# Esercizio: apprendimento di congiunzioni di letterali

Algoritmo **Find-S**

/* trova l'ipotesi più specifica consistente con l'insieme di apprendimento */

- input: insieme di apprendimento $Tr$
- inizializza l'ipotesi corrente $h$ alla ipotesi più specifica
  $h \equiv l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \wedge \cdots \wedge l_m \wedge \neg l_m$
- per ogni istanza positiva di apprendimento $(x, true) \in Tr$
  - rimuovi da $h$ ogni letterale (affermato o negato) che non è soddisfatto da $x$
- restituisci $h$

# Esempio di applicazione: $m = 5$

| Esempio (positivo) | ipotesi corrente |
|---|---|
| | $h_0 \equiv l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \wedge l_3 \wedge \neg l_3 \wedge l_4 \wedge \neg l_4 \wedge l_5 \wedge \neg l_5$ |
| 1 1 0 1 0 | $h_1 \equiv l_1 \wedge l_2 \wedge \neg l_3 \wedge l_4 \wedge \neg l_5$ |
| 1 0 0 1 0 | $h_2 \equiv l_1 \wedge \neg l_3 \wedge l_4 \wedge \neg l_5$ |
| 1 0 1 1 0 | $h_3 \equiv l_1 \wedge l_4 \wedge \neg l_5$ |
| 1 0 1 0 0 | $h_4 \equiv l_1 \wedge \neg l_5$ |
| 0 0 1 0 0 | $h_5 \equiv \neg l_5$ |

Notare che $h_0 \leq_g h_1 \leq_g h_2 \leq_g h_3 \leq_g h_4 \leq_g h_5$

Inoltre, ad ogni passo si sostituisce $h_i$ con un' ipotesi $h_{i+1}$ che costituisce una *generalizzazione minima* di $h_i$ consistente con l'esempio corrente.

Quindi **Find-S** restituisce l'ipotesi più specifica consistente con $Tr$

# Osservazioni su **Find-S**

**Find-S** è in effetti uno schema di algoritmo che si può applicare a spazi di istanze e ipotesi diversi da quelli visti.

L'idea base dell'algoritmo è quella di effettuare una *generalizzazione minima* dell'ipotesi corrente quando questa non è più consistente con l'esempio corrente.

Notare che ogni volta che l'ipotesi corrente $h$ viene *generalizzata* ad una nuova ipotesi $h'$ ($h' \geq_g h$), tutti gli esempi positivi visti in precedenza continuano ad essere soddisfatti dalla nuova ipotesi $h'$ (infatti, poiché $h' \geq_g h$, si ha che
$\forall x \in X,\ (h(x) = 1) \rightarrow (h'(x) = 1)$)

Infine, se il concetto da apprendere è contenuto in $\mathcal{H}$, tutti gli eventuali esempi negativi sono soddisfatti automaticamente dalla ipotesi restituita da **Find-S** in quanto questa è l'ipotesi consistente più specifica, cioè quella che attribuisce il minor numero possibile di 1 alle istanze in $X$.

Esiste un motivo per preferire l'ipotesi consistente più specifica ?

# Principali Paradigmi di Apprendimento: Richiamo

Apprendimento Supervisionato:

- dato in insieme di esempi pre-classificati, $Tr = \{(x^{(i)}, f(x^{(i)}))\}$, apprendere una descrizione generale che incapsula l'informazione contenuta negli esempi (regole valide su tutto il dominio di ingresso)
- tale descrizione deve poter essere usata in modo predittivo (dato un nuovo ingresso $\tilde{x}$ predire l'output associato $f(\tilde{x})$)
- si assume che un esperto (o maestro) ci fornisca la supervisone (cioè i valori della $f()$ per le istanze $x$ dell'insieme di apprendimento)

**Find-S** è un algoritmo di apprendimento supervisionato

# Dati

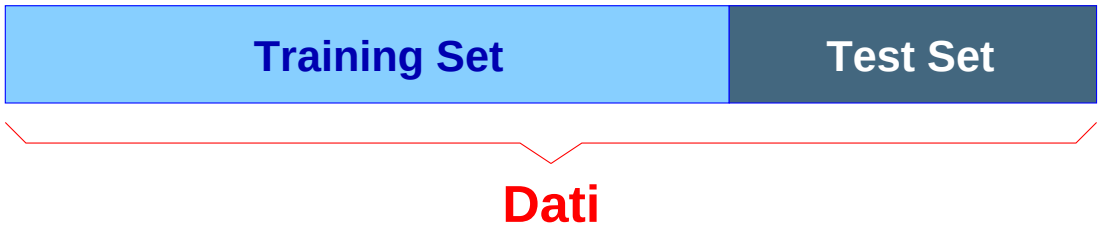
Consideriamo il paradigma di Apprendimento Supervisonato

Dati a nostra disposizione (off-line)

$$\text{Dati} = \{(x^{(1)}, f(x^{(1)})), \ldots, (x^{(N)}, f(x^{(N)}))\}$$

Suddivisione tipica ($N = N_{tr} + N_{ts}$) :

- $\text{Training Set} = \{(x^{(1)}, f(x^{(1)})), \ldots, (x^{(N_{tr})}, f(x^{(N_{tr})}))\}$
  usato direttamente dall'algoritmo di apprendimento;
- $\text{Test Set} = \{(x^{(1)}, f(x^{(1)})), \ldots, (x^{(N_{ts})}, f(x^{(N_{ts})}))\}$
  usato alla fine dell'apprendimento per stimare la bontà della soluzione.

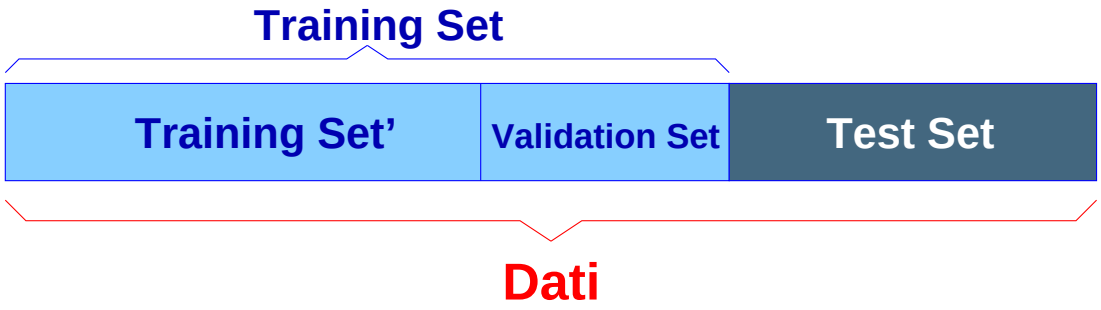


# Dati (cont.)

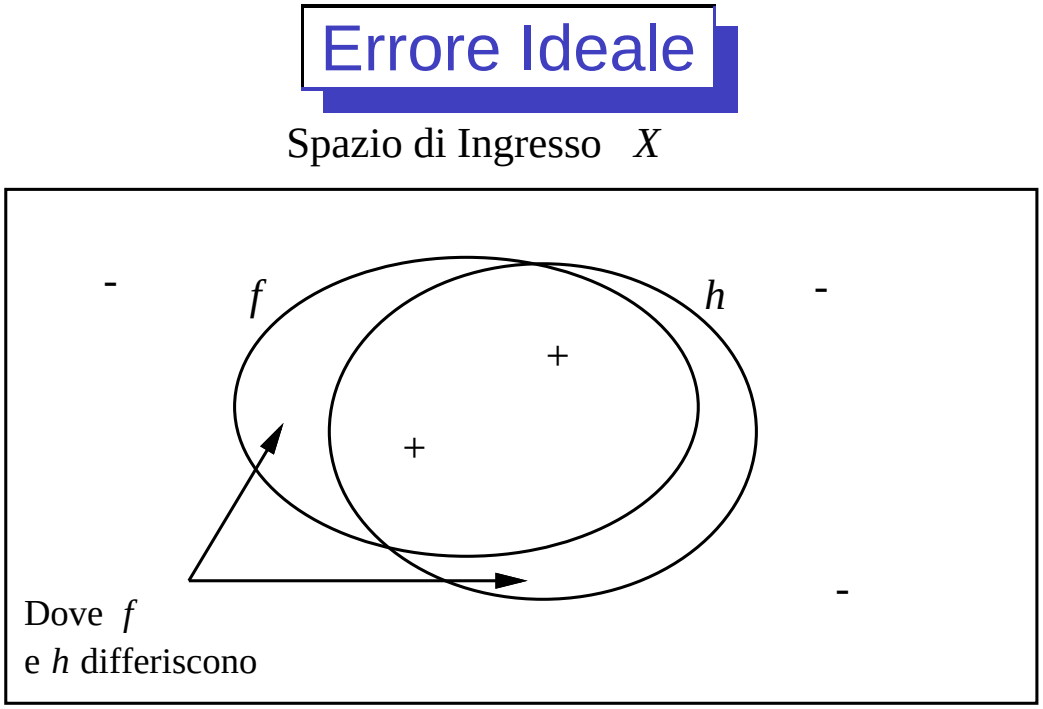
Se $N$ abbastanza grande il Training Set è ulteriormente suddiviso in due sottoinsiemi ($N_{tr} = N_{\widehat{tr}} + N_{val}$) :

- Training Set' $= \{(x^{(1)}, f(x^{(1)})), \ldots, (x^{(N_{\widehat{tr}})}, f(x^{(N_{\widehat{tr}})})\}$
  usato direttamente dall'algoritmo di apprendimento;
- Validation Set $= \{(x^{(1)}, f(x^{(1)})), \ldots, (x^{(N_{val})}, f(x^{(N_{val})})\}$
  usato indirettamente dall'algoritmo di apprendimento.

Il Validation Set serve per scegliere l'ipotesi $h \in \mathcal{H}$ migliore fra quelle consistenti con il Training Set'



# Errore Ideale

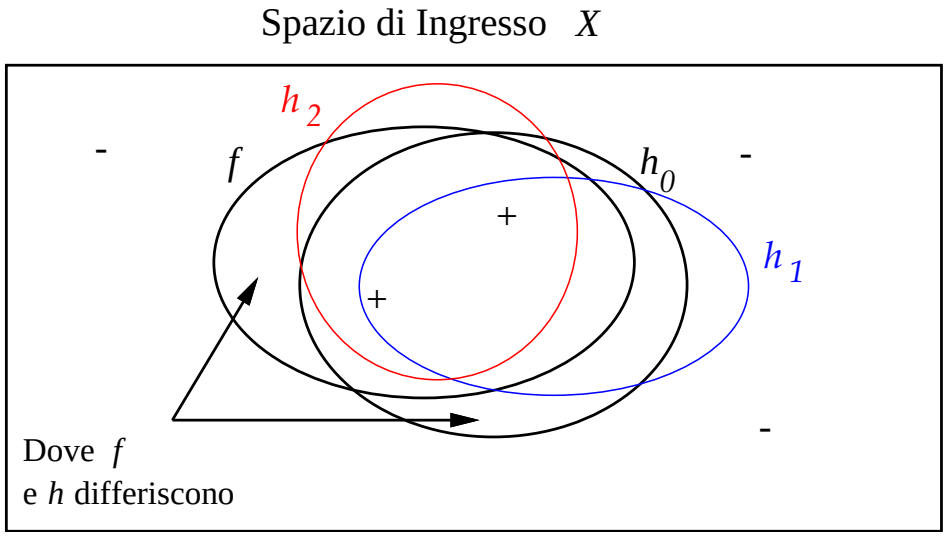



Supponiamo che la funzione $f$ da apprendere sia una funzione booleana (concetto):

$$f : X \to \{0, 1\} \ (\{-, +\})$$

**Def:** L'**Errore Ideale** ($error_{\mathcal{D}}(h)$) di una ipotesi $h$ rispetto al concetto $f$ e la distribuzione di probabilità $\mathcal{D}$ (probabilità di osservare l'ingresso $x \in X$) è la probabilità che $h$ classifichi erroneamente un input selezionato a caso secondo $\mathcal{D}$: $error_{\mathcal{D}}(h) \equiv \underset{x \in \mathcal{D}}{Pr}[f(x) \neq h(x)]$

# Errore di Apprendimento




Dato $Tr$ = Training Set', più ipotesi possono essere consistenti: $h_0$, $h_1$, $h_2$ quale scegliere ?

**Def:** L'**Errore Empirico** ($error_{Tr}(h)$) di una ipotesi $h$ rispetto a $Tr$ è il numero di esempi che $h$ classifica erroneamente: $error_{Tr}(h) \equiv \#\{(x, f(x)) \in Tr | f(x) \neq h(x)\}$

**Def:** Una ipotesi $h \in \mathcal{H}$ è sovraspecializzata (**overfit**) $Tr$ se $\exists h' \in \mathcal{H}$ tale che $error_{Tr}(h) < error_{Tr}(h')$, ma $error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$.

Il Validation Set serve per cercare di selezionare l'ipotesi migliore (evitare **overfit**).

# PAC learning

E' possibile sviluppare una teoria che non ha bisogno del Validation Set e che è in grado di dirmi di quanti esempi di apprendimento ho bisogno per ottenere una ipotesi che risponda a determinati criteri di qualità ?

Per l'apprendimento di concetti una possibile risposta (parziale) è la teoria del PAC Learning (Probably Approximately Correct Learning)

Assunzioni: tutti gli input ed output sono binari, e i dati non contengono rumore e sono estratti da $X$ secondo una distribuzione di probabilità $\mathcal{D}$ *arbitraria* ma *stazionaria*.

Vantaggi:

- è possibile definire dei bound sul numero di esempi di apprendimento per garantire che con una certa probabilità un algoritmo di apprendimento è in grado di restituire una ipotesi con errore ideale basso (approssimazione)
- è possibile caratterizzare formalmente la *complessità* computazionale nell'apprendere determinate classi di concetti

# PAC learning

Consideriamo una class $C$ di possibili concetti target (funzioni da apprendere) definite su un insieme di istanze $X$ di lunghezza $m$ (ad esempio stringhe binarie), ed un algoritmo di apprendimento $L$ che utilizza uno spazio delle ipotesi $\mathcal{H}$.

**Definizione:** $C$ è PAC-apprendibile da $L$ usando $\mathcal{H}$ se per tutti i

- $c \in C$,
- distribuzioni $\mathcal{D}$ su $X$,
- $\epsilon$ tali che $0 < \epsilon < 1/2$,
- $\delta$ tali che $0 < \delta < 1/2$,

l'algoritmo di apprendimento $L$ con probabilità almeno $(1 - \delta)$ restituisce un'ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) \leq \epsilon$, in tempo che è **polinomiale** in $1/\epsilon$, $1/\delta$, $m$, e $size(c)$ (spazio di memoria che serve per rappresentare $c$).

# PAC learning

E' stato dimostrato che se si assume:

- $c \in \mathcal{H}$
- $L$ consistente, cioè restituisce una ipotesi consistente con l'insieme di apprendimento (**Find-S** è un algoritmo consistente)

allora con probabilità almeno $(1 - \delta)$, l'algoritmo di apprendimento $L$ restituisce un'ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) \leq \epsilon$ a patto che il numero di esempi di apprendimento $n$ soddisfi la seguente disuguaglianza:

$$n \geq \frac{1}{\epsilon}(ln(|\mathcal{H}|) + ln(\frac{1}{\delta}))$$

dove $|\mathcal{H}|$ è la cardinalità dello spazio delle ipotesi e $ln()$ è il logaritmo in base naturale

# PAC learning

E' la congiunzione di $m$ letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$ ? Si !

Infatti:

- ogni congiunzione di $m$ letterali è inclusa in $\mathcal{H}_4$
- **Find-S** è un algoritmo consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poichè $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

$$n \geq \frac{1}{\epsilon}((m+1)ln(3) + ln(\frac{1}{\delta}))$$

  quindi $n$ è polinomiale in $1/\epsilon$, $1/\delta$, $m$, e $size(c)$ (che non compare)

- per ogni esempio di apprendimento **Find-S** esegue una elaborazione che è lineare con l'ipotesi corrente (che è di dimensione maggiore o uguale a $size(c)$) e la dimensione dell'input ($m$), quindi di nuovo polinomiale, e globalmente polinomiale per $Tr$

Quindi tutte le condizioni per la PAC-apprendibilità sono soddisfatte!

# PAC learning

Usando la disuguaglianza ed altre considerazioni si è potuto dimostrare che alcune classi di concetti non sono PAC-apprendibili considerando un particolare algoritmo di apprendimento e spazio delle ipotesi.

In particolare si è dimostrato che:

- se $\mathcal{H}$ contiene tutte le funzioni booleane realizzabili su $X$ allora $C = \mathcal{H}$ non è PAC-apprendibile da algoritmi consistenti.
- esistono classi di concetti $C$ che non sono apprendibili da algoritmi consistenti che utilizzano $C$ come spazio delle ipotesi, ma diventano PAC-apprendibili se come spazio delle ipotesi si usano opportuni $C \subset \mathcal{H}$

Un problema con l'uso della disuguaglianza è che questa diventa inutile se $\mathcal{H}$ non è finito

Quello che in realtà interessa è caratterizzare la "potenza" di uno spazio delle ipotesi: la VC-dimension (Vapnik-Chervonenkis) fornisce una risposta in questa direzione

# VC-dimension

**Definizione:** Frammentazione (Shattering)

Dato $S \subset X$, $S$ è frammentato (shattered) dallo spazio delle ipotesi $\mathcal{H}$ se e solo se

$$\forall S' \subseteq S,\ \exists h \in \mathcal{H},\ \text{tale che}\ \forall x \in S,\ h(x) = 1 \Leftrightarrow x \in S'$$

($\mathcal{H}$ realizza tutte le possibili dicotomie di $S$)

**Definizione:** VC-dimension

La VC-dimension di uno spazio delle ipotesi $\mathcal{H}$ definito su uno spazio delle istanze $X$ è data dalla cardinalità del sottoinsieme più grande di $X$ che è frammentato da $\mathcal{H}$:

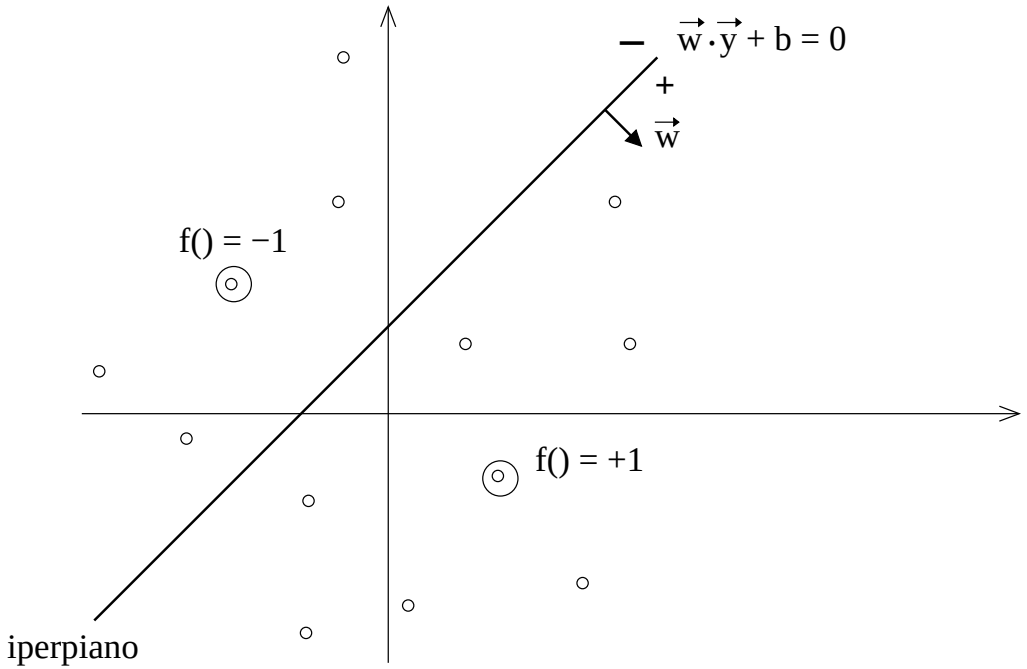
$$VC(\mathcal{H}) = \max_{S \subseteq X} |S| :\ \mathcal{H}\ \text{frammenta}\ S$$

$VC(\mathcal{H}) = \infty$ se $S$ non è limitato

# VC-dimension: Esempio

Quale è la VC-dimension di $\mathcal{H}_1$ ?

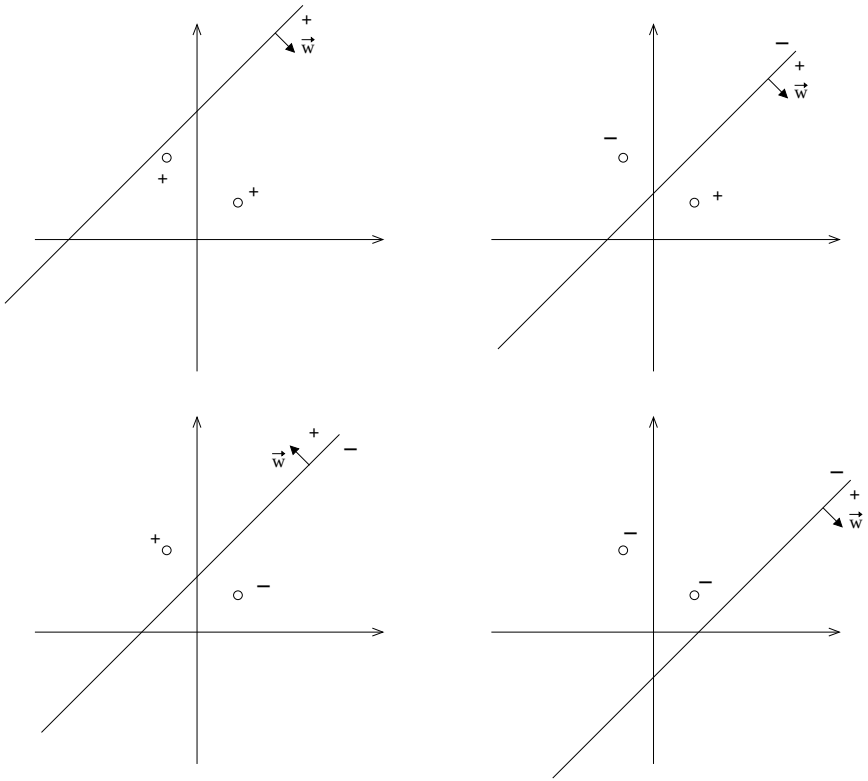
$$\mathcal{H}_1 = \{f_{(\vec{w},b)}(\vec{y}) | f_{(\vec{w},b)}(\vec{y}) = sign(\vec{w} \cdot \vec{y} + b), \vec{w} \in \mathbb{R}^2, b \in \mathbb{R}\}$$



# VC-dimension: Esempio

Quale è la VC-dimension di $\mathcal{H}_1$ ?

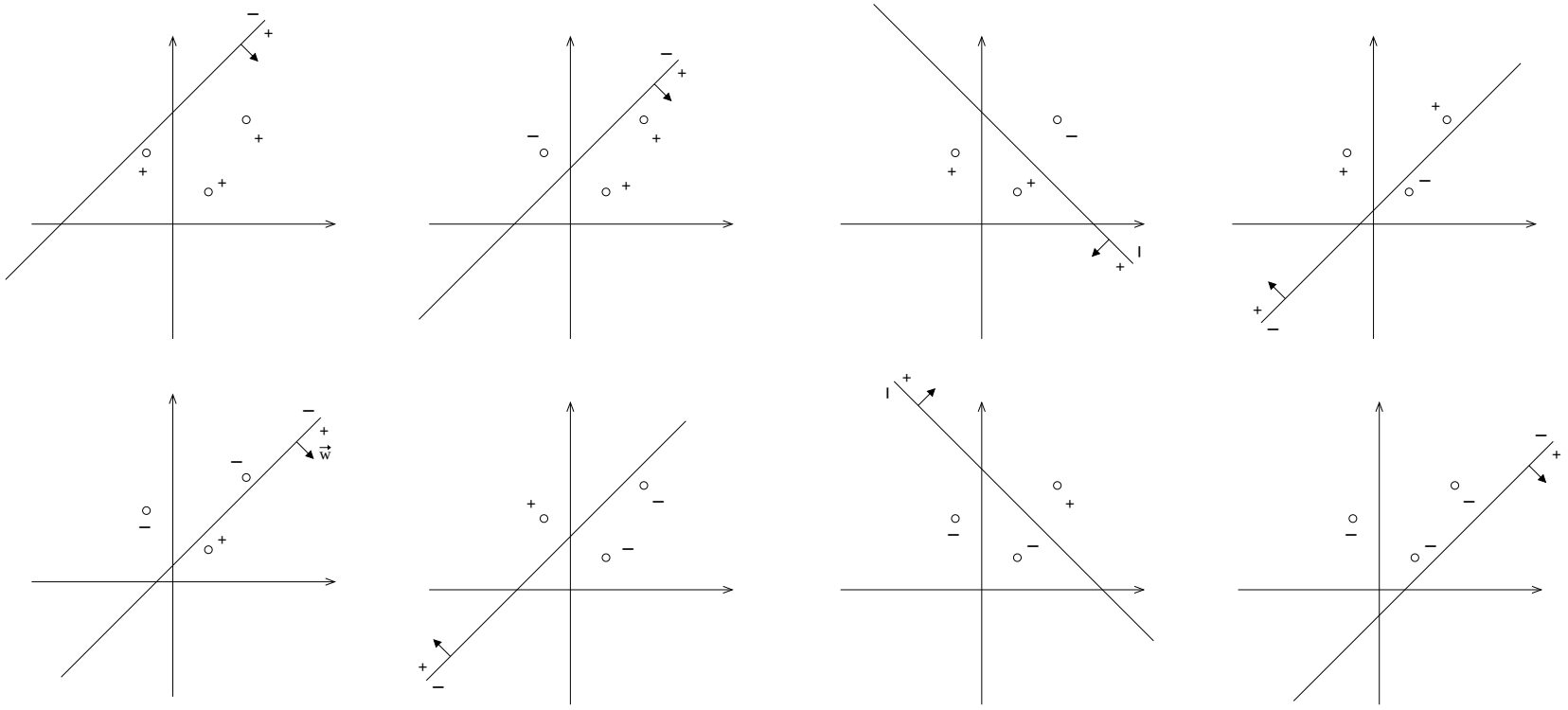
$VC(\mathcal{H}) \geq 1$ banale. Vediamo cosa succede con 2 punti:



# VC-dimension: Esempio

Quale è la VC-dimension di $\mathcal{H}_1$ ?

Quindi $VC(\mathcal{H}) \geq 2$. Vediamo cosa succede con 3 punti:



# VC-dimension: Esempio

Quale è la VC-dimension di $\mathcal{H}_1$ ?

Quindi $VC(\mathcal{H}) \geq 3$. Cosa succede con 4 punti ?

# PAC learning

E' stato dimostrato che se si assume:

- $c \in \mathcal{H}$
- $L$ consistente, cioè restituisce una ipotesi consistente con l'insieme di apprendimento (**Find-S** è un algoritmo consistente)

allora con probabilità almeno $(1 - \delta)$, l'algoritmo di apprendimento $L$ restituisce un'ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) \leq \epsilon$ a patto che il numero di esempi di apprendimento $n$ soddisfi la seguente disuguaglianza:

$$n \geq \frac{1}{\epsilon}(4 \log_2(\frac{1}{\delta}) + 8VC(\mathcal{H}) \log_2(\frac{13}{\epsilon}))$$

Notare che vale $VC(\mathcal{H}) \leq \log_2(|\mathcal{H}|)$