

Figura 16: Simulazione Monte Carlo con l'algoritmo di Metropolis per il modello di Ising 2D con dimensioni 50x50 e condizioni al contorno periodiche. $J = 1$ e $h = 0$. Ogni simulazione consta di $100 \cdot 10^6$ di passi.

Parte VIII

Algoritmi di ottimizzazione

Cominciamo con il problema di trovare il minimo (o il massimo) di una funzione $f(x) : \mathbb{R} \rightarrow \mathbb{R}$. Qui parleremo sempre di minimi. Basta moltiplicare f per -1 per passare da un problema di minimo a uno di massimo.

31 Metodo della sezione aurea

Consideriamo $f(x) : \mathbb{R} \rightarrow \mathbb{R}$ continua e tre punti lungo x : $a < b < c$. Supponiamo valga: $f(b) < f(a)$ e $f(b) < f(c)$ allora c'è sicuramente (almeno) un minimo di f in $[a, c]$. Possiamo trovarlo con il seguente algoritmo iterativo (vedi Fig. 17) :

1. prendiamo $a < b < c$ tali che $f(b) < f(a)$ e $f(b) < f(c)$
2. se $(c - b) > (b - a)$ prendiamo $x_0 \in [b, c]$:
 - (a) se $f(x_0) > f(b)$ allora ho un minimo in $[a, x_0]$, pongo $c = x_0$ e torno a (1)
 - (b) altrimenti ho un minimo in $[b, c]$, pongo $a = b$, $b = x_0$ e torno a (1)
3. se $(c - b) > (b - a)$ prendiamo $x_0 \in [a, b]$:
 - (a) se $f(x_0) > f(b)$ allora ho un minimo in $[x_0, c]$, pongo $a = x_0$ e torno a (1)
 - (b) altrimenti ho un minimo in $[a, b]$, pongo $b = x_0$, $c = b$ torno a (1)

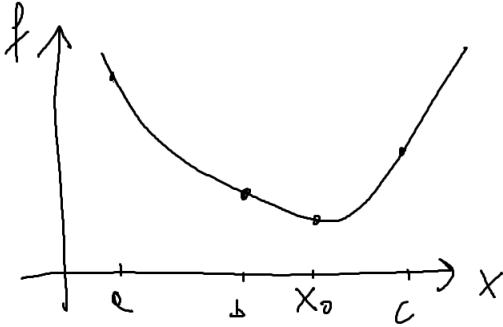


Figura 17: Minimizzazione di una funzione tramite il metodo della sezione aurea

Vogliamo trovare un metodo per scegliere x_0 in maniera tale che i due possibili nuovi intervalli di ricerca abbiano lunghezza uguale. Supponiamo di essere nel caso $(c - b) > ((b - a))$ e poniamo:

$$w = \frac{b - a}{c - a} \quad (315)$$

$$z = \frac{x_0 - b}{c - a} \quad (316)$$

Se $f(x_0) > b$ allora il nuovo intervallo di ricerca è $[a, x_0]$ con lunghezza $(w + z)(c - a)$, altrimenti l'intervallo è $[b, c]$ con lunghezza $(1 - w)(c - a)$. I due intervalli sono uguali se:

$$z = 1 - 2w \quad (317)$$

Quindi dato w possiamo ricavare z e quindi x_0 . Dobbiamo però ancora determinare il primo w . Supponiamo $f(x_0) < f(b)$. Imponiamo di avere sempre le stesse proporzioni tra i punti a, b, x_0 e b, x_0, c , ossia:

$$\frac{b - a}{c - a} = \frac{x_0 - b}{b - x_0} \quad (318)$$

$$w = \frac{z}{1 - w} \quad (319)$$

Sostituendo Eq. 317 e troviamo:

$$w = \frac{1 - 2w}{1 - w} \quad (320)$$

$$w^2 - 3w + 1 = 0 \quad (321)$$

Prendiamo la sola soluzione soluzione entro $[0, 1]$:

$$w = \frac{3 - \sqrt{5}}{2} \simeq 0.38197 \quad (322)$$

Vediamo ora in pratica come funziona. Cominciamo con scegliere a, c e poi poniamo $b = a + w(c - a)$. Abbiamo un minimo se $f(b) < f(a)$ e $f(b) < f(c)$ in tal caso, scegliamo $x_0 = b + z(c - a)$. Ora abbiamo due casi:

Se $f(x_0) < b$ ricominciamo con b, x_0, c e si verifica facilmente che $(x_0 - b)/(c - b) = w$.

Se $f(x_0) > b$ ricominciamo con a, x_0, b , in tal caso la situazione è *ribaltata* infatti si verifica facilmente che $(x_0 - b)/(x_0 - a) = w$ Quindi sceglieremo il nuovo punto di prova $x'_0 = b - z(x_0 - a)$. Notiamo che w è proprio la sezione aurea.

32 Approssimazione con una parabola

In molti casi $f(x)$ è approssimabile con una parabola in prossimità del minimo con una parabola. Valutiamo f in $a < b < c$. e approssimiamo:

$$f(x) \approx \alpha + \beta x + \gamma x^2 \quad (323)$$

I coefficienti α, β, γ sono univocamente determinati dalla richiesta che la funzione approssimante passi per $(a, f(a)), (b, f(b)), (c, f(c))$. Poi il minimo approssimato di f è nel punto:

$$x_0 = -\frac{\beta}{2\gamma} \quad (324)$$

svolgendo i calcoli si trova:

$$x_0 = b - \frac{1}{2} \frac{(b-a)^2(f(b)-f(c)) - (b-c)^2(f(b)-f(a))}{(b-a)(f(b)-f(c)) - (b-c)(f(b)-f(a))} \quad (325)$$

In diversi casi è disponibile anche la derivata di f , $f'(x) = df(x)/dx$. In tal caso mi bastano due punti $a < b$ per trovare la parabola. Mi basta conoscere f' in uno dei due punti che supponiamo sia b . Allora, i parametri β e γ sono dati da:

$$\beta = f'(b) - 2\gamma b \quad (326)$$

$$\gamma = \frac{(f(a) - f(b)) - f'(b)(a - b)}{(a^2 - b^2) - 2b(a - b)} \quad (327)$$

33 Metodi basati sul gradiente

Consideriamo il problema di trovare il minimo (o i minimi) di una generica funzione $F(x_1, \dots, x_N)$ rispetto ai parametri x_1, \dots, x_N .

Supponiamo sia possibile calcolare il gradiente:

$$\nabla F(x_1, \dots, x_N) = \begin{pmatrix} \frac{\partial}{\partial x_1} F(x_1, \dots, x_N) \\ \frac{\partial}{\partial x_2} F(x_1, \dots, x_N) \\ \vdots \\ \frac{\partial}{\partial x_N} F(x_1, \dots, x_N) \end{pmatrix} \quad (328)$$

Allora possiamo trovare il minimo spostandoci lungo la direzione opposta al gradiente. Secondo il seguente algoritmo detto *Steepest descend*.

1. Dati i valori iniziali x_1^i, \dots, x_N^i , calcolo $\nabla F(x_1^i, \dots, x_N^i)$
2. Calcolo il parametro λ tale che $F\left(x_1^i - \lambda \frac{\partial}{\partial x_1} F(x_1^i, \dots, x_N^i), \dots, x_N^i - \lambda \frac{\partial}{\partial x_N} F(x_1^i, \dots, x_N^i)\right)$ sia minimo
3. Pongo per il valore di λ trovato $x_1^{i+1} = x_1^i - \lambda \frac{\partial}{\partial x_1} F(x_1^i, \dots, x_N^i), \dots, x_N^{i+1} = x_N^i - \lambda \frac{\partial}{\partial x_N} F(x_1^i, \dots, x_N^i)$ e ritorno al punto 2

Talvolte è conveniente usare un valore di λ prefissato senza ricavarlo ad ogni iterazione.

Una metoda affine basato sui gradienti è quello del *Gradiente Coniugato* che permette di scegliere la direzione lineare di minimizzazione in maniera più efficiente. Indichiamo con \mathbf{u} il vettore direzione di ricerca. Il metodo steepest-descent pone:

$$\mathbf{u}^i = -\frac{\nabla f(\mathbf{x}^i)}{|\nabla f(\mathbf{x}^i)|} \quad (329)$$

Il metodo del gradiente coniugato genera delle direzioni di ricerca in maniera tale che la minimizzazione lungo la direzione successiva non rovini le minimizzazioni lungo le direzioni precedenti. Supponiamo di approssimare f in serie di Taylor attorno all'origine del sistema di riferimento:

$$f(\mathbf{x}) \approx f(0) + \sum_{i=1,N} \frac{\partial f(0)}{\partial x_i} x_i + \frac{1}{2} \sum_{i,j=1,N} \frac{\partial^2 f(0)}{\partial x_i \partial x_j} x_i x_j \quad (330)$$

che scriviamo in forma vettoriale come:

$$f(\mathbf{x}) = c - \mathbf{b} \cdot \mathbf{x} + \frac{1}{2} \mathbf{x} \cdot \hat{A} \mathbf{x} \quad (331)$$

con:

$$\mathbf{b} = -\nabla f(0) \quad (332)$$

$$A_{ij} = \frac{\partial^2 f(0)}{\partial x_i \partial x_j} \quad (333)$$

Supponiamo di minimizzare lungo \mathbf{u} fino al punto di minimo \mathbf{x}_u in quel punto il gradiente di f sarà ortogonale a \mathbf{u} . Vogliamo ora trovare la direzione di minimizzazione \mathbf{v} tale che spostandomi da \mathbf{x}_u lungo \mathbf{v} il gradiente di f non acquisti componenti lungo \mathbf{u} . Tale gradiente lo scrivo come:

$$\nabla f(\mathbf{x}_u + \epsilon\mathbf{v}) = -\mathbf{b} + \hat{A}(\mathbf{x}_u + \epsilon\mathbf{v}) \quad (334)$$

quindi richiediamo che

$$\mathbf{u} \cdot (-\mathbf{b} + \hat{A}(\mathbf{x}_u + \epsilon\mathbf{v})) = 0 \quad (335)$$

Siccome la relazione

$$\mathbf{u} \cdot (-\mathbf{b} + \hat{A}(\mathbf{x}_u)) = 0 \quad (336)$$

è soddisfatta per dalla condizione di minimo di \mathbf{x}_u allora la direzione \mathbf{v} deve soddisfare la relazione:

$$\mathbf{u} \cdot \hat{A}\mathbf{v} = 0 \quad (337)$$

Il metodo del gradiente coniugato genera una serie di direzioni \mathbf{h}^i tutte coniugate tra di loro. Poniamo per la prima iterazione:

$$\mathbf{g}^0 = \mathbf{h}^0 \quad (338)$$

e per le iterazioni successive usiamo la regola:

$$\mathbf{g}^{i+1} = \mathbf{g}^i - \lambda_i \hat{A}\mathbf{h}^i \quad (339)$$

$$\mathbf{h}^{i+1} = \mathbf{g}^{i+1} + \gamma_i \hat{A}\mathbf{h}^i \quad (340)$$

con:

$$\lambda_i = \frac{\mathbf{g}^i \cdot \mathbf{g}^i}{\mathbf{h}^i \hat{A}\mathbf{h}^i} = \frac{\mathbf{g}^i \cdot \mathbf{h}^i}{\mathbf{h}^i \hat{A}\mathbf{h}^i} \quad (341)$$

$$\gamma_i = \frac{\mathbf{g}^{i+1} \cdot \mathbf{g}^{i+1}}{\mathbf{g}^i \cdot \mathbf{g}^i} \quad (342)$$

Si può mostrare che questo assicura che per $j < i$

$$\mathbf{g}^i \cdot \mathbf{g}^j = 0 \quad (343)$$

$$\mathbf{h}^i \hat{A}\mathbf{h}^j = 0 \quad (344)$$

$$\mathbf{g}^i \cdot \mathbf{h}^j = 0 \quad (345)$$

Ora ci poniamo il problema che la matrice \hat{A} non è conosciuta. Possiamo usare il seguente teorema: Se per un i ho che $\mathbf{g}^i = -\nabla f(\mathbf{x}^i)$ e minimizzando lungo \mathbf{h}^i trovo il minimo di f nel punto \mathbf{x}^{i+1} allora risulta $\mathbf{g}^{i+1} = -\nabla f(\mathbf{x}^{i+1})$.

Dimostrazione: Abbiamo:

$$\mathbf{g}^i = \mathbf{b} - \hat{A}\mathbf{x}^i \quad (346)$$

$$\mathbf{g}^{i+1} = \mathbf{b} - \hat{A}(\mathbf{x}^i + \lambda \mathbf{h}^i) = \mathbf{g}^i - \lambda \hat{A}\mathbf{h}^i \quad (347)$$

con λ determinato dalla relazione di minimo $\mathbf{g}^{i+1} \cdot \mathbf{h}^i = 0$ ossia $0 = \mathbf{g}^i \cdot \mathbf{h}^i - \lambda \mathbf{h}^i \cdot \hat{A}\mathbf{h}^i$ da cui si ritrova:

$$\lambda = \frac{\mathbf{g}^i \cdot \mathbf{h}^i}{\mathbf{h}^i \cdot \hat{A}\mathbf{h}^i} \quad (348)$$

Come si voleva dimostrare.

La scelta di γ in Eq. 341 prende il nome di Fletcher-Reeves. Spesso viene preferita la variante di Polak-Ribiere

$$\gamma_i = \frac{(\mathbf{g}^{i+1} - \mathbf{g}^i) \cdot \mathbf{g}^{i+1}}{\mathbf{g}^i \cdot \mathbf{g}^i} \quad (349)$$

che risulta più stabile.

34 Dynamic relaxation

Introduciamo un sistema dinamico fittizio associando ad ogni parametro λ_i un' energia cinetica fittizia $\frac{1}{2} \left(\frac{d\lambda_i}{dt} \right)^2$ e una forza frenante (frizione) $-f(t) \frac{d\lambda_i}{dt}$ quindi andremo a propagare le equazioni del moto fittizio:

$$\begin{cases} \frac{d^2\lambda_1(t)}{dt^2} = -f(t) \frac{d\lambda_1}{dt} - \frac{\partial F(\lambda_1, \dots, \lambda_N)}{\partial \lambda_1} \\ \dots \\ \frac{d^2\lambda_N(t)}{dt^2} = -f(t) \frac{d\lambda_N}{dt} - \frac{\partial F(\lambda_1, \dots, \lambda_N)}{\partial \lambda_N} \end{cases} \quad (350)$$

abbassando il termine di frizione fino al suo annullarsi in modo da trovare un punto di equilibrio in cui i parametri restano costanti nel tempo. Tale punto è un punto di minimo perché avremo $\frac{\partial F(\lambda_1^i, \dots, \lambda_N^i)}{\partial \lambda_i} = 0$ per tutti gli i .

35 Simulated annealing

L'idea è quella di considerare il sistema dinamico fittizio dal punto di vista termodinamico. Consideriamo una distribuzione canonica per i parametri $\lambda_1, \dots, \lambda_N$:

$$p(\lambda_1, \dots, \lambda_N) = \frac{1}{Z} e^{-\frac{F(\lambda_1, \dots, \lambda_N)}{T}} \quad (351)$$

dove T è una temperatura fittizia e Z è una costante di normalizzazione che non conosciamo. L'idea è quella di usare un algoritmo come quello di Metropolis per campionare lo spazio dei parametri ad una certa temperatura che verrà poi abbassata in maniera da portare il sistema nel suo stato di energia (o meglio di F) più basso.

Parte IX

Machine Learning

36 Il problema della classificazione

Supponiamo di avere dei campioni (*samples*) che appartengono a classi diverse. Ogni classe è contraddistinta da un nome o etichetta (*label*). I campioni sono definite come un insieme di dati numerici (*features*). Un esempio famoso è la classificazione dei fiori Iris. Le features sono le lunghezze e larghezze di petalo e sepalo e le classi sono le specie di Iris (Virginica, Versicolor, Setosa). Il database contiene dati di 150 Iris 50 per specie. In generale per ogni sample i le features sono date dal vettore reale \mathbf{x}^i , il vettore ha dimensione N con N numero delle features, e la classe è data dal numero intero y^i . Ci poniamo il problema di creare un algoritmo capace di predire l'appartenenza di un campione ad una certa classe dopo essere stato *addestrato* su un certo numero di campioni detto insieme di addestramento. Per capire la qualità di un classificatore vado a vedere come si comporta su un insieme di campioni, detto di test, diverso da quello di addestramento.

37 Il Perceptron

Nel 1957 è stato proposto il primo classificatore che ha la struttura di un neurone artificiale. Per ogni feature $j = 1, N$ viene introdotto un peso w_j . Il perceptron può essere visto come una funzione di \mathbf{x} dipendente dai parametri \mathbf{w} , detti pesi, e da un ulteriore parametro reale θ che restituisce 1 se il campione appartiene ad una certa classe e -1 in caso contrario. Tale funzione viene scritta come:

$$\tilde{\sigma}(z) = \begin{cases} 1 & z > \theta \\ -1 & z \leq \theta \end{cases} \quad (352)$$

con

$$z = \sum_{i=1,N} w_i x_i \quad (353)$$

Di solito si preferisce introdurre una feature aggiuntiva sempre paria 1: $x_0 = 1$ in tale maniera il parametro θ viene assorbito dal peso aggiuntivo w_0 . L'algoritmo diventa

$$\sigma(z) = \begin{cases} 1 & z > 0 \\ -1 & z \leq 0 \end{cases} \quad (354)$$

con

$$z = \sum_{i=0,N} w_i x_i \quad (355)$$

dove σ è ora la funzione gradino. Rosenblatt introdusse un algoritmo (*learning rule*) per trovare i pesi \mathbf{w} da un insieme di campioni di addestramento \mathbf{x}^i .

1. Parto con \mathbf{w} vicino a 0
2. ciclo sui campioni i , calcolo il risultato attuale $\hat{y}^i = \sigma(\mathbf{w} \cdot \mathbf{x}^i)$
3. aggiorno i pesi secondo la regola $\mathbf{w} = \mathbf{w} + \eta(y^i - \hat{y}^i)\mathbf{x}^i$ con η disolito $\in [0, 1]$

Dopo l'iniziale entusiasmo ci si accorse che il Perceptron funziona solo per problemi che sono linearmente separabili ossia se esiste una iperpiano di dimensione $N - 1$ che divide i campioni delle due classi. Infatti nel caso $N = 2$ abbiamo che:

$$z = w_0 + w_1 x_1 + w_2 x_2 \quad (356)$$

è la condizione $z = 0$ è definita dalla retta:

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2} \quad (357)$$

Per problemi non separabili l'algoritmo di Rosenblatt non converge e diventa instabile.

38 L'Adaline

L' *Adaptive linear neuron* introdotto nel 1960 permette la convergenza anche nel caso di problemi non completamente separabili inoltre presenta una struttura di neurone artificiale ripresa da modelli più recenti ed avanzati. Infatti viene sia introdotta una activation function $\phi(\mathbf{w} \cdot \mathbf{x})$ sia introdotta una funzione di errore $J(\mathbf{w})$ la minimizzazione di tale funzione permette di trovare i pesi. Poi il risultato del classificatore è dato, al solito, dalla funzione soglia $\sigma(\phi(\mathbf{w} \cdot \mathbf{x}))$

Nell'adaline la funzione di attivazione è semplicemente l'identità: $\phi(z) = z$ mentre la funzione di errore è definita come:

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1, N_{samples}} (y^i - \phi(\mathbf{w} \cdot \mathbf{x}^i))^2 \quad (358)$$

Per la minimizzazione devo calcolare il gradiente ∇J :

$$\frac{\partial J}{\partial w_j} = \sum_{i=1, N_{samples}} (y^i - \mathbf{w} \cdot \mathbf{x}^i)(-x_j^i) \quad (359)$$

Spesso come algoritmo di minimizzazione viene utilizzato lo steepest descent (semplice) ossia utilizzando la regola:

$$\mathbf{w} = \mathbf{w} - \eta \nabla J \quad (360)$$

con η chiamato *learning rate*.

Siccome spesso capita di avere a che fare con insiemi di addestramento molto numerosi viene sovente scelto l'algoritmo detto *stochastic gradient descent* in cui si sceglie in maniera casuale, a turno, un solo sample i e si aggiornano i pesi secondo la regola:

$$\mathbf{w} = \mathbf{w} + \eta(y^i - \mathbf{w} \cdot \mathbf{x}^i)\mathbf{x}^i \quad (361)$$

39 One versus all

Sia il Perceptron che l'Adaline forniscono l'informazione se un sample appartiene o meno ad una certa classe. Molte volte però ho a che fare con più di due classi. In tal caso per l'assegnazione si usa la strategia OVA (one versus all). Prima si addestrano tanti classificatori quante classi ci sono. Poi si assegna l'appartenenza ad una classe al classificatore che dà la funzione di attivazione $\phi(\mathbf{w} \cdot \mathbf{x})$ più elevata.

40 Standardizzazione

Succede che le features possono essere distribuite su scale di grandezza differenti. In tal caso è più efficiente rinormalizzarle in maniera che abbiano tutte lo stesso centro (che poniamo a 0) e la stessa deviazione standard che poniamo a 1. Se la feature j ha media u_j e deviazione standard σ_j , usiamo la trasformazione:

$$x'_j = \frac{x_j - u_j}{\sigma_j} \quad (362)$$

41 Regressione Logistica

Talvolta oltre l'assegnazione ad una classe vogliamo anche che il nostro classificatore associa una probabilità a tale assegnazione. Questo viene usato ad esempio nelle previsioni metereologiche per la classe *oggi piove*.

Cominciamo col mappare la probabilità $p \in [0, 1]$ nell'intervallo $[-\infty, +\infty]$:

$$f(p) = \log\left(\frac{p}{1-p}\right) \quad (363)$$

Tale funzione prende il nome di funzione logistica *logit*.

La *logistic regression* è analoga alla Adalina ma con funzione di attivazione ϕ e funzione di errore J differenti. Richiediamo che $\phi(z)$ dia una probabilità:

$$\text{logit}(\phi(\mathbf{w} \cdot \mathbf{x})) = \mathbf{w} \cdot \mathbf{x} \quad (364)$$

Poniamo $z = \mathbf{w} \cdot \mathbf{x}$ e risolvendo troviamo:

$$\phi(z) = \frac{1}{1 + e^{-z}} \quad (365)$$

Tale funzione prende il nome di *sigmoidea*.

Dobbiamo ora fornire la funzione errore. Per prima cosa ridefiniamo le y^i in maniera che $\in [0, 1]$ e usiamo una funzione a scalino:

$$\sigma'(p) = \begin{cases} 1 & p > 0.5 \\ 0 & p \leq 0.5 \end{cases} \quad (366)$$

Cominciamo con una prima funzione performance:

$$L(\mathbf{w}) = \prod_{i=1, N_{samples}} (\phi(z^i))^{y^i} (1 - \phi(z^i))^{1-y^i} \quad (367)$$

dove la funzione nella produttoria per $y^i = 1$ è pari a $\phi(z^i)$ altrimenti è pari a $(1 - \phi(z^i))$ tale funzione va massimizzata. Il problema è equivalente a massimizzare $\log(L)$. Scegliamo quindi la funzione errore da minimizzare:

$$J(\mathbf{w}) = -\log L(\mathbf{w}) \quad (368)$$

$$= \sum_{i=1, N_{samples}} (-y^i \log(\phi(z^i)) - (1 - y^i) \log(1 - \phi(z^i))) \quad (369)$$

La minimizzazione richiede il calcolo di ∇J :

$$\frac{\partial J}{\partial w_j} = \sum_{i=1, N_{samples}} \left(-\frac{y^i}{\phi(z^i)} \frac{\partial \phi(z^i)}{\partial w_j} - \frac{1-y^i}{1-\phi(z^i)} \left(-\frac{\partial \phi(z^i)}{\partial w_j} \right) \right) \quad (370)$$

dove:

$$\frac{\partial \phi(z^i)}{\partial w_j} = \frac{\partial}{\partial w_j} \left(\frac{1}{1 + e^{-z^i}} \right) = -\frac{1}{(1 + e^{-z^i})^2} (-e^{-z^i}) x_j^i \quad (371)$$

$$= \frac{1}{(1 + e^{-z^i})} \frac{e^{-z^i} + 1 - 1}{(1 + e^{-z^i})} x_j^i \quad (372)$$

$$= \frac{1}{(1 + e^{-z^i})} \left(1 - \frac{1}{1 + e^{-z^i}} \right) x_j^i \quad (373)$$

$$= \phi(z^i)(1 - \phi(z^i))x_j^i \quad (374)$$

Inserendo troviamo:

$$\frac{\partial J}{\partial w_j} = \sum_{i=1, N_{samples}} (-y^i + \phi(z^i)) x_j^i \quad (375)$$

42 Regolarizzazione

Un inconveniente tipico della classificazione è il problema del *overfitting*. Questo è di solito accompagnato, se si usano features rinormalizzate, da una distribuzione irregolare dei pesi \mathbf{w} . Per alleviare tale problema si regolarizza la funzione errore, dato $C > 0$:

$$J(\mathbf{w}) = C J(\mathbf{w}) + \frac{1}{2} \mathbf{w} \cdot \mathbf{w} \quad (376)$$

il termine $\mathbf{w} \cdot \mathbf{w}$ è minimo per una distribuzione uniforme dei pesi. Facendo diventare C piccolo si induce uniformità in \mathbf{w} .

43 Support Vector Machine

Supponiamo, inizialmente, di lavorare con un problema linearmente separabile. Useremo la notazione vettoriale per rappresentare samples \mathbf{x}^i nello spazio delle features. Non includeremo però la dimensione 0-esima. Le Support Vector Machines forniscono una classificazione $-1, 1$. L'idea è di trovare un iper-piano nello spazio delle features che separi le soluzioni positive 1 da quelle negative -1 . Se il problema è separabile tale iperpiano esiste ed inoltre ci saranno (praticamente sicuramente) più piani con tale proprietà. Dobbiamo quindi scegliere l'iperpiano migliore. Vogliamo massimizzare la distanza tra l'iperpiano e i samples dell'insieme di addestramento più vicini ad esso. Tali samples vengono chiamati support vectors (vedi Fig. 18)

Chiamiamo \mathbf{x}^{pos} il support vector della classe $+1$ e \mathbf{x}^{neg} il support vector della classe -1 . Consideriamo la seguente funzione:

$$\phi(\mathbf{x}) = w_0 + \mathbf{w} \cdot \mathbf{x} \quad (377)$$

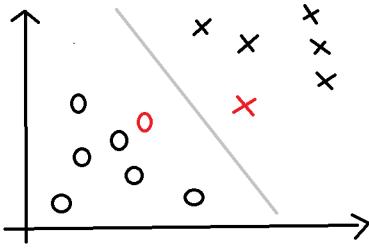


Figura 18: Samples di classe 1 (x), di classe -1 (o) e support vectors (rosso)

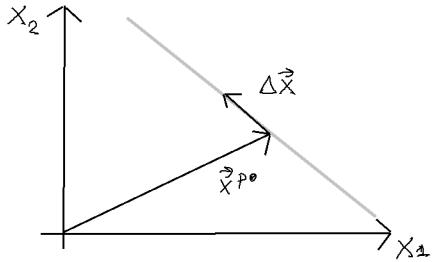


Figura 19: Margine passante per \mathbf{x}^{pos}

e scegiamo i pesi in maniera tale che $\phi \geq 1$ nella parte dello spazio a cui appartengo i samples positivi e $\phi \leq -1$ nella parte dello spazio a cui appartengo i samples negativi. Chiamiamo margini gli iperpiani $\phi(\mathbf{x}) = 1$ e $\phi(\mathbf{x}) = -1$ e scegiamo i pesi in maniera che i support vectors stiano sui margini:

$$w_0 + \mathbf{w} \cdot \mathbf{x}^{pos} = 1 \quad (378)$$

$$w_0 + \mathbf{w} \cdot \mathbf{x}^{neg} = -1 \quad (379)$$

$$(380)$$

Dimostriamo ora che \mathbf{w} è perpendicolare ai margini. Infatti se prendiamo $\mathbf{x}^{pos} + \Delta\mathbf{x}$ appartenente al margine di o \mathbf{x}^{pos} (vedi Fig. 19) risulta:

$$w_0 + \mathbf{w} \cdot (\mathbf{x}^{pos} + \Delta\mathbf{x}) = 0 \quad (381)$$

$$(w_0 + \mathbf{w} \cdot \mathbf{x}^{pos}) + \mathbf{w} \cdot \Delta\mathbf{x} = 0 \quad (382)$$

$$\mathbf{w} \cdot \Delta\mathbf{x} = 0 \quad (383)$$

Il nostro obiettivo è massimizzare la distanza d tra i due margini. Essa può essere facilmente calcolata (vedi Fig. 20):

$$d = (\mathbf{x}^{pos} - \mathbf{x}^{neg}) \cdot \left(\frac{\mathbf{w}}{|\mathbf{w}|} \right) \quad (384)$$

$$= (\mathbf{x}^{pos} + w_0 - (\mathbf{x}^{neg} - w_0)) \cdot \left(\frac{\mathbf{w}}{|\mathbf{w}|} \right) \quad (385)$$

$$= \frac{2}{|\mathbf{w}|} \quad (386)$$

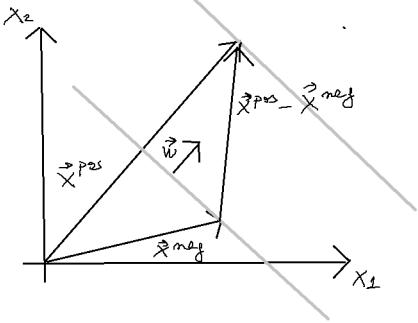


Figura 20: Distanza tra i margini

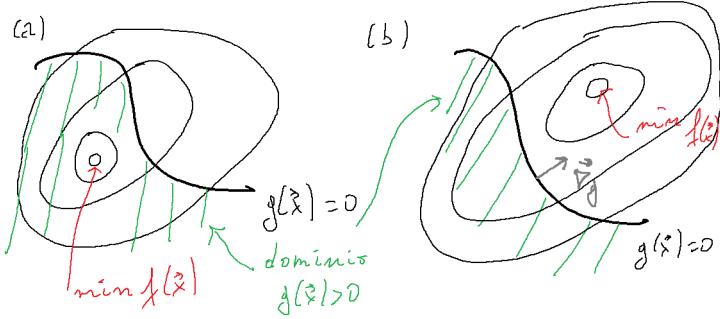


Figura 21: Minimizzazione di una funzione soggetta a condizione di diseguaglianza

Si preferisce però minimizzare la funzione costo o errore:

$$J(\mathbf{w}, w_0) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} \quad (387)$$

J va minimizzata con la condizione che i samples siano classificati correttamente ossia con l'insieme di condizioni:

$$y^i(w_0 + \mathbf{w} \cdot \mathbf{x}^i) \geq 1 \quad (388)$$

con $i = 1, N_{samples}$. Notiamo che sovente non si riescono a separare linearmente in maniera completa le due classi di samples. Per estendere la SVM a questo caso introduciamo un insieme di valori soglia $\xi^i \geq 0$ per $i = 1, N_{samples}$ e usiamo condizioni *rilassate*:

$$\begin{cases} w_0 + \mathbf{w} \cdot \mathbf{x}^i \geq 1 - \xi^i & \text{per } y^i = 1 \\ w_0 + \mathbf{w} \cdot \mathbf{x}^i \leq -1 + \xi^i & \text{per } y^i = -1 \end{cases} \quad (389)$$

Poi minimizziamo la seguente funzione costo:

$$J(\mathbf{w}, w_0, \xi^1, \dots, \xi^{N_{samples}}) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \left(\sum_{i=1, N_{samples}} \xi^i \right) \quad (390)$$

con C parametro fissato. Tale funzione va minimizzata sotto le condizioni di Eq. 389.

44 Moltiplicatori di Lagrange per condizioni di diseguaglianza

Supponiamo di cercare il minimo di $f(\mathbf{x})$ con la condizione $g(\mathbf{x}) \geq 0$. Abbiamo due casi.

(I) Nel primo il minimo di f sta nella parte di spazio dove $g > 0$ (vedi Fig. 21 (a)). In tal caso il minimo è lo stesso con o senza condizioni.

(II) Altrimenti il minimo deve trovarsi sul luogo dei punti (iper-superficie) $g(\mathbf{x}) = 0$ (vedi Fig. 21 (b)). Infatti se per assurdo il minimo si trovasse in un punto \mathbf{x}' con $g(\mathbf{x}') > 0$ allora potrei muovermi lungo la direzione $\nabla f(\mathbf{x}')$ ed il valore di f diminuirebbe. Inoltre abbiamo che sulla iper-superficie $g(\mathbf{x}) = 0$ $\nabla g(\mathbf{x})$ è localmente ortogonale ad essa. Nel punto di minimo anche $\nabla f(\mathbf{x})$ è ortogonale a tale iper-superficie. Inoltre $\nabla g(\mathbf{x}) \parallel \nabla f(\mathbf{x})$ quindi posso scrivere $\nabla f(\mathbf{x}) = \lambda \nabla g(\mathbf{x})$ con $\lambda > 0$.

i due casi possono essere raggruppati insieme tramite la ricerca del minimo di:

$$\min_{\mathbf{x}} \max \lambda f(\mathbf{x}) - \lambda g(\mathbf{x}) \quad \text{con } \lambda \geq 0 \quad (391)$$

Infatti nella parte di spazio $g(\mathbf{x}) > 0$ avremo che $\lambda = 0$ (caso (I)); altrimenti la soluzione \mathbf{x}' soddisferà $\nabla(f(\mathbf{x}') - \lambda g(\mathbf{x}')) = 0$ e $g(\mathbf{x}') = 0$. Pertanto \mathbf{x}' starà sulla ipersuperficie $g(\mathbf{x}) = 0$. Notiamo che nel caso (I), in cui il minimo non sta sul bordo, l'equazione $\nabla(f(\mathbf{x}') - \lambda g(\mathbf{x}')) = 0$ è soddisfatta solo per $\lambda < 0$.

45 Lagrangian duality

Rittrattiamo ora il problema della minimizzazione di una funzione $f(\mathbf{x})$ sottoposta al vincolo $g(\mathbf{x}) \geq 0$. Chiamiamo p^* la soluzione:

$$p^* = \min_{\mathbf{x}} f(\mathbf{x}) \quad \text{con } g(\mathbf{x}) \geq 0 \quad (392)$$

definiamo:

$$J(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & g(\mathbf{x}) \geq 0 \\ +\infty & g(\mathbf{x}) < 0 \end{cases} \quad (393)$$

allora:

$$p^* = \min_{\mathbf{x}} J(\mathbf{x}) \quad (394)$$

Definiamo:

$$H(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x}) \quad (395)$$

allora:

$$\max_{\lambda} H(\mathbf{x}, \lambda) = \begin{cases} f(\mathbf{x}) & g(\mathbf{x}) \geq 0 \\ +\infty & g(\mathbf{x}) < 0 \end{cases} \quad (396)$$

$$= J(\mathbf{x}) \quad (397)$$

Quindi:

$$p^* = \min_{\mathbf{x}} \max_{\lambda} H(\mathbf{x}, \lambda) \quad (398)$$

Proviamo ad invertire l'ordine di max e min:

$$H(\mathbf{x}, \lambda) \leq J(\mathbf{x}) \quad (399)$$

$$\min_{\mathbf{x}} H(\mathbf{x}, \lambda) \leq \min_{\mathbf{x}} J(\mathbf{x}) \quad (400)$$

$$\min_{\mathbf{x}} H(\mathbf{x}, \lambda) \leq p^* \quad (401)$$

$$\max_{\lambda} \min_{\mathbf{x}} H(\mathbf{x}, \lambda) \leq p^* \quad (402)$$

Allora $d^* = \max_{\lambda} \min_{\mathbf{x}} H(\mathbf{x}, \lambda)$ è un limite inferiore per p^* . Se $d^* = p^*$ si parla di *strong duality*, in tal caso il vincolo è strettamente soddisfatto $g(\mathbf{x}_{min}) > 0$ (caso (I) della sezione precedente). Altrimenti si parla di *weak duality*.

46 Minimizzazione della funzione costo di una SVM

Consideriamo la minimizzazione della funzione costo J di Eq. 387 soggetta ai vincoli di Eq. 388. Introduciamo i moltiplicatori di Lagrange λ_i per $i = 1, N_{samples}$ e la funzione:

$$J'(\mathbf{w}, w_0, \lambda_1, \dots, \lambda_{N_{samples}}) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} - \sum_{i=1, N_{samples}} \lambda_i y^i (w_0 + \mathbf{w} \cdot \mathbf{x}^i - 1) \quad (403)$$

Dobbiamo prima massimizzare J' rispetto alle λ_i con la condizione $\lambda_i \geq 0$ e poi dobbiamo minimizzare rispetto a \mathbf{w} e w_0 . Supponiamo strong duality e invertiamo le due operazioni. Consideriamo le condizione di minimo rispetto a \mathbf{w} e w_0 :

$$\frac{\partial J'}{\partial \mathbf{w}} = 0 \quad (404)$$

$$\mathbf{w} - \sum_{i=1, N_{samples}} \lambda_i y^i \mathbf{x}^i = 0 \quad (405)$$

e

$$\frac{\partial J'}{\partial w_0} = 0 \quad (406)$$

$$- \sum_{i=1, N_{samples}} \lambda_i y^i = 0 \quad (407)$$

Da cui ricaviamo:

$$\mathbf{w} = \sum_{i=1, N_{samples}} \lambda_i y^i \mathbf{x}^i \quad (408)$$

$$\sum_{i=1, N_{samples}} \lambda_i y^i w_0 = 0 \quad (409)$$

$$(410)$$

che sostituiamo in Eq. 403:

$$J''(\lambda_1, \dots, \lambda_{N_{samples}}) = \frac{1}{2} \sum_{i,j=1, N_{samples}} \lambda_i \lambda_j \mathbf{x}^i \mathbf{x}^j - \sum_{i,j=1, N_{samples}} \lambda_i \lambda_j \mathbf{x}^i \mathbf{x}^j + \sum_{i=1, N_{samples}} \lambda_i \quad (411)$$

$$= -\frac{1}{2} \sum_{i,j=1, N_{samples}} \lambda_i \lambda_j \mathbf{x}^i \mathbf{x}^j + \sum_{i=1, N_{samples}} \lambda_i \quad (412)$$

$$(413)$$

J'' va ora massimizzata rispetto alle λ_i con i vincoli $\lambda_i \geq 0$.

47 Estensione a problemi non linearmente separabili