Concept Learning
[IIA – Lect._____]

Alessio Micheli

micheli@di.unipi.it

Dipartimento di Informatica
Università di Pisa - Italy

Computational Intelligence & Machine Learning Group

DRAFT, please do not circulate! *2023*

Overview

Dip. Informatica
University of Pisa

- Concept Learning from examples as research in the Hypothesis Space
  - an opportunity to have a look of a simple symbolic learning system (propositional) (rule based)
  - as the basis for the subsequent introduction of flexible models

- Discrete space (symbolic representation of the target function), structure of the hypothesis space
- Representation: examples: Conjunction of literals
- Search Alg.: Find-S, candidate elimination
- Inductive Bias

Learning Problem (*alla* Mitchell)

Dip. Informatica
University of Pisa

- Learning as *Improving with experience at some task*

- Improve over task T,
- with respect to performance measure P,
- based on experience E.

Tasks: Supervised Learning

Dip. Informatica
University of Pisa

- Given: Training examples as <input,output> = <***x***,*d*>

  for an unknown function *f* (known only at the given points of example)
  - Target value: desiderate value *d* or *t* or *y* … is given by the teacher according to *f*(***x***) to label the data
- Find: A *good* approximation to *f* (i.e. a hypothesis h that can used for prediction on unseen data ***x***′)

- Target *d* (or *y or t*): a categorical or numerical label
  - *Classification*: *f*(***x***) returns the (assumed) correct class for ***x***

    f(***x***) is a *discrete-valued function*
  - *Regression*: approximate a real-valued target function (in R o R^K)

NOW

Note on notation: in the following we follow the Mitchell with slight differences w.r.t the rest of the course. E.g. c wil be target function, vectors ***x*** *will be not in bold in the following, not italic for c, h, x …*

Concept Learning

Dip. Informatica
University of Pisa

Def

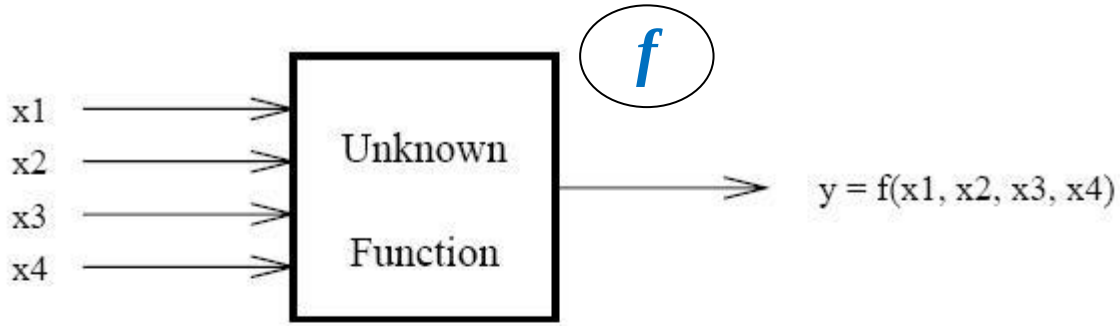
- **Concept Learning** : inferring *a Boolean function* from positive and negative training examples
- c: X → {t,f} or {+,-} or {yes, no} or {0,1}, X: *instance space*
- Examples of concepts: category "cat" or "enjoy-sport"

Defs

- **Def:** "Example": <x,c(x)> (or <x,d>) in D (or *TR set*)
- **Def:** h:X→{0,1} **satisfies** x if h(x)=1
- **Def:** a hypothesis h is **consistent** with
  - An example <x,c(x)>, x in X, if h(x)=c(x)
  - D, if h(x)=c(x) for each training example <x,c(x)> in D.

An example: Learning Boolean functions

Dip. Informatica
University of Pisa



Find the function s.t.

Example	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

Table 1

This is an **ill posed** (inverse) problem:
We may violate either
existence, uniqueness,
stability of the solution or solutions

Learning Boolean functions: ill-posed

Dip. Informatica
University of Pisa

- There are $2^{16} = 2^{2^4} = 65536$ possible Boolean functions over four input features. We can't figure out which one is correct until we've seen every possible input-output pair.
- After 7 examples, we still have 2^9 possibilities.

x_1	x_2	x_3	x_4	y
0	0	0	0	?
0	0	0	1	?
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	?
1	0	0	0	?
1	0	0	1	1
1	0	1	0	?
1	0	1	1	?
1	1	0	0	0
1	1	0	1	?
1	1	1	0	?
1	1	1	1	?

Def.

In the general case: $|H| = 2^{\#\text{-instances}} = 2^{2^n}$

for binary inputs/outputs, *n= input dimension*

***Lookup table** model* ⟶

Guide to next solutions

Dip. Informatica
University of Pisa

- **Work with a restricted hypothesis space:** we start choosing a space of hypotheses *H* that is considerably smaller than the space of all possible functions! (*language bias*)

- We will see:
- **(Simple) conjunctive rules** (in a finite discrete H)
- **Linear functions** (in an infinite continuous H)

Next slides

Dip. Informatica
University of Pisa

- How to organize and efficiently search through a (discrete) space of hypothesis: some algorithms for a very restricted set of hypotheses
  - **Conjunctive rules**
  - **No noisy data**

- Repetita on the notation: in this lecture we make an exception, as we follow the Mitchell's book notation. However, this is irrelevant with respect to the main track of the course (majority of the lectures).

Conjunctive Rules

Dip. Informatica
University of Pisa

- How many distinct h? = How many simple conjunctive rules (with "and") in the example of Table 1?
  - 4 inputs: True, 4 single, 6 couple, 4 triple, 1 quadruple: 16

- In the general case:
- Positive literals, e.g.: $h_1 = l_2$, $h_2 = (l_1$ ***and*** $l_2)$, $h_3 = true$, …
  - **Simple conjunctive rules**
  - $|H| = 2^n$ (image l_i as a bit of a string of length n)
- Literals (also ***not***(l_i))
  - $|H| = 3^n+1$ (why ? : **Exercise 1**)

- **Exercise 2**: Can you find "simple conjunctive rules" *consistent* with table 1 ? (hint: *h* is not consistent if you find a counterexample in Table 1)

Training Examples for Concept Enjoy Sport

Dip. Informatica
University of Pisa

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

Target

Sky	Temp	Humid	Wind	Water	Fore-cast	Enjoy Sport
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

example

Representing Hypothesis

Dip. Informatica
University of Pisa

- Hypothesis *h* is a conjunction of constraints on attributes
- 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 (e.g.* l_i *AND* **not**(l_i)*)*
- Example: hypothesis *h* (h in the following)

Sky Temp Humid Wind Water Forecast

< Sunny ? ? Strong ? Same> ⟶ *Note: this is a function h, not an input example!*

Corresponding to boolean function:

Sky=Sunny ∧ (and) Wind=Strong ∧ (and) Forecast=Same

< Ø Ø Ø Ø Ø Ø > Most specific

< ? ? ? ? ? ? > Most general

Prototypical Concept Learning Task

Dip. Informatica
University of Pisa

Def

Given:

- **Instances** X : Possible days decribed by the attributes *Sky, Temp, Humidity, Wind, Water, Forecast*
- **Target function** c: EnjoySport X $\rightarrow$ {0,1}
- **Hypotheses H**: conjunction of (a finite set of) literals e.g.

  < Sunny ? ? Strong ? Same > $\longrightarrow$ h is a function (x an input)
- **Training examples** D : positive and negative examples of the target function: $<x_1, c(x_1)>, \ldots, <x_n, c(x_l)>$ (l examples)

Find:

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

Learning: searching in the hypotheses space H

Inductive Learning Hypothesis

Dip. Informatica
University of Pisa

- Assumpition here (in this lecture): ”Any hypothesis found to approximate the target function well over the training examples, will also approximate the target function well over the unobserved examples”.

- h(x)=c(x) for all x in D (i.e. *consistent* with D)
- h(x)=c(x) for all x in X (?)

- (?) = I.e. The Foundamental issue of the ML:
  - theoretical and empirical analysis

Number of Instances, Concepts, Hypotheses

Dip. Informatica
University of Pisa

- Sky: Sunny, Cloudy, Rainy (3 values!!!)
- AirTemp: Warm, Cold
- Humidity: Normal, High
- Wind: Strong, Weak
- Water: Warm, Cold
- Forecast: Same, Change

- The representation chosen for H determines the search space

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

#distinct concepts : $2^{96} = 2^{\#\text{-instances}}$

sunny/cloudy/rainy/?/Ø

warm/cold/?/Ø

#syntactically distinct hypotheses (conjunctions): 5*4*4*4*4*4=5120

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

(because all h with Ø are equivalen to "*false*")

In general: High dimension, even infinite: Structuring the search space may help in searching more efficiently!

General to Specific Ordering

Dip. Informatica
University of Pisa

- Consider two hypotheses:
  - h_1=< Sunny,?,?,Strong,?,?>
  - h_2=< Sunny,?,?,?,?,?>
- Set of instances covered by h_1 and h_2:

h_2 imposes fewer constraints than h_1 and therefore classifies more instances x as positive [i.e. h(x)=1].

Def

Def: 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 (written $h_j \geq h_k$) if and only if

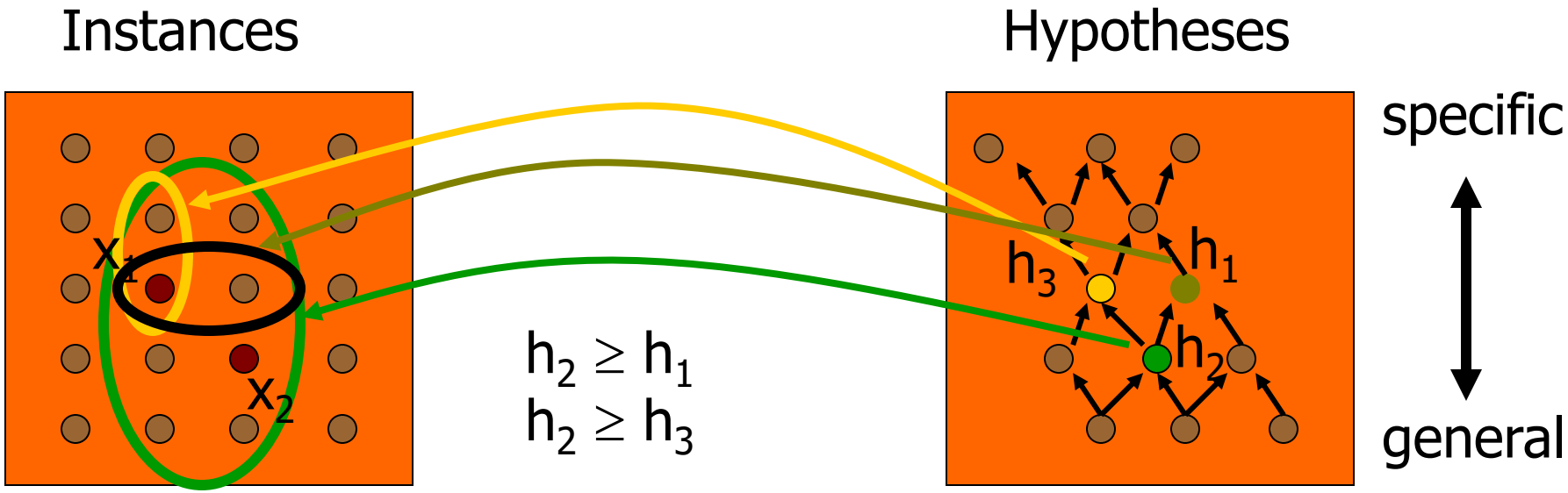
$$\forall x \in X : [(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$

Examples over binary l_i : $l_1 \geq (l_1 \, and \, l_2)$, l_1 versus l_2 : not comparable

- The relation $\geq$ imposes a partial order (p.o.) over the hypothesis space H that is utilized by many concept learning methods.
- We can take advantage of this p.o. to efficently organize the search in H

A structure over H (partial order)

Dip. Informatica
University of Pisa



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,?>

Find-S Algorithm

Dip. Informatica
University of Pisa

Exploit m.g.t. partial order to efficiently search consistent h (*without explicitly enumerating* every h in H)

Def

1. Initialize h to the most specific hypothesis in H
2. For each **positive** training instance x
   - For each attribute a_i in h

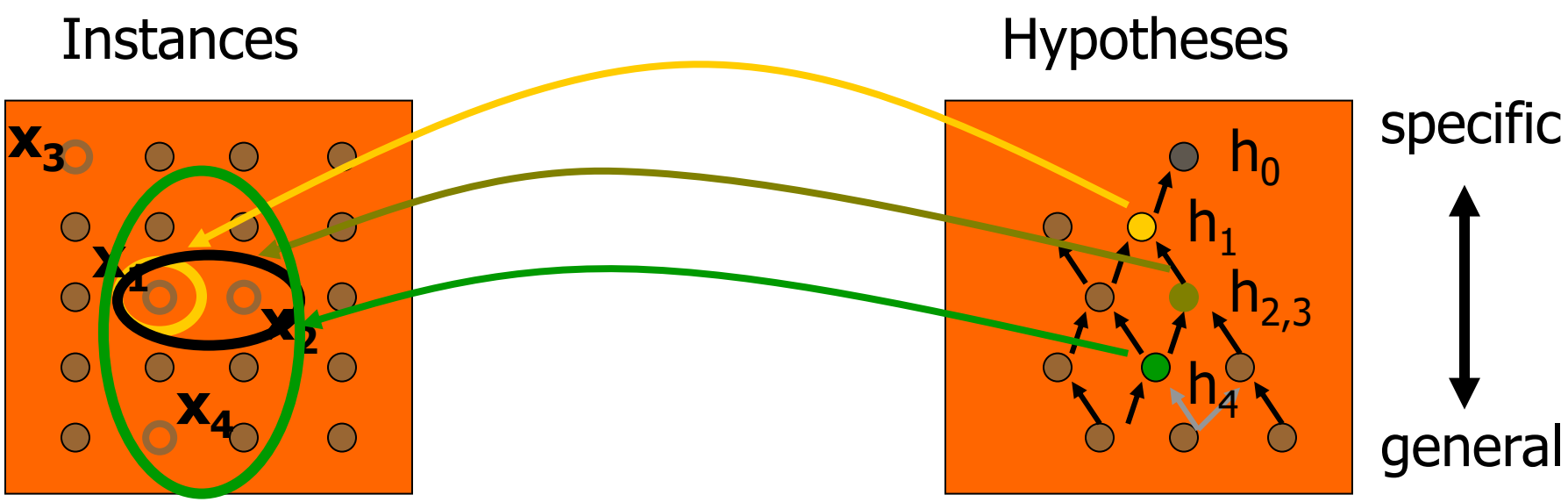
     If the a_i in h is satisfied by x

     then do nothing

     else replace a_i in h by the next more general constraint that is satisfied by x *[e.g. remove from h literals not satisfying x]*
3. Output hypothesis h

Hypothesis Space Search by Find-S

Dip. Informatica
University of Pisa



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=< Ø, Ø, Ø, Ø, Ø, Ø,>

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

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

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

Properties of Find-S

Dip. Informatica
University of Pisa

- Hypothesis space described by conjunctions of attributes (hard limitations !)
- Find-S will output the most specific hypothesis within H that is consistent with the positive training examples
- The output hypothesis *h* will also be consistent with the negative examples, provided the target concept is contained in H.
  - Because $c \geq h$

Complaints about Find-S

Dip. Informatica
University of Pisa

- 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.
- Can't tell when training data is inconsistent, as it ignores negative training examples: *no noise tollerance !*
- **Why prefer the most specific hypothesis?**
- **What if there are multiple maximally specific hypothesis?**

Version Spaces

Dip. Informatica
University of Pisa

Key idea: ouput a description of the set of *all* h consinstent with D

We can do it without enumerating them all

- Consistent(h,D) := $\forall <x,c(x)> \in D \;\; h(x)=c(x)$

Def

- 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:

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

List-Then Eliminate Algorithm

Dip. Informatica
University of Pisa

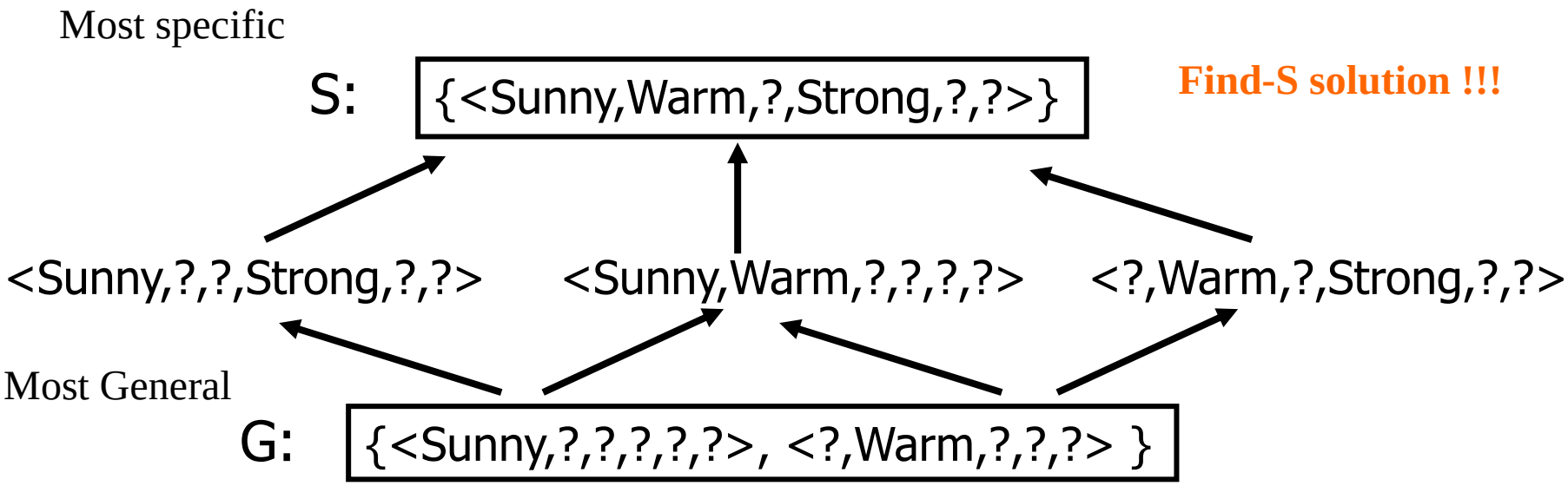
1. *VersionSpace* $\leftarrow$ a list containing every hypothesis in H
2. For each training example <x,c(x)>

   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*

Irrealistic: exuastive enumeration of all h in H

A better representation for the Version Space (by G and S)

Dip. Informatica
University of Pisa



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> +

Exercise: check that all the h in VS are consistent with D

Representing Version Spaces

Dip. Informatica
University of Pisa

Def

- The **general boundary**, G, of version space $VS_{H,D}$ is the set of maximally general members (of H consistent with D).
- The **specific boundary**, S, of version space $VS_{H,D}$ is the set of maximally specific members (of H consistent with D).
- **Theorem**: Every member of the version space lies between these boundaries

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

where $x \geq y$ means x is more general or equal than y

Proof: see cap 2 Mitchell

We can make *a complete search* of consistent hypotheses using the two boundaries S and G for the VS,
i.e. not restricted to S as for Find-S

Candidate Elimination Algorithm

Dip. Informatica
University of Pisa

$G \leftarrow$ maximally general hypotheses in H

S $\leftarrow$ 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 (def. of VS)

For each hypothesis s in S that is not consistent with d

[generalize S]

- remove s from S.
- Add to S all minimal generalizations h of s such that
  - h consistent with d
  - Some member of G is more general than h ($\leftarrow^*$)
- Remove from S any hypothesis that is more general than another hypothesis in S

Candidate Elimination Algorithm

Dip. Informatica
University of Pisa

If d is a negative example

Remove from S any hypothesis that is inconsistent* with d [*h is 1*]

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

[specialize G]

- remove g from G.
- Add to G all minimal specializations h of g such that
  - h consistent with d
  - Some member of S is more specific than h (←* key point in the example later)
- Remove from G any hypothesis that is less general than another hypothesis in G

Example Candidate Elimination

Dip. Informatica
University of Pisa

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

Dip. Informatica
University of Pisa

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,?,?,?> }

SPECIALIZE **G**: Cerco le "minimal specilization to make x3 negative" (s.t some member of S is more specific than h)

x4: Non è più consistente con G (non dà + su x4) (primo statement dell'alg.)

Exercise

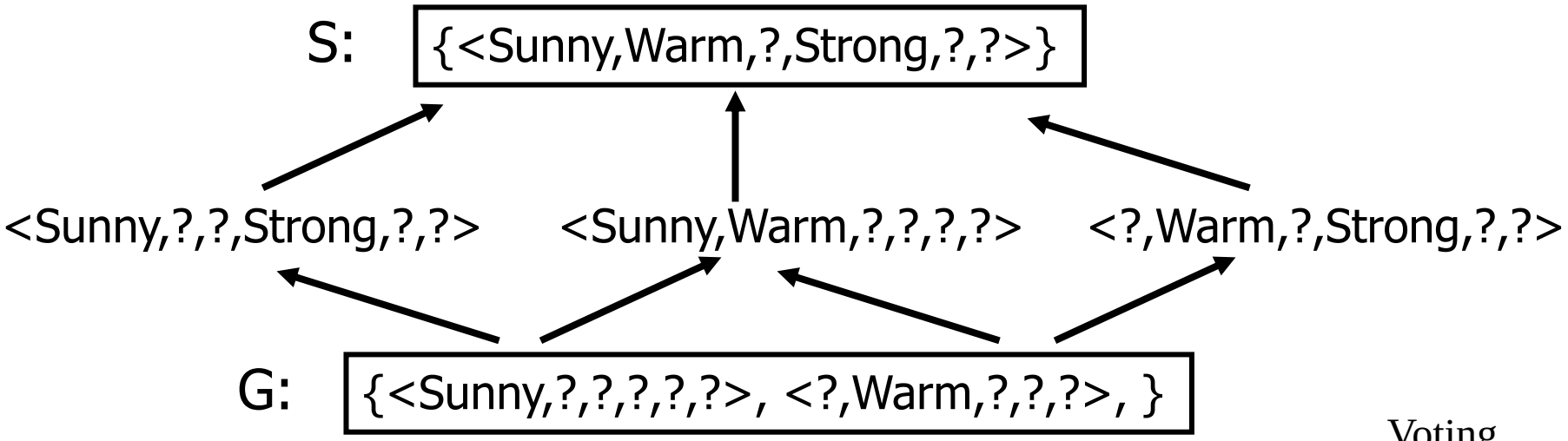
Dip. Informatica
University of Pisa

Exercise: try find-s and cand. elimination on the example slide 7 (table 1).

Add "not(l_i)" to the space of Simple conjunctive rules (see <Conjunctive Rules> slide),
noting that 0 → not(l_i), 1 → l_i, ? → there is no literal.

Classification of New Data

Dip. Informatica
University of Pisa



Voting

x_5 = <Sunny Warm Normal Strong Cool Change> +6/0
x_6 = <Rainy Cold Normal Light Warm Same> - 0/6
x_7 = <Sunny Warm Normal Light Warm Same> ? 3/3
x_8 = <Sunny Cold Normal Strong Warm Same> ? 2/4

Rejection

Majority vote

Note: x in D (TR set) are not useful to evaluate

Inductive Bias and its role

Biased Hypothesis Space

Dip. Informatica
University of Pisa

- Our hypothesis space is unable to represent a simple disjunctive target concept : (Sky=Sunny) v (Sky=Cloudy)
- Candidate elimination on:

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 : {}

Bias: We assume that the hypothesis space H contain the target concept c. In other words that c can be described by a conjunction (by AND operator) of literals.

Unbiased Learner

Dip. Informatica
University of Pisa

- Idea: Choose H that expresses every teachable concept, that means H is the set of all possible subsets of X: 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 ? ? ?> v <? ? ? ? ? Change>
- H surely contains the target concept.

- *What for generalitazion ?*

Unbiased Learner

Dip. Informatica
University of Pisa

What are S and G in this case?

Assume positive examples (x_1, x_2, x_3) and negative examples (x_4, x_5)

$$S : \{ (x_1 \vee x_2 \vee x_3) \} \qquad G : \{ \neg (x_4 \vee x_5) \}$$

The only examples that are unambiguolsy classified are the training examples themselves (H can represent them).
In other words in order to learn the target concept one would have to present every single instance in X as a training example.

Def

Property: An unbiased learner is *unable to generalize*:
Proof: Each unobserved instance will be classified positive by precisely half the hypothesis in VS and negative by the other half (*rejection*).
Indeed:
$\forall$h consistent with x_i (test), $\exists$ h' identical to h except $h'(x_i) <> h(x_i)$, $h \in VS \rightarrow h' \in VS$ (they are identical on D)

Futility of Bias-Free Learning

Dip. Informatica
University of Pisa

- A learner that makes no prior assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances.
- (Restriction, preference) bias not only assumed for efficiency, it is *needed for generalization*
  - However, does no tell us (quantify) which one is the best solution for generalization

- **Trivial Example** (TR= Training Set, TS= Test Set):

	X	c(x)	H={x, not(x), **0**, **1**}
TR	0	0	VS={x,**0**}
TS	1	? →	Can be 1 or 0 … Unless you use all X as TR set.

- Issue: characterize the bias for different learning approaches

Inductive Bias

Dip. Informatica
University of Pisa

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 by L after training on D_c.

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

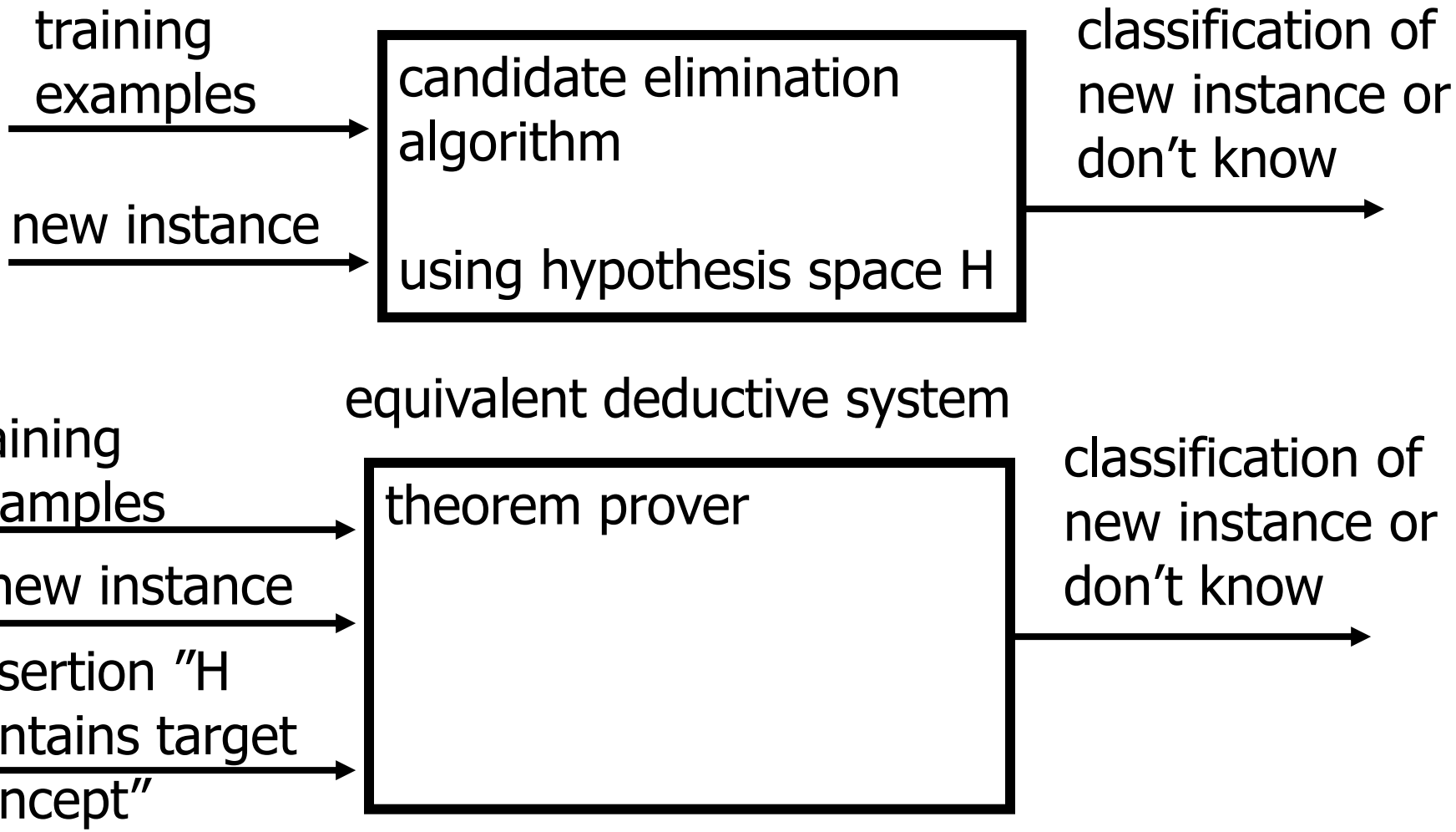
Def

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

Where A |— B means that A logically entails B (follows deductively from).

Inductive Systems and Equivalent Deductive Systems

Dip. Informatica
University of Pisa



Three Learners with Different Biases

Dip. Informatica
University of Pisa

- Rote learner (*lookup table*): Store examples, classify x if and only if it matches a previously observed example.
  - No inductive bias → no generalization
- Version space candidate elimination algorithm.
  - Bias: The hypothesis space contains the target concept (conj. of attributes) |H|= 973 versus 10^{28}.
- 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 (seen as positive examples).
  - In other words: we have a *language bias* due to the AND on the literals plus the *search bias* due to the preference of the most specific hyphotesis

Other symbolic - rule based approaches

Dip. Informatica
University of Pisa

Overcome the conjunctive restriction toward *flexible* models:

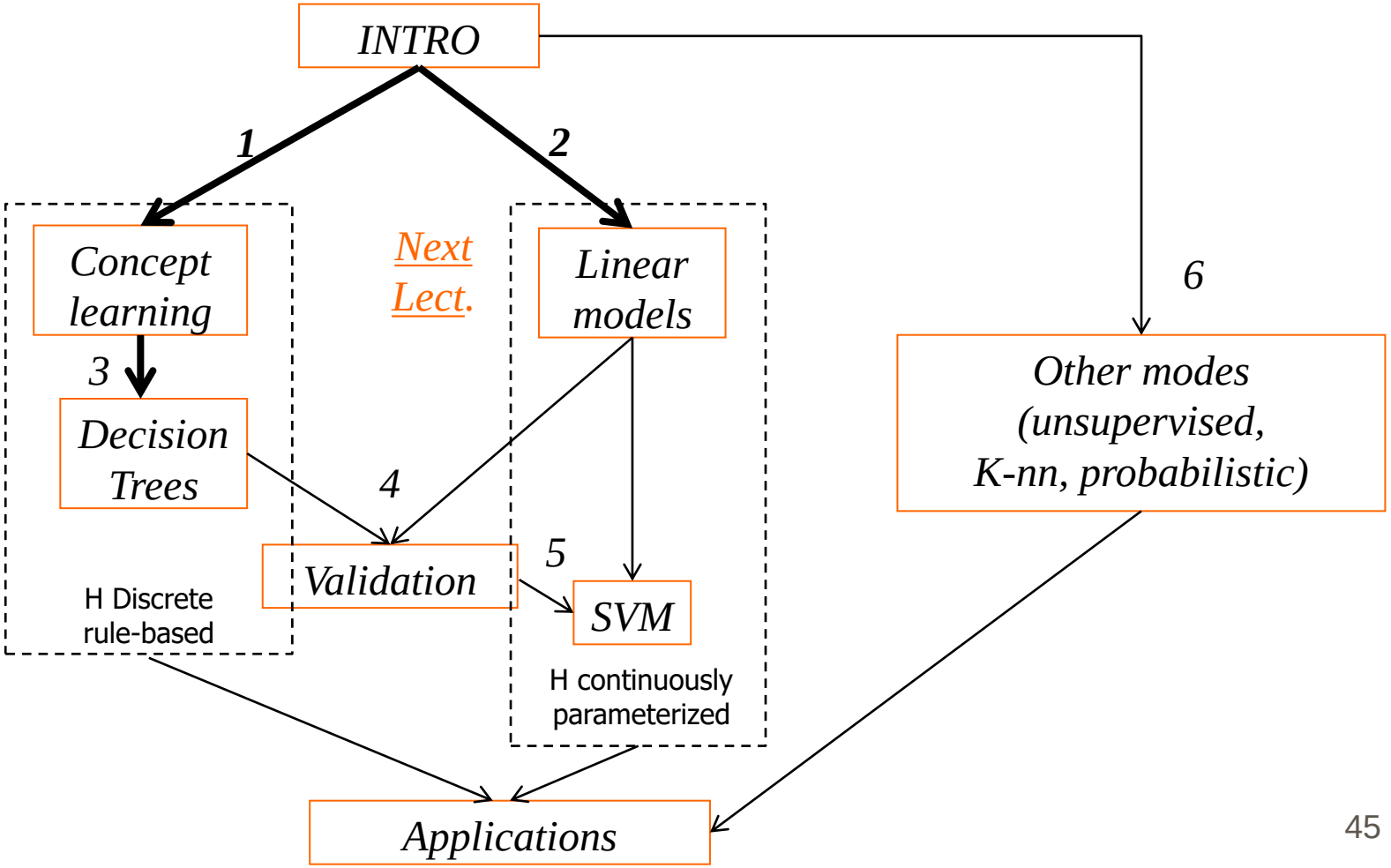
Many approaches in ML, e.g.:

- **Decision trees (a future lecture) (chap. 3 Mitchell)**
- Genetic Algorithms:
  - encode each rule set as a bit string and uses genetic search operator to explore this H
- Inductive Logic Programming (chap. 10 Mitchell):
  - First-order logic rules that contain variables (much more expressive than propositional rules)
  - Automatic inferring *Prolog* programs from examples (collection of Horn clauses)

In the course flow

Dip. Informatica
University of Pisa

In the course: example of *symbolic rules*, now (next lecture) we move to *numerical eq.* language (H continuous space) with *linear models* [see 2 parallel tracks]



Bibliographic references (this lecture)

Dip. Informatica
University of Pisa

- T. M. Mitchell, *Machine learning*, McGraw-Hill, 1997:
  chapter 1 e 2 .

- Other: AIMA ed. III: 19.1