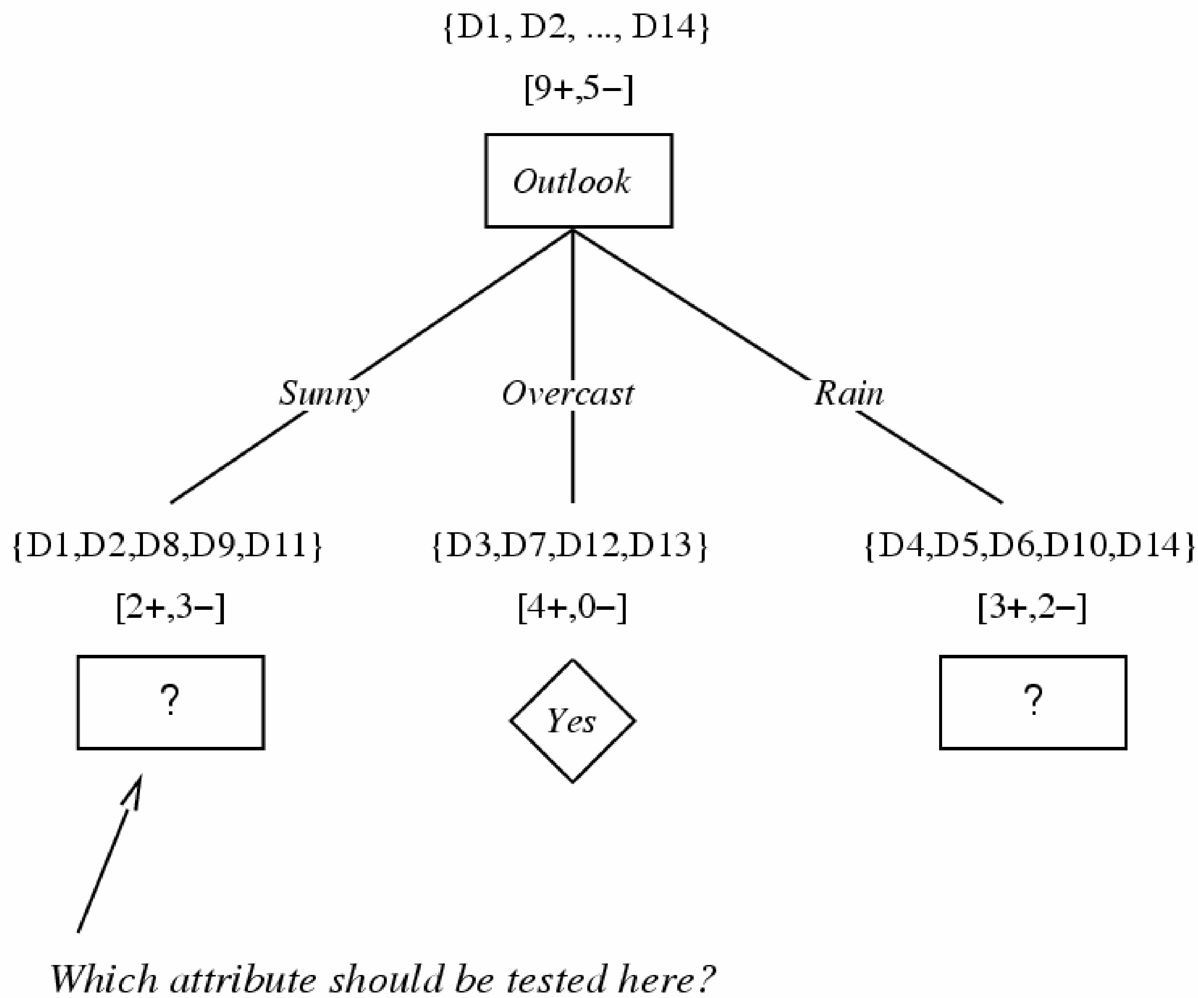


Gain (S, Humidity)

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

Gain (S, Wind)

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



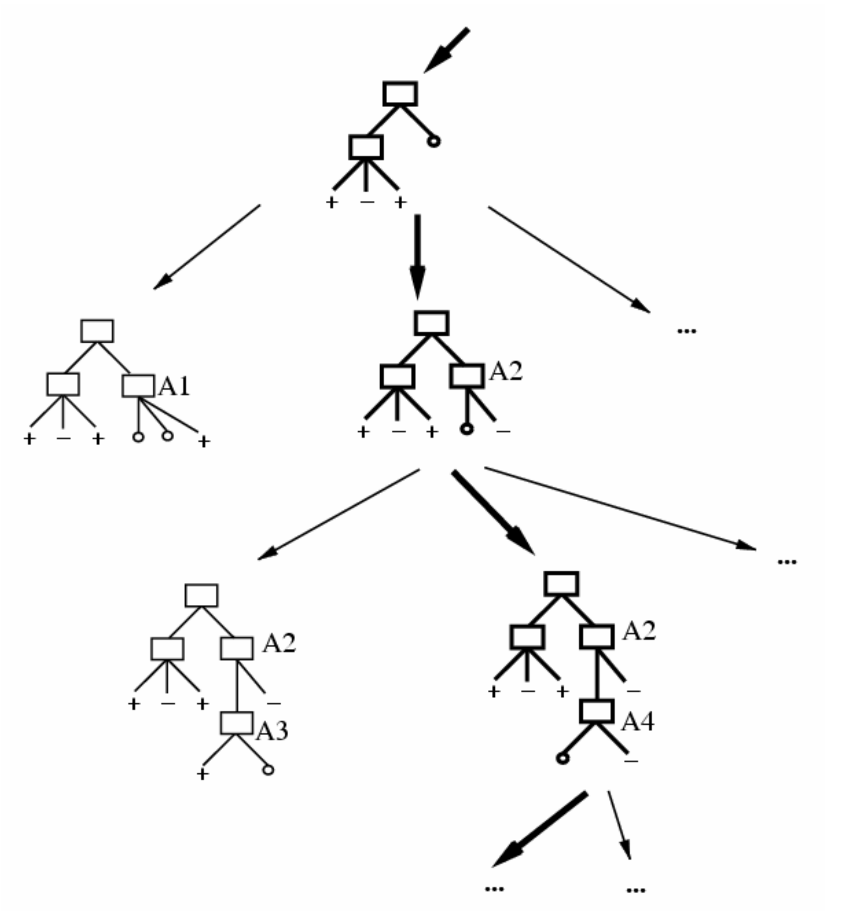
$S_{sunny} = \{D1,D2,D8,D9,D11\}$

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

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

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

Which Tree Should We Output?



- 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

Occam's Razor

Why prefer short hypotheses?

Argument in favor:

- Fewer short hyps. than long hyps.

$\rightarrow$ a short hyp that fits data unlikely to be coincidence

$\rightarrow$ a long hyp that fits data might be coincidence

Argument opposed:

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:

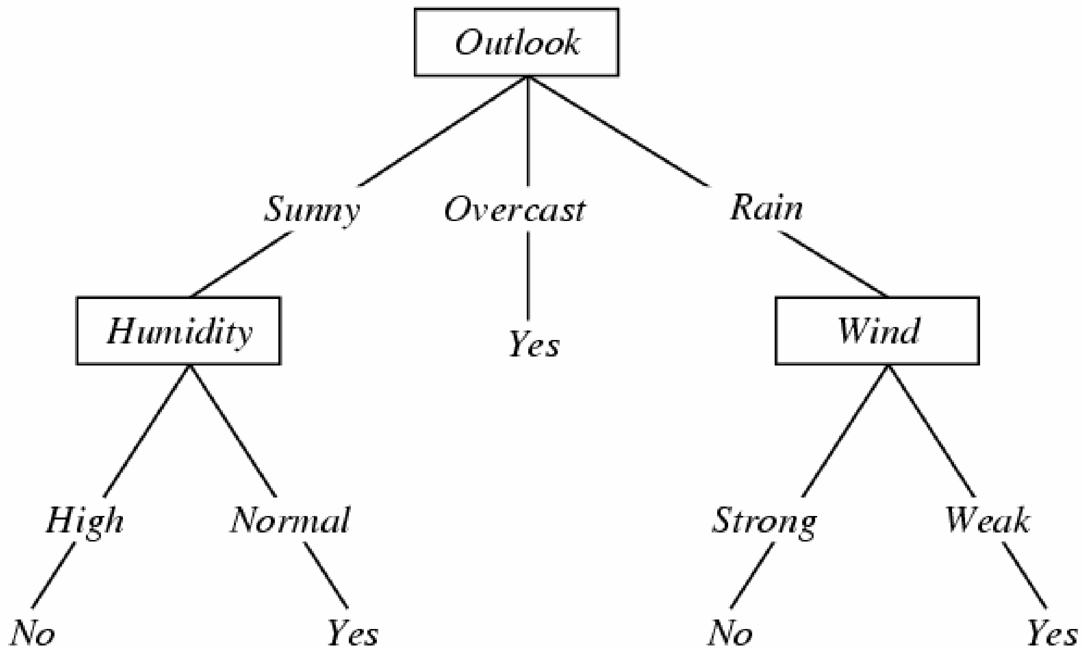
- There are many ways to define small sets of hyps
- e.g., all trees with a prime number of nodes that use attributes beginning with "Z"
- What's so special about small sets based on *size* of hypothesis??

Overfitting in Decision Trees

Consider adding noisy training example #15:

$Sunny,\ Hot,\ Normal,\ Strong,\ PlayTennis = No$

What effect on earlier tree?



Overfitting

Consider error of hypothesis h over

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

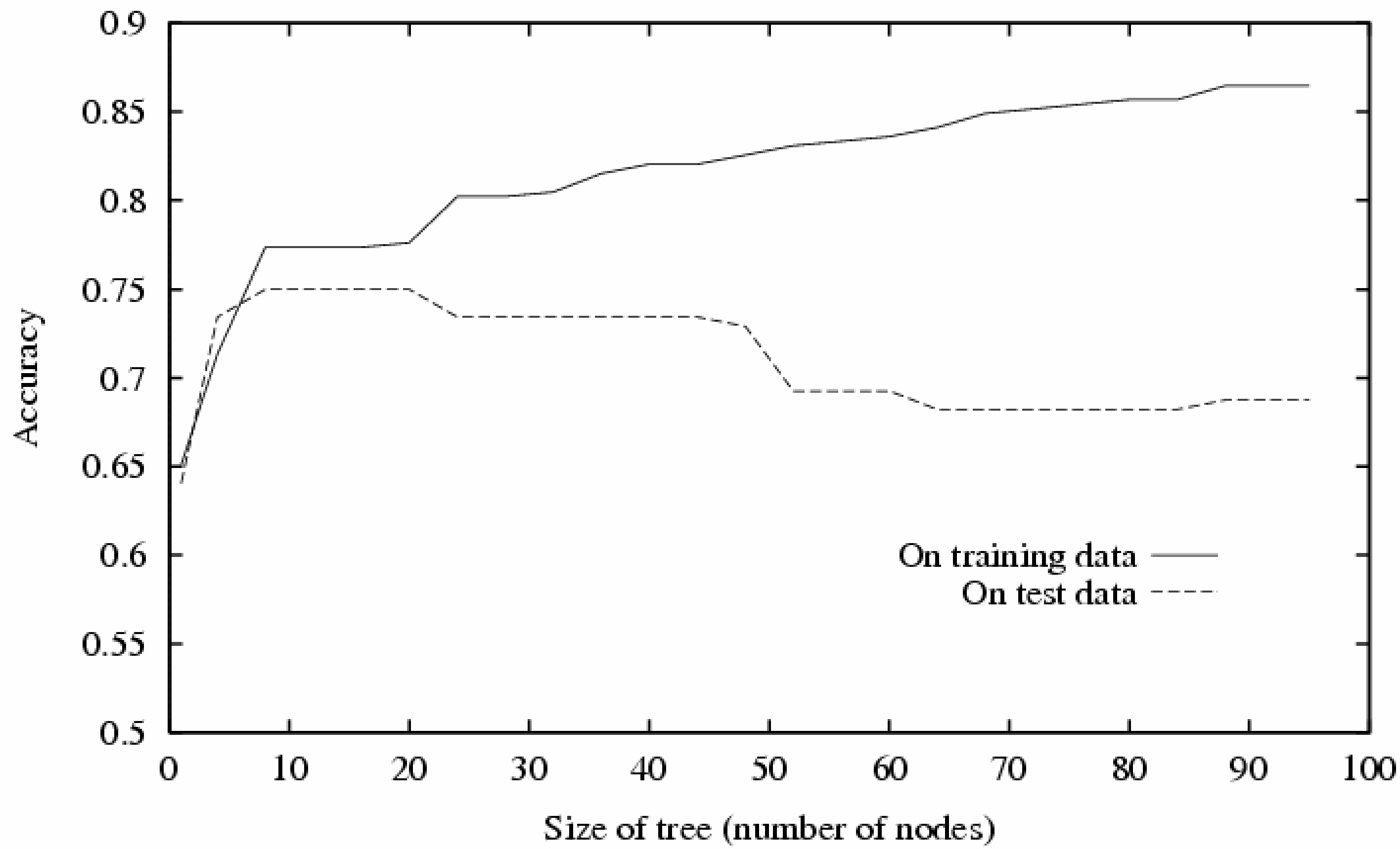
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

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

and

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

Overfitting in Decision Tree Learning



Avoiding 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

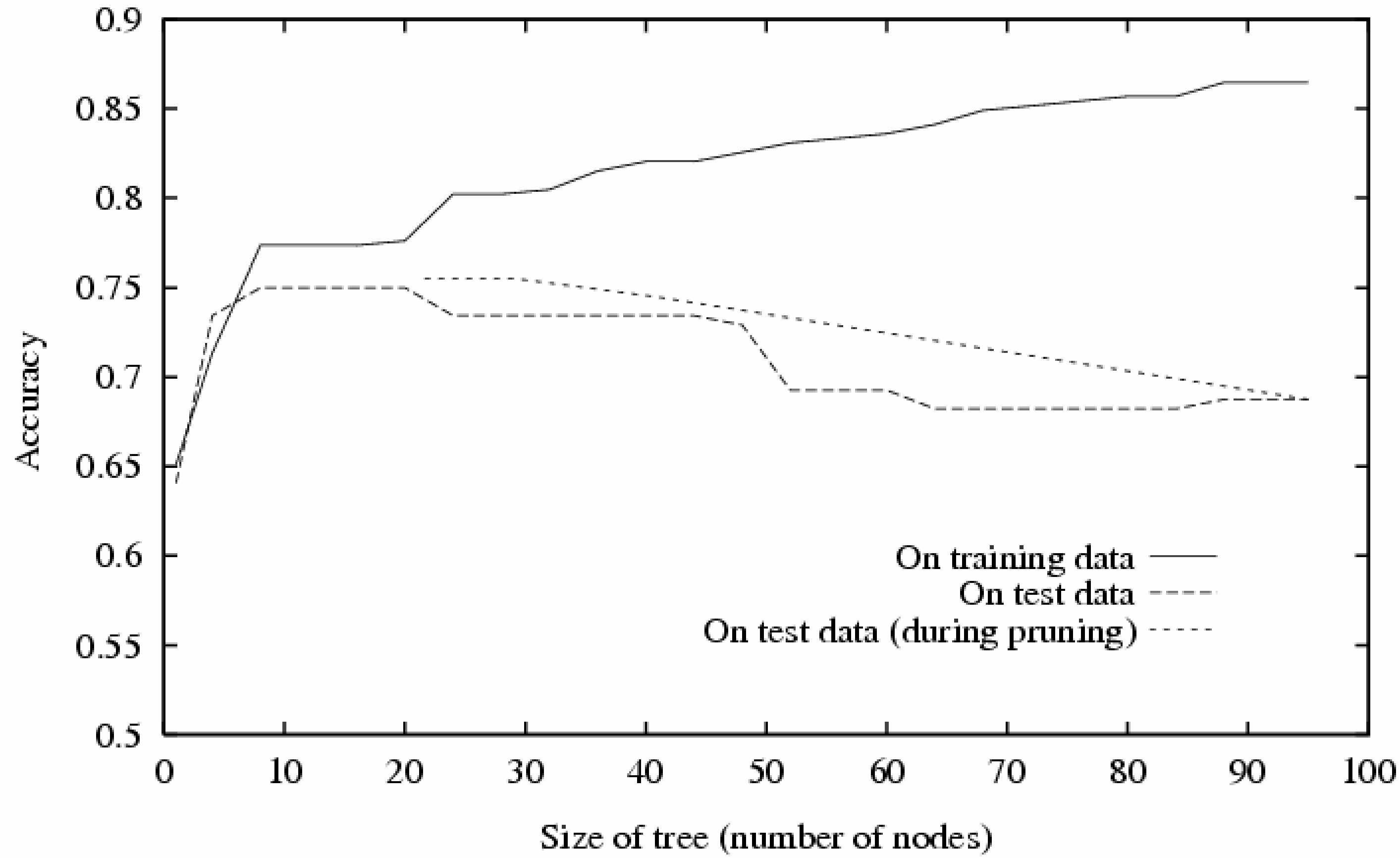
Create tree that classifies *training* set correctly

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?

Effect of Reduced-Error Pruning

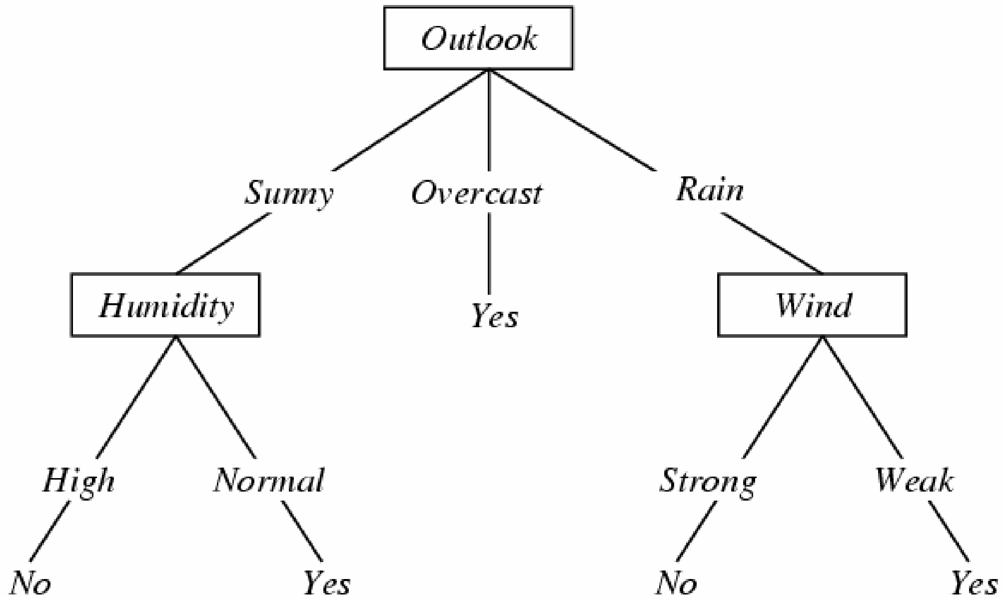


Rule Post-Pruning

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)

Converting A Tree to Rules



$$\begin{array}{ll} \text{IF} & (Outlook = Sunny) \wedge (Humidity = High) \\ \text{THEN} & PlayTennis = No \end{array}$$

$$\begin{array}{ll} \text{IF} & (Outlook = Sunny) \wedge (Humidity = Normal) \\ \text{THEN} & PlayTennis = Yes \end{array}$$

Continuous Valued Attributes

Create a discrete attribute to test continuous

- $Temperature = 82.5$
- $(Temperature > 72.3) = t, f$

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

Unknown Attribute Values

What if some examples missing values of A?
Use training example anyway, sort through tree

- If node n tests A, assign most common value of A among other examples sorted to node n
- assign most common value of A among other examples with same target value
- assign probability p_i to each possible value v_i of A
  - assign fraction p_i of example to each descendant in tree

Classify new examples in same fashion

What you should know:

- Well posed function approximation problems:
  - Instance space, X
  - Sample of labeled training data { $<x_i, y_i>$}
  - Hypothesis space, H = { f: X→Y }

- Learning is a search/optimization problem over H
  - Various objective functions
    - minimize training error (0-1 loss)
    - among hypotheses that minimize training error, select shortest

- Decision tree learning
  - Greedy top-down learning of decision trees (ID3, C4.5, ...)
  - Overfitting and tree/rule post-pruning
  - Extensions…

Questions to think about (1)

- Why use Information Gain to select attributes in decision trees? What other criteria seem reasonable, and what are the tradeoffs in making this choice?

Questions to think about (2)

- ID3 and C4.5 are heuristic algorithms that search through the space of decision trees. Why not just do an exhaustive search?

Questions to think about (3)

- Consider target function f: <x1,x2> → y, where x1 and x2 are real-valued, y is boolean. What is the set of decision surfaces describable with decision trees that use each attribute at most once?