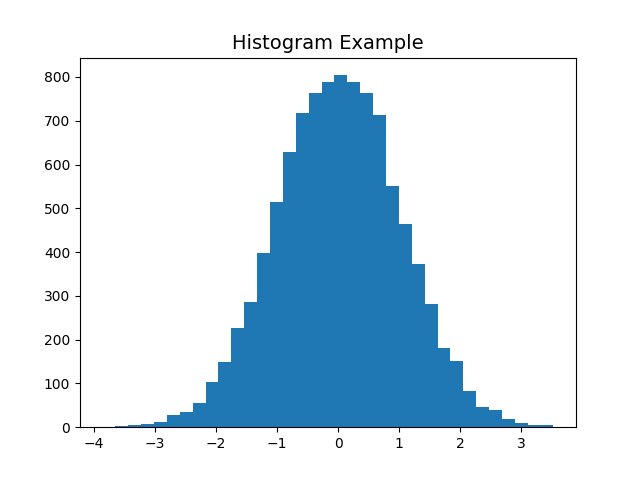
## **Biểu đồ mô tả một thuộc tính**

### **Giới thiệu**

Data Visualization cho các thuộc tính có thể được thực hiện độc lập hoặc kết hợp với một số thuộc tính khác. Việc hiển thị dữ liệu độc lập được thực hiện với các biểu đồ đơn biến như biểu đồ histogram, biểu đồ cột, biểu đồ tròn. Các biểu đồ đơn biến có thể giúp chúng ta minh họa dữ liệu qua một khoảng thời gian, hay minh họa mối quan hệ so sánh dữ liệu giữa các danh mục riêng hoặc so sánh một dữ liệu so với toàn thể.

### **Biểu đồ Histogram**



Histogram (biểu đồ phân bố) là một dạng biểu đồ thể hiện tần suất dạng cột. Nó mô tả dữ liệu một cách đơn giản mà không làm mất bất cứ thông tin thống kê nào của dữ liệu. Histogram cho thấy hình thái phân bố của dữ liệu, qua đó có thể đánh giá chất lượng quá trình thu thập dữ liệu, thiết lập mục tiêu và xu hướng khắc phục cho từng vấn đề. Biểu đồ trả lời các câu hỏi sau:

* Kiểu phân bố dữ liệu?
* Độ rộng dữ liệu như thế nào?
* Dữ liệu có đối xứng hay không?
* Có dữ liệu nào nằm ngoài hay không?

Biểu đồ phân bố thường có dạng phân bố đối xứng, hình chuông. Chính vì thế, hình dạng, "độ trơn" của biểu đồ được dùng để đánh giá vấn đề của dữ liệu nhằm phát hiện ra những nguyên nhân đặc biệt đang tác động đến dữ liệu từ đó đưa ra các điều chỉnh, cải tiến cụ thể. Một số hình dạng biểu đồ thường gặp:

* Phân bố chuẩn (hình chuông)
* Phân bố hai đỉnh
* Phân bố lệch (trái/phải)
* Phân bố răng cưa
* Phân bố bình nguyên
* Phân bố đảo nhỏ
* Phân bố hai đỉnh độc lập

#### Thành phần chính trong biểu đồ Histogram

* Trục X: thể hiện các lớp (classes). Mỗi lớp là các giá trị dữ liệu trong một khoảng nào đó được dùng để xây dựng biểu đồ tần suất. Mỗi cột trong biểu đồ thể hiện cho một lớp.
* Độ rộng của lớp chính là độ rộng của cột trong biểu đồ.
* Trục Y: thể hiện chiều cao của mỗi cột hình chữ nhật trong biểu đồ, diện tích của cột biểu diễn tần số của lớp tương ứng.
* Độ rộng của dải dữ liệu là sai khác giữa giá trị lớn nhất và nhỏ nhất.

#### Ưu điểm và nhược điểm của biểu đồ Histogram

*Ưu điểm:* Biểu đồ phân bố cho chúng ta biết bốn vấn đề sau:

* Trực quan hóa dữ liệu tốt.
* Có thể dùng để so sánh với đường cong chuông (Bell Curve) của phân phối chuẩn.
* Thể hiện giá trị thường xuất hiện nhất (mode).
* Mức độ thường xuất hiện của mỗi giá trị (tần suất).
* Hình dạng của phân bố.
* Mối quan hệ giữa các dữ liệu và các giới hạn yêu cầu (max, min. average).

*Nhược điểm:*

* Chỉ hiển thị tần suất của dữ liệu, không hiển thị được chính xác giá trị của dữ liệu do dữ liệu đã bị phân vào các lớp.
* Khó dùng để so sánh nhiều danh mục.

#### Vẽ biểu đồ Histogram bằng thư viện Matplotlib

* Câu lệnh vẽ biểu đồ Histogram:

plt.hist(x, bins = None, ’…’)

Trong đó:

x: danh sách dữ liệu.

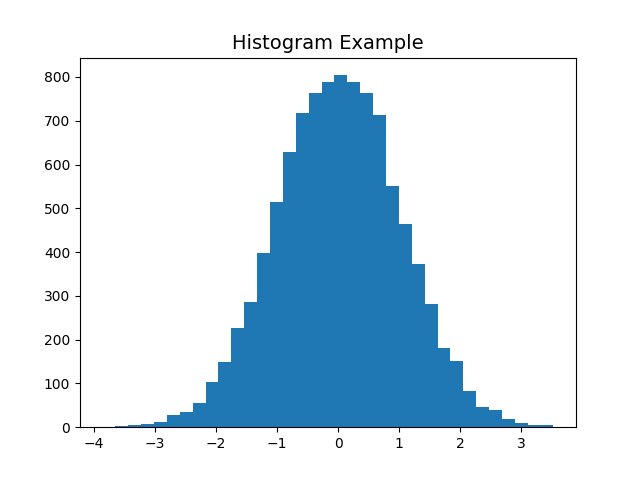
bins: số lớp (số cột) dữ liệu.

‘…’: các tham số cài đặt thuộc tính biểu đồ

Ví dụ

import matplotlib.pyplot as plt  
import numpy as np  
gaussian\_numbers = np.random.normal(size=10000)  
plt.hist(gaussian\_numbers, bins = 35)  
plt.title('Histogram Example', fontsize = 14)  
plt.show()

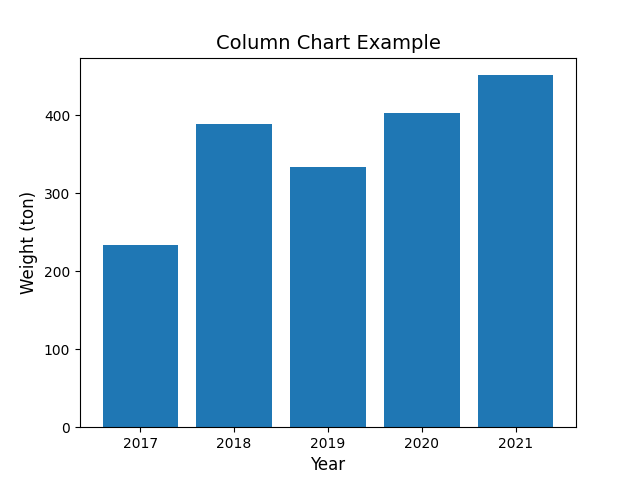
Kết quả:



* Các tham số cài đặt thuộc tính biểu đồ Histogram:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tham số** | **Giá trị** | **Ý nghĩa sử dụng** | **Ví dụ minh họa** |
| range | Tuples or None | Chỉ định phạm vi sử dụng dữ liệu. Mặc định là (x.min(), x.max()) | plt.hist(gaussian\_numbers, bins = 35, range=(-1,1)) |
| density | boolean | Vẽ hàm mật độ xác suất  [Bài đọc](https://math.stackexchange.com/questions/2666834/what-is-the-difference-between-frequency-and-density-in-a-histogram) | plt.hist(gaussian\_numbers, bins = 35, density = True) |
| cumulative | Boolean or None or -1 | Vẽ hàm phân phối tích lũy. Nếu density = True thì vẽ hàm mật độ tích lũy với tổng là 1.  ***Cộng dồn với nhau cho các giá trị nhỏ hơn trong mỗi bin.*** | plt.hist(gaussian\_numbers, bins = 35, cumulative=True) |
| histtype | { 'bar', 'barstacked', 'step', 'stepfilled'} | Cài đặt kiểu vẽ biểu đồ. | plt.hist(gaussian\_numbers, bins = 35, histtype='step') |
| orientation | {'vertical', 'horizontal'} | Cài đặt hướng hiển thị của biểu đồ | plt.hist(gaussian\_numbers, bins = 35, orientation='horizontal') |

### **Biểu đồ cột**

******

Một trong những biểu đồ phổ biến nhất được sử dụng trong Data Visualization là biểu đồ cột. Các biểu đồ cột được dùng để so sánh sự khác biệt dữ liệu giữa các mốc thời gian rời rạc hay các danh mục cố định. Dữ liệu phổ biến nhất cho biểu đồ cột là những tập dữ liệu được phân chia thành các loại/danh mục và không bị ràng buộc thứ tự. Sử dụng biểu đồ cột giúp người xem dễ hình dung và so sánh dữ liệu của một thuộc tính nào đó.

#### Thành phần chính trong biểu đồ cột

* Trục X: hiển thị một khoảng thời gian hoặc tập các danh mục.
* Trục Y: hiển thị các giá trị số tương ứng với mỗi danh mục.
* Các cột: Dữ liệu được hiển thị bằng các cột, mỗi cột đại diện cho một danh mục cụ thể. Chiều cao của mỗi cột đại diện cho một giá trị số cụ thể (ví dụ: tổng của các giá trị trong danh mục mà nó đại diện) hoặc biểu diễn tần suất hay tần số. Các danh mục có thể là số lượng dân số, nhóm tuổi hoặc vị trí địa lý…
* Tất cả các cột phải biểu diễn chung một thuộc tính của dữ liệu.
* Mỗi cột trong biểu đồ phải có chiều rộng bằng nhau.
* Khoảng cách giữa mỗi cột phải giống nhau.
* Các cột biểu diễn dữ liệu có thể ở dạng đứng hoặc nằm ngang.

#### Ưu điểm và nhược điểm của biểu đồ cột

*Ưu điểm:*

* Biểu đồ cột dễ đọc, dễ hiểu, trực quan đơn giản.
* Minh họa rõ xu hướng của dữ liệu tốt hơn so với bảng.
* Giúp ước tính các giá trị quan trọng như max, min, mức độ chênh lệch dữ liệu giữa các danh mục một cách nhanh chóng.

*Nhược điểm:*

* Đôi khi, biểu đồ cột không thể hiện được quy luật các mẫu, nguyên nhân, tác động, v.v.
* Nó có thể dễ dàng bị thao túng để mang lại thông tin giả mạo. Ví dụ, nếu chọn tỉ lệ trên trục Y quá lớn thì sự chênh lệch dữ liệu có thể xuất hiện không đáng kể, trong khi thực tế nó có thể rất quan trọng.

#### Vẽ biểu đồ cột bằng thư viện Matplotlib

* Câu lệnh vẽ biểu đồ cột:

plt.bar(x, y, '…')

trong đó:

x: danh sách tọa độ trục ngang hoặc danh sách các danh mục hoặc các mốc thời gian.

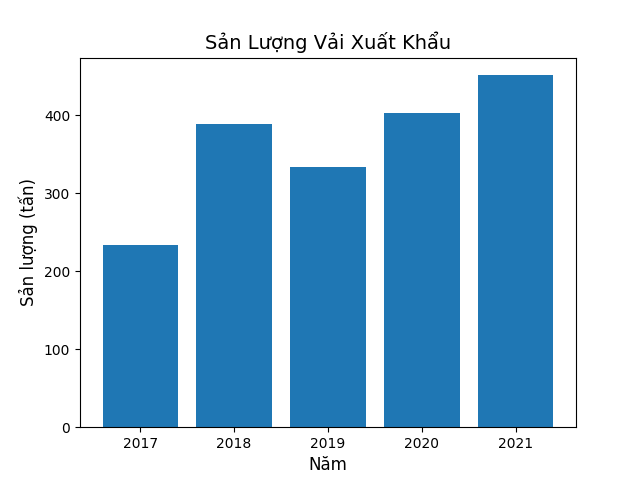
y: danh sách giá trị số tương ứng với các danh mục.

‘…’: các tham số cài đặt thuộc tính biểu đồ

Ví dụ

import matplotlib.pyplot as plt  
x = ['2017', '2018', '2019', '2020', '2021']  
y = [234, 389, 333, 402, 451]  
plt.bar(x, y)  
plt.title('Sản Lượng Vải Xuất Khẩu', fontsize = 14)  
plt.xlabel('Năm', fontsize = 12)  
plt.ylabel('Sản lượng (tấn)', fontsize = 12)  
plt.show()

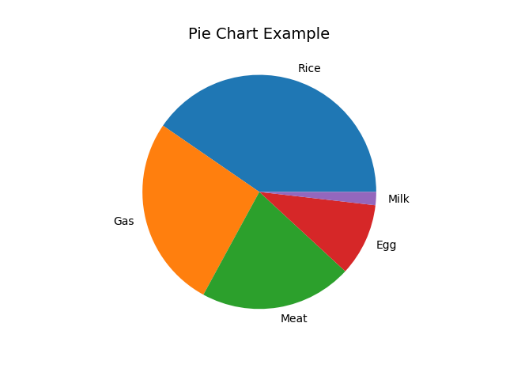
Kết quả:



* Các tham số cài đặt thuộc tính biểu đồ cột:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tham số** | **Giá trị** | **Ý nghĩa sử dụng** | **Ví dụ minh họa** |
| width | float or array-like | Thay đổi chiều rộng các cột | plt.bar(x, y, width=0.5) |
| bottom | float or array-like | Cài đặt giá trị cơ sở của trục Y (mặc định là 0) | plt.bar(x, y, bottom= 100) |
| align | ‘edge’ or ‘center’ | Căn chỉnh vị trí của cột so với tọa độ x (mặc định là center) | plt.bar(x, y, bottom= 100) |
| color | Color or list of color | Cài đặt một màu hoặc danh sách các màu cho các cột | plt.bar(x, y, color = ['red', 'green', 'blue']) |
| edgecolor | Color or list of color | Cài đặt một màu hoặc danh sách các màu cho các đường viền của cột | plt.bar(x, y, edgecolor = 'red') |
| linewidth | Float or array-like | Cài đặt chiều rộng đường viền cột. | plt.bar(x, y, linewidth = 3, edgecolor = 'red') |
| tick\_label | str or list of str | Cài đặt nhãn cho các cột. | plt.bar(x, y, tick\_label = ['Năm 1', 'Năm 2', 'Năm 3', 'Năm 4', 'Năm 5']) |

### **Biểu đồ tròn**

******

Biểu đồ tròn được sử dụng để biểu diễn tỉ lệ phần trăm của các thành phần so với tổng thể. Nó cũng có thể sử dụng để thể hiện thị phần của mỗi giá trị trong tập dữ liệu (tần số một biến rời rạc). Loại biểu đồ này hữu ích trong việc thể hiện cấu thành của bộ dữ liệu và tương quan giữa các thành phần so với tổng thể.

Ví dụ, minh họa phần trăm tổng doanh thu của mỗi loại sản phẩm

#### Thành phần chính trong biểu đồ tròn

* Tổng các thành phần là 100%, không thành phần nào có giá trị âm.
* Chỉ dùng biểu đồ tròn khi ***số lượng thể loại ít hơn 6***.
* ***Không dùng*** biểu đồ tròn nếu tỉ lệ giữa các thể ***loại gần tương đương nhau***.
* Nên sắp xếp giá trị các thể loại để dễ hiểu hơn (ví dụ từ lớn đến nhỏ).
* Tránh sử dụng dạng 3D hoặc nghiêng (bóp méo biểu đồ).

#### Ưu nhược điểm của biểu đồ tròn

*Ưu điểm*

* Tóm tắt một tập dữ liệu lớn ở dạng trực quan.
* Giúp dễ dàng hình dung được giá trị % mà các thành phần đóng góp vào tổng thể.

*Nhược điểm*

* Không thể hiện được giá trị chính xác.
* Có thể cần nhiều biểu đồ hình tròn để hiển thị các thay đổi theo thời gian.
* Không thể hiện được các quy luật, nguyên nhân, tác động, v.v.
* Dễ dàng bị thao túng để mang lại thông tin giả mạo.

#### Vẽ biểu đồ tròn bằng thư viện Matplotlib

* Câu lệnh vẽ biểu đồ tròn

plt.pie(sizes, labels = labels, '...')

Trong đó:

sizes: danh sách giá trị các danh mục.

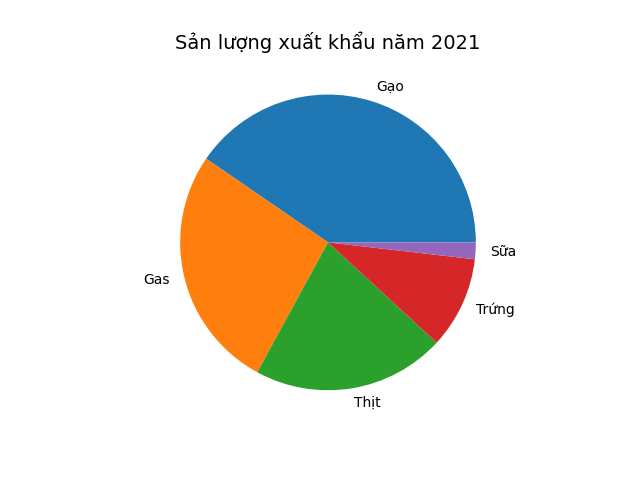
labels: danh sách nhãn các danh mục.

‘…’: các tham số.

Ví dụ

import matplotlib.pyplot as plt  
labels = ['Gạo', 'Gas', 'Thịt', 'Trứng', 'Sữa']  
sizes = [443, 292, 231, 110, 20]  
plt.pie(sizes, labels = labels)  
plt.title('Sản lượng xuất khẩu năm 2021', fontsize = 14)  
plt.show()

Kết quả:



* Các tham số cài đặt thuộc tính biểu đồ tròn:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tham số** | **Giá trị** | **Ý nghĩa sử dụng** | **Ví dụ minh họa** |
| **colors** | list of color | Cài đặt màu sắc cho các thành phần | plt.pie(sizes, labels = labels, colors = ['r', 'g', 'b', 'y', 'c']) |
| **autopct** | None or str or callable | Dùng để gán nhãn % cho từng thành phần. Nếu là str, str sẽ được nối vào sau giá trị số của từng thành phần để hiển thị. Nếu là một hàm, hàm đó sẽ được gọi. | plt.pie(sizes, labels = labels, autopct='%1.2f%%') |
| **pctdistance** | float | Tỷ lệ giữa tâm của mỗi thành phần và phần bắt đầu của văn bản được tạo ra bởi tính năng autopct . Bỏ qua nếu autopct là None | plt.pie(sizes, labels = labels, autopct='%1.2f%%', pctdistance=0.4) |
| **shadow** | bool | Vẽ bóng mờ cho các thành phần. Mặc định là không (false) | plt.pie(sizes, labels = labels, shadow= True) |
| **normalize** | None or boolean | Khi tổng giá trị các thành phần nhỏ hơn 1, nếu normalize = False thì chỉ vẽ 1 phần của hình tròn. | sizes = [0.2, 0.3, 0.2, 0.1, 0.1]; plt.pie(sizes, labels = labels, normalize = False) |
| **labeldistance** | Float or None | Khoảng cách tính từ tâm mà tại đó các nhãn được vẽ. Nếu là None, nhãn sẽ không được vẽ, nhưng được lưu trữ để sử dụng trong legend() | plt.pie(sizes, labels = labels, labeldistance=0.7) |
| **startangle** | float | Góc quay bắt đầu của hình tròn tính từ trục X, ngược chiều kim đồng hồ. | plt.pie(sizes, labels = labels, startangle=45) |
| **radius** | float | Bán kính của hình tròn. Mặc định là 1 | plt.pie(sizes, labels = labels, radius=1.3) |
| **counterclock** | bool | Mặc định là ngược chiều kim đồng hồ | plt.pie(sizes, labels = labels, counterclock= False) |
| frame | bool | Khung cho biểu đồ. Mặc định là Không | plt.pie(sizes, labels = labels, frame=True) |

### **Tổng kết**

Qua bài đọc này, chúng ta đã tìm hiểu được một số định nghĩa, ý nghĩa sử dụng, các thành phần chính, cách sử dụng thư viện Matplotlib để vẽ và tinh chỉnh một số biểu đồ:

* Biểu đồ Histogram
* Biểu đồ Cột
* Biểu đồ Tròn