Apprendimento PAC

PAC Learning (Probably Approximately Correct Learning)

Assume che: input ed output sono binari, non c'è rumore, le istanze sono estratte da X concordemente ad una distribuzione di probabilità $\mathcal{D}$ *arbitraria* ma *stazionaria*.

Il framework di apprendimento PAC cerca di rispondere alle seguenti domande:

- Sotto quali condizioni apprendere con successo è possibile o impossibile ?
- Sotto quali condizioni si può assicurare che un particolare algoritmo di apprendimento apprenda con successo ?

Apprendimento PAC

Consideriamo una classe C di possibili concetti target (funzioni che vogliamo apprendere) definita su uno Spazio delle Istanze X (con istanze di dimensione m), ed un algoritmo di apprendimento L che utilizza uno Spazio delle Ipotesi $\mathcal{H}$.

Def.: C è PAC-apprendibile da L usando $\mathcal{H}$ se per ogni

- $c \in C$,
- distribuzione $\mathcal{D}$ su X,
- ϵ tale che $0 < \epsilon < 1/2$,
- δ tale che $0 < \delta < 1/2$,

l'algoritmo di apprendimento L con probabilità almeno $(1 - \delta)$ restituisce una 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 necessario per rappresentare c).

Apprendimento PAC

Si può dimostrare che assumendo:

- $c \in \mathcal{H}$
- L consistente, cioè che restituisce una ipotesi h consistente con Tr, o equivalentemente $h \in VS_{\mathcal{H},Tr}$ (ad esempio, **Find-S** è consistente)

allora, con probabilità almeno $(1-\delta)$, L restituisce una ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) < \epsilon$ se il numero di esempi di apprendimento n soddisfa 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 naturale

Prova...

Idea base: bisogna dare un limite al numero di esempi necessario ad assicurare che il Version Space (ricordiamo che L è consistente) non contiene ipotesi non "accettabili":

$$(\forall h \in VS_{\mathcal{H},Tr})\ error_{\mathcal{D}}(h) < \epsilon$$

Prova...

$$(\forall h \in VS_{\mathcal{H},Tr})\ error_{\mathcal{D}}(h) < \epsilon$$

Primo risultato: se $|\mathcal{H}| < \infty$, $|Tr| = n > 0$ e Tr è costituito da esempi di un concetto target c estratti indipendentemente ed a caso, allora per ogni $0 \leq \epsilon \leq 1$, la probabilità che la disuguaglianza **NON** sia soddisfatta è minore od uguale a $|\mathcal{H}|e^{-\epsilon n}$:

- siano $h_1, \ldots, h_k$ tutte le ipotesi con $error_{\mathcal{D}}(h) \geq \epsilon$; la disuguaglianza NON è soddisfatta se e solo se ALMENO una di tali ipotesi è consistente con Tr
- la probabilità che una di tali ipotesi sia consistente con un singolo esempio è $(1-\epsilon)$, e quindi per n esempi indipendenti la probabilità è $(1-\epsilon)^n$
- poiché esistono k di tali ipotesi, la probabilità che ci interessa è a più $k(1-\epsilon)^n \leq |\mathcal{H}|(1-\epsilon)^n$ (poiché $k \leq |\mathcal{H}|$)
- infine, poiché $(1-\epsilon) \leq e^{-\epsilon}$ se $0 \leq \epsilon \leq 1$, abbiamo $|\mathcal{H}|(1-\epsilon)^n \leq |\mathcal{H}|e^{-\epsilon n}$

Prova...

Cosa abbiamo ottenuto: abbiamo un limite superiore alla probabilità che usando un Tr con n esempi, il Version Space contenga qualche ipotesi CATTIVA, cioè che L restituisca h, una delle ipotesi cattive, per cui $error_{\mathcal{D}}(h) \geq \epsilon$!

Quindi, se vogliamo che tale probabilità sia inferiore a livello desiderato δ

$$|\mathcal{H}|e^{-\epsilon n} \leq \delta$$

bisogna usare un valore per n per cui

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

o, alternativamente, se si usa un tale valore per n, con probabilità $1 - \delta$ l'ipotesi h restituita da L avrà $error_{\mathcal{D}}(h) < \epsilon$

Apprendimento PAC: esempio $\mathcal{H}_4$

Congiunzione di m letterali

- Spazio delle Istanze $\rightarrow$ stringhe di m bit: $X = \{s | s \in \{0,1\}^m\}$
- Spazio delle Ipotesi $\rightarrow$ tutte le sentenze logiche che riguardano i letterali $l_1, \ldots, l_m$ (anche in forma negata, $\neg l_i$) e che contengono solo l'operatore $\wedge$ (**and**):

$$\mathcal{H} = \{f_{\{i_1,\ldots,i_j\}}(s) | f_{\{i_1,\ldots,i_j\}}(s) \equiv L_{i_1} \wedge L_{i_2} \wedge \cdots \wedge L_{i_j},$$
$$\text{dove } L_{i_k} = l_{i_k} \text{ oppure } \neg l_{i_k}, \{i_1, \ldots, i_j\} \subseteq \{1, \ldots, 2m\}\}$$

Notare che se in una formula un letterale compare sia affermato che negato, allora la formula ha sempre valore di verità $false$ (formula non soddisfacibile)

Quindi, tutte le formule che contengono almeno un letterale sia affermato che negato sono equivalenti alla funzione che vale sempre $false$

Apprendimento PAC: esempio $\mathcal{H}_4$

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

Apprendimento PAC: esempio $\mathcal{H}_4$

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 $\mathcal{H}_4$

Apprendimento PAC: esempio $\mathcal{H}_4$

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 consistente

Apprendimento PAC: esempio $\mathcal{H}_4$

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 consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

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

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

Apprendimento PAC: esempio $\mathcal{H}_4$

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 consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

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

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

- per ogni esempio di apprendimento **Find-S** impiega tempo lineare nella dimensione della ipotesi corrente (e tale dimensione è $\geq size(c)$) e nella dimensione dell'input (m), quindi di nuovo polinomiale, e globalmente è polinomiale per Tr

Apprendimento PAC: esempio $\mathcal{H}_4$

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 consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

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

  quindi n è polinomiale in $1/\epsilon$, $1/\delta$, m, e $size(c)$ (che non compare)
- per ogni esempio di apprendimento **Find-S** impiega tempo lineare nella dimensione della ipotesi corrente (e tale dimensione è $\geq size(c)$) e nella dimensione dell'input (m), quindi di nuovo polinomiale, e globalmente è polinomiale per Tr

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

Apprendimento PAC

Usando la disuguaglianza precedente ed altre considerazioni è possibile mostrare che alcune classi di concetti non sono PAC-apprendibili dato uno specifico algoritmo di apprendimento L e $\mathcal{H}$.

In particolare è possibile mostrare che:

- se $\mathcal{H}$ contiene tutte le funzioni booleane definite su X allora $C = \mathcal{H}$ non è PAC-apprendibile da algloritmi consistenti
- esistono classi di concetti C che non sono PAC-apprendibili da algoritmi consistenti che usano C come Spazio delle Ipotesi, tuttavia diventano PAC-apprendibili se uno Spazio delle Ipotesi "più grande" è usato, cioè $C \subset \mathcal{H}$

Il problema con la disuguaglianza data è che questa non può essere usata se $|\mathcal{H}| = \infty$

Tuttavia, il fattore chiave non è quante funzioni diverse sono contenute in $\mathcal{H}$, ma quante funzioni "utili" sono in $\mathcal{H}$: VC-dimension dà una risposta a questa domanda

Apprendimento PAC

Si può mostrare che, se assumiamo:

- $c \in \mathcal{H}$
- L consistente

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

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

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

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

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

- per ogni S tale che $\mathcal{H}$ frammenta S abbiamo $|\mathcal{H}| \geq 2^{|S|}$, infatti $\mathcal{H}$ può implementare tutte le possibili dicotomie di S, che sono esattamente $2^{|S|}$.
- scegliendo un S tale che $|S| = VC(\mathcal{H})$, otteniamo $|\mathcal{H}| \geq 2^{VC(\mathcal{H})}$

Quindi, applicando $\log_2$ ad entrambi i lati della disuguaglianza, possiamo concludere che $\log_2(|\mathcal{H}|) \geq VC(\mathcal{H})$

Mistake Bounds per algoritmi On-line

Quando si considerano algoritmi on-line per l'apprendimento di concetti, è ragionevole essere interessati ad un limite superiore al numero di errori commessi prima di apprendere *esattamente* il concetto target

Il **Modello Mistake Bound** è stato definito per questo scopo:

- le istanze x_i sono presentate ad L una alla volta
- data una istanza x, L deve "indovinare" il valore target $c(x)$
- solo dopo, il valore corretto è fornito ad L ai fini dell'apprendimento
- se la predizione di L era sbagliata, allora si ha un errore (mistake)

Bisogna rispondere alla seguente domanda:

> "Quanti errori farà L prima di apprendere esattamente il concetto target ?"

Mistake bound per la versione on-line di Find-S

L'algoritmo **Find-S** può essere usato in versione on-line !

/* versione on-line di Find-S per congiunzione di m letterali */

inizializza h alla ipotesi più specifica $h \equiv l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \wedge \cdots \wedge l_m \wedge \neg l_m$

do forever /* in effetti, fino a quando arrivano esempi */

leggi la nuova istanza x e predici $h(x)$

leggi $c(x)$, cioè il valore target per x

/* errore ! */

if $(h(x) \neq c(x)$ rimuovi da h ogni letterale che non è soddisfatto da x

Assumendo $c \in \mathcal{H}$, e assenza di rumore negli esempi

E' possibile dare un limite superiore al numero di errori ?

Mistake bound per la versione on-line di Find-S

E' possibile dare un limite superiore al numero di errori ?

Si!

- l'ipotesi iniziale contiene $2m$ letterali
- dopo il primo sbaglio (che occorre subito!), solo m letterali rimangono nella ipotesi corrente
- dopo ogni altro errore, almeno un letterale è rimosso dalla ipotesi corrente

Quindi il numero totale di errori che **Find-S** commette prima di convergere alla ipotesi corretta è $\leq m + 1$

Mistake bound per Halving

L'algoritmo **Halving** è una versione on-line di **Candidate-Elimination**

/* Aggiorna il Version Space (VS) on-line */

inizializza S e G come in **Candidate-Elimination**

do forever /* in effetti, fino a che VS contiene più di 1 ipotesi */

leggi una nuova istanza x e predici tramite voto a maggioranza delle ipotesi in VS

leggi $c(x)$, cioè il valore target per x

aggiorna S e G in modo da rimuovere da VS le ipotesi h per cui $h(x) \neq c(x)$

/* un errore occorre solo se la maggioranza delle ipotesi in VS sono sbagliate */

Assumendo $c \in \mathcal{H}$, $|\mathcal{H}| < \infty$, e nessun rumore negli esempi

E' possibile dare un limite superiore al numero di errori ?

Mistake bound per Halving

E' possibile dare un limite superiore al numero di errori ?

Si!

- il VS iniziale contiene $|\mathcal{H}|$ ipotesi
- dopo un errore (che occorre quando il voto a maggioranza è sbagliato), ALMENO $|VS|/2$ ipotesi sono rimosse da VS

Quindi il numero totale di errori che **Halving** commette prima di convergere alla ipotesi corretta è $\leq log_2|\mathcal{H}|$

ATTENZIONE: **Halving** può convergere alla ipotesi corretta senza errori !! Accade quando il voto a maggioranza è sempre corretto e solo le ipotesi minoritarie sono rimosse da VS

Optimal Mistake Bounds

Fino ad ora abbiamo considerato limiti al numero di errori (caso pessimo) per algoritmi *specifici*.

E' possibile dare il più basso fra i limiti di errore su tutti i possibili algoritmi di apprendimento (*optimal mistake bound*) ?

- Assumiamo $C = \mathcal{H}$
- Sia $M_L(c)$ il massimo su tutte le possibili sequenze di esempi di apprendimento del numero di errori commesso da L per apprendere esattamente il concetto $c \in C$
- Sia $M_L(C) \equiv \max_{c \in C} M_L(c)$

Definizione: Sia C una classe non vuota di concetti. L' **optimal mistake bound** per C, denotato *Opt(C)*, è il minimo su tutti i possibili algoritmi di apprendimento L di $M_L(C)$:

$$Opt(C) \equiv \min_{L \in AlgoritmiApprendimento} M_L(C)$$

Optimal Mistake Bounds

Littlestone (1987) ha mostrato che

$$VC(C) \leq Opt(C) \leq M_{Halving}(C) \leq log_2(|C|)$$

Un esempio finale di Mistake Bound...

Fino ad ora abbiamo visto esempi di algoritmi di apprendimento capaci di trattare con esempi di apprendimento consistenti (cioè che non danno luogo a contraddizioni)

Non è difficile modificare **Halving** in modo da ottenere un algoritmo capace di trattare dati inconsistenti: Algoritmo **Weighted-Majority**

- l'idea base è di assegnare ad ogni ipotesi (ma possiamo estendere l'idea ad ogni insieme di predittori) un peso che è usato per pesare il voto associato ad ogni ipotesi
- quindi, invece di considerare il voto a maggioranza, consideriamo il voto a maggioranza pesata
- quando una ipotesi commette un errore, invece di rimuoverla, si moltiplica il peso a lei associata per un fattore $\beta < 1$ (riduzione del peso)

Weighted-Majority

/* Weighted-Majority: l' algoritmo Halving... con pesi! */

per ogni ipotesi (o predittore) h_j definire il peso w_j, inizialmente uguale a 1

do forever /* in effetti, fino a che ci sono esempi */

leggi una nuova istanza x

calcola $q_0 \leftarrow \sum_{j:h_j(x)=0} w_j$ e $q_1 \leftarrow \sum_{j:h_j(x)=1} w_j$

if $q_0 > q_1$ **then** predici 0

if $q_1 > q_0$ **then** predici 1

if $q_0 = q_1$ **then** predici 0 o 1 a caso

read $c(x)$, i.e. the target value for x

for each h_j **do**

if $h_j(x) \neq c(x)$ **then** $w_j \leftarrow \beta w_j$ /* $0 \leq \beta < 1$ */

Cosa accade se $\beta = 0$?

Mistake bound per Weighted-Majority

Teorema: Sia $seq \equiv (x_1, c(x_1)), (x_2, c(x_2)), \ldots$ una qualunque sequenza di esempi di allenamento e sia k il numero minimo di errori commesso su seq da una qualunque ipotesi dello Spazio delle Ipotesi. Allora il numero di errori su seq commesso da **Weighted-Majority** usando $\beta = \frac{1}{2}$ è al più

$$2.4(k + \log_2(|\mathcal{H}|))$$

(lo proviamo!)

Per $0 \leq \beta < 1$, Littlestone e Warmuth (1991) hanno provato che il bound di sopra diventa

$$\frac{k \log_2 \frac{1}{\beta} + \log_2(|\mathcal{H}|)}{\log_2 \frac{2}{1+\beta}}$$

(questo non lo proviamo...)

Prova...

Per provare il teorema confrontiamo il peso finale w^* della migliore ipotesi h^*, cioè quella che produce il numero minore di errori k, con la somma finale dei pesi su tutte le ipotesi $W = \sum_{j=1}^{|\mathcal{H}|} w_j$ (naturalmente abbiamo $w^* \leq W$)

- **fatto 1:** $w^* = (\frac{1}{2})^k$, infatti h^* commette esattamente k errori
- **fatto 2:** per ogni errore di **Weighted-Majority**, W si riduce di al più $\frac{3}{4}W$, infatti se **Weighted-Majority** commette un errore, tutte le ipotesi h_j che hanno contribuito alla maggioranza pesata avranno il loro peso ridotto di metà; quindi almeno $\frac{1}{2}W$ (il voto a maggioranza pesata è $\geq \frac{1}{2}W$) del peso totale è ridotto a metà, cioè il nuovo peso totale è **minore o uguale a**

$$\underbrace{\frac{1}{2}W}_{\text{minoranza}} + \underbrace{\frac{1}{2}(\frac{1}{2}W)}_{\text{maggioranza}} = \frac{3}{4}W$$

- **fatto 3:** se M è il numero totale di errori prodotti da **Weighted-Majority**, allora per il peso totale finale $W \leq |\mathcal{H}|(\frac{3}{4})^M$, a causa del fatto 2 e la condizione iniziale $W = |\mathcal{H}|$

Prova...

Ricordando che $w^* \leq W$, ed a causa dei fatti 1-3, abbiamo

$$(\tfrac{1}{2})^k = w^* \leq W \leq |\mathcal{H}|(\tfrac{3}{4})^M$$

Prendendo il logaritmo (in base 2) dei termini più a sinistra e più a destra, otteniamo

$$k \log_2(\tfrac{1}{2}) \leq \log_2(|\mathcal{H}|) + M \log_2(\tfrac{3}{4})$$

ed isolando M otteniamo (notare che $\log_2(\frac{3}{4})$ è una quantità negativa)

$$M \leq \frac{k + \log_2(|\mathcal{H}|)}{-\log_2(\frac{3}{4})} \leq 2.4(k + \log_2(|\mathcal{H}|))$$