Valutazione dei risultati della classificazione

Gianluca Amato

Corso di Laurea Specialistica in Economia Informatica
Università "G. D'Annunzio" di Chieti-Pescara
anno accademico 2004-2005

Accuratezza dei classificatori

Tasso di errore vero

- Una volta ottenuto un classificatore, bisogna stimarne l'accuratezza.
  - ci serve una misura della prestazione del classificatore
  - per i problemi di classificazione, una misura naturale è il tasso di errore.
- Supponiamo che
  - I sia l'insieme di tutte le istanze possibili
  - Pr è una distribuzione di probabilità su I
  - c(I) è la classe "vera" di I
  - h(I) è la classe "predetta" di I
- Si definisce allora il tasso di errore vero (true error rate) come
  - $error_I(h) = Pr\ \{ i \in I \mid h(I) \neq c(I) \}$

Tasso di errore su un campione

- Nella maggior parte dei casi I è infinito e il true error rate non è calcolabile
- Si considera allora un campione finito $S \subseteq I$ e si calcola tasso di errore sul campione S

$$error_S(h)=\frac{1}{n}\sum_{x\in S}\delta(f(x),h(x))$$

  dove $\delta(f(x),h(x))$ è 0 se $f(x)=h(x)$, 1 altrimenti.
- Si stima $error_I(h)$ sulla base di $error_S(h)$
  - sotto opportune ipotesi, $error_S(h)$ è "vicino" a $error_I(h)$
- Invece del tasso di errore, si può definire il tasso di successo
  - tutto procedere esattamente come per il tasso di errore

Intervalli di confidenza

- Il metodo standard per stimare il vero tasso di errore è avere un insieme S di istanze di testing, estratte in maniera indipendente dalle istanze di addestramento.
- Possiamo allora affermare che $error_I(h)$ cade in un certo intervallo, con una determinata confidenza.
- Esempio 1:
  - $error_S(h)$=25%, |S|=1000
  - con confidenza dell'80%, $error_I(h) \in [0.233, 0.268]$
- Esempio 2:
  - $error_S(h)$=25% ma |S|=100
  - con confidenza dell'80%, $error_I(h) \in [0.203, 0.313]$

Processi di Bernoulli e binomiale

- Consideriamo la variabile casuale E: I → {0,1} con
  - E(x)=δ(f(x),h(x))
- La variabile E ha una distribuzione di Bernoulli con parametro $p=error_I(h)$
  - infatti, con prob. p vale 1 e con prob. 1-p vale 0
  - media = p
  - varianza = p(1-p)
- n * $error_S(h)$ è semplicemente la somma di n variabili di Bernoulli indipendenti, dunque ha una distribuzione binomiale
  - media = np
  - varianza = np(1-p)

Lo stimatore $error_S(h)$

- Quindi, su $error_S(h)$ possiamo dire che ha
  - media: $error_I(h)$
  - varianza: 1/n * $error_I(h)$ (1-$error_I(h)$)
- Quindi $error_S(h)$ è uno stimatore non distorto di $error_I(h)$
- Vogliamo calcolare un intervallo di confidenza per $error_S(h)$
  - ovvero, dato un livello di confidenza *c*, vogliamo trovare un intervallo $[e_1,e_2]$ tale che

$$Pr\{e_1 \leqslant |error_S(h) - error_I(h)| \leqslant e_2\} = c$$

Intervalli di confidenza per variabili normali (1)

- Il calcolo di un intervallo di confidenza per una distribuzione binomiale è faticoso
  - tuttavia, per valori di n elevati, una distribuzione binomiale di media p e varianza v è approssimabile con una distribuzione gaussiana con stessa media e varianza.
- Se X è una distribuzione gaussiana di media 0 e varianza 1, esistono tabelle (e algoritmi) che danno il valore di
  - Pr { X >= z }
- Siccome X è simmetrica rispetto all'asse y, allora
  - Pr { -z <= X <= z } = 1 – 2*P{ X >= z}
- Possiamo allora calcolare gli intervalli di confidenza per X

Intervalli di confidenza per variabili normali (2)

- Consideriamo la segunte tabella

P{X>=z}	z
0,01%	3,72
0,50%	2,58
1,00%	2,33
5,00%	1,65
10,00%	1,28
20,00%	0,84
40,00%	0,25

- L'intervallo di confidenza per X con confidenza del 90% è [-1.65,1.65]
  - in quanto Pr {-1.65 <= X <= 1.65} = 0.9

Intervalli di confidenza per $error_I(h)$

- Se p=$error_I(h)$ e se supponiamo di approssimare la distribuzione binomiale con una gaussiana,

$$\frac{error_S(h)-p}{\sqrt{p(1-p)/n}}$$

  è una variabile gaussiana di media 0 e varianza 1
- Dato un valore di confidenza c, sappiamo trovare z tale che

$$Pr\left\{-z \leqslant \frac{error_S(h)-p}{\sqrt{p(1-p)/n}} \leqslant z\right\} = c$$

- Risolvendo su p otteniamo

$$p = \left(error_S(h) + \frac{z^2}{2n} \pm z\sqrt{\frac{f}{n} - \frac{f^2}{n} + \frac{z^2}{4n}}\right) / \left(1 + \frac{z^2}{n}\right)$$

Esempi di intervalli di confidenza

- Invece di un semplice esempio statico, questa è una tabella in OpenOffice Calc per determinare gli intervalli di confidenza

Dati		Intervallo risultante [p1,p2]	
errorS(h)	25,00%	p1	0,203
n	100	p2	0,313
c	0,8		
z	1,28		

- Attenzione che per valori di *n* molto piccoli, l'approssimazione gaussiana non è più indicata.
  - diciamo che per *n*>100 non c'è problema

Errore di sostituzione

- Il modo più immediato per scegliere S è usare l'insieme di addestramento.
  - si parla di errore di sostituzione (resubstitution error)
- La stima è troppo ottimistica!!
  - occorre utilizzare due insiemi distinti, uno per l'addestramento e l'altro per il test.
- E` importante che l'insieme di test non sia utilizzato in alcun modo nella fase di costruzione del modello
  - ad esempio, nella fase di "reduced error pruning" non vada usato l'insieme di test, ma un terzo insieme di dati, indipendente sia da quello di addestramento che da quello di test
- Una volta compiuta la stima, si può rimettere l'insieme di test dentro quello di addestramento, e ricalcolare il modello.

Metodo holdout

- Holdout: metodo con cui si divide l'insieme di dati in una parte usata per l'addestramento e una per il testing.
  - tipicamente 1/3 è usato per il test e 2/3 per l'addestramento
- Problema: il campione potrebbe non essere rappresentativo
  - ad esempio, alcune classi poco numerose potrebbero mancare nell'insieme di addestramento
  - le versione avanzate di holdout usano la tecnica della stratificazione.
    - le proporzioni tra le varie classi nell'insieme di dati originario devono essere le stesse che si riscontrano negli insiemi di test e di addestramento.
  - il problema comunque rimane se il numero di istanze è esiguo.

Cross validation

- Si divide l'insieme di dati in k parti
  - ripeto, per tutti i valori di i da 1 a k:
    - addestro il sistema con tutti i dati tranne quelli della partizione i
    - uso la partizione i per calcolare il tasso di errore
  - calcolo il tasso di errore finale come la media dei k tassi di errore che ho ottenuto in questo modo
  - si parla di k-fold cross validation.
- Che valore scegliere per k?
  - da numerosi esperimenti si è visto che 10 è un buon valore
- Si può al solito ricorrere alla versione stratificata, che è la più utilizzata.

Leave one out cross validation (1)

- Se si sceglie k=N (numero totale di istanze), si ha la leave one out cross validation.
- Vantaggi:
  - fa il massimo uso dei dati a disposizione
  - non ci sono campionamenti casuali
- Svantaggi:
  - computazionalmente oneroso (tranne che per il metodo k-NN) in quanto addestrare il sistema ben N volte.
  - la stratificazione non è possibile
    - anzi, l'insieme di test è “garantito” non essere stratificato

Leave one out cross validation (2)

- Caso estremo:
  - insieme di istanze I infinito, completamente casuale e con due classi nella stessa proporzione
    - siccome le istanze sono completamente casuali, qualunque modello generato avrà tasso di errore vero del 50%
  - per un insieme di dati casuali, il migliore classificatore non può far altro che predirre, per una nuova istanza, la classe che ha probabilità maggiore nell'insieme di apprendimento
  - supponendo un campione che ha lo stesso numero di istanze per ogni classe:
    - ogni fold del LOO-CV, la classe opposta alla istanza di test è maggioritaria
    - la predizione sarà sempre scorretta, dando un errore stimato del 100%
  - dunque otteniamo una stima molto pessimistica

Bootstrap

- Il metodo di cross-validation o di holdout usano un campionamento senza rimpiazzo
  - una volta che una istanza è stata scelta per una determinata partizione, non può essere scelta di nuovo
- I metodi di bootstrap usano invece un campionamento con rimpiazzo
  - da un insieme di dati di N istanze, ne vengono scelte N con rimpiazzo
    - le istanze scelte fanno parte dell'insieme di addestramento
    - quelle che non sono mai state scelte fanno parte dell'insieme di testing.
  - si ripete il procedimento con campioni differenti
  - si tenta, insomma, di recuperare la relazione tra l'insieme di istanze di addestramento S e l'insieme di tutte le istanze I, sfruttando la relazione tra S e i suoi sotto-campioni.

Bootstrap (2)

- In particolare utilizziamo il metodo del 0.632 bootstrap.
  - ogni istanza ha probabilità 1 – 1/N di non essere scelta
  - pertanto la sua probabilità di far parte dell'insieme di test è

$$\left(1-\frac{1}{N}\right)^N \simeq e^{-N} \simeq 0.368$$

  dal limite notevole $\lim_{n\to\infty}\left(1+\frac{x}{n}\right)^n = e^x$
- L'insieme di addestramento conterrà quindi circa il 63.2% delle istanze.

Bootstrap (3)

- La stima del tasso di errore ottenuto dal bootstrap è pessimistica
  - l'insieme di addestramento contiene solo il 63.2% delle istanze (contro il 90% del 10-fold CV e il quasi 100 della LOO-CV)
- Spesso, la si bilancia con l'errore di sostituzione, ottenendo

$$err = 0.632 \cdot err_{\text{istanze di test}} + 0.368 \cdot err_{\text{istanze di addestramento}}$$

- Il procedimento si ripete varie volte e si mediano i risultati.

Bootstrap (4)

- Si è rivelato molto efficace per piccoli insiemi di dati ma ha dei problemi
- Caso limite:
  - considera un insieme causale di dati, per cui ogni modello ha tasso di errore vero del 50%
  - il migliore classificatore possibile otterrà un errore del 0% sull'insieme di addestramento e del 50% su un insieme di test
  - il metodo del bootstrap stima per questo classificatore un tasso di errore del 0.632 * 50% = 31.6%
    - eccessivamente ottimistico

Valutazione di previsioni probabilistiche

Accuratezza di predizioni probabilistiche

- Finora abbiamo usato come misura di accuratezza il tasso di errore (o il tasso di successo)
- La maggior parte dei classificatori possono essere modificati per produrre distribuzioni di probabilità
  - vorremmo tener conto di questa distribuzione per valutare l'errore
  - se la predizione è che al 90% la classe di una istanza x è true, mentre in realtà la classe di x è false, siamo di fronte a un errore più grave che se la probabilità prevista fosse stata solo del 60%.
- Decidere se tenere conto o no delle probabilità nella stima della bontà di un classificatore dipende dall'utilizzo che se ne fa
  - ad esempio, se la stima verrà vagliata da un esperto

Funzioni di perdita

- Siano C_1,... C_k le possibili classi, si chiama funzione di perdita una funzione a valori reali con due parametri:
  - una distribuzione di probabilità su C_1... C_k
  - un intero da 1 a k che indica la vera classe dell'istanza
- data una istanza $i \in I$
  - h(i) è la distribuzione di probabilità calcolata dal classificatore
  - c(i) è il vero valore della istanza I
- data un funzione di perdita δ, l'errore vero rispetto a δ è

$$error_I(h) = E[\delta(h(i), c(i))]$$

dove la media è calcolata sulla distribuzione di probabilità su I.

Errore vero ed errore campionario

- L'errore vero non è calcolabile, e si ricorre quindi all'errore campionario.
- Dato $S \subseteq I$, definiamo l'errore sul campione

$$error_S(h) = \sum_{i \in S} \delta(h(i), c(i))$$

- Il problema è al solito quello di approssimare $error_I(h)$ con $error_S(h)$

Funzione di perdita quadratica (1)

- Una delle funzioni di perdita più utilizzate è la funzione di perdita quadratica

$$\delta(p,c)=\sum_{j=1}^{k}(p_j-a_j)^2$$

dove $a_j=1$ se j=c, altrimenti $a_j=0$, ovvero

$$\delta(p,c)=\sum_{j\neq c}p_j^2+(1-p_c)^2$$

Funzione di perdita quadratica (2)

- Giustificazione
  - supponiamo che la variabile classe sia casuale, e indipendente dagli altri attributi
  - minimizzare la perdita quadratica vuol dire utilizzare un classificatore che produce come output p la vera distribuzione di probabilità p^* delle classi
- Infatti

$$error_I(h)=E[\sum_j(p_j-a_j)^2]=\sum_j(E[p_j^2]-2E[p_ja_j]+E[a_j^2])=$$
$$=\sum_j(p_j^2-2p_jp_j^*+p_j^*)=\sum_j((p_j-p_j^*)^2+p_j^*(1-p_j^*))$$

Funzione di perdita di informazione

- Nella teoria dei codici, se c è un simbolo che occorre con probabilità p_c, il numero di bit necessari alla sua rappresentazione è $\log_2 p_c$
  - supponendo di usare il miglior codice possibile
- Definisco sulla base di questa osservazione la funzione di perdita:

$$\delta(p,c)=-\log_2 p_c=-\sum_{j=1}^{k}a_j\log_2 p_j$$

- Nel caso di attributo classe casuale e indipendente

$$error_I(h)=E[-\sum_j a_j\log_2 p_j]=-\sum_j p_j^*\log_2 p_j$$

e studiando questa funzione si vede che è minimizzata per $p=p^*$

Quale funzione di perdita scegliere?

- Al solito, dipende dalla situazione
- La perdita quadratica
  - prende in considerazione tutte le classi
  - non può mai essere superiore a 2
- La perdita di informazione
  - considera solo la probabilità della classe corretta
  - può essere infinita quando la probabilità di una classe è 0
  - è legata al problema della rappresentazione delle informazioni e al "*minimum description length principle*"
- MDL principle: il metodo dei classificazione migliore è quello che minimizza il numero di bit necessari per codificare il modello e i dati

Il costo degli errori

Costo degli errori

- In realtà, gli errori che si commettono non sempre hanno lo stesso costo.
- Alcune situazioni in cui alcuni errori sono più gravi di altri
  - diagnosi precoce di un guasto
  - predizione macchie di petrolio
- In queste situazioni, la maggior parte delle volte non c'è nessuna situazione "degenere"
  - il classificatore che sceglie sempre la classe false (nessun guasto, nessuna macchia) avrebbe un accuratezza altissima
- Noi vogliamo pesare di più l'errore di predirre false quando la classe vera è true piuttosto che il contrario.

Matrice di confusione

- Gli errori commessi dal classificatore possono essere visualizzati in una matrice di confusione

classi vere\predette	*Classe 1*	*Classe 2*	*Classe 3*	*...*	*Classe k*
Classe 1	23	3	4		2
Classe 2	..	17	..		..
Classe 3	..	..	15		..
....					
Classe k	..	..	..		93

- L'errore campionario si ottiene dalla somma dei valori di tutte le celle, esclusa la diagonale
- In generale, si possono voler in maniere diversa le varie celle

Apprendimento sensibile ai costi

- La maggior parte degli schemi di apprendimento non sono sensibili ai costi
  - si pensi agli algoritmi basati sugli alberi di decisione
- Metodi standard per ottenere metodi sensibili ai costi in una situazione a due classi
  - Ricampionare i dati in accordo ai costi
    - se il costo di non accorgersi di una macchia di petrolio è più alto di quello di dare un falso allarme, ricampiono i miei dati in modo che le istanze di addestramento con vere macchie di petrolio siano in numero maggiore di quelle senza macchie
  - Pesare le istanze in accordo ai costi
    - abbiamo visto che gli alberi di decisioni pesano le istanze quando devono gestire attributi mancanti
    - la stessa cosa si può fare per assegnare pesi diverse alle istanze, a seconda della classe della stessa.

Stimare i costi

- In realtà, raramente si ha una idea precisa di quanto siano i costi dei vari errori
  - dipende anche da fattori difficili da ponderare, come il costo di acquisire dati di addestramento
- Di solito, le decisioni vengono prese confrontando tra loro differenti scenari
- Esempio:
  - si vuole valutare un spedizione in massa di una offerta promozionale a un milione di abitazioni
    - stando ai dati precedenti, la proporzione che risponde positivamente all'offerta è lo 0.1% (1000 abitazioni)
    - un tool di data mining identifica un sottoinsieme di 100.000 abitazioni con tasso di risposta dello 0.4% (400 abitazioni)
  - che fare?

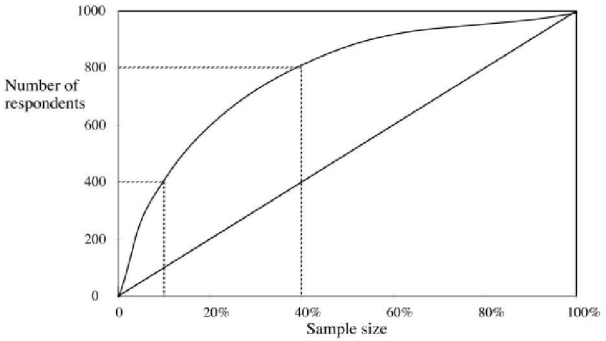
Lift chart (1)

- Si può avere una comoda visualizzazione di vari scenari con il lift chart.
  - nella terminologia del marketing, l'aumento del tasso di risposta dell'esempio precedente è chiamato "lift factor"
- Le istanze di test sono ordinate in ordine decrescente di probabilità di essere un vero positivo.

Posizione	*Probabilità*	*Classe effettiva*
1	0,95	yes
2	0,93	yes
3	0,93	no
4	0,88	yes
...	...	...

Lift chart (2)

- Per ogni *i* tra 1 ed *N* (numero istanze)
  - si considera il campione costituito dalle prime *i* istanze
  - si traccia un punto su un piano cartesiano dove
    - l'asse x è la frazione i/N in percentuale (la dimensione del campione)
    - l'asse y è il numero di veri positivi in quel campione

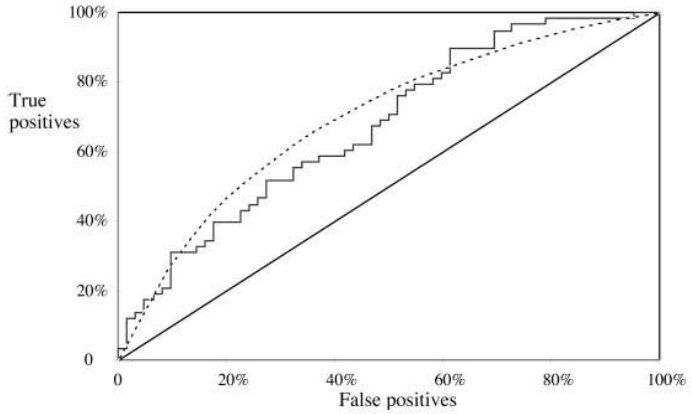


Curve ROC (1)

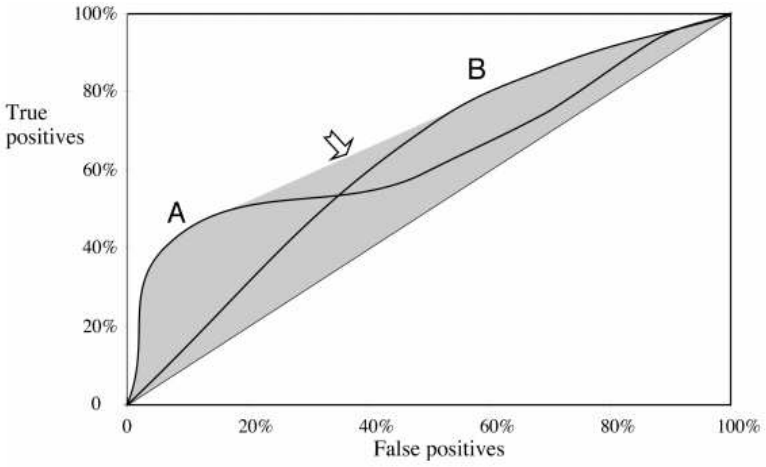
- Le curve ROC sono simili ai lift chart
  - ROC sta per "receiver operating characteristics"
  - usato nel riconoscimento di segnali per mostrare il trade off tra numero di segnai riconosciuti e numero di falsi allarmi
- Differenze con i lift chart
  - l'asse x è la percentuale di falsi positivi (e non la dimensione del campione)
  - l'asse y è la percentuale di veri positivi (e non il valore assoluto)
- Se il numero di veri positivi è molto basso (ad esempio nel caso di marketing diretto), non c'è molta differenza tra curve ROC e lift chart.

Curve ROC (2)

- Esempio di curva ROC per lo stesso insieme di dati di prima



Curve ROC per due classificatori



L'inviluppo convesso

- Dati due classificatori, possiamo raggiungere qualunque punto nell'inviluppo convesso delle due curve ROC
- Siano
  - t_1, f_1 i valori di true positive e false positive scelti per il 1° classificatore
  - t_2, f_2 i valori di true positive e false positive scelti per il 1° classificatore
  - fissato un $q \in [0,1]$ se il primo schema è usato per predirre, in maniera casuale, 100% * q dei casi e il secondo per i restanti, il nuovo schema combinato avrà
    - true positive = $q * t_1 + (1-q) * t_2$
    - false positive = $q * f_1 + (1-q) * f_2$
  - al variare di q, percorre tutta la retta da (f_1,t_1) a (f_2,t_2)

Valutazione di previsione numeriche

Valutazione di predizioni numeriche

- Stesse strategie adottare per la classificazione
  - insiemi di test, cross validation, etc..
- Cambia la funzione di errore
  - se p(i) è il valore predetto per la istanza I e a(i) il valore effettivo
  - sia S ⊆ I un insieme di istanze di cardinalità *n*
  - la funzione di errore più usata è l'errore quadratico medio

$$eqm = \frac{\sum_{i \in S} (p(i) - a(i))^2}{n}$$

  - semplice da manipolare matematicamente
  - sotto opportune ipotesi, corrisponde a massimizzare la verosimiglianza dei dati

Altre funzioni di errore

- La radice dell'errore quadratico medio è espressa nella stesse unità di misura dei dati

$$reqm = \sqrt{\frac{\sum_{i \in S} (p(i) - a(i))^2}{n}}$$

- l'errore assoluto medio è meno sensibile agli outlier

$$ram = \frac{\sum_{i \in S} |p(i) - a(i)|}{n}$$

- talvolta in queste formule si sostituisce l'errore assoluto *p(i)-a(i)* con l'errore relativo alla previsione *(p(i)-a(i))/p(i)*

Misure relative alla media

- Talvolta vogliamo tener conto di quanto il nostro classificatore è migliore di quello banale che semplicemente predice sempre la media dei dati
- Definiamo allora l'errore quadratico relativo come

$$rqr = \frac{\sum_{i \in S} (p(i) - a(i))^2}{\sum_{i \in S} (a(i) - \bar{a})^2}$$

dove $\bar{a} = \frac{1}{n} \sum_{i \in S} a(i)$ e l'errore assoluto relativo

$$rar = \frac{\sum_{i \in S} |p(i) - a(i)|}{\sum_{i \in S} |a(i) - \bar{a}|}$$

Il coefficiente di correlazione (1)

- Misura la correlazione statistica tra il valore reale e quello predetto

$$corr = \frac{S_{PA}}{\sqrt{S_P S_A}}$$

dove S_{PA}, S_P ed S_A sono stimatori rispettivamente della
  - covarianza tra P ed A
  - varianza di P
  - varianza di A
- In formule

$$S_{PA} = \frac{\sum_{i \in S} (p(i) - \bar{p})(a(i) - \bar{a})}{n-1} \quad S_P = \frac{\sum_{i \in S} (p(i) - \bar{p})^2}{n-1} \quad S_A = \frac{\sum_{i \in S} (a(i) - \bar{a})^2}{n-1}$$

Il coefficiente di correlazione (2)

- Assume valore tra -1 e +1
  - valori elevati sono indice di buone prestazione
- E` indipendente dalla scala
  - se moltiplico tutti i p(i) o tutti gli a(i) per un certo fattore k, il risultato non cambia

Quale misura di errore usare?

- Dipende dalla situazione
- In molti casi, comunque, il metodo di classificazione migliore rimane migliore indipendentemente dal tipo di misura di errore utilizzata

	A	B	C	D
Root mean-squared error	67.8	91.7	63.3	57.4
Mean absolute error	41.3	38.5	33.4	29.2
Root relative squared error	42.2%	57.2%	39.4%	35.8%
Relative absolute error	43.1%	40.1%	34.8%	30.4%
Correlation coefficient	0.88	0.88	0.89	0.91

Combinazione di classificatori

Boosting e Bagging

- Sono due metodi standard per combinare dei classificatori C_1, ... C_T per produrre un classificatore C* più accurato.
- Analogia con i medici
  - Supponiamo di voler diagnosticare una malattia. Possiamo rivolgerci a vari medici invece che ad uno solo.
  - Bagging: prendo le risposte di tutti i medici e considero come diagnosi valida quella prodotta in maggioranza.
  - Boosting: peso la diagnosi di ogni medico in base agli errori che egli ha commesso in precedenza

Bagging

- Sia S un insieme di *s* istanze.
- Per ogni t in {1, ... , T}
  - determino l'insieme S_t ottenuto campionando *s* valori con rimpiazzo
  - ottengo un classificatore C_t
- Per una nuova istanza, provo tutti i classificatori e scelgo la risposta che occorre con più frequenza.

Boosting

- Sia S un insieme di *s* istanze.
- Si applica ad algoritmi di classificazione che possono supportare istanze pesate
  - Ogni istanza ha un peso *w* (inizialmente uguale per tutte)
- Per ogni t in {1, ... , T}
  - addestro il classificatore C_t tenendo conto dei pesi attuali.
  - modifico i pesi in modo tale che, durante l'apprendimento di C_{t+1}, contino di più le istanze classificate incorrettamente da C_t.
- Per una nuova istanza, uso tutti i classificatori e peso i risultati in base alla loro accuratezza sull'insieme di addestramento.

Bibliografia

Bibliografia

- Tom M. Mitchell, *Machine Learning,* McGraw-Hill
  - capitolo 5
- Ian H. Witten, Eibe Frank, Data Mining: *Practical Machine Learning Tools and Techniques with Java Implementations,* Morgan Kaufmann
  - capitolo 5