## Outline

[read Chapter 2]
[suggested exercises 2.2, 2.3, 2.4, 2.6]

- Learning from examples
- General-to-specific ordering over hypotheses
- Version spaces and candidate elimination algorithm
- Picking new examples
- The need for inductive bias

Note: simple approach assuming no noise, illustrates key concepts

## Training Examples for *EnjoySport*

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |

What is the general concept?

## Representing Hypotheses

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |

Many possible representations

Here, $h$ is conjunction of constraints on attributes

Each constraint can be

- a specfic value (e.g., $Water = Warm$)
- don't care (e.g., "$Water = ?$")
- no value allowed (e.g.,"Water=$\emptyset$")

For example,

| Sky | AirTemp | Humid | Wind | Water | Forecst |
|---|---|---|---|---|---|
| $\langle$*Sunny* | ? | ? | *Strong* | ? | *Same*$\rangle$ |

Classify everything negative

$$\langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$$

Classify everything positive

$$\langle ?, ?, ?, ?, ?, ? \rangle$$

## Prototypical Concept Learning Task

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |

- **Given:**
  - Instances $X$: Possible days, each described by the attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
  - Target function $c$: $EnjoySport : X \rightarrow \{0, 1\}$
  - Hypotheses $H$: Conjunctions of literals. E.g.

    $$\langle ?, Cold, High, ?, ?, ? \rangle.$$

  - Training examples $D$: Positive and negative examples of the target function

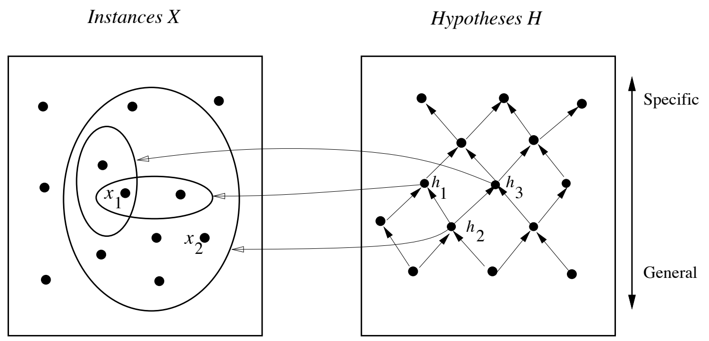
    $$\langle x_1, c(x_1) \rangle, \ldots \langle x_m, c(x_m) \rangle$$

- **Determine:** A hypothesis $h$ in $H$ such that $h(x) = c(x)$ for all $x$ in $D$.

**The inductive learning hypothesis:** Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples.

## Instance, Hypotheses, and More-General-Than

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |



$x_1$= <*Sunny, Warm, High, Strong, Cool, Same*>
$x_2$= <*Sunny, Warm, High, Light, Warm, Same*>

$h_1$= <*Sunny, ?, ?, Strong, ?, ?*>
$h_2$= <*Sunny, ?, ?, ?, ?, ?*>
$h_3$= <*Sunny, ?, ?, ?, Cool, ?*>

## FIND-S **Algorithm**

1. Initialize $h$ to the most specific hypothesis in $H$
2. For each positive training instance $x$
   - For each attribute constraint $a_i$ in $h$

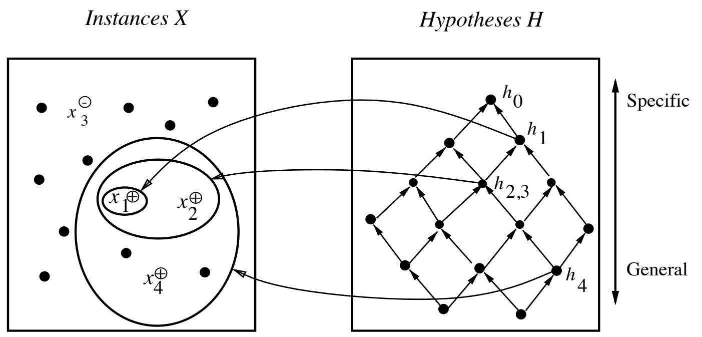
     If the constraint $a_i$ in $h$ is satisfied by $x$

     Then do nothing

     Else replace $a_i$ in $h$ by the next more general constraint that is satisfied by $x$
3. Output hypothesis $h$

## **Hypothesis Space Search by** FIND-S

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |



$h_0$ = <∅, ∅, ∅, ∅, ∅, ∅>

$x_1$ = *<Sunny Warm Normal Strong Warm Same>*, + $h_1$ = *<Sunny Warm Normal Strong Warm Same>*

$x_2$ = *<Sunny Warm High Strong Warm Same>*, + $h_2$ = *<Sunny Warm ? Strong Warm Same>*

$x_3$ = *<Rainy Cold High Strong Warm Change>*, - $h_3$ = *<Sunny Warm ? Strong Warm Same>*

$x_4$ = *<Sunny Warm High Strong Cool Change>*, + $h_4$ = *<Sunny Warm ? Strong ? ? >*

Generalisation only as far as necessary!

IF hypothesis is conjuction of attribute constraints

THEN guarantees most specific hypothesis in $H$ that is consistent with the positive examples

also consistent with the negative examples provided target concept $\in H$, training examples correct

## Complaints about FIND-S

- Can't tell whether it has learned concept
- Can't tell when training data inconsistent
- Picks a maximally specific $h$ (why?)
- Depending on $H$, there might be several!

## Version Spaces

A hypothesis $h$ is **consistent** with a set of training examples $D$ of target concept $c$ if and only if $h(x) = c(x)$ for each training example $\langle x, c(x)\rangle$ in $D$.

$$Consistent(h, D) \equiv (\forall \langle x, c(x)\rangle \in D)\ h(x) = c(x)$$

The **version space**, $VS_{H,D}$, with respect to hypothesis space $H$ and training examples $D$, is the subset of hypotheses from $H$ consistent with all training examples in $D$.

$$VS_{H,D} \equiv \{h \in H | Consistent(h, D)\}$$

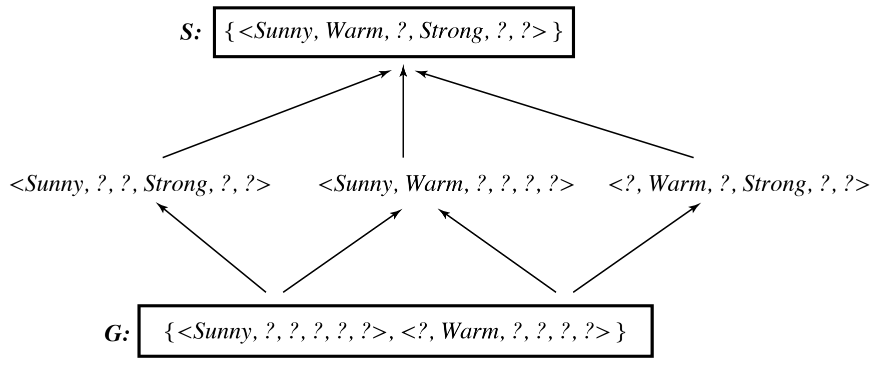
## The List-Then-Eliminate Algorithm:

1. $VersionSpace \leftarrow$ a list containing every hypothesis in $H$
2. For each training example, $\langle x, c(x) \rangle$

   remove from $VersionSpace$ any hypothesis $h$ for which $h(x) \neq c(x)$
3. Output the list of hypotheses in $VersionSpace$

## Example Version Space

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |



## Concept Learning As Search

| Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|
| Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| Sunny | Warm | High | Strong | Warm | Same | Yes |
| Rainy | Cold | High | Strong | Warm | Change | No |
| Sunny | Warm | High | Strong | Cool | Change | Yes |

- # instances
  - $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$
- # classifications
  - $2^{96}$
- # syntactically distinct hypotheses
  - 5120
- $\langle ., ., \phi, ., ., . \rangle$
  - represents no instance
- # semantically distinct hypotheses
  - 975

When large hypothesis space $H$ (possibly infinite) then efficient search strategies required to find hypothesis that <u>best</u> fits the data.

## Representing Version Spaces

The **General boundary**, G, of version space $VS_{H,D}$ is the set of its maximally general members

The **Specific boundary**, S, of version space $VS_{H,D}$ is the set of its maximally specific members

Every member of the version space lies between these boundaries

$$VS_{H,D} = \{h \in H | (\exists s \in S)(\exists g \in G)(g \geq h \geq s)\}$$

where $x \geq y$ means $x$ is more general or equal to $y$

## Candidate Elimination Algorithm

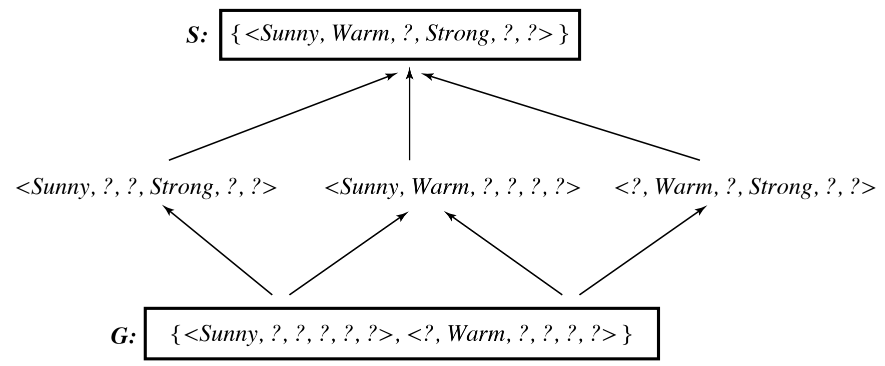
```
$G \leftarrow$ maximally general hypotheses in $H$, $\langle ?, ?, ?, ?, ?, ? \rangle$
$S \leftarrow$ maximally specific hypotheses in $H$, $\langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$
For each training example $d$, do
  • If $d$ is a positive example
      – Remove from $G$ any hypothesis inconsistent with $d$
      – For each hypothesis $s$ in $S$ that is not consistent with $d$
          ∗ Remove $s$ from $S$
          ∗ Add to $S$ all minimal generalizations $h$ of $s$ such that
            1. $h$ is consistent with $d$, and
            2. some member of $G$ is more general than $h$
          ∗ Remove from $S$ any hypothesis that is more general than another hypothesis in $S$
  • If $d$ is a negative example
      – Remove from $S$ any hypothesis inconsistent with $d$
      – For each hypothesis $g$ in $G$ that is not consistent with $d$
          ∗ Remove $g$ from $G$
          ∗ Add to $G$ all minimal specializations $h$ of $g$ such that
            1. $h$ is consistent with $d$, and
            2. some member of $S$ is more specific than $h$
          ∗ Remove from $G$ any hypothesis that is less general than another hypothesis in $G$
```

## Example Trace

$S_0$: {<Ø, Ø, Ø, Ø, Ø, Ø>}

$G_0$: {<?, ?, ?, ?, ?, ?>}

## What Next Training Example?



What query is best, most informative?

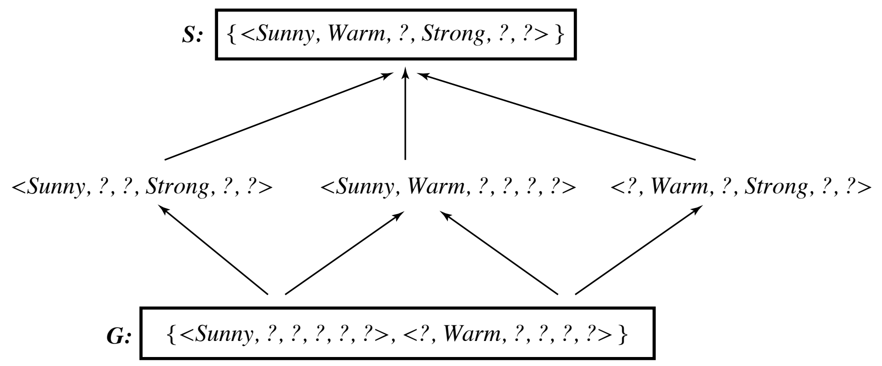
A query is satisfied by # / 2 of the hypotheses, and not satisfied by the other half

For example

$$\langle Sunny, Warm, Normal, Light, Warm, Same \rangle$$

Note: Answer comes from nature or teacher
# experiments: $\log_2 |VS|$

## How Should These Be Classified?



$$\langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change \rangle$$

$$\langle Rainy\ Cool\ Normal\ Light\ Warm\ Same \rangle$$

$$\langle Sunny\ Warm\ Normal\ Light\ Warm\ Same \rangle$$

$$\langle Sunny\ Cold\ Normal\ Strong\ Warm\ Same \rangle$$

## A Biased Hypothesis Space

| Example | Sky | Temp | Humid | Wind | Water | Forecst | EnjoySpt |
|---|---|---|---|---|---|---|---|
| 1 | Sunny | Warm | Normal | Strong | Cool | Change | Yes |
| 2 | Cloudy | Warm | Normal | Strong | Cool | Change | Yes |
| 3 | Rainy | Warm | Normal | Strong | Cool | Change | No |

$$S_2 : \langle ?\ Warm\ Normal\ Strong\ Cool\ Change \rangle$$

## What Justifies this Inductive Leap?

$$+\ \langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change \rangle$$

$$+\ \langle Sunny\ Warm\ Normal\ Light\ Warm\ Same \rangle$$

---

$$S:\ \langle Sunny\ Warm\ Normal\ ?\ ?\ ? \rangle$$

Why believe we can classify the unseen

$$\langle Sunny\ Warm\ Normal\ Strong\ Warm\ Same \rangle$$

## An UNBiased Learner

Idea: Choose $H$ that expresses every teachable concept (i.e., $H$ is the power set of $X$)

Consider $H' =$ disjunctions, conjunctions, negations over previous $H$. E.g.,

$$\langle \mathit{Sunny}\ \mathit{Warm}\ \mathit{Normal}\ ?\ ?\ ?\rangle \ \vee\ \neg\langle ?\ ?\ ?\ ?\ ?\ \mathit{Change}\rangle$$

What are $S$, $G$ in this case?

S $\leftarrow$

G $\leftarrow$

## Inductive Bias

Consider

- concept learning algorithm $L$
- instances $X$, target concept $c$
- training examples $D_c = \{\langle x, c(x)\rangle\}$
- let $L(x_i, D_c)$ denote the classification assigned to the instance $x_i$ by $L$ after training on data $D_c$.

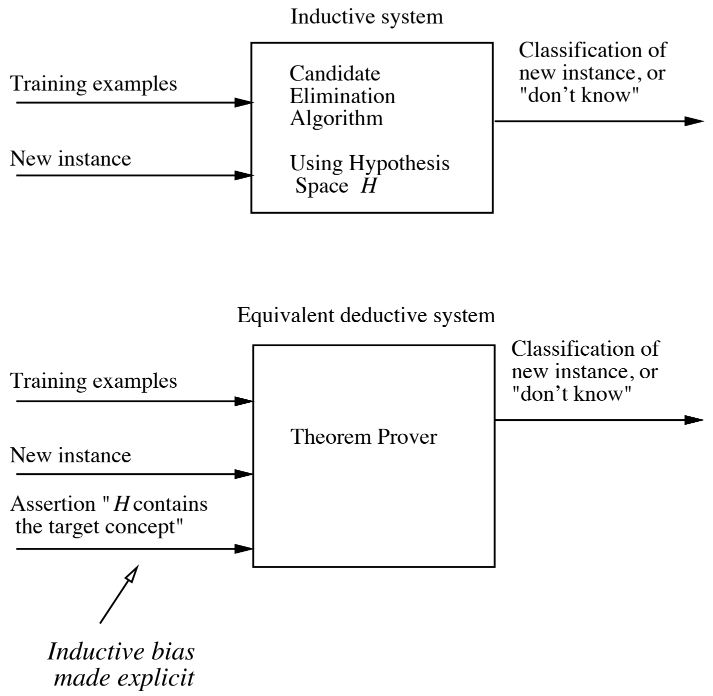
**Definition**:

The **inductive bias** of $L$ is any minimal set of assertions $B$ such that for any target concept $c$ and corresponding training examples $D_c$

$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$$

where $A \vdash B$ means $A$ logically entails $B$

## Inductive Systems and Equivalent Deductive Systems


Inductive system
Training examples
New instance
Candidate Elimination Algorithm
Using Hypothesis Space $H$
Classification of new instance, or "don't know"
Equivalent deductive system
Training examples
New instance
Assertion "$H$ contains the target concept"
Theorem Prover
Classification of new instance, or "don't know"
Inductive bias made explicit

## Three Learners with Different Biases

1. *Rote learner:* Store examples, Classify $x$ iff it matches previously observed example.
2. *Version space candidate elimination algorithm*
3. *Find-S*

## Summary Points

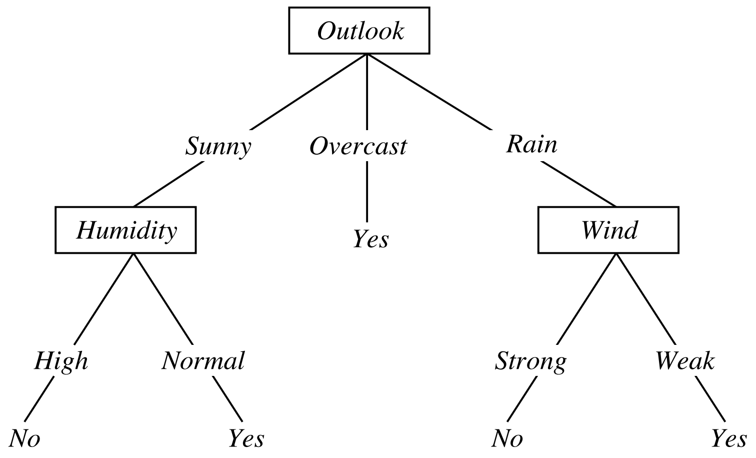
1. Concept learning as search through $H$
2. General-to-specific ordering over $H$
3. Version space candidate elimination algorithm
4. $S$ and $G$ boundaries characterize learner's uncertainty
5. Learner can generate useful queries
6. Inductive leaps possible only if learner is biased
7. Inductive learners can be modelled by equivalent deductive systems

## Decision Tree Learning

[read Chapter 3]
[recommended exercises 3.1, 3.4]

- Decision tree representation
- ID3 learning algorithm
- Entropy, Information gain
- Overfitting

## Decision Tree for *PlayTennis*



The instance

- Outlook = Sunny
- Temperature = Hot
- Humidity = High
- Wind = Strong

→ No

## 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+ .05-
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+ .22-
| | | 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-
```

## Decision Trees

Decision tree representation:

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

How would we represent:

- $\wedge, \vee$, XOR
- $(A \wedge B) \vee (C \wedge \neg D \wedge E)$
- ~~$M$ of $N$~~

## 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:

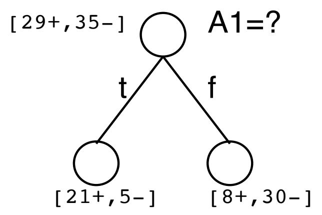
- Equipment or medical diagnosis
- Credit risk analysis
- Modeling calendar scheduling preferences

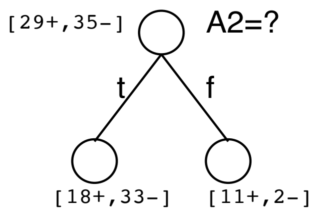
## Top-Down Induction of Decision Trees

Main loop:

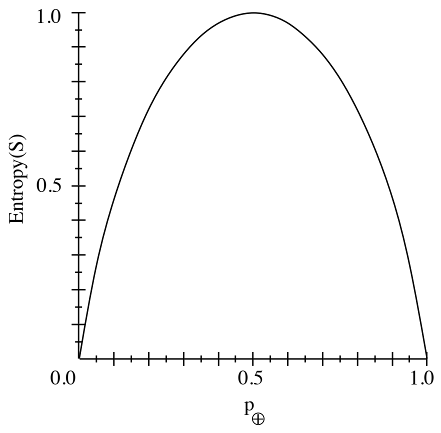
1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign $A$ as decision attribute for *node*
3. For each value of $A$, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?





## Entropy



- $S$ 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

*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 $S$:

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

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

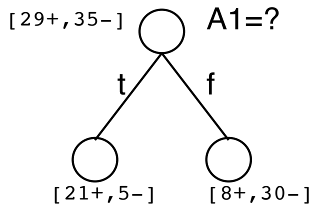
with c-wise classification, we get

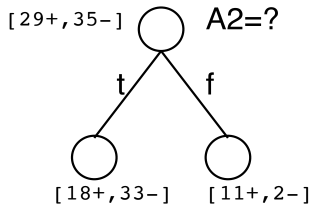
$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

## Information Gain

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

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





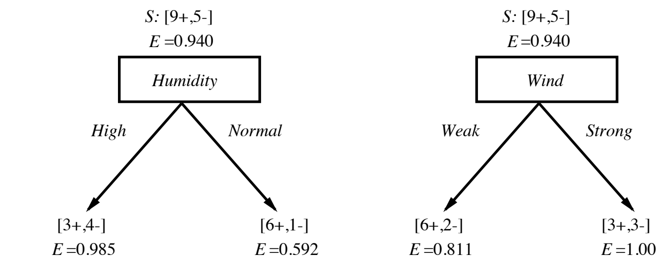
## Training Examples

| 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

| 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 |

**Which attribute is the best classifier?**

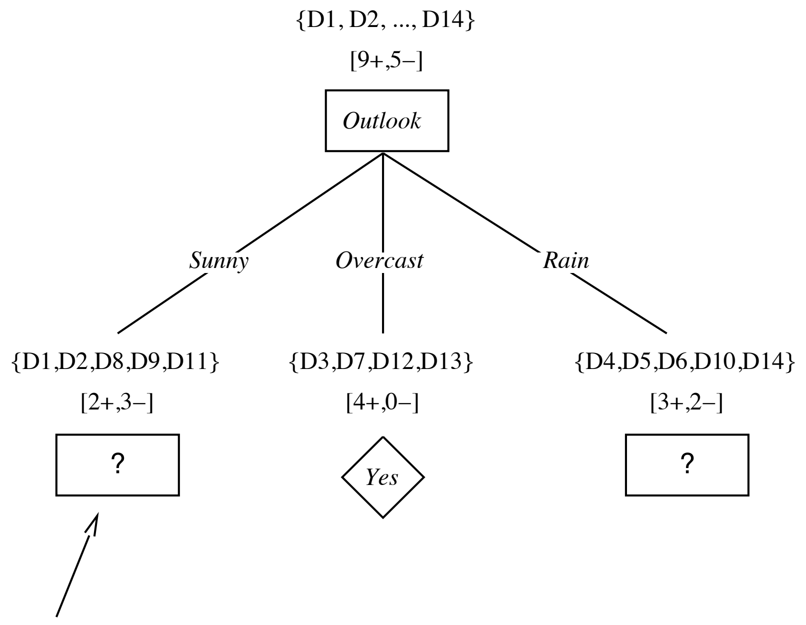


*Gain (S, Humidity )*
= .940 - (7/14).985 - (7/14).592
= .151

*Gain (S, Wind)*
= .940 - (8/14).811 - (6/14)1.0
= .048

$$Gain(s, Outlook) = 0.246$$

$$Gain(s, Temperature) = 0.029$$



*Which attribute should be tested here?*

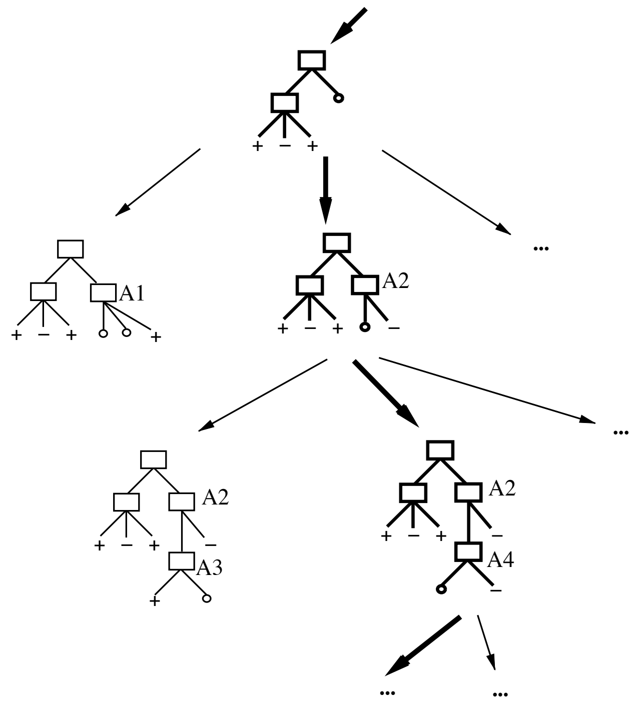
$S_{sunny}$ = {D1,D2,D8,D9,D11}

*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

## Hypothesis Space Search by ID3



ID3 algorithms perform a single to complex hill-climbing searching

The evaluation function = information gain

## 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

Note $H$ is the power set of instances $X$

$\rightarrow$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:

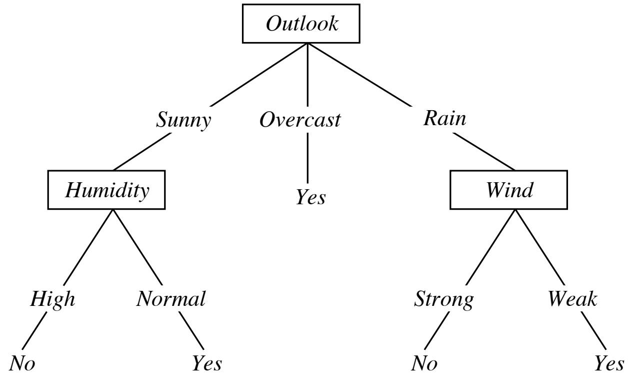
- There are many ways to define small sets of hyps
- e.g., all trees with a prime number of nodes that use attributes beginning with "Z"
- What's so special about small sets based on *size* of hypothesis??

## Overfitting in Decision Trees

Consider adding noisy training example #15:

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

What effect on earlier tree?



## Overfitting

Consider error of hypothesis $h$ over

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

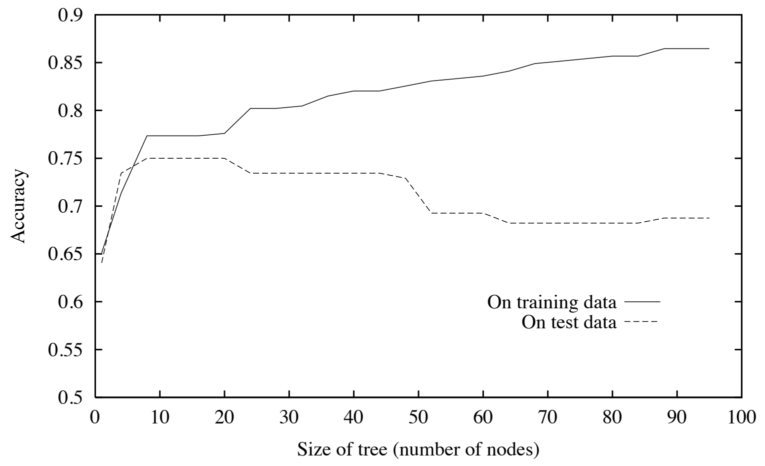
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

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

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

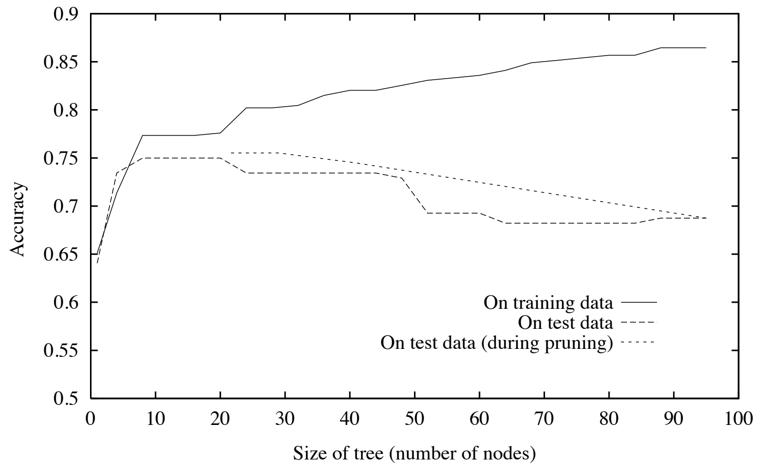
Split data into *training* and *validation* set

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

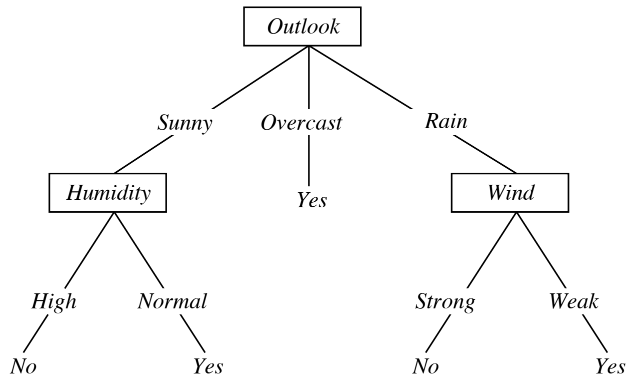


## Rule Post-Pruning

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



IF $(Outlook = Sunny) \wedge (Humidity = High)$
THEN $PlayTennis = No$

IF $(Outlook = Sunny) \wedge (Humidity = Normal)$
THEN $PlayTennis = Yes$

...

## 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

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

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

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

## Attributes with Costs

Consider

- medical diagnosis, $BloodTest$ has cost \$150
- robotics, $Width\_from\_1ft$ has cost 23 sec.

How to learn a consistent tree with low expected cost?
One approach: replace gain by

- Tan and Schlimmer (1990)

$$\frac{Gain^2(S, A)}{Cost(A)}.$$

- Nunez (1988)

$$\frac{2^{Gain(S,A)} - 1}{(Cost(A) + 1)^w}$$

where $w \in [0, 1]$ determines importance of cost

## 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