哈爾濱工業大學深圳
Harbin Institute of Technology Shenzhen

Machine Learning

Chapter 2 Concept Learning and The General-to-specific Ordering

Outline

- Learning from examples
- General-to-specific ordering of hypotheses
- Version spaces and candidate elimination algorithm
- Inductive bias

2.1 Introduction

- Concept
  - Concept can be viewed as describing some subset of objects or events defined over a large set. Or a boolean-valued function defined over this larger set
  - IsBird(animal)

- Concept Learning
  - Inferring a boolean-valued function from training examples of its input and output

2.2 A Concept Learning Task

- An example
  - Target Concept: "days on which my friend Aldo enjoys his favourite water sports"
  - Task: predict the value of " Enjoy Sport " for an arbitrary day based on the values of the other attributes

    EnjoySport(day)

Table2-1Positive and negative training examples for the target concept EnjoySport

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

- Hypothesis
  - Understand Hypothesis
    - Concept, boolean-valued function
  - Hypothesis Representation
    - A conjunction of constraints on attributes (bias)
    - Each constraint can be:
      - A specific value : e.g. *Water=Warm*
      - A don't care value : e.g. *Water=?*
      - No value allowed (null hypothesis): e.g. *Water=Ø*

– Hypothesis Example: h

Sky	Temp	Humid	Wind	Water	Forecast
< Sunny	?	?	Strong	?	Same >

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

// every day is a positive example

< ϕ, ϕ, ϕ, ϕ, ϕ, ϕ>

// no day is a positive example

Description of The EnjoySport Concept Learning Task

Given:

- Instances X : Possible days decribed by the attributes *Sky, Temp, Humidity, Wind, Water, Forecast*
- Hypotheses H: each h is a conjunction of literals e.g.

  $<$ Sunny ? ? Strong ? Same $>$
- Training examples D : positive and negative examples of the target function: $<x_1,c(x_1)>,\ldots, <x_n,c(x_n)>$
- Target concept c: EnjoySport $X \rightarrow \{0,1\}$

Determine:

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

2.2.1 Notation

- Instance: x
- The set of instances: X
- Target concept: c
- Training examples: x
- The set of training examples: D
- Positive examples: $c(x)=1$
- Negative examples: $c(x)=0$
- Hypothesis: h
- The set of all possible hypotheses: H

The goal : find a hypothesis h such that $h(x)=c(x)$ for all x in X

2.2.2 The Inductive Learning Hypothesis

- Inductive Learning
  - Get common rules from examples
  - Only guarantee that the output hypothesis fits the target concept over the training data

- Fundamental Assumption of Inductive Learning
  - Any hypothesis found to approximate the target function well over the training examples, will also approximate the target function well over the unobserved examples.

2.3 Concept Learning as Search

- Concept Learning can be viewed as a searching problem
  - Searching Space： hypotheses space defined by hypothesis representation
  - Goal： find the hypothesis that best fits the training examples

- Understanding the Relations among Hypothesis representation , Hypothesis Space, and Program
  - Hypothesis Space for EnjoySport
    - Sky: Sunny, Cloudy, Rainy
    - AirTemp: Warm, Cold
    - Humidity: Normal, High
    - Wind: Strong, Weak
    - Water: Warm, Cold
    - Forecast: Same, Change

#distinct instances : 3*2*2*2*2*2 = 96

#distinct concepts defined on instance space: 2^{96}

#syntactically distinct hypotheses :
5*4*4*4*4*4=5120

#semantically distinct hypotheses :
1+4*3*3*3*3*3=973

2.3.1 General-to-Specific Ordering of Hypotheses

- Consider two hypotheses:
  - h_1=< Sunny,?,?,Strong,?,?>
  - h_2=< Sunny,?,?,?,?,?>
  - Set of instances classified positive by h_1 and h_2
    - h_2 imposes fewer constraints than h_1 and therefore classifies more instances x as positive h(x)=1

- Definition of more_general_than_or_equal_to
  - Satisfy:
    - For any x and h, we say x satisfies h if and only if h(x)=1
    - x=(sunny, cold, high, weak, cold, change) satisfies h_2=< Sunny,?,?,?,?,?>
  - 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 if and only if

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

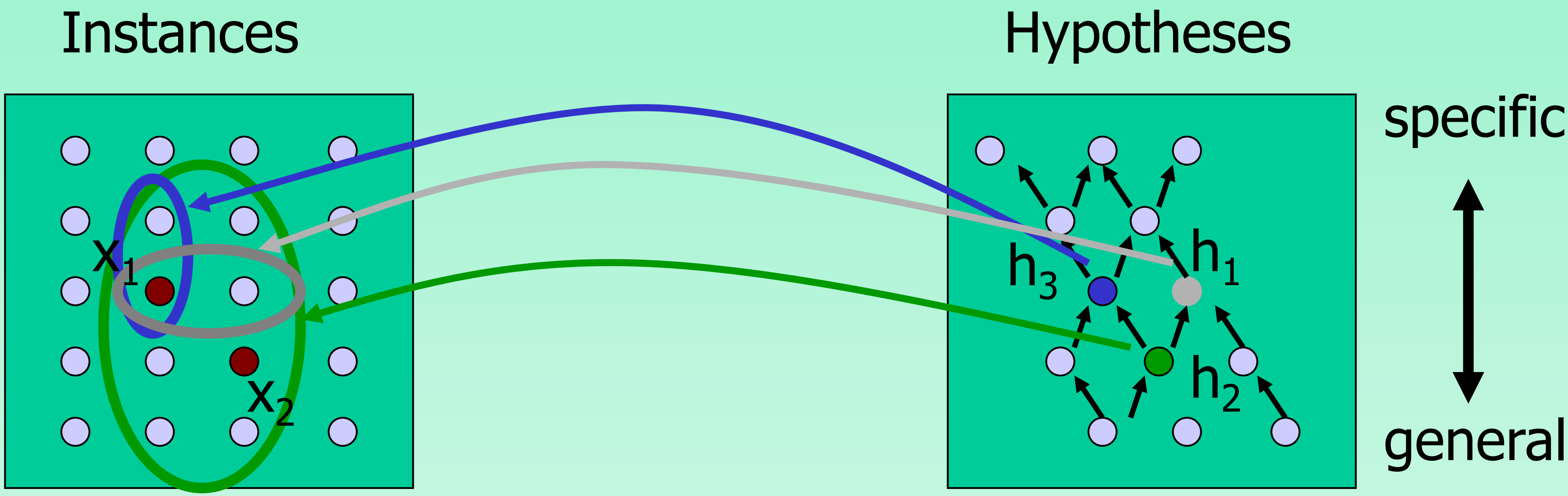
    - Written h_j more_general_than_or_equal_to h_k， or $h_j \geq_g h_k$

- The relation ≥ imposes a partial order (偏序：自反，反对称，传递) over the hypothesis space H

- Strictly more_general_than
  - $h_j >_g h_k$， if and only if， $(h_j \geq_g h_k) \wedge \neg (h_k \geq_g h_j)$

- More_specific_than
  - $h_j \leq_g h_k$， if and only if， $h_k \geq_g h_j$

more_general_than example



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,?>

$$h_2 \geq h_1$$
$$h_2 \geq h_3$$

2.4 Find-S： Finding a Maximally Specific Hypothesis

- Introduction
  - Use the more_general_than partial ordering to organize the search for a hypothesis
  - Begin with the most specific possible hypothesis in H, and then generalize this hypothesis each time it fails to cover an observed positive training example

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

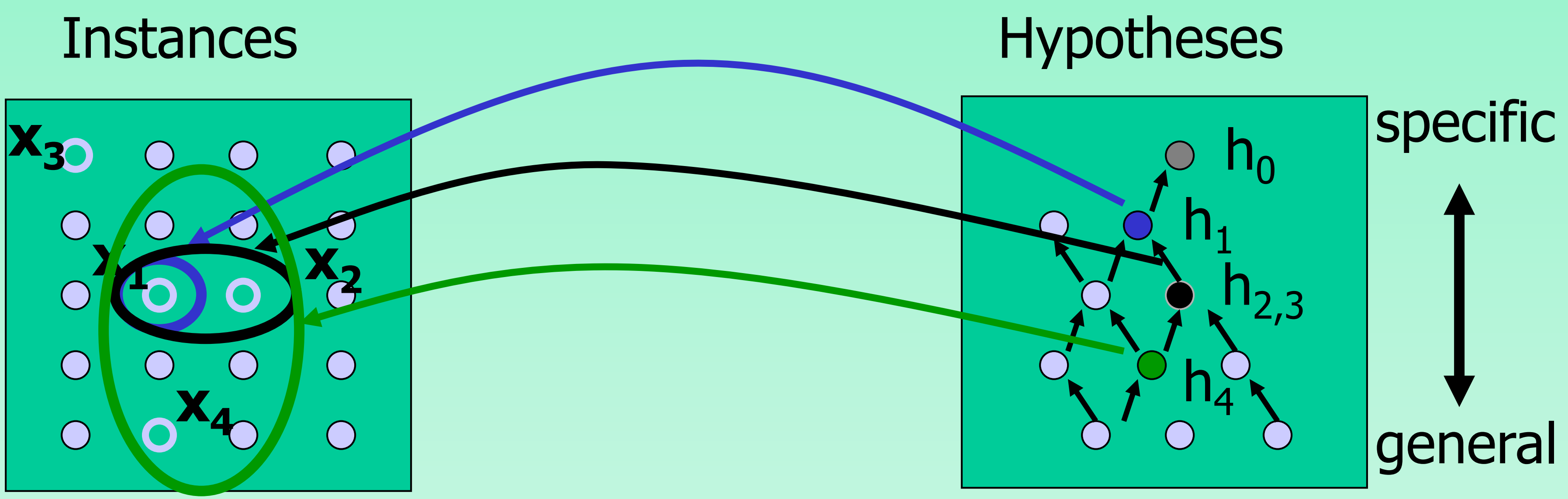
     If the constraint a_i in h is satisfied by x

     then do nothing

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

- For negative example， makes no change to h
  Why?

Hypothesis Space Search by Find-S



h_0=< Ø, Ø, Ø, Ø, Ø, Ø,>

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

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

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

$h_{2,3}$=< Sunny,Warm,?, Strong,Warm,Same>

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

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

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

Properties of Find-S

- Hypothesis space described by conjunctions of attributes

- Find-S will output the most specific hypothesis that is consistent with the positve training examples

- The output hypothesis will also be consistent with the negative examples, provided the target concept is contained in H and trainning samples are correct.

Complaints about Find-S

- Can't tell if the learner has converged to the target concept
  - in the sense that it is unable to determine whether it has found the *only* hypothesis consistent with the training examples, or there are still other hypthotheis
- Can't tell if training examples are consistent, as it ignores negative training examples.
- Why prefer the most specific hypothesis?
- What if there are multiple maximally specific hypothesis?

2.5 Version Spaces and the Candidate-Elimination Algorithm

- Candidate-Elimination VS Find-S
  - Find-S only output one of many hypotheses from H that might fit the training data
  - candidate-elimination output a description of the set of all hypotheses consistent with the training examples
  - candidate-elimination computes the description of this set without explicitly enumerating all of its members
    - This is accomplished by using the more_general_than partial ordering
  - Find-S and candidate-elimination are limited by noisy training data

2.5.1 Representation

- Definition of Consistent
  - A hypothesis h is **consistent** with a set of training examples D if and only if h(x)=c(x) for each example <x,c(x)> in D.

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

  - Difference between consistent and satifies
- Version Space
  - Represent the set of all hypotheses consistent with the training data
  - Contains all plausible(貌似正确) versions of the target concept
  - Definition:
    - The **version space**, $VS_{H,D}$, with respect to hypothesis space H, and training set D, is the subset of hypotheses from H consistent with all training examples:

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

2.5.2 The List-Then-Eliminate Algorithm

- List-Then Eliminate Algorithm

  List all of its members to represent the VS

  1. *VersionSpace* ← a list containing every hypothesis in H
  2. For each training example <x,c(x)>

     remove from *VersionSpace* any hypothesis that is inconsistent with the training example $h(x) \neq c(x)$
  3. Output the list of hypotheses in *VersionSpace*
- advantages
  - Guarantee to output all hypotheses consistent with training data
- disadvantages
  - Requires exhaustively enumerating all hypotheses in H---an unrealistic requirement

2.5.3 A More Compact Representation for Version Spaces

- A More Compact Representation
  - VS is represented by its most general and least general members
  - These members form general and specific boundary sets that delimit the VS
  - EnjoySport Sample

Version Space Example

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

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

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

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> +

- The **general boundary**, G, With respect to H and D is the set of maximally general members of H consistent with D.

$G = \{g \in H \mid \text{Consist}(g,D) \text{ and } (\neg\exists\, g' \in H)\, [(\exists\, g' > g) \text{ and } \text{Consist}(g',D)]\}$

- The **specific boundary**, S, With respect to H and D is the set of maximally specific members of H consistent with D.

$S = \{s \in H \mid \text{Consist}(s,D) \text{ and } (\neg\exists\, s' \in H)\, [(\exists\, s > s') \text{ and } \text{Consist}(s',D)]\}$

- Version Space representation theorem
  - Every member of the version space lies between these boundaries

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

    where $x \geq y$ means x is more general or equal than y
  - (proof)
    - (1) every h satisfying the right side is in $VS_{H,D}$
    - (2) every member of $VS_{H,D}$ satisfies the right side

2.5.4 Candidate-Elimination Learning Algorithm

G ← maximally general hypotheses in H

S ← maximally specific hypotheses in H

For each training example d=<x,c(x)>

If d is a positive example

Remove from G any hypothesis that is 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 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

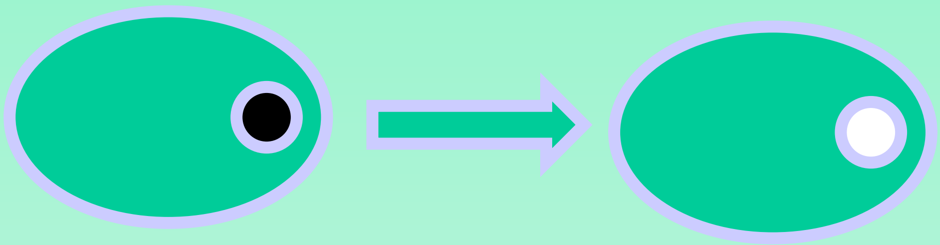
If d is a negative example

Remove from S any hypothesis that is inconsistent with d

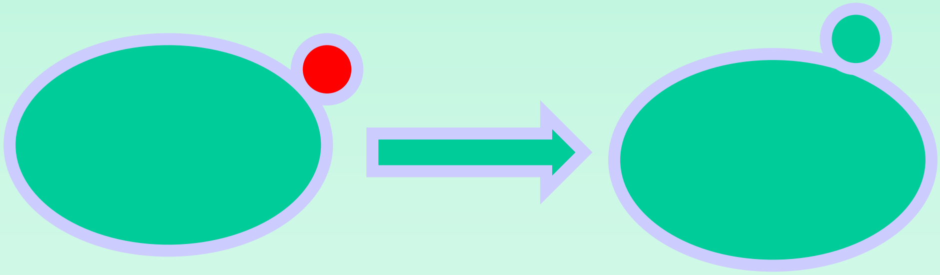
For each hypothesis g in G that is not consistent with d

- remove g from G.
- Add to G all minimal specializations h of g such that
  - h 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

- minimal specializations for g



- minimal generalizations for s



2.5.5 Example Candidate Elimination

S: ~~{<∅, ∅, ∅, ∅, ∅, ∅ >}~~

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

x_1 = <Sunny Warm Normal Strong Warm Same> +

S: ~~{< Sunny Warm Normal Strong Warm Same >}~~

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

x_2 = <Sunny Warm High Strong Warm Same> +

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

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

Example Candidate Elimination

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

G: ~~{<?, ?, ?, ?, ?, ?>}~~

x_3 = <Rainy Cold High Strong Warm Change> -

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

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

x_4 = <Sunny Warm High Strong Cool Change> +

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

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

2.6 Remarks On VS and C-E

- Will the C-E Converge to the Correct Hypothesis?
  - No error in the training examples
  - There is some h in H that correctly describes the target concept
- IF the training data contains errors
  - Given sufficient additional training data the learner will detect the inconsistency by noticing that the S and G boundary sets converge to an empty VS
- The target concept can’t be described in the hypothesis representation
  - A similar symptom

- What Training Example Should the Learner Request Next?
  - The optimal query strategy is to generate instances that satisfy exactly half the hypotheses in the current VS. So the concept can be found with only $\log_2|VS|$

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

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

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

x_1 = <Sunny Warm Normal Light Warm Same>

- How Can Partially Learned Concepts Be Used?
  - the target concept has not yet been fully learned, it is possible to classify certain examples with the some degree of confidence
  - Sample 2.6

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 >2+4-

2.7 Inductive Bias

- Problems about C-E
  - What if the target concept is not contained in H
  - Can we avoid this difficult by using a full H
  - How does the size of H influence the ability of the algorithm to generalize to unobserved instances
  - How does the size of the H influence the number of training examples

2.7.1 A Biased Hypothesis Space

- Our hypothesis space is unable to represent a simple disjunctive target concept :

  (Sky=Sunny) v (Sky=Cloudy)

x_1 = <Sunny Warm Normal Strong Cool Change> +
x_2 = <Cloudy Warm Normal Strong Cool Change> +

S : { <?, Warm, Normal, Strong, Cool, Change> }

x_3 = <Rainy Warm Normal Strong Cool Change > -

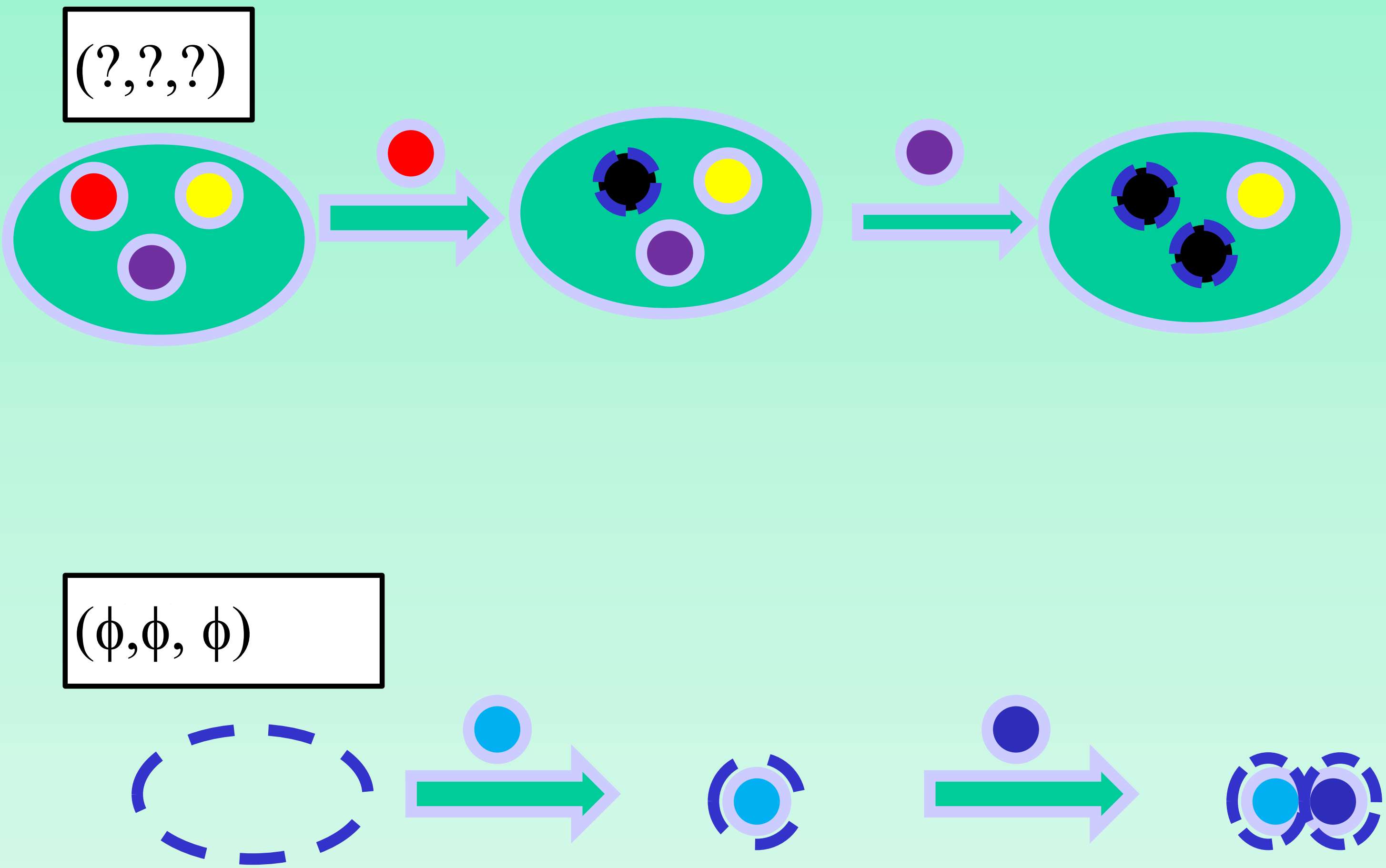
S : {}

2.7.2 An Unbiased Learner

- Idea: Choose H that expresses every teachable concept, that means H is the set of all possible subsets of X called the power set P(X)
  - $|X|=96$, $|P(X)|=2^{96} \sim 10^{28}$ distinct concepts
  - H = disjunctions, conjunctions, negations
  - e.g. <Sunny Warm Normal ? ? ?> or <Sunny or cloudy ? ? ? ? Change>
  - H surely contains the target concept.

- What are S and G in this case?
  - Assume positive examples (x_1, x_2, x_3) and negative examples (x_4, x_5)
  - S : { $<x_1 \vee x_2 \vee x_3>$ } , G : { $\neg <x_4 \vee x_5>$ }
  - The only examples that are classified are the training examples themselves. (why?)
    - Each unobserved instance will be classified positive by precisely half the hypothesis in VS and negative by the other half.
    - In order to learn the target concept, one would have to present every single instance in X as a training example.

Bias-free Learning



- EnjoySport in an unbiased way
  - Unable to generalize beyond the observed examples
  - In order to converge to a single ,final target concept, have to present every single instance in X as a training example

2.7.3 The Futility of Bias-Free Learning

- A fundamental property of inductive inference:

  a learner that makes no a priori assumption regarding the identity of the target concept has no rational（合理）basis for classifying any unseen instances

- Prior assumption required by inductive learning is the inductive bias

- Definition of The inductive bias :

Consider:

– Concept learning algorithm L
– Instances X, target concept c
– Training examples $D_c=\{<x,c(x)>\}$
– Let $L(x_i,D_c)$ denote the classification assigned to instance x_i.

Definition:

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

$$(\forall x_i \in X)[B \wedge D_c \wedge x_i] \mid\!\!-\!\!- L(x_i, D_c)$$

Where y |-- z means that z follows deductively from y.

Inductive Systems and Equivalent Deductive Systems

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 target concept" →

theorem prover

→ classification of new instance or don't know

- Inductive bias of candidate-elimination algorithm
  - $\{c \in H\}$(why?)
- Three Learners with Different Biases
  - Rote learner: Store examples classify x if and only if it matches a previously observed example.
    - No inductive bias
  - Version space candidate elimination algorithm.
    - Bias: The hypothesis space contains the target concept.
  - Find-S
    - Bias: The hypothesis space contains the target concept and all instances are negative instances unless the opposite is entailed by its other knowledge.

- More strongly biased methods make more inductive leaps, classifying a greater proportion of unseen instance

- Different forms of inductive bias:
  - Categorical assumptions that completely rule out (排除…的可能性) certain concepts
  - Rank order the hypothesis by stating preferences
  - Implicit in the learner and unchangeable by the learner