Phân tích dữ liệu với R/Python:
[P2] – XỬ LÝ VÀ TRỰC QUAN HÓA DỮ LIỆU VỚI PYTHON

ThS. Nguyễn Quang Phúc
phucnq@uel.edu.vn

Thư viện Numpy

NỘI DUNG

1. Giới thiệu Numpy
2. Mảng 1 chiều
3. Mảng nhiều chiều

1. Giới thiệu Numpy

NumPy (Numerical Python) là một thư viện mở, chủ yếu hỗ trợ việc tính toán dữ liệu mảng (array) một hoặc nhiều chiều có kích thước lớn.

Cài đặt: *pip install numpy*

(trong môi trường Anaconda, numpy đã được tích hợp sẵn)

Sử dụng: *import numpy as np*

https://numpy.org/

2. Mảng 1 chiều

Khởi tạo

- Hàm **np.array()**: *np.array([phần tử 1, phần tử 2,...., phần tử n])*

```
arr1 = np.array([6, 6.5, 4, 5.5, 7, 8.5])
print(arr1)
print(type(arr1))
print(arr1.dtype)
```

```
[6.  6.5 4.  5.5 7.  8.5]
<class 'numpy.ndarray'>
float64
```

- Hàm **np.asarray()**:

```
list_sample = [5, 7, 6.5, 8, 9.5]
tuple_sample = (6, 4, 5.5, 8.5)
arr2 = np.asarray(list_sample)
arr3 = np.asarray(tuple_sample)
print(arr2)
print(arr3)
```

```
[5.  7.  6.5 8.  9.5]
[6.  4.  5.5 8.5]
```

2. Mảng 1 chiều

Khởi tạo

- Hàm **np.zeros()**, **np.ones()**:

```
arr_zeros = np.zeros(4)
arr_ones = np.ones(3, dtype=int)
print(arr_zeros)
print(arr_ones)
```

```
[0. 0. 0. 0.]
[1 1 1]
```

- Hàm **np.arange(start, stop, step, dtype)**:

```
arr1 = np.arange(2, 10, 1.5)
arr2 = np.arange(6)
print(arr1)
print(arr2)
```

```
[2.  3.5 5.  6.5 8.  9.5]
[0 1 2 3 4 5]
```

2. Mảng 1 chiều

Khởi tạo

- Hàm **np.linspace(start, stop, num, endpoint, dtype)**:

```
arr = np.linspace(5, 15, 6)
print(arr)
```

```
[ 5.  7.  9. 11. 13. 15.]
```

- Hàm **np.random.rand()/randn()/randint()**:

```
arr1 = np.random.rand(3)
arr2 = np.random.randn(4)
arr3 = np.random.randint(10, 45, 5)
print(arr1)
print(arr2)
print(arr3)
```

```
[0.84104516 0.51868114 0.40900915]
[-0.70898656  0.7423577  -1.04844593 -1.99692462]
[10 41 21 32 17]
```

2. Mảng 1 chiều

Khởi tạo

- Hàm **np.random.uniform(low, high, size)**:

```
arr1 = np.random.uniform(0.0, 5.0, 20)
print(arr1)
```

```
[3.12796281 4.37809727 2.04093551 3.97151646 4.34181853 2.92307961
 4.61853109 2.32980698 3.87701855 3.27931855 1.18309134 2.45093926
 4.85516246 4.87401567 3.27884145 0.889924   0.23298822 4.74315178
 0.85233009 0.04807371]
```

- Hàm **np.random.normal(loc, scale, size)**:

%.0: Gia tri trung binh
1.0: do lech chuan 1

```
arr2 = np.random.normal(5.0, 1.0, 10000)
print(arr2)
```

```
[5.22391698 5.05607245 5.44958119 ... 4.26805925 5.79051086 5.04896192]
```

2. Mảng 1 chiều

Truy xuất

- Ví dụ:

```
arr = np.random.randint(10, 80, 8)
print(arr)
print(arr[1])
print(arr[-2])
print(arr[[1,3,4]])
print(arr[2:5])
print(arr[arr < 40])
```

```
[60 20 32 31 59 51 32 28]
20
32
[20 31 59]
[32 31 59]
[20 32 31 32 28]
```

2. Mảng 1 chiều

Truy xuất

- Ví dụ:

```
arr = np.random.randint(10, 80, 8)
print(arr)
indices = np.where(arr>40)
print(indices)
print(arr[indices])
print(np.extract(arr>35,arr))
print(np.extract(np.mod(arr,2)==0, arr))
```

```
[31 59 15 78 46 20 13 51]
(array([1, 3, 4, 7]),)
[59 78 46 51]
[59 78 46 51]
[78 46 20]
```

2. Mảng 1 chiều

Truy xuất

- Một vài giá trị thống kê cơ bản:

```
arr = np.random.randint(5, 45, 7)
print(arr)
print(np.min(arr)) # giá trị min
print(np.argmin(arr)) # index phần tử min
print(np.max(arr)) # giá trị max
print(np.argmax(arr)) # index phần tử max
print(np.mean(arr)) # giá trị trung bình
print(np.median(arr)) # giá trị trung vị
print(np.std(arr)) # độ lệch chuẩn
```

```
[22  5 17 44 21 27 25]
5
1
44
3
23.0
22.0
10.862780491200215
```

2. Mảng 1 chiều

Thêm phần tử

```
arr = np.array([4, 2, 15, 7, 9, 5, 11, 8])
print(arr)
```

```
[ 4  2 15  7  9  5 11  8]
```

```
arr = np.append(arr, [6, 3]) # Thêm phần tử vào cuối
print(arr)
```

```
[ 4  2 15  7  9  5 11  8  6  3]
```

```
arr = np.insert(arr, 2, [6, 1]) # Thêm tại vị trí bất kỳ
print(arr)
```

```
[ 4  2  6  1 15  7  9  5 11  8  6  3]
```

Xóa phần tử

```
arr = np.array([4, 2, 15, 7, 9, 5, 11, 8])
arr = np.delete(arr, [2, 4]) # Xóa phần tử theo index
print(arr)
```

```
[ 4  2  7  5 11  8]
```

2. Mảng 1 chiều

Sắp xếp

```
arr = np.array([4, 2, 15, 7, 9, 5, 11, 8])
print(np.sort(arr))
print(np.sort(arr)[::-1])
```

```
[ 2  4  5  7  8  9 11 15]
[15 11  9  8  7  5  4  2]
```

Tính toán số học

```
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
print(arr1 + arr2)
print(arr1 - arr2)
print(arr1 * arr2)
print(arr1 / arr2)
print(arr1 % arr2)
```

```
[5 7 9]
[-3 -3 -3]
[ 4 10 18]
[0.25 0.4  0.5 ]
[1 2 3]
```

3. Mảng nhiều chiều

Khởi tạo

```
arr = np.array([[1, 2, 3], [4, 5, 6]])

print(arr)
```

```
[[1 2 3]
 [4 5 6]]
```

```
print(arr.shape) # Kích thước array
```

```
(2, 3)
```

```
arr = np.array([[[1, 2, 3], [4, 5, 6]],
                [[7, 8, 9], [10, 11, 12]]])

print(arr)
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

```
print(arr.shape) # Kích thước array
```

```
(2, 2, 3)
```

2: so phan tu, ma tran, 2: 2 dong, 3: c

3. Mảng nhiều chiều

Khởi tạo

```
arr = np.zeros([2, 3, 4], dtype=int)
print(arr)
```

```
[[[0 0 0 0]
  [0 0 0 0]
  [0 0 0 0]]

 [[0 0 0 0]
  [0 0 0 0]
  [0 0 0 0]]]
```

```
arr = np.ones([2, 3], dtype=int)
print(arr)
```

```
[[1 1 1]
 [1 1 1]]
```

3. Mảng nhiều chiều

Truy xuất

```
arr = np.array([[[1, 2, 3], [4, 5, 6]],
                [[7, 8, 9], [10, 11, 12]]])
print(arr)
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

```
print(arr[0])
```

```
[[1 2 3]
 [4 5 6]]
```

```
print(arr[1][0][2])
```

```
9
```

```
print(arr[1][1][1])
```

```
11
```

3. Mảng nhiều chiều

Truy xuất

```
arr = np.array([[[1, 2, 3], [4, 5, 6]],
                [[7, 8, 9], [10, 11, 12]]])
print(arr)
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

→ Truy xuất các trị thống kê cơ bản tương tự như mảng 1 chiều

```
print(np.max(arr))
```

```
12
```

```
print(np.mean(arr[1]))
```

```
9.5
```

```
print(arr[np.where(arr>8)])
```

```
[ 9 10 11 12]
```

3. Mảng nhiều chiều

Cập nhật giá trị

```
arr = np.array([[[1, 2, 3], [4, 5, 6]],
                [[7, 8, 9], [10, 11, 12]]])
print(arr)
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

```
arr[1][0][2] = 10
print(arr[1][0])
```

```
[ 7  8 10]
```

3. Mảng nhiều chiều

Sắp xếp

```
arr = np.array([[[1, 3, 2], [6, 4, 5]],
                [[7, 9, 8], [12, 10, 11]]])
print(np.sort(arr))
```

```
[[[ 1  2  3]
  [ 4  5  6]]

 [[ 7  8  9]
  [10 11 12]]]
```

3. Mảng nhiều chiều

Tính toán số học

```
arr1 = np.array([[1, 2, 3], [4, 5, 6]])
arr2 = np.array([1, 2, 3])
```

```
print(arr1 + arr2)
```

```
[[2 4 6]
 [5 7 9]]
```

```
print(arr1 - arr2)
```

```
[[0 0 0]
 [3 3 3]]
```

```
print(arr1 * arr2)
```

```
[[ 1  4  9]
 [ 4 10 18]]
```

```
print(arr1 / arr2)
```

```
[[1.  1.  1. ]
 [4.  2.5 2. ]]
```

→ Tính toán tương tự cho các phép toán số học khác

3. Mảng nhiều chiều

Chuyển đổi dòng ←→ cột

```
arr1 = np.array(range(12))
print(arr1)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11]
```

```
arr_reshape = arr1.reshape(3,4)
print(arr_reshape)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
```

```
arr2 = arr_reshape.reshape(1,-1)
print(arr2)
```

```
[[ 0  1  2  3  4  5  6  7  8  9 10 11]]
```

```
arr3 = arr_reshape.flatten()
print(arr3)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11]
```

Thư viện Pandas

NỘI DUNG

1. Giới thiệu Pandas
2. Cấu trúc dữ liệu trong Pandas
3. Làm việc với dữ liệu rỗng

1. Giới thiệu Pandas

Pandas là một thư viện mở, được sử dụng rộng rãi trong việc xử lý dữ liệu, ứng dụng trong nhiều lĩnh vực như kinh tế, khoa học, thống kê, …

Cài đặt: *pip install pandas*

(trong môi trường Anaconda, pandas đã được tích hợp sẵn)

Sử dụng: *import pandas as pd*

2. Cấu trúc dữ liệu trong Pandas

Series

Series là một dạng cấu trúc dữ liệu *tương tự như mảng 1 chiều* nhưng *được diễn tả theo chiều dọc*. Tất cả phần tử trong Series đều được gán chỉ mục index.

```
ser = pd.Series([2, 4, 6, 8])
print(ser)
```

```
0    2
1    4
2    6
3    8
dtype: int64
```

```
arr_price = np.array([76.3, 23.1, 102.4])
arr_symbol = np.array(['FPT', 'ACB', 'VNM'])
ser = pd.Series(arr_price, index=arr_symbol)
print(ser)
```

```
FPT     76.3
ACB     23.1
VNM    102.4
dtype: float64
```

```
dic = {'FPT':76.3, 'ACB':23.1,
'VNM':102.4}
ser = pd.Series(dic)
print(ser)
```

2. Cấu trúc dữ liệu trong Pandas

Series

- **Truy xuất**

```
FPT      76.3
ACB      23.1
VNM     102.4
dtype: float64
```

```
print(ser['ACB'])
```

```
23.1
```

```
print(ser[2])
```

```
102.4
```

```
print(ser[1:])
```

```
ACB      23.1
VNM     102.4
dtype: float64
```

```
print(ser[['FPT', 'VNM']])
```

```
FPT      76.3
VNM     102.4
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

Series

- **Truy xuất**

```
FPT      76.3
ACB      23.1
VNM     102.4
dtype: float64
```

```
print(ser.size)
print(len(ser))
```

```
3
3
```

```
print(ser.values)
```

```
[ 76.3  23.1 102.4]
```

```
print(ser.index)
```

```
Index(['FPT', 'ACB', 'VNM'], dtype='object')
```

```
print(ser.axes)
```

```
[Index(['FPT', 'ACB', 'VNM'], dtype='object')]
```

2. Cấu trúc dữ liệu trong Pandas

Series

- **Truy xuất**

```
dic = {'FPT':76.3, 'ACB':23.1, 'VNM':102.4,
       'AGH': 7.8, 'FLC':3.5, 'HTC':24.2}
ser = pd.Series(dic)
print(ser)
```

```
FPT      76.3
ACB      23.1
VNM     102.4
AGH       7.8
FLC       3.5
HTC      24.2
dtype: float64
```

```
print(ser.head(3))
```

```
FPT      76.3
ACB      23.1
VNM     102.4
dtype: float64
```

```
print(ser.tail(2))
```

```
FLC      3.5
HTC     24.2
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

Series

➢ **Truy xuất**

```
FPT      76.3
ACB      23.1
VNM     102.4
AGH       7.8
FLC       3.5
HTC      24.2
dtype: float64
```

```
print(ser.mean())
print(ser.std())
print(ser.describe())
```

```
39.550000000000004
40.285419198514
count      6.000000
mean      39.550000
std       40.285419
min        3.500000
25%       11.625000
50%       23.650000
75%       63.275000
max      102.400000
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

Series

- **Cập nhật**

```
FPT      76.3
ACB      23.1
VNM     102.4
dtype: float64
```

```
ser['FPT'] = 81
ser[2] = 106
print(ser)
```

```
FPT      81.0
ACB      23.1
VNM     106.0
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

Series

➢ Xóa phần tử

```
FPT      76.3
ACB      23.1
VNM     102.4
AGH       7.8
FLC       3.5
HTC      24.2
dtype: float64
```

```
print(ser.drop(ser.index[[0, 2]]))
```

```
ACB     23.1
AGH      7.8
FLC      3.5
HTC     24.2
dtype: float64
```

```
print(ser.drop(['FLC', 'AGH']))
```

```
FPT      76.3
ACB      23.1
VNM     102.4
HTC      24.2
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

Series

➢ Tính toán số học

```
FPT      76.3
ACB      23.1
VNM     102.4
AGH       7.8
FLC       3.5
HTC      24.2
dtype: float64
```

```
print(ser + 2)
```

```
FPT      78.3
ACB      25.1
VNM     104.4
AGH       9.8
FLC       5.5
HTC      26.2
dtype: float64
```

x: gia tri hien tai cua du lieu

```
print(ser.map(lambda x:x*2))
```

```
FPT     152.6
ACB      46.2
VNM     204.8
AGH      15.6
FLC       7.0
HTC      48.4
dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

DataFrame có cấu trúc dữ liệu được tổ chức theo dòng và cột.

```
list_sample = [['PNJ', 180.1, 182], ['VIB', 22.3, 21.2],
['VIC', 46.2, 45.6], ['VNM', 150, 146.1]]
df = pd.DataFrame(list_sample, columns=['Symbol',
'Open', 'Close'])
print(df)
```

```
  Symbol   Open  Close
0    PNJ  180.1  182.0
1    VIB   22.3   21.2
2    VIC   46.2   45.6
3    VNM  150.0  146.1
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Đọc dữ liệu từ file .csv/.xlsx**

```
import pandas as pd
df = pd.read_csv('employee.csv')
print(df)
```

```
   Id        Name         Dob                  Role
0   1   Tuấn Kiệt  12/02/2000          Web Developer
1   2  Khánh Hưng  22/04/2003                 Tester
2   3     Gia Hân  06/08/2002       Business Analyst
3   4     Ngọc Tú  02/02/2001  Mobile App Developer
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Đọc dữ liệu từ file .csv/.xlsx**

$ pip install xlrd

$ pip install openpyxl → .xlsx

```
import pandas as pd
df = pd.read_excel('Sales.xlsx')
```

	Week	Sales_Volume	Price	Ads_Cost
0	1	350	5.50000	3.30000
1	2	460	7.50000	3.30000
2	3	350	8.00000	3.00000
3	4	430	8.00000	4.50000
4	5	350	6.80000	3.00000

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Đọc dữ liệu từ file .csv/.xlsx**

```
df = pd.read_csv('./data/TCB_2018_2020.csv')
print(df)
```

```
           Date    High    Low    Open   Close     Avg      Volume
0    2018-06-04  105.00  102.4  102.40  102.40  102.49   2811840.0
1    2018-06-05  106.00   96.0   99.10   96.00  100.19   1689500.0
2    2018-06-06   96.00   91.0   95.00   92.00   92.98   1901680.0
3    2018-06-07   98.40   93.1   94.50   98.40   97.00   1476540.0
4    2018-06-08  105.20   99.5  101.00  105.20  103.83   2008500.0
..          ...     ...    ...     ...     ...     ...         ...
646  2020-12-25   29.60   27.7   27.95   29.55   28.69  18797140.0
647  2020-12-28   30.30   29.5   29.90   29.70   29.98  15787300.0
648  2020-12-29   29.90   29.5   29.65   29.75   29.75  13958770.0
649  2020-12-30   30.55   29.7   29.80   29.90   30.06  16629440.0
650  2020-12-31   31.75   29.9   29.95   31.50   30.63  16486510.0

[651 rows x 7 columns]
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Đọc dữ liệu từ file .csv/.xlsx**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.head())
```

	High	Low	Open	Close	Avg	Volume
Date						
2018-06-04	105.0	102.4	102.4	102.4	102.49	2811840.0
2018-06-05	106.0	96.0	99.1	96.0	100.19	1689500.0
2018-06-06	96.0	91.0	95.0	92.0	92.98	1901680.0
2018-06-07	98.4	93.1	94.5	98.4	97.00	1476540.0
2018-06-08	105.2	99.5	101.0	105.2	103.83	2008500.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

➢ Truy xuất dữ liệu theo điều kiện

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
# Xuất dữ liệu với điều kiện giá đóng cửa lớn hơn 98
print(df[df['Close']>98])
```

	High	Low	Open	Close	Avg	Volume
Date						
2018-06-04	105.0	102.4	102.4	102.4	102.49	2811840.0
2018-06-07	98.4	93.1	94.5	98.4	97.00	1476540.0
2018-06-08	105.2	99.5	101.0	105.2	103.83	2008500.0
2018-06-11	109.0	103.2	104.8	109.0	106.71	1229830.0
2018-06-12	108.0	102.9	108.0	105.0	104.93	1183630.0
2018-06-13	108.0	104.5	105.0	105.4	105.39	888330.0
2018-06-14	106.5	103.5	106.5	105.0	105.03	792130.0
2018-06-15	105.5	103.5	105.5	105.2	104.84	2558933.0
2018-06-18	105.0	98.0	104.1	100.0	101.89	2042807.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo cột**

```
print(df[["High", "Low"]].tail())
```

```
             High   Low
Date
2020-12-25  29.60  27.7
2020-12-28  30.30  29.5
2020-12-29  29.90  29.5
2020-12-30  30.55  29.7
2020-12-31  31.75  29.9
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo cột**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', header=None)
print(df[[0, 2, 3]].tail())
```

	0	2	3
647	2020-12-25	27.7	27.95
648	2020-12-28	29.5	29.9
649	2020-12-29	29.5	29.65
650	2020-12-30	29.7	29.8
651	2020-12-31	29.9	29.95

Lưu ý*: chỉ truy xuất được dữ liệu cột theo index khi DataFrame được khởi tạo với index mặc định.

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo dòng**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.loc['2020-06-15'])
```

```
High            21.10
Low             20.05
Open            21.10
Close           20.10
Avg             20.45
Volume    2367220.00
Name: 2020-06-15, dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo dòng**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.loc[['2019-06-10', '2020-06-10']])
```

	High	Low	Open	Close	Avg	Volume
Date						
2019-06-10	21.90	21.30	21.80	21.50	21.61	2432830.0
2020-06-10	21.95	21.25	21.35	21.85	21.66	2611700.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo dòng**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.iloc[0]) # Lấy dòng đầu tiên
```

```
High          105.00
Low           102.40
Open          102.40
Close         102.40
Avg           102.49
Volume    2811840.00
Name: 2018-06-04, dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo dòng**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.iloc[[0, 2]]) # Lấy nhiều dòng
```

```
             High    Low   Open  Close     Avg     Volume
Date
2018-06-04  105.0  102.4  102.4  102.4  102.49  2811840.0
2018-06-06   96.0   91.0   95.0   92.0   92.98  1901680.0
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo dòng**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
print(df.iloc[35:41]) # Lấy nhiều dòng liên tiếp
```

	High	Low	Open	Close	Avg	Volume
Date						
2018-07-23	27.85	26.6	27.30	26.60	27.28	920690.0
2018-07-24	27.50	26.3	27.00	26.40	26.78	1019120.0
2018-07-25	26.85	24.9	26.80	26.75	26.13	4238010.0
2018-07-26	26.25	25.5	26.00	25.80	25.85	777430.0
2018-07-27	26.20	25.6	25.80	25.95	25.91	3207660.0
2018-07-30	26.20	25.9	25.95	26.10	26.05	1549238.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo phần tử**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
# Truy xuất giá đóng cửa ngày 20-08-2019
print(df.loc['2019-08-20', 'Close'])
```

```
21.55
```

```
print(df.loc['2020-12-25':, 'Open'])
```

```
Date
2020-12-25    27.95
2020-12-28    29.90
2020-12-29    29.65
2020-12-30    29.80
2020-12-31    29.95
Name: Open, dtype: float64
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Truy xuất dữ liệu theo phần tử**

```
df = pd.read_csv('./data/TCB_2018_2020.csv', index_col=0)
# Truy xuất dòng thứ 5 và cột đầu tiên
print(df.iloc[4, 0])
```

```
105.2
```

```
# Truy xuất dòng từ dòng thứ 648 với tất cả các cột
print(df.iloc[648:, :])
```

```
             High   Low   Open  Close    Avg      Volume
Date
2020-12-29  29.90  29.5  29.65  29.75  29.75  13958770.0
2020-12-30  30.55  29.7  29.80  29.90  30.06  16629440.0
2020-12-31  31.75  29.9  29.95  31.50  30.63  16486510.0
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Xóa dữ liệu

```
df = pd.read_csv('./data/SampleData.csv', index_col=0)
print(df)
```

```
        Price    PE
Symbol
VNM     104.0  16.0
REE      22.3   5.8
DHG      61.4   7.8
FPT      30.4   5.4
```

```
del df['Price'] # Xóa cột Price
print(df)
```

```
            PE
Symbol
VNM       16.0
REE        5.8
DHG        7.8
FPT        5.4
```

```
# Xóa dòng có index là 2
print(df.drop(df.index[2]))
```

```
        Price    PE
Symbol
VNM     104.0  16.0
REE      22.3   5.8
FPT      30.4   5.4
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Thêm dữ liệu

```
df = pd.read_csv('./data/SampleData.csv', index_col=0)
print(df)
```

```
        Price    PE
Symbol
VNM     104.0  16.0
REE      22.3   5.8
DHG      61.4   7.8
FPT      30.4   5.4
```

```
# Thêm cột giá USD với tỷ giá 1USD = 23.000 VND
df['Usd'] = df['Price']/23
print(df)
```

```
        Price    PE       Usd
Symbol
VNM     104.0  16.0  4.521739
REE      22.3   5.8  0.969565
DHG      61.4   7.8  2.669565
FPT      30.4   5.4  1.321739
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

➢ Thêm dữ liệu

```
df = pd.read_csv('./data/SampleData.csv')
print(df)
```

```
  Symbol  Price    PE
0    VNM  104.0  16.0
1    REE   22.3   5.8
2    DHG   61.4   7.8
3    FPT   30.4   5.4
```

```
# Thêm dòng vào cuối df
df.loc[df.shape[0]] = ['VCB', 113.6, 23.09]
print(df)
```

```
  Symbol  Price     PE
0    VNM  104.0  16.00
1    REE   22.3   5.80
2    DHG   61.4   7.80
3    FPT   30.4   5.40
4    VCB  113.6  23.09
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Sắp xếp dữ liệu**

```
sales_2020 = pd.DataFrame({'sales': [450, 360, 550, 480]},
index=['Mar', 'Jun', 'Feb', 'Apr'])
print(sales_2020)
```

```
     sales
Mar    450
Jun    360
Feb    550
Apr    480
```

```
print(sales_2020.sort_index())
```

```
     sales
Apr    480
Feb    550
Jun    360
Mar    450
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

➢ Sắp xếp dữ liệu

```
sales_2020 = pd.DataFrame({'sales': [450, 360, 550, 480]},
index=['Mar', 'Jun', 'Feb', 'Apr'])
print(sales_2020)
```

```
     sales
Mar    450
Jun    360
Feb    550
Apr    480
```

```
sales_2021 = pd.DataFrame({'sales': [650, 600,
700, 680]}, index=['Feb', 'Mar', 'Apr', 'Jun'])
print(sales_2020.reindex(sales_2021.index))
```

```
     sales
Feb    550
Mar    450
Apr    480
Jun    360
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Nhóm dữ liệu**: Khi các phần tử trong 1 cột có sự lặp lại về giá trị (các biến có tính phân loại).

```
df = pd.read_csv('./data/SampleData2.csv')
print(df)
```

	Symbol	Price	PE	Group
0	VNM	104.0	16.0	high
1	REE	22.3	5.8	low
2	DHG	61.4	7.8	high
3	FPT	30.4	5.4	low
4	AGF	7.8	13.0	low

```
print(df.groupby('Group').mean())
```

	Price	PE
Group		
high	82.700000	11.900000
low	20.166667	8.066667

sum(), **count**(), …

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df = pd.read_csv('./data/SampleData2.csv')
df1 = df[['Symbol', 'Price', 'Group']]
df2 = df[['Symbol', 'PE', 'Group']]
```

```
print(df1)
```

	Symbol	Price	Group
0	VNM	104.0	high
1	REE	22.3	low
2	DHG	61.4	high
3	FPT	30.4	low
4	AGF	7.8	low

```
print(df2)
```

	Symbol	PE	Group
0	VNM	16.0	high
1	REE	5.8	low
2	DHG	7.8	high
3	FPT	5.4	low
4	AGF	13.0	low

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df_concat = pd.concat([df1, df2])
print(df_concat)
```

	Symbol	Price	Group	PE
0	VNM	104.0	high	NaN
1	REE	22.3	low	NaN
2	DHG	61.4	high	NaN
3	FPT	30.4	low	NaN
4	AGF	7.8	low	NaN
0	VNM	NaN	high	16.0
1	REE	NaN	low	5.8
2	DHG	NaN	high	7.8
3	FPT	NaN	low	5.4
4	AGF	NaN	low	13.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df_concat = pd.concat([df1, df2], join='inner')
print(df_concat)
```

→ Loại bỏ các cột không trùng nhau sử dụng tham số **join = 'inner'** (mặc định join = 'outer'

```
  Symbol Group
0    VNM  high
1    REE   low
2    DHG  high
3    FPT   low
4    AGF   low
0    VNM  high
1    REE   low
2    DHG  high
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Gộp dữ liệu**

```
df_concat = pd.concat([df1, df2], axis=1)
print(df_concat)
```

	Symbol	Price	Group	Symbol	PE	Group
0	VNM	104.0	high	VNM	16.0	high
1	REE	22.3	low	REE	5.8	low
2	DHG	61.4	high	DHG	7.8	high
3	FPT	30.4	low	FPT	5.4	low
4	AGF	7.8	low	AGF	13.0	low

→ Gộp dữ liệu theo dòng tham số **axis = 1** (mặc định axis = 0)

2. Cấu trúc dữ liệu trong Pandas

DataFrame

- **Gộp dữ liệu**

```
df_append = df1.append(df2)
print(df_append)
```

	Symbol	Price	Group	PE
0	VNM	104.0	high	NaN
1	REE	22.3	low	NaN
2	DHG	61.4	high	NaN
3	FPT	30.4	low	NaN
4	AGF	7.8	low	NaN
0	VNM	NaN	high	16.0
1	REE	NaN	low	5.8
2	DHG	NaN	high	7.8
3	FPT	NaN	low	5.4
4	AGF	NaN	low	13.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df_merge = pd.merge(df1, df2)
print(df_merge)
```

	Symbol	Price	Group	PE
0	VNM	104.0	high	16.0
1	REE	22.3	low	5.8
2	DHG	61.4	high	7.8
3	FPT	30.4	low	5.4
4	AGF	7.8	low	13.0

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df = pd.read_csv('./data/SampleData2.csv')
df1 = df[['Symbol', 'Price', 'Group']]
df1 = df1.drop(df1.index[3]) # Xóa FPT
df2 = df[['Symbol', 'PE', 'Group']]
```

```
print(df1)
```

```
  Symbol  Price Group
0    VNM  104.0  high
1    REE   22.3   low
2    DHG   61.4  high
4    AGF    7.8   low
```

```
print(df2)
```

```
  Symbol    PE Group
0    VNM  16.0  high
1    REE   5.8   low
2    DHG   7.8  high
3    FPT   5.4   low
4    AGF  13.0   low
```

2. Cấu trúc dữ liệu trong Pandas

DataFrame

Gộp dữ liệu

```
df_merge = pd.merge(df1, df2)
print(df_merge)
```

	Symbol	Price	Group	PE
0	VNM	104.0	high	16.0
1	REE	22.3	low	5.8
2	DHG	61.4	high	7.8
3	AGF	7.8	low	13.0

```
df_merge = pd.merge(df1, df2,
how='outer')
print(df_merge)
```

	Symbol	Price	Group	PE
0	VNM	104.0	high	16.0
1	REE	22.3	low	5.8
2	DHG	61.4	high	7.8
3	AGF	7.8	low	13.0
4	FPT	NaN	low	5.4

3. Làm việc với dữ liệu rỗng

Kiểm tra dữ liệu rỗng

```
df = pd.read_csv('./data/SampleData_NaN.csv')
print(df)
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.8	0.7	4.7
1	UNI	2015	76.7	13.8	NaN
2	UNI	2014	26.6	0.9	7.3
3	UNI	2013	NaN	0.8	6.5
4	UNI	2012	NaN	16.0	NaN
5	UNI	2011	NaN	-7.7	3.2
6	UNI	2010	56.7	12.4	11.5
7	UNI	2009	92.7	NaN	12.7
8	UNI	2008	70.1	1.4	4.5
9	UNI	2007	226.8	8.6	21.6

3. Làm việc với dữ liệu rỗng

Kiểm tra dữ liệu rỗng

```
df = pd.read_csv('./data/SampleData_NaN.csv')
print(df.isnull())
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	False	False	False	False	False
1	False	False	False	False	True
2	False	False	False	False	False
3	False	False	True	False	False
4	False	False	True	False	True
5	False	False	True	False	False
6	False	False	False	False	False
7	False	False	False	True	False
8	False	False	False	False	False
9	False	False	False	False	False

3. Làm việc với dữ liệu rỗng

Kiểm tra dữ liệu rỗng

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Kiểm tra dữ liệu rỗng cho từng cột
print(df.isnull().any())
```

```
Symbol          False
Year            False
Sales            True
Net Profit       True
Stock Price      True
dtype: bool
```

```
# Kiểm tra dữ liệu rỗng cho toàn bộ DataFrame
print(df.isnull().values.any())
```

```
True
```

3. Làm việc với dữ liệu rỗng

Kiểm tra dữ liệu rỗng

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Kiểm tra số lượng dữ liệu rỗng cho từng cột
print(df.isnull().sum())
```

```
Symbol        0
Year          0
Sales         3
Net Profit    1
Stock Price   2
dtype: int64
```

```
# Kiểm tra số lượng dữ liệu rỗng cho toàn bộ DataFrame
print(df.isnull().sum().sum()) → 6
```

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Xóa dòng chứa phần tử rỗng
df_delete_na_by_row = df.dropna(axis=0)
print(df_delete_na_by_row)
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.8	0.7	4.7
2	UNI	2014	26.6	0.9	7.3
6	UNI	2010	56.7	12.4	11.5
8	UNI	2008	70.1	1.4	4.5
9	UNI	2007	226.8	8.6	21.6

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Xóa cột chứa phần tử rỗng
df_delete_na_by_col = df.dropna(axis=1)
print(df_delete_na_by_col)
```

	Symbol	Year
0	UNI	2016
1	UNI	2015
2	UNI	2014
3	UNI	2013
4	UNI	2012
5	UNI	2011
6	UNI	2010
7	UNI	2009
8	UNI	2008
9	UNI	2007

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Fill giá trị 100 cho phần tử rỗng
df_fill_na_100 = df.fillna(100)
print(df_fill_na_100)
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.8	0.7	4.7
1	UNI	2015	76.7	13.8	100.0
2	UNI	2014	26.6	0.9	7.3
3	UNI	2013	100.0	0.8	6.5
4	UNI	2012	100.0	16.0	100.0
5	UNI	2011	100.0	-7.7	3.2
6	UNI	2010	56.7	12.4	11.5
7	UNI	2009	92.7	100.0	12.7
8	UNI	2008	70.1	1.4	4.5
9	UNI	2007	226.8	8.6	21.6

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Fill phần tử rỗng với giá trị liền kề phí dưới
df_fill_na_bfill = df.fillna(method='bfill')
print(df_fill_na_bfill)
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.8	0.7	4.7
1	UNI	2015	76.7	13.8	7.3
2	UNI	2014	26.6	0.9	7.3
3	UNI	2013	56.7	0.8	6.5
4	UNI	2012	56.7	16.0	3.2
5	UNI	2011	56.7	-7.7	3.2
6	UNI	2010	56.7	12.4	11.5
7	UNI	2009	92.7	1.4	12.7
8	UNI	2008	70.1	1.4	4.5
9	UNI	2007	226.8	8.6	21.6

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Fill phần tử rỗng với giá trị liền kề phí trên
df_fill_na_ffill = df.fillna(method='ffill')
print(df_fill_na_ffill)
```

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.8	0.7	4.7
1	UNI	2015	76.7	13.8	4.7
2	UNI	2014	26.6	0.9	7.3
3	UNI	2013	26.6	0.8	6.5
4	UNI	2012	26.6	16.0	6.5
5	UNI	2011	26.6	-7.7	3.2
6	UNI	2010	56.7	12.4	11.5
7	UNI	2009	92.7	12.4	12.7
8	UNI	2008	70.1	1.4	4.5
9	UNI	2007	226.8	8.6	21.6

3. Làm việc với dữ liệu rỗng

Xử lý dữ liệu rỗng

Dữ liệu rỗng có thể được xử lý bằng việc *xóa bỏ* hoặc *điền giá trị mới*.

```
df = pd.read_csv('./data/SampleData_NaN.csv')
# Fill phần tử rỗng với giá trị nội suy
df_fill_na_interpolate = df.interpolate()
print(df_fill_na_interpolate)
```

**Khi có nhiều phần tử rỗng liên tục nhau nên sử dụng phương pháp nội suy giá trị.*

	Symbol	Year	Sales	Net Profit	Stock Price
0	UNI	2016	33.800	0.7	4.70
1	UNI	2015	76.700	13.8	6.00
2	UNI	2014	26.600	0.9	7.30
3	UNI	2013	34.125	0.8	6.50
4	UNI	2012	41.650	16.0	4.85
5	UNI	2011	49.175	-7.7	3.20
6	UNI	2010	56.700	12.4	11.50
7	UNI	2009	92.700	6.9	12.70
8	UNI	2008	70.100	1.4	4.50
9	UNI	2007	226.800	8.6	21.60

Thư viện

Matplotlib & Seaborn

NỘI DUNG

1. Giới thiệu Matplotlib
2. Các loại biểu đồ thông dụng với Matplotlib
3. Giới thiệu Seaborn
4. Các loại biểu đồ trong Seaborn

1. Giới thiệu Matplotlib

Matplotlib là một thư viện phổ biến trong Python, phục vụ chính cho mục đích vẽ đồ thị mô tả dữ liệu, hỗ trợ đa dạng các loại đồ thị.

Cài đặt: *pip install matplotlib*

(trong môi trường Anaconda, matplotlib đã được tích hợp)

Sử dụng: *import* ***matplotlib.pyplot*** *as plt # cách 1*

from ***matplotlib*** *import* ***pyplot*** *as plt # cách 2*

https://matplotlib.org/

2. Biểu đồ thông dụng với Matplotlib

Thiết lập thông số cấu hình chung cho biểu đồ

```
plt.rcParams['figure.figsize'] = (10,8)
plt.rcParams['figure.dpi'] = 200
plt.rcParams['font.size'] = 13
# plt.rcParams['savefig.dpi'] = 200
# plt.rcParams['legend.fontsize'] = 'large'
# plt.rcParams['figure.titlesize'] = 'medium'
# plt.rcParams["legend.loc"] = 'best'
```

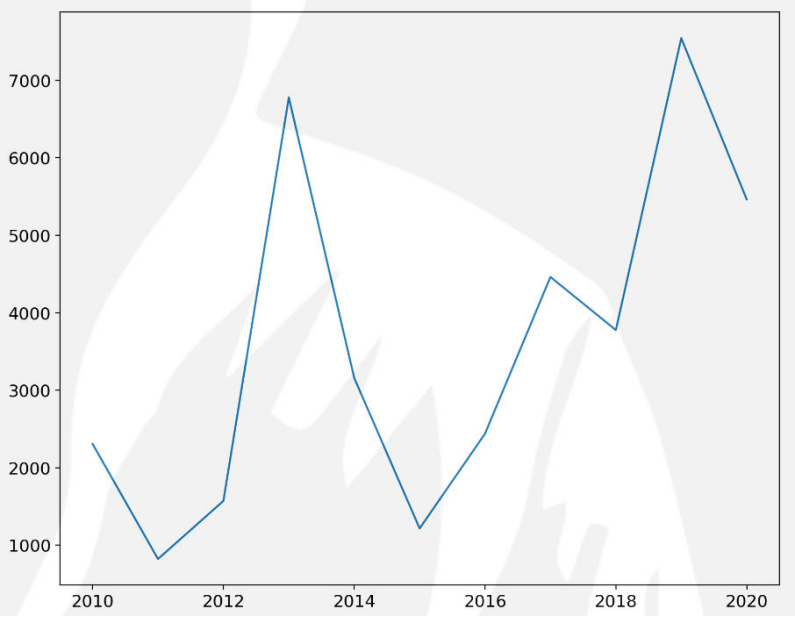
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ đường

```
df = pd.read_csv('./data/NetProfit.csv')
dat = df[['Year', 'VIC']]
print(dat)
```

	Year	VIC
0	2010	2306.9
1	2011	821.3
2	2012	1571.3
3	2013	6779.5
4	2014	3158.6
5	2015	1215.7
6	2016	2439.5
7	2017	4462.4
8	2018	3776.7
9	2019	7545.9
10	2020	5464.6

```
plt.plot('Year', 'VIC', data=dat)
plt.show()
```



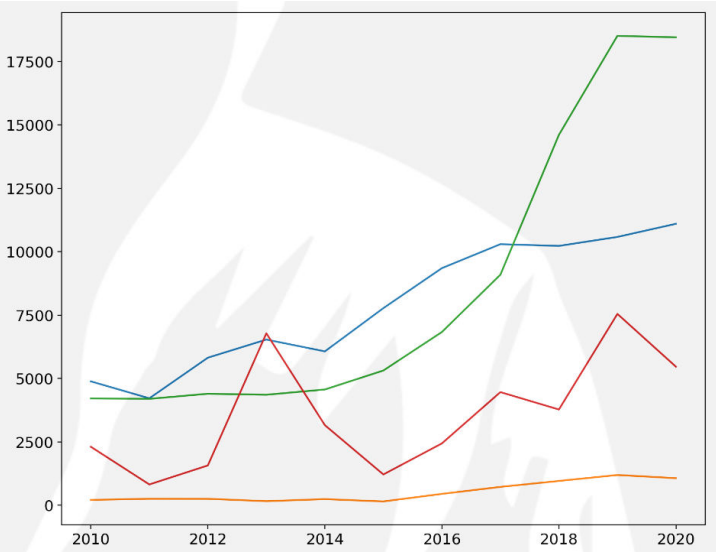
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ đường

```
df = pd.read_csv('./data/NetProfit.csv')
print(df)
```

	Year	VNM	PNJ	VCB	VIC
0	2010	4886.2	212.0	4214.5	2306.9
1	2011	4218.2	257.1	4196.8	821.3
2	2012	5819.5	254.4	4397.5	1571.3
3	2013	6534.1	163.2	4358.1	6779.5
4	2014	6068.8	242.5	4565.6	3158.6
5	2015	7773.4	152.3	5313.9	1215.7
6	2016	9350.3	449.6	6831.7	2439.5
7	2017	10295.7	724.9	9091.1	4462.4
8	2018	10227.3	959.9	14605.6	3776.7
9	2019	10581.2	1193.9	18510.9	7545.9
10	2020	11098.9	1069.3	18451.3	5464.6

```
plt.plot('Year', 'VNM', data=df)
plt.plot('Year', 'PNJ', data=df)
plt.plot('Year', 'VCB', data=df)
plt.plot('Year', 'VIC', data=df)
plt.show()
```

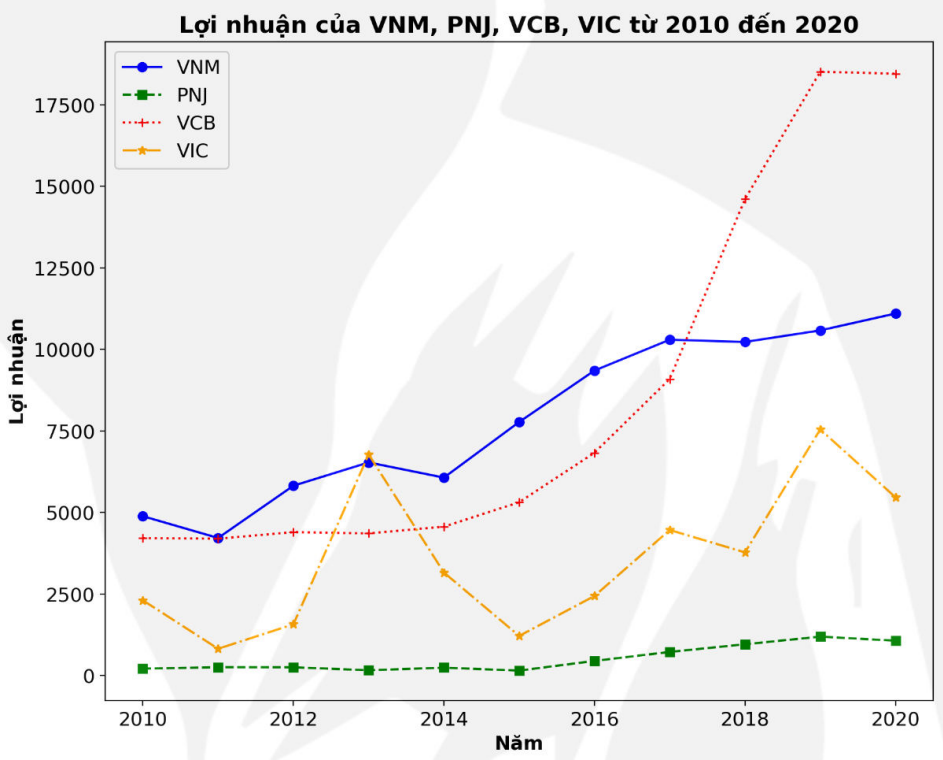


2. Biểu đồ thông dụng với Matplotlib

Biểu đồ đường

```
plt.plot('Year', 'VNM', data=df, color='b', linestyle='-', marker='o')
plt.plot('Year', 'PNJ', data=df, color='g', linestyle='--', marker='s')
plt.plot('Year', 'VCB', data=df, color='#FF0000', linestyle=':', marker='+')
plt.plot('Year', 'VIC', data=df, color='orange', linestyle='-.', marker='*')
plt.title("Lợi nhuận của VNM, PNJ, VCB, VIC từ 2010 đến 2020", fontweight='bold')
plt.xlabel("Năm", fontweight='bold')
plt.ylabel("Lợi nhuận", fontweight='bold')
plt.legend()
plt.show()
```

	Year	VNM	PNJ	VCB	VIC
0	2010	4886.2	212.0	4214.5	2306.9
1	2011	4218.2	257.1	4196.8	821.3
2	2012	5819.5	254.4	4397.5	1571.3
3	2013	6534.1	163.2	4358.1	6779.5
4	2014	6068.8	242.5	4565.6	3158.6
5	2015	7773.4	152.3	5313.9	1215.7
6	2016	9350.3	449.6	6831.7	2439.5
7	2017	10295.7	724.9	9091.1	4462.4
8	2018	10227.3	959.9	14605.6	3776.7
9	2019	10581.2	1193.9	18510.9	7545.9
10	2020	11098.9	1069.3	18451.3	5464.6



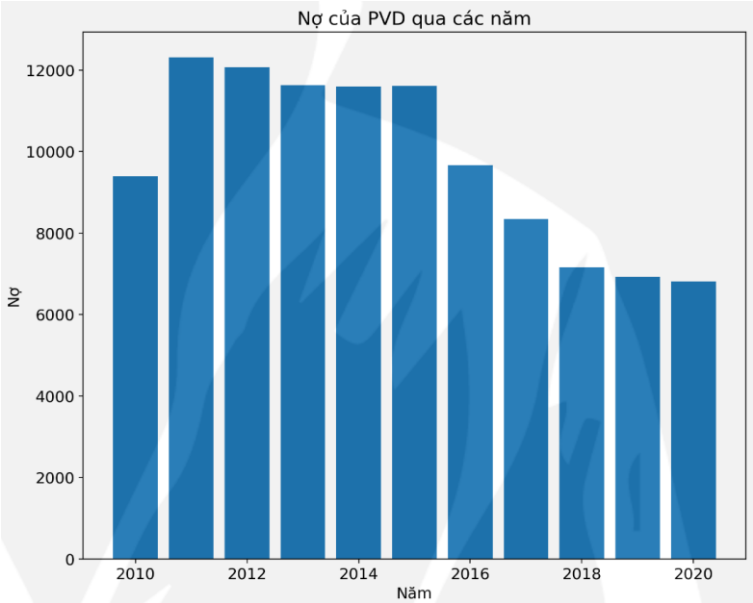
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ cột

```
df = pd.read_csv('./data/PVD_Asset.csv')
print(df)
```

	Year	Liabilities	Equity
0	2010	9396.8	5226.9
1	2011	12313.7	6202.2
2	2012	12066.5	6992.1
3	2013	11624.5	9838.2
4	2014	11591.7	11478.6
5	2015	11611.5	13303.7
6	2016	9667.3	13475.4
7	2017	8344.5	13472.9
8	2018	7153.6	13850.2
9	2019	6923.3	13968.5
10	2020	6814.2	14042.0

```
plt.bar('Year', 'Liabilities', data=df)
plt.title("Nợ của PVD qua các năm")
plt.xlabel("Năm")
plt.ylabel("Nợ")
plt.show()
```



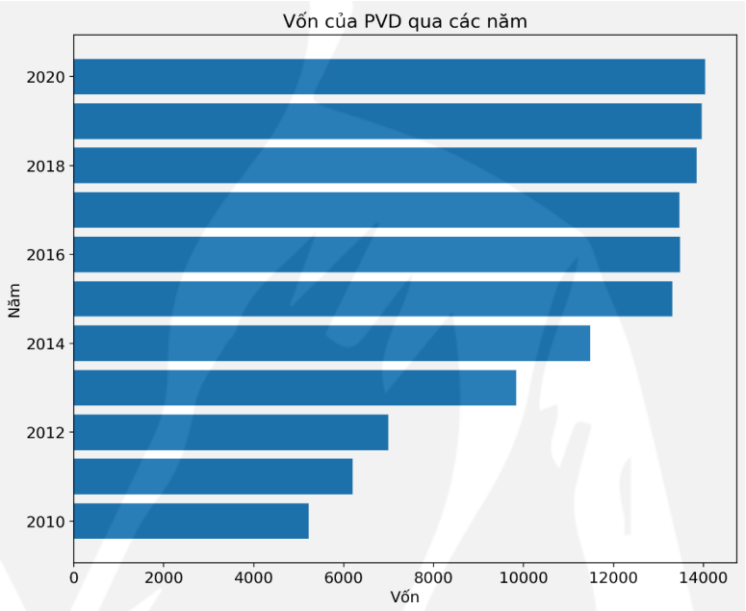
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ cột

```
df = pd.read_csv('./data/PVD_Asset.csv')
print(df)
```

	Year	Liabilities	Equity
0	2010	9396.8	5226.9
1	2011	12313.7	6202.2
2	2012	12066.5	6992.1
3	2013	11624.5	9838.2
4	2014	11591.7	11478.6
5	2015	11611.5	13303.7
6	2016	9667.3	13475.4
7	2017	8344.5	13472.9
8	2018	7153.6	13850.2
9	2019	6923.3	13968.5
10	2020	6814.2	14042.0

```
plt.barh('Year', 'Equity', data=df)
plt.title("Vốn của PVD qua các năm")
plt.xlabel("Vốn")
plt.ylabel("Năm")
plt.show()
```

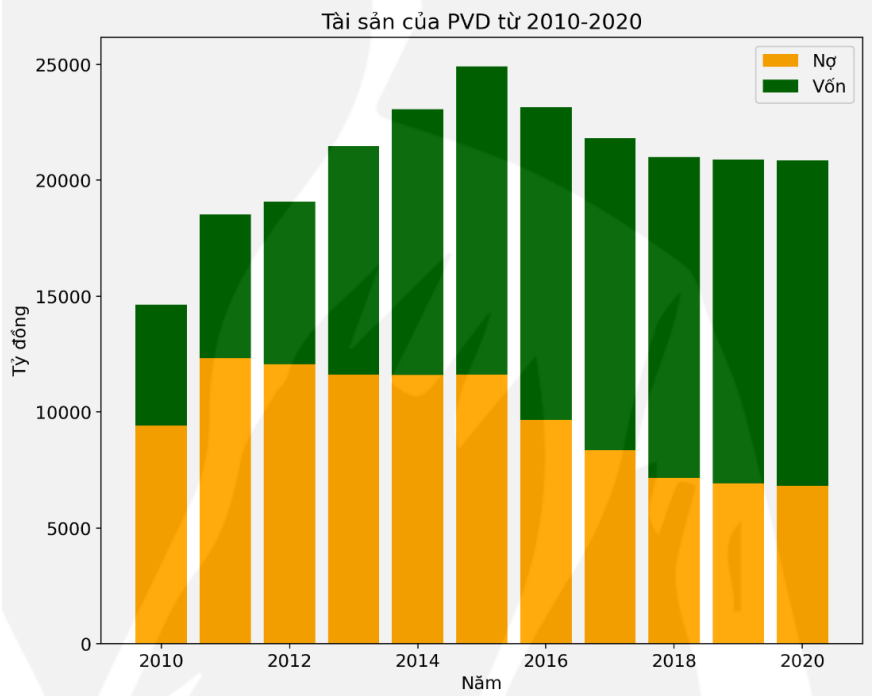


2. Biểu đồ thông dụng với Matplotlib

Biểu đồ cột

	Year	Liabilities	Equity
0	2010	9396.8	5226.9
1	2011	12313.7	6202.2
2	2012	12066.5	6992.1
3	2013	11624.5	9838.2
4	2014	11591.7	11478.6
5	2015	11611.5	13303.7
6	2016	9667.3	13475.4
7	2017	8344.5	13472.9
8	2018	7153.6	13850.2
9	2019	6923.3	13968.5
10	2020	6814.2	14042.0

```
plt.bar('Year', 'Liabilities', data=df, color='orange', label="Nợ")
plt.bar('Year', 'Equity', data=df, bottom='Liabilities',
color='darkgreen', label="Vốn")
plt.title("Tài sản của PVD từ 2010-2020")
plt.xlabel("Năm")
plt.ylabel("Tỷ đồng")
plt.legend()
plt.show()
```



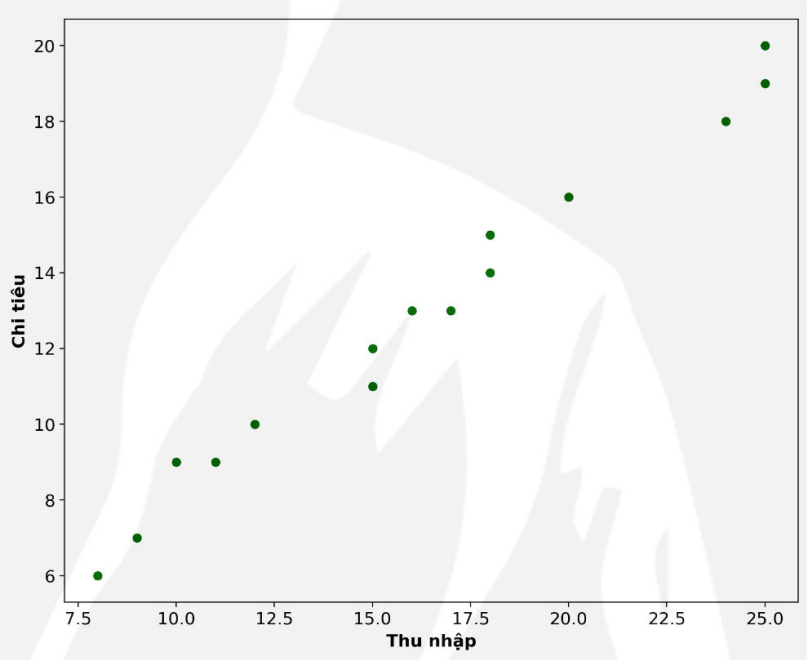
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ phân tán

```
df = pd.read_csv('./data/Income.csv')
print(df)
```

	Income	Expenditure
0	8	6
1	9	7
2	10	9
3	11	9
4	12	10
5	15	12
6	15	11
7	16	13
8	17	13
9	18	15
10	18	14
11	20	16
12	24	18
13	25	20
14	25	19

```
plt.scatter('Income', 'Expenditure', data=df, color='darkgreen')
plt.xlabel('Thu nhập', fontweight='bold')
plt.ylabel('Chi tiêu', fontweight='bold')
plt.show()
```

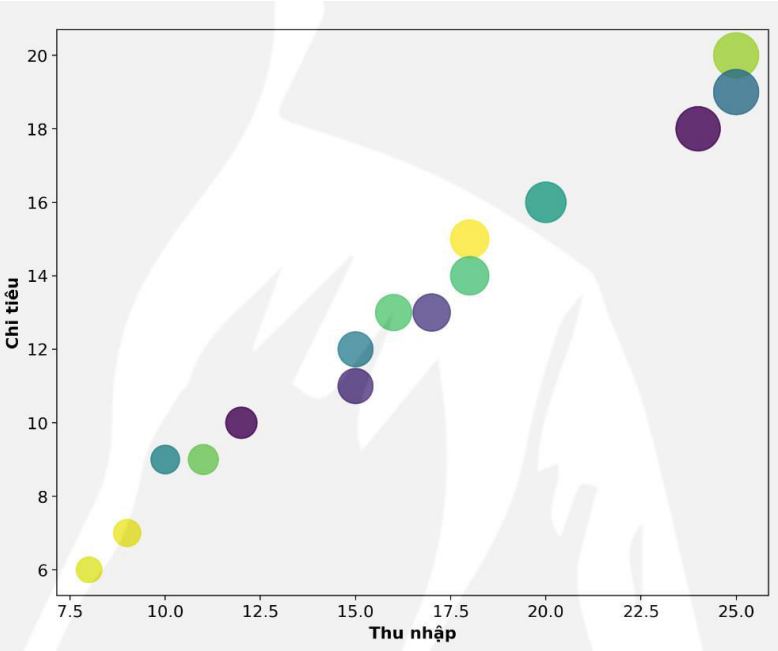


2. Biểu đồ thông dụng với Matplotlib

Biểu đồ phân tán

	Income	Expenditure
0	8	6
1	9	7
2	10	9
3	11	9
4	12	10
5	15	12
6	15	11
7	16	13
8	17	13
9	18	15
10	18	14
11	20	16
12	24	18
13	25	20
14	25	19

```
colors = np.random.rand(df.shape[0]) # Random màu ngẫu nhiên
area = df['Income'].values * 50 # Kích thước điểm dữ liệu
plt.scatter('Income', 'Expenditure', data=df, c=colors,
s=area,alpha=0.8)
plt.xlabel('Thu nhập', fontweight='bold')
plt.ylabel('Chi tiêu', fontweight='bold')
plt.show()
```



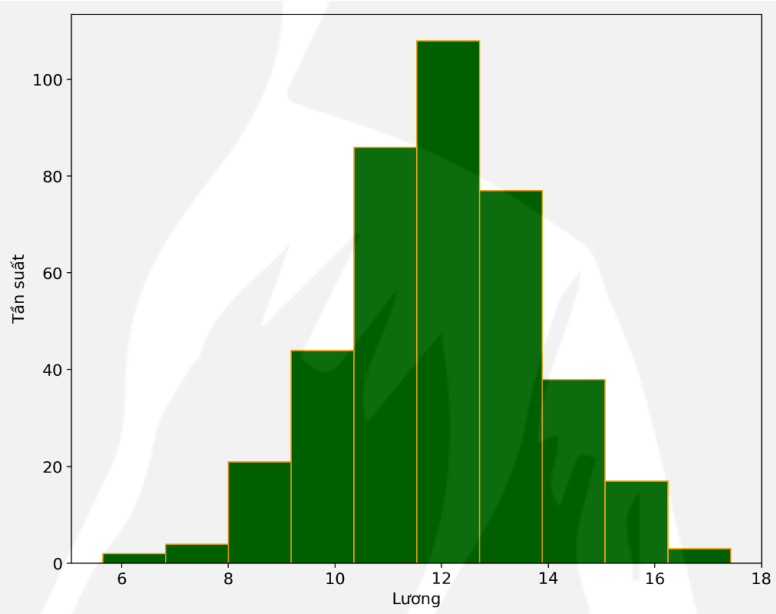
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ tần suất

Biểu đồ tần suất (histogram) là một dạng biểu đồ cột cho thấy *sự thay đổi, biến động* của một tập hợp dữ liệu theo những *hình dạng nhất định*.

Vd: vẽ biểu đồ tần suất thể hiện mức lương của 400 nhân viên (lương trung bình 12 triệu đồng, mức chênh lệch 2 triệu đồng).

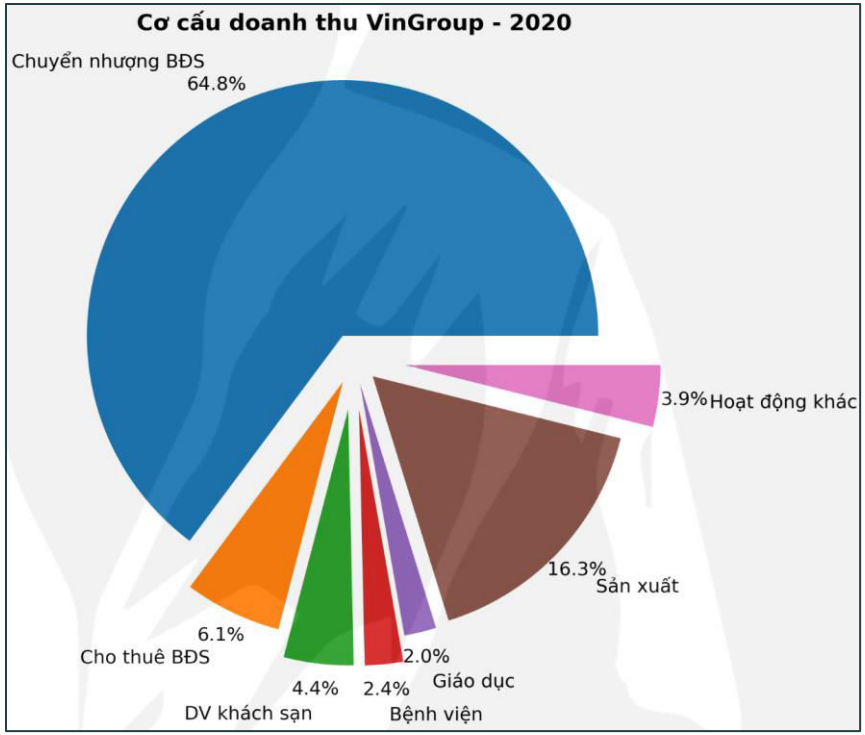
```
dat = np.random.normal(12, 2, 400)
plt.hist(dat, color='darkgreen',
edgecolor='orange')
plt.xlabel('Lương')
plt.ylabel('Tần suất')
plt.show()
```



2. Biểu đồ thông dụng với Matplotlib

Biểu đồ tròn

```
lbls = ['Chuyển nhượng BĐS', 'Cho thuê BĐS', 'DV khách sạn', 'Bệnh viện', 'Giáo dục', 'Sản xuất', 'Hoạt động khác']
income = [71.576, 6.788, 4.869, 2.675, 2.244, 18.007, 4.304]
explode = [0.1, 0.1, 0.2, 0.2, 0.1, 0.1, 0.2]
plt.pie(income, labels= lbls, explode=explode, autopct='%1.1f%%', pctdistance=1.1, labeldistance=1.2)
plt.title('Cơ cấu doanh thu VinGroup - 2020', fontweight='bold')
# plt.legend(loc='upper right')
plt.show()
```

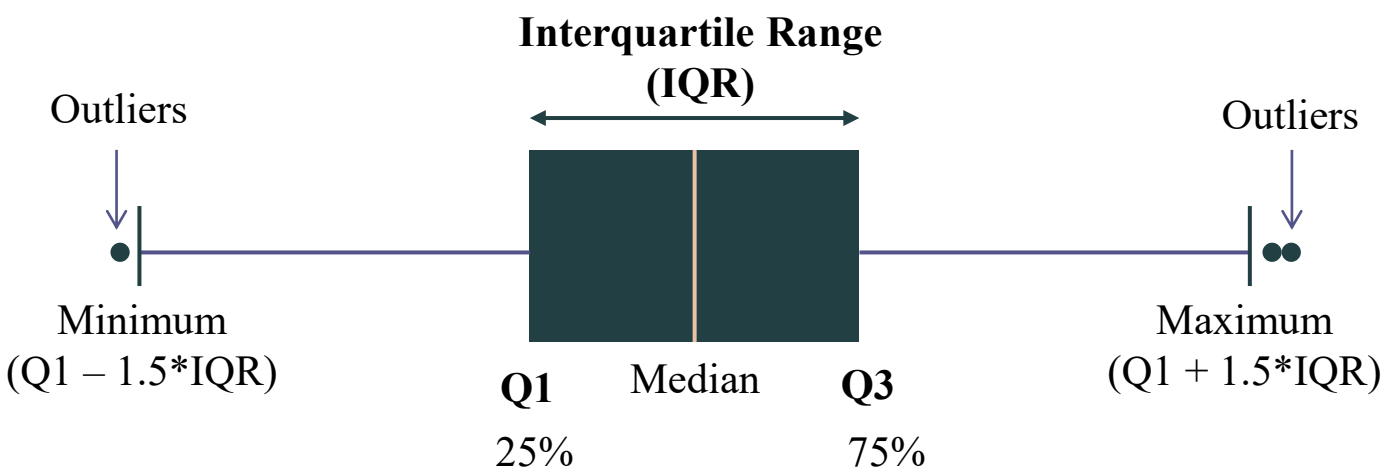


2. Biểu đồ thông dụng với Matplotlib

Biểu đồ Boxplot

Boxplot là một dạng biểu đồ thể hiện *phân phối dữ liệu*, cho biết *độ dàn trải các điểm dữ liệu*, *dữ liệu có đối xứng không*, *phân bố rộng hay hẹp*, *giá trị min-max* và *các điểm ngoại lệ*.

Dữ liệu được phân phối theo phạm vi **"tứ phân vị"** chia dữ liệu thành 4 phần:



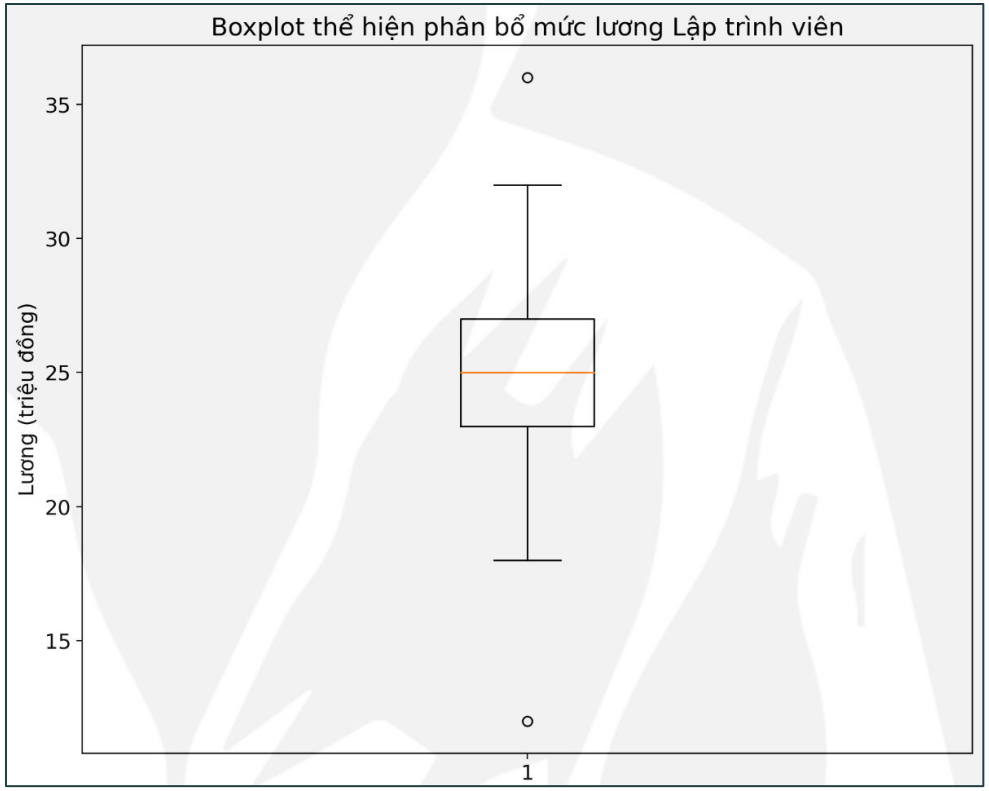
2. Biểu đồ thông dụng với Matplotlib

Biểu đồ Boxplot

Vd: xét dữ liệu khảo sát mức lương của lập trình viên như sau:

```
dat = pd.read_csv('./data/Salary_of_Developer.csv')
print(dat)
```

```
plt.boxplot(dat)
plt.ylabel("Lương (triệu đồng)")
plt.title("Boxplot thể hiện phân bổ mức lương Lập trình viên")
plt.show()
```



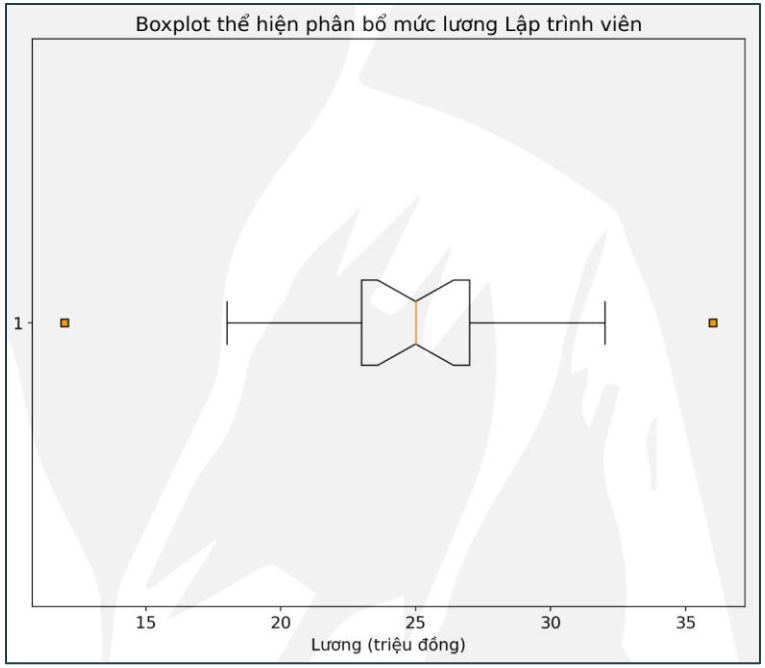
	Salary
0	25
1	23
2	23
3	26
4	27
5	22
6	18
7	12
8	27
9	26
10	28
11	22
12	26
13	25
14	24
15	24
16	28
17	32
18	36
19	25

2. Biểu đồ thông dụng với Matplotlib

Biểu đồ Boxplot

Vd: xét dữ liệu khảo sát mức lương của lập trình viên như sau:

```
orange_square = dict(markerfacecolor='orange', marker='s')
plt.boxplot(dat, notch=True, flierprops=orange_square, vert=False)
plt.xlabel("Lương (triệu đồng)")
plt.title("Boxplot thể hiện phân bổ mức lương Lập trình viên")
plt.show()
```

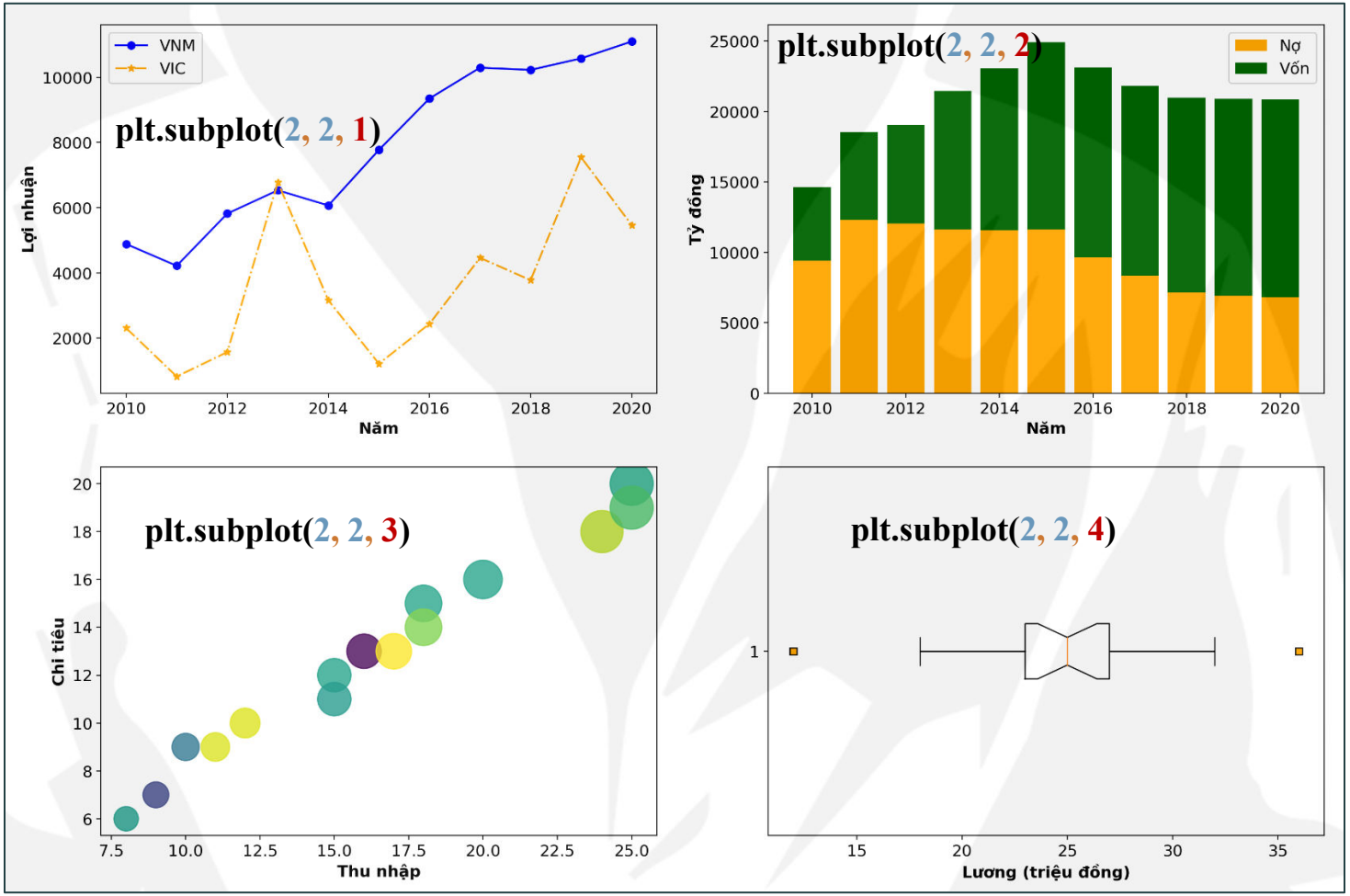


	Salary
0	25
1	23
2	23
3	26
4	27
5	22
6	18
7	12
8	27
9	26
10	28
11	22
12	26
13	25
14	24
15	24
16	28
17	32
18	36
19	25

2. Biểu đồ thông dụng với Matplotlib

Biểu đồ kết hợp

plt.subplot(rows, columns, index)



3. Giới thiệu Seaborn

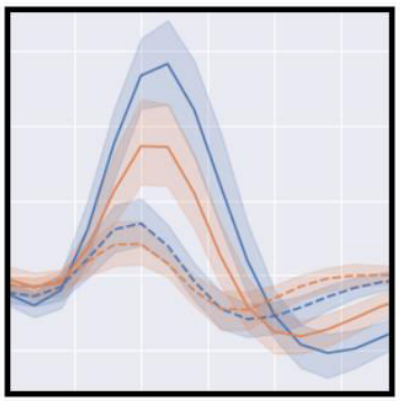
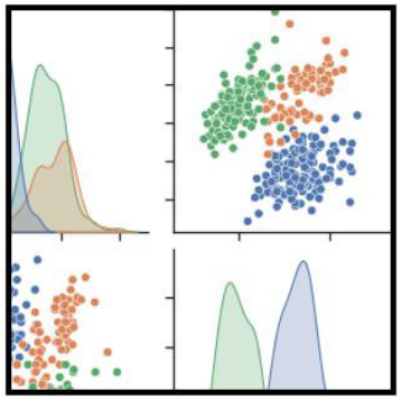
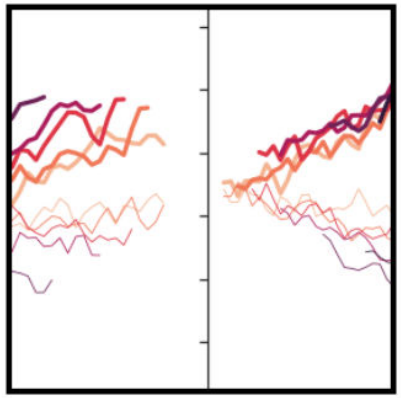
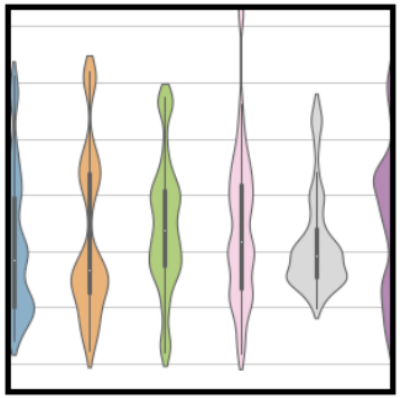
Seaborn là một thư viện được xây dựng dựa trên nền tảng của Matplotlib, phục vụ chính cho mục đích vẽ đồ thị trực quan hóa dữ liệu.

Cài đặt: *pip install **seaborn***

(trong môi trường Anaconda, seaborn đã được tích hợp)

Sử dụng: *import **seaborn** as sns*

https://seaborn.pydata.org/

3. Giới thiệu Seaborn

Thiết lập cấu hình chung

```
import seaborn as sns
sns.set(style='darkgrid') # Thiết lập style cho biểu đồ
import ssl
# Cấu hình ssl cho phép tải dữ liệu mẫu thông qua thư viện
ssl._create_default_https_context = ssl._create_unverified_context
```

```
# Show sample datasets
sample_datasets = sns.get_dataset_names()
print(sample_datasets)
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds',
'dots', 'exercise', 'flights', 'fmri', 'gammas', 'geyser', 'iris', 'mpg',
'penguins', 'planets', 'tips', 'titanic']
```

4. Các loại biểu đồ trong Seaborn

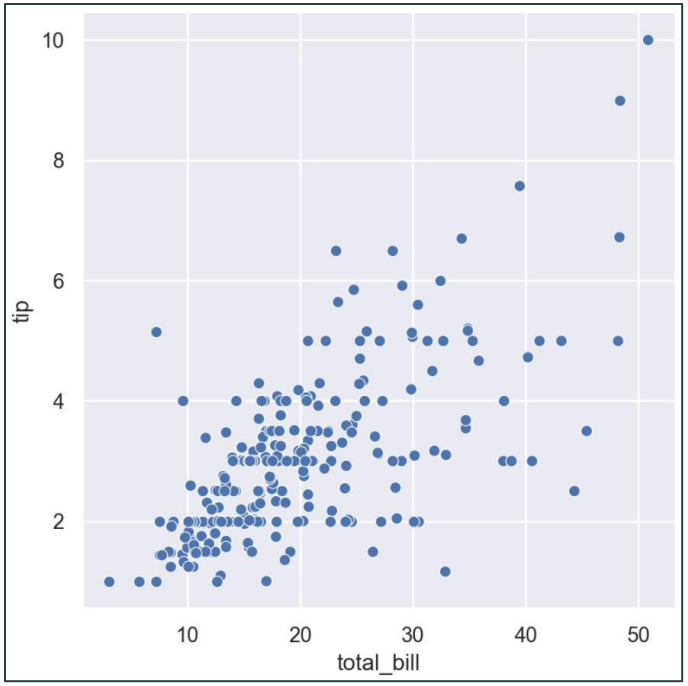
Biểu đồ với biến liên tục

relplot()

```
tips = sns.load_dataset("tips")
print(tips.head())
```

```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

```
sns.relplot(x='total_bill',
y='tip', data=tips)
```



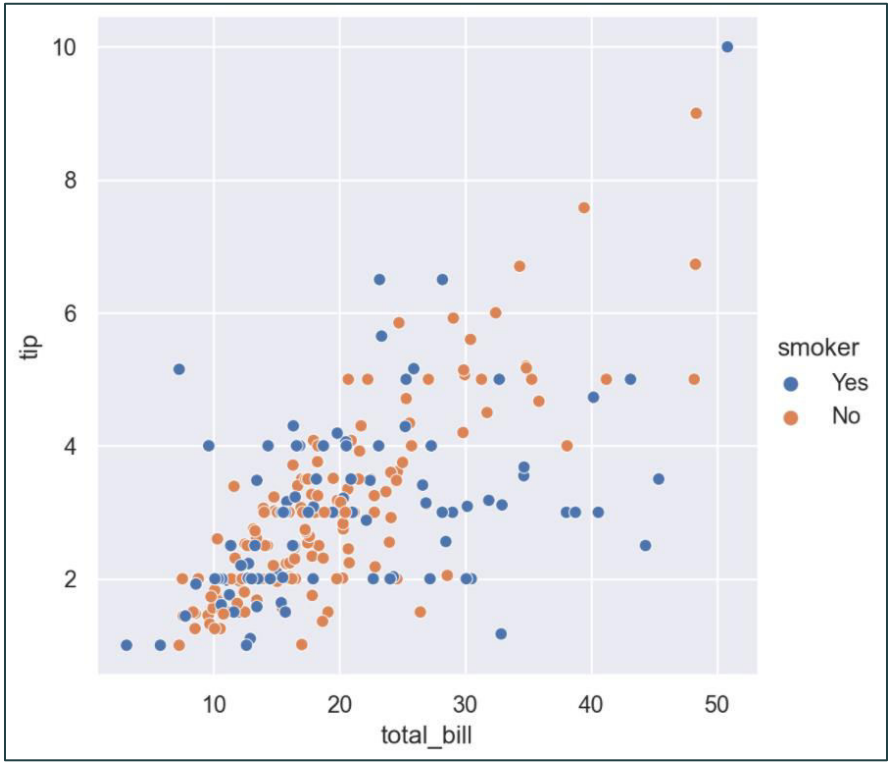
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến liên tục

- **relplot()**

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
sns.relplot(x='total_bill',
y='tip', data=tips,
hue='smoker')
```



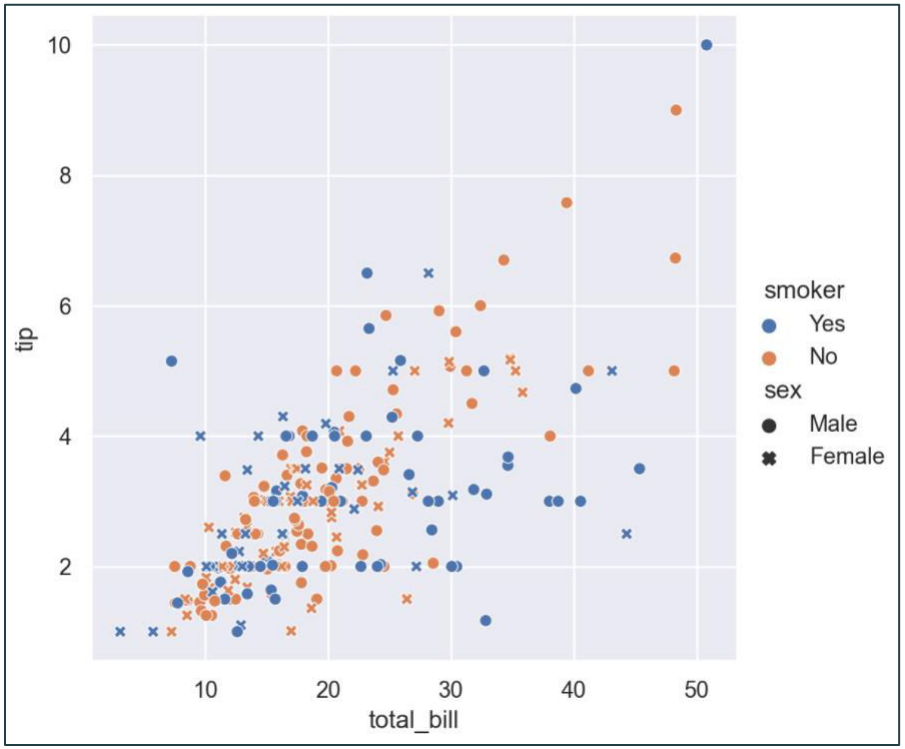
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến liên tục

- **relplot()**

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
sns.relplot(x='total_bill',
y='tip', data=tips,
hue='smoker', style='sex')
```



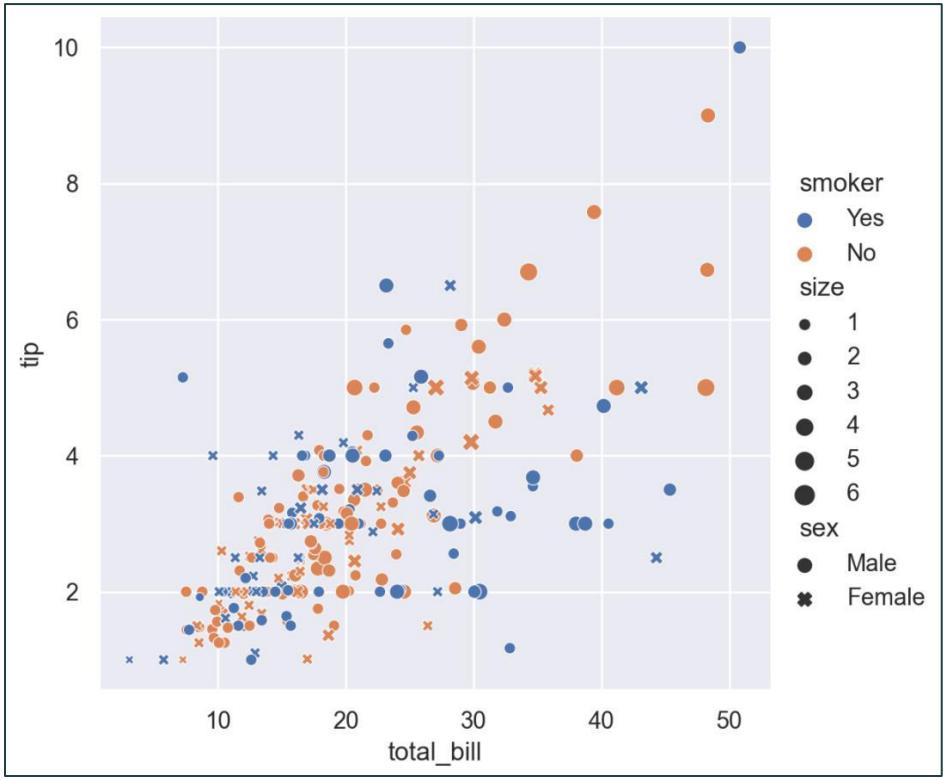
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến liên tục

- **relplot()**

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
sns.relplot(x='total_bill',
y='tip', data=tips,
hue='smoker', style='sex',
size='size')
```



4. Các loại biểu đồ trong Seaborn

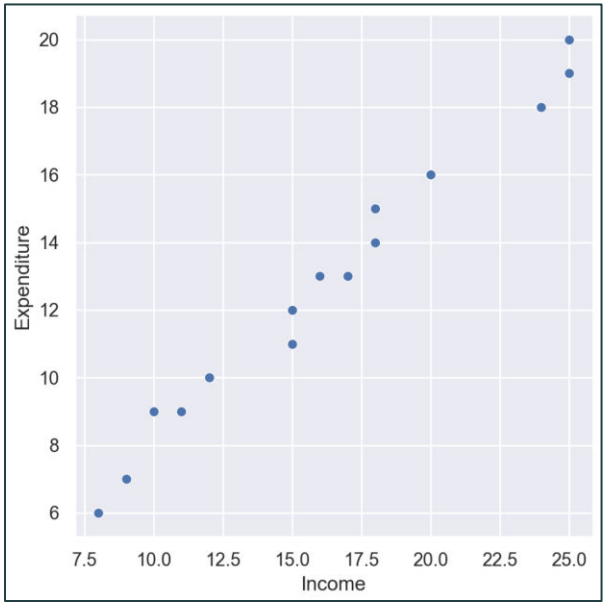
Biểu đồ với biến liên tục

➢ relplot()

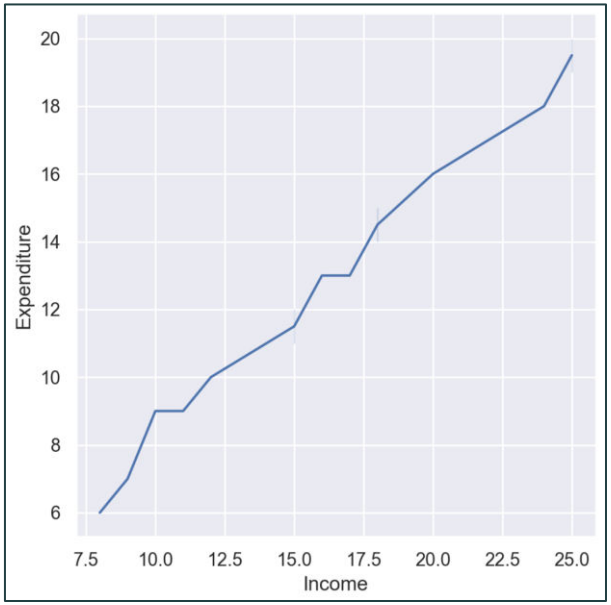
```
dat = pd.read_csv('./data/Income.csv')
print(dat)
```

	Income	Expenditure
0	8	6
1	9	7
2	10	9
3	11	9
4	12	10
5	15	12
6	15	11
7	16	13
8	17	13
9	18	15
10	18	14
11	20	16
12	24	18
13	25	20
14	25	19

```
sns.relplot(x='Income', y='Expenditure', data=dat,
kind='scatter')
```



kind='scatter'



kind='line'

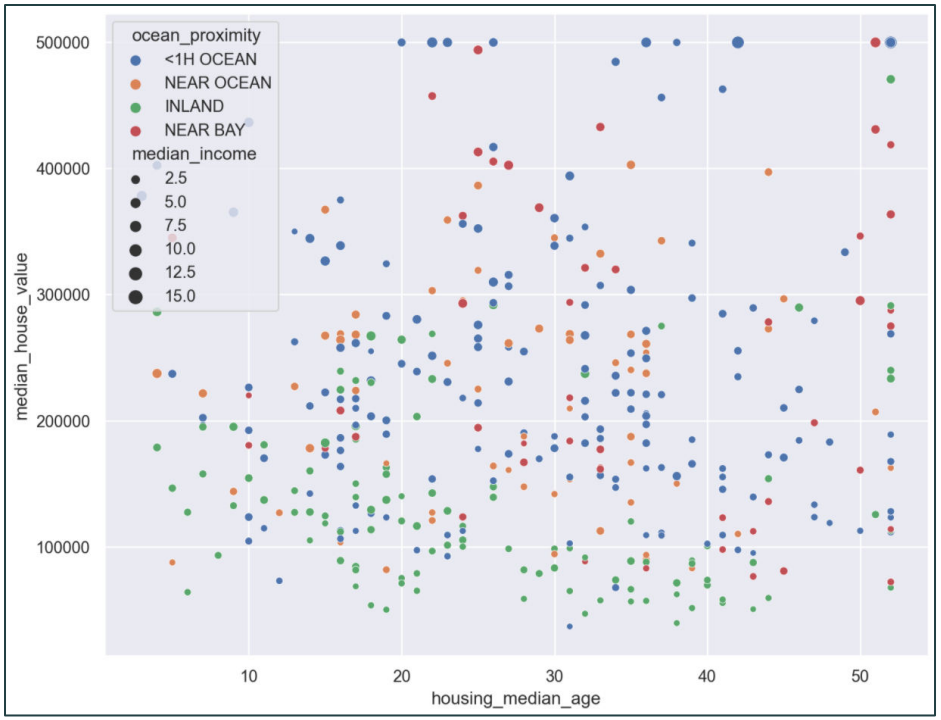
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến liên tục

- **scatterplot()**

  relplot() với kind='scatter'

```
df = pd.read_csv('./data/Housing.csv')
# Randomly select 400 rows from the df
dat = df.sample(400)  df: {DataFrame: (20640, 10)}
sns.scatterplot(x='housing_median_age',
  y='median_house_value', data=dat, hue='ocean_proximity',
  size='median_income')  dat: {DataFrame: (400, 10)}
```



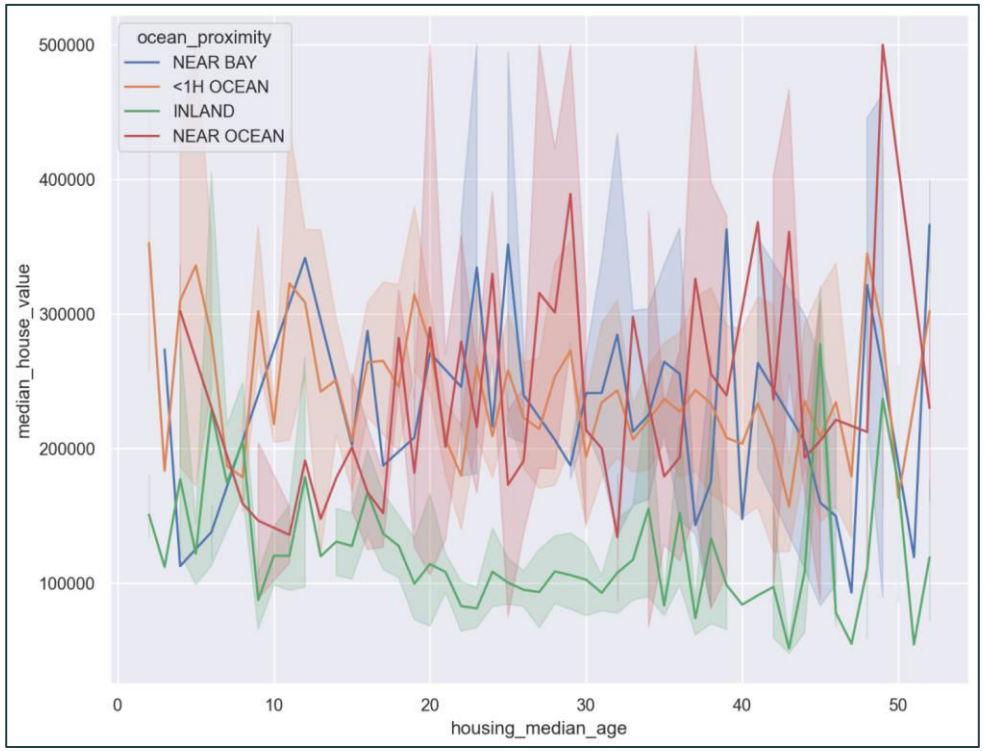
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến liên tục

- **lineplot()**

relplot() với kind= 'line'

```
df = pd.read_csv('./data/Housing.csv')
# Randomly select 1000 rows from the df
dat = df.sample(1000)  df: {DataFrame: (20640, 10)}
sns.lineplot(x='housing_median_age', y='median_house_value',
  data=dat, hue='ocean_proximity')  dat: {DataFrame: (1000, 10)}
```



4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến phân loại

- **catplot()**

<table>
<tr><th>Phân tán</th><th>Phân phối</th><th>Ước tính</th></tr>
<tr><td rowspan="2">stripplot() → catplot() với kind=‘strip’</td><td>boxplot() → catplot() với kind=‘box’</td><td rowspan="2">pointplot() → catplot() với kind=‘point’</td></tr>
<tr><td rowspan="2">violinplot() → catplot() với kind=‘violin’</td></tr>
<tr><td rowspan="2">swarmplot() → catplot() với kind=‘warm’</td><td rowspan="2">barplot() → catplot() với kind=‘bar’</td></tr>
<tr><td>boxenplot() → catplot() với kind=‘boxen’</td></tr>
</table>

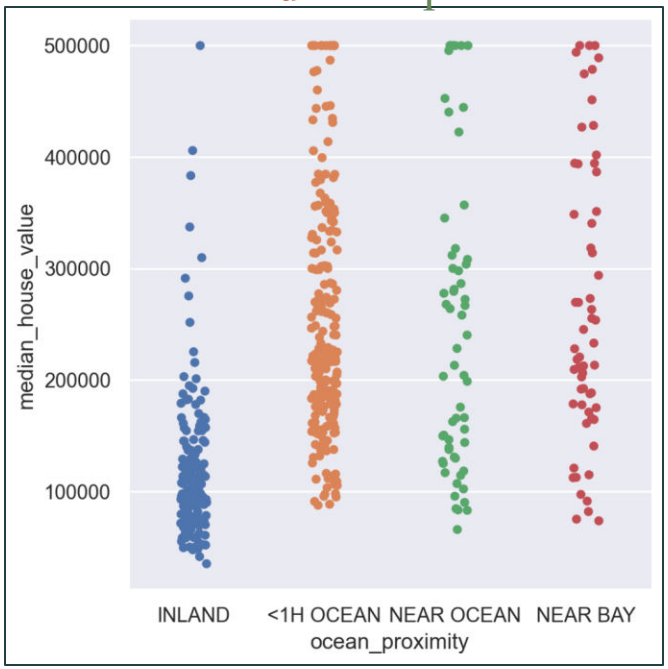
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến phân loại

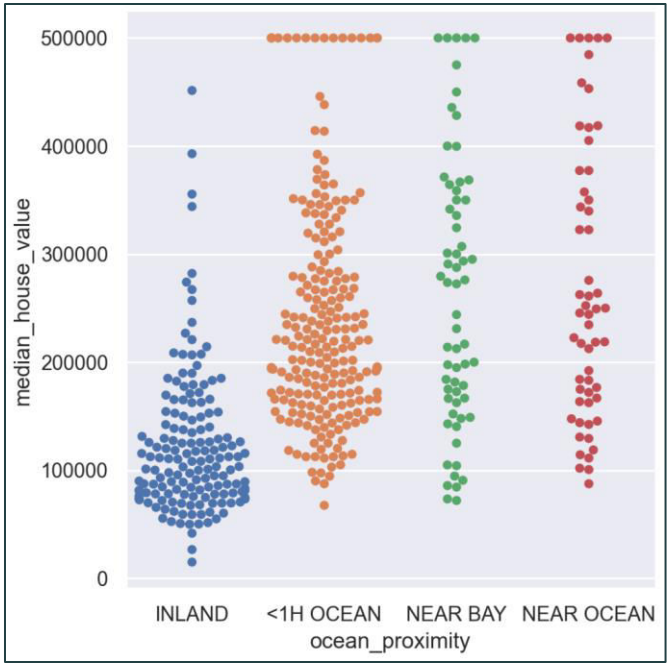
- **catplot()**

```
df = pd.read_csv('./data/Housing.csv')
# Randomly select 500 rows from the df
dat = df.sample(500)  df: {DataFrame: (20640, 10)}
sns.catplot(x='ocean_proximity', y='median_house_value',
  data=dat)  dat: {DataFrame: (500, 10)}
```

kind = 'strip'



kind = 'swarm'



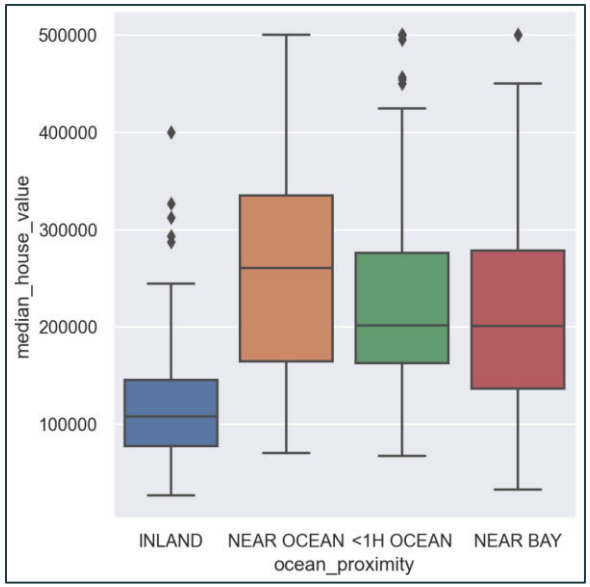
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến phân loại

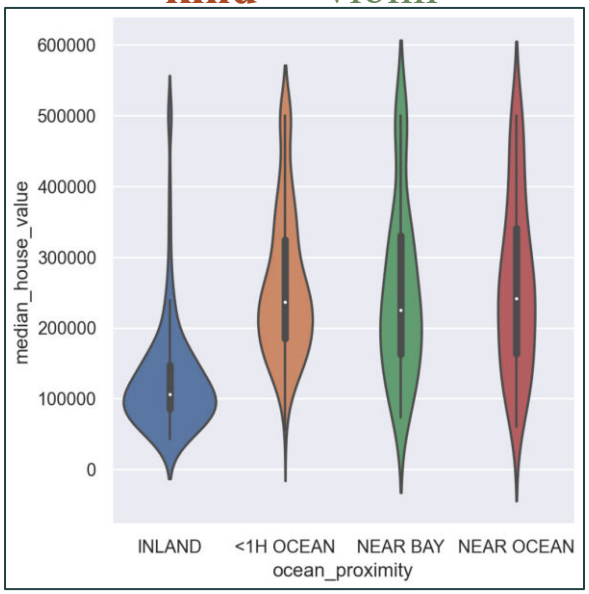
- **catplot()**

```
df = pd.read_csv('./data/Housing.csv')
# Randomly select 500 rows from the df
dat = df.sample(500)  df: {DataFrame: (20640, 10)}
sns.catplot(x='ocean_proximity', y='median_house_value',
  data=dat, kind='box')  dat: {DataFrame: (500, 10)}
```

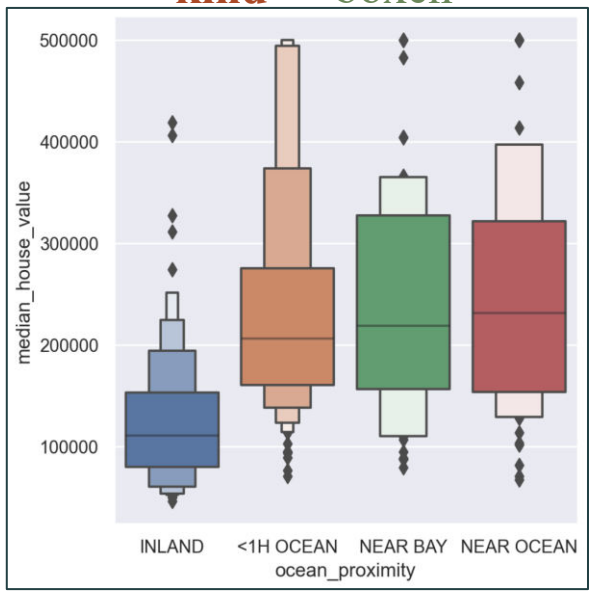
kind = 'box'



kind = 'violin'



kind = 'boxen'



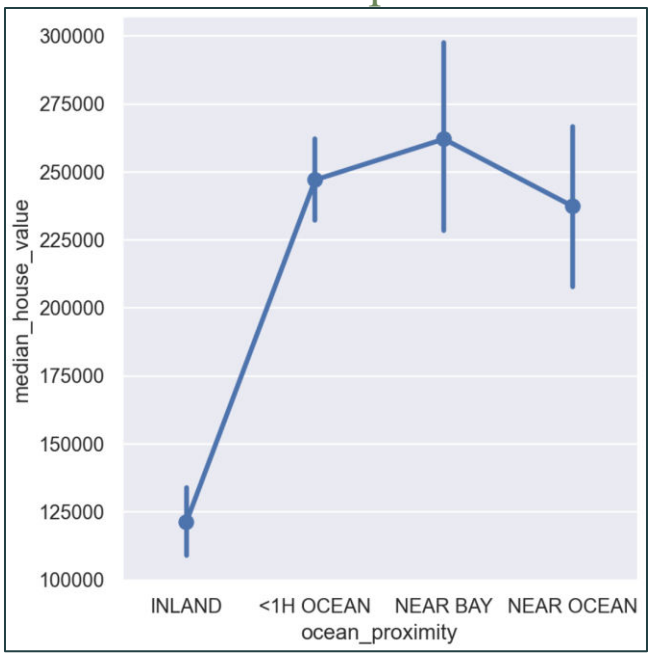
4. Các loại biểu đồ trong Seaborn

Biểu đồ với biến phân loại

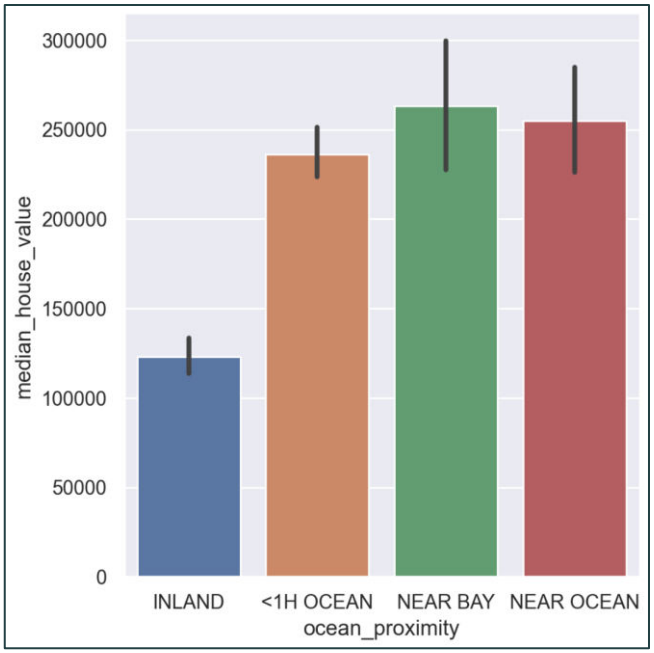
- **catplot()**

```
df = pd.read_csv('./data/Housing.csv')
# Randomly select 500 rows from the df
dat = df.sample(500)  df: {DataFrame: (20640, 10)}
sns.catplot(x='ocean_proximity', y='median_house_value',
  data=dat, kind='point')  dat: {DataFrame: (500, 10)}
```

kind = 'point'



kind = 'bar'



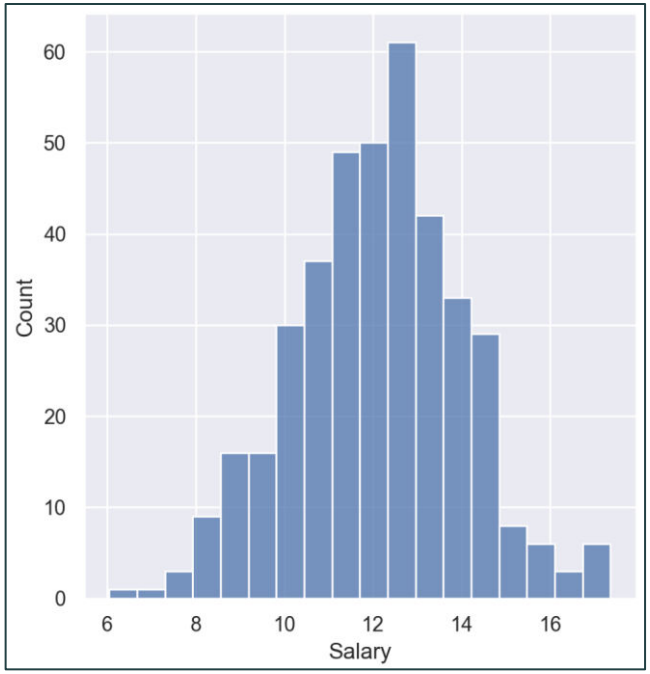
4. Các loại biểu đồ trong Seaborn

Biểu đồ phân phối (tần suất)

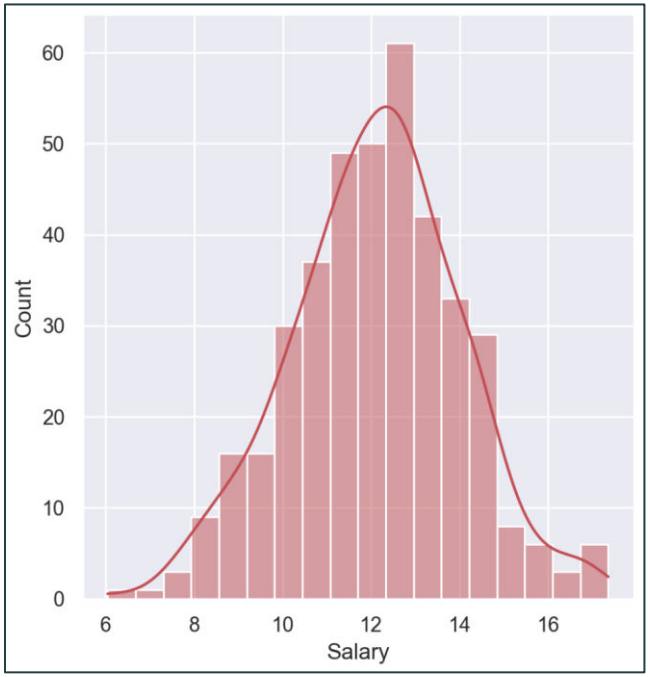
➢ distplot()

```
np.random.seed(10) # -> Random với giá trị không đổi
dat = np.random.normal(12, 2, 400)
```

```
sns.displot(dat)
plt.xlabel('Salary')
```




```
sns.displot(dat, kde=True, color='r')
plt.xlabel('Salary')
```

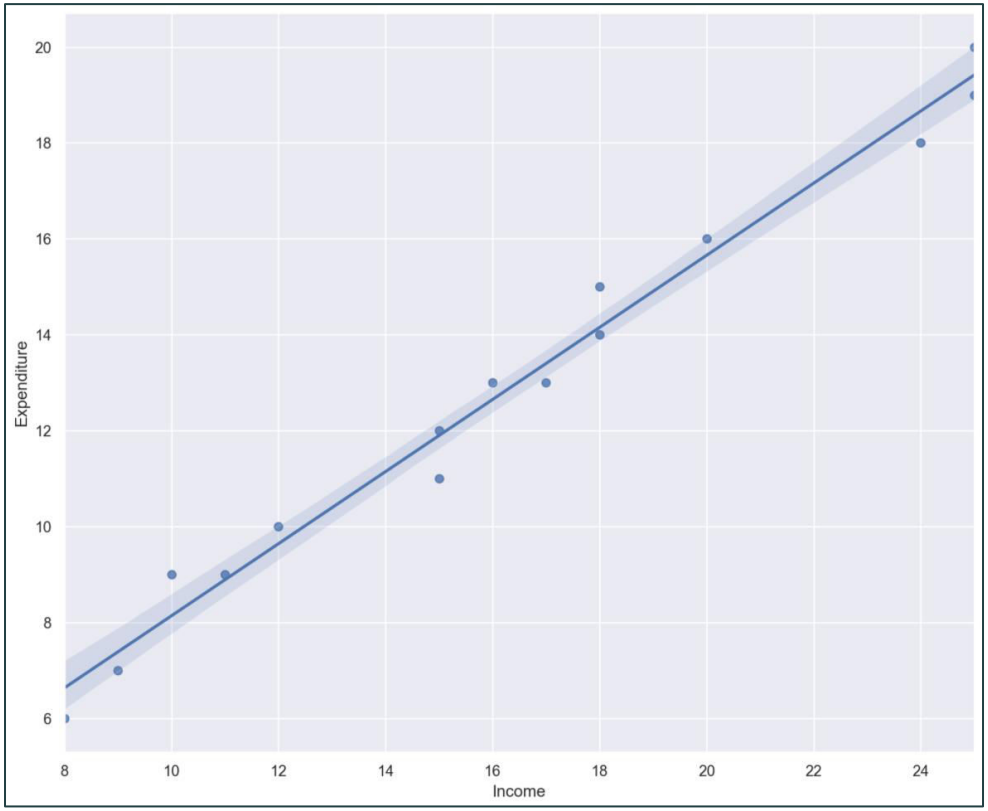


4. Các loại biểu đồ trong Seaborn

Biểu đồ hồi quy

- **regplot()**

```
df = pd.read_csv('./data/Income.csv')
sns.regplot(x='Income', y='Expenditure', data=df)
```

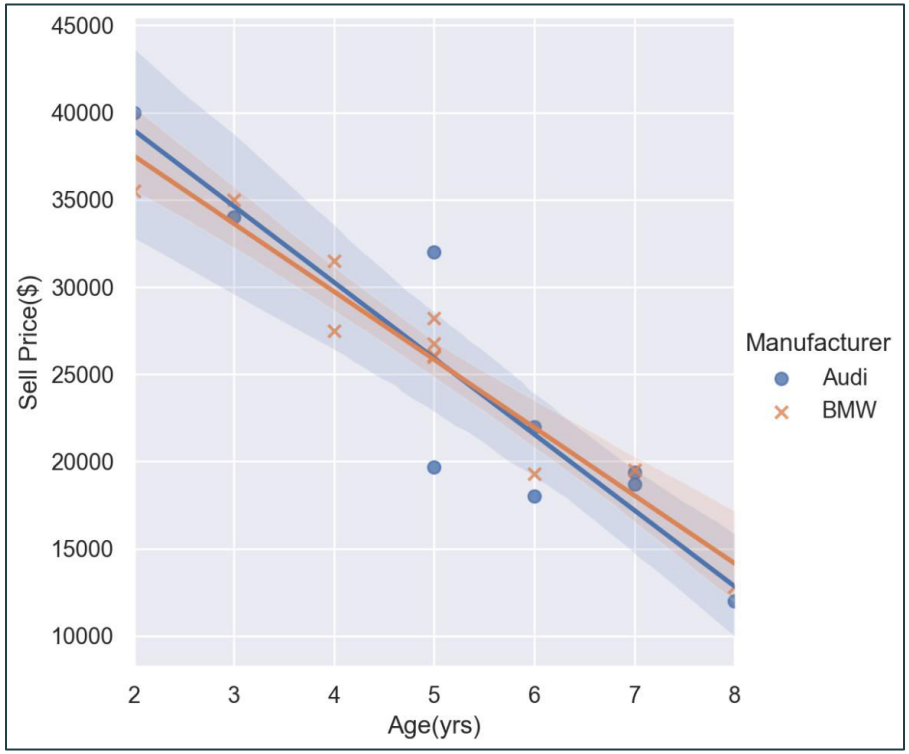


4. Các loại biểu đồ trong Seaborn

Biểu đồ hồi quy

- **lmplot()**

```
df = pd.read_csv('./data/Car_Prices.csv')
sns.lmplot(x='Age(yrs)', y='Sell Price($)', data=df, hue='Manufacturer',
markers=['o', 'x'])
```



4. Các loại biểu đồ trong Seaborn

Biểu đồ nhiệt

- **heatmap()**

```
flights_long = sns.load_dataset('flights')
flights = flights_long.pivot('month', 'year', 'passengers')
```

flights_long

```
     year month  passengers
0    1949   Jan         112
1    1949   Feb         118
2    1949   Mar         132
3    1949   Apr         129
4    1949   May         121
..    ...   ...         ...
139  1960   Aug         606
140  1960   Sep         508
141  1960   Oct         461
142  1960   Nov         390
143  1960   Dec         432

[144 rows x 3 columns]
```

flights

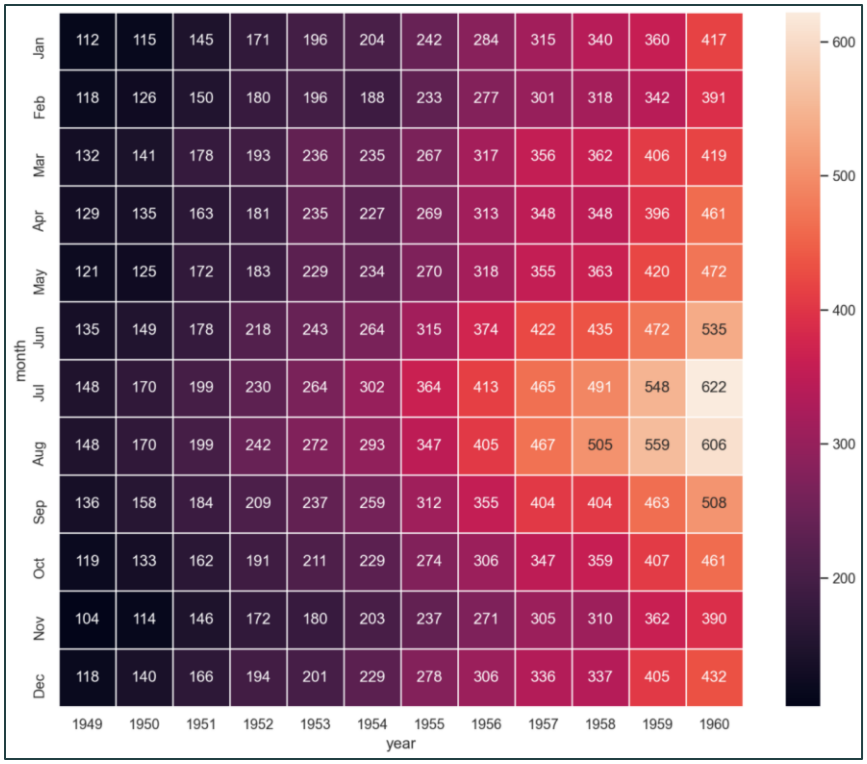
	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960
Jan	112	115	145	171	196	204	242	284	315	340	360	417
Feb	118	126	150	180	196	188	233	277	301	318	342	391
Mar	132	141	178	193	236	235	267	317	356	362	406	419
Apr	129	135	163	181	235	227	269	313	348	348	396	461
May	121	125	172	183	229	234	270	318	355	363	420	472
Jun	135	149	178	218	243	264	315	374	422	435	472	535
Jul	148	170	199	230	264	302	364	413	465	491	548	622
Aug	148	170	199	242	272	293	347	405	467	505	559	606
Sep	136	158	184	209	237	259	312	355	404	404	463	508
Oct	119	133	162	191	211	229	274	306	347	359	407	461
Nov	104	114	146	172	180	203	237	271	305	310	362	390
Dec	118	140	166	194	201	229	278	306	336	337	405	432

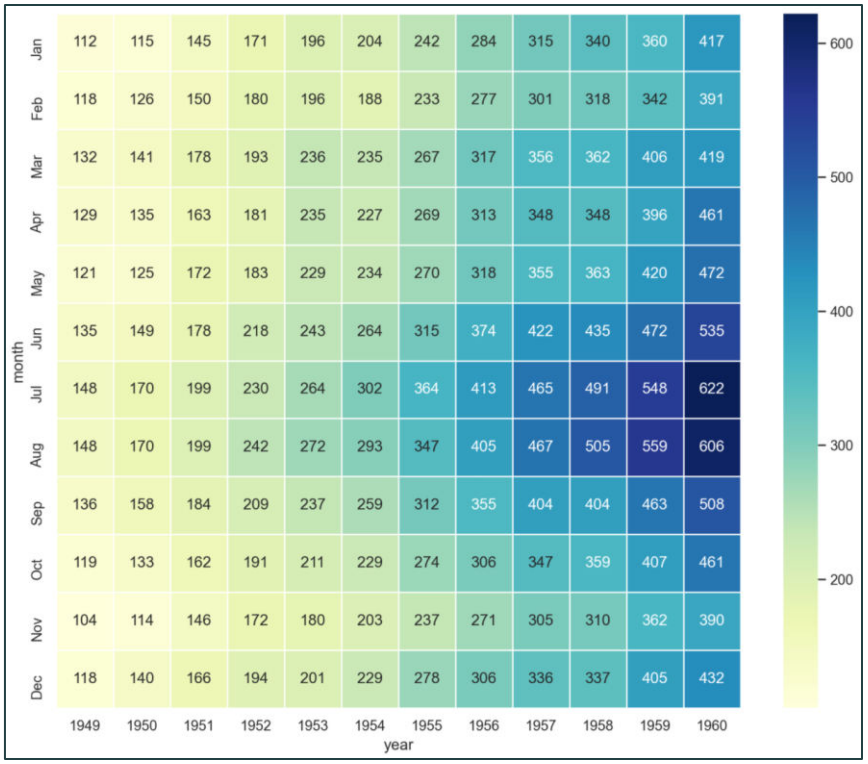
4. Các loại biểu đồ trong Seaborn

Biểu đồ nhiệt

- **heatmap()**

```
sns.heatmap(flights, annot=True, fmt='d', linewidths=.5)
```





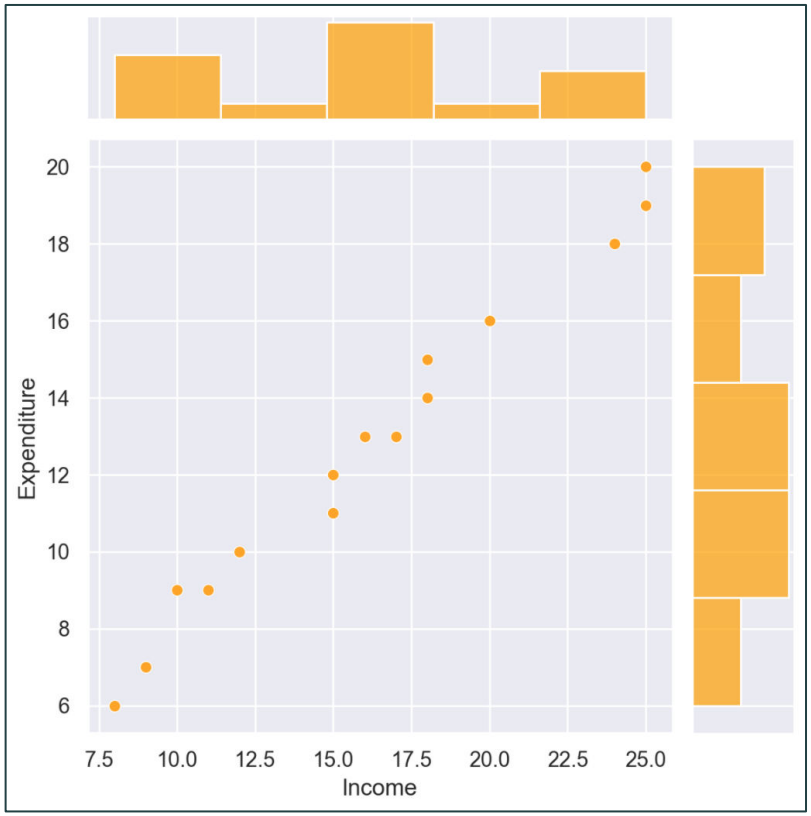
cmap='YlGnBu'

4. Các loại biểu đồ trong Seaborn

Biểu đồ kết hợp

- **jointplot()**

```
df = pd.read_csv('./data/Income.csv')
sns.jointplot(x='Income', y='Expenditure', data=df, color='orange')
```

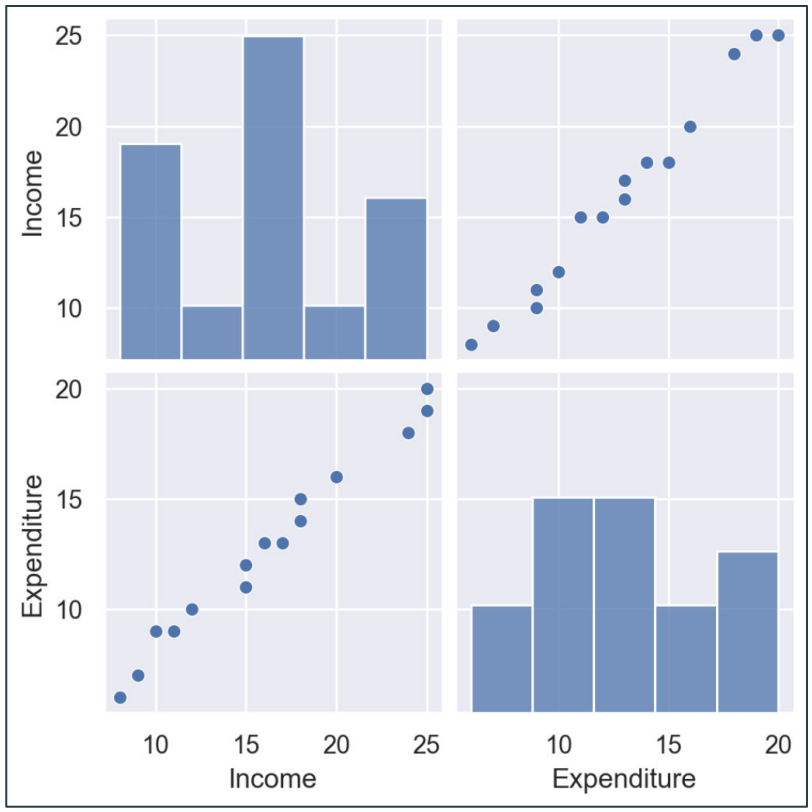


4. Các loại biểu đồ trong Seaborn

Biểu đồ kết hợp

- **pairplot()**

```
df = pd.read_csv('./data/Income.csv')
sns.pairplot(df[['Income','Expenditure']])
```

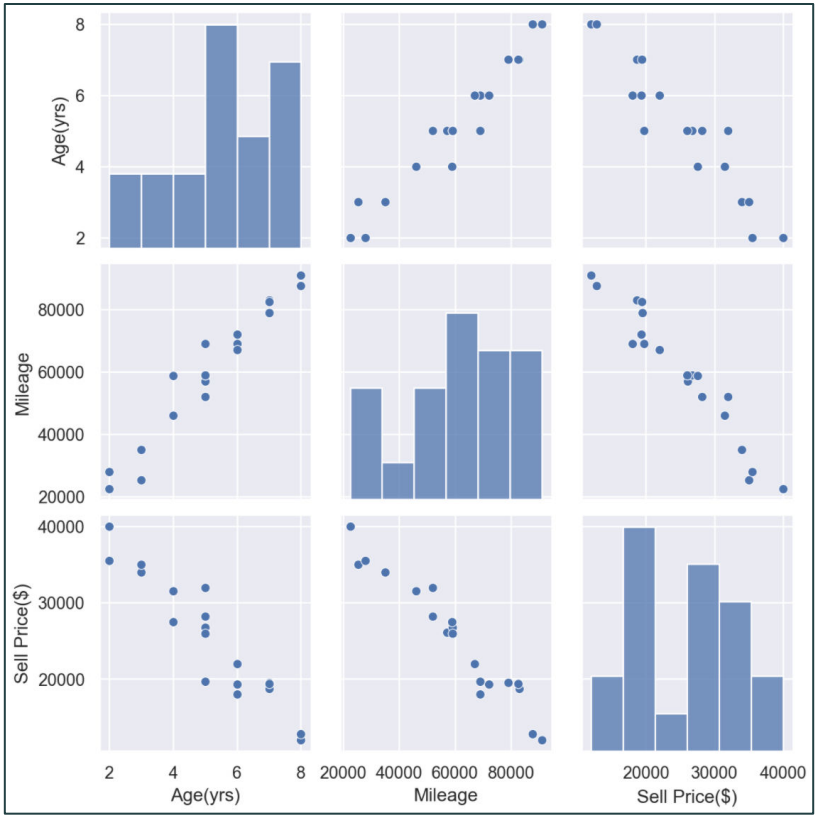


4. Các loại biểu đồ trong Seaborn

Biểu đồ kết hợp

- **pairplot()**

```
df = pd.read_csv('./data/Car_Prices.csv')
sns.pairplot(df[['Age(yrs)','Mileage', 'Sell Price($)']])
```



Q & A