Matematyczne aspekty analizy danych
semestr zimowy 2024/2025

Dr Anna Muranova
UWM w Olsztynie

Wykład 10

Współczynnik korelacji liniowej oraz Współczynnik determinacji

Współczynnik korelacji liniowej definiuje się następująco:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}},$$

gdzie $r_{xy} \in [-1, 1]$. *Współczynnik determinacji* R^2 jest zdefiniowane jako

$$R^2 := \frac{\sum_{i=1}^{n}(\hat{Y}_i - \overline{Y})^2}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2} \geq 0,$$

gdzie:

- Y_i – i-ta obserwacja zmiennej Y,
- $\hat{Y}_i$ – wartość teoretyczna zmiennej objaśnianej (na podstawie modelu),
- $\overline{y}$ – średnia arytmetyczna empirycznych wartości zmiennej objaśnianej.

Twierdzenie

$$R^2 = r_{xy}^2$$

Python: numpy R2

```
import numpy as np

RAM = np.array([2, 4, 8, 16])
price = np.array([12, 16, 28, 62])
mprice = np.mean(price)
n = len(RAM)
beta2 = (n*sum(RAM*price) - sum(RAM)*sum(price))/
            (n*sum(RAM**2) - (sum(RAM))**2)
print(beta2)
beta1 = (sum(price)-beta2*sum(RAM))/n
print(beta1)

pricehat = beta1 + RAM*beta2
R2 = sum((pricehat-mprice)**2)/sum((price-mprice)**2)
print(R2)
#0.9868244287681628
nRAM = RAM - np.mean(RAM)
nprice = price - np.mean(price)
r = sum(nRAM*nprice/(sum(nRAM**2)*sum(nprice**2))**0.5)
print(r**2)
#0.9868244287681627
```

Python: stats R2

```
import numpy as np
from scipy import stats

RAM = np.array([2, 4, 8, 16])
price = np.array([12, 16, 28, 62])

reg= stats.linregress(RAM, price)
print(reg.rvalue**2)
#0.9868244287681629
```

Istotność statystyczna

- Czy to możliwe, ze widzimy liniową zależność z przyczyn losowych?
- Czy korelacja w danych jest przypadkowa?
- Jak zyskać 95% pewność ze korelacja pomiędzy dwoma zjawiskami jest istotna, a nie przypadkowa?

Istotność statystyczna: hipotezy

Niech ρ – współczynnik korelacje w populacje, a r – w probie.

Hipoteza zerowa:
$H_0 : \rho = 0$ – brak związku Hipoteza alternatywna:
$H_0 : \rho \neq 0$ –związek istnieje (korelacja dodatnia lub ujemna)

W regresji liniowej do testowania hipotez używamy rozkład t zamiast rozkładu normalnego.

Jeśli mamy n punktów dla regresji liniowej, to patrzymy na rozklad t z $n-1$ stopniami swobody.

Porównujemy 95% zakres ufności z wartością testowej:

$$t = \frac{r}{\sqrt{\frac{1-r^2}{n-2}}},$$

gdzie r - współczynnik korelacje, n – rozmiar próby.

Jeśli nasza wartość testowa wypada poza krytycznym zakresem $(-ppf(00.025), ppf(0.975))$ ufności, to przyjmujemy ze korelacja była nieprzypadkowa.

Istotność statystyczna: przykład

Dla naszego przykładu:

```
from scipy.stats import t

n = 4
dolna = t(n-1).ppf(0.025)#ppf-odwrotna gestosci
gorna = t(n-1).ppf(0.975)

print(dolna, gorna)#-3.1824463052842638 3.182446305284263
```

$$t = \frac{r}{\sqrt{\frac{1-r^2}{n-2}}} = \frac{0.99}{\sqrt{\frac{1-0.99^2}{2}}} = 9.92,$$

– nie przypadkowa.

Przy pomocy wartości p:

```
from scipy.stats import t

n = 4
p =1 - t(n-1).cdf(9.92)
print(p)#p=0.001<0.05
```

Błąd standardowy estymacje

Jednym ze sposobów mierzenia ogólnego błędu regresji liniowej jest SSE, czyli suma kwadratów błędów. Poznaliśmy ją wcześniej, kiedy podnosiliśmy do kwadratu każdą resztę i sumowali wyniki. Jeśli $\hat{y}$ to każda przewidywana wartość na linii, a y reprezentuje każdą rzeczywistą wartość w danych, wzór wygląda tak:

$$SSE = \sum_{i=1}^{n} (\hat{y} - y_i)^2$$

W naszym przykładzie:

Pojemność RAM	Cena
2	12
4	16
8	28
16	62

$$\hat{Y} = [9.456, 16.742, 31.314, 60.458]$$

$$SSE = (2.544)^2 + (0.742)^2 + (3.314)^2 + (1.542)^2 = 20.38$$

Błąd standardowy estymacji

Liczyć się jako

$$S_e = \sqrt{\frac{SSE}{n-2}} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y} - y_i) * *2}{n-2}}$$

Dlaczego $n-2$, a nie $n-1$ jak w wielu innych poprzednich obliczeniach wariancji? Bez zagłębiania w dowody matematyczne, to dlatego ze regresja liniowa ma dwie zmienne, a nie tylko jedną, wiec musimy zwiększyć niepewność o 1 w naszych stopniach swobody.

Błąd standardowy estymacji bardzo przypomina odchylenie standardowe. To nie przypadek - jest on właśnie odchyleniem standardowy regresji liniowej. W naszym przykładzie:

$$S_e = \sqrt{\frac{20.38}{4-2}} = \sqrt{\frac{20.38}{2}} = 3.19$$

SSE w numpy

```
import numpy as np

RAM = np.array([2, 4, 8, 16])
price = np.array([12, 16, 28, 62])
mprice = np.mean(price)
n = len(RAM)
beta2 = (n*sum(RAM*price) - sum(RAM)*sum(price))/
            (n*sum(RAM**2) - (sum(RAM))**2)
beta1 = (sum(price)-beta2*sum(RAM))/n

pricehat = beta1 + RAM*beta2
SSE = sum((pricehat-price)**2)
print(SSE)#20.38
Se = (SSE/(n-2))**0.5
print(Se)#3.19
```

Przedział przewidywania

Jeżeli zbudowaliśmy regresją liniową, to mamy wzór:

$$\hat{y} = \beta_1 + \beta_2 x$$

To oznacza, po podanym x' możemy obliczyć $\hat{y}'$.

Ponieważ to będzie oszacowanie wartości y' (mniej więcej jako średnia y' dla tych x'), to możemy tez podać cały przedział wartości y, odpowiadające podanemu progu ufności (tradycyjnie 95%).

Wzór na margines błędu:

$$E = t_{0,025} \cdot S_e \sqrt{1 + \frac{1}{n} + \frac{n(x' - \overline{x})^2}{n(\sum x^2) - (\sum x)^2}}$$

Wtedy 95% przedział ufności:

$$[y' - E, y' + E]$$

```
from scipy.stats import t

t = t(n-2).ppf(0.975)
```

Uwaga! $(n - 2)$

Przedział przewidywania: przykład

Pojemność RAM	Cena
2	12
4	16
8	28
16	62

Jeżeli zbudowaliśmy regresją liniową, to mamy wzór:

$$\hat{y} = 2.17 + 3.643x$$

Margines błędu:

$$E = t_{0,025} \cdot S_e \sqrt{1 + \frac{1}{n} + \frac{n(x' - \overline{x})}{n(\sum x^2) - (\sum x)^2}}$$

Z numpy: $t = 4,3$. Niech nas interesuje, ile kosztuje pamięć 64.

$$\hat{y}' = 2.17 + 3.643 \cdot 64 = 235.322$$

Dalej

$$E = 4.3 \cdot 3.19 \sqrt{1 + \frac{1}{4} + \frac{4(64 - 7.5)^2}{4(2^2 + 4^2 + 8^2 + 16^2) - (30)^2}} = 73.8$$

To oznacza, 95% pamięci Ram 64 kosztuje 235 ± 73.

Przedział przewidywania: przykład, $x' = 12$

Pojemność RAM	Cena
2	12
4	16
8	28
16	62

Jeżeli zbudowaliśmy regresją liniową, to mamy wzór:

$$\hat{y} = 2.17 + 3.643x$$

Margines błędu:

$$E = t_{0,025} \cdot S_e \sqrt{1 + \frac{1}{n} + \frac{n(x' - \overline{x})}{n(\sum x^2) - (\sum x)^2}}$$

Z numpy: $t = 4,3$. Niech nas interesuje, ile kosztuje pamięć 12.

$$\hat{y}' = 2.17 + 3.643 \cdot 12 = 45.886$$

Dalej

$$E = 4.3 \cdot 27.58 \sqrt{1 + \frac{1}{4} + \frac{4(12 - 7.5)}{4(2^2 + 4^2 + 8^2 + 16^2) - (30)^2}} = 16.4$$

To oznacza, 95% pamięci Ram 12 kosztuje od 29 do 61. (Mniejsze zakres)

Przedział przewidywania: numpy

```
import numpy as np
from scipy.stats import t

...[SSE, Se]

t = t(n-2).ppf(0.975)
print(t)
xprime = 64
yprime = beta1 + xprime*beta2
print(yprime)#235.35
mRAM= np.mean(RAM)
E = t*Se*(1 +1/n+n*(xprime-mRAM)**2/(4*sum(RAM**2)-sum(RAM)**2))**0.5
print(E)
```

Podział danych na treningowe i testowe (sprawdzanie krzyżowe)

Podstawowa technika, której używają praktycy uczenia maszynowego, aby ograniczyć problem nadmiernego dopasowania, sprawdzanie krzyżowe, tzn. podział danych na treningowe i testowe, w którym zwykle 2/3 danych odkłada się do testów, a pozostałych 2/3 używa się do treningu (możliwe są też inne proporcje). Treningowy zbiór danych wykorzystuje się do dopasowania regresji liniowej, a testowy – do zmierzenia jak dobrze działa regresja liniowa na wcześniej nieznanych danych.

Techniki tej używa się zasad metodach nadzorowanego uczenia maszynowego, m.in. w regresje logistycznej i sieciach neuronowych.

Jeżeli zbiór danych jest mały, są inne sposoby dzielenia zbioru danych niż 2/3 i 1/3. Jeśli masz tak mały zbiór danych, prawdopodobnie lepszy byłby podział w połączeniu ze sprawdzaniem krzyżowym albo nawet sprawdzaniem krzyżowym z wyłączeniem jednego elementu (ang. leave-one-out).

Przykład (za mały żeby dlatego teoretyczny)

Dane są w tablice. Jeśli podzielmy dane na testowy ($x = 2$) i treningowe (inne).

Pojemność RAM	Cena
2	12
4	16
8	28
16	62

$$\hat{\beta}_2 = \frac{n\sum_{i=1}^{n} X_i Y_i - \left(\sum_{i=1}^{n} X_i\right)\left(\sum_{i=1}^{n} Y_i\right)}{n\sum_{i=1}^{n} X_i^2 - \left(\sum_{i=1}^{n} X_i\right)^2} = \frac{3(64 + 224 + 992) - 28 \cdot 106}{3(336) - 784}$$

$$\approx 3.89$$

oraz

$$\hat{\beta}_1 = \frac{1}{n}\left(\sum_{i=1}^{n} Y_i - \hat{\beta}_2\left(\sum_{i=1}^{n} X_i\right)\right) = \frac{1}{3}\left(106 - 3.89 \cdot 28\right) \approx -0.97$$

Test: 2*3.89-0.97=6.8

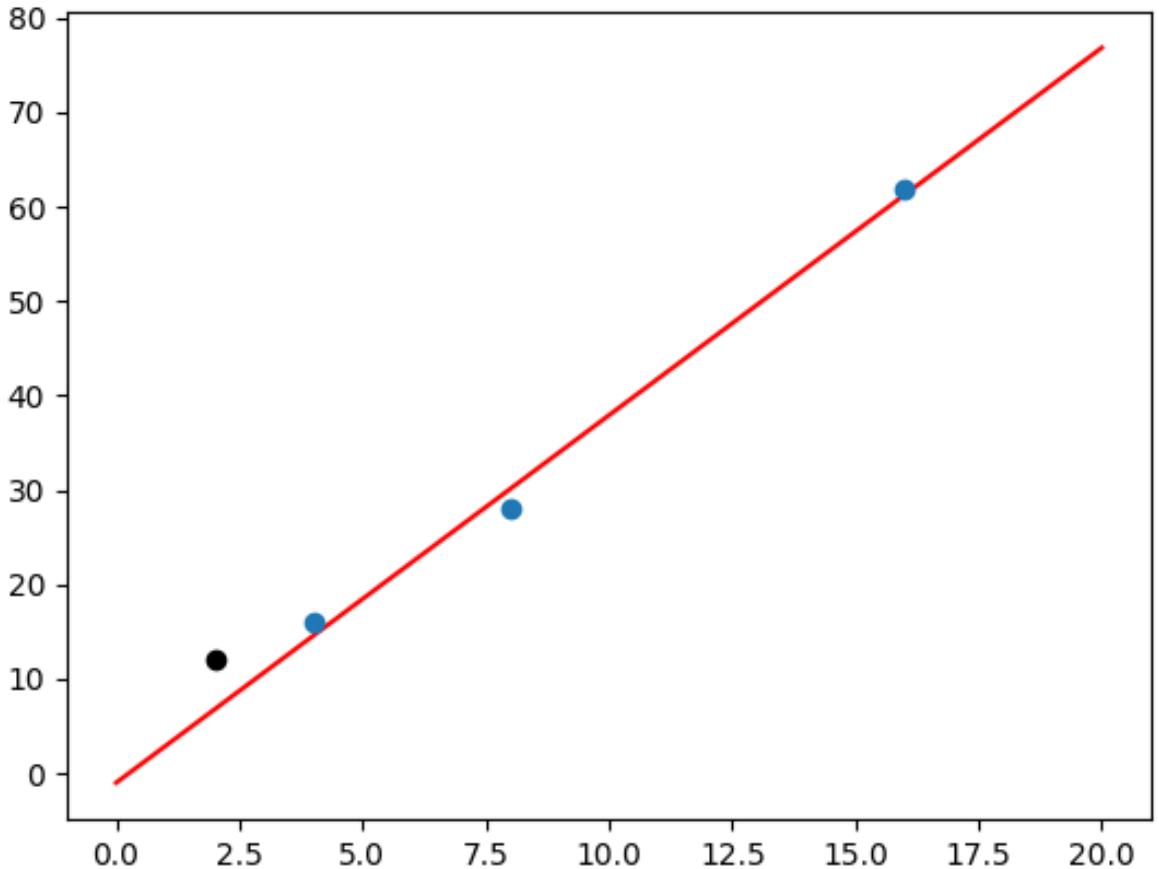
Przykład (za mały żeby dlatego teoretyczny): numpy

```
import numpy as np
import matplotlib.pyplot as plt

RAM = np.array([4, 8, 16])
price = np.array([16, 28, 62])
n = len(RAM)
beta2 = (n*sum(RAM*price) - sum(RAM)*sum(price))/
            (n*sum(RAM**2) - (sum(RAM))**2)
print(beta2)
beta1 = (sum(price)-beta2*sum(RAM))/n
print(beta1)

x = np.linspace(0,20)
y = beta1 + x*beta2
plt.plot(x,y,'r')
plt.plot(RAM, price,'o')
plt.plot([2], [12], color='black', marker='o')
plt.show()
```

Przykład (za mały żeby dlatego teoretyczny): numpy



Regresja liniowa dwóch zmiennych

Szukamy $\beta_0, \beta_1, \beta_2$ dla $f(x, y) = \beta_0 + \beta_1 x + \beta_2 y$ metodą najmnieszich kwadratów:

$$S(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} (f(x_i, y_i) - z_i)^2 \rightarrow min$$

$$S(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} [z_i - f(x_i, y_i)]^2 = \sum_{i=1}^{n} [z_i - (\beta_0 + \beta_1 x_i + \beta_2 y_i)]^2$$

Kiedy suma kwadratów odchyleń S jest najmniejsza?

$$S(\beta_0,\beta_1,\beta_2)=\sum_{i=1}^{n}[z_i-f(x_i,y_i)]^2=\sum_{i=1}^{n}[z_i-(\beta_0+\beta_1x_i+\beta_2y_i)]^2$$

$$\begin{cases}\frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_2}=0\\ \frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_1}=0\\ \frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_0}=0\end{cases}\Leftrightarrow\begin{cases}-2\sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))y_i=0\\ -2\sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))x_i=0\\ -2\sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))=0\end{cases}\Leftrightarrow$$

$$\Leftrightarrow\begin{cases}\sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))y_i=0\\ \sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))x_i=0\\ \sum_{i=1}^{n}(z_i-(\beta_0+\beta_1x_i+\beta_2y_i))=0\end{cases}\Leftrightarrow$$

$$\begin{cases}\beta_2(\sum_{i=1}^{n}y_i^2)+\beta_1(\sum_{i=1}^{n}x_iy_i)+\beta_0(\sum_{i=1}^{n}y_i)=\sum_{i=1}^{n}z_iy_i\\ \beta_2(\sum_{i=1}^{n}x_iy_i)+\beta_1(\sum_{i=1}^{n}x_i^2)+\beta_0(\sum_{i=1}^{n}x_i)=\sum_{i=1}^{n}z_ix_i\\ \beta_2(\sum_{i=1}^{n}y_i)+\beta_1(\sum_{i=1}^{n}x_i)+n\beta_0=\sum_{i=1}^{n}z_i\end{cases}$$

Przykład

x	-1	-1	0	1	2
y	-1	0	1	2	3
z	0	-1	0	1	2

$$\begin{cases} \beta_2(\sum_{i=1}^n y_i^2) + \beta_1(\sum_{i=1}^n x_i y_i) + \beta_0(\sum_{i=1}^n y_i) = \sum_{i=1}^n z_i y_i \\ \beta_2(\sum_{i=1}^n x_i y_i) + \beta_1(\sum_{i=1}^n x_i^2) + \beta_0(\sum_{i=1}^n x_i) = \sum_{i=1}^n z_i x_i \\ \beta_2(\sum_{i=1}^n y_i) + \beta_1(\sum_{i=1}^n x_i) + n\beta_0 = \sum_{i=1}^n z_i \end{cases}$$

$$\begin{cases} 15\beta_2 + 9\beta_1 + 5\beta_0 = 8 \\ 9\beta_2 + 7\beta_1 + \beta_0 = 6 \\ 5\beta_2 + \beta_1 + 5\beta_0 = 2 \end{cases}$$

$$\beta_0 = 1, \quad \beta_1 = 2, \quad \beta_2 = -1,$$

$$z = 1 + 2x - y.$$

Regresja liniowa dla kilka zmiennych niezależnych

Niech dany będzie zbiór danych zaobserwowanych $(Z_i, X_{i1}, \ldots, X_{im})_{i=1}^n$. Model regresji liniowej zakłada, że istnieje liniowa (afiniczna) relacja pomiędzy zmienną zależną Z_i a wektorem X_i. Dokładniej, model ten jest postaci

$$Z_i = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_p X_{im} = \mathsf{X}_i^{\mathsf{T}} \beta, \qquad i = 1, \ldots, n,$$

Powyższe n równań można zapisać w sposób macierzowy: $\mathsf{Z} = \mathsf{X}\beta$, gdzie:

$$\mathsf{Z} = \begin{pmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{pmatrix}, \qquad \mathsf{X} = \begin{pmatrix} \mathsf{X}_1^{\mathsf{T}} \\ \mathsf{X}_2^{\mathsf{T}} \\ \vdots \\ \mathsf{X}_n^{\mathsf{T}} \end{pmatrix} = \begin{pmatrix} 1 & X_{11} & \ldots & X_{1m} \\ 1 & X_{21} & \ldots & X_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n1} & \ldots & X_{nm} \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}.$$

Najczęściej do obliczenia $\hat{\beta}$ wykorzystuje się, jak i w przypadku jednej zmiennej niezależnej, klasyczna metoda najmniejszych kwadratów i jej pochodne.
Dla klasycznej metody najmniejszych kwadratów:

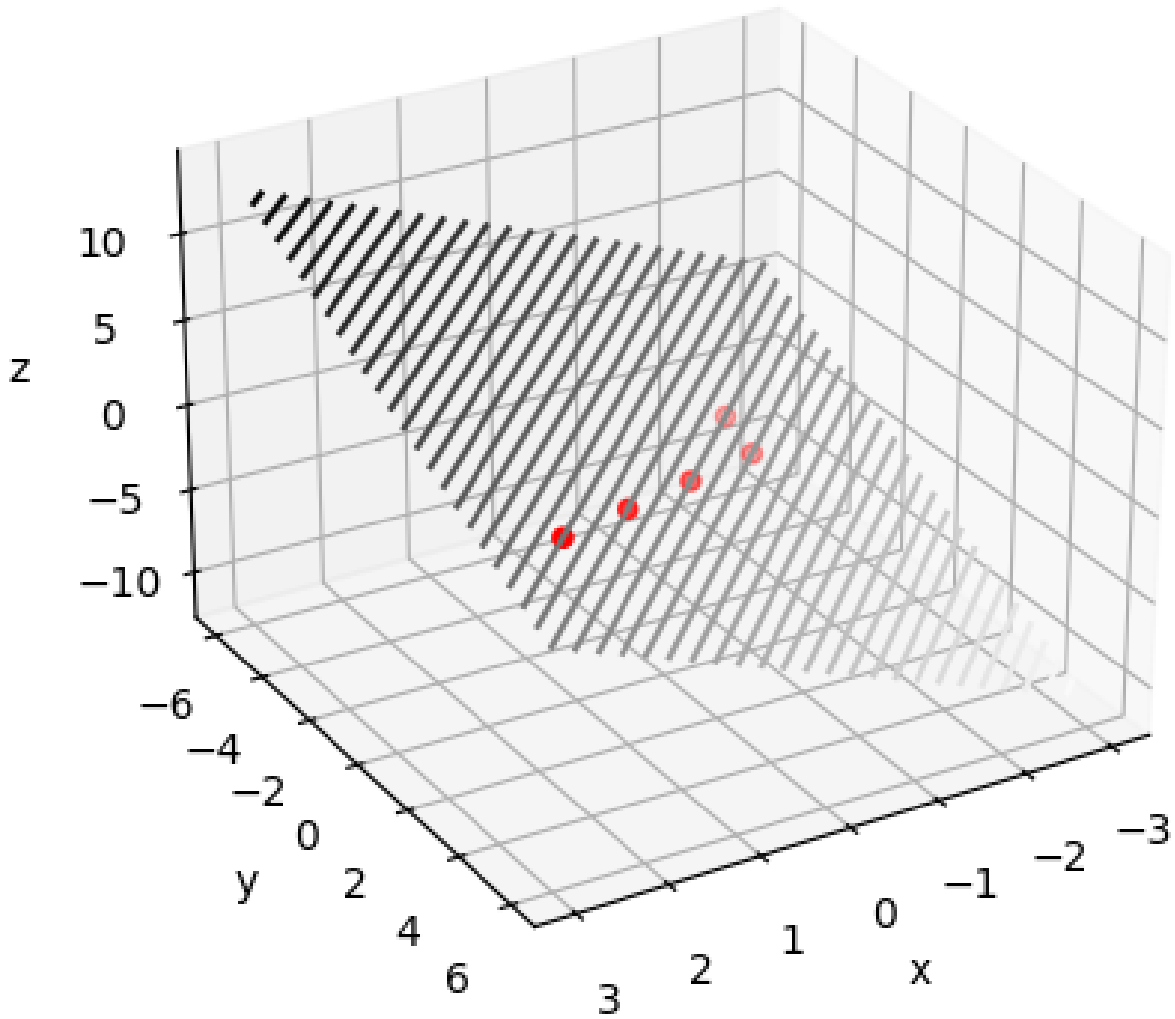
$$\hat{\beta} = \left(\mathsf{X}^{\mathsf{T}}\mathsf{X}\right)^{-1} \mathsf{X}^{\mathsf{T}}\mathsf{Z}.$$

Obliczenia w numpy dla przykładu

```
import numpy as np

xi = np.array([-1,-1,0,1,2])
yi = np.array([-1,0,1,2,3])
zi = np.array([0,-1,0,1,2])
n = len(xi)

X = np.array([np.ones(n),xi,yi]).transpose()
Z = zi.transpose()
beta = np.linalg.inv(X.transpose()@X)@X.transpose()@Z
print(beta)
```


z
10
5
0
−5
−10
y
−6
−4
−2
0
2
4
6
x
3
2
1
0
−1
−2
−3

Kod numpy obrazku 1

```
import numpy as np
import matplotlib.pyplot as plt


xi = np.array([-1,-1,0,1,2])
yi = np.array([-1,0,1,2,3])
zi = np.array([0,-1,0,1,2])
n = len(xi)

def f(x, y):
    return 1+2*x-y

x = np.linspace(-3, 3, 30)
y = np.linspace(-6, 6, 30)

X, Y = np.meshgrid(x, y)
Z = f(X, Y)
```

Kod numpy obrazku 2

```
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.contour3D(X, Y, Z, 50, cmap='binary')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
ax.scatter3D(xi, yi, zi, color='r')
plt.show()
```

Współczynnik determinacji

Współczynnik determinacji R^2 jest zdefiniowane jako

$$R^2 := \frac{\sum_{i=1}^{n}(\hat{Z}_i - \overline{Z})^2}{\sum_{i=1}^{n}(Z_i - \overline{Z})^2} \geq 0,$$

gdzie:

- Z_i – i-ta obserwacja zmiennej Z,
- $\hat{Z}_i$ – wartość teoretyczna zmiennej objaśnianej (na podstawie modelu),
- $\overline{Z}$ – średnia arytmetyczna empirycznych wartości zmiennej objaśnianej.

Nasz przykład:

x	-1	-1	0	1	2
y	-1	0	1	2	3
z	0	-1	0	1	2

$$z = 1 + 2x - y.$$

$\hat{z}$	0	-1	0	1	2

$R^2 = 1.$

Kod numpy (przyklad)

```
import numpy as np

xi = np.array([-1,-1,0,1,2])
yi = np.array([-1,0,1,2,3])
zi = np.array([0,-1,0,1,2])
n = len(xi)

X = np.array([np.ones(n),xi,yi]).transpose()
Z = zi.transpose()
beta = np.linalg.inv(X.transpose()@X)@X.transpose()@Z
print(beta)
Zhat = beta[0]+beta[1]*xi+beta[2]*yi
print(Zhat)
print(np.sum((Zhat-np.mean(zi))**2)/
        np.sum((Z-np.mean(zi))**2))
#0.9999999999999998
```

Ważona metoda najmniejszych kwadratów

Ta metoda (WLS), znana również jako ważona regresja liniowa, to uogólnienie zwykłych metod najmniejszych kwadratów i regresji liniowej, w których do regresji włączana jest wiedza o nierównej wariancji błędu (heteroscedastyczność), przy nieskorelowanych błędach. W tej metodzie dopasowanie linijnej regresji polega na minimalizacji sumy:

$$S(\beta_1, \beta_2) = \sum_{i=1}^{n} W_{ii}[Y_i - f(X_i)]^2 = \sum_{i=1}^{n}[W_{ii}\, Y_i - (\beta_1 + \beta_2 X_i)]^2,$$

gdzie W_{ii} są wagami. Probe z mniejszą wariancją muszą mieć większą wagę, nprz.

$$W_{ii} = \frac{1}{\sigma_i^2}.$$

Uogólniona metoda najmniejszych kwadratów

Uogólniona metoda najmniejszych kwadratów po raz pierwszy została opisana przez Alexandra Aitkena w 1935 roku. Ta metoda używa się, jeżeli błędy są skorelowane. W tej metodzie dopasowanie linijnej regresji polega na minimalizacji

$$S(\beta) = (Y - \beta_1 - \beta_2 X)^{\mathsf{T}} \Omega^{-1} (Y - \beta_1 - \beta_2 X),$$

gdzie Ω jest specjalną macierzą wariancji błędów: $\Omega = \mathrm{Cov}(\varepsilon|X)$.

(nie wchodzimy w szczegóły)

Regresja wielomianowa

Niech mamy dwie cechy x i y. Regresja jest metodą budowania krzywej zależności Y od X, na podstawie ich wartości w probie, $X_1, \ldots, X_n$ oraz $Y_1, \ldots, Y_n$. Założenie modelu regresji wielomianowej:

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \cdots + \beta_m X_i^m \ (i = 1, 2, \ldots, n)$$

gdzie $\beta_1, \beta_2, \ldots \beta_m$ są parametrami.

Współczynnik determinacji

Współczynnik determinacji R^2 jest zdefiniowane jako

$$R^2 := \frac{\sum_{i=1}^{n}(\hat{Y}_i - \overline{Y})^2}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2} \geq 0,$$

gdzie:

- Y_i – i-ta obserwacja zmiennej Y,
- $\hat{Y}_i$ – wartość teoretyczna zmiennej objaśnianej (na podstawie modelu),
- $\overline{y}$ – średnia arytmetyczna empirycznych wartości zmiennej objaśnianej.

Metoda najmniejszych kwadratów dla $m = 2$

Szukamy parabolą:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

Znaleźć parametry $\beta_0, \beta_1 \beta_2$ parabolę $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2$ metodą najmnieszich kwadratów.

$$S(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} [y_i - f(x_i)]^2 = \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x + \beta_2 x^2)]^2$$

Kiedy suma kwadratów odchyleń S jest najmniejsza?

$$S(\beta_0,\beta_1,\beta_2)=\sum_{i=1}^{n}[y_i-f(x_i)]^2=\sum_{i=1}^{n}[y_i-(\beta_0+\beta_1x+\beta_2x^2)]^2$$

$$\begin{cases}\frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_2}=0\\ \frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_1}=0\\ \frac{\partial S(\beta_0,\beta_1,\beta_2)}{\partial\beta_0}=0\end{cases}\Leftrightarrow\begin{cases}-2\sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))x_i^2=0\\ -2\sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))x_i=0\\ -2\sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))=0\end{cases}\Leftrightarrow$$

$$\Leftrightarrow\begin{cases}\sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))x_i^2=0\\ \sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))x_i=0\\ \sum_{i=1}^{n}(y_i-(\beta_2x_i^2+\beta_1x_i+\beta_0))=0\end{cases}\Leftrightarrow$$

$$\begin{cases}\beta_2(\sum_{i=1}^{n}x_i^4)+\beta_1(\sum_{i=1}^{n}x_i^3)+\beta_0(\sum_{i=1}^{n}x_i^2)=\sum_{i=1}^{n}y_ix_i^2\\ \beta_2(\sum_{i=1}^{n}x_i^3)+\beta_1(\sum_{i=1}^{n}x_i^2)+\beta_0(\sum_{i=1}^{n}x_i)=\sum_{i=1}^{n}y_ix_i\\ \beta_2(\sum_{i=1}^{n}x_i^2)+\beta_1(\sum_{i=1}^{n}x_i)+n\beta_0=\sum_{i=1}^{n}y_i\end{cases}$$

Przykład

Załóżmy, że mamy pięć punktów doświadczalnych danych w tabeli:

x	-2	-1	0	1	2
y	5	2	1	2	4

Znaleźć parametry $\beta_0, \beta_1, \beta_2$ parabolę metodą najmnieszich kwadratów.

$$\begin{cases} \beta_2(\sum_{i=1}^n x_i^4) + \beta_1(\sum_{i=1}^n x_i^3) + \beta_0(\sum_{i=1}^n x_i^2) = \sum_{i=1}^n y_i x_i^2 \\ \beta_2(\sum_{i=1}^n x_i^3) + \beta_1(\sum_{i=1}^n x_i^2) + \beta_0(\sum_{i=1}^n x_i) = \sum_{i=1}^n y_i x_i \\ \beta_2(\sum_{i=1}^n x_i^2) + \beta_1(\sum_{i=1}^n x_i) + n\beta_0 = \sum_{i=1}^n y_i \end{cases}$$

$$\begin{cases} \beta_2 \cdot 34 + \beta_1 \cdot 0 + \beta_0 \cdot 10 = 40 \\ \beta_2 \cdot 0 + \beta_1 \cdot 10 + \beta_0 \cdot 0 = -2 \\ \beta_2 \cdot 10 + \beta_1 \cdot 0 + 5\beta_0 = 14 \end{cases}$$

$\beta_0 = 38/35 \approx 1.08, \beta_1 = -1/5 = -0.2, \beta_2 = 6/7 \approx 0.85$

Współczynnik determinacji: przykład

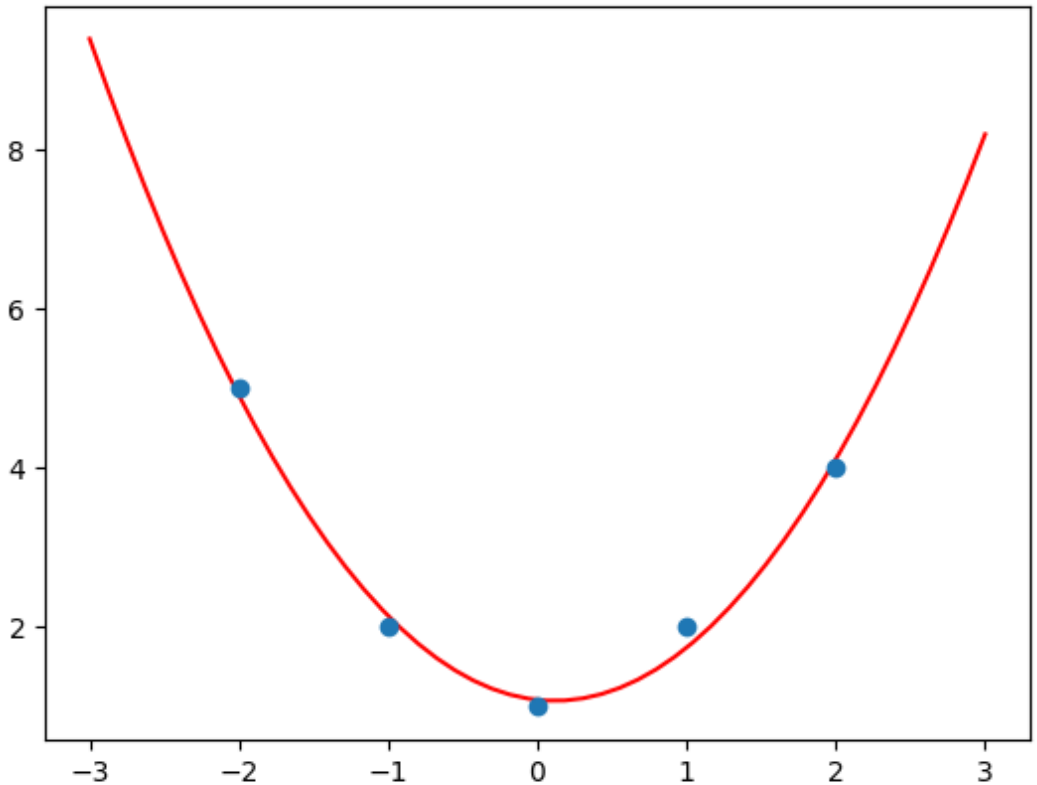
x	-2	-1	0	1	2
y	5	2	1	2	4
$\hat{y}$	4.9	2.1	1.1	1.7	4.1

$\beta_0 = 38/35 \approx 1.08, \beta_1 = -1/5 = -0.2, \beta_2 = 6/7 \approx 0.85$
Współczynnik determinacji R^2 jest zdefiniowane jako

$$R^2 := \frac{\sum_{i=1}^{n}(\hat{Y}_i - \overline{Y})^2}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2} = 0.98$$

Współczynnik zbieżności ϕ^2 jest zdefiniowane jako

$$\phi^2 = 1 - R^2.$$


8
6
4
2
−3
−2
−1
0
1
2
3

Kod numpy obrazku

```
import numpy as np
import matplotlib.pyplot as plt

beta0=38/35
beta1 = -1/5
beta2=6/7
print(beta0, beta1, beta2)

xi = np.array([-2,-1,0,1,2])
yi = np.array([5,2,1,2,4])
n = len(xi)
x = np.linspace(-3,3)
plt.plot(x,beta0 + x*beta1 + x**2*beta2,'r')
plt.plot(xi, yi,'o')
plt.show()
```

Regresja wielomianowa: postać macierzowa

Niech mamy dwie cechy x i y. Regresja jest metodą budowania krzywej zależności Y od X, na podstawie ich wartości w probie, $X_1, \ldots, X_n$ oraz $Y_1, \ldots, Y_n$. Założenie modelu regresji wielomianowej:

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \cdots + \beta_m X_i^m \ (i = 1, 2, \ldots, n)$$

gdzie $\beta_1, \beta_2, \ldots \beta_m$ są parametrami.
Model można zapisać w postaci macierzowej:

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_n \end{pmatrix} = \begin{pmatrix} 1 & X_1 & X_1^2 & \ldots & X_1^m \\ 1 & X_2 & X_2^2 & \ldots & X_2^m \\ 1 & X_3 & X_3^2 & \ldots & X_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_n & X_n^2 & \ldots & X_n^m \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{pmatrix},$$

tzn. $\mathrm{Y} = \mathrm{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$. i używając klasyczną metodą najmniejszych kwadratów dla wielu zmiennych otrzymujemy

$$\hat{\boldsymbol{\beta}} = \left(\mathrm{X}^{\mathsf{T}}\mathrm{X}\right)^{-1} \mathrm{X}^{\mathsf{T}}\mathrm{Y}.$$

zakładając $m < n$, co jest wymagane, aby macierz była odwracalna; wówczas, ponieważ X jest macierzą Vandermonde'a, warunek odwracalności jest spełniony, jeśli wszystkie wartości x_i są różne. Wtedy rozwiązanie metodą najmniejszych kwadratów jest unikalne.

Obliczenia przykładu w numpy

```
import numpy as np
import matplotlib.pyplot as plt


xi = np.array([-2,-1,0,1,2])
yi = np.array([5,2,1,2,4])
n = len(xi)

X = np.array([np.ones(n),xi,xi**2]).transpose()
print(X)
Y = yi.transpose()
beta = np.linalg.inv(X.transpose()@X)@X.transpose()@Y
print(beta)

#lub latwiej
print(np.polyfit(xi, yi, 2))
```

Kod numpy współczynnika determinacji

```
import numpy as np

xi = np.array([-2,-1,0,1,2])
yi = np.array([5,2,1,2,4])
n = len(xi)

beta = np.polyfit(xi, yi, 2)[::-1]

yhat = beta[0]+beta[1]*xi+beta[2]*xi**2
print(yhat)
print(np.sum((yhat-np.mean(yi))**2)
       /np.sum((yi-np.mean(yi))**2))
```