CHAPTER-2

CONCEPT LEARNING & THE GENERAL-TO-SPECIFIC ORDERING.

concept learning ~~Task~~ means inferring a boolean valued func. from training examples of its i/p & o/p.

A Concept learning Task:

- Consider an example, where a friend enjoys water sport. Concept learning task is "Enjoy sport".

example	Sky	Air Temp	Humidity	wind	water	Forcast	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

Table: 2.1

→ The "Attribute" - Enjoy sport rep. whether the person enjoys water sport that day or not.

→ Task: To learn to predict "enjoysport" of an arbitrary day based on other values of attributes.

→ Hypothesis rep^n:- Conjuction of Constraints on the instance attributes.
- So Hypothesis is a vector having 6 Constraints here, i.e, sky, Airtemp, humidity, wind, water & forecast.
- For each attribute, Hypothesis will be:
  (i). "?" = Any value is acceptable for this attribute
  (ii) Specify a single reqd. value ex:- warm.
  (iii) "ϕ" - No value is acceptable.

→ Positive Example: - If an instance 'x' satisfies all constraints of hypothesis 'h' then 'h' classifies 'x' as +ve Example ⇒ $h(x)=1$.

Ex: ① (?, cold, High, ?, ?, ?) ⇒ on cold day with high humidity
② (?, ?, ?, ?, ?, ?) ⇒ enjoy sports every day ⇒ every day +ve
③ (ϕ, ϕ, ϕ, ϕ, ϕ, ϕ) ⇒ No day is a positive example

→ The "Enjoy sport" Concept Learning Task:-

Given:-

Instances X: possible days, each described by attributes
1. sky (sunny, cloudy & rainy)
2. Air Temp (warm & cold)
3. Humidity (Normal & High)
4. Wind (strong & weak)
5. water (warm & cool)
6. Forecast (same & change)

Hypothesis H: Each Hypothesis is described by constraint on attributes
"?" → any value acceptable
"ϕ" → no value acceptable.
specific value.

Target concept c: Enjoy sport: $X \rightarrow \{0, 1\}$.

Training Examples D: +ve & -ve examples of Target func.

Determine:-

A hypothesis "h" in H such that
$h(x) = c(x) \ \forall x \text{ in } X$.

→ Notation:-

- Instances: Set of items over which the concept is defined is called set of Instances, X.

- Target Concept: The concept ~~or~~ func. to be learned is called Target Concept, c.
It can be any boolean valued func. defined over the instances X.
$c: X \rightarrow \{0, 1\}$.
ex:- In "enjoy sport" concept, If
$c(x) = 1$ ⇒ if Enjoy spor = yes
& $c(x) = 0$ " " " No.

- <u>Training examples</u> :- when learning a target Concept a learner is provided a set of training examples. They Consist of an instance x from X, along with its target Concept value $c(x)$.
  - If $c(x) = 1 \Rightarrow$ Positive Training examples
  - If $c(x) = 0 \Rightarrow$ Negative "
  - It is written in ordered pairs $\langle x, c(x) \rangle$ for defining a training example.
    Here, x = instance & $c(x)$ = Target Concept Value
  - The set of available Training set examples are denoted by "D".

- <u>All possible Hypotheses</u> :- - Denoted by H.
  - ~~given a target concept c~~
  - using a set of Training examples, the learner must hypothesize or estimate "c".
  - so, the set of all possible hypotheses is H inorder to identify of target concept.
  - Each hypothesis in set "H" is denoted by "h" which is a boolean value defined over X.
    $\Rightarrow h : X \rightarrow \{0, 1\}$.

- The goal of the learner is to find a hypothesis 'h' such that $\boxed{h(x) = c(x) \ \forall \ x \in X}$

→ <u>The Inductive Learning Hypothesis</u> :-
It is a hypothesis which is found to approximate target func. well, over a sufficiently large set of training examples, it will also approximate the target func. well, over other unobserved examples.

Concept learning as Search:

→ Concept learning is like a task of searching through a large space of hypotheses (defined by hypothesis repⁿ)
→ Here, goal is to find hypothesis ; which fits the best to the training examples.

Ex: - In " Enjoysport " learning task , H = Hypotheses set
X = Instances set

The Attribute Sky has 3 possible values & Remaing AirTemp, Humidity, wind, water & forecast has 2 possible values.

Thus there are $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$ instances in X .

→ coming to Hypotheses there are $5 \times 4 \times 4 \times 4 \times 4 \times 4 = 5120$ "Syntactically distinct Hypotheses in H .

→ Every hypotheses with one or more "ϕ" symbol rep. empty set of instance ⇒ Instance is −ve .

→ Thus now " Semantically distinct Hypotheses" are

Only $1 + (4 \cdot 3 \cdot 3 \cdot 3 \cdot 3 \cdot 3) = 973$.

General - to - specific Ordering of Hypotheses:

Defⁿ:- Let h_j and h_k be boolean valued funcs., defined over X . Then 'h_j' is "more-general than-or-equal to" h_k ($h_j \geq_g h_k$) iff

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

Ex:- $h_1 = \langle$ sunny, ?, ?, strong, ?, ? $\rangle$
$h_2 = \langle$ Sunny, ?, ?, ?, ?, ? $\rangle$

Consider set of instances classified as +ve by h_1 & h_2

As seen, any instance classified +ve by h_1 will also be classified +ve by h_2

$\Rightarrow h_2$ is more general than h_1

($\because$ h_2 imposes fewer constraints on instance, also, it classifies more instances to be +ve)

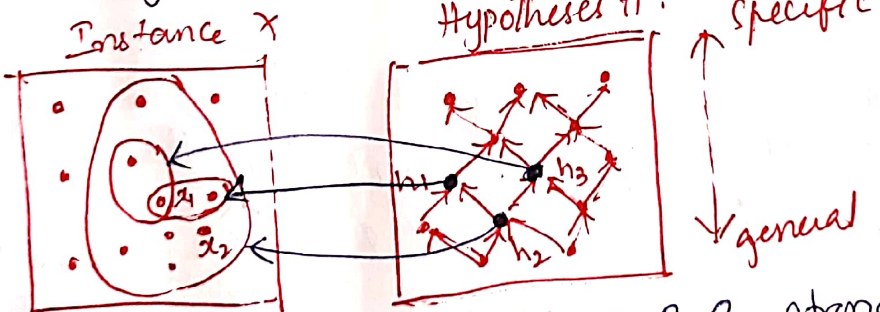
→ In general, for any instance "x" in the set of Instances X and any hypothesis "h" in set of hypotheses 'H' we say, x satisfies h iff $h(x)=1$.

→ Given 2 hypotheses h_k & h_j, if any instance satisfying h_k also satisfies h_j then h_j is more general or equal to h_k.

→ strictly, If $(h_j \geq_g h_k) \wedge (h_k \ngeq_g h_j)$ then h_j is more_general_than h_k. (or) h_k is more_specific_than h_j. (Inverse)

→ Example, Assume 3 hypotheses h_1, h_2 and h_3 from "Enjoy sport" example.

~~Also, $h_2 \geq_g h_1$~~ \ ~~since, every instance~~



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, ?>

→ Here we see that, $h_2 \geq_g h_1$ because every instance that satisfies h_1 also satisfies h_2; ||ly $h_2 \geq_g h_3$

→ $\geq_g$ relⁿ depends only on instances & not on the classification of them.

→ A $\geq_g$ relⁿ defines a partial order over hypothesis space (7)
H. (Relⁿ is Reflexive, antisymmetric & Transitive).
→ The structure is partial order ⇒ there may be pairs of hypotheses such as h_1 & h_3 such that $h_1 \not\geq_g h_3$ and $h_3 \not\geq_g h_1$.

Find-S: Finding a Maximally Specific Hypothesis:

→ It is an applⁿ of more general than partial ordering.
→ Here, we begin with most specific possible hypothesis in H & generalize it, when it fails to cover an observed +ve Training example.
(classify as +ve).

→ Find-S Algorithm: -

1. Initialise h to most specific hypothesis in H.
2. For each +ve Training instance x.
   - For each attribute constraint a_i in h
     if constraint a_i is satisfied by x
       then do nothing
     else
       Replace a_i in h with next more general constraint satisfied by x.
3. O/P hypothesis h.

→ Example: -
- consider "EnjoySport task".

(i) Initialise h to most specific hypothes in in H.

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

(ii) From table 2.1, none of constraints are satisfied by this so it is replaced by next more general constraint that fits this example.

h ← ⟨Sunny, Warm, Normal, Strong, Warm, Same⟩

[Its too specific be'coz all instances are -ve except one +ve Traing example, (See Table 2.1 to relate)]

→ The second Training example forces algo. to further generalise h, which can be done by substituting any attribute value in 'h' by a "?".

h ← ⟨Sunny, warm, ?, Strong, warm, Same⟩

→ Upon encountering a -ve Training example (3rd one) Find_S algo simply ignores it. So no revision of "h" is needed.

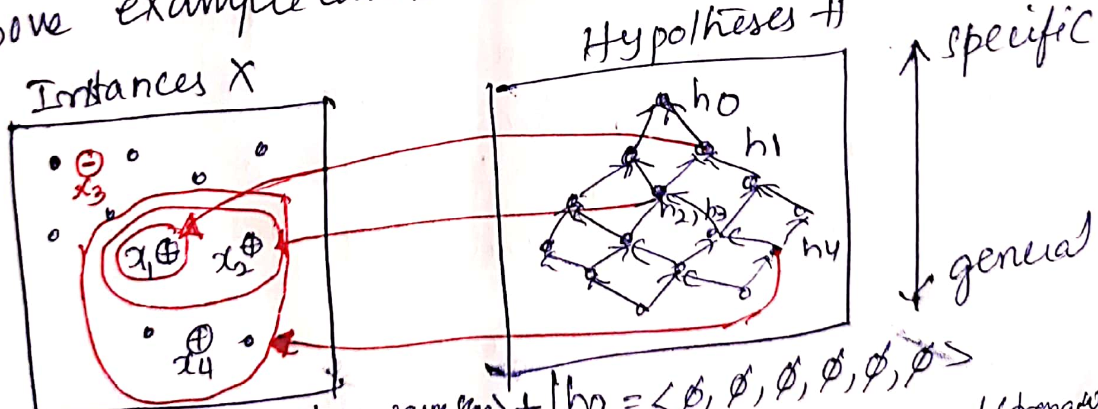
→ the 4th example (+ve) leads to further generalisation of "h".

h ← ⟨sunny, warm, ?, strong, ?, ?⟩

→ In this Algo., "More_general_than" partial ordering is thus used to organise the search for an acceptable Hypothesis.

→ At each step, the hypothesis is generalised as far as necessary to cover the new +ve examples thus making the hypothesis more consistent with Training examples. Hence it is named as "Find_S" Algorithm.

→ Above example can be illustrated further with foll. fig.



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 = ⟨ϕ, ϕ, ϕ, ϕ, ϕ, ϕ⟩
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, ?, ?⟩

→ "Find - S" is guaranteed to o/p most specific hypothesis within H that is consistant with +ve training example Finally, it is also consistent with -ve examples provided the correct target concept is contained in H & also provided the training examples are correct.

→ Few Voids in this Algo. in the form of Questions are:
* Has learner converged to correct target concept?
* why prefer most specific hypothesis?
* Are Training examples consistent
* what if there are several maximally specific consistent hypotheses?

Version spaces and "The Candidate Elimination Algorithm":

→ Helps to address limitations of Find_S Algo.

→ "Find_S" Algo gives as an o/p, only single hypothesis from H that is consistent with Training examples.

→ "Candidate_Elimination" Algo on other hand, o/ps a set of "all hypotheses" that are conistent with Training examples.

→ It also uses "More_general than" partial ordering, but produces o/p without explicitly enumerating all of its members. Hence it is more compact repn of set of consistent Hypotheses.

→ Both "Find_S" & "Candidate_Elimination" Algos. perform poorly when noisy Training data is given.

→ Appln's of "Candidate_Elimination" Algo:
- Learning Irregularities in chemical mass spectroscopy
- Learning Control rules for Heuristic search

→ Representation :-

- Consistent :-

A Hypothesis 'h' is said to be consistent with a set of training examples 'D' iff $h(x) = c(x)$ for each example $\langle x, c(x) \rangle$ in D.

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

NOTE :

Satisfy	Vs. Consistent
An example 'x' satisfies a hypothesis 'h' when $h(x) = 1$ irrespective of x is +ve or -ve example of Target Concept	x is consistent with h is not only dependent on target concept but also its compulsory that $h(x) = c(x)$.

- Version space :-

It is denoted as $VS_{H,D}$ w.r.t hypothesis space H & training examples D.

It is a subset of Hypotheses from H consistent with Training examples in D.

→ The "List-Then-Eliminate" Algorithm :-

It is used to rep. the version space by simply listing all of its members.

Algo :-

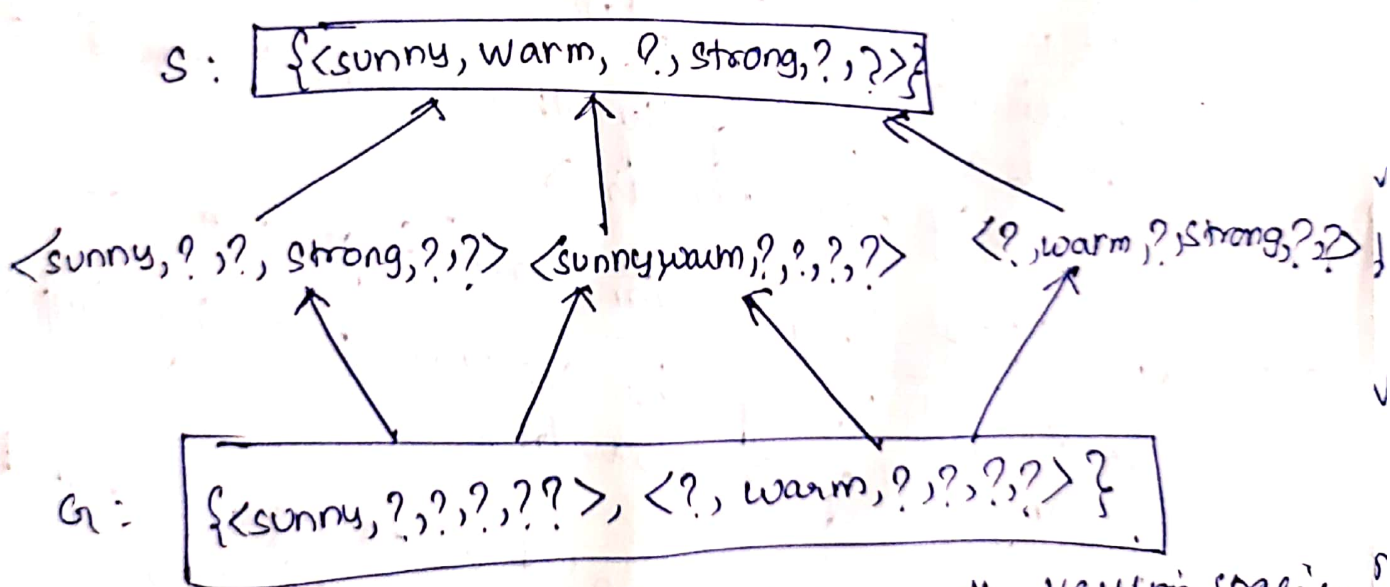
1. version space ← a list containing every hypothesis in H
2. For each Training example, $\langle x, c(x) \rangle$
   - remove from "version space" any hypotheses 'h' for which $h(x) \neq c(x)$
3. O/p. the list of hypotheses in version space

- Limitation :-
  - * It can be applied on finite Hypotheses space H,
  - * Exhaustive enumeration of all 'h' in H is reqd.
- Advantages:-
  - * guarantees to o/p all hypotheses consistent with training data.

→ **More Compact representation for Version spaces:-**

- Here, version space is rep. by most general & least general members, thus setting boundaries for version space. They form general boundary set & specific boundary set.
- For this, we use example of "Enjoy sport task" & its o/p of "Find-S" algorithm, which is:

h = <sunny, warm, ?, strong, ?, ?>.



- In "candidate Elimination" Algo., the version space is rep by storing only its most general members, "G" & and most specific members "S".
- General Boundary "G":- G w.r.t Hypothesis space H & training data D is set of maximally general members of H consistent with D

$$G \equiv \{g \in H \mid consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge consistent(g', D)]$$

22

- specific boundary S:
S w.r.t H & D is set of minimally general members of H Consistent with D

$$S \equiv \{s \in H \mid Consistent(s, D) \land (\neg \exists s' \in H)[(s >_g s') \land Consistent(s', D)]\}$$

- Version space repⁿ theorem:-

Let X be arbitrary set of instances
Let H be set of a boolean valued Hypotheses defined over X
Let $C : X - \{0,1\}$ be an arbitrary Target Concept defined over X.
Let D be an arbitrary set of training examples $\{\langle x, c(x)\rangle\}$.
Let D be ~~an arbitrary~~ ... S&G are well defined,

$\forall$ X, H, C&D such that S&G are well defined,

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

proof:-
* Every h satisfying RHS is in $VS_{H,D}$.
* Every member of $VS_{H,D}$ satisfies RHS

This must be proved.
Let 'g' is an arbitrary member of G
's' " " " S
'h' " " " H
such that $g \geq_g h \geq_g s$.
⇒ s must be satisfied by all +ve examples in D (as per definition).
$\because h \geq_g s$ ⇒ it should also be satisfied by all +ve examples in D

As per defⁿ of G, g can't be satisfied by a -ve example in D
$\because g \geq_g h$ ⇒ h also " " " "
⇒ h is consistent with D ⇒ h is member of $VS_{H,D}$

→ "Candidate_Elimination" Learning Algorithm:—

1. Initialise G to set of maximally general Hypotheses in H
2. " S " " " specific " "
3. For each training example 'd', do.
   - If d is positive example
     (i) Remove from 'G' any hypothesis inconsistent with 'd'.
     (ii) For each hypothesis 's' in S that is inconsistent with 'd':
       (a) Remove 's' from 'S'.
       (b) Add to 's' all minimal generalization 'h' of 's' such that
         (i) h is consistent with d
         (ii) some member of G is more general than h.
       (c) Remove from S, any hypothesis i.e more general than another hypothesis in S.
   - If d is Negative example.
     (i) Remove from 's' any hypothesis inconsistent with 'd'.
     (ii) For each hypothesis 'g' in G i.e not consistent with 'd'
       (a) Remove 'g' from 'G'.
       (b) Add to 'G' all minimal generalisation 'h' of 'g' such that
         (i) h is consistent with d
         (ii) some member of S is more specific than h.
       (c) Remove from G, any hypothesis i.e less general than another hypothesis in G.

Example:-

→ Initialise G & S

$G_0 \leftarrow \{<?,?,?,?,?,?>\}$ Most general hypothesis

$S_0 \leftarrow \{<\phi, \phi, \phi, \phi, \phi, \phi>\}$ " specific " "

→ Candidate_Elimination Trace :-

Training Examples :

1. <sunny, Warm, Normal, strong, warm, same>, Enjoy sport = yes.

2. <sunny, warm, ~~Normal~~ high, strong, warm, same>, Enjoysport = yes.

- S_0 & G_0 are initial boundary sets. ~~sets~~
- Training Examples 1 & 2 force S boundary to become more general, as in "Find S" Algo.
- They don't have any effect on G boundary.

S_0 : $\{<\phi, \phi, \phi, \phi, \phi, \phi>\}$

↓

S_1 : {<sunny, warm, Normal, strong, warm, same>}

↓

S_2 : {<sunny, warm, ?, strong, warm, same>}

G_0, G_1, G_2 : $\{<?,?,?,?,?,?>\}$

→ Candidate_Elimination Trace 2:-

Training Example:

3. <Rainy, Cold, High, strong, Warm, change>, Enjoysport = NO

∵ its a -ve example, it will now force G2 boundary to be specialised to G3. & there will be several alternative maximally general hypotheses that are included.

S_2, S_3: {<sunny, warm, ?, strong, warm, same>}

G3: {<sunny, ?, ?, ?, ?, ?>, <?, warm, ?, ?, ?, ?>, <?, ?, ?, ?, ?, same>}

G2: {<?, ?, ?, ?, ?, ?>}

→ Candidate_Elimination Trace 3:-

Training Example:

4. <Sunny, Warm, high, strong, cool, change>, Enjoy sport = Yes.

∵ +ve Training example, it generalizes 'S' boundary from S3 to S4.

Also one member of G3 should be deleted since it is no longer more general than S4 boundary.

S3: {<sunny, warm, ?, strong, warm, same>}

S4: {<sunny, warm, ?, strong, ?, ?>}

G4: {<sunny, ?, ?, ?, ?, ?> <?, warm, ?, ?, ?, ?>}

G3: {<sunny, ?, ?, ?, ?, ?> <?, warm, ?, ?, ?, ?> <?, ?, ?, ?, ?, same>}

Final version :-

- Set S4 & G4 delimit version space of all hypotheses consistent with set of incrementally observed training examples.
- The foll. learned version space is independent of the seq. in which training examples present.
- S & G boundaries move monotonically closer, delimiting smaller & smaller version space of candidate hypothesis

S4: {<sunny, warm, ?, strong, ?, ?>}

<sunny, ?, ?, strong, ?, ?> <sunny, warm, ?, ?, ?, ?> <?, warm, ?, strong, ?, ?>

G4: {<sunny, ?, ?, ?, ?, ?>, <?, warm, ?, ?, ?, ?>}

Remarks On Version space & "Candidate - Elimination":

→ "Version space" learned by the Candidate - Elimination algorithm will converge towards the hypothesis that correctly describes the target concept, provided
  (i) There are no errors in the training examples.
  (ii) There is some Hypothesis in H that correctly describes the target concept.

→ Also if training data contains errors, Algo. will remove correct target concept from Version space & may create an empty version space.

→ If the learner is allowed to conduct experiments in which it chooses next instance (a.k.a. Query) then it obtains correct classification for this instance from an external oracle (ex: Nature or Teacher).

Example:-
For "Enjoy sport" task, the "Query stratigy" would be to choose an instance that would be classified +ve by some of these hypotheses, but negative by others such as:
  <Sunny, warm, Normal, Light, warm, same>

→ With optimal Query strategy, size of Version space is reduced by half with each new example: The correct target concept can therefore be found with only $\lceil \log_2 |VS| \rceil$ experiments

→ It is possible to classify certain examples, with same degree of confidence as if the target concept had been uniquely identified, even if it uses partially learned concept.

→ Consider an instance A, which was not amongst the training examples, the learner will classify it as +ve. since, condition will be met iff instance satisfies every member of S. (with definition of more_general_than).

→ Ily For instance B, it will be classified Negative by every hypothesis in version space; given a partially learned concept. (Just check if instance satisfies none of members of G)

→ For instance C, Half of version space hypotheses classify it +ve & ½ as −ve. Hence Learner can't classify this instance confidently.

→ Incase of instance D, it is classified +ve by 2 of Version Space hypotheses & −ve by other 4 ⇒ Negative classification it is.

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

Inductive Bias:-

→ "candidate_Elimination" Algo converges iff. accurate training examples & its initial hypothesis space contain target concept.

→ A Biased Hypothesis space:-

- Assume that hypothesis space contains unknown target concept.
- In "Enjoy sport" example, hypothesis space was restricted to include only conjunctions of attribute values.
- Becoz of this restriction, hypothesis space is unable to rep. even simple disjunctive target concepts such as "sky = Sunny or sky = cloudy".

Ex	sky	Air Temp	Humidity	Wind	water	Forecast	Enjoysport
1	sunny	warm	Normal	strong	cool	change	Yes
2	cloudy	warm	"	"	"	"	Yes
3	Rainy	warm	"	"	"	"	NO

There are no hypotheses which are consistent with these 3 examples above.

- The maximally specific hypothesis from H is
  S_2 : (?, warm, Normal, strong, cool, change)

- Problem:-
  The learner is biased to consider only conjunctive hypotheses.

→ An unbiased learner:-

Powerset:- A hypothesis space capable of representing every "Teachable concept" i.e capable of rep. every possible subset of instances of X.
⇒ Set of all subsets of X is called Powerset of X.

ex:- X for "Enjoy sports" is 96.

So, Distinct subsets $= 2^{|x|} = 2^{96} = 10^{28}$.

- Reformulate "Enjoy sports" Task in unbiased way, lets define new hypothesis space H'. H' rep. every subset of instances ⇒ corresponds to powerset of X. Lets allow arbitrary disjunctions, conjunctions & negations also.

⇒ "sky = sunny" or sky = cloudy" can be defined as
<sunny, ?, ?, ?, ?, ?> v <cloudy, ?, ?, ?, ?, ?>.

- If a "Candidate_Elimination" Algo is used then S boundary is disjunction of observed +ve example whereas G boundary will be negated disjunction of -ve examples. So, we have to present every single instance in X as Training example, instead of S & G.

→ Futility of Bias-Free learning:-

* Fundamental property of Inductive Inference
- A learner that makes no a priori assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances.

- Inductive learning requires some form of prior assumptions or Inductive bias to classify new instances.

- Consider an arbitrary learning algo. 'L' & an arbitrary set of Training data $D_c = \{<x, c(x)>\}$ arbitrary Target concept.
Here c = some arbitrary Target concept.

- After Training, 'L' classifies a new instance x_i

- Let $L(x_i, D_c)$ denotes that classification (+ve or -ve)

- Inductive Inference step performed by L is:

$$(D_c \wedge x_i) \succ L(x_i, D_c).$$

Scanned with CamScanner

where "$\succ$" symbol means Inductively inferred,
$\Rightarrow y \succ z \Rightarrow x$ is inductively inferred from y.

- Assume L = candidate Elimination Algo.
  D_c = Training data of "Enjoy sport" Task.
  Here we should also define inductive bias of "L".

<u>Definition</u>:-
inductive bias of 'L' is any minimal set of Assertions 'B' such that for any target concept c & corresp. Training examples D_c

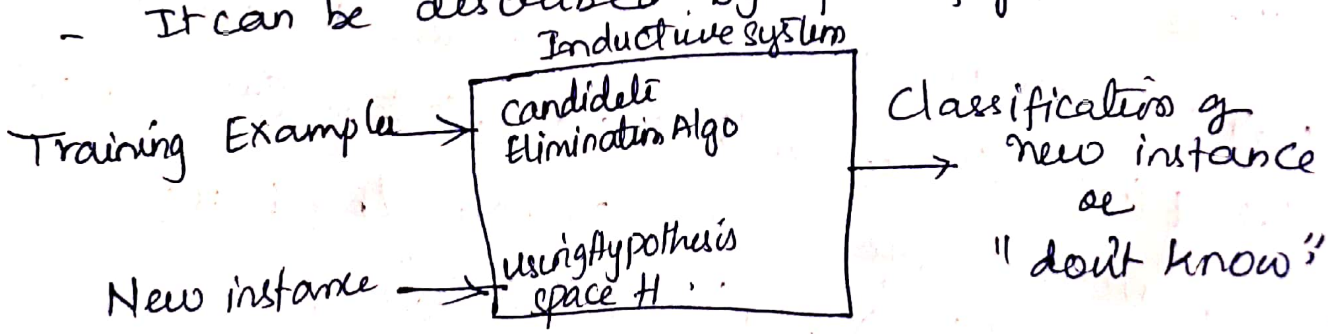
$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$$

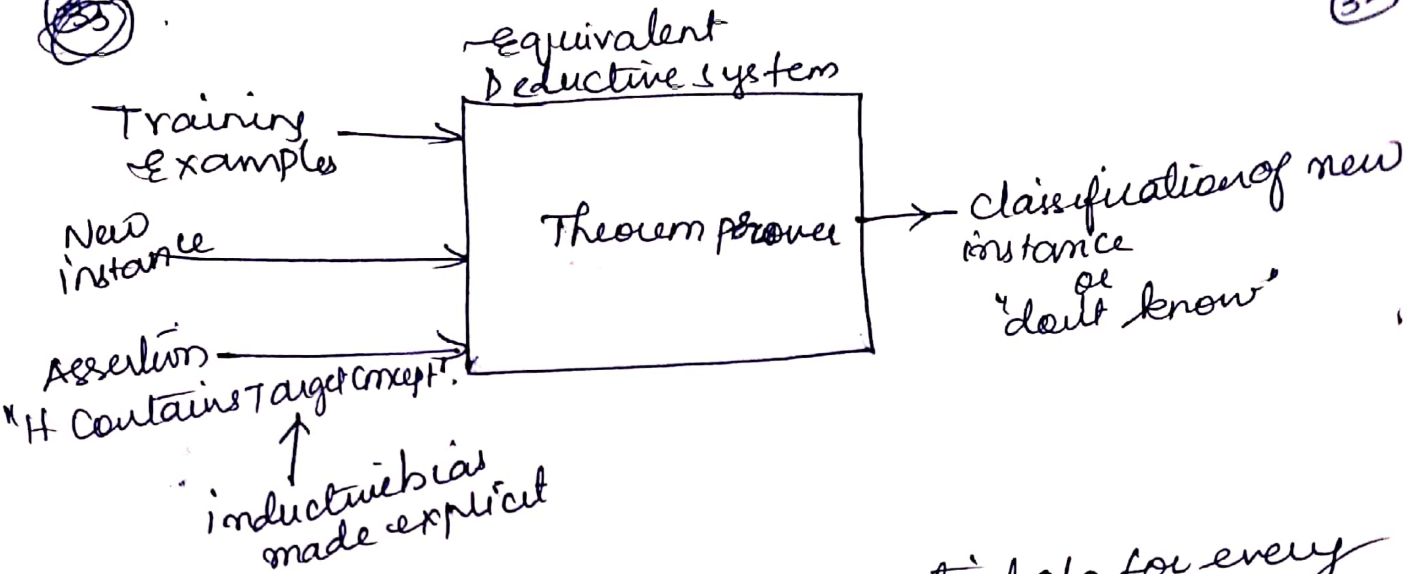
[<u>Note</u> :- $y \vdash z \Rightarrow z$ follows deductively from y
i.e z is provable from y]

- Given D_c, use candidate_Elimination Algo.
- compute Version space VS_{H,D_c}
- classify new instance x_i (+ve or −ve)
- Inductive bias : Assumption that $c \in H$.
- The classification $L(x_i, D_c)$ follows deductively from $B = \{c \in H\}$ along with Data D_c & description of instance x_i
- so, if we assume $c \in H$ then it follows deductively that $c \in VS_{H,D_c}$. Also, classification $L(x_i, D_c)$ is to be unanimous vote of all hypothesis in VS_{H,D_c}

$$\therefore \quad c(x_i) = L(x_i, D_c)$$

- It can be described by foll. fig:





These 2 sys. will produce identical o/p for every possible i/p set of training set & new instances set X.

→ Inductive Bias Advantages:-

(i) provides Non procedural means of characterizing their policy for generalizing beyond observed data

(ii) Allows Comparison of diff. learners accdg to strength of inductive bias.

→ The 3 Algo. Example from Weakest to strongest bias are:

(i) Rote-Learner:
- simply storing each observe Training example in memory
- It has no inductive bias.
- Instances are classified by looking them up in memory. If found, the stored classification is returned else sys. refuses to classify the new instance.

(ii) Candidate_Elimination:-
- New instances are classified only when all members of the current version space agrees on the classification, else it refuses to classify new instance. It has strong inductive bias

(iii) Find-S:-
- Finds most specific hypothesis consistent with the Training examples. Then it uses this hypothesis to classify the subsequent instances. It has a stronger inductive bias