Probabilità e statastica - Foglio 1, esercizio 15 A.A. 2024-25

Basso Kevin De Fina Giuseppe Mantoan Matteo Rampazzo Filippo
Vigolo Davide

25 marzo 2025

Indice

0.1 Minimizzazione della funzione φ

Data la funzione di errore:

$$\varphi(a,b) = \sum_{i=1}^{n} \left(y_i - (ax_i + b)\right)^2,$$

vogliamo trovare (a^*, b^*) che la minimizzano.

0.1.1 Calcolo delle derivate parziali

Le derivate parziali rispetto ad a e b sono:

$$\frac{\partial \varphi}{\partial a} = -2\sum_{i=1}^{n} x_i \left(y_i - (ax_i + b)\right) = 0,$$

$$\frac{\partial \varphi}{\partial b} = -2\sum_{i=1}^{n} \left(y_i - (ax_i + b)\right) = 0.$$

0.1.2 Sistema di equazioni

Dividendo per -2 e riorganizzando:

$$\begin{cases} \displaystyle\sum_{i=1}^{n} x_i \left(y_i - (ax_i + b)\right) = 0, & \text{(Equazione 1)} \\ \displaystyle\sum_{i=1}^{n} \left(y_i - (ax_i + b)\right) = 0. & \text{(Equazione 2)} \end{cases}$$

0.1.3 Soluzione per b (Equazione 2)

Dall'Equazione 2:

$$\begin{aligned} 0 &= \sum_{i=1}^{n} \left(y_i - (ax_i + b)\right) \\ &= \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} ax_i + \sum_{i=1}^{n} b \\ &= \sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i - nb \\ b &= \frac{1}{n}\sum_{i=1}^{n} y_i - \frac{a}{n}\sum_{i=1}^{n} x_i. \\ b &= \overline{y} - a\overline{x}. \end{aligned}$$

0.1.4 Soluzione per a (Equazione 1)

Sostituendo $b = \bar{y} - a\bar{x}$ nell'Equazione 1:

$$\begin{aligned} 0 &= \sum_{i=1}^{n} x_i \left(y_i - (ax_i + b)\right) \\ &= \sum_{i=1}^{n} x_i \left(y_i - ax_i - (\bar{y} - a\bar{x})\right) \\ &= \sum_{i=1}^{n} x_i(y_i - \bar{y}) - a\sum_{i=1}^{n} x_i(x_i - \bar{x}) \end{aligned}$$

Riscriviamo le somme:

$$\sum_{i=1}^{n} x_i(y_i - \bar{y}) = \sum_{i=1}^{n}(x_i - \bar{x} + \bar{x})(y_i - \bar{y}) = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) + \bar{x}\sum_{i=1}^{n}(y_i - \bar{y}),$$

dove $\sum_{i=1}^{n}(y_i - \bar{y}) = 0$ dato che $\sum_{i=1}^{n}(y_i)$ è n volte la media, quindi uguale a $\sum_{i=1}^{n}(\bar{y})$, da questo segue che:

$$\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) + \bar{x}\sum_{i=1}^{n}(y_i - \bar{y}) = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}). \tag{1}$$

Analogamente:

$$\sum_{i=1}^{n} x_i(x_i - \bar{x}) = \sum_{i=1}^{n} x_i^2 - \bar{x}x_i = \sum_{i=1}^{n} x_i^2 - 2\bar{x}x_i + \bar{x}^2 + \bar{x}x_i - \bar{x}^2$$

$$= \sum_{i=1}^{n}(x_i - \bar{x})^2 + \bar{x}\sum_{i=1}^{n}(x_i - \bar{x}) \tag{2}$$

dove $\sum_{i=1}^{n}(x_i - \bar{x}) = 0$ per lo stesso motivo di sopra, quindi:

$$\sum_{i=1}^{n} x_i(x_i - \bar{x}) = \sum_{i=1}^{n}(x_i - \bar{x})^2.$$

Sostituendo nell'Equazione 1:

$$\begin{aligned}\sum_{i=1}^{n} (y_i - (ax_i + b)) &= \sum_{i=1}^{n} x_i(y_i - \bar{y}) - a\sum_{i=1}^{n} x_i(x_i - \bar{x}) \text{ per 1) e 2)} \\ &= \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) - a\sum_{i=1}^{n}(x_i - \bar{x})^2 = 0.\end{aligned}$$

Isolando a:

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

0.1.5 Soluzione finale

I coefficienti ottimali sono:

$$\boxed{a^* = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \quad b^* = \bar{y} - a^*\bar{x}}$$

0.1.6 Conclusione

Si nota che $\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$ è la covarianza tra x e y (cov_{xy}) e che $\sum_{i=1}^{n}(x_i - \bar{x})^2$ è la varianza di x (var_x), quindi possiamo riscrivere a^* come:

$$a^* = \frac{\mathrm{cov}_{xy}}{\mathrm{var}_x}.$$

Nota: La formula per a^* è il *coefficiente angolare* della retta di regressione, mentre b^* è l'*intercetta* che assicura che la retta passi per il punto $(\bar{x}, \bar{y})$.

0.2 Codice

0.2.1 Calcolo della retta di regressione

File "regression.py"

```python
import statistics

"""
    dato un campione bi-variato in x, y calcola:
        - a* -> il coefficiente angolare della retta di regressione
        - b* -> l'intercetta di tale retta

    nota: usiamo statistics.mean() per una maggiore stabilita
          numerica nel risultato, verificato tramite vari test
"""
def regression(x, y):
    # calcolo media dei valori in x e y
    media_x = statistics.mean(x)
    media_y = statistics.mean(y)

    # calcolo numeratore e denominatore tramite
    # formula precedentemente ricavata
    num = sum((xi - media_x) * (yi - media_y) for xi, yi in zip(x, y))
    den = sum((xi - media_x) ** 2 for xi in x)

    # calcolo di a* e b*
    a_s = num / den
    b_s = media_y - a_s * media_x
    return a_s, b_s
```

0.2.2 Import dei dati

File "main.py"

```python
def leggi_da_file(nome_file):
    with open(nome_file, 'r') as f:
        data = f.readlines()
    data = [line.strip().split(',') for line in data]
    return data
```

0.2.3 Principal Component Analysis

File "pca.py". Il codice è stato realizzato seguendo i principi presenti in questo articolo.

```python
import numpy as np

def costruisci_mat_covarianza(data):
    return np.cov(data, rowvar=False)

def decomposizione_spettrale(matrix):
    autovalori, autovettori = np.linalg.eig(matrix)
    return autovettori.T, autovalori  # Le righe sono autovettori

def pca(data):
    data = np.array(data, dtype=float)

    # 1. Standardizzazione dei dati
    media = np.mean(data, axis=0)
    std = np.std(data, axis=0, ddof=1)  # Usa la deviazione standard del campione (ddof=
```

```
    data_std = (data - media) / std

    # 2. Calcola la matrice di covarianza dei dati standardizzati
    matrice_covarianza = costruisci_mat_covarianza(data_std)

    # 3. Decomposizione spettrale
    autovettori, autovalori = decomposizione_spettrale(matrice_covarianza)

    # 4. Ordina gli autovettori per autovalori (decrescente)
    indici_ordinati = np.argsort(autovalori)[::-1]
    autovettori = autovettori[indici_ordinati, :]  # Ordina le righe
    autovalori = autovalori[indici_ordinati]

    # 5. Seleziona i primi 2 autovettori (righe)
    max_autovettori = autovettori[:2, :]

    # 6. Proietta i dati standardizzati
    dati_proiettati = (max_autovettori @ data_std.T).T

    return dati_proiettati
```

0.2.4 Stampa dei grafici e dei valori ricavati

File "main.py"

```
from matrix import *
from regression import *
from pca import *

def main():
    # Lettura dei dati
    data = leggi_da_file('./dati/dati.csv')
    tmin = [float(row[1]) for row in data]
    tmed = [float(row[2]) for row in data]
    tmax = [float(row[3]) for row in data]
    ptot = [float(row[4]) for row in data]
    data = list(zip(tmin, tmed, tmax, ptot))

    a,b = regression(tmin, tmed) # coefficienti della retta di regressione
        a*x + b
    print(f"Coefficienti della retta di regressione lineare (tmin, tmed): a
        = {a}, b = {b}")

    # primo campione bivariato tmin,tmed
    plt.figure(1)
    plt.plot(tmin, tmed, 'o', label='Dati')
    # per ogni punto xi nel dataset, valuta la retta di regressione in quel
        punto
    plt.plot(tmin, [a * xi + b for xi in tmin], 'r', label='Retta di
        regressione')
    plt.ylabel('Tmed')
    plt.xlabel('Tmin')
    plt.title('Regressione lineare sul primo campione bivariato (tmin,tmed)
        ')
    plt.legend()
    plt.show()

    # secondo campione bivariato tmin,ptot
```

```
    a2, b2 = regression(tmin, ptot)
    print(f"Coefficienti della retta di regressione lineare (tmin, ptot): a
        = {a2}, b = {b2}")

    plt.figure(2)
    plt.plot(tmin, ptot, 'o', label='Dati')
    # per ogni punto xi nel dataset, valuta la retta di regressione in quel
         punto
    plt.plot(tmin, [a2 * xi + b2 for xi in tmin], 'r', label='Retta di
        regressione')
    plt.xlabel('Tmin')
    plt.ylabel('Ptot')
    plt.title('Regressione lineare sul secondo campione bivariato (tmin,
        ptot)')
    plt.legend()
    plt.show()

    dati_proiettati = pca(data)
    # Plot dei dati proiettati sulle prime due componenti principali
    plt.figure(3)
    plt.scatter(dati_proiettati[:, 0], dati_proiettati[:, 1])
    plt.xlabel('Componente 1')
    plt.ylabel('Componente 2')
    plt.title('PCA')
    plt.legend()
    plt.show()


if __name__ == "__main__":
    main()
```

0.3 Grafici

0.3.1 Tmin vs Tmed e retta di regressione

Output: "Coefficienti della retta di regressione lineare (tmin, tmed): $a = 0.9299473114832919, b = 4.515694867377192$".

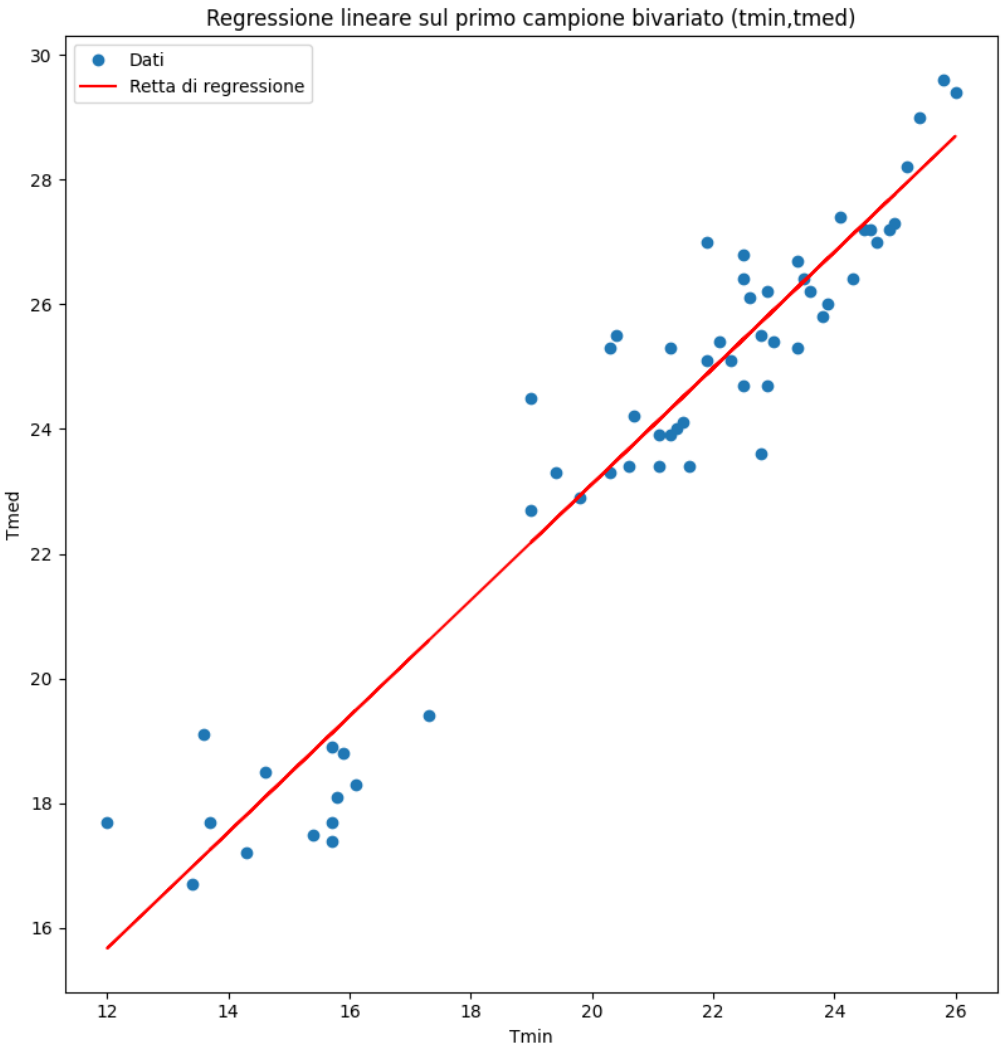

Figura 1: Tmin vs Tmed

0.3.2 Tmin vs Ptot e retta di regressione

Output: "Coefficienti della retta di regressione lineare (tmin, ptot): $a = -0.7340932767048642, b = 18.131124956593744$".

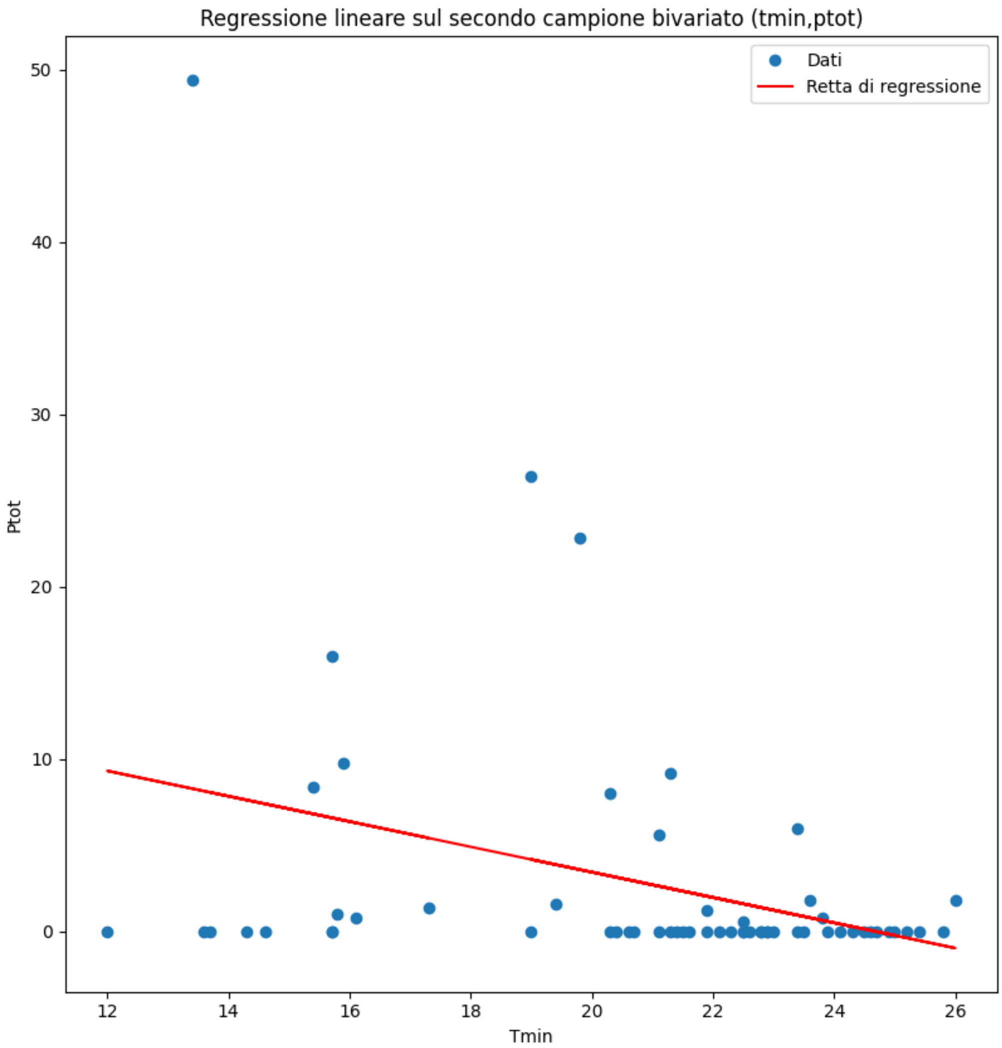

Figura 2: Tmin vs Ptot

0.3.3 Principal Component Analysis

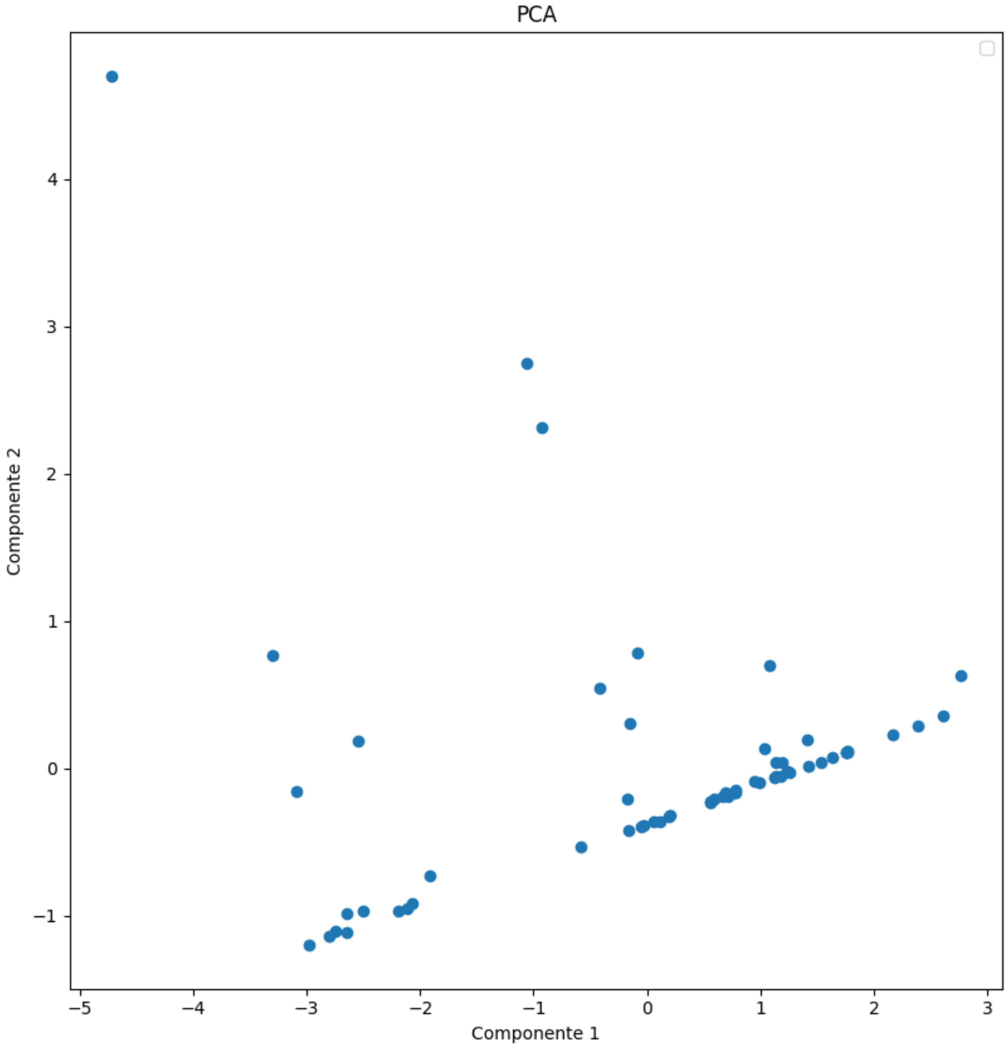

Figura 3: Principal Component Analysis sul campione 4-variato, plot delle prime due componenti principali.