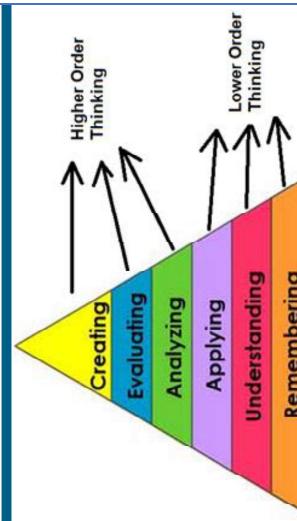


# Chương 1

## NHẬP VÀ XUẤT BỘ DỮ LIỆU



### Mục tiêu



Thang đo BLOOM

- Biết cách đặt vấn đề trong phân tích.
- Hiểu dữ liệu trước khi phân tích.
- Vận dụng phương pháp nhập và xuất bộ dữ liệu
- Biết các thư viện trong Python hỗ trợ phân tích dữ liệu.

## Nội dung

1. Tại sao phải phân tích dữ liệu?
2. Đặt vấn đề trong phân tích
3. Phương pháp hiểu bộ dữ liệu
4. Các thư viện trong Python hỗ trợ phân tích
5. Kỹ thuật nhập bộ dữ liệu
6. Kỹ thuật xuất bộ dữ liệu
7. Phân tích cơ bản

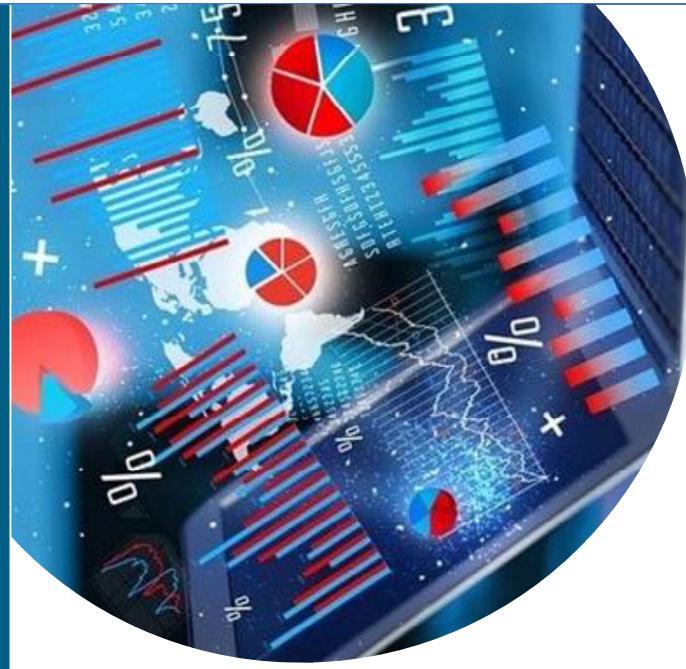
3

### 1. Tại sao phải phân tích dữ liệu?

- Dữ liệu có ở mọi nơi.

- Lĩnh vực phân tích dữ liệu/khoa học dữ liệu giúp chúng ta có thể trả lời các câu hỏi từ dữ liệu.

- Phân tích dữ liệu đóng vai trò quan trọng:**
  - Tìm ra các thông tin có giá trị, hoặc bất bình thường...
  - Trả lời các câu hỏi.
  - Có thể dùng để dự báo/tiên lượng...



4



## 2. Đặt vấn đề trong phân tích

Làm sao thẩm định giá xe cũ?



Giá xe đưa ra không nên quá cao, nhưng cũng không quá thấp.

5



## 2. Đặt vấn đề trong phân tích

Ước tính giá xe đã qua sử dụng?

- Làm thế nào để xác định giá xe tốt nhất?
- Có bộ dữ liệu về giá xe và các đặc điểm kèm theo? Trong đó, đặc tính nào của xe ảnh hưởng đến giá xe?
  - ✓ Màu?
  - ✓ Nhãn hiệu xe?
  - ✓ Mẫu lực? ⇔ Công suất?
  - ✓ Nhiên liệu?
  - ✓ ...
- Đặt câu hỏi phù hợp cho dữ liệu đang có?



6



## 2. Đặt vấn đề trong phân tích

### Tìm/thu thập dữ liệu về giá xe cũ (xe đã qua sử dụng)

```
3.?, alfa-romero,gas,std,two,convertible,rwd,front,88.06,168.80,64.10,48.80,64.10,168.80,64.10,48.80,2548,dohc,four,130,mpfi,3.47,2.68,9.08,111,5000,21,27,13495
3.?, alfa-romero,gas,std,two,convertible,rwd,front,88.60,168.80,64.10,48.80,2548,dohc,four,130,mpfi,3.47,2.68,9.08,111,5000,21,27,16500
1.?, alfa-romero,gas,std,two,hatchback,rwd,front,94.50,171.20,65.50,52,40,2823,ohcv,six,152,mpfi,2.68,3.47,9.08,154,5000,19,26,16500
2,164,audi,gas,std,four,sedan,fwd,front,99.80,176.60,60,54,30,2337,ohc,four,109,mpfi,3.19,3.40,10.00,102,5500,24,38,19950
2,164,audi,gas,std,four,sedan,fwd,front,99.40,176.60,66,49,54,30,2824,ohc,five,136,mpfi,3.19,3.40,8.00,115,5500,18,22,17450
2.?, audi,gas,std,two,sedan,fwd,front,95.80,177.30,53.19,2507,ohc,five,136,mpfi,3.19,3.40,8.50,110,5500,19,25,15150
1,158,audi,gas,std,two,sedan,fwd,front,105.80,192.70,71,40,55.76,2844,ohc,five,136,mpfi,3.19,3.40,8.50,110,5500,19,25,17710
1.?, audi,gas,std,four,wagon,fwd,front,105.80,192.70,71,40,55.70,2954,ohc,five,136,mpfi,3.19,3.40,8.50,110,5500,19,25,18940
1,158,audi,gas,turbo,two,sedan,fwd,front,105.80,192.70,71,49,55.90,3086,ohc,five,131,mpfi,3.13,3.40,8.30,140,5500,1,20,23875
0.?, audi,gas,turbo,two,hatchback,fwd,front,99.50,178.20,67.90,52,00,3053,ohc,five,131,mpfi,3.13,3.40,7.00,160,5500,16,22,?
2,192,bmw,gas,std,two,sedan,rwd,front,101.20,176.80,64.30,2395,ohc,four,108,mpfi,3.50,2.80,8.80,101,5800,23,29,16325
0,192,bmw,gas,std,four,sedan,rwd,front,101.20,176.80,64.80,54,30,2395,ohc,four,108,mpfi,3.50,2.80,8.80,101,5800,23,29,16325
0,188,bmw,gas,std,two,sedan,rwd,front,101.20,176.80,64.80,54,30,2710,ohc,six,164,mpfi,3.31,3.19,9.00,121,4250,21,28,28970
0,188,bmw,gas,std,four,sedan,rwd,front,101.20,176.80,64.80,54,30,2765,ohc,six,164,mpfi,3.31,3.19,9.00,121,4250,21,28,21105
1.?, bmw,gas,std,four,sedan,rwd,front,103.50,189.80,66.90,55.70,3055,ohc,six,164,mpfi,3.31,3.19,9.00,121,4250,20,25,24565
0.,bmw,gas,std,four,sedan,rwd,front,103.50,189.80,66.90,55.70,3230,ohc,six,109,mpfi,3.62,3.42,9.00,18,5400,16,22,30160
0.,?bmw,gas,std,two,sedan,rwd,front,103.50,193.80,67.90,53.70,3380,ohc,six,209,mpfi,3.62,3.39,8.00,182,5400,16,22,41315
0.,?bmw,gas,std,four,sedan,rwd,front,110.00,197.00,70.90,56.30,3505,ohc,six,209,mpfi,3.62,3.39,8.00,182,5400,15,28,35880
2,121,chevrolet,gas,std,two,hatchback,fwd,front,88.40,141.10,60.30,53.20,1488,1,three,61,2bb1,2.91,3.03,3.11,9.50,48,5100,47,53,5151
1.98,chevrolet,gas,std,two,hatchback,fwd,front,94.50,155.90,63.60,52,00,1874,ohc,four,90,2bb1,3.03,3.11,9.60,70,5400,38,43,6295
0,81,chevrolet,gas,std,four,sedan,fwd,front,94.50,158.80,53.90,52,00,1999,ohc,four,90,2bb1,3.03,3.11,9.60,70,5400,38,43,5575
1,118,dodge,gas,std,two,hatchback,fwd,front,93.70,157.30,59,63,80,50,80,1876,ohc,four,90,2bb1,2.97,3.23,9.41,68,5500,37,41,5572
1,118,dodge,gas,std,two,hatchback,fwd,front,93.70,157.30,59,63,80,50,80,1876,ohc,four,90,2bb1,2.97,3.23,9.40,68,5500,31,38,5377
```

Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/>

7



## 2. Đặt vấn đề trong phân tích

**Chúng ta muốn mua/bán  
một căn nhà => tham định  
giá?**



**Thinking About  
Selling  
Your House?**

Có thể chọn làm  
đồ án môn học

8



## 2. Đặt vấn đề trong phân tích



Các nhà kinh doanh muốn dự  
trữ lúa, gạo khi nào là hợp lý  
nhất?



Có thể chọn làm  
đồ môn học

9



## 2. Đặt vấn đề trong phân tích



Giá heo?



Data scientists



10



## 2. Đặt vấn đề trong phân tích

### Bài tập nhỏ làm trên lớp 15 phút



11



## 3. Phương pháp hiệu bộ dữ liệu

### Giới thiệu bộ dữ liệu

- Bộ dữ liệu về giá xe cũ.
- Bộ dữ liệu mở: Jeffrey C. Schlemmer
  - Nguồn:  
<https://archive.ics.uci.edu/ml/machine-learning-databases/autos/>



Data Set Characteristics:	Multivariate	Number of Instances:	205	Area:	N/A
Attribute Characteristics:	Categorical, Integer, Real	Number of Attributes:	26	Date Donated	1987-05-19
Associated Tasks:	Regression	Missing Values ?	Yes	Number of Web Hits:	434925

12



### 3. Phương pháp hiệu bộ dữ liệu

#### Bộ dữ liệu thô

```
3 ?, alfa-romero,gas, std, two, convertible, fwd, front, 88, 60, 168, 80, 64, 10, 48, 80, 2548, dohc, four, 130,mpfi, 3, 47, 2, 68, 9, 00, 111, 5000, 21, 27, 13495
3 ?, alfa-romero,gas, std, two, convertible, fwd, front, 88, 60, 168, 80, 64, 10, 48, 80, 2548, dohc, 3, 47, 2, 68, 9, 00, 111, 5000, 21, 27, 16500
1 ?, alfa-romero,gas, std, two, hatchback, fwd, front, 94, 50, 171, 20, 65, 50, 52, 40, 2823, ohcv,six,152,mpfi,2, 68, 3, 47, 9, 00, 154, 5000, 19, 26, 16500
2,164, audi, gas, std, four, sedan, fwd, front, 99, 38, 176, 60, 66, 20, 54, 38, 2337,ohc, four, 109,mptri, 3,19, 3, 40, 10, 00, 102, 500, 24, 30, 1950
2,164, audi, gas, std, four, sedan, 4wd, front, 99, 48, 176, 60, 66, 40, 54, 38, 2824,ohc, five,136,mpfi, 3,19, 3, 40, 8, 00, 115, 5500, 18, 22, 17450
2 ?, audi, gas, std, two, sedan, fwd, front, 99, 80, 177, 30, 66, 30, 53, 10, 2507,ohc, five,136,mpfi, 3,19, 3, 40, 8, 50, 110, 5500, 19, 25, 15250
1,158, audi, gas, std, four, sedan, fwd, front, 105, 80, 192, 70, 71, 40, 55, 70, 2954,ohc, five,136,mpfi, 3,19, 3, 40, 8, 50, 110, 5500, 19, 25, 18920
1,158, audi, gas, std, four, wagon, fwd, front, 105, 80, 192, 70, 71, 40, 55, 90, 3886,ohc, five,131,mpfi, 3, 13, 3, 40, 8, 30, 140, 5500, 17, 20, 23875
0 ?, audi, gas, turbo, two, hatchback, 4wd, front, 99, 56, 178, 20, 67, 90, 52, 00, 3853,ohc, five,131,mpfi, 3,13, 3, 40, 7, 00, 166, 5500, 16, 22, ?
2,192, bmw, gas, std, two, sedan, fwd, front, 101, 20, 176, 80, 64, 80, 54, 30, 2395,ohc, four, 108,mpfi, 3, 50, 2, 80, 8, 80, 101, 5800, 23, 29, 16440
0,192, bmw, gas, std, four, sedan, "rwd", front, 101, 20, 176, 80, 64, 80, 54, 30, 2395,ohc, four, 108,mpfi, 3, 50, 2, 80, 8, 80, 101, 5800, 23, 29, 16425
0,188, bmw, gas, std, two, sedan, "rwd", front, 101, 20, 176, 80, 64, 80, 54, 30, 2710,ohc, six,164,mpfi, 3, 31, 3, 19, 9, 00, 121, 4250, 21, 28, 20970
0,188, bmw, gas, std, four, sedan, "rwd", front, 101, 20, 176, 80, 64, 80, 54, 30, 2765,ohc, six,164,mpfi, 3, 31, 3, 19, 9, 00, 121, 4250, 21, 28, 21105
1,2, bmw, gas, std, four, sedan, "rwd", front, 103, 50, 189, 90, 66, 90, 55, 70, 3055,ohc, six,164,mpfi, 3, 31, 3, 19, 9, 00, 121, 4250, 20, 25, 24565
0,?, bmw, gas, std, "four", sedan, "rwd", front, 103, 50, 189, 90, 66, 90, 55, 70, 3230,ohc, six,209,mpfi, 3, 62, 3, 39, 8, 00, 182, 5400, 16, 22, 30760
0,?, bmw, gas, std, two, sedan, "rwd", front, 103, 50, 193, 80, 67, 90, 53, 70, 3380,ohc, six,209,mpfi, 3, 62, 3, 39, 8, 00, 182, 5400, 16, 22, 41315
0,?, bmw, gas, std, four, sedan, "rwd", front, 110, 00, 197, 90, 70, 90, 56, 30, 3505,ohc, six,209,mpfi, 3, 62, 3, 39, 8, 00, 182, 5400, 15, 20, 36880
2,121, chevrolet, gas, std, two, hatchback, fwd, front, 88, 40, 141, 10, 60, 30, 53, 20, 1488, 1, three, 61, 2bb1, 2, 91, 3, 03, 9, 50, 48, 5100, 47, 53, 5151
1,98, chevrolet, gas, std, two, hatchback, fwd, front, 94, 50, 155, 90, 63, 60, 52, 00, 1874,ohc, four, 98, 2bb1, 3, 03, 3, 11, 9, 60, 70, 5400, 38, 43, 6295
0,81, chevrolet, gas, std, four, sedan, fwd, front, 94, 50, 158, 80, 63, 60, 52, 00, 1909,ohc, four, 90, 2bb1, 3, 03, 3, 11, 9, 60, 70, 5400, 38, 43, 6575
1,118, dodge, gas, std, two, hatchback, fwd, front, 93, 70, 157, 30, 63, 80, 50, 88, 1876,ohc, four, 90, 2bb1, 2, 97, 3, 23, 9, 40, 68, 5500, 31, 38, 6377
1,118, dodge, gas, std, two, hatchback, fwd, front, 93, 70, 157, 30, 63, 80, 50, 88, 1876,ohc, four, 90, 2bb1, 2, 97, 3, 23, 9, 40, 68, 5500, 31, 38, 6377
```

Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data>

13



### 3. Phương pháp hiệu bộ dữ liệu

#### Phương pháp

- Biết rõ bản chất từng dòng dữ liệu.
- Hiểu các mô tả từng thuộc tính (cột, biến) của bộ dữ liệu.
- Xem xét kiểu dữ liệu và phạm vi giá trị từng thuộc tính.
- Thuộc tính để liên kết dữ liệu.
- Thuộc tính liên quan vẫn đề đặt ra.

	Attribute1	Attribute2	Attribute3	Attribute4
0	...	...	...	...
1	...	...	...	...
2	...	...	...	...
3	...	...	...	...
n	...	...	...	...

14



### 3. Phương pháp hiệu bộ dữ liệu

#### Mô tả các thuộc tính của bộ dữ liệu

Attribute Information:	Attribute:	Attribute Range:	Attribute:	Attribute Range:
1. symboling:	symboling:	-3, -2, -1, 0, 1, 2, 3.	10. wheel-base:	continuous from 86.6 120.9.
2. normalized-losses:	normalized-losses:	continuous from 65 to 256.	11. length:	continuous from 141.1 to 208.1.
3. make:	make:	alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugeot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, volvo std, turbo.	12. width:	continuous from 60.3 to 72.3.
4. fuel-type:	fuel-type:	four, two.	13. height:	continuous from 47.8 to 59.8.
5. aspiration:	aspiration:	hardtop, wagon, sedan, hatchback, convertible.	14. curb-weight:	continuous from 1488 to 4066.
6. num-of-doors:	num-of-doors:	4wd, fwd, rwd.	15. engine-type:	dohc, dohcvt, l, ohc, ohcf, ohcv, rotor.
7. body-style:	body-style:	front, rear.	16. num-of-cylinders:	eight, five, four, six, three, twelve, two.
8. drive-wheels:	drive-wheels:	continuous from 86.6 120.9.	17. engine-size:	continuous from 61 to 326.
9. engine-location:	engine-location:	continuous from 141.1 to 208.1.	18. fuel-system:	lbbl, 2bbbl, 4bbbl, idti, mfi, mpfi, spdi, spf.
10. wheel-base:	wheel-base:	continuous from 47.8 to 59.8.	19. bore:	continuous from 2.07 to 4.17.
11. length:	length:	continuous from 141.1 to 208.1.	20. stroke:	continuous from 7 to 23.
			21. compression-ratio:	continuous from 7 to 23.
			22. horsepower:	continuous from 48 to 288.
			23. peak-rpm:	continuous from 4150 to 6600.
			24. city-mpg:	continuous from 13 to 49.
			25. highway-mpg:	continuous from 16 to 54.
			26. price:	continuous from 5118 to 45400.

Jeffrey C. Schlemmer

Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.names>

15



### 3. Phương pháp hiệu bộ dữ liệu

#### Mô tả các thuộc tính của bộ dữ liệu

No.	Attribute name	attribute range	No.	Attribute name	attribute range
1	symboling	-3, -2, -1, 0, 1, 2, 3.	14	curb-weight	continuous from 1488 to 4066.
2	normalized-losses	continuous from 65 to 256.	15	engine-type	dohc, dohcvt, l, ohc, ohcf, ohcv, rotor.
3	make	audi, bmw, etc.	16	num-of-cylinders	eight, five, four, six, three, twelve, two.
4	fuel-type	diesel, gas.	17	engine-size	continuous from 61 to 326.
5	aspiration	std, turbo.	18	fuel-system	lbbl, 2bbbl, 4bbbl, idti, mfi, mpfi, spdi, spf.
6	num-of-doors	four, two.	19	bore	continuous from 2.54 to 3.94.
7	body-style	hardtop, wagon, etc.	20	stroke	continuous from 2.07 to 4.17.
8	drive-wheels	4wd, fwd, rwd.	21	compression-ratio	continuous from 7 to 23.
9	engine-location	front, rear.	22	horsepower	continuous from 48 to 288.
10	wheel-base	continuous from 86.6 120.9.	23	peak-rpm	continuous from 4150 to 6600.
11	length	continuous from 141.1 to 208.1.	24	city-mpg	continuous from 13 to 49.
12	width	continuous from 60.3 to 72.3.	25	highway-mpg	continuous from 16 to 54.
13	height	continuous from 47.8 to 59.8.	26	price	continuous from 5118 to 45400.

Jeffrey C. Schlemmer

Chú ý, các có thể gọi **thuộc tính** là **cột/biến**, hoặc gọi là **attribute**, hoặc gọi là **feature**.

16



### 3. Phương pháp hiệu bộ dữ liệu

#### Mô tả các thuộc tính của bộ dữ liệu

- Mức độ rủi ro bảo hiểm/1 xe
- Giá trị +3 cho thấy ô-tô có rủi ro cao.
- 3, là khá an toàn.

No.	Attribute name	attribute range
1	symboling	-3, -2, -1, 0, 1, 2, 3.
2	normalized-losses	continuous from 65 to 256.
3	make	audi, bmw, etc.
4	fuel-type	diesel, gas.
5	aspiration	std, turbo.
6	num-of-doors	four, two.
7	body-style	hardtop, wagon, etc.
8	drive-wheels	4wd, fwd, rwd.
9	engine-location	front, rear.
10	wheel-base	continuous from 86.6 to 120.9.
11	length	continuous from 141.1 to 208.1.
12	width	continuous from 60.3 to 72.3.
13	height	continuous from 47.8 to 59.8.

Jeffrey C. Schlemmer

17



### 3. Phương pháp hiệu bộ dữ liệu

#### Mô tả các thuộc tính của bộ dữ liệu

No.	Attribute name	attribute range
1	Trọng lượng của xe	continuous from 1488 to 4066.
2	Loại động cơ	dohc, dohcvt, i, ohc, ohcf, rotor.
3	Số lượng bánh xe	eight, five, four, six, three, twelve, two.
4	Dung tích động cơ	continuous from 61 to 326.
5	Loại hệ thống nhiên liệu	1bbl, 2bbl, 4bbl, idi, mpfi, spdi, spfi.
6	Đường kính xi-lanh động cơ	bore
7	Chiều dài thì động cơ	stroke
8	Hệ số nén của động cơ	compression-ratio
9	Mã lực	horsepower
10	Đồng hồ xe thể hiện thông tin	peak-rpm
11	Tiêu thụ nhiên liệu chạy trong Tp	city-mpg
12	Tiêu thụ nhiên liệu chạy trong XL	highway-mpg
13	Giá xe	price

Jeffrey C. Schlemmer

...

18



### 3. Phương pháp hiệu bộ dữ liệu

#### Thuộc tính liên đến vấn đề đặt ra

No.	Attribute name	attribute range	No.	Attribute name	attribute range
1	symboling	-3, -2, -1, 0, 1, 2, 3.	14	curb-weight	continuous from 1488 to 4066.
2	normalized-losses	continuous from 65 to 256.	15	engine-type	dohc, dohcv, l, ohc, ohcf, ohcv, rotor.
3	make	audi, bmw, etc.	16	num-of-cylinders	eight, five, four, six, three, twelve, two.
4	fuel-type	diesel, gas.	17	engine-size	continuous from 61 to 326.
5	aspiration	std, turbo.	18	fuel-system	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi.
6	num-of-doors	four, two.	19	bore	continuous from 2.54 to 3.94.
7	body-style	hardtop, wagon, etc.	20	stroke	continuous from 2.07 to 4.17.
8	drive-wheels	4wd, fwd, rwd.	21	compression-ratio	continuous from 7 to 23.
9	engine-location	front, rear.	22	horsepower	continuous from 48 to 288.
10	wheel-base	continuous from 86.6 to 120.9.	23	peak-rpm	continuous from 4150 to 6600.
11	length	continuous from 141.1 to 208.1.	24	city-mpg	continuous from 13 to 49.
12	width	continuous from 60.3 to 72.3.	25	highway-mpg	continuous from 16 to 54.
13	height	continuous from 47.8 to 59.8.	26	price	continuous from 5118 to 45400.

By Jeffrey C. Schlemmer

Target(Label)

Attributes description. Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.names>

19



### 3. Phương pháp hiệu bộ dữ liệu

#### Bài tập áp dụng

Tìm và tải các bộ dữ liệu trên UCI, Kaggle, Kdnuggets hoặc nguồn khác ...  
thuộc các chủ đề sau:

1. Giá điện thoại, laptop cũ
2. Giá nhà, đất
3. Nông nghiệp
4. Giá cổ phiếu
5. hoặc các chủ đề tự chọn khác

Sau đó, mô tả chi tiết các bộ dữ liệu.

...

20



## 4. Các thư viện trong Python hỗ trợ phân tích



21



## 4. Các thư viện trong Python hỗ trợ phân tích

**Trong khóa học này có ba gói thư viện chính:**

- 4.1.** Thư viện hỗ trợ xử lý dữ liệu
- 4.2.** Thư viện hỗ trợ trực quan dữ liệu
- 4.3.** Thư viện hỗ trợ các thuật toán phân tích

22



## 4. Các thư viện trong Python hỗ trợ phân tích

### 4.1. Thư viện hỗ trợ xử lý dữ liệu



- Biểu diễn các cấu trúc dữ liệu
- Công cụ xử lý dữ liệu



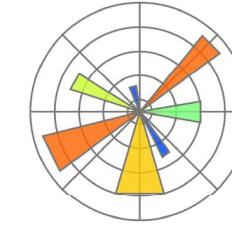
- Biểu diễn mảng và ma trận lớn đa chiều.
- Hàm toán học cao cấp.

23



### 4. CácthưviệntrongPythonhỗtrợphân tích

### 4.2. Thư viện hỗ trợ trực quan dữ liệu



- Phác họa đồ thị theo ý muốn.
- 3 chiều



- Cung cấp API
- violin plot, displot, heatmap, cluster map, time series...

24



## 4. Các thư viện trong Python hỗ trợ phân tích

### 4.3. Thư viện hỗ trợ thuật toán phân tích



- Học máy: Classification, Regression, Clustering
- Chọn mô hình và xử lý dữ liệu.



- Chuẩn hóa dữ liệu
- Thăm dò dữ liệu.
- Mô hình thống kê
- Kiểm định thống kê.

25



## 4. Các thư viện trong Python hỗ trợ phân tích

### Bài tập nhỏ:

Nhập (import) các thư viện sau vào trong dự án (project) phân tích dữ liệu

1. Numpy
2. Pandas
3. Matplotlib
4. Seaborn
5. Sk-learn
6. Statsmodels

Sau đó, in ra thông tin, phiên bản các thư viện

26

## 4. Các thư viện trong Python hỗ trợ phân tích

### Hướng dẫn:

1. Dùng lệnh “`pip`” để cài đặt các thư viện nếu chưa có trong trình thông dịch cụ của Python.
2. Dùng lệnh “`import`” để nhập các thư viện vào dự án .
3. Gọi phương thức “`version`” để xuất thông tin và phiên bản của thư viện

27



## 5. Kỹ thuật nhập bộ dữ liệu

### Nhập bộ dữ liệu

- Là quá trình tải và đọc dữ liệu từ nhiều nguồn.
- Loại: `.CSV`, `.XLSX`, `.HDF`, `.JSON`...
- Nguồn:
  - Tại máy tính: `/Documents/mydata.csv`
  - Qua mạng: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data>

#### Kỹ thuật: dùng pandas để đọc

#### Kết quả: dataframe (gọi tắt df)

Result → dataframe

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	Nan
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	LSU	5000000.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0	6-8	235.0	Louisville	1709560.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Ohio State	18245460.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN	6-9	260.0	Ohio State	2569560.0
6	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0

28



## 5. Kỹ thuật nhập bộ dữ liệu

### Nhập bộ dữ liệu

- Dùng các phương pháp sau để nhập bộ dữ liệu phục vụ phân tích.

Loại	Định dạng	Phương thức đọc
Comma-separated Values (csv)	csv	pandas.read_csv()
Excel sheet	excel	pandas.read_excel()
SQL database	sql	pandas.read_sql()
Hierarchical Data Format (HDF)	hdf	pandas.read_hdf()
JSON string	json	pandas.read_json()
	...	

29



## 5. Kỹ thuật nhập bộ dữ liệu

### Bộ dữ liệu thô

```
3 ?, alfa-romero,gas,std,two,convertible,rwd,front,88,60,168,80,64,10,48,80,2548,dohc,four,130,mpfi,3,47,2,68,9,00,111,5000,21,27,13495
3 ?, alfa-romero,gas,std,two,convertible,rwd,front,88,60,168,80,64,10,48,80,2548,dohc,four,130,mpfi,3,47,2,68,9,00,111,5000,21,27,16500
1 ?, alfa-romero,gas,std,two,hatchback,rwd,front,94,50,171,20,65,50,52,40,2823,ohcv,six,1152,mpfi,2,68,3,47,9,00,154,8000,19,26,16500
2,164, audi,gas,std,four,sedan,fwd,front,99,88,176,60,66,20,54,30,2337,ohc,four,109,mpfi,3,19,3,49,10,00,182,5500,24,36,19500
2,164, audi,gas,std,four,sedan,fwd,front,99,40,176,60,66,40,54,30,2824,ohc,five,136,mpfi,3,19,3,49,8,00,115,5000,18,22,17450
2 ?, audi,gas,std,two,sedan,fwd,front,99,88,177,30,66,30,53,10,2507,ohc,five,136,mpfi,3,19,3,49,8,50,110,5000,19,25,17710
1,158, audi,gas,std,four,sedan,fwd,front,105,80,192,70,71,40,55,70,2844,ohc,five,136,mpfi,3,19,3,40,8,50,110,5000,19,25,18900
1 ?, audi,gas,std,four,sedan,fwd,front,105,80,192,70,71,40,55,70,2954,ohc,five,136,mpfi,3,19,3,40,8,50,110,5000,19,25,18900
1,158, audi,gas,turbo,four,wagon,fwd,front,105,80,192,70,71,40,55,70,2954,ohc,five,131,mpfi,3,13,3,40,8,30,140,5500,17,20,23875
0 ?, audi,gas,turbo,two,hatchback,4wd,front,99,50,178,20,67,90,52,00,3053,ohc,five,131,mpfi,3,13,3,40,7,00,160,5500,16,22,?
2,192,bmw,gas,std,two,sedan,fwd,front,101,28,176,80,64,80,54,30,2395,ohc,four,108,mpfi,3,39,2,80,8,80,101,5000,23,29,16925
0,192,bmw,gas,std,four,sedan,rwd,front,101,20,176,80,64,80,54,30,2395,ohc,four,108,mpfi,3,39,2,80,8,80,101,5000,23,29,16925
0,188,bmw,gas,std,two,sedan,rwd,front,101,20,176,80,64,80,54,30,2749,ohc,six,164,mpfi,3,31,3,19,9,00,121,4250,21,28,20970
0,188,bmw,gas,std,four,sedan,rwd,front,101,20,176,80,64,80,54,30,2765,ohc,six,164,mpfi,3,31,3,19,9,00,121,4250,21,28,21105
1 ?, bmw,gas,std,four,sedan,rwd,front,103,50,189,00,66,90,55,70,230,ohc,four,108,mpfi,3,31,3,19,9,00,121,4250,20,25,24565
0 ?, bmw,gas,std,four,sedan,rwd,front,103,50,189,00,66,90,55,70,230,ohc,four,108,mpfi,3,62,3,39,8,00,182,5000,16,22,41315
0 ?, bmw,gas,std,two,sedan,rwd,front,103,50,193,80,67,90,53,70,3380,ohc,six,209,mpfi,3,62,3,39,8,00,182,5000,16,22,41315
0 ?, bmw,gas,std,four,sedan,rwd,front,110,00,197,00,70,90,56,30,3505,ohc,six,209,mpfi,3,62,3,39,8,00,182,5000,15,20,36880
2,121,chevrolet,gas,std,two,hatchback,fwd,front,88,40,141,10,60,30,53,20,1488,1,three,51,2bb1,2,91,3,03,9,50,48,5100,47,53,5151
1,98,chevrolet,gas,std,two,hatchback,fwd,front,94,50,155,90,63,60,52,00,1874,ohc,four,90,2bb1,3,03,3,11,9,60,70,5400,38,43,6295
0,1,chevrolet,gas,std,four,sedan,fwd,front,94,50,158,80,63,60,52,00,1809,ohc,four,90,2bb1,3,03,3,11,9,60,70,5400,38,43,6295
1,118,dodge,gas,std,two,hatchback,fwd,front,93,70,157,36,63,80,50,88,1876,ohc,four,90,2bb1,2,97,3,23,9,41,68,5500,37,41,5572
1,118,dodge,gas,std,two,hatchback,fwd,front,93,70,157,36,63,80,50,88,1876,ohc,four,90,2bb1,2,97,3,23,9,40,68,5500,31,38,6377
```

Data Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data>



## 5. Kỹ thuật nhập bộ dữ liệu

### Áp dụng

```
import pandas as pd

path = "https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data"

headers = ["symboling", "normalized-losses", "make", "fuel-type", "aspiration",
           "num-of-doors", "body-style", "drive-wheels", "engine-location",
           "wheel-base", "length", "width", "height", "curb-weight", "engine-type",
           "num-of-cylinders", "engine-size", "fuel-system", "bore", "stroke",
           "compression-ratio", "horsepower", "peak-rpm", "city-mpg", "highway-mpg", "price"]
```

```
df = pd.read_csv(path, names=headers)
```

31



## 5. Kỹ thuật nhập bộ dữ liệu

### Áp dụng

symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	... engine-size	fuel-system	bore	stroke		
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47
3	2	164	audi	gas	std	four	sedan	4wd	front	99.8	...	109	mpfi	3.19	3.40
4	2	164	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40
5	2	?	audi	gas	std	two	sedan	twd	front	99.8	...	136	mpfi	3.19	3.40
6	1	158	audi	gas	std	four	sedan	twd	front	105.8	...	136	mpfi	3.19	3.40
7	1	?	audi	gas	std	four	wagon	twd	front	105.8	...	136	mpfi	3.19	3.40
8	1	158	audi	gas	turbo	four	sedan	twd	front	105.8	...	131	mpfi	3.13	3.40
9	0	?	audi	gas	turbo	two	hatchback	4wd	front	99.5	...	131	mpfi	3.13	3.40

32



## 5. Kỹ thuật nhập bộ dữ liệu

### Xem bộ dữ liệu

- df => xuất hết bộ dữ liệu (xuất hết df)
- df.head(n) xuất n dòng đầu tiên của bộ dữ liệu.
- df.tail(n) xuất n dòng cuối của bộ dữ liệu.

```
1 | df.tail()
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	... engine-size	fuel-system	bore	stroke
200	-1	95	volvo	gas	std	four	sedan	rwd	front	109.1	...	141	mpfi	3.78
201	-1	95	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78
202	-1	95	volvo	gas	std	four	sedan	rwd	front	109.1	...	173	mpfi	3.58
203	-1	95	volvo	diesel	turbo	four	sedan	rwd	front	109.1	...	145	idi	3.01
204	-1	95	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78
							...					33		



## 5. Kỹ thuật nhập bộ dữ liệu

### Bài tập nhỏ

Đọc bộ dữ liệu và tổ chức theo kiểu df (dataframe).

Với df có được c, dùng các phương thức sau:

1. iloc()
2. loc()
3. ix()

Để chọn mẫu dữ liệu trong df.



## 6. Kỹ thuật xuất bộ dữ liệu

### Xuất bộ dữ liệu

- Lưu lại dữ liệu bất cứ lúc nào trong quá trình phân tích xây ra biến đổi.
- Dùng các phương pháp sau để xuất bộ dữ liệu để phân tích.

Loại	Định dạng	Phương thức xuất
Comma-separated Values (csv)	csv	df.to_csv()
Excel sheet	excel	df.to_excel()
SQL database	sql	df.to_sql()
Hierarchical Data Format (HDF)	hdf	df.to_hdf()
JSON string	json	df.to_json()
	...	35



## 6. Kỹ thuật xuất bộ dữ liệu

### Áp dụng

symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	... engine-size	fuel-system	bore	stroke
1	161	mitsubishi	gas	turbo	two	hatchback	fwd	front	93.0	...	98	spdi	3.03
3	197	toyota	gas	std	two	hatchback	rwd	front	102.9	...	171	mpfi	3.27
1	168	toyota	gas	std	two	hatchback	rwd	front	94.5	...	98	mpfi	3.24
3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47
1	122	nissan	gas	std	four	sedan	fwd	front	94.5	...	97	2bbl	3.15
													3.29

```
df.to_csv("datasetV2.csv")
```



## 6. Kỹ thuật xuất bộ dữ liệu

### Tóm tắt

Loại	Định dạng	Phương thức đọc	Phương thức xuất
Comma-separated Values (csv)	csv	pandas.read_csv()	df.to_csv()
Excel sheet	excel	pandas.read_excel()	df.to_excel()
SQL database	sql	pandas.read_sql()	df.to_sql()
Hierarchical Data Format (HDF)	hdf	pandas.read_hdf()	df.to_hdf()
JSON string	json	pandas.read_json()	df.to_json()
		...	

37



## 7. Phân tích cơ bản

### Phương pháp

- Hiểu dữ liệu trước khi bắt đầu phân tích.
- Kiểm tra kiểu dữ liệu => Tìm các vấn đề tiềm ẩn trong bộ dữ liệu.
- Kiểm tra sự phân bố dữ liệu, tức độ trai dữ liệu.
- Tóm tắt dữ liệu.



38



## 7. Phân tích cơ bản

### Kiểu dữ liệu

- Kiểm tra kiểu dữ liệu để phát hiện quá trình lỗi tìm kiếm không khớp nhau khi chuyển từ dữ liệu thô sang các kiểu dữ liệu trong pandas.

Kiểu dữ liệu trong Python	Kiểu dữ liệu trong pandas	Ý nghĩa
string	object	Kiểu chuỗi
int	int64	Số nguyên
float	float64	Số thực
datetime	datetime64	Kiểu thời gian
	...	

- Dùng `dataframe.dtypes` để kiểm tra kiểu dữ liệu.

39



## 7. Phân tích cơ bản

### Áp dụng

```
1 df.dtypes
symboling          int64
normalized-losses    object
make                object
fuel-type            object
aspiration          object
num-of-doors        object
body-style          object
drive-wheels        object
engine-location     object
wheel-base          float64
length              float64
width               float64
height              float64
curb-weight         int64
engine-type          object
num-of-cylinders    object
engine-size          int64
fuel-system          object
bore                object
stroke              float64
compression-ratio   object
horsepower          object
peak-rpm             object
city-mpg             int64
highway-mpg          int64
price               object
dtype: object
```

...

40

## 7. Phân tích cơ bản

### Phân bố dữ liệu

- Trả về tóm tắt thống kê.
  - Tính toán các thông số trong thống kê mô tả (sẽ học trong chương 3).
  - Sử dụng:
    - `dataframe.describe()`:
    - `dataframe.describe(include = "all")`:
- => tóm tắt xu hướng trung tâm, độ phân tán và hình dạng của phân bố của tập dữ liệu, không bao gồm các giá trị NaN.

41

## 7. Phân tích cơ bản

### Áp dụng `dataframe.describe()`

1 df.describe()

	symboling	wheel-base	length	width	height	curb-weight	engine-size	compression-ratio	city-mpg	highway-mpg
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	0.834146	98.756585	174.049268	65.907805	53.724878	2555.566584	126.907317	10.142537	25.219512	30.751220
std	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	3.972040	6.542142	6.886443
min	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	7.000000	13.000000	16.000000
25%	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	8.600000	19.000000	25.000000
50%	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	9.000000	24.000000	30.000000
75%	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	9.400000	30.000000	34.000000
max	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	23.000000	49.000000	54.000000

42



## 7. Phân tích cơ bản

### Áp dụng `dataframe.describe(include = "all")`

```
1 | df.describe(include="all")
```



	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio
count	205.000000	205	205	205	205	205	205	205	205	205	205	205.000000	205	205	205	205.000000
unique	NaN	52	22	2	2	3	5	3	2	NaN	...	NaN	8	39	37	NaN
top	NaN	?	toyota	gas	std	four	sedan	fwd	front	NaN	...	NaN	mpfi	3.62	3.40	NaN
freq	NaN	41	32	185	168	114	96	120	202	NaN	...	NaN	94	23	20	NaN
mean	0.834146	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	98.756585	...	126.907317	NaN	NaN	NaN	10.142537
std	1.245307	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	6.021776	...	41.642693	NaN	NaN	NaN	3.972040
min	-2.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	86.600000	...	61.000000	NaN	NaN	NaN	7.000000
25%	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	94.500000	...	97.000000	NaN	NaN	NaN	8.600000
50%	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	97.000000	...	120.000000	NaN	NaN	NaN	9.000000
75%	2.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	102.400000	...	141.000000	NaN	NaN	NaN	9.400000
max	3.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	120.900000	...	326.000000	NaN	NaN	NaN	23.000000

11 rows x 26 columns

43

```
1 | df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   symbolic         205 non-null    int64  
 1   normalized-losses 205 non-null    int64  
 2   make             205 non-null    object 
 3   fuel-type        205 non-null    object 
 4   aspiration       205 non-null    object 
 5   num-of-doors     205 non-null    object 
 6   body-style       205 non-null    object 
 7   drive-wheels    205 non-null    float64
 8   engine-location  205 non-null    object 
 9   wheel-base       205 non-null    float64
 10  length           205 non-null    float64
 11  width            205 non-null    float64
 12  height           205 non-null    float64
 13  curb-weight      205 non-null    int64  
 14  engine-type      205 non-null    object 
 15  num-of-cylinders 205 non-null    int64  
 16  engine-size      205 non-null    float64
 17  fuel-system      205 non-null    object 
 18  bore              205 non-null    float64
 19  stroke            205 non-null    float64
 20  compression-ratio 205 non-null    float64
 21  horsepower        205 non-null    object 
 22  peak-rpm          205 non-null    object 
 23  city-mpg          205 non-null    int64  
 24  highway-mpg       205 non-null    int64  
 25  price             205 non-null    object(16)
dtypes: float64(5), int64(5), object(16)
memory usage: 41.8+ KB
```

## 7. Phân tích cơ bản



### Tóm tắt dữ liệu

- Dùng `dataframe.info()` xuất bảng tóm tắt ngắn gọn của `DataFrame`.

#### Bao gồm:

- Số dòng, số cột.
- Chỉ số dòng, cột.
- Kiểu dữ liệu
- Số lượng giá trị non-null.
- Sử dụng bộ nhớ cho bộ dữ liệu.

44



Thank you

## Hỏi - Đáp

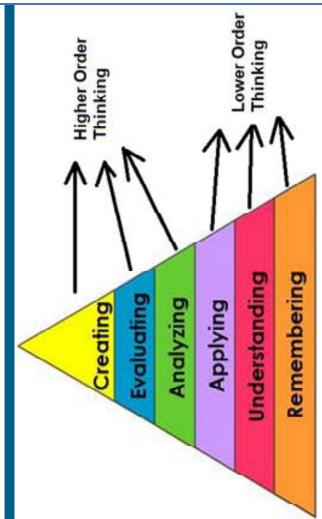


## Chương 2

# SẮP XẾP DỮ LIỆU



### Mục tiêu



Thang đo BLOOM

- Xác định và xử lý các giá trị bị khuyết.
- Vận dụng định dạng dữ liệu.
- Vận dụng chuẩn hóa dữ liệu.
- Chuyển đổi giá trị trong bộ dữ liệu.



## Nội dung

1. Tiền xử lý dữ liệu trước khi phân tích.
2. Xử lý giá trị bị khuyết.
3. Định dạng dữ liệu.
4. Chuẩn hóa dữ liệu.
5. Biến đổi giá trị số thành giá trị phân loại.
6. Biến đổi giá trị phân loại thành giá trị số.



3



## Sắp xếp dữ liệu - Data Wrangling

*Tại sao phải sắp xếp dữ liệu?*



4



## 1. Tiền xử lý dữ liệu

Quá trình chuyển đổi hoặc ánh xạ dữ liệu từ dạng “thô” sang một định dạng khác, để chuẩn bị dữ liệu trước cho phân tích.

(Joseph Santarcangelo, Ph.D., Data Scientist at IBM)

- Thuật ngữ tương tự:

- Data Cleaning
- Data Wrangling

**“ 80 percent of a data scientist's valuable time is spent simply finding, cleansing, and organizing data, leaving only 20 percent to actually perform analysis... ”**

IBM Data Analytics

5



## 1. Tiền xử lý dữ liệu

### Nhiệm vụ chính

- Tìm và xử lý giá trị khuyết
- Định dạng dữ liệu
- Chuẩn hóa dữ liệu
- Chuyển giá trị số thành các giá trị phân loại
- Chuyển các giá trị phân loại thành giá trị số



6



## 2. Xử lý giá trị bị khuyết

### Giá trị bị khuyết?

- Giá trị bị khuyết là giá trị bị thiếu (trống/rỗng) trong bộ dữ liệu (hoặc df)
- Xảy ra khi một mẫu nào đó thiếu giá trị của một thuộc tính (*hay feature*) nào đó.

- Thông thường, giá trị khuyết trong dữ liệu thô được thể hiện dưới dạng:

▪ Dấu chấm hỏi “?”

▪ Dấu chấm “.”, “Non”, “Null”

▪ “N/A”, “NA”, “NaN”, “NaN”

▪ Hoặc “0” (tùy tình huống)

▪ Hoặc để “ô trống” .

	Attribute1	Attribute2	Attribute3	Attribute4
0	...	...	...	...
1	...	...	?	...
2	...	?	...	
3	?	...		N/A
...	...	N/A	...	...
n	...	...	0	...
...	...	...	...	...

7



## 2. Xử lý giá trị bị khuyết

### Ví dụ giá trị bị khuyết

symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke
0	3	? alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68
1	3	? alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68
2	1	? alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47
3	2	164 audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40
4	2	164 audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40
5	2	? audi	gas	std	two	sedan	fwd	front	99.8	...	136	mpfi	3.19	3.40
6	1	158 audi	gas	std	four	sedan	fwd	front	105.8	...	136	mpfi	3.19	3.40
7	1	? audi	gas	std	four	wagon	fwd	front	105.8	...	136	mpfi	3.19	3.40
8	1	158 audi	gas	turbo	four	sedan	fwd	front	105.8	...	131	mpfi	3.13	3.40
9	0	? audi	gas	turbo	two	hatchback	4wd	front	99.5	...	131	mpfi	3.13	3.40
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

8



## 2. Xử lý giá trị bị khuyết

### Phương pháp xử lý dữ liệu bị thiếu?

- Hiểu rõ nguồn dữ liệu đang có.
- Kiểm tra nguồn thu thập dữ liệu.
- Loại bỏ giá trị bị thiếu
  - Loại bỏ biến/thuộc tính
  - Loại bỏ nguyên mẫu dữ liệu
- Thay thế giá trị bị thiếu
  - Thay thế bằng giá trị trung bình (*average*)
  - Thay thế bằng tần suất (*frequency*)
- Dùng thuật toán để diễn khuyết giá trị(\*)

\*Ref: <https://scikit-learn.org/stable/modules/impute.html>



9



## 2. Xử lý giá trị bị khuyết

### Phương pháp 1: Tìm và loại bỏ các giá trị bị khuyết

DataFrame.dropna(**axis=0**, how='any', thresh=None, subset=None, inplace=False)

#### Parameters

**axis {0 or 'index', 1 or 'columns', 'default'}**

Determine if rows or columns which contain missing values are removed.

**0, or 'index'**: Drop rows which contain missing values.

**1, or 'columns'**: Drop columns which contain missing value.

**how {'any', 'all'}**, default 'any'

Determine if row or column is removed from DataFrame, when we have at least one NA or all NA.

'any': If any NA values are present, drop that row or column.

'all': If all values are NA, drop that row or column.

**thresh int, optional**

Require that many non-NA values.

**subset array-like, optional**

Labels along other axis to consider, e.g. if you are dropping rows these would be a list of columns to include.

**inplace bool, default False**

If True, do operation inplace and return None.

#### Returns

**DataFrame**

DataFrame with NA entries dropped from it.



10



## 2. Xử lý giá trị bị khuyết

### Các phương thức khác

STT	Phương thức	Mô tả
1	DataFrame.isna	Xác định các giá trị khuyết
2	DataFrame.notna	Xác định các giá trị <b>không</b> khuyết
3	DataFrame.fillna	Thay thế giá trị khuyết
4	Series.dropna	Loại bỏ các giá trị khuyết
5	Index.dropna	Trả về các chỉ số không có giá trị NA / NaN



11



## 2. Xử lý giá trị bị khuyết

### Áp dụng

df.dropna(subset = ["price"], axis = 0, inplace = True)

horsepower	peak-rpm	price
...	...	...
114	5400	16845
160	5300	19045
134	5500	NaN
106	4800	22470
114	5400	22625
102	5500	13950
...	...	...



11



Chú ý:

- axis = 0 loại bỏ dòng
- axis = 1 loại bỏ cột



12



## 2. Xử lý giá trị bị khuyết

### Chú ý

```
df.dropna(subset = ["price"], axis = 0)
```

```
df.dropna(subset = ["price"], axis = 0, inplace = True)
```

Ref.: <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html>



13



## 2. Xử lý giá trị bị khuyết

**Phương pháp 2: Thay thế các giá trị bị khuyết bằng giá trị khác**

```
df.replace(  
    to_replace= None,  
    value= None,  
    inplace= False,  
    limit= None,  
    regex= False,  
    method= 'pad' )
```

Giải thích:

- **to\_replace**: ký hiệu bị khuyết cần thay thế. Ví dụ: NaN, "?", " ", .
- **value**: giá trị mới. Ví dụ mean, mode...
- **inplace**: True/False

Ref.: <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.replace.html>



14



## 2. Xử lý giá trị bị khuyết

**Áp dụng** `mean = df[“horsepower”].mean()`  
`df[“horsepower”].replace(to_replace=np.nan, value=mean)`

horsepower	peak-rpm	price
...	...	...
114	5400	16845
160	5300	19045
134	5500	21485
NaN	4800	22470
114	5400	22625
102	5500	13950
...	...	...

↑

horsepower	peak-rpm	price
...	...	...
114	5400	16845
160	5300	19045
134	5500	21485
124	4800	22470
114	5400	22625
102	5500	13950
...	...	...

↓

15



## 2. Xử lý giá trị bị khuyết

### Dùng thuật toán

- Có thể dùng thêm mô-đun `preprocessing` trong `sci-kitlearn` để hỗ trợ tiền xử lý dữ liệu.
- Ref: <https://scikit-learn.org/stable/modules/preprocessing.html>





## 2. Xử lý giá trị bị khuyết

### Bài tập

- Làm sao phát hiện (detect) các giá trị bị khuyết theo dạng ký tự đặc biệt?

Gợi ý: Sau khi chuyển thành NaN thì dùng hai phương thức:

- isnull()
- notnull()

Sau đó, thống kê số lượng giá trị bị khuyết.

=> Từ đó phân tích là nên loại bỏ (drop) hay thay thế (replace)?

Trong bài LAB cũng sẽ có bài tương tự như thế này.

17



## 2. Xử lý giá trị bị khuyết

### Bài tập

- Tìm thêm một số giải pháp khác cho việc phát hiện các giá trị khuyết.
- Dựa vào các bộ dữ liệu trong các bài tập trước, dùng các phương pháp đó minh họa.
- Tìm hiểu **Nearest neighbors** của **KNNImputer** trong **sklearn** để điền giá trị khuyết.



18



## 2. Xử lý giá trị bị khuyết

### Gợi ý

```
import numpy as np
from sklearn.impute import KNNImputer

X = [[1, 2, np.nan],
     [3, 4, 3],
     [np.nan, 6, 5],
     [8, 8, 7]]
```

```
imputer = KNNImputer(n_neighbors=2)
```

```
imputer.fit_transform(X)

array([[1., 2., 4.],
       [3., 4., 3.],
       [5.5, 6., 5.],
       [8., 8., 7.]])
```

19



## 3. Định dạng dữ liệu

### Giới thiệu

- Dữ liệu thường được thu thập từ những nơi khác nhau và được lưu trữ ở các định dạng khác nhau => nên không nhất quán.
- Chuyển đổi dữ liệu dựa theo một tiêu chuẩn tổng quát chung, và cho phép người dùng so sánh ý nghĩa giữa chúng. (~ *qui về một “đầu mối” chung*)
- Ví dụ, cùng thể hiện địa chỉ là “Tp. Hồ Chí Minh” nhưng lại có nhiều thể hiện:
  - Hồ Chí Minh
  - Ho Chi Minh
  - Tp. HCM
  - .....

20



### 3. Định dạng dữ liệu

Ví dụ	Ren_luyen	Tam_tru	diem
...	...	...	...
78	KTX	5.7	
87	Ký túc xá	8.6	
78	Túc xá	2.7	
98	KTX	8.5	
100	Ký túc xá	9.3	
99	Ký túc	7.5	
...	...	...	...

#### Chưa định dạng

- Gây nhầm lẫn
- Khó tổng hợp
- Khó so sánh

21

Ví dụ	Ren_luyen	Tam_tru	diem
...	...	...	...
78	KTX	5.7	
87	Ký túc xá	8.6	
78	Túc xá	2.7	
98	KTX	8.5	
100	Ký túc xá	9.3	
99	Ký túc	7.5	
...	...	...	...

#### Đã định dạng

- Rõ ràng
- Dễ tổng hợp
- Dễ phân biệt

### 3. Định dạng dữ liệu

#### Tạo biến mới, hoặc cập nhật lại biến



```
df.rename ( mapper=None,
           index=None,
           columns=None,
           axis=None,
           copy=True,
           inplace=False,
           level=None,
           errors='ignore',
           )
```

#### Parameter:

- columns**: dict-like or function. Alternative to specifying axis (mapper, axis=1 is equivalent to columns=mapper).
- axis** {0 or **index**, 1 or **columns**}, default 0. Axis to target with mapper. Can be either the axis name ('index', 'columns') or number (0, 1). The default is 'index'.

**Returns:** DataFrame



22



### 3. Định dạng dữ liệu

#### Áp dụng

- Chuyển đổi đơn vị tính “**mpg**” sang kiểu mới “**L/100km**” cho bộ dữ liệu.

- 1 MPG = 235,2 Lít mỗi 100 kilômét [l/100km]

	city-mpg	city-L/100km
0	21	11.190476
1	21	11.190476
2	19	12.368421
3	24	9.791667
4	18	13.055556
...	...	...
...	...	...
23		



### 3. Định dạng dữ liệu

#### Định dạng lại kiểu dữ liệu (Sai kiểu dữ liệu gốc)

- Đối khi bị sai về kiểu dữ liệu của một trong bộ dữ liệu, chủ yếu là trong dữ liệu thô.

```
1 df['price']
```

	df.dtypes
0	13495
1	16500
2	16500
3	13950
4	17450
...	...
200	16845
201	19045
202	21485
203	22470
204	22625

Name: price, Length: 205, dtype: object

```
dtype: object
```

```
1 df.dtypes
```

	df.dtypes
symboling	int64
normalized-losses	object
make	object
fuel-type	object
aspiration	object
num-of-doors	object
body-style	object
drive-wheels	object
engine-location	object
wheel-base	float64
length	float64
width	float64
height	float64
curb-weight	int64
engine-type	object
num-of-cylinders	object
engine-size	int64
fuel-system	object
bore	object
stroke	int64
compression-ratio	float64
horsepower	float64
peak-rpm	object
city-mpg	object
highway-mpg	int64
price	object

```
dtype: object
```

...





### 3. Định dạng dữ liệu

#### So sánh kiểu dữ liệu trong Python và Pandas

- Có nhiều kiểu dữ liệu trong Pandas. Cụ thể:

- Objects: "A", "hello"....
- Int64: 1, 2, 3, 4, 5...
- Float64: 10.5, 23.56, 56.9...
- ....

Bài tập nhỏ: Tìm hiểu thêm các kiểu dữ liệu khác trong Pandas?

25



### 3. Định dạng dữ liệu

#### Chỉnh sửa đúng kiểu dữ liệu

Phương pháp	Mô tả
<code>dataframe.dtypes</code>	Kiểm tra kiểu dữ liệu
<code>dataframe.astype('data type')</code>	Chuyển kiểu dữ liệu

- Chuyển kiểu dữ liệu object cột "price" sang kiểu số nguyên (`int`)  
`df['price'] = df['price'].astype('int')`

26



## 4. Chuẩn hóa dữ liệu

### Giới thiệu (Chuẩn hóa dữ liệu <=> Data Normalization)

- Đồng nhất giá trị của các biến trong các phạm vi giá trị khác nhau thành cùng miền giá trị.

```
1 df[['length', 'width', 'height']]
```

length width height

0	168.8	64.1	48.8
1	168.8	64.1	48.8
2	171.2	65.5	52.4
3	176.6	66.2	54.3
4	176.6	66.4	54.3
5	177.3	66.3	53.1
6	192.7	71.4	55.7
7	192.7	71.4	55.7
8	192.7	71.4	55.9

	scale	[150,250]	[50,100]	[50,100]
impact	large	small	small	small

⇒ Sau khi chuẩn hóa xong thì đặc trưng của các mẫu không thay đổi.



27



## 4. Chuẩn hóa dữ liệu

### Ví dụ

Hoc_phi	Ren_luyen	diem
...	...	...
12000000	78	5.7
19000000	87	8.6
18500000	78	2.7
13700000	98	8.5
19700000	100	9.3
16700000	99	7.5
...	...	...



Hoc_phi	Ren_luyen	diem
...	...	...
0.12	0.78	0.57
0.19	0.87	0.86
0.185	0.78	0.27
0.137	0.98	0.85
0.197	1	0.93
0.167	0.99	0.75
...	...	...

### Non-normalized

- “Hoc\_phi”, “Ren\_luyen”, “diem” khác miền dữ liệu
- Khó so sánh. “Hoc\_phi” ảnh hưởng nhiều hơn

### Normalized

- Cùng miền
- ít ảnh hưởng



28



## 4. Chuẩn hóa dữ liệu

### Phương pháp chuẩn hóa dữ liệu

- Hướng tiếp cận cơ bản để chuẩn hóa như:

$$x_{new} = \frac{x_{old}}{x_{max}}$$

scale  
(chia tỉ lệ)

$$x_{new} = \frac{x_{old} - xmin}{x_{max} - xmin}$$

min\_max

$$x_{new} = \frac{x_{old} - \mu}{\sigma}$$

z-score



29



## 4. Chuẩn hóa dữ liệu

### Phương pháp scale

	length	width	height	
0	168.8	64.1	48.8	
1	168.8	64.1	48.8	
2	171.2	65.5	52.4	df['length'] = df['length'] / df['length'].max()
3	176.6	66.2	54.3	df['width'] = df['width'] / df['width'].max()
4	176.6	66.4	54.3	df['height'] = df['height'] / df['height'].max()
5	177.3	66.3	53.1	
6	192.7	71.4	55.7	
7	192.7	71.4	55.7	
8	192.7	71.4	55.9	



30

## 4. Chuẩn hóa dữ liệu

### Phương pháp max\_min

```
df['length'] = (df['length'] - df['length'].min()) / (df['length'].max() - df['length'].min())
df['width'] = (df['width'] - df['width'].min()) / (df['width'].max() - df['width'].min())
df['height'] = (df['height'] - df['height'].min()) / (df['height'].max() - df['height'].min())
```

	length	width	height
0	168.8	64.1	48.8
1	168.8	64.1	48.8
2	171.2	65.5	52.4
3	176.6	66.2	54.3
4	176.6	66.4	54.3
5	177.3	66.3	53.1
6	192.7	71.4	55.7
	...		

31



## 4. Chuẩn hóa dữ liệu

### Phương pháp z-score

```
df['length'] = (df['length'] - df['length'].mean()) / df['length'].std()
df['width'] = (df['width'] - df['width'].mean()) / df['width'].std()
df['height'] = (df['height'] - df['height'].mean()) / df['height'].std()
```

	length	width	height
0	168.8	64.1	48.8
1	168.8	64.1	48.8
2	171.2	65.5	52.4
3	176.6	66.2	54.3
4	176.6	66.4	54.3
5	177.3	66.3	53.1
6	192.7	71.4	55.7

32



32



## 4. Chuẩn hóa dữ liệu

### Sử dụng statsmodels cho z-score

```
scipy.stats.zscore(a, axis=0, ddof=0, nan_policy='propagate')
```

#### Parameters:

- **a**: array\_like. An array like object containing the sample data.
- **axis**: int or None, optional. Axis along which to operate. Default is 0. If None, compute over the whole array a.

#### Returns: zscore array\_like

The z-scores, standardized by mean and standard deviation of input array a.



33



## 4. Chuẩn hóa dữ liệu

### Áp dụng

```
1 | scipy.stats.zscore(df[['length', 'width', 'height']])  
  
array([[-0.42652147, -0.84478235, -2.0204173 ],  
      [-0.42652147, -0.84478235, -2.0204173 ],  
      [-0.23151305, -0.19056612, -0.54352748],  
      [ 0.2072559 , 0.13654199, 0.23594216],  
      [ 0.2072559 , 0.23000146, 0.23594216],  
      [ 0.26413336, 0.18327172, -0.25635445],  
      [ 1.5154374 , 2.56648799, 0.8102882 ],  
      [ 1.5154374 , 2.56648799, 0.8102882 ],  
      [ 1.5154374 , 2.56648799, 0.8102882 ],  
      [ 0.33726152, 0.93094742, -0.70762635],
```



34



## 5. Chuyển giá trị số thành các giá trị phân loại

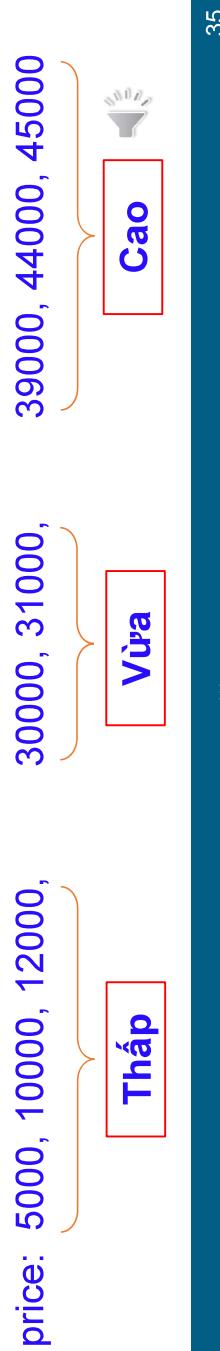
### Giới thiệu

- Giá trị phân loại (categorical variable)
- Nhóm các giá trị số thành các giá trị phân loại (còn gọi là “bin”)
- Hay, gom nhóm các giá trị số thành một tập “bin”

### Ví dụ:

Thuộc tính “price” có miền giá trị 5,000 đến 45,500

Được chuyển đổi thành các giá trị như “Thấp”, “Vừa”, “Cao”



35



## 5. Chuyển giá trị số thành các giá trị phân loại

### Phương pháp:

- Bước #1: Tìm Max và Min
  - Bước #2: Xác định số bin cần tạo (tức số lượng các giá trị phân loại)
- Dùng:
- `numpy.linspace(start, stop, num=...)` => trả về list các mốc giá trị.
  - `pandas.cut(X, bins, right=True, labels=None/Group, retbins=False, precision=3, include_lowest=False, duplicates='raise', ordered=True)`

	Bin 1	Bin 2	Bin 3	Bin 4	Bin 5	Bin 6	Bin 7	Bin 8	Bin 9	Bin 10
Range	0. to 3412.9	3412.9 to 6825.8	6825.8 to 10238.7	10238.7 to 13651.6	13651.6 to 17064.5	17064.5 to 20477.4	20477.4 to 23890.3	23890.3 to 27303.2	27303.2 to 30716.1	30716.1 to 34129.

36



## 5. Chuyển giá trị số thành các giá trị phân loại

### Áp dụng

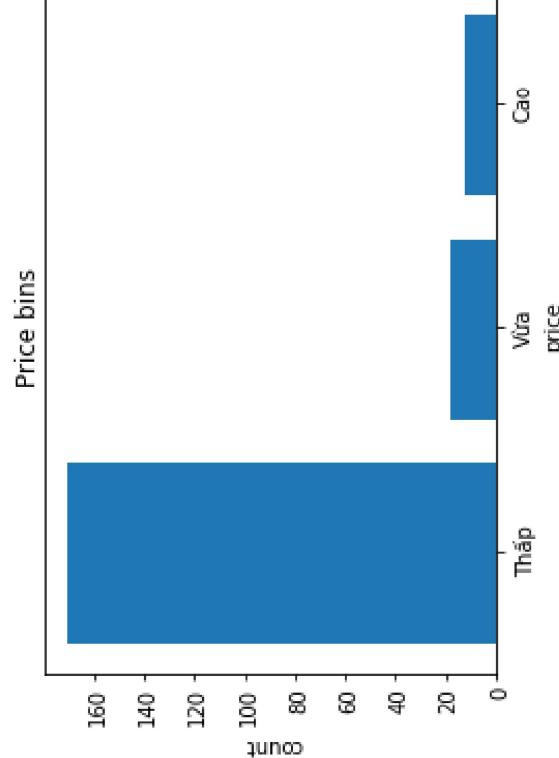
price	price-bin
7299	Thấp
22018	Vừa
8238	Thấp
34184	Cao
41315	Cao
16558	Thấp
6338	Thấp
...	...

37



## 5. Chuyển giá trị số thành các giá trị phân loại

### Trực quan các giá trị phân loại



```
%matplotlib inline
import matplotlib as plt
from matplotliblib import pyplot
pyplot.bar(group_names,
df[["price-bin"]].value_counts())
plt.pyplot.xlabel("price")
plt.pyplot.ylabel("count")
plt.pyplot.title("Price bins")
```

38



## 6. Chuyển các giá trị phân loại thành giá trị số

### Khó khăn?

- Các mô hình thống kê/ML không thể lấy biến là các kiểu **object/string**

fuel-type	drive-wheels	body-style	price
gas	fwd	hatchback	8949
gas	rwd	hatchback	18399
gas	fwd	wagon	8921
gas	fwd	wagon	14399
gas	fwd	wagon	8921
diesel	rwd	sedan	16900
gas	rwd	wagon	12440
gas	rwd	sedan	15580
		...	

39



Vì vậy, phải mã hóa (encode) sang dạng số để các mô hình phân tích có thể dùng được.



6. Chuyển các giá trị phân loại thành giá trị số

### Giải pháp:

- Thêm các biến khác biểu diễn cho duy nhất mỗi giá trị phân loại.
- Chỉ định **0** hoặc **1** cho mỗi giá trị phân loại.
- Đây là phương pháp mã hóa "**one-hot**", hoặc gọi là "**one-hot encoding**"

Name	Fuel	.....	gas	diesel
A	gas	.....	1	0
B	diesel	.....	0	1
C	gas	.....	1	0
D	gas	.....	1	0



40



## 6. Chuyển các giá trị phân loại thành giá trị số

### Kỹ thuật

signature:

```
pd.get_dummies(  
    data,  
    prefix=None,  
    prefix_sep='__',  
    dummy_na=False,  
    columns=None,  
    sparse=False,  
    drop_first=False,  
    dtype=None,  
    ) -> 'DataFrame'
```

#### Docstring:

Convert categorical variable into dummy/indicator variables.



41



## 6. Chuyển các giá trị phân loại thành giá trị số

### Áp dụng

- Dùng phương thức `pandas.get_dummies()`
- Chuyển giá trị phân loại sang giá trị số (**0**, hoặc **1**)

fuel-type	diesel	gas
gas	0	1
gas	0	1
gas	0	1
diesel	1	0
gas	0	1

```
pd.get_dummies(df['fuel-type'])
```

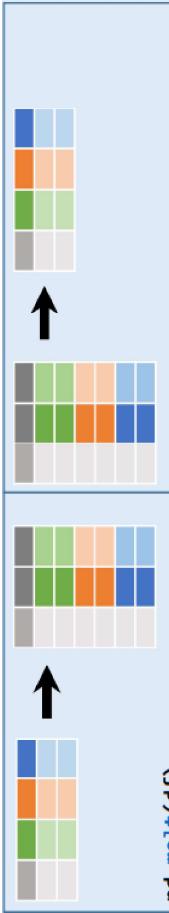
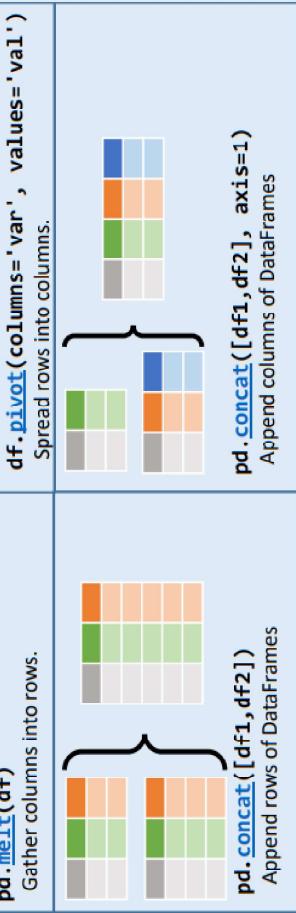
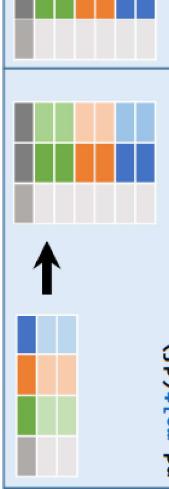


42



## Tóm tắt

Ref.: [https://pandas.pydata.org/Pandas\\_Cheat\\_Sheet.pdf](https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf)

<p><code>pd.melt(df)</code> Gather columns into rows.</p> 	<p><code>df.pivot(columns='var', values='val')</code> Spread rows into columns.</p> 
<p><code>pd.concat([df1, df2])</code> Append rows of DataFrames</p> 	<p><code>pd.concat([df1, df2], axis=1)</code> Append columns of DataFrames</p> 



Thank you

## Hỏi - Đáp



46

# PHẦN TÍCH THĂM DÒ DỮ LIỆU

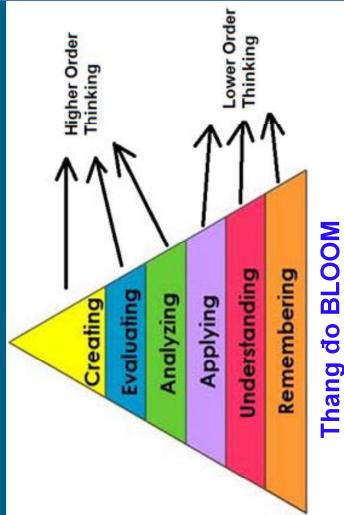
## Chương 3



## Mục tiêu



- Vận dụng thống kê mô tả
- Vận dụng độ tương quan
- Phân tích gom nhóm dữ liệu cơ bản
- Vận dụng phân tích phương sai



2

## Nội dung



1. Phân tích thăm dò dữ liệu
2. Thống kê mô tả
3. Gom nhóm dữ liệu
4. Phân tích ANOVA

3



## Bộ dữ liệu

Mình họa học lý thuyết và thực hành:

[https://raw.githubusercontent.com/datasetshub/ds105/master/EDA\\_automobile.csv](https://raw.githubusercontent.com/datasetshub/ds105/master/EDA_automobile.csv)

symboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	width	height	curb-weight	engine-type	num-of-cylinders	engine-size	
0	3	122	alfa-romero	std	twc	convertible	nwd	front	88.6	0.81148	0.850278	48.8	2548	dohc	four	130
1	3	122	alfa-romero	std	twc	convertible	nwd	front	88.6	0.81148	0.850278	48.8	2548	dohc	four	130
2	1	122	alfa-romero	std	twc	hatchback	nwd	front	94.5	0.822681	0.909722	52.4	2823	ohcv	six	152
3	2	164	audi	std	four	sedan	fwd	front	99.8	0.848630	0.919444	54.3	2337	ohc	four	109
4	2	164	audi	std	four	sedan	4wd	front	99.4	0.848630	0.922222	54.3	2824	ohc	five	135
5	2	122	audi	std	two	sedan	fwd	front	99.8	0.851994	0.920833	53.1	2507	ohc	five	135
6	1	158	audi	std	four	sedan	fwd	front	105.8	0.925997	0.991667	55.7	2844	ohc	five	135
7	1	122	audi	std	four	wagon	fwd	front	105.8	0.925997	0.981667	55.7	2954	ohc	five	135
8	1	158	audi	turbo	four	sedan	fwd	front	105.8	0.925997	0.981667	55.9	3086	ohc	five	131
9	2	192	bmw	std	twc	sedan	nwd	front	101.2	0.849592	0.900000	54.3	2395	ohc	four	103
								...							4	



## 1. Phân tích thăm dò dữ liệu

### EDA?

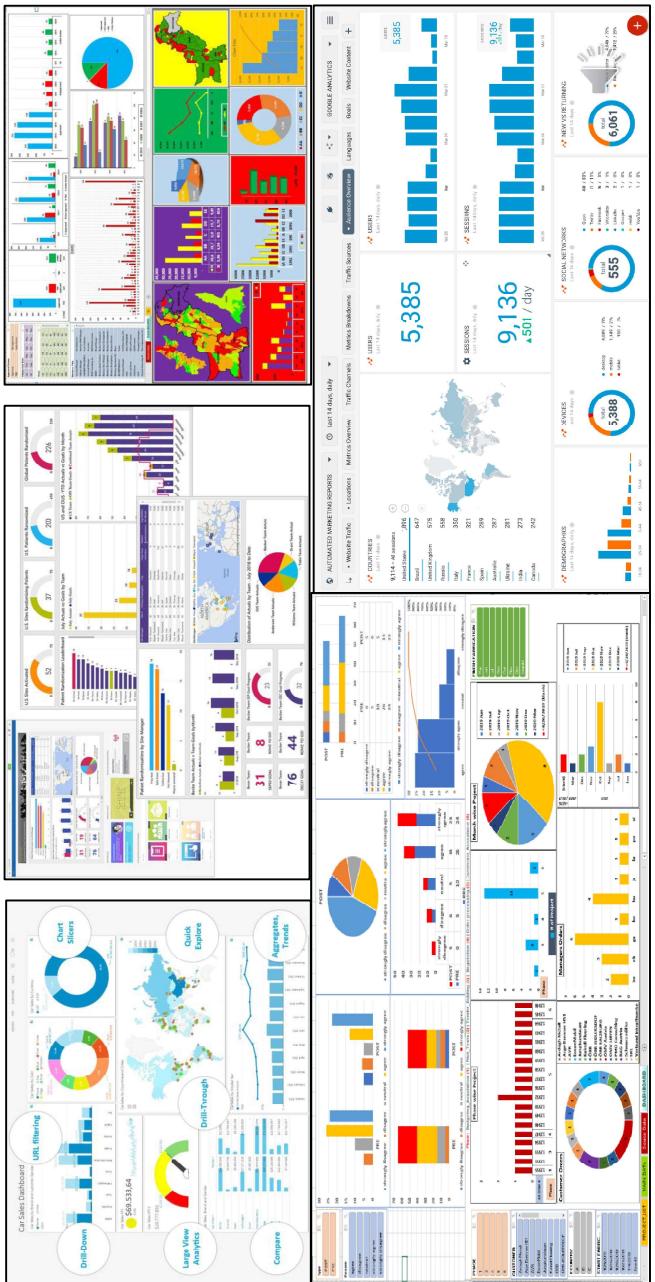
- EDA là một cách tiếp cận để phân tích dữ liệu. (*khi bắt đầu từ 1 nguồn mới*)
- Các bước sơ bộ trong phân tích dữ liệu để:
  - Tóm tắt các đặc tính chính của dữ liệu
  - Hiểu rõ hơn về tập dữ liệu
  - Phát hiện mối quan hệ khác nhau giữa các biến/thuộc tính/cột
  - Trích xuất/chọn ra các biến quan trọng

- Câu hỏi giải quyết vấn đề đặt ra:

“Những đặc điểm nào có ảnh hưởng nhất đến giá xe?”



## 1. Phân tích thăm dò dữ liệu

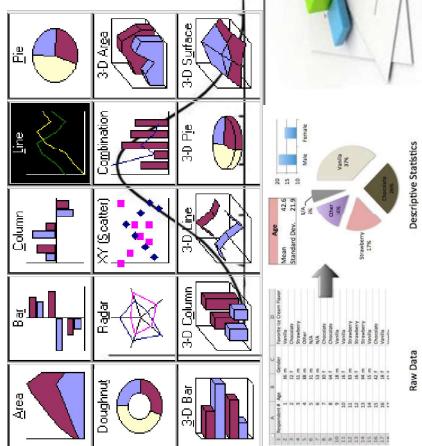


6

## 2. Thống kê mô tả

### Giới thiệu

- Mô tả các đặc tính cơ bản của dữ liệu.
- Đưa ra các tóm tắt ngắn về mẫu và thang đo của dữ liệu.
- Là một trong những hướng tiếp cận để thăm dò dữ liệu.



7



## 2. Thống kê mô tả

### Thăm dò các đặc tính sau:

- 2.1. Hướng trung tâm (Central tendencies)
- 2.2. Độ phân tán (Dispersion)
- 2.3. Hình dáng của dữ liệu (Shape)
- 2.4. Tương quan (Correlation)



8



## 2. Thống kê mô tả

### Thăm dò chi tiết các đặc tính sau:

- 2.1. Central tendencies (Hướng trung tâm)
  - Mean
  - Median
  - Mode
- 2.2. Dispersion (Độ phân tán)
  - Range
  - Quartile
  - Interquartile Range (IQR)
  - Variance
  - Standard deviation (std)
  - Coefficient of Variation (CV)
- 2.3. Shape (Hình dáng)
  - Skewness
  - Kurtosis
- 2.4. Correlation (Tương quan)

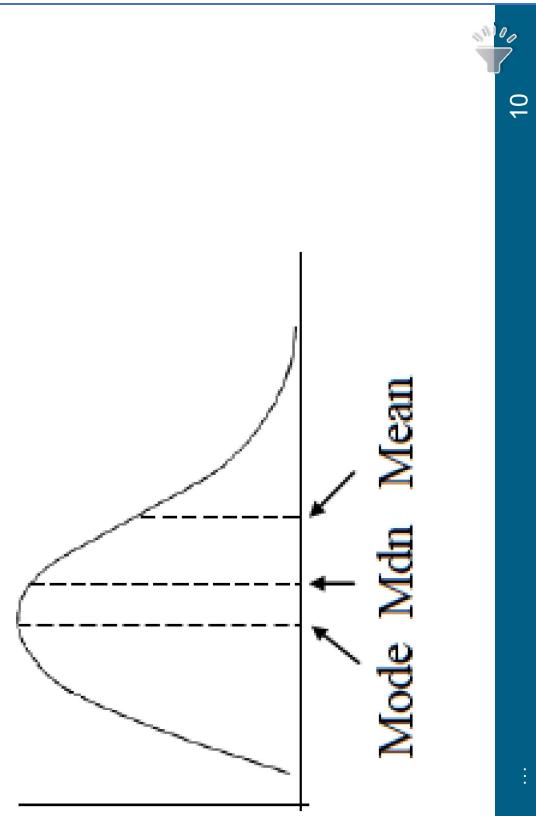


9

## 2.1. Central tendencies (Hướng trung tâm)

### Giới thiệu

- Là thước đo tóm tắt, khái quát mức độ tổng thể của từng biến trong bộ dữ liệu
- Gồm
  - Mean
  - Median
  - Mode



### 2.1. Central tendencies

#### Mean

- Trung bình/bình quân của một bộ dữ liệu.
- Trong thống kê mô tả thì mean là trung bình cộng.
- Không phụ thuộc thứ tự các phần tử trong bộ dữ liệu.
- Công thức:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

#### Ý nghĩa của mean?

- Hướng trung tâm về mặt giá trị.

## 2.1. Central tendencies

### Median

- Các giá trị của biến được sắp xếp có thứ tự thì giá trị ở giữa gọi là median (**trung vị**).
- Hoặc, median là giá trị nằm ở giữa tập dữ liệu sau khi được sắp xếp.

### Ý nghĩa của median là gì?

- Thể hiện đặc trưng cho điểm giữa của sự phân tán giá trị của dữ liệu, còn được gọi là thống kê xu hướng trung tâm, hay thống kê vị trí.
- 50% giá trị mẫu là nhỏ hơn trung vị.
- Trung tâm về mặt vị trí.



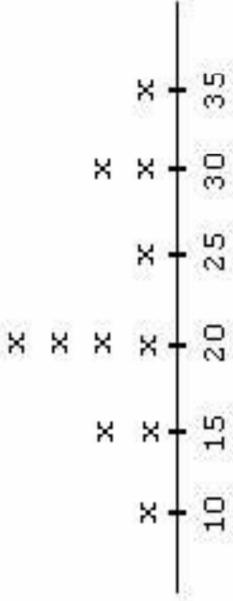
12



## 2.1. Central tendencies

### Mode

- Có thể gọi là yếu vị.
- Là giá trị xuất hiện nhiều nhất trong bộ dữ liệu.
- Tập dữ liệu có 1 mode, hoặc tập dữ liệu có đến 2 hoặc 3 mode, hoặc tập dữ liệu không có mode nào.



### Ý nghĩa của mode?

- Trung tâm về mức độ tập trung dữ liệu.



13

## 2.1. Central tendencies

### Phương pháp

Phương thức	Mô tả
df.mean()	Tính trung bình
df.median()	Tính trung vị
df.mode()	Tính yếu vị
df.describe()	Tính cả 3 trung bình, trung vị, yếu vị



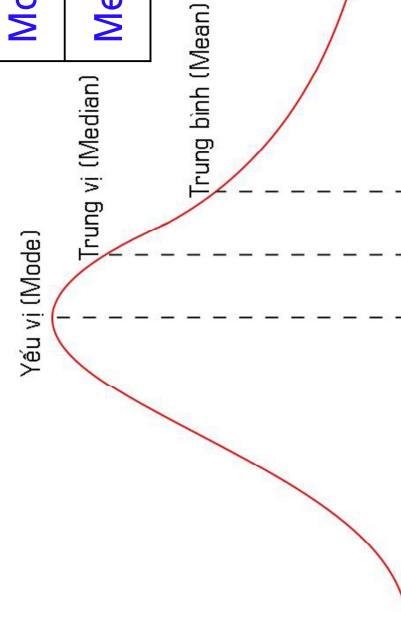
14



## 2.1. Central tendencies

### Đồ thị thể hiện

Mean = Median = Mode	Đối xứng (symmetry)
Mode < Median < Mean	Tích cực (positive)
Mean < Median < Mode	Không tốt (negative)



15



## 2.1. Central tendencies

### Bài tập

- Tính mean, median, mode cho biến “price” và “length”



16



## 2.2. Dispersion (Hướng phân tán)

### Tính toán các đại lượng sau

- Range - Khoảng biến thiên
- Quartile - Tứ phân vị
- Interquartile Range (IQR) - Độ trai giữa (khoảng tứ phân vị)
- Variance - Phương sai
- Standard deviation (std) - Độ lệch chuẩn
- Coefficient of Variation (CV) - Hệ số biến thiên



17



## 2.2. Dispersion

### Range

- Khoảng biến thiên
- Cho bộ dữ liệu với các giá trị quan sát được sắp xếp theo thứ tự:

$$X_1, X_2, X_3, \dots, X_n$$

Trong đó:

- Min =  $X_1$
- Max =  $X_n$

- Công thức:  $\text{Range} = \text{Max} - \text{Min} = X_n - X_1$

**Ý nghĩa của Range trong thống kê là gì?**

- Khoảng biến thiên sẽ bị ảnh hưởng bởi các giá trị ngoại lệ (Outliers)



18

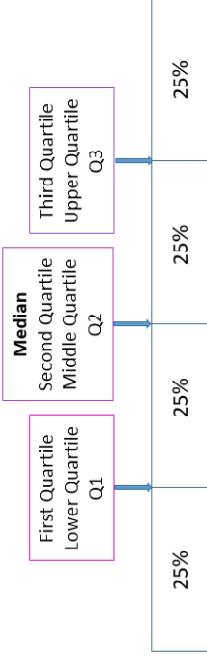


## 2.2. Dispersion

### Quartile

- Chia tập dữ liệu thành 04 phần có số lượng quan sát nhau nhau, mỗi phần có chứa khoảng 1/4, tức 25% giá trị quan sát.

- Quartile cung cấp thông tin về các giá trị quan sát được phân bố từ Min tới Max.



19



## 2.2. Dispersion

### Quartile

- Phân vị mẫu thứ nhất  $Q_1$  (lower quartile)
- Phân vị mẫu thứ hai  $Q_2$  = Median, còn gọi là sample median.

- Phân vị mẫu thứ ba  $Q_3$  (upper quartile)
- Tính từ phân vị:

$$Q_1 = X_{(q1)}$$

$$Q_3 = X_{(q3)}$$

với

$$q_1 = \frac{n+1}{4}$$

$$q_3 = \frac{3(n+1)}{4}$$



## 2.2. Dispersion

### Interquartile Range (IQR)

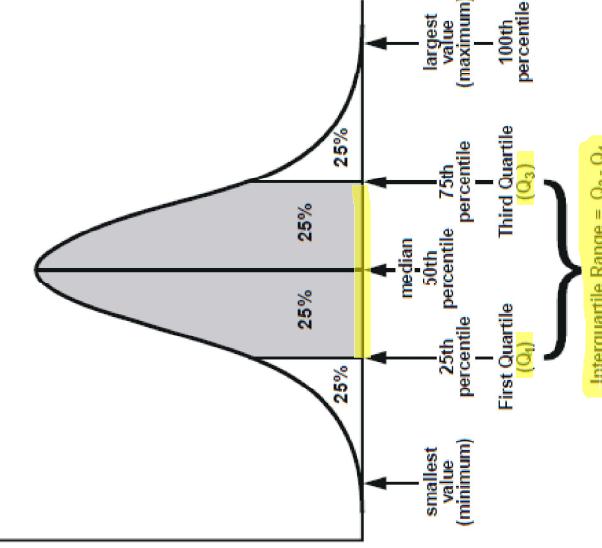
- Độ trai giữa (khoảng từ phân vị)

$$IQR = Q_3 - Q_1$$

### Ý nghĩa của IQR?

- Thể hiện khoảng rộng hay “sự phân tán” của tập số.

The middle half of the observations in a frequency distribution lie within the interquartile range





## 2.2. Dispersion

### Variance (Phương sai)

- Phương sai của một tập mẫu là trung bình của bình phương độ lệch của các quan sát so với trung bình của chúng.

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2$$

- Phương sai mẫu thì  $n - 1$  để bù độ lệch gây ra do cách dùng giá trị trung bình

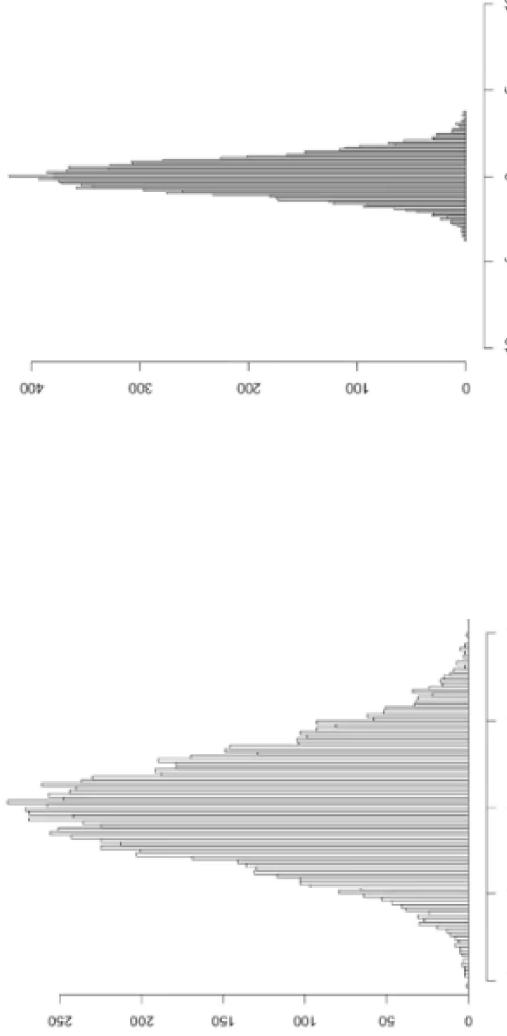
### Ý nghĩa của phương sai?

- Đo lường mức độ lan rộng (sự phân tán/phân bố) của dữ liệu xung quanh giá trị trung bình.

22



## 2.2. Dispersion



Thể hiện phương sai khác nhau

23





## 2.2. Dispersion

### Standard deviation (std)

- Standard deviation (std): Độ lệch chuẩn
- Độ lệch chuẩn đo độ rộng của dữ liệu, được trải rộng xung quanh giá trị trung bình.
- Công thức:

$$\text{std} = \sqrt{s^2}$$



24



## 2.2. Dispersion

### Coefficient of Variation (CV)

- Là hệ số biến thiên.
- CV thường dùng để đo mức độ phân tán của các biến xung quanh giá trị trung bình của nó.
- Công thức:

$$CV = \frac{\sigma}{\mu} \times 100\%$$



25



## 2.2. Dispersion

### Áp dụng - `describe()`

- Phương thức `describe()` thực hiện tóm tắt các thông số thống kê của các biến trong bộ dữ liệu.

`df.describe()`

	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size
count	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000
mean	0.840796	122.000000	98.797015	0.837102	0.915126	53.766667	2555.666667	126.875622
std	1.254802	31.99625	6.066366	0.059213	0.029187	2.447822	517.296727	41.546834
min	-2.000000	65.000000	86.600000	0.678039	0.837500	47.800000	1488.000000	61.000000
25%	0.000000	101.000000	94.500000	0.801538	0.890278	52.000000	2169.000000	98.000000
50%	1.000000	122.000000	97.000000	0.832292	0.909722	54.100000	2414.000000	120.000000
75%	2.000000	137.000000	102.400000	0.881788	0.925000	55.500000	2926.000000	141.000000
max	3.000000	256.000000	120.900000	1.000000	1.000000	59.800000	4066.000000	326.000000

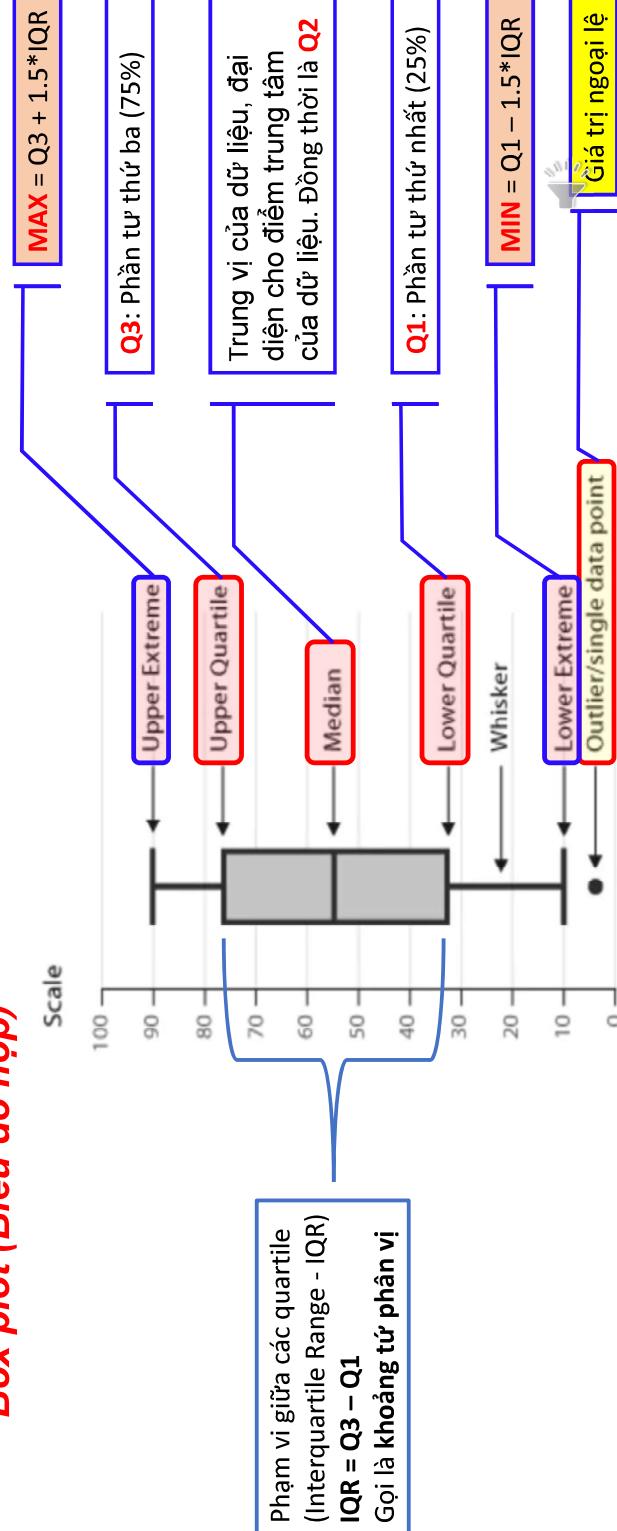
...

26



### Box plot (Biểu đồ hộp)

## 2.2. Dispersion



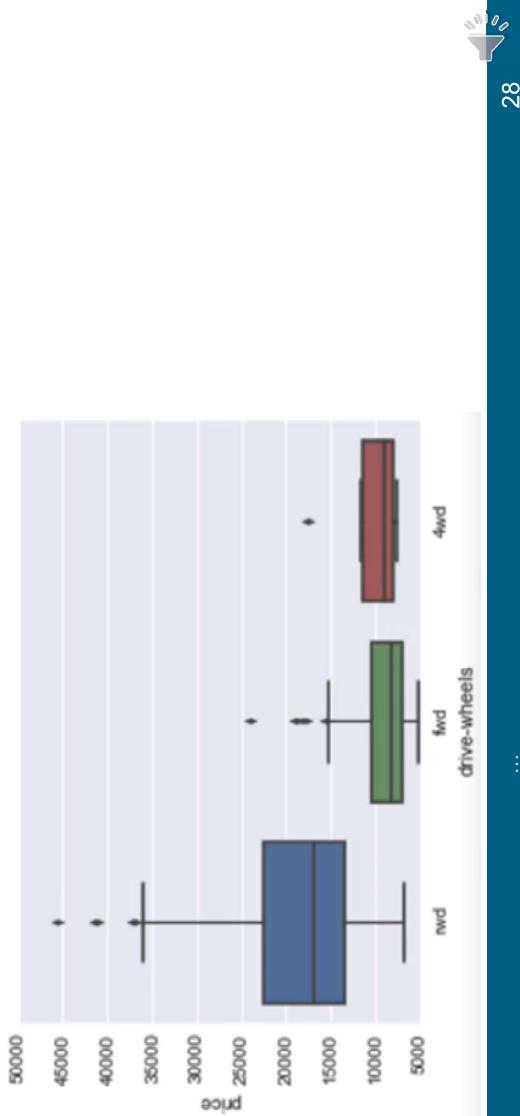
27



## 2.2. Dispersion

### Box plot – Áp dụng minh họa

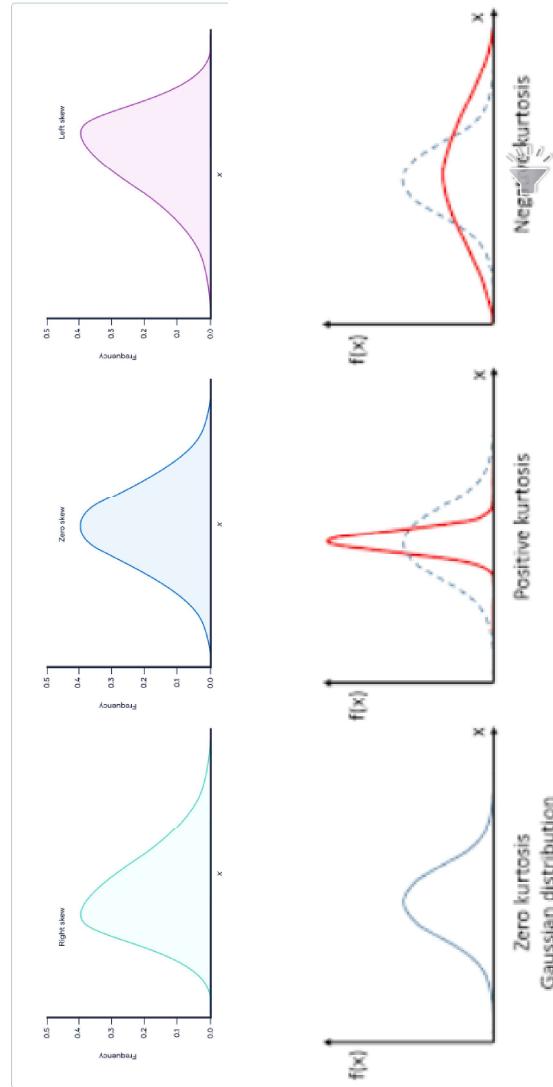
```
sns.boxplot(x= "drive-wheels", y= "price", data=df)
```



## 2.3. Shape of Data (Hình dạng của dữ liệu)

- Hình dạng của dữ liệu được do bằng

- **Skewness (Độ lệch)**
- **Kurtosis (Độ nhọn)**



### 2.3. Hình dạng của dữ liệu

#### Skewness

- Dùng giá trị trung bình làm cơ sở trung tâm để xác định sự chuyển dịch của dữ liệu.
- Công thức:

$$skewness = \frac{1}{N} \frac{\sum_{i=1}^N (x_i - \mu)^3}{\sigma^3}$$

$$skewness = \frac{n}{(n-1)(n-2)} \frac{\sum_{i=1}^n (X_i - \bar{X})^3}{s^3}$$

...  
30  
...

### 2.3. Hình dạng của dữ liệu

#### Skewness

Lệch trái	Cân đối	Lệch phải
Negatively Skewed	Normal (no skew)	Positively Skewed



The normal curve represents a perfectly symmetrical distribution

...  
31  
...



### 2.3. Hình dạng của dữ liệu

#### Ý nghĩa hệ số Skewness trong phân tích dữ liệu

- Sử dụng hệ số skewness kết hợp với biểu đồ để đánh giá phân phối của bộ dữ liệu.
- Đánh giá độ lệch nghiêng so với giá trị trung bình theo xu thế tích cực hay tiêu cực.



32



### 2.3. Hình dạng của dữ liệu

#### Kurtosis

- Kurtosis dựa vào độ cao của điểm đỉnh (peakedness).
- Kurtosis càng cao thì càng có nhiều ngoại lệ và đuôi càng dài.
- Công thức:

$$\text{kurtosis} = \frac{1}{N} \frac{\sum_{j=1}^N (x_j - \mu)^4}{\sigma^4}$$

$$\text{kurtosis} = \frac{n(n+1)}{(n-1)(n-2)(n-3)} \frac{\sum_{j=1}^n (x_j - \bar{x})^4}{s^4}$$

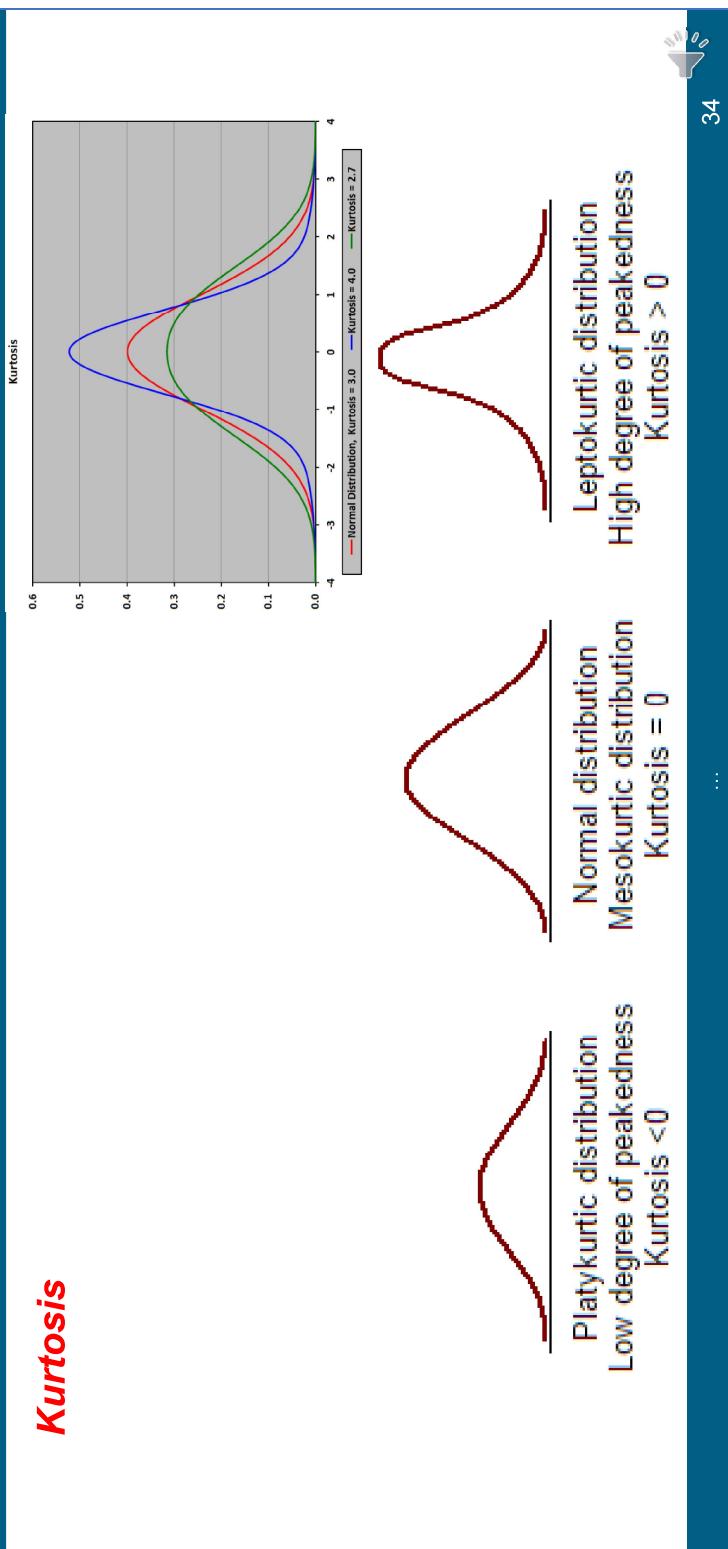


33



### 2.3. Hình dạng của dữ liệu

#### Kurtosis



34



### 2.3. Hình dạng của dữ liệu

#### Ý nghĩa hệ số Kurtosis trong phân tích dữ liệu

- Hệ số kurtosis là một thước đo tốt để đánh giá có những vấn đề nào liên quan đến các giá trị ngoại lệ trong bộ dữ liệu.

- Khi giá trị **kurtosis lớn** cho thấy tập dữ liệu gấp ván đề nghiêm trọng đối với các giá trị ngoại lệ

35

## 2.4. Correlation (Độ tương quan)

### Correlation là gì?

- Thể hiện mức độ **phụ thuộc lẫn nhau** của các biến khác nhau.
- Ảnh hưởng giữa hai chiều dữ liệu.

- Ví dụ về tương quan:

- Ung thư gan có liên quan đến rượu, bia
- Giá vàng có liên quan giá dầu

- Tương quan **KHÔNG** thể hiện mối quan hệ kéo theo.

- **KHÔNG** có tính “bắc cầu”

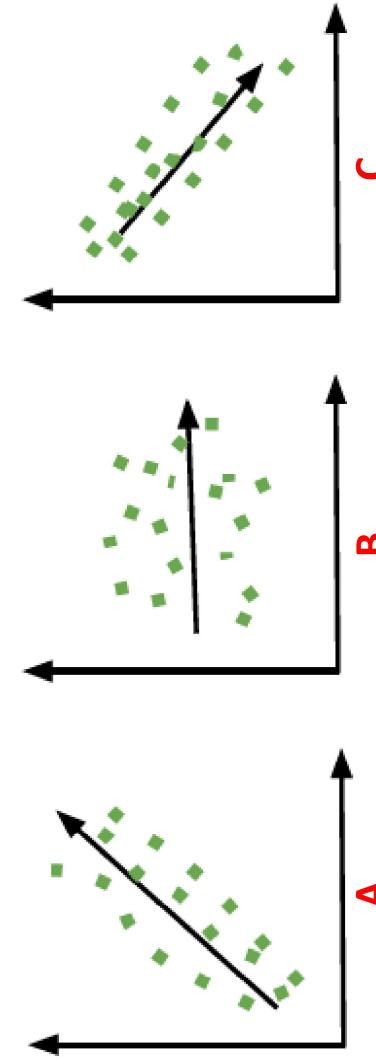


37

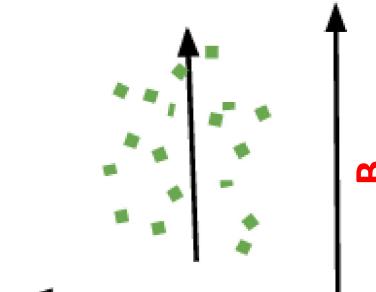


## 2.4. Correlation

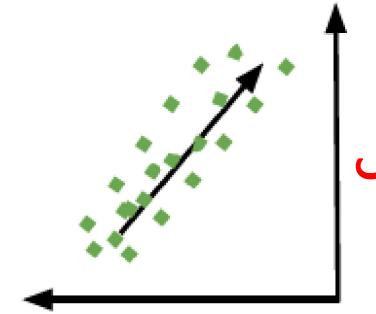
- Để thể hiện mối quan hệ giữa 2 biến là “mạnh” hay “yếu”..., chúng ta sử dụng **correlation**



Positive  
Correlation



Zero  
Correlation



Negative  
Correlation



38

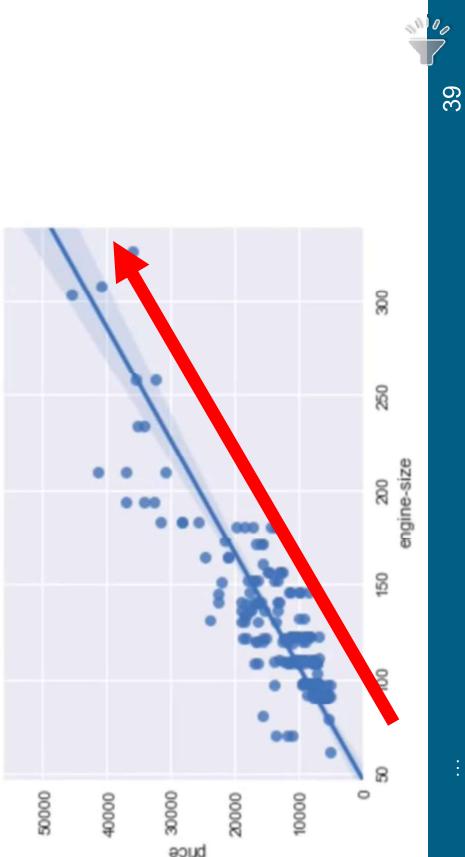


## 2.4. Correlation

### Dùng trực quan

- Tương quan giữa hai biến engine-size và price

```
sns.regplot(x= "engine-size", y= "prices", data=df)  
plt.ylim(0, )
```



39

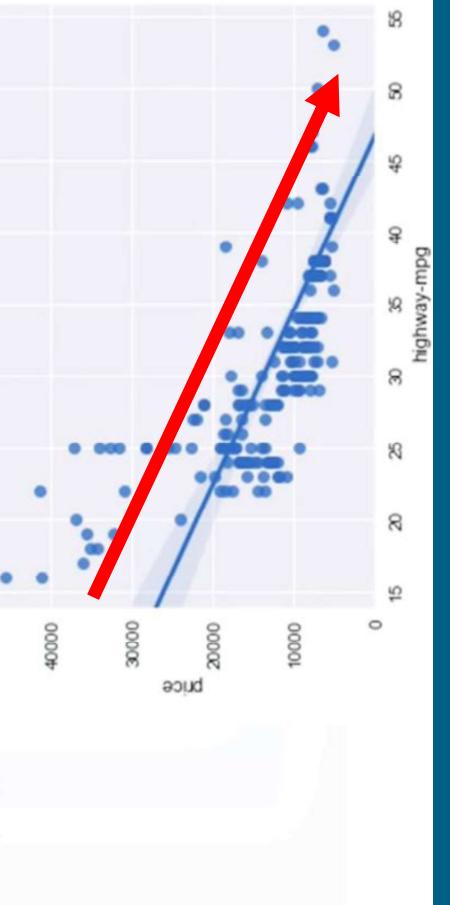


## 2.4. Correlation

### Dùng trực quan

- Tương quan giữa hai biến highway-mpg và price

```
sns.regplot(x= "highway-mpg", y= "prices", data=df)  
plt.ylim(0, )
```



40

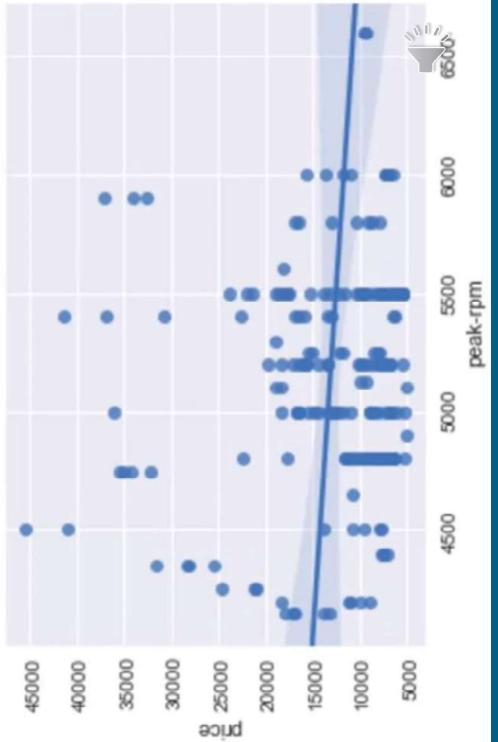


## 2.4. Correlation

### Dùng trực quan

- Tương quan (yếu) giữa hai biến peak-rmp và price

```
sns.regplot(x= "peak-rmp", y= "prices", data=df)  
plt.ylim(0, )
```



41



## 2.4. Correlation

### Hệ số tương quan

- Thể hiện mối quan hệ giữa 2 biến là “mạnh” hay “yếu” ...

	Hệ số tương quan		
	Đồng thuận	Không đồng thuận	
<b>Mạnh</b>	0.8 đến 1	-1 đến -0.8	
<b>Vừa</b>	0.5 đến 0.8	-0.8 đến -0.5	
<b>Yếu</b>	0.3 đến 0.5	-0.5 đến -0.3	
<b>Không</b>	0 đến 0.3	-0.3 đến 0	



42



## 2.4. Correlation - Statistics

### Tương quan Pearson

Dùng để đánh giá mức độ tương quan tuyến tính

- Đánh giá tương quan giữ 2 biến.
  - Hệ số tương quan tuyến tính
    - P-value: thể hiện mức độ chắc chắn về mối tương quan đã tính toán
- Hệ số tương quan**
  - Close to **+1**: Large positive relationship
  - Close to **-1**: Large negative relationship
  - Close to **0**: No relationship

### P-value

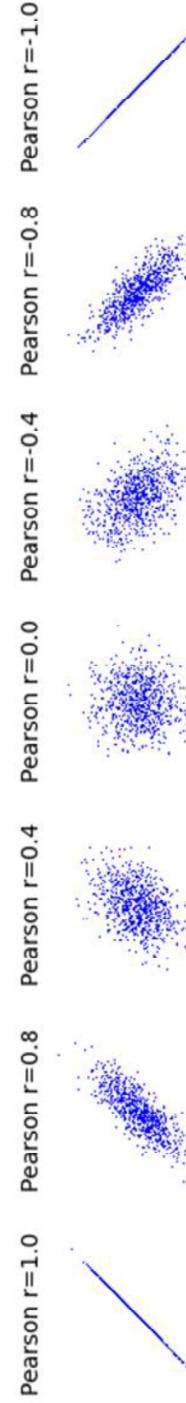
- P-value < 0.001 **Strong**
- P-value < 0.05 **Moderate** (Vừa phải)
- P-value < 0.1 **Weak**
- P-value > 0.1 **No**

43



## 2.4. Correlation - Statistics

### Tương quan Pearson



44

## 2.4. Correlation - Statistics

### Minh họa tương quan Pearson

- Xét mối quan hệ tương quan giữa 2 biến horsepower và price?

```
Pearson_coef, p_value=stats.personr [ 'horsepower' ], df [ 'price' ] ]
```

- Hệ số tương quan Pearson: **0.81**

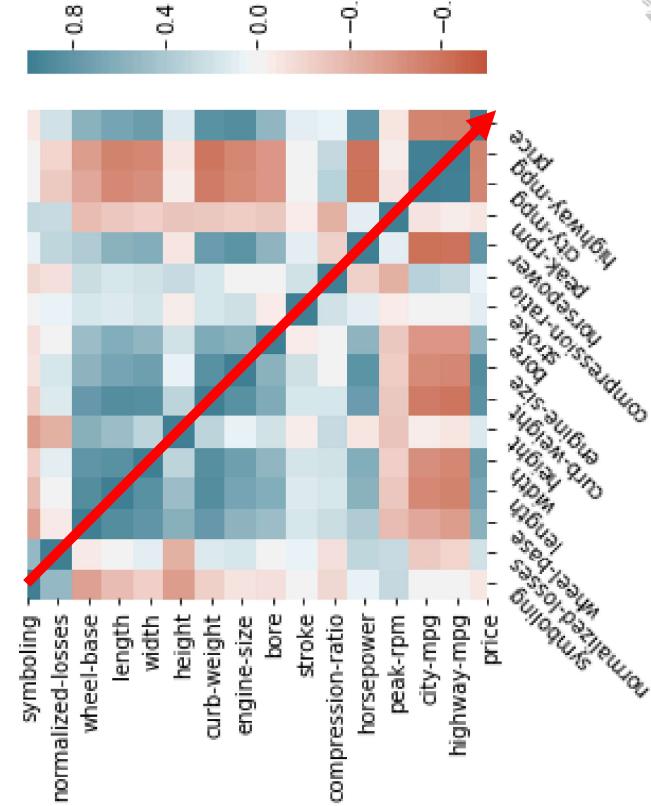
- Giá trị P-value: **9.35e-48**

45



## 2.4. Correlation - Statistics

### Bản đồ nhiệt tương quan (Correlation heatmap)



46





### 3. Gom nhóm

#### Phương thức `df.groupby()`

- Để gom nhóm dữ liệu, dùng phương thức `groupby()`:
  - Áp dụng được cho các biến dạng phân loại
  - Gom nhóm dữ liệu thành các phân loại khác.
  - Áp dụng được cho đơn biến hoặc đa biến.
- Đánh giá xem liệu có mối ảnh hưởng nào về giá của các giá trị trong hai biến phân loại “drive-wheels” và “body-style”.

...  
47



### 3. Gom nhóm

- Buớc #1**  
`df_test = df[['drive-wheels', 'body-style', 'price']]  
df_test`

- Buớc #2**  
`df_grp = df_test.groupby(['drive-wheels', 'body-style'], as_index = False).mean()`

	drive-wheels	body-style	price
0	rwd	convertible	13495.0
1	rwd	convertible	16500.0
2	rwd	hatchback	16500.0
3	fwd	sedan	13950.0
4	4wd	sedan	17450.0
5	fwd	sedan	15250.0
6	fwd	sedan	17710.0
7	fwd	wagon	18920.0
8	fwd	sedan	23875.0
10	rwd	sedan	16430.0
12	rwd	wagon	16954.222222
...			



### 3. Gom nhóm

#### Phương thức Pivot()

- Một biến được trình bày theo cột, các biến còn lại trình bày theo dòng

```
df_pivot = df_grp.pivot(index= 'drive-wheels', columns='body-style')
```

	price	convertible	hardtop	hatchback	sedan	wagon
body-style						
drive-wheels						
4wd	20239.229524	20239.229524	7603.000000	12647.333333	9095.750000	
fwd	11595.000000	8249.000000	8396.387755	9811.800000	9997.333333	
rwd	23949.600000	24202.714286	14337.777778	21711.833333	16994.222222	

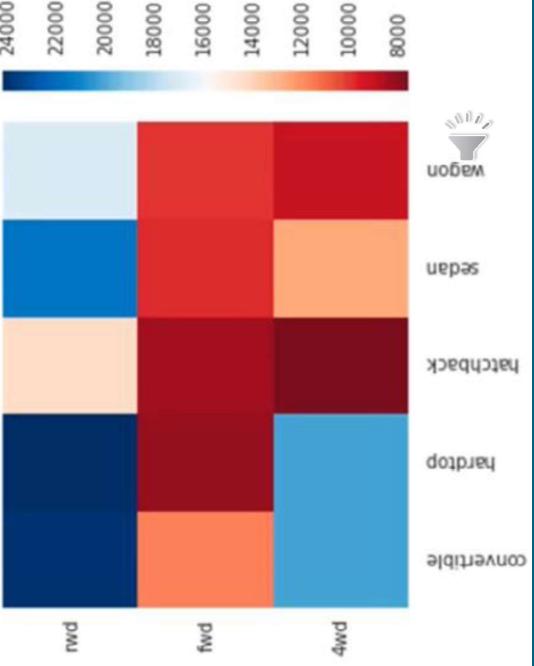
49

### 3. Gom nhóm

#### Dùng heatmap

```
plt.pcolor(df_pivot, cmap='RdBu')
plt.colorbar()
plt.show()
```

- Quy tắc màu của heatmap?



50



### 3. Gom nhóm

#### Phương thức `value_counts()`

- Phương thức `value_counts()` dùng để đếm các giá trị dạng phân loại.
- Ví dụ, đếm các giá trị trong biến “drive-wheels”

```
drive_wheels_counts=df[ "drive-wheels" ].value_counts() .to_frame()

drive_wheels_counts.rename(columns={ 'drive-wheels' : 'value_counts' , inplace=True })

drive_wheels_counts.index.name= 'drive-wheels'
```

	value_counts
drive-wheels	
fwd	118
rwd	75
4wd	8

51



### 4. Phân tích ANOVA

#### Tại sao phân tích ANOVA (phân tích phương sai)?

- Xem xét khả năng ảnh hưởng giữa các nhóm khác nhau trong cùng một biến dạng phân loại.

- Phương pháp t-test dùng để kiểm định hai nhóm (biến binary).

- Tương tự, Point Biserial correlation

- Ví dụ: so sánh xe chạy bằng xăng hay dầu, loại nào ảnh hưởng đến giá?

- Phân tích ANOVA áp dụng cho kiểm định cho nhiều nhóm.
  - Ví dụ: kiểm tra xem **honda**, **mec**, **toyota** và **bmw**, hãng nào ảnh hưởng đến giá xe hơn?

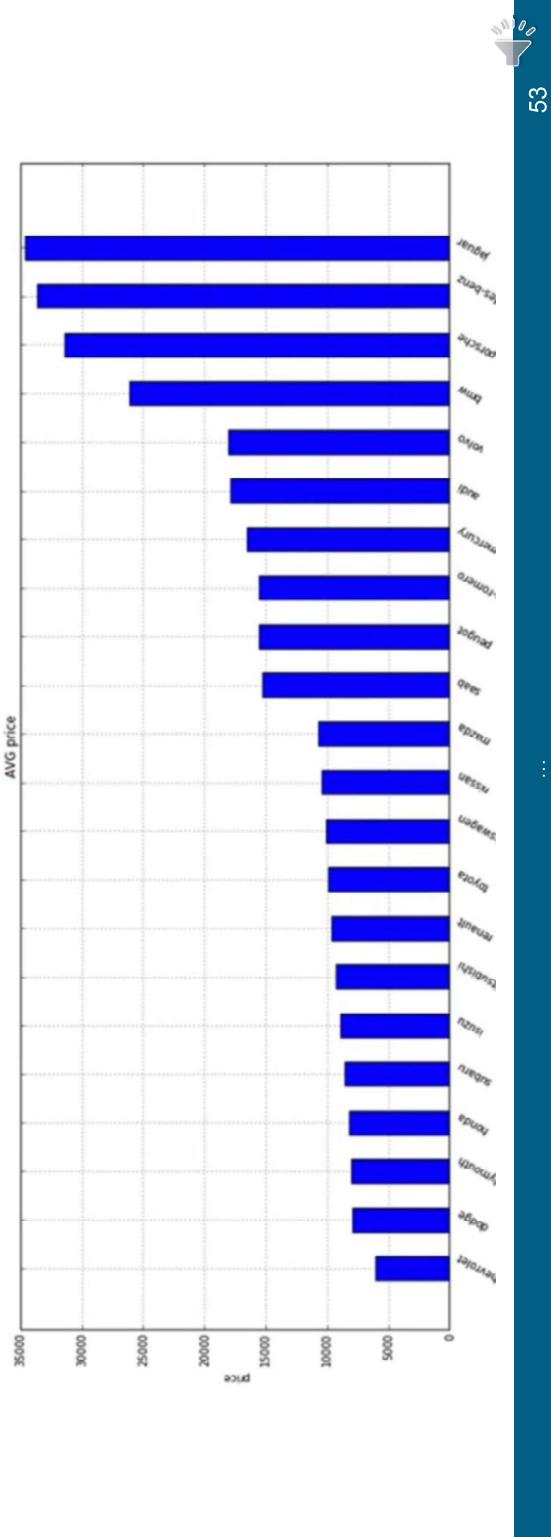


52



## 4. Phân tích ANOVA

- Ví dụ: so sánh xe **honda** và **bmw**, hãng nào ảnh hưởng giá cao hơn?



53



## 4. Phân tích ANOVA

**Phân tích ANOVA cần xác định hai giá trị sau?**

- F-test** được tính bằng:

Bình phương trung bình giữa các nhóm (*variation between groups*)  
\_\_\_\_\_  
Bình phương trung bình trong mỗi nhóm (*variation within group*)

- P-value:** (confidence degree/mức độ tự tin) kết quả thu được có ý nghĩa thống kê hay không.

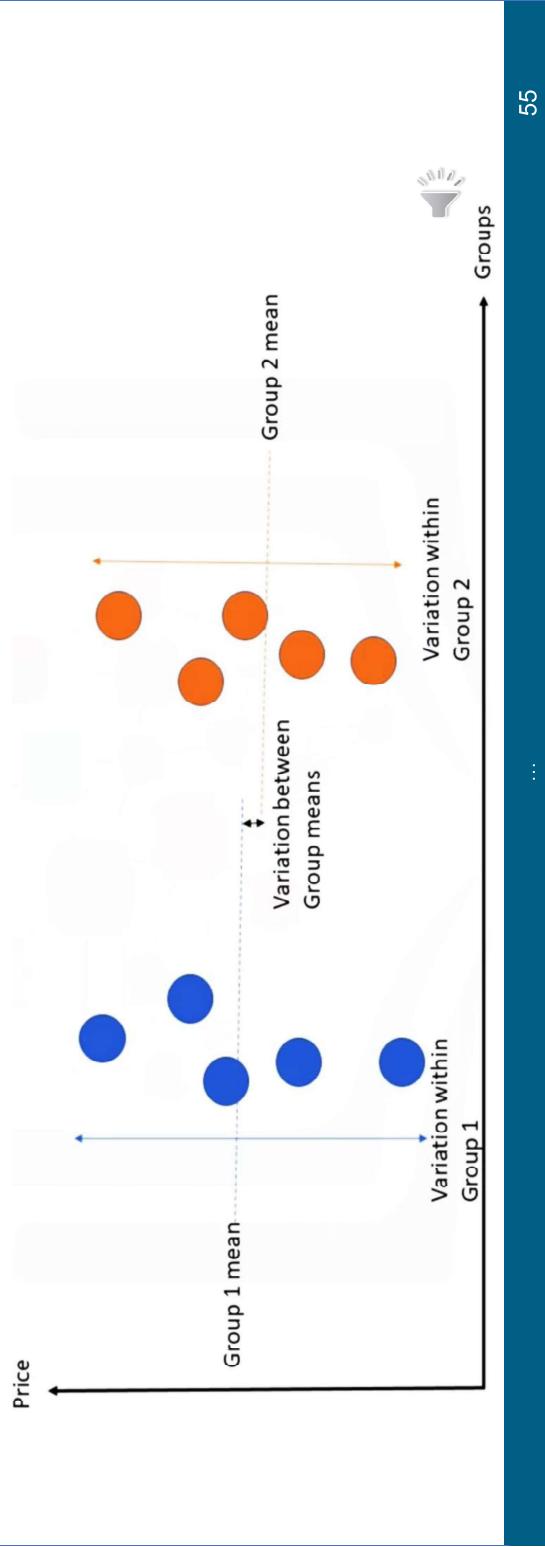


54

## 4. Phân tích ANOVA

### ANOVA – F-test

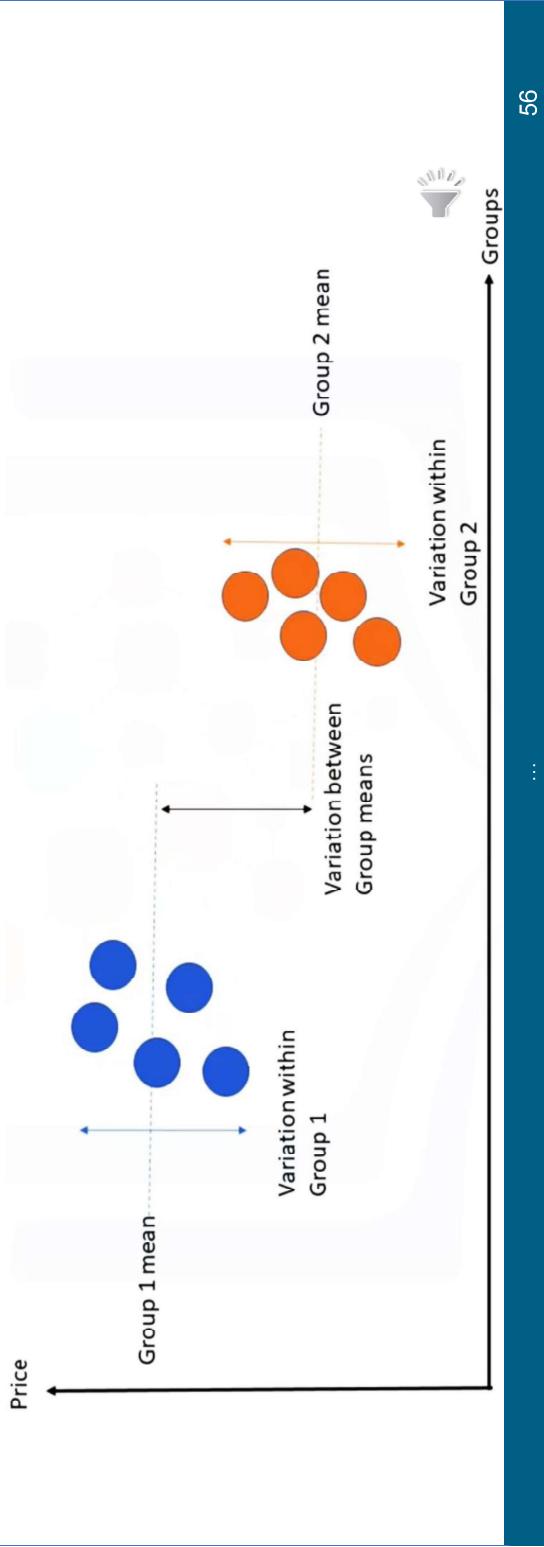
- F-test (nhỏ) thì khả năng ảnh hưởng (yếu) giữa các biến phân loại và biến mục tiêu



## 4. Phân tích ANOVA

### ANOVA – F-test

- F-test (lớn) thì khả năng ảnh hưởng (mạnh) giữa các biến phân loại và biến mục tiêu



## 4. Phân tích ANOVA

### Cài đặt

```
import scipy.stats as stats

fvalue, pvalue = stats.f_oneway(groupA,
                                 groupB,
                                 groupC, ...)
```

...

57

## 4. Phân tích ANOVA



### Áp dụng

- **Bước #1:**  
grouped\_anova = df\_anova.groupby(['make'])
- **Bước #2:**

#### ➤ Giữa **honda** và **subaru**

```
anova_result = stats.f_oneway(grouped_anova.get_group('honda')[['price']],
                                 grouped_anova.get_group('subaru')[['price']])
```

```
F_onewayResult(statistic=0.19744030127462606, pvalue=0.6609478240622193)
```

#### ➤ Giữa **honda** và **jaguar**

```
anova_result = stats.f_oneway(grouped_anova.get_group('honda')[['price']],
                                 grouped_anova.get_group('jaguar')[['price']])
```

```
F_onewayResult(statistic=400.925870564337, pvalue=1.0586193512077862e-11)
```

...

58



## Kết luận

### Công việc cần làm trong phân tích thăm dò dữ liệu

- Kích cỡ bộ dữ liệu (Size - Shape).
  - Ý nghĩa từng biến/cột.
  - Phân tích thăm dò => Thông kê mô tả => Thăm dò từng biến.
  - Đánh giá giá trị ngoại lệ.
  - Xét độ tương quan (thống kê).
- => Trích xuất biến quan trọng ảnh hưởng đến mục tiêu.



59



Thank you

## Hỏi - Đáp

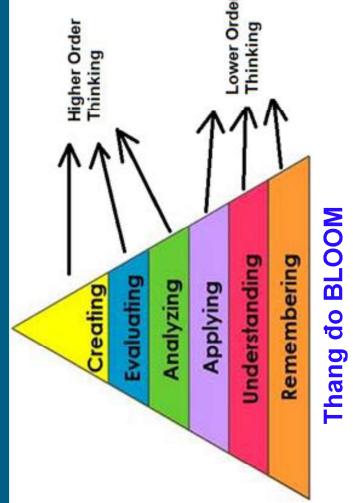


62

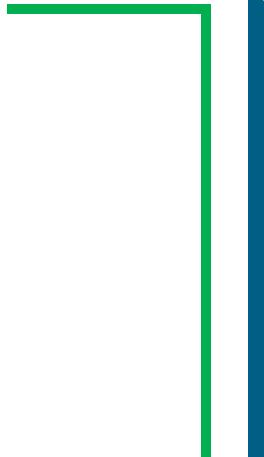
## Chương 4 PHÁT TRIỂN MÔ HÌNH



## Mục tiêu



- Vận dụng hồi quy tuyến tính đơn và đa biến
- Biết đánh giá mô hình dùng trực quan
- Vận dụng hồi quy đa thức và kỹ thuật Pipeline
- Áp dụng thang đo R-squared và MSE dùng để đánh giá tập mẫu



2

## Nội dung



1. Phát triển mô hình
2. Hồi quy tuyến tính đơn biến
3. Hồi quy tuyến tính đa biến
4. Đánh giá mô hình dùng trực quan
5. Hồi quy đa thức
6. Kỹ thuật Pipeline / pip, lin/
7. Thang đo MSE và R-squared ( $R^2 - R^{\wedge}2$ )
8. Đánh giá



3



## Tóm tắt

- **Đặt vấn đề (Chương 1):**  
*Chúng ta nên bán xe đã qua sử dụng với giá bao nhiêu?*

- **Hướng giải quyết (Chương 2):**  
*Tiền xử lý dữ liệu liên quan để phục vụ phân tích.*

- **Phân tích thăm dò (Chương 3):**  
*Những đặc điểm nào có ảnh hưởng nhất đến giá xe?*

- Phát triển mô hình để dự đoán (Chương 4):

*Chúng ta xác định giá hợp lý cho xe đã qua sử dụng như thế nào?*

4



## Tóm tắt

- **Phân tích thăm dò - EDA (Chương 3):**  
*Những đặc điểm nào có ảnh hưởng nhất đến giá xe?*
- **Kết quả (tham khảo):** Các biến quan trọng đã thăm dò được tìm thấy là:

Numerical feature	Categorical feature
<ul style="list-style-type: none"><li>• curb-weight</li><li>• engine-size</li><li>• length</li><li>• width</li><li>• horsepower</li><li>• city-mpg</li><li>• highway-mpg</li><li>• wheel-base</li><li>• bore</li></ul>	<ul style="list-style-type: none"><li>• drive-wheels</li></ul>

5



## Bộ dữ liệu

### **Bộ dữ liệu dùng để phát triển mô hình**

Lý thuyết	Thực hành
<a href="https://raw.githubusercontent.com/datasets/hub/master/Model_Dataset_Lab.csv">https://raw.githubusercontent.com/datasets/hub/master/Model_Dataset_Lab.csv</a>	<a href="https://raw.githubusercontentcontent.com/datasets/hub/ds105/master/Model_Database_Lab.csv">https://raw.githubusercontentcontent.com/datasets/hub/ds105/master/Model_Database_Lab.csv</a>



## 1. Phát triển mô hình

### **Giới thiệu**

- Mô hình có thể được xem là một phương trình toán học được dùng để dự đoán (predict) một giá trị hoặc nhiều giá trị.
- Biểu diễn mối liên quan hệ một hoặc nhiều **biến độc lập** (independent) với các **biến phụ thuộc** (dependent).

Biến độc lập  
'city-mpg'  
39



Mô hình



Biến phụ thuộc  
'price'  
7000



## 1. Phát triển mô hình

### Giới thiệu

- Thông thường, dữ liệu càng phù hợp, liên quan thì mô hình càng chính xác.



- Khi nào dữ liệu là phù hợp?  
=> Thu thập => Đạt chuẩn. Vậy, dùng phương pháp gì để kiểm tra?



## 1. Phát triển mô hình

### Giới thiệu

- Để hiểu tại sao nhiều dữ liệu (thuộc tính) là quan trọng hơn, hãy xem xét tình huống sau:

- Có hai chiếc xe gần như giống hệt nhau.
- Thực tế, đối khi xe màu xanh bán với giá thấp hơn.



## 1. Phát triển mô hình

### Giới thiệu

- Ngoài việc có được nhiều dữ liệu (tốt, phù hợp, có liên quan, đạt chuẩn), chúng ta có thể thử các loại mô hình khác để có kết quả tốt hơn.

• Trong phần này, chúng ta sẽ tập trung xây dựng các loại mô hình sau:

1. Hồi quy tuyến tính đơn biến.
2. Hồi quy tuyến tính đa biến.
3. Hồi quy đa thức
4. Kỹ thuật Pipeline

Tại sao chọn hồi quy?

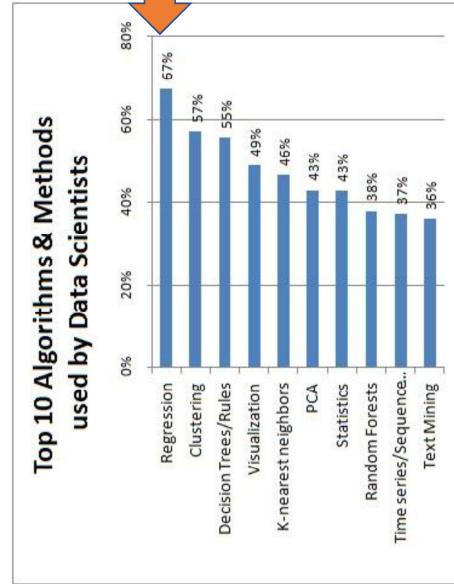
...

10

...

## 1. Phát triển mô hình

Tại sao chọn hồi quy?



Năm 2016

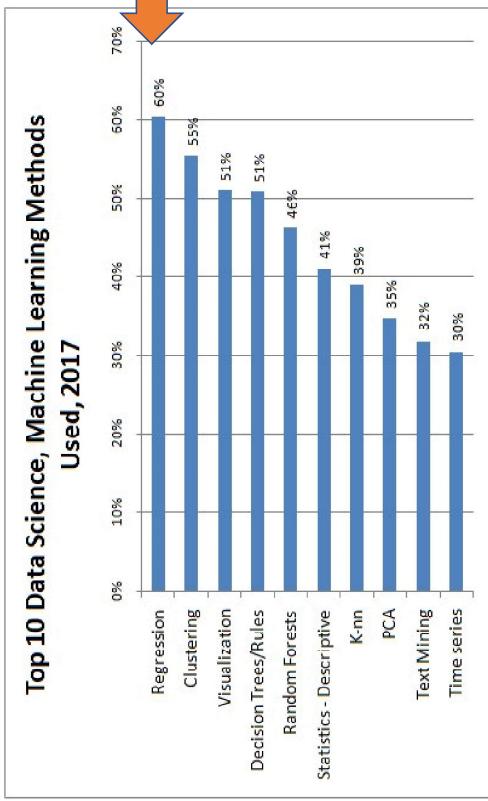
...

11



## 1. Phát triển mô hình

Tại sao chọn hồi quy?



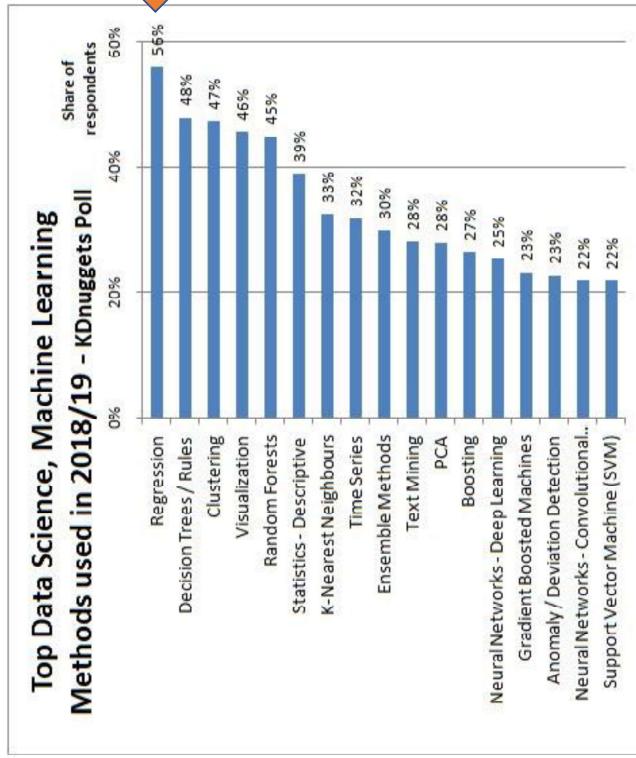
Năm 2017

... 12 ...



## 1. Phát triển mô hình

Tại sao chọn hồi quy?



Năm 2018-2019

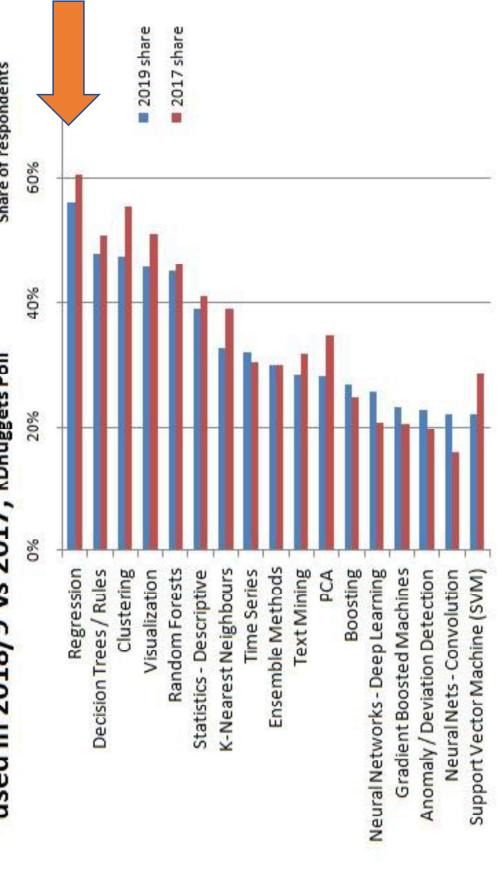
... 13 ...



## 1. Phát triển mô hình

Tại sao chọn hồi quy?

### Top Data Science, Machine Learning Methods, Algorithms used in 2018/9 vs 2017, KDnuggets Poll



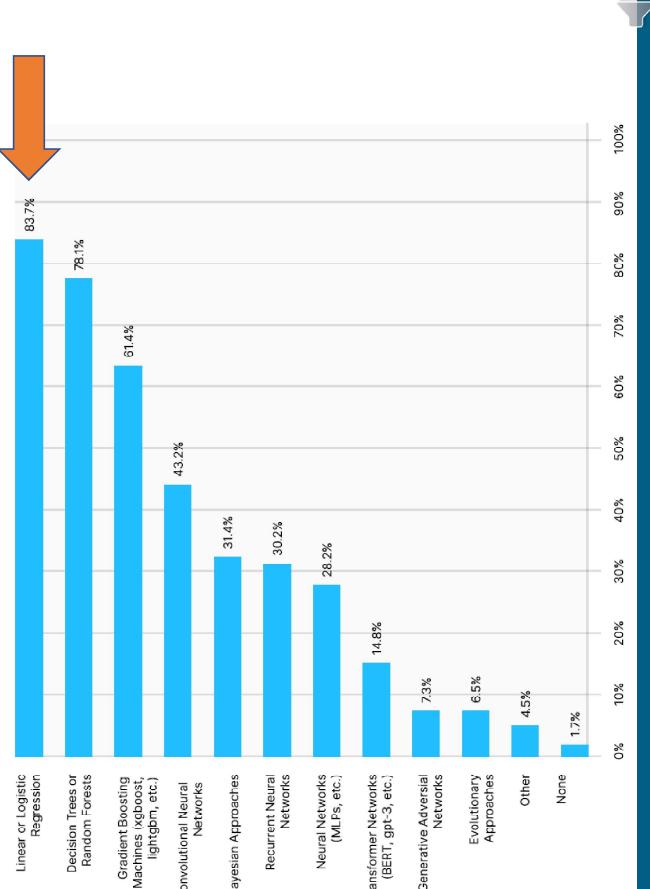
14

So sánh năm 2018-19 và 2017

## 1. Phát triển mô hình

Tại sao chọn hồi quy?

### METHODS AND ALGORITHMS USAGE



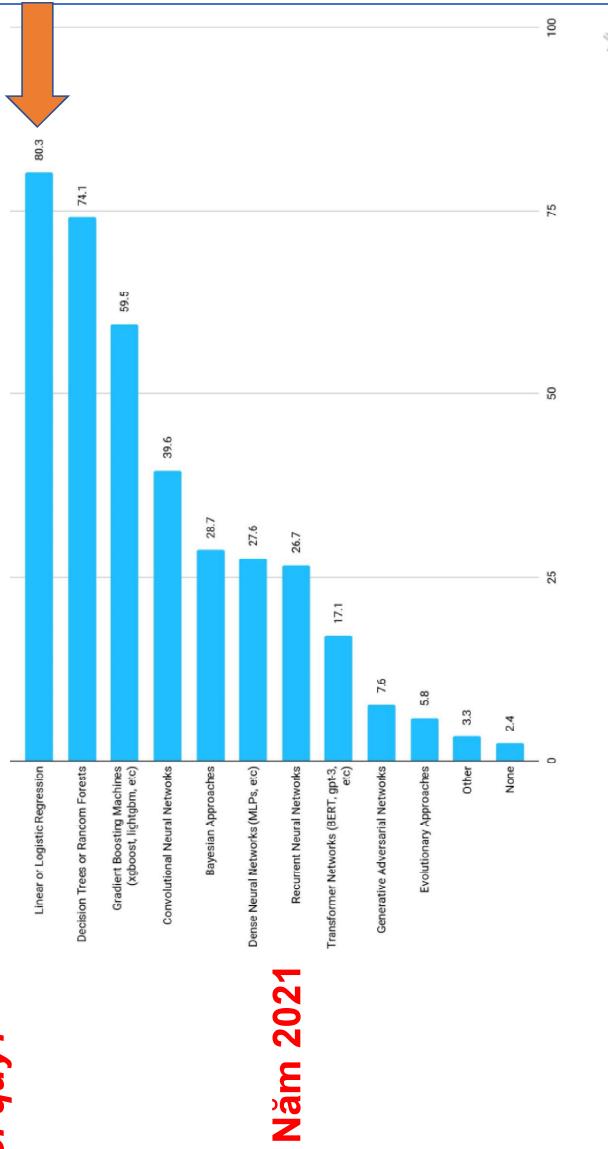
15

Năm 2020



## 1. Phát triển mô hình

Tại sao chọn hồi quy?



Năm 2021



## 1. Phát triển mô hình

Chú ý

Biến độc lập ~~~~~ Biến phụ thuộc ~~~~~ => Target

	feature 1	feature 2	feature 3	feature 4	price	type
0	5.1	3.5	1.4	0.2	4394	2
1	4.9	3.0	1.4	0.2	4336	1
2	4.7	3.2	1.3	0.2	6561	3
3	4.6	3.1	1.5	0.2	4137	2
4	5.0	3.6	1.4	0.2	1762	1
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	8530	3
146	6.3	2.5	5.0	1.9	464	3
147	6.5	3.0	5.2	2.0	2680	4
148	6.2	3.4	5.4	2.3	3591	2
149	5.9	3.0	5.1	1.8	6003	3

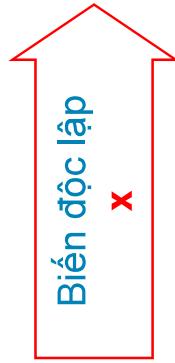




## 2. Hồi quy tuyến tính đơn biến

### Giới thiệu

- Hồi quy tuyến tính đơn biến sẽ dùng **một biến độc lập** để đưa ra dự đoán.



Dự đoán



18



## 2. Hồi quy tuyến tính đơn biến

### Mô hình - Model

- **x**: Biến độc lập (hay biến predictor)
- **y**: Biến phụ thuộc (hay biến target )

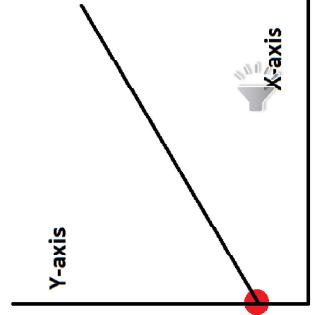
$$y = b_0 + b_1 x$$

•  **$b_0$** : là intercept

•  **$b_1$** : là slope

Thể hiện độ dốc

Thể hiện điểm chấn đầu của mô hình



19



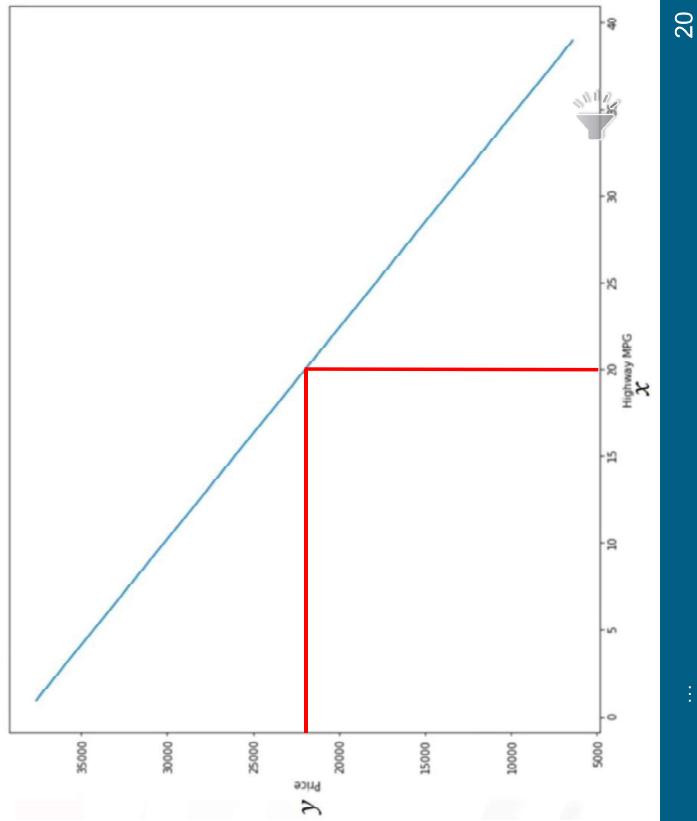
## 2. Hồi quy tuyến tính đơn biến

### Dự đoán - Prediction

- Dùng mô hình hiện tại để dự đoán kết quả.

Ví dụ ta có mô hình:

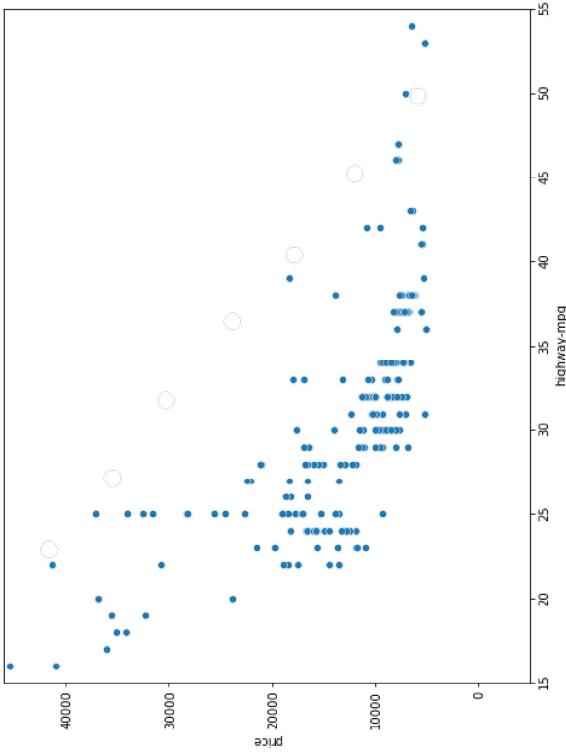
- $y = 38423 - 821x$
- $y = 38423 - 821(20)$
- $y = 22003$



## 2. Hồi quy tuyến tính đơn biến

### Xây dựng mô hình (gọi tắt là fit)

Fit

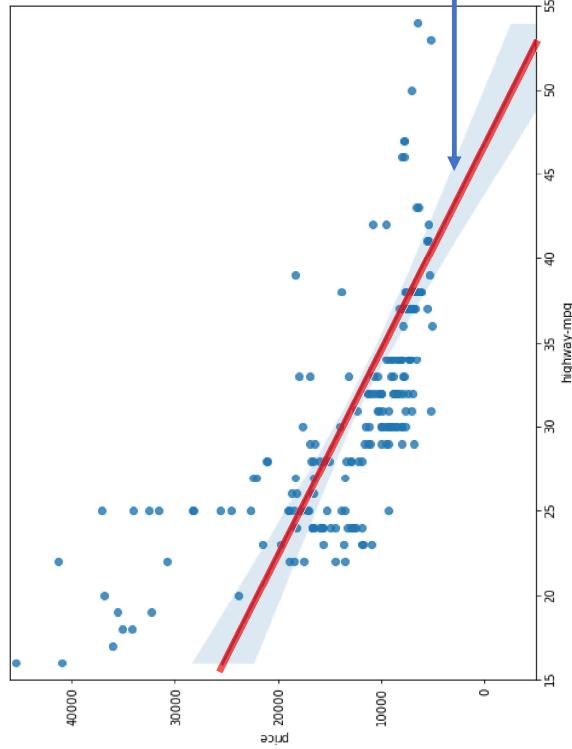


$$y = b_0 + b_1 x$$



## 2. Hồi quy tuyến tính đơn biến

Xây dựng mô hình (gọi tắt là fit)



22



## 2. Hồi quy tuyến tính đơn biến

Xây dựng mô hình hồi quy tuyến tính đơn biến

Chuẩn bị:

- **X:** Biến độc lập - Biến dùng để dự đoán (predictor variable)
- **Y:** Biến phụ thuộc - Biến mục tiêu (target variable)

Cài đặt:

1. import linear\_model từ gói scikit-learn

```
from sklearn.linear_model import LinearRegression
```

2. Khởi tạo đối tượng LinearRegression dùng constructor:

```
lm=LinearRegression()
```



24



## 2. Hồi quy tuyến tính đơn biến

### Xây dựng mô hình hồi quy tuyến tính đơn biến

3. Chọn biến độc lập X và biến phụ thuộc Y

```
X = df[['highway-mpg']]  
Y = df['price']
```

4. Gọi phương thức lm.fit(X, Y) để xây dựng mô hình, tìm  $b_0$  và  $b_1$

```
lm.fit(X, Y)
```

X	Yhat
10	13400
...	...
15	10300
...	...

5. Dự đoán

```
Yhat=lm.predict(X)
```



25



## 2. Hồi quy tuyến tính đơn biến

### Xây dựng mô hình hồi quy tuyến tính đơn biến

- Xuất kết quả intercept ( $b_0$ ):

```
lm.intercept_
```

Kết quả: 38423.305858

- Xuất kết quả slope ( $b_1$ ):

```
lm.coef_
```

Kết quả: -821.73337832

- Mô hình thể hiện mối quan hệ giữa Price và Highway\_mpg:

Price = 38423.31 - 821.73 \* highway-mpg

$Y = b_0 + b_1 X$

...



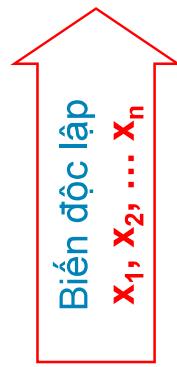
26



### 3. Hồi quy tuyến tính đa biến

#### Giới thiệu

- Hồi quy tuyến tính đa biến sẽ dùng **nhiều biến độc lập** để đưa ra dự đoán.



Hồi quy tuyến tính  
đa biến

Dự đoán



27



### 3. Hồi quy tuyến tính đa biến

#### Model

Phương pháp này được sử dụng để biểu diễn mối quan hệ giữa:

- Hai hoặc nhiều biến độc lập **X** (hay biến predictor)
- Một biến phụ thuộc **Y** (hay biến target)

$$Y = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + b_4X_4 + \dots$$

- **b<sub>0</sub>** intercept ( $X = 0$ )
- **b<sub>1</sub>** coefficient hoặc parameter của **X<sub>1</sub>**
- **b<sub>2</sub>** coefficient hoặc parameter of **X<sub>2</sub>** và tương tự...



28



### 3. Hồi quy tuyến tính đa biến

#### Xây dựng mô hình hồi quy tuyến tính đa biến

##### 1. Chọn 4 biến độc lập

```
Z = df[ [ 'horsepower', 'curb-weight', 'engine-size', 'highway-mpg' ] ]
```

##### 2. Xây dựng mô hình bằng phương thức fit():

```
lm.fit(z, df[ 'price' ])
```

##### 3. Dự đoán

```
Yhat=lm.predict(X)
```

	$x_1$	$x_2$	$x_3$	$x_4$	
3	5	-4	3		
:	:	:	:	:	
2	4	2	-4		
					...

33



### 3. Hồi quy tuyến tính đa biến

#### Xây dựng mô hình hồi quy tuyến tính đa biến

##### 1. Lấy giá trị intercept ( $b_0$ )

```
lm.intercept  
-15678.742628061467
```

##### 2. Lấy coefficients ( $b_1, b_2, b_3, b_4$ )

```
lm.coef  
array([52.65851272, 4.69878948, 81.95906216, 33.58258185])
```

##### 3. Mô hình:

```
Price = -15678.74 + (52.66) * horsepower + (4.70) * curb-weight + (81.96) * engine-size + (33.58) * highway-mpg
```

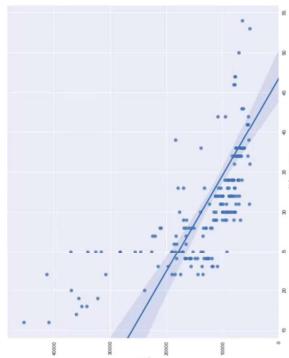


34

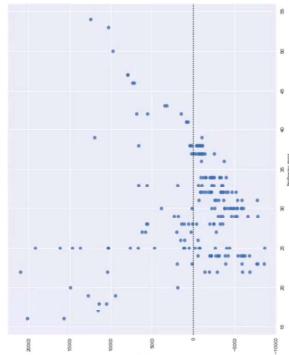


## 4. Đánh giá mô hình dùng trực quan

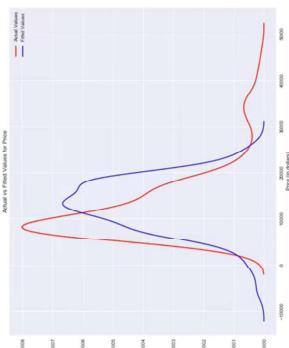
### 4.1. Regression plot



### 4.2. Residual plot



### 4.3. Distribution plot



35



## 4.1 Regression plot

- Tại sao phải dùng đến **regression plot**?

• Biểu đồ hồi quy là một tiêu chí đánh giá tốt về mô hình. Vì thế hiện:

- Mối quan hệ giữa các biến.
- Độ mạnh của mối tương quan (correlation).
- Chiều hướng của các mối quan hệ đồng thuận và không đồng thuận (positive and negative)



36



## 4.1 Regression plot

Regression plot cho chúng ta thấy sự kết hợp của:

- Sự tản xạ của các điểm dữ liệu sẽ đại diện cho một ý nghĩa (target) khác nhau.
- Đường hồi quy tuy nhiên tính được xây dựng (fit) dựa trên các điểm dữ liệu.



37



### 4.1. Regression plot

Cài đặt Regression plot

```
import seaborn as sns
```

```
sns.regplot(x="highway-mpg", y="price", data=df)
```



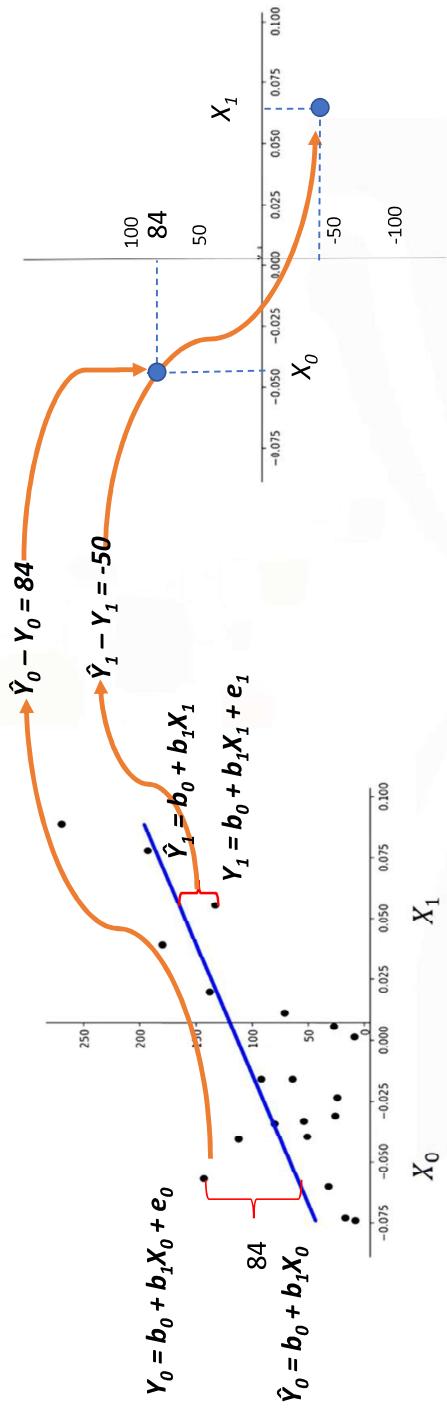
38



## 4.2. Residual Plot

### Residual Plot (Phản dure, hoặc errors)

Y-axis: residual



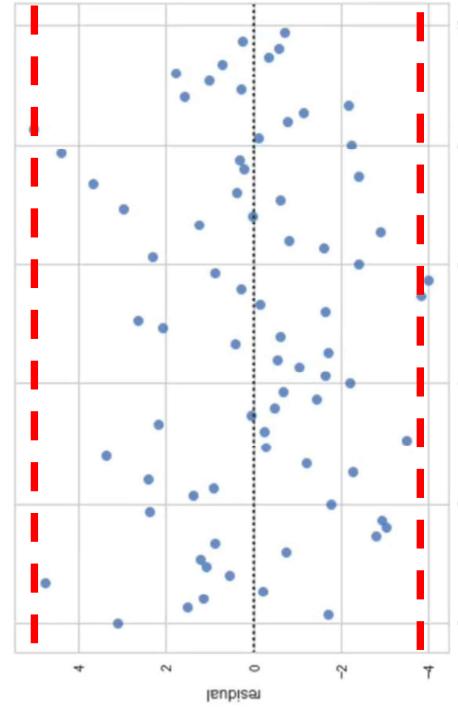
X-axis: the predictor variable or fitted values

39



## 4.2. Residual Plot

### Trường hợp 1



• Quan sát các điểm trải đều trong residual:

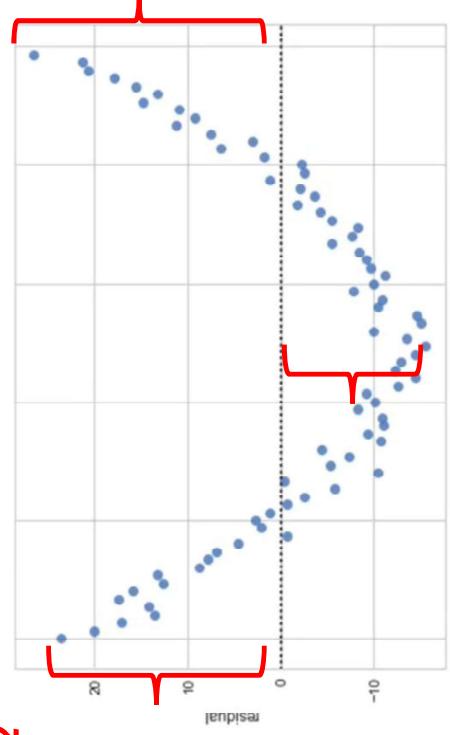
- (Ngẫu nhiên) trải ra xung quanh trục thì mô hình tuyến tính là phù hợp



40

## 4.2. Residual Plot

### Trường hợp 2



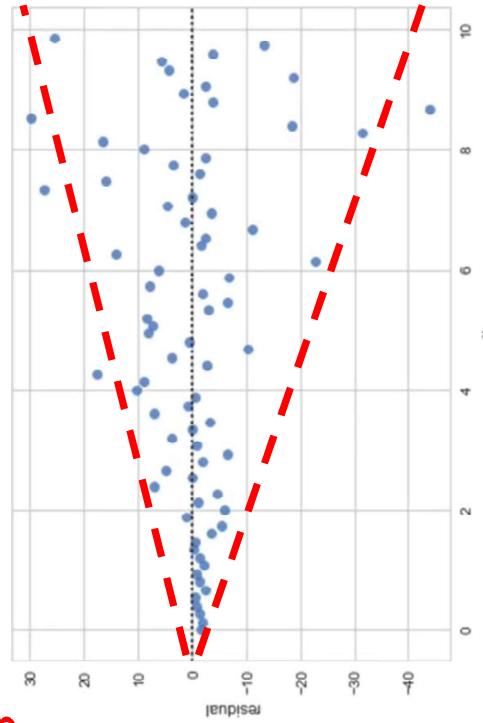
- Không ngẫu nhiên trải ra xung quanh trục x
- Tìm mô hình khác phù hợp hơn

Phi tuyến tính

41

## 4.2. Residual plot

### Trường hợp 3



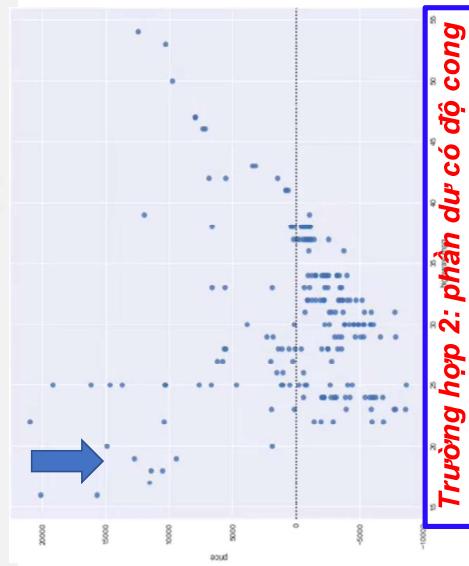
- Không tập trung quanh trục, phần dư tăng theo x
- Mô hình không chính xác

42

## 4.2. Residual plot

Cài đặt Residual plot

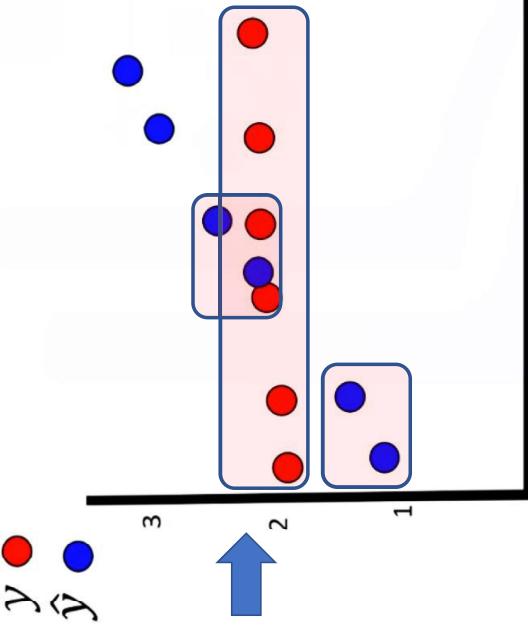
```
import seaborn as sns  
  
sns.residplot(df['highway-mpg'], df['price'])
```



43

## 4.3. Distribution plot

Distribution Plot



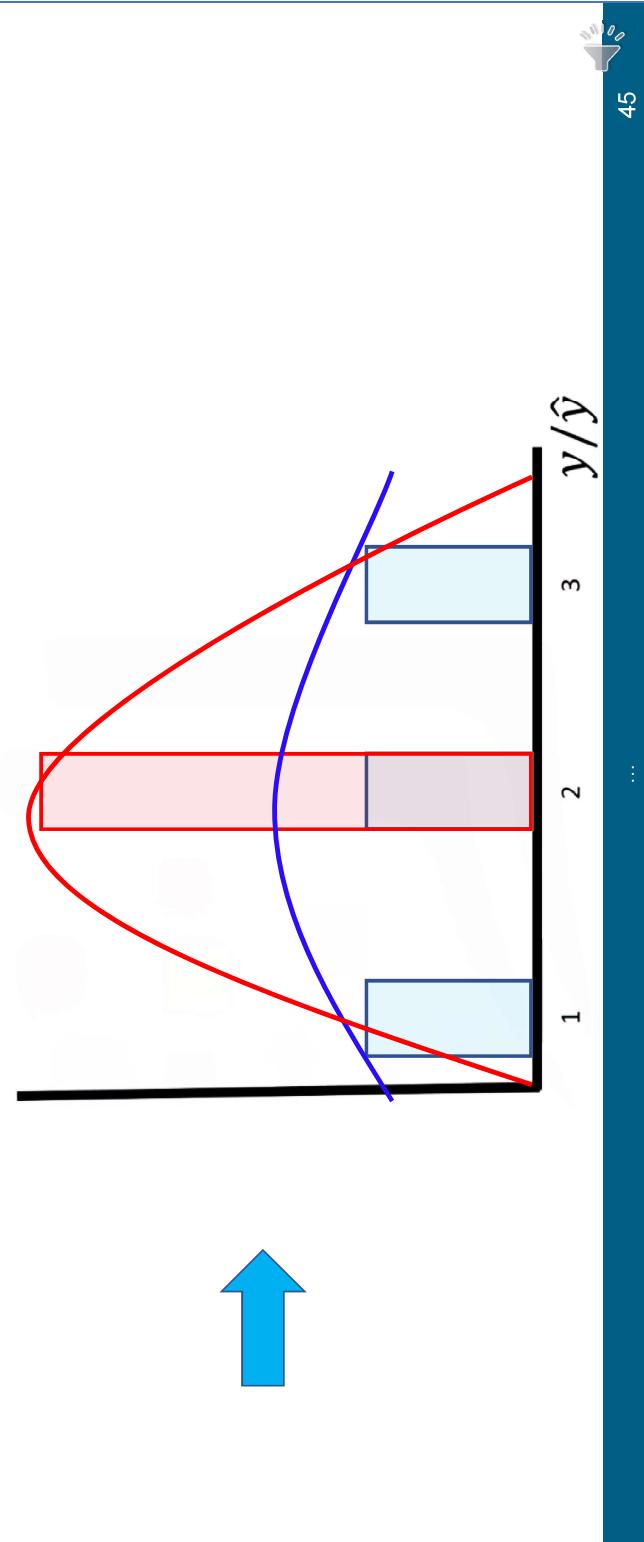
$x$

44



### 4.3. Distribution plot

#### Distribution Plots

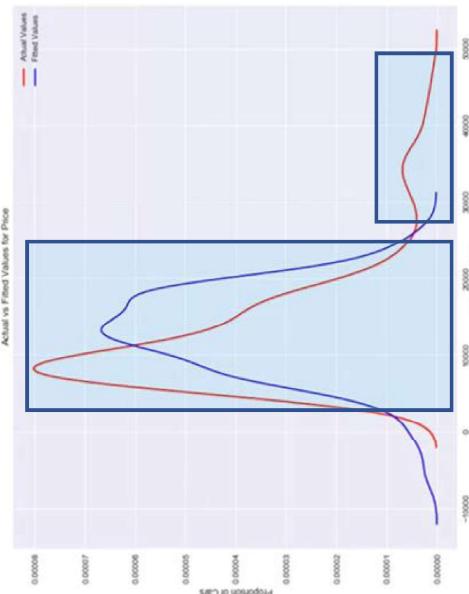


### 4.3. Distribution plot



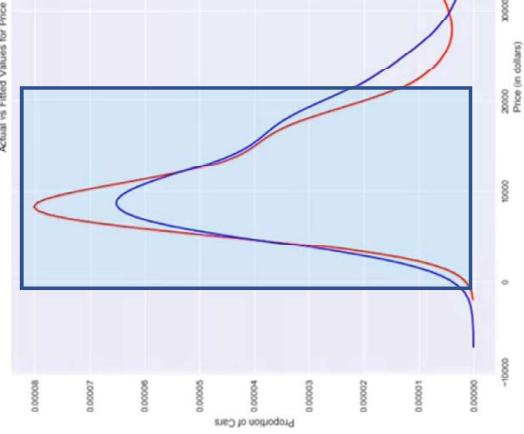
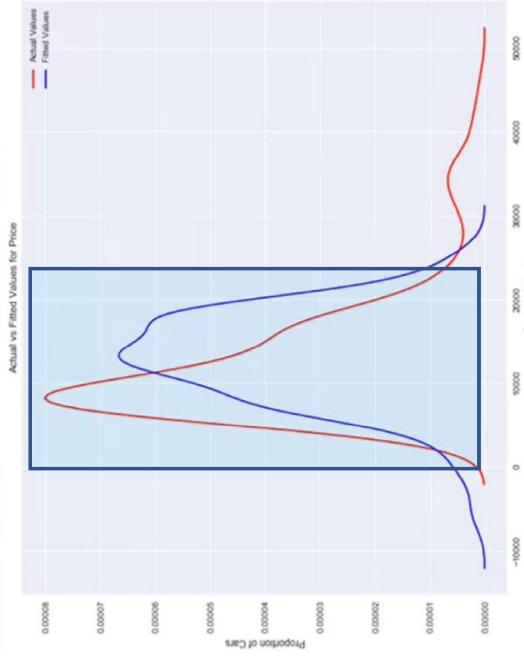
**Ví dụ:** So sánh đường phân phối chuẩn giữa

- Giá trị đã dự đoán từ kết quả mô hình (Đường màu xanh)
- Giá trị thực tế (Đường màu đỏ)



## 4.3. Distribution plot

Ví dụ



47

A

B



## 4.3. Distribution plot

Cài đặt Distribution plot

```
import seaborn as sns
```

```
ax1 = sns.distplot(df['price'], hist=False, color="r", label="Actual Value")
```

```
sns.distplot(Yhat, hist=False, color="b", label="Fitted Values", ax=ax1)
```

...

...

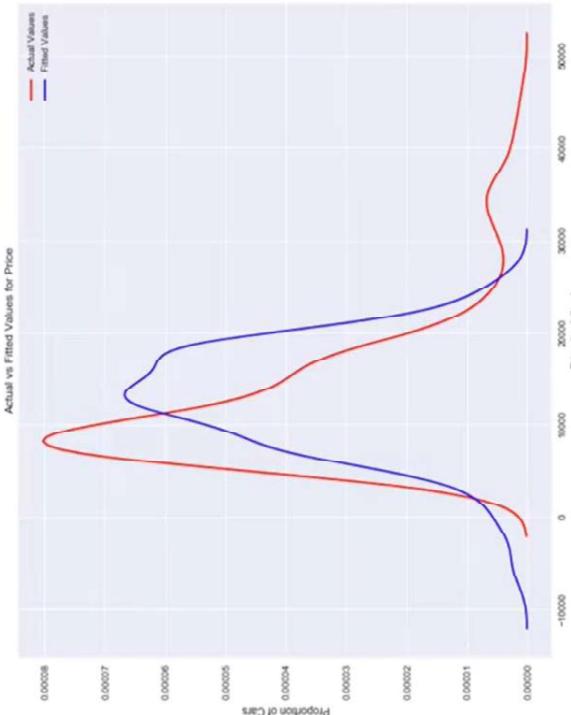


...

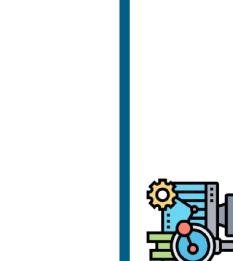
48

## 4.3. Distribution plot

### Kết quả



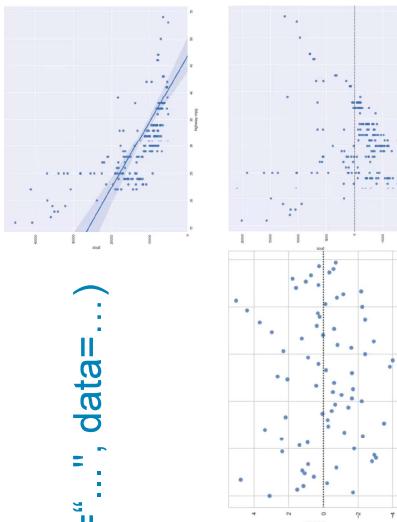
49



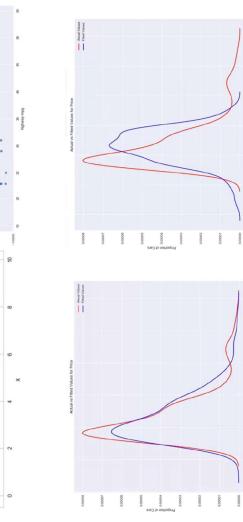
## 4. Đánh giá mô hình dùng trực quan

### Tóm lại

- Regression Plot => `regplot(x="...", y="...", data=...)`



- Residual Plot => `residplot(df.a, df.b)`



- Distribution Plot => `distplot()`

50





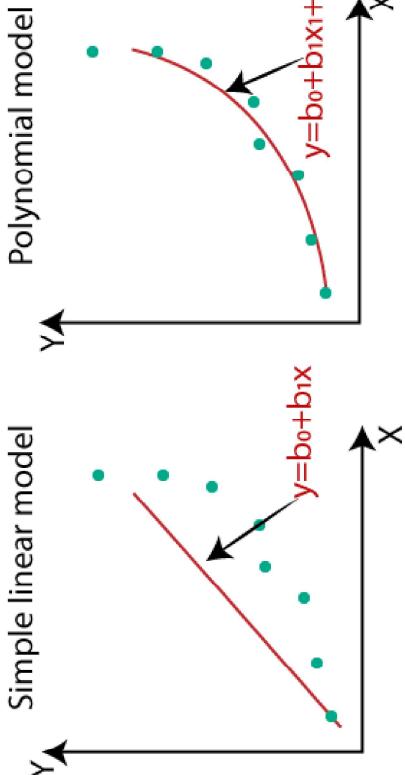
## 5. Hồi quy đa thức

### Hồi quy đa thức - Polynomial regression

- Là một trường hợp đặc biệt của mô hình hồi quy tuyến tính tổng quát.
- Biểu diễn các mối quan hệ **cong phi tuyến** (**curvilinear**) rất tốt.

#### Quan hệ cong phi tuyến

- Bình phương
- Hoặc thiết lập các bậc cao hơn  
cho biến độc lập



51



## 5. Hồi quy đa thức

### Cong phi tuyến

- Bậc 2 (Quadratic – 2<sup>nd</sup> order)

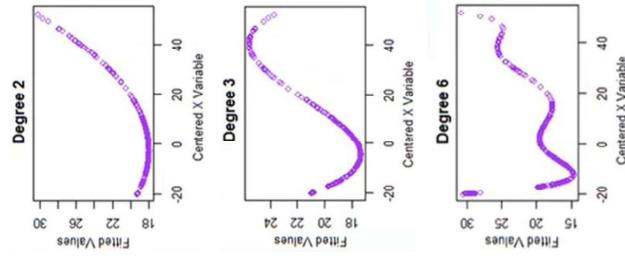
$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2$$

- Bậc 3 (Cubic – 3<sup>rd</sup> order)

$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2 + b_3 (x_1)^3$$

- Bậc cao hơn

$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2 + b_3 (x_1)^3 + \dots$$



52



## 5. Hồi quy đa thức

### Xây dựng mô hình

#### 1. Xây dựng mô hình bậc 3

```
f=np.polyfit(x,y,3)
```

```
p=np.polyd1(f)
```

#### 2. Xuất mô hình

```
print(p)
```

$$-1.557(x_1)^3 + 204.8(x_1)^2 + 8965x_1 + 1.37 \times 10^5$$

53

...

...

```
x = df['highway-mpg']  
y = df['price']
```

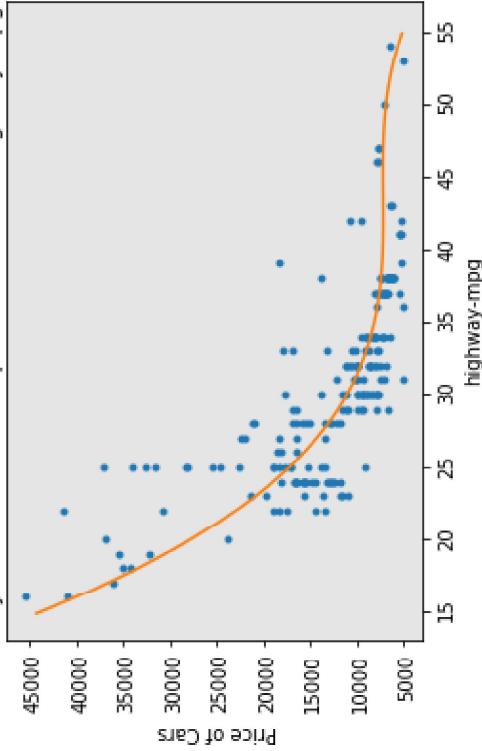
### Mô hình

- Kết quả:  $-1.557(x_1)^3 + 204.8(x_1)^2 + 8965x_1 + 1.37 \times 10^5$



## 5. Hồi quy đa thức

Polynomial Fit with Matplotlib for Price ~ Highway-mpg



54



## 5. Hồi quy đa thức

### Đa biến và bậc cao hơn

- Mô hình

$$y\text{Hat} = b_0 + b_1 \textcolor{red}{X}_1 + b_2 \textcolor{red}{X}_2 + b_3 \textcolor{red}{X}_1 \textcolor{red}{X}_2 + b_4 (\textcolor{red}{X}_1)^2 + b_5 (\textcolor{red}{X}_2)^2 + \dots$$
$$y\text{Hat} = b_0 + b_1 \textcolor{red}{a} + b_2 \textcolor{red}{b} + b_3 \textcolor{red}{c} + b_4 \textcolor{red}{d} + b_5 \textcolor{red}{e} + \dots$$

- Các bước:

- Bước #1: Biến đổi các biến (Transform=>feature)

- Bước #2: Dùng LinearRegression để Fit - xây dựng



55



## 5. Hồi quy đa thức

### Bước #1: Biến đổi biến

- Sử dụng mô-đun **PolynomialFeatures** trong package **preprocessing** của **sklearn**



```
from sklearn.preprocessing import PolynomialFeatures
```

```
pr=PolynomialFeatures(degree=2)
```

```
pr.fit_transform([[1, 2]])
```

```
array([1., 1., 2., 1., 2., 4.])
```

Chuyển [a, b] thành bậc 2  
Kết quả: [1, a, b, a^2, ab, b^2]

56



## 5. Hồi quy đa thức

### Bước #1: Biến đổi biến

- Sử dụng mô-đun **PolynomialFeatures** trong package **preprocessing** của **sklearn**

```
from sklearn.preprocessing import PolynomialFeatures
```

```
pr=PolynomialFeatures(degree=2, include_bias=False)
```

```
pr.fit_transform([[1, 2]])
```

```
array([[[1., 2., 1., 2., 4.]]])
```

Chuyển [a, b] thành bậc 2  
Kết quả: [a, b, a^2, ab, b^2]

...

57



## 5. Hồi quy đa thức

### Bước #1: Biến đổi biến

- Áp dụng vào bộ dữ liệu hiện tại

```
from sklearn.preprocessing import PolynomialFeatures
```

```
pr = PolynomialFeatures(degree=2, include_bias=False)
```

```
X_poly = pr.fit_transform(df[['horsepower', 'cubic-weight']])
```

```
X_poly.shape
```

(201, 5)



58



## 5. Hồi quy đa thức

### Bước #2: Dùng LinearRegression để Fit

- Áp dụng vào bộ dữ liệu

```
from sklearn.linear_model import LinearRegression
```

```
lm = LinearRegression()
```

```
lm.fit(x_poly, df['price'])
```

```
LinearRegression()
```

```
lm.intercept_
```

```
17062.609692007238
```

```
lm.coef_
```

```
array([-1.07696940e+02, -1.80653670e+01, 5.48648597e-01, -5.38543914e-02,
```

```
5.82043770e-03])
```

```
...
```

```
59
```



## 5. Hồi quy đa thức

### Khuyến nghị

- Mô hình:

$$y\hat{H}at = b_0 + b_1 \mathbf{X}_1 + b_2 \mathbf{X}_2 + b_3 \mathbf{X}_1 \mathbf{X}_2 + b_4 (\mathbf{X}_1)^2 + b_5 (\mathbf{X}_2)^2 + \dots$$

- **Khuyến nghị** xây dựng theo bước sau:

- **Bước #1: Chuẩn hóa** (Normalize)
- **Bước #2: Biến đổi các biến** (Transform=>feature)
- **Bước #3: Dùng LinearRegression để Fit - xây dựng**



## 5. Hồi quy đa thức

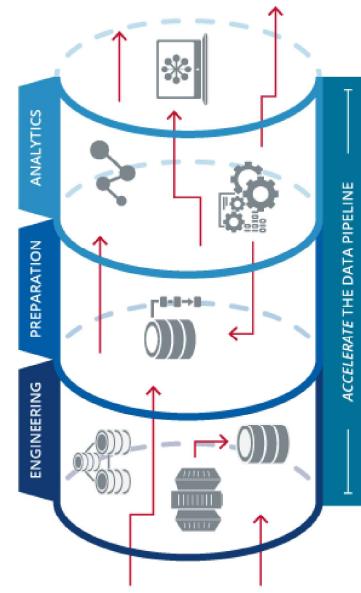
### Chuẩn hóa

- Nên chuẩn hóa các biến khi áp dụng nhiều chiều.
- Ví dụ, thực hiện **chuẩn hóa cho biến horsepower và highway-mpg**

```
from sklearn.preprocessing import StandardScaler  
SCALE=StandardScaler()  
SCALE.fit(x_data[['horsepower', 'highway-mpg']])  
...  
62
```

## Pipeline

### 6. Kỹ thuật Pipeline





## 6. Kỹ thuật Pipeline

### Pipeline

- Là một thư viện trong **scikit-learn**.
- Tập hợp các phép biến đổi của các bước để tìm ra mô hình.
- Mục tiêu của kỹ thuật pipeline là thể hiện các quá trình phân tích thành một **qui trình tự động**.

```
sklearn.pipeline.Pipeline
```

[source]

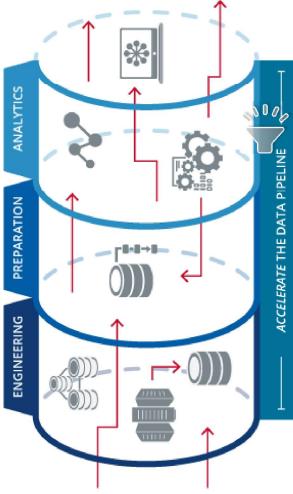
```
class sklearn.pipeline.Pipeline(steps, memory=None, verbose=False)
```

Pipeline of transforms with a final estimator.

Sequentially apply a list of transforms and a final estimator. Intermediate steps of the pipeline must be 'transforms', that is, they must implement `fit` and `transform` methods. The final estimator only needs to implement `fit`. The transforms in the pipeline can be cached using `memory` argument.

The purpose of the pipeline is to assemble several steps that can be cross-validated together while setting different parameters. For this, it enables setting parameters of the various steps using their names and the parameter name separated by a `'__'`, as in the example below. A step's estimator may be replaced entirely by setting the parameter with its name to another estimator, or a transformer removed by setting it to `'passthrough'` or `None`.

...



64

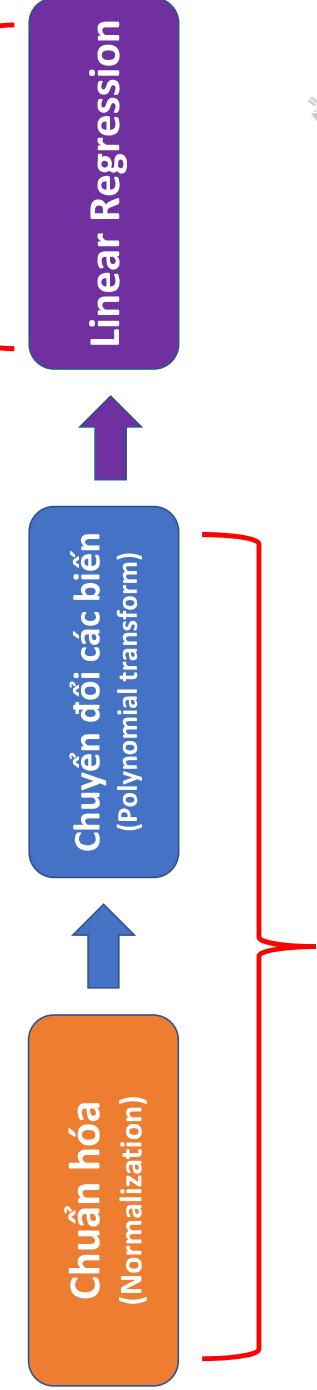


## 6. Kỹ thuật Pipeline

### Pipeline

- Để có được một mô hình dùng để **dự đoán** thì phải trải qua nhiều bước:

Mô hình => **dự đoán**



65



## 6. Kỹ thuật Pipeline

### Cài đặt kỹ thuật Pipeline

- Nhập thư viện hỗ trợ có liên quan

```
from sklearn.preprocessing import PolynomialFeatures  
from sklearn.linear_model import LinearRegression  
from sklearn.preprocessing import StandardScaler  
from sklearn.pipeline import Pipeline
```



66

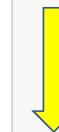
...



## 6. Kỹ thuật Pipeline

### Pipeline Constructor

```
Input=[('scale',StandardScaler()),  
      ('polynomial',PolynomialFeatures(include_bias=False)),  
      ('model',LinearRegression())]
```



- Dùng Pipeline constructor để khởi tạo đối tượng

```
pipe=Pipeline(Input)
```



pipeline object



67



## 6. Kỹ thuật Pipeline

### Pipeline Constructor

- Dùng phương thức `fit()` để xây dựng mô hình từ **đổi tương pipeline**

```
pipe.fit(z,y)
```

```
Pipeline(steps=[('scale', StandardScaler()),
               ('polynomial', PolynomialFeatures(include_bias=False)),
               ('model', LinearRegression())])
```

```
ypipe=pipe.predict(z)
ypipe[0:4]
```

```
array([13102.74784201, 13102.74784201, 18225.54572197, 10390.29636551])
```



68

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
```

```
Input=[('scale',StandardScaler()),
      ('polynomial', PolynomialFeatures(include_bias=False)),
      ('model',LinearRegression())]
```

```
pipe=Pipeline(Input)
pipe
```

```
Pipeline(steps=[('scale', StandardScaler()),
               ('polynomial', PolynomialFeatures(include_bias=False)),
               ('model', LinearRegression())])
```

```
pipe.fit(z,y)
```

```
Pipeline(steps=[('scale', StandardScaler()),
               ('polynomial', PolynomialFeatures(include_bias=False)),
               ('model', LinearRegression())])
```

```
ypipe=pipe.predict(z)
ypipe[0:4]
```

```
array([13102.74784201, 13102.74784201, 18225.54572197, 10390.29636551])
```





## 7. Thang đo đánh giá

### Giới thiệu các thang đo

- Thang đo là 1 giá trị dùng để định lượng mức độ phù hợp của mô hình với bộ dữ liệu.

- Hai thang đo quan trọng để xác định mức độ phù hợp của mô hình:

7.1. Mean Squared Error (**MSE**)

7.2. R-squared (**R<sup>2</sup>**)

Có thể gọi là:  
Sai số bình phương trung bình  
Sai số toàn平方 trung bình

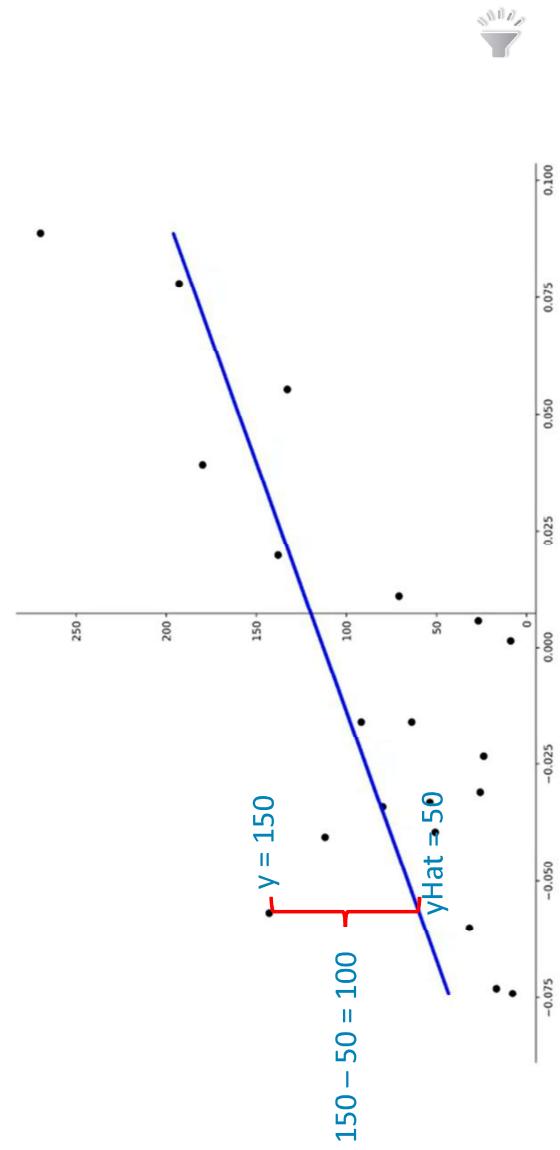
Hệ số xác định

70



### 7.1. Mean Squared Error (MSE)

#### Mean Squared Error (MSE)

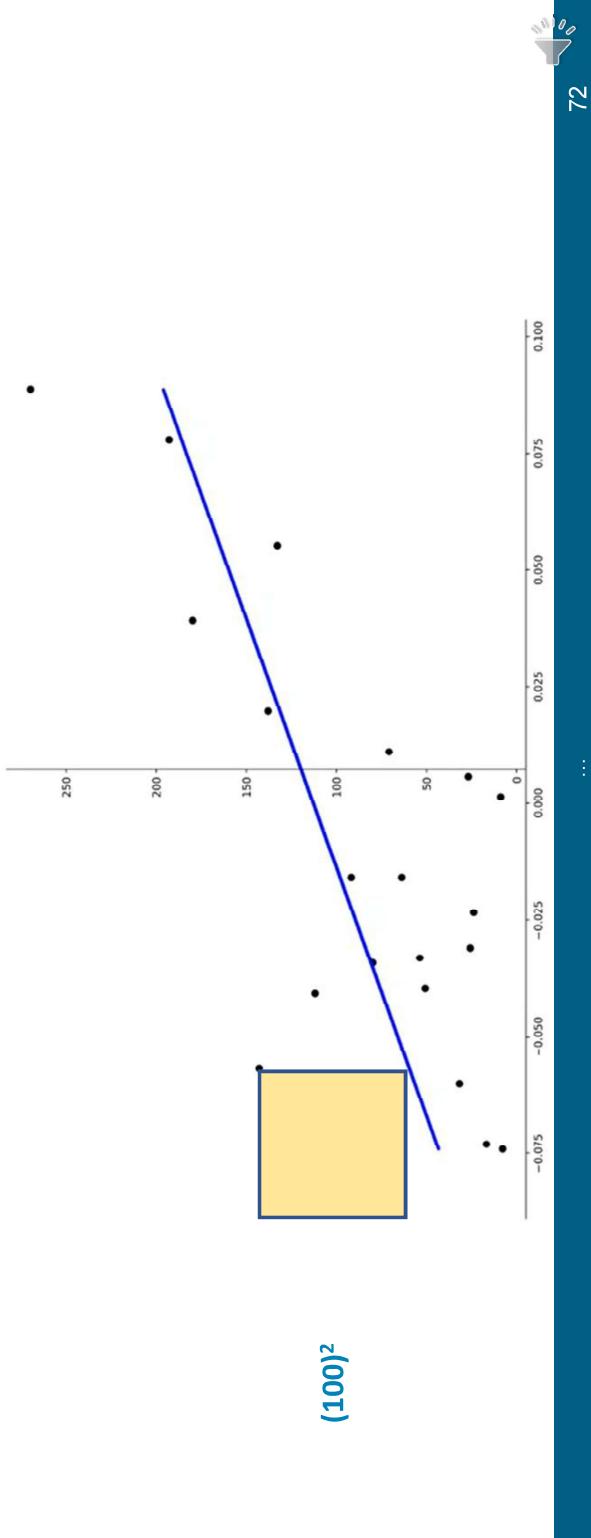


71



## 7.1. Mean Squared Error (MSE)

### Mean Squared Error (MSE)



## 7.1. Mean Squared Error (MSE)

### Mean Squared Error (MSE)



Số lượng mẫu



## 7.1. Mean Squared Error (MSE)

### Mean Squared Error (MSE)

- Sử dụng mô-đun **metrics** trong **sklearn** để tính thang đo MSE như sau:

```
from sklearn.metrics import mean_squared_error  
  
mean_squared_error(df['price'], Y_predict_simple_fit)
```

3163502.944639888

- Thủ tính bằng cách thủ công?



74

...



## 7.2 R-squared ( $R^2$ )

### R-squared ( $R^2$ )

- Hệ số xác định (coefficient of determination)  $\Leftrightarrow$  R-squared ( $R^2$ )
- Là một thang đo để xác định độ phù hợp của dữ liệu so với **đường hồi quy** đã được fit.
- $R^2$ : thể hiện tỉ lệ phần trăm thay đổi (sự biến thiên) của biến **target** ( $Y$ ), được giải thích bởi mô hình tuyến tính.



75

...



## 7.2 R-squared ( $R^2$ )

### Công thức tính R-squared ( $R^2$ )

$$R^2 = 1 - \frac{\text{MSE của đường hồi quy}}{\text{MSE của đường trung bình}}$$



## 7.1. R-squared ( $R^2$ )

### MSE của đường trung bình

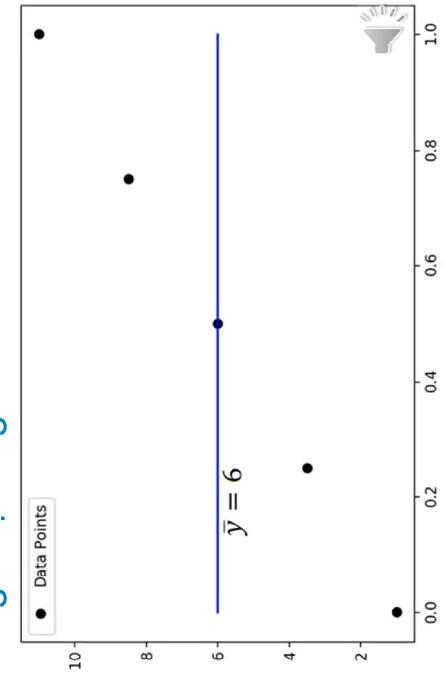
X là biến độc lập

Y là biến phụ thuộc

- Y là trung bình cộng của biến Y, tức giá trị trung bình của mỗi điểm dữ liệu

- Ví dụ hình bên cạnh  $\bar{y} = 6$

Tính MSE của Y



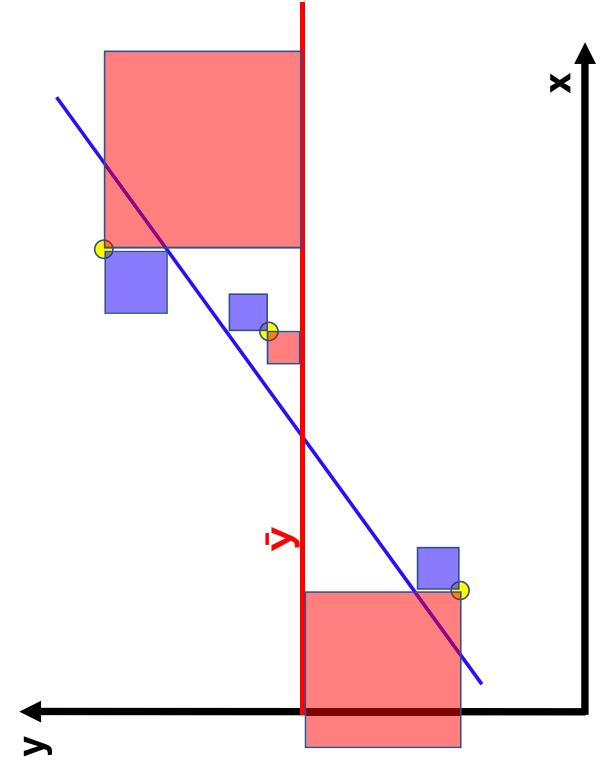
## 7.1. R-squared ( $R^2$ )



Y

### Phương pháp

- Đường màu xanh thể hiện đường hồi quy.
- Các hình vuông màu xanh thể hiện MSE của đường hồi quy.
- Đường màu đỏ thể hiện giá trị trung bình của các điểm dữ liệu (còn gọi là đường trung bình  $\bar{y}$ )
- Các hình vuông màu đỏ thể hiện MSE của đường màu đỏ.



78

=> Diện tích của **hình vuông** màu xanh nhỏ hơn nhiều so với diện tích của **hình vuông** màu **đỏ**.

## 7.1. R-squared ( $R^2$ )



### Phương pháp

- Nếu đường hồi quy là **1** đường tốt thì tỉ lệ các diện tích của MSE tiến về 0.

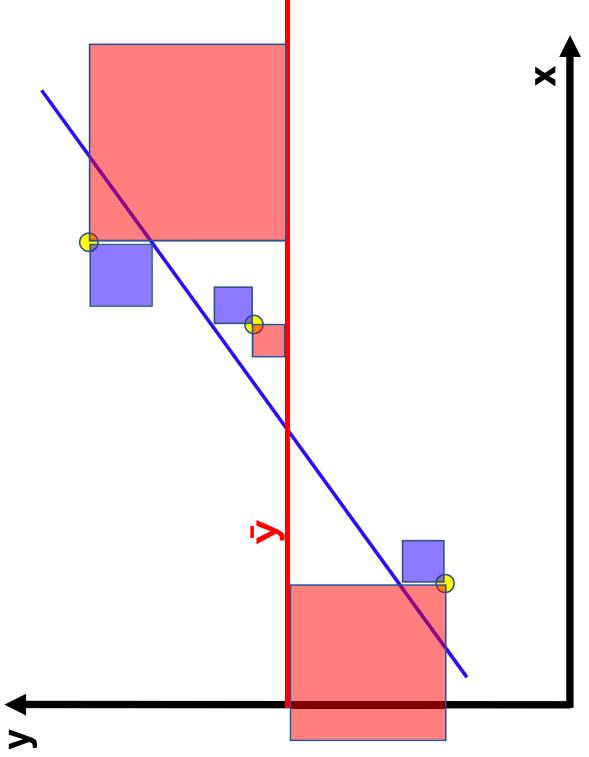
$$\frac{\text{MSE của đường hồi quy}}{\text{MSE của đường trung bình } (\bar{y})} = \frac{\text{Hình vuông màu xanh} + \text{Hình vuông màu xanh} + \dots}{\text{Hình vuông màu đỏ} + \text{Hình vuông màu đỏ} + \dots} = \text{tiến về 0}$$

79



## 7.1. R-squared ( $R^2$ )

### Phương pháp

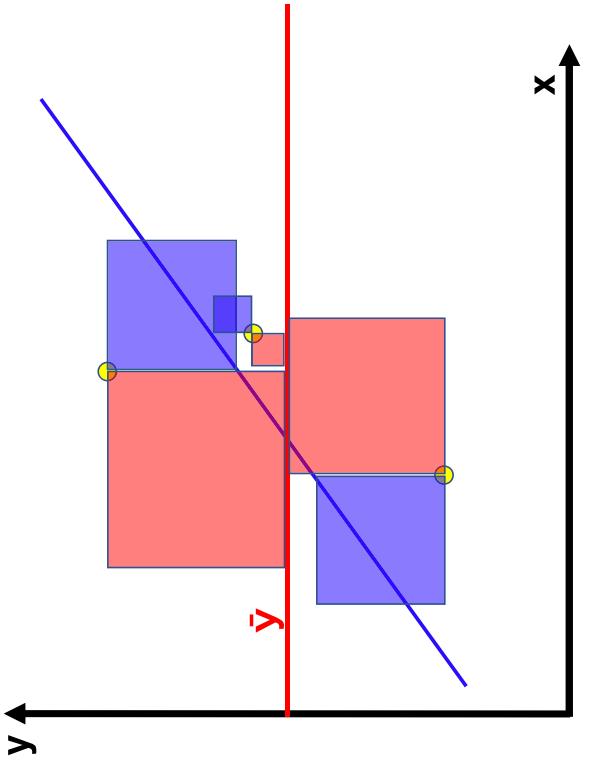


80

...

## 7.1. R-squared ( $R^2$ )

### Phương pháp



83

## 7.1. R-squared ( $R^2$ )

### Tính $R^2$

- Giá trị MSE thuộc đoạn [0; 1]
  - Gọi phương thức score() trong đối tượng mô hình lm để tính  $R^2$
- ```
X = df[['highway-mpg']]  
Y = df['price']
```

```
lm.fit(X, Y)
```

```
lm.score(X, Y)  
0.496591188
```



85

## 8. Đánh giá



Để xác định một kết quả fit tốt nhất (final best fit), chúng ta nên xem xét các tiêu chí sau:

1. Các giá trị dự đoán có hợp lý không?
2. Sử dụng trực quan.
3. Thang đo định lượng  $R^2$  (tiến về 1) và MSE
4. So sánh các mô hình lẫn nhau.



86



## 8. Đánh giá

### Đánh giá kết quả dự đoán

- Xây dựng mô hình

```
lm.fit (df ['highway-mpg'] , df ['prices'])
```

- Kết quả dự đoán cho biến highway-mpg có giá trị là 30

```
lm.predict (30)
```

- Kết quả: \$13771.30

```
lm.coef_
```

-821.73337832

```
Price = 38423.31 - 821.73 * highway-mpg
```

...  
87



## 8. Đánh giá

### Đánh giá kết quả dự đoán

- Sử dụng import numpy

```
import numpy as np
```

- Dùng phương thức `arrange()` để tạo một mảng giá trị từ 1 đến 100  
`new_input=np.arange(1,101,1).reshape(-1,1)`

|   |   |     |    |     |
|---|---|-----|----|-----|
| 1 | 2 | ... | 99 | 100 |
|---|---|-----|----|-----|



## 8. Đánh giá

**Đánh giá kết quả dự đoán**

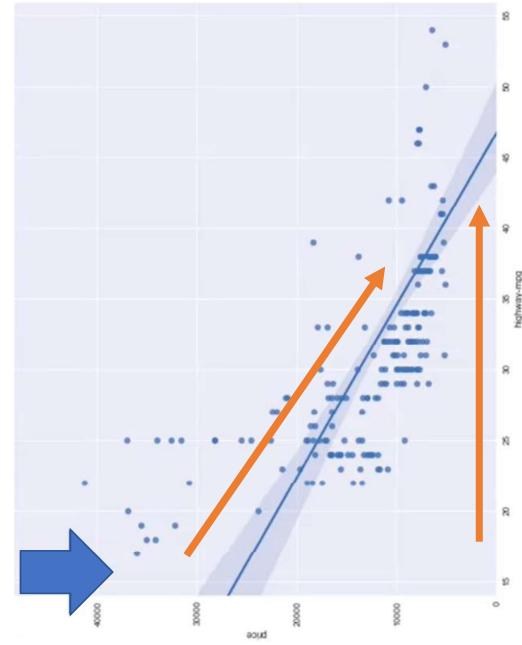
```
yhat=lm.predict(new_input)
```

```
array([ 34314.6, 33896.5, 33972.893910151., 32692.90558823, 31027.705451326., 30205.97207494, 27740.77193997, 26919.03858165, 24853.53842668, 23632.10504836, 21166.9049134., 20345.17155308, 17879.97140011., 17058.23802179, 13771.30480585 /, 10484.37099521, 11306.10437353, 8019.1708025., 4732.23734696, 1445.3038367., -1841.62962962., -5128.5631929., -8415.49670619., -11702.4302198., -14989.36337277., -18276.29724606., -15153.23807523., -24850.13647263., -28137.09778592., -31124.03129921., -3471.88982578., -41284.83183907., 33414.6389655, 33972.893910151., 32692.90558823, 31027.705451326., 30205.97207494, 27740.77193997, 26919.03858165, 24853.53842668, 23632.10504836, 21166.9049134., 20345.17155308, 17879.97140011., 17058.23802179, 13771.30480585 /, 10484.37099521, 11306.10437353, 8019.1708025., 4732.23734696, 1445.3038367., -1841.62962962., -5128.5631929., -8415.49670619., -11702.4302198., -14989.36337277., -18276.29724606., -15153.23807523., -24850.13647263., -28137.09778592., -31124.03129921., -3471.88982578., -41284.83183907., 32671.1720295 /, 29384.23869662, 26097.30518333, 21801.37167004, 19523.43816575, 16236.50464347, 13771.57113018, 9662.63716889, 6375.7041036 /, 3088.77095031, 623.57045535, -2663.36307794, -5950.29657123, -9237.23008451, -12524.16359718, -1345.89697612, -1581.0911109, -16632.83084891, -1998.03062438, -23394.01417676, -25671.89763095, -28958.83116424, -32245.76467753, -35538.68819082 /, -38819.6317041 /, -42106.56521739., 35975.0772319, 32671.1720295 /, 29384.23869662, 26097.30518333, 21801.37167004, 19523.43816575, 16236.50464347, 13771.57113018, 9662.63716889, 6375.7041036 /, 3088.77095031, 623.57045535, -198.16292977, -169.16292977, -3485.09643262, -6772.-02949455, -10058.96346284, -13345.89697612, -14165.83084891, -17454.56386773, -20741.49738102 /, -23066.69715159, -26493.63102927, -29780.56454256, -33067.49805585, -36463.36307794, -39641.363084891, -43156.94137146, -46463.09846075 /, -42998.29859571])
```

68

## 8. Đánh giá

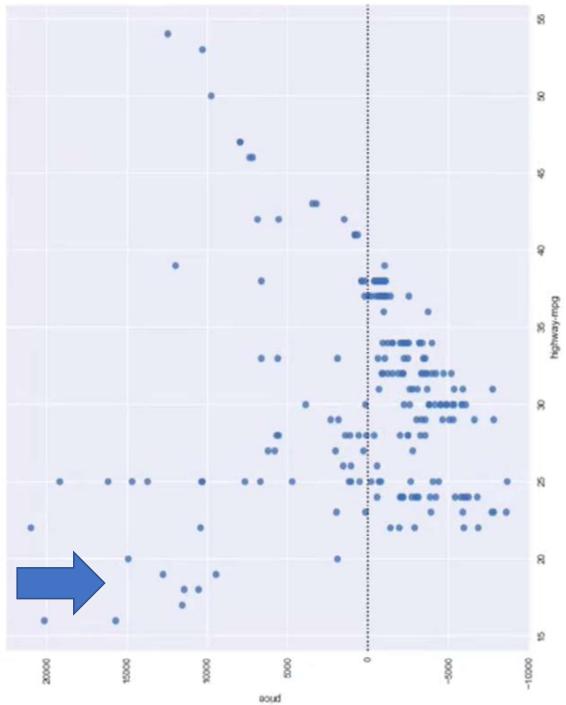
## Dùng trực quan để đánh giá – Regression plot





## 8. Đánh giá

Dùng trực quan để đánh giá – *Residual plot*

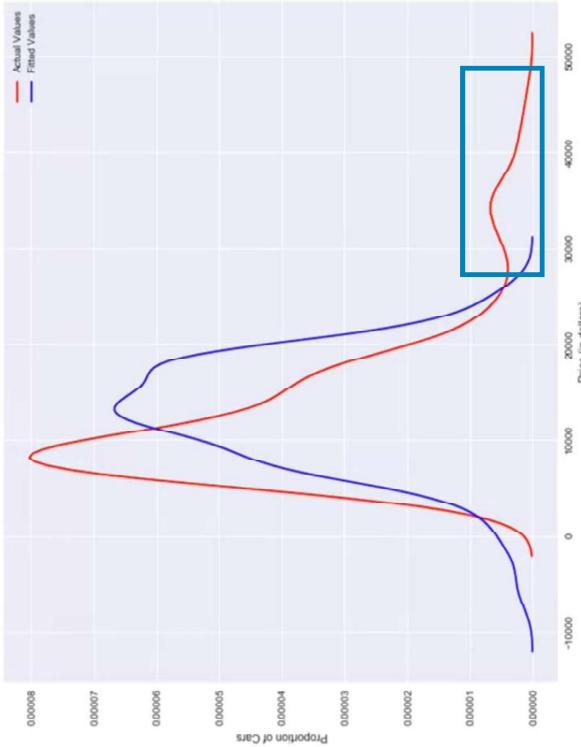


91



## 8. Đánh giá

Dùng trực quan để đánh giá – *Distribution plot*



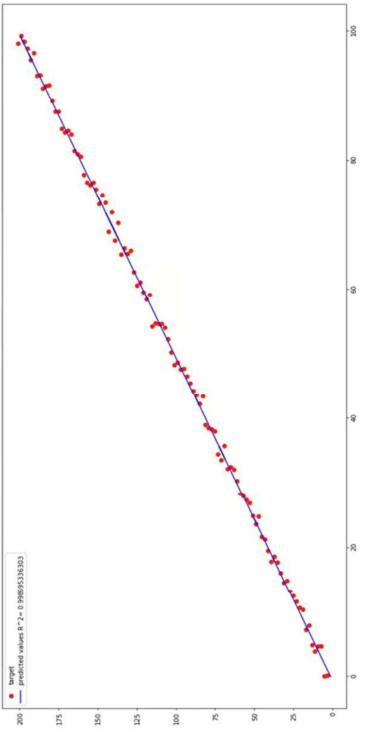
92



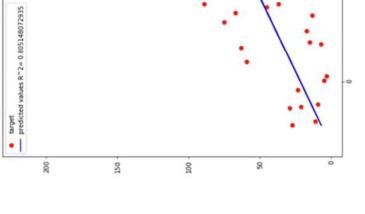


## 8. Đánh giá

**$R^2$**



A



B



93



## 8. Đánh giá

**Bài tập: So sánh giữa hồi quy tuyến tính đa biến và đơn biến**

1. Có phải MSE thấp hơn thì luôn luôn có một đường fit tốt hơn?
  - Không nhất thiết. Tại sao?
2. MSE của mô hình MLR sẽ luôn thấp hơn MSE của mô hình SLR, vì các sai số của dữ liệu sẽ giảm khi nhiều biến có trong mô hình. Giải thích?
3. Hồi quy đa thức cũng sẽ có MSE thấp hơn MLR và SLR. Giải thích?
4.  $R^2$

Chương sau sẽ giúp chúng ta tìm ra hướng tốt hơn để đánh giá mô hình.

94



97



Thank you



## Hỏi - Đáp

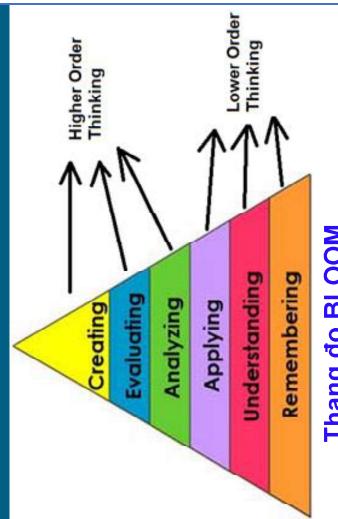


# Chương 5

## ĐÁNH GIÁ MÔ HÌNH



### Mục tiêu



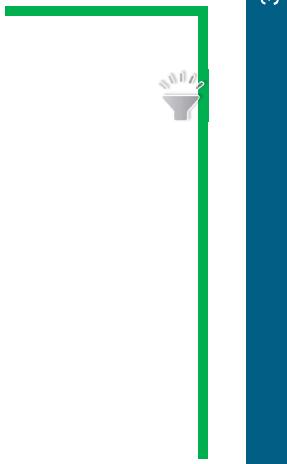
Thang đo BLOOM

- Hiểu đánh giá mô hình
- Phân tích Over-fitting, Under-fitting và hiểu cách chọn mô hình
- Hiểu hồi quy Ridge /rɪdʒ/
- Hiểu Grid Search /grɪd/
- Phân tích để sàng lọc mô hình



## Nội dung

- 1. Kỹ thuật đánh giá mô hình
- 2. Overfitting - Underfitting - Chọn mô hình
- 3. Hồi quy Ridge
- 4. Grid Search



3

## Bộ dữ liệu



## Lý thuyết và Thực hành:

[https://raw.githubusercontent.com/datasetshub/ds105/master/  
Model-Evaluation-and-Refinement.csv](https://raw.githubusercontent.com/datasetshub/ds105/master/Model-Evaluation-and-Refinement.csv)



4

## 1. Kỹ thuật đánh giá mô hình



- 1.1. Phương pháp đánh giá mô hình
- 1.2. Hiệu suất
- 1.3. Kiểm chứng chéo
- 1.4. Tóm tắt các thang đo đánh giá



5

### 1.1. Phương pháp đánh giá mô hình



#### Giới thiệu

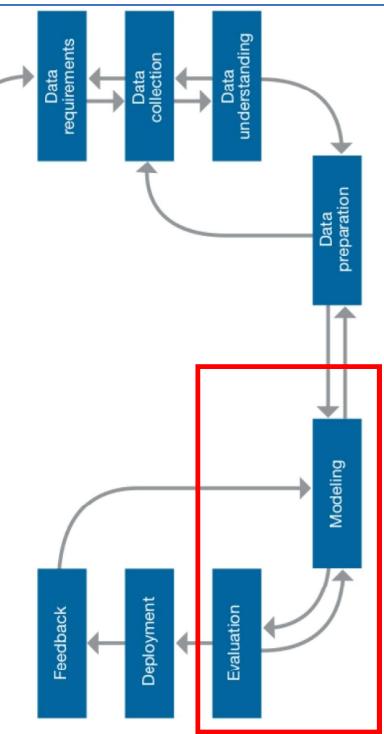
- Đánh giá tập mẫu cho phép chúng ta biết được mô hình có phù hợp với dữ liệu được dùng để phát triển mô hình.

- Vấn đề đặt ra:

- Không biết được mô hình có thể **độ chính xác** trị mới tốt như thế nào.

- Giải pháp

- **Đánh giá tập dữ liệu dùng để phát triển mô hình** (gọi là training set)
  - Tập dùng để **kiểm tra** (gọi là testing set)



John B. Rollins, Ph.D.

6



## 1.1. Phương pháp đánh giá mô hình

### Training/Testing Set

- Bộ dữ liệu



Tách bộ dữ liệu thành:

- Training set (70%);
- Testing set (30%);
- Phát triển mô hình với tập huấn luyện (training set)
- Dùng tập kiểm thử (testing set) để đánh giá hiệu suất dự đoán của mô hình.
- Khi hoàn thành đánh giá mô hình nên sử dụng lại tất cả dữ liệu để phát triển lại mô hình để có hiệu suất tốt nhất.



7



## 1.1. Phương pháp đánh giá mô hình

### Hàm train\_test\_split()

- Tách bộ dữ liệu thành hai bộ train và test bằng kỹ thuật ngẫu nhiên

```
from sklearn.model_selection import train_test_split  
x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.3, random_state=0)
```

- **x\_data**: biến độc lập
- **y\_data**: biến phụ thuộc => df['price']
- **x\_train, y\_train**: biến độc lập và phụ thuộc của bộ train
- **x\_test, y\_test**: biến độc lập và phụ thuộc của bộ test
- **test\_size**: kích cỡ bộ test, đơn vị % (30%)
- **random\_state**: số dùng để làm cơ sở kiểm soát việc xáo trộn dữ liệu.



8



## 1.2. Hiệu suất

### Generalization performance (Hiệu suất tổng thể, hay Hiệu suất chung)

- Mô hình dự đoán chính xác phần lớn các trường hợp của bộ **test** thì mô hình đó đạt được độ khai quát hóa tốt (tức là *generalization performance*).
- Lỗi tổng thể (*generalization error*) là thang đo thể hiện độ tốt của bộ **train** dùng để phát triển mô hình.
- Lỗi tổng thể được tính bằng cách sử dụng bộ **test** đưa vào mô hình.

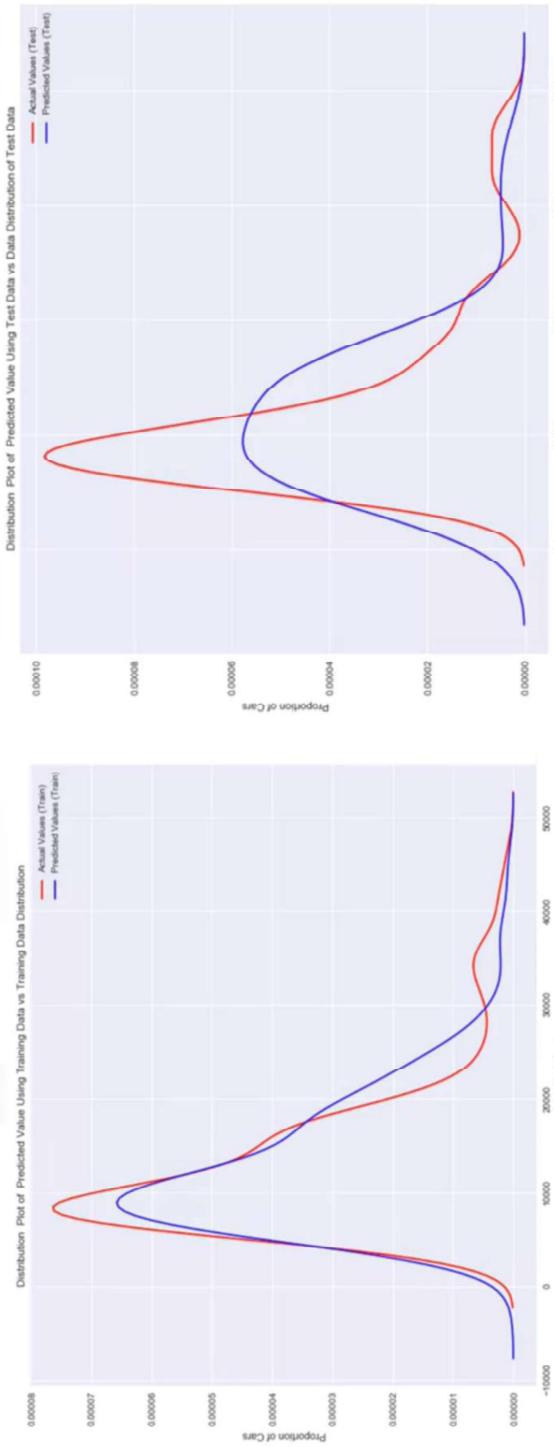
...

9



## 1.2. Hiệu suất

### Generalization error



...

10



### 1.3. Kiểm chứng chéo

#### Cross validation (*k*-Fold Cross-Validation)

- Cross validation (kiểm chứng chéo) là phương pháp thống kê lấy mẫu để kiểm tra đánh giá độ chính xác mô hình.
  - So sánh và chọn ra mô hình tốt nhất.
- Sử dụng phương pháp Cross-Validation với *k*-Fold => tương đương: ***k*-Fold Cross-Validation**



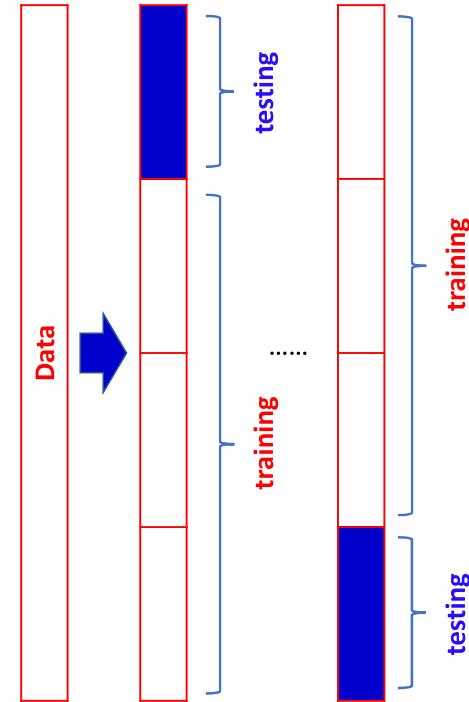
11



### 1.3. Kiểm chứng chéo

#### Cross validation (*k*-Fold Cross-Validation)

- Số liệu đánh giá ngoài mẫu phổ biến nhất.
- Sử dụng dữ liệu hiệu quả hơn (dữ liệu được sử dụng cho cả train và test)



12



### 1.3. Kiểm chứng chéo

#### Cross validation (*k*-Fold Cross-Validation)

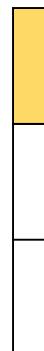
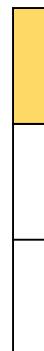
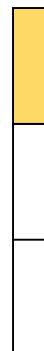
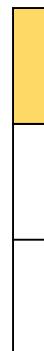


| Train  | Validation | Kết quả |
|--------|------------|---------|
| Fold 1 | Fold 2     | a1      |
| Fold 1 | Fold 2     | Fold 5  |
| Fold 1 | Fold 2     | Fold 3  |
| Fold 1 | Fold 2     | Fold 4  |
| Fold 1 | Fold 5     | Fold 4  |
| Fold 5 | Fold 2     | Fold 4  |
| Fold 5 | Fold 3     | Fold 4  |
|        |            | ...     |
|        |            | 13      |



### 1.3. Kiểm chứng chéo

#### Hàm cross\_val\_score()





### 1.3. Kiểm chứng chéo

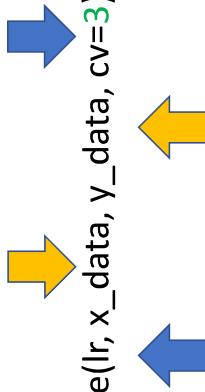
#### Hàm cross\_val\_score()

```
from sklearn.model_selection import cross_val_score  
  
scores = cross_val_score(lr, x_data, y_data, cv=3)  
  
np.mean(scores)
```



15

...



### 1.3. Kiểm chứng chéo

#### Hàm cross\_val\_predict()

|       |       |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | $x_9$ |
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | $x_9$ |
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | $x_9$ |



16

...



### 1.3. Kiểm chứng chéo

#### Hàm `cross_val_predict()`

- Trả về giá trị dự đoán của mỗi phần tử trong bộ test
- Có interface (giao diện) sử dụng giống như `cross_val_score()`

```
from sklearn.model_selection import cross_val_predict
```

```
yhat= cross_val_predict (lr2e, x_data, y_data, cv=3)
```

...  
17



### 1.4. Các thang đo đánh giá

#### Tóm tắt các thang đo đánh giá

- In-Sample Evaluation:

- **Mean Squared Error (MSE):** gọi `mean_squared_error()` trong `metrics`
- **R-squared (R^2):** gọi `score()` trong mô hình.

- Out-Sample Evaluation:

- **Cross validation (CV):** gọi `cross_val_score()` trong `model_selection` để thảm định chéo mô hình hồi quy.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^n (p_i - a_i)^2}{n}}$$

...  
18



## 1.4. Các thang đo đánh giá

### Tóm tắt các thang đo đánh giá

- Ngoài ra, còn các thang đo như:

- Relative Squared Error (**RSE**)

$$RSE = \frac{\sum_{i=1}^n (p_i - a_i)^2}{\sum_{i=1}^n (\bar{a} - a_i)^2}$$

- Mean Absolute Error (**MAE**)

$$MAE = \frac{\sum_{i=1}^n |p_i - a_i|}{n}$$

- Relative Absolute Error (**RAE**)

$$RAE = \frac{\sum_{i=1}^n |p_i - a_i|}{\sum_{i=1}^n |\bar{a} - a_i|}$$

=> Nhiệm vụ: tìm hiểu thêm các thang đo đã liệt kê bên trên.



19



## 2. Overfitting - Underfitting – Chọn mô hình

- 2.1. Overfitting (mô hình quá khớp)
- 2.2. Underfitting (mô hình chưa khớp)
- 2.3. Chọn mô hình
- 2.4. Minh họa



20



## 2.1. Overfitting

### Overfitting (Mô hình quá khớp)

- Là mô hình áp dụng đạt kết quả **cao (tốt)** cho bộ dữ liệu **train** nhưng áp dụng đạt kết quả **thấp (không tốt)** cho bộ dữ liệu **test**.

#### Nguyên nhân:

- Do dữ liệu nhiễu, hoặc dữ liệu bất thường (**outliers**) được chọn góp phần đưa ra quy luật mô hình.
- Hay, dữ liệu nhiễu/bất thường tham gia train góp phần “bẻ gãy” đường hồi quy “=> **làm mềm mại đường hồi quy**”



21



## 2.2. Underfitting

### Underfitting (Mô hình chưa khớp)

- Là mô hình chưa có độ chính xác cao đối với tập dữ liệu test sau khi phát triển.

#### Nguyên nhân và cách khắc phục:

- Do tập dữ liệu train chưa đạt chuẩn (cách kiểm tra: thực hiện phân tích thăm dò lại bộ dữ liệu)
  - Bổ sung thêm dữ liệu train.
  - Hoặc đổi thuật toán.



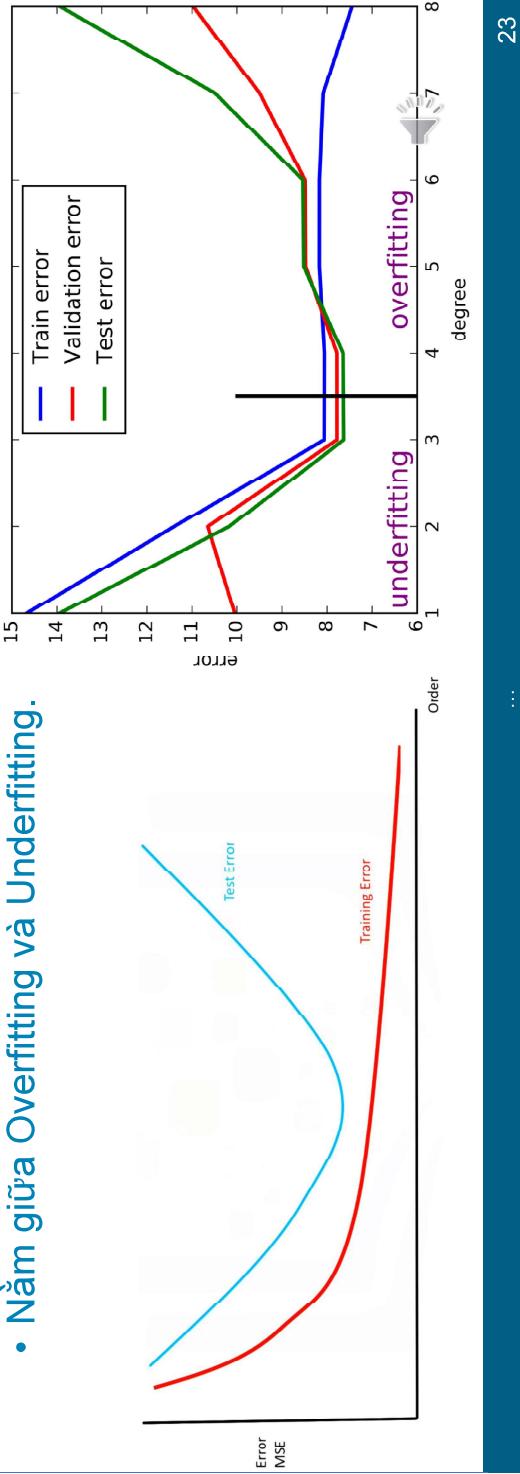
22



## 2.3. Chọn mô hình

### Model selection => Good Fitting

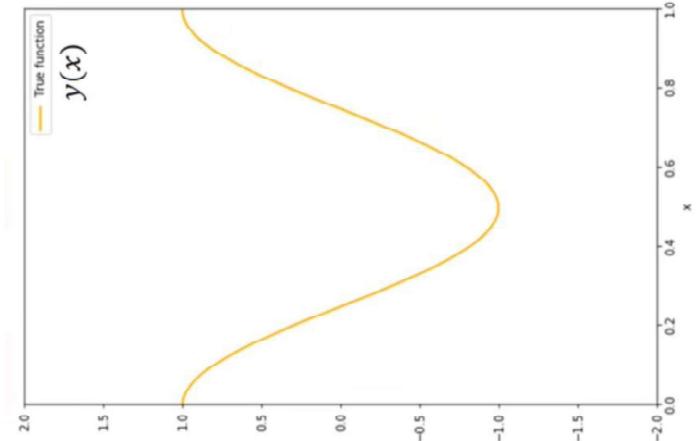
- Chọn mô hình làm sao để vừa khớp (Good Fitting).
- Nằm giữa Overfitting và Underfitting.



23



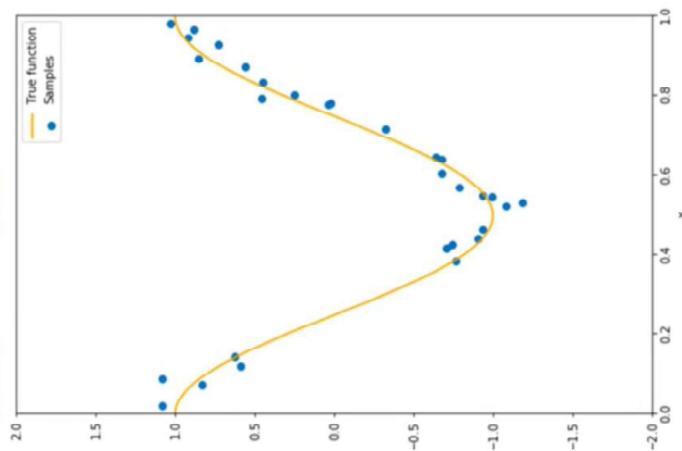
## 2.3. Chọn mô hình



24



## 2.3. Chọn mô hình

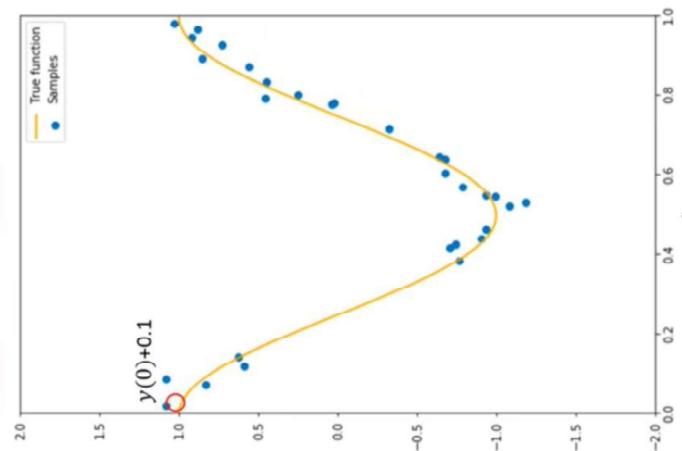


$y(x) + \text{noise}$



25

## 2.3. Chọn mô hình

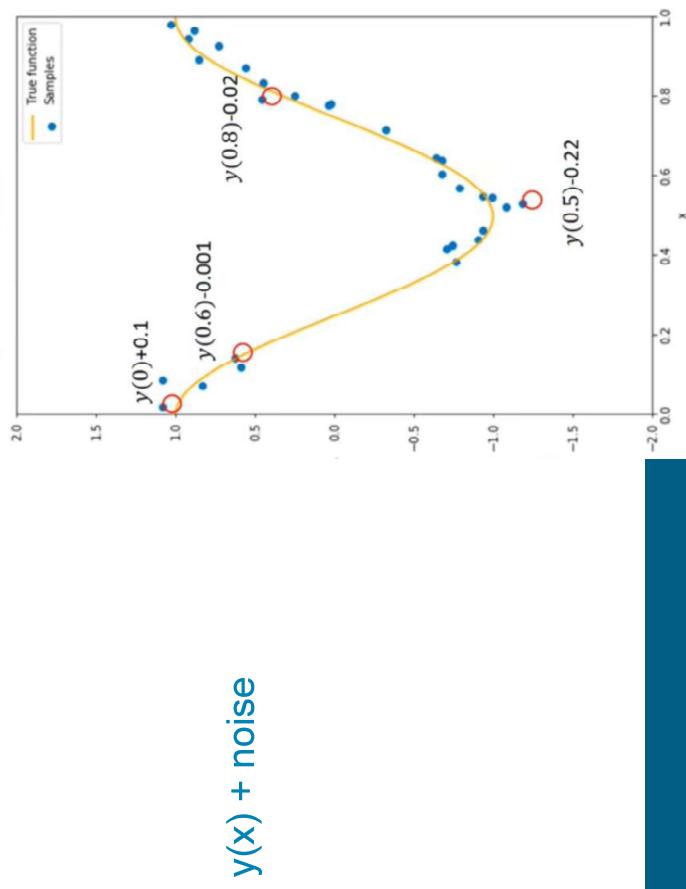


$y(x) + \text{noise}$



26

## 2.3. Chọn mô hình

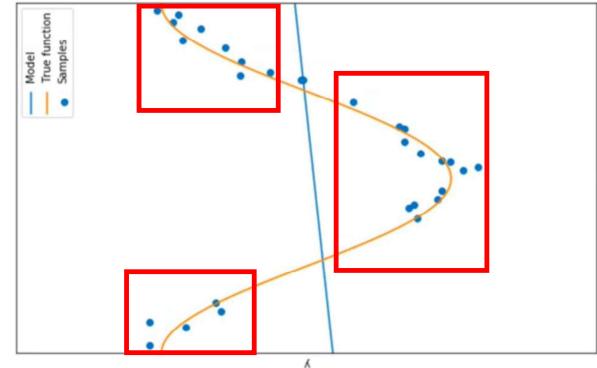


27

## 2.3. Chọn mô hình



$$y = b_0 + b_1 x$$

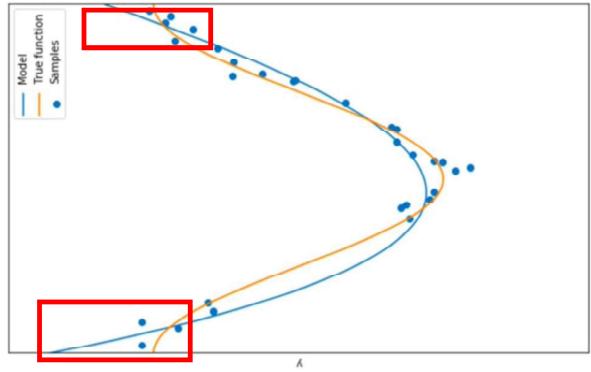


28

### 2.3. Chọn mô hình



$$y = b_0 + b_1x + b_2x^2$$

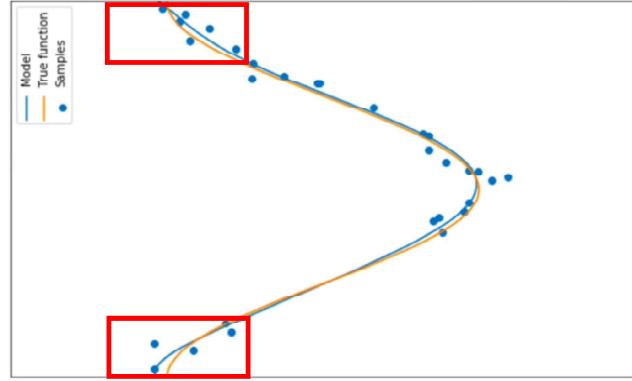


29

### 2.3. Chọn mô hình

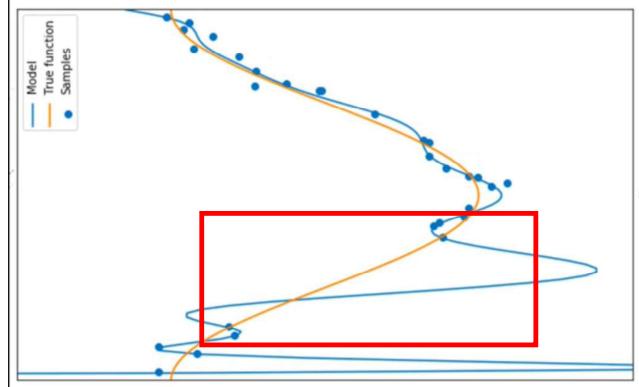


$$y = b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7 + b_8x^8$$



30

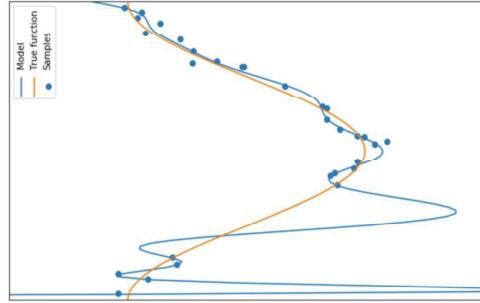
## 2.3. Chọn mô hình



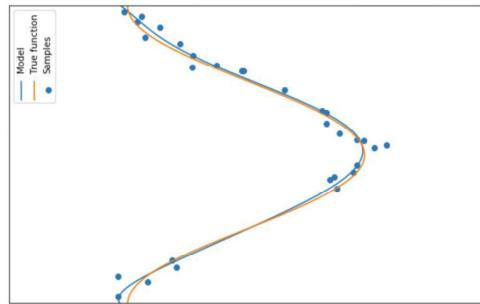
31

$$y = b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7 + b_8x^8 + \dots + b_{15}x^{15} + b_{16}x^{16}$$

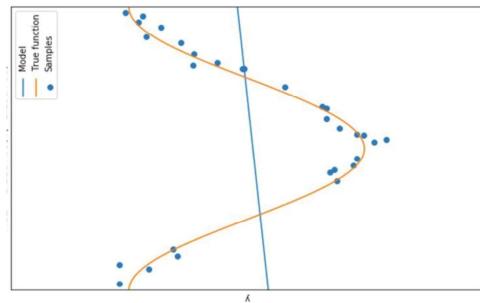
## 2.3. Chọn mô hình



c



b

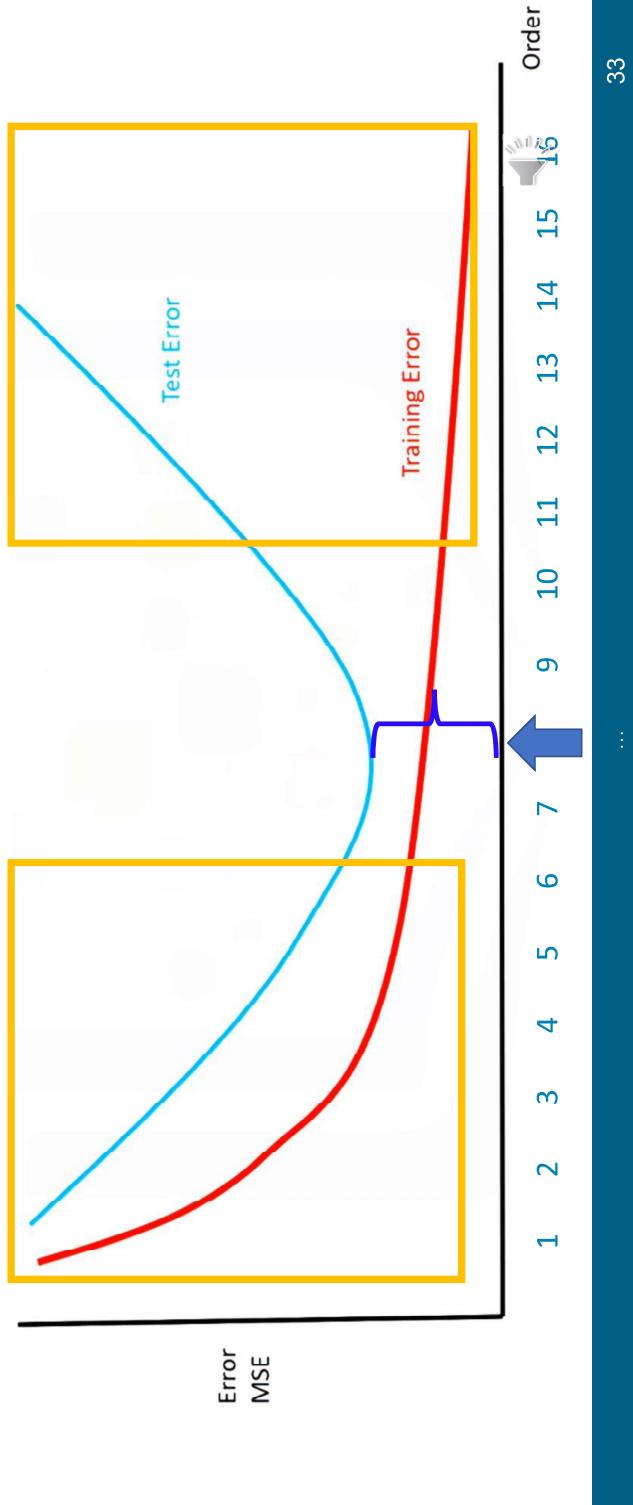


a

32



## 2.3. Chọn mô hình



## 2.4. Minh họa

### Code

```
Rsqu_test=[]
```

```
order=[1, 2, 3, 4]
```

```
for n in order:
```

```
pr=PolynomialFeatures(degree=n)
```

```
x_train_pr=pr.fit_transform(x_train[['horsepower']])
```

```
x_test_pr=pr.transform(x_test[['horsepower']])
```

```
lr.fit(x_train_pr,y_train)
```

```
Rsqu_test.append(lr.score(x_test_pr,y_test))
```





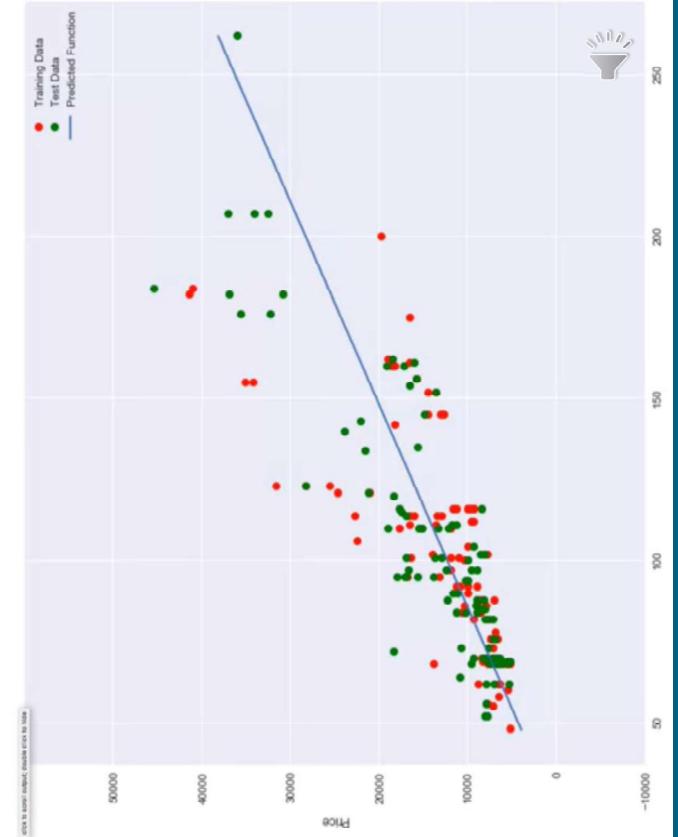
## 2.4. Minh họa

### Chọn mô hình



## 2.4. Minh họa

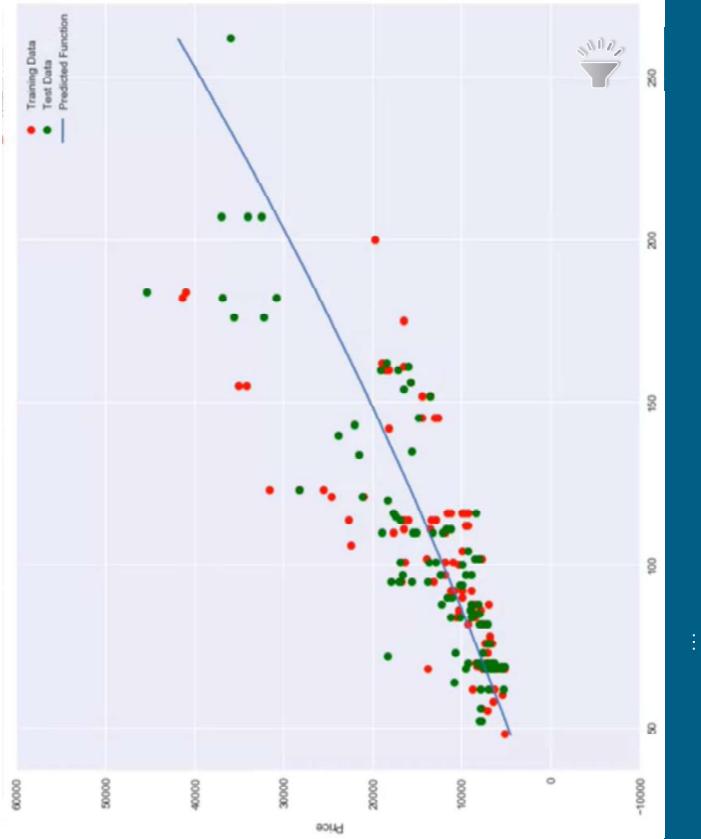
### Chọn mô hình



Bậc  $n = 1$

## 2.4. Minh họa

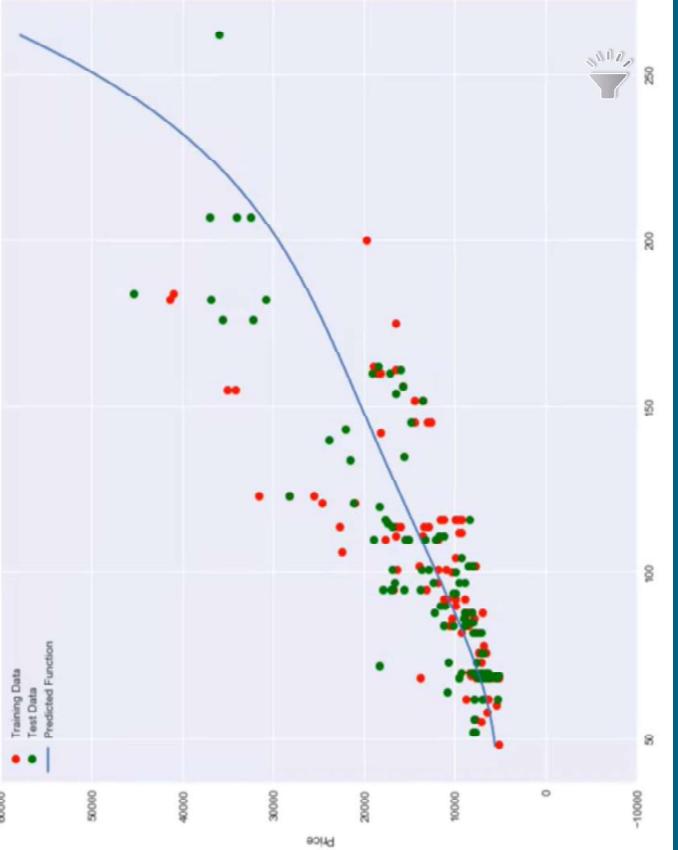
### Chọn mô hình



Bậc  $n = 2$

## 2.4. Minh họa

### Chọn mô hình



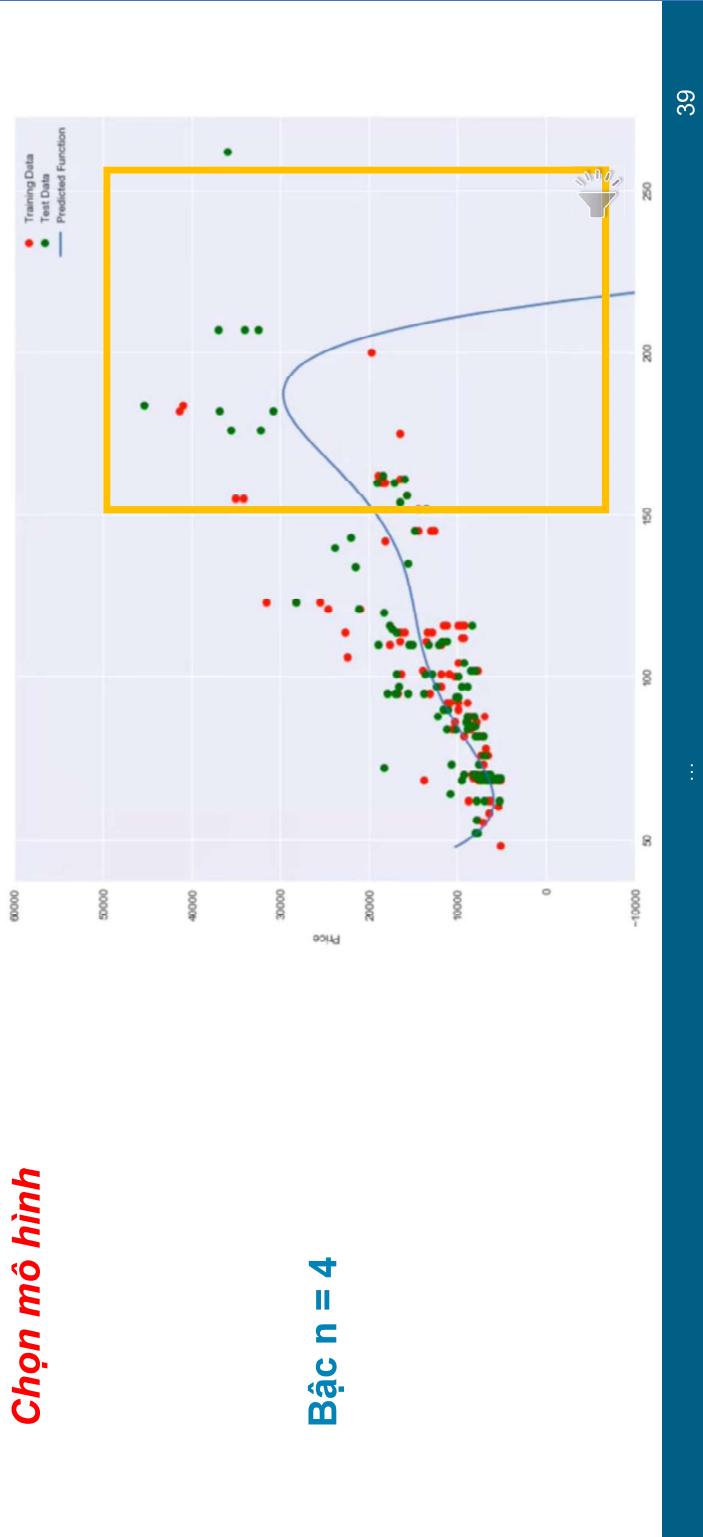
Bậc  $n = 3$





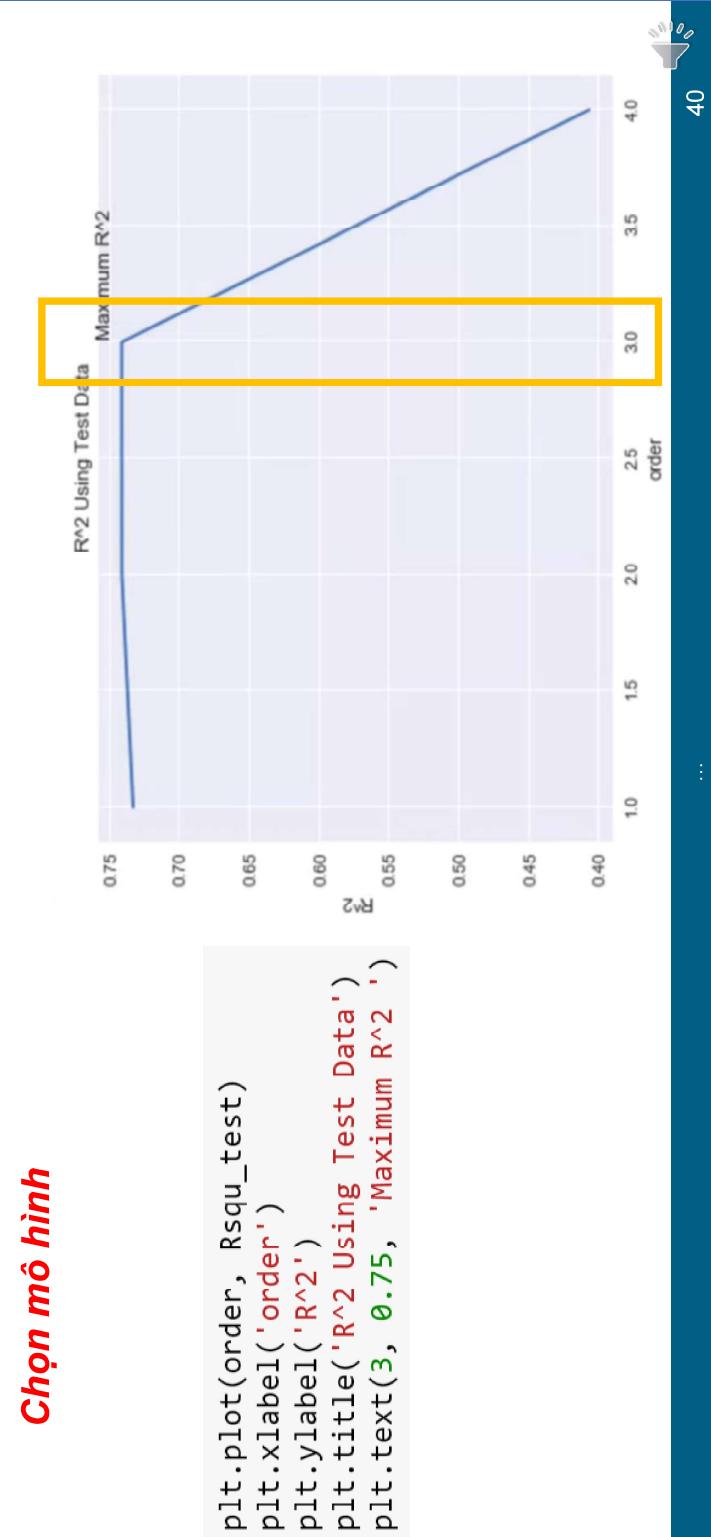
## 2.4. Minh họa

### Chọn mô hình



## 2.4. Minh họa

### Chọn mô hình



```

Rsqu_test = []

order = [1, 2, 3, 4]
for n in order:
    pr = PolynomialFeatures(degree=n)

    x_train_pr = pr.fit_transform(x_train[['horsepower']])
    x_test_pr = pr.fit_transform(x_test[['horsepower']])

    lr.fit(x_train_pr, y_train)

    Rsqu_test.append(lr.score(x_test_pr, y_test))

plt.plot(order, Rsqu_test)
plt.xlabel('order')
plt.ylabel('R^2')
plt.title('R^2 Using Test Data')
plt.text(3, 0.75, 'Maximum R^2')

```

41



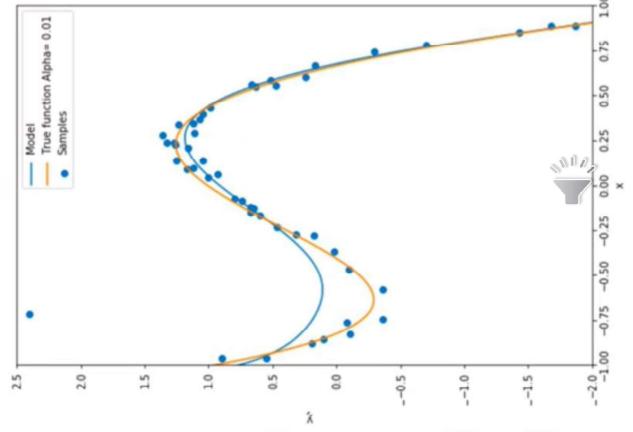
### 3. Hồi quy Ridge

#### Hồi quy Ridge:

- Cũng là một mô hình hồi quy tuyến tính phân tích mối quan hệ giữa các biến độc lập và biến phụ thuộc.
- Điều chỉnh mô hình sao cho giảm thiểu các vấn đề overfitting.

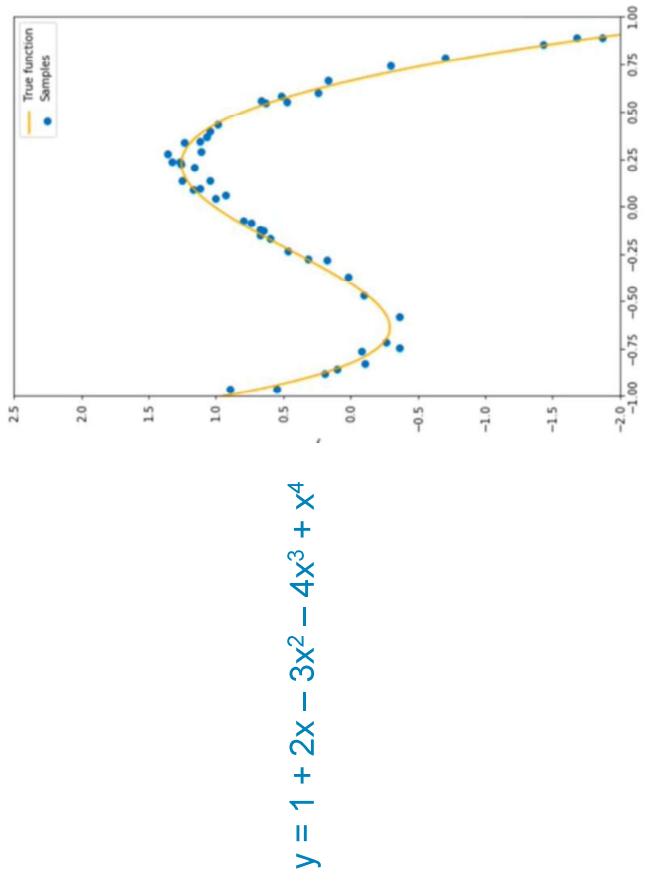
#### **Khi nào áp dụng hồi quy Ridge?**

- Áp dụng khi bộ dữ liệu có các biến độc lập có mối liên hệ với nhau.
- Áp dụng khi giải quyết overfitting.



42

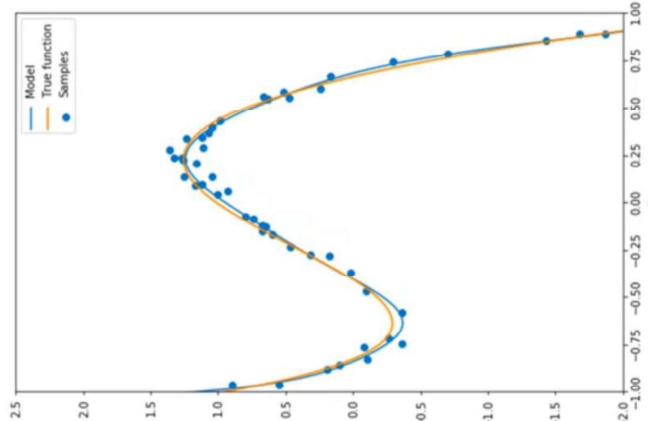
### 3. Hồi quy Ridge



43



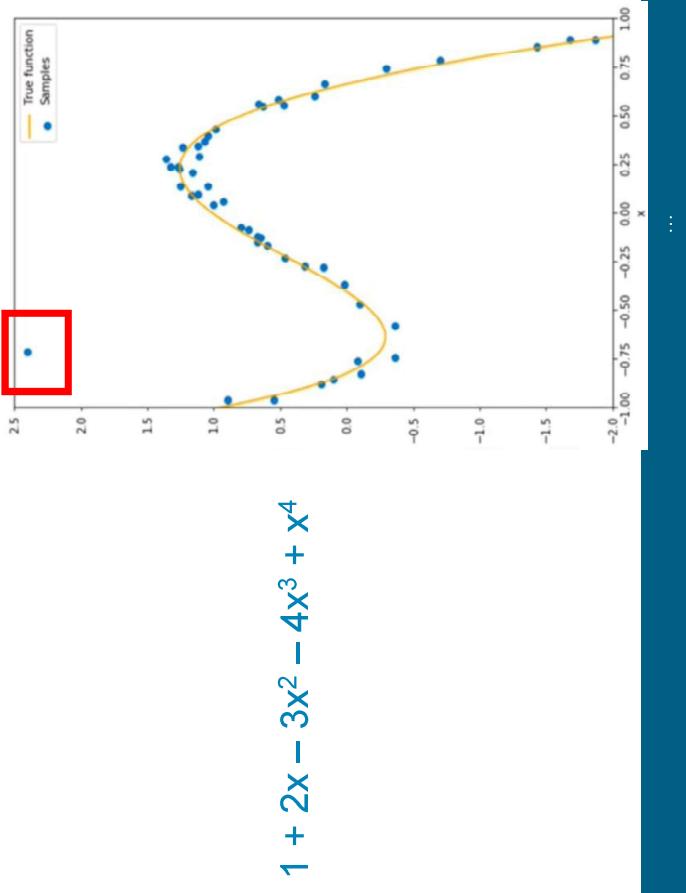
### 3. Hồi quy Ridge



44



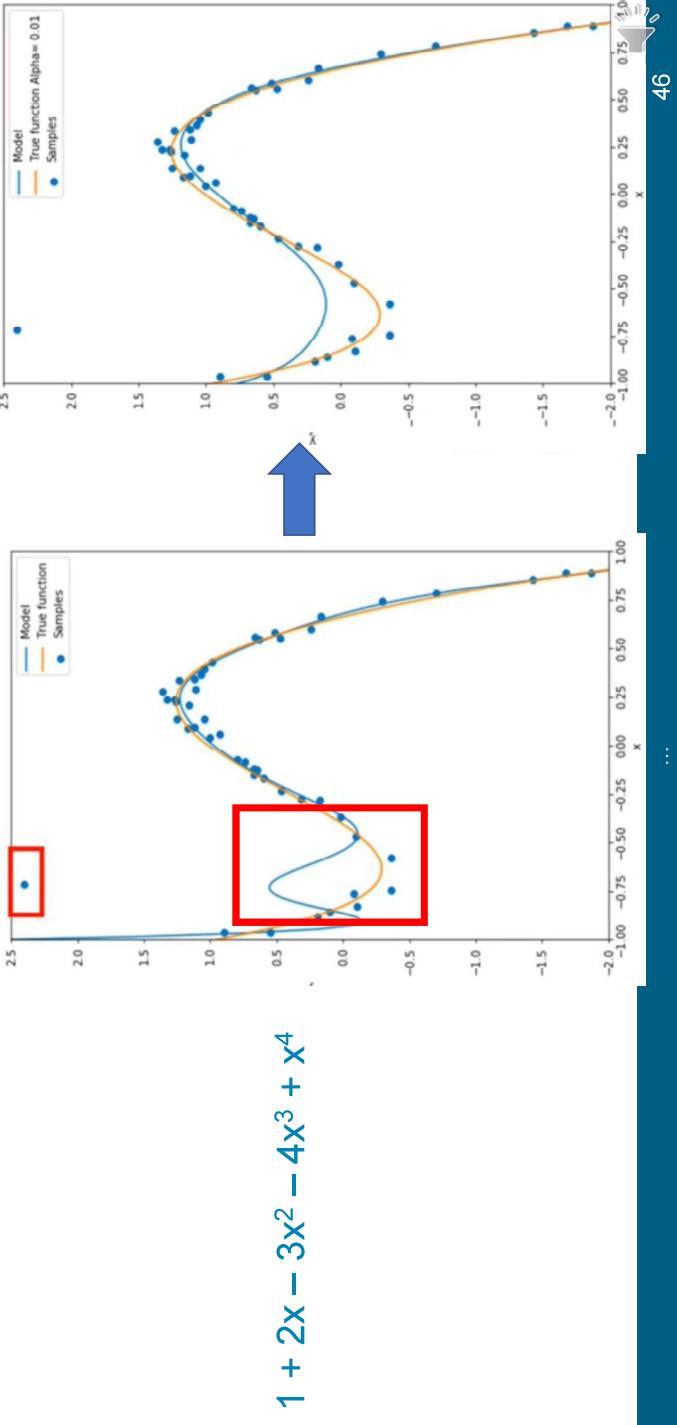
### 3. Hồi quy Ridge



### 3. Hồi quy Ridge



#### Hệ số điều chỉnh alpha

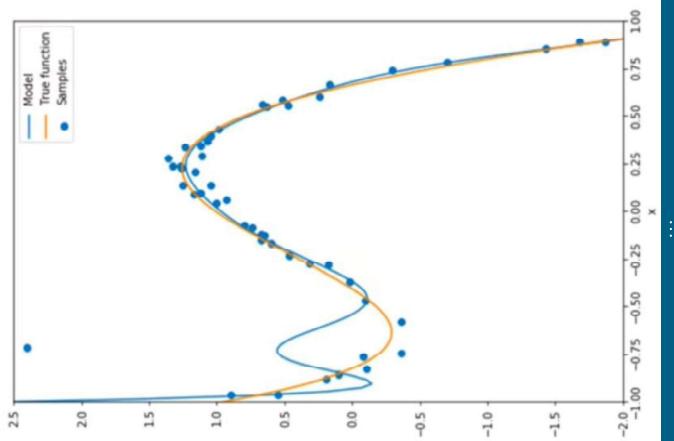




### 3. Hồi quy Ridge

Hệ số điều chỉnh alpha

| Alpha |
|-------|
| 0     |
| 0.001 |
| 0.01  |
| 1     |
| 10    |



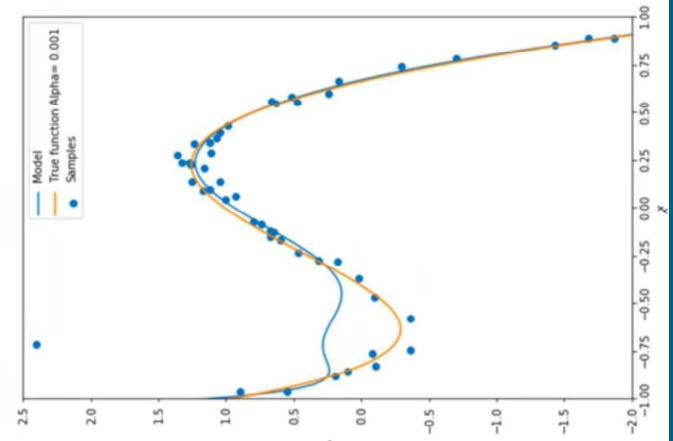
48



### 3. Hồi quy Ridge

Hệ số điều chỉnh alpha

| Alpha |
|-------|
| 0     |
| 0.001 |
| 0.01  |
| 1     |
| 10    |

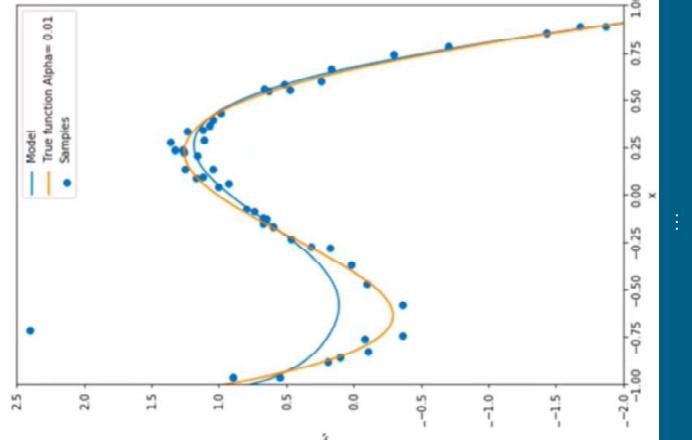


49



### 3. Hồi quy Ridge

Hệ số điều chỉnh alpha

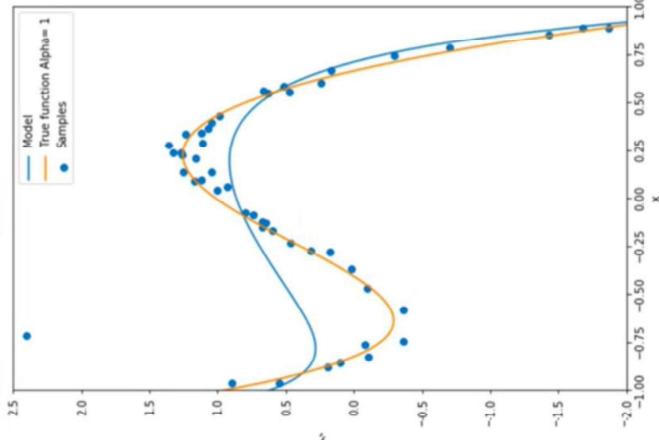


| Alpha |
|-------|
| 0     |
| 0.001 |
| 0.01  |
| 1     |
| 10    |



### 3. Hồi quy Ridge

Hệ số điều chỉnh alpha

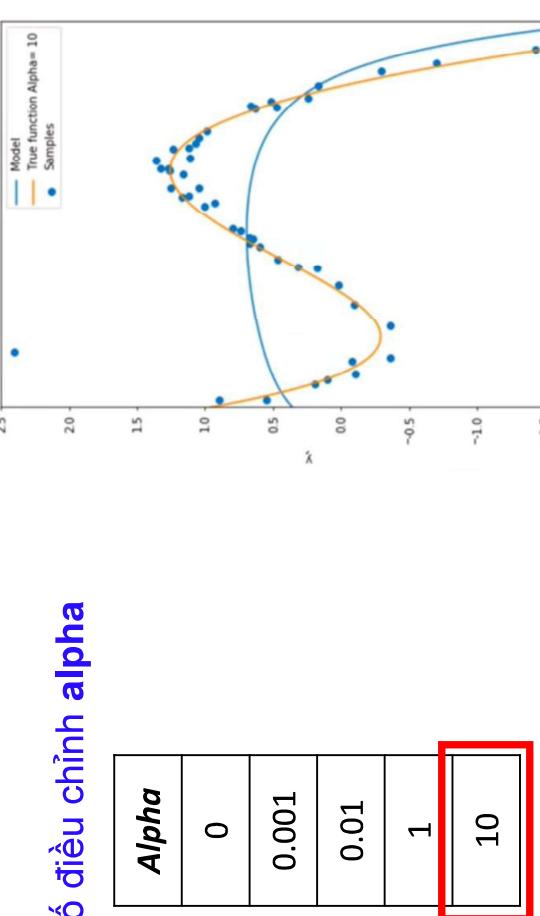


| Alpha |
|-------|
| 0     |
| 0.001 |
| 0.01  |
| 1     |
| 10    |



### 3. Hồi quy Ridge

Hệ số điều chỉnh alpha



Vậy, đưa vào đầu để xác định hệ số điều chỉnh alpha là tốt nhất?



52



### 3. Hồi quy Ridge

Cài đặt

```
from sklearn.linear_model import Ridge
```

```
RidgeModel=Ridge(alpha=0.1)
```

```
RidgeModel.fit(X,y)
```

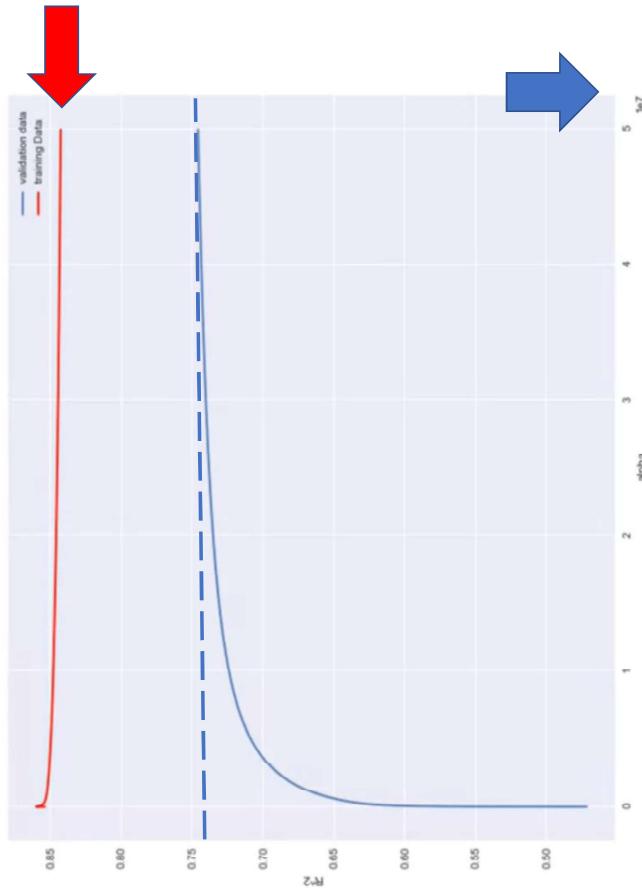
```
Yhat=RidgeModel.predict(X)
```



54



### 3. Hồi quy Ridge



55



Thank you



## Hỏi - Đáp



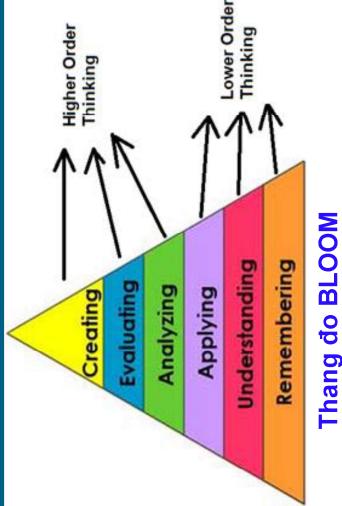
58

# TRỰC QUAN DỮ LIỆU

## Chương 6



## Mục tiêu



- Phân loại trực quan dữ liệu.
- Hiểu kiến trúc Matplotlib.
- Vận dụng phác họa cơ bản.
- Áp dụng tiền xử lý bộ dữ liệu trước khi trực quan.

2

## Nội dung



1. Giới thiệu và phân loại trực quan dữ liệu.
2. Kiến trúc Matplotlib.
3. Các biểu đồ (phác họa) cơ bản.
4. Bộ dữ liệu tình hình nhập cư đến Canada.

3



## 1. Giới thiệu và phân loại trực quan dữ liệu

- 1.1. Giới thiệu trực quan dữ liệu
- 1.2. Các phân loại trực quan dữ liệu



4

### 1.1. Giới thiệu trực quan dữ liệu

#### Data Visualization?

- Là mô tả dữ liệu dưới dạng các hình ảnh trực quan như bảng, biểu đồ, đồ thị...
- Sao cho dễ hiểu và dễ hình dung nhất.



#205



5



## 1.1. Giới thiệu trực quan dữ liệu

### Tại sao phải trực quan dữ liệu?

- Phân tích thăm dò dữ liệu.
- Truyền đạt dữ liệu rõ ràng.
- Chia sẻ, thể hiện sự chính xác của dữ liệu.
- Sử dụng chúng để hỗ trợ các đề xuất cho các bên liên quan khác.



6



## 1.1. Giới thiệu trực quan dữ liệu

Khi tạo một trực quan từ dữ liệu, khuyến nghị:

- Hiệu quả (effective)
- Hấp dẫn (attractive)
- Linh hoạt (impactive)

Bên cạnh đó, cần phải thể hiện rõ thông điệp cần truyền tải qua sản phẩm trực quan.



7



## 1.2. Các phân loại trực quan dữ liệu

### Độ phân bố – Độ trãi



...

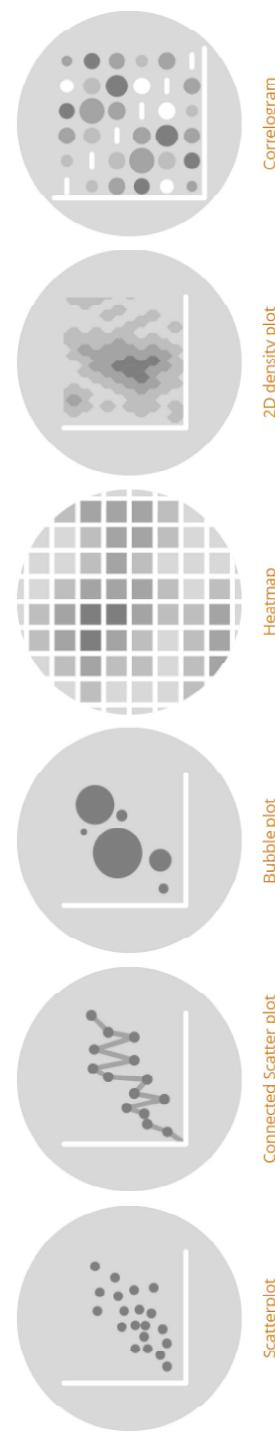
8

### Tương quan



## 1.2. Các phân loại trực quan dữ liệu

### Tương quan



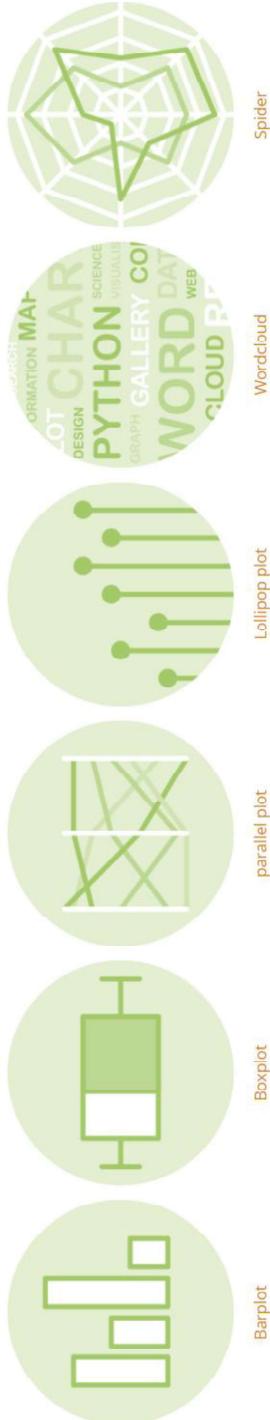
...

9



## 1.2. Các phân loại trực quan dữ liệu

### So sánh

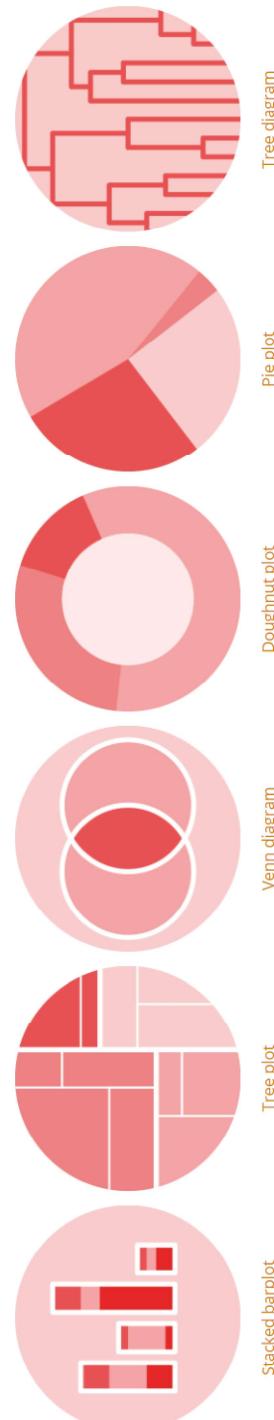


10



## 1.2. Các phân loại trực quan dữ liệu

### So sánh với tổng thể



11





## 1.2. Các phân loại trực quan dữ liệu

### Tiến triển

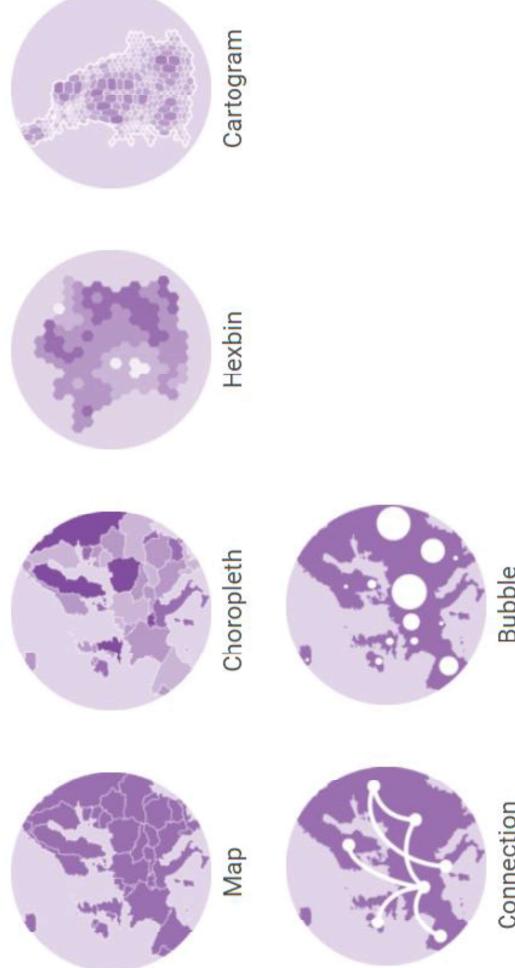


...  
12



## 1.2. Các phân loại trực quan dữ liệu

### Bản đồ



13



## 1.2. Các phân loại trực quan dữ liệu

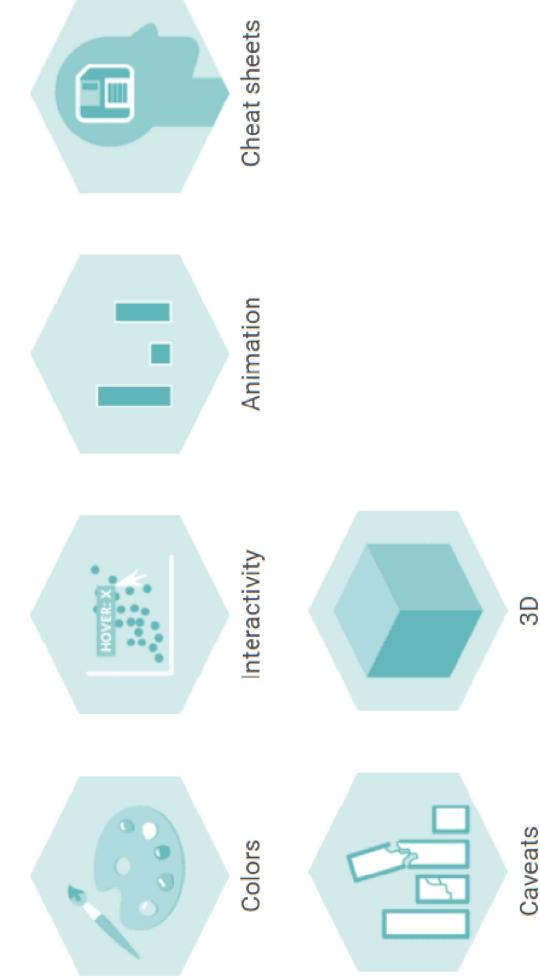
### *Luồng dịch chuyển*



14

## 1.2. Các phân loại trực quan dữ liệu

### *Thể hiện khác*



15





## 1.2. Các phân loại trực quan dữ liệu

### Tham khảo

- Phương pháp trực quan và vẽ: <https://matplotlib.org/tutorials/index.html>
- Tham khảo thêm tại: <https://python-graph-gallery.com/>



THE PYTHON  
GRAPH GALLERY

16

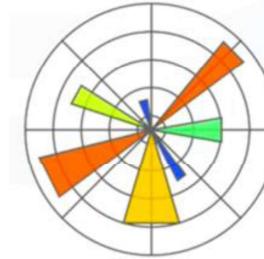


17



## 2. Matplotlib

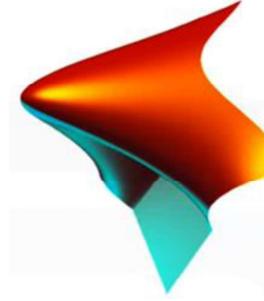
### Lịch sử



John Hunter (1968 – 2012)



EEG/ECoG Visualization Tool

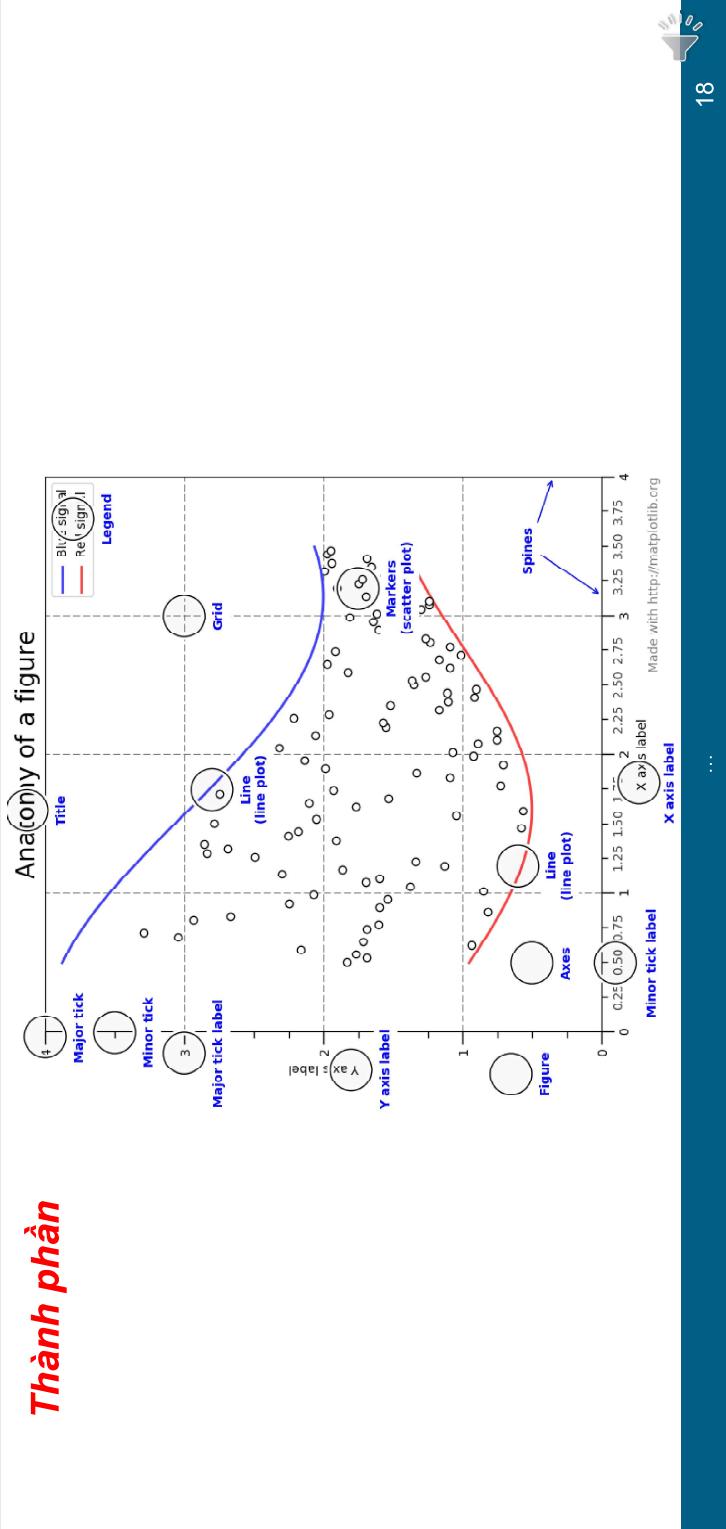


Analogous to Matlab  
scripting interface



## 2. Matplotlib

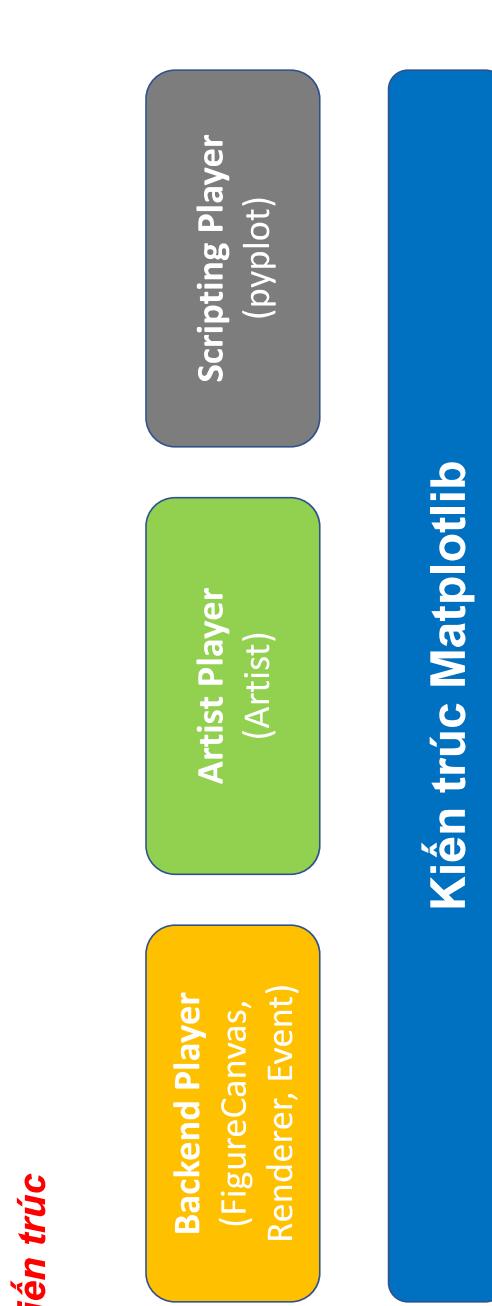
### Thành phần



### Kiến trúc



## 2. Matplotlib





## 2. Matplotlib

### Backend player

- Có ba lớp giao diện trừu tượng được tích hợp (abstract interface class)

#### 1. **FigureCanvas**: matplotlib.backend\_bases.FigureCanvas

- Định nghĩa khu vực/vùng mà hình được vẽ.

#### 2. **Renderer**: matplotlib.backend\_bases.Renderer

- Vẽ những gì trên FigureCanvas

#### 3. **Event**: matplotlib.backend\_bases.Event

- Xử lý sự kiện của người dùng, như gõ phím, click chuột...



20



## 2. Matplotlib

### Artist player

- Gồm một đối tượng chính – **Artist object**: cho phép vẽ lên nền canvas

- Có thể khởi tạo hai loại đối tượng **Artist**:

#### 1. Primitive Artist: Line2D, Rectangle, Circle, Text, AxesImage...

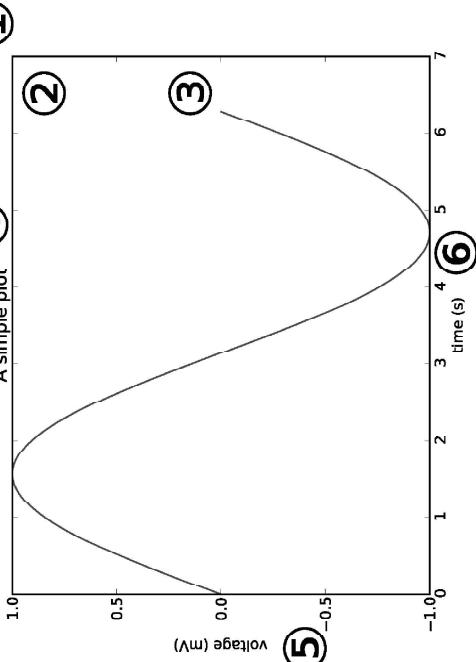
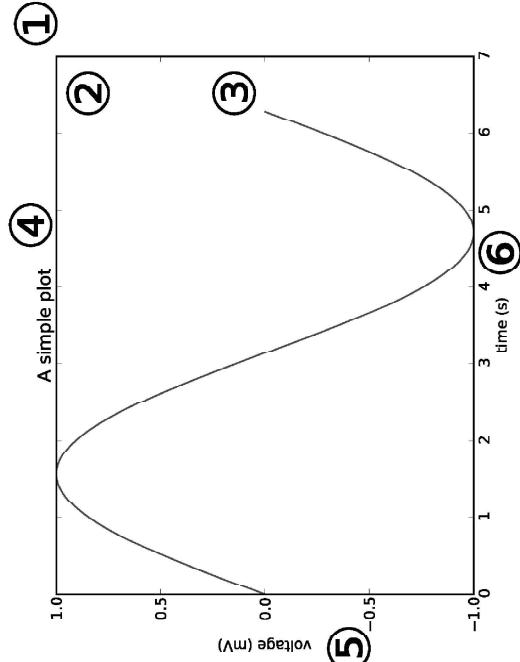
#### 2. Container Artist: Axis, Axes, Figure, và Tick

- Mỗi container artist có thể chứa các container artist khác cũng như chứa các primitive artist.



## 2. Matplotlib

### Artist Player



22



## 2. Matplotlib

### Artist Player

```
1 from matplotlib.backends.backend_agg import FigureCanvasAgg as FigureCanvas
2 from matplotlib.figure import Figure
3 fig = Figure()
4 canvas = FigureCanvas(fig)
5
6 # Tạo dữ liệu 10_000 số ngẫu nhiên
7 import numpy as np
8 x = np.random.randn(10_000)
9
10 # Tạo một Axes artist
11 # Được thêm tự động vào vùng chưa figure
12 # 111 Là quy ước của MATLAB: 1 dòng, 1 cột và dùng 1 cell đầu
13 ax = fig.add_subplot(111)
14 ax.hist(x, 100) #100 bin
15
16 ax.set_title('Normal distribution with $\mu=0, \sigma=1$')
17 fig.savefig('matplotlib_histogram.png')
```

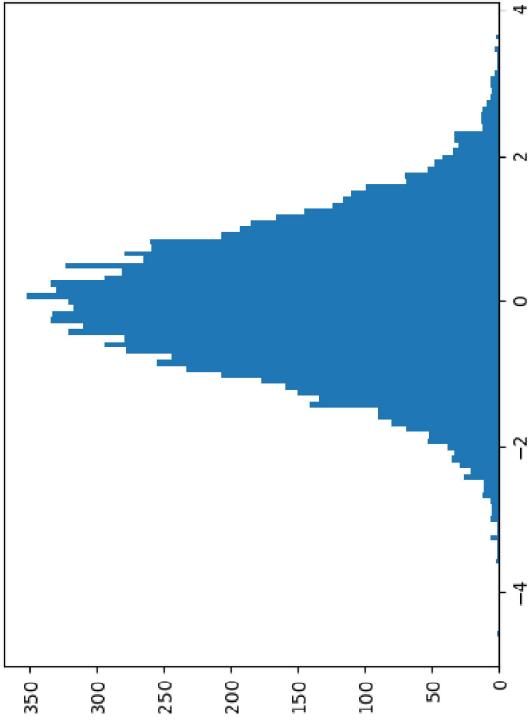
23



## 2. Matplotlib

### Artist Player

Normal distribution with  $\mu = 0, \sigma = 1$



- Kết quả

24



## 2. Matplotlib

### Scripting player

- Chủ yếu là gói **pyplot** cung cấp một giao diện dễ dùng hơn **Artist layer**.
- Tạo ra một histogram của 10\_000 giá trị ngẫu nhiên bằng cách dùng **pyplot**.

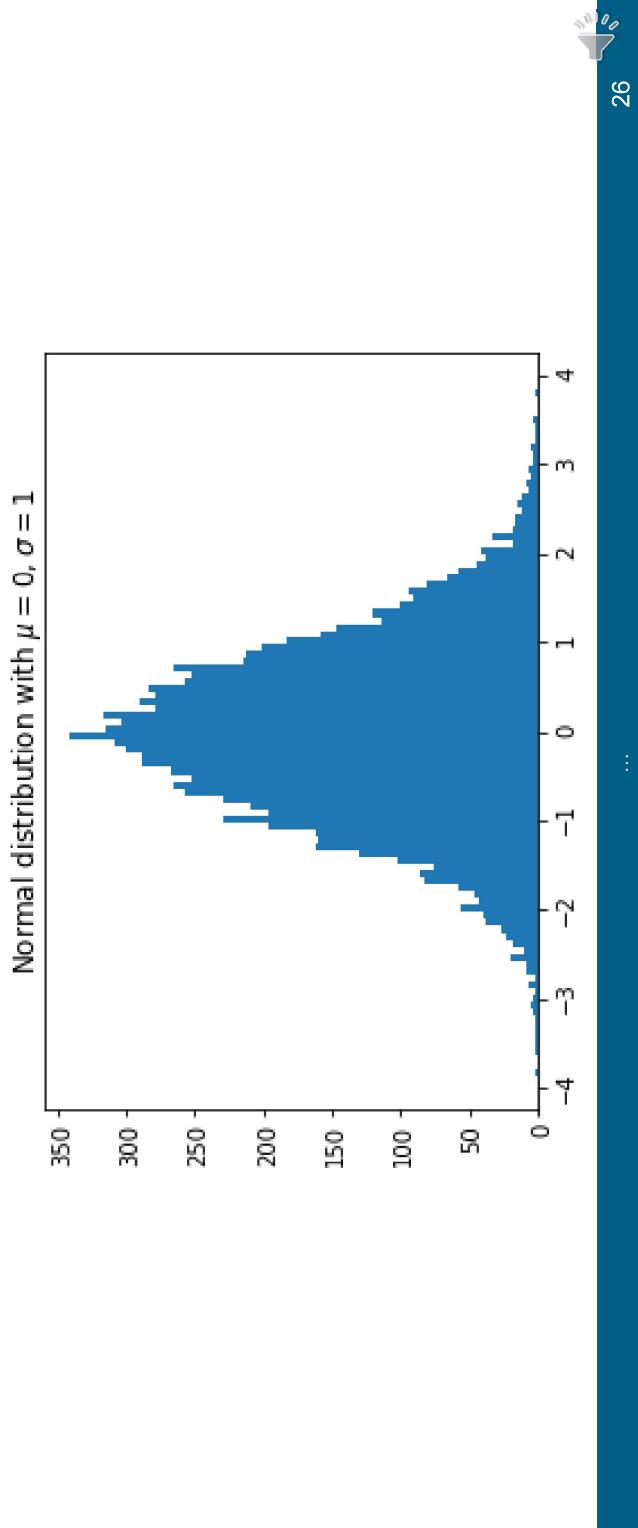
```
1 import matplotlib.pyplot as plt →
2 import numpy as np
3
4 x = np.random.randn(10000) ←
5 plt.hist(x, 100) →
6 plt.title('Normal distribution with $\mu=0, \sigma=1$')
7 plt.savefig('matplotlib_histogram.png')
8 plt.show()
```



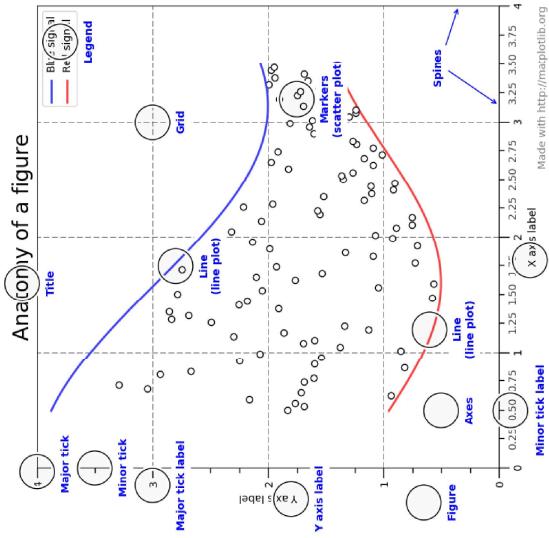
25

## 2. Matplotlib

### Scripting player



## 2. Matplotlib



Code mẫu: <https://matplotlib.org/examples/showcase/anatomy.html>



## 2. Matplotlib

Tài liệu tham khảo

<https://www.aosabook.org/en/matplotlib.html>

<https://matplotlib.org/resources/index.html>

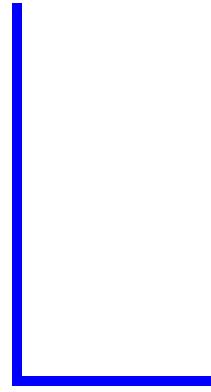


28

...



## 3. Phác họa cơ bản



- 3.1. Phương thức plot trong Matplotlib.pyplot
- 3.2. Phương thức plot trong DataFrame



29



### 3.1. Phương thức plot trong matplotlib.pyplot

```
plot(*args, scalex=True, scaley=True, data=None, **kwargs)
```

- Phác họa đường (line) hoặc điểm dữ liệu (marker)
- Giao diện sử dụng khác:

```
plot([x], y, [fmt], *, data=None, **kwargs)
```

```
plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)
```

- **x, y**: hai mảng dữ liệu cho các điểm dữ liệu
- **fmt: str => fmt = '[color][marker][line]'**. Ví dụ: 'ro', 'g-'
- Tham số trong **\*\*kwargs**: thiết lập thuộc tính cho đường



Ref.: [https://matplotlib.org/devdocs/api/\\_as\\_gen/matplotlib.plot.html](https://matplotlib.org/devdocs/api/_as_gen/matplotlib.plot.html)

30

...

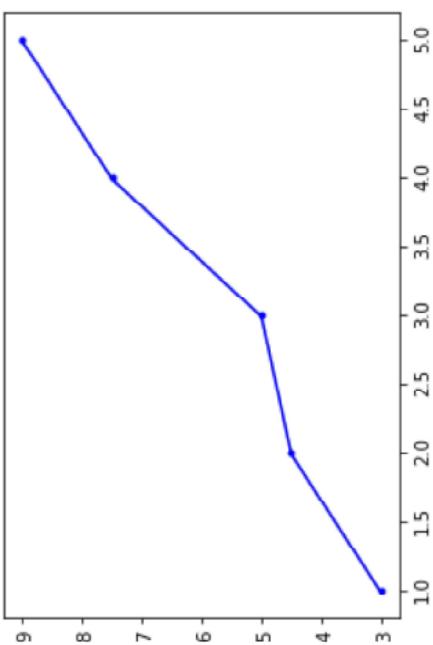


### 3.1. Phương thức plot trong matplotlib.pyplot

*Áp dụng phương thức plot() => phác họa Line2D*

```
plt.plot([1, 2, 3, 4, 5], [3, 4.5, 5, 7.5, 9], 'b.-', linewidth=4)
```

```
plt.show()
```



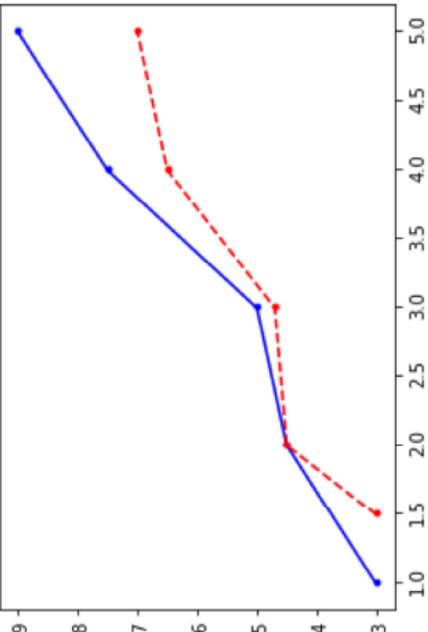
31



### 3.1. Phương thức plot trong matplotlib.pyplot

**Áp dụng phương thức `plot()` => phác họa Line2D => 2 hoặc nhiều line**

```
plt.plot([1, 2, 3, 4, 5], [3, 4.5, 5, 7.5, 9], 'b.-',
         [1.5, 2, 3, 4, 5], [3, 4.5, 4.7, 6.5, 7], 'r.--')
plt.show()
```



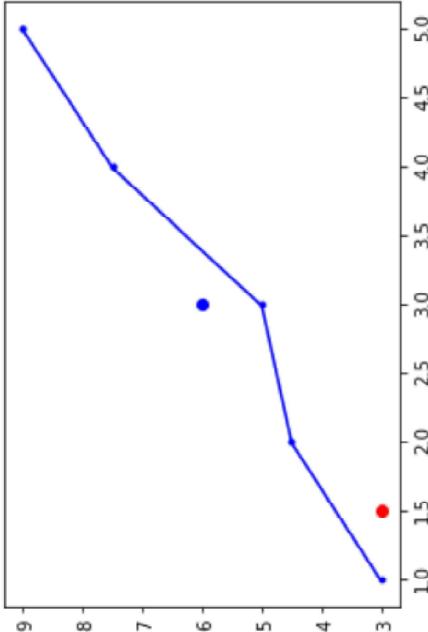
32



### 3.1. Phương thức plot trong matplotlib.pyplot

**Áp dụng phương thức `plot()` => phác họa Line2D và marker**

```
plt.plot([1, 2, 3, 4, 5], [3, 4.5, 5, 7.5, 9], 'b.-',
         [1.5, 3, 'ro', 3, 6, 'bo')
plt.show()
```



33



## 3.2. Phương thức plot trong dataframe

DataFrame.plot(\*args, \*\*kwargs)

- Phác họa đưa vào dữ liệu loại **Series** hoặc **DataFrame**.

- Đối số **kind**:

- ‘**line**’ : line plot (mặc định)
- ‘**bar**’ : vertical bar plot
- ‘**barh**’ : horizontal bar plot
- ‘**box**’ : boxplot
- ‘**density**’ : giống như ‘**kde**’
- ‘**area**’ : area plot
- ‘**scatter**’ : scatter plot (chỉ dùng cho DataFrame)
- ‘**pie**’ : pie plot
- ‘**hexbin**’ : hexbin plot (chỉ dùng cho DataFrame)



Ref.: <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.plot.html>

34

...



## 3.2. Phương thức plot trong dataframe

### Áp dụng phương thức plot()

```
1 data = {'Viet Nam': [10, 21, 18, 23, 26, 30, 33, 31, 36, 39],  
2   'Thai Lan': [15, 17, 22, 20, 24, 29, 32, 34, 30, 32]}  
3  
4 df = pd.DataFrame(data=data)  
5 df
```

|   | Viet Nam | Thai Lan |
|---|----------|----------|
| 0 | 10       | 15       |
| 1 | 21       | 17       |
| 2 | 18       | 22       |
| 3 | 23       | 20       |
| 4 | 26       | 24       |
| 5 | 30       | 29       |
| 6 | 33       | 32       |
| 7 | 31       | 34       |
| 8 | 36       | 30       |
| 9 | 39       | 32       |



35

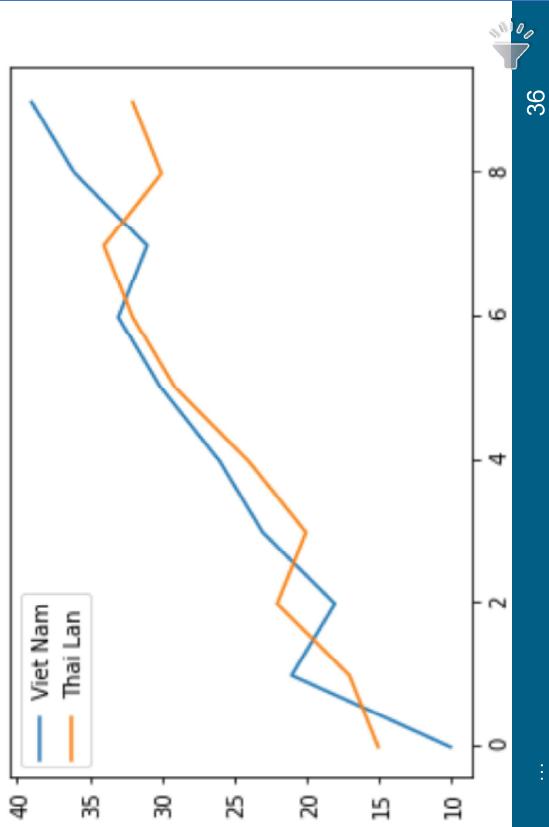


## 3.2. Phương thức plot trong dataframe

### Áp dụng phương thức `plot()`

```
1 df.plot(kind='line')
```

<AxesSubplot:>



## 4. Bộ dữ liệu tình hình nhập cư đến Canada

### Giới thiệu bộ dữ liệu

- Bộ phân dân số trực thuộc Liên hợp quốc đã tổng hợp dữ liệu dân số liên quan đến nhiều quốc gia.
- Đối với mỗi quốc gia, dữ liệu hàng năm về dòng người di cư được thống kê cập nhật.
- Trong phần minh họa trực quan dữ liệu, sử dụng chủ yếu dữ liệu của Liên Hợp Quốc về nhập cư vào Canada.

- Đường dẫn: <https://github.com/datasetshub/ds105/blob/master/Canada.xlsx>



## 4. Bộ dữ liệu tình hình nhập cư đến Canada

### Giới thiệu bộ dữ liệu

<https://www.un.org/en/development/desa/population/migration/data/empirical2/migrationflows.asp>

The screenshot shows a table titled "International Migration Flows to and from Selected Countries: The 2015 Revision". The table has columns for Year (1980-1988), Region Name, and various countries/regions. The table is part of a larger UN document with a header and footer.

|    | A              | B          | C                | D                | E          | F    | G    | H    | I    | J    | K    | L    | M    |      |
|----|----------------|------------|------------------|------------------|------------|------|------|------|------|------|------|------|------|------|
| 1  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 2  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 3  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 4  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 5  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 6  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 7  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 8  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 9  |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 10 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 11 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 12 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 13 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 14 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 15 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 16 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 17 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 18 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 19 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 20 |                |            |                  |                  |            |      |      |      |      |      |      |      |      |      |
| 21 | Classification | Type       | Coverage         | AreaName         | RegionName | 1980 | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | 1987 | 1988 |
| 22 | Immigrants     | Citizens   | Northern America | Northern America | 1471       | 1641 | 1426 | 1094 | 1187 | 1134 | 1454 | 2734 | 34   |      |
| 23 | Immigrants     | Foreigners | Eastern Africa   | Eastern Africa   | 33         | 44   | 32   | 42   | 30   | 61   | 100  |      |      |      |
| 24 | Immigrants     | Foreigners | Middle Africa    | Northern Africa  | 1100       | 1268 | 1348 | 936  | 842  | 840  | 1115 | 1864 | 22   |      |
| 25 | Immigrants     | Foreigners | South Africa     | Southern Africa  | 1041       | 1126 | 791  | 387  | 327  | 725  | 1480 | 13   |      |      |
| 26 | Immigrants     | Foreigners | Western Africa   | Western Africa   | 306        | 301  | 210  | 222  | 271  | 319  | 427  | 1316 | 6    |      |
| 27 | Immigrants     | Foreigners | Africa Total     | Africa Total     | 3951       | 4363 | 3819 | 2671 | 2639 | 2650 | 3782 | 7494 | 75   |      |
| 28 | Immigrants     | Foreigners | Central Asia     | Central Asia     | 6836       | 8835 | 5481 | 3254 | 2624 | 2979 | 3416 | 5403 | 56   |      |
| 29 | Immigrants     | Foreigners | Eastern Asia     | Eastern Asia     | ...        | ...  | ...  | ...  | ...  | ...  | ...  | ...  | ...  |      |
| 30 | Immigrants     | Foreigners | ...              | ...              | ...        | ...  | ...  | ...  | ...  | ...  | ...  | ...  | ...  | 38   |



## 4. Bộ dữ liệu tình hình nhập cư đến Canada

### Đọc bộ dữ liệu

```
1 import pandas as pd
2 import numpy as np
```

```
1 pip install xlrd #Nếu chưa có thì install
```

```
1 import pandas as pd
2 df_can = pd.read_excel('dataset/Canada.xlsx',
3 sheet_name='Canada by Citizenship',
4 skiprows=range(20),
5 skipfooter=2)
6 df_can.shape
7
```

(195, 51)



## 4. Bộ dữ liệu tình hình nhập cư đến Canada

### Kết quả

```
1 df_can.head()  
2 df_can.tail()
```

| Type | Coverage   | OdName     | AREA           | AreaName | REG     | RegName | DEV                | DevName | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | 1987 | 1988 | 1989 |
|------|------------|------------|----------------|----------|---------|---------|--------------------|---------|--------------------|------|------|------|------|------|------|------|------|------|
| 0    | Immigrants | Foreigners | Afghanistan    | 935      | Asia    | 5501    | Southern Asia      | 902     | Developing regions | 16   | 39   | 47   | 71   | 340  | 496  | 741  | 828  | 1076 |
| 1    | Immigrants | Foreigners | Albania        | 908      | Europe  | 925     | Southern Europe    | 901     | Developed regions  | 1    | 0    | 0    | 0    | 0    | 1    | 2    | 2    | 3    |
| 2    | Immigrants | Foreigners | Algeria        | 903      | Africa  | 912     | Northern Africa    | 902     | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69   | 132  | 242  |
| 3    | Immigrants | Foreigners | American Samoa | 909      | Oceania | 957     | Polynesia          | 902     | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 1    | 0    | 1    |
| 4    | Immigrants | Foreigners | Andorra        | 908      | Europe  | 925     | Southern Europe    | 901     | Developed regions  | 0    | 0    | 0    | 0    | 0    | 2    | 0    | 0    | 0    |
| ...  |            |            |                |          |         |         |                    |         |                    |      |      |      |      |      |      |      |      |      |
| 190  | Immigrants | Foreigners | Viet Nam       | 935      | Asia    | 920     | South-Eastern Asia | 902     | Developing regions | 1191 | 1829 | 2162 | 3404 | 7583 | 5907 | 2741 | 1406 | 1411 |
| 191  | Immigrants | Foreigners | Western Sahara | 903      | Africa  | 912     | Northern Africa    | 902     | Developing regions | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 192  | Immigrants | Foreigners | Yemen          | 935      | Asia    | 922     | Western Asia       | 902     | Developing regions | 1    | 2    | 1    | 6    | 0    | 18   | 7    | 12   | 18   |
| 193  | Immigrants | Foreigners | Zambia         | 903      | Africa  | 910     | Eastern Africa     | 902     | Developing regions | 11   | 17   | 11   | 7    | 16   | 9    | 15   | 23   | 44   |
| 194  | Immigrants | Foreigners | Zimbabwe       | 903      | Africa  | 910     | Eastern Africa     | 902     | Developing regions | 72   | 114  | 102  | 44   | 32   | 29   | 43   | 68   | 99   |
| ...  |            |            |                |          |         |         |                    |         |                    |      |      |      |      |      |      |      |      |      |

```
df_can.head()
```

```
df_can.tail()
```



Thank you



## Hỏi - Đáp



43

# CÔNG CỤ TRỰC QUAN DỮ LIỆU

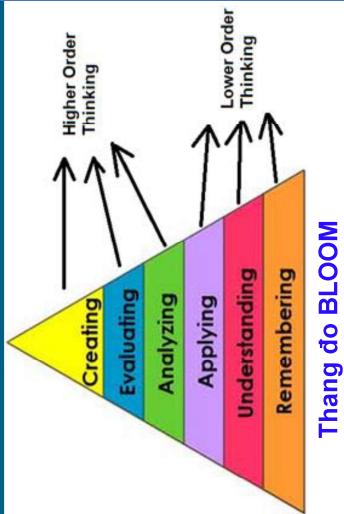
## Chương 7



## Mục tiêu



- Hiểu và vận dụng line plot
- Hiểu và vận dụng area plot
- Hiểu và vận dụng histogram
- Hiểu và vận dụng bar chart
- Hiểu và vận dụng pie chart
- Hiểu và vận dụng box plot
- Hiểu và vận dụng scatter plot
- Hiểu và vận dụng waffle chart
- Hiểu và vận dụng word cloud



2

## Nội dung



1. Các bước trực quan dữ liệu
2. Line plot
3. Area plot
4. Histogram
5. Bar chart
6. Pie chart
7. Box plot
8. Scatter plot
9. Waffle chart
10. Word cloud

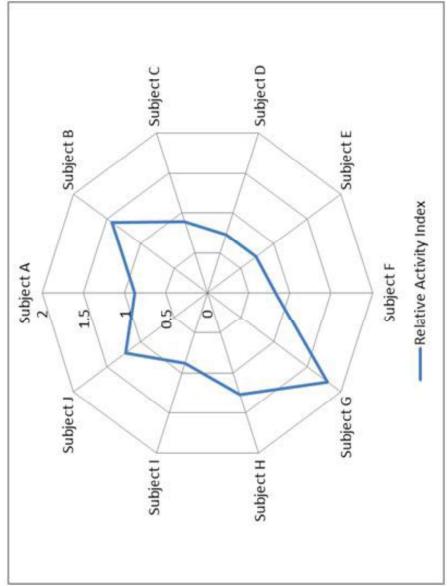


3



## 1. Các bước trực quan dữ liệu

- Bước 1** - Làm rõ câu hỏi, hoặc vấn đề cần thể hiện từ dữ liệu.
- Bước 2** - Hiểu dữ liệu đang có, và bắt đầu với những phác họa cơ bản
- Bước 3** - Xác định thông điệp cần truyền tải của trực quan được xây dựng, đưa ra các thông tin cốt lõi nhất.
- Bước 4** - Chọn họa đồ thích hợp.
- Bước 5** - Dùng màu, kích thước, tỉ lệ, số chiêu và nhấn thông tin tập trung vào thông điệp chính cần truyền tải.



Ref.: <https://www.elsevier.com/connect/a-5-step-guide-to-data-visualization>

4



## 1. Các bước trực quan dữ liệu

### Bộ dữ liệu

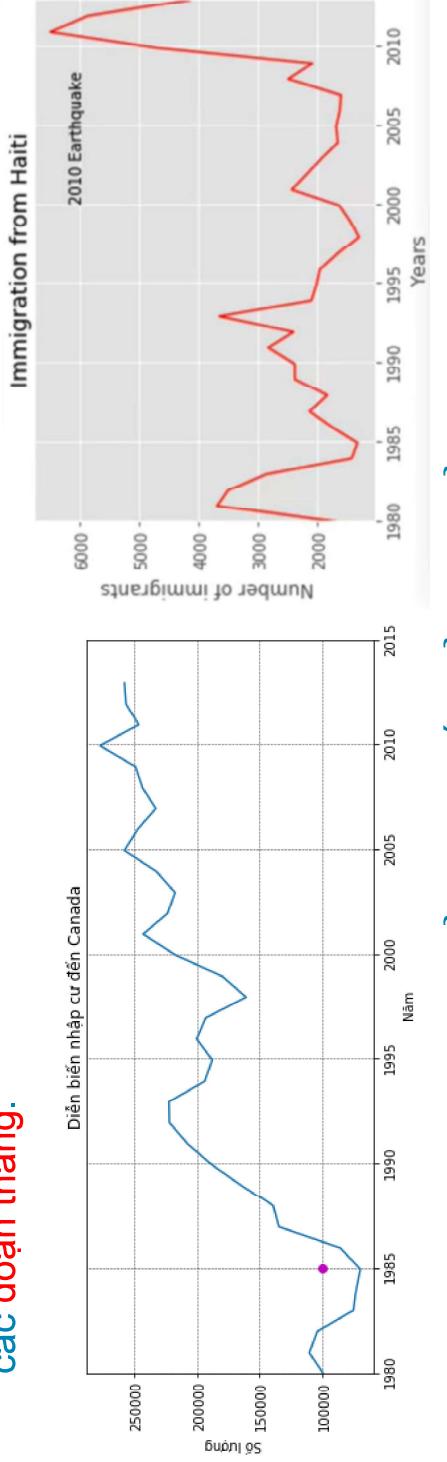
| Country | Continent           | Region                          | DevName            | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|--------------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia      | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe    | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa    | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia          | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4       | Andorra             | Europe                          | Southern Europe    | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 1    | 1    | 1    |
| 5       | Angola              | Africa                          | Middle Africa      | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Developing regions | 0                  | 0    | 0    | 0    | 42   | 52   | ... | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |      |
| 7       | Argentina           | South America                   | Developing regions | 368                | 426  | 626  | 241  | 237  | 196  | ... | 1591 | 1153 | 847  | 620  | 540  | 467  | 459  | 278  | 263  | 282  |      |

5



## 2. Line plot

- Line plot (Biểu đồ đường) là một loại biểu đồ hiển thị thông tin dưới dạng **một loạt các điểm dữ liệu** (gọi là các “**marker**”) và được kết nối nhau bởi **các đoạn thẳng**.



- Dùng trong trường hợp thể hiện **sự tiến triển/phát triển**.

6



## 2. Line plot

### Áp dụng vào bộ dữ liệu thống kê tình hình di cư của các nước đến Canada

- Câu hỏi: Thể hiện số lượng di cư (sự biến động, diễn biến) của Haiti đến Canada qua từng năm (*tức từ năm 1980 đến 2013*)?

#### Thực hiện:

- Bước #1: Thể hiện diễn biến di cư của Haiti đến Canada từ 1980 đến 2013
- Bước #2: Hiểu dữ liệu => xác định số dân di cư các năm, chọn các cột...
- Bước #3: Làm rõ biến động/diễn biến dân di cư
- Bước #4: Chọn họa đồ => line plot... (còn loại nào nữa không?)
- Bước #5: Hiệu chỉnh Kích thước hình, màu sắc, biểu tượng...



7



## 2. Line plot

### Bộ dữ liệu

- Tóm tắt bộ dữ liệu để hiểu dữ liệu, cần chỉ định rõ dữ liệu dùng để trực quan.

| Country | Continent           | Region                          | DevName            | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|--------------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia      | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe    | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa    | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia          | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4       | Andorra             | Europe                          | Southern Europe    | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 1    |
| 5       | Angola              | Africa                          | Middle Africa      | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Developing regions | 0                  | 0    | 0    | 0    | 42   | 52   | ... | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |      |
|         |                     |                                 |                    | ...                |      |      |      |      |      |     |      |      |      |      |      |      |      |      |      |      |      |
|         |                     |                                 |                    |                    |      |      |      |      |      |     |      |      |      |      |      |      |      |      | 8    |      |      |



## 2. Line plot

### Bộ dữ liệu

- Xử lý cột “Country” sẽ là index

| Continent      | Region  | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|----------------|---------|-----------------|--------------------|------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| Afghanistan    | Asia    | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340  | 496 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| Albania        | Europe  | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0    | 1   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| Algeria        | Africa  | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| American Samoa | Oceania | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| Andorra        | Europe  | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0    | 2   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 1    | 1    | 1    |

Thông điệp cần muốn diễn đạt: thể hiện số lượng di cư (sự biến động, diễn biến) của **Haiti** đến Canada qua từng năm? => **dùng biểu đồ đường**.

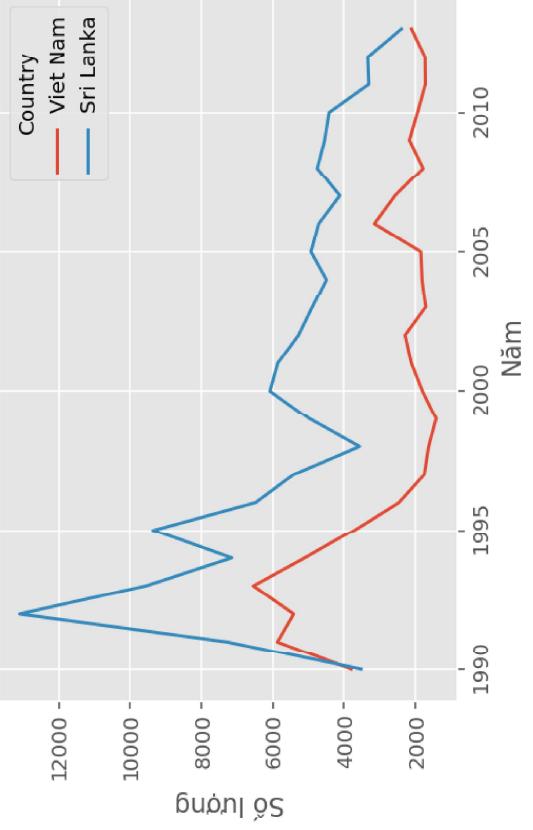




## 2. Line plot

Bài tập

So sánh Việt Nam và Sri Lanka nhập cư đến Canada



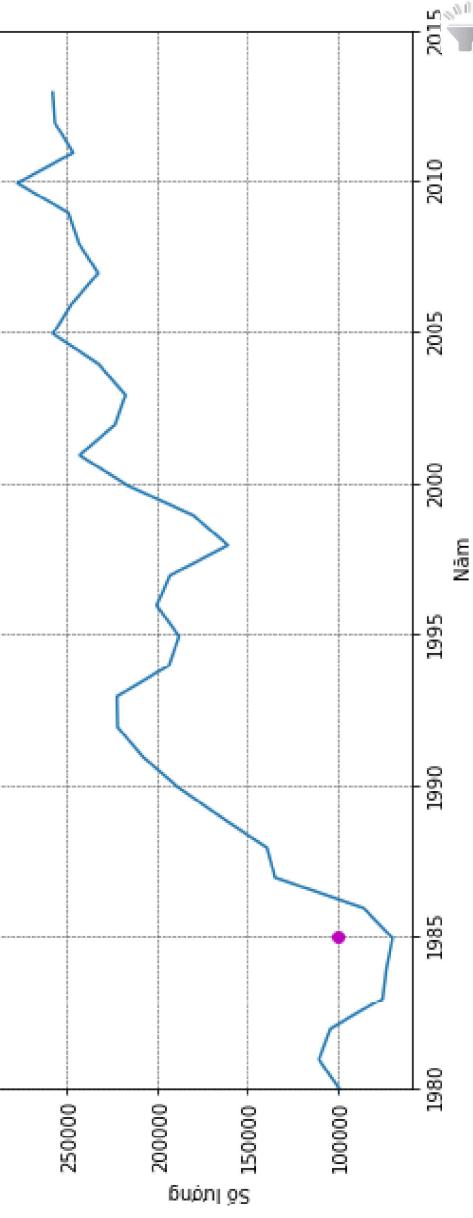
12



## 2. Line plot

Bài tập

Diễn biến nhập cư đến Canada



13



### 3. Area plot

#### Giới thiệu

- Area plot còn được gọi là **area chart** hoặc **area graph**.
- Thường được sử dụng để **biểu diễn tổng số tích lũy (cumulated totals)** bằng cách sử dụng **số (number)** hoặc **tỷ lệ phần trăm (percentage)** theo thời gian.
- Dựa trên **line plot**, hay là dạng mở rộng từ **line plot**.



14



### 3. Area plot

#### Câu hỏi

- Mô tả xu hướng nhập cảng của năm quốc gia có tỷ lệ nhập cảng cao nhất vào Canada từ năm 1980 đến 2013?

| Country | Continent      | Region  | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|----------------|---------|-----------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan    | Asia    | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1758 | 2203 | 2635 | 2004 |      |
| 1       | Albania        | Europe  | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria        | Africa  | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa | Oceania | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    |      |
| 4       | Andorra        | Europe  | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    |      |
| 5       | Angola         | Africa  | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |

Dữ liệu ban đầu

15



### 3. Area plot

#### Xử lý

- Chuyển cột “Country” làm index.
- Tính tổng di cư của các nước => cột “Total”

| Continent      | Region  | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | 1986 | ... | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | Total |
|----------------|---------|-----------------|--------------------|------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|-------|
| Country        |         |                 |                    |      |      |      |      |      |      |     |      |      |      |      |      |      |      |      |      |       |
| Afghanistan    | Asia    | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340  | 496 | ...  | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004  |
| Albania        | Europe  | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0    | 1   | ...  | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603   |
| Algeria        | Africa  | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44   | 69  | ...  | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331  |
| American Samoa | Oceania | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 6     |
| Andorra        | Europe  | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0    | 2   | ...  | 0    | 1    | 1    | 0    | 0    | 0    | 0    | 1    | 15    |
| Angola         | Africa  | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3    | 5   | ...  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45    |
|                |         |                 |                    |      |      |      |      |      |      |     | ...  |      |      |      |      |      |      |      |      |       |
|                |         |                 |                    |      |      |      |      |      |      |     |      |      |      |      |      |      |      |      | 16   |       |



### 3. Area plot

#### Xử lý

- Xử lý chọn ra top 5 quốc gia có số lượng di cư đến Canada nhiều nhất.

```

1 df_can.sort_values(['Total'], ascending=False, axis=0, inplace=True) ↓
2 df_top5 = df_can.head() ↓
3 df_top5 = df_top5.loc[:, 1980:2013].transpose() ↓
4
1 df_top5.head()

```

| Country | India | China | United Kingdom of Great Britain and Northern Ireland | Ireland | Philippines | Pakistan |
|---------|-------|-------|------------------------------------------------------|---------|-------------|----------|
| 1980    | 8880  | 5123  |                                                      |         | 22045       | 6051     |
| 1981    | 8670  | 6682  |                                                      |         | 24796       | 5921     |
| 1982    | 8147  | 3308  |                                                      |         | 20620       | 5249     |
| 1983    | 7338  | 1863  |                                                      |         | 10015       | 4562     |
| 1984    | 5704  | 1527  | Dữ liệu chuẩn bị plot                                |         | 10170       | 3801     |
|         |       |       |                                                      |         |             | 668      |



### 3. Area plot

#### Vẽ Area plot

```
1 df_top5.plot(kind='area', stacked=False, figsize=(20, 10),)  
2  
3 plt.title('Top 5 quốc gia có lượng di lớn nhất đến Canada')  
4 plt.ylabel('Số lượng di cư')  
5 plt.xlabel('Năm')  
6  
7 plt.show()
```



18

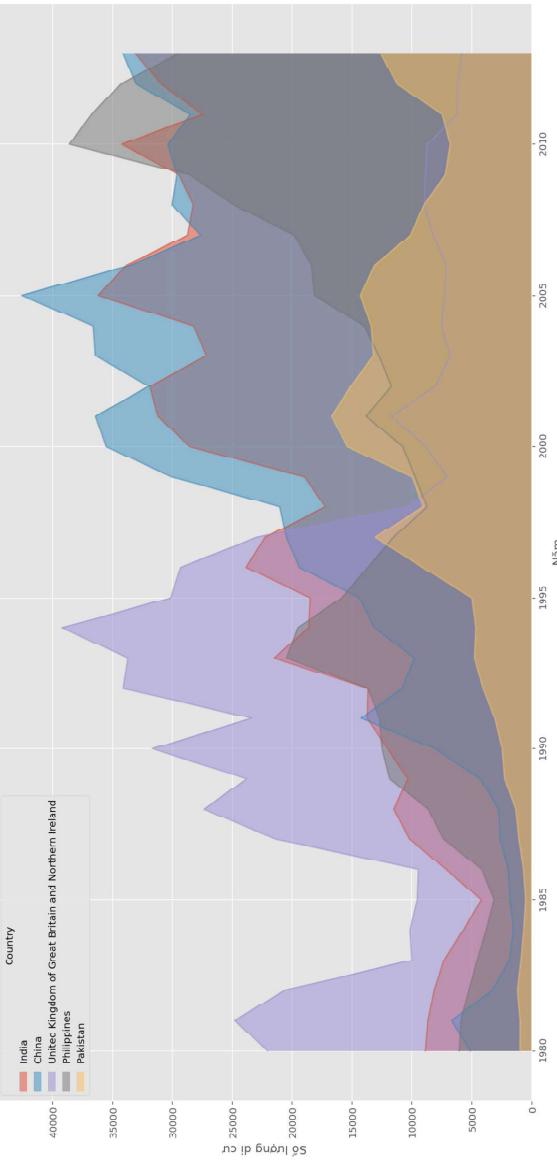
...



### 3. Area plot

#### Vẽ Area plot

Top 5 quốc gia có lượng di lớn nhất đến Canada



19

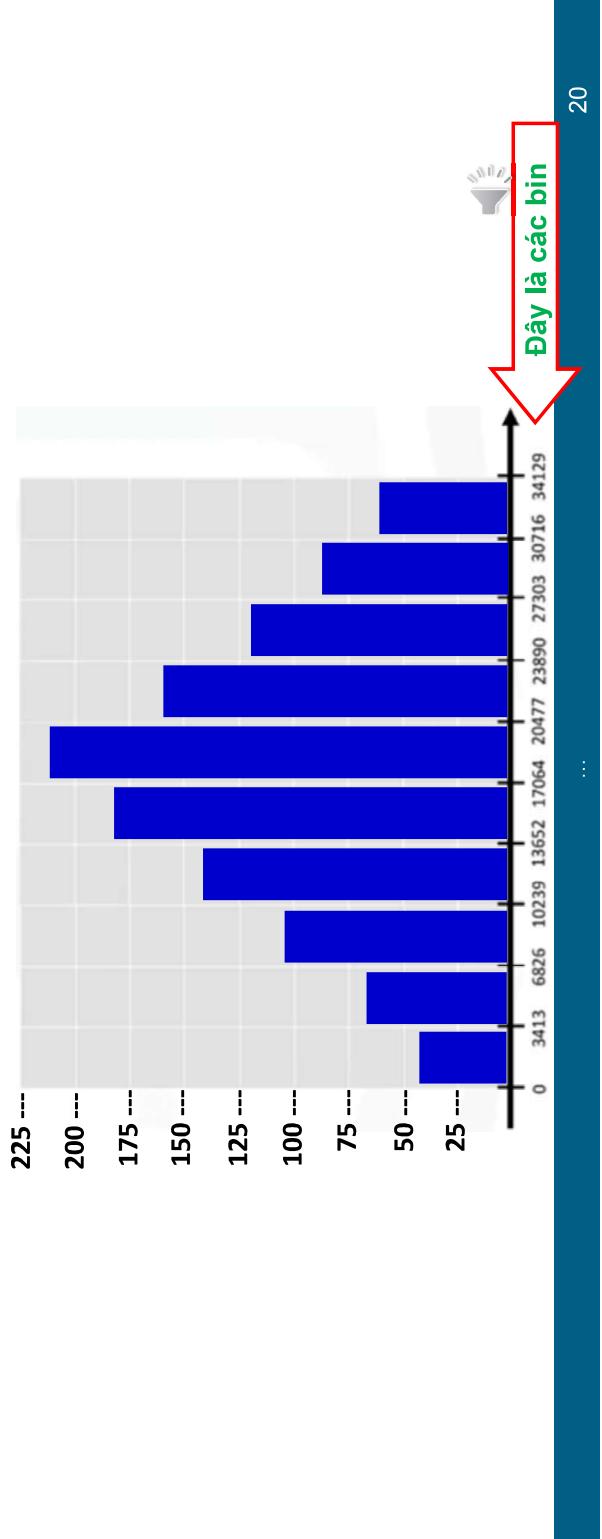


## 4. Histogram

### Giới thiệu

(frequency distribution)

- Histogram là một biểu đồ biểu diễn phân佈 tần suất của một biến.



## 4. Histogram

### Câu hỏi

- Thể hiện tần suất số quốc gia nhập cư đến Canada trong năm 2010.

| Country | Continent           | Region                          | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|-----------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    |      |
| 4       | Andorra             | Europe                          | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 1    | 1    |      |
| 5       | Angola              | Africa                          | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 296  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Caribbean       | Developing regions | 0    | 0    | 0    | 0    | 42   | 52  | ...  | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |

## 4. Histogram

### Chuẩn bị dữ liệu

```
1 df_can[2010]
```

```
Country          34235
India           30391
China           8724
United Kingdom of Great Britain and Northern Ireland 38617
Philippines      6811
Pakistan         ...
...
San Marino        1
New Caledonia      0
Marshall Islands     0
Western Sahara       0
Palau             0
Name: 2010, Length: 195, dtype: int64
```

22



## 4. Histogram

### Chuẩn bị dữ liệu



```
1 count, bin_edges = np.histogram(df_can[2010])
```

```
1 count
```

```
array([176, 14, 2, 0, 0, 0, 1, 1], dtype=int64)
```

```
1 bin_edges
```

```
array([ 0., 3861.7, 7723.4, 11585.1, 15446.8, 19308.5, 23170.2,
27031.9, 30893.6, 34755.3, 38617. ])
```



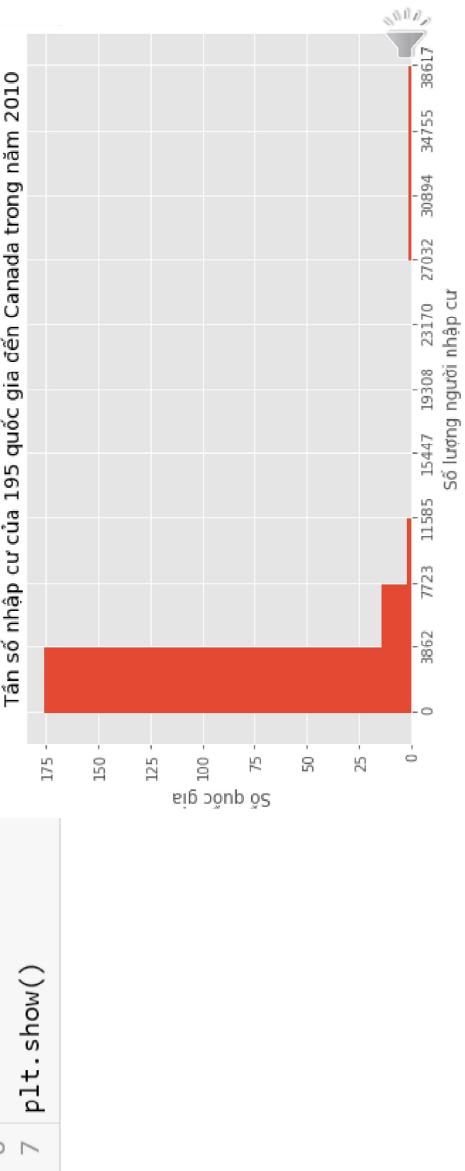
23



## 4. Histogram

Vẽ

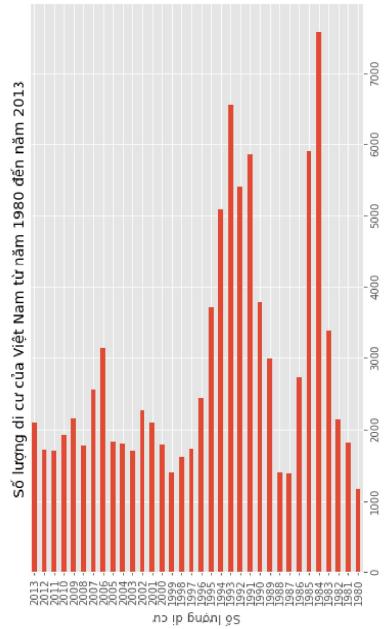
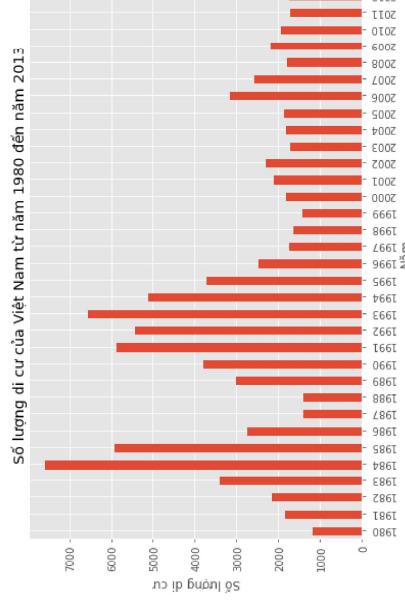
```
1 df_can[2010].plot(kind='hist', figsize=(9, 5), xticks=bin_edges)
2
3 plt.title('Tần số nhập cư của 195 quốc gia đến Canada trong năm 2010')
4 plt.ylabel('Số quốc gia')
5 plt.xlabel('Số lượng người nhập cư')
6
7 plt.show()
```



## 5. Bar chart

Giới thiệu

- Không giống như histogram, bar chart thường được sử dụng để so sánh các giá trị của biến tại một thời điểm nhất định.





## 5. Bar chart

### Câu hỏi

- Mô tả số lượng người nhập cư từ Việt Nam đến Canada từ năm 1980 đến 2013

| Country | Continent           | Region                          | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|-----------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    |      |
| 4       | Andorra             | Europe                          | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 1    | 1    |      |
| 5       | Angola              | Africa                          | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Caribbean       | Developing regions | 0    | 0    | 0    | 0    | 42   | 52  | ...  | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |
|         |                     |                                 |                 |                    |      |      |      |      |      |     | ...  |      |      |      |      |      |      |      |      |      |      |
|         |                     |                                 |                 |                    |      |      |      |      |      |     |      |      |      |      |      |      |      |      | 26   |      |      |



## 5. Bar chart

### Chuẩn bị dữ liệu

```
1 df_viet = df_can.loc['Viet Nam', 1980:2013]
2 df_viet
```

|      |      |
|------|------|
| 1997 | 1752 |
| 1998 | 1631 |
| 1999 | 1419 |
| 2000 | 1803 |
| 2001 | 2117 |
| 2002 | 2291 |
| 2003 | 1713 |
| 2004 | 1816 |
| 2005 | 1852 |
| 2006 | 3153 |
| 2007 | 2574 |
| 2008 | 1784 |
| 2009 | 2171 |
| 2010 | 1942 |
| 2011 | 1723 |
| 2012 | 1731 |
| 2013 | 2112 |

Name: Viet Nam, dtype: object



## 5. Bar chart

Vẽ Bar chart

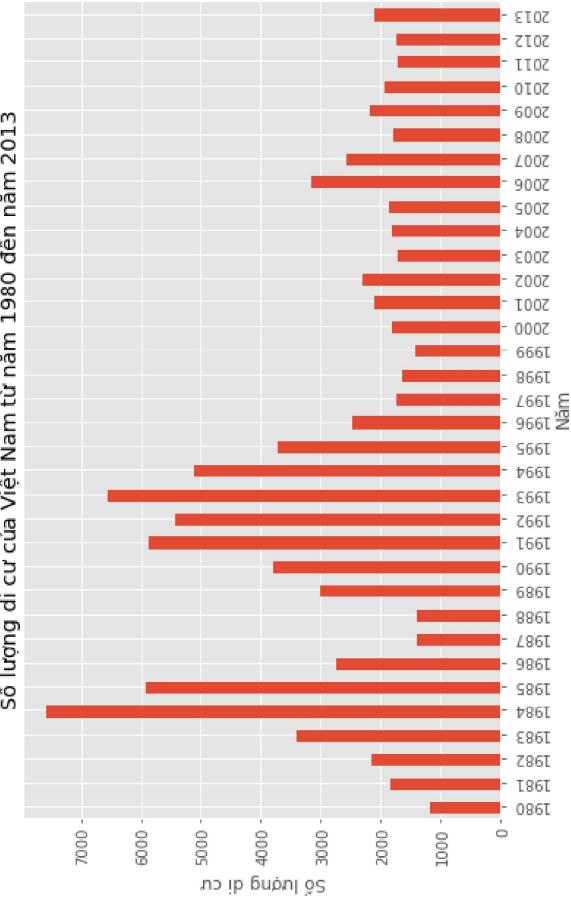
```
1 df_viet.plot(kind='bar', figsize=(10, 6))
2 plt.xlabel('Năm')
3 plt.ylabel('Số lượng di cư')
4 plt.title('Số lượng di cư của Việt Nam từ năm 1980 đến năm 2013')
5
6 plt.show()
```

28



## 5. Bar chart

Vẽ Bar chart



29



## 5. Bar chart

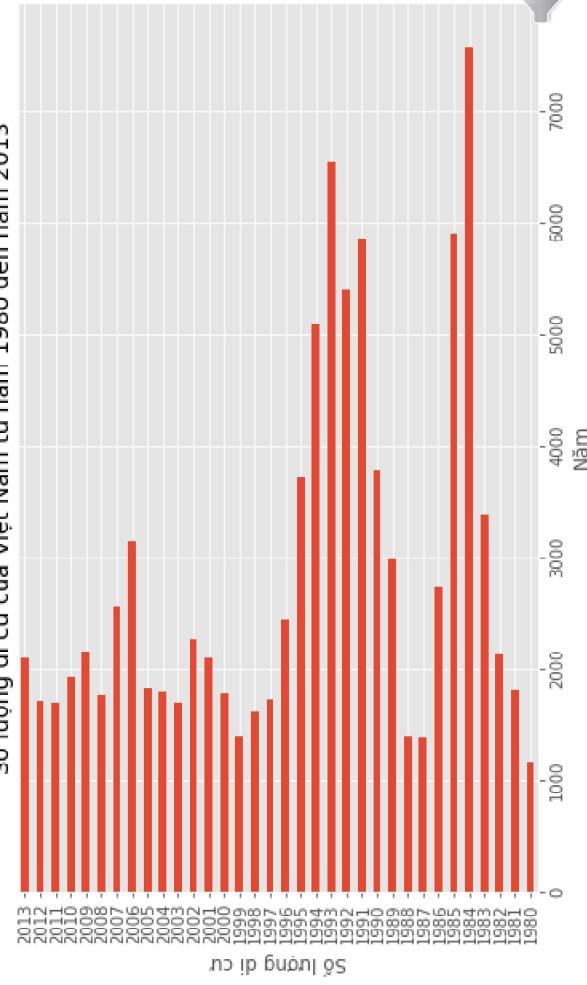
Vẽ Bar chart

```
1 df_viet.plot(kind='barh', figsize=(10, 6))
2
3 plt.xlabel('Năm')
4 plt.ylabel('Số lượng di cư')
5 plt.title('Số lượng di cư của Việt Nam từ năm 1980 đến năm 2013')
6
7 plt.show()
```



## 5. Bar chart

Vẽ Bar chart

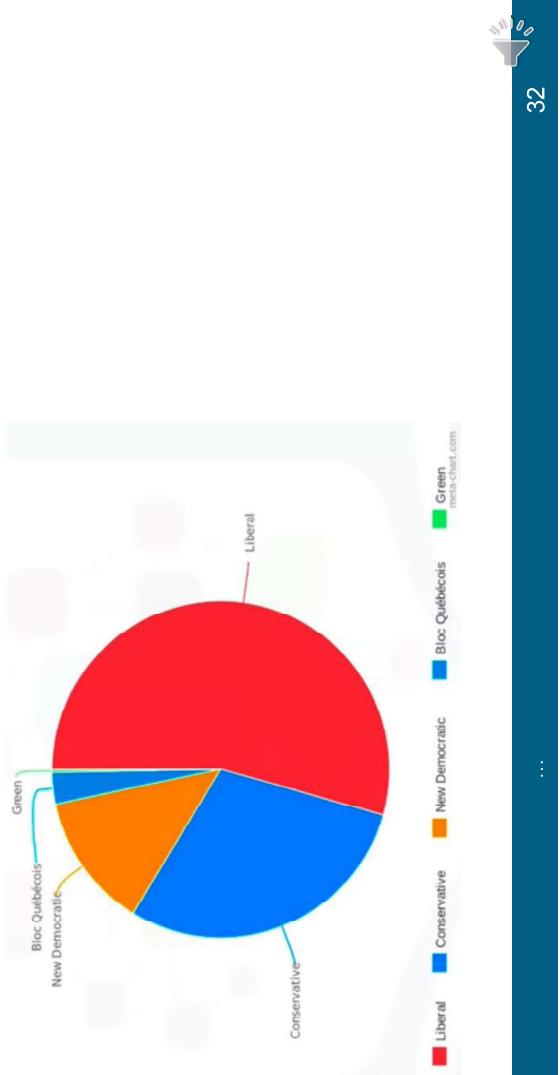




## 6. Pie chart

### Pie Chart

- Pie chart là một biểu đồ thống kê tròn được chia thành các lát (slices) để minh họa tỉ lệ.



32



## 6. Pie chart

### Câu hỏi

- Thể hiện tỉ lệ nhập cư của mỗi lục địa vào Canada từ năm 1980 đến 2013.

| Country | Continent           | Region                          | DevName            | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|--------------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia      | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe    | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa    | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia          | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    |      |
| 4       | Andorra             | Europe                          | Southern Europe    | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 0    | 0    | 1    | 0    | 0    | 1    | 1    |      |
| 5       | Angola              | Africa                          | Middle Africa      | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Developing regions | 0                  | 0    | 0    | 0    | 42   | 52   | ... | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |      |

33



## 6. Pie chart

### Chuẩn bị dữ liệu

```
1 df_continents = df_can.groupby('Continent', axis=0).sum()  
2 df_continents
```

| Continent                       | 1980  | 1981  | 1982  | 1983  | 1984  | 1985  | 1986  | 1987  | 1988  | 1989  | 1989 ... | 2005   | 2006   | 2007   | 2008   | 2009   | 2010   | 2011   | 2012   | 2013   | Total   |
|---------------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|
| Africa                          | 3951  | 4363  | 3819  | 2671  | 2639  | 2650  | 3782  | 7494  | 7552  | 9894  | ...      | 27523  | 29188  | 28284  | 29890  | 34534  | 40892  | 35441  | 38083  | 338543 | 618948  |
| Asia                            | 31025 | 34314 | 30214 | 24696 | 27274 | 23650 | 28739 | 43203 | 47454 | 60256 | ...      | 159253 | 149054 | 133459 | 139894 | 141434 | 163845 | 146694 | 152218 | 155075 | 3317794 |
| Europe                          | 39760 | 44802 | 42720 | 24638 | 22237 | 20844 | 24370 | 46698 | 54726 | 60893 | ...      | 35955  | 33053  | 33495  | 34692  | 35078  | 33425  | 26778  | 29177  | 28691  | 1410947 |
| Latin America and the Caribbean | 13081 | 15215 | 16769 | 15427 | 13678 | 15171 | 21179 | 28471 | 21924 | 25060 | ...      | 24747  | 24676  | 26011  | 26547  | 26867  | 28818  | 27856  | 27173  | 24950  | 765148  |
| Northern America                | 9378  | 10030 | 9074  | 7100  | 6661  | 6543  | 7074  | 7705  | 6469  | 6790  | ...      | 8394   | 9613   | 9463   | 10190  | 8995   | 8142   | 7677   | 7892   | 3503   | 241142  |
| Oceania                         | 1942  | 1839  | 1675  | 1018  | 878   | 920   | 904   | 1200  | 1181  | 1539  | ...      | 1585   | 1473   | 1693   | 1834   | 1860   | 1834   | 1548   | 1679   | 1775   | 55174   |



## 6. Pie chart

### Vẽ Pie chart

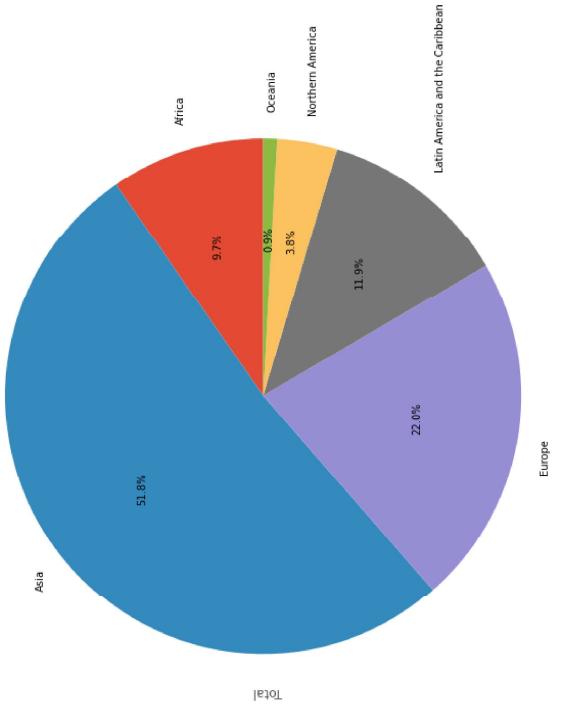
```
1 df_continents['Total'].plot(kind='pie',  
2                             figsize=(10, 10),  
3                             autopct='%1.1f%%',  
4                             )  
5  
6 plt.title('Các lục địa đến Canada giai đoạn 1980 - 2013')  
7 plt.axis('equal')  
8 plt.show()
```



## 6. Pie chart

### Vẽ Pie chart

Các lục địa đến Canada giai đoạn 1980 - 2013



Tỉ lệ nhập cư của mỗi lục địa vào Canada từ năm 1980 đến 2013.

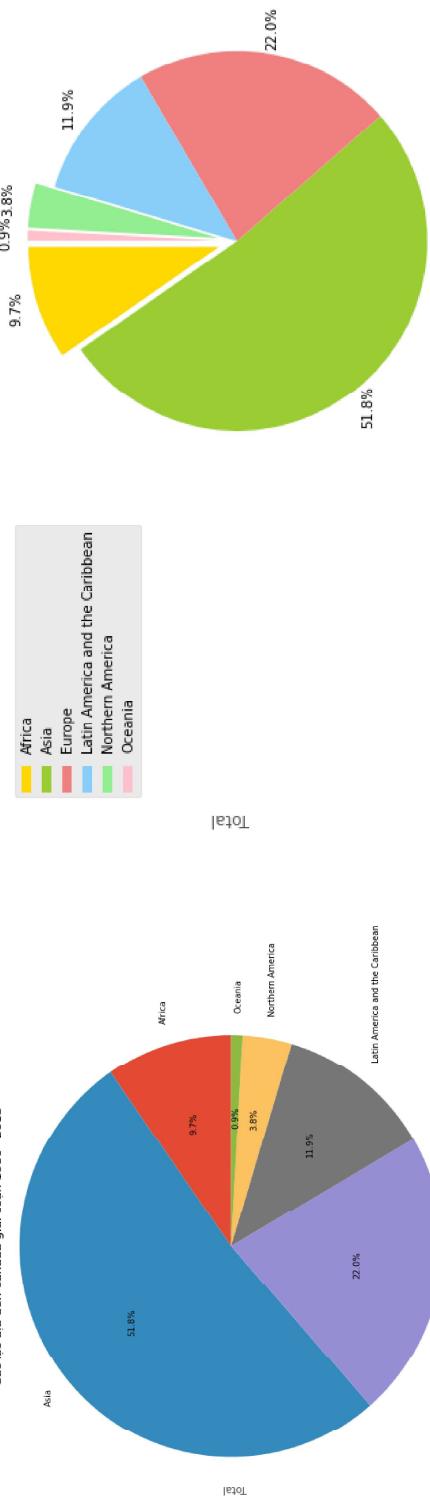
36



## 6. Pie chart

### Vẽ Pie chart

Các lục địa đến Canada giai đoạn 1980 - 2013



**Bài tập về nhà:**  
Tìm hiểu cách vẽ biểu đồ tròn hiển thị theo kiểu khác.

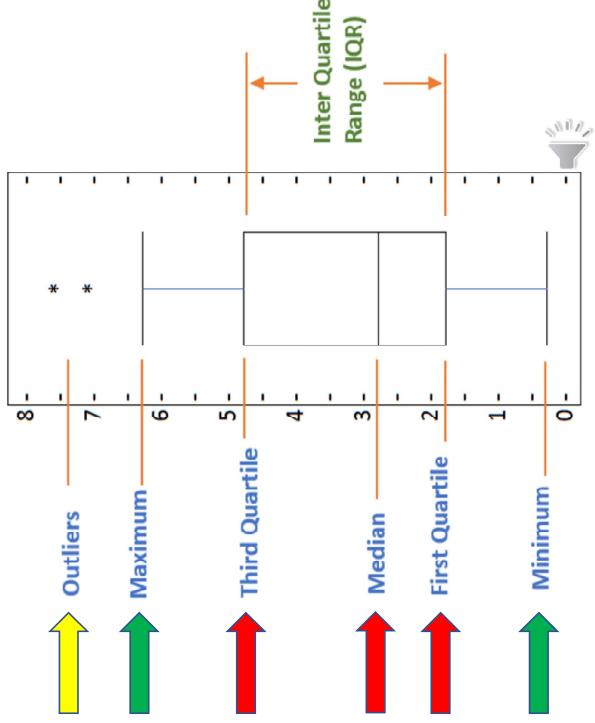
37



## 7. Box plot

### Box Plot

- Box plot là biểu đồ thể hiện sự phân bố của dữ liệu.



38



## 7. Box plot

### Câu hỏi

- Thể hiện sự phân bố người nhập cư Việt Nam đến Canada từ 1980 đến 2013

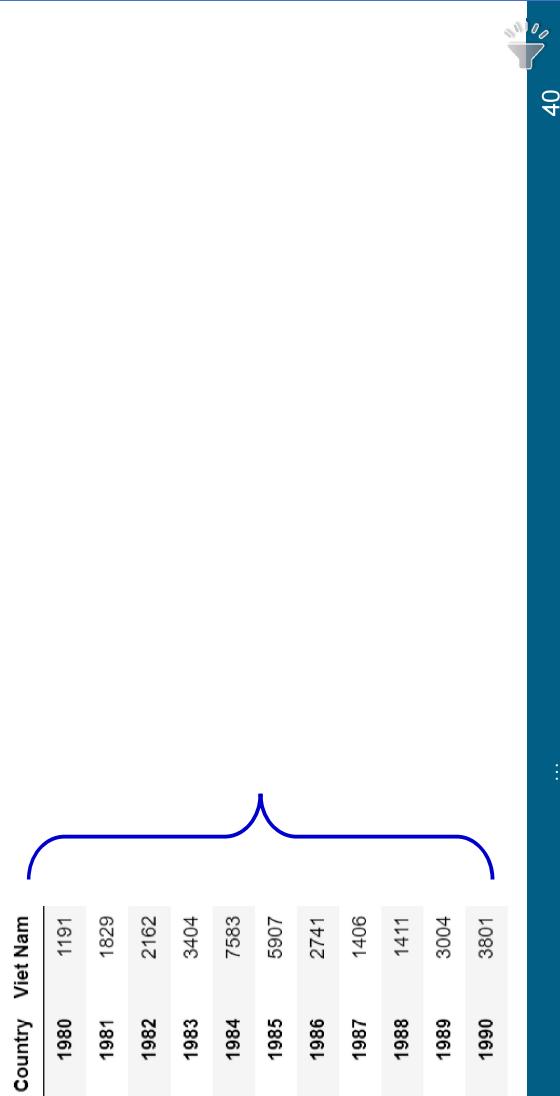
| Country | Continent           | Region                          | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|---------------------|---------------------------------|-----------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan         | Asia                            | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania             | Europe                          | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria             | Africa                          | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa      | Oceania                         | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 1    | 1    | 0    | 0    | 0    |      |
| 4       | Andorra             | Europe                          | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 1    | 1    |      |
| 5       | Angola              | Africa                          | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |
| 6       | Antigua and Barbuda | Latin America and the Caribbean | Caribbean       | Developing regions | 0    | 0    | 0    | 0    | 42   | 52  | ...  | 14   | 24   | 32   | 15   | 32   | 38   | 27   | 37   | 51   | 25   |
|         |                     |                                 |                 |                    |      |      |      |      |      |     | ...  |      |      |      |      |      |      |      |      |      |      |

39

## 7. Box plot

### Chuẩn bị dữ liệu

```
1 df_Viet = df_can.loc[['Việt Nam'], 1980:2013].transpose()  
2 df_Viet|
```



## 7. Box plot

### Vẽ Box plot

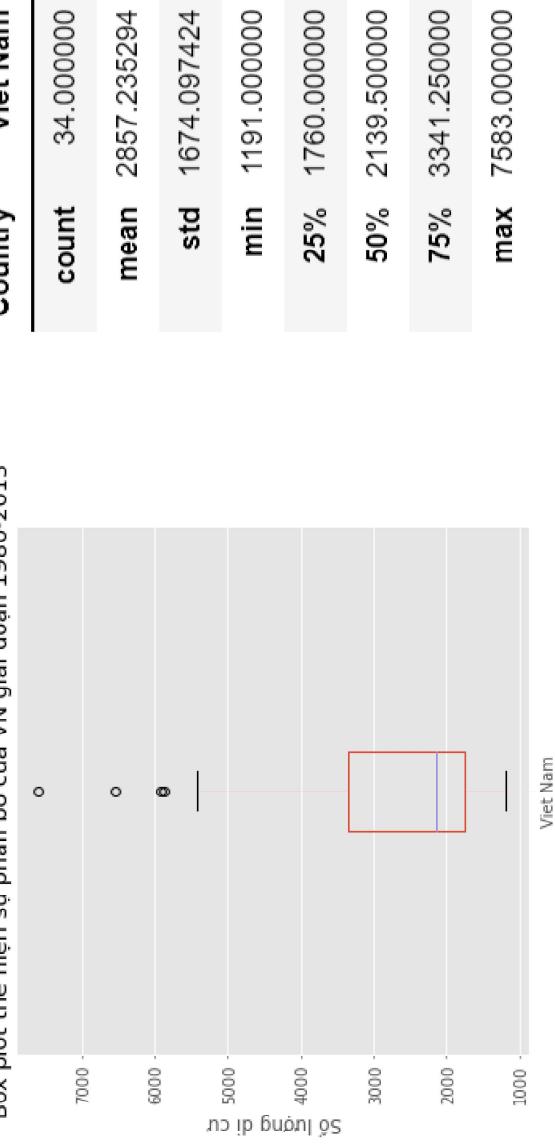
```
1 df_Viet.plot(kind='box', figsize=(6, 6))  
2 plt.title('Box plot thể hiện sự phân bố của VN giai đoạn 1980-2013')  
3 plt.ylabel('Số lượng di cư')  
4  
5 plt.show()
```



## 7. Box plot

### Vẽ Box plot

Box plot thể hiện sự phân bố của VN giai đoạn 1980-2013



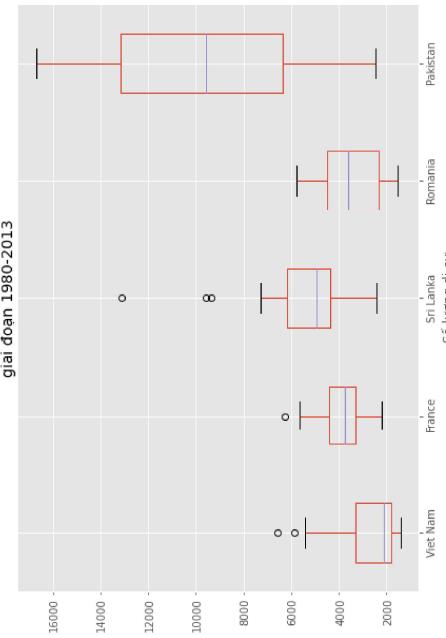
...  
42 Viet Nam



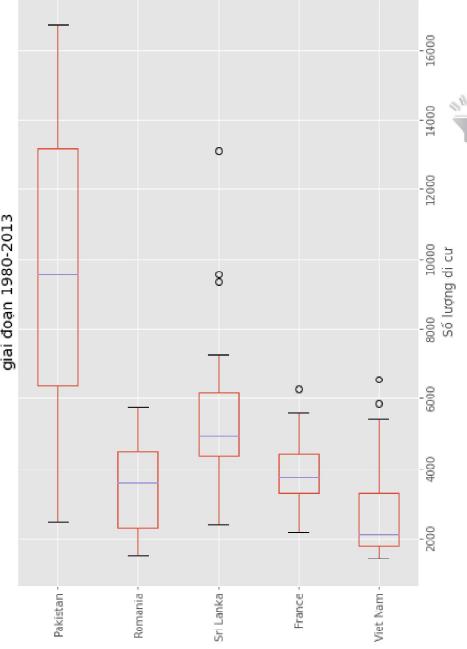
## 7. Box plot

### Bài tập

Box plot so sánh sự phân bố Việt Nam - France - Sri Lanka - Romania - Pakistan  
giai đoạn 1980-2013



Box plot so sánh sự phân bố Việt Nam - France - Sri Lanka - Romania - Pakistan  
giai đoạn 1980-2013





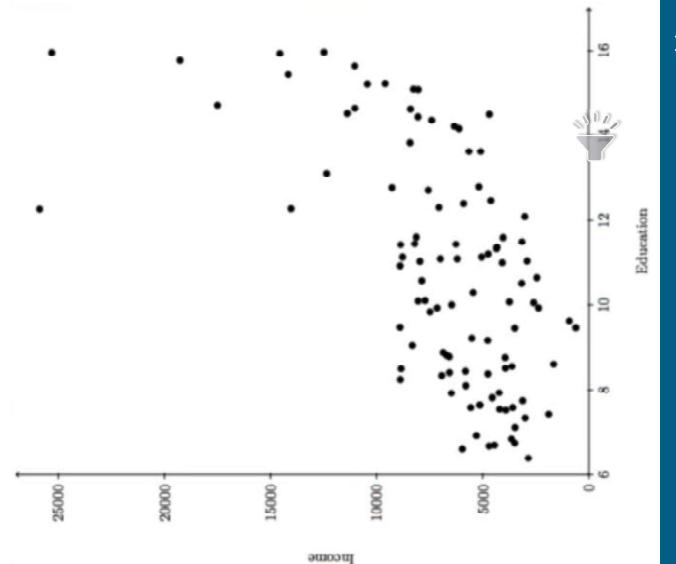
## 8. Scatter plot

### Scatter Plot

- Scatter plot là một loại biểu đồ hiển thị các giá trị liên quan đến hai biến với nhau.

- Tức biểu diễn mối quan hệ **giữa hai biến**.

- Dựa vào ảnh táng xạ của các điểm dữ liệu => có thể kết luận mối quan hệ hai biến.



44



## 8. Scatter plot

### Thông điệp

- Thể hiện mối liên hệ giữa năm và tổng số lượng di cư đến Canada từ năm 1980 đến 2013.

| Country | Continent      | Region  | DevName         | 1980               | 1981 | 1982 | 1983 | 1984 | 1985 | ... | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 |      |
|---------|----------------|---------|-----------------|--------------------|------|------|------|------|------|-----|------|------|------|------|------|------|------|------|------|------|------|
| 0       | Afghanistan    | Asia    | Southern Asia   | Developing regions | 16   | 39   | 39   | 47   | 71   | 340 | ...  | 2978 | 3436 | 3009 | 2652 | 2111 | 1746 | 1758 | 2203 | 2635 | 2004 |
| 1       | Albania        | Europe  | Southern Europe | Developed regions  | 1    | 0    | 0    | 0    | 0    | 0   | ...  | 1450 | 1223 | 856  | 702  | 560  | 716  | 561  | 539  | 620  | 603  |
| 2       | Algeria        | Africa  | Northern Africa | Developing regions | 80   | 67   | 71   | 69   | 63   | 44  | ...  | 3616 | 3626 | 4807 | 3623 | 4005 | 5393 | 4752 | 4325 | 3774 | 4331 |
| 3       | American Samoa | Oceania | Polynesia       | Developing regions | 0    | 1    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |
| 4       | Ancorra        | Europe  | Southern Europe | Developed regions  | 0    | 0    | 0    | 0    | 0    | 0   | ...  | 0    | 0    | 1    | 1    | 0    | 0    | 0    | 1    | 1    |      |
| 5       | Angola         | Africa  | Middle Africa   | Developing regions | 1    | 3    | 6    | 6    | 4    | 3   | ...  | 268  | 295  | 184  | 106  | 76   | 62   | 61   | 39   | 70   | 45   |

Dữ liệu ban đầu

45





## 8. Scatter plot

### Chuẩn bị dữ liệu

```
1 df_sum = pd.DataFrame(df_can.loc[:,1980:2013].sum(axis=0))  
2 df_sum.reset_index(inplace = True)  
3 df_sum.columns = ['year', 'total']  
4  
5 df_sum.head(10)
```

| year | total |        |
|------|-------|--------|
| 0    | 1980  | 99137  |
| 1    | 1981  | 110563 |
| 2    | 1982  | 104271 |
| 3    | 1983  | 75550  |
| 4    | 1984  | 73477  |
| 5    | 1985  | 69978  |
| 6    | 1986  | 86048  |
| 7    | 1987  | 134771 |
| 8    | 1988  | 139306 |
| 9    | 1989  | 164432 |
|      |       | ...    |

Dữ liệu đã được xử lý

46



## 8. Scatter plot

### Vẽ Scatter plot

```
1 df_sum.plot(kind='scatter', x='year', y='total', figsize=(10, 6), color='red')  
2  
3 plt.title('Tổng số lượng di cư đến Canada giai đoạn 1980-2013')  
4 plt.xlabel('Năm')  
5 plt.ylabel('Số lượng nhập cư')  
6  
7 plt.show()
```

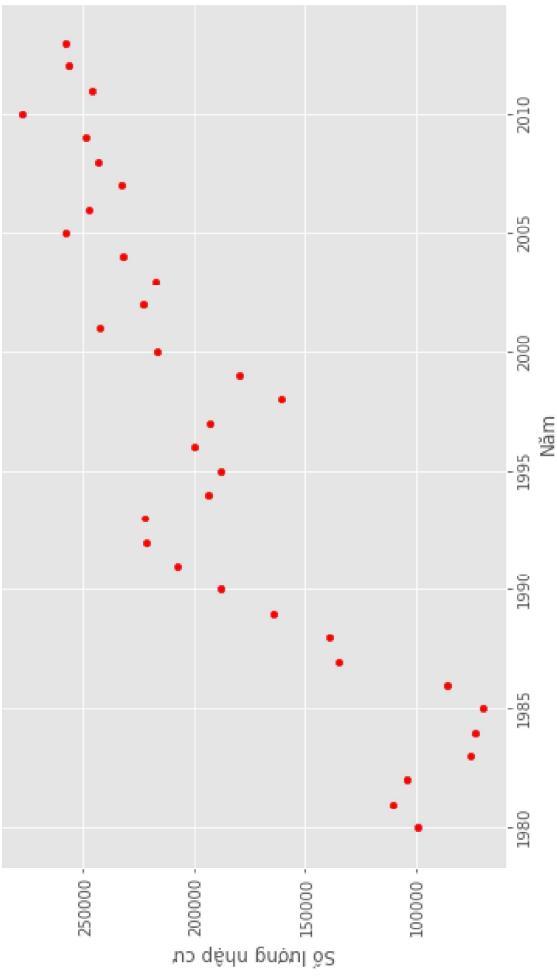
47



## 8. Scatter plot

### Vẽ Scatter plot

Tổng số lượng di cư đến Canada giai đoạn 1980-2013



48

### pandas.DataFrame.plot

```
DataFrame.plot(x=None, y=None, kind='line', ax=None, subplots=False, sharex=False, layout=None,  
figsize=None, use_index=True, title=None, grid=None, legend=True, style=None, logx=False, logy=False,  
loglog=False, xticks=None, yticks=None, xlim=None, ylim=None, rot=None, fontsize=None, colormap=None,  
table=False, yerr=None, xerr=None, secondary_y=None, sort_columns=False, sort_index=False, **kwargs) [source]
```

Make plots of DataFrame using matplotlib / pylab.

New in version 0.17.0: Each plot kind has a corresponding method on the DataFrame.plot accessor:  
df.plot(kind='line') is equivalent to df.plot.line().

data : DataFrame

x : label or position, default None

y : label, position or list of label, positions, default None

Allows plotting of one column versus another

kind : str

- 'line' : line plot (default)
- 'bar' : vertical bar plot
- 'barh' : horizontal bar plot
- 'hist' : histogram
- 'box' : boxplot
- 'kde' : Kernel Density Estimation plot
- 'density' : same as 'kde'
- 'area' : area plot
- 'pie' : pie plot
- 'scatter' : scatter plot
- 'hexbin' : hexbin plot



**Đường dẫn:** <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.html>

49

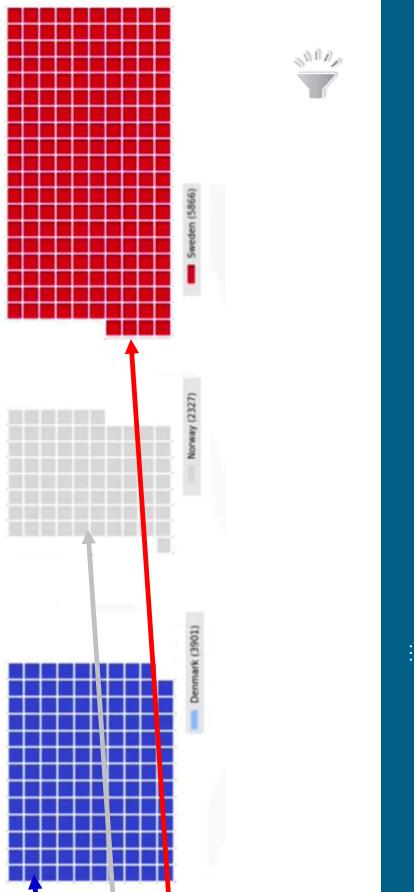


## 9. Waffle chart

### Giới thiệu

- Waffle chart là một trực quan khá thú vị được tạo ra để trình bày mục tiêu hướng đến quá trình.

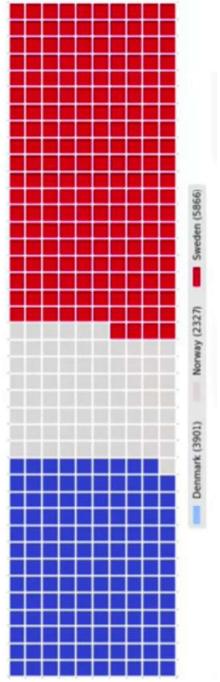
| Country | Total Immigrants |
|---------|------------------|
| Denmark | 3901             |
| Norway  | 2327             |
| Sweden  | 5866             |



## 9. Waffle chart

### Waffle Chart

| Country | Total Immigrants |
|---------|------------------|
| Denmark | 3901             |
| Norway  | 2327             |
| Sweden  | 5866             |







## 10. Word cloud

### **Ứng dụng – Hạn chế**

#### **Một vài ứng dụng của word cloud:**

- Phân tích phản hồi của khách hàng.
- Xác định từ khóa mới trong tối ưu công cụ tìm kiếm (SEO).



54

#### **Một vài hạn chế của word cloud:**

- Word cloud không hoàn hảo trong mọi tình huống.
- Dữ liệu phải được tối ưu cho mọi ngữ cảnh.

...



## 10. Word cloud

### **Cài đặt word cloud**

- Hướng dẫn tạo Word Cloud trong buổi thực hành.



55

...



58



Thank you

## Hỏi - Đáp

