UNIVERSITÀ DEGLI STUDI DI BERGAMO

Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Control Automation Lab

IDENTIFICAZIONE DEI MODELLI E ANALISI DEI DATI (IMAD)

Corso di Laurea Magistrale in INGEGNERIA INFORMATICA

Lezione 3: Regressione lineare

SPEAKER
Prof. Mirko Mazzoleni

PLACE
Università degli Studi di Bergamo

Syllabus

Parte I: sistemi statici

1. **Richiami di statistica**
2. **Teoria della stima**

   2.1 Proprietà degli stimatori

3. **Stima a minimi quadrati**

   3.1 Stima di modelli lineari

   3.2 Algoritmo del gradient descent

4. **Stima a massima verosimiglianza**

   4.1 Proprietà della stima

   4.2 Stima di modelli lineari

5. **Regressione logistica**

   5.1 Stima di un modello di regressione logistica

6. **Fondamenti di machine learning**

   6.1 Bias-Variance tradeoff

   6.2 Overfitting

   6.3 Regolarizzazione

   6.4 Validazione

7. **Cenni di stima Bayesiana**

   7.1 Probabilità congiunte, marginali e condizionate

   7.2 Connessione con Filtro di Kalman

IMAD

Parte I: sistemi statici

Stima parametrica $\hat{\theta}$

- **θ deterministico**
  - ***NO assunzioni su ddp dei dati***
    - ✓ Stima parametri popolazione
    - ✓ Stima modello lineare: minimi quadrati
  - ***SI assunzioni su ddp dei dati***
    - ✓ Stima massima verosimiglianza parametri popolazione
    - ✓ Stima modello lineare: massima verosimiglianza
    - ✓ Regressione logistica
- **θ variabile casuale**
  - ***SI assunzioni su ddp dei dati***
    - ✓ Stima Bayesiana

Machine learning

Parte II: sistemi dinamici

Stima parametrica $\hat{\theta}$

- **θ deterministico**
  - ***NO assunzioni su ddp dei dati***
    - ✓ Modelli lineari di pss
    - ✓ Predizione
    - ✓ Identificazione
    - ✓ Persistente eccitazione
    - ✓ Analisi asintotica metodi PEM
    - ✓ Analisi incertezza stima (numero dati finito)
    - ✓ Valutazione del modello

Outline

1. Stima a minimi quadrati
2. Funzione di costo
3. Gradient descent
4. Proprietà dello stimatore a minimi quadrati
5. Esercizi con codice

Outline

1. **Stima a minimi quadrati**
2. Funzione di costo
3. Gradient descent
4. Proprietà dello stimatore a minimi quadrati
5. Esercizi con codice

Stima a minimi quadrati (least squares)

Abbiamo finora descritto i dati $\mathcal{D} = \{y(1), y(2), \dots, y(N)\}$ in termini della loro media e varianza, dando degli stimatori per queste quantità

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} y(i) \qquad\qquad S_{N-1}^2 = \frac{1}{N-1} \cdot \sum_{i=1}^{N} (y(i) - \hat{\mu})^2$$

Supponiamo ora di **voler descrivere** (cioè, assumiamo che i dati abbiamo questa struttura) i dati tramite una **relazione lineare**

$$y(i) = \theta_0 + \theta_1 \varphi_1(i) + \cdots + \theta_{d-1}\varphi_{d-1}(i)$$

Stima a minimi quadrati (least squares)

Obiettivo: Supponiamo di avere a disposizione N dati $\mathcal{D} = \{(\boldsymbol{\varphi}(1), y(1)), \dots, (\boldsymbol{\varphi}(N), y(N))\}$.

Trovare la relazione tra le variabili di input (regressori, features) $\boldsymbol{\varphi} \in \mathbb{R}^{(d-1)\times 1}$ e una variabile di output $y \in \mathbb{R}$, usando un **modello lineare**

$$y(i) = \theta_0 + \theta_1 \varphi_1(i) + \cdots + \theta_{d-1}\varphi_{d-1}(i) + \epsilon(i) = \sum_{j=0}^{d-1} \theta_j\, \varphi_j(i) + \epsilon(i)$$

$$= \underset{1\times d}{\boldsymbol{\varphi}^\top(i)}\, \underset{d\times 1}{\boldsymbol{\theta}} + \underset{1\times 1}{\epsilon(i)}$$

$$[\;\cdots\;] \quad \begin{bmatrix} \vdots \end{bmatrix}$$

i-esima osservazione

- $\varphi_0 = 1$
- $\boldsymbol{\varphi} = [\varphi_0, \varphi_1, \dots, \varphi_{d-1}]^\top \in \mathbb{R}^{d\times 1}$
- $\boldsymbol{\theta} = [\theta_0, \theta_1, \dots, \theta_{d-1}]^\top \in \mathbb{R}^{d\times 1}$

- Il vettore $\boldsymbol{\theta} \in \mathbb{R}^{d\times 1}$ è il **vettore dei parametri**
- Il vettore $\boldsymbol{\varphi}(i) \in \mathbb{R}^{d\times 1}$ è il **vettore delle features** per la i-esima osservazione
- La quantità $\epsilon(i) \in \mathbb{R}$ è l'errore dovuto ad una non perfetta spiegazione di $y(i)$ tramite $\boldsymbol{\varphi}(i)$

Esempio (stimare il prezzo delle case)

Singola feature φ_3

Variabile di output y

Numero di osservazioni N

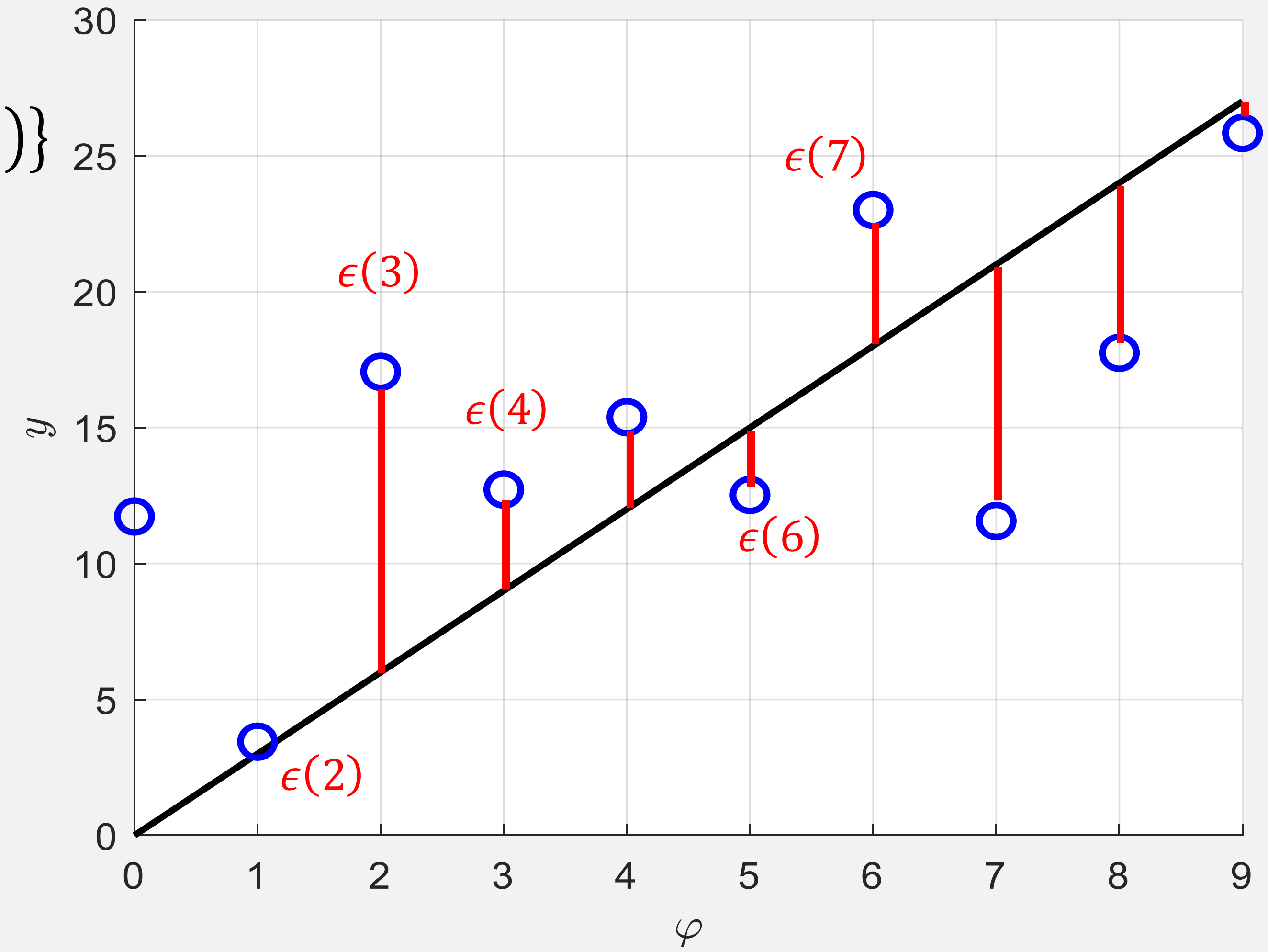
Area (feet^2)	# Camere da letto	# Piani	Età	Prezzo $(1000\$)$
2104	5	1	45	115
1416	3	2	40	150
1534	2	1	30	210
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Singola osservazione (regressore\features vector) $\boldsymbol{\varphi}$

- Il numero delle righe è il numero di osservazioni N
- L'osservazione i-esima è il vettore $\boldsymbol{\varphi}(i) = [\varphi_1(i) \quad \varphi_2(i) \quad \varphi_3(i) \quad \varphi_4(i)]^\top \in \mathbb{R}^{4x1}$
- Ogni regressore $\boldsymbol{\varphi}$ ha associata una risposta $y \in \mathbb{R}$ che vogliamo stimare

QUIZ!

Nel grafico seguente, quante **osservazioni** abbiamo? $\{(\varphi(1), y(1)), \ldots, (\varphi(N), y(N))\}$



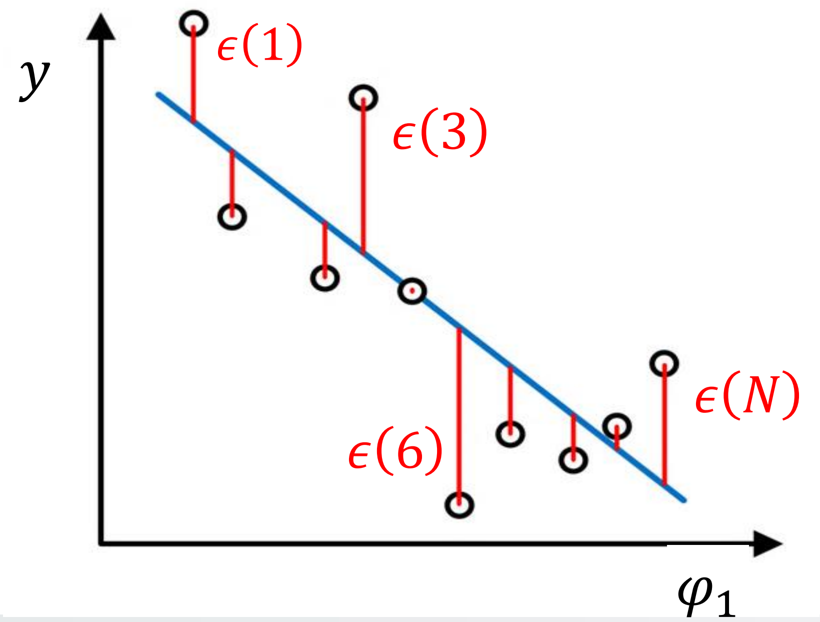
- ❑ $N = 10$ osservazioni
- ❑ $N = 7$ osservazioni
- ❑ $N = 9$ osservazioni

Interpretazione geometrica

Caso scalare (retta)

In questo caso c'è **un solo regressore** φ_1 e **due parametri** θ_0, θ_1

$$y(i) = \theta_0 + \theta_1\varphi_1(i) + \epsilon(i)$$



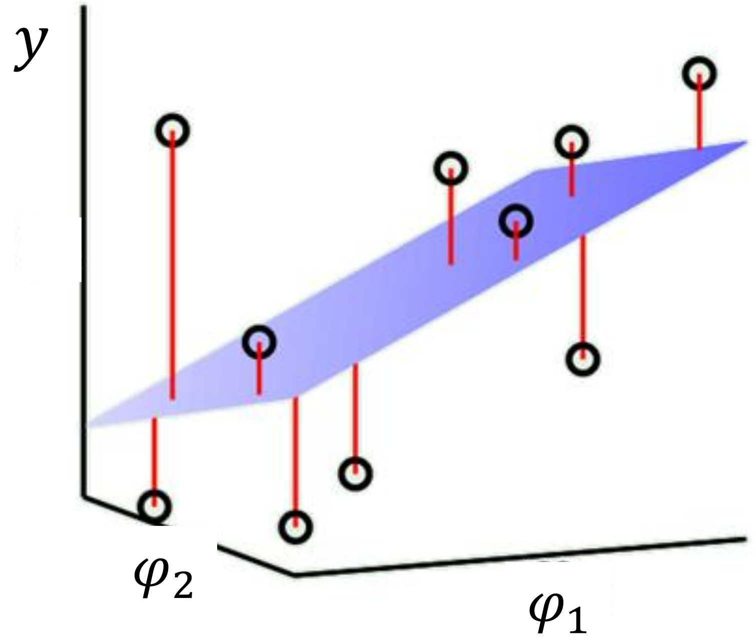
Caso con 2 regressori (piano)

In questo caso ci sono **due regressori** φ_1, φ_2 e **tre parametri** $\theta_0, \theta_1, \theta_2$

$$y(i) = \theta_0 + \theta_1\varphi_1(i) + \theta_2\varphi_2 + \epsilon(i)$$

Esempio

- y: peso [kg]
- φ_1: altezza [m]
- φ_2: età



Outline

1. Stima a minimi quadrati
2. **Funzione di costo**
3. Gradient descent
4. Proprietà dello stimatore a minimi quadrati
5. Esercizi con codice

Funzione di costo

Regressione lineare: modello lineare + minimi quadrati

Il metodo della regressione lineare stima i parametri $\boldsymbol{\theta}$ minimizzando l'errore quadratico tra **output osservati** e **stimati** dal modello lineare

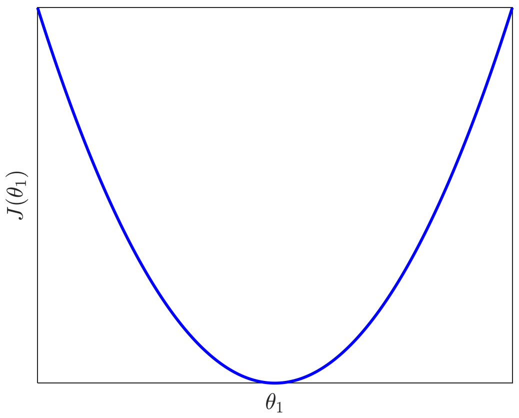
$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

Funzione di costo (cifra di merito)

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\Big(y(i) - \underset{1\times d}{\boldsymbol{\varphi}^{\top}(i)}\,\underset{d\times 1}{\boldsymbol{\theta}}\Big)^2 = \frac{1}{N}\sum_{i=1}^{N}\epsilon(i)^2$$

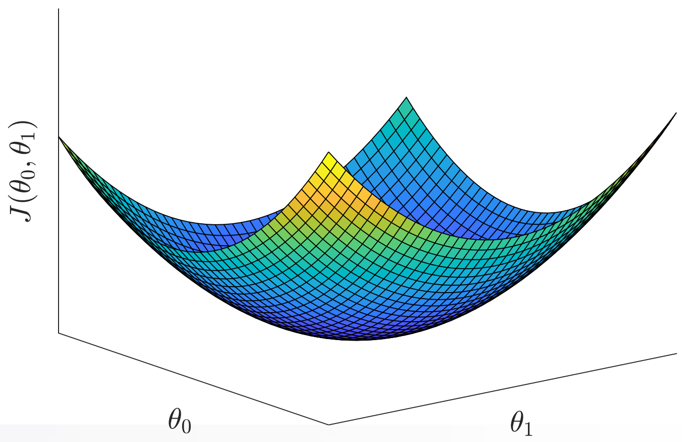
Caso scalare senza intercetta, $\theta_0 = 0$

$$y(i) = \theta_1\varphi_1(i) + \epsilon(i)$$



Caso scalare con intercetta, $\theta_0 \neq 0$

$$y(i) = \theta_0 + \theta_1\varphi_1(i) + \epsilon(i)$$



Minimizzazione della funzione di costo

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(y(i) - \boldsymbol{\varphi}(i)^{\top}\boldsymbol{\theta}\right)^2$$

$$\underset{d\times 1}{\nabla J(\boldsymbol{\theta})} = \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \underset{d\times 1}{\mathbf{0}} \Rightarrow \frac{2}{N}\sum_{i=1}^{N} \underset{d\times 1}{\boldsymbol{\varphi}(i)} \cdot \underset{1\times 1}{\left(y(i) - \underset{1\times d}{\boldsymbol{\varphi}^{\top}(i)}\underset{d\times 1}{\boldsymbol{\theta}}\right)} = \mathbf{0} \;\Rightarrow \sum_{i=1}^{N}\boldsymbol{\varphi}(i)y(i) - \sum_{i=1}^{N}\boldsymbol{\varphi}(i)\boldsymbol{\varphi}^{\top}(i)\boldsymbol{\theta} = \mathbf{0}$$

$$\Rightarrow \left[\sum_{i=1}^{N}\underset{d\times d}{\boldsymbol{\varphi}(i)\boldsymbol{\varphi}^{\top}(i)}\right]\underset{d\times 1}{\boldsymbol{\theta}} = \sum_{i=1}^{N}\underset{d\times 1}{\boldsymbol{\varphi}(i)y(i)}$$

$$\boxed{\Rightarrow \underset{d\times 1}{\hat{\boldsymbol{\theta}}} = \left[\sum_{i=1}^{N}\underset{d\times d}{\boldsymbol{\varphi}(i)\boldsymbol{\varphi}^{\top}(i)}\right]^{-1} \cdot \left[\sum_{i=1}^{N}\underset{d\times 1}{\boldsymbol{\varphi}(i)y(i)}\right]}$$

Poiché il modello è **lineare nei parametri** e la misura dell'errore è **quadratica**, la funzione di costo è **convessa** → ammette un **minimo unico** (globale)

Nel caso della regressione lineare, il minimo può anche essere trovato in **forma chiusa**

Funzione di costo: caso matriciale

Possiamo esprimere il problema della regressione lineare usando delle matrici

Vettore dei regressori $\boldsymbol{\varphi}^\top(1)$ $\quad 1 \times d$

$$\underset{N \times d}{X} = \begin{bmatrix} 1 & \varphi_1(1) & \varphi_2(1) & \cdots & \varphi_{d-1}(1) \\ 1 & \varphi_1(2) & \varphi_2(2) & & \varphi_{d-1}(2) \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & \varphi_1(N) & \varphi_2(N) & \cdots & \varphi_{d-1}(N) \end{bmatrix} \qquad \underset{d \times 1}{\boldsymbol{\theta}} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_{d-1} \end{bmatrix} \qquad \underset{N \times 1}{Y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} \qquad \underset{N \times 1}{E} = \begin{bmatrix} \epsilon(1) \\ \epsilon(2) \\ \vdots \\ \epsilon(N) \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{\varphi}^\top(1) \\ \boldsymbol{\varphi}^\top(2) \\ \vdots \\ \boldsymbol{\varphi}^\top(N) \end{bmatrix} \qquad \underset{N \times 1}{Y} = \underset{N \times d}{X} \overset{d \times 1}{\boldsymbol{\theta}} + \underset{N \times 1}{E} \Rightarrow$$

$$\boxed{J(\boldsymbol{\theta}) = \frac{1}{N} \|Y - X\boldsymbol{\theta}\|_2^2 = \frac{1}{N} \underset{1 \times N}{(Y - X\boldsymbol{\theta})^\top} \underset{N \times 1}{(Y - X\boldsymbol{\theta})}}$$

Funzione di costo: caso matriciale

È utile ricordare queste proprietà di derivazione matriciale (https://en.wikipedia.org/wiki/Matrix_calculus)

$$\boxed{\nabla_{\boldsymbol{x}}(\underset{1\times d}{\boldsymbol{x}^\top} \cdot \underset{d\times d}{A} \cdot \underset{d\times 1}{\boldsymbol{x}}) = \underset{d\times d}{(A + A^\top)} \cdot \underset{d\times 1}{\boldsymbol{x}} \qquad \nabla_{\boldsymbol{x}}(\underset{1\times d}{\boldsymbol{x}^\top} \cdot \underset{d\times 1}{\boldsymbol{b}}) = \underset{d\times 1}{\boldsymbol{b}}}$$

$$J(\boldsymbol{\theta}) = \frac{1}{N}(Y - X\boldsymbol{\theta})^\top(Y - X\boldsymbol{\theta}) = \frac{1}{N}(\underset{1\times N}{Y^\top}\overset{N\times 1}{Y} - \underset{1\times N}{Y^\top}\underset{N\times d}{X} \cdot \overset{d\times 1}{\boldsymbol{\theta}} - \underset{1\times d}{\boldsymbol{\theta}^\top} \cdot \underset{d\times N}{X^\top}\overset{N\times 1}{Y} + \underset{1\times d}{\boldsymbol{\theta}^\top} \cdot \underset{d\times N}{X^\top}\underset{N\times d}{X} \cdot \overset{d\times 1}{\boldsymbol{\theta}})$$

$$= \frac{1}{N}(Y^\top Y - 2 \cdot \boldsymbol{\theta}^\top \cdot X^\top Y + \boldsymbol{\theta}^\top \cdot X^\top X \cdot \boldsymbol{\theta})$$

$$\nabla J(\boldsymbol{\theta}) = \mathbf{0} \Rightarrow \frac{1}{N}(-2X^\top Y + 2X^\top X\boldsymbol{\theta}) = \mathbf{0} \Rightarrow$$

$$\boxed{\underset{d\times 1}{\hat{\boldsymbol{\theta}}} = \underset{d\times d}{(X^\top X)^{-1}}\underset{d\times N}{X^\top}\underset{N\times 1}{Y}}$$

Normal equations

$$\boxed{\hat{\boldsymbol{\theta}} = (X^\top X)^{-1} X^\top Y}$$

Normal equations

Cosa succede se $X^\top X$ **non è invertibile**? → Si usa la **pseudo-inversa**. In MatLab:

```
theta_hat = pinv(X'*X)*X*Y
```

- **Regressori ridondanti** (linearmente dipendenti)
  - ✓ $\varphi_1 = \text{altezza in m}$
  - ✓ $\varphi_2 = \text{altezza in feet}$
- **Troppi regressori** (e.g. $N \leq d$)
  - ✓ Rimuovere qualche regressore
  - ✓ Usare regolarizzazione (la vedremo più avanti...)

- Il metodo delle normal equation è **lento** se d è molto grande
  - ✓ Per risolvere questo problema, si usano **metodi iterativi** come il **gradient descent**

Outline

1. Stima a minimi quadrati
2. Funzione di costo
3. **Gradient descent**
4. Proprietà dello stimatore a minimi quadrati
5. Esercizi con codice

Gradient descent

Il **gradient descent** è un metodo **iterativo** per minimizzare le funzioni differenziabili (ovvero funzioni in cui possiamo calcolare le derivate in ogni punto del dominio)

Consideriamo prima il **caso scalare** (abbiamo un solo parametro $\theta \in \mathbb{R}$ da stimare)

Dato un valore iniziale $\hat{\theta}^{(0)}$, la stima $\hat{\theta}^{(k+1)}$ del parametro θ all'iterazione $k + 1$ è:

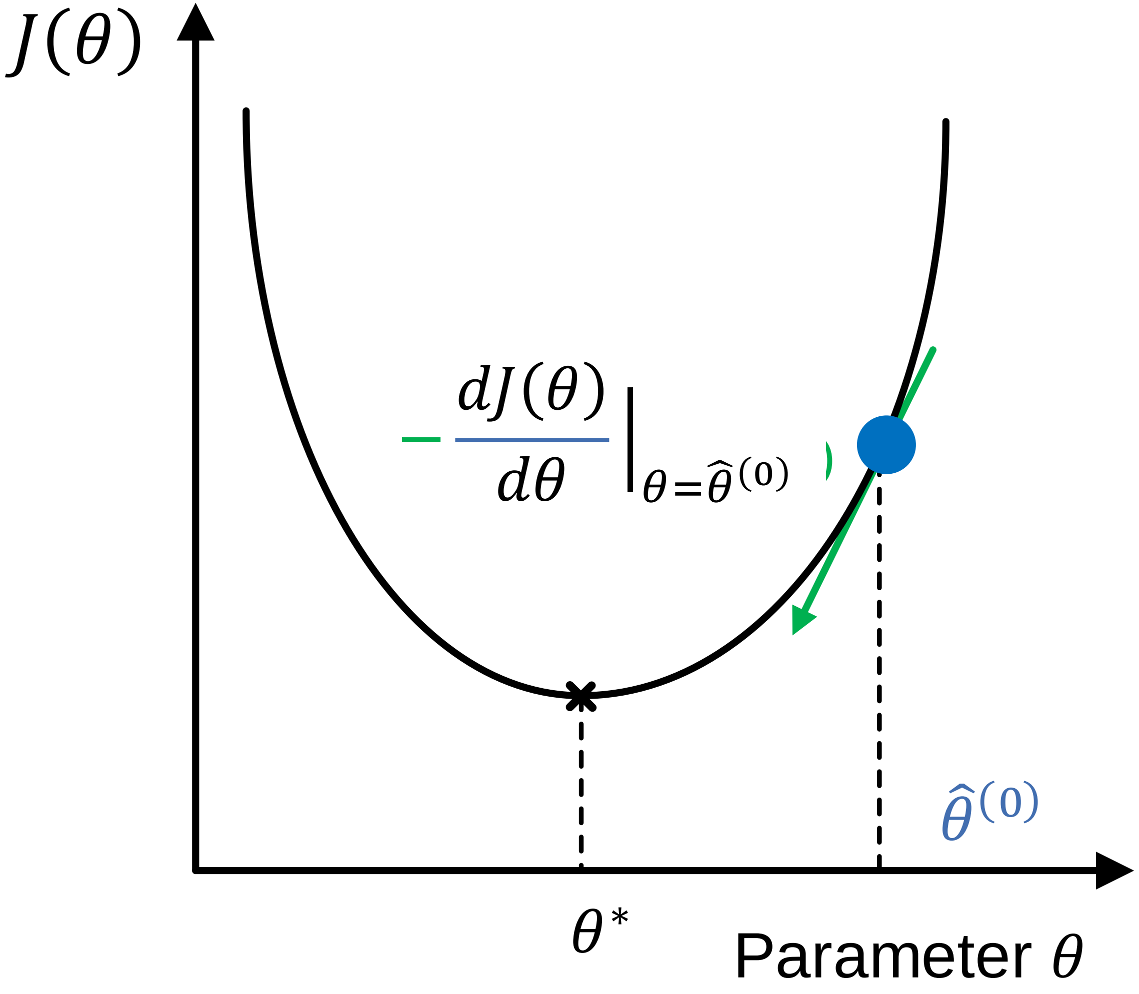
$$\boxed{\underset{1\times 1}{\hat{\theta}^{(k+1)}} = \underset{1\times 1}{\hat{\theta}^{(k)}} - \underset{1\times 1}{\alpha} \cdot \underset{1\times 1}{\frac{\partial J(\theta)}{\partial \theta}}\bigg|_{\theta=\hat{\theta}^{(k)}}}$$

$\alpha \in \mathbb{R}_{>0}$: learning rate

Gradient descent

Caso scalare $\theta \in \mathbb{R}$

$$\boxed{\hat{\theta}^{(k+1)} = \hat{\theta}^{(k)} - \alpha \cdot \frac{\partial J(\theta)}{\partial \theta}\bigg|_{\theta=\hat{\theta}^{(k)}}}$$



$$\frac{dJ(\theta)}{d\theta}\bigg|_{\theta=\hat{\theta}^{(k)}} > 0 \Rightarrow \quad \hat{\theta}^{(k+1)} < \hat{\theta}^{(k)}$$

La nuova stima è più vicina al valore ottimale θ^*

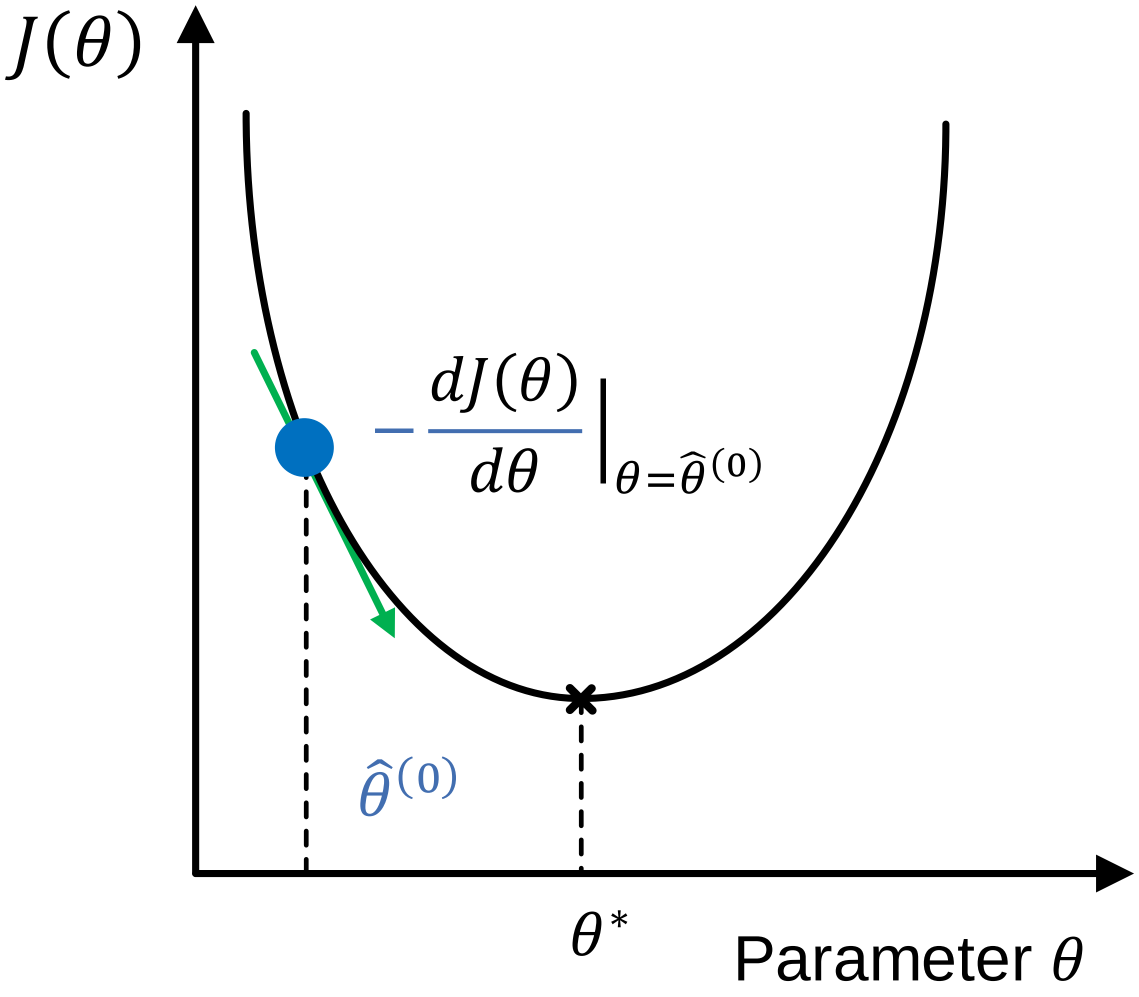
Gradient descent

Caso scalare $\theta \in \mathbb{R}$

$$\hat{\theta}^{(k+1)} = \hat{\theta}^{(k)} - \alpha \cdot \frac{\partial J(\theta)}{\partial \theta}\bigg|_{\theta=\hat{\theta}^{(k)}}$$

$$\frac{dJ(\theta)}{d\theta}\bigg|_{\theta=\hat{\theta}^{(k)}} < 0 \Rightarrow \quad \hat{\theta}^{(k+1)} > \hat{\theta}^{(k)}$$

La nuova stima è più vicina al valore ottimale θ^*



Gradient descent

Nel caso generale **multivariabile** (i.e. stimare un vettore di parametri $\boldsymbol{\theta} \in \mathbb{R}^{d\times 1}$), dobbiamo sostituire la derivata con il **vettore gradiente** $\nabla J(\boldsymbol{\theta}) \in \mathbb{R}^{d\times 1}$

Dato un valore iniziale $\widehat{\boldsymbol{\theta}}^{(0)}$, la stima $\widehat{\boldsymbol{\theta}}^{(k+1)}$ del vettore di parametri $\boldsymbol{\theta}$ all'iterazione $k + 1$ è:

$$\boxed{\underset{d\times 1}{\widehat{\boldsymbol{\theta}}^{(k+1)}} = \underset{d\times 1}{\widehat{\boldsymbol{\theta}}^{(k)}} - \underset{1\times 1}{\alpha} \cdot \underset{d\times 1}{\nabla J(\boldsymbol{\theta})}\Big|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}^{(k)}}}$$

$\alpha \in \mathbb{R}_{>0}$: learning rate

Gradient descent: trick computazionale

Quando sono presenti più regressori (caso **multivariabile**) è utile normalizzarne i valori, in modo che l'algoritmo del gradient descent «faccia meno fatica» a raggiungere il minimo

Calcolo la media per ogni regressore (che non sia quello dell'intercetta)

$$\hat{\mu}_j = \frac{1}{N}\sum_{i=1}^{N} \varphi_j(i) \qquad j = 1, \dots, d-1$$

Calcolo la varianza per ogni regressore (che non sia quello dell'intercetta)

$$\hat{\sigma}_j^2 = \frac{1}{N}\sum_{i=1}^{N} \left(\varphi_j(i) - \mu_j\right)^2 \quad j = 1, \dots, d-1$$

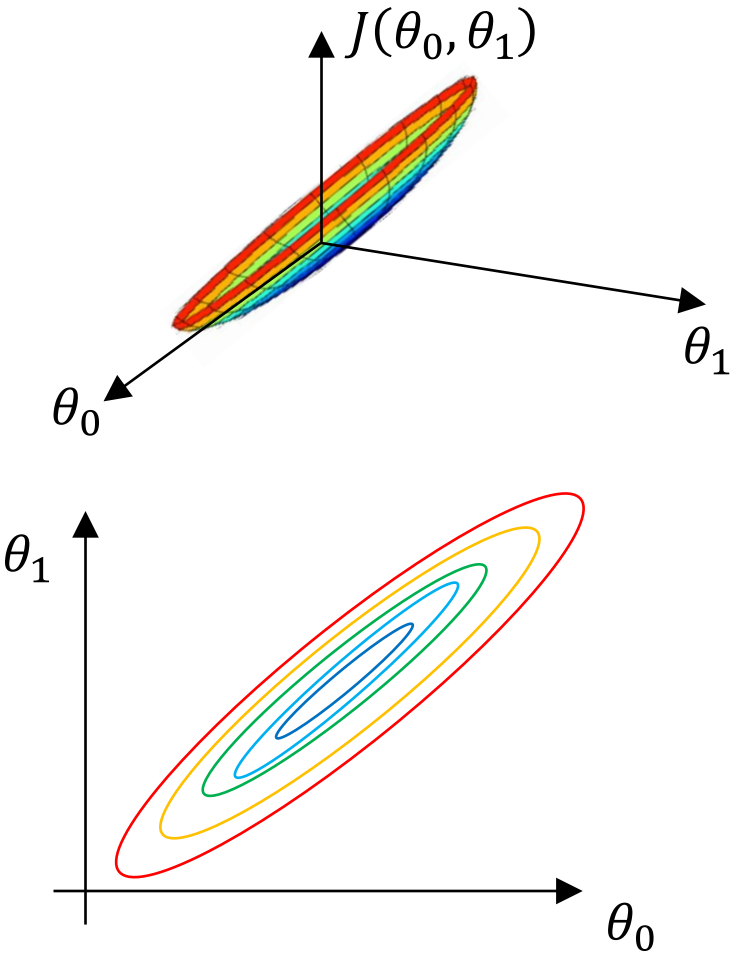
Sottraggo media e divido per deviazione standard

$$\boxed{\varphi_j(i) = \frac{\varphi_j(i) - \hat{\mu}_j}{\sqrt{\hat{\sigma}_j^2}} \qquad j = 1, \dots, d-1}$$

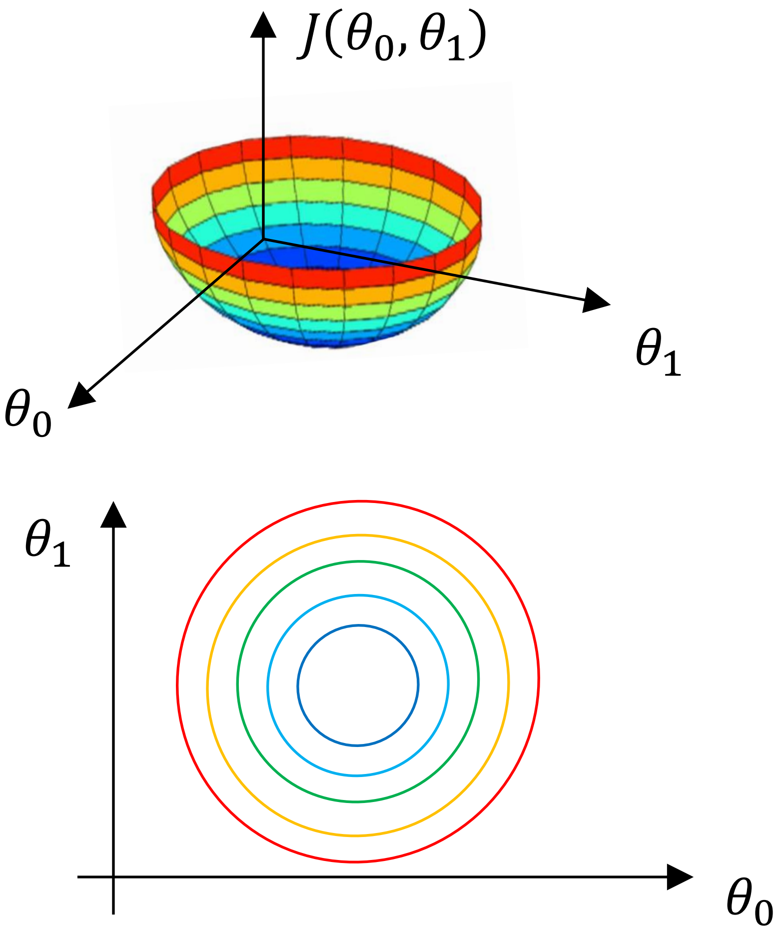
Normalizzare i nuovi dati usando LA STESSA MEDIA E LA STESSA VARIANZA calcolata sul dataset usato per stimare il modello

Gradient descent: trick computazionale

Regressori non normalizzati



Regressori normalizzati



Outline

Proprietà dello stimatore a minimi quadrati

Dubbio legittimo: come si comporta lo **stimatore a minimi quadrati** di un modello lineare nel caso in cui il sistema vero (che genera i dati) sia effettivamente lineare?

$$y(i) = \boldsymbol{\varphi}^\top(i)\boldsymbol{\theta}^0 + \epsilon(i)$$

Supponiamo che $\epsilon(i)$ sia una variabile casuale a media nulla, con una certa varianza λ^2

Nota: non stiamo assumendo nessuna specifica distribuzione di probabilità su $\epsilon(i)$

Proprietà dello stimatore a minimi quadrati (nel caso del sistema di cui sopra)

- Lo stimatore è **corretto:** $\mathbb{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}^0$
- Supponendo inoltre che i rumori siano incorrelati $\mathbb{E}[\epsilon(i)\epsilon(j)] = 0, \forall i \neq j$, lo stimatore è **consistente:** $\mathrm{Var}[\hat{\boldsymbol{\theta}}] = \lambda^2 \cdot (X^\top X)^{-1} = \lambda^2 \cdot P$

Outline

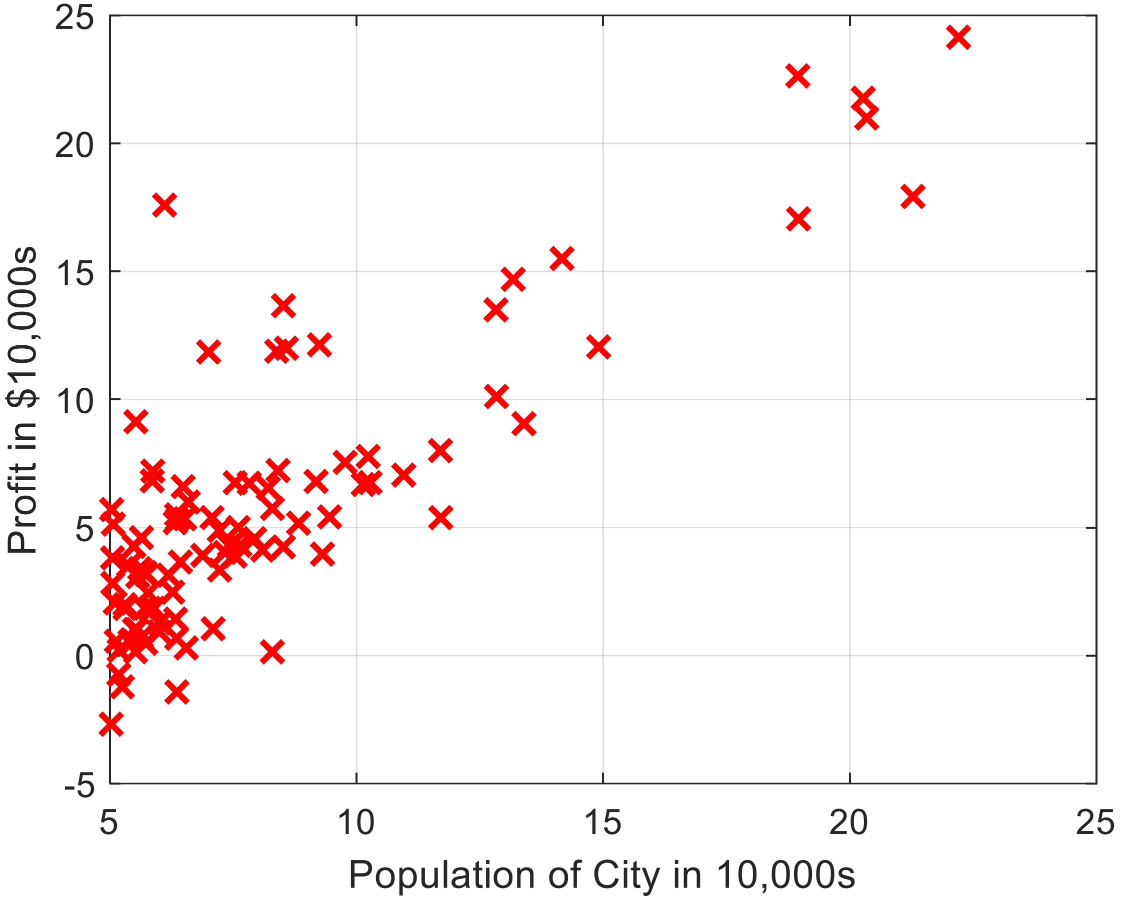
1. Stima a minimi quadrati
2. Funzione di costo
3. Gradient descent
4. Proprietà dello stimatore a minimi quadrati
5. **Esercizi con codice**

Esercizio 1: Stima dei profitti di un ristorante

Problema: il CEO di un franchising di ristoranti che sta valutando diverse città per l'apertura di un nuovo ristorante.

La catena ha già ristoranti in varie città e sono disponibili dati di profitti e di popolazione di questa città.

L'obiettivo è utilizzare questi dati per selezionare in quale città aprire la nuova attività



- Ogni città è descritta da:
  - ✓ φ_1: Popolazione $[\text{in } 10000 \text{ unità}]$

✓ L'output y è il profitto $[\text{in } 10000\$\,]$

- Il dataset consiste di $N = 97$ città con $\varphi_1(i)$, e $y(i)$, per $i = 1, \ldots, N$

Esercizio 2: stima dei prezzi delle case

Vogliamo **stimare il prezzo** delle case a Portland, Oregon. L'output y è quindi il prezzo

- Ogni casa è descritta da:
  - ✓ φ_1: Area $[\text{feet}^2]$
  - ✓ φ_2: Numero di camere da letto

- Il dataset consiste di $N = 47$ case con $\varphi_1(i), \varphi_2(i)$ e $y(i)$, per $i = 1, \dots, N$

$$y(i) = \boldsymbol{\varphi}^\top(i)\boldsymbol{\theta} + \epsilon(i) \qquad \underset{3\times 1}{\boldsymbol{\varphi}(i)} = [1 \quad \varphi_1(i) \quad \varphi_2(i)]^\top$$

$$\underset{47\times 3}{X} = \begin{bmatrix} \boldsymbol{\varphi}^\top(1) \\ \boldsymbol{\varphi}^\top(2) \\ \vdots \\ \boldsymbol{\varphi}^\top(N) \end{bmatrix} \qquad \underset{3\times 1}{\boldsymbol{\theta}} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} \qquad \underset{47\times 1}{Y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(47) \end{bmatrix}$$

```
% Read data from file
data = csvread('ex2data.txt');
X = data(:, 1:2); % Features
y = data(:, 3); % Price
N = length(y); % Number of data

% Add intercept term to X
X = [ones(N, 1) X];

% Calculate the parameters from the normal equation
theta_hat = pinv(X'*X)*X'*y;
```

```
% Estimate the price of a 1650 sq-ft, 3 br house
price_hat = [1 3 1650]*theta_hat;
```

Punto non visto durante la stima di $\boldsymbol{\theta}$

Calcolare e implementare il gradiente

Come calcoliamo il gradiente? Supponiamo che il nostro modello sia

$$\boxed{y = \theta_0 + \theta_1 \cdot \varphi + \epsilon}$$

$$J(\theta_0, \theta_1) = \frac{1}{N} \sum_{i=1}^{N} \left(y(i) - \theta_0 - \theta_1 \cdot \varphi(i)\right)^2 \quad \Rightarrow \quad \underset{2\times 1}{\nabla J(\theta_0, \theta_1)} = \begin{bmatrix} \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0} & \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1} \end{bmatrix}^\top$$

$$\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0} = \frac{2}{N} \sum_{i=1}^{N} \left(y(i) - \theta_0 - \theta_1 \cdot \varphi(i)\right) \cdot (-1) \quad = -\frac{2}{N} \underset{1\times N}{X(:,1)^\top} \cdot \underset{N\times 1}{(Y - X\boldsymbol{\theta})}$$

$$\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1} = \frac{2}{N} \sum_{i=1}^{N} \left(y(i) - \theta_0 - \theta_1 \cdot \varphi(i)\right) \cdot \left(-\varphi(i)\right) \quad = -\frac{2}{N} \underset{1\times N}{X(:,2)^\top} \cdot \underset{N\times 1}{(Y - X\boldsymbol{\theta})}$$

Calcolare e implementare il gradiente

In generale, se abbiamo **più di un regressore** (ovvero, un vettore $\boldsymbol{\varphi} = [1\ \varphi_1,\ \varphi_2, \dots, \varphi_{d-1}]^\top \in \mathbb{R}^{d\times 1}$) possiamo implementare il gradient descent come di seguito:

```
For {
```

$$\theta_0 = \theta_0 - \alpha \cdot \frac{2}{N}\sum_{i=1}^{N}\left(y(i) - \boldsymbol{\varphi}^\top(i)\boldsymbol{\theta}\right)\cdot(-1)$$

$$\theta_1 = \theta_1 - \alpha \cdot \frac{2}{N}\sum_{i=1}^{N}\left(y(i) - \boldsymbol{\varphi}^\top(i)\boldsymbol{\theta}\right)\cdot\left(-\varphi_1(i)\right)$$

$$\vdots$$

$$\theta_{d-1} = \theta_{d-1} - \alpha \cdot \frac{2}{N}\sum_{i=1}^{N}\left(y(i) - \boldsymbol{\varphi}^\top(i)\boldsymbol{\theta}\right)\cdot\left(-\varphi_{d-1}(i)\right)$$

```
}
```

UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Dipartimento
di Ingegneria Gestionale,
dell'Informazione e della Produzione