Concept Learning
in
Machine Learning

Dr. Kuppusamy .P

Associate Professor / SCOPE

Learning - Introduction

- Learning involves acquiring general concepts from specific training samples.

E.g.

- People continuously learn general concepts or categories such as "bird," "car," "situations in which I should study more in order to pass the exam," etc.
- Each such concept can be viewed as describing some subset of objects or events defined over a larger set (e.g., the subset of animals that constitute birds)

Hypothesis

- Taking a sample data from the population
- Null hypothesis is defined as the conclusion based on sample data
- Alternate hypothesis- against null (prove based on the complete data)
- Researchers need to prove alternate hypothesis
- **E.g.,** Son – fever – caused by ice-cream (Null Hypothesis)
- Dad says no it is not due to ice-cream, it might be due to various reasons (alternate hypothesis)
- Various reasons for the fever would be taken and analyze it to prove that alternate hypothesis to be accepted.

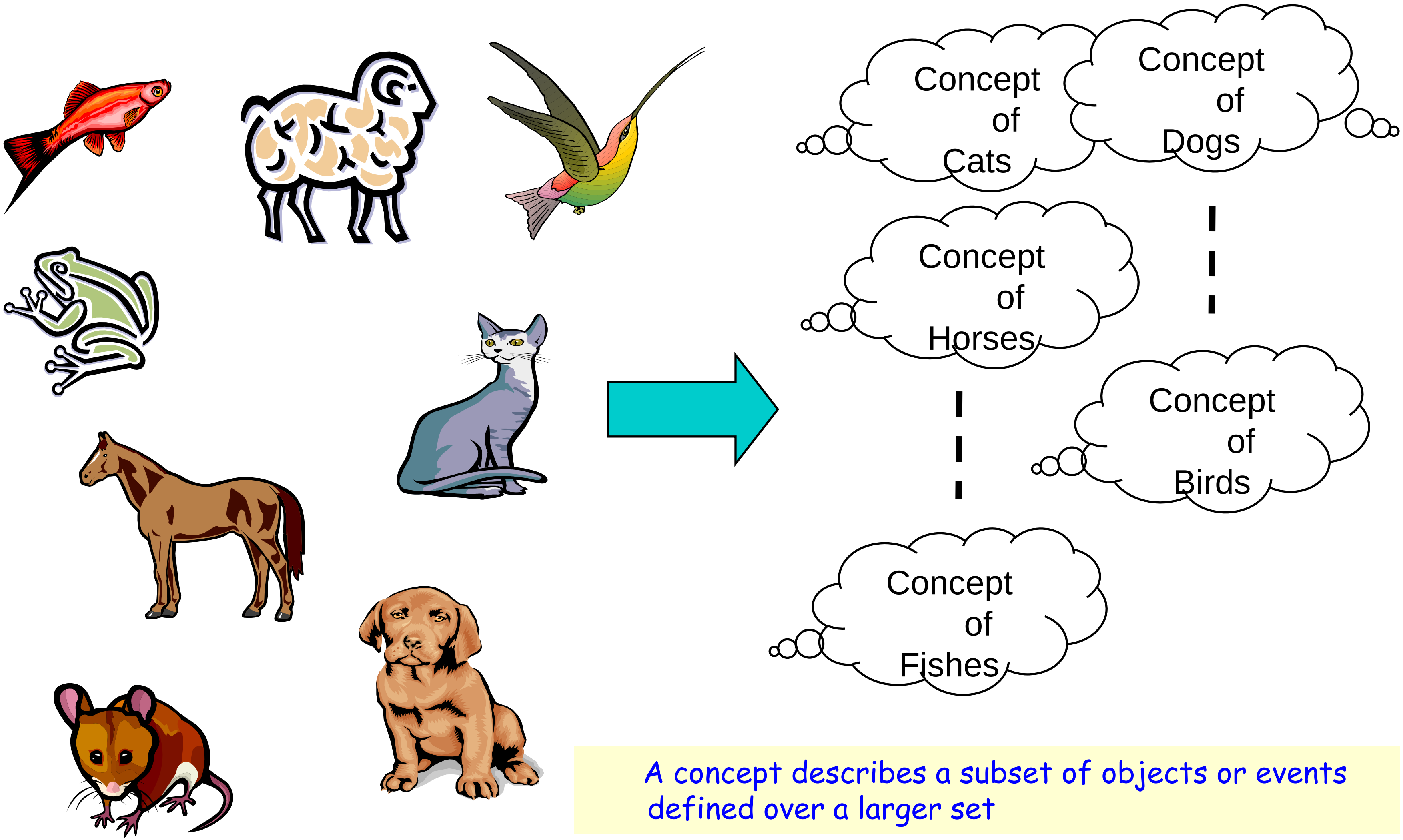
Hypothesis Space and Hypothesis Constraints

- Set of hypothesis from the given training examples.

Constraints

- Hypothesis shown as a vector of some constraints on the attributes.
- Hypothesis consists of 3 constraints.
  - ➢ Totally free of constraint -? (Constraint does not have an important role)
  - ➢ Strong constraint on the value of the attribute: specified by the exact required value for that particular attribute
  - ➢ Strong constraint regardless of the value of the attribute

What is a Concept ?



A concept describes a subset of objects or events defined over a larger set

Concept Learning - learning based on symbolic representations

- Acquire/Infer the general definition of a concept from given a (labeled) examples as positive (members) and negative (non-members) category is Concept Learning.
- Concept Learning represent as Approximate Boolean-valued function from training examples.
- Each concept can be considered as a **Boolean-valued** (true/false or yes/no) function defined over larger set(e.g., a function defined over all animals, whose value is true for birds and false for other animals)

  - Concept learning can be formulated as a problem of searching through a predefined potential hypotheses space for the hypothesis that best fits the training examples

  - Take advantage of a naturally occurring structure over the hypothesis space
    - General-to-specific ordering of hypotheses

Training Examples for *EnjoySport*

- Concept to be learned
  - "Days on which enjoys favorite water sport"

Attributes

Days

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

Concept to be learned

- Days (examples/instances) are represented by a set of attributes.

- What is the general concept ?
  - The task is to learn to predict the value of *EnjoySport* for an arbitrary day based on the values of other attributes
  - Learn a (a set of) hypothesis representation(s) for the concept

Representing Hypotheses

- Many representations possible for hypotheses *h* that contains conjunction of constraints on attributes.
- For Each attribute, hypothesis can either
  - indicate by a "?' (any value is acceptable for this attribute),
  - specify a single required value (e.g., Warm) for the attribute, or
  - indicate by a " Ø" that no value is acceptable (e.g."*Water=Ø* ").
    - Sky AirTemp Humid Wind Water Forecast

      < *Sunny* ? ? *Strong* ? *Same* >

- Most general hypothesis

  < ? ? ? ? ? ? > All are positive examples
- Most specific hypothesis

  < Ø Ø Ø Ø Ø Ø > All are negative examples

Concept Learning Task using Notations

- Given
  - Instances X (set all possible days): each day is represented by 6 attributes
    Sky, *AirTemp*, *Humidity*, *Wind*, *Water*, *Forecast*

    (*Sunny, Cloudy, Rainy*) (*Warm, Cold*) (*Normal, High*) (*Strong, Week*) (*Warm, Cool*) (*Same, Change*)
  - Target concept/function c : *EnjoySport* $X \rightarrow \{0, \ 1\}$

    "No" "Yes"
  - Hypotheses *H* : Conjunctions of Literals. E.g.,

    <?,*Cold*, *High*, ?, ?, ? >
  - Training examples *D* : Positive and negative examples (members/nonmembers) of the target function (concept)

    $<x_1,c(x_1)>, <x_2,c(x_2)>, \ldots, <x_m,c(x_m)>$

    target concept value
- Determine
  - A hypothesis *h* in *H* (an approximate target function) such that
    $h(x)=c(x)$ for all x in D

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
  - Assumption of Inductive Learning
    - The best hypothesis regarding the unseen instances is the hypothesis that best fits the observed training data

Viewing Learning as a Search Problem

- Concept learning can be viewed as the task of searching through a large space of hypotheses.
- Goal: This search finds the hypothesis that best fits the training examples

<u>Instance space *X*</u>

Sky (*Sunny/Cloudy/Rainy*)
AirTemp (*Warm/Cold*)
Humidity (*Normal/High*)
Wind (*Strong/Weak*)
Water (*Warm/Cool*)
Forecast (*Same/Change*)

=> 3*2*2*2*2*2=96 instances

<u>Hypothesis space *H*</u>

Syntactically distinct hypotheses
= 5*4*4*4*4*4=5120
Semantically distinct hypotheses
=1+4*3*3*3*3*3=973

Viewing Learning as a Search Problem

Hypothesis space *H*

Syntactically distinct hypotheses = 5*4*4*4*4*4=5120

- (Includes Two more values for attributes: ? and 0)
- Every hypothesis containing one or more 0 symbols represents the empty set of instances; that is, it classifies every instance as negative.

Semantically distinct Hypotheses values (Only one more value for attributes: ?, and one hypothesis representing empty set of instances)

- Semantically distinct hypotheses =1(empty set Ø) + 4*3*3*3*3*3

  = 973

Each hypothesis is represented as a conjunction of constraints

E.g.,

< Ø Warm Normal Strong Cool Same >

< Sunny Ø Normal Strong Cool Change>

Viewing Learning As a Search Problem

- Study of learning algorithms that examine different strategies for searching the hypothesis space, e.g.,
  - *Find-S* Algorithm
  - *List-Then-Eliminate* Algorithm
  - *Candidate Elimination* Algorithm

- How to exploit the naturally occurring structure in the hypothesis space ?
  - Relations among hypotheses ,
  - e.g.,
    - General-to-Specific-Ordering

General-to-Specific-Ordering of Hypothesis

- Many concept learning algorithms organize the search through the hypothesis space by taking advantage of a naturally occurring structure over it
  - *"general-to-specific ordering"*

$h_1 = <Sunny, ?, ?, Strong, ?, ?>$

$h_2 = <Sunny, ?, ?, ?, ?, ?>$

Suppose that h_1 and h_2 classify positive examples

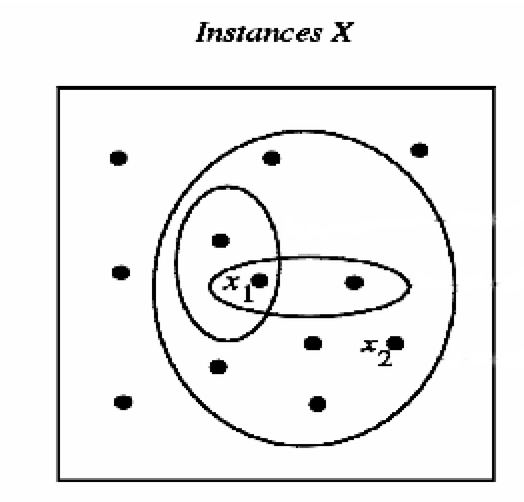
- *h_2 is more general than h_1*
  - h_2 imposes fewer constraints on instances
  - h_2 classify more positive instances than h_1

More-General-Than Partial Ordering

- Definition
  - Let h_j and h_k be Boolean-valued functions defined over X. Then h_j is *more general than* h_k ($h_j >_g h_k$) if and only if

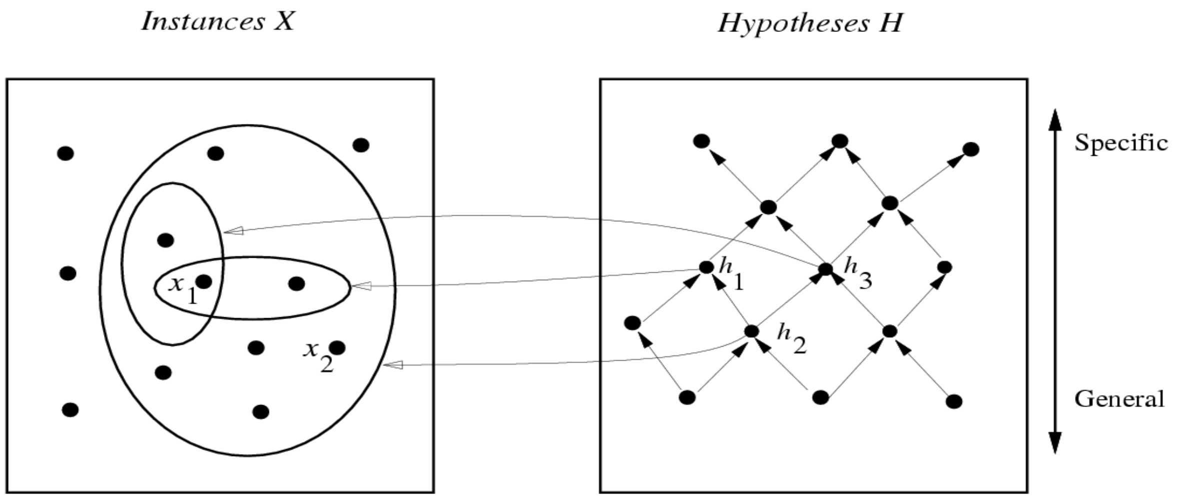
$$(\forall x \in X)\ [(h_k(x)=1) \rightarrow (h_j(x)=1)]$$

x satisfies h_k



- We also can define the *more-specific-than* ordering

General-to-Specific Ordering of Hypotheses



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, ?>

- Suppose instances are classified positive by h_1, h_2 , h_3
  - h_2 (imposing fewer constraints) are *more general than* h_1 and h_3
  - $h_1 \overset{?}{\longleftrightarrow} h_3$

partial order relation
- antisymmetric, transitive

$$h_a \geq_g h_b,\ h_b \geq_g h_c \Rightarrow h_a \geq_g h_c$$

Find-S Algorithm

- Find a maximally specific hypothesis by using the *more-general-than* partial ordering to organize the search for a hypothesis consistent with the observed training examples

$$h \leftarrow \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$$

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

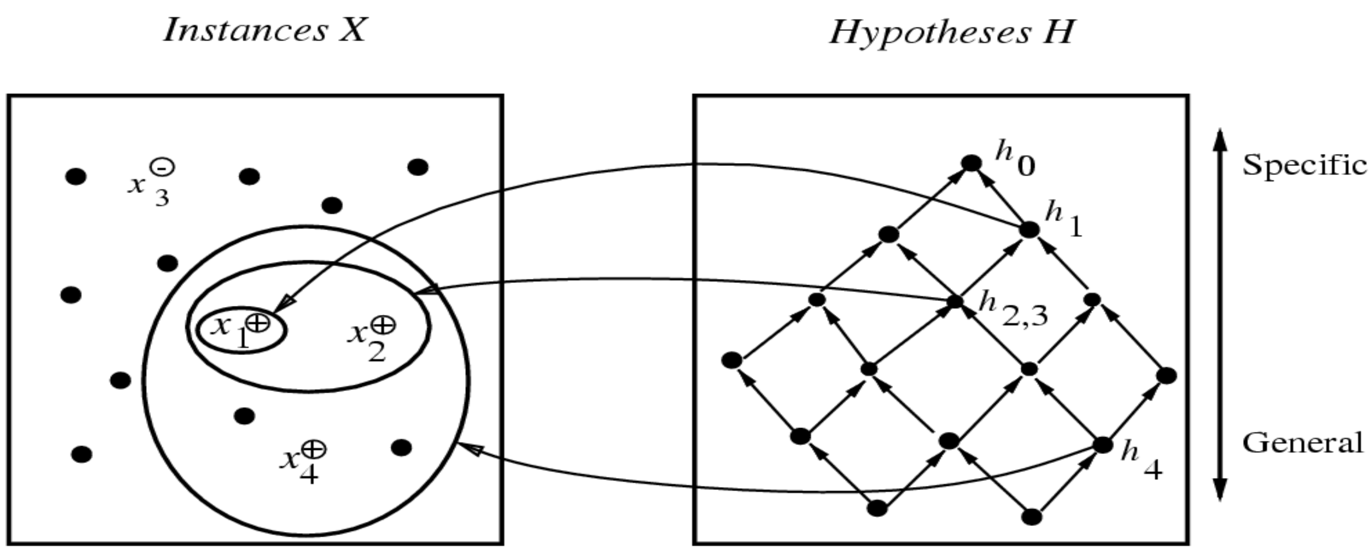
     If the constraint a_i in h is satisfied by x

     Then do nothing

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

Find-S Algorithm

- Hypothesis Space Search by ***Find-S***



h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

x_1 = *<Sunny Warm Normal Strong Warm Same>*, + $\quad$ h_1 = *<Sunny Warm Normal Strong Warm Same>*

x_2 = *<Sunny Warm High Strong Warm Same>*, + $\quad$ h_2 = *<Sunny Warm ? Strong Warm Same>*

x_3 = *<Rainy Cold High Strong Warm Change>*, - $\quad$ h_3 = *<Sunny Warm ? Strong Warm Same>* no change!

x_4 = *<Sunny Warm High Strong Cool Change>*, + $\quad$ h_4 = *<Sunny Warm ? Strong ? ? >*

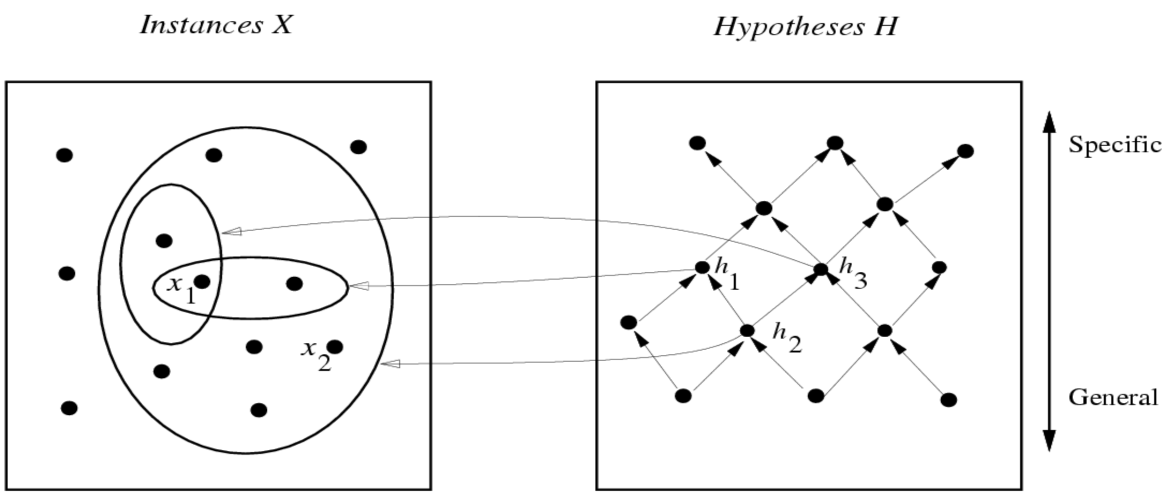
  - Substitute a "?" in place of any attribute value in *h* that is not satisfied by the new example.
  - Find S algorithm ignores the negative examples. So, no change in h_3.

Find-S Algorithm

- Why *F-S* never check a negative example ?
  - The hypothesis *h* found by it is the most specific one in *H*
  - Assume the target concept *c* (true) is also in *H* which will cover both the training and unseen **positive** examples
    - ♦ *c* is more general than *h*

can be represented as
a conjunction of attributes

  - So, the training data contains no errors, then the current hypothesis h can never require a revision in response to a negative example
  - Because the target concept will not cover the negative examples, thus neither will the hypothesis *h*



Drawbacks about *Find-S*

- Can not tell whether it has learned concept
  (Output only one. Many other consistent hypotheses may exist!)

- Picks a maximally specific *h* (why?)
  (Find a most specific hypothesis consistent with the training data)

- Can not tell when training data inconsistent
  - What if there are noises or errors contained in training examples

- Depending on *H*, there might be several maximally specific consistent hypotheses!

Consistence of Hypotheses Representation

- 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*.

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

Indexing	Article	InLibrary	Price	Editions	Buy
Scopus	Journal	No	Affordable	One	No
Scopus	Magazine	No	Expensive	Many	Yes

- *Assume*
    - h1 = (?, ?, No, ?, Many) - Consistent
    - h2 = (?, ?, No, ?, ?) - Inconsistent

Version Space Representation

- 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

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

  - A subspace of hypotheses
  - Contain all plausible (reasonable) versions of the target concepts

List-Then-Eliminate Algorithm

1. *VersionSpace* $\leftarrow$ a list containing all hypotheses in *H*

2. For each training example, $<x, c(x)>$

   remove from *VersionSpace* any hypothesis *h* for which $h(x) \neq c(x)$

   – i.e., eliminate hypotheses that inconsistent with any training examples

   – The *VersionSpace* shrinks as more examples are observed

3. Output the list of hypotheses in *VersionSpace*

Consistent Hypotheses and Version Space

- Assume the attributes, sky → (sunny, rainy) values and

  Temp→ (high, normal) values

- Instance space X : (sunny, high), (sunny, normal), (rainy, high), rainy, normal) – 4 possible training samples.

SNO	Sky	Temp
1	Sunny	High
2	Sunny	Normal
3	Rainy	High
4	Rainy	Normal

- *Consider ? and Ø values for each attribute for forming H*
- ***Hypothesis Space H:*** (sunny, high), (sunny, normal), (sunny, ?), (sunny, *Ø*), (rainy, high), rainy, normal), (rainy, ?), (rainy, *Ø*), (*Ø*, *sunny), (Ø, rainy), (Ø, Ø), (Ø, ?), (?, sunny), (?, rainy), (?, Ø), (?, ?),*
- **Semantically distinct Hypothesis:** (sunny, high), (sunny, normal), (sunny, ?), (rainy, high), rainy, normal), (rainy, ?), *(Ø, Ø (?, sunny), (?, rainy), (?, ?),*
  - *If Ø in hypothesis, it does not match with any example. So, ignore Ø hypothesis combination.*

List and Eliminate Algorithm - Example

- **Version Space:** (sunny, high), (sunny, normal), (sunny, ?), (rainy, high), rainy, normal), (rainy, ?), *(Ø, Ø* (*?, sunny), (?, rainy), (?, ?)*
- *Training examples*

Sky	Temp	Target
Sunny	High	Yes
Sunny	Normal	Yes

- Consider one training example at a time for determining consistency.
- Consistent Hypothesis are:

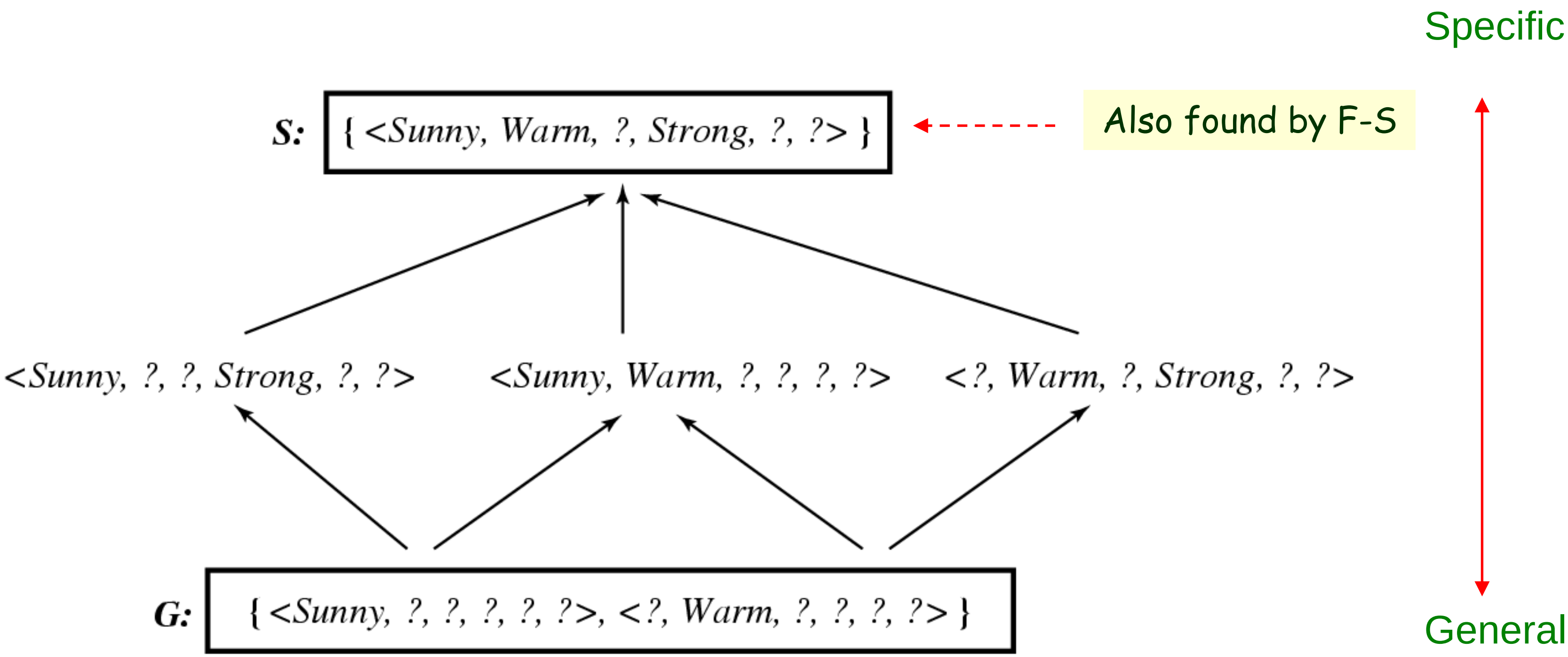
  (sunny, ?) and (?, ?)
- Rest of hypothesis are inconsistent.

Drawbacks of *List-Then-Eliminate*

- The algorithm requires exhaustively enumerating all hypotheses in *H. It consumes more time*
- Hypothesis should be finite.
- If insufficient (training) data is available, the algorithm will output a huge set of hypotheses consistent with the observed data.

Example Version Space

- Employ a much more compact representation of the version space in terms of its most general and least general (most specific) members.



Arrows mean more-general-than relations

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

Representing Version Space

- The **General boundary** *G*, of version space $VS_{H,D}$ is the set of its maximally general members

$$G \equiv \left\{ g \in H \middle| Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)] \right\}$$

- The **Specific boundary** *S*, of version space $VS_{H,D}$ is the set of its maximally specific members

$$S \equiv \left\{ s \in H \middle| Consistent(s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)] \right\}$$

- Every member of the version space lies between these boundaries

$$VS_{H,D} = \left\{ h \in H \middle| (\exists s \in S)(\exists g \in G)\ g \geq_g h \geq_g s \right\}$$

  – Version Space Representation Theorem

Candidate Elimination Algorithm

- $G \leftarrow$ maximally general hypotheses in H

$$G_0 \leftarrow \{\langle ?, ?, ?, ?, ?, ? \rangle\}$$

Should be specialized

- $S \leftarrow$ maximally specific hypotheses in H

$$S_0 \leftarrow \{\langle \phi, \phi, \phi, \phi, \phi, \phi \rangle\}$$

Should be generalized

Candidate Elimination Algorithm

- For each training example *d*, do
  - 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* 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*
        (i.e., partial-ordering relations exist)

positive training examples force the S boundary become increasing general

Candidate Elimination Algorithm

- 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

negative training examples force the G boundary become increasing specific

Example Trace

S_0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

G_0: {<?, ?, ?, ?, ?, ?>}

- If x_1. is positive example start with G boundary. If negative, start with S boundary

Example Trace

S_0: {< ∅, ∅, ∅, ∅, ∅, ∅ >}

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

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

G_0, G_1, G_2: {<*?, ?, ?, ?, ?, ?*>}

- 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* (i.e., partial-ordering relations exist)

Training examples:

1 . <*Sunny, Warm, Normal, Strong, Warm, Same*>, *Enjoy Sport = Yes*

2 . <*Sunny, Warm, High, Strong, Warm, Same*>, *Enjoy Sport = Yes*

Example Trace

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

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

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

< ? ? Normal ? ? ?>

?

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

Training Example:

3. <*Rainy, Cold, High, Strong, Warm, Change*>, *Enjo*

- 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*

- *G2 has six ways to be minimally specified*
  - <sunny,?,?,?,?,?>, <?,warm,?,?,?,?>, *<?,? ,Normal,?,?,? >* , *<?,? ,?,?,?,same >*.
  - *Check above h with examples which are checked until now i.e.,* x_3.
  - *<?,? ,Normal,?,?,? >* is inconsistent for x_3. So, ignored in G_3.

Example Trace

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

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

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

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

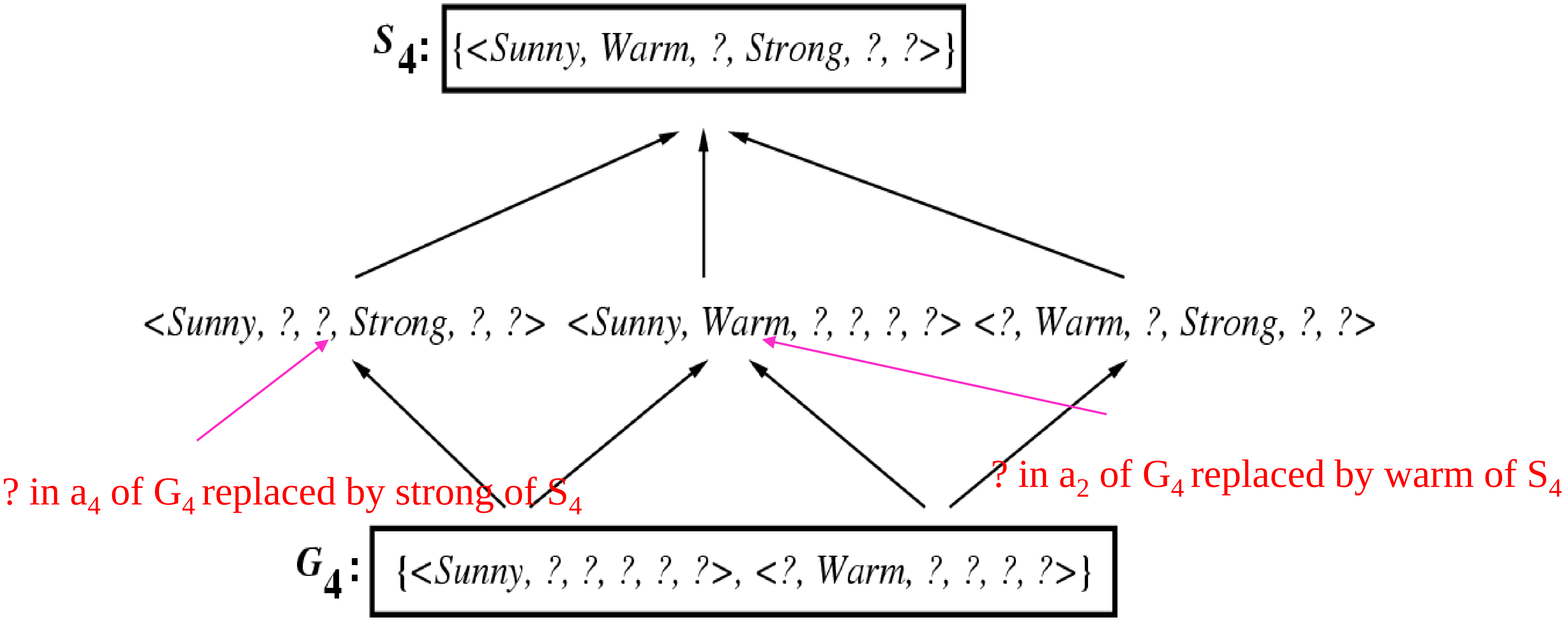
- For each training example *d*, do
  - If *d* is a positive example
    - Remove from *G* any hypothesis inconsistent with *d*
    - For each hypothesis *s* in *S* that is not consistent with *d*
      - Remove *s* from *S*
      - Add to *S* all minimal generalizations *h* of *s* such that
        - » *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* (i.e., partial-ordering relations exist)

Training Example:

4. <*Sunny, Warm, High, Strong, Cool, Change*>, *EnjoySport = Yes*

- Here, more than one hypothesis (two hypothesis in G_4,and one hypothesis S_4) exist. So, find some more possibilities.

Consider one from G and one from S for new possibilities.



- *S* and *G* boundaries move monotonically closer to each other, delimiting a smaller and smaller version space.
- Total six hypothesis available for given problem.

Candidate Elimination Remarks

- Is candidate algorithm converging to correct hypothesis and is there no error?
- Is obtained hypothesis exactly defining the target concept?
- If not, What training example should the learner (machine) request next?
  1. Training Examples taken externally or
  2. Training Examples request on its own (learner). It increases the H and VS size.
  - However, Version space (VS) should be reduced with each new training example.
  - If VS is reduced, then need log |VS|. Otherwise, it needs more than log |VS|.

Learning Types

Inductive Learning

- Machine derive (learning knowledge) the rules from training examples

Deductive Learning

- Existing rules (learned knowledge) are applying on the given training examples.

Biased Hypothesis Space

- Machine is learning using some examples. Then it utilizes already learned knowledge (from few examples) for new training examples.
- It **does not consider all kind of training examples** (Biased). So, it should include all kind of hypothesis.

E.g., sunny ^ warm ^ normal ^ strong ^ cool ^ change ➔ Yes

- If forecast value is ‘same’, then learner (machine) returns ‘No’ as target, even which is not important attribute.

Biased Hypothesis Space Example

- Biased hypothesis space
  - Restrict the hypothesis space to include only **conjunctions** of attribute values i.e., bias the learner (machine) to consider (learn) only conjunctive hypothesis.
- Can't represent disjunctive target concepts *"Sky=Sunny* **or** *Sky=Cloud"*

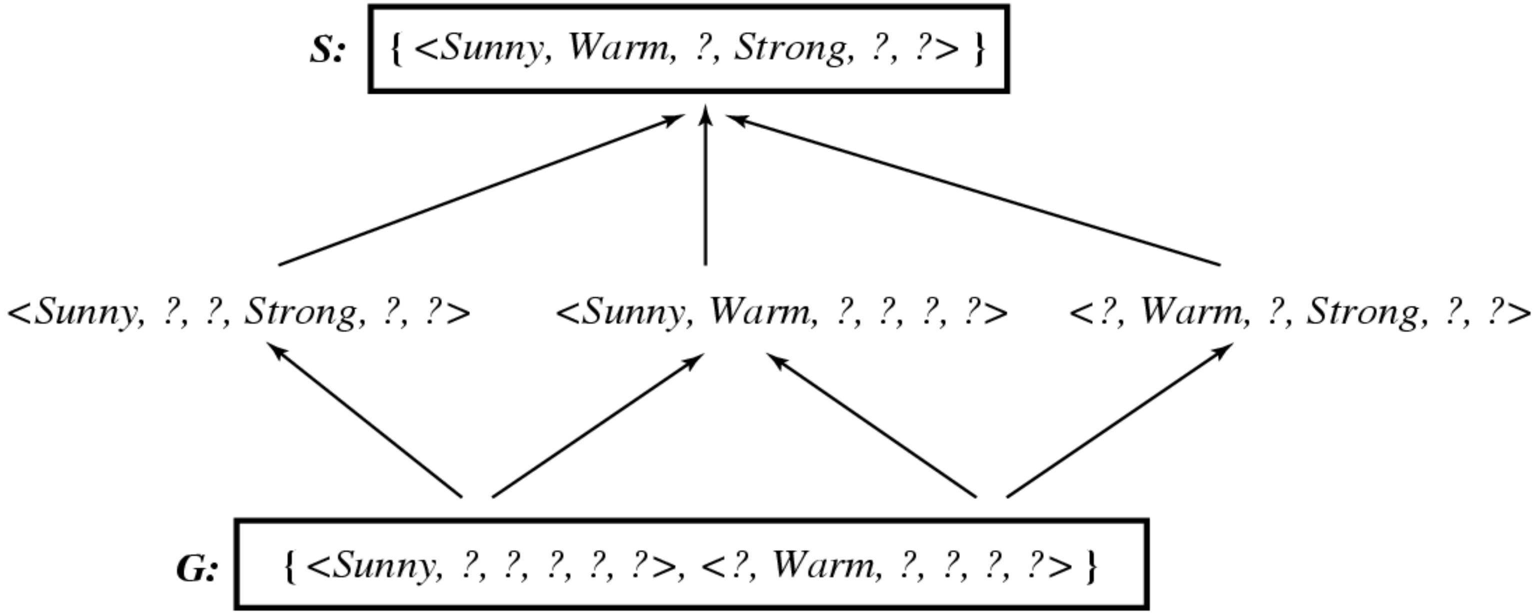
Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Cool	Change	Yes
2	Cloudy	Warm	Normal	Strong	Cool	Change	Yes
3	Rainy	Warm	Normal	Strong	Cool	Change	No

Lets consider the only one hypothesis in VS:
<?, Warm, Normal, Strong, Cool, Change>, *Check the h is consistent or inconsistent?*

At third example,
S2: <?, Warm, Normal, Strong, Cool, Change> → *becomes inconsistent,* $h(x_3) \neq c(x_3)$

- For third new example, $h(x_3)$ is positive. But $c(x_3)$ is No (-Ve). So, 'h' is considered as **biased hypothesis** based on first 2 examples.
- In Candidate algorithm, if h is inconsistent then h is removed from Version Space (No h in VS i.e., zero hypothesis in VS). It is called **biased hypothesis space.**
- To solve this issue, learner should use unbiased hypothesis using **power set** of D.

Partially (Biased) learned concepts used in Classification



A	Sunny	Warm	Normal	Strong	Cool	Change	?	Yes
B	Rainy	Cold	Normal	Light	Warm	Same	?	No
C	Sunny	Warm	Normal	Light	Warm	Same	?	?
D	Sunny	Cold	Normal	Strong	Warm	Same	?	?

- For x_3, three hypothesis provides +ve and three h provides –ve. So, learner unable to classify.
- For x_4, two h provides +ve and four h provides –ve. So, learner classify as majority –ve.

Unbiased Hypothesis Space

- Machine is learned using power set of (all) training examples in the instance space X. So, it will provide solution to all kind of examples.
- In our example, 6 attributes in X.
- Possible (power set) instances : $|X| = 3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$
- Target concepts (possible hypothesis) = $2^{|X|} = 2^{96}$.
- So, Learner should learn huge unbiased hypothesis space.
- It requires more learning time to apply these all possible hypothesis on every new training example.
- But, learner should learn in less time.

Unbiased Learner Example

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

- Consider Learner learns a task in unbiased approach by defining new hypothesis space H`.
- H` denotes every subset (possible combination) of instances.
- To define H`, can use arbitrary **disjunctions** of **positive** example and **negation** of **disjunction** of **negative** examples.
- E.g.,

  Target concept: *Sky=Sunny* **or** *Sky=Cloudy* " described as

  <sunny, ?, ?, ?, ?, ?> v *<cloudy, ?, ?, ?, ?, ?>*

Unbiased Hypothesis Space Example

E.g., Let consider Five examples instance Space X.

- In that three positive x_1, x_2, x_3 and two negative examples x_4, x_5.
- Specific boundary of VS contains h with disjunction of +ve examples S= (x1 v x2 v x3)
- General boundary of VS contains h with negation of disjunction -ve examples G = ~(x4 v x5).

Disadvantage:

- **Very expressive hypothesis** i.e. S boundary always be with disjunction of +ve observed examples and G boundary always be with negated disjunction of -ve observed examples.
- But Learner completely unable to generalize beyond the observed training examples i.e., may or may not correctly predict the new unobserved example.
- Hence, every h should check with all training examples in this approach. Due to unable to generalizing, there is possibility of misclassification.

The Futility (uselessness) of Bias-free (Unbiased) Learning

CE Assumptions:

- All conjunctions of attribute values for hypothesis to be considered. Disjunctions of attribute values not considered.

Property of inductive inference:

- Learner not have prori assumptions regarding target concept has no rational basis for classifying **any** new unobserved examples.
  - For generalizing the model, learner should have priori assumptions.

Idea of Inductive Bias

- Learner generalizes beyond the observed training examples to infer the classification of new unobserved example.
- Inductive bias with **inductive inference:**
  - Learner algorithm L which is trained using random set of training example. So, D_c (learning outcome of D) = {x, c(x)} of some random target concept c.
  - After training, Consider Learner receives new example 'x_i' that should be classified as +ve or -ve.
  - $L(x_i, D_c)$ denotes classification that is assigned to new instance x_i by Learner L using learned knowledge from D_c i.e., $L(x_i, D_c)$ can be inductively inferred from learned knowledge D_c using the trained data D i.e. $(D_c \wedge x_i) \succ L(x_i, D_c)$
  - $\succ$ - "inductively inferred from"

Inductive Bias with New Examples

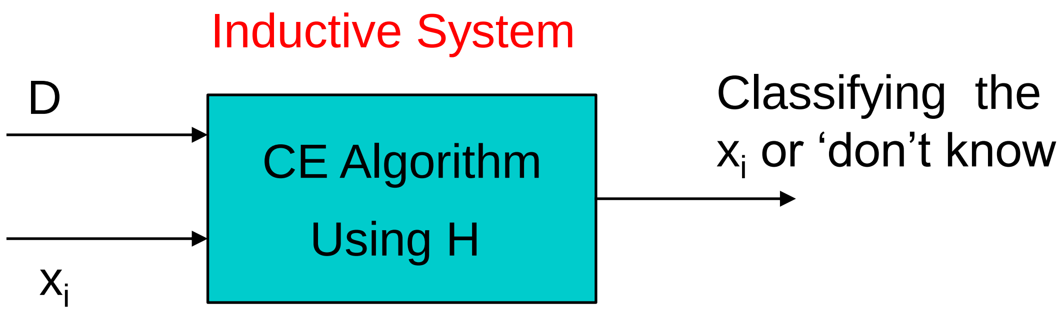
Instance	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport	
A	Sunny	Warm	Normal	Strong	Cool	Change	?	Yes
B	Rainy	Cold	Normal	Light	Warm	Same	?	No
C	Sunny	Warm	Normal	Light	Warm	Same	?	?
D	Sunny	Cold	Normal	Strong	Warm	Same	?	?

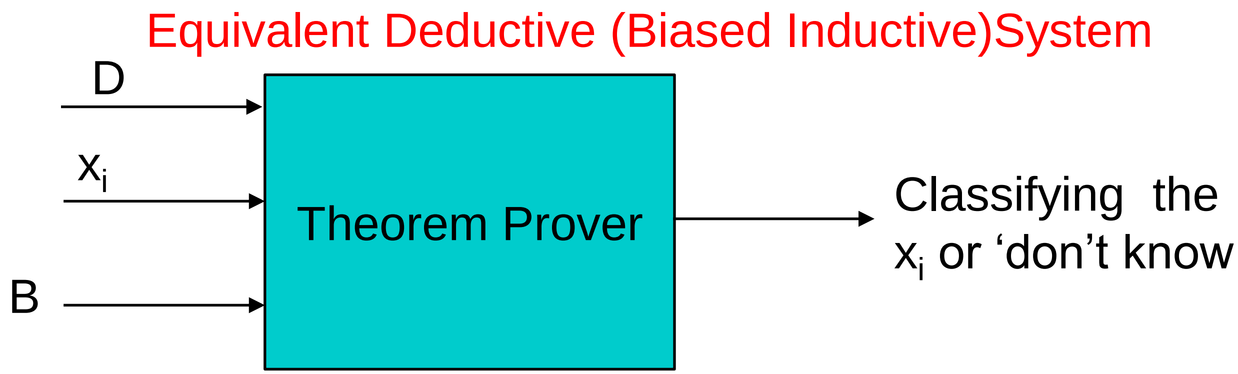
- For instance A, $L(x_i, D_c)$ = (EnjoySport = 'Yes') based on inductively inferred from earlier specific hypothesis in H during training using D_c.
- Additional assumptions could be added to $(D_c \wedge x_i)$ so that $L(x_i, D_c)$ would follow deductively.
- Define inductive bias of L to be minimal set of assertions (assumptions) B such that for all new instances x_i.

$$(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)$$

- $\vdash$ - "$L(x_i, D_c)$ follows deductively from $(B \wedge D_c \wedge x_i)$"
- Inductive bias make the learner to **justify** its inductive inferences as deductive inferences using set of additional assertions B.
- Inductive bias of CE algorithm is the target concept c that is in H.

Modelling inductive system by equivalent deductive system





Inductive Bias Advantages

- Generalizing the model beyond the observed training examples.
- Inductive bias learning is better than different learners Rote learner (algorithm), CE algorithm, Find S algorithm

References

1. Tom M. Mitchell, Machine Learning, McGraw Hill , 2017.
2. EthemAlpaydin, Introduction to Machine Learning (Adaptive Computation and Machine Learning), The MIT Press, 2017.
3. Wikipedia