Concept Learning, Version Space learning, Candidate Elimination Algorithm

By

Dr. Sonali Patil

General Terms

- Concept learning: Concept learning is basically learning task of the machine (Learn by Train data)
- General Hypothesis: Not Specifying features to learn the machine.

- G = {'?', '?','?','?'…}: Number of attributes
- Specific Hypothesis: Specifying features to learn machine (Specific feature)
- S= {'pi','pi','pi'…}: Number of pi depends on number of attributes.
- Version Space: It is intermediate of general hypothesis and Specific hypothesis. It not only just written one hypothesis but a set of all possible hypothesis based on training data-set.

Candidate elimination algorithm

Space

obj(X, Y, Z)

Concept

obj(X, Y, ball) obj(X, red, Z) obj(small, Y, Z) obj(small, red, Z)

obj(X, red, ball) obj(small, Y, ball)

obj(small, orange, ball)

obj(small, red, ball)

4. Learning in version space

Generalization operators in version space

- **Replace constants with variables**

  color(ball, red) color(X, red)

- **Remove literals from conjunctions**

  shape(X, round) ∧ size(X, small) ∧ color(X, red) shape(X, round) ∧ color(X, red)

- **Add disjunctions**

  shape(X, round) ∧ size(X, small) ∧ color(X, red)

  shape(X, round) ∧ size(X, small) ∧ (color(X, red) ∨ color(X, blue))

- **Replace an class with the superclass in is-a relations**

is-a(tom, cat) is-a(tom, animal)

Candidate elimination algorithm

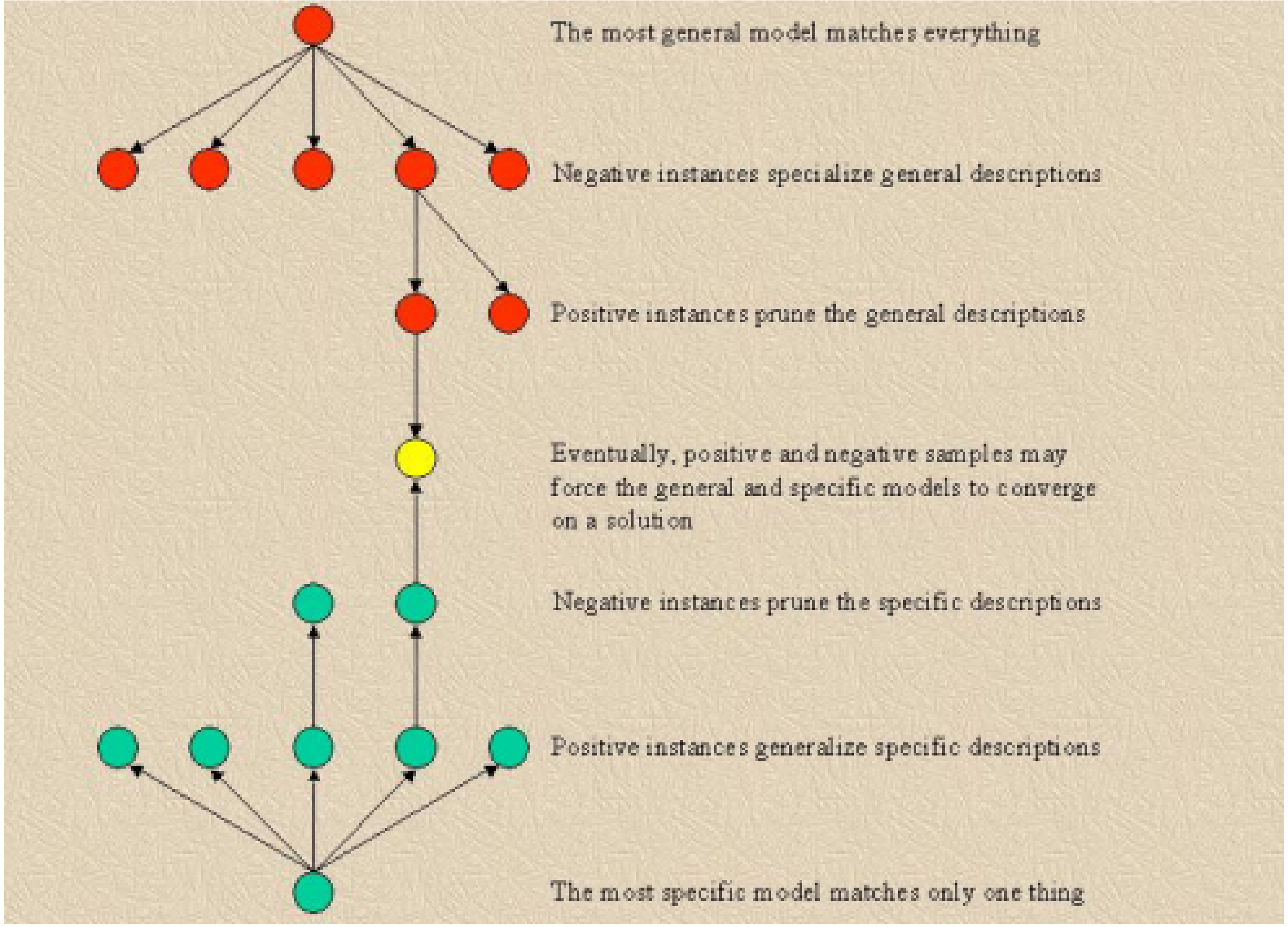
Version space

- **Version space** = the set of all concept descriptions which are consistent with the learning/training examples.
- **What is the idea?** = reduce the version space based on learning examples

- 1 algorithm – from specific to general
- 1 algorithm – from general to specific
- 1 algorithm – bidirectional search = candidate elimination algorithm

5

Candidate elimination algorithm



Version Space Diagram

Generalization and specialization

- Ideally, the learned concept must be general enough to cover all positive examples and also must be specific enough to exclude all negative examples.

- one concept that would cover all sets of exclusively positive instances would simply be obj(X, Y, Z). However, this concept is probably too general, because it implies that all instances belong to the target concept.
- One way to avoid overgeneralization is to generalize as little as possible to cover positive examples; another is to use negative instances to eliminate overly general concepts

Generalization and specialization

Concept induced from positive examples only

Concept induced from positive and negative examples

Figure 10.6 The role of negative examples in preventing overgeneralization.

- negative instances prevent overgeneralization by forcing the learner to specialize concepts in order to exclude negative instances.

Algorithm for searching from specific to general

1. Initialize **S** with the first ex+
2. Initialize **N** with the empty set
3. **for** every learning example **repeat**
   3.1 **if** ex+, ***p***, **then**
   for each **s** ∈ **S** **repeat**
   \- **if** ***s*** does not cover ***p*** **then** replace ***s*** with

the most specific generalization which covers ***p*** - Remove from **S** all hypothesis more general than other hypothesis from **S**

- Remove from **S** all hypothesis which cover an ex- from **N**

3.2 **if** ex-, ***n***, **then**

- Remove from **S** all hypothesis which cover ***n***

- Add ***n*** to **N** (to check for overgeneralization) **end**

Algorithm for searching from specific to general

S: { }

S: { obj(small, red, ball) } S: { obj(small, Y, ball) } S: { obj(X, Y, ball) }

Positive: obj(small, red, ball) Positive: obj(small, white, ball) Positive:

obj(large, blue, ball)

Algorithm for searching from general to specific

1. Initialize **G** with the most general description 2. Initialize **P** with the empty set

3. **for** every learning example **repeat**

3.1 **if** ex-, ***n***, **then**

for each ***g*** ∈ **G** **repeat**

- **if** ***g*** covers ***n*** **then** replace ***g*** with the most general specialization which does not cover ***n*** - Remove from **G** all the hypothesis more specific than other hypothesis in **G**

- Remove from **G** all hypothesis which does not cover the positive examples from **P**

3.2 **if** ex+, ***p***, **then**

- Remove from **G** all the hypothesis that does not cover ***p***

- Add ***p*** to **P** (to check for overspecialization) **end**

Algorithm for searching from general to specific

G: { obj(X, Y, Z) }

G: { obj(large, Y, Z), obj(X, white, Z), obj(X, Y, ball) }

G: { obj(large, Y, Z), obj(X, white, Z),
Negative: obj(small, red, brick)

G: {obj(X, white, Z), obj(X, Y, ball) }

Positive: obj(large, white,

G: obj(X, Y, ball)

ball)

Negative: obj(large, blue, cube) Positive:

obj(X, blue, Z), obj(X, Y, ball), obj(X, Y, cube) } obj(small, blue, ball)

Algorithm for searching in version space

1. Initialize **G** with the most general description **2.** Initialize **S** with the first ex+

3. **for** every learning example **repeat**

3.1 **if** ex+, ***p***, **then**

3.1.1 Remove from **G** all the elements that does not cover ***p***

3.1.2 **for** each ***s*** ∈ **S** **repeat**

- **if** ***s*** does not cover ***p*** **then** replace ***s*** with the most specific generalization which covers ***p*** - Remove from **S** all hypothesis more general than other hypothesis in **S**

- Remove from **S** all hypothesis more general than other hypothesis in **G**

Algorithm for searching in version space - cont

3.2 if ex-, ***n***, **then**

3.2.1 Remove from **S** all the hypothesis that cover ***n***

3.2.2 for each ***g*** ∈ **G** **repeat**

- **if** ***g*** covers ***n*** **then** replace ***g*** with the most general specialization which does not cover ***n*** - Remove from **G** all hypothesis more specific than other hypothesis in **G**

- Remove from **G** all hypthesis more specific than other hypothesis in **S**

4. if G = **S** and card(**S**) = 1 **then** a concept is found

5. if G = **S** = { } **then** there is no concept consistent with all hypothesis

end

Algorithm for searching in version space

G: { obj(X, Y, Z) }
S: { }

G: { obj(X, red, ball) }
S: { obj(X, red, ball) }

Positive: obj(small, red, ball)

G: { obj(X, Y, Z) }
S: { obj(small, red, ball) }

G: { obj(X, red, Z) }
S: { obj(small, red, ball) }

Negative: obj(small, blue, ball) Positive:

G: { obj(X, red, Z) }
S: { obj(X, red, ball) }

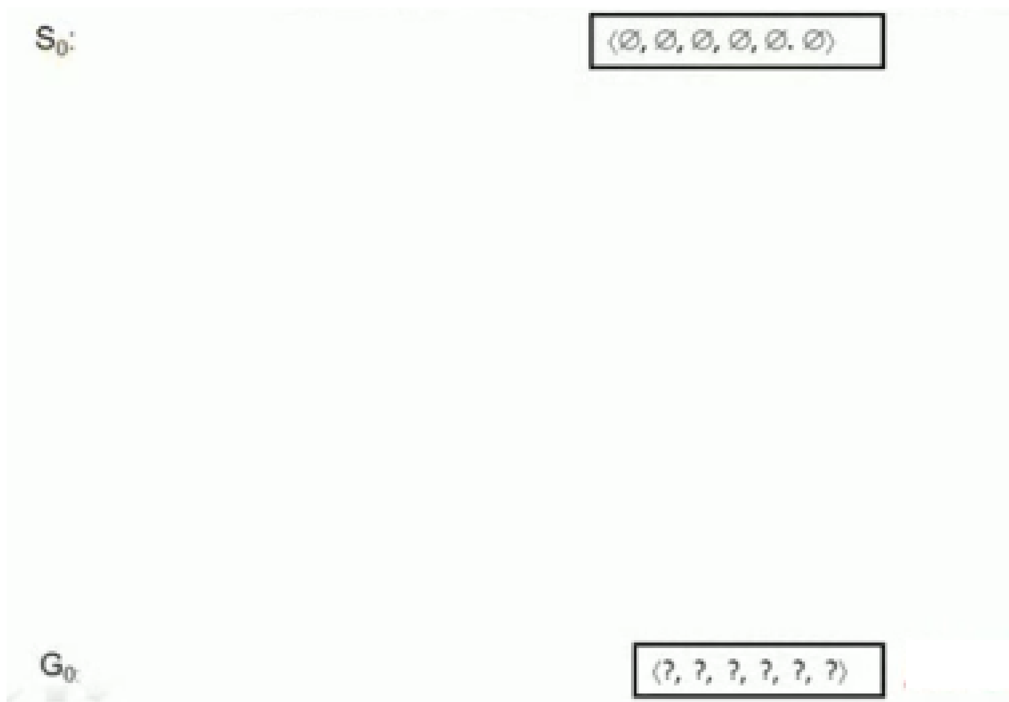
obj(large, red, ball)

Negative: obj(large, red, cube)

17

Example

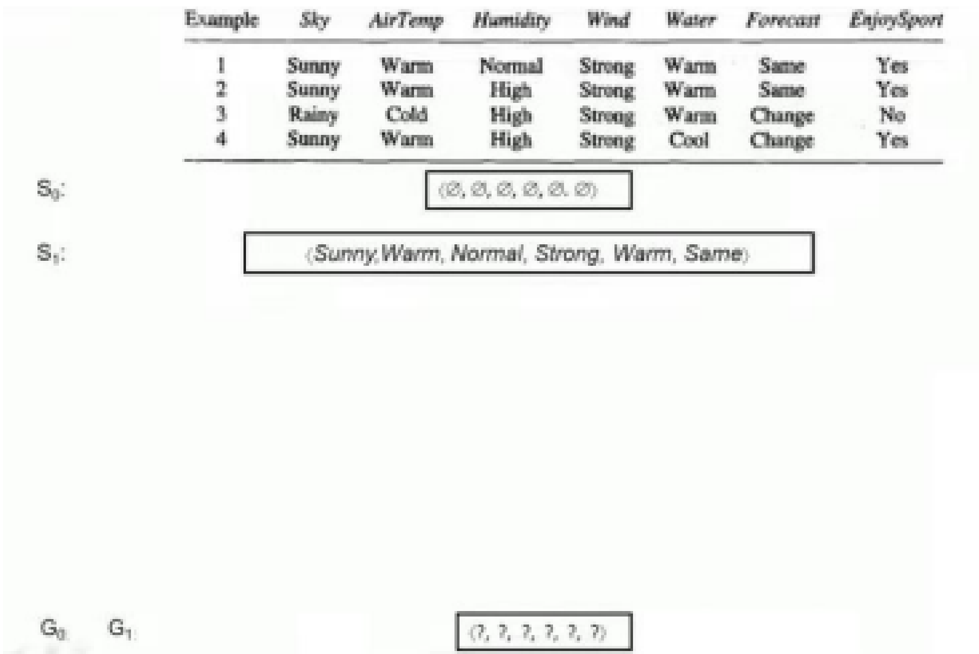
Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rain	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes



- First example is positive, we go to generic boundary and check If the hypothesis at the generic boundary is consistent with the input/training sample or not. If consistent, we will retain the generic hypothesis else we have to write the next general hypothesis

 Compare G0 with first sample , All question marks matches with sample1, hence the classification is positive(yes) which is consistent with the label of the sample1. G1 same as G0

• Now go to specific boundary and check if the hypothesis at the specific boundary is consistent with the input/training sample or not Compare S0 with first sample. S0 has all null. No match hence negative classification which is not consistent with the label of sample1 (positive/yes). Replce null with sample1



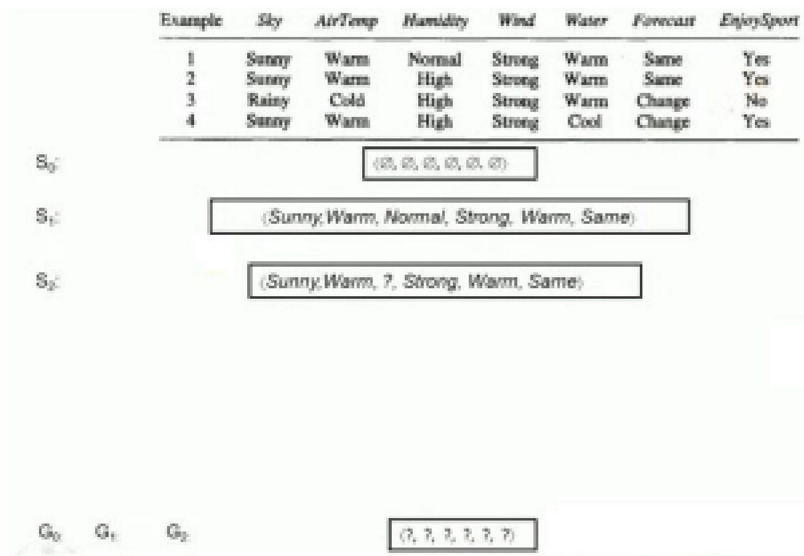
Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

• Second example is positive, we go to generic boundary and check If the hypothesis at the generic boundary is consistent with the input/training sample or not. If consistent, we will retain the generic hypothesis else we have to write the next general hypothesis

- Compare G1 with second sample , All question marks matches with sample2, hence the classification is positive(yes) which is consistent with the label of the sample2. G2 same as G1

- Now go to specific boundary and check if the hypothesis at the specific boundary is consistent with the input/training sample or not

   Compare S1 with second sample. Retain Sunny and warm as they are matching with sample2. Normal not matching with High, negative classification. Expected is positive. Replace Normal with ?. Strong, warm and same matching, hence consistent. Retain



- Third example is negative, we go to specific boundary and check If the hypothesis at the specific boundary is consistent with the input/training sample or not. If consistent, we will retain the specific hypothesis else we have to write the next specific hypothesis

- Compare S2 with third sample , Sunny not matching with rainy, hence negative classification which is consistent with the label of sample3. Retain. S3=S2

- Now go to generic boundary and check if the hypothesis at the specific boundary is consistent with the input/training sample or not. If yes, retain. If No, we will write all hypothesis which are consistent with all the training examples/samples seen till now
  - G2 is all ?s. Matches with sample3. Positive classification. Expected is negative (label of sample3). Not consistent. Hence write all hypothesis which are consistent with sample1, sample2 and sample3. For this consider one ? at a time. Ex first ?. The first attribute is Rainy in sample3. Substitute opposite of Rainy in place of first ? Which is Sunny and rest all ? Will be retain. Now consider second ? And repeat same. Now for all hypothesis formed so, check for consistency with Sample1, sample2 and sample3 (samples seen till now). Retain consistent. Remove inconsistent (red ones)

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

S_0: ⟨∅, ∅, ∅, ∅, ∅, ∅⟩

S_1: ⟨Sunny, Warm, Normal, Strong, Warm, Same⟩

S_2: S_3: ⟨Sunny, Warm, ?, Strong, Warm, Same⟩

G_3: ⟨Sunny,?,?,?,?,?⟩ ⟨?,Warm,?,?,?,?⟩ ⟨?,?,Normal,?,?,?⟩ ⟨?, ?,?,?,Cool,?⟩ ⟨?,?,?,?,?,Same⟩

G_0 G_1 G_2 ⟨?, ?, ?, ?, ?, ?⟩

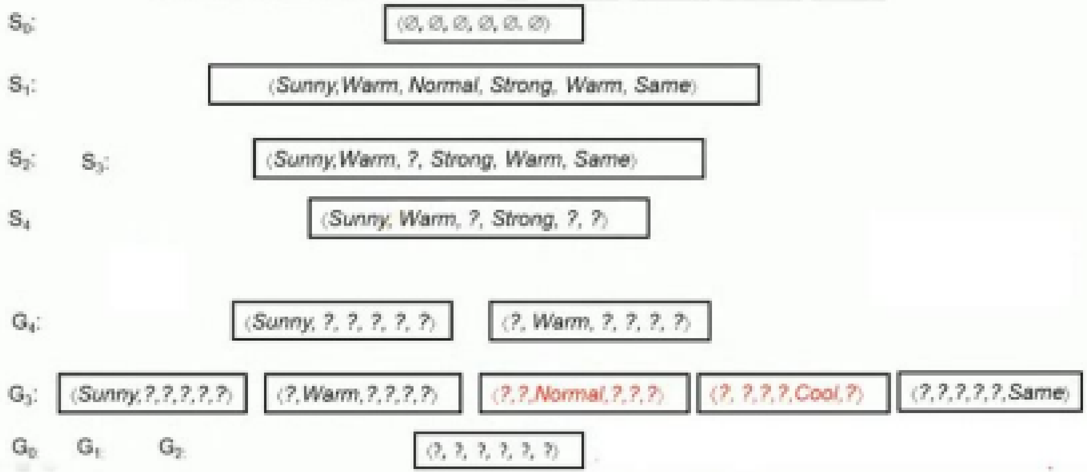
• Fourth example is positive, we go to generic boundary and check If the hypothesis at the generic boundary is consistent with the input/training sample or not. If consistent, we will retain the generic hypothesis else we have to write the next general hypothesis

 Compare all hypothesis in G3 with the training sample4. Retain those ones in G4 which are consistent with Sample4 and remove other inconsistent hypothesis

• Now go to specific boundary and check if the hypothesis at the specific boundary is consistent with the input/training sample or not Compare S3 with sample4. Sunny, warm matches. ? matches with high. Warm not matching with Cool and Same not matching with change. Replace Warm and

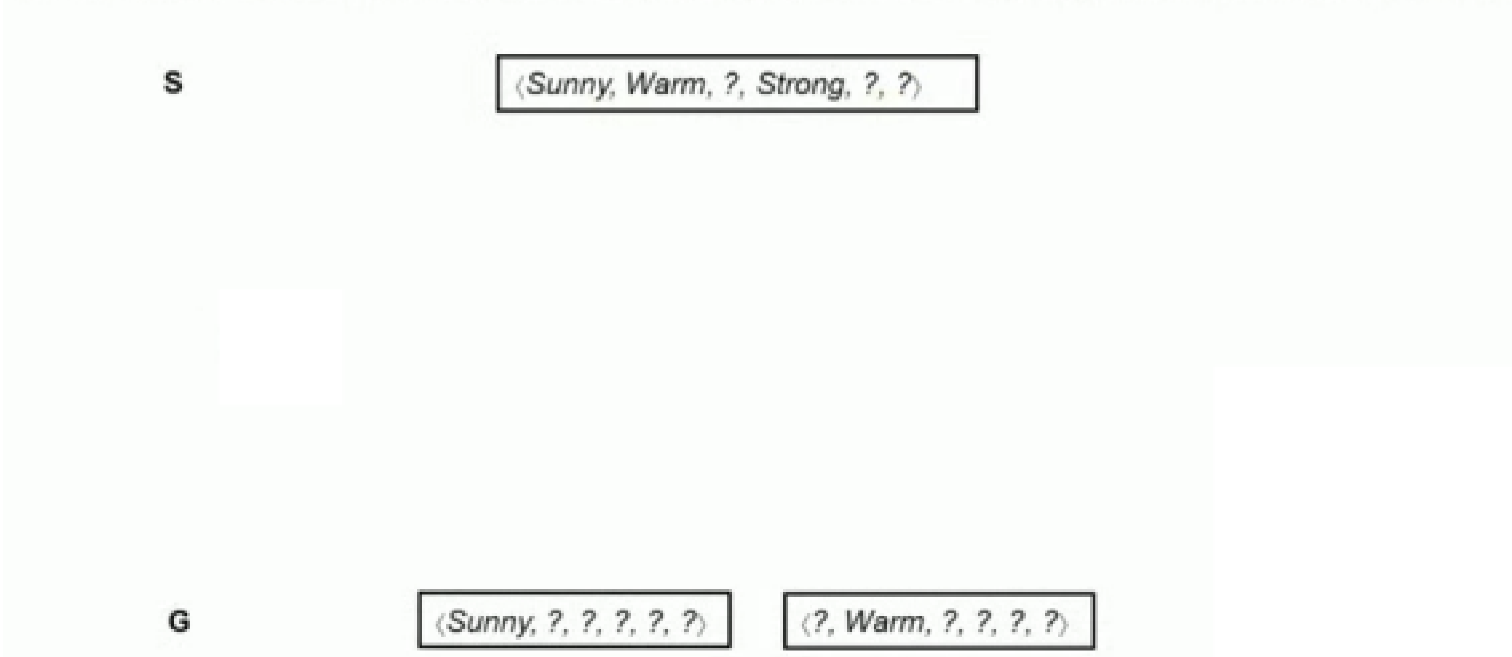
Same in S3 with ? to get S4.

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

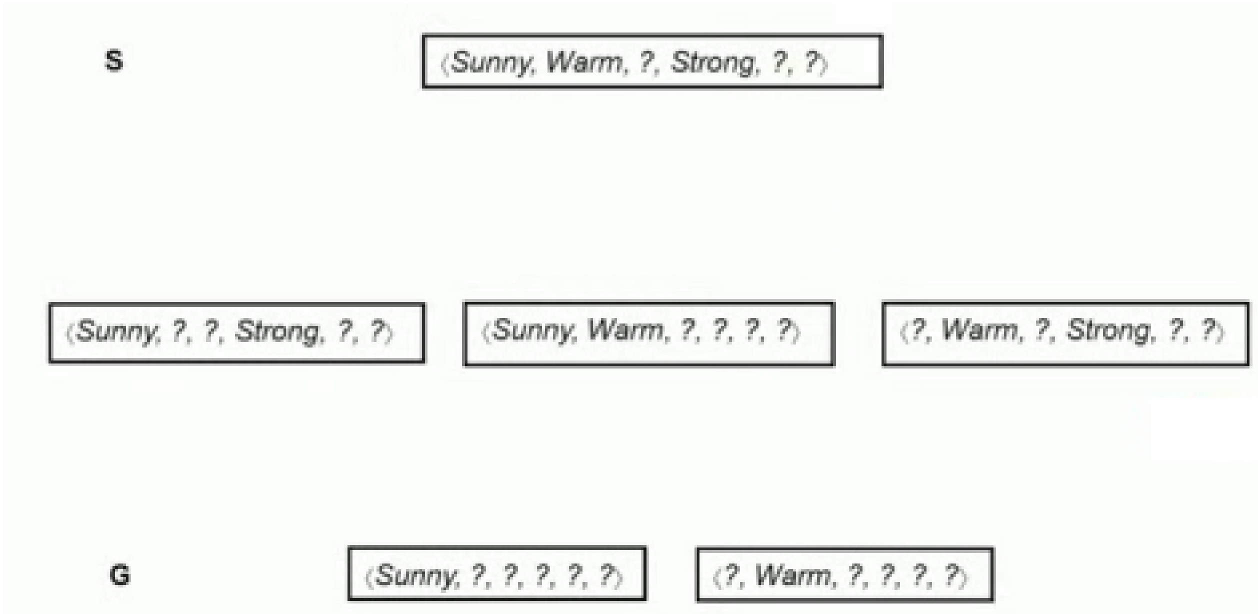


- All training samples are over
- S4 and G4 not same. There are more than one hypothesis. So we need to write few more hypothesis considering S4 and G4. If S4 would have been same as G4, it would have been perfect classification and we would not have written more hypothesis

Learned Version Space by Candidate Elimination Algorithm



- All training samples are over
- S4 and G4 not same. There are more than one hypothesis. So we need to write few more hypothesis considering S4 and G4. If S4 would have been same as G4, it would have been perfect classification and we would not have written more hypothesis. Compare S4 with all hypothesis in G4 one by one. Warm and ? not matching so replace ? By Warm etc. Here consistent hypothesis are 6



Candidate Elimination Algo Ex

Candidate Elimination Algorithm

Size	Color	Shape	Class / Label
Big	Red	Circle	No
Small	Red	Triangle	No
Small	Red	Circle	Yes
Big	Blue	Circle	No
Small	Blue	Circle	Yes

S0: (0, 0, 0)

S1: (0, 0, 0)

S2: (0, 0, 0)

S3: (Small, Red, Circle)

S4: (Small, Red, Circle)

S5: (Small, ?, Circle)

S: G: (Small, ?, Circle)

G5: (Small, ?, Circle)

G4: (Small, ?, Circle)

G3: (Small, ?, Circle)

G2: (Small, Blue, ?) (Small, ?, Circle) (?, Blue, ?) (Big, ?, Triangle) (?, Blue, Triangle)

G1: (Small, ?, ?) (?, Blue, ?) (?, ?, Triangle)

G0: (?, ?, ?)