Alma Mater Studiorum · Università di Bologna

**SCUOLA DI SCIENZE**
**Corso di Laurea Magistrale in Matematica**

# Metodi di ottimizzazione per l'analisi dati: il metodo di Kaczmarz per la regressione lineare

**Tesi di Laurea in Analisi Numerica**

**Relatore:**
**Prof.ssa**
**MARGHERITA PORCELLI**

**Presentata da:**
**EMANUELE**
**CIAMMARICONE**

**Anno Accademico 2022-2023**

# Indice

# Introduzione

Il presente elaborato tratta di metodi numerici per l'ottimizzazione nel campo dell'analisi dei dati. In questo campo, si affronta solitamente un problema di minimo di una funzione obiettivo reale di variabili reali. Un metodo classico per la risoluzione di questi tipi di problemi è il metodo del gradiente, un metodo iterativo del primo ordine, cioè che utilizza il gradiente della funzione obiettivo per generare una successione di approssimazioni della soluzione del problema di minimo.

A metà del XX secolo sono stati proposti i metodi di ottimizzazione stocastica e tra questi vi è il metodo del gradiente stocastico, una generalizzazione del metodo del gradiente in cui non è necessario avere l'informazione completa del gradiente per definire l'iterazione. Invece, si calcola una approssimazione del gradiente attraverso una funzione che dipende da una variabile aleatoria e che, generalmente, ha valore atteso uguale al gradiente della funzione obiettivo. Soprattutto nei casi in cui il costo del calcolo del gradiente risulta elevato, questo metodo risulta vantaggioso dal punto di vista computazionale e risulta particolarmente indicato nell'analisi dei dati [9, 10].

In questa tesi ci concentriamo su un modello particolare per l'analisi di dati, la regressione lineare. Il problema di ottimizzazione è dunque un problema ai minimi quadrati lineari in cui la matrice dei coefficienti rappresenta un insieme di dati (uno per riga) e le loro caratteristiche (una per colonna). In generale, la matrice è sovradeterminata, in quanto abbiamo insiemi di dati con tanti più campioni raccolti rispetto alle loro caratteristiche. Per questo problema è stato proposto in letteratura il metodo di Kaczmarz, che risulta una variante del metodo del gradiente stocastico per la regressione lineare.

Nel primo capitolo di questa tesi viene presentata una panoramica di alcune delle principali famiglie di problemi di ottimizzazione che nascono nell'analisi dei dati, mentre nei capitoli successivi analizziamo il metodo del gradiente, il metodo del gradiente stocastico e quello di Kaczmarz, con relativa analisi di convergenza dei metodi. Nell'ultimo capitolo sono raccolti gli esperimenti numerici ottenuti con i metodi studiati per ottimizzare un modello di regressione lineare per la simulazione delle vendite di farmaci di una azienda farmaceutica. I dati riguardanti queste vendite sono stati ottenuti grazie al tirocinio fatto presso l'azienda Analytics Network, un ente di consulenza informatica con sede a Casalecchio di Reno (BO). Durante questo tirocinio, è stato sviluppato un codice per costruire un modello di regressione lineare adatto al problema. A conclusione dell'elaborato, viene descritto il modello e vengono confrontati i risultati ottenuti con le varie tecniche studiate.

# Capitolo 1

# Panoramica sui problemi di ottimizzazione nell'analisi dei dati

## 1.1 Preliminari

In un problema di ottimizzazione regolare e non vincolato l'obiettivo è quello di minimizzare una funzione regolare di una o più variabili reali. La formulazione standard è la seguente

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

dove $f : \mathbb{R}^n \to \mathbb{R}$ è regolare, generalmente derivabile con continuità.

Come affrontato in [10], in questo elaborato assumeremo spesso l'ipotesi che le funzioni siano *convesse* per le loro proprietà sui minimi (si veda il Capitolo 2 di [10]).

**Definizione 1.1.** Un insieme $\Omega \subseteq \mathbb{R}^n$ si dice convesso se

$$\forall x, y \in \Omega \Rightarrow (1-\alpha)x + \alpha y \in \Omega, \ \forall \alpha \in [0,1]. \tag{1.2}$$

Cioè, per ogni coppia di punti di $\Omega$, $(x, y)$, il segmento che collega $x$ e $y$ è tutto contenuto in $\Omega$.

Gli insiemi convessi di cui tratteremo in questo elaborato saranno anche insiemi *chiusi*.

**Definizione 1.2.** Una funzione $f : \mathbb{R}^n \to \mathbb{R}$ si dice *convessa* se

$$f((1-\alpha)x - \alpha y) \leq (1-\alpha)f(x) + \alpha f(y), \ \forall x, y \in \mathbb{R}^n, \ \forall \alpha \in [1,0]. \tag{1.3}$$

Cioè, la linea che collega il punto $(x, f(x))$ con $(y, f(y))$ giace interamente "sopra" il grafico della funzione $f$.

**Teorema 1.1.** *Sia dato il problema di minimo (1.1), con $f$ convessa e $\Omega$ insieme convesso e chiuso. Allora:*

1. *Ogni soluzione locale è anche una soluzione globale.*

*2. L'insieme di soluzioni del problema (1.1) è un insieme convesso.*

*Dimostrazione.* Si veda il secondo capitolo del libro [10]. □

La principale proprietà di una funzione convessa segue dal teorema di Taylor. Supposto di avere una funzione $f$ derivabile, allora:

$$f(x+\alpha(y-x)) = f(x) + \alpha\nabla f(x)^T(y-x) + o(\alpha) \leq (1-\alpha)f(x) + \alpha f(y)$$

poiché $f$ è convessa. Semplifico $f(x)$ ed isolo $f(y)$

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + o(1)$$

Per $\alpha \downarrow 0$, $o(1)$ svanisce, perciò

$$f(y) \geq f(x) + \nabla f(x)^T(y-x),\ \forall x, y \in dom(f). \tag{1.4}$$

Da questa proprietà si dimostra il seguente teorema.

**Teorema 1.2.** *Sia $f$ una funzione convessa derivabile con derivata continua. Allora se $\nabla f(x^*) = 0$, si ha che $x^*$ è un minimo globale.*

**Definizione 1.3.** Sia $f : \mathbb{R}^n \to \mathbb{R}$ una funzione convessa e derivabile con derivata continua. Essa si dice *fortemente convessa con modulo di convessità $\gamma$* se esiste una valore $\gamma > 0$ tale che

$$f((1-\alpha)x + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y) - \frac{\gamma}{2}\alpha(1-\alpha)\|x-y\|_2^2, \tag{1.5}$$

per ogni $x$ e $y$ nel dominio di $f$

Una condizione equivalente alla forte convessità per funzioni differenziabili è la seguente:

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\gamma}{2}\|y-x\|^2. \tag{1.6}$$

Per mostrare questa osservazione si procede in maniera simile a quanto visto per ottenere (1.4). Per le funzioni fortemente convesse vale la seguente condizione sufficiente sui minimi.

**Teorema 1.3.** *Sia $f$ funzione fortemente convessa e differenziabile con derivata continua. Allora se $\nabla f(x^*) = 0$, si ha che $x^*$ è l'unico punto di minimo globale di $f$.*

Per quanto visto, quindi, lo studio dei punti di minimo globale per funzioni convesse o fortemente convesse risulta molto più semplice del caso generale non convesso.

## 1.2 Modelli nell'analisi dei dati[1]

Nel campo dell'analisi dati, un problema tipico richiede di determinare un modello che possa mettere in relazione un *insieme di dati* (o *data set*) raccolti riguardanti un certo evento o esperimento. Vengono spesso aggiunti vincoli strutturali che si pensa siano adatti alla descrizione del problema e che il modello deve rispettare.

In generale, un data set può essere formalizzato come un insieme di coppie del tipo

$$\mathcal{D} := \{(a_j, y_j), j = 1, \dots, m\}$$

in cui gli $a_j$ sono vettori o matrici e sono dette *features* o *caratteristiche dei dati*, mentre gli $y_j$ possono essere scalari o vettori e sono detti *osservazioni* o *etichette* in base al tipo di problema.

A partire da questo insieme, si vuole "imparare" una relazione tra features e osservazioni, cioé si vuole trovare una funzione $\phi$ tale che $\phi(a_j) \approx y_j$, in una misura di errore da definire. Questa ricerca viene detta fase di *training* o *learning*.

Spesso calcolare questa $\phi$ vuol dire determinare i parametri che la definiscono, che indichiamo con $x$ e solitamente sono un vettore o una matrice. Si cercano quindi i parametri "migliori" minimizzando una funzione obiettivo della forma

$$\mathcal{L}_{\mathcal{D}} := \frac{1}{m} \sum_{j=1}^{m} l(a_j, y_j; x). \tag{1.7}$$

La funzione $\mathcal{L}_{\mathcal{D}}$ calcola in media per ogni dato quanta distanza c'è tra $y_j$ e $\phi(a_j)$ tramite il parametro $x$, distanza indicata attraverso la funzione $l(a_j, y_j; x)$ che è detta *loss function*.

Una volta calcolato $x$ minimizzando $\mathcal{L}_{\mathcal{D}}$ sull'insieme $\mathcal{D}$, l'obiettivo è assegnare a nuovi dati che non appartengono al data set di partenza un valore plausibile che chiamiamo *previsione*. Dato un elemento $\hat{a}$ che non appartiene a $\mathcal{D}$, poniamo come la previsione di $\hat{a}$ il valore di $\phi(\hat{a})$, dove $\phi$ è validata tramite $x$ calcolato nella fase di training. Può succedere che la funzione $\phi$ presenti qualità particolari, ad esempio che solo alcune delle caratteristiche dei dati siano rilevanti per un calcolo ottimale dei parametri e quindi ci si possa basare solo su quelli (si chiama *feature selection*). Oppure, se $x \in M_m(\mathbb{R})$ (spazio delle matrici reali quadrate di dimensione $m$), può essere una matrice di un certo tipo, come di basso rango, sparsa o che contiene tutti i dati in un certo sottospazio.

Gli $y_j$, invece, sono di natura diversa in base al problema affrontato.

- Se $y_j \in \mathbb{R}$, $j = 1, \dots, m$, si parla di problema di *regressione*.

  L'esempio più semplice è quello della regressione lineare, in cui la funzione $\phi$ è lineare rispetto a $x$. Consideriamo il data set **Air**[2], in cui sono state raccolte le misurazioni delle concentrazioni di gas inquinanti in una città italiana ogni ora per 12 mesi, dal marzo 2004 a febbraio 2005. Come fatto in [1], si può costruire una regressione lineare che leghi la concentrazione di benzene nell'aria (valori $y$) con altri

[1] Le sezioni che seguono sono tratte da [10]

[2] https://archive.ics.uci.edu/ml/datasets/Air+Quality

sette parametri che sono l'umidità relativa, la temperatura e poi le concentrazioni di ossido di stagno, di ossido di titanio, di ossido di azoto di tungsteno, di diossido di azoto di tungsteno e ossido di indio. Grazie ad un insieme di dati $a_j$ riguardanti i sette parametri e prendendo come loss function da minimizzare $\|Ax - y\|$, con $A$ matrice dei vettori $a_j$, allora si ottiene un valore approssimativo delle caratteristiche $x$ che sono poi lo strumento usato per le nuove previsioni[3].

- Se $y_j$ è una etichetta che può assumere solo valori interi da un insieme $\{1, \ldots, M\}$, si tratta di un problema di *classificazione*. Si dirà *binario* quando $M = 2$, altrimenti è detto problema di classificazione *multiclasse*. L'obiettivo è assegnare a quale classe appartiene ogni dato.

- Può accadere che nel data set non siano note le $y_j$ ma solo gli $a_j$ e si possono fare comunque studi riguardo a questi problemi. In questo caso, si può essere interessati a fare un raggruppamento in *cluster* dei dati oppure uno o ad identificare sottospazi di dimensione bassa che li contengano. In questo caso, gli $y_j$ vengono determinati durante la costruzione della funzione $\phi$. Nel caso dei cluster, per esempio, $y_j$ risulterà il cluster a cui $a_j$ appartiene e $\phi$ è la funzione che assegna ad ogni $a_j$ il suo cluster.

La formulazione di problemi di ottimizzazione per l'analisi dei dati può avere delle caratteristiche che la rendono complessa. Per esempio, i dati a disposizione possono essere incompleti (se mancano alcuni degli $a_j$ o $y_j$), corrotti o con del rumore.

Un altro rischio è l'*overfitting*. Quando troviamo i parametri ottimali $x$, questi possono funzionare molto bene sul data set di partenza e darne una buona rappresentazione, ma il nostro interesse il più delle volte è poter predire bene anche le etichette dei dati che sono fuori da $\mathcal{D}$. Se sull'insieme di partenza si ha un errore molto basso ma fuori dall'insieme il modello non è molto buono, allora si parla di overfitting e di fatto non rappresenta la soluzione migliore. Questo accade anche perchè i dati dentro e fuori il data set possono essere di natura diversa (Esempio: se $\mathcal{D}$ riguarda le vendite di certi prodotti solo in alcuni mesi dell'anno, questo non genera un modello robusto sulle vendite di tutto l'anno).

Cambiare leggermente la funzione obiettivo è una possibilità spesso usata per avere un modello più robusto. Si aggiunge un termine detto *funzione di regolarizzazione* o *regolarizzatore*, indicato qui sotto con $pen(x)$ (da *penalty function*), alla loss-function vista in (1.7)

$$\min_{x \in \Omega} \mathcal{L}_{\mathcal{D}}(x) + \lambda pen(x). \tag{1.8}$$

In (1.8) $\Omega$ è l'insieme dei valori ammissibili per $x$, cioè contiene i vincoli imposti in base al problema. Ad esempio, in alcune applicazioni, i valori negativi di $x$ potrebbero non essere significativi, si richiede allora che i parametri siano maggiori o uguali a zero. Lo scalare $\lambda \geq 0$ è il parametro di regolarizzazione e la sua scelta ha un ruolo importante

[3]Per approfondire la regressione multilineare si veda [4]

poiché bilancia l'aderenza ai dati con la complessità della $\phi$ da determinare. Più piccolo il valore di $\lambda$, più il modello "fitterà" con gli elementi di $\mathcal{D}$; più alto il valore, meno complesso sarà il modello.

**Esempio di regolarizzazione**

Supponiamo di avere una funzione obiettivo della forma quadratica

$$f(x) = \frac{1}{2}x^T Ax + b^T x \tag{1.9}$$

con $A$ matrice simmetrica e semidefinita positiva di dimensione $n$ e poi $b \in \mathbb{R}^n$. $A$ quindi ha autovalori $\mu_i \geq 0$. Abbiamo che

$$\nabla f(x) = Ax - b, \quad \nabla^2 f(x) = A \geq 0.$$

Con il termine di regolarizzazione otteniamo una nuova funzione fatta così

$$\tilde{f}(x) = \frac{1}{2}x^T Ax + b^T x + \lambda \|x\|_2^2, \ \lambda > 0. \tag{1.10}$$

dove con $\|\cdot\|_2$ indichiamo la norma 2 euclidea. L'Hessiana di questa nuova funzione $\tilde{f}$

$$\nabla^2 \tilde{f}(x) = A + \lambda I$$

è una matrice definita positiva poiché i suoi autovalori $\mu_i + \lambda$ sono tutti strettamente positivi. Avendo Hessiana definita positiva, la funzione quadratica $\tilde{f}$ risulta una funzione fortemente convessa e questa condizione ci dice che un minimo locale è anche globale (quindi unicità della soluzione). Cioè, abbiamo visto che una leggera modifica della funzione di partenza permette di acquisire una caratteristica molto importante. Il punto critico è la scelta di un $\lambda$ adeguato, che dia regolarizzazione senza cambiare in maniera eccessiva la funzione.

Nelle prossime sezioni, vengono presentati alcuni problemi tipici nella data science che sono casi particolari del problema (1.8).

## 1.3 Problemi ai minimi quadrati: la regressione lineare

Uno dei problemi di ottimizzazione più conosciuti e studiati è il problema ai minimi quadrati lineare. Consideriamo il data set $\mathcal{D} = \{(a_j, y_j) \in \mathbb{R}^n \times \mathbb{R} | j = 1, \ldots, m\}$ e indichiamo con $A$ la matrice le cui righe sono i vettori $a_j^T$. L'idea è porre $\phi(a) := a^T x$ come approssimazione del valore $y_j$ calcolando l'errore di approssimazione usando la metrica euclidea. Notiamo che la funzione $\phi$ può presentare anche un termine reale $\beta$ come termine noto della funzione lineare. Vogliamo quindi risolvere il seguente problema

$$\min_x \frac{1}{2m} \sum_{j=1}^{m} \left(a_j^T x - y_j\right)^2 = \min_x \frac{1}{2m} \|Ax - y\|_2^2. \tag{1.11}$$

Il problema può essere regolarizzato aggiungendo un termine di regolarizzazione per dare al modello particolari caratteristiche ed esistono varie possibilità. Vediamo le due principali tecniche di regolarizzazione. La prima è detta *ridge regression* e comporta l'aggiunta di un termine di penalità con norma $l_2$, ottenendo

$$\min_x \frac{1}{2m}\|Ax - y\|_2^2 + \lambda\|x\|_2^2, \tag{1.12}$$

con $\lambda$ parametro strettamente positivo. La soluzione di questo problema è meno sensibile alle perturbazioni dei dati $(a_j, y_j)$.

Un'altra formulazione è la LASSO (Least Absolute Shrinkage and Selection Operator)

$$\min_x \frac{1}{2m}\|Ax - y\|_2^2 + \lambda\|x\|_1$$

che promuove soluzioni sparse, quindi con pochi elementi non nulli. Questo permette la cosiddetta *feature selection*: la posizione delle componenti non nulle di $x$ indica quali sono le componenti degli $a_j$ utili a calcolare il valore delle osservazioni $y_j$ Ovviamente, un modello che dipende da poche caratteristiche è più semplice e permette una estrapolazione più facile dei risultati. In particolare, quando si ha un nuovo dato fuori dal data set di cui si vuole fare la previsione, solo alcune componenti sono necessarie e non tutte grazie alla feature selection.

## 1.4 Problemi di fattorizzazione di matrice

In molti problemi della data analysis è richiesto di trovare una matrice di rango basso a partire da un insieme di dati. Un problema così può essere pensato come una generalizzazione del problema ai minimi quadrati in cui i dati $a_j$ non sono vettori ma matrici.

Perciò, consideriamo $A_j \in M_{n\times p}(\mathbb{R})$, per ogni $j = 1, \ldots, m$ e poi $y_j \in \mathbb{R}$, cerchiamo una matrice $X \in M_{n\times p}(\mathbb{R})$ che risolva

$$\min_X \frac{1}{2m}\sum_{j=1} m\left(\langle A_j, X\rangle - y_j\right)^2, \tag{1.13}$$

dove $\langle A, B\rangle := traccia(A^T B)$. Le osservazioni con cui lavoriamo possono essere di tipo diverso, tra i più comuni ci sono $y_j$ che sono combinazioni lineari randomiche (per cui gli $A_j$ vengono scelti in maniera indipendente e identicamente distribuita secondo una distribuzione data) oppure sono ad elementi singoli (per cui gli $A_j$ presentano un 1 in una determinata posizione e 0 nelle altre).

Anche di questo problema si può studiare la versione regolarizzata

$$\min_X \frac{1}{2m}\sum_{j=1}^{m}\left(\langle A_j, X\rangle - y_j\right)^2 + \lambda\|X\|_*, \tag{1.14}$$

che restituisce una soluzione di basso rango. Con $\|X\|_*$ indichiamo la norma nucleare della matrice $X$, cioè la somma dei valori singolari della matrice. Il rango basso della

soluzione si ottiene grazie all'uso della norma nucleare, che ha un ruolo analogo alla norma 1 nel problema (1.12), in cui la presenza della norma 1 restituisce vettori sparsi.

## 1.5 Support Vector Machines

Il Support Vector Machines è un problema classico di ottimizzazione nato negli anni '60.

Sia $\{(a_j, y_j)\}$ il data set con $a_j \in \mathbb{R}^n$ e $y_j \in \{-1, 1\}$, il problema chiede di trovare il vettore $x \in \mathbb{R}^n$ e lo scalare $\beta \in \mathbb{R}$ tali che

$$\begin{aligned} a_j^T x - \beta \geq 1, \text{ con } y_j = +1, \\ a_j^T x - \beta \leq -1, \text{ con } y_j = -1. \end{aligned} \tag{1.15}$$

Una coppia $(x, \beta)$ che soddisfa queste due condizioni definisce un *iperpiano separatore* in $\mathbb{R}^n$, cioè un iperpiano che divide lo spazio in due sottospazi, uno contenente tutti i casi "positivi" $\{a_j | y_j = 1\}$, l'altro tutti i casi "negativi" $\{a_j | y_j = -1\}$. Si può dimostrare che si ottiene l'iperpiano che massimizza il margine tra le due classi minimizzando $\|x\|^2$. Con margine si intende la distanza tra iperpiano e il punto più vicino all'iperpiano della classe.

Il problema di trovare l'iperpiano migliore può essere impostato come problema di ottimizzazione. Definiamo la funzione obiettivo da minimizzare che ci aiuta a questo scopo

$$H(x, \beta) = \frac{1}{m} \sum_{j=1}^{m} \max(1 - y_j(a_j^T x - \beta), 0). \tag{1.16}$$

Quando il $j$-esimo addendo della sommatoria è nullo, vuol dire che le condizioni (1.15) sono soddisfatte, altrimenti ci sarebbe un residuo di errore positivo e si avrebbe $H > 0$. Se anche non esistesse alcuna coppia $(x, \beta)$ tale che $H(x, \beta)$ sia uguale a zero, si può comunque cercare il miglior $(x, \beta)$ che minimizza l'equazione e aver quindi un iperpiano che meglio di tutti gli altri divide i due casi.

Nuovamente, si può aggiungere regolarità al problema grazie ad un termine di secondo grado

$$H(x, \beta) = \frac{1}{m} \sum_{j=1}^{m} \max(1 - y_j(a_j^T x - \beta), 0) + \frac{1}{2}\lambda \|x\|_2^2. \tag{1.17}$$

Con un $\lambda$ abbastanza piccolo e se un iperpiano separatore esiste, la soluzione di (1.17) è l'iperpiano con il margine massimo, che risulta una proprietà utile a classificare i dati nelle due casistiche. Se ad esempio ci fosse un nuovo dato $a_j$, il modello così definito saprebbe dire in maniera abbastanza ragionevole a quale sottospazio appartiene. Un iperpiano che, non essendo di margine massimo, passa vicino alla nuvola di punti di partenza, in generale non riuscirebbe a fare lo stesso lavoro. La Figura 1.1 restituisce un'idea di questo concetto.

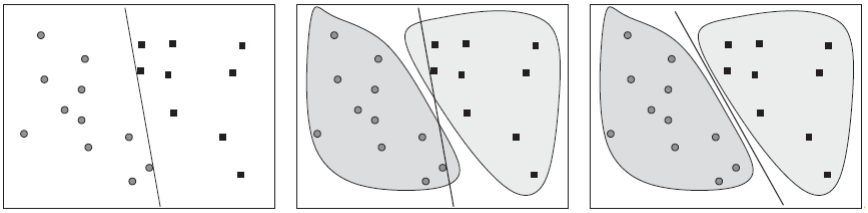


Figura 1.1: Nei grafici si vedono due esempi di iperpiani separatori. Nell'immagine di sinistra l'iperpiano non ha margine massimo, per cui può non restituire buone previsioni per gli eventuali dati da studiare, come ad esempio si vede nella nuvola di punti dell'immagine centrale. A destra, l'iperpiano di massimo margine divide bene le nuvole di dati.[10]

Può accadere invece che non sia possibile trovare un iperpiano che separi bene i casi positivi da quelli negativi. Una soluzione può essere l'utilizzo di una funzione non lineare $\psi$ che trasforma i vettori riga $a_j$ per poi applicare il metodo SVM sulle immagini. Per cui, le condizioni da porre su $x$ e $\beta$ sarebbero

$$\begin{aligned} \psi(a_j)^T x - \beta \geq 1, \text{ con } y_j = 1, \\ \psi(a_j)^T x - \beta \leq -1, \text{ con } y_j = -1. \end{aligned} \tag{1.18}$$

Da cui si ottiene poi come funzione obiettivo

$$H(x, \beta) = \frac{1}{m} \sum_{j=1}^{m} \max(1 - y_j(\psi(a_j)^T x - \beta), 0) + \frac{1}{2}\lambda \|x\|_2^2. \tag{1.19}$$

Dopo aver minimizzato la funzione, si applica la trasformazione inversa e si ottiene l'insieme $\{a|\psi(a)^T x - \beta = 0\}$ come classificatore dei dati, un classificatore più potente di quelli lineari. Da notare che l'insieme trovato è non lineare e potrebbe essere anche disconnesso.

La Figura 1.2 dà un esempio di come si possa adoperare una separazione non lineare.

## 1.6 Regressione logistica

La regressione logistica può essere pensata come una versione debole della classificazione fatta mediante Support Vector Machines binario. Dato un nuovo vettore di dati $a$, invece di cercare la funzione $\phi$ che cerca di predire a quale classe appartiene questo dato, il metodo della regressione logistica prova a dare una stima delle probabilità che ha $a$ di appartenere ad una classe o all'altra. Si cerca di determinare la "odds function" $p$ che dipende dai parametri $x \in \mathbb{R}^n$ definita come

$$p(a; x) := \frac{1}{1 + exp(a^T x)}, \tag{1.20}$$

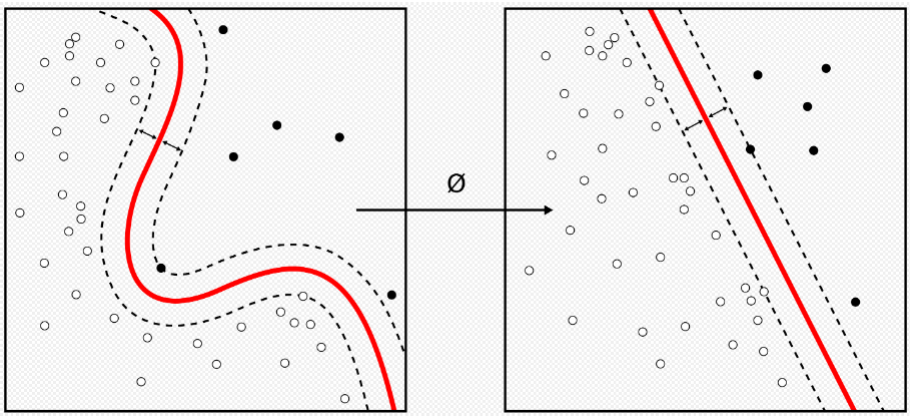



Figura 1.2: (https://en.wikipedia.org/wiki/ Support_vector_machine) Un esempio di mappa non lineare che permette la classificazione quando invece un iperpiano separatore non esiste .

e $x$ viene scelto in modo tale che

$$\begin{aligned} p(a_j; x) &\approx 1, \text{ quando } y_j = 1 \\ p(a_j; x) &\approx 0, \text{ quando } y_j = -1. \end{aligned} \tag{1.21}$$

Per trovare la $x$ ottimale, usiamo la funzione di verosimiglianza logaritmica negativa e ne cerchiamo il minimo. In generale, dato un campione di dati osservati $b$, supponiamo che seguano una certa distribuzione di probabilità $P$ e siano quindi la realizzazione di una $v.a.$ $B \sim P$. Sia $P$ dipendente da uno o più parametri $\theta$ in partenza incogniti, quindi $P = P_\theta$, allora *la funzione di verosimiglianza* è una funzione che assume valore $P_\theta(B = b)$, in funzione dei $\theta$, cioè la probabilità che la variabile $B$ assuma esito $b$ (nel nostro esempio, $b = a$, $\theta = x$).

I parametri che massimizzano questa funzione sono i valori ottimali (in quanto si hanno le odds function più vicine a 1). I calcoli risultano più comodi nell'ipotesi in cui i dati siano raccolti dalla distribuzione in maniera indipendente. Così si ha un prodotto delle odds function, poi si cambia il massino nell'opposto del minimo ed infine si applica il logaritmo che è una funzione monotona (si parla infatti di log-likelihood).

Allora la funzione di veorsimiglianza si può scrivere come somma di loss function:

$$L(x) := -\frac{1}{m}\left[\sum_{j:y_j=-1} \log(1 - p(a_j; x)) + \sum_{j:y_j=1} \log(p(a_j; x))\right]. \tag{1.22}$$

Notiamo che per come è definito, $p(a_j; x) \in (0, 1)$, $\forall a, x$. Perciò si ha che $\log(1 - p(a_j; x)) < 1$ e $\log p(a_j; x) < 1$. Inoltre, quando $p(a_j; x)$ soddisfa una delle condizioni in (1.21), allora il termine nella sommatoria di (1.22) è negativo e molto vicino a zero. Con il meno davanti diventa positivo e più è piccolo il valore e più la $x$ si avvicina ad essere un parametro ottimale.

Come visto nel problema ai minimi quadrati, con l'aggiunta di un termine $\lambda\|x\|_1$ si può ottenere la feature selection. Lo scalare $\lambda > 0$ è detto parametro di regolarizzazione.

La tecnica della regressione logistica si può estendere a problemi con $M > 2$ classi. È un metodo molto usato nell'analisi dati moderna ed un esempio classico è il riconoscimento vocale, in cui le $M$ classi sono tutti i vari fonemi che l'uomo può emettere e il sistema deve classificare.

Il problema di regressione logistica multiclasse richiede quindi diverse odds function che indichiamo con $p_k$, con $k \in \{1, \dots, M\}$. Ognuna è parametrizzata da un vettore $x_{[k]} \in \mathbb{R}^n$ (indicato così per distinguerlo dalle componenti $x_k$). Si ha quindi

$$p_k(a; X) := \frac{\exp\left(a^T x_{[k]}\right)}{\sum_{l=1}^{M} \exp a^T x_{[l]}},\ k = 1, 2, \dots, M, \tag{1.23}$$

dove $X := \{x_{[k]} \in \mathbb{R}^n | k = 1, 2, \dots, M\}$. Queste funzioni sono ottenute applicando la funzione "softmax" al vettore che ha componente $k$ uguale a $a^T x_{[k]}$. La funzione softmax, in generale, prende un vettore di $\mathbb{R}^M$ e restituisce un altro vettore lungo uguale, le cui componenti sono gli esponenziali normalizzati delle componenti del vettore, come in (1.23). Il vettore così ottenuto (e quindi le $p_k$ definite sopra) sono funzioni a valori positivi, con $p_k(a; X) \in (0, 1)$, ed inoltre $\sum_{l=1}^{M} p_l(a; X) = 1$.

Ogni dato $a_j$ ha la sua etichetta $y_j$ che è un vettore di $\mathbb{R}^M$ fatto così

$$(y_j)_k = \begin{cases} 1 & \text{quando } a_j \text{ appartiene alla classe } k \\ 0 & \text{altrimenti.} \end{cases} \tag{1.24}$$

L'obiettivo è trovare gli $x_{[k]}$ tali che

$$\begin{aligned} p_k(a_j; X) &\approx 1 \text{ quando } y_{jk} = 1 \\ p_k(a_j; X) &\approx 0 \text{ quando } y_{jk} = 0 \end{aligned} \tag{1.25}$$

e risolviamo questo minimizzando di nuovo la funzione di verosimiglianza negativa

$$L(X) := -\frac{1}{m} \sum_{j=1}^{m} \left[ \sum_{l=1}^{M} y_{jl} (x_{[l]}^T a_j) - \log \left( \sum_{l=1}^{M} \exp x_{[l]}^T a_j \right) \right]. \tag{1.26}$$

**Osservazione**

Da questi esempi si può notare come molti problemi di ottimizzazione si possano scrivere attraverso una funzione obiettivo della forma (1.7) di cui si cerca il minimo. Problemi di classificazione, regressione e decisionali rientrano nella classe dei problemi di *minimizzazione del rischio empirico.*

Sia $P$ una distribuzione da cui sono generati dei dati $(x, y) \in \mathcal{D}$ e sia $l(u, v)$ una loss function, definiamo la *funzione di rischio atteso* come

$$R[f] := \mathbb{E}_{(x,y)\sim P}[l(f(x), y)] \tag{1.27}$$

dove il valore atteso è calcolato rispetto alla distribuzione $P$ sullo spazio dei dati $(x, y)$. La funzione $l$ è la loss function che misura la "perdita" nell'assegnare $f(x)$ quando la stima vera è $y$. Cioè, $R$ misura in media quanto si perde assumendo come legge dei dati $f$ rispetto alla distribuzione di probabilità vera $P$.

Trovare la $f$ che minimizza il rischio spesso non è facile, soprattutto dal punto di vista computazionale. La (1.27) è difficile da usare nei problemi pratici. Solitamente si lavora con un insieme di campioni $(x_i, y_i)$ presi i.i.d. e si considera una stima della funzione di rischio attraverso la cosiddetta funzione di *rischio empirico*

$$R_{emp}(f) := \frac{1}{N} \sum_{i=1}^{N} l(f(x_i), y_i), \tag{1.28}$$

una funzione che risulta della forma vista in (1.7). Le funzioni di rischio appena definite sono legate dal fatto che il rischio empirico in media è uguale al rischio teorico (come mostrato in [10], Capitolo 5)

$$\mathbb{E}[R_{emp}(f)] = R(f) \tag{1.29}$$

Quindi i problemi che rientrano in questa classe possono essere affrontati come l'equazione (1.7) usando alcuni metodi ben conosciuti. Tra questi, una classe importante sono i metodi del gradiente, presentati nel prossimo capitolo.

# Capitolo 2

# Metodi del gradiente per l'ottimizzazione non vincolata

I *metodi del gradiente* sono metodi iterativi utilizzati per risolvere problemi di minimizzazione di funzioni differenziabili non vincolati facendo uso del gradiente della funzione.

Consideriamo $f : \mathbb{R}^n \to \mathbb{R}$ differenziabile, vogliamo approssimare la soluzione del problema (1.1) definito in precedenza.

Assumiamo che $f$ abbia gradiente Lipschitziano con costante $L > 0$, cioè che soddisfi

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \ \forall x, y \in \mathbb{R}^n, \tag{2.1}$$

(viene detto anche che $f$ è *L-smooth*). Data un'approssimazione iniziale $x^0 \in \mathbb{R}$, un metodo iterativo del gradiente genera una successione di punti $\{x^k\}_{k=0,1,\ldots}$ attraverso la formula

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k) \tag{2.2}$$

dove $\alpha_k$ è detto *lunghezza del passo*. Il metodo del gradiente è detto anche metodo di più ripida discesa, poiché massimizza il prodotto scalare $\nabla f(x^k)^T d^k$, con $d^k \in \mathbb{R}^n$.

Per approfondire lo studio di questi metodi, presentiamo il seguente lemma

**Lemma 2.1** (Lemma di discesa)**.** *Data una funzione $f : \mathbb{R}^n \to \mathbb{R}$ L-smooth, allora*

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{L}{2}\|y - x\|^2, \ \forall x, y \in \mathbb{R}^n. \tag{2.3}$$

*In particolare, sia $y = x + \alpha d$, $\alpha > 0$, $d \in \mathbb{R}^n$, allora:*

$$f(x + \alpha d) \leq f(x) + \alpha \nabla f(x)^T d + \alpha^2 \frac{L}{2}\|d\|^2, \ \forall x, y \in \mathbb{R}^n. \tag{2.4}$$

*Dimostrazione.* Usiamo il teorema di Taylor:

$$\begin{aligned} f(x) &= f(y) + \int_0^1 \nabla f(x+\tau(x-y))^T(x-y)\,d\tau = \\ &= f(y) + \nabla f(y)^T(x-y) + \int_0^1 (\nabla f(x+\tau(y-x)) - \nabla f(y))^T(x-y)\,d\tau \leq \\ &\leq f(y) + \nabla f(y)^T(x-y) + \int_0^1 \|\nabla f(x+\tau(y-x)) - \nabla f(y)\|\|x-y\|\,d\tau \leq \\ &\leq f(y) + \nabla f(y)^T(x-y) + L\int_0^1 \|x-y\|^2\tau\,d\tau \\ &= f(y) + \nabla f(y)^T(x-y) + \frac{L}{2}\|x-y\|^2. \end{aligned}$$

□

Sostituendo $x = x^k$ e $d = -\nabla f(x^k)$ nella disuguaglianza (2.4) e minimizzando il membro di destra della disequazione rispetto ad $\alpha$, si ottiene il minimo per $\alpha = 1/L$. Dato questo valore, consideriamo l'iterazione del metodo di discesa

$$x^{k+1} = x^k - \frac{1}{L}\nabla f(x^k),\ k = 0, 1, \ldots.$$

Si ottiene che

$$f(x^{k+1}) = f\left(x^k - \frac{1}{L}\nabla f(x^k)\right) \leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|^2. \tag{2.5}$$

La condizione trovata in (2.5) è una condizione di decrescita sufficiente fondamentale che ci permette di studiare l'analisi di convergenza del metodo del gradiente

### Velocità di convergenza del metodo di più ripida discesa: caso generale

Supponiamo di avere $f$ non convessa, con derivata continua e Lipschitziana. Supponiamo inoltre che sia limitata inferiormente. Grazie a (2.5) possiamo mostrare la convergenza sublineare del metodo.

**Teorema 2.1.** *Sia $f$ una funzione differenziabile con derivata continua e Lipschitziana e sia $f \geq \bar{f}$, con $\bar{f}$ costante. Allora applicando il metodo di massima discesa con punto iniziale $x^0$ e lunghezza di passo $\alpha_k = 1/L$, si ha per ogni $T \geq 0$ che*

$$\min_{0\leq k\leq T-1} \|\nabla f(x^k)\| \leq \sqrt{\frac{2L[f(x^0) - f(x^T)}{T}} \leq \sqrt{\frac{2L[f(x^0) - \hat{f}]}{T}}. \tag{2.6}$$

*Dimostrazione.* Si ha che

$$\min_{0\leq k\leq T-1} \|\nabla f(x^k)\| = \sqrt{\min_{0\leq k\leq T-1} \|\nabla f(x^k)\|^2} \leq \sqrt{\frac{1}{T}\sum_{k=0}^{T-1} \|\nabla f(x^k)\|^2}$$

(il minimo è minore o uguale della media). Da (2.5),

$$\|\nabla f(x^k)\|^2 \leq 2L(f(x^k) - f(x^{k+1})).$$

Sommando fino al termine $T-1$ e sfruttando la serie telescopica risultante si ha

$$\sum_{k=0}^{T-1} \|\nabla f(x^k)\|^2 \leq 2L \sum_{k=0}^{T-1} [f(x^k) - f(x^{k+1})] = 2L[f(x^0) - f(x^T)]. \tag{2.7}$$

Infine $f$ è limitata dal basso da $\bar{f}$, perciò

$$\min_{0\leq k\leq T-1} \|\nabla f(x^k)\| \leq \sqrt{\frac{1}{T}\sum_{k=0}^{T-1} \|\nabla f(x^k)\|^2} \leq \sqrt{\frac{2L}{T}[f(x^0) - f(x^T)]} \leq \sqrt{\frac{2L[f(x^0) - \bar{f}]}{T}}.$$

□

Questo risultato ci dice che tra le prime $T-1$ iterate se ne ha una per cui la norma del gradiente in quel punto è minore di $\sqrt{2L[f(x^0) - \bar{f}]/T}$, cioè si ha convergenza sublineare di primo tipo[1]. Altra conseguenza di questo risultato è che dal passaggio (2.7) si ha che per una funzione limitata dal basso ogni punto di accumulazione della successione $\{x_k\}$ è anche un punto stazionario.

### Velocità di convergenza del metodo di più ripida discesa: caso convesso

**Teorema 2.2.** *Sia $f$ una funzione convessa e con gradiente continuo e Lipschitziano, di costante $L$. Assumiamo che $x^*$ sia soluzione del problema (1.1) per una tale $f$. Allora il metodo di più ripida discesa con $\alpha_k = 1/L$ genera una successione di iterate $\{x^k\}_{k=0}^{\infty}$ che soddisfa*

$$f(x^T) - f(x^*) \leq \frac{L}{2T}\|x^0 - x^*\|^2, \ \forall T > 0. \tag{2.8}$$

*Dimostrazione.* Poiché $f$ è convessa, si ha che $f(x^*) \geq f(x^k) + \nabla f(x^k)^T(x^* - x^k)$ e la sostituiamo in (2.5). Perciò per ogni $k = 0, 1, 2, \ldots$ vale che

$$\begin{aligned} f(x^{k+1}) &\leq f(x^*) + \nabla f(x^k)^T(x^k - x^*) - \frac{1}{2L}\|\nabla f(x^k)\|^2 \\ &= f(x^*) + \frac{L}{2}\left(\|x^k - x^*\|^2 - \|x^k - x^* - \frac{1}{L}\nabla f(x^k)\|^2\right) \\ &= f(x^*) + \frac{L}{2}\left(\|x^k - x^*\|^2 - \|x^{k+1} - x^*\|^2\right). \end{aligned}$$

[1]Sia $\phi_k$ una misura di errore alla iterazione $k$ (ad esempio la norma del gradiente o $\|x^k - x^*\|$, con $x^*$ soluzione, ecc), si dice che l'algoritmo ha tasso di convergenza sublineare se esiste una costante positiva $C$ per cui:

- tipo 1, $\phi_k \leq C/\sqrt{k}$
- tipo 2, $\phi_k \leq C/k$
- tipo 3, $\phi_k \leq C/k^2$

dove nella prima uguaglianza abbiamo usato l'identità $a^2 - (a-b)^2 = 2ab - b^2$, con $a = x^k - x^*$ e $b = \frac{1}{L}\nabla f(x^k)$, mentre nella seconda abbiamo usato semplicemente la definizione di $x^{k+1}$.

Sommiamo su $k = 0, \ldots, T-1$ e poi notiamo la somma telescopica che ne risulta (notazione: $f^* = f(x^*)$)

$$\begin{aligned}\sum_{k=0}^{T-1}(f(x^{k+1}) - f^*) &\leq \frac{L}{2}\sum_{k=0}^{T-1}\left(\|x^k - x^*\|^2 - \|x^{k+1} - x^*\|^2\right) \\ &= \frac{L}{2}\left(\|x^0 - x^*\|^2 - \|x^T - x^*\|^2\right) \leq \frac{L}{2}\|x^0 - x^*\|^2.\end{aligned}$$

La successione $\{f(x^k)\}_k$ è non negativa per la condizione (2.5), perciò abbiamo che $\forall T > 0$

$$f(x^T) - f^* \leq \frac{1}{T}\sum_{k=0}^{T-1}(f(x^{k+1}) - f^*) \leq \frac{L}{2T}\|x^0 - x^*\|^2.$$

□

**Velocità di convergenza del metodo di più ripida discesa: caso fortemente convesso**

Richiamiamo due proprietà delle funzioni fortemente convesse.

- $f$ è limitata dal basso da una funzione quadratica con hessiana $\gamma I$, cioè

$$f(y) \geq f(x) + \nabla f(x)^T(y - x) + \frac{\gamma}{2}\|y - x\|^2. \tag{2.9}$$

- Se $f$ ha gradiente con constante di Lipschitz $L$, è limitata dall'alto da un'analoga funzione quadratica con Hessiana $LI$,

$$f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}\|y - x\|^2.$$

Da questo effetto "sandwich" otteniamo convergenza lineare per il metodo del gradiente.

**Teorema 2.3.** *Sia $f$ una funzione differenziabile con gradiente Lipschitziano di costante $L$ e fortemente convessa, con modulo di convessità $\gamma$. Allora $f$ ha un unico punto di minimo $x^*$ e il metodo di massima discesa con lunghezza di passo $\alpha_k = 1/L$ genera una successione $\{x_k\}_{k=0}^{\infty}$ che soddisfa*

$$f(x^{k+1}) - f(x^*) = \left(1 - \frac{\gamma}{L}\right)(f(x^k) - f(x^*)),\ k = 0, 1, \ldots \tag{2.10}$$

*Dimostrazione.* Esistenza ed unicità del punto di minimo segue dalle proprietà delle funzioni fortemente convesse (si veda Teorema (3.3.14) di [9]). Consideriamo la condizione

(2.9) e minimizziamo entrambi i membri della disequazione rispetto a $y$. A sinistra si ottiene minimo per $y = x^*$, mentre a destra si ha minimo per $y = x - \frac{\nabla f(x)}{\gamma}$. Sostituiamo i valori ottimali in (2.9)

$$\begin{aligned}
&\min_y f(y) \geq \min_y [f(x) + \nabla f(x)^T (y - x) + \frac{\gamma}{2}\|y - x\|^2] \\
&\Rightarrow f(x^*) \geq f(x) - \nabla f(x)^T \left(\frac{1}{\gamma}\nabla f(x)\right) + \frac{\gamma}{2}\|\frac{1}{\gamma}\nabla f(x)\|^2 \\
&\Rightarrow f(x^*) \geq f(x) - \frac{1}{2\gamma}\|f(x)\|^2.
\end{aligned}$$

Isoliamo la norma del gradiente

$$\|f(x)\|^2 \geq 2\gamma[f(x) - f(x^*)].$$

E ora sostituiamo questa condizione in (2.5) ed otteniamo

$$f(x^{k+1}) = f\left(x^k - \frac{1}{L}\nabla f(x^k)\right) \leq f(x^k) - \frac{1}{2L}\|f(x^k)\|^2 \leq f(x^k) - \frac{\gamma}{L}(f(x^k) - f^*).$$

Sottraiamo ad entrambi i membri $f^*$ e si ha la tesi. □

Osserviamo che dopo $T$ passi, si ha

$$f(x^T) - f^* \leq \left(1 - \frac{\gamma}{L}\right)^T (f(x^0) - f^*)$$

cioè si ha convergenza lineare con costante $1 - \frac{\gamma}{L}$.

## 2.1 Metodi di line-search

In generale, il metodo del gradiente rientra in un insieme più ampio di algoritmi, detti *algoritmi di line-search*.

Consideriamo il problema (1.1), con $f$ funzione $L$-smooth ma non necessariamente convessa. Cerchiamo il minimo attraverso un'iterazione della forma

$$x^{k+1} = x^k + \alpha_k d^k, \ k = 0, 1, \ldots \tag{2.11}$$

dove $d^k$ soddisfa la seguente condizione, detta *di discesa*

$$\nabla f(x^k)^T d^k < 0$$

che assicura che, preso un $\alpha_k$ abbastanza piccolo, allora $f(x^k + \alpha_k d^k) < f(x^k)$. Il passo $d^k$, detto *direzione di discesa*. Imponendo su $\alpha_k$ e $d^k$ alcune condizioni, si può ottenere una maggiorazione simile a quella vista in (2.5) per ottenere poi i risultati di convergenza del metodo risultante.

Per quanto riguarda la direzione, dato un certo $\eta > 0$, richiediamo che valga

$$\nabla f(x^k)^T d^k \leq -\eta \|\nabla f(x^k)\| \|d^k\|. \tag{2.12}$$

Mentre per la lunghezza del passo supponiamo valide le *condizioni deboli di Wolfe*, per $c_1$ e $c_2$ costanti tali che $0 < c_1 < c_2 < 1$:

$$\begin{aligned} f(x^k + \alpha_k d^k) &\leq f(x^k) + c_1 \alpha_k \nabla f(x^k)^T d^k \\ \nabla f(x^k + \alpha_k d^k)^T d^k &\geq c_2 \nabla f(x^k)^T d^k. \end{aligned} \tag{2.13}$$

La prima condizione di Wolfe garantisce che ci sia una decrescita ad ogni passo di almeno una frazione $c_1$ della quantità assicurata dallo sviluppo di Taylor. La seconda condizione, invece, dice che la derivata direzione di $f$ lungo $d^k$ è molto meno negativa quando calcolata nell'$\alpha_k$ scelto, rispetto ad $\alpha = 0$. In questo modo il passo non risulta troppo corto. Si può dimostrare che un $\alpha_k$ che soddisfi le (2.13) esiste sempre (si veda [9]).

Sotto queste condizioni (2.12) e (2.13) vale il seguente teorema per i metodi line-search

**Teorema 2.4.** *Sia $f$ differenziabile ed L-smooth ed inoltre sia $f \geq \bar{f}$, con $\bar{f}$ costante. Applichiamo il metodo line-search con $d^k$ tale che valga (2.12), con $\eta > 0$ fissato, ed $\alpha_k$ che gode delle condizioni di Wolfe (2.13), con $c_1$, $c_2$ costanti tali che $0 < c_1 < c_2 < 1$. Allora per ogni $T \geq 1$ si ha che*

$$\min_{0 \leq k \leq T-1} \|\nabla f(x^k)\| \leq \sqrt{\frac{L}{\eta^2 c_1 (1 - c_2)}} \sqrt{\frac{f(x^0) - \bar{f}}{T}}. \tag{2.14}$$

*Dimostrazione.* Usiamo la seconda condizione di Wolfe:

$$\begin{aligned} -(1 - c_2) \nabla f(x^k)^T d^k &= -\nabla f(x^k)^T d^k + c_2 \nabla f(x^k)^T d^k \leq \\ &\leq [\nabla f(x^k + \alpha_k d^k) - \nabla f(x^k)]^T d^k \leq L \alpha_k |d^k\|^2 \end{aligned}$$

per la condizione di Lipschitz sul gradiente. Confrontiamo il primo e ultimo termine di questa catena e otteniamo la seguente condizione su $\alpha_k$

$$\alpha_k \geq -\frac{(1 - c_2)}{L} \frac{\nabla f(x^k)^T d^k}{\|d^k\|^2}$$

da utilizzare nella prima condizione di Wolfe. Usando anche (2.12), si ha che

$$\begin{aligned} f(x^{k+1}) = f(x^k + \alpha_k d^k) &\leq f(x^k) + c_1 \alpha_k \nabla f(x^k)^T d^k \\ &\leq f(x^k) - \frac{c_1 (1 - c_2)}{L} \frac{(\nabla f(x^k)^T d^k)^2}{\|d^k\|^2} \\ &\leq f(x^k) - \frac{c_1 (1 - c_2)}{L} \eta^2 \|\nabla f(x^k)\|^2. \end{aligned}$$

Isoliamo la norma del gradiente

$$\|\nabla f(x^k)\|^2 \leq \frac{L}{c_1(1-c_2)\eta^2}\left(f(x^k) - f(x^{k+1})\right). \tag{2.15}$$

E si procede come si è fatto nella dimostrazione del Teorema 2.1, sfruttando la serie telescopica. □

Osserviamo che da (2.15), applicando il limite ad entrambi i membri, si ha

$$\lim_{k\to\infty} \|\nabla f(x^k)\| = 0,$$

perciò ogni punto di accumulazione $\bar{x}$ della successione $\{x^k\}_k$ è tale che $\nabla f(\bar{x}) = 0$. Se $f$ fosse anche convessa, allora questo ci garantirebbe che $\hat{x}$ è soluzione del problema (1.1). Quando $f$ non è convessa, $\bar{x}$ può essere un punto di minimo, ma anche un punto di massimo o di sella.

# Capitolo 3

# Il metodo del gradiente stocastico e il metodo di Kaczmarz

L'*algoritmo del gradiente stocastico* (abbreviato con SG) è un metodo numerico molto conosciuto ed utilizzato nel Machine Learning. Si tratta di un metodo simile al metodo di discesa del gradiente, ma l'aggiornamento dell'iterazione avviene usando una stima del gradiente invece del gradiente stesso. Si procede quindi in maniera simile per cercare il minimo della funzione che si sta studiando.

Data una funzione $f : \mathbb{R}^n \to \mathbb{R}$ di cui vogliamo calcolare il minimo e che supponiamo essere regolare e convessa, cerchiamo un vettore $g(x, \xi) \in \mathbb{R}^n$ che sia una approssimazione di $\nabla f(x)$ secondo la seguente relazione

$$\nabla f(x) = \mathbb{E}_\xi[g(x, \xi)], \tag{3.1}$$

dove $x \in R^n$, $\xi$ è una variabile aleatoria che appartiene allo spazio che indichiamo con $\Xi$ e ha distribuzione $P$. Con $\mathbb{E}_\xi$ indichiamo il valore atteso fatto sulla variabile $\xi$ secondo la sua distribuzione $P$.

Quando $g$ soddisfa l'equazione (3.1), si dice che $g(x, \xi)$ è una stima senza bias o non distorta di $\nabla f$.

Dato un punto iniziale $x^0 \in \mathbb{R}^n$, il metodo procede in questo modo:

$$x^{k+1} = x^k - \alpha_k g(x^k, \xi^k), \tag{3.2}$$

per $k \geq 0$, con $\xi^k$ che viene scelto in maniera indipendente dalle variabili aleatorie precedenti e con $\alpha_k$ che viene detta *lunghezza del passo* (o *steplength*).

L'idea è che il vettore $g(x, \xi)$ non sia la direzione di discesa più veloce, ma comunque imponga al metodo una discesa in media e per questo si avvicina al minimo della funzione. L'iterazione viene quindi aggiornata in maniera meno ottimale ad ogni passo, ma può avere altri vantaggi, ad esempio il vettore $g(x^k, \xi^k)$ può essere molto meno costoso da calcolare rispetto al gradiente.

Una questione critica del metodo è la scelta di $\alpha_k$, che non deve essere né troppo grande, né troppo piccolo. Nel caso del metodo di massima discesa, abbiamo visto che

$\alpha_k = 1/L$ garantisce la convergenza del metodo, dove $L$ è la costante di Lipschitz di $\nabla f$. SG si comporta in maniera simile ma non ha le stesse proprietà. Un esempio ci dà un'idea: se poniamo $x^0 = x^*$, con $x^*$ punto di minimo, il metodo di discesa non ha altre direzioni da percorrere per far avanzare il metodo, mentre il metodo del gradiente stocastico, avendo il termine aleatorio, potrebbe avere $g(x^0, \xi^0)$ non nullo e allontanarsi dalla soluzione vera.

## 3.1 Il metodo del gradiente stocastico per problemi con somme finite

Consideriamo una funzione obiettivo del tipo

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x), \tag{3.3}$$

con $N$ solitamente molto grande rispetto a $n$, dimensione di $x$. Esempi di questo tipo di problemi sono stati presentati nel Capitolo 1, cioè sono i problemi con funzioni obiettivo che risultano dall'uso della funzione di rischio empirico.

Calcolare il gradiente di questa funzione avrebbe un costo elevato perché bisognerebbe calcolare i singoli gradienti delle funzioni $f_i$, con $i = 1, \ldots, N$ ed $N$ grande appunto. Supponiamo invece di selezionare ad ogni iterazione un indice $i_k \in \{1, \ldots, N\}$ e si considera come direzione di ricerca il gradiente della funzione che ha questo indice, $f_{i_k}$.

L'iterazione ha quindi la seguente forma

$$x^{k+1} = x^k - \alpha_k \nabla f_{i_k}(x^k). \tag{3.4}$$

L'indice $i_k$ può essere scelto in due maniere:

- ciclica sui numeri $\{1, \ldots, N\}$, cioè $i_k = (k \ mod \ N) + 1$, $k = 1, 2, \ldots$ (detto *metodo del gradiente incrementale*)
- randomica, ad esempio si può considerare la variabile aleatoria $\xi^k$ che assume valore $i_k \in 1, \ldots, N$ con probabilità $P(i_k) = 1/N$, $\forall k$, e si avrebbe che $g(x^k, \xi^k) = \nabla f_{i_k}(x^k)$. Questo caso rientra in quello del gradiente stocastico.

## 3.2 Il metodo di Kaczmarz

Consideriamo un problema ai minimi quadrati

$$\min_{x \in \mathbb{R}^n} f(x) := \min_{x \in \mathbb{R}^n} \frac{1}{2N} \sum_{i=1}^{N} (a_i^T x - b_i)^2 = \min_{x \in \mathbb{R}^n} \frac{1}{2N} \|Ax - b\|^2. \tag{3.5}$$

con $A$ matrice $N \times m$, $N \geq m$, i vettori $a_i^T$ sono le righe della matrice $A$ e $b \in \mathbb{R}^N$. Si può vedere lo stesso problema come un sistema lineare $Ax = b$ sovradeterminato di cui cerchiamo la soluzione che minimizza la norma del residuo.

Il metodo di Kaczmarz viene proposto nel 1937 dal polacco Stefan Kaczmarz come metodo iterativo per la soluzione di sistemi lineari. Viene poi riscoperto negli anni '70 nell'ambito della computer tomography, in cui è conosciuto anche con il nome di *Algebraic Recustruction Technique (ART)* [2], ed infatti è stato un metodo implementato nel primo scanner medico [3]. Recentemente [8] è stato adattato al caso di sistemi lineari sovradeterminati con l'uso di tecniche randomiche.

Di fatto è uno dei metodi più popolari per la risoluzione di problemi (3.5). In verità, risulta essere il metodo del gradiente stocastico applicato ai problemi ai minimi quadrati.

Dato un punto iniziale $x^0$, sviluppiamo l'iterazione del gradiente stocastico in questo particolare caso, considerando $\alpha_k = 1$ ed il fatto che

$$g(x, \xi) = a_{i_k}(a_{i_k}^T x - b_{i_k}) \tag{3.6}$$

con $i_k$ scelto in maniera randomica. Aggiungiamo l'ipotesi per cui $\|a_i\| = 1$, $i = 1, \ldots, N$, che in generale possiamo ottenere dividendo $a_i$ per la sua norma (come in precedenza, $\|\cdot\|$ è la norma 2 euclidea). Si ha quindi che

$$x^{k+1} = x^k + \frac{b_i - a_i^T x^k}{\|a_i\|^2} a_i, \tag{3.7}$$

dove $b = (b_1, \ldots, b_N)^T$.

Supponiamo esista $x^*$ tale che $a_i^T x^* = b_i$, $\forall i$, cioè $x^*$ realizza il minimo della funzione $f$, con $f(x^*) = 0$. Allora

$$x^{k+1} = x^k - a_{i_k}(a_{i_k}^T x^k - b_{i_k}) = x^k - a_{i_k} a_{i_k}^T (x^k - x^*)$$

Confrontiamo l'iterata con la soluzione

$$x^{k+1} - x^* = x^k - a_{i_k} a_{i_k}^T (x^k - x^*) - x^* = (I - a_{i_k} a_{i_k}^T)(x^k - x^*) = \prod_{j=1}^{k} (I - a_{i_j} a_{i_j}^T)(x^0 - x^*)$$

dove $I$ indica la matrice identità $N$-dimensionale.

Osserviamo che in ogni iterazione del metodo si ha una proiezione dell'iterata attuale sull'iperpiano di equazione $a_{i_k}^T x = b_{i_k}$. Infatti in letteratura questo metodo è considerato anche un caso particolare di POCS (projection onto Convex Sets)[5]. Questa proprietà ci dice che se due righe successive della matrice sono simili, allora sono il vettore normale di due iperpiani abbastanza vicini e quindi l'iterazione avanza di poco verso il minimo da calcolare.

### Vantaggi della scelta randomica sulla scelta ciclica

Attraverso un esempio, mostriamo che la scelta degli indici $i_k = (k \ mod \ N) + 1$ porta ad una lenta convergenza del metodo di Kaczmarz, che risulta invece più veloce scegliendo $i_k \in \{1, \ldots, N\}$ con probabilità $1/N$. Supponiamo che la matrice $A$ sia composta dalle righe $a_i^T$

$$a_i = \begin{bmatrix} \cos(i\omega_N) \\ \sin(i\omega_N) \end{bmatrix}, \quad i = 1, \ldots, N \tag{3.8}$$

con $\omega_N := \pi/N$ e $N \geq 3$. Siano $b_i = 0$, $\forall i$, così risulta $x^* = 0$. Procediamo usando il metodo di Kaczmarz e ne mostriamo la convergenza calcolando $\mathbb{E}[x^k - x^0]$.

Osserviamo innanzitutto che $\langle a_i, a_{i+1} \rangle = \cos(\omega_N)$ e poi che le matrici $M_i := I - a_i a_i^T$ sono semidefinite positive per ogni $i$. Vale l'identità

$$\mathbb{E}_j[M_j] = \frac{1}{N}\sum_{i=1}^{N} M_i = \frac{1}{N}\sum_{i=1}^{N}(I - a_i a_i^T) = \frac{1}{N}(I - \frac{1}{2}I) = \frac{1}{2}I^1 \tag{3.9}$$

Sia ora $a_{i_k}$ scelto dall'insieme $\{a_1, \ldots, a_N\}$ con probabilità uniforme e scelte indipendenti le une dalle altre ad ogni iterazione. L'errore atteso è

$$\begin{aligned} \mathbb{E}_{i_k}[x^{k+1} - x^* | x^k] = \mathbb{E}_{i_k}[(I - a_i a_i^T)(x^k - x^*) | x^k] = \\ \mathbb{E}_{i_k}[(I - a_i a_i^T)](x^k - x^*) = \frac{1}{2}(x^k - x^*), \end{aligned} \tag{3.10}$$

avendo usato l'equazione (3.9) per ottenere $1/2$. Allora possiamo mostrare che

$$\mathbb{E}[x^k - x^*] = \mathbb{E}_{i_0,\ldots,i_{k-1}}\left[\prod_{j=1}^{k-1}(I - a_{i_j} a_{i_j}^T)(x^0 - x^*)\right] = {}^2$$

$$= \left[\prod_{j=0}^{k-1}\mathbb{E}_{i_j}(M_{i_j})\right](x^0 - x^*) = \left[\mathbb{E}_{i_j}(M_{i_j})\right]^{k-1}(x^0 - x^*) = 2^{-k}(x^0 - x^*).$$

Da questo si vede che il valore atteso dell'errore decresce esponenzialmente con tasso di convergenza $1/2$.

Se usassimo in questo caso il metodo standard con scelta ciclica dei pedici, avremmo una diversa velocità di convergenza.

Siano $i_k = k + 1$, con $k = 1, 2, \ldots, N - 1$. Definiamo i vettori

$$\hat{a}_i = \begin{bmatrix} \sin(-i\omega_N) \\ \cos(-i\omega_N) \end{bmatrix}, \quad i = 1, \ldots,$$

poi notiamo che vale $M_i = I - a_i a_i^T = \hat{a}_i \hat{a}_i^T$ ed inoltre che $\langle \hat{a}_i, \hat{a}_{i+1} \rangle = cos(\omega_N)$. Segue quindi che

$$\prod_{j=1}^{k} M_j = \hat{a}_k \hat{a}_1^T \prod_{j=1}^{k-1}\langle \hat{a}_j, \hat{a}_{j+1} \rangle = \hat{a}_k \hat{a}_1^T (\cos(\omega_N))^{k-1}.$$

[1] Per dimostrarla si sfruttano le proprietà di parità e disparità di coseno e seno, per cui si ha simmetria nei valori: $\cos(i\omega_N) = \cos((N-i+1)\omega_N)$ e $\sin(i\omega_N) = \sin((N-i)\omega_N)$ e nel caso di $N$ dispari, $\cos(\pi/2) = 0$ e $\sin(\pi) = 0$.

[2] Le scelte sono indipendenti, quindi il valore atteso si scambia con il prodotto. $\mathbb{E}_{i_j}$ indica il valore atteso calcolato rispetto alla scelta di $i_j$.

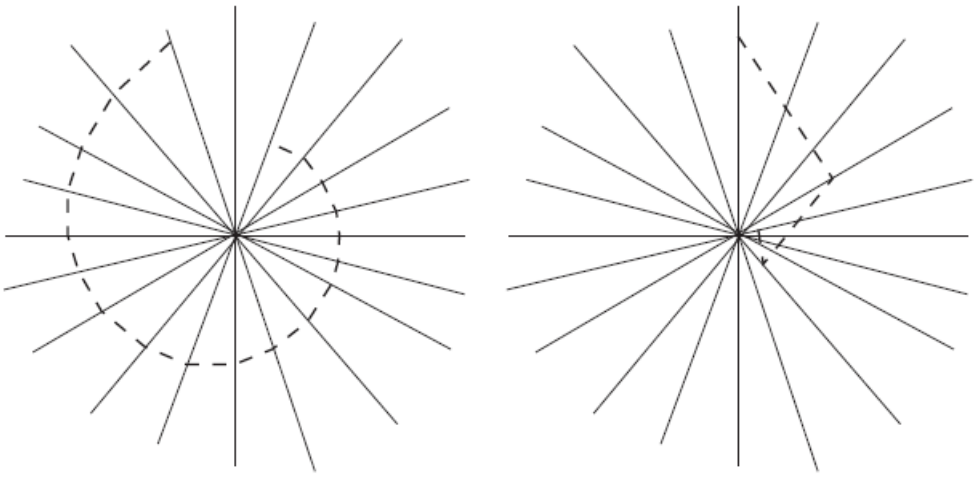


Figura 3.1: Convergenza del metodo di Kaczmarz. A sinistra è mostrata la convergenza con scelta di indice ciclico, la quale è più lenta della convergenza con scelta stocastica, nell'immagine di destra. [10]

Da cui

$$\|x^k - x^*\| = \left\| \prod_{i=1}^{k} (I - a_i a_i^T)(x^0 - x^*) \right\| = (\cos(\omega_N))^{k-1} \left| \hat{a}_1^T \left( x^0 - x^* \right) \right|.$$

Se inizializziamo $x^0 = (0,1)^T$, abbiamo che $\hat{a}_1^T(x^0 - x^*) = \|x^0 - x^*\|$, per cui

$$\|x^k - x^*\| = (\cos(\omega_N))^{k-1} \|x^0 - x^*\|, \ k = 1, 2, \ldots$$

Quindi il metodo ha convergenza lineare con tasso di convergenza pari a $\cos^{k-1}(\omega_N) \approx 1 - (1/2)(\pi/N)^2$, più lenta del tasso di convergenza di $1/2$ raggiunta con il metodo stocastico. Osserviamo comunque che abbiamo fatto un confronto tra un errore deterministico nel caso ciclico e un valore atteso dell'errore nel caso stocastico.

In Figura 3.1 viene mostrato un confronto visivo di quanto visto nell'esempio.

## 3.3 Una condizione utile per lo studio della convergenza del metodo del gradiente stocastico

Per studiare la convergenza del metodo dello SG partiamo considerando una funzione $f : \mathbb{R}^n \to \mathbb{R}$ convessa su cui applichiamo il metodo con il passo visto nell'equazione (3.2) e supponendo che valga la condizione di non bias di $g(x, \xi)$ in (3.1).

Il punto centrale dell'analisi sta nella seguente maggiorazione

$$\mathbb{E}_\xi \left[ \|g(x, \xi)_2^2\| \right] \leq L_g^2 \|x - x^*\| + B^2, \ \forall x \tag{3.11}$$

cioè supponiamo di poter maggiorare il valore atteso di $g$ con la norma dell'errore rispetto a $x^*$ soluzione del problema di minimo, a meno di due costanti $L_g$ e $B$ non negative. Notiamo che il valore atteso è calcolato su $\xi$ e deve valere per ogni $x$, è quindi possibile avere $g$ illimitata ad esempio.

Osserviamo poi che se $L_g = 0$, non possiamo avere $f$ fortemente convessa su un dominio illimitato. Questo perché se $f$ fosse fortemente convessa con modulo di convessità $\gamma$ si avrebbe dalla teoria di questa classe di funzioni che

$$\|\nabla f(x)\| \geq \frac{\gamma}{2}\|x - x^*\|, \ \forall x. \tag{3.12}$$

Inoltre varrebbe la disuguaglianza di Jensen, per cui

$$\|\nabla f(x)\| = \|\mathbb{E}[g(x,\xi)]\|^2 \leq \mathbb{E}[\|g(x,\xi)\|^2].$$

Con $L_g = 0$ avrei questo valore atteso limitato da una costante $B^2$. Ma dalla condizione (3.12) e con il dominio illimitato della funzione, $\nabla f$ può diventare grande a piacimento e quindi le due condizioni non possono valere contemporaneamente.

Mostriamo ora che la condizione (3.11) può essere ottenuta sempre nei vari casi, per poi usarla nello studio della convergenza.

Nel seguito assumiamo che le $\{\xi^k\}$ nella definizione di $g(x^k, \xi^k)$ sono scelte indipendenti e identicamente distribuite da una distribuzione fissata.

### 3.3.1 Caso 1: Gradiente limitato ($L_g = 0$)

Supponiamo che la funzione del gradiente stocastico $g(\cdot,\cdot)$ sia una funzione limitata quasi certamente per ogni $x$. Allora la condizione (3.11) vale con $L_g = 0$.

Prendiamo per esempio la funzione obiettivo (1.21) del caso della regressione logistica vista nella Sezione 1.6

$$f(x) = \frac{1}{N}\sum_{i=1}^{N}(-y_i x^T a_i + \log(1 + \exp(x^t a_i))), \tag{3.13}$$

dove $(a_i, y_i)$ sono i dati, con $y_i \in 0, 1$, $i = 1, 2, \ldots, N$. Il valore di $\xi$ viene estratto con probabilità uniforme nell'insieme $1, \ldots, N$, per cui il gradiente aleatorio è

$$g(x, i) = \left(-y_i + \frac{\exp(x^T a_i)}{1 + x^T a_i}\right) a_i.$$

Poiché la quantità tra parentesi ha norma minore di 1, la condizione (3.11) vale ponendo $L_g = 0$ e $B = \sup_{i=1,\ldots,N} \|a_i\|_2$.

### 3.3.2 Caso 2: Aggiunta di rumore gaussiano

Consideriamo un esempio semplice di gradiente stocastico, in cui

$$g(x, \xi) = \nabla f(x) + \xi, \tag{3.14}$$

cioè il vettore stocastico è uguale al gradiente vero con l'aggiunta di un vettore di rumore gaussiano, con media nulla e varianza $\sigma^2 I$. Dato che $\mathbb{E}[\xi] = 0$, la proprietà di non distorsione (3.1) è valida. In questo caso l'iterazione procede come

$$x^{k+1} = x^k - \alpha_k(\nabla f(x^k) + \xi^k)$$

molto simile al metodo di discesa del gradiente. Abbiamo in questo caso che

$$\mathbb{E}[\|g(x,\xi)^2\|] = \|\nabla f(x)\|^2 + 2\nabla f(x)^T\mathbb{E}[\xi] + \mathbb{E}[\|\xi\|^2] = \|\nabla f(x)\|^2 + n\sigma^2. \tag{3.15}$$

Notando che $\|\nabla f(x)\|^2 = \|\nabla f(x) - \nabla f(x^*)\|^2 \leq L^2\|x - x^*\|^2$, con $L$ costante di Lipschitz di $\nabla f$, la (3.11) è soddisfatta con $L_g = L$ e con $B = \sqrt{n}\sigma$.

### 3.3.3 Caso 3: Metodo di Kaczmarz stocastico

Consideriamo la funzione obiettivo (3.5) per problemi ai minimi quadrati. Assumiamo $a_i \neq 0$ ma non necessariamente $\|a_i\| = 1$, per ogni $i$. Supponiamo esista $x^* \in \mathbb{R}^n$ per cui $f(x^*) = 0$, cioè tale che $a_i^T x^* = b_i,\ \forall i = 1, \ldots, N$. Usiamo questa condizione in (3.5)

$$\begin{aligned} f(x) &= \frac{1}{2N}\sum_{i=1}^{N}(a_i^T x - b_i)^2 = \frac{1}{2N}\sum_{i=1}^{N}(a_i^T x - a_i^T x^*)^2 \\ &= \frac{1}{2N}\sum_{i=1}^{N}(a_i^T(x - x^*))^2 = \frac{1}{2N}\sum_{i=1}^{N}((x - x^*)^T a_i a_i^T (x - x^*)). \end{aligned}$$

Sia $\xi$ variabile aleatoria con esiti $\{1, 2 \ldots, N\}$, abbiamo che

$$g(x,i) = a_i a_i^T(x - x^*).$$

Perciò, il valore atteso della norma di $g$ viene

$$\mathbb{E}[\|g(x,i)\|^2] = \mathbb{E}[\|a_i\|^2 |a_i^T(x - x^*)|^2] \leq \mathbb{E}[\|a_i\|^4]\|x - x^*\|^2$$

e quindi la condizione (3.11) è valida anche nel caso del metodo di Kaczmarz ponendo $L_g = \mathbb{E}[\|a_i\|^4]^{1/2}$ e $B = 0$.

### 3.3.4 Caso 4: Gradiente incrementale

Consideriamo una funzione obiettivo come quella vista in (3.3) e supponiamo che per ogni $i$, la funzione $\nabla f_i$ abbia constante di Lipschitz $L_i$. Sia $\xi$ variabile aleatoria con distribuzione discreta uniforme e che ha come esiti $\{1, \ldots, N\}$. Per ogni $i$-esimo termine $f_i$ definiamo $x^{*i}$ come un punto per cui $\nabla f_i(x^{*i}) = 0$. Allora abbiamo che

$$\begin{aligned}\mathbb{E}_\xi[\|g(x,\xi)\|^2] &= \mathbb{E}_i[\|\nabla f_i(x)\|^2] = \mathbb{E}_i[\|\nabla f_i(x) - \nabla f_i(x^{*i})\|^2] \leq \\ &\leq \mathbb{E}[L_i^2\|x - x^{*i}\|^2] \leq \mathbb{E}[2L_i^2\|x - x^*\| + 2L_i^2\|x^{*i} - x^*\|^2] = {}^3 \\ &= \frac{2}{N}\sum_{i=1}^{N} L_i^2\|x - x^*\|^2 + \frac{2}{N}\sum_{i=1}^{N} L_i^2\|x^{*i} - x^*\|^2.\end{aligned}$$

Quindi in questo caso la (3.11) vale ponendo

$$L_g^2 = \frac{2}{N}\sum_{i=1}^{N} L_i^2, \quad B^2 = \frac{2}{N}\sum L_i^2\|x^{*i} - x^*\|.$$

Notiamo che se $x^{*i} = x^*$, $\forall i$, allora si avrebbe $B = 0$ come si è visto nel caso del metodo di Kaczmarz randomizzato.

## 3.4 Analisi di convergenza

Procediamo con lo studio calcolando la decrescita dell'errore attraverso due tipi di misure. La prima misura è data dal valore atteso del quadrato dell'errore, $\mathbb{E}[\|x - x^*\|^2]$, con $x^*$ soluzione vera e il valore atteso è calcolato rispetto alla variabile aleatoria $\xi^k$ alla iterazione $k$. Questo tipo di misura funziona bene quando la funzione obiettivo $f$ è fortemente convessa, per cui abbiamo un unico punto di minimo.

La seconda misura di ottimalità considera la differenza tra il valore della funzione obiettivo $f(x)$ e il valore $f^*$ calcolato nel punto di minimo $x^*$. Si tratta di una misura utile quando $f$ è una funzione convessa (non è necessario che lo sia fortemente).

Quando $f$ è fortemente convessa, entrambe le misure possono essere maggiorate una con l'altra utilizzando la costante di Lipschitz $L$ e il modulo di convessità $\gamma$ (basta usare le proprietà di convessità e di forte convessità).

Consideriamo la prima misura. Utilizziamo la definizione di iterazione con il metodo SG per sviluppare il bound superiore

$$\begin{aligned}\|x^{k+1} - x^*\|^2 &= \|x^k - \alpha_k g(x^k, \xi^k) - x^*\|^2 = \\ &= \|x^k - x^*\|^2 - 2\alpha_k g(x^k, \xi^k)^T(x^k - x^*) + \alpha_k^2\|g(x^k, \xi^k)\|^2.\end{aligned} \tag{3.16}$$

Applichiamo il valore atteso ad entrambi i membri rispetto alle variabili aleatorie che l'algoritmo incontra fino al passo $k$ e studiamo singolarmente gli addendi che sono a destra dell'uguale. Dette $i_0, \ldots, i_k$ queste variabili, partiamo dal termine centrale della somma $\mathbb{E}[g(x^k, \xi^k)^T(x^k - x^*)]$ e applichiamo la proprietà del valore atteso di una

[3] Nel passaggio centrale abbiamo aggiunto e tolto $x^*$ e poi abbiamo usato l'identità $\|a + b\|^2 \leq \|a\|^2 + \|b\|^2 + 2\|a\|\|b\| \leq 2\|a\|^2 + 2\|b\|^2$). Abbiamo poi sviluppato il valore atteso.

attesa condizionata. Ricordando poi che $x^k$ non dipende da $\xi^k$ ma solo dai precedenti $\xi^0, \ldots, \xi^{k-1}$, otteniamo

$$\begin{aligned}\mathbb{E}[g(x^k,\xi^k)^T(x^k-x^*)] &= \mathbb{E}\left[\mathbb{E}_{\xi^k}[g(x^k,\xi^k)^T(x^k-x^*)|\xi^0,\xi^1,\ldots,\xi^{k-1}]\right] \\ &= \mathbb{E}\left[\mathbb{E}_{\xi^k}[g(x^k,\xi^k)]^T(x-x^*)\right] = \mathbb{E}\left[\nabla f(x^k)^T(x^k-x^*)\right].\end{aligned}$$

in cui abbiamo applicato il Lemma di Freezing (si veda [7], pag. 244) considerando che $x^k$ dipende da $\xi^0, \ldots, \xi^{k-1}$ e non da $\xi^k$, il quale a sua volta è indipendente dalle variabili aleatorie precedenti. La condizione di non distorsione (3.1) conclude questa catena di uguaglianze.

Si procede in maniera analoga per il terzo termine

$$\mathbb{E}[\|g(x^k,\xi^k)\|_2^2] = \mathbb{E}\left[\mathbb{E}_{\xi^k}[\|g(x^k,\xi^k)\|_2^2|\xi^0,\ldots,\xi^{k-1}]\right] \leq \mathbb{E}[L_g^2\|x^k-x^*\|_2^2 + B^2].$$

in cui abbiamo usato nuovamente il Lemma di Freezing e poi la maggiorazione vista in (3.11).

Per comodità nei prossimi passi ci serviamo della notazione

$$A_k := \mathbb{E}[\|x^k - x^*\|^2]. \tag{3.17}$$

Perciò dall'uguaglianza (3.16) otteniamo la disuguaglianza seguente

$$A_{k+1} \leq (1+\alpha_k^2 L_g^2)A_k - 2\alpha_k \mathbb{E}\left[\nabla f(x^k)^T(x^k-x^*)\right] + \alpha_k^2 B^2. \tag{3.18}$$

Dalla condizione (3.18) dimostriamo la convergenza di SG in ognuno dei casi presentati nella Sezione precedente 3.3.

### 3.4.1 Caso 1: $L_g = 0$

Se $L_g = 0$ l'espressione (3.18) diventa

$$A_{k+1} \leq A_k - 2\alpha_k \mathbb{E}\left[\nabla f(x^k)^T(x^k-x^*)\right] + \alpha_k^2 B^2. \tag{3.19}$$

Definiamo ora

$$\lambda_k = \sum_{j=0}^{k} \alpha_j \tag{3.20}$$

e poi

$$\bar{x}^k = \lambda_k^{-1} \sum_{j=0}^{k} \alpha_j x^j. \tag{3.21}$$

Studiamo l'iterata media ottenuta sommando tra loro i passi fino al $k$-esimo e pesandoli rispetto alla lunghezza del passo $\alpha_k$. Confrontiamo il valore della funzione obiettivo in questo punto $\bar{x}^k$ rispetto al valore ottimale. Definiamo prima $D_0 := \|x^0 - x^*\|$ errore

quadratico iniziale, con $x^0$ punto di partenza e $x^*$ soluzione e notiamo anche che $A_0 = D_0^2$. Date $T$ iterazioni, usando la convessità di $f$ e la definizione (3.21),vale la seguente stima

$$\mathbb{E}[f(\bar{x}^T) - f(x^*)] \leq \mathbb{E}[\lambda_T^{-1} \sum_{j=0}^{k} \alpha_j f(x^j) - \sum_{j=0}^{T} \frac{\alpha_j}{\lambda_T} f(x^*)] \tag{3.22}$$

$$\leq \mathbb{E}\left[\lambda_T^{-1} \sum_{j=0}^{T} \alpha_j (f(x^j) - f(x^*))\right] \tag{3.23}$$

$$\leq \lambda_T^{-1} \sum_{j=0}^{T} \alpha_j \mathbb{E}[\langle \nabla f(x^j), x^j - x^* \rangle] \tag{3.24}$$

$$\leq \lambda_T^{-1} \sum_{j=0}^{T} \left[\frac{1}{2}(A_j - A_{j+1}) + \frac{1}{2}\alpha_j^2 B^2\right] \tag{3.25}$$

$$= \frac{1}{2\lambda_T}\left[A_0 - A_{T+1} + B^2 \sum_{j=0}^{T} \alpha_j^2\right] \leq \tag{3.26}$$

$$\leq \frac{D_0^2 + B^2 \sum_{j=0}^{T} \alpha_j^2}{2 \sum_{j=0}^{T} \alpha_j}. \tag{3.27}$$

dove (3.22) segue dalla convessità di $f$ e la definizione (3.21), (3.24) viene nuovamente dalla convessità di $f$, usando (1.4), e dalla sublinearità del valore atteso, (3.25) segue da (3.18) ed infine (3.27) si ottiene eliminando il termine negativo $-A_{T+1}$.

Questa maggiorazione ci permette di dimostrare la seguente proposizione

**Proposizione 3.1** (Nemirowski et al, 2009)**.** *Sia $f$ una funzione convessa con $L_g = 0$ e applichiamo il metodo del gradiente stocastico per $T$ passi con lunghezza di passo fissato $\alpha > 0$. Definiamo*

$$\alpha_{opt} = \frac{D_0}{B\sqrt{T+1}} \text{ e } \theta := \frac{\alpha}{\alpha_{opt}}$$

*Allora abbiamo la seguente maggiorazione*

$$\mathbb{E}[f(\bar{x}^T) - f(x^*)] \leq (\frac{1}{2}\theta + \frac{1}{2}\theta^{-1})\frac{BD_0}{\sqrt{T+1}}. \tag{3.28}$$

*Dimostrazione.* La dimostrazione segue da (3.27) ponendo $\alpha_j = \alpha = \theta\alpha_{opt} = \theta\frac{D_0}{B\sqrt{T+1}}$, per cui abbiamo che

$$\mathbb{E}\left[f(\bar{x}^T - f^*)\right] \leq \frac{D_0^2 + B^2(T+1)\alpha^2}{2(T+1)\alpha} =$$

$$= \frac{D_0^2 B}{2\sqrt{T+1}D_0\theta} + \frac{B^2 D_0 \theta}{2\sqrt{T+1}B} = \left(\frac{1}{2}\theta^{-1} + \frac{1}{2}\theta\right)\frac{BD_0}{\sqrt{T+1}}$$

□

Si ottiene la migliore maggiorazione quando $\theta = 1$ e cioè quando $\alpha = \alpha_{opt}$.

### 3.4.2 Caso 2: $B = 0$

Nel caso in cui $B = 0$, l'espressione (3.18) diventa

$$A_{k+1} \leq (1 + \alpha_k^2 L_g^2) A_k - 2\alpha_k \mathbb{E}\left[\nabla f(x^k)^T (x^k - x^*)\right]. \tag{3.29}$$

Sia $f$ fortemente convessa, con modulo di convessità $\gamma > 0$, abbiamo in generale che

$$\nabla f(x)^T (x - x^*) \geq \gamma \|x - x^*\|^2. \tag{3.30}$$

(segue dalla proprietà (1.6)).

Allora la (3.29) diventa

$$A_{k+1} \leq (1 - 2m\alpha_k + L_g^2 \alpha^2) A_k \tag{3.31}$$

Scegliamo $\alpha_k = \alpha$ costante prendendo un valore in $(0, 2m/L_g^2)$ (così il coefficiente in (3.31) rimane tra 0 e 1). Il valore di $\alpha$ che ottimizza il coefficiente qui sopra è $\alpha = m/L_g^2$. Con questa scelta abbiamo che la disuguaglianza diventa

$$A_{k+1} \leq \left(1 - \frac{m^2}{L_g^2}\right) A_k \leq \left(1 - \frac{m^2}{L_g^2}\right)^k D_0^2. \tag{3.32}$$

Con la condizione trovata possiamo calcolare il numero di iterazioni $T$ necessarie per avere un errore atteso al di sotto di una soglia $\epsilon > 0$ fissata:

$$(1 - \frac{m^2}{L_g^2})^T D_0^2 < \epsilon,$$

isolando il termine con $T$ e applicando il log naturale

$$T = \left\lceil \frac{L_g^2}{m^2} \log\left(\frac{D_0^2}{\epsilon}\right) \right\rceil.$$

#### Caso speciale: il metodo di Kaczmarz

Il risultato ottenuto con $B = 0$ può essere applicato anche al caso del metodo di Kaczmarz per problemi ai minimi quadrati sovradeterminati. In questo caso il risultato di convergenza del metodo può essere migliorato sfruttando la struttura del problema.

Richiamiamo la funzione obiettivo ai minimi quadrati vista in (3.3) e l'iterazione di Kaczmarz (3.4) della forma

$$x^{k+1} = x^k - a_{i_k}(a_{i_k}^T x^k - b_{i_k}),$$

dove $i_k$ è scelto con probabilità uniforme nell'insieme di indici, $\alpha_k = 1$ costante e $a_i^T$ è l'$i$-esima riga di $A$; $a_i$ di norma 1. $x^*$ è il punto tale che $a_i^T x^* = b_i$, $\forall i$ (non è detto sia unico!).Abbiamo che

$$\begin{aligned} \|x^{k+1} - x^*\|^2 &= \|x^k - a_{i_k}(a_{i_k}^T x^k - b_{i_k}) - x^*\|^2 \\ &= \|x^k - x^*\|^2 - 2(a_{i_k}^T x^k - b_{i_k}) a_{i_k}^T (x^k - x^*) + (a_{i_k}^T x^k - b_{i_k})^2 \end{aligned}$$

notando che $a_{i_k}^T(x^k - x^*) = a_{i_k}^T x^k - b_{i_k}$ e che quindi il termine centrale sarebbe $-2(a_{i_k}^T x^k - b_{i_k})^2$ si ha

$$\|x^{k+1} - x^*\|^2 = \|x^k - x^*\|^2 - (a_{i_k}^T x^k - b_{i_k})^2$$

Sia $\lambda_{min,nz}$ il più piccolo autovalore non zero della matrice $A^T A$. Fissiamo il punto $x^*$ che minimizza $\|x - x^*\|$ tra i punti che soddisfano $Ax^* = b$ (come detto, $x^*$ può non essere unico). Allora preso questo punto, applichiamo il valore atteso

$$\begin{aligned}
\mathbb{E}\left[\|x^{k+1} - x^*\|^2 | x^k\right] &\leq \|x^k - x^*\|^2 - \mathbb{E}_{i_k}\left[\left(a_{i_k}^T x^k - b_{i_k}\right)^2\right] \\
&= \|x^k - x^*\|^2 - \frac{1}{N}\sum_{i=1}^{N}\left(a_i^T x^k - b_i\right)^2 \\
&= \|x^k - x^*\|^2 - \frac{1}{N}\|Ax^k - b\|^2 \\
&\leq \|x^k - x^*\|^2 - \frac{1}{N}\|Ax^k - Ax^*\|^2 \\
&\leq \|x^k - x^*\|^2 - \frac{1}{N}\|A\|^2\|x^k - x^*\|^2 \\
&\leq \left(1 - \frac{\lambda_{min,nz}}{N}\right)\|x^k - x^*\|^2.
\end{aligned}$$

Nell'ultimo passaggio abbiamo usato che $\|A\|_2^2 = \rho(A^T A) \geq \lambda_{min,nz}$, con $\rho$ raggio spettrale.

Definiamo $D_k := \min_{x:Ax=b}\|x^k - x\|^2$ e, poiché $D_{k+1} \leq \|x^{k+1} - x^*\|^2$ e $D_k = \|x^k - x^*\|^2$, allora si ha

$$\mathbb{E}[D_{k+1}] \leq \mathbb{E}[\|x^{k+1} - x^*\|^2] \leq \left(1 - \frac{\lambda_{min,nz}}{N}\right)\mathbb{E}[D_k].$$

Otteniamo quindi una velocità di convergenza migliore in questo caso rispetto a (3.32).

### 3.4.3 Caso 3: $L_g$ e $B$ non nulli

Supponiamo di avere il caso generale, con la funzione $f$ fortemente convessa e $L_g$ e $B$ non nulli in (3.11). Usiamo nuovamente la condizione (3.30), da cui otteniamo

$$A_{k+1} \leq (1 - 2m\alpha_k + \alpha_k^2 L_g^2)A_k + \alpha_k^2 B^2. \tag{3.33}$$

#### Lunghezza di passo fissato

Consideriamo prima una lunghezza di passo $\alpha_k = \alpha$ costante, con $\alpha \in (0, 2m/L_g^2)$ (in questo modo, il coefficiente di $A_k$ rimane compreso tra 0 e 1). Iteriamo la condizione

(3.33) $k$ volte:

$$\begin{aligned} A_k &\leq (1-2m\alpha+\alpha^2L_g^2)A_{k-1}+\alpha^2B^2 \leq \\ &\leq (1-2m\alpha+\alpha^2L_g^2)^2A_{k-2}+(1-2m\alpha+\alpha^2L_g^2)\alpha^2B^2+\alpha^2B^2 \leq \dots \\ &\leq (1-2m\alpha+\alpha^2L_g^2)^kD_0^2+\alpha^2B^2\sum_{j=0}^{k-1}(1-2m\alpha+\alpha^2L_g^2)^j \leq \\ &\leq (1-2m\alpha+\alpha^2L_g^2)^kD_0^2+\frac{\alpha^2B^2}{1-(1-2m\alpha+\alpha^2L_g^2)} = \\ &= (1-2m\alpha+\alpha^2L_g^2)^kD_0^2+\frac{\alpha B^2}{2m+\alpha L_g^2}. \end{aligned} \tag{3.34}$$

Il termine al membro di destra non scende mai sotto la soglia $\frac{\alpha B^2}{2m+\alpha L_g^2}$, qualsiasi sia l'iterazione a cui siamo arrivati. Si può vedere questo comportamento nella pratica: l'iterazione converge ad una palla intorno al punto ottimale, con raggio della palla maggiorato da $\frac{\alpha B^2}{2m+\alpha L_g^2}$. Una volta dentro, le iterazioni rimbalzano all'interno della palla senza arrivare a $x^*$. Si può provare a ridurre il raggio riducendo il valore di $\alpha$, ma questo comportamento rimane e si perderebbe la convergenza lineare data dal primo termine elevato alla $k$ in (3.34). La quantità tra parentesi sarebbe molto vicina a 1.

Si può trovare una buona correzione a questo problema usando le *epoche* (si veda Sezione 3.5.1).

### Lunghezza di passo decrescente

Un secondo approccio è quello di considerare $\alpha_k$ decrescente, con una decrescita circa simile a $\frac{1}{k}$. Un esempio è prendere

$$\alpha_k = \frac{\gamma}{k_0+k}$$

con $\gamma$ e $k_0$ costanti da determinare (per questo sono detti *iperparametri*). Una proposizione mostra che si può ottenere un bound dell'errore della forma

$$A_k \leq \frac{Q}{k_0+k}$$

per qualche $Q$, come mostrato nella seguente proposizione.

**Proposizione 3.2.** *Sia $f$ una funzione fortemente convessa con modulo di convessità $\gamma$. Applichiamo il metodo del gradiente stocastico con*

$$\alpha_k = \frac{1}{2m(L_g^2/2m^2+k)},\ k=1,2,3\dots$$

*allora per una certa costante $c_0$ abbiamo che*

$$\mathbb{E}[|x^k-x^*\|^2] \leq \frac{c_0B^2/2m}{L_g^2/2m^2+k},\ k=0,1,2,\dots$$

*Dimostrazione.* Nella dimostrazione si procede per induzione[10]. □

## 3.5 Aspetti implementativi

Alcuni utili strumenti per implementare il metodo del gradiente stocastico.

### 3.5.1 Epoche

In una epoca vengono fatti un certo numero di iterazioni del metodo con un passo fissato e all'epoca successiva si decide se modificare o no la lunghezza di passo. Una strategia spesso utilizzata è quella di scegliere una epoca lunga $T$ iterazioni e poi ridurre $\alpha$ di un fattore $\gamma \in (0, 1)$, Si ha quindi inizialmente il passo $\alpha$ mentre dopo $k$ epoche si procede con un passo uguale a $\alpha\gamma^{k-1}$. Una scelta ragionevole di $\gamma$ può essere $\gamma \in (0.8, 0.9)$, ma la scelta di questo iperparametro e della lunghezza delle epoche rimane un punto critico nell'implementazione del metodo.

Un altro esempio popolare è quello chiamato *epoch doubling*, in cui dopo $T$ passi con $\alpha$ fissato, si continua per altri $2T$ passi e lunghezza di passo pari a $\alpha/2$. E poi ancora si raddoppia la lunghezza dell'epoca e si dimezza la lunghezza di passo.

Usando le epoche si ottiene un metodo più robusto di quello visto con la decrescita del passo ad ogni iterazione $k$ visto in precedenza.

### 3.5.2 Minibatching

Se applichiamo SG ad una funzione alle somme finite come in (3.3), spesso ad ogni passo non si fa l'aggiornamento usando un unico termine del gradiente, ma si sceglie un piccolo insieme di termini, detto *minibatch*, di cardinalità fissata, diciamo $p$. Quindi, scelto $S_k \subset \{1, 2, \ldots, N\}$, con $|S_k| = p$, l'iterazione $k$ del metodo ha la forma

$$x^{k+1} = x^k - \alpha_k \frac{1}{p} \sum_{i \in S_k} \nabla f_i(x^k).$$

Se la scelta del sottoinsieme $S_k$ avviene in maniera uniforme e i.i.d. ad ogni iterazione, allora l'analisi di convergenza vista fino a questo punto si può applicare anche in questo caso. Il vantaggio del minibatch sta nello stimare il gradiente con una varianza minore rispetto al singolo termine. Si ha quindi una migliore velocità di convergenza in valore atteso. D'altro canto, ogni iterazione risulta $p$ volte più costosa. Infatti anche $p$ rimane un iperparametro la cui scelta influenza sostanzialmente la performance pratica del metodo. Rimane comunque un approccio universalmente utilizzato quando si implementa il metodo SG.

# Capitolo 4

# Esperimenti numerici

In questo capitolo studiamo il comportamento numerico dei metodi studiati nel Capitolo 3 per il problema di regressione lineare. A tal fine, consideriamo un data set noto in letteratura (vedi Sezione 4.1) e un data set reale legato al problema di previsione di vendite studiato durante il tirocinio presso Analytics Network (vedi Sezione 4.2). Analytics Network fa parte della business unit di DataScience di Var Group insieme ad altre aziende ed in particolare è l'azienda specializzata nell'utilizzo delle tecniche di Machine Learning. Il Machine Learning è una disciplina che utilizza approcci algoritmici che permettono di modellare fenomeni anche complessi sfruttando le relazioni presenti nei dati. Grazie a queste tecniche, Analytics Network riesce a fornire *insight* utili alle aziende per veicolare azioni di business.

## 4.1 Data set esistente in letteratura: il data set Air

Abbiamo lavorato con il data set **Air**[1], in cui sono raccolte le misurazioni di concentrazione di gas inquinanti in una città italiana prese ogni ora da marzo 2004 fino a febbraio 2005. Lo abbiamo fatto usando il linguaggio Matlab e confrontiamo i risultati ottenuti con il lavoro svolto in [6] in cui lo stesso problema è stato studiato usando il linguaggio Python e un modello di reti neurali.

Come fatto anche in [1], l'obiettivo è quello di dare una previsione della concentrazione di benzene nell'aria in funzione di 7 variabili: la concentrazione di ossido di stagno, la concentrazione di ossido di titanio, la concentrazione di ossido di azoto di tungsteno, la concentrazione di diossido di azoto di tungsteno, la concentrazione di ossido di indio, la temperatura dell'ambiente e l'umidità relativa. Abbiamo quindi eliminato gli altri valori presenti nel data set.

Inoltre, nella colonna della concentrazione di benzene sono presenti dei valori pari a $-200$ che indicano che i rilevatori non hanno potuto raccogliere il dato in quella determinata ora. Abbiamo non considerato queste righe, passando quindi da un data set con 9357 esempi ad uno con 8991 righe.

[1] https://archive.ics.uci.edu/ml/datasets/Air+Quality

Per finire la preparazione del data set, abbiamo scalato i valori in modo da ottenere tutte quantità dentro l'intervallo $[0, 1]$. Sia $D$ l'insieme dei dati, $D \in \mathbb{R}^N \times \mathbb{R}^{d_x+1}$, dove con $d_x$ indichiamo il numero di predittori (cioè 7 in questo caso) mentre la colonna in più è quella delle osservazioni da predire. Per ogni colonna, calcoliamo

$$\begin{cases} m_j = \min_{i=1,\dots,N} D_{i,j} \\ M_j = \max_{i=1,\dots,N} D_{i,j} \end{cases} \tag{4.1}$$

con $j = 1, \dots, d_x + 1$ e $D_{i,j} \in D$. Otteniamo una nuova matrice dei dati che indichiamo con $A$ i cui elementi sono

$$a_{i,j} = \frac{D_{i,j} - m_j}{M_j - m_j}, \tag{4.2}$$

per ogni $i = 1, \dots, N$ e $j = 1, \dots, d_x + 1$.

Dividiamo infine il data set in una parte di *training* e una di *test*. Per la parte di training abbiamo preso il 70% degli esempi. In particolare, i primi nove mesi dello studio sono stati finalizzati all'addestramento del modello, i restanti 3 mesi alla fase di testing.

**Applicazione del metodo di Kaczmarz stocastico**

Abbiamo implementato in una funzione Matlab il metodo di Kaczmarz (Sezione 3.2), dati in input il punto di partenza $x^0$, il data set $A$, il vettore delle reali concentrazioni di benzene $b$ e il numero di epoche computazionali $EGE$.

La funzione crea un ciclo che si interrompe quando viene raggiunto il numero massimo di epoche attraverso un contatore e, ad ogni ciclo, applica l'aggiornamento secondo il metodo di Kaczmarz: per prima cosa estrae casualmente una riga dalle $ntrain$ righe della matrice di allenamento[2], con $N = ntrain + ntest$ ed $N$ numero totale di righe di A, poi calcola l'approssimazione stocastica del gradiente come derivata della funzione quadratica associata all'indice della riga estratta ($g = (a_i^T x - b_i)a_i$) ed infine modifica il vettore indicato con $x$ aggiungendo il gradiente stocastico calcolato in precedenza diviso per la norma della riga della matrice al quadrato, visto che il metodo richiede che le righe siano di norma unitaria.

Il ciclo si chiude calcolando la norma del gradiente e il valore della funzione obiettivo nei punti trovati. Viene aggiornato il contatore *ngrad* che tiene il conto di quanta parte di epoche sono state calcolate fino ad ora (ad ogni aggiornamento si aggiunge una quantità pari a $1/ntrain$). Insieme all'iterazione corrente, queste tre variabili vengono salvate in una riga della matrice $M$, uno dei due output della funzione. Finito il ciclo, il vettore $x$ finale è il vettore dei parametri cercati.

Lo pseudocodice è presentato nell'Algoritmo 1.

Il metodo di Kaczmarz, essendo stocastico, ad ogni iterazione restituisce valori di parametri diversi e, di conseguenza, previsioni diverse. Per questo motivo, abbiamo eseguito il codice 10 volte e preso la media delle previsioni per avere una stima che

[2]Questa procedura è stata realizzata in Matlab con il comando *readsample*, il quale dati in input due numeri $n$ e $k$, restituisce $k$ numeri estratti tra i valori 1 e $n$ in maniera casuale, con probabilità uniforme e senza reinserimento.

**Algorithm 1** Il metodo di Kaczmarz

**Require:** $x \in \mathbb{R}^n,\ A \in \mathbb{R}^{ntrain \times n},\ b \in \mathbb{R}^{ntrain},\ EGE > 0$
**Ensure:** $x\_sol,\ M$
  $k = 0$,
  **while** $ngrad < EGE$ **do**
    $k = k + 1$,
    Scegli un indice $i$ in $\{1, \ldots, ntrain\}$ in maniera randomica (distribuzione uniforme).
    Seleziona dalla matrice $A$ la riga $a_i^T$,
    Calcola il gradiente stocastico: $g = \frac{a_i}{\|a_i\|^2}(a_i^T x - b_i)$,
    $ngrad = ngrad + 1/ntrain$,
    $x = x - g$,
    $M(k) = [k\ ngrad\ \|g\|\ \frac{1}{2ntrain}\|Ax - b\|_2^2]$,
  **end while**
  $x\_sol = x$

dipenda di meno dalla singola iterazione eseguita. Abbiamo preso come punto di partenza l'origine $x^0 = 0$ e come valore di epoche 0.7 (e quindi il metodo costa un totale di 0.7 volte il calcolo del gradiente della funzione obiettivo).

Abbiamo confrontato i risultati ottenuti con quelli studiati in [6], in cui lo stesso problema è stato studiato attraverso un modello di reti neurali con due livelli nascosti. Nel primo livello sono presenti 7 neuroni mentre nel secondo 5. Una funzione lineare e la funzione sigmoide sono state scelte come funzioni di attivazione della rete. Nelle Figure 4.1 e 4.2 si possono apprezzare i risultati ottenuti con i due modelli.

Poiché si tratta di un modello non lineare, ci aspettavamo risultati migliori con il modello che usa le reti neurali (Figura 4.1). Di fatto, però, la previsione ottenuta attraverso il metodo di Kaczmarz per la regressione lineare (Figura 4.2) si avvicina molto e per questo l'abbiamo trovata soddisfacente.

Inoltre le Figure 4.3 e 4.4 mostrano l'errore relativo basso ottenuto sia dal modello con le reti neurali, sia da quello lineare, con un valore di errore relativo sempre al di sotto dello 0.05, con il primo modello che presenta un errore migliore poiché risulta più potente, come detto in precedenza. Nelle quattro figure si può notare un picco molto alto di errore, quello relativo al decimo giorno, che risulta anche l'errore più grande in valore assoluto di tutta la serie ed è più difficile da prevedere.

## 4.2 Problema affrontato durante il tirocinio: previsione di vendite

### 4.2.1 Presentazione del problema

Durante il tirocinio presso Analytics Network è stato studiato un data set che contiene i dati di vendita di farmaci per una azienda cliente. Questa azienda si occupa di raccogliere le richieste di migliaia di farmacie in Italia e di rispondere giorno per giorno alla domanda.

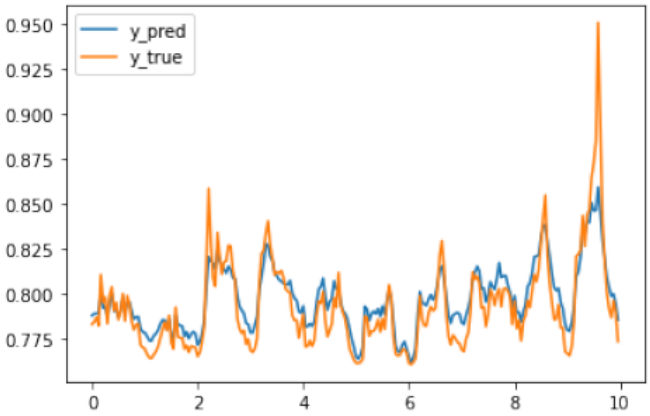



Figura 4.1: Andamento reale (giallo) e andamento previsto (blu) della concentrazione di Benzene nei primi 10 giorni di studio, modello con reti neurali. Figura tratta dalla tesi [6]

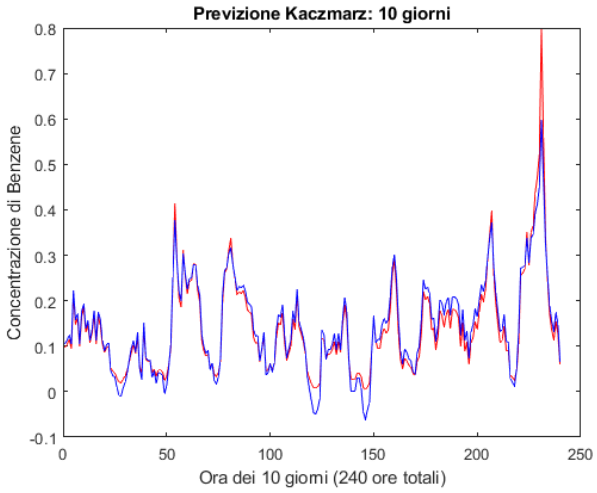



Figura 4.2: Andamento reale (rosso) e andamento previsto (blu) della concentrazione di Benzene nei primi 10 giorni di studio, metodo di Kaczmarz per regressione lineare. Valore in ascissa misurato in ore.

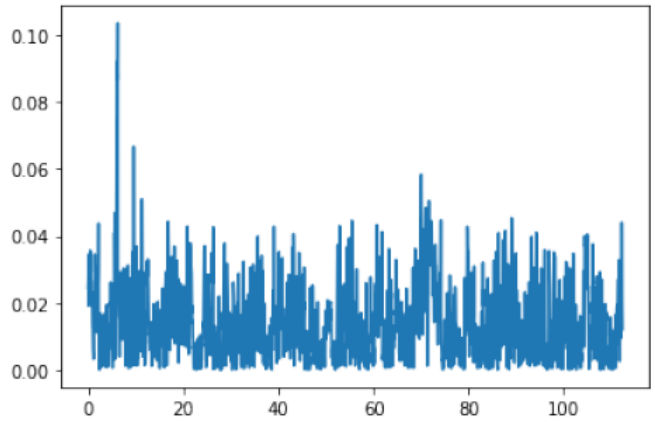



Figura 4.3: Errore relativo sui 112 giorni di studio, metodo delle reti neurali. Figura tratta dalla tesi [6]

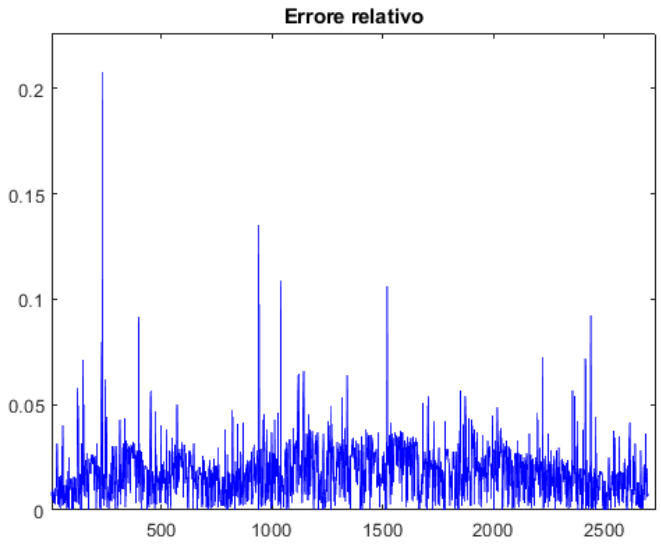



Figura 4.4: Errore relativo sui 112 giorni di studio, metodo di Kaczmarz. Valore in ascissa misurato in ore.

Fino ad ora la Signora Carla[3] (dipendente dell'azienda) si è occupata di questo segnando ogni volta con carta e penna gli ordini ricevuti e avendo un quadro totale a fine giornata per chiedere le quantità corrette il giorno seguente.

L'azienda cliente si è messa in contatto con Analytics Network per vedere se ci fosse un modo per rendere automatico questo processo e trovare una soluzione che potesse ottimizzare i costi.

L'obiettivo è quindi cercare un modello di previsione che possa predire in quale quantità ogni articolo farmaceutico verrà venduto nel futuro prossimo in modo da rispondere alla domanda del mercato, facendo però attenzione a limitare i pezzi in eccesso i quali portano a perdite, per due motivi:

- i pezzi invenduti vengono stoccati in magazzino con un certo costo;
- i prodotti che rimangono in magazzino troppo a lungo rischiano di arrivare a data di scadenza e rimanere invenduti, per una spesa quindi che non porta guadagno.

Abbiamo affrontato il problema usando un modello semplice di previsione quale la *regressione lineare* attraverso un approccio statistico e utilizzando il linguaggio Python.

### 4.2.2 Dati forniti dall'azienda e preparazione del data set

L'azienda cliente ci ha messo a disposizione alcuni insiemi di dati per lo studio. Questi riguardano le vendite di più di seicento articoli farmaceutici nel periodo che va dal primo gennaio 2020 al 31 dicembre 2022 (quindi un periodo lungo tre anni).

Il primo passo è stato prendere questi dati e riordinarli per lo studio che volevamo farne. Questa fase è detta *Data preparation*. Innanzitutto, i dati sono stati importati in Python e salvati in strutture dataframe con l'ausilio dell'estensione *Pandas*. I *dataframe* sono data structure simili a tabelle con doppie etichette che si possono riempire con diversi tipi di dati (valori, stringhe, date, ecc) e che sono comode perché permettono varie operazioni tra righe e colonne (ad esempio, somme tra righe, selezione, ecc).

Dopo averle salvate, c'è stato bisogno di correggere l'etichetta del *'Codice Articolo'*. Questo perché nell'arco di tre anni, il mercato di ogni articolo può cambiare, ad esempio alcuni possono entrare in commercio o esserne tolti, altri essere sostituiti da nuovi prodotti. Vorremmo quindi che se due o più codici rappresentano la stessa richiesta, vengano salvati, e quindi studiati, sotto un unico codice articolo (in particolare, il codice ora in uso). Questo processo è detto *Phase in - Phase out* e l'abbiamo sviluppato sfruttando la struttura dizionario di Python: abbiamo posto come chiavi il codice dell'articolo sostituito e come relativo valore il codice dell'articolo che ne ha preso il posto. Infine, il comando *.map()* ha applicato in automatico la sostituzione seguendo il dizionario.

Step successivo, i dati di vendita e giacenza sono salvati con cadenza giornaliera. Abbiamo pensato di ottimizzare gli ordini considerando intervalli di tempo con frequenza di due settimane. Ci è sembrata una buona scelta per provare a ridurre il numero di nuovi ordini da fare ma comunque non avere periodi troppo lunghi e difficili da prevedere.

[3]Nome di fantasia

Abbiamo posto il sabato come primo giorno delle due settimane perché era il giorno in cui non venivano fatte richieste visto che l'azienda la domenica rimane chiusa. Quindi ogni intervallo bisettimanale lo indichiamo con il sabato con cui inizia.

### 4.2.3 Feature engineering

Con quanto detto fino ad ora, abbiamo costruito un dataframe con al suo interno il numero di pezzi che sono stati venduti per ogni codice articolo e per ogni bisettimana degli anni 2020-2021-2022.

Prima di iniziare con il modello matematico, abbiamo standardizzato i dati, poiché ogni articolo è caratterizzato da grandezze di ordini diversi. Ad esempio, alcuni possono essere richiesti in quantità superiori alle 200 unità ogni due settimane, mentre altri possono non raggiungere le 5 unità lungo un uguale intervallo. La standardizzazione permette di lavorare con questi dati contemporaneamente scalandone i valori rispetto ad un riferimento scelto. Altrimenti l'andamento di alcuni codici articolo influirebbe in malo modo sugli altri. Abbiamo scelto come riferimento la media delle vendite bisettimanali per ogni articolo. Una prima idea era di usare la mediana, cioè il 50° percentile della storia di un articolo, ma in alcuni casi la mediana era nulla e questo impediva la standardizzazione. La mediana risultava nulla quando più della metà delle bisettimane dell'articolo erano con 0 vendite, quindi nei casi in cui o l'articolo da un certo punto in poi non era più in vendita, o era stato appena messo in commercio o soffriva di forte stagionalità ed era quindi venduto solo in determinati periodi.

A questo punto, abbiamo scelto e costruito i predittori utili a creare il modello:

- $Q_{t-1}$, cioè gli ordini ricevuti nella bisettimana precedente;
- $MM2$, quanti ordini ci sono stati nelle due bisettimane precedenti, cioè nell'ultimo mese;
- $MM3$, $MM4$, $MM6$, analoghi al punto precedente, quindi indicano rispettivamente un mese e mezzo, due mesi, tre mesi precendenti il periodo attuale;
- *trend breve*, calcolato come rapporto tra i pezzi venduti nel periodo precedente e i pezzi venduti nel mese e mezzo precedente (in simboli $\frac{Q_{t-1}}{MM3}$);
- *trend lungo*, calcolato come rapporto tra i pezzi venduti nel mese precedente e i pezzi venduti nei tre mesi precedenti (in simboli $\frac{MM2}{MM6}$);
- *cluster*, come si può vedere nei grafici 4.5 e 4.6, raggruppa i codici articolo in 4 insiemi in base a volatilità e densità di zeri: smooth, intermittent, erratic, lumpy;
- *stagionalità*, poiché alcuni articoli hanno un maggiore o minore numero di vendite in base alla stagione in cui siamo (ad esempio, i farmaci per l'influenza hanno maggiori vendite d'inverno e meno d'estate, mentre le creme solari viceversa).

Nota: La variabile di *stagionalità* è stata costruita assegnando ad ogni articolo e ad ogni bisettimana il rapporto tra la somma delle vendite di quell'articolo nella stagione

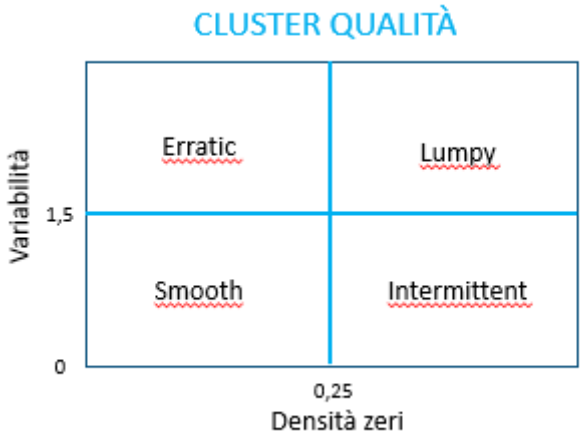



Figura 4.5: Scelta della suddivisione in cluster dei codici articolo. Ad esempio, un articolo *lumpy* presenta una alta variabilità e una alta densità di zeri.

in cui siamo diviso la somma delle vendite di tutto l'anno (riferito all'anno 2020). Abbiamo usato il primo anno di dati per costruire questo valore per inserirlo nel dataframe riguardante i successivi due anni.

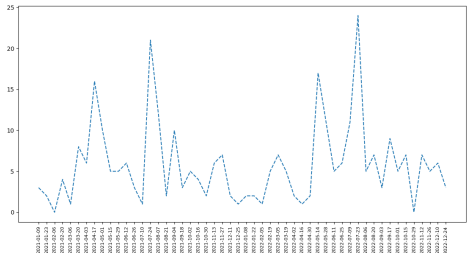
Una prima idea era stata quella di considerare semplicemente una variabile categorica che indicasse in che stagione fosse la bisettimana di riferimento. In questa maniera però avremmo avuto una condizione di stagionalità unica per tutti i codici articolo. Perciò, un prodotto che si vende principalmente in estate e uno che si vende principalmente in inverno avrebbero addestrato il modello in senso opposto, "annullandosi".

Un'altra variabile che inizialmente volevamo considerare era il periodo di Lockdown avvenuto tra marzo e maggio 2020. Ma a causa della costruzione della variabile di stagionalità, il primo anno di dati non fa più parte dell'insieme di addestramento e non abbiamo quindi utilizzato questa informazione.

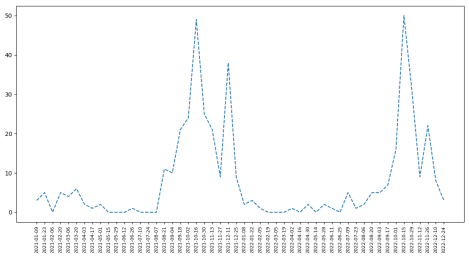
### 4.2.4 Modellizzazione

Scelti i predittori, iniziamo a definire il modello. Abbiamo scelto di lavorare con un modello di regressione lineare, come quello presentato nel problema (1.12), per la sua semplicità e perché è un buon modello per iniziare lo studio di un problema. Lo abbiamo fatto attraverso i metodi implementati nel pacchetto *sklearn* (conosciuto anche come *Scikit-learn*) del linguaggio Python, di cui ora descriviamo alcuni comandi.

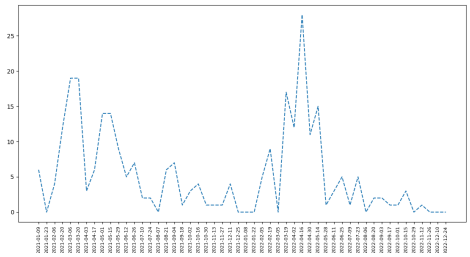
Per prima cosa, dividiamo i dati in due gruppi, uno per costruire il modello (*fase di training*), il secondo per vedere se il modello è valido (*fase di test*). Nell'insieme di testing abbiamo considerato le bisettimane che fanno parte di un mese estivo, un mese autunnale ed uno invernale, tutti del 2022, per avere quindi circa tre mesi di dati abbastanza variegati. In totale risultano 8 bisettimane per articolo. I restanti dati riguardanti le altre 44 bisettimane hanno composto l'insieme di training.



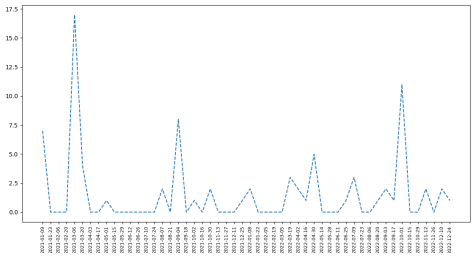
(a) *Serie smooth.*



(b) Serie lumpy.

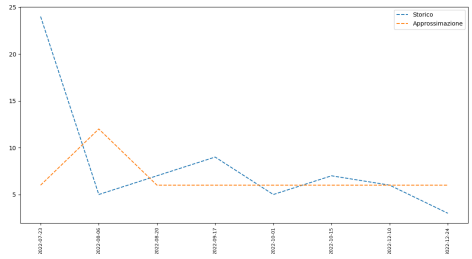


(c) *Serie erratic.*

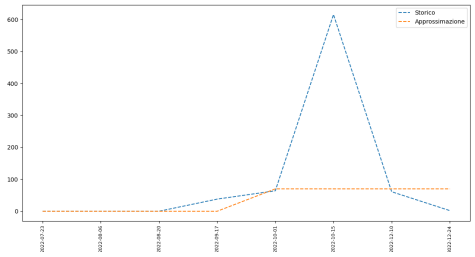


(d) Serie intermittent

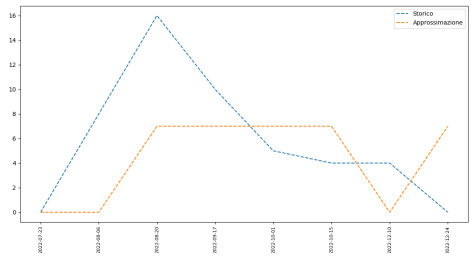
Figura 4.6: Per ogni cluster, un esempio dell'andamento delle vendite del prodotto lungo gli anni 2021 e 2022. Se ne riconoscono alcune caratteristiche, ad esempio nella serie intermittent si notano i tanti zeri che intervallano l'andamento delle vendite, oppure le serie lumpy e erratic hanno grandi salite e discese dovute all'alta variabilità.



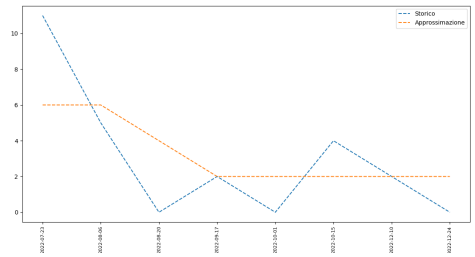

(a) *Serie smooth.*




(b) Serie lumpy.




(c) *Serie erratic.*




(d) Serie intermittent

Figura 4.7: Confronto tra la serie storica (blu) e l'approssimazione trovata (giallo) per ogni tipo di cluster. Sull'asse delle ascisse le bisettimane che fanno parte dell'insieme di test. Le previsioni seguono in parte l'andamento dell'articolo, a volte riuscendo anche ad avere errore nullo. I picchi elevati (come quelli presenti nelle serie erratic e lumpy) rimangono molto difficili da prevedere.

Il comando "$sm.OLS$"[4] prende in input la matrice dei dati di training e il vettori con i relativi valori reali di vendita e costruisce il modello calcolando i coefficienti dell'iperpiano che minimizza l'errore. Il comando "$.fit()$" permette di applicare il modello ai dati di testing che abbiamo. Infine, il comando "$.predict()$" fa agire il modello contro le variabili di test e permette di trovare una previsione di vendite per quelle bisettimane appartenenti all'insieme di test.

Dopo aver corretto la standardizzazione dei dati per riportarli sulla unità di misura di partenza, possiamo confrontare la previsione trovata con i dati veri che abbiamo a disposizione.

Nella Figura 4.7 sono mostrati questi confronti tra serie storica e approssimazione calcolata per un codice articolo di ogni cluster.

[4]Non è stata trovata una documentazione approfondita riguardo il comando *sm.OLS*, che quindi viene usato come una black box.

### 4.2.5 Misure di errore e diagnostica

Per misurare l'errore commesso dal modello abbiamo usato due strumenti statistici come le *misure di errore* $MAE$ e $WAPE$.

**Definizione 4.1.** Dati $X$ e $Y$ osservazioni dello stesso fenomeno, di cui $Y$ rappresenta il valore reale del fenomeno mentre $X$ la previsione usando un nostro modello. Indicate con $x_i$ e $y_i$ le rispettive osservazioni, si definisce $MAE$ (*Mean Absolute Error*) la quantità

$$MAE(X,Y) = \frac{\sum_{i=1}^{n} |x_i - y_i|}{n} \tag{4.3}$$

e misura l'errore che c'è tra la previsione fatta rispetto ai valori reali. Questa misura di errore ci indica in media di quanti pezzi in valore assoluto abbiamo sbagliato con il nostro modello rispetto ai veri dati.

**Definizione 4.2.** Si definisce, invece, $WAPE$ (*Weighted Average Percentage Error*) la quantità

$$WAPE(X,Y) = \frac{\sum_{i=1}^{n} |x_i - y_i|}{\sum_{i=1}^{n} |y_i|} \tag{4.4}$$

che si tratta di una media pesata rispetto all'osservazione vera del valore assoluto dell'errore commesso. Il *WAPE* ci indica la percentuale di errore che abbiamo commesso sulle previsioni. Ad esempio, se il valore risulta uguale a 0.3, vuol dire che mediamente commettiamo un errore del 30% sulle previsioni.

Osservare insieme MAE e WAPE è utile nei casi in cui gli articoli hanno vendite medio basse. Se considerassimo ad esempio un articolo che mediamente vende 1 unità e in una previsione diciamo che vende 2, l'errore risulterebbe del 100% ma si avrebbe un errore assoluto di 1.

Applichiamo queste formule nel nostro caso e vediamo che

$$MAE = 2.78, \quad WAPE = 1.05,$$

cioè in media l'errore commesso è tra i 2 e i 3 pezzi e sbagliamo di circa il 105% sulle previsioni. L'approssimazione non è molto buona, soprattutto considerando che molti articoli vengono venduti nell'ordine delle 3/4 unità al massimo ogni due settimane, in particolare quelli che fanno parte del cluster smooth o intermittent (alcuni articoli smooth fanno eccezione con picchi molto alti).

Abbiamo quindi provato alcune considerazioni alternative per migliorare la precisione, sfruttando la divisione nei cluster. Si possono seguire i risultati con le Tabelle 4.1 e 4.2.

- Per prima cosa abbiamo osservato $MAE$ e $WAPE$ degli articoli in ogni cluster per vedere se si ha cattiva precisione in tutti i casi: si vede subito che con gli articoli lumpy si ha una pessima previsione. Ci sono meno errori con gli articoli smooth e erratic ma gli errori sono molto grandi a causa di alcuni articoli particolarmente richiesti. Infine, gli articoli intermittent vengono sbagliati il 27% delle volte con errore basso, ma questo tipo di articoli tendono ad avere poche richieste quindi sono errori relativi importanti.

| | $MAE$ col modello completo | $MAE$ col modello separato | $MAE$ del 95° percentile |
|---|---|---|---|
| smooth | 17.20 | 2.73 | 1.62 |
| intermittent | 2.16 | 1.19 | 0.82 |
| erratic | 116.1 | 4.66 | 3.42 |
| lumpy | 111.58 | 3.23 | 1.06 |

Tabella 4.1: MAE dei modelli costruiti sui singoli cluster nei tre casi visti: modello generale ristretto ai cluster, modelli separati secondo i cluster, modello costruito escludendo gli errori più grandi.

| | $WAPE$ col modello completo | $WAPE$ col modello separato | $WAPE$ del 95° percentile |
|---|---|---|---|
| smooth | 0.14 | 1.02 | 0.69 |
| intermittent | 0.48 | 1.02 | 0.92 |
| erratic | 0.04 | 1.05 | 0.98 |
| lumpy | 0.03 | 1.00 | 0.96 |

Tabella 4.2: WAPE dei modelli costruiti sui singoli cluster nei tre casi visti: modello generale ristretto ai cluster, modelli separati secondo i cluster, modello costruito escludendo gli errori più grandi.

- Il secondo tentativo è stato quello di creare un modello di regressione lineare separatamente per ogni cluster seguendo lo schema fatto in precedenza. Si hanno così quattro modelli in teoria più adatti ai singoli cluster. Dalle misure presentate si può notare un migliore errore assoluto (gli errori medi di tutte le tipologie sono sotto le 5 unità), ma c'è un errore relativo più grande (gli articoli erratic ad esempio passano da un $WAPE$ bassissimo al 4% ad uno alto al 105%). Questo è accettabile quando si guardano articoli con vendite molto basse e su un periodo di tempo lungo come quello delle due settimane che abbiamo utilizzato.
- Infine, abbiamo provato a vedere se le previsioni sono state abbastanza buone almeno per una parte dei dati, tenendo fuori dal conto gli errori più grandi. Abbiamo escluso i dati che avevano errore sopra al 95° percentile, arrendendoci al fatto che alcuni casi non possono essere previsti. Ricalcolato $MAE$ e $WAPE$ dei valori di test fino al 95° percentile, si vede che il $WAPE$ ha un buon miglioramento per tutti i cluster (circa di dieci punti percentuali per ognuno) e il $MAE$ diminuisce in tutti i casi, in particolare per i cluster con alta volatilità (erratic e lumpy).

## 4.3 Metodo di Kaczmarz per il problema di previsione vendite

Abbiamo provato ad applicare il metodo di Kaczmarz sul data set che abbiamo ottenuto alla fine della data preparation vista nella Sezione 4.2.2 e in questa sezione ne riportiamo i risultati ottenuti in ambiente Matlab.

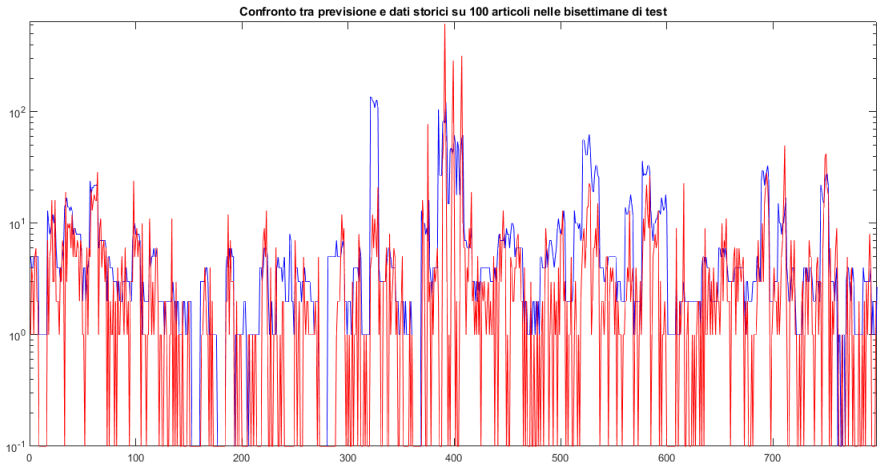



Figura 4.8: Previsione secondo Kaczmarz (in blu) e serie storica (in rosso) di circa 100 articoli sulle 8 bisettimane di testing. Grafico con ordinata in scala logaritmica, per questo gli zeri sono indicati con il valore di $10^{-1}$. In alcuni punti il modello riesce a seguire la serie storica.

I parametri in input dell'Algoritmo 1 che abbiamo scelto sono stati: $x^0 = 0$, $A$ e $b$ riguardanti il data set detto in precedenza, $EGE = 0.7$.

La previsione ottenuta dalla media delle 10 previsioni compiute risulta con le seguenti metriche di errore

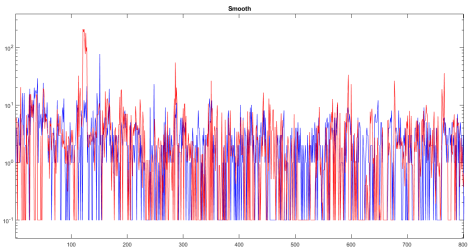
$$MAE = 2.66,\ WAPE = 1.00.$$

L'approssimazione rimane non molto buona, ma leggermente migliore rispetto a quanto visto con la regressione lineare in ambiente Python.

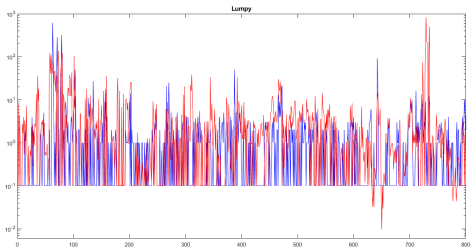
La Figura 4.8 riporta il confronto tra la previsione ed i valori veri di circa 160 articoli sulle 8 bisettimane appartenenti al data set di testing. Si possono notare alcuni errori molto grandi in valore assoluto, soprattutto tra i primi e gli ultimi valori mostrati. Nel resto dell'andamento la previsione segue abbastanza bene la serie storica e il numero di pezzi venduti e previsti risultano identici circa nel 22% dei casi.

Come fatto anche nella Sezione 4.2.5, guardiamo anche l'errore ottenuto nei singoli cluster per vedere quali tipi di articoli danno maggiori problemi. A seguire abbiamo applicato il metodo di Kaczmarz dividendo il data set in quattro insiemi più piccoli secondo la divisione in cluster e costruendo un modello per ogni caso. In questo caso, il metodo è stato eseguito con una epoca computazionale leggermente minore ($EGE = 0.5$) poiché il metodo raggiunge convergenza molto velocemente. Nelle Tabelle 4.3 e 4.4 sono raccolti gli errori sui cluster su entrambi i modelli. Confrontiamo i risultati ottenuti con quelli della ottenuti nella Sezione 4.2.5.

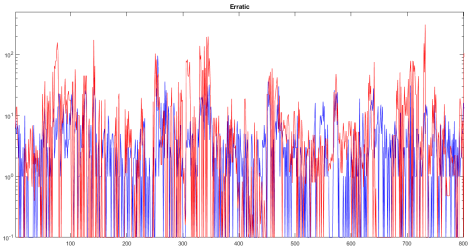
Possiamo notare come i modelli separati in questo caso funzionino molto meno rispetto a quanto visto con la regressione lineare della Sezione 4.2.4, in cui si vedeva un



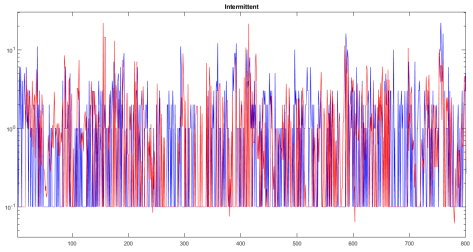

(a) *Serie smooth.*




(b) Serie lumpy.




(c) *Serie erratic.*




(d) Serie intermittent

Figura 4.9: Confronto tra la previsione ottenuta attraverso il metodo di Kaczmarz (blu) e la serie reale dei dati (rosso). Si vede bene la difficoltà nell'avere buone previsioni.

leggero miglioramento almeno nell'errore in valore assoluto. Invece il WAPE peggiora sia con la regressione calcolata con il metodo di Kaczmarz che con la regressione lineare ottenuta precedentemente.

Le serie *intermittent* risultano quelle con valori di MAE più bassi ma, allo stesso tempo, con valori di WAPE più alti, cioè sbagliamo sempre la previsione su questi articoli, ma sempre di pochi pezzi. Questi articoli hanno una alta densità di zeri, che non aiuta la previsione. Probabilmente il fattore di stagionalità da noi costruito avrebbe bisogno di più dati per essere addestrato meglio.

Nella Figura 4.9 sono riportate le previsioni fatte con i modelli costruiti per ogni cluster su tutti gli elementi dell'insieme di test relativo al singolo cluster. Già dai grafici si nota una qualità della previsione non molto alta.

Infine, anche in questo caso abbiamo calcolato i valori di MAE e WAPE che si ottengono escludendo il 5% degli errori più grandi, anche questi valori sono presenti nelle Tabelle 4.3 e 4.4. Il calcolo di MAE e WAPE considerando gli errori al di sotto del 95° percentile risultano nettamente migliori rispetto al caso di regressione lineare della Sezione 4.2.4, in particolare con le serie *erratic* e *lumpy*. Questo dà valore al modello da noi costruito perché vuol dire che una parte dei casi riusciamo a prevederli, ma non è ancora un risultato soddisfacente.

Ci aspettiamo risultati migliori usando il metodo del gradiente stocastico dove si può agire sulla scelta del passo e del minibatch (vedi Sezione 3.5.2) per migliorare la

| *Kaczmarz* | $MAE$ dal modello completo | $MAE$ col modello separato | $MAE$ del 95° percentile |
|---|---|---|---|
| smooth | 2.12 (17.20) | 3.73 (2.73) | 1.66 (1.62) |
| intermittent | 1.28 (2.16) | 1.49 (1.19) | 1.00 (0.82) |
| erratic | 4.46 (116.1) | 10.48 (4.66) | 1.98 (3.42) |
| lumpy | 3.96 (111.58) | 7.98 (3.23) | 1.16 (1.06) |

Tabella 4.3: MAE del modello ottenuto con il metodo di Kaczmarz rispetto ai singoli cluster nei tre possibili casi. Tra parentesi i valori calcolati con la regressione lineare della Tabella 4.1.

| *Kaczmarz* | $WAPE$ dal modello completo | $WAPE$ col modello separato | $WAPE$ del 95° percentile |
|---|---|---|---|
| smooth | 0.60 (0.14) | 1.06 (1.02) | 0.62 (0.69) |
| intermittent | 1.18 (0.48) | 1.37 (1.02) | 1.01 (0.92) |
| erratic | 0.39 (0.04) | 0.93 (1.05) | 0.34 (0.98) |
| lumpy | 0.58 (0.03) | 1.17 (1.00) | 0.31 (0.96) |

Tabella 4.4: WAPE del modello ottenuto con il metodo di Kaczmarz rispetto ai singoli cluster nei tre possibili casi. Tra parentesi i valori calcolati con la regressione lineare della Tabella 4.2.

precisione, mantenendo basso il costo computazionale.

## 4.4 Commenti conclusivi

In conclusione, in questo elaborato abbiamo presentato i principali esempi di problemi di ottimizzazione che nascono nel campo dell'analisi dei dati e poi ne abbiamo studiato un caso particolare, quello dei problemi ai minimi quadrati.

L'esempio di cui ci siamo occupati è motivato dall'esperienza di tirocinio effettuato presso l'azienda Analytics Network che riguardava lo studio delle vendite di un'azienda cliente. Abbiamo costruito un modello di regressione lineare che abbiamo ottimizzato usando la libreria Python *sklearn*. Le previsioni ottenute attraverso questo approccio non sono state soddisfacenti, nonostante alcune modifiche apportate al modello seguendo la divisione in cluster data dalle caratteristiche degli articoli studiati. Come visto nelle Tabelle 4.3 e 4.4, l'errore risulta consistente.

Abbiamo tentato di cambiare qualcosa nell'approccio provando ad ottimizzare il modello attraverso il metodo di ottimizzazione di Kaczmarz. Il metodo costruisce una successione di iterazioni che approssimano il minimo della funzione obiettivo. Si tratta di un metodo del gradiente stocastico e ha un basso costo computazionale per iterazione. La previsione ottenuta seguendo questa strada è risultata migliore della precedente guardando le misure di errore ottenute, ma non abbastanza da dare una previsione adeguata sull'andamento dei dati.

Si potrebbe avere un miglioramento del modello aumentando i dati a disposizione, soprattutto perché una parte dei dati è stata usata per la costruzione di uno dei predittori

della regressione lineare. Infine, poiché la regressione lineare è uno dei modelli più semplici nel campo dell'analisi dati, una ulteriore strada per migliorare i risultati potrebbe essere quella di usare dei modelli più sofisticati (come i modelli con reti neurali).

# Bibliografia


[1] S. De Vito et al. «On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario». In: *Sensors and Actuators B: Chemical* 129.2 (2008), pp. 750–757. ISSN: 0925-4005. DOI: https://doi.org/10.1016/j.snb.2007.09.060. URL: https://www.sciencedirect.com/science/article/pii/S0925400507007691.

[2] G. T. Herman. *Fundamentals of Computerized Tomography*. Springer, 2009. DOI: https://doi.org/10.1007/978-1-84628-723-7.

[3] G. N. Hounsfield. «Computerized transverse axial scanning (tomography)». In: *The British journal of radiology vol. 46,552* (1973). DOI: 10.1259/0007-1285-46-552-1016.

[4] G. James et al. *An Introduction to Statistical Learning*. Springer, 2021.

[5] H. Stark K.M.Sezan. «Applications of convex projection theory to image recovery in tomography and related areas». In: *Acad. Press, San Diego* (1987).

[6] E. Nesterini. «Metodi del primo ordine stocastici per problemi di ottimizzazione in machine learning». In: (2020). Università di Firenze.

[7] A. Pascucci. *Teoria della Probabilità*. Springer Milano, 2020. ISBN: 978-88-470-4000-7. DOI: https://doi.org/10.1007/978-88-470-4000-7.

[8] T. Strohmer e R. Vershynin. «A randomized Kaczmarz algorithm with exponential convergence». In: (2007). arXiv: math/0702226 [math.NA].

[9] S. J. Wright. «Optimization algorithms for data analysis». In: *The Mathematics of Data* 25 (2018), p. 49.

[10] S. J. Wright. *Optimization for Data Analysis*. Cambridge University Press, 2022.