Giới thiệu
Hồi qui tuyến tính

Prerequisites:

- Khái niệm cơ bản về Xác suất thống kê

Created by:
Le Ngoc Thach

Learning is to build your own assets.

Nội dung

- ✓ Mục tiêu
- ✓ Ý tưởng
- ✓ Ước tính tham số
- ✓ Ví dụ với Python

Mục tiêu
của hồi qui tuyến tính

Các phân tích chính

- ✓ Analysis of difference (khác biệt)
- ✓ Association analysis (liên quan)
- ✓ **Correlation analysis & prediction (tương quan và tiên lượng)**

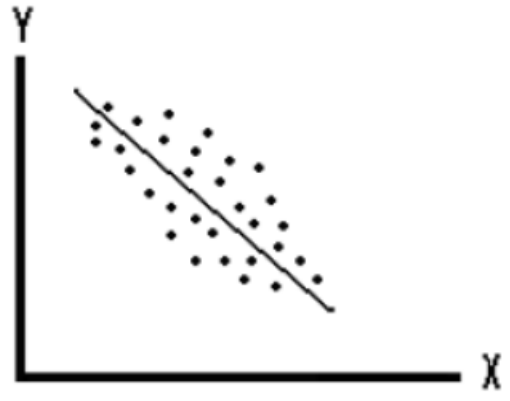
Hồi qui tuyến tính

Phân tích tương quan
(Correlation analysis)

Khái niệm tương quan (correlation)

- ✓ Khi hai biến số (x và y) có liên quan với nhau
- ✓ Mối liên quan có thể cùng chiều hay nghịch đảo
- ✓ Ví dụ: mối liên quan giữa tiêu thụ chocolate và giải Nobel (?)

Positive Correlation



Negative Correlation

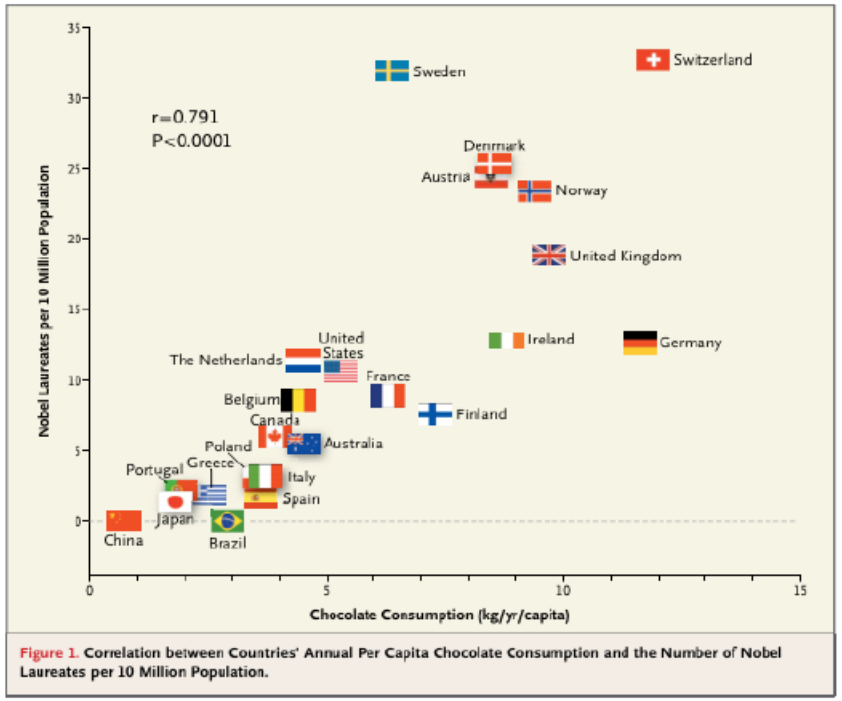
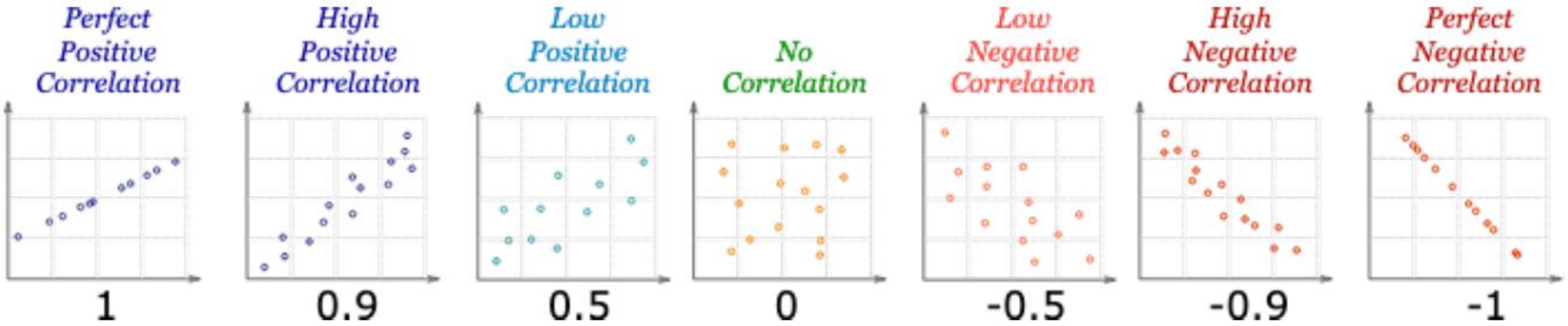


Figure 1. Correlation between Countries' Annual Per Capita Chocolate Consumption and the Number of Nobel Laureates per 10 Million Population.

Tương quan giữa 2 biến liên tục



Làm sao định lượng mối liên quan?

Làm thế nào để mô tả mối tương quan tuyến tính?

- Gọi X và Y là 2 biến ngẫu nhiên từ n quan sát
- Đo lường độ biến thiên: **phương sai (variance)**

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

- Chúng ta cần một thước đo độ "hiệp biến" giữa X và Y
- Covariance là trung bình của tích số X và Y

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

Ước tính hệ số tương quan

- Covariance có đơn vị đo lường (X * Y).
- Coefficient of correlation (r) giữa X và Y là một standardized covariance – không có đơn vị đo lường
- r định nghĩa như sau:

$$r = \frac{\mathrm{cov}(x, y)}{\sqrt{\mathrm{var}(x) \times \mathrm{var}(y)}} = \frac{\mathrm{cov}(x, y)}{SD_x \times SD_y}$$

Obesity data (Vietnam)

- ✓ Nghiên cứu cắt ngang >1100 nam và nữ (Việt Nam)
- ✓ Mục tiêu: ước tính hệ số tương quan giữa tỉ trọng cơ thể (bmi) và tỉ trọng mỡ (pcfat)

```
import pandas as pd

data = pd.read_csv('https://thachln.github.io/datasets/obesity.csv')

data.head()
```

	id	gender	height	weight	bmi	age	bmc	bmd	fat	lean	pcfat
0	1	F	150	49	21.8	53	1312	0.88	17802	28600	37.3
1	2	M	165	52	19.1	65	1309	0.84	8381	40229	16.8
2	3	F	157	57	23.1	64	1230	0.84	19221	36057	34.0
3	4	F	156	53	21.8	56	1171	0.80	17472	33094	33.8
4	5	M	160	51	19.9	54	1681	0.98	7336	40621	14.8

Pandas dataframe.corr()

- Với k biến, chúng ta có k(k – 1)/2 hệ số tương quan
- Mục tiêu: tính toán tất cả mối tương quan
- data.corr()

	id	height	weight	...	fat	lean	pcfat
id	1.000000	-0.075752	0.002212	...	0.035708	-0.018045	0.043131
height	-0.075752	1.000000	0.597667	...	-0.063494	0.768167	-0.479821
weight	0.002212	0.597667	1.000000	...	0.575840	0.840478	0.056556
bmi	0.057696	-0.016403	0.786917	...	0.769213	0.453855	0.440918
age	0.051628	-0.373116	-0.048198	...	0.217547	-0.194388	0.307100
bmc	-0.012571	0.691945	0.610006	...	0.007272	0.739428	-0.403526
bmd	0.049471	0.381603	0.340560	...	-0.060155	0.434807	-0.304407
fat	0.035708	-0.063494	0.575840	...	1.000000	0.115051	0.824162
lean	-0.018045	0.768167	0.840478	...	0.115051	1.000000	-0.445575
pcfat	0.043131	-0.479821	0.056556	...	0.824162	-0.445575	1.000000

[10 rows x 10 columns]

Pandas dataframe.corr()

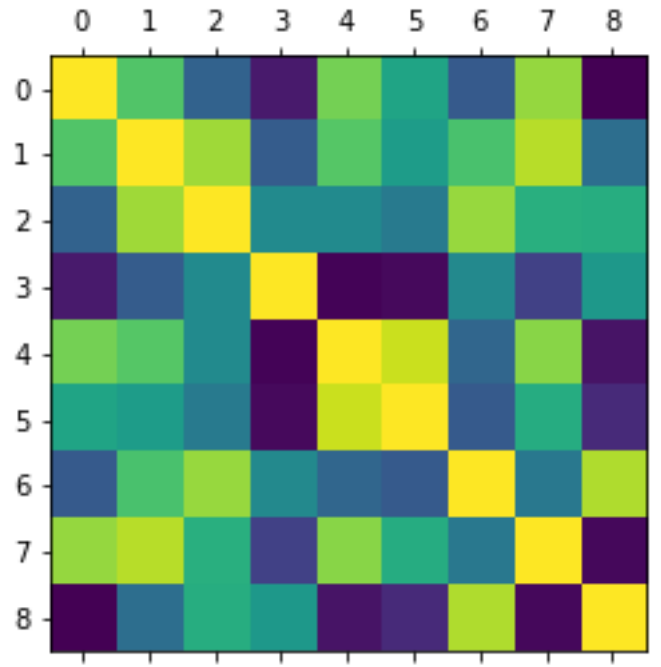
- ✓ data.loc[:, data.columns != 'id'].corr()

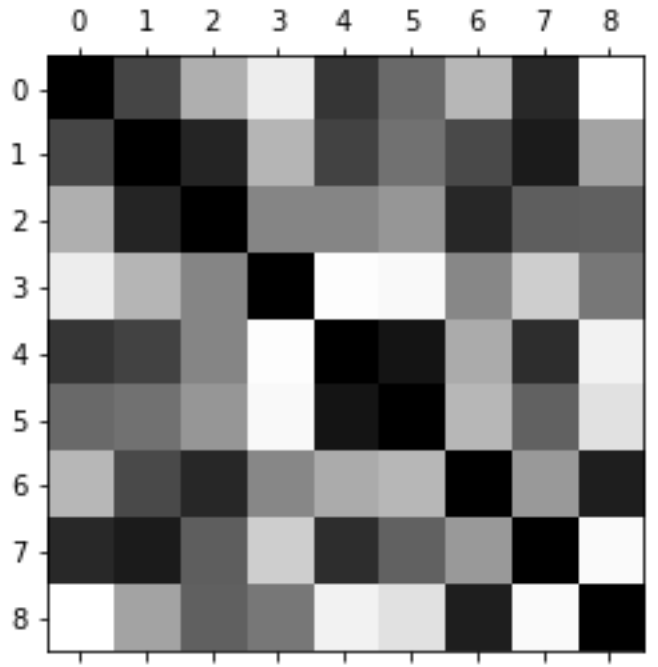
	height	weight	bmi	...	fat	lean	pcfat
height	1.000000	0.597667	-0.016403	...	-0.063494	0.768167	-0.479821
weight	0.597667	1.000000	0.786917	...	0.575840	0.840478	0.056556
bmi	-0.016403	0.786917	1.000000	...	0.769213	0.453855	0.440918
age	-0.373116	-0.048198	0.228628	...	0.217547	-0.194388	0.307100
bmc	0.691945	0.610006	0.225461	...	0.007272	0.739428	-0.403526
bmd	0.381603	0.340560	0.128636	...	-0.060155	0.434807	-0.304407
fat	-0.063494	0.575840	0.769213	...	1.000000	0.115051	0.824162
lean	0.768167	0.840478	0.453855	...	0.115051	1.000000	-0.445575
pcfat	-0.479821	0.056556	0.440918	...	0.824162	-0.445575	1.000000

[9 rows x 9 columns]

Visualization

- ✓ corr_data = data.loc[:, data.columns != 'id'].corr()
- ✓ import matplotlib.pyplot as plt
- ✓ plt.matshow(corr_data)
- ✓ plt.show()

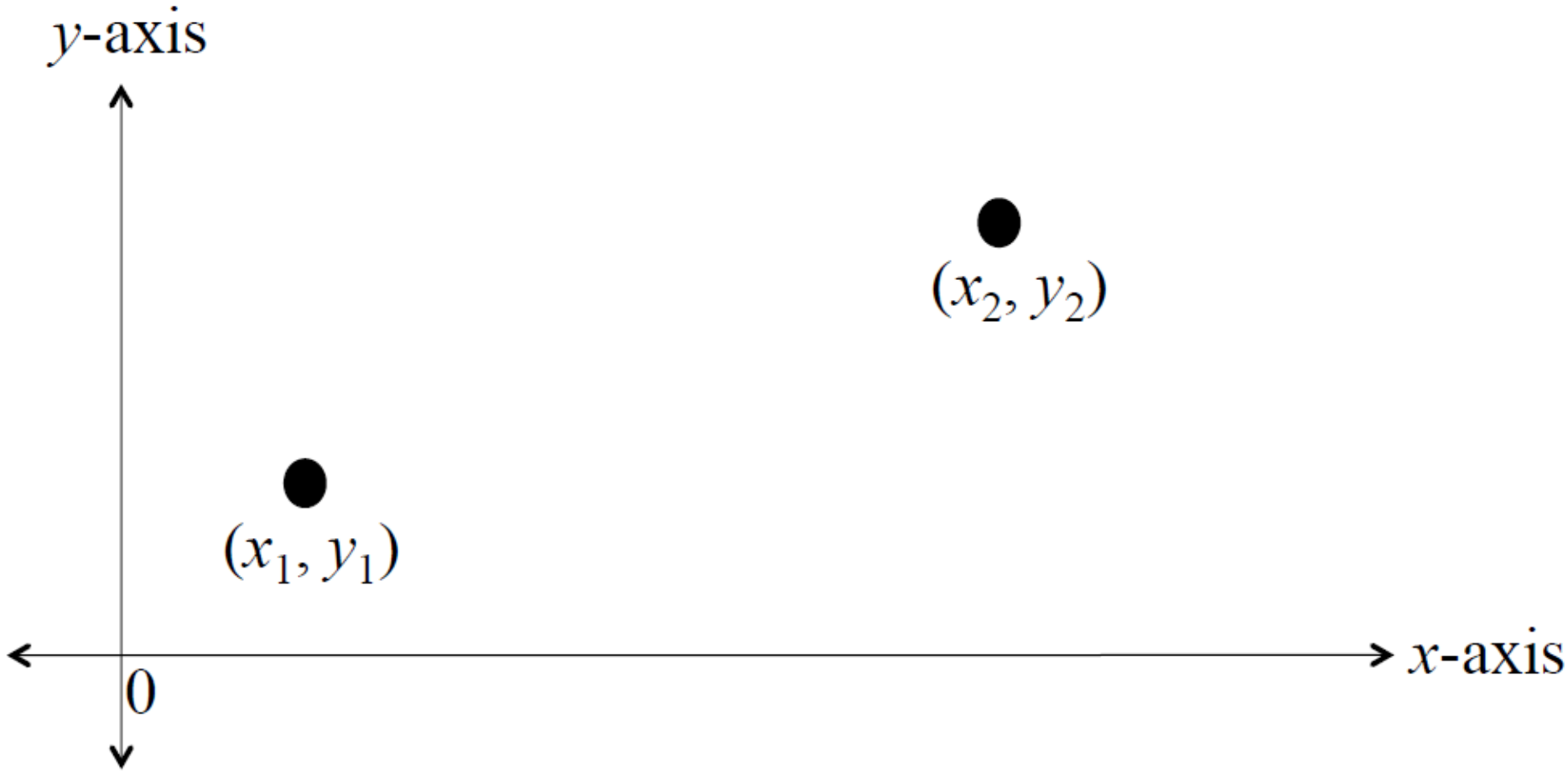




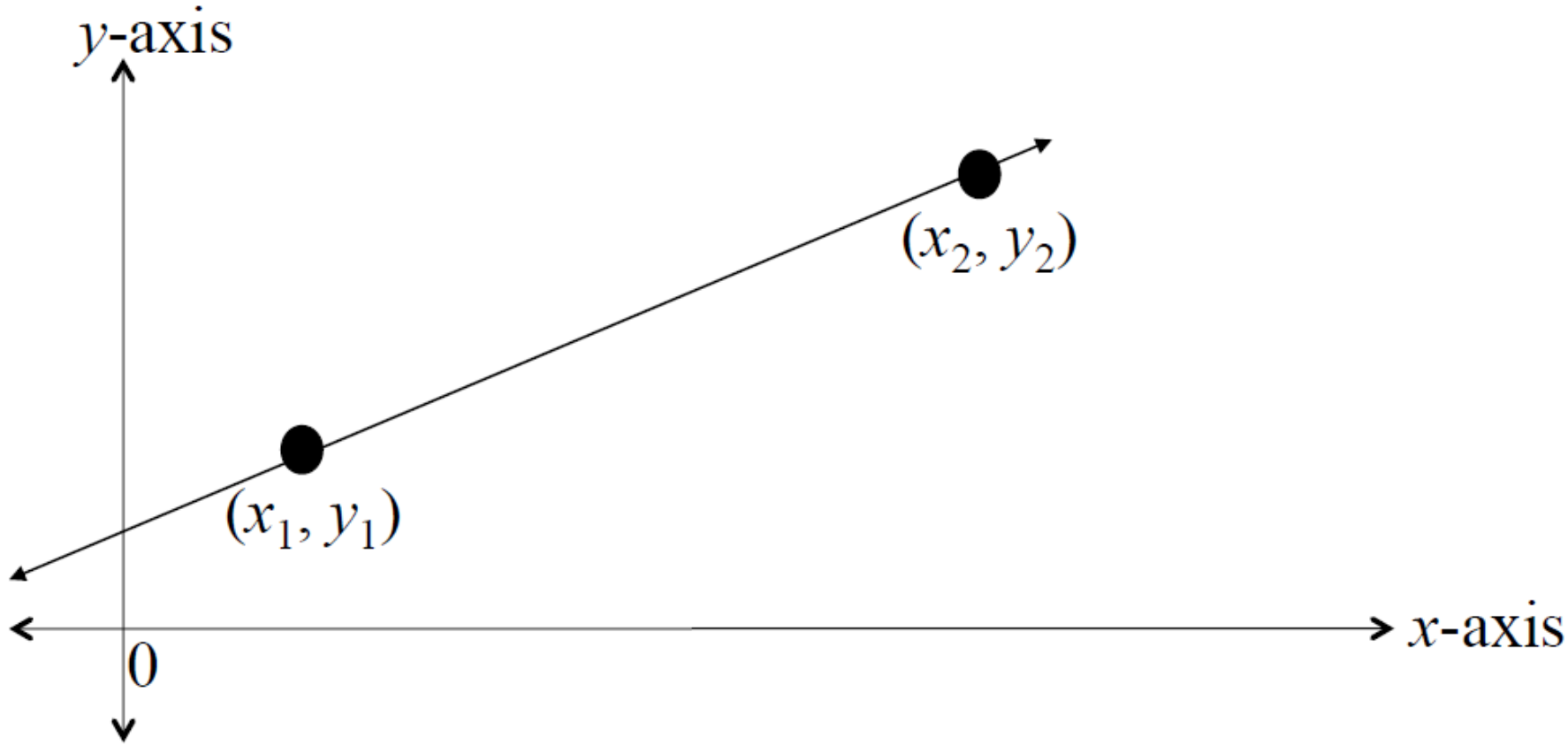
- ✓ plt.matshow(corr_data, cmap=plt.cm.gray_r)

Ý tưởng đằng sau mô hình hồi qui tuyến tính

Cho hai điểm (x_1, y_1) và (x_2, y_2)



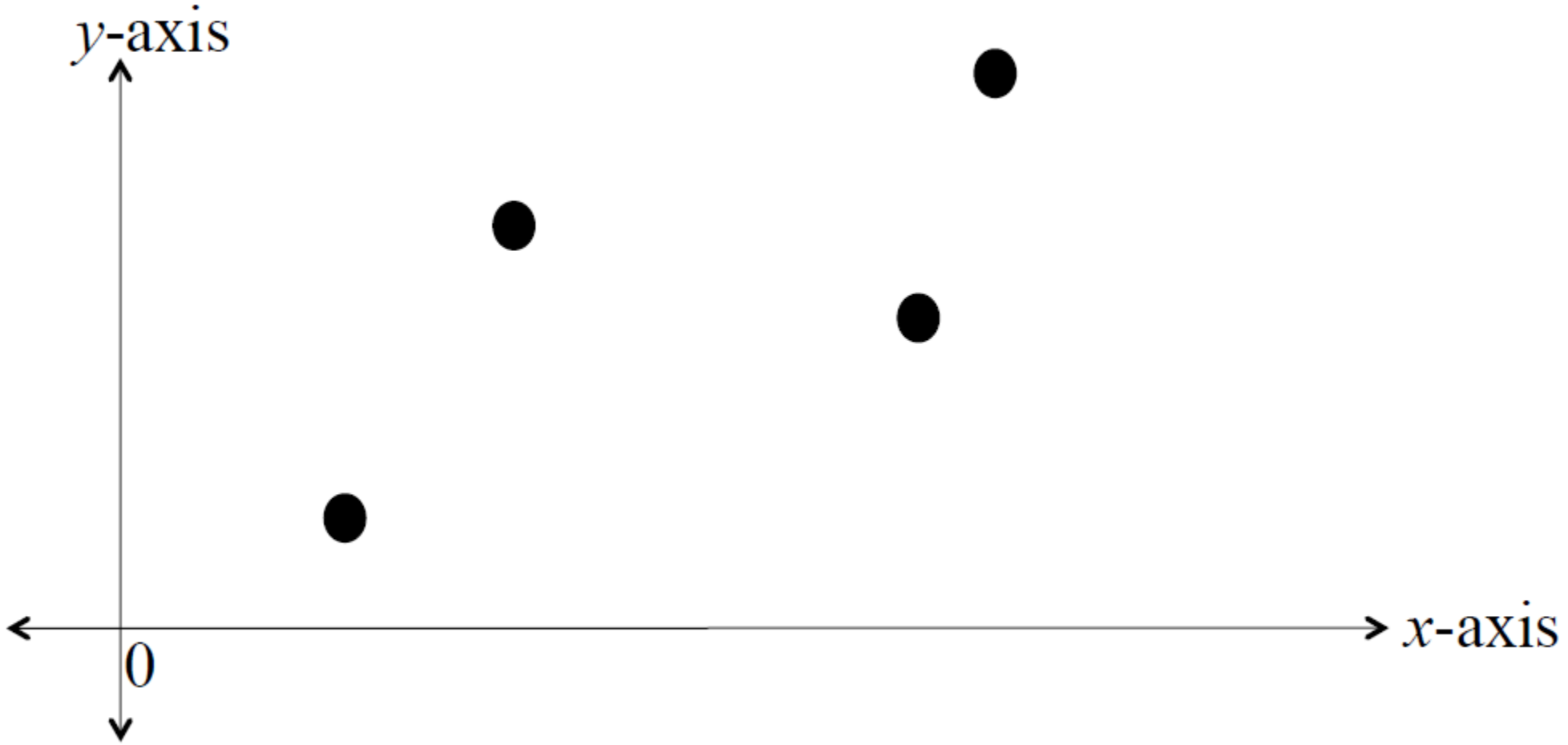
Làm sao để "phát triển" một phương trình nối 2 điểm này?



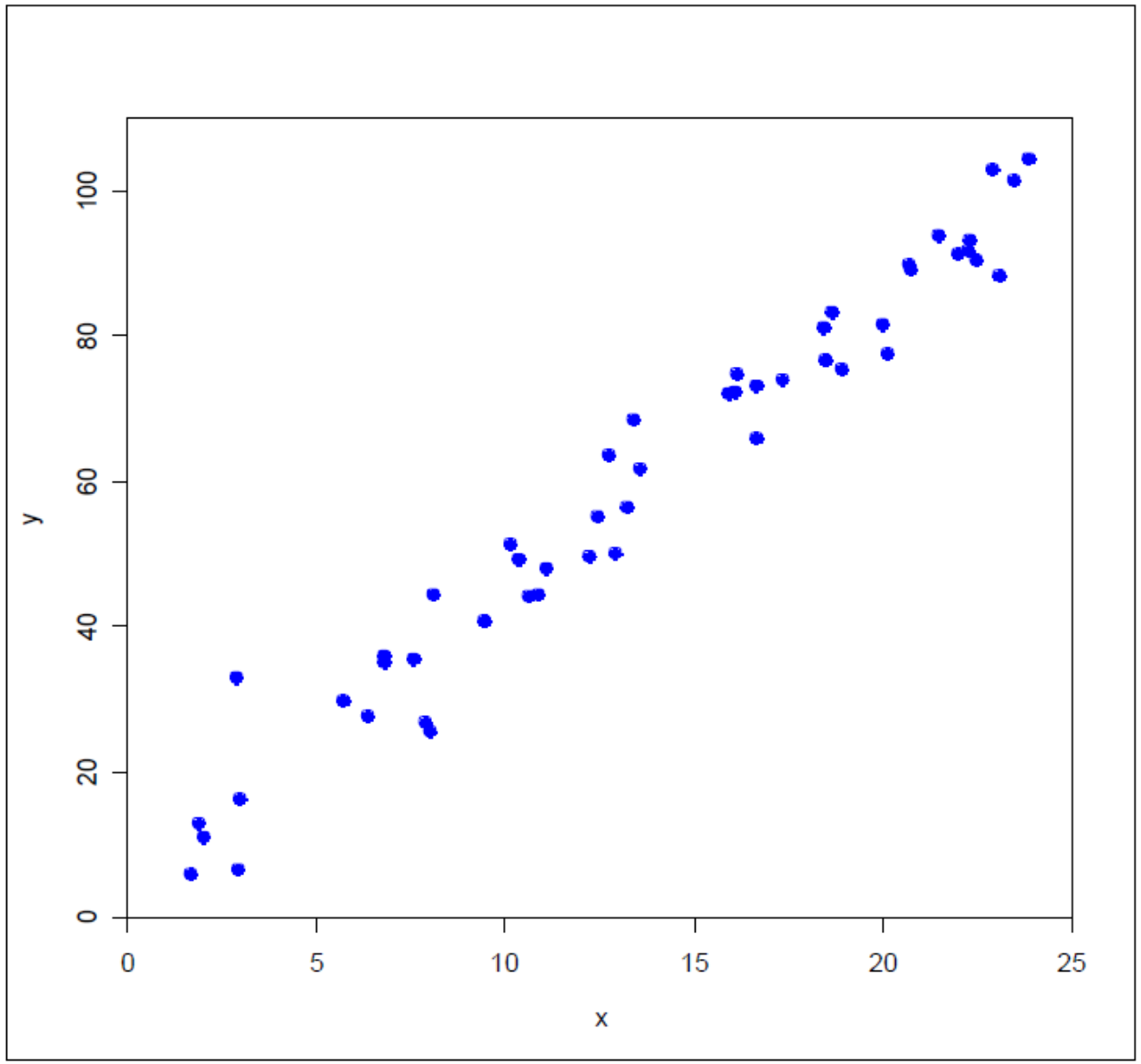
$$slope = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$$

- Tìm gradient (slope)
- Tìm giá trị khởi đầu (intercept) của y khi x=0

Khi có nhiều điểm thì sao?



Và rất rất nhiều điểm



Mô hình hồi qui tuyến tính

- **Simple linear regression model**
- ***Y*** **-** biến phụ thuộc (response variable, dependent variable, v.v.)
  - *Y* là biến liên tục
- ***X*** - biến độc lập (predictor variable, independent variable, v.v.)
  - *X* là biến liên tục hay không liên tục

Mô hình hồi qui tuyến tính

- Mô hình:

$$Y = \alpha + \beta X + \varepsilon$$

α : intercept

β : slope / gradient

ε : sai số ngẫu nhiên (random error – những dao động về Y trong mỗi giá trị X)

Giả định

- Mối liên quan giữa X và Y là tuyến tính (linear) về *tham số*
- *X* không có sai số ngẫu nhiên
- Giá trị của *Y* độc lập với nhau (vd, Y_1 không liên quan với Y_2) ;
- Sai số ngẫu nhiên (e): phân bố chuẩn, trung bình 0, phương sai bất biến

$$\varepsilon \sim N(0, \sigma^2)$$

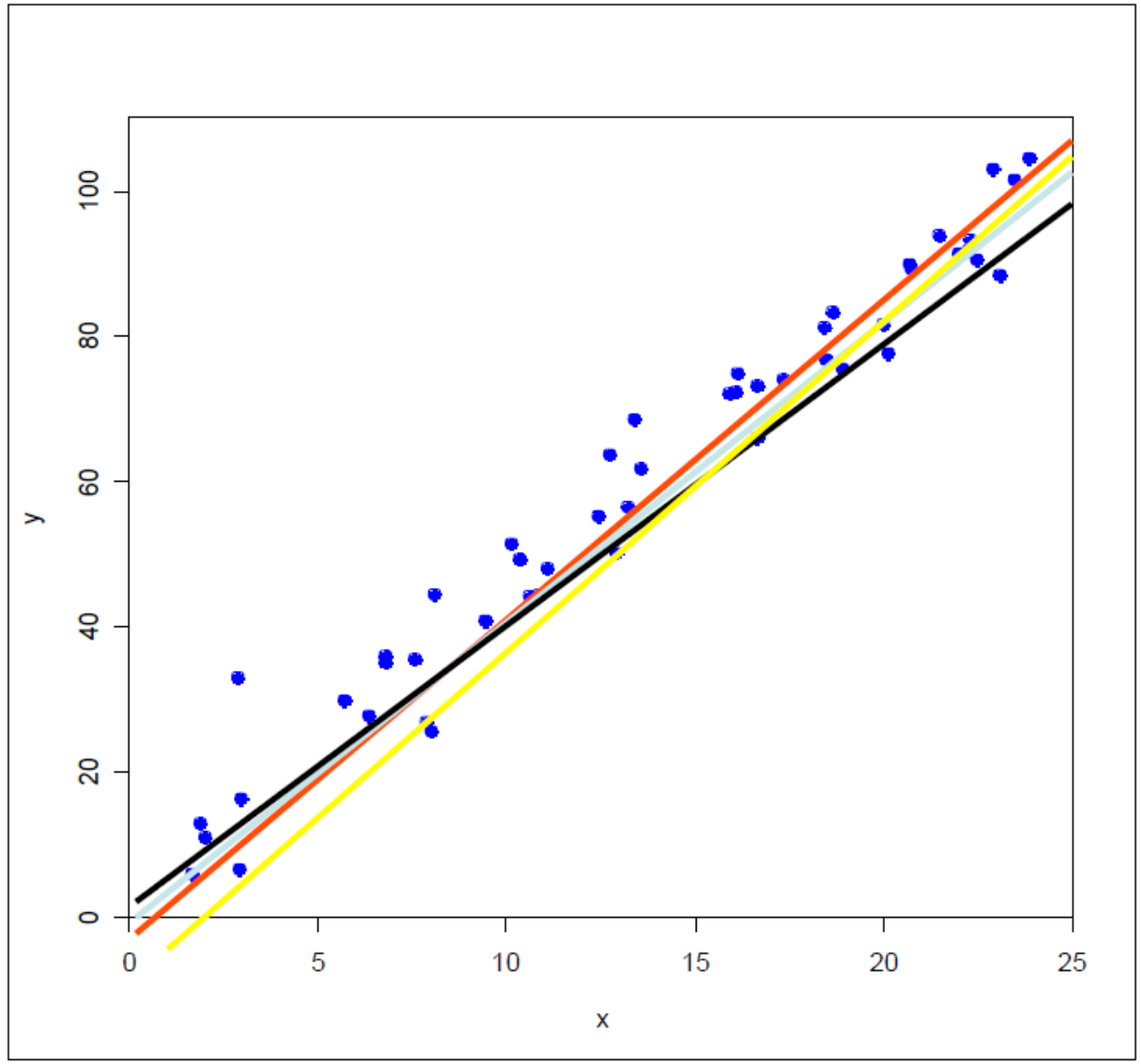
Ước tính tham số (Parameters)

Mục tiêu

- Mô hình

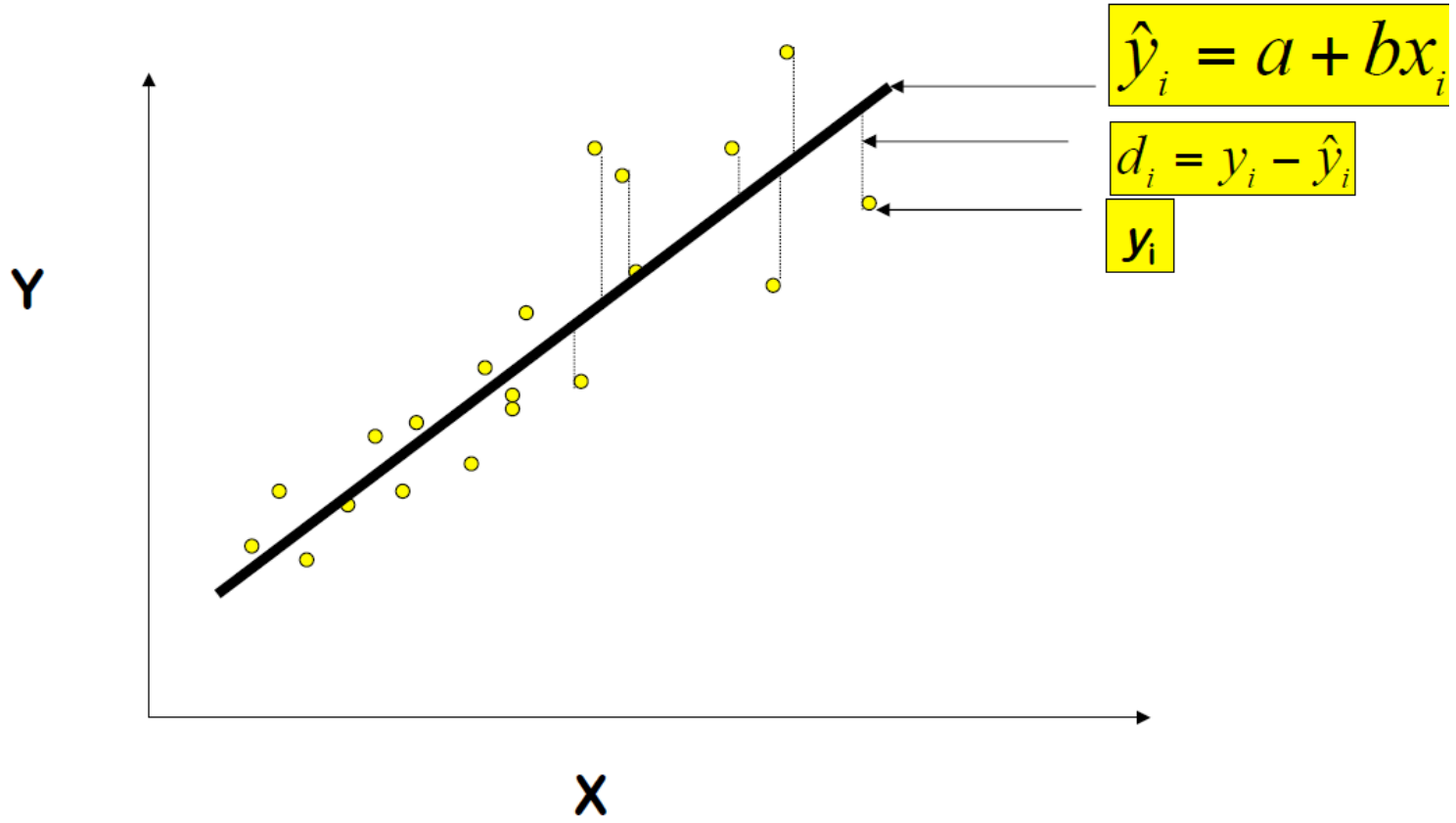
$$Y = \alpha + \beta X + \varepsilon$$

- Chúng ta không biết α và β
- Nhưng có thể dùng dữ liệu thí nghiệm / thực tế để ước tính 2 tham số đó
- Ước số (estimate) của α và β là *a* và *b*



- Có thể tính bằng mắt
- Nhưng không khách quan (biased)
- Chúng ta cần sự nhất quán -- consistency

Tiêu chuẩn để tìm tham số



Tìm công thức (estimator) để tính a và b sao cho tổng d^2 là nhỏ nhất →

Least square method = Bình phương nhỏ nhất

Ước tính bằng Python

Dữ liệu tuyển thủ Nam của đội bóng đá VN

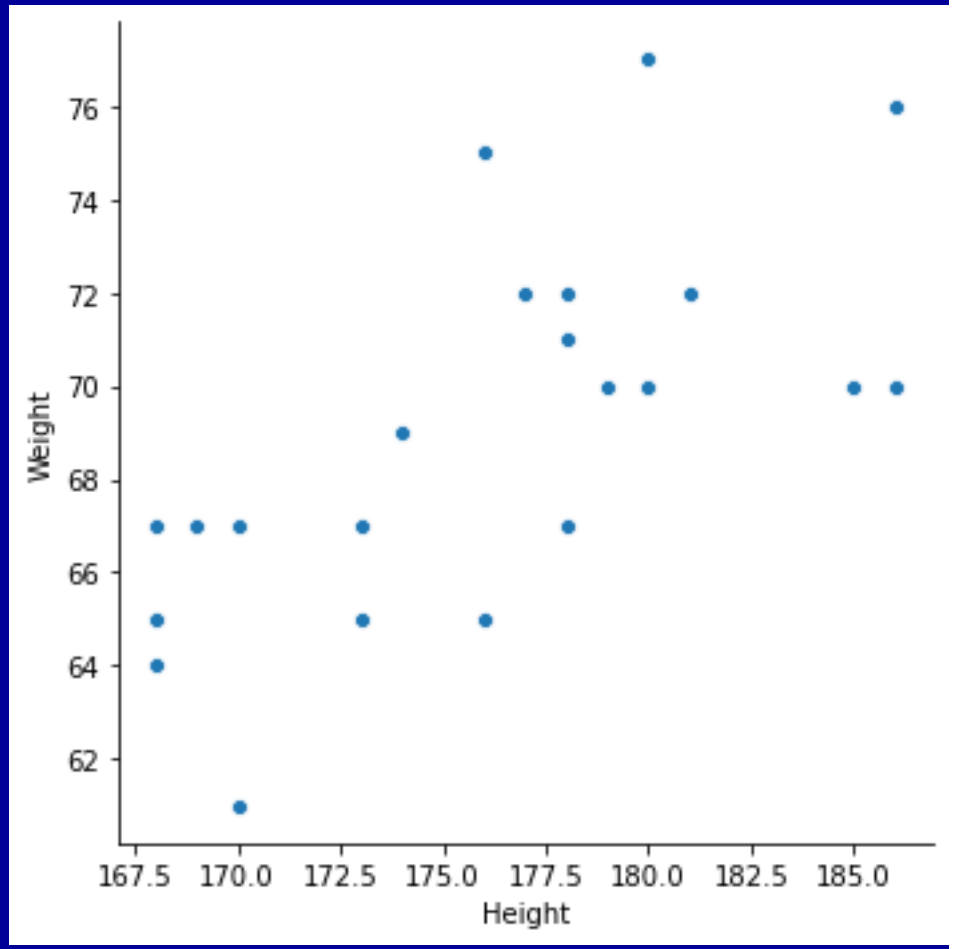
```
import pandas as pd
df = pd.read_csv('https://thachln.github.io/datasets/TuyenVN_2019.csv')
df.columns
df.head()
```

data - DataFrame

Index	id	gender	height	weight	bmi	age	bmc	bmd	fat	lean	pcfat
0	1	F	150	49	21.8	53	1312	0.88	17802	28600	37.3
1	2	M	165	52	19.1	65	1309	0.84	8381	40229	16.8
2	3	F	157	57	23.1	64	1230	0.84	19221	36057	34
3	4	F	156	53	21.8	56	1171	0.8	17472	33094	33.8
4	5	M	160	51	19.9	54	1681	0.98	7336	40621	14.8
5	6	F	153	47	20.1	52	1358	0.91	14904	30068	32.2
6	7	F	155	58	24.1	66	1546	0.96	20233	35599	35.3
7	8	M	167	65	23.3	50	2276	1.11	17749	43301	28
8	9	M	165	54	19.8	61	1778	0.96	10795	38613	21.1
9	10	F	158	60	24	58	1404	0.86	21365	35534	36.6
10	11	F	155	48	20	36	1889	1.06	13458	32261	28.3
11	12	M	165	65	23.9	50	1878	0.97	18100	43391	28.6
12	13	F	155	40	16.6	78	1001	0.78	11865	26264	30.3
13	14	M	158	57	22.8	84	1772	1.02	13398	40313	24.1
14	15	F	154	59	24.9	43	1717	1.03	19154	37239	33
15	16	M	150	70	31.1	49	2084	1	16540	49512	24.3

Dữ liệu tuyển thủ Nam của đội bóng đá VN - Plot dữ liệu

```
import pandas as pd
df = pd.read_csv('https://thachln.github.io/datasets/TuyenVN_2019.csv')

# Quan sát liên quan giữa Chiều cao và Cân nặng
import seaborn as sns
sns.relplot(x="Height", y="Weight", data=df)
```



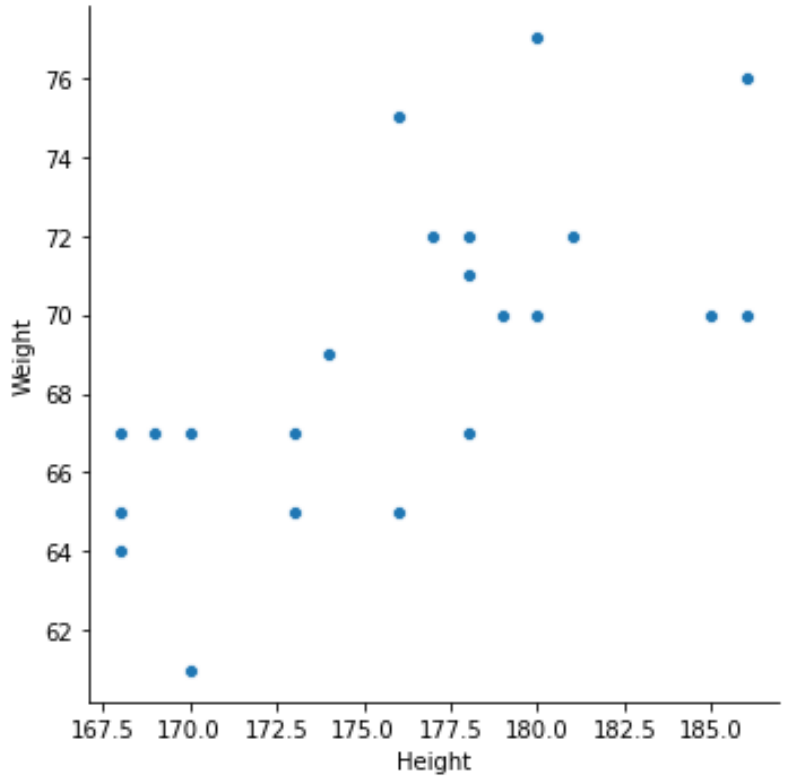
Tìm hệ số coefficient và intercept

```
import pandas as pd
df = pd.read_csv('https://thachln.github.io/datasets/TuyenVN_2019.csv')

model = LinearRegression()
model.fit(df[['Height']], df[['Weight']])

LinearRegression(copy_X = True, fit_intercept = True, n_jobs = None, normalize = False)

intercept = model.intercept_
print('intercept = {}'.format(intercept))

coefficent = model.coef_
print('coefficient = {}'.format(coefficent))
```

```
intercept = [-14.57925878]
coefficient = [[0.47474584]]
```

Code đầy đủ

```
import pandas as pd
import seaborn as sns
from sklearn.linear_model import LinearRegression

df = pd.read_csv('https://thachln.github.io/datasets/TuyenVN_2019.csv')
df.columns
sns.relplot(x="Height", y="Weight", data=df)

model = LinearRegression()
model.fit(df[['Height']], df[['Weight']])

LinearRegression(copy_X = True, fit_intercept = True, n_jobs = None, normalize = False)

intercept = model.intercept_
print('intercept = {}'.format(intercept))

coefficent = model.coef_
print('coefficient = {}'.format(coefficent))
```



Diễn giải kết quả

- ✓ intercept = [-14.57925878]
- ✓ coefficient = [[0.47474584]]

- ✓ Phương trình:

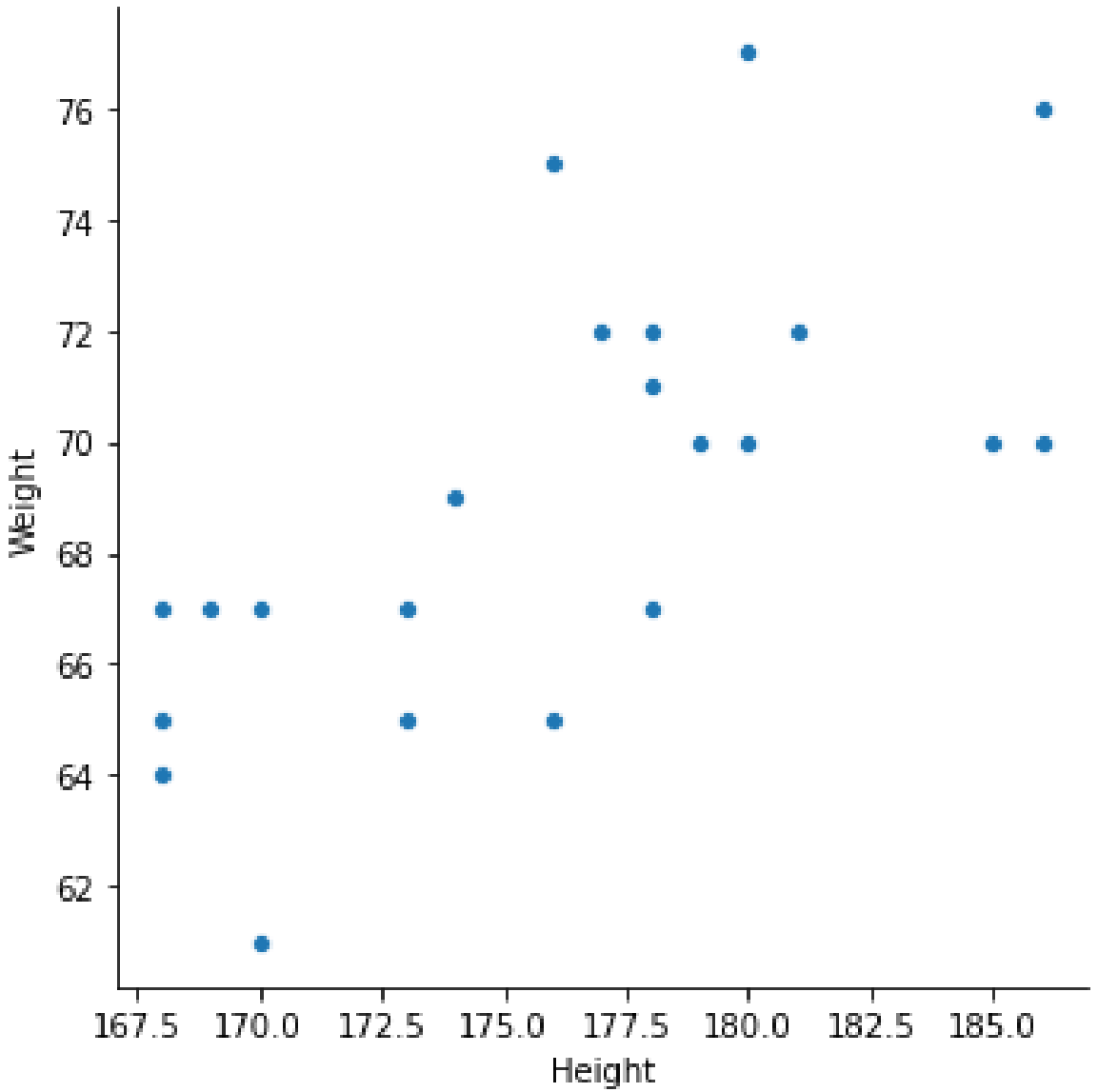
Weight = 0.47 * Height - 14.68

Height (cm)

Weight (kg)

Diễn giải:

Người có chiều cao tăng 1cm thì cân nặng tăng 0.47 kg.



References

1) https://ThachLN.github.io

Good luck!