Concept learning and General to Specific Ordering

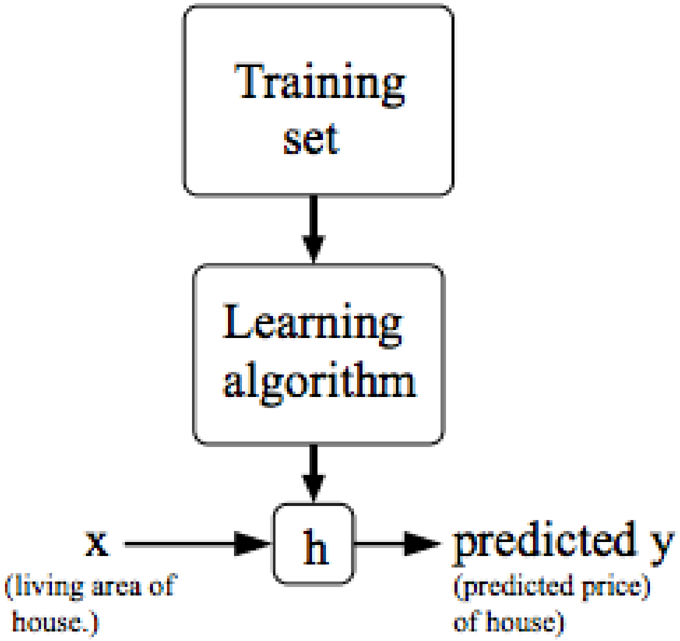
Chapter 2

Recall – Chapter 1

- Define Machine Learning
- Types of Machine Learning
- Design of Machine Learning system
- Linear regression
- Least mean square error minimization
- Gradient descent
- Logistical regression
- Maximum likelihood estimation
- Issues in Machine Learning

General representation of supervised learning

Assume there is a training data set of house area (sq.ft) and its corresponding price.
Aim: Using given data set, predict the price of house for any given living area.
Process: Learning algorithm generates a target function (h) hypothesis that maps input x to output y.
Outcome: Accuracy of hypothesis function is measured using cost/error function



h : x y

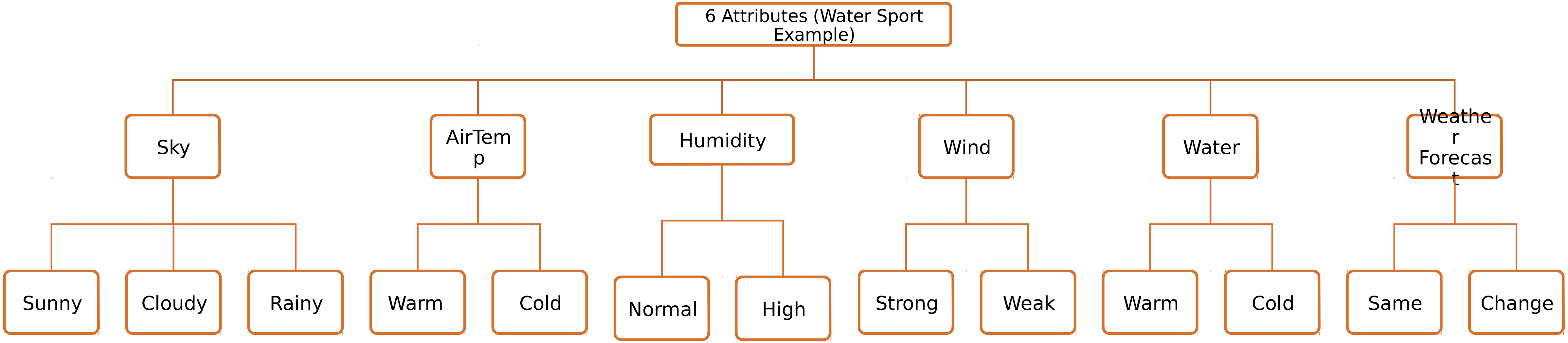
Learning: Induce general function from specific training example

Concept Learning: Inferring a Boolean valued function from training examples of its input and output

Concept learning task

Example: A person enjoys his favorite water sport based on the following attributes.

Each hypothesis is a vector of six attributes. It consists of a conjunction od constraints on the instance attributes.



6 Attributes (Water Sport Example)

Sky

AirTem
p

Humidity

Wind

Water

Weathe
r
Forecas
t

Sunny

Cloudy

Rainy

Warm

Cold

Normal

High

Strong

Weak

Warm

Cold

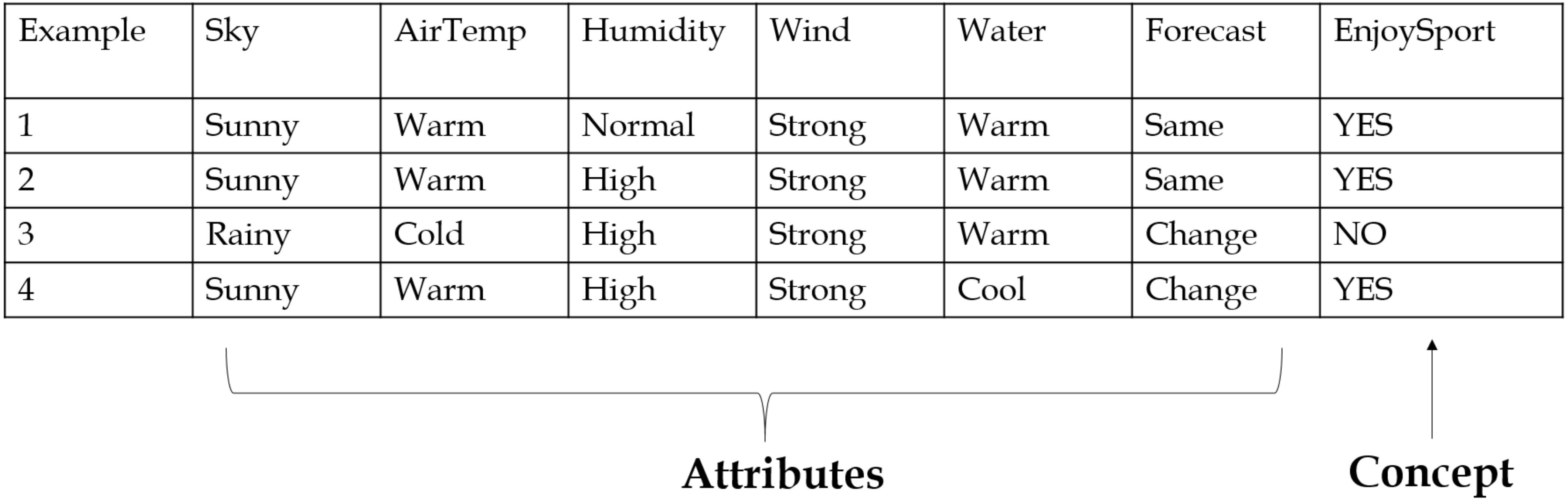
Same

Change

Enjoy Sport
?

Training Example set -D

The set of 4 instances with 3 positive and one negative training examples



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

Attributes

Concept

Concept learning task example

Given

- ***Instances X*** : set of all possible days, each described by the attributes
    - Sky – (values: Sunny, Cloudy, Rainy)
    - AirTemp – (values: Warm, Cold)
    - Humidity – (values: Normal, High)
    - Wind – (values: Strong, Weak)
    - Water – (values: Warm, Cold)
    - Forecast – (values: Same, Change)
- ***Target Concept (Function) c*** : EnjoySport : X □ {0,1}
- ***Hypotheses H*** : Each hypothesis is described by a conjunction of constraints on the attributes.
- ***Training Examples D*** : positive and negative examples of the target function

Determine

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

- For each attribute, the hypothesis will either
  - Indicate by “?” that any value is acceptable (don’t care)
  - Specify a single required value for the attribute (specific)
  - Indicate by “ø” that no value is acceptable (no value)
- A hypothesis:

Sky	AirTemp	Humidity	Wind	Water	Forecast
<Sunny,	?,	?,	Strong,	?,	Same >

The most general hypothesis – every day is a positive example

(?, ?, ?, ?, ?, ?)

- The most specific hypothesis – no day is positive example

(ø, ø, ø, ø, ø, ø)

For the example data discussed in previous table,

- Number of distinct instances?

96 (3*2*2*2*2*2)

- Number of syntactically distinct hypotheses within H?

5120 (5*4*4*4*4*4) [? , ø added to each]

- Number of semantically distinct hypotheses within H?

973 (1 + 4*3*3*3*3*3) [? Added to each + one null entry]

Aim: Find the hypotheses that best fit the training data

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.

  - *Can at best guarantee that the output hypothesis fits the target concept over the training data.*
  - *The assumption is that the best hypothesis regarding unseen instances is the hypothesis that best fits the observed training data.*

General to specific ordering of hypotheses

Consider two hypotheses

h1 = < Sunny, ?, ?, Strong, ?,? >

h2 = < Sunny, ?,?,?,?,? >

Number of instances classified as positive by h2

Number of instances classified as positive by h1

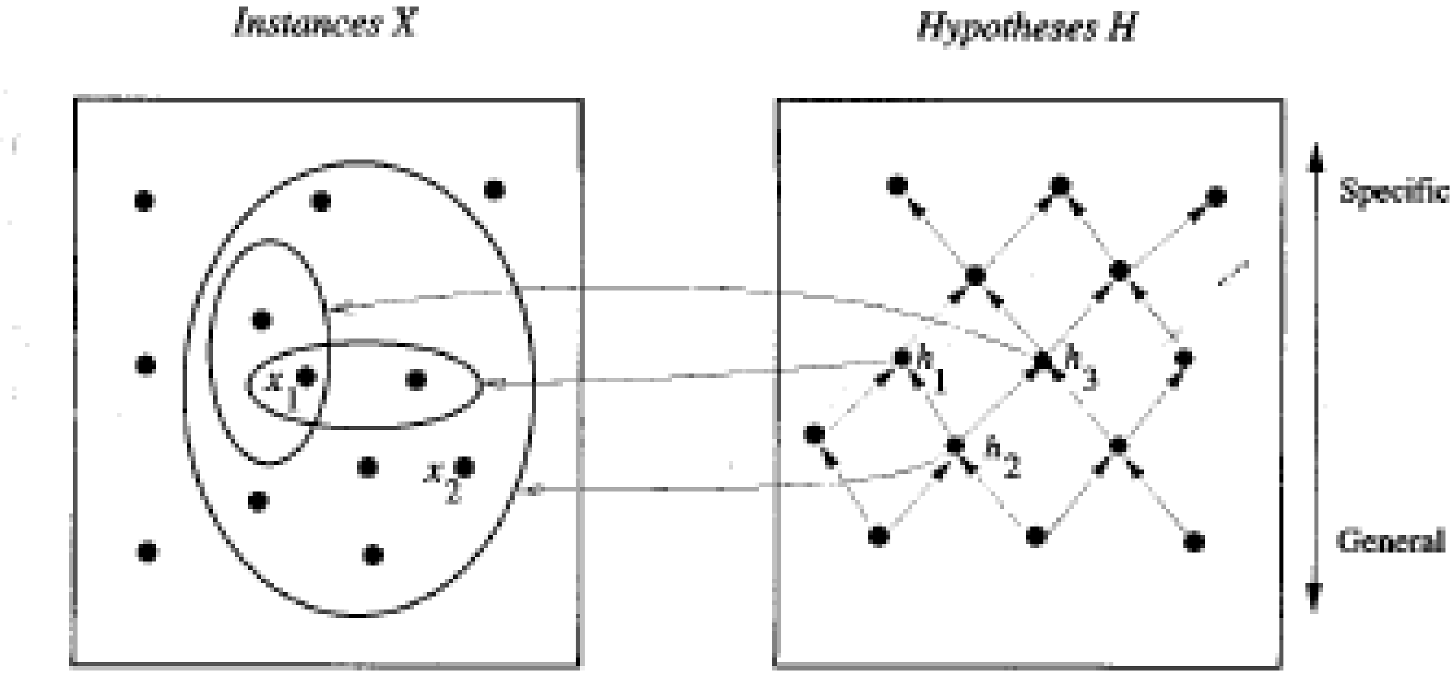
More-General-Than Relation

- For any instance x in X and hypothesis h in H, we say that x satisfies h if and only if h(x) = 1.

More-General-Than-Or-Equal Relation:

- Let h1 and h2 be two boolean-valued functions defined over X.
- Then h1 is ***more-general-than-or-equal-to*** h2 (written h1 ≥ h2) if and only if any instance that satisfies h2 also satisfies h1.
- h1 is ***more-general-than*** h2 (h1 > h2) if and only if h1≥h2 is true and h2≥h1 is false. We also say h2 is ***more-specific-than*** h1.

More-General Relation



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

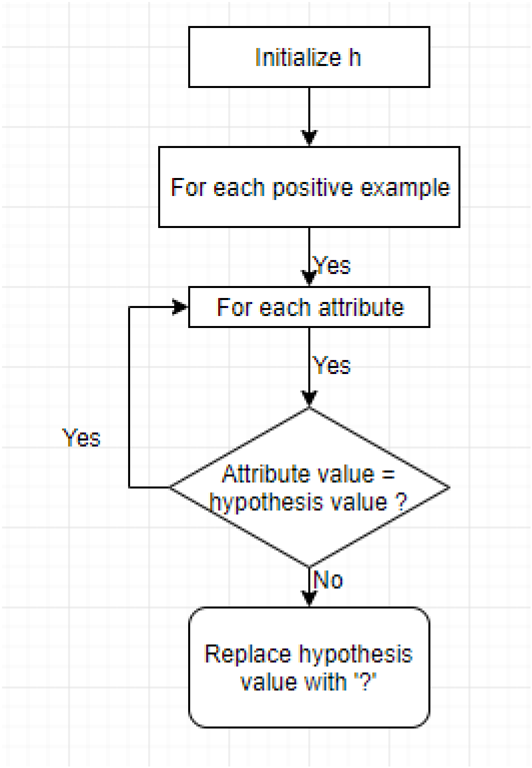
h_1= <Sunny, ?, ?, Strong, ?, ?>
h_2= <Sunny, ?, ?, ?, ?, ?>
h_3= <Sunny, ?, ?, ?, Cool, ?>

Find-S Algorithm : Finding a maximally specific hypothesis

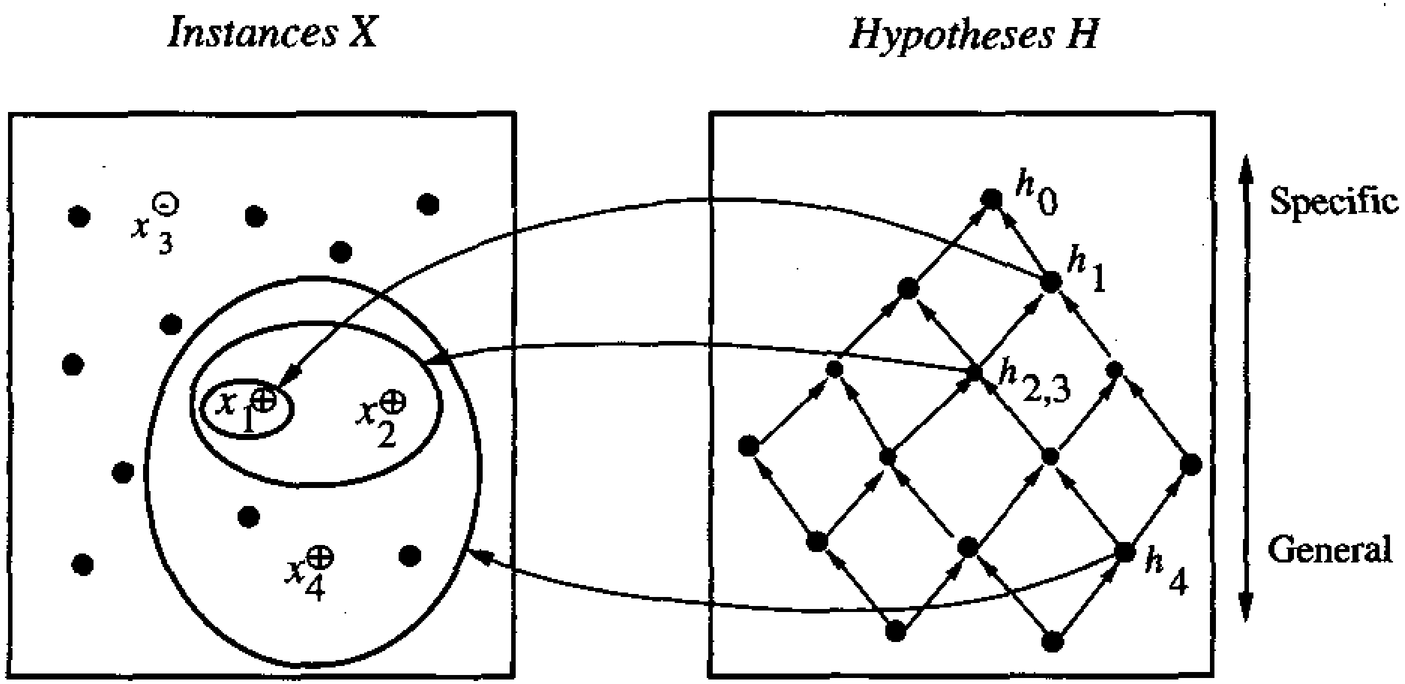
- FIND-S Algorithm starts from the most specific hypothesis and generalize it by considering only positive examples.
- FIND-S algorithm ignores negative examples.
- FIND-S algorithm finds the most specific hypothesis within H that is consistent with the positive training examples.

FIND-S Algorithm

```
Initialize h to the most specific hypothesis in H.
For each positive training instance x
    For each attribute constraint a, in h
    If the constraint a, is satisfied by x
        Then do nothing
    Else replace a, in h by the next more general constraint that is satisfied by x
Output hypothesis h
```


Initialize h
For each positive example
Yes
For each attribute
Yes
Yes
Attribute value = hypothesis value ?
No
Replace hypothesis value with '?'

- Initialize: $h \leftarrow (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
- Replace: $h \leftarrow (Sunny, Warm, Normal, Strong, Warm, Same)$
- Refine: $h \leftarrow (Sunny, Warm, ?, Strong, Warm, Same)$
- No modification in example 3 - Negative
- Further: $h \leftarrow (Sunny, Warm, ?, Strong, ?, ?)$



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

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

x_3 = <*Rainy Cold High Strong Warm Change*>, -

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

h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

h_1 = <*Sunny Warm Normal Strong Warm Same*>

h_2 = <*Sunny Warm ? Strong Warm Same*>

h_3 = <*Sunny Warm ? Strong Warm Same*>

h_4 = <*Sunny Warm ? Strong ? ?* >

Unanswered Questions by FIND-S Algorithm

- Has FIND-S converged to the correct target concept?
- Why prefer the most specific hypothesis?
- Are the training examples consistent?
- What if there are several maximally specific consistent hypotheses?

FIND-S outputs a hypothesis from H, that is consistent with the training examples, this is just one of many hypotheses from H that might fit the training data equally well.

Candidate-Elimination Algorithm

- The key idea in the Candidate-Elimination algorithm is to output a description of the set of all hypotheses **consistent** with the training examples.
- Candidate-Elimination algorithm computes the description of this set without explicitly enumerating all of its members.
- This is accomplished by using the **more-general-than** partial ordering and maintaining a compact representation of the set of consistent hypotheses

Consistent Hypothesis

- 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 < x, c(x) > in D.

Consistent (h, D)

Version Spaces

- The Candidate-Elimination algorithm represents the set of all hypotheses consistent with the observed training examples.

- This subset of all hypotheses is called the ***version space*** because it contains all plausible versions of the target concept.

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

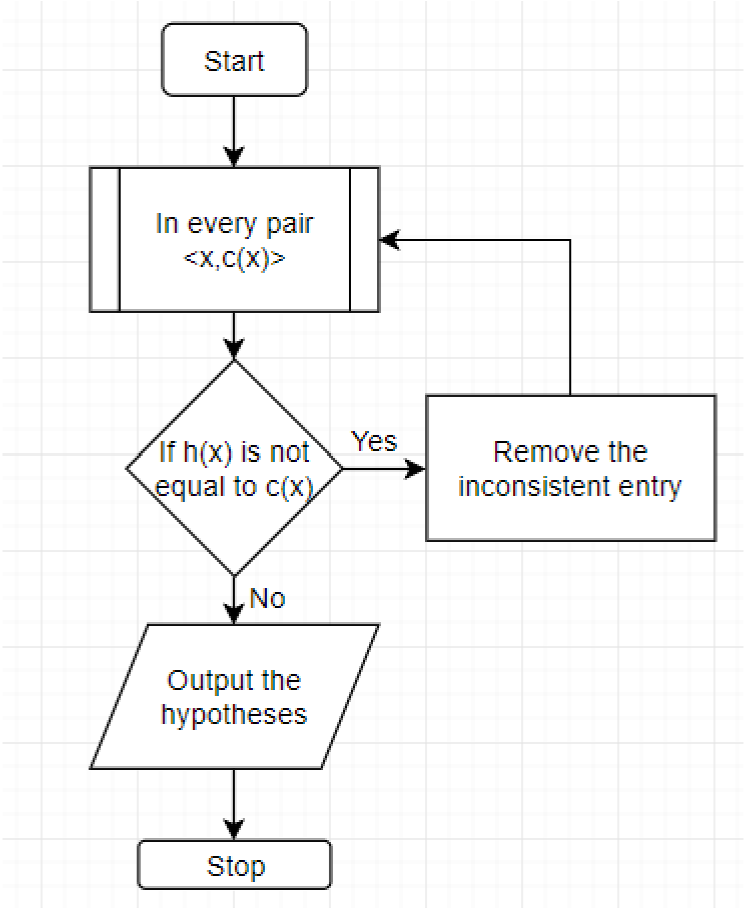
List-Then-Eliminate Algorithm

- List-Then-Eliminate algorithm **initializes the version space** to contain all hypotheses in H, then **eliminates any hypothesis found inconsistent** with any training example.
- The version space of candidate hypotheses thus shrinks as more examples are observed, until ideally just one hypothesis remains that is consistent with all the observed examples.
- List-Then-Eliminate algorithm can be applied whenever the hypothesis space H is finite.

List-Then-Eliminate Algorithm

- VersionSpace ← a list containing every hypothesis in H.

```
For each training example < x, c(x) >
        Remove from VersionSpace any hypothesis h for
        which $h(x) \neq c(x)$
Output the list of hypotheses in VersionSpace
```



Compact Representation of Version Spaces

- The General boundary, G, of version space is the set of its *maximally general members* that are consistent with the given training set

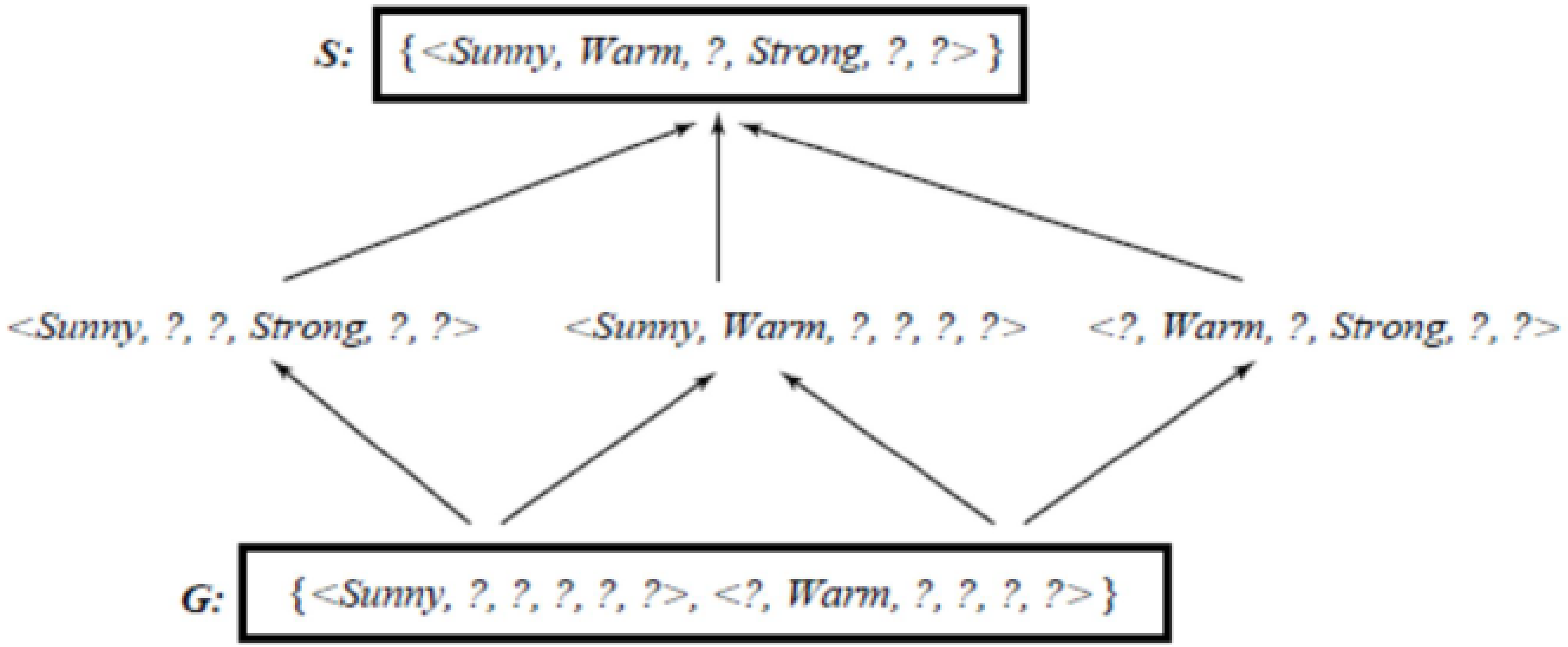
- The Specific boundary, S, of version space is the set of its *maximally specific members* that are consistent with the given training set

- Every member of the version space lies between these boundaries (S, G)

$$\epsilon \ H \mid (\ s \ \epsilon \ S) \ (\ g \ \epsilon \ G) \ (g \geq h \geq s)\}$$

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

Example Version Space



Candidate Elimination Algorithm – Positive Training Examples

Input: training set

Output:

- G = maximally general hypotheses in H
- S = maximally specific hypotheses in H

Algorithm:

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

  (a) h is consistent with d, and

  (b) 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 – Negative Training Examples

- ***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
      - (a) h is consistent with d, and
      - (b) some member of S is more specific than h
    - Remove from G any hypothesis that is less general than another hypothesis in G

Candidate Elimination Algorithm - Summary

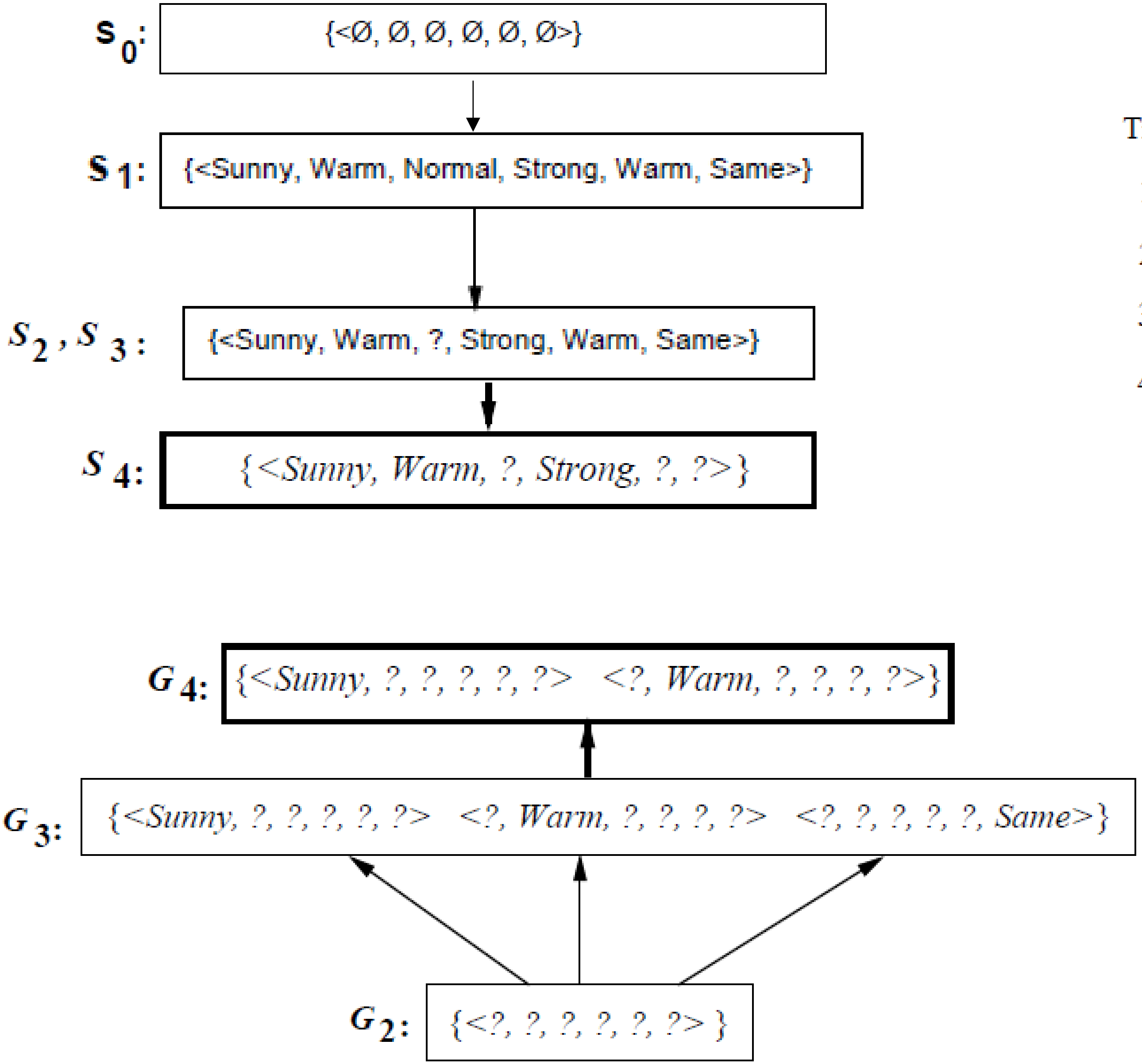
For positive training example

- Tend to generalize specific hypothesis

For negative training example

- Tend to move towards specific from general hypothesis

Example Trace



S_0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

S_1: {<Sunny, Warm, Normal, Strong, Warm, Same>}

S_2 , S_3 : {<Sunny, Warm, ?, Strong, Warm, Same>}

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

G_4: {*<Sunny, ?, ?, ?, ?, ?> <?, Warm, ?, ?, ?, ?>*}

G_3: {*<Sunny, ?, ?, ?, ?, ?> <?, Warm, ?, ?, ?, ?> <?, ?, ?, ?, ?, Same>*}

G_2: {*<?, ?, ?, ?, ?, ?>* }

Training examples:

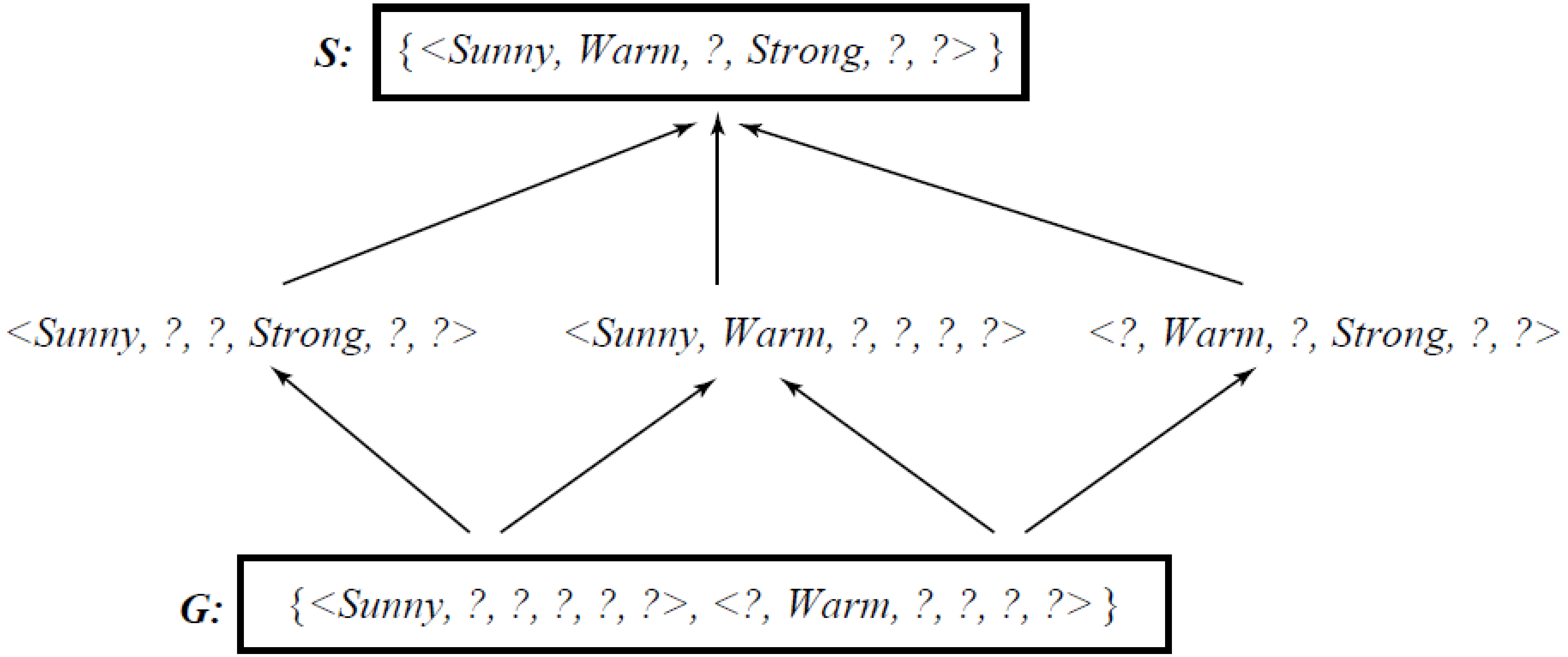
1. <Sunny, Warm, Normal, Strong, Warm, Same>, Enjoy-Sport?=Yes
2. <Sunny, Warm, High, Strong, Warm, Same>, Enjoy-Sport?=Yes
3. *<Rainy, Cold, High, Strong, Warm, Change>, EnjoySport=No*
4. *<Sunny, Warm, High, Strong, Cool, Change>, EnjoySport = Yes*

Properties of the two Sets

- S can be seen as the summary of the positive examples
- Any hypothesis more general than S covers all positive examples
- Other hypotheses fail to cover at least one positive example

- G can be seen as the summary of the negative examples
- Any hypothesis more specific than G covers no previous negative example
- Other hypothesis cover at least one positive example

Resulting Version Space



Properties

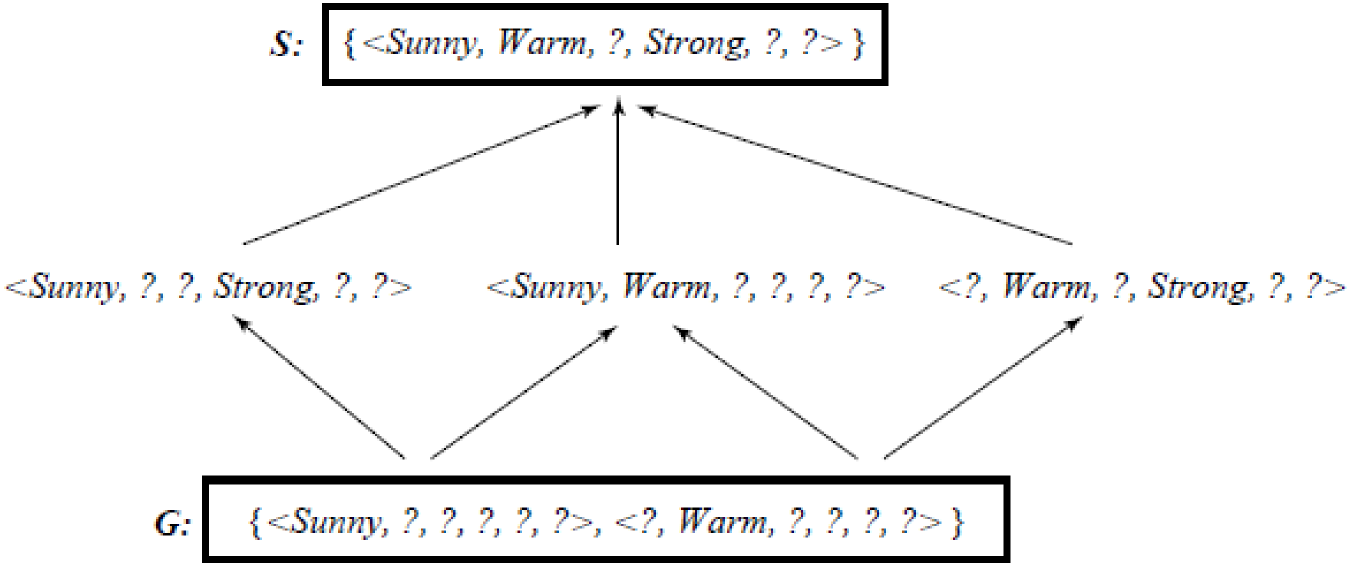
- **If there is a consistent hypothesis then the algorithm will converge to S = G = {h} when enough examples are provided**
- False examples may cause the removal of the correct h
- **If the examples are inconsistent, S and G become empty**
- This can also happen, when the concept to be learned is not in H

What Next Training Example?

If the algorithm is allowed to select the next example, which is best?

Ideally, choose an instance that is classified positive by half and negative by the other half of the hypothesis in Version Space. In either case (positive or negative example), this will eliminate half of the hypothesis. E.g: <Sunny Warm Normal Light Warm Same>

How should these be classified?



S: {<Sunny, Warm, ?, Strong, ?, ?> }

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

G: {<Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?> }

- $\langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change \rangle$
- $\langle Rainy\ Cool\ Normal\ Light\ Warm\ Same \rangle$
- $\langle Sunny\ Warm\ Normal\ Light\ Warm\ Same \rangle$

Classification

- Classify a new example as positive or negative, if all hypotheses in the version space agree in their classification
- Otherwise:

  Rejection or

  Majority vote

Inductive Bias

- What if target concept not contained in hypothesis space?
- Should we include every possible hypothesis?
- How does this influence the generalization ability?

An Unbiased Learner

Idea: Choose H that expresses every teachable concept

- H corresponds to the power set of X |H| =

  much bigger than before, where |H| = 937

- Consider H’ = disjunctions, conjunctions, negations over previous H. E.g.,

<Sunny Warm Normal ? ? ?> ¬ <? ? ? ? ? Change>

- It holds h(x) = 1 if x satisfies the logical expression.
- What are S, G in this case?

Inductive Bias

- Concept learning algorithm L
- Instances X, target concept c
- Training examples
- Let denote the classification assigned to the instance by L after training on data , e.g. EnjoySport = yes

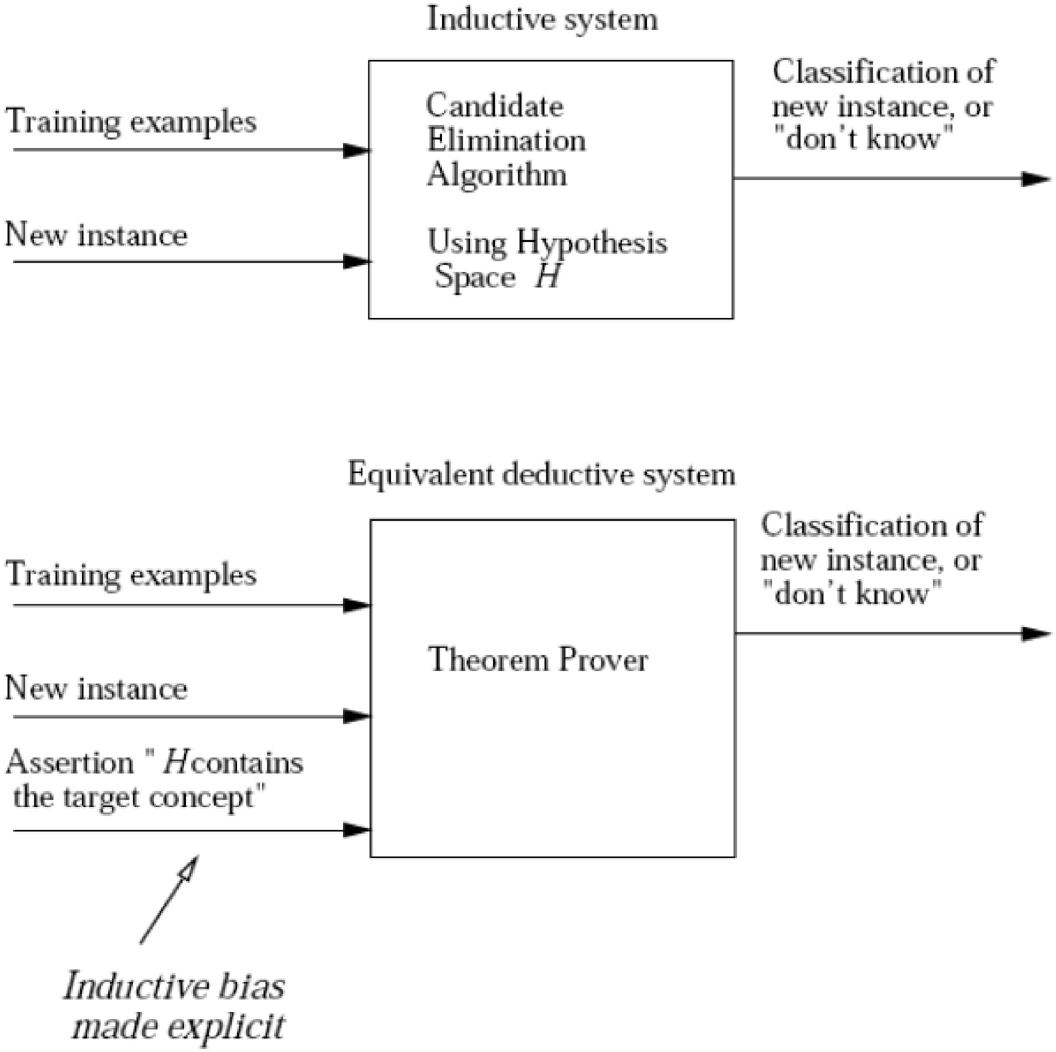
Definition: The inductive bias of L is any minimal set of assertions B such that for any target concept c and corresponding training examples

where A B means A logically entails B.

Inductive Bias for Candidate Elimination

- Assume a training set . The algorithm computes the version space
- is classified by unanimous voting this way is computed.
- Conjecture: B = {c H} is the inductive bias
- From c H it follows that c is a member of the version space.
- = k implies that all members of , including c, vote for class k. Therefore c(= k =
- This means, that the output of the learner can be logically deduced from
- The inductive bias of the Candidate Elimination Algorithm is: c is in H

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

- In the general case, it’s much more difficult to determine the inductive bias
- Often properties of the learning algorithm have to be included, e.g. its search strategy
- Inductive bias is of
  - *Rote learner :* Store examples, Classify x iff it matches previously observed example. No inductive bias (no generalization)
  - *Candidate elimination algorithm :* c is in H
  - *Find-S :* c is in H and that all instances are negative examples unless the opposite is entailed by its training data

A good generalization capability of course depends on the appropriate choice of the inductive bias!

Summary

- Concept learning as search through H
- General-to-specific ordering over H
- Version space candidate elimination algorithm
- S and G boundaries characterize learner’s uncertainty
- Learner can generate useful queries
- Inductive bias

Exercise

1. For the training example of EnjoySport, how would the number of possible instances and possible hypotheses increase with the addition of the attribute *Watercurrent,* which can take on the values *Light, Moderate,* or *Strong?*

Ans:

No. of instances = 96 * 3 = **288**

No. of syntactically distinct hypothesis = 5120 * 5 = **25,600**

No. of semantically distinct hypothesis = 1 + 972*4 = **3,889**

Exercise

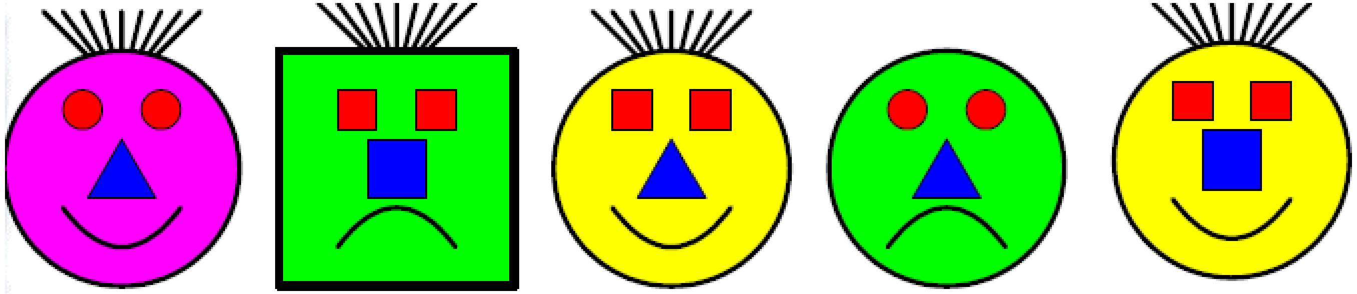
2. Use **Find-S algorithm** to find the specific hypothesis that best describes the given training data set D to **predict** whether the unknown smiley face has the expression of smile or not.

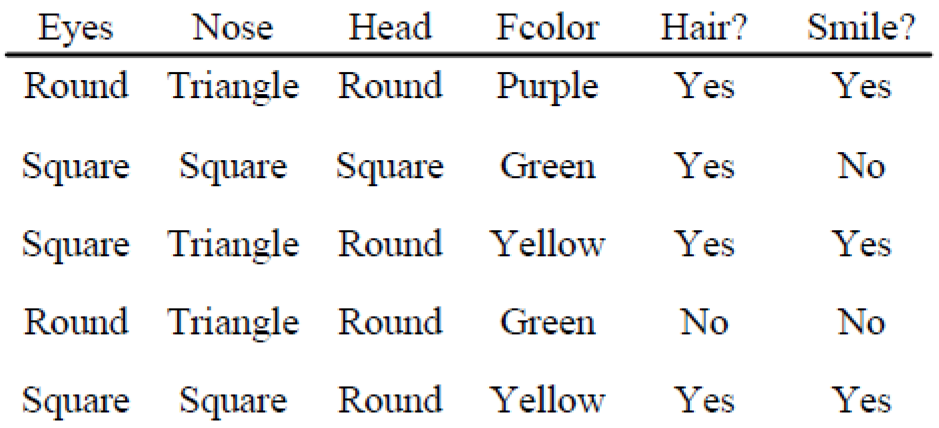
Features : Eyes, Nose, Head, Face_Color, Hair_Presence

Also, compute the total number of instances, syntactical and semantical hypotheses.

Draw the hypotheses space diagram

For the same training example dataset, find the consistent hypothesis using candidate elimination algorithm,.





Eyes	Nose	Head	Fcolor	Hair?	Smile?
Round	Triangle	Round	Purple	Yes	Yes
Square	Square	Square	Green	Yes	No
Square	Triangle	Round	Yellow	Yes	Yes
Round	Triangle	Round	Green	No	No
Square	Square	Round	Yellow	Yes	Yes

⇩

Eyes	Nose	Head	Fcolo r	Hair	Smile?
R	T	R	P	Y	Y
S	S	S	G	Y	N
S	T	R	Y	Y	Y
R	T	R	G	N	N
S	S	R	Y	Y	Y

Solution : Find-S

Sl. No	Eyes	Nose	Head	Fcolor	Hair	Smile ?
1	R	T	R	P	Y	Y
2	S	S	S	G	Y	N
3	S	T	R	Y	Y	Y
4	R	T	R	G	N	N
5	S	S	R	Y	Y	Y

h_0 = (ø, ø, ø, ø, ø)

h_1 = (R, T, R, P, Y) from example 1

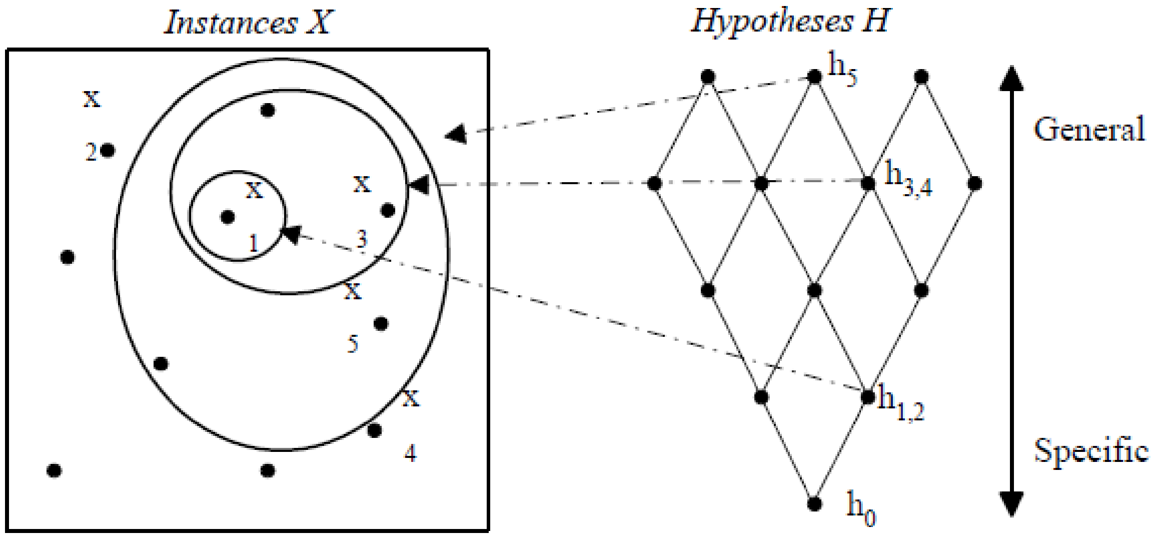
h_2 = h_1 Since negative example (2) is ignored

h_3 = (?, T, R, ?, Y) from example 3

h_4 = h_3 Since negative example (4) is ignored

h_5 = (?, ?, R, ?, Y) from example 5

Hence the most specific hypothesis that best describes this training set D is given as:
S = (?, ?, R, ?, Y)



h_0=<ϕ,ϕ,ϕ,ϕ,ϕ >

x_1=<Round,Triangle,Round,Purple,Yes> +
x_2=<Square,Square,Square,Green,Yes> -
x_3=<Square,Triangle,Round,Yellow,Yes> +
x_4=<Round,Triangle,Round,Green,No> -
x_5=<Square,Square,Round,Yellow,Yes> +

h_1=<Round,Triangle,Round,Purple,Yes>
h_2=<Round,Triangle,Round,Purple,Yes>
h_3=<?,Triangle,Round,?,Yes>
h_4=<?,Triangle,Round,?,Yes>
h_5=<?,?,Round,?,Yes>

Solution : Candidate Elimination

Sl. No	Eyes	Nose	Head	Fcolor	Hair	**Smile ?**
1	R	T	R	P	Y	**Y**
2	S	S	S	G	Y	**N**
3	S	T	R	Y	Y	**Y**
4	R	T	R	G	N	**N**
5	S	S	R	Y	Y	**Y**

S_0 = (ø, ø, ø, ø, ø)

S_1 = (R, T, R, P, Y) from positive example (1)

S_2 , S_1 = (R, T, R, P, Y) No change in S with negative example (2)

S_3 = (?, T, R, ?, Y) from positive example (3)

S_4 , S_3 = (?, T, R, ?, Y) No change in S with negative example (2)

S_5 = (?, ?, R, ?, Y) from positive example (5)

G_5 = (?, ?, R, ?, Y) G is updated by removing inconsistent entry (?, T, ?, ?, Y)

Here S = G , algorithm stops.
Output: The consistent hypothesis that satisfies 'D is (?, ?, R, ?, Y)';
same as Find – S algorithm

G_4 = { (?, T, ? ,? Y), (?, ?, R, ? ,Y)} from negative example (4)

Click Here to derive from Version Space

G_3 = { (?, T, ? ,? ?), (?, ?, R, ? ,?)} G is updated by removing the inconsistent entries {(R, ?, ? ,?, ?), (?, ?, ?, P, ?) }

G_2 = {(R, ?, ? ,?, ?), (?, T, ? ,? ?), (?, ?, R, ? ,?), (?, ?, ?, P, ?) }from negative example (2)

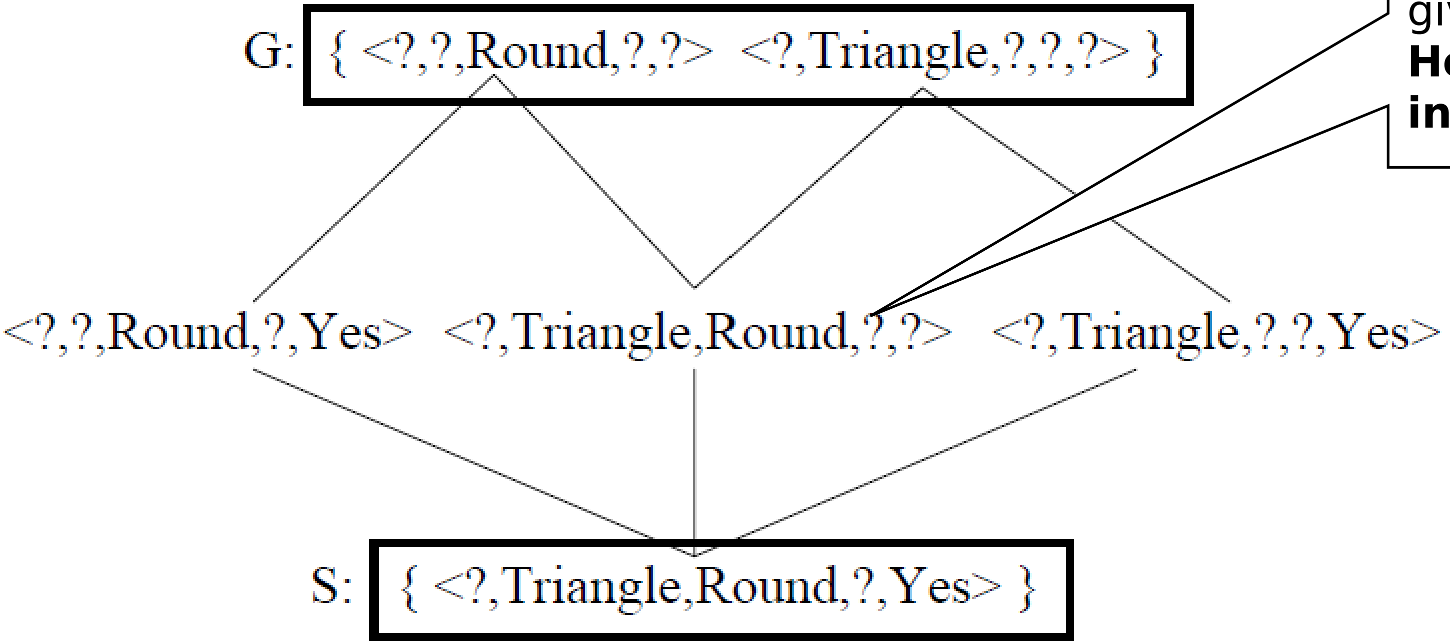
Consider the opposite attribute which is given in negative example.

Take one attribute at a time to make it more specific and write all combinations that are consistent with updated S.

G_1, G_0 = (?, ?, ?, ?, ?) No change in G since it is consistent with S_1

G_0 = (?, ?, ?, ?, ?)

Version Space



G: { <?,?,Round,?,?> <?,Triangle,?,?,?> }

<?,?,Round,?,Yes> <?,Triangle,Round,?,?> <?,Triangle,?,?,Yes>

S: { <?,Triangle,Round,?,Yes> }

From negative example (4) Triangle, Round with any other combination gives negative output. **Hence this is inconsistent**

Click Here :Go Back

Exercise

3. Consider the reverse order of training example given in Q2. Find the consistent hypothesis using Candidate Elimination algorithm.

Solution: The reversed training data set s :

Sl. No	Eyes	Nose	Head	Fcolor	Hair	**Smile ?**
1	S	S	R	Y	Y	**Y**
2	R	T	R	G	N	**N**
3	S	T	R	Y	Y	**Y**
4	S	S	S	G	Y	**N**
5	R	T	R	P	Y	**Y**

= (ø, ø, ø, ø, ø)

$S_{2,'}$ = (S, S, R, Y, Y) from positive example (1)

$S_{4,'}$ S_3 = (S, ?, R, Y, Y) from positive example (3)

S_5 = (?, ?, R, ?, Y) from positive example (5)

= (?, ?, R, ?, Y) G is updated by removing inconsistent entries

= {(S, ?, R ,?, ?), (S, ?, ? ,Y ?), (R, ?, ?, Y ,?), (?, ?, R, Y, ?), (?, ?, ? ,Y Y), (R, ?, ?, ? ,Y), **(?, ?, R, ?, Y)** } from negative example (4)

= { (S, ?, ? ,? ?), (?, ?, ?, Y ,?), (?, ?, ?, ? ,Y)} G is updated by removing inconsistent en

= {(S, ?, ? ,?, ?), (?, S, ? ,? ?), (?, ?, ?, Y ,?), (?, ?, ?, ?, Y) } from negative example (2)

= (?, ?, ?, ?, ?)

Exercise – (Example for non existence of specific - consistent hypothesis)

4. Consider the training data set given below to classify whether a particular disease is identified or not based on the symptoms. Use candidate elimination method to find the consistent hypothesis.

Sl. No	Running nose	Cough	Reddened skin	fever	**Classification**
1	+	+	+	--	**Positive**
2	+	+	--	--	**Positive**
3	--	--	+	+	**Positive**
4	+	--	--	--	**Negative**
5	--	--	--	--	**Negative**
6	--	+	+	--	**Negative**

Solution: $S_0 = (\varnothing, \varnothing, \varnothing, \varnothing)$

= (+,+,+,--) from positive example (1)

= (+,+,?,--) from positive example (2)

$S_3 = (?, ?, ?, ?)$ from positive example (3)

= (?,?,?,?) from positive example (3)

$S_4 = (?, ?, ?, ?)$ Since it is negative example

= (?, ?, ?, ?) Since it is negative example

↕ Inconsistent

$G_4 = (-, ?, ?, ?), (?, +, ?, ?), (?, ?, +, ?), (?, ?, ?, -)$ from negative example (4)

= (--, ?, ?, ?), (?, +, ?, ?), (?, ?, +, ?), (?, ?, ?, --) from negative example (4)

$G_3, G_2, G_1, G_0 = (?, ?, ?, ?)$

= (?, ?, ?, ?)

Since both, S and G = (?,?,?,?) and the remaining example makes S and G inconsistent, the required hypothesis is **S = G = ø,**

Exercise – case where S ≠ G till the end

5. Use CE algorithm to define the version space that satisfies the training example given below

Sl. No	Hair	Body	Regular Visitor	Pose	Smile	Smart	**Class**
1	Blond	Thin	Yes	Arrogant	Toothy	No	**Positive**
2	Brown	Thin	No	Natural	Pleasant	Yes	**Negative**
3	Blond	Plump	Yes	Goofy	Pleasant	No	**Positive**
4	Black	Thin	No	Arrogant	None	No	**Negative**
5	Blond	Plump	No	Natural	Toothy	Yes	**Negative**

Instances

Hair	3
Body	2
Visitor	2
Pose	3
Smile	3
Smart	2

Solution:

Sl. No	Hair	Body	Visitor	Pose	Smile	Smart	Class
1	BLD	T	Y	A	T	N	P
2	BRN	T	N	N	P	Y	N
3	BLD	P	Y	G	P	N	P
4	BLK	T	N	A	N	N	N
5	BLD	P	N	N	T	Y	N

= (ø, ø, ø, ø, ø, ø)

S_2,′ = (BLD, T, Y, A, T, N) from positive example (1)

S_5,′ S_4,′ S_3 = **(BLD, ?, Y, ?, ?, N)** from positive example (3)

G_5 = **(?, ?, Y ,?, ?, ?)** from negative example (5)

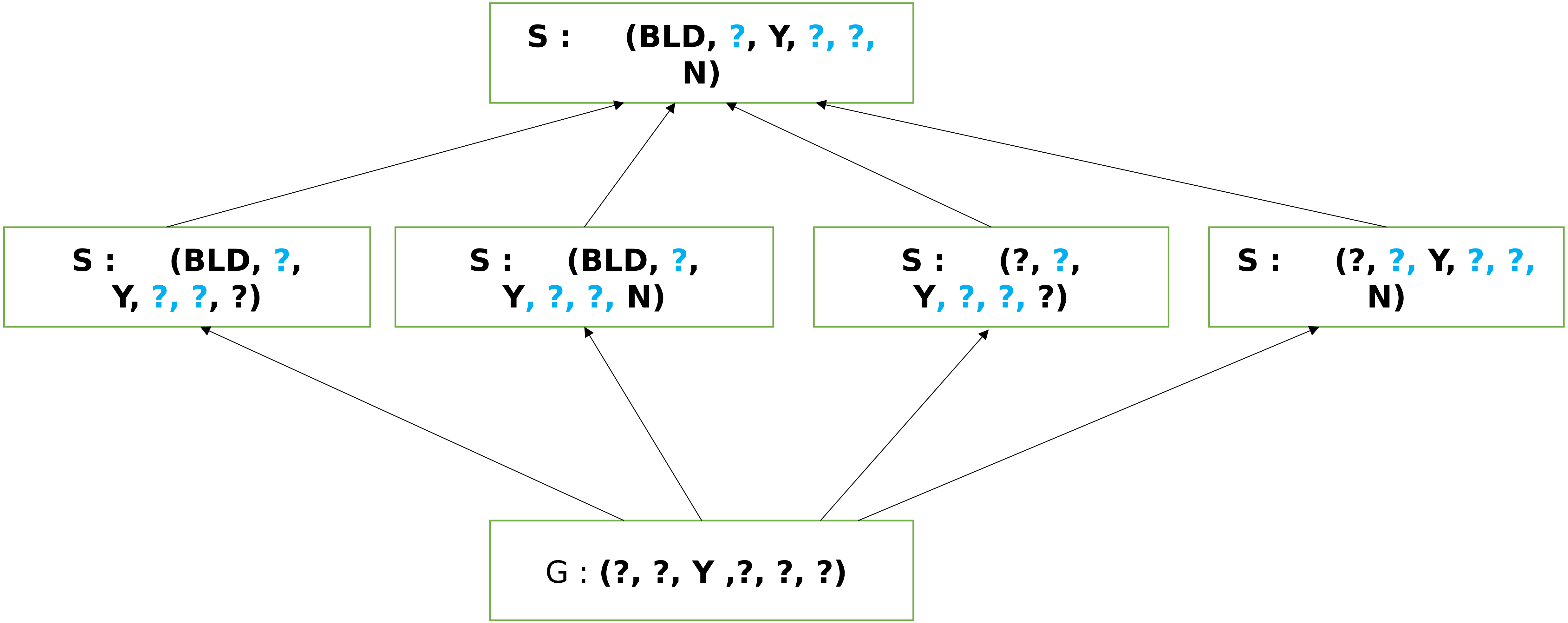
G_4 = { (BLD, ?, ?, ? ,? ?), (?, ?, Y ,?, ?, ?)} from negative example (4)

{ (BLD, ?, ?, ? ,? ?), (?, ?, Y ,?, ?, ?), (?, ?, ?, ?, ? ,N)} G is updated by removing inconsistent entry

G_2 = {(BLD, ?, ?, ? ,?, ?), (?, ?, Y, ? ,? ?), (?, ?, ?, A, ? ,?), (?, ?, ?, ?, T, ?) , (?, ?, ?, ?, ?, N) } from negative example (2)

= (?, ?, ?, ?, ?)

Version Space



S : (BLD, ?, Y, ?, ?, N)

S : (BLD, ?, Y, ?, ?, ?)

S : (BLD, ?, Y, ?, ?, N)

S : (?, ?, Y, ?, ?, ?)

S : (?, ?, Y, ?, ?, N)

G : (?, ?, Y ,?, ?, ?)

Exercise – With first negative training example

6. For the training data set given below, find the consistent hypothesis using CE algorithm

Sl. No	Size	Color	Shape	LABEL
1	Big	Red	Circle	NEGATIVE
2	Small	Red	Triangle	NEGATIVE
3	Small	Red	Circle	POSITIVE
4	Big	Blue	Circle	NEGATIVE
5	Small	Blue	Circle	POSITIVE

Sl. No	Size	Color	Shape	LABEL
1	Big	Red	Circle	NEGATIVE
2	Small	Red	Triangle	NEGATIVE
3	Small	Red	Circle	POSITIVE
4	Big	Blue	Circle	NEGATIVE
5	Small	Blue	Circle	POSITIVE

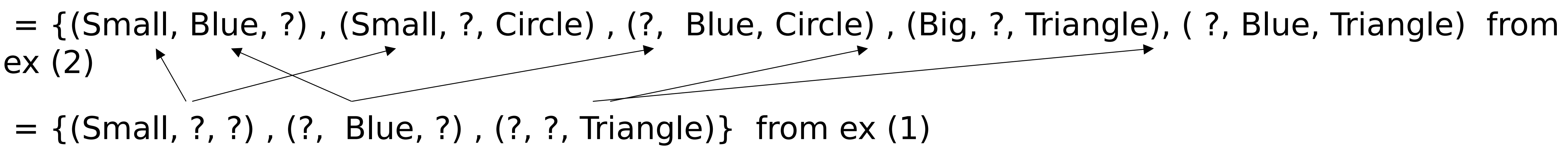
Solution

$S_2,\ S_1\ S_0 = (\varnothing, \varnothing, \varnothing)$ = (ø, ø, ø)

$S_4,\ \ S_3 = (\text{Small, Red, Circle})$ from ex (3)

$S_5 = \{(\text{Small, ?, Circle})\} = G_5$ ➔ Final output

$G_3 = \{(\text{Small, ?, Circle})\}$ Updating G consistent with S



= {(Small, Blue, ?) , (Small, ?, Circle) , (?, Blue, Circle) , (Big, ?, Triangle), (?, Blue, Triangle) from ex (2)

= {(Small, ?, ?) , (?, Blue, ?) , (?, ?, Triangle)} from ex (1)

$G_0 = (?, ?, ?)$