Biểu diễn dữ liệu

20-01-2021

1 Giới thiệu

Trực quan hóa dữ liệu (Data Visualization) là việc thể hiện dữ liệu dưới dạng hình ảnh, giúp cho con người nắm bắt nhanh về dữ liệu.

Để hiểu rõ tầm quan trọng của việc trực quan hóa dữ liệu, cùng theo dõi ví dụ sau dây.

```
In [2]: import pandas as pd
        import matplotlib
        import matplotlib.pyplot as plt

        %matplotlib notebook
        plt.style.use('ggplot')
        matplotlib.rc("lines", markeredgewidth=0.5)

        quartet = pd.read_csv('https://raw.githubusercontent.com/Levytan/MIS.2019/mas
        quartet
```

Out[2]:

Observation	x1	y1	x2	y2	x3	y3	x4	y4
1	10	8.04	10	9.14	10	7.46	8	6.58
2	8	6.95	8	8.14	8	6.77	8	5.76
3	13	7.58	13	8.74	13	12.74	8	7.71
4	9	8.81	9	8.77	9	7.11	8	8.84
5	11	8.33	11	9.26	11	7.81	8	8.47
6	14	9.96	14	8.10	14	8.84	8	7.04
7	6	7.24	6	6.13	6	6.08	8	5.25
8	4	4.26	4	3.10	4	5.39	19	12.50
9	12	10.84	12	9.13	12	8.15	8	5.56
10	7	4.82	7	7.26	7	6.42	8	7.91
11	5	5.68	5	4.74	5	5.73	8	6.89

Đây là bộ bốn dữ liệu (x, y) được nhà thống kê Francis Anscombe xây dựng vào năm 1973. Bộ bốn dữ liệu này có thống kê mô tả gần giống nhau.

```
In [3]: quartet.describe().loc[['count', 'mean', 'std']]
```

Out[3]:

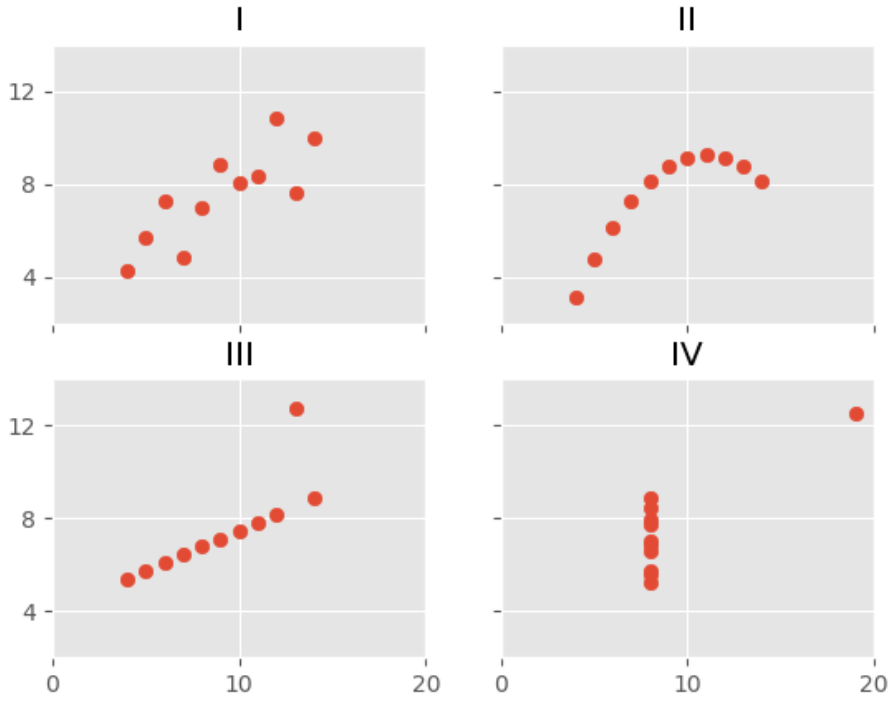
	x1	y1	x2	y2	x3	y3	\
count	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000	
mean	9.000000	7.500909	9.000000	7.500909	9.000000	7.500000	
std	3.316625	2.031568	3.316625	2.031657	3.316625	2.030424	

```
                x4         y4
count   11.000000  11.000000
mean     9.000000   7.500909
std      3.316625   2.030579
```

Nếu chỉ dựa vào thống kê mô tả, một người có thể cho rằng bốn bộ dữ liệu này gần giống nhau, tuy nhiên, sự khác biệt được thể hiện khi biểu diễn bằng hình ảnh các bộ dữ liệu này.

```
In [4]: fig, axes = plt.subplots(2, 2, sharex = True, sharey = True)
        axes[0, 0].set(xlim = (0, 20), ylim = (2, 14))
        axes[0, 0].set(xticks = (0, 10, 20), yticks = (4, 8, 12))

        axes[0, 0].scatter(quartet.x1, quartet.y1)
        axes[0, 0].set_title('I')
        axes[0, 1].scatter(quartet.x2, quartet.y2)
        axes[0, 1].set_title('II')
        axes[1, 0].scatter(quartet.x3, quartet.y3)
        axes[1, 0].set_title('III')
        axes[1, 1].scatter(quartet.x4, quartet.y4)
        axes[1, 1].set_title('IV')
```



Bộ bốn Anscombe

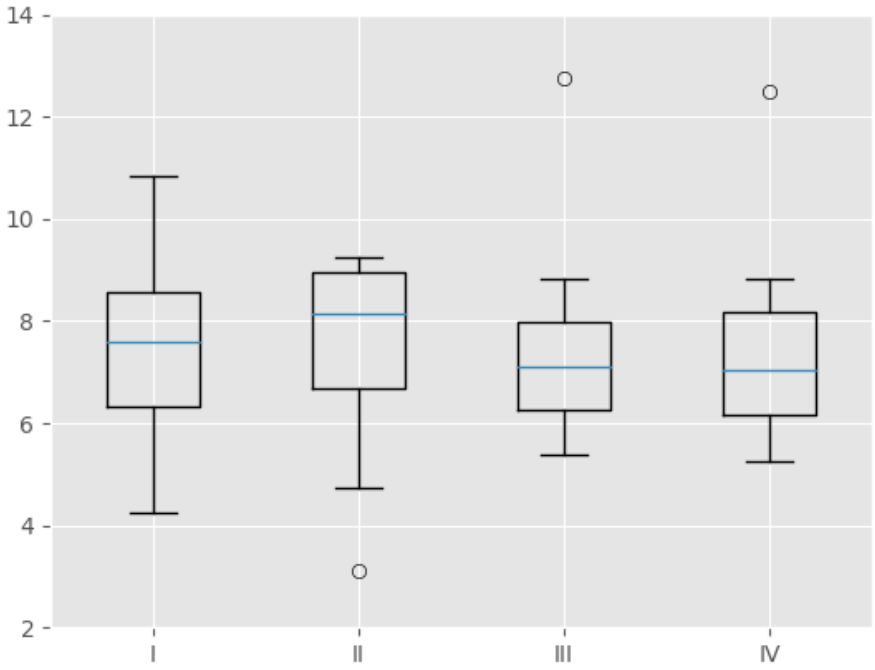
2 Box plot

2.1 Công dụng của boxplot

Bước đầu tiên trong phân tích dữ liệu đó chính là việc nắm bắt được phân bổ của dữ liệu (tập trung, phân tán, đối xứng, lệch). Và boxplot chính là công cụ để thực hiện điều đó.

Cùng xem boxplot các boxplot sau :

```
In [5]: plt.ylim(2, 14)
        plt.boxplot([quartet.y1, quartet.y2, quartet.y3, quartet.y4], labels = ['I',
```



Boxplot của Bộ bốn Anscombe

Từ các boxplot trên, có thể thấy:

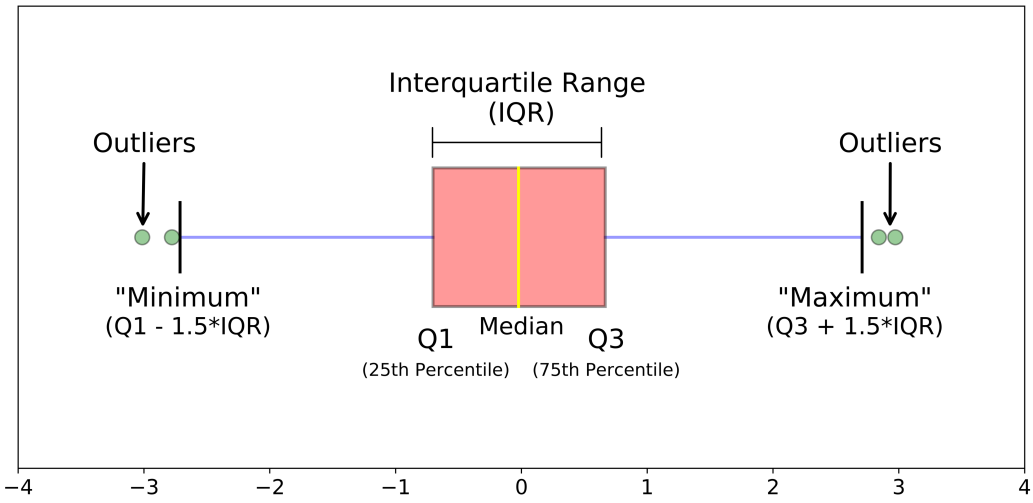
- Dữ liệu I phân tán rộng và khá đối xứng.
- Dữ liệu II lệch về bên phải.
- Dữ liệu III phân tán hẹp.

2.2 Giới thiệu

Biểu đồ hộp (Box plot) là biểu đồ diễn tả 5 vị trí phân bố của dữ liệu, đó là: min, tứ phân vị thứ nhất (Q1), trung vị (median), tứ phân vị thứ 3 (Q3) và max.

Trong đó:

- `median`: giá trị trung vị của bộ dữ liệu.
- `Q1`: trung vị của dữ liệu nằm giữa trung vị và giá trị nhỏ nhất.
- `Q3`: trung vị của dữ liệu nằm giữa trung vị và giá trị lớn nhất.
- `IQR`: bằng `Q3 - Q1`.
- `min`: điểm dữ liệu nhỏ nhất mà lớn hơn `Q1 - 1.5*IQR`.



Boxplot

- `max`: điểm dữ liệu lớn nhất mà nhỏ hơn `Q3 + 1.5*IQR`.
- `outlier`: các điểm dữ liệu nằm ngoài khoảng `(min, max)`.

2.3 Cách vẽ boxplot

Để vẽ boxplot, bạn có thể dùng phương thức của `.plot()` của `Series` hoặc dùng hàm `plt.boxplot()`

```
In [6]: # dùng phương thức .plot()
        quartet.y1.plot(kind = 'box')
```

```
Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x1b214dbb7b8>
```

```
In [8]: # dùng plt.boxplot()
        plt.boxplot(quartet.y3)
```

3 Histogram

3.1 Công dụng

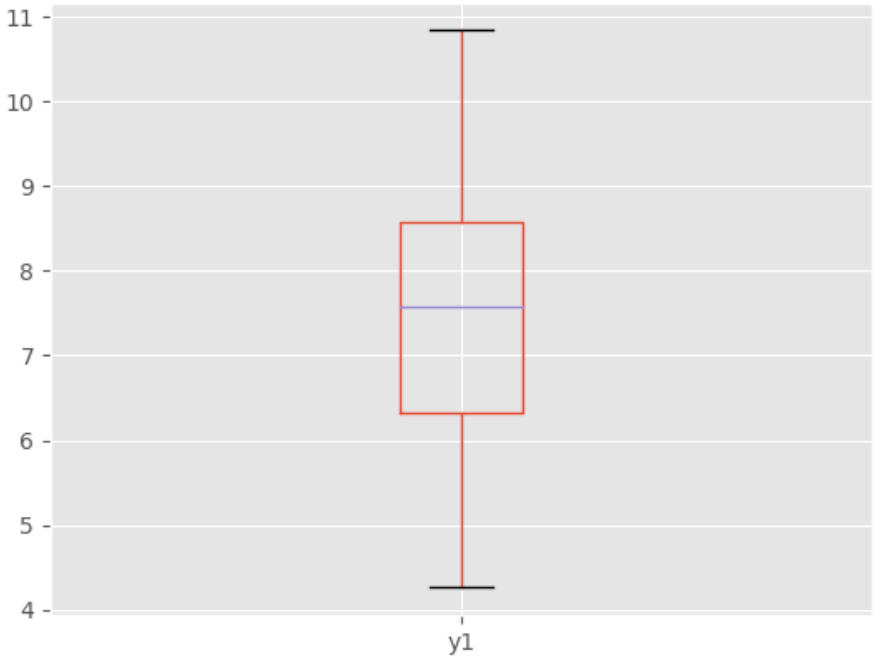
Tuy boxplot có thể cho thấy phân bổ của dữ liệu, nhưng lại không cho thấy được tần suất của dữ liệu. Để biểu diễn chính xác hơn dữ liệu, người ta dùng đến `histogram` (biểu đồ tần suất).

Histogram có hình dạng như sau :

Histogram là một dạng đồ thị cho phép bạn khám phá, hiển thị dạng phân phối tần suất của một tập dữ liệu liên tục. Nó cho phép chúng ta kiểm tra dạng phân phối (chẳng hạn, phân phối chuẩn), điểm dị biệt, độ trôi, độ nhọn của tập dữ liệu.

3.2 Cách vẽ Histogram

Để vẽ boxplot, bạn có thể dùng phương thức của `.plot(kind = 'hist')` của `Series` hoặc dùng hàm `plt.hist()`.



Vẽ bằng `.plot()` của Series

```
In [10]: iris.sepal_width.plot(kind = 'hist', bins = 20)
```

```
In [11]: plt.hist(iris.petal_width, bins = 5)
```

4 KDE chart

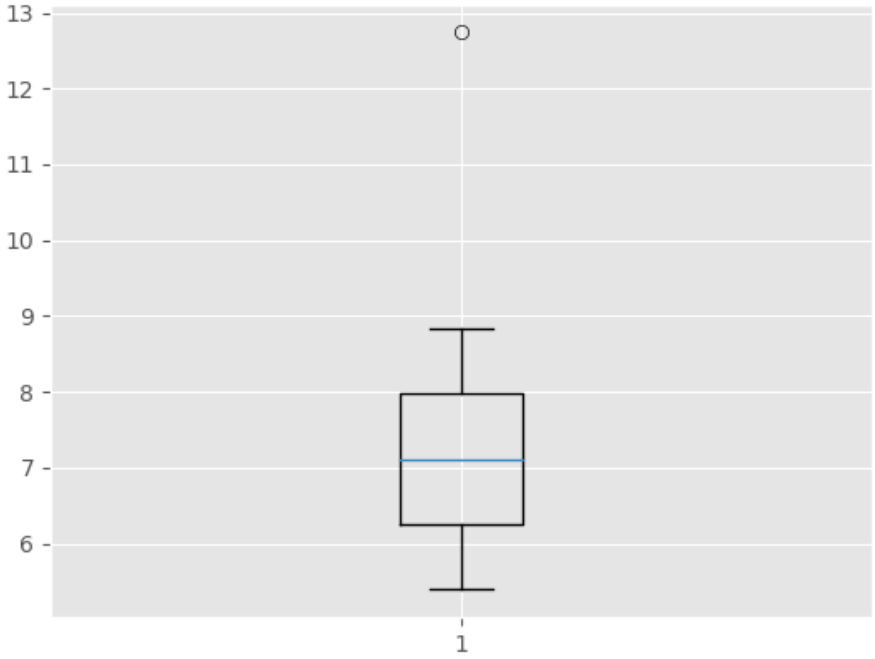
4.1 Công dụng

Histogram có một điểm yếu là hình dạng của nó phụ thuộc vào việc chia số bins. Vì vậy, đôi khi người ta còn dùng một công cụ khác là KDE (kernel density estimator).

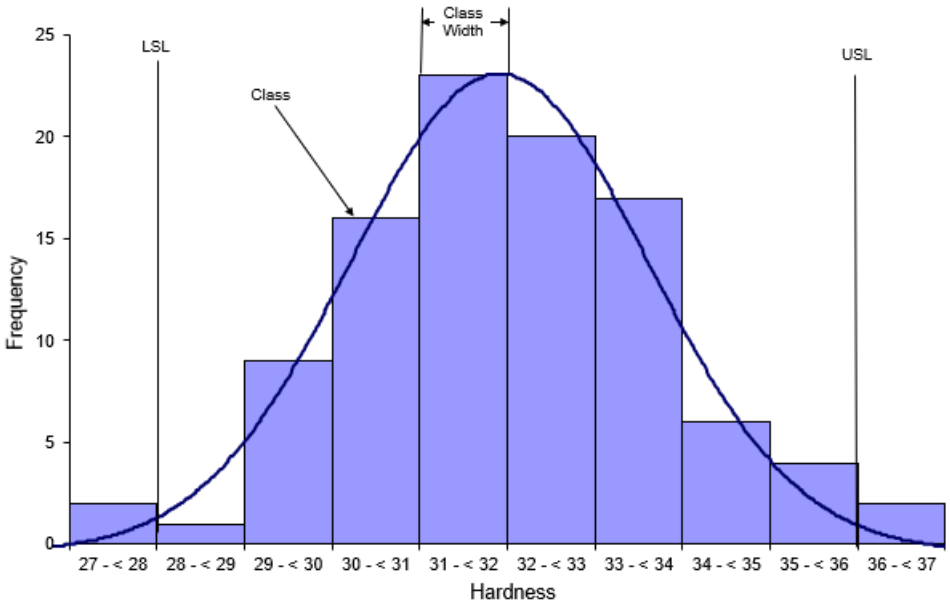
4.2 Cách vẽ

Tương tự như histogram, bạn có thể dùng `.plot(kind = 'kde')` của `Series`.

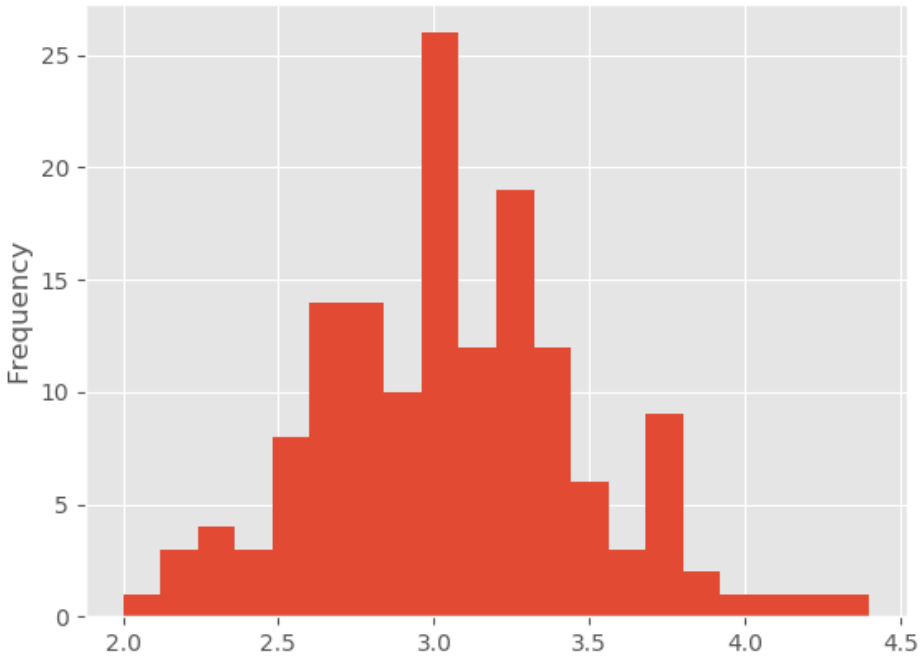
```
In [12]: iris.sepal_width.plot(kind = 'kde')
```



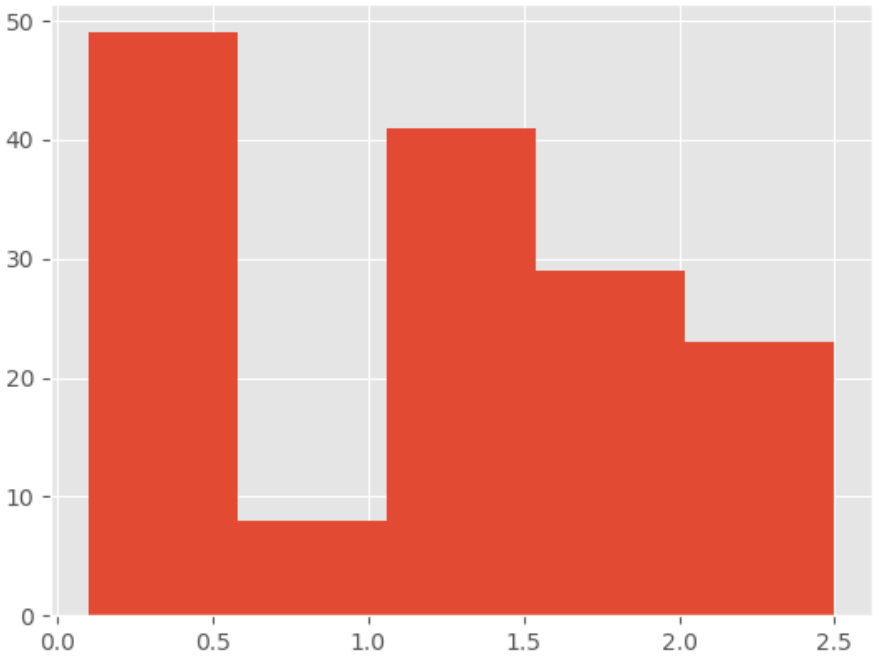
Vẽ bằng `plt.boxplot()`



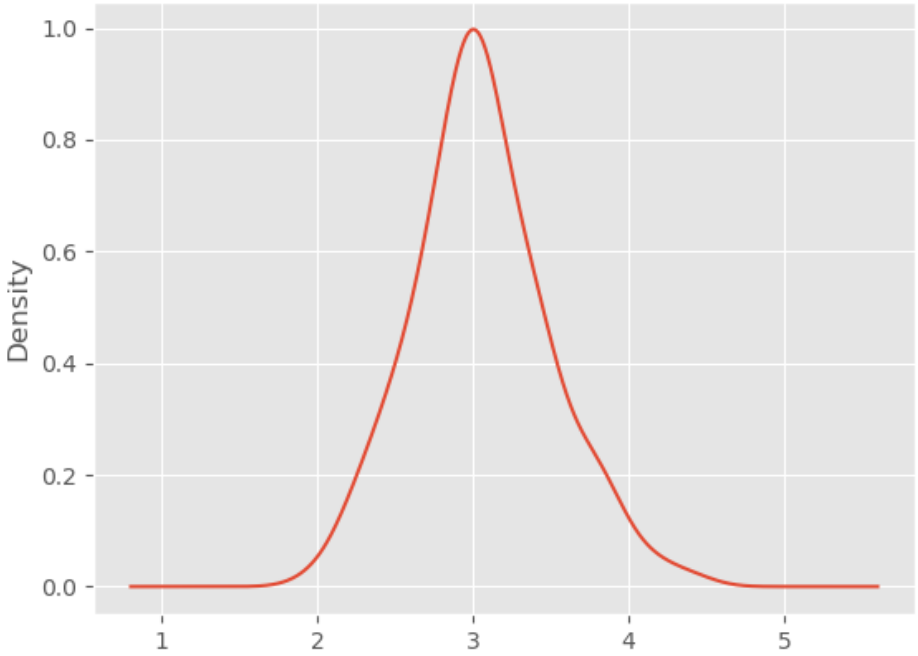
Histogram



Vẽ bằng `.plot(kind = 'hist')`



Vẽ bằng `plt.hist()`



Vẽ bằng `.plot(kind = 'kde')`