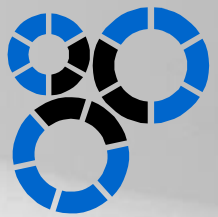
Machine Learning
CSE - 465

Lecture - 06

Outline

- List- Then- Eliminate Algorithm
- Candidate Elimination Algorithm



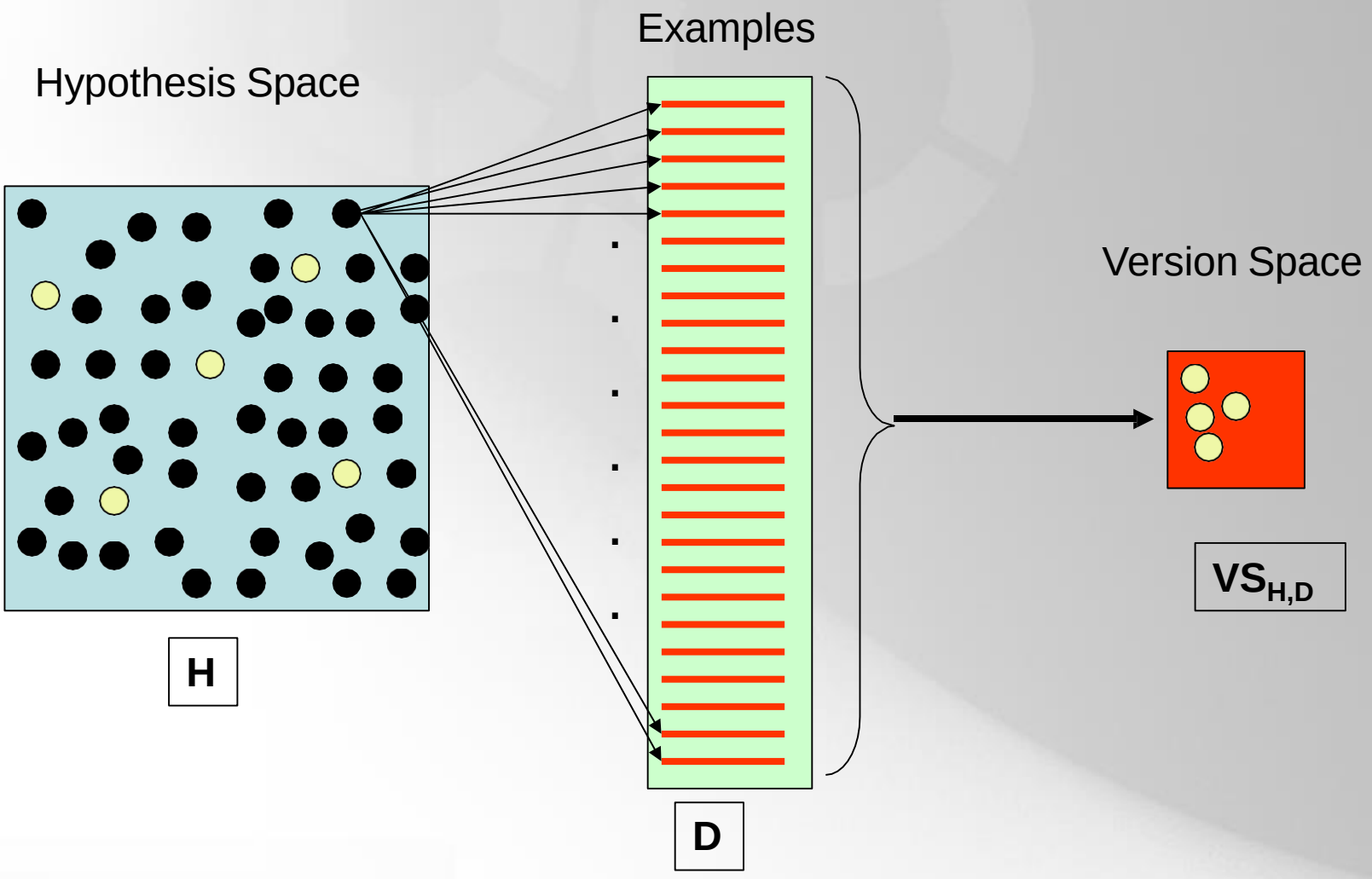
LIST-THEN-ELIMINATE Algorithm to Obtain Version Space

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 $VersionSpace$

LIST-THEN-ELIMINATE Algorithm to Obtain Version Space



Candidate-Elimination Algorithm

The **Candidate-Elimination** algorithm finds all describable hypotheses that are consistent with the observed training examples

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

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

Hypothesis is derived from examples regardless of whether x is positive or negative example

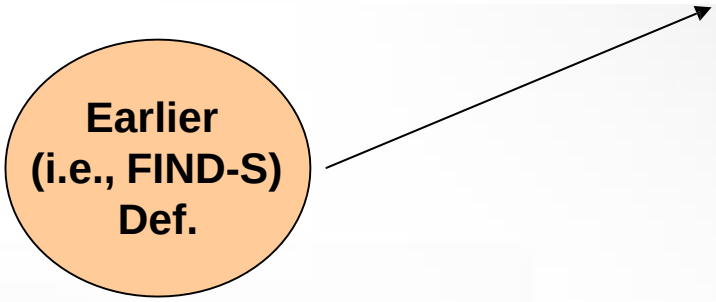
Candidate-Elimination Algorithm

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

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

Definition: Let h_j and h_k be boolean-valued functions defined over X. Then h_j is **more_general_than_or_equal_to** h_k (written $h_j \geq_g h_k$) if and only if

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$



LIST-THEN-ELIMINATE Algorithm to Obtain Version Space

- In principle, the **LIST-THEN-ELIMINATE** algorithm can be applied whenever the hypothesis space H is finite.

- It is guaranteed to output all hypotheses consistent with the training data.

- Unfortunately, it requires exhaustively enumerating all hypotheses in H-an unrealistic requirement for all but the most trivial hypothesis spaces.

Candidate-Elimination Algorithm

- The **CANDIDATE-ELIMINATION** algorithm **works on the same principle** as the above **LIST-THEN-ELIMINATE** algorithm.

- It employs a much more **compact representation of the version space**.

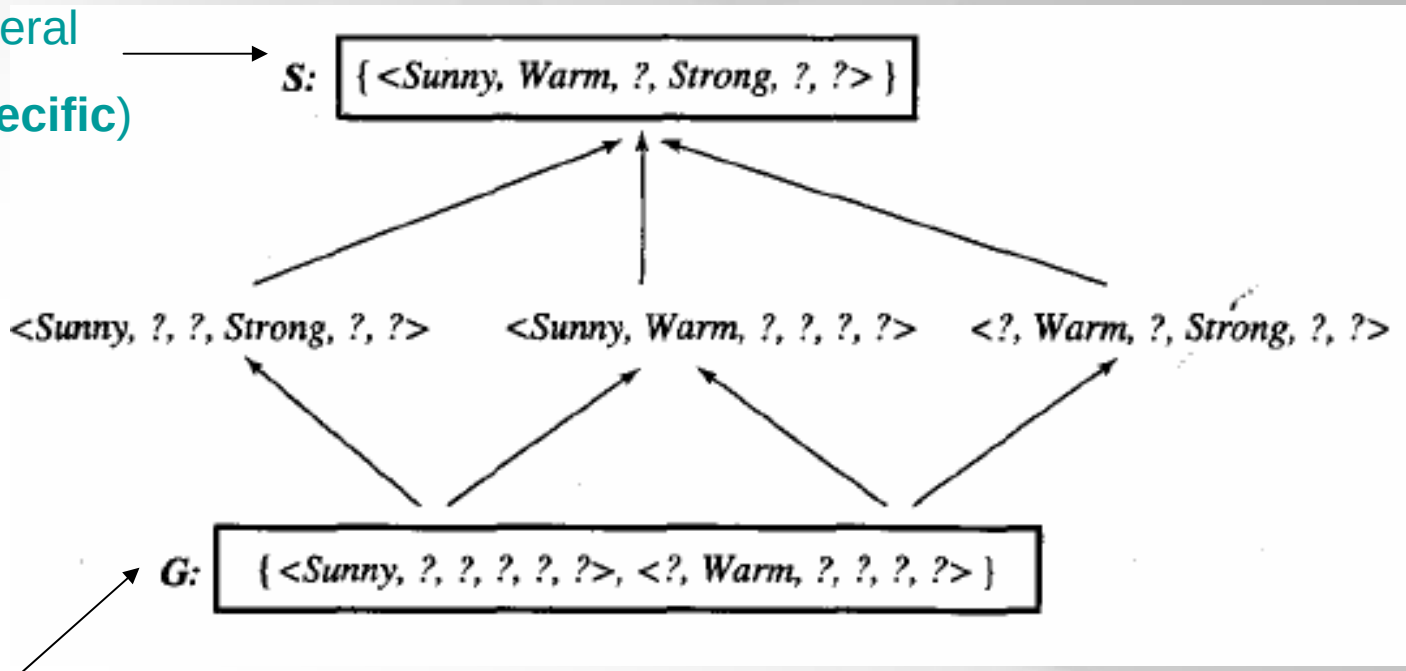
- In this the **version space is represented by its most general and least general members (Specific)**.

- These members form general and specific boundary sets that delimit the version space within the partially ordered hypothesis space.

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Least General

(**Most Specific**)



Most General

(Least Specific)

Candidate-Elimination Algorithm

Initialize G to the set of maximally general hypotheses in H
Initialize S to the set of 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
      - h is consistent with d, and 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
      - h is consistent with d, and some member of S is more specific than h
    - Remove from G any hypothesis that is less general than another hypothesis in G

Example

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

G0 ← {<?, ?, ?, ?, ?, ?>}

Initialization

S0 ← {<Ø, Ø, Ø, Ø, Ø, Ø >}

G0 ← {<?, ?, ?, ?, ?, ?>}

S0 ← {<Ø, Ø, Ø, Ø, Ø, Ø >}

Iteration 1

x1 = <Sunny, Warm, Normal, Strong, Warm, Same>

G1 ← {<?, ?, ?, ?, ?, ?>}

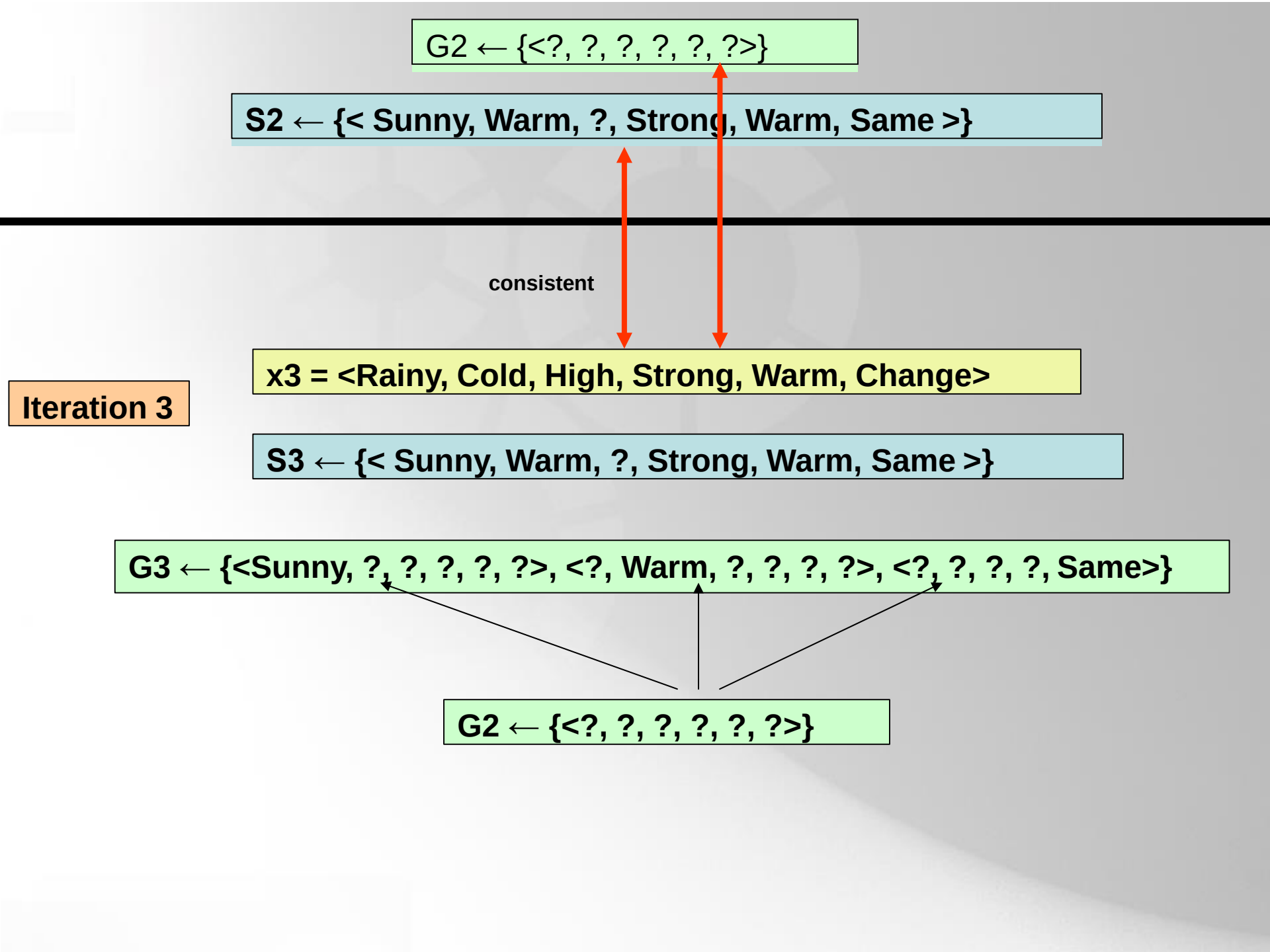
S1 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}

Iteration 2

x2 = <Sunny, Warm, High, Strong, Warm, Same>

G2 ← {<?, ?, ?, ?, ?, ?>}

S2 ← {< Sunny, Warm, ?, Strong, Warm, Same >}


G2 ← {<?, ?, ?, ?, ?, ?>}
S2 ← {< Sunny, Warm, ?, Strong, Warm, Same >}
consistent
x3 = <Rainy, Cold, High, Strong, Warm, Change>
Iteration 3
S3 ← {< Sunny, Warm, ?, Strong, Warm, Same >}
G3 ← {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?>, <?, ?, ?, ?, Same>}
G2 ← {<?, ?, ?, ?, ?, ?>}

S3 ← {< Sunny, Warm, ?, Strong, Warm, Same >}

G3 ← {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?>, <?, ?, ?, ?, Same>}

Iteration 4

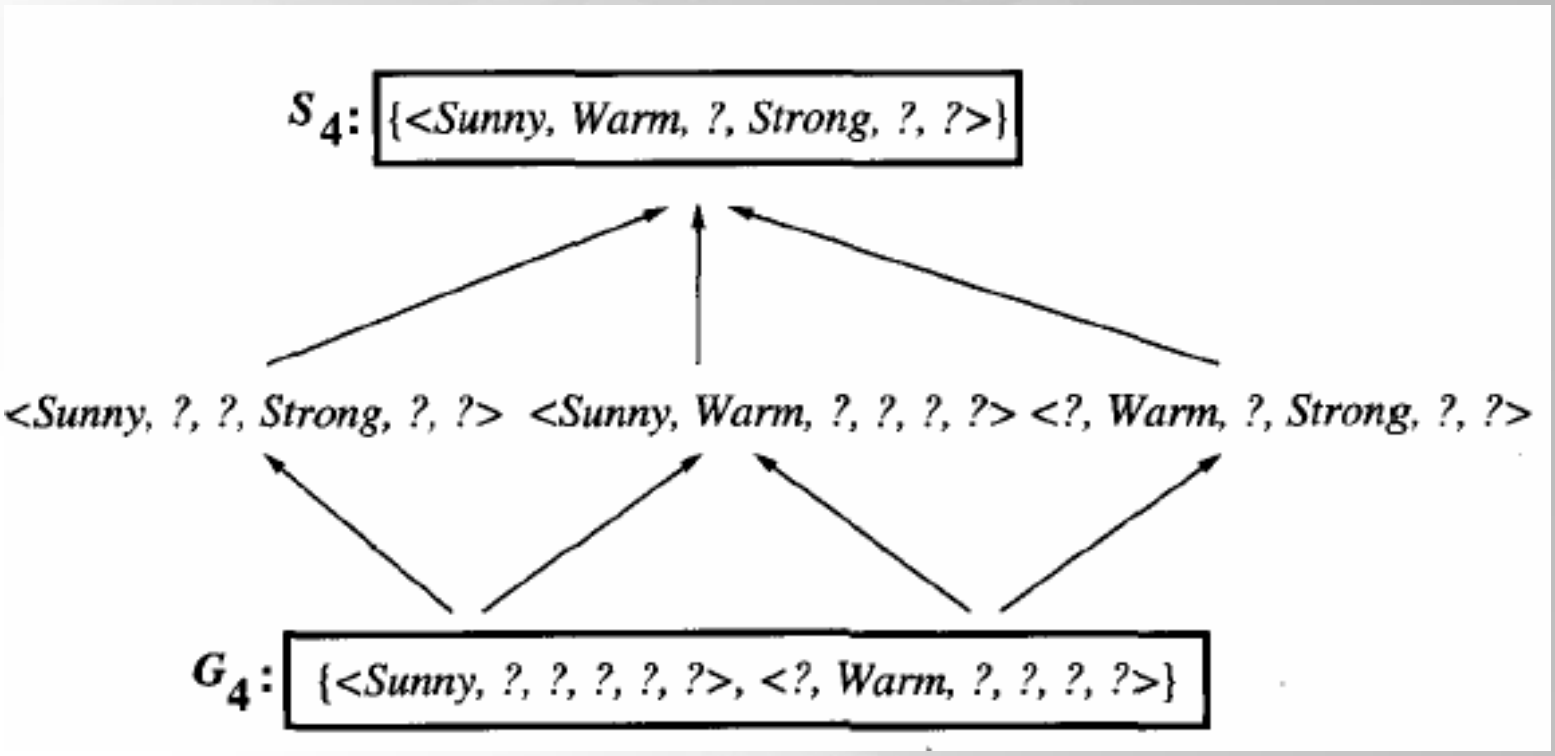
x4 = <Sunny, Warm, high, Strong, Cool, Change>

S4 ← {< Sunny, Warm, ?, Strong, ?, ? >}

G4 ← {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?>}

G3 ← {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?>, <?, ?, ?, ?, Same>}

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No


S_4: {<Sunny, Warm, ?, Strong, ?, ?>}
<Sunny, ?, ?, Strong, ?, ?>
<Sunny, Warm, ?, ?, ?, ?>
<?, Warm, ?, Strong, ?, ?>
G_4: {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?>}

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

<*Sunny, Warm, ?, Strong, ?, ?*>

<*Sunny, ?, ?, Strong, ?, ?*>

<*Sunny, Warm, ?, ?, ?, ?*>

<*?, Warm, ?, Strong, ?, ?*>

<*Sunny, ?, ?, ?, ?, ?*>

<*?, Warm, ?, ?, ?, ?*>

Remarks on Version Spaces and Candidate-Elimination

The version space learned by the **CANDIDATE-ELIMINATION** algorithm will converge toward the hypothesis that correctly describes the target concept, provided

(1) there are no errors in the training examples, and

(2)there is some hypothesis in H that correctly describes the target concept.

What will Happen if the Training Contains errors ?

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	**No**
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

G0 ← {<?, ?, ?, ?, ?, ?>}

S0 ← {<Ø, Ø, Ø, Ø, Ø, Ø >}

Iteration 1

x1 = <Sunny, Warm, Normal, Strong, Warm, Same>

G1 ← {<?, ?, ?, ?, ?, ?>}

S1 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}

Iteration 2

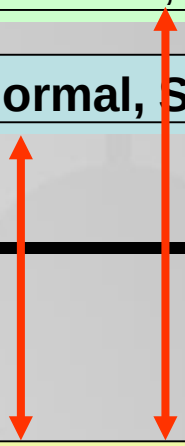
x2 = <Sunny, Warm, High, Strong, Warm, Same>

G2 ← {<?, ?, Normal, ?, ?, ?>}

S2 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}

G2 ← {<?, ?, Normal, ?, ?, ?>}

S2 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}



consistent

x3 = <Rainy, Cold, High, Strong, Warm, Change>

Iteration 3

S3 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}

G3 ← {<?, ?, Normal, ?, ?, ?>}

G3 ← {<Sunny,?,?,?,?,?> <?, Warm,?,?,?,?> <?, ?, Normal, ?, ?, ?>
<?,?,?,?,?, Same>}

S3 ← {< Sunny, Warm, Normal, Strong, Warm, Same >}

G3 ← {<?, ?, Normal, ?, ?, ?>}

S4 ← {< Sunny, Warm, ?, Strong, ?, ? >}

Iteration 4

x4 = <Sunny, Warm, high, Strong, Cool, Change>

S4 ← { }

G4 ← { }

Empty

G4 ← {<?, ?, Normal, ?, ?, ?>}

Remarks on Version Spaces and Candidate-Elimination

The target concept is exactly learned when the S and *G* boundary sets converge to a single, identical, hypothesis.

Remarks on Version Spaces and Candidate-Elimination

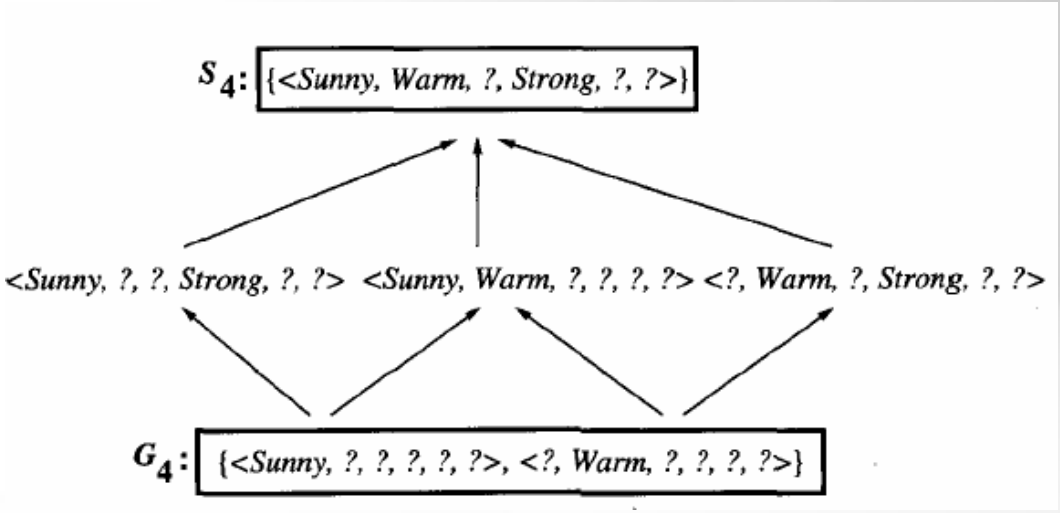
How Can Partially Learned Concepts Be Used?

Suppose that no additional training examples are available beyond the four in our example. And the learner is now required to classify new instances that it has not yet observed.

The target concept is exactly learned when the S and *G* boundary sets converge to a single, identical, hypothesis.

Remarks on Version Spaces and Candidate-Elimination

Instance	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
A	Sunny	Warm	Normal	Strong	Cool	Change	?
B	Rainy	Cold	Normal	Light	Warm	Same	?
C	Sunny	Warm	Normal	Light	Warm	Same	?
D	Sunny	Cold	Normal	Strong	Warm	Same	?



<Sunny, Warm, ?, Strong, ?, ?>

<Sunny, ?, ?, Strong, ?, ?>

<Sunny, Warm, ?, ?, ?, ?>

<?, Warm, ?, Strong, ?, ?>

<Sunny, ?, ?, ?, ?, ?>

<?, Warm, ?, ?, ?, ?>

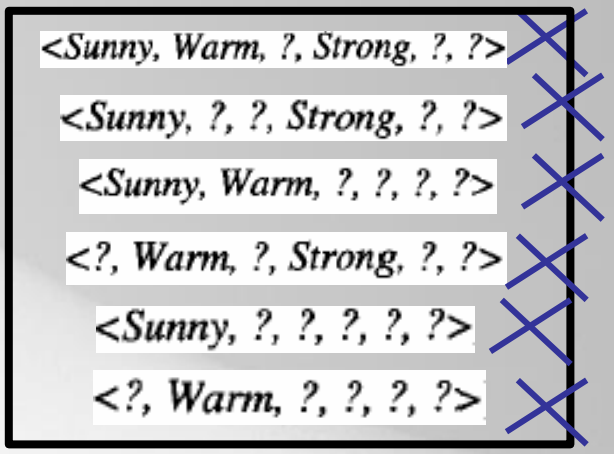
Remarks on Version Spaces and Candidate-Elimination

A	Sunny	Warm	Normal	Strong	Cool	Change

All six hypotheses satisfied

B	Rainy	Cold	Normal	Light	Warm	Same

All six hypotheses dissatisfied

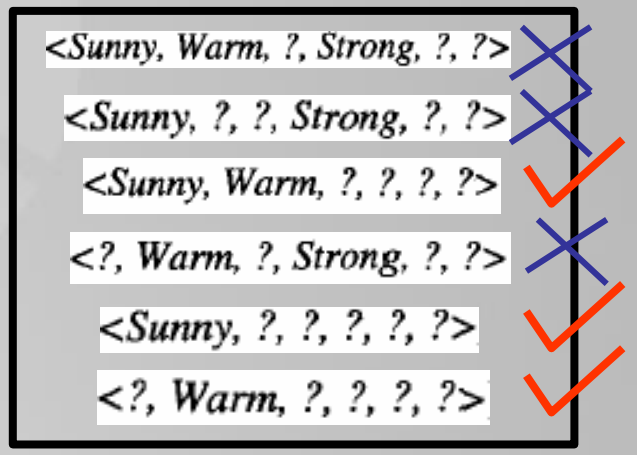


Remarks on Version Spaces and Candidate-Elimination

C Sunny Warm Normal Light Warm Same

Three hypotheses satisfied

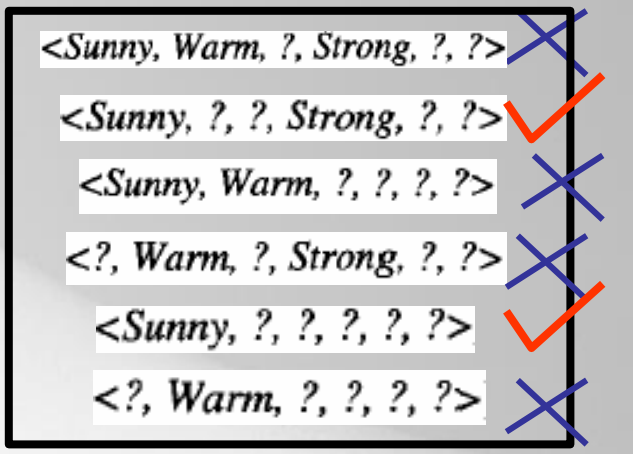
Three hypotheses not satisfied



D Sunny Cold Normal Strong Warm Same

Two hypotheses satisfied

Four hypotheses not satisfied



Thank You