Computational Game Theory

Vincenzo Bonifaci

24 maggio 2012

5 Regret Minimization

Consideriamo uno scenario in cui un agente deve selezionare, più volte nel tempo, una decisione tra un insieme di N disponibili: $X = \{1, 2, \ldots, N\}$. Ad ogni passo t, l'agente sceglie, deterministicamente o probabilisticamente, una delle N azioni. Formalmente, vogliamo un algoritmo online che scelga, al tempo t, una distribuzione di probabilità $p^t = (p_1^t, p_2^t, \ldots, p_N^t)$.

In seguito a ciascuna scelta, l'ambiente (o un "avversario") associa ad ogni azione i una perdita $\ell_i^t \in \{0, 1\}$ (osservabile dall'algoritmo). La perdita (istantanea) dell'algoritmo, ℓ_A^t, sarà la media pesata della perdita delle azioni: $\ell_A^t = \sum_{i \in X} p_i^t \ell_i^t$. Il processo si ripete per $t = 1, 2, \ldots$

Ad esempio, possiamo immaginare uno scenario in cui le N azioni corrispondano a N titoli di borsa e che $\ell_i^t = 1$ se il titolo i perde quota, $\ell_i^t = 0$ se il titolo i tiene. In questo caso una distribuzione di probabilità può essere vista come una diversificazione del portafoglio titoli.

Dopo T passi di questo processo possiamo considerare le seguenti quantità:

- La perdita complessiva dell'azione i: $L_i^T = \sum_{t=1}^T \ell_i^t$.
- La perdita complessiva dell'algoritmo A: $L_A^T = \sum_{t=1}^T \ell_A^t$.
- La perdita complessiva dell'azione migliore: $L_{\min}^T = \min_{i \in X} L_i^T$.
- Il *rimorso* (regret) dell'algoritmo: la differenza $R = L_A^T - L_{\min}^T$.
- Il *rimorso per unità di tempo* dopo T passi: R/T.

Un buon algoritmo per regret minimization dovrebbe approssimativamente minimizzare la quantità R/T. In particolare sarebbe desiderabile avere $R/T \to 0$ per $T \to \infty$.

5.1 Algoritmi per Regret Minimization

Teorema 5.1. *Per qualunque algoritmo deterministico A, esiste una sequenza di perdite tali che $L_A^T = T$ e $L_{\min}^T \leq T/N$. Il rimorso per unità di tempo di A è quindi almeno $1 - 1/N$.*

Dimostrazione. Si consideri la sequenza che, al passo t, assegna perdita 1 all'azione selezionata dall'algoritmo A e 0 a tutte le altre azioni. Questa sequenza è ben definita, poiché l'algoritmo è deterministico. Per costruzione abbiamo $L_A^T = T$, poiché l'algoritmo perde 1 ad ogni passo.

Inoltre, tra tutti gli $N \cdot T$ valori ℓ_i^t, con $i = 1, \ldots, N$, $t = 1, \ldots, T$, esattamente T sono pari ad 1 (quelli corrispondenti alle scelte dell'algoritmo A), tutti gli altri sono pari a 0. Il numero medio di perdite complessive per azione è quindi T/N. Deve quindi esistere un'azione con perdita al più T/N. □

Algoritmo GREEDY. Un semplice algoritmo deterministico è il seguente:

1. Nella prima fase, seleziona deterministicamente l'azione con indice 1.
2. Sia $S^t = \{i \in X : L_i^t = L_{\min}^t\}$.

   Al tempo t, seleziona deterministicamente l'azione con indice minimo nell'insieme S^{t-1}.

Teorema 5.2. *Per l'algoritmo* GREEDY *si ha il bound*

$$L_{\text{GREEDY}}^T \leq N \cdot L_{\min}^T + N - 1.$$

Dimostrazione. Osserviamo che se $L_{\min}^t = L_{\min}^{t-1}$, allora $S^t \subseteq S^{t-1}$. Ad ogni passo t nel quale GREEDY subisce una perdita pari ad 1 e $L_{\min}^t$ non aumenta, la cardinalità dell'insieme S^t diminuisce strettamente. Poiché $1 \leq |S^t| \leq N$, ciò può avvenire al massimo N volte prima che $L_{\min}^t$ aumenti di 1. Quindi GREEDY subisce al più N perdite tra un incremento di $L_{\min}^t$ e il successivo. Più precisamente, $L_{\text{GREEDY}}^t \leq N \cdot L_{\min}^t + N - |S^t|$. □

Algoritmo Randomized Greedy (RG).

1. Inizializza $p_i^1 = 1/N$ per ciascun $i \in X$.
2. Sia $S^t = \{i \in X : L_i^t = L_{\min}^t\}$.

   Al tempo t, poni

$$p_i^t = \begin{cases} 1/|S^{t-1}| & \text{se } i \in S^{t-1} \\ 0 & \text{se } i \notin S^{t-1}. \end{cases}$$

Teorema 5.3. *Per l'algoritmo Randomized Greedy* (RG) *si ha il bound*

$$L_{\text{RG}}^T \leq (1 + \ln N) \cdot L_{\min}^T + \ln N.$$

Dimostrazione. Ad ogni passo t nel quale $L_{\min}^t = L_{\min}^{t-1}$ e l'insieme S^t passa da cardinalità n' a cardinalità $n' - k$, la perdita (attesa) dell'algoritmo RG è esattamente k/n', poiché ogni azione in S^{t-1} ha peso $1/n'$. Poiché $k/n' \leq 1/n' + 1/(n'-1) + \ldots + 1/(n'-k+1)$, possiamo assumere che S^t diminuisca di un elemento alla volta. Finché $L_{\min}^t$ non aumenta, quindi, la perdita dell'algoritmo è al massimo

$$1/N + 1/(N-1) + \ldots + 1/2 + 1/1 \leq 1 + \ln N.$$

Più precisamente, quanto detto mostra che

$$L_{\text{RG}}^t \leq (1 + \ln N) \cdot L_{\min}^t + 1/N + 1/(N-1) + \ldots + 1/(|S^t| + 1).$$

□

Algoritmo Weighted Majority (WM) [Littlestone & Warmuth, 1994].

1. Sia η un parametro tale che $0 \leq \eta \leq 1/2$.
2. Inizializza $w_i^1 = 1$ per ogni $i \in X$.

   Al tempo t:
3. Associa all'azione i peso $w_i^t = (1-\eta)^{L_i^t}$. Equivalentemente,

$$w_i^t = \begin{cases} w_i^{t-1} & \text{se } \ell_i^t = 0, \\ (1-\eta)w_i^{t-1} & \text{se } \ell_i^t = 1. \end{cases}$$

4. Poni $p_i^t = w_i^t / \sum_{j=1}^N w_j^t$.

Teorema 5.4. *Per l'algoritmo Weighted Majority (WM) si ha il bound*

$$L_{\text{WM}}^T \leq (1+\eta)L_{\min}^T + \frac{\ln N}{\eta}.$$

Di conseguenza, se $\eta = \sqrt{(\ln N)/T}$ *e* $T \geq 4 \ln N$*, si ha*

$$L_{WM}^T \leq L_{\min}^T + 2\sqrt{T \ln N}$$

e quindi

$$\frac{R}{T} \to 0 \text{ quando } T \to \infty.$$

Dimostrazione. Consideriamo la seguente funzione (di "potenziale"):

$$\Phi^t := \sum_{i \in X} w_i^t.$$

Poiché i pesi w non vengono mai aumentati dall'algoritmo, Φ è una funzione non crescente nel tempo. Mostriamo ora che quando l'algoritmo WM incorre un costo significativo, Φ deve decrescere notevolmente. Questo, unito al fatto che $\Phi^{T+1} \geq \max_i w_i^{T+1} = (1-\eta)^{L_{\min}^T}$, ci permetterà di arrivare alla conclusione voluta.

Sia $F^t := \left(\sum_{i:\ell_i^t=1} w_i^t\right)/\Phi^t$ la frazione del peso totale Φ^t che si ha sulle azioni che incorrono perdita 1 al tempo t; quindi F^t è anche la perdita attesa dell'algoritmo WM al tempo t. Ora, l'algoritmo moltiplica il peso di ciascuna delle azioni con perdita 1 per un fattore $1-\eta$, mentre i pesi restanti sono inalterati. Quindi,

$$\Phi^{t+1} = \Phi^t - \eta F^t \Phi^t = (1 - \eta F^t)\Phi^t.$$

(In altre parole, la proporzione di peso rimossa dal sistema ad ogni passo è esattamente proporzionale alla perdita attesa dell'algoritmo online. Usando il fatto che $\Phi^1 = N$ e $\Phi^{T+1} \geq (1-\eta)^{L_{\min}^T}$, otteniamo

$$(1-\eta)^{L_{\min}^T} \leq \Phi^{T+1} = \Phi^1 \prod_{t=1}^T (1 - \eta F^t) = N \prod_{t=1}^T (1 - \eta F^t).$$

Prendendo i logaritmi di ciascuna espressione, e ricordando che $\ln(1-x) \leq -x$,

$$\begin{aligned} L^T_{\min} \ln(1-\eta) &\leq (\ln N) + \sum_{t=1}^{T} \ln(1-\eta F^t) \\ &\leq (\ln N) - \sum_{t=1}^{T} \eta F^t \\ &= (\ln N) - \eta L^T_{\mathrm{WM}}. \end{aligned}$$

Riarrangiando i termini, e ricordando che $-\ln(1-x) \leq x + x^2$ per $x \in [0, 1/2]$,

$$\begin{aligned} L^T_{\mathrm{WM}} &\leq \frac{-L^T_{\min} \ln(1-\eta)}{\eta} + \frac{\ln N}{\eta} \\ &\leq (1+\eta) L^T_{\min} + \frac{\ln N}{\eta}. \end{aligned}$$

Quando $\eta = \sqrt{(\ln N)/T}$ e $T \geq 4 \ln N$, si ha $\eta \leq 1/2$ e quindi il bound può essere applicato; notando che si ha sempre $L^T_{\min} \leq T$, otteniamo

$$\begin{aligned} L^T_{\mathrm{WM}} &\leq L^T_{\min} + \eta L^T_{\min} + \frac{\ln N}{\eta} \\ &\leq L^T_{\min} + \sqrt{(\ln N)/T} \cdot T + \frac{\ln N}{\sqrt{(\ln N)/T}} \\ &= L^T_{\min} + 2\sqrt{T \ln N}. \end{aligned}$$

□

Osservazione 5.1. Osserviamo che per poter usare il valore $\eta = \sqrt{(\ln N)/T}$ nell algoritmo avremmo bisogno di conoscere in anticipo il valore T del numero di passi, informazione che in realtà non è a disposizione di un algoritmo online. A questo problema si può ovviare mantenendo una stima del valore di T e raddoppiandola quando il numero di passi è diventato troppo grande. Si può dimostrare che questa versione modificata mantiene un valore del regret che è entro un fattore costante da quello del teorema: $L^T_{\mathrm{WM-online}} \leq L^T_{\min} + O(\sqrt{T \ln N})$.

Osservazione 5.2. L'algoritmo può essere generalizzato per trattare il caso di perdite anche frazionarie, quindi in $[0, 1]$ anziché in $\{0, 1\}$. In questo caso la regola di aggiornamento dei pesi è

$$w_i^t = (1-\eta)^{\ell_i^t} \cdot w_i^{t-1}.$$

Il bound del Teorema 5.4 può essere esteso a questa generalizzazione.

5.2 Regret Minimization e giochi a somma zero

Consideriamo un gioco a somma zero Γ a due giocatori, con insiemi delle azioni S_1 e S_2 e funzioni di utilità u_1 e u_2. Abbiamo visto come esista un valore $v \in \mathbb{R}$ associabile a Γ che è esattamente l'utilità

che si ha all'equilibrio: il primo giocatore può ottenere v, ma non più di v, e il secondo giocatore può perdere v, ma non meno di v; v è detto il *valore di* Γ.

Mostriamo che se il gioco Γ viene giocato ripetutamente e uno dei giocatori segue una strategia per la minimizzazione del rimorso, questo gli garantisce di convergere al valore v nel tempo. Non è quindi necessario che il giocatore sappia risolvere un problema di programmazione lineare per giocare Γ al meglio; è sufficiente giocare molte volte, se si fa tesoro dell'esperienza attraverso un algoritmo di apprendimento quale Weighted Majority.

Teorema 5.5. *Sia Γ un gioco a somma zero di valore v. Se il secondo giocatore gioca Γ per T passi usando un algoritmo A dal rimorso R, allora la sua perdita media $\frac{1}{T}L_A^T$ è al più $v + R/T$.*

Dimostrazione. Sia q la strategia mista che corrisponde alle frequenze osservate delle azioni giocate dal giocatore 1; in altre parole, $q_j = \sum_{t=1}^{T} x_j^t/T$, dove x_j^t è la probabilità data dal giocatore 1 all'azione j al tempo t. Per la teoria dei giochi a somma zero, il giocatore 2 ha una azione (pura) $b_k \in S_2$ che è una miglior risposta alla strategia mista q del giocatore 1. Quindi, il giocatore 2 ha costo atteso $\leq v$ quando il giocatore 1 gioca q e il giocatore 2 gioca b_k.

In altre parole, se il giocatore 2 avesse sempre giocato b_k, avrebbe pagato un costo complessivo $v \cdot T$. Quindi $L_{\min}^T \leq L_k^T \leq v \cdot T$. Poiché il giocatore 2 sta seguendo un algoritmo A con rimorso R, abbiamo

$$L_A^T \leq L_{\min}^T + R \leq v \cdot T + R$$
$$\frac{1}{T}L_A^T \leq v + \frac{R}{T}.$$

□