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

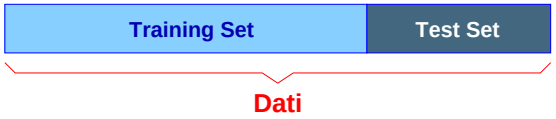
Consideriamo il paradigma di Apprendimento Supervisonato

Dati a nostra disposizione (off-line)

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

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

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

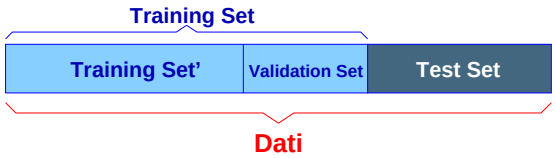


Dati (cont.)

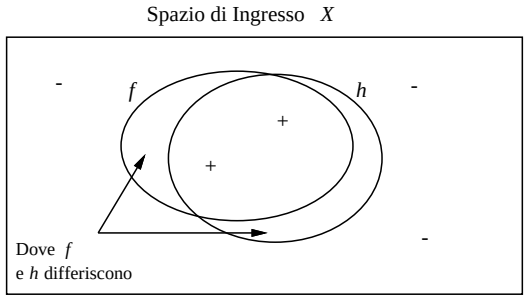
Se N abbastanza grande il Training Set è ulteriormente suddiviso in due sottoinsiemi ($N_{tr} = N_{\widehat{tr}} + N_{val}$) :

- 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

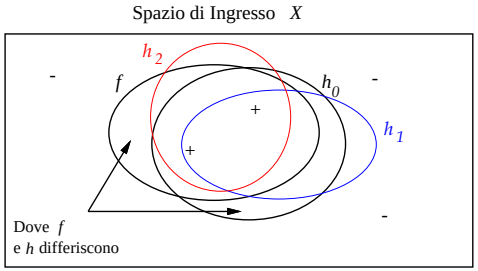


Supponiamo che la funzione f da apprendere sia una funzione booleana (concetto):

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

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 classifi chi erroneamente un input selezionato a caso secondo $\mathcal{D}$: $error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$

Errore di Apprendimento



Dato Tr = Training Set', pi`u 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 classifi ca 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 defi nire 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) defi nite 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,
- ϵ tali che $0 < \epsilon < 1/2$,
- δ 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 fi nito

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 $\mathcal{H}$ defi nito su uno spazio delle istanze X è data dalla cardinalità del sottoinsieme pi`u grande di X cheè frammentato da $\mathcal{H}$:

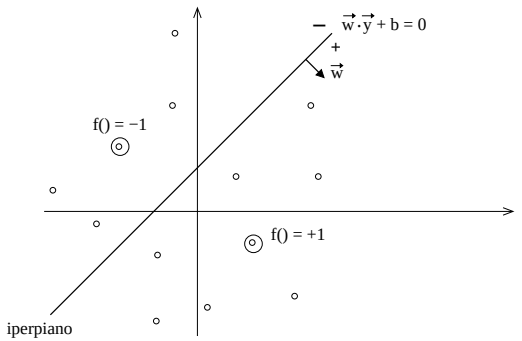
$$VC(\mathcal{H}) = \max_{S \subseteq X} |S| :\ \mathcal{H} \text{ frammenta } S$$

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

VC-dimension: Esempio

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

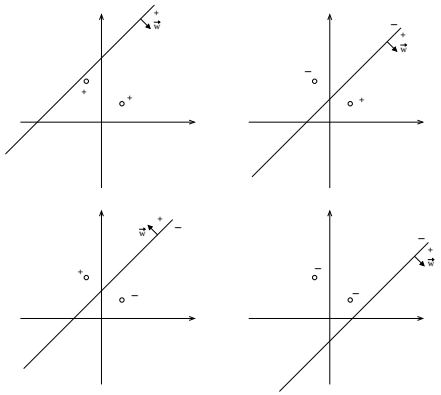
$\mathcal{H}_1 = \{f_{(\vec{w},b)}(\vec{y}) | f_{(\vec{w},b)}(\vec{y}) = sign(\vec{w} \cdot \vec{y} + b), \vec{w} \in \mathbb{R}^2, b \in \mathbb{R}\}$



VC-dimension: Esempio

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

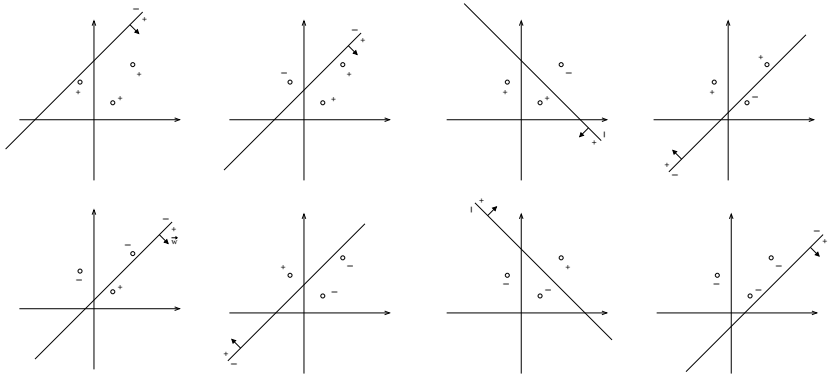
$VC(\mathcal{H}) \geq 1$ banale. Vediamo cosa succede con 2 punti:



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 $\mathcal{H}_1$?

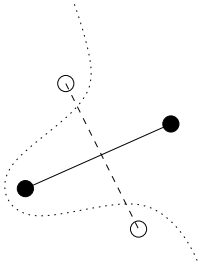
Quindi $VC(\mathcal{H}) \geq 3$. Cosa succede con 4 punti ?

VC-dimension: Esempio

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

Quindi $VC(\mathcal{H}) \geq 3$. Cosa succede con 4 punti ? Non si riesce a frammentare 4 punti!!

Infatti esisteranno sempre due coppie di punti che se unite con un segmento provocano una intersezione fra i due segmenti e quindi, ponendo ogni coppia di punti in classi diverse, per separarli non basta una retta, ma occorre una curva. Quindi $VC(\mathcal{H}) = 3$



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

VC-dimension

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

- Per ogni S tale che $\mathcal{H}$ frammenta S si ha $|\mathcal{H}| \geq 2^{|S|}$, infatti $\mathcal{H}$ può realizzare tutte le possibili dicotomie di S, che sono esattamente $2^{|S|}$.
- Scegliendo un S per cui vale $|S| = VC(\mathcal{H})$ si ottiene $|\mathcal{H}| \geq 2^{VC(\mathcal{H})}$

Quindi, applicando $\log_2$ ad entrambi i membri dell'ultima disuguaglianza, possiamo concludere che $\log_2(|\mathcal{H}|) \geq VC(\mathcal{H})$

Compiti di Classifi cazione più Complessi

In molte applicazioni del mondo reale non è sufficiente apprendere funzioni booleane con ingressi binari.

Gli Alberi di Decisione sono particolarmente adatti a trattare:

- istanze rappresentate da coppie attributo-valore;
- funzioni target con valori di output discreti (in generale più di 2 valori);
- esempi di apprendimento che possono contenere errori e/o avere valori mancanti.

Inoltre, algoritmi di apprendimento per Alberi di Decisione sono in genere molto veloci.

Per questi motivi gli Alberi di Decisione sono molto utilizzati in applicazioni pratiche.

Giocare a Tennis!!

E' la giornata ideale per giocare a Tennis ?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No