(F94) Metodi statistici per l'apprendimento

Consistenza e algoritmi nonparametrici

Docente: *Nicolò Cesa-Bianchi* versione 16 aprile 2018

Fissiamo un modello statistico (D, η) per classificazione binaria. Per ogni algoritmo di apprendimento A, denotiamo con $A(S_m)$ il classificatore prodotto da A quando il training set S_m contenente m esempi è fornito in input. Denotiamo con $\mathrm{er}\big(A(S_m)\big)$ il rischio di questo classificatore rispetto al modello (D, η). Quindi, se $A(S_m) = h$ allora $\mathrm{er}\big(A(S_m)\big) = \mathbb{P}\big(h(\boldsymbol{X}) \neq Y\big)$. Infine, indichiamo con $\mathrm{er}(f^*)$ il rischio del classificatore Bayesiano ottimo f^*, sempre rispetto a (D, η).

L'algoritmo A è detto consistente rispetto a (D, q) quando

$$\lim_{m\to\infty} \mathbb{E}\Big[\mathrm{er}\big(A(S_m)\big)\Big] = \mathrm{er}(f^*)$$

dove il valore atteso è rispetto all'estrazione del training set S_m da (D, η). Quindi, la consistenza è una proprietà asintotica che certifica la capacità dell'algoritmo di raggiungere in media le prestazioni del Bayesiano ottimo al crescere del training set.

Se indichiamo con k-NN l'algoritmo k-Nearest Neighbor, allora per ogni (D, η) vale

$$\lim_{m\to\infty} \mathbb{E}\Big[\mathrm{er}\big(k\text{-NN}(S_m)\big)\Big] \leq \mathrm{er}(f^*) + 2\sqrt{\frac{\mathrm{er}(f^*)}{k}}\ . \tag{1}$$

Perciò per ogni k fissato k-NN non è consistente. Possiamo confrontare (1) con il risultato

$$\mathbb{E}\Big[\mathrm{er}\big(k\text{-NN}(S_m)\big)\Big] \leq \left(1 + \sqrt{\frac{8}{k}}\right)\mathrm{er}(f^*) + \mathcal{O}\big(k\, m^{-1/(d+1)}\big)\ .$$

enunciato in precedenza sotto l'assunzione

$$\big|\eta(\boldsymbol{x}) - \eta(\boldsymbol{x}')\big| \leq c\,\big\|\boldsymbol{x} - \boldsymbol{x}'\big\| \qquad \text{per ogni } \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}\ . \tag{2}$$

Se in (1) rendiamo k dipendente da m in modo che cresca non troppo velocemente, cioè se $k = k_m$ tale che $k_m \to \infty$ e $k_m = o(m)$, allora è possibile dimostrare che k-NN diventa consistente.

Sotto determinate assunzioni su (D, η), è possibile dimostrare che anche l'algoritmo greedy di costruzione di classificatori ad albero diventa consistente quando ogni foglia è finale per un numero sufficiente di punti del training set (dettagli omessi).

Il motivo per cui un algoritmo consistente può risultare in pratica peggiore di uno non consistente è legato alla velocità di convergenza. Infatti, la consistenza per ogni modello (D, η) può essere pagata con una velocità di convergenza *arbitrariamente lenta* al Bayes error, come stabilito dal seguente risultato.

Teorema 1 (No Free Lunch) *Sia $a_1, a_2, \ldots$ una successione convergente a zero di numeri positivi tali che $\frac{1}{16} \geq a_1 \geq a_2 \geq \cdots$. Per ogni algoritmo di apprendimento A esiste un modello statistico (D, η) tale che $\mathrm{er}(f^*) = 0$ e simultaneamente $\mathbb{E}\big[\mathrm{er}\big(A(S_m)\big)\big] \geq a_m$ per ogni $m \geq 1$.*

Si noti che se $\text{er}(f^*) = 0$ allora dev'essere $\eta(\boldsymbol{x}) \in \{0, 1\}$ per ogni $\boldsymbol{x}$. Ovvero η è discontinua e la condizione (2) non può essere soddisfatta. Si noti anche che il Teorema 1 non impedisce a un algoritmo consistente di convergere rapidamente per specifiche distribuzioni (D, η). Analogamente, un algoritmo non consistente può convergere rapidamente a specifici valori di rischio a patto che siano superiori al Bayes error. Il Teorema 1 dice solo che se A arriva al Bayes error, allora non è possibile che ci arrivi in fretta per tutte le distribuzioni

Gli algoritmi consistenti, come k-NN e gli algoritmi per i predittori ad albero, vengono anche detti algoritmi **nonparametrici**. Questa terminologia si riferisce al fatto che i classificatori prodotti non sono rappresentabili con un numero predeterminato di parametri (come ad esempio succede per i classificatori lineari). Infatti, la struttura di un classificatore nonparametrico non è fissata, ma viene determinata dai dati di training (si pensi alla costruzione di un predittore ad albero guidata dal training set).

In generale,

- Se il training set è piccolo conviene utilizzare un algoritmo parametrico.
- Se d è grande conviene usare un algoritmo parametrico perché anche quando η soddisfa (2) la convergenza nonparametrica è comunque dell'ordine $\mathcal{O}\big(m^{-1/(d+1)}\big)$ mentre quella parametrica è dell'ordine $\mathcal{O}(m^{-1/2})$.
- Negli altri casi (dataset grandi con pochi attributi) possiamo provare ad usare un algoritmo nonparametrico.