UNIVERSITÀ DI PISA
Dipartimento di Informatica

MACHINE LEARNING

Appunti dalle lezioni del Prof. Alessio Micheli

Pasquale Miglionico
Guido Narduzzi
Enrico Negri
Filippo Quattrocchi
Francesco Zigliotto

ANNO ACCADEMICO 2019–2020

INDICE

1 INTRODUZIONE

1.1 LEZIONE DI GIOVEDÌ 26 SETTEMBRE

1.1.1 Contestualizzazione

L'obiettivo del Machine Learning è insegnare ad un sistema un compito preciso, costruendo un modello utilizzabile per predire il corretto output, dopo aver esaminato un gran numero di esempi. Tale metodo di apprendimento è detto *per generalizzazione*.

È utile per esempio quando l'approccio teorico a un determinato problema è difficilmente praticabile, o quando i dati in input sono poco accurati, affetti da errore o incompleti.

Osservazione 1.1. Il Machine Learning non consente di prevedere fenomeni *casuali* (come i numeri della lotteria). Inoltre non permette di inferire relazioni di *causalità* tra due fenomeni tramite la sola analisi dei dati.

Definizione 1.2. (Machine Learning). Il *Machine Learning* studia e propone metodi per inferire funzioni o correlazioni che, a partire da dati osservati, producano un buon *fit* dei *samples* forniti, generalizzandoli con ragionevole accuratezza.

1.1.2 Terminologia

Un *machine learning system* si compone di *dati*, *tasks*, *modelli*, *algoritmi di apprendimento* e *validazione*.

Definizione 1.3. (Dati). I *dati* rappresentano le *esperienze disponibili*. Possono essere organizzati in un certo numero l di istanze x_p (*samples*, *instances*), ciascuno contenente n attributi (*features*). Con $x_{p,j}$ indicheremo l'attributo j-esimo della p-esima istanza.

Osservazione 1.4. Se un attributo può assumere un numero finito di valori, risulta spesso conveniente rappresentarlo come un vettore di dimensione k (dove k è il numero dei valori possibili) con componenti tutte nulle a parte una.

Esempio 1.5. Se i valori possibili sono i tre colori *rosso* (R), *verde* (G), *blu* (B), si pone

$$R = (1,0,0), \quad G = (0,1,0), \quad B = (0,0,1). \tag{1.1}$$

Definizione 1.6. (Rumore, outliers). Il *rumore* è l'aggiunta di fattori esterni dovuta al processo di misura (e non alla legge soggiacente). Gli *outliers* sono dati che si collocano molto lontani rispetto agli altri.

I *tasks* sono in genere di due tipologie (ma ce ne sono altre):

Definizione 1.7. (Supervised learning). Nel *supervised learning* sono dati dei *samples* di una funzione f ignota, nella forma

$$\langle input, output \rangle$$

si tratta di trovare una buona approssimazione di f. Gli input sono detti anche *variabili indipendenti*, gli ouput *variabili dipendenti* o *risposte*. Se f è a valori discreti, il problema si dice di *classificazione*. Se l'output è costituito da valori reali, allora si parla di *regressione*.

Definizione 1.8. (Unsupervised learning). Nell'*unsupervised learning* non si dispone di input e output nelle istanze, ma solo di dati non etichettati. Un problema tipico è quello di raggruppare tali dati secondo determinati criteri.

Noi ci occuperemo soprattutto di supervised learning.

Definizione 1.9. (Modello, ipotesi). Il modello cerca di descrivere la relazione tra i dati con un *linguaggio*, legato alla rappresentazione dei dati. Le *ipotesi* sono le funzioni h_w proposte dal modello per approssimare la "vera" funzione f. Le ipotesi sono indicizzate da parametri (w) e formano uno *spazio delle ipotesi* H.

In generale non esiste un modello *ottimo*: se un modello di apprendimento è il migliore in qualche problema, sarà peggiore di altri in altri problemi. Questo concetto è noto come *No Free Lunch Theorem*, ovvero "non c'è un pranzo gratis" (mah, sarà qualche detto inglese, ndr). In ogni caso, questo non significa che tutti i modelli siano equivalenti.

Definizione 1.10. (Algoritmo di apprendimento). Un *algoritmo* di apprendimento si occupa di cercare nello spazio delle ipotesi H (di un modello fissato) la migliore approssimazione della funzione f.

Come definiamo *buona approssimazione*? Si utilizza una *loss function* $L\big(h(x), d\big)$, che misura la distanza tra $h(x)$ e d, dove d è il valore osservato (cioè $f(x)$ più eventualmente il rumore).

Definizione 1.11. (Errore). L'errore è definito da

$$E = \frac{1}{l}\sum_{i=1}^{l} L\big(h(x_i), d_i\big) \tag{1.2}$$

dove x_i sono le istanze e $d_i = f(x_i)$ i valori osservati.

Esempio 1.12. Nei problemi di regressione spesso come loss function si usa

$$L\big(h(x), d\big) = \big(d - h(x)\big)^2 \tag{1.3}$$

e in tal caso l'errore si dice *errore quadratico medio* (MSE).

Se siamo di fronte a un problema di classificazione, allora è più conveniente usare la loss function che vale 1 se i suoi due argomenti sono uguali (e dunque la classificazione è corretta) e 0 altrimenti.

Osservazione 1.13. In Machine Learning, quando si parla di *performance*, si fa riferimento all'accuratezza predittiva, non all'efficienza computazionale.

1.2 LEZIONE DI VENERDÌ 27 SETTEMBRE

1.2.1 Fitting e overfitting

Definizione 1.14. (Validazione). La *validazione* valuta la capacità di generalizzazione di una determinata ipotesi, misurandone l'accuratezza.

In generale non possiamo assumere che se un'ipotesi h approssima bene la funzione f sui samples di allenamento, allora h approssima f anche su nuove istanze. C'è per esempio il problema dell'*overfitting*:

Definizione 1.15. (Overfitting). L'*overfitting* avviene quando sottostimiamo l'errore sperimentale nel fitting e quindi aumenta il *vero* errore sui dati sconosciuti. L'overfitting avviene quindi se il modello è troppo complesso e quindi è in grado di *fittare il rumore*.

Esempio 1.16. (Fitting polinomiale). Supponiamo di avere una funzione reale e di voler risolvere il problema di regressione con un *fit polinomiale*. Le ipotesi sono della forma

$$y(x, w) = \sum_{i=0}^{M} w_i x^i \tag{1.4}$$

dove w è il vettore dei coefficienti w_i. Si cerca di minimizzare l'errore quadratico medio. Si osserva che se il polinomio ha grado troppo basso i *training samples* non vengono fittati correttamente (*underfitting*). Aumentando troppo il grado del polinomio nel modello, l'errore sui *training samples* diminuisce, mentre quello sui *test samples* aumenta (e i coefficienti del polinomio aumentano di molto in modulo): questo comportamento è tipico dell'*overfitting*.

Infine, a parità di grado, si nota che aumentando il numero di dati il fitting migliora molto, indipendentemente dal fatto che ci sia molto rumore.

1.2.2 Setting semplificato

Formalmente, quindi, disponiamo di

- una funzione f ignota da approssimare;
- un modello con un relativo spazio di ipotesi H;
- una *loss function* L;
- una distribuzione di probabilità $P(x, d)$ per lo spazio dei dati, dove $d(x)$ corrisponde alla distribuzione dei dati sperimentali ad x fissato (d corrisponde al valore di $f(x)$ misurato sperimentalmente, quindi affetto da rumore, e per lo stesso x naturalmente si possono avere più misure).

L'obiettivo teorico sarebbe minimizzare il *rischio*, cioè il *vero errore su tutti i dati*:

$$R = \int L(h_w(x), d)\, dP(x, d). \tag{1.5}$$

È tuttavia più praticabile lavorare con il *rischio empirico*, cioè l'errore sui *training samples*:

$$R_{\text{emp}} = \frac{1}{l} \sum L(d - h_w(x), d). \tag{1.6}$$

Chiaramente non bisogna minimizzare il rischio empirico, per via dell'overfitting. Occorre considerare insieme il rischio empirico e la complessitò.

Si può mostrare che, con probabilità $1 - \delta$, vale

$$R \leq R_{\text{emp}} + \varepsilon(1/l, VC, 1/\delta) \tag{1.7}$$

dove l è il numero di samples, VC è la "complessità del modello" (il *grado* del polinomio, nell'esempio del fit polinomiale) ed ε è un'opportuna funzione: solitamente si assume che ε sia direttamente proporzionale ai suoi due primi argomenti. Infatti, per grandi valori di l, il rischio R è minore. Mentre per grande complessità del modello il rischio empirico R_{emp} è minore ma il rischio R può aumentare, per via dell'overfitting.

1.2.3 Validazione

La validazione è composta da due fasi:

- la selezione del modello (*model selection*) si occupa di scegliere il modello più adatto a trattare il problema;
- il giudizio del modello (*model assessment*) valuta la capacità predittiva del modello su *test samples*.

In particolare si divide l'insieme di dati in TR (*training set*), VL (*validation set*) and TS (*test set*). A questo punto avviene la *model selection*, in cui

- si fissa un determinato modello e si usano i dati in TR per trovare la funzione h che minimizzi il rischio empirico;
- si usano i dati in VL per determinare la bontà del modello;
- si cicla sui due step sopra al fine di determinare il modello migliore.

Una volta ottenuto il modello migliore (già pronto per l'utilizzo, con tutti parametri fissati dai dati in TR), si effettua il *model assessment* e si valuta il comportamento sui nuovi dati presenti nel TS.

Esempio 1.17. (k-fold cross-validation). Per esempio, si può dividere l'insieme di dati per la *model selection* in k sottoinsiemi $D_1, \ldots, D_k$ e poi, per ogni $j = 1, \ldots, k$ utilizzare D_j come VL e tutti gli altri D_i come TR.

1.2.4 Matrice di confusione

[...]

2 ESEMPI

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

2.1 LEZIONE DI DOMENICA 22 SETTEMBRE

2.1.1 Alcune indicazioni

Un paio di proposte, per uniformità:

- Ogni lezione inizia con `\lecture{⟨data⟩}`, dove la data è del tipo `29/2`.
- Per creare paragrafi numerati, non va usato `\section`, ma `\subsection`;
- Non usate il grassetto, ma il *corsivo*, soprattutto quando s'introduce un *nuovo termine*, e poi si può usare il nuovo termine anche senza corsivo;
- Usate `equation` per le equazioni, in modo che vengano tutte numerate; le equazioni a fine frase terminano con il punto, ma magari evitiamo altro tipo di punteggiatura alla fine di equazioni.
- usare molto gli ambienti `definition`, `theorem`, `example`, `remark`...
- negli elenchi, il ";" alla fine degli *item* e il punto alla fine dell'ultimo. Ma se sono frasi lunghe va bene anche il punto dappertutto.

Definizione 2.1. (Tensore). Un *tensore* è ciò che ruota come un tensore.

$$e^z = \sum_{n=0}^{+\infty} \frac{z^n}{n!}. \tag{2.1}$$

```c
#include <stdio.h>
int main()
{
  // printf() displays the string inside quotation
  printf("Hello, World!");
  return 0;
}
```