

# CONCEPT LEARNING AND THE GENERAL-TO-SPECIFIC ORDERING

[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

# The Concept Learning Problem

## Concept:

- Subset of objects defined over a set
- Boolean-valued function defined over a set
- Set of instances
- Syntactic definition

## Concept Learning

- Inferring a Boolean-valued function from training examples of its input and output

Automatic inference of the general definition of some concept, given examples labelled as members or non-members of the concept

- **Given:**

A set  $E = \{e_1, e_2, \dots, e_n\}$  of *training* instances of concepts, each labelled with the name of a concept  $C_1, C_2, \dots, C_k$  to which it belongs

**Determine:**

The definitions of each of  $C_1, C_2, \dots, C_k$  which correctly cover  $E$ . Each definition is a *concept description*

## Example: Learning Conjunctive Boolean Concepts

Instances space:  $\{0, 1\}^n$

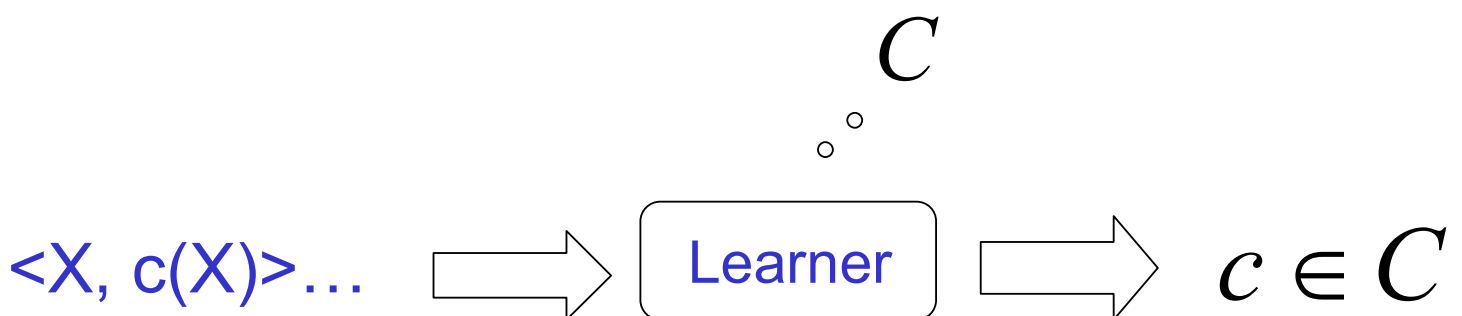
Concept is binary function  $c: \{0, 1\}^n \rightarrow \{0, 1\}$

Inputs:  $n$ -bit patterns

Outputs: 0 or 1

$\mathcal{C}$  = set of all  $c$  which have conjunctive representation

Learning task: Identify a conjunctive concept that is consistent with the examples



## Example: Learning Conjunctive Boolean Concepts

- Learning algorithm:

1. Initialize:  $L = \{x_1, \bar{x}_1, \dots, x_n, \bar{x}_n\}$

2. Predict the label on the input  $X$  based on the conjunction of literals in  $L$

3. If a *mistake is made*, eliminate the offending literals from  $L$

- Theorem:

The above algorithm is guaranteed to learn any conjunctive Boolean concept given a non-contradictory sequence of examples in a noise-free environment. The bound on the number of mistakes is  $n + 1$ .

# Concept Learning

## Learning Input–Output Functions

- Target function  $f \in F$ : unknown to the learner
- Hypothesis  $h \in H$ , about what  $f$  might be  
 $H$  — Hypothesis space
- Instance space  $X$ : domain of  $f, h$
- Output space  $Y$ : range of  $f, h$
- Example: an ordered pair  $(x, y)$ ,  $x \in X$  and  $f(x) = y \in Y$
- $F$  and  $H$  may or may not be the same!
- Training set  $E$ : a *multi-set of examples*
- Learning algorithm  $L$ : a procedure which given some  $E$ , outputs an  $h \in H$

# Dimensions of Concept Learning

Representation:

## 1. Instances

- Symbolic
- Numeric

## 2. Hypotheses (i.e., concept description)

- Attribute-value (propositional logic)
- Relational (first-order logic)

Semantic associated with both representations

Level of learning

- Symbolic
- Sub-symbolic

Method of learning

1. Bottom-up (covering)
2. Top-down

# Case Study: Concept of EnjoySport

Sky	Temp	Humid	Wind	Water	Forecast	EnjoySport
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

(Many possible representations)

Here,  $h$  is conjunction of constraints on attributes

Each constraint can be

- a specific 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      Forecast  
 $\langle Sunny \quad ? \quad ? \quad Strong \quad ? \quad Same \rangle$

# Prototypical Concept Learning Task

- **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

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

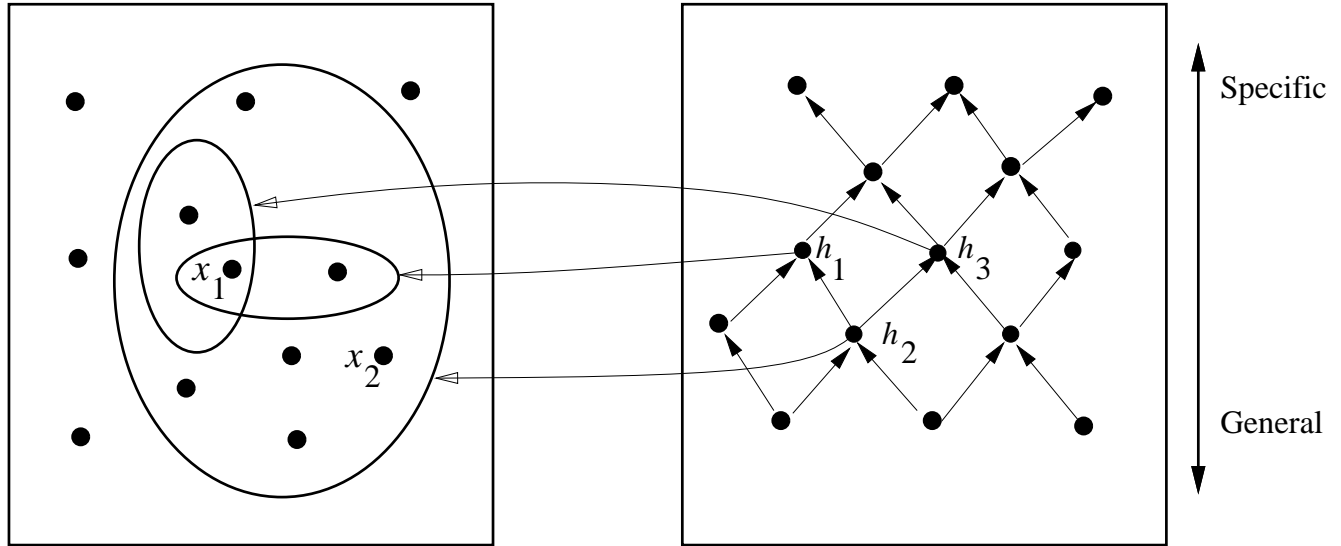
- **Determine:** A hypothesis  $h$  in  $H$  such that  $h(x) = c(x)$  for all  $x$  in  $D$ .

**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

*Instances X*

*Hypotheses H*



$x_1 = \langle \text{Sunny, Warm, High, Strong, Cool, Same} \rangle$

$x_2 = \langle \text{Sunny, Warm, High, Light, Warm, Same} \rangle$

$h_1 = \langle \text{Sunny, ?, ?, Strong, ?, ?} \rangle$

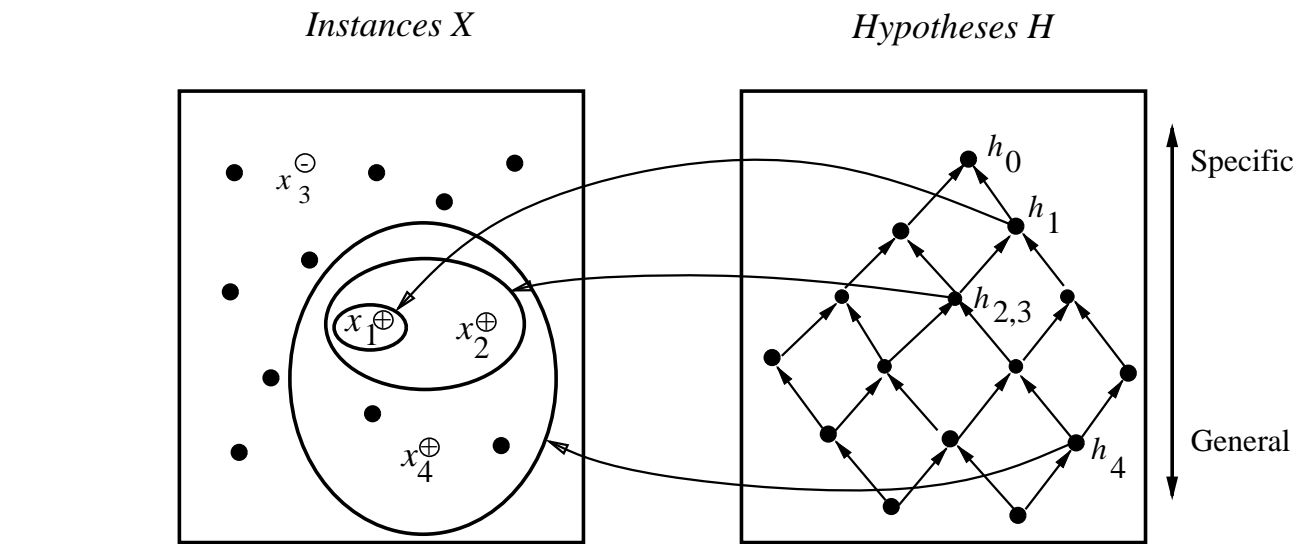
$h_2 = \langle \text{Sunny, ?, ?, ?, ?, ?} \rangle$

$h_3 = \langle \text{Sunny, ?, ?, ?, Cool, ?} \rangle$

## 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



$x_1 = \langle \text{Sunny Warm Normal Strong Warm Same} \rangle, +$   
 $x_2 = \langle \text{Sunny Warm High Strong Warm Same} \rangle, +$   
 $x_3 = \langle \text{Rainy Cold High Strong Warm Change} \rangle, -$   
 $x_4 = \langle \text{Sunny Warm High Strong Cool Change} \rangle, +$

$h_0 = \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$   
 $h_1 = \langle \text{Sunny Warm Normal Strong Warm Same} \rangle$   
 $h_2 = \langle \text{Sunny Warm ? Strong Warm Same} \rangle$   
 $h_3 = \langle \text{Sunny Warm ? Strong Warm Same} \rangle$   
 $h_4 = \langle \text{Sunny Warm ? Strong ? ?} \rangle$

## 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  $D$ .

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

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  $D$ .

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

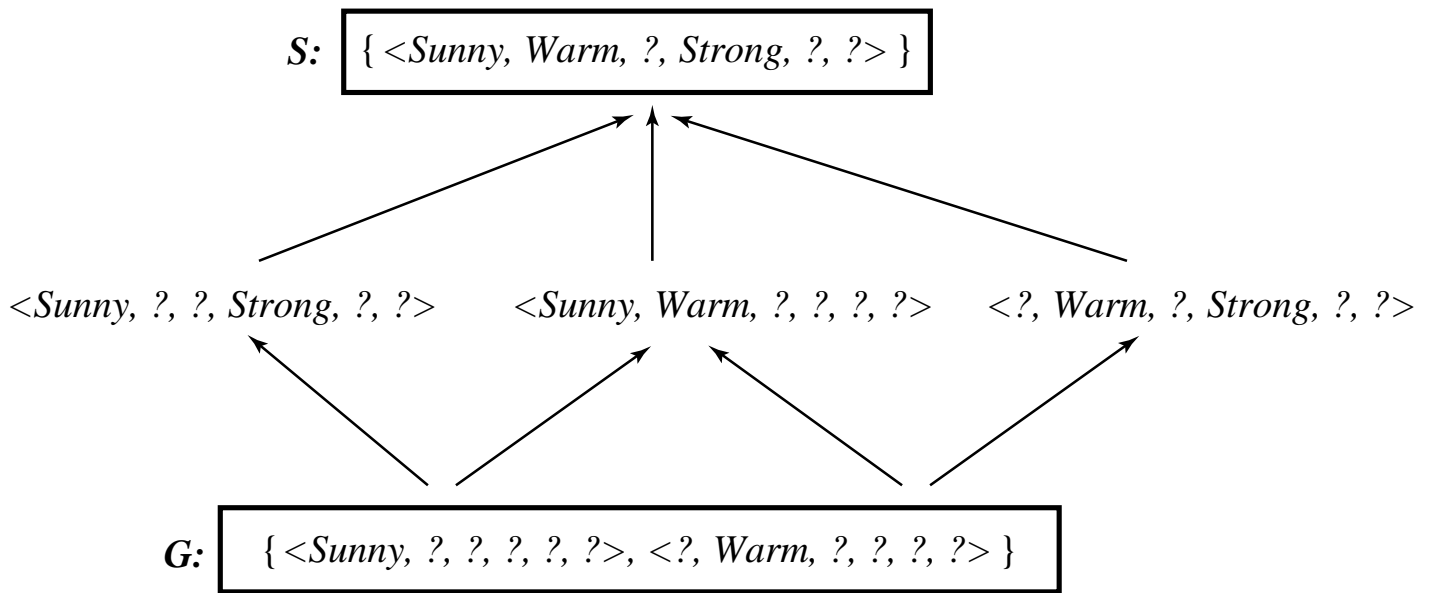
## The List-Then-Eliminate **Algorithm**:

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



## 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

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

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

# Candidate Elimination Algorithm

$G \leftarrow$  maximally general hypotheses in  $H$

$S \leftarrow$  maximally specific hypotheses in  $H$

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  $S$

## Candidate Elimination Algorithm (Continued)

- 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**<sub>0</sub>: {<∅, ∅, ∅, ∅, ∅, ∅>}

**G**<sub>0</sub>: {<?, ?, ?, ?, ?, ?>}

# Selecting New Training Instances

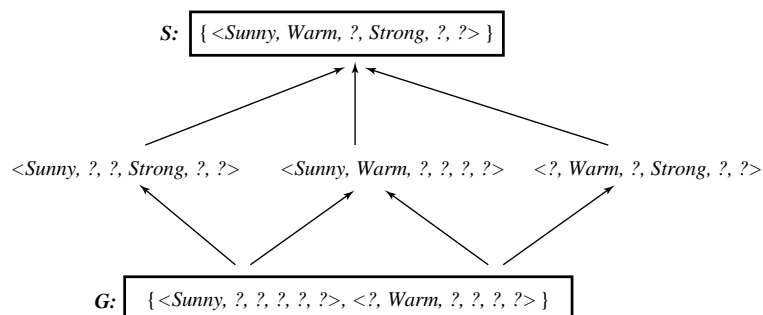
- We assumed that teacher provided examples to learner
- What if learner select its own instances for learning?
  1. A bunch of new instances are given to learner, without classification as 0/1
  2. Learner selects a new instance

Such selected instance is called a *query*

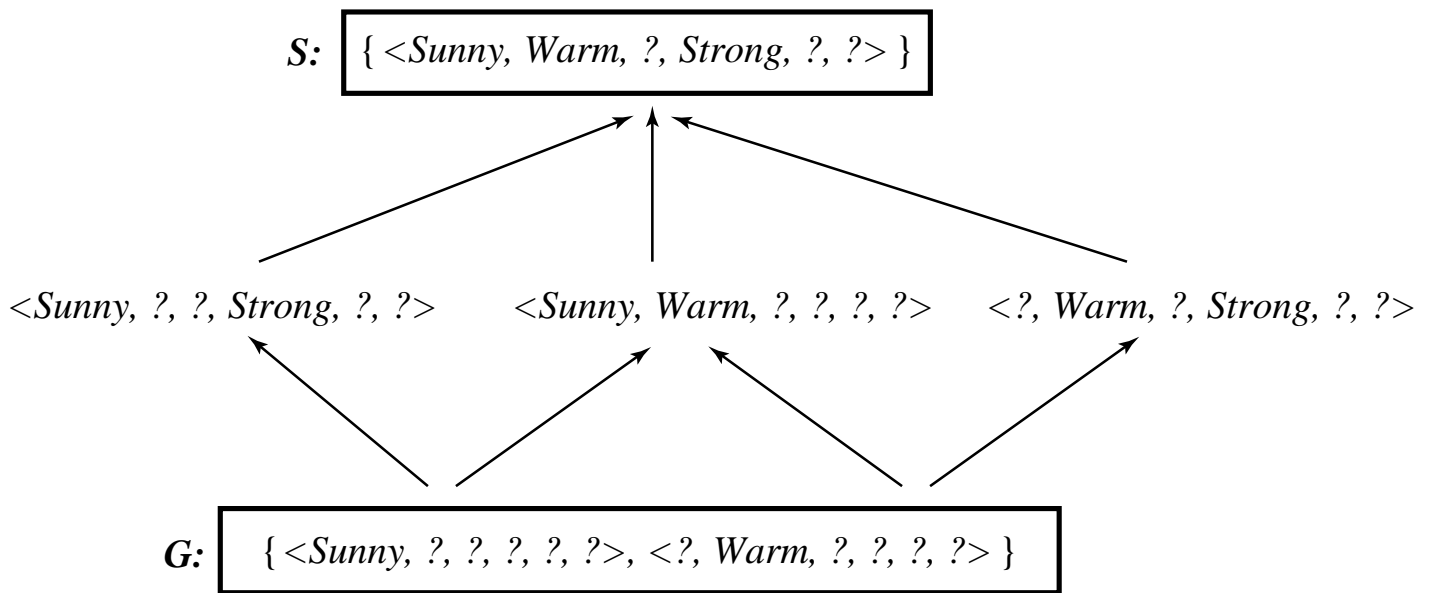
How to select new instance?

- Choose the one that comes closest to matching half of the hypotheses in the version space

3. Learner requests teacher for the correct classification of query
4. Learner updates the version space accordingly



# How Should These Be Classified?



*<Sunny Warm Normal Strong Cool Change>*

*<Rainy Cool Normal Light Warm Same>*

*<Sunny Warm Normal Light Warm Same>*

## What Justifies this Inductive Leap?

+ *<Sunny Warm Normal Strong Cool Change>*

+ *<Sunny Warm Normal Light Warm Same>*

⇓

S : *<Sunny Warm Normal ? ? ?>*

## Why believe we can classify the unseen

*<Sunny Warm Normal Strong Warm Same>*

# Convergence and Bias

- Candidate-Elimination algorithm converge toward the true target concept, provided

1. There are no errors in the training examples

If there is error: The algorithm will remove the true target concept from the version space, because it will eliminate all hypotheses that are inconsistent with each training example.

2.  $H$  contains some  $h$  that correctly describes the target concept

If not: Then see example below for learning disjunctive concepts such as

$Sky = Sunny$  or  $Sky = Cloudy$

Sky	Temp	Humid	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Cool	Change	Yes
Cloudy	Warm	Normal	Strong	Cool	Change	Yes
Rainy	Warm	Normal	Strong	Cool	Change	No

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

$S_2$  is overly general since it covers the third training instance

Learner is biased to consider only conjunctive hypotheses. It cannot learn disjunctive concepts.

# An Un-Biased 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.,

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

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

$S \leftarrow$

$G \leftarrow$

Problem: Learner is unable to generalize beyond the observed training examples

1.  $S$  boundary: disjunction of observed positives examples
2.  $G$  boundary: negated disjunction of observed negative examples

Only the observed training examples will be un-ambiguously classified by  $S$  and  $G$

Only the observed training examples will be unanimously classified by the version space

# Learning and Bias

- Absolute bias: Restricted hypothesis space bias

Weaker bias: more open to experience, more expressive hypothesis space, lower generalizability

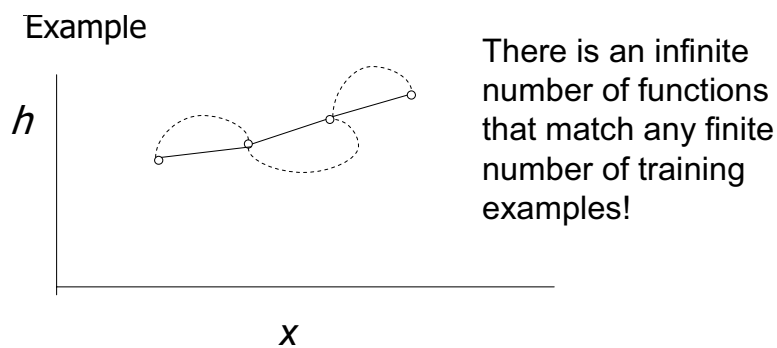
Stronger bias: higher generalizability

- Implicit vs explicit bias

- Preferential bias: Selection based on some ordering criteria

Occam's Razor: Simpler (shorter) hypotheses preferred

Learning in practice requires a tradeoff between complexity of hypothesis space and goodness of fit



Bias free function learning is impossible!

# 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  $D_c$

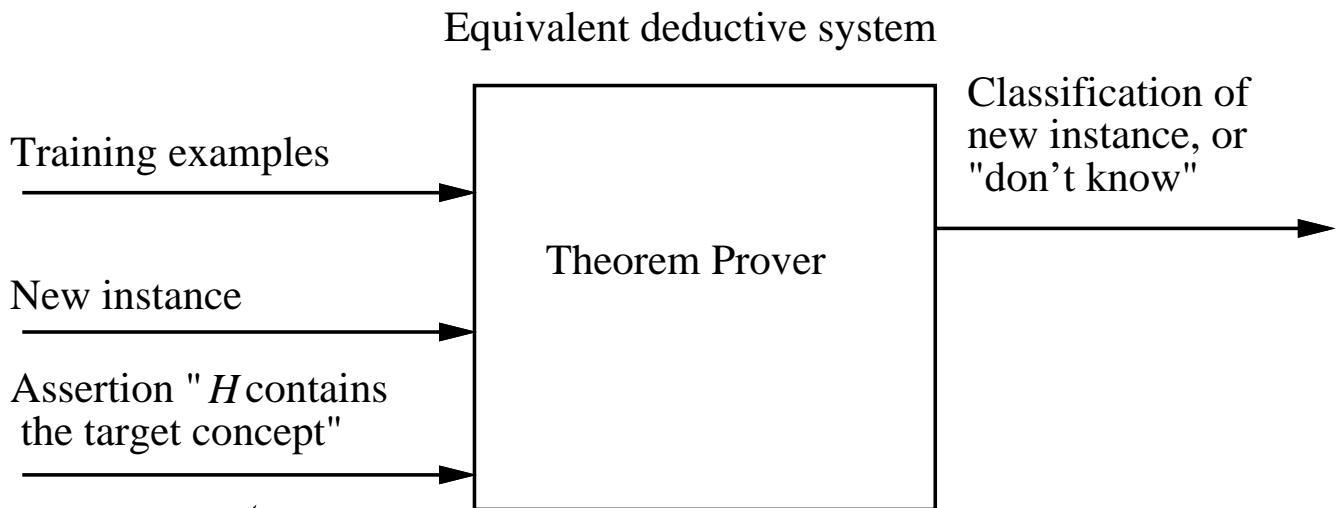
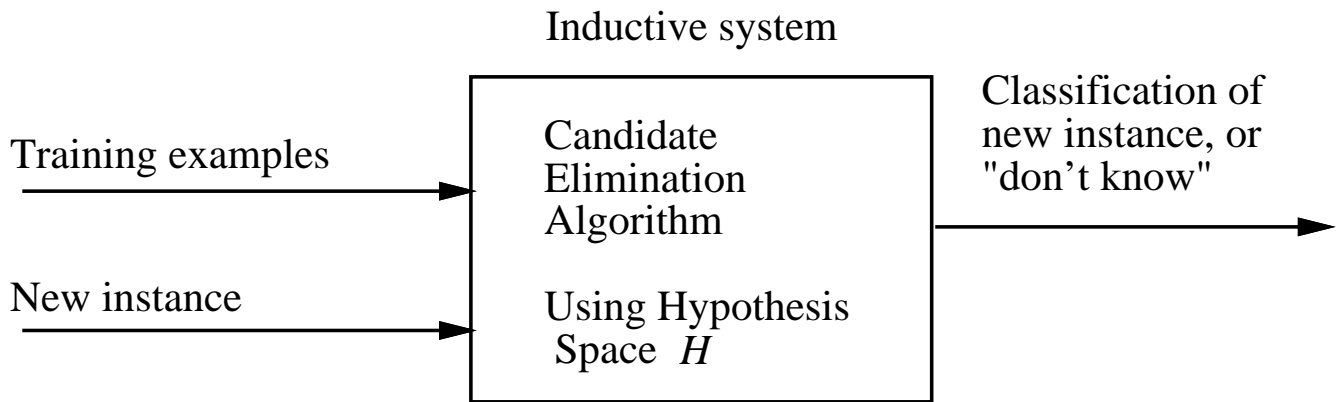
$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$$

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

# Inductive Systems and Equivalent Deductive Systems

**Inductive Learning** A hypothesis (e.g. a classifier) that is consistent with a sufficiently large number of representative training examples is likely to accurately classify novel instances drawn from the same universe

With stronger bias, there is less reliance on the training data



*Inductive bias  
made explicit*

## Three Learners with Different Biases

1. *Rote learner*: Store examples, Classify  $x$  iff it matches previously observed example. No bias
2. *Version space candidate elimination algorithm*: Stronger bias
3. *Find-S*: Strongest bias

### 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