Concept Learning (Classification)

MSc. Bui Quoc Khanh

khanhbq@hanu.edu.vn

Classification: Problem Definition

- Classification is the task of assigning a class label to an instance based on its attribute values, e.g.,
  - Assign a label "mammal" or "non-mammal" to an instance representing a living being
  - Assign a label "reliable" or "non-reliable" to an instance representing a bank customer
  - Assign a newspaper article to a topic (sport, politics, science, ..)
- Classification is a predictive task, as it allows to predict the label of a new instance based on past instances

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- A training set S is a set of examples (instances), each associated with a label (class)
- Each example is characterized by a set of attributes (properties), i.e., Body Temp, Aquatic, Aerial, Legs, Hibernates, Gives Birth
- The association between examples and class labels is called target function

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- Target function:
  - f: {Example} -> {Class Label}
  - f(human) = f(Warm, No, No, Yes, No, Yes) = mammal
  - f(python) = f(Cool, No, No, No, Yes, No) = reptile
- Training set S = {<e, f(e)>}, where f is the target function

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- A hypothesis is a proposition formula defined over the attributes
- Conjunctive hypothesis for concept “mammal”
  - h: body temp=?∧ aquatic=?∧aerial=no ∧ legs=?∧hibernates=no ∧ gives birth=yes
- h is a conjunction of constraints on attributes
- ?: don’t care constraint (any value)
- Shorthand: <?, ?, no, ?, no, yes>

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- According to h = <?, ?, no, ?, no, yes> a mammal is a living being which is not aerial, does not hibernates and gives birth (no matter for the other attributes)
- Is h consistent with the training data?

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- h = <?, ?, no, ?, no, yes>
- humans and whales both satisfy the above hypothesis h, so h(human)=mammal and h(whale)=mammal - they are both mammals according to h
- Since f(human) = mammal and f(whale) = mammal, h is consistent with the positive examples

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- h = <?, ?, no, ?, no, yes>
- Pythons and Salmons both do not satisfy the above formula, hence h(python) ≠ mammal and h(salmon) ≠ mammal, so are non-mammal according to h
- Since f(python) ≠ mammal and f(salmon) ≠ mammal, h is consistent with the negative examples

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- h = <?, ?, no, ?, no, yes> is consistent with all examples
- Hence, h is a model representing the concept "mammal" (according to our training set)
- The larger and the more statistically significant the training set, the more reliable the model
- Concepts cannot be exactly learned from limited data, only approximated

The Mammal example

The mammal training set

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Class
human	Warm	No	No	Yes	No	Yes	mammal
Python	Cool	No	No	No	Yes	No	reptile
salmon	Cool	Yes	No	No	No	No	fish
Whale	Cool	Yes	No	No	No	Yes	mammal

- A turtle is <cool, yes, no, yes, no, no> - Is the turtle a mammal?
- Turtle does not belong to the training set – no SQL query
- The model of “mammal” is h=<?, ?, no, ?, no, yes>
- Since turtle does not satisfy h, the answer is NO (according to our model of mammal)

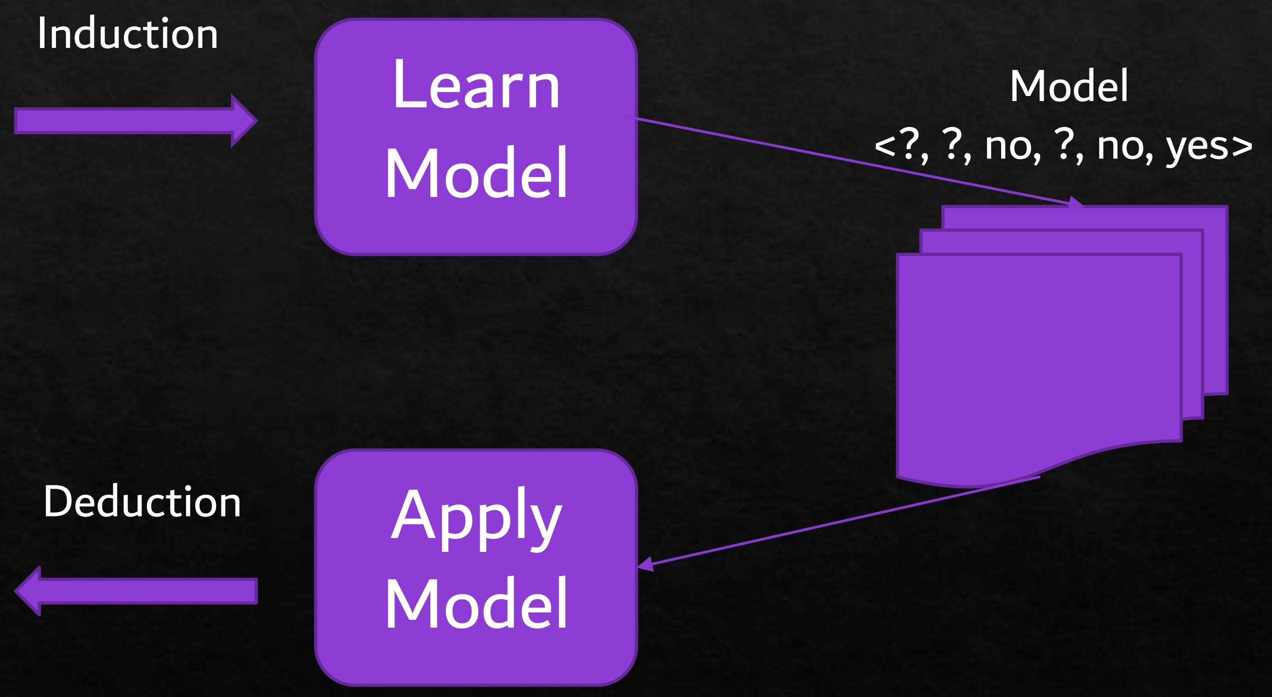
Classification

Training data

Name	BT	...	mammal
Human	warm	...	YES
Python	cool	..	NO
Salmon	cool	...	NO
Whale	Cool	...	YES
...	...	..	...

New instances

Name	BT	...	mammal
Turtle	cool	...	NO
dog	warm	...	YES
Cat	warm	...	YES



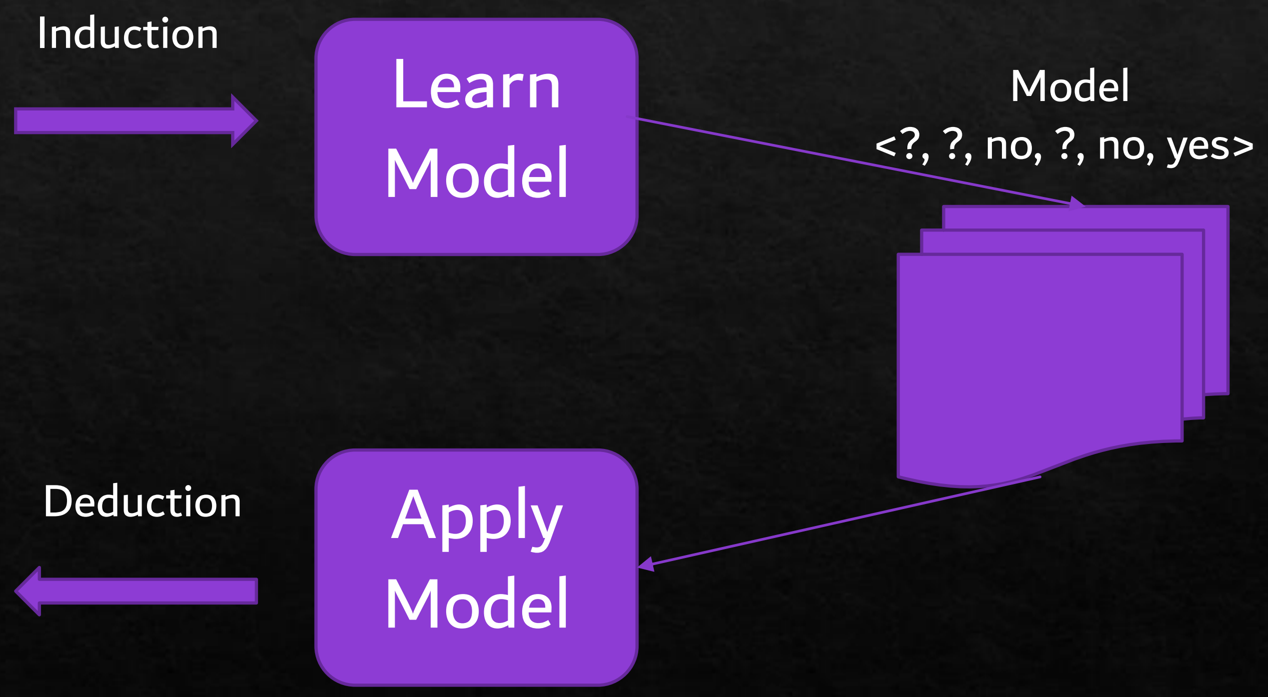
Classification

Training data

Name	BT	...	mammal
Human	warm	...	YES
Python	cool	..	NO
Salmon	cool	...	NO
Whale	Cool	...	YES
...	...	..	...

New instances

Name	BT	...	mammal
Turtle	cool	...	NO
dog	warm	...	YES
Cat	warm	...	YES



Hypothesis Space

- **Hypothesis**: propositional expression defined over the attributes of the training set
- **Hypothesis Language**: the kind of propositional expressions used for representing hypotheses (conjunctions, disjunctions, etc.)
- **Hypothesis Space**: set of all possible hypotheses (e.g., the set of all propositional formulas, expressible in the given language, over the attribute values)

Conjunctive Hypothesis Space

- The hypotheses space is the set of all possible conjunctions over the attributes of the training set
- Given conjunction $h=<h_1, ..., h_n>$ and example $e = <e_1, ..., e_n>$, we say that e satisfies h if e "fits into" h, that is,
  - for each i=1,n, either hi=? or hi=ei
- Let h = <?, ?, no, ?, no, yes> be a hypothesis
  - e = <cool, yes, no, yes, no, yes> satisfies h
  - e = <cool, yes, yes, yes, no, yes> does not satisfy h

Conjunctive Hypothesis Space

- If h is a hypothesis for class c and e satisfies h then h(e)=c, otherwise h(e) ≠ c – e.g. if h is a hypothesis for mammal and e is a living being that satisfies h, then e is a mammal according to h
- h is consistent with example e in S if
  - e is positive and satisfies h, i.e., h(e)=c
  - e is negative and does not satisfy h, i.e., h(e) ≠ c
- h is consistent with S if it is consistent with any example in S
- h is a model for S if it is consistent with S

The binary setting

- A binary problem is one with class c ∈ {0, 1}, that is f: {examples}-> {0,1} - conventionally, 1 is the positive class
- If e satisfies h, then h(e)=1, otherwise h(e)=0
- h is consistent with e if h(e) = f(e), for any example e
- h is a model if it is consistent with any example in the training set

Classification: Problem Definition

- Let S be a set of training examples

$$S = \{<e, f(e)>\} \text{ where}$$

  - e is an example $<e_0, \ldots, e_n>$
  - f the target function and f(e) is the target concept (or class label) of e
- Let H be a given hypothesis space
- **Classification problem:** Learn from S a hypothesis in $h \in H$ which is a model for S, i.e., h is consistent with the examples in S

Classification as search problem

- Search Problem: Given a search space H and a training set S=<e,f(e)>, find $h \in H$ (if any) consistent with the training data, i.e., such that

$$h(e)=f(e), \text{ for any example } e \text{ in } S$$

Classification as search
Naive approach

- *Input*: training set S={<e,f(e)>}, hypothesis space H
- *begin* Naïve Search
  - For each training example <e, f(e) >
    - remove from H any hypothesis h for which h(e) <> f(e) – h is indeed not consistent with S
  - Output H – (possibly empty) set of models
- *end*

Classification as search
Naive approach

- H = { ... <cool, no, ?, ?, yes, no>, ..., <warm, no, ? ,?,no, ?>, ... }
- h= <cool, no, ?, ?, yes, no> does not satisfy human, so h(human) = no
- Since f(human) = yes, h is not consistent
- Hence, h is removed from H

Id	Body temp	Aquatic	Aerial	Legs	Hibernates	Gives birth	Mammal
human	Warm	No	No	Yes	No	Yes	yes
Python	Cool	No	No	No	Yes	No	no
salmon	Cool	Yes	No	No	No	No	no
Whale	Cool	Yes	No	No	No	Yes	yes

The size of the hypothesis space

- Given that the number of possible values for each attribute are the following
  - Body Temp = 3, Aquatic=3, Aerial=3, Legs=3, Hibernates=3, Gives Birth=3 (recall that ? is a possible value for each attribute)

  the number of conjunctive hypotheses in H is

$$s = 3^6 = 729$$

- In general, the size s of H is exponential in the number n of attributes, i.e., $s = m^n$ - assuming that m is number of possible values for each attribute

Classification as search
Naive approach

- The Naïve approach is in general inefficient, because it requires to exhaustively enumerate all hypotheses in H for each training example
- **Remedy**: exploiting the partial order among hypotheses

Ordering the hypothesis space

- Given a training set S and hypotheses h and g over S, h ≥ g (read "h is more general than g" or "g is more specific than h") if

$$g(e)=1 \rightarrow h(e)=1$$

- that is, if e satisfies g then e satisfies h as well
- Intuitively, h covers more examples then g
- Syntactically, h ≥ g, with h = $<a_1, \ldots, a_n>$, g = $<b_1, \ldots, b_n>$ if • for each i=1,n, either a_i=? or $a_i=b_i$.
- (H, ≥) is a partial order (reflexive, antisymmetric and transitive properties)

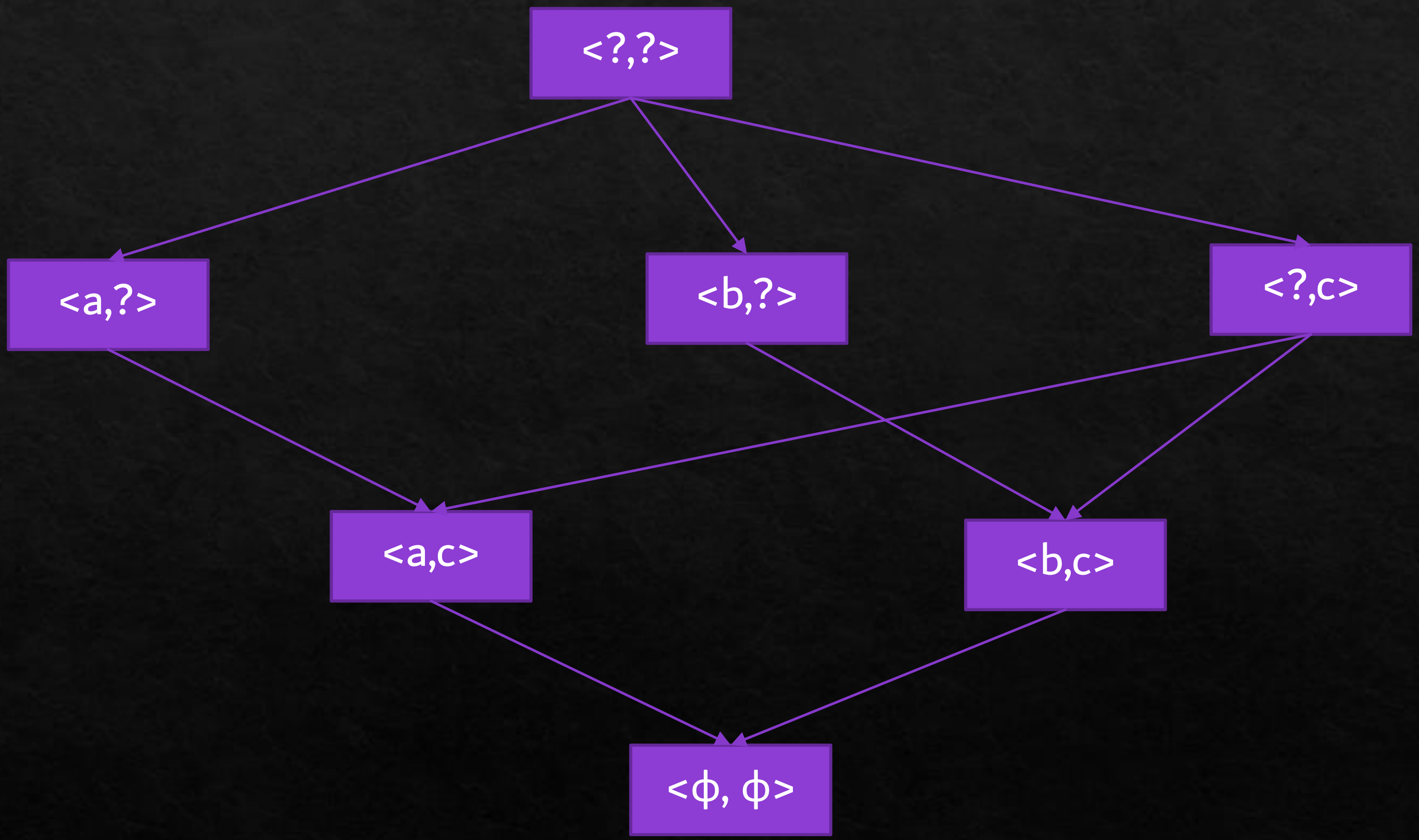
Ordering the hypothesis space

- h_1=<a,b,?,?,?,?>, h_2=<a,b,c,?,?,?>, h_3=<x,y,z,w,?,e> are such that
  - $h_1 \geq h_2$, and
  - h_3 is incomparable with both h_1 and h_2
- h=<?, ?, ?, ?, ?, ?> is the most general hypothesis – only don't care values, thus h is satisfied by any example
- h=<ϕ, ϕ, ϕ, ϕ, ϕ, ϕ> is the most specific hypothesis – no value allowed, thus h is satisfied by no example

Ordering the hypothesis space

- Minimal generalization: replace one value by “?”
  - h_1=<a,b,c,d,?,?> -> h_1=<a,b,?,d,?,?>
- Minimal specialization: replace “?” by a value
  - h_1=<a,b,c,d,?,?> -> h_1=<a,b,c,d,e,?>

The hypotheses space has a structure



- h=<x,y>
- x ∈ {a,b}
- y ∈ {c}

Training Examples for EnjoySport

Sky	Temp	Humid	Wind	Water	Forecast	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

- Days on which Aldo enjoys his favorite water sport
- Task: given the above training set, find a model, i.e., a hypothesis consistent with the training data to predict the value of ”Enjoy Sport” for an arbitrary day based on the values of the other attributes

Ordering the hypothesis space

Sky	Temp	Humid	Wind	Water	Forecast	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

- h: <sunny, warm, ?, ?, ?, ?>,
- g: <sunny, warm, ?, ?, warm,?>
- Syntactically, h ≥ g
- h is satisfied by examples 1,2 and 4
- g is satisfied by examples 1 and 2

By definition h ≥ g, where

- $h = <a_1, ..., a_n>$,
- $g = <b_1, ..., b_n>$

if for each i=1,n, either a_i=? or $a_i=b_i$

The Find-S Algorithm

- Starting from the most specific hypothesis, it examines all positive examples, one at a time – negative examples are simply ignored
- At each step, it performs a minimum generalization of the current hypothesis to make it consistent with the current example (and the previous ones)
- After having examined all positive examples, it outputs the most specific hypothesis h which is consistent with all positive examples (possibly <?,...,?>)
- if there is a conjunction h' consistent with the training set, then h=h'
- Otherwise, h will cover some negative examples – so it is not a model

The Find-S Algorithm

- Assumption: there exists a conjunctive model
- Input: the training set S; the space (H,≥) of all possible conjunctions over the examples in S
- Begin
  - h = <ϕ,..., ϕ>
  - For each positive example e in S
    - if e does not satisfy h then replace h by the next most general hypothesis according to ≥ that is satisfied by e
  - Output hypothesis h
- end

Find-S Algorithm
the EnjoySport example

More general

<ϕ, ϕ, ϕ, ϕ, ϕ, ϕ> S_0

<sunny,warm,normal, strong, warm,same>

<sunny,warm,normal,strong,warm,same> S_1

<sunny,warm,high, strong, warm,same>

<sunny,warm,?,strong,warm,same> S_2

<sunny,warm,high, strong, cool, change>

<Sunny, Warm, ?, Strong, ?, ? > S_3

Find-S Algorithm
the EnjoySport example

- Starting from <ϕ,..., ϕ>, FindS progressively generalizes the current hypothesis to make it consistent with all positive examples so far examined
- Eventually Find-S outputs
  - h = <Sunny, Warm, ?, Strong,?,? >
- h is a model for the EnjoySport dataset – WHY?
- h is the most specific model

Find-S Algorithm

- The conjunction <? ? 0> is a model for this training set
- FindS correctly computes the above model
  - h0 = <ϕ,ϕ,ϕ>
  - h1 = <0 1 0>
  - h2 = <? ? 0> -- (Z=0 or not Z)

X	Y	Z	C
0	0	1	0
0	1	0	1
1	0	0	1

Problems with Find-S Algorithm

Dataset 1			
X	Y	Z	C
0	0	1	0
0	1	0	1
1	0	0	1

Dataset 2			
X	Y	Z	C
0	0	0	0
0	1	0	1
1	0	0	1

Dataset	Conjunctive Model	FindS output
1	<?,?,0> -- Z=0	<?,?,0>
2	None	<?,?,0>

- Dataset 1 has a conjunctive model, which then is generated by FindS
- Dataset 2 has no conjunctive model
- The two training sets have the same positive examples
- Hence, FindS generates the same output <?,?,0> for both datasets (irrespective of negative examples)
- FindS over Dataset2 generates a hypothesis which is not a model as it covers the negative example – indeed a conjunctive model does not exist

Problems with Find-S Algorithm

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	Very high	Strong	Warm	Same	No

- No conjunctive hypothesis consistent with the training set exists
- Output FindS: h = <sunny, warm, ?, strong, warm, same >
- h not a model – satisfies one negative example (the red one)

FindS revisited

- FindS outputs the most specific conjunction h consistent with all positive examples in the training set S
- Of course, h is a model provided that it does not cover any negative examples

Find-S Algorithm Revisited

- Remove the assumption about the existence of a conjunctive model
- Input: the training set S; the space (H,≥) of all possible conjunctions over the examples in S
- begin
  - h = <ϕ, ϕ,…, ϕ>
  - for each positive example e in S
    - If e does not satisfy h then replace h by the next most specific hypothesis in H according to ≥ that is satisfied by e
  - if there is any negative example satisfying h then
    - Output “no conjunctive model exists”
  - else
    - Output hypothesis h
- end

Concluding Remarks on FindS

- Correctness: The output h (if any) is a model
- Completeness: If a (conjunctive) model h’ exists, then h=h’

Complaints with FindS

- It picks a maximally specific model (if any) - is this reasonable?
- In general, there might be several correct conjunctive hypothesis!
- Version Space

Version Space

- In general, there are more hypotheses that are consistent with the training set, e.g.,
  - h_0 = <Sunny, Warm, ?, Strong, ?, ? > provided by FindS
  - h_1 = <?, Warm, ?, Strong, ?, ? >
  - h_2 = <?, Warm, ?, ?, ?, ? >
- where $h_2 \geq h_1 \geq h_0$

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

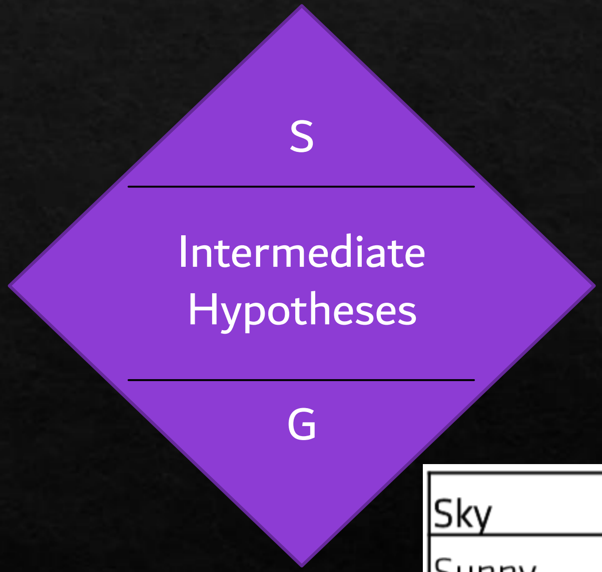
Version Space

- Given a hypothesis space H and training set S, the version space $VS_{H,S}$ is the set of all hypotheses in H consistent with S

$$VS_{H,S} = \{h \in H \mid consistent(h,S)\}$$

Representing Version Spaces

- Specific boundary S: most specific hypothesis in VS
- General boundary G: set of most general hypotheses in VS



S: <Sunny, Warm, ?, Strong, ?, ? >
G: <Sunny, ?, ?, ?, ?, ? >, <?, warm, ?,?,?,?>

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

Candidate Elimination algorithm

- The algorithm starts with
  - Specific boundary S = <ϕ, ϕ,..., ϕ>
  - General boundary G = {<?,?,...,?>}
- For each example e in the training set the hypotheses in S and G are modified in order to make them consistent with e
- In particular
  - hypotheses in G are specialized
  - hypotheses in S are generalized
- so restricting the version space to eliminate hypotheses not consistent with e

Candidate-elimination the EnjoySport example

S_0

<ϕ, ϕ, ϕ, ϕ, ϕ, ϕ>

1st positive example

e = <sunny,warm,normal, strong,warm,sam>

Is e consistent with G? YES– do nothing

Is e consistent with S? NO - generalize

G_0

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

Candidate-elimination
the EnjoySport example

<ϕ, ϕ, ϕ, ϕ, ϕ, ϕ> S_0

<sunny,warm,normal, strong, warm,same>

<sunny,warm,normal,strong,warm,same> S_1

<sunny,warm,high, strong, warm,same>

<sunny,warm,?,strong,warm,same> S_2

<sunny,warm,high, strong, cool, change>

<?,?,?,?,?,?> S_3

Minimal Generalizations

Candidate-elimination
the EnjoySport example

S

<sunny,warm,normal,strong,warm,same>

1st negative example
e=<Rainy,Cold,High,Strong,Warm,Change>

Is e consistent with G? YES– do nothing
Is e consistent with S? NO - generalize

G

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

Candidate-elimination
the EnjoySport example

S

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

1st negative example
e=<Rainy,Cold,High,Strong,Warm,Change>

{<sunny,?,?,?,?,?>, <?,warm,?,?,?,?>, <?,?,normal,?,?,?>,
<?,?,?,weak,?,?>, <?,?,?,?,cool,?>, <?,?,?,?,?,same>}

Every hypothesis in G must
- Not satisfy the negative example - TRUE
- Satisfy all positive examples - FALSE

G

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

Minimal specialization

Candidate-elimination the EnjoySport example

S

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

Not more general than S - remove

1st negative example
e=<Rainy,Cold,High,Strong,Warm,Change>

{<sunny,?,?,?,?,?>, <?,warm,?,?,?,?>, <?,?,normal,?,?,?>, <?,?,?,weak,?,?>, <?,?,?,?,cool,?>, <?,?,?,?,?,same>}

Every hypothesis in G must

- Not satisfy the negative example - TRUE
- Satisfy all positive examples - FALSE

Minimal specialization

G

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

Candidate-elimination the EnjoySport example

S

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

1st negative example
e=<Rainy,Cold,High,Strong,Warm,Change>

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

Every hypothesis in G must
- Not satisfy the negative example - TRUE
- Satisfy all positive examples - FALSE

So it is a model

G

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

Minimal specialization

Candidate-elimination the EnjoySport example

S <sunny,warm,?,strong,warm,same>

G {<sunny,?,?,?,?,?>, <?,warm,?,?,?,?>, <?,?,?,?,?,same>}

Candidate-elimination the EnjoySport example

S

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

3rd positive example
e=<Sunny,Warm,High,Strong,Cool,Change>

is e consistent with S? NO - generalize

G

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

Candidate-elimination
the EnjoySport example

S

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

3rd positive example
e=<Sunny,Warm,High,Strong,Cool,Change>

G

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

Candidate-elimination
the EnjoySport example

S

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

3rd positive example
e=<Sunny,Warm,High,Strong,Cool,Change>

not more general
than s - remove

G

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

Candidate-elimination the EnjoySport example

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

No more example

G {<sunny,?,?,?,?,?>, <?,warm,?,?,?,?>

Candidate-elimination
the EnjoySport example

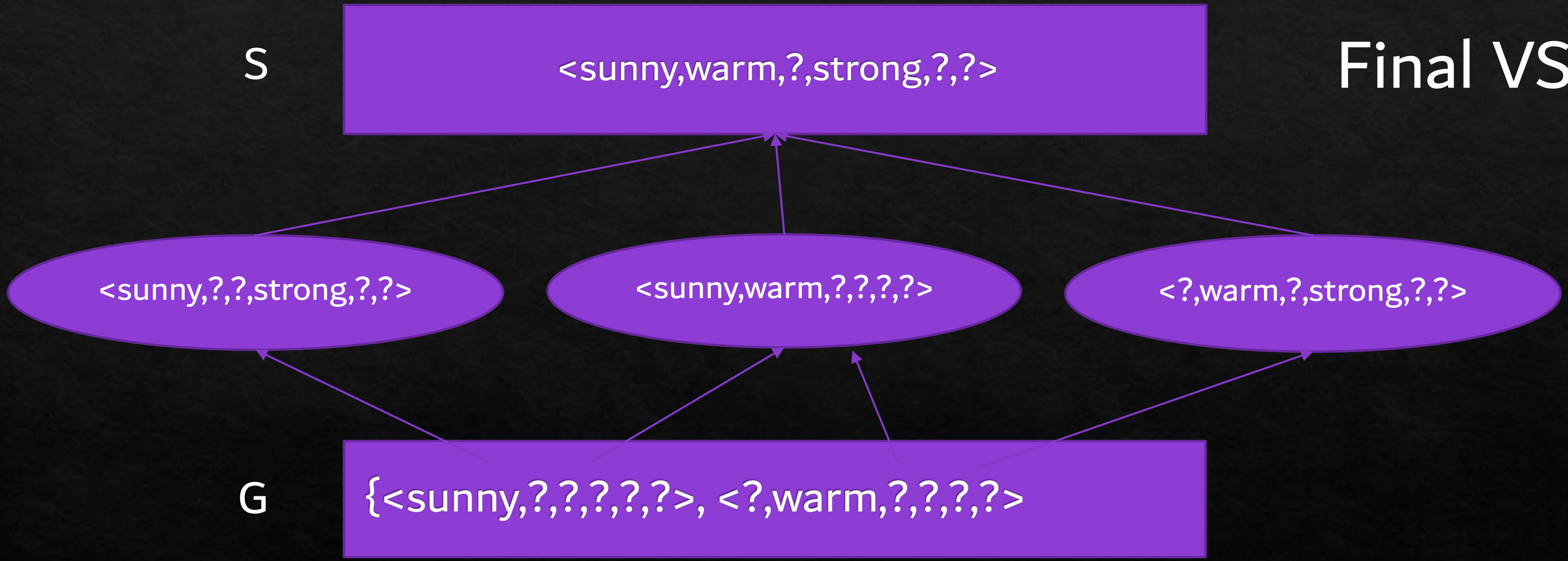
S <sunny,warm,?,strong,?,?>

S is the Specific boundary and G the General boundary of the version space

G {<sunny,?,?,?,?,?>, <?,warm,?,?,?,?>

Candidate-elimination the EnjoySport example

Final VS



All possible hypotheses compatible with the examples

Version Space Theorem

- Consider 3 hypotheses from our example VS
  - s=<sunny, warm, ?, strong, ?, ?> in S
  - g=<sunny, ?, ?, ?, ?, ?> in G
  - h=<sunny, ?, ?, strong, ?, ?>
  - $g \geq h \geq s$
- $h \geq s$ -> any (positive) examples classified by s is classified by h
- $g \geq h$ -> any (negative) examples not classified by g is not classified by h
- Since s and g are both consistent, h is consistent as well
- Every hypothesis “in between” S and G is consistent, i.e., is a model

Candidate elimination algorithm

- Candidate Elimination
- S={<ϕ … ϕ>}; G{<? …, ?>}
- for each training example e in the training set do
- If e is a positive example
  - If s ∈ S is s.t. s(e)=0 (i.e., s is not consistent with e)
    - Perform a minimal generalization h of s such that h(e)=1 (h is consistent with e)
    - If exists g ∈ G s.t. not g ≥ h remove g from G

Candidate elimination algorithm (cont'ed)

- If e is a negative example
  - If s ∈ S is s.t. s(e)=1 (i.e., s is not consistent with e) remove s from S
  - else for each hypothesis g ∈ G s.t. g(e)=1 (i.e., g is not consistent with e)
    - Perform all minimal specializations h of g such that h(e)=0 (h is consistent with e)
    - If exists g ∈ G s.t. not g ≥ s remove g from G
- } //end for
- Output S,G

Candidate elimination
An Example

The target function is a boolean function

X	Y	Z	C
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0

S0 = {<ϕ , ϕ , ϕ >}
G0 = {<?,?,?>}

S1 = {<0, 0, 0>}
G1 = {<?,?,?>}

S2 = {<0,0,0>}
G2= {<?,?,0>} //removed <1,?,?>and <?,1,?> from G2

S3 = {<0,?,0>} C= not X not Z
G3= {<?,?,0>} C= not Z

- The algorithm provides two conjunctions compatible with the training data
- Since no conjunction does exist in between S3 and G3 (G3 is a minimal generalization of S3)

VS=S3 U G3

Candidate elimination
An Example

The target function is a boolean function

X	Y	Z	C
1	0	1	0
0	0	0	1
1	1	0	1
1	0	0	1

- Find final S and G of the given training set.
- Can we infer something from the result? Can we find a model for that data?

Candidate elimination
An Example

The target function is a boolean function

X	Y	Z	C
1	0	1	0
0	0	0	1
1	1	0	1
1	0	0	0

- Find final S and G of the given training set.
- Can we find a conjunctive hypothesis?