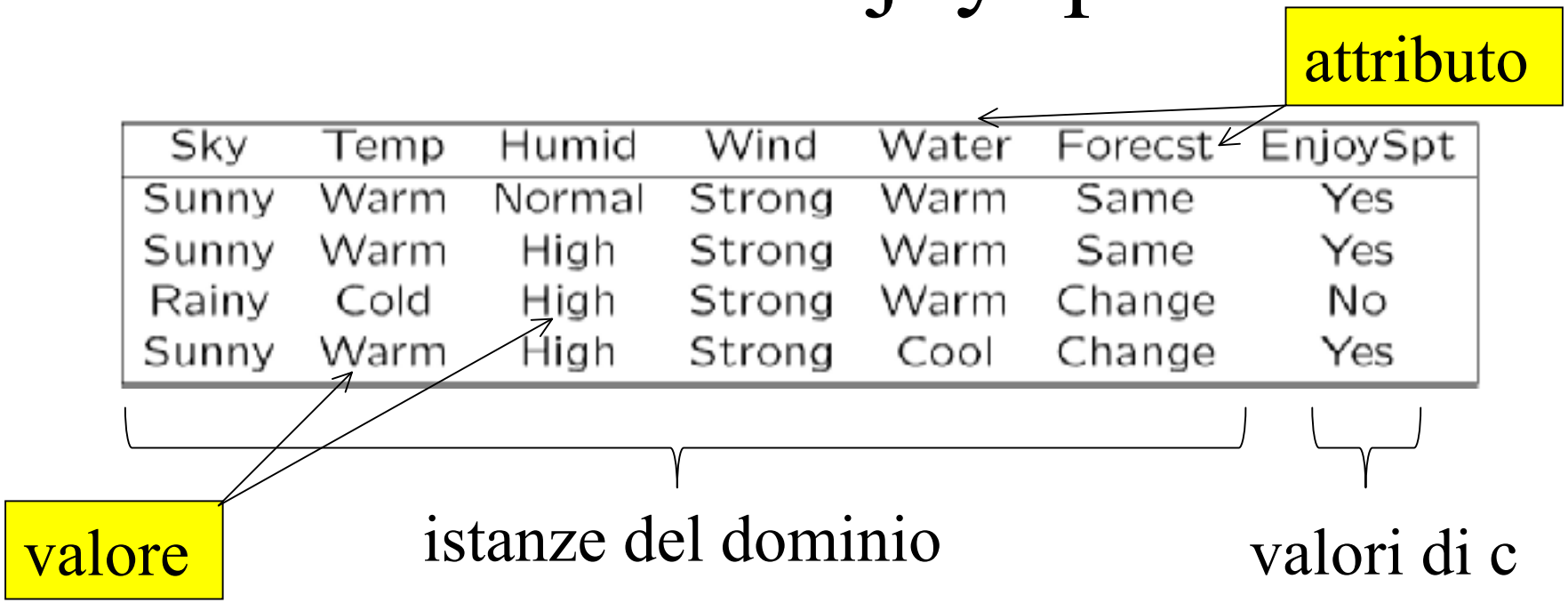
Apprendimento di Concetti da Esempi

Concept Learning

Che cos'è l'apprendimento di concetti

- Inferire una funzione booleana (**funzione obiettivo** o **concetto**) a partire da esempi di addestramento dati da coppie (input, output) della funzione stessa
  - Indicata con c o f
- $c : X \rightarrow \{ 0, 1 \}$
- X è l'insieme delle istanze del dominio
- $x \in X$ è una generica istanza
- L'insieme di addestramento D è un insieme di coppie $(x, c(x)) \in X \times \{ 0, 1 \}$ dove:
  - x è un esempio **positivo** se c(x) = 1, **negativo** altrimenti

Esempi di Addestramento per il concetto EnjoySport



Qual è il concetto generale?

Assunzione: gli esempi non contengono errori (**non sempre vero!**)

Un altro esempio: apprendere un profilo di utente per web browsing

D=

	Dominio	Piattaforma	Browser	Giorno	Schermo	Continente	**Click?**
Esempi D	edu	Mac	Net3	Lu	XVGA	America	Si
	com	Mac	NetCom	Lu	XVGA	America	Si
	com	PC	IExpl	Sab	VGA	Asia	No
	org	Unix	Net2	Gio	XVGA	Europa	Si

- Assunzione: gli esempi non contengono errori (NON SEMPRE VERO!!!)
- “Click” è il nostro concetto c ed assume valori in {0,1}

Quali sono le ipotesi per il concetto c?

- Obiettivo: determinare una **ipotesi** h tale che: $h(x)=c(x)\ \forall x$ in X
  - Ma noi conosciamo solo i valori assunti da c per gli esempi di addestramento in D
- L'insieme delle possibili ipotesi è detto **spazio delle ipotesi** H
- Molte possibili rappresentazioni per l'ipotesi h (dipende dalla rappresentazione di c)

Algoritmi di apprendimento da esempi

- Due semplici algoritmi
  - **Find-S** e **Version Space**
- Si applicano a:
  - funzioni booleane $c : X \rightarrow \{ 0,1 \}$
  - *rappresentazione monomiale*, cioè congiunzioni di k letterali dove k è compreso fra 1 ed n, n è il numero di attributi (*feature*) utilizzati per rappresentare ogni istanza x di X
  - es.: $\overline{a_1} \wedge a_2 \wedge a_5$ esprimibile anche come un vettore $(1, 0, ?, ?, 1)$

Rappresentare le ipotesi

- Supponiamo che le ipotesi h in H abbiano la forma di congiunzione di k letterali ($k \leq 6$ nell'esempio EnjoySport)
- Ogni letterale può essere:
  - Un valore specifico dell'attributo (es. Sunny, Rainy per l'attributo Sky)
  - Un "don't care" (?)
  - Non può assumere valori ($\varnothing$)
- Es.
  - h = (Sunny, ?, ?, Strong, ?, ?)
- Quante ipotesi posso formulare?
  - $|H| = (|Sky|+2)*(|Temp|+2)*(|Humid|+2)*(|Wind|+2)*(|Water|+2)*(|Forecst|+2)$
    $= 4*4*4*4*4*4$

Ipotesi “eccellenti”

- L’ipotesi più generale (tutti gli esempi sono positivi):
  - (?, ?, ?, ?, ?, ?)
- L’ipotesi più specifica (nessun esempio è positivo):
  - (∅, ∅, ∅, ∅, ∅, ∅)

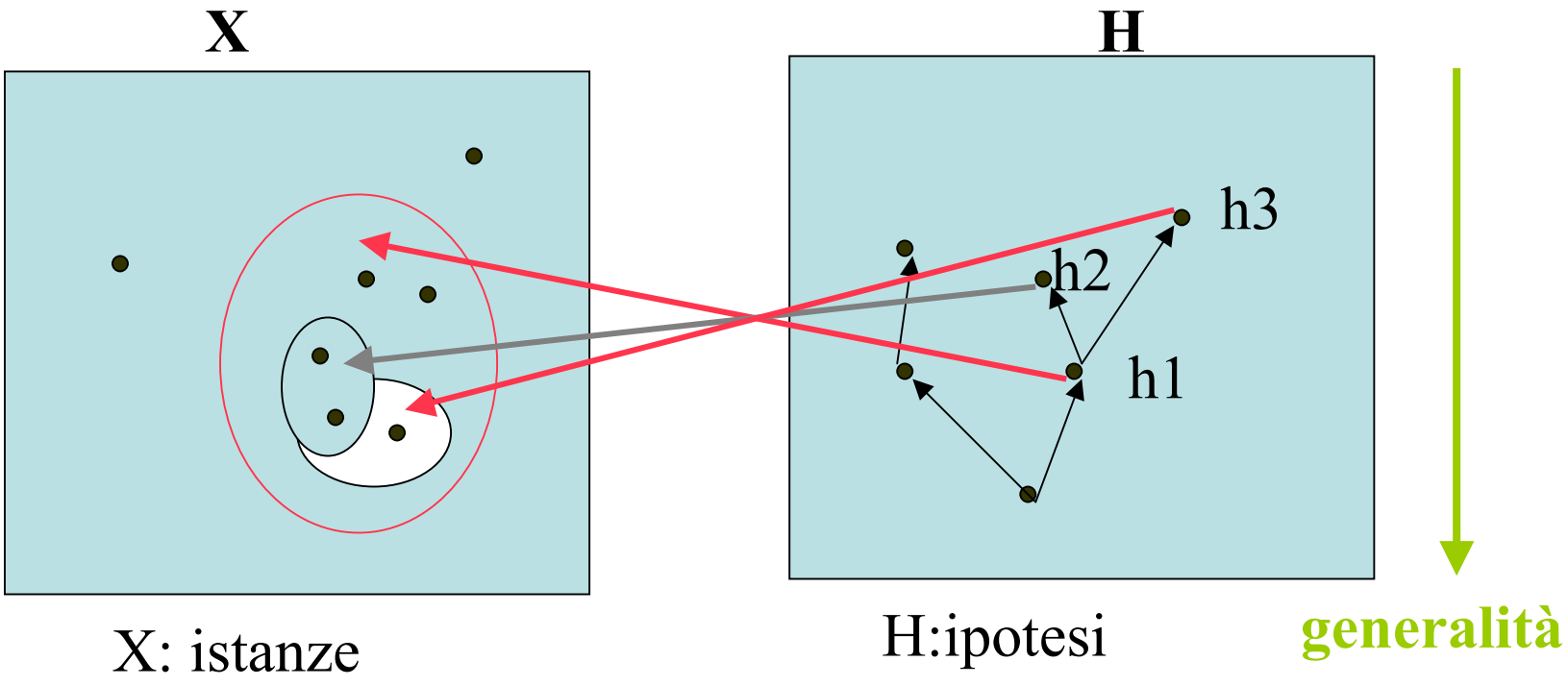
Ipotesi dell'apprendimento induttivo

- La nostra ipotesi h può garantire la coerenza con gli esempi di addestramento D, ma non necessariamente con tutte le possibili istanze in X
- Ogni ipotesi che approssima il comportamento della funzione obiettivo su un numero "sufficentemente grande" di esempi lo approssimerà sufficentemente anche su campioni non osservati
- Perché ciò è vero? Dalla teoria statistica del campionamento, (per estrarre parametri di popolazioni da campioni), e dalla *computational learning theory* (vedremo tutto questo più avanti nel corso)

Apprendimento di concetti=ricerca nello spazio delle ipotesi

- La ricerca può essere aiutata ordinando parzialmente le ipotesi
- Un ordinamento può essere fatto sulla base della generalità delle h
  - Def. $h_1 \geq_{gen} h_2 \;\; iff \;\; \forall x \in X, h_2(x) = 1 \rightarrow h_1(x) = 1$
  - h_2 è vera solo se lo è h_1
- Es.:
  - h_1 =(Sunny, ?, ?, ?, ?, ?)
  - h_2 =(Sunny, ?, ?, Strong, ?, ?)

Ordinamenti per generalità sullo spazio delle ipotesi



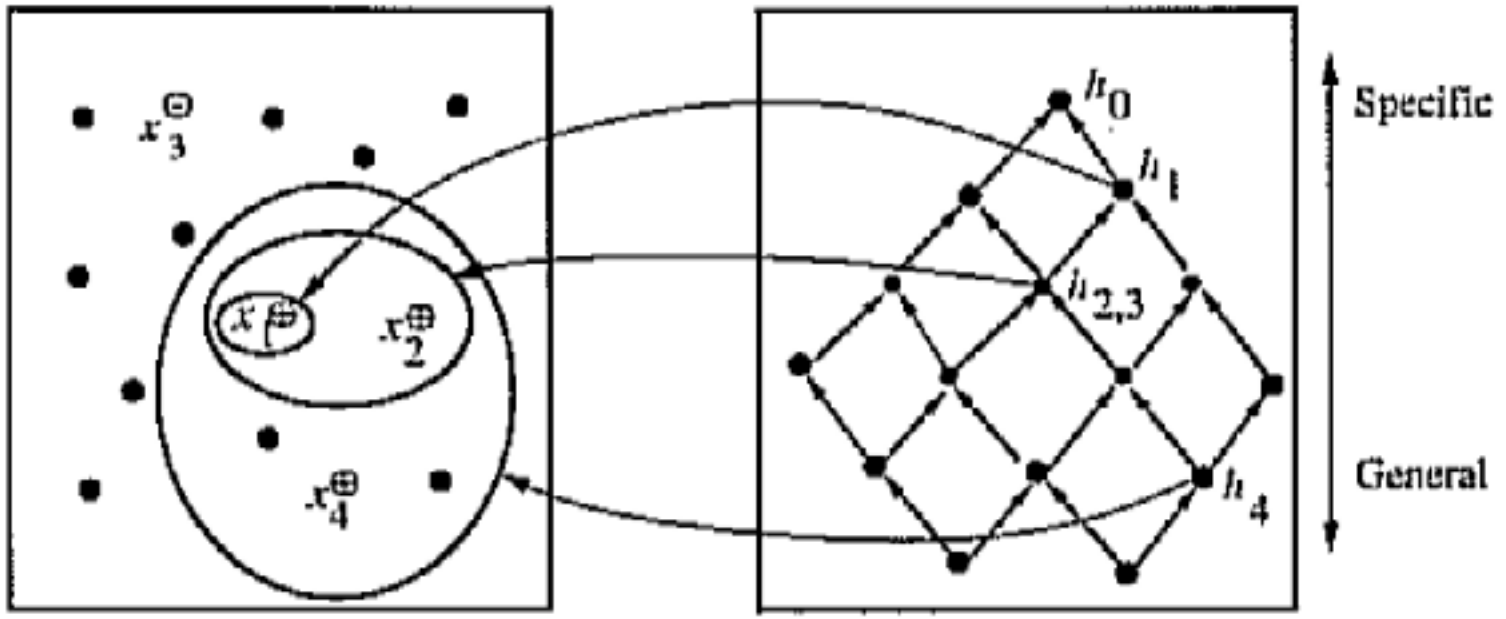
Ogni ipotesi “copre” cioè soddisfa, zero o più istanze classificate dell’insieme D in X

FIND-S: cerca l'ipotesi massimamente specifica

(per apprendere funzioni booleane)

1. Inizializza h come l'ipotesi **più specifica** in H
2. Per ogni esempio positivo $(x,1) \in D$ **esegui**:
   - Per ogni condizione su un attributo a_i in h, **esegui**:
     - Se la condizione a_i non è soddisfatta da x, sostituisci a_i in h con la condizione immediatamente più generale che sia soddisfatta da x
3. Emetti l'ipotesi h

Esempio



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 = <∅, ∅, ∅, ∅, ∅, ∅>

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 ? ?* >

Problemi dell'algoritmo Find-S

- **Convergenza**: non è chiaro quando (e se) ha appreso un modello del concetto c (condizione arresto?)
- **Consistenza**: poiché gli esempi negativi vengono ignorati, non è possibile rilevare inconsistenze
- Trova un'ipotesi **massimamente specifica** (poca capacità di generalizzare)
- In funzione di H, possono esistere **parecchie ipotesi massimamente specifiche**.

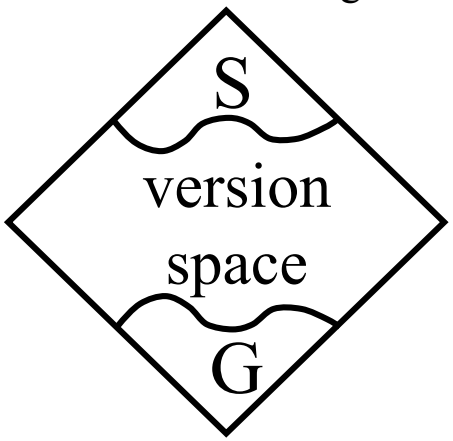
Algoritmo dello **spazio delle versioni**

- *Def.* Un'ipotesi h è **consistente** con un insieme di esempi di addestramento D del concetto obiettivo c iff $h(x)=c(x) \ \forall (x,c(x)) \in D$
- Uno **spazio delle versioni** $VS_{H,D}$ dove H è lo spazio delle ipotesi e D il training set, è il subset di H consistente con D
- Come ottenere $VS_{H,D}$?
  - Elencare tutte le ipotesi ed eliminare quelle non consistenti ad ogni nuovo esempio in D: non praticabile!
  - Algoritmo **VersionSpace**

Rappresentazione dello spazio delle versioni

- **Idea chiave**: memorizzare solo le ipotesi "di confine", utilizzando l'ordinamento parziale delle ipotesi in H
- Insieme G in $VS_{H,D}$: è l'insieme di ipotesi massimamente generali
- Insieme S in $VS_{H,D}$: è l'insieme di ipotesi massimamente specifiche
- Ogni ulteriore ipotesi $h \in VS_{H,D}$ giace nello spazio compreso fra G e S

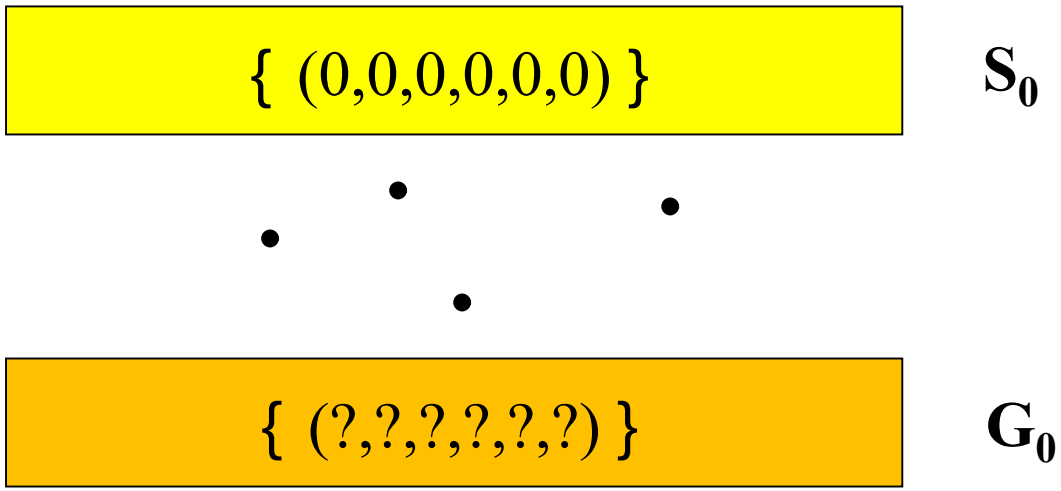
$$VS_{H,D} = \{ h \in H \mid \exists s \in S, \exists g \in G \text{ t.c. } g \geq_{gen} h \geq_{gen} s \}$$



Algoritmo VersionSpace (1)

G ← le ipotesi più generali in H

S ← le ipotesi più specifiche in H



Algoritmo VersionSpace (2)

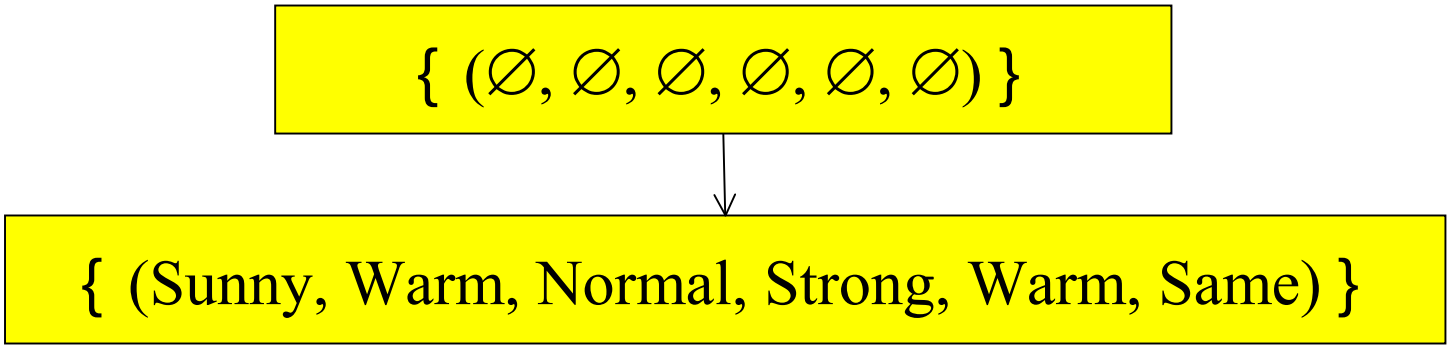
- Per ogni esempio $d = (x, c(x)) \in D$, esegui:
- Se d è positivo ($c(x)=1$), esegui:
  - Rimuovi da G le ipotesi inconsistenti con d
  - Per ogni ipotesi s in S che non sia consistente con d esegui:
    - Rimuovi s da S
    - Aggiungi ad S tutte le ipotesi h che siano **generalizzazioni minime** di s, e **consistenti** con d, e $\exists g \in G, g \geq h$
    - Rimuovi da S ogni ipotesi s che sia più generale di altre ipotesi in S

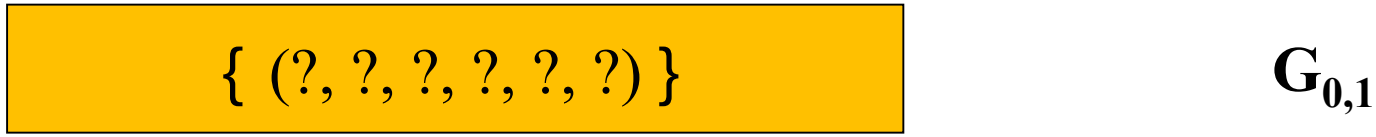
Esempio (EnjoySport)

$\{ (\varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing) \}$ S_0

$\{ (?, ?, ?, ?, ?, ?) \}$ G_0

Esempio (EnjoySport)

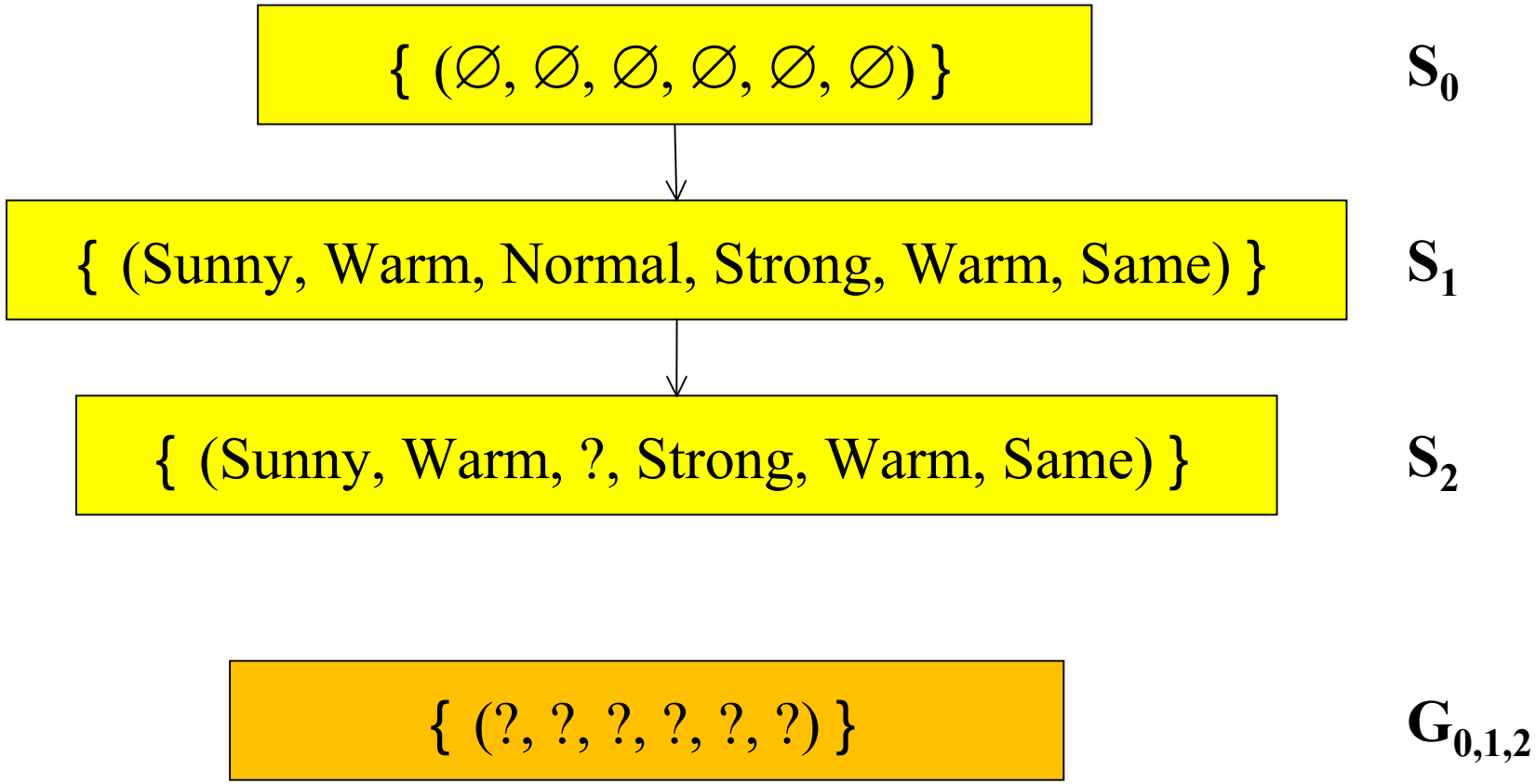




Esempio di addestramento:

d_1 = ((Sunny, Warm, Normal, Strong, Warm, Same), 1)

Esempio (EnjoySport)



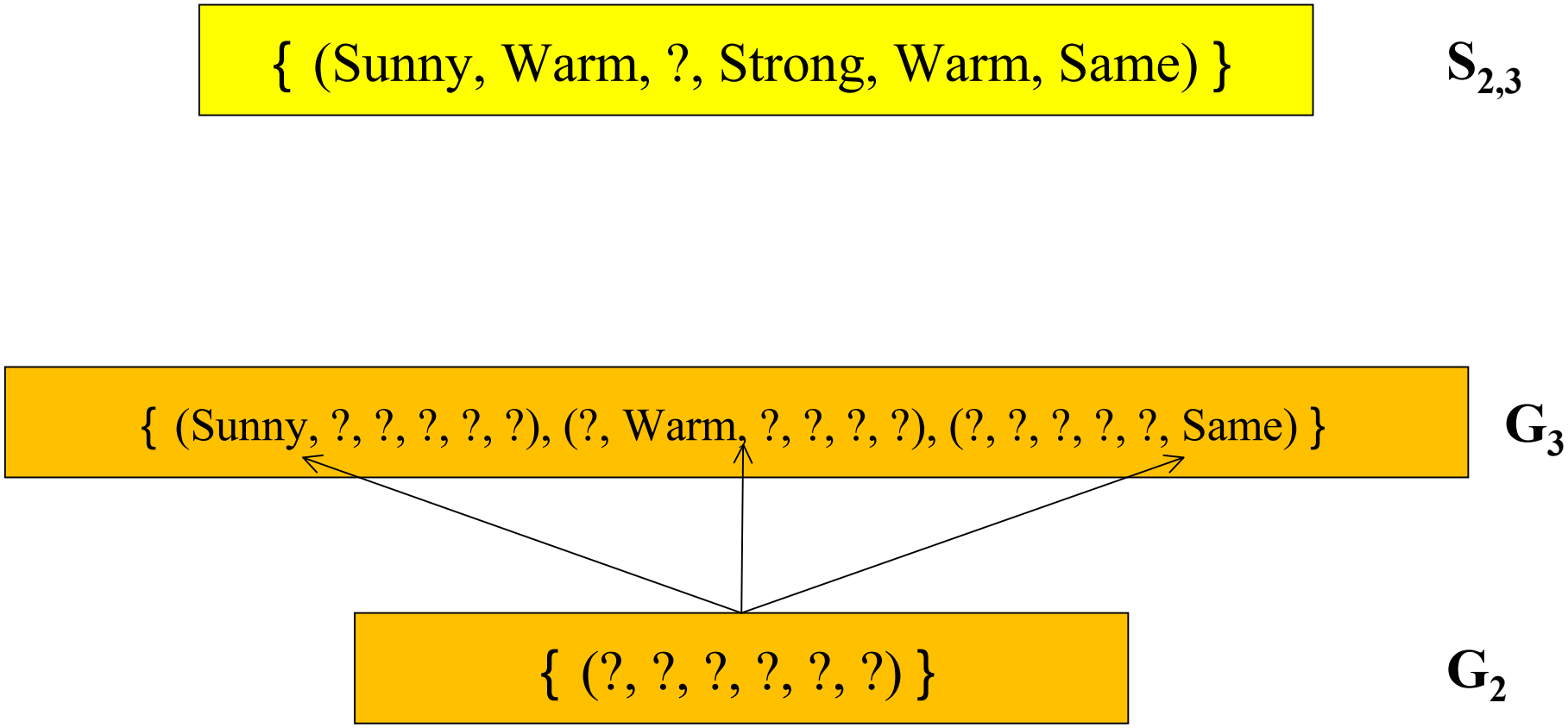
Esempio di addestramento:

d_2 = ((Sunny, Warm, High, Strong, Warm, Same), 1)

Algoritmo VersionSpace (3)

- Se d è un esempio negativo:
  - Elimina in S ogni ipotesi h inconsistente con d
  - Per ogni ipotesi *g* in G inconsistente con d:
    - Elimina *g* da G
    - Aggiungi a G tutte le specializzazioni minime h di *g* tali che h sia consistente con d, ed esistano in S ipotesi più specifiche di h (cioè, h non "sconfini" in S)
    - Elimina da G ogni ipotesi *g'* che sia meno generale di un'altra ipotesi *g"* in G
      - not $\forall(g',g'')\ g' \in G,\ g'' \in G$ t.c. $g'' \geq_{gen} g'$)

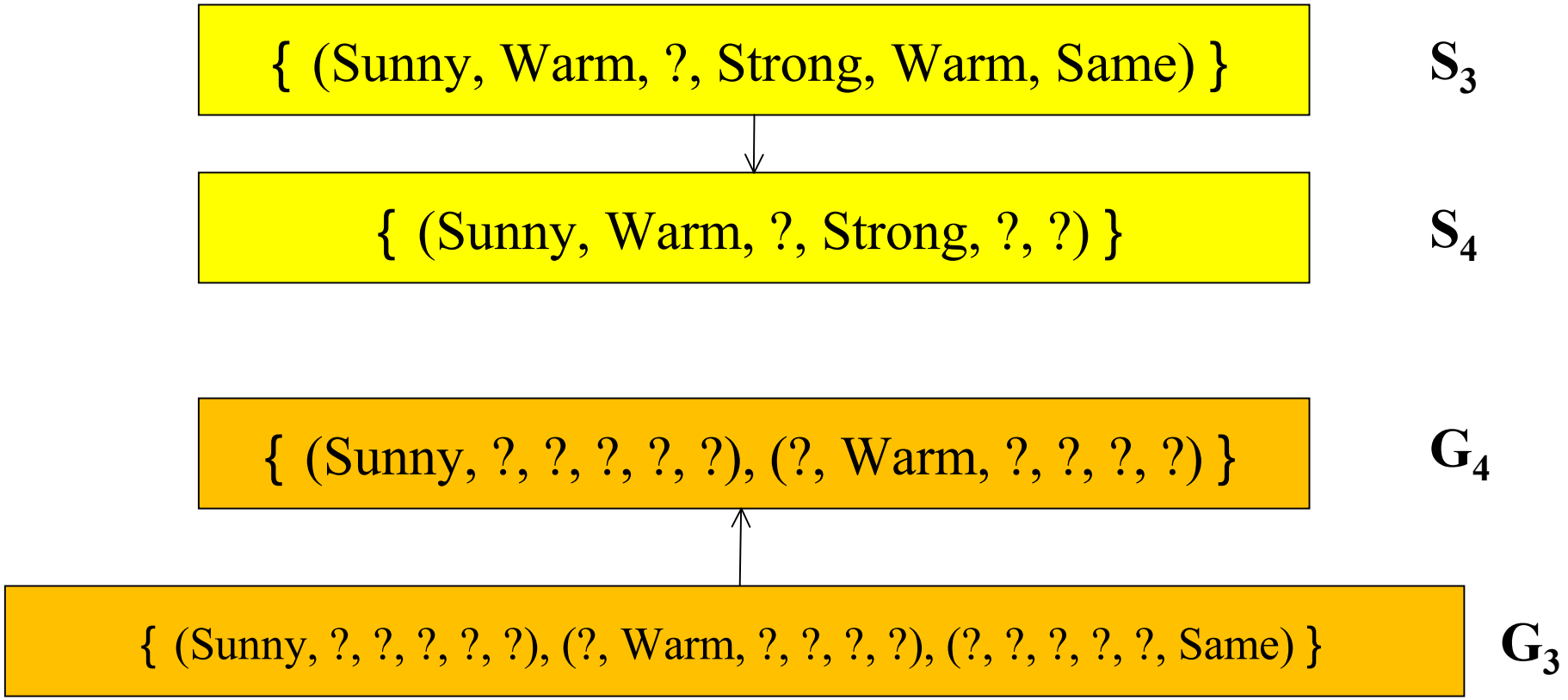
Esempio (EnjoySport)



Esempio di addestramento:

d_3 = ((Rainy, Cold, High, Strong, Warm, Change), 0)

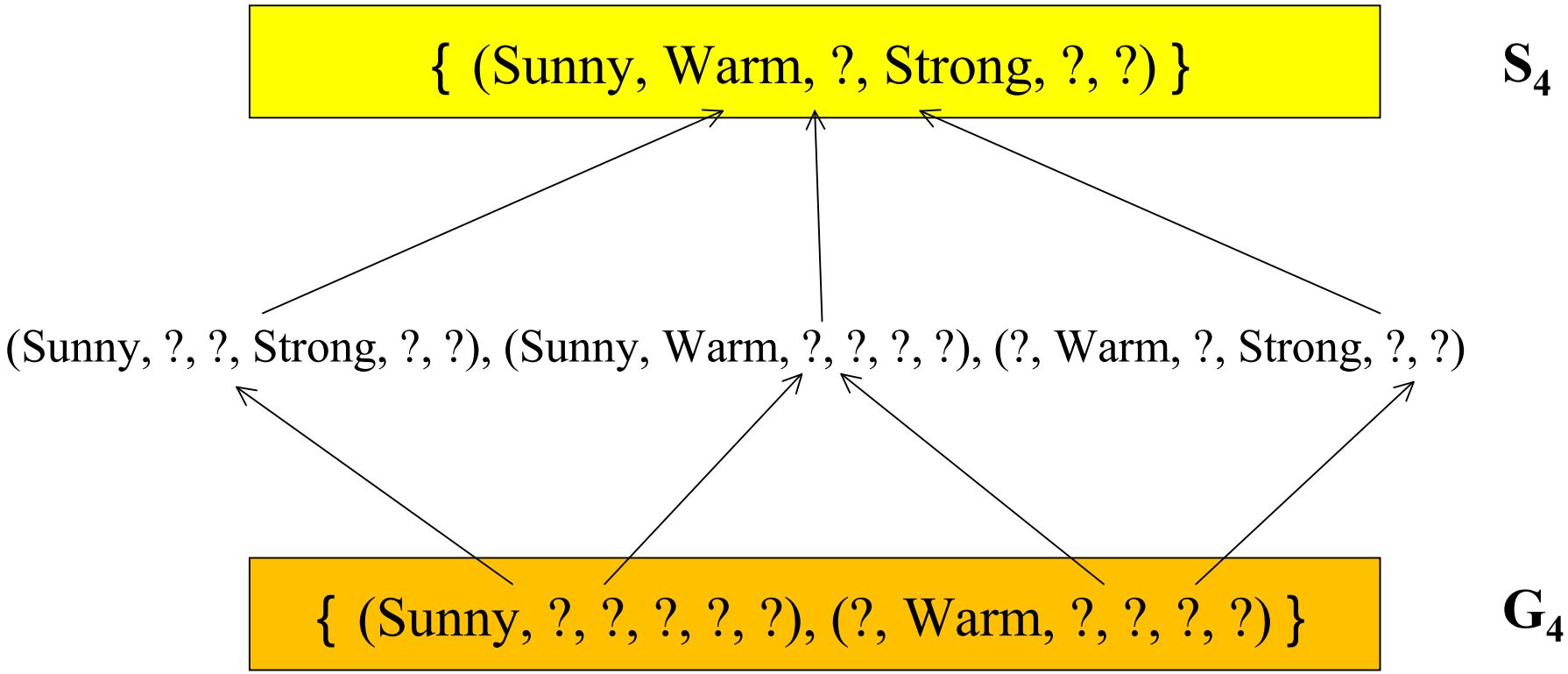
Esempio (EnjoySport)



Esempio di addestramento:

d_4 = ((Sunny, Warm, High, Strong, Cool, Change), 1)

Lo spazio delle versioni finale per EnjoySport



Un altro esempio: apprendere una funzione 3-monomiale

S_0 { $(\varnothing, \varnothing, \varnothing)$ }

$((\overline{x}, \overline{y}, ?), +)$

G_0 { $(?, ?, ?)$ }

Un altro esempio: apprendere una funzione 3-monomiale

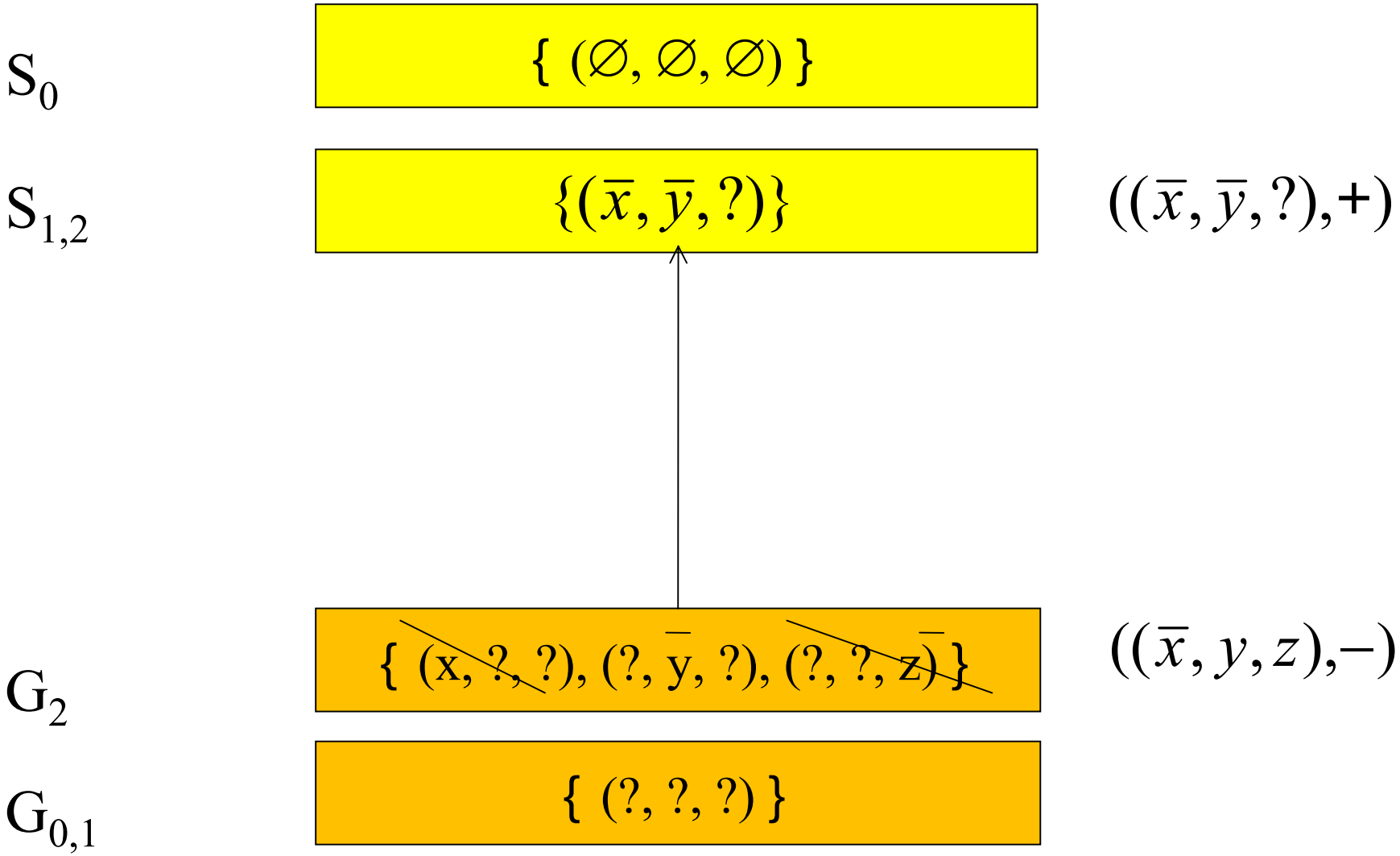
S_0 $\{ (\varnothing, \varnothing, \varnothing) \}$

S_1 $\{(\bar{x}, \bar{y}, ?)\}$ $((\bar{x}, \bar{y}, ?), +)$

$((\bar{x}, y, z), -)$

$G_{0,1}$ $\{ (?, ?, ?) \}$

Un altro esempio: apprendere una funzione 3-monomiale



Complessità

- Supponiamo che la funzione obiettivo, o concetto, abbia una forma monomiale

$$c_n = a_{n-1} \wedge a_{n-2} \wedge \ldots a_0$$

$$a_i \in \{0,1\},$$

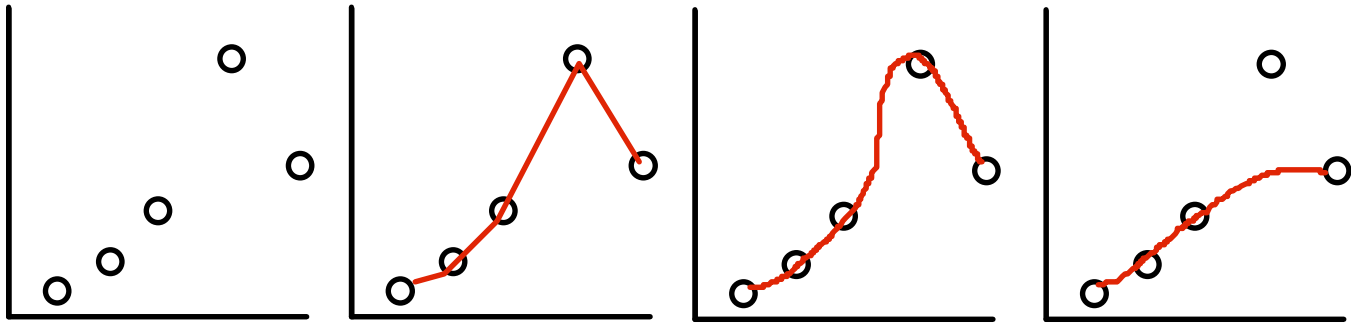
- Per esempio,supponiamo che il concetto “esatto” sia

  la funzione c(x) = 1 (sempre vero!) per cui h = (?,?,….?)

- Se gli esempi in D sono descritti da n attributi booleani, es. d_0=(<1,0,0,…,0>,1), occorrono nel caso peggiore un numero esponenziale in n di esempi per apprendere il concetto (devo avere evidenza del fatto che c=1 per ogni valore di ogni attributo!)

Bias

- "Bias" o inclinazione è ogni criterio utilizzato (oltre ai dati di training) per preferire un'ipotesi ad un'altra



- Inclinazioni:
  - la scelta del ***modello di rappresentazione*** per H (ad esempio, h è una congiunzione di variabili booleane, o una rete neurale..)
  - La scelta dell'***algoritmo di ricerca*** nello spazio H (ad esempio, la scelta dell'algoritmo Find-S ha un'inclinazione per ipotesi massimamente specifiche)

Uno spazio “biased”

- Se scelgo di utilizzare k-monomials, come nel caso precedente, non posso rappresentare ipotesi del tipo:

$$(warm \wedge cloudy) \vee (cool \wedge sunny)$$

- La scelta della funzione c ha dunque orientato l’apprendimento verso un sottoinsieme di ipotesi che non è detto sia adatto ad apprendere soluzioni per il problema considerato

Un apprendista privo di inclinazioni

- Se non voglio un “bias”, scelgo lo spazio H che esprima **ogni concetto apprendibile** (cioè, $H \equiv 2^X$ l’insieme delle potenze di X)
- H’: l’insieme delle ipotesi che si ottengono combinando ipotesi di H (congiunzione di letterali) mediante gli operatori $\neg \wedge \vee$
  - Es.: (Sunny, Warm, Normal, ?, ?, ?) $\wedge$ $\neg$ (?, ?, ?, ?, ?, Change)
- Dato $D = \{ (x_1, 1), (x_2, 1), (x_3, 1), (x_4, 0), (x_5, 0) \}$
- Quanto valgono S e G?

Un apprendista privo di inclinazioni

- $S \leftarrow \{ x_1 \vee x_2 \vee x_3 \}$
  - (si limita ad accettare gli esempi positivi)
- $G \leftarrow \{ \neg x_4 \wedge \neg x_5 \}$
  - (si limita a escludere gli esempi negativi)
- Ogni nuovo esempio aggiunge una **disgiunzione** a S (se +) o una **congiunzione** a G (se -)
- Ma in questo modo, i soli esempi classificabili da S e G sono gli esempi osservati! Per convergere, bisognerebbe far osservare all'algoritmo ogni possibile elemento di X!

In assenza di un'inclinazione non è possibile apprendere alcuna generalizzazione degli esempi osservati!

Un'inclinazione è necessaria!

- La "bontà" di un sistema di apprendimento discende dalla adeguatezza dell'inclinazione prescelta
- Come scegliere una buona inclinazione?
  - Conoscenza del dominio
  - Conoscenza della sorgente dei dati di apprendimento
  - Semplicità e generalità (rasoio di Occam)

Alcuni aspetti correlati all'apprendimento di concetti da esempi

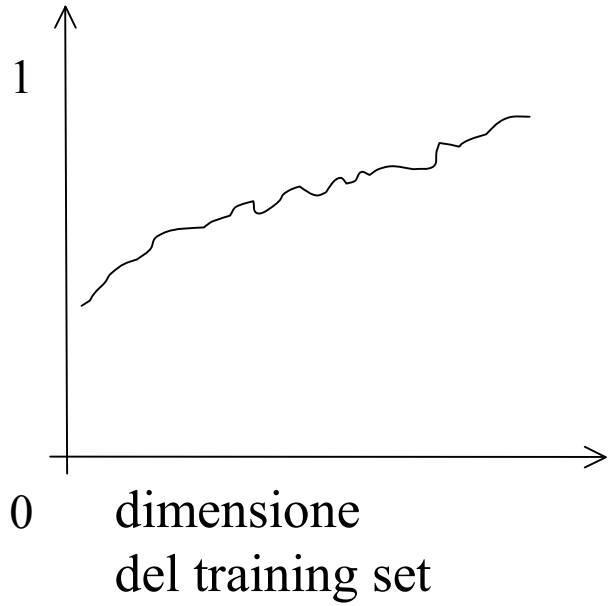
1. Dimensionamento del training set D (studio delle curve di apprendimento)
2. Sovradattamento
3. Rumore
4. Attributi irrilevanti

1. Studio delle Curve di apprendimento

- **Problema**: se sottopongo al sistema un insieme di esempi **insufficienti**, o **rumorosi**, o **ambigui**, posso apprendere ipotesi il cui potere predittivo è insufficente.
- Un algoritmo di appendimento è buono se può prevedere con buona precisione la classificazione di esempi non visti

1. Curve di apprendimento

- Come variane le prestazioni del sistema di apprendimento all'aumentare del training set
- Misura di prestazione:
  - **Accuratezza** = $n^{TP}+n^{TN}/(n^{TP}+n^{TN}+n^{FP}+n^{FN})$
    - n^{TP} = numero di veri positivi
    - n^{TN} = numero di veri negativi
    - n^{FP} = numero di falsi positivi
    - n^{FN} = numero di falsi negativi



2. Sovradattamento

- *Def.* Dato uno spazio di ipotesi H, una ipotesi h si dice **sovradattata** (**overfit**) rispetto ai dati di apprendimento D se esiste una ipotesi alternativa h' tale che h commette un errore minore di h' sul set di addestramento D, ma h' produce un errore minore sui dati di test T.

2. Sovradattamento

- Quali sono le cause del sovradattamento?
  - **Errori nella classificazione** dei dati in D
  - **Esempi idiosincratici** (la distribuzione di probabilità dei fenomeni in D non è la stessa di X, es. molti esempi di **balene** nell'apprendimento di una definizione di **mammifero**)
  - Regolarità incidentali fra i dati (es. molti pazienti che soffrono di diabete abitano in una certa area - ma nell'area è stata fatta una campagna di prevenzione!)
  - Attributi irrilevanti (colore dei capelli nell'esempio dell'agente esaminatore)

3. Rumore

- In alcuni casi, due o più esempi con la stessa descrizione possono avere classificazioni diverse (questo può essere dovuto ad ambiguità, oppure ad errori o a un insieme di attributi non sufficientemente descrittivo)
- Esempio: optical character recognition

4. Attributi irrilevanti

- Supponiamo di considerare il problema della predizione del lancio di un dato.
- Consideriamo i seguenti attributi che descrivono gli esempi:
  - Ora, mese, colore del dado
- Se nessuna descrizione si ripete, il sistema sarà solo in grado di costruire un modello predittivo totalmente inattendibile

Conclusioni

- Apprendimento da esempi:
  - Rappresentare lo spazio delle istanze
  - Rappresentare la funzione obiettivo, detta in questo caso funzione di classificazione c
  - Creare un insieme di addestramento
  - Identificare un algoritmo
- Problemi:
  - La scelta della funzione c determina una "**inclinazione**"
  - La scelta degli attributi per rappresentare le istanze ha anche essa un rilievo: ad esempio **attributi irrilevanti** possono influire negativamente
  - Il dimensionamento di D ha effetti sull'apprendimento (curve di adattamento e problema dell'**overfitting** o **sovradattamento**)
  - La presenza di **rumore** o attributi non noti in D può rallentare l'apprendimento, o impedirlo nel caso di algoritmi che apprendono solo ipotesi consistenti con D

Conclusioni (2)

- Abbiamo studiato due “semplici” algoritmi:

  Find-S e VS
- Quali problemi pongono, rispetto a quelli elencati nella precedente slide?
  - Forte “bias” determinato dalla scelta di c: un k-monomial
  - Apprendono solo classificatori CONSISTENTI con D, quindi non sono toleranti al rumore!!

Prossimo algoritmo: **alberi di decisione**.
Minore bias, tollera rumore