Fondamenti di Machine Learning
Metodi Matematici per il Machine Learning

Slide a cura di Silvia Bonettini e Luca Zanni
Giorgia Franchini giorgia.franchini@unimore.it

University of Modena and Reggio Emilia, Italy
Department of Physics, Informatics and Mathematics

6 Novembre 2022

Struttura lezione

- Formalizzazione di un problema di classificazione binaria
- Logistic loss function
- Backtracking non monotono
- Problema test giocattolo
- Problema test di libreria.

Un problema di classificazione binaria

Dato il training set

$$\{(a_i, b_i), \quad a_i \in \mathbb{R}^m, \ b_i \in \{-1, 1\}, \quad i = 1, \ldots, N\}$$

determinare

$$f : \mathbb{R}^m \to \mathbb{R}$$

per classificare nuovi eventi $a \in \mathbb{R}^m$ mediante la valutazione di $sign(f(a))$.

Fissiamo lo spazio delle ipotesi, cioè la classe di funzioni con la quale descrivere la relazione tra i dati a_i e b_i del training set

$$\mathcal{H} = \left\{ f_{w,c} : \mathbb{R}^m \to \mathbb{R}, \quad | \quad f_{w,c}(x) = w^T x + c, \quad w, x \in \mathbb{R}^m, \ c \in \mathbb{R} \right\}$$

Obiettivo di un algoritmo di Machine Learning: determinare (w^*, c^*) affinchè la funzione $f_{w^*,c^*}(x)$ descriva adeguatamente la relazione tra i dati del training set, evitando fenomeni di overfitting

Come determinare (w^*, c^*)?

Una strategia per determinare la funzione di decisione $f_{w^*,c^*}(x)$ consiste nell'individuare (w^*, c^*) come soluzione del seguente problema di ottimizzazione:

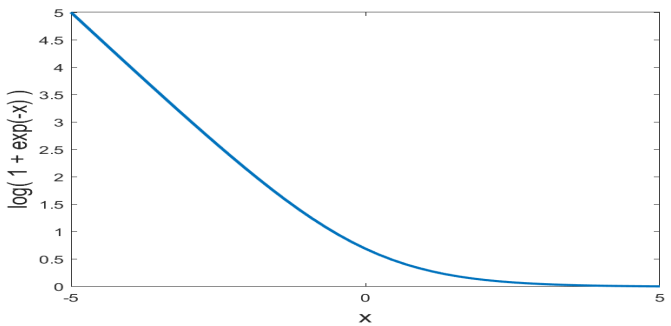
$$\min_{w,c} F(w,c) = \min_{w,c} \frac{1}{N}\sum_{i=1}^{N} V(b_i, f_{w,c}(a_i)) + \frac{\lambda}{2}\|f_{w,c}\|^2$$

per opportune scelte della loss function $V(b, f_{w,c}(a))$ e della misura della complessità della funzione di decisione $\|f_{w,c}\|$.

In questa esercitazione:

- $V(b, f_{w,c}(a)) = \log(1 + \exp(-bf(a))) = \log(1 + \exp(-b(w^T a + c))$

logistic loss function



- $\|f_{w,c}\| = \left\| \begin{bmatrix} w \\ c \end{bmatrix} \right\| = \sqrt{\begin{bmatrix} w \\ c \end{bmatrix}^T \begin{bmatrix} w \\ c \end{bmatrix}} = \sqrt{\sum_{i=1}^{m} w_i^2 + c^2}$

  (regolarizzazione in norma ℓ_2)

Come risolvere il problema $\min_{w,c} F(w, c)$?

Per risolvere il problema di ottimizzazione utilizzeremo il metodo del gradiente nella versione proposta da Barzilai-Borwein:

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}, \qquad k = 0, 1, 2, \ldots$$

- direzione di discesa:

$$d^{(k)} = -\rho_k \nabla F(x^{(k)}), \qquad \rho_k = \frac{{s^{(k)}}^T s^{(k)}}{{s^{(k)}}^T y^{(k)}}$$

$$s^{(k)} = x^{(k)} - x^{(k-1)}, \qquad y^{(k)} = \nabla F(x^{(k)}) - \nabla F(x^{(k-1)}).$$

  Si ricordi che le incognite del problema sono i pesi (w, c), cioè $x = \begin{bmatrix} w \\ c \end{bmatrix}$

- strategia di backtracking in versione monotona (regola di Armijo) o anche in versione non monotona

- test di arresto: $\left\| \nabla F(x^{(k)}) \right\| \leq tol$

Backtracking monotono o non monotono

La condizione che deve essere soddisfatta dalla procedura di backtracking era

$$F(x^{(k)} + \alpha d^{(k)}) \leq F(x^{(k)}) + \sigma\alpha\nabla F(x^{(k)})^T d^{(k)}$$

Questa condizione crea una successione monotona decrescente $\{F(x^{(k)})\}_{k=0,1,\ldots}$

Quando la procedura di backtracking è combinata con la regola di Barzilai-Borwein, per consentire che la lunghezza di passo ρ_k proposta dalla regola venga accettata e non eccessivamente ridotta, si utilizza una versione non monotona della procedura di backtracking nella quale la condizione da soddisfare è:

$$F(x^{(k)} + \alpha d^{(k)}) \leq F_{ref} + \sigma\alpha\nabla F(x^{(k)})^T d^{(k)}$$

dove

$$F_{ref} = \max\{F(x^{k-j}), \quad j = 0, 1, \ldots, M\}, \qquad M \in \mathbb{N}$$

Quando $M = 0$ si ritrova la classica regola di Armijo; quando $M > 0$ la successione $\{F(x^{(k)})\}_{k=0,1,\ldots}$ può risultare non monotona decrescente.

Cosa ci serve per implementare il metodo del gradiente?

Il metodo del gradiente richiede

- il calcolo della funzione obiettivo

$$F(x) = F(w, c) = \frac{1}{N} \sum_{i=1}^{N} \log(1 + \exp(-b_i(w^T a_i + c)) + \frac{\lambda}{2} \left\| \begin{bmatrix} w \\ c \end{bmatrix} \right\|^2$$

- il calcolo del gradiente della funzione obiettivo. Tenendo presente che

$$\frac{\partial F(w,c)}{\partial w_j} = \frac{1}{N} \left[\sum_{i=1}^{N} \frac{\exp(-b_i(w^T a_i + c))}{1+\exp(-b_i(w^T a_i + c))} (-b_i (a_i)_j) \right] + \lambda w_j, \quad j = 1, \ldots, m$$

$$\frac{\partial F(w,c)}{\partial c} = \frac{1}{N} \left[\sum_{i=1}^{N} \frac{\exp(-b_i(w^T a_i + c))}{1+\exp(-b_i(w^T a_i + c))} (-b_i) \right] + \lambda c$$

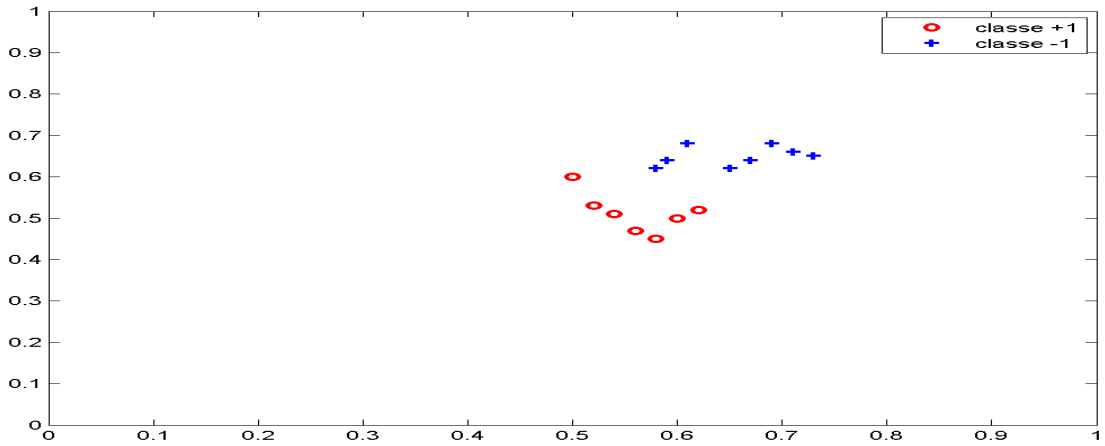
segue

$$\nabla F(w, c) = \frac{1}{N} \sum_{i=1}^{N} \begin{bmatrix} \frac{-b_i \exp(-b_i(w^T a_i + c))}{1+\exp(-b_i(w^T a_i + c))} a_i \\ \\ \frac{-b_i \exp(-b_i(w^T a_i + c))}{1+\exp(-b_i(w^T a_i + c))} \end{bmatrix} + \lambda \begin{bmatrix} w \\ c \end{bmatrix}$$

Problemi test

Consideriamo due problemi test:

- Problema test giocattolo: 15 punti nel piano, 7 di classe 1 e 8 di classe -1



- Problema test di libreria: dataset *w8a*
  Disponibile, assieme a molti altri datasets, all'indirizzo
  https://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/
  Training set di $N = 49749$ esempi di dim. $m = 300$; test set di 14951 esempi.
  Gli esempi sono costituiti da componenti binarie. Ogni esempio è memorizzato in formato "compresso", cioè per ogni esempio sono memorizzate solo le componenti non nulle
  (label index1:feature1 index2:feature2 ...)

Problema test giocattolo

L'obiettivo è individuare la funzione di decisione del tipo

$$f_{w,c}(x) = w^T x + c, \quad w \in \mathbb{R}^2, \quad c \in \mathbb{R}$$

determinando la soluzione (w^*, c^*) del problema

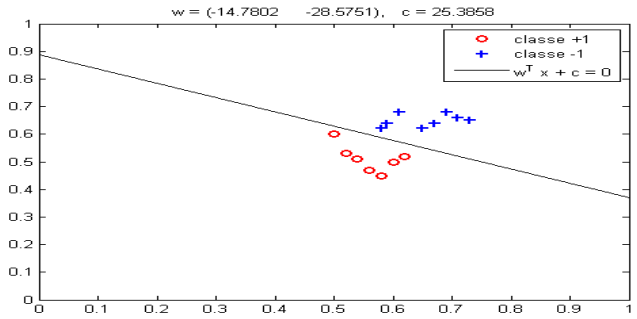
$$\min_{w,c} F(w, c) = \min_{w,c} \frac{1}{N} \sum_{i=1}^{N} \log(1 + \exp(-b_i(w^T a_i + c)) + \frac{\lambda}{2} \left\| \begin{bmatrix} w \\ c \end{bmatrix} \right\|^2$$

Nell'esercitazione vengono forniti i programmi Matlab per risolvere il problema:

- function "logistic_l2_c.m" per calcolo funzione obiettivo
- function "grad_logistic_l2_c.m" per calcolo gradiente della funzione obiettivo
- function "BB1_solver.m" per metodo del gradiente di Barzilai-Borwein
- function "backtracking.m" per procedura di backtracking monotona o non monotona
- script "main_test_dim2_c.m" per chiamata di BB1_solver e costruzione grafici

Osservazioni sui risultati del problema test giocattolo

grafico funzione di decisione per $\lambda = 10^{-4}$

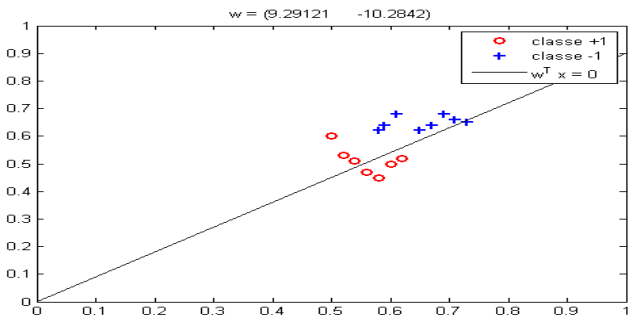


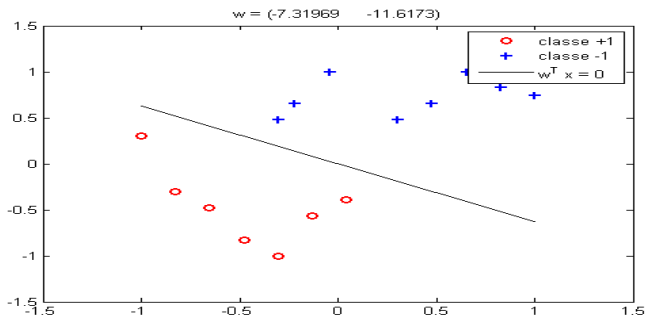
- BB1 monotono: 90 it. BB1 non monontono: 27 it. steepest: 13823 it.

- se si lavora con la classe di funzioni seguente

$$\mathcal{H} = \left\{ f_w : \mathbb{R}^m \to \mathbb{R}, \quad | \quad f_w(x) = w^T x, \quad w, x \in \mathbb{R}^m \right\}$$

conviene scalare i dati in modo che ogni componente sia in $[-1,\ 1]$

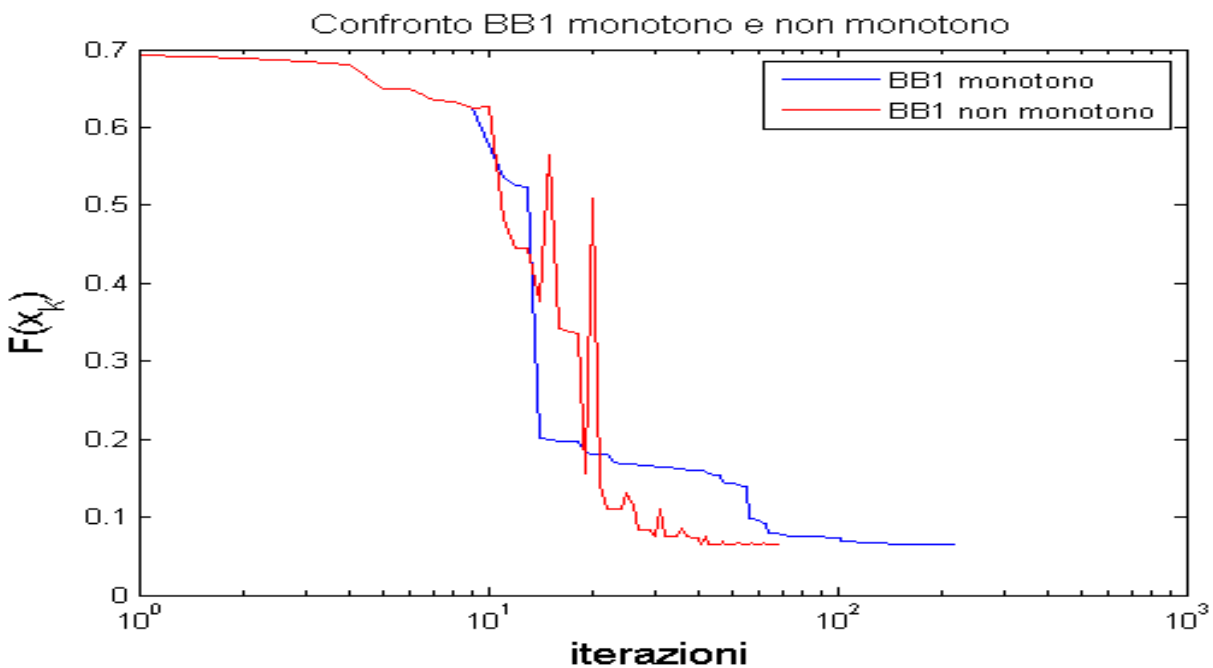




Osservazioni sul backtraking monotono e non monotono

Quando si utilizza un metodo del gradiente nella versione Barzilai-Borwein l'uso della procedura di backtracking non monotono può essere utile per accelerare la convergenza.

Il seguente grafico mostra la discesa della funzione obiettivo sul problema test giocattolo ($\lambda = 10^{-5}$) nei due casi ($M = 5$ nella versione non monotona)



Problema test di libreria

Il dataset riguarda un problema di classificazione di pagine web con una rappresentazione degli esempi basata su 300 componenti binarie. Per l'addestramento della metodologia di learning si utilizza la versione del dataset di dimensione 49749 e per testare la metodologia si usa un test set di 14951 esempi.
Anche in questo caso l'obiettivo è individuare la funzione di decisione del tipo

$$f_{w,c}(x) = w^T x + c, \quad w \in \mathbb{R}^m, \quad c \in \mathbb{R}$$

determinando la soluzione (w^*, c^*) del problema

$$\min_{w,c} F(w,c) = \min_{w,c} \frac{1}{N} \sum_{i=1}^{N} \log(1 + \exp(-b_i(w^T a_i + c)) + \frac{\lambda}{2} \left\| \begin{bmatrix} w \\ c \end{bmatrix} \right\|^2$$

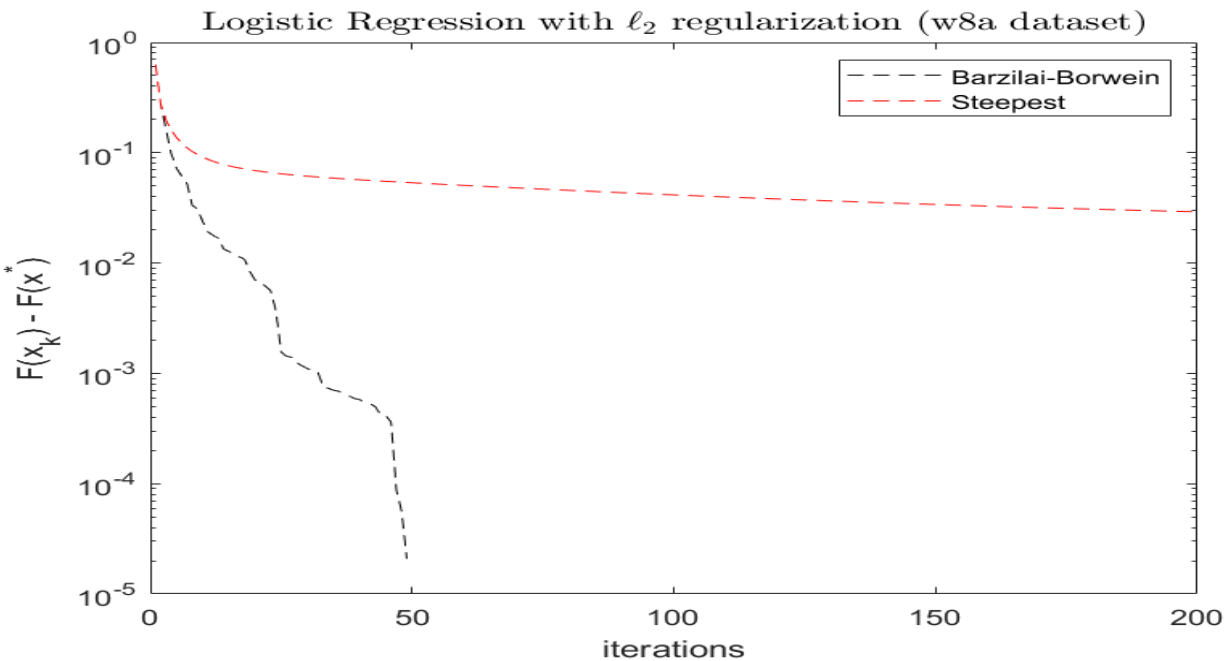
Nell'esercitazione vengono forniti, oltre alle funzioni già usate nel problema test giocattolo, i seguenti programmi Matlab:

- script "main_dataset_w8a_c.m" per chiamata metodi BB1 e steepest descent e per la costruzione del grafico sulla discesa della funzione obiettivo;
- function "ACCURACY_c.m" per il calcolo delle misure di accuratezza relative ad una funzione di predizione applicata ad un test set
- function "misure_accuracy_c.m" per la lettura dei parametri che definiscono la funzione di predizione e la chiamata a "ACCURACY_c.m";

Il metodo BB1 soddisfa il test di arresto ($\|\nabla F(x^{(k)})\| \leq 10^{-4}$) in 49 iterazioni (6.95 sec).

Il metodo steepest descent non soddisfa il test di arresto in 200 iterazioni (28.28 sec).

Il grafico che segue mostra la discesa della funzione obiettivo durante il procedimento di ottimizzazione.



Misure di accuratezza per la classificazione binaria

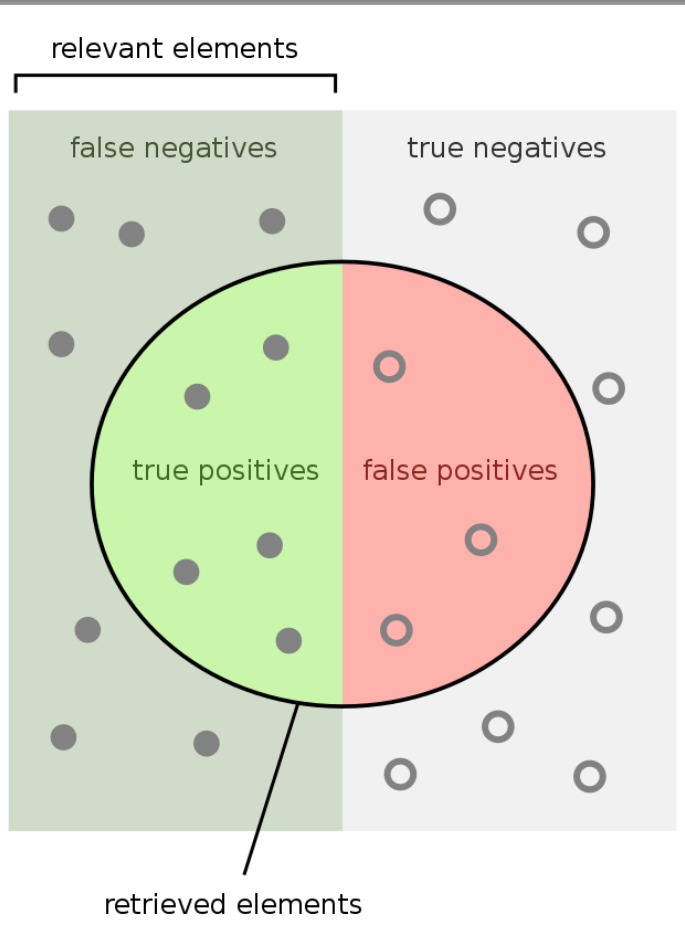
Usando la notazione

- TP = True Positive
- TN = True Negative
- FP = False Positive
- FN = False Negative
- N = numero esempi testati

si possono definire le seguenti misure di accuratezza:

- Accuracy $= \dfrac{TP + TN}{N}$
- Precision $= \dfrac{TP}{TP + FP}$
- Recall $= \dfrac{TP}{TP + FN}$
- F1 $= 2 \cdot \dfrac{precision \cdot recall}{precision + recall}$

Misure di accuratezza per la classificazione binaria



How many retrieved items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{retrieved elements}}$$

How many relevant items are retrieved?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

Matrice di confusione

		Predicted	
		Negative (**N**) **-**	Positive (**P**) **+**
Actual	Negative **-**	True Negatives (T**N**)	False Positives (F**P**) **Type I error**
	Positive **+**	False Negatives (F**N**) **Type II error**	True Positives (T**P**)