

KHOA KỸ THUẬT VÀ CÔNG NGHỆ  
BỘ MÔN CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH  
HỌC KỲ I, NĂM HỌC 2024-2025

**TÌM HIỂU JUPYTER BOOK VÀ BIÊN SOẠN  
TÀI LIỆU HƯỚNG DẪN LẬP TRÌNH  
TRỰC QUAN HÓA DỮ LIỆU TRÊN PYTHON**

*Giáo viên hướng dẫn:*  
ThS. Hà Thị Thúy Vi

*Sinh viên thực hiện:*  
Họ tên: Trần Ngọc Hành  
MSSV: 110122219  
Lớp: DA22TTA

Trà Vinh, tháng 12 năm 2024

### NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

Trà Vinh, ngày ..... tháng ..... năm .....

**Giáo viên hướng dẫn**

(Ký tên và ghi rõ họ tên)

### NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

Trà Vinh, ngày ..... tháng ..... năm .....  
**Thành viên hội đồng**  
(Ký tên và ghi rõ họ tên)

## LỜI CẢM ƠN

Lời nói đầu, Em xin cảm ơn cô Hà Thị Thúy Vi đã hỗ trợ, hướng dẫn tận tình chúng em trong thời gian em làm thực tập cơ sở ngành , những kiến thức mà cô đã dạy chúng em sẽ là hành trang quý báo trên con đường học vấn và phát triển sự nghiệp tương lai rộng mở của chúng em. Cô đã luôn kiên nhẫn, nhiệt tình trong việc truyền đạt kiến thức và kinh nghiệm quý báo, giúp chúng em vượt qua những khó khăn và thử thách trong quá trình học tập và nghiên cứu.

Những lời khuyên, góp ý của cô không chỉ là kim chỉ nam cho sự phát triển của đồ án môn học này mà còn là nguồn động viên, khích lệ tinh thần lớn lao cho chúng em.

Chúng em xin hứa sẽ tiếp tục nỗ lực không ngừng để không phụ lòng cô đã dành cho chúng em.

Xin chân thành cảm ơn Cô.

Trà Vinh, 29 Tháng 12 năm 2024

**Trần Ngọc Hành**

## MỤC LỤC

|   |    |
|---|----|
| MỞ ĐẦU .....  | 7  |
| CHƯƠNG 1: TỔNG QUAN .....   | 9  |
| CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT .....                                    | 10 |
| 2.1. Cơ sở lý thuyết .....  | 10 |
| 2.1.1. Giới thiệu về trực quan hóa dữ liệu .....                        | 10 |
| 2.1.2. Khái quát về Jupyter Book .....                                  | 10 |
| 2.2. Công cụ và công nghệ sử dụng .....                                 | 17 |
| 2.2.1. Jupyter Book .....   | 17 |
| 2.2.2. Ngôn ngữ lập trình Python .....                                  | 17 |
| 2.2.3. Các công cụ hỗ trợ .....   | 18 |
| 2.3. Phần mềm và tài nguyên sử dụng .....                               | 20 |
| 2.3.1. Phần mềm .....   | 20 |
| 2.3.2. Tài nguyên học thuật .....                                       | 20 |
| 2.4. Mô hình tổ chức nội dung sách .....                                | 21 |
| 2.4.1. Cấu trúc tài liệu .....  | 21 |
| 2.4.2. Phân tích cách sử dụng Jupyter Book .....                        | 24 |
| CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU .....                                | 25 |
| 3.1. Chuẩn bị công cụ và môi trường làm việc .....                      | 25 |
| 3.1.1. Cài đặt Python .....   | 25 |
| 3.1.2. Cài đặt Jupyter Book: .....                                      | 25 |
| 3.1.3. Cài đặt các thư viện hỗ trợ (Matplotlib, Seaborn, Plotly): ..... | 26 |
| 3.1.4. Cài đặt Anaconda để quản lý môi trường làm việc: .....           | 26 |
| 3.1.5. Cài đặt Visual Studio Code để soạn thảo nội dung sách: .....     | 27 |
| 3.2. Xây dựng cấu trúc và nội dung sách .....                           | 27 |
| 3.3. Tích hợp mã nguồn và trực quan hóa dữ liệu .....                   | 29 |
| 3.4. Xây dựng và xuất bản sách .....                                    | 35 |
| CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU .....                                      | 36 |
| 4.1. Kết quả đạt được .....   | 36 |
| 4.2. Đánh giá hiệu năng .....   | 36 |
| 4.3. Trải nghiệm người dùng .....                                       | 37 |
| 4.4. Giao diện chức năng nghiên cứu .....                               | 37 |

|   |           |
|---|-----------|
| 4.5. Những khó khăn và cách khắc phục .....         | 37        |
| 4.6. Một số nội dung của tài liệu đã biên soạn..... | 37        |
| <b>CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b> | <b>38</b> |
| 5.1. Kết luận .....                                 | 38        |
| 5.2. Hướng phát triển .....                         | 38        |
| <b>DANH MỤC TÀI LIỆU THAM KHẢO .....</b>            | <b>39</b> |
| <b>PHỤ LỤC .....</b>                                | <b>40</b> |

## DANH MỤC HÌNH ẢNH

|  |    |
|--|----|
| Hình 2.1 Biểu đồ cột.....                    | 11 |
| Hình 2.2 Biểu đồ đường .....                 | 12 |
| Hình 2.3 Biểu đồ tròn .....                  | 12 |
| Hình 2.4 Histogram .....                     | 13 |
| Hình 2.5 Box Plot .....                      | 13 |
| Hình 2.6 Violin plot.....                    | 14 |
| Hình 2.7 Scatter plot.....                   | 14 |
| Hình 2.8 Bubble chart.....                   | 15 |
| Hình 2.9 Heatmap.....                        | 15 |
| Hình 2.10 Treemap .....                      | 16 |
| Hình 2.11 Network graph .....                | 16 |
| Hình 2.12 Geographic map.....                | 17 |
| Hình 2.13 Tiêu đề cấp 6 .....                | 18 |
| Hình 2.14 Danh sách không có thứ tự .....    | 18 |
| Hình 2.15 Danh sách có thứ tự .....          | 18 |
| Hình 2.16 Liên kết hình ảnh .....            | 19 |
| Hình 2.17 Chèn mã inline.....                | 19 |
| Hình 2.18 Chèn khôi mã.....                  | 19 |
| Hình 2.19 Liên kết chéo .....                | 19 |
| Hình 2.20 Cấu trúc tài liệu .....            | 23 |
| Hình 3.1 Logo Python .....                   | 25 |
| Hình 3.2 Logo Jupyter Book .....             | 26 |
| Hình 3.3 Logo Anaconda .....                 | 26 |
| Hình 3.4 Logo Visual Studio Code .....       | 27 |
| Hình 3.5 Thư mục dự án.....                  | 27 |
| Hình 3.6 _config.yml cấu hình tài liệu ..... | 28 |
| Hình 3.7 _toc.yml Cấu trúc nội dung .....    | 28 |
| Hình 3.8 Biểu đồ đường Matplotlib .....      | 29 |
| Hình 3.9 Tạo bộ cục nhiều biểu đồ.....       | 30 |
| Hình 3.10 Biểu đồ cột Seaborn.....           | 31 |
| Hình 3.11 Biểu đồ ghép.....                  | 32 |

---

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

|  |    |
|--|----|
| Hình 3.12 Biểu đồ tương tác với plotly ..... | 33 |
| Hình 3.13 Biểu đồ bản đồ .....               | 34 |
| Hình 3.14 Xuất sang định dạng PDF .....      | 35 |

## TÓM TẮT ĐỒ ÁN

### Vấn đề nghiên cứu:

Hiện nay, việc sử dụng các công cụ trực quan hóa dữ liệu đã trở thành một yêu cầu quan trọng trong lĩnh vực phân tích và khoa học dữ liệu. Python, với các thư viện mạnh mẽ như Matplotlib, Seaborn, và Plotly, là một trong những ngôn ngữ phổ biến nhất để xây dựng các biểu đồ và minh họa dữ liệu.

Tuy nhiên, tài liệu hướng dẫn chi tiết, có cấu trúc rõ ràng để hỗ trợ người học từ cơ bản đến nâng cao chưa thực sự phong phú. Do đó, đề tài tập trung vào việc:

Nghiên cứu Jupyter Book - một công cụ mạnh mẽ để biên soạn, tổ chức và xuất bản tài liệu trực tuyến.

Xây dựng tài liệu hướng dẫn lập trình trực quan hóa dữ liệu bằng Python, tích hợp lý thuyết và ví dụ thực tế.

### Các hướng tiếp cận:

Nghiên cứu công cụ Jupyter Book: Tìm hiểu cấu trúc, tính năng, cách biên tập tài liệu, và các định dạng xuất bản như HTML và PDF.

Phân tích các thư viện trực quan hóa dữ liệu: Khám phá Matplotlib, Seaborn, và Plotly, tập trung vào khả năng tạo biểu đồ, tùy chỉnh, và tương tác.

Xây dựng nội dung tài liệu: Chia tài liệu thành các chương rõ ràng, từ cài đặt môi trường, giới thiệu dữ liệu, đến các kỹ thuật trực quan hóa nâng cao.

### Cách giải quyết vấn đề:

Cấu trúc nội dung và tài liệu được biên soạn thành các chương cụ thể, bao gồm:

- Giới thiệu về lập trình trực quan hóa dữ liệu.
- Cài đặt môi trường và công cụ Python.
- Trình bày các thư viện Matplotlib, Seaborn, và Plotly.
- Ví dụ thực tiễn và các kỹ thuật nâng cao.
- Tích hợp trực quan hóa vào bài tập thực tế.

Sử dụng Jupyter Book:

Xây dựng tài liệu sử dụng định dạng Markdown và Jupyter Notebook.

---

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

Tích hợp hình ảnh, mã nguồn Python, và kết quả trực quan.

Xuất bản tài liệu dưới dạng website và file PDF để phục vụ học tập và tham khảo.

### **Một số kết quả đạt được:**

Tài liệu hướng dẫn hoàn chỉnh: Tài liệu gồm 10 chương với các nội dung từ cơ bản đến nâng cao, cung cấp các ví dụ minh họa chi tiết về trực quan hóa dữ liệu.

Ứng dụng Jupyter Book: Đã biên soạn và xuất bản thành công tài liệu dưới dạng website trực tuyến và PDF, hỗ trợ người học dễ dàng truy cập.

Hiệu quả thực tiễn: Người học nắm vững cách sử dụng các thư viện Matplotlib, Seaborn, và Plotly để trực quan hóa dữ liệu và Tăng khả năng áp dụng vào các dự án thực tế trong phân tích dữ liệu và khoa học dữ liệu

## MỞ ĐẦU

### Lí do chọn đề tài:

Tầm quan trọng của trực quan hóa dữ liệu trong thời đại số: Trong kỷ nguyên dữ liệu, khả năng trực quan hóa thông tin từ dữ liệu một cách hiệu quả đóng vai trò quan trọng trong việc hỗ trợ ra quyết định. Trực quan hóa không chỉ giúp truyền tải thông tin phức tạp thành hình ảnh dễ hiểu mà còn giúp người dùng khám phá, phân tích và tương tác với dữ liệu.

Sự phổ biến và sức mạnh của Python trong phân tích dữ liệu: Python là ngôn ngữ lập trình phổ biến nhờ cú pháp đơn giản và hệ sinh thái thư viện mạnh mẽ, đặc biệt trong lĩnh vực phân tích và trực quan hóa dữ liệu. Các thư viện như Matplotlib, Seaborn, và Plotly cung cấp khả năng tạo biểu đồ từ cơ bản đến nâng cao, đáp ứng nhu cầu từ người mới học đến chuyên gia.

Thiếu tài liệu hướng dẫn có cấu trúc: Mặc dù có nhiều tài liệu hướng dẫn trực tuyến, chúng thường rời rạc, thiếu tính hệ thống và không phù hợp với những người mới học hoặc cần tài liệu để giảng dạy. Việc biên soạn một tài liệu chi tiết, kết hợp lý thuyết và thực hành, đồng thời có tính tương tác là cần thiết để giúp người học tiếp cận hiệu quả hơn với lập trình trực quan hóa dữ liệu.

Ưu điểm của Jupyter Book trong giáo dục và nghiên cứu: Jupyter Book là một công cụ mạnh mẽ để xây dựng tài liệu giảng dạy và sách điện tử với khả năng tích hợp lý thuyết, mã nguồn và kết quả thực thi. Đây là nền tảng lý tưởng để phát triển tài liệu hướng dẫn lập trình, đặc biệt là trong lĩnh vực trực quan hóa dữ liệu.

Ứng dụng thực tế và tiềm năng phát triển: Đề tài không chỉ phù hợp với việc học và giảng dạy mà còn có tiềm năng ứng dụng vào nhiều lĩnh vực như khoa học dữ liệu, kinh tế, môi trường và xã hội. Việc sử dụng Jupyter Book để xuất bản tài liệu giúp tài liệu dễ dàng tiếp cận hơn, từ đó hỗ trợ cộng đồng người học và nhà nghiên cứu.

### Mục đích chọn đề tài:

Biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python một cách hệ thống và dễ tiếp cận.

Nghiên cứu và ứng dụng Jupyter Book để tạo sách điện tử tương tác, tích hợp lý thuyết và thực hành.

---

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

Hỗ trợ người học và giảng viên trong việc học tập và giảng dạy lập trình trực quan hóa dữ liệu.

Phát triển tài liệu giáo dục chất lượng cao, ứng dụng trong phân tích dữ liệu thực tiễn.

### **Đối tượng nghiên cứu:**

Jupyter Book: Cách cài đặt, cấu hình và sử dụng Jupyter Book để biên soạn tài liệu học tập, Tích hợp nội dung lý thuyết, mã nguồn Python, và biểu đồ trực quan vào sách điện tử tương tác.

Các thư viện trực quan hóa dữ liệu trên Python:

Matplotlib: Thư viện cơ bản để tạo biểu đồ và đồ họa dữ liệu.

Seaborn: Công cụ mở rộng trực quan hóa với các biểu đồ nâng cao.

Plotly: Thư viện hỗ trợ biểu đồ tương tác mạnh mẽ và linh hoạt.

Người học lập trình và phân tích dữ liệu: Sinh viên, nhà nghiên cứu, và người làm việc trong lĩnh vực khoa học dữ liệu cần tài liệu hướng dẫn rõ ràng, dễ hiểu, người mới học lập trình Python hoặc muốn tìm hiểu trực quan hóa dữ liệu để ứng dụng vào công việc.

### **Phạm vi nghiên cứu:**

Tập trung vào các thư viện trực quan hóa dữ liệu Python: Matplotlib, Seaborn, và Plotly.

Sử dụng Jupyter Book để biên soạn tài liệu học tập tương tác.

Ứng dụng cho việc học tập, giảng dạy và phân tích dữ liệu cơ bản đến trung cấp.

## CHƯƠNG 1: TỔNG QUAN

Trong thời đại dữ liệu, trực quan hóa thông tin là công cụ quan trọng giúp phân tích và trình bày dữ liệu một cách hiệu quả. Python với các thư viện mạnh mẽ như Matplotlib, Seaborn, và Plotly đã trở thành ngôn ngữ phổ biến trong lập trình trực quan hóa dữ liệu. Tuy nhiên, việc thiếu tài liệu hướng dẫn có hệ thống và dễ tiếp cận đã gây khó khăn cho người học và giảng viên.

Jupyter Book là công cụ hiện đại cho phép xây dựng tài liệu học tập dưới dạng sách điện tử tương tác. Với khả năng tích hợp lý thuyết, mã nguồn, và kết quả thực thi, Jupyter Book hỗ trợ hiệu quả cho việc biên soạn tài liệu lập trình, đặc biệt là trong lĩnh vực trực quan hóa dữ liệu.

Đề tài này tập trung vào việc tìm hiểu Jupyter Book và các thư viện trực quan hóa dữ liệu trên Python, sau đó biên soạn một tài liệu hướng dẫn có cấu trúc rõ ràng, tích hợp lý thuyết và thực hành. Nội dung tài liệu bao gồm từ kiến thức cơ bản đến ứng dụng thực tế, phục vụ cho sinh viên, giảng viên, và người làm việc trong lĩnh vực phân tích dữ liệu.

**Đối tượng và ứng dụng:**

**Đối tượng:** Người học lập trình, sinh viên, giảng viên, và người làm việc với dữ liệu.

**Ứng dụng:** Hỗ trợ giảng dạy và học tập trực quan hóa dữ liệu, áp dụng trong phân tích dữ liệu các lĩnh vực như kinh doanh, khoa học, và công nghệ.

**Ý nghĩa thực tiễn:** Đề tài không chỉ giúp người học tiếp cận dễ dàng hơn với lập trình trực quan hóa dữ liệu mà còn đóng góp tài liệu giáo dục chất lượng cao, phù hợp với xu hướng sử dụng tài liệu điện tử và học tập tương tác trong thời đại số.

## CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

### 2.1. Cơ sở lý thuyết

#### 2.1.1. Giới thiệu về trực quan hóa dữ liệu

Trực quan hóa dữ liệu đóng vai trò quan trọng trong phân tích và truyền tải thông tin. Cụ thể:

Hỗ trợ phân tích dữ liệu: Giúp làm rõ các mối quan hệ, xu hướng và mẫu dữ liệu trong các tập dữ liệu lớn. Ví dụ: Biểu đồ đường giúp phát hiện xu hướng theo thời gian, biểu đồ phân tán thể hiện mối tương quan giữa hai biến.

Truyền tải thông tin hiệu quả: Cung cấp hình ảnh trực quan dễ hiểu, hỗ trợ việc trình bày dữ liệu phức tạp trong báo cáo hoặc bài thuyết trình.

Hỗ trợ ra quyết định: Biểu đồ và đồ thị giúp các nhà quản lý hoặc nhà khoa học dữ liệu đưa ra quyết định nhanh chóng và chính xác hơn.

Các nguyên tắc cơ bản:

Màu sắc: Chọn màu sắc dễ phân biệt, hài hòa và phù hợp với đối tượng mục tiêu. Tránh sử dụng quá nhiều màu gây rối.

Bố cục: Đảm bảo sự rõ ràng, trực quan, tập trung vào thông tin chính. Tiêu đề, trực và chú thích cần sắp xếp hợp lý để dễ đọc.

Loại biểu đồ: Chọn loại biểu đồ phù hợp với mục đích sử dụng:

Biểu đồ thanh: So sánh các nhóm dữ liệu.

Biểu đồ tròn: Minh họa tỷ lệ phần trăm.

Biểu đồ phân tán: Tìm mối tương quan giữa hai biến.

#### 2.1.2. Khái quát về Jupyter Book

##### Tổng quan về Jupyter Book

Jupyter Book là công cụ mã nguồn mở giúp tạo tài liệu đa phương tiện kết hợp giữa:

Nội dung văn bản: Viết bằng Markdown để trình bày lý thuyết và khái niệm.

Mã lệnh: Sử dụng Jupyter Notebook để trình bày và thực thi mã Python.

Hình ảnh và biểu đồ: Minh họa dữ liệu trực quan.

Jupyter Book đặc biệt hữu ích trong biên soạn tài liệu học thuật, sách giáo khoa và hướng dẫn lập trình.

### Lợi ích của Jupyter Book

Kết hợp lý thuyết và thực hành: Tích hợp lý thuyết với mã lệnh thực thi trực tiếp, giúp người học hiểu và thực hành dễ dàng.

Khả năng tương tác cao: Cho phép người đọc chạy và chỉnh sửa mã trực tiếp trên tài liệu.

Tổ chức nội dung rõ ràng: Sử dụng Markdown và Notebook để xây dựng cấu trúc chương mục cụ thể.

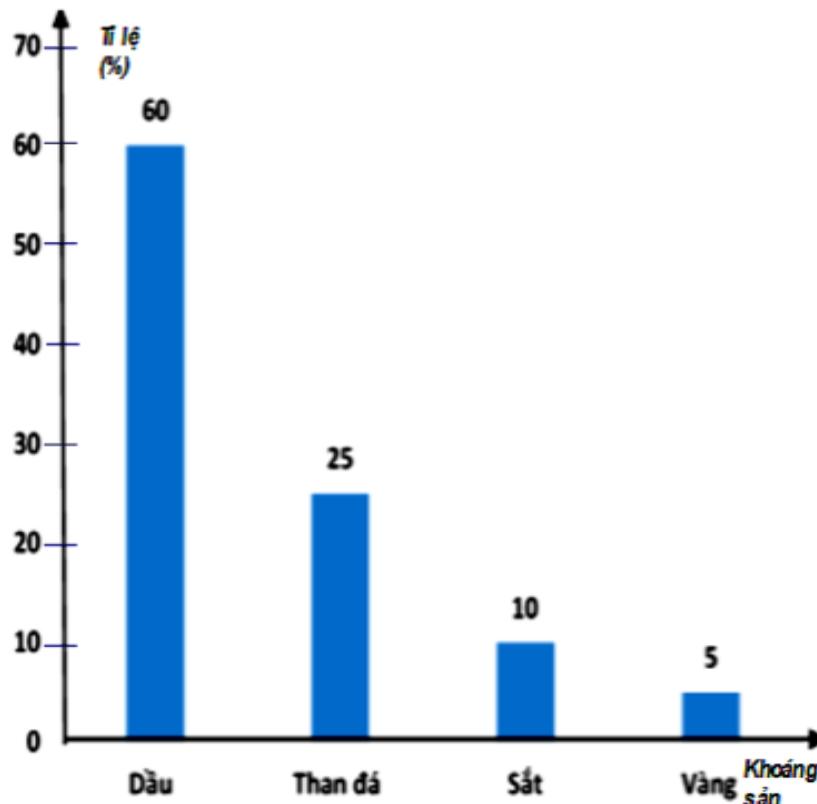
Xuất bản đa dạng: Hỗ trợ xuất bản dưới dạng website, PDF, hoặc ebook.

Tích hợp với hệ sinh thái Python: Hỗ trợ các thư viện như Matplotlib, Seaborn, Plotly để minh họa dữ liệu trực quan.

Trực quan hóa dữ liệu là một công cụ mạnh mẽ giúp chúng ta chuyển đổi dữ liệu thô thành những hình ảnh dễ hiểu, từ đó rút ra những insights sâu sắc. Có rất nhiều kỹ thuật trực quan hóa khác nhau, mỗi kỹ thuật phù hợp với một loại dữ liệu và câu hỏi nghiên cứu khác nhau.

#### Biểu đồ cơ bản:

**Biểu đồ cột:** Sử dụng các cột để biểu diễn dữ liệu theo danh mục. Thường dùng để so sánh các giá trị giữa các danh mục.

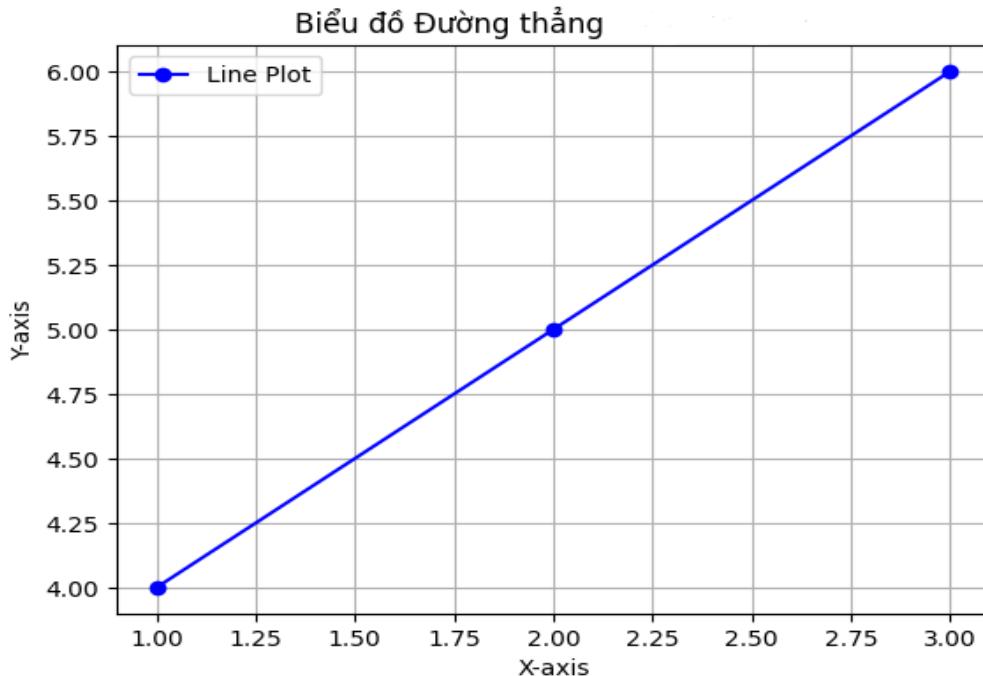


Hình 2.1: Biểu đồ cột

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

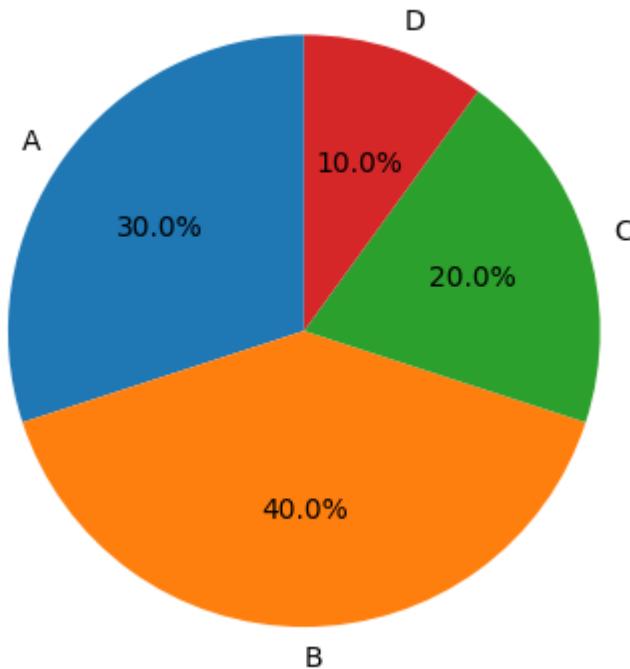
---

**Biểu đồ đường:** Sử dụng đường để thể hiện sự thay đổi của dữ liệu theo thời gian hoặc một biến số liên tục khác. Thường dùng để thể hiện xu hướng.



Hình 2.2: Biểu đồ đường

**Biểu đồ tròn:** Sử dụng các phần của một hình tròn để biểu diễn tỷ lệ phần trăm của các danh mục trong một tổng thể.



Hình 2.3: Biểu đồ tròn

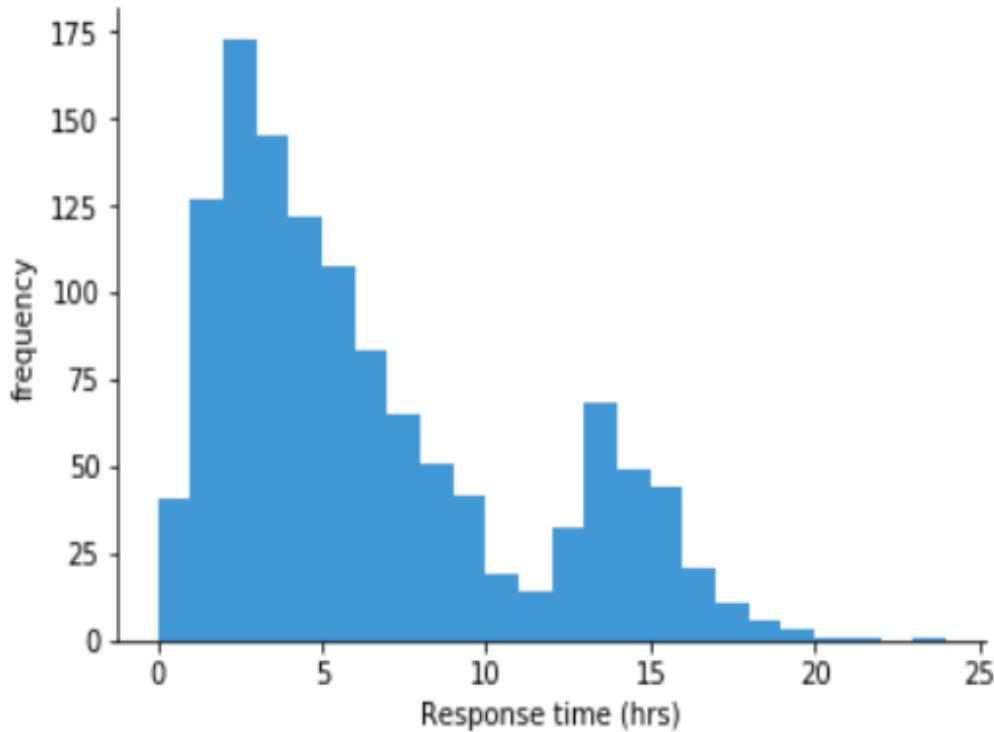
**Biểu đồ phân bố:**

---

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

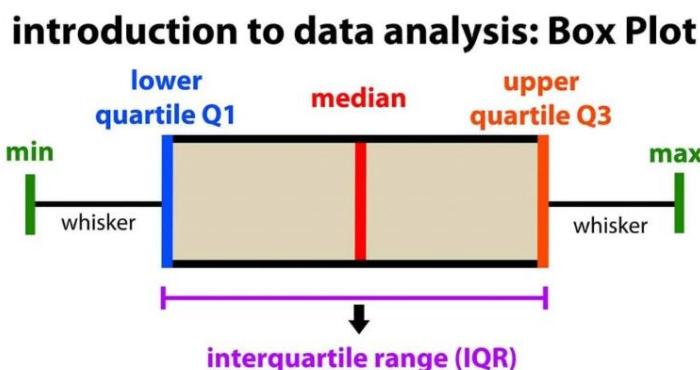
---

**Histogram:** Sử dụng các hình chữ nhật để biểu diễn tần suất của các giá trị trong một khoảng dữ liệu. Thường dùng để xem xét phân phối của dữ liệu.



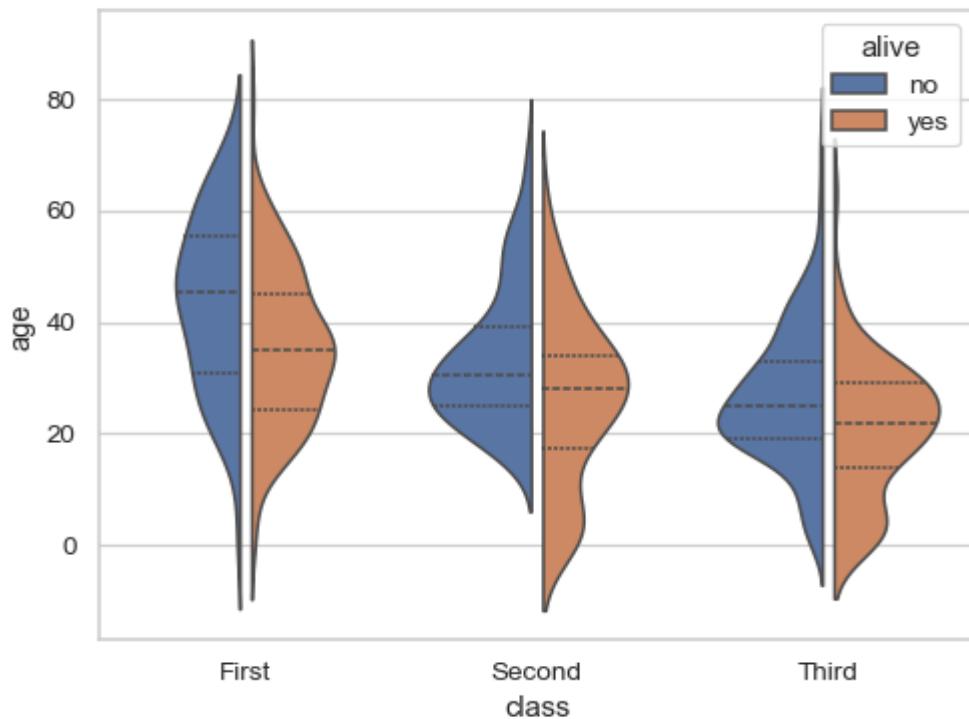
Hình 2.4: Histogram

**Box plot:** Sử dụng một hộp và các đường để biểu diễn phân bố của dữ liệu, bao gồm các giá trị ngoại lệ.



Hình 2.5: Box Plot

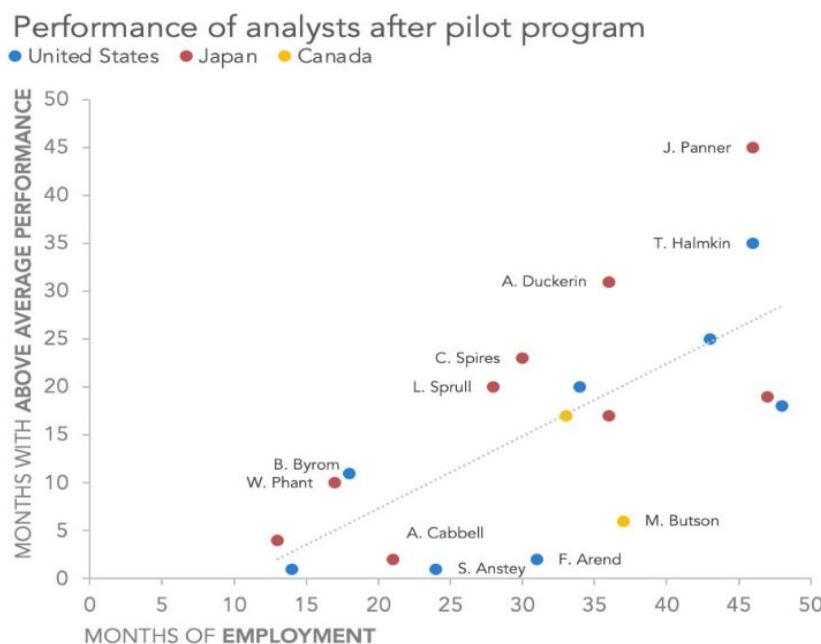
**Violin plot:** Tương tự như box plot nhưng sử dụng mật độ để thể hiện hình dạng phân bố.



Hình 2.6: Violin plot

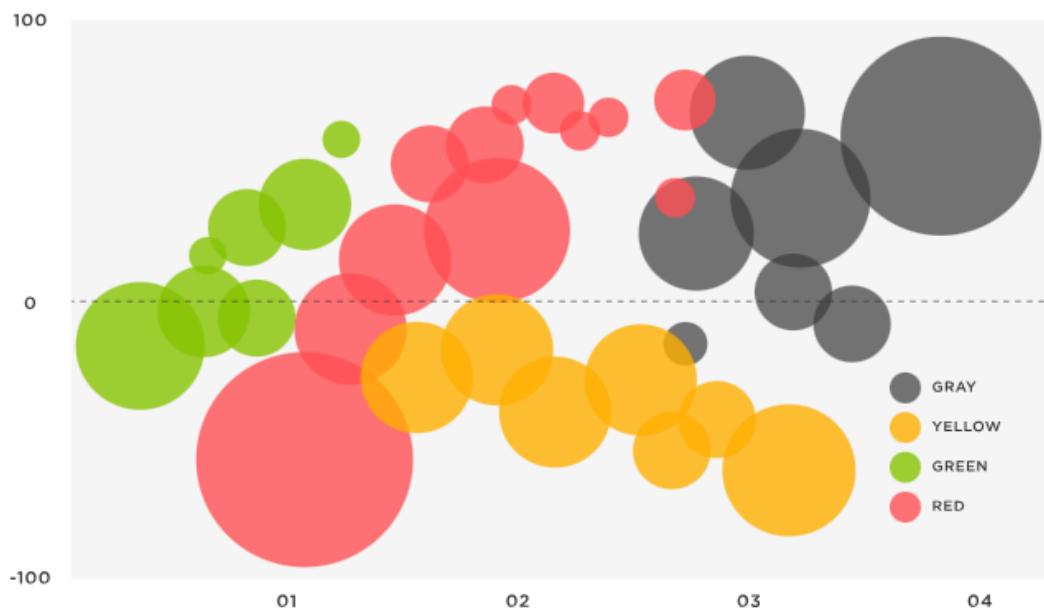
### Biểu đồ mối quan hệ:

**Scatter plot:** Sử dụng các điểm để biểu diễn mối quan hệ giữa hai biến số liên tục



Hình 2.7: Scatter plot

**Bubble chart:** Giống như scatter plot nhưng kích thước của các điểm biểu diễn một biến thứ ba.



Hình 2.8: Bubble chart

**Heatmap:** Sử dụng màu sắc để biểu diễn giá trị của các ô trong một ma trận. Thường dùng để thể hiện mối quan hệ giữa hai biến số phân loại.



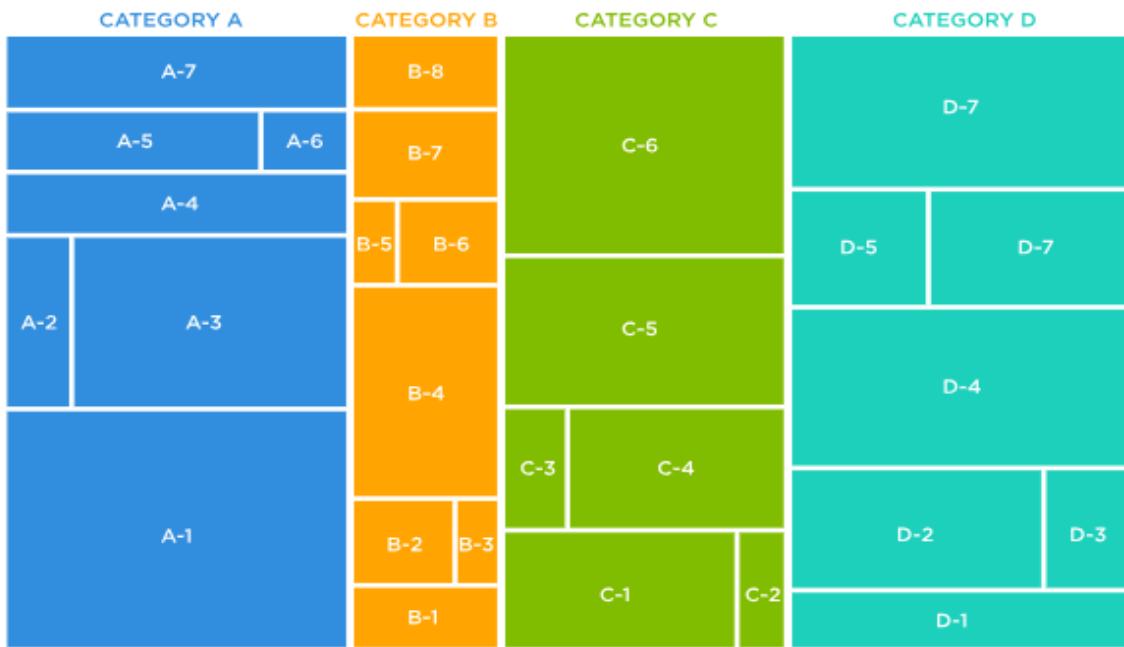
Hình 2.9: Heatmap

### Các kỹ thuật khác:

**Treemap:** Sử dụng các hình chữ nhật có kích thước khác nhau để biểu diễn dữ liệu phân cấp.

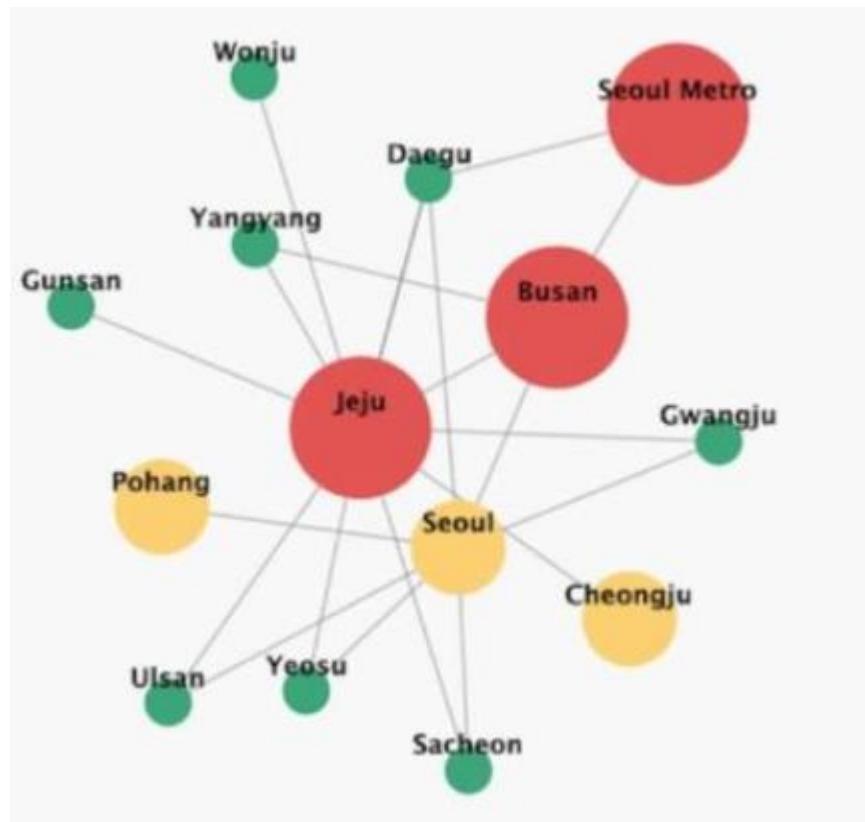
Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---



Hình 2.10: Treemap

**Network graph:** Sử dụng các nút và cạnh để biểu diễn mối quan hệ giữa các đối tượng.



Hình 2.11: Network graph

**Geographic map:** Sử dụng bản đồ để hiển thị dữ liệu địa lý.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python



Hình 2.12: Geographic map

## 2.2. Công cụ và công nghệ sử dụng

### 2.2.1. Jupyter Book

Phiên bản sử dụng và lý do lựa chọn: Phiên bản mới nhất của Jupyter Book được chọn nhằm đảm bảo tích hợp tốt nhất với các công cụ và thư viện Python, đồng thời hỗ trợ xuất bản tài liệu dưới dạng website, PDF, hoặc ebook.

Cách cấu hình: Jupyter Book được cấu hình để hỗ trợ Markdown và Jupyter Notebook, giúp kết hợp nội dung văn bản, mã lệnh và biểu đồ trực quan trong một tài liệu duy nhất. Các bước cấu hình bao gồm cài đặt thư viện cần thiết, thiết lập tệp cấu hình `_config.yml`, và tổ chức cấu trúc thư mục.

### 2.2.2. Ngôn ngữ lập trình Python

#### Các thư viện chính:

Matplotlib: Tạo biểu đồ cơ bản như biểu đồ cột, đường, hoặc hình tròn.

Seaborn: Hỗ trợ tạo các biểu đồ nâng cao, dễ dàng tùy chỉnh với giao diện trực quan.

Plotly: Cung cấp khả năng tạo biểu đồ tương tác như biểu đồ 3D, bản đồ nhiệt, hoặc biểu đồ phân tán động.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

Tích hợp Python với Jupyter Notebook: Python được sử dụng kết hợp với Jupyter Notebook để trình bày lý thuyết cùng mã lệnh thực hành, cho phép người học chạy mã trực tiếp và xem kết quả tức thì.

### 2.2.3. Các công cụ hỗ trợ

Markdown: Ngôn ngữ đơn giản để viết tài liệu lý thuyết với cấu trúc rõ ràng và dễ đọc.

#### Cú pháp cơ bản của Markdown:

Tiêu đề: Sử dụng # để tạo tiêu đề các cấp từ 1 đến 6

```
# Tiêu đề cấp 1  
## Tiêu đề cấp 2  
### Tiêu đề cấp 3  
#### Tiêu đề cấp 4  
##### Tiêu đề cấp 5  
##### Tiêu đề cấp 6
```

Hình 2.13: Tiêu đề 6 cấp

#### Định dạng văn bản:

*In nghiêng*: \*in nghiêng\* hoặc \_in nghiêng\_

**In đậm**: \*\*in đậm\*\*

*In nghiêng và đậm*: \*\*\*in nghiêng và đậm\*\*\*

~~gạch ngang~~

#### Danh sách:

Danh sách không thứ tự:

- Mục 1
- Mục 2
- Mục con

Hình 2.14: Danh sách không có thứ tự

Danh sách có thứ tự:

1. Mục 1
2. Mục 2

Hình 2.15: Danh sách có thứ tự

#### Liên kết và hình ảnh:

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

[Liên kết](<https://example.com>)  
![Mô tả hình ảnh](path/to/image.png)

Hình 2.16: Liên kết hình ảnh

### Code Block:

Bạn có thể sử dụng khối mã với Markdown:

Chèn mã inline:

Đây là một đoạn mã `print("Hello, World!")`.

Hình 2.17: Chèn mã inline

Chèn khối mã:

```
```python
data.rename(columns={'OldName': 'NewName'}, inplace=True)
```
```

Hình 2.18: chèn khối mã

### Liên kết chéo (Cross-referencing) :

Tạo liên kết đến các phần hoặc tài liệu khác trong Jupyter Book:

- [Chương 1: Giới thiệu](chapter1.md)
- [1.1: Mục Tiêu Của Cuốn Sách](chapter1.md#muc-tieu-cua-cuon-sach)
- [1.2: Lợi ích của trực quan hóa dữ liệu](chapter1.md#loi-ich-cua-truc-quan-hoa-du-lieu)
- [1.3: Vì sao chọn python cho trực quan hóa dữ liệu](chapter1.md#vi-sao-chon-python-cho-truc-quan-hoa-du-lieu)
- [1.4: Tổng quan về các chương](chapter1.md#tong-quan-vecac-chuong)
- [1.5: Cách sử dụng cuốn sách](chapter1.md#cach-su-dung-cuon-sach)
- [1.6 Các lĩnh vực ứng dụng của trực quan hóa dữ liệu](chapter1.md#cac-linh-vuc-ung-dung-cua-truc-quan-hoa-du-lieu)
- [1.7 Thực hành và ứng dụng](chapter1.md#thuc-hanh-va-ung-dung)
- [1.8 So sánh các công cụ trực quan hóa](chapter1.md#so-sanh-cac-cong-cu-truc-quan-hoa)
- [1.9 Tương lai của trực quan hóa dữ liệu](chapter1.md#tuong-lai-cua-truc-quan-hoa-du-lieu)
- [1.10 Tổng kết chương 1](chapter1.md#tong-ket-chuong-1)

Hình 2.19: Liên kết chéo

Jupyter Notebook: Công cụ chính để thực thi mã lệnh Python, minh họa kết quả và tích hợp nội dung trực quan.

GitHub/Git: Hỗ trợ quản lý phiên bản, theo dõi lịch sử chỉnh sửa và xuất bản tài liệu trực tuyến.

Visual Studio Code/JupyterLab: Các môi trường soạn thảo và phát triển hiệu quả, tích hợp mạnh mẽ với Jupyter Notebook và các công cụ khác.

Anaconda: là một công cụ phổ biến giúp quản lý các môi trường Python và thư viện, cung cấp các công cụ như Jupyter Notebook. Việc sử dụng Anaconda giúp đơn giản hóa quá trình cài đặt và quản lý các thư viện cần thiết cho dự án, bao gồm Matplotlib, Seaborn, Plotly và các thư viện khác.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

## 2.3. Phần mềm và tài nguyên sử dụng

### 2.3.1. Phần mềm

Python (phiên bản cụ thể): Phiên bản Python được sử dụng trong dự án này là Python 3.x, với các thư viện chính như Matplotlib, Seaborn, Plotly, và Pandas. Việc sử dụng Python 3.x đảm bảo tính tương thích với các thư viện và công cụ mới nhất, đồng thời đáp ứng yêu cầu về hiệu suất và tính năng của các công cụ phân tích và trực quan hóa dữ liệu.

Jupyter Book và các plugin mở rộng: Jupyter Book là công cụ chính để biên soạn tài liệu, cho phép kết hợp nội dung văn bản, mã lệnh Python, và biểu đồ trực quan. Các plugin mở rộng của Jupyter Book có thể được sử dụng để hỗ trợ tính năng xuất bản, tương tác hoặc định dạng tài liệu.

jupyter-book: Plugin chính để tạo và xuất bản tài liệu từ Markdown và Jupyter Notebook.

sphinx-book-theme: Plugin giúp tạo giao diện đẹp và dễ dàng điều hướng cho sách.

nbsphinx: Plugin hỗ trợ tích hợp Jupyter Notebook vào tài liệu Sphinx.

### 2.3.2. Tài nguyên học thuật

#### Các sách và tài liệu tham khảo:

"Python Data Science Handbook" của Jake VanderPlas: Cung cấp nền tảng vững chắc về các thư viện phân tích và trực quan hóa dữ liệu bằng Python, rất hữu ích cho việc hiểu sâu về cách sử dụng Matplotlib, Seaborn, và Plotly.

"Data Visualization with Python and Matplotlib" của Kyran Dale: Tài liệu chi tiết về cách sử dụng Matplotlib trong việc trực quan hóa dữ liệu.

"Interactive Data Visualization with Python" của Abhishek Kumar: Hướng dẫn chi tiết về cách tạo biểu đồ tương tác với Plotly.

#### Nguồn tài nguyên mã nguồn mở từ cộng đồng:

GitHub Repositories: Các mã nguồn mở trên GitHub, như kho lưu trữ ví dụ về trực quan hóa dữ liệu, mẫu Jupyter Book hoặc Jupyter Notebooks, giúp người học và người phát triển có thể tham khảo, học hỏi và đóng góp.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

Kaggle Datasets: Kaggle cung cấp nhiều bộ dữ liệu mở, có thể sử dụng trong các ví dụ minh họa trực quan hóa dữ liệu. Đây là nguồn tài nguyên phong phú để thực hành và thử nghiệm.

Plotly, Seaborn, và Matplotlib Documentation: Tài liệu chính thức từ các thư viện này sẽ cung cấp các hướng dẫn chi tiết về cách sử dụng các thư viện trong việc tạo ra các biểu đồ và đồ thị trực quan.

Jupyter Community: Các diễn đàn và nhóm trực tuyến nơi người dùng có thể thảo luận, trao đổi về các vấn đề và giải pháp khi làm việc với Jupyter Book và các công cụ liên quan.

## 2.4. Mô hình tổ chức nội dung sách

### 2.4.1. Cấu trúc tài liệu

Cuốn sách được tổ chức thành nhiều chương, mỗi chương có mục tiêu rõ ràng và nội dung chi tiết, với mục đích giúp người đọc hiểu và thực hành trực quan hóa dữ liệu bằng Python thông qua các công cụ như Jupyter Book. Dưới đây là cấu trúc của cuốn sách:

#### Chapter 1 (Giới thiệu)

Mục tiêu: Cung cấp cái nhìn tổng quan về trực quan hóa dữ liệu, giải thích tầm quan trọng của việc sử dụng biểu đồ và đồ thị trong việc phân tích và truyền tải thông tin.

Nội dung: Khái niệm cơ bản về trực quan hóa dữ liệu và vai trò của trực quan hóa trong việc hỗ trợ ra quyết định, nhận diện xu hướng và làm rõ các mối quan hệ trong dữ liệu giới thiệu về các loại biểu đồ cơ bản và nâng cao, cùng với những nguyên tắc cơ bản khi thiết kế trực quan.

#### Chapter 2 (Hướng dẫn cài đặt)

Mục tiêu: Hướng dẫn cách cài đặt môi trường và công cụ cần thiết để làm việc với Python và Jupyter Book.

Nội dung: Cài đặt Python, Jupyter Notebook, và các thư viện cần thiết như Matplotlib, Seaborn, Plotly và hướng dẫn cài đặt Jupyter Book và cách cấu hình môi trường làm việc và giới thiệu các công cụ hỗ trợ như Visual Studio Code và GitHub để quản lý phiên bản tài liệu.

#### Chapter 3 (Giới thiệu về Dữ liệu)

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

Mục tiêu: Giới thiệu về các loại dữ liệu phổ biến trong phân tích dữ liệu và cách chuẩn bị dữ liệu để thực hiện trực quan hóa.

Nội dung: Các loại dữ liệu: Dữ liệu số, dữ liệu phân loại, dữ liệu chuỗi thời gian, v.v.... Cách làm sạch và xử lý dữ liệu trước khi đưa vào trực quan hóa và giới thiệu về các bộ dữ liệu mẫu và cách sử dụng chúng trong các bài thực hành.

### **Chapter 4 (Thống kê mô tả)**

Mục tiêu: Hướng dẫn các khái niệm cơ bản trong thống kê mô tả và cách sử dụng chúng để phân tích dữ liệu.

Nội dung: Các phép đo cơ bản: Trung bình, trung vị, phương sai, độ lệch chuẩn và có nhiều cách tính toán và diễn giải các chỉ số thống kê mô tả sử dụng trực quan hóa để minh họa các khái niệm thống kê mô tả.

### **Chapter 5 (Trực quan hóa dữ liệu với Matplotlib)**

Mục tiêu: Giới thiệu về thư viện Matplotlib và cách tạo các biểu đồ cơ bản trong Python.

Nội dung: Cách sử dụng Matplotlib để tạo biểu đồ thanh, đường, hình tròn, biểu đồ phân tán và cách tùy chỉnh biểu đồ: Đổi màu sắc, kiểu đường, tiêu đề, trực, v.v..... Giới thiệu các tính năng nâng cao của Matplotlib như biểu đồ con (subplot) và đồ thị 3D.

### **Chapter 6 (Trực quan hóa dữ liệu với Seaborn)**

Mục tiêu: Hướng dẫn sử dụng thư viện Seaborn để tạo các biểu đồ nâng cao và đẹp mắt.

Nội dung: Tạo biểu đồ phân phối, biểu đồ hộp (box plot), biểu đồ nhiệt (heatmap) và Cách sử dụng các tính năng của Seaborn để tùy chỉnh màu sắc và kiểu dáng biểu đồ và có các kỹ thuật trực quan hóa nâng cao như vẽ các mối quan hệ giữa các biến (pair plots, scatter plots).

### **Chapter 7 (Trực quan hóa dữ liệu với Plotly)**

Mục tiêu: Giới thiệu về thư viện Plotly và cách tạo các biểu đồ tương tác.

Nội dung: Cách tạo biểu đồ tương tác như biểu đồ phân tán động, biểu đồ 3D, và bản đồ nhiệt (heatmap) và tích hợp các tính năng tương tác như zoom, hover thông tin, và thay đổi kiểu dáng và giới thiệu về Plotly Dash để xây dựng ứng dụng trực quan hóa dữ liệu tương tác.

### **Chapter 8 (Kỹ thuật trực quan hóa dữ liệu nâng cao)**

---

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

Mục Tiêu: Cung cấp các kỹ thuật trực quan hóa nâng cao để giải quyết các bài toán phức tạp trong dữ liệu.

Nội dung: Các biểu đồ động, biểu đồ phân loại, và biểu đồ mạng và sử dụng các thư viện nâng cao như Plotly, Bokeh để tạo các biểu đồ tùy chỉnh áp dụng các kỹ thuật trực quan hóa vào các bài toán thực tế như phân tích dữ liệu lớn và mô hình học máy.

### **Chapter 9 (Trực quan hóa trong các dự án)**

Mục tiêu: Hướng dẫn cách áp dụng các kỹ thuật trực quan hóa dữ liệu vào các dự án thực tế.

Nội dung: Cách sử dụng trực quan hóa để hỗ trợ quyết định trong các dự án thực tế (ví dụ: phân tích kinh doanh, nghiên cứu thị trường) và Giới thiệu cách tích hợp trực quan hóa dữ liệu vào các báo cáo, thuyết trình, hoặc ứng dụng web và có các ví dụ thực tế về sử dụng trực quan hóa trong các dự án.

### **Chapter 10 (Tài liệu tham khảo)**

Mục tiêu: Cung cấp các nguồn tài liệu tham khảo hữu ích cho người học và người nghiên cứu.

Nội dung: Các sách, bài báo, và nghiên cứu liên quan đến trực quan hóa dữ liệu và Python và các tài nguyên học trực tuyến, khóa học, và diễn đàn giúp người học tiếp tục mở rộng kiến thức và các tài nguyên mã nguồn mở từ cộng đồng giúp người học nâng cao kỹ năng lập trình và trực quan hóa dữ liệu.

```
format: jb-book
root: intro
chapters:
- file: thu_muc
- file: chapter1
- file: chapter2
- file: chapter3
- file: chapter4
- file: chapter5
- file: chapter6
- file: chapter7
- file: chapter8
- file: chapter9
- file: chapter10
```

Hình 2.20: Cấu trúc tài liệu

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

#### **2.4.2. Phân tích cách sử dụng Jupyter Book**

Tổ Chức File: Markdown (.md) và Notebook (.ipynb): Cuốn sách sử dụng Markdown (.md) cho phần lý thuyết và Jupyter Notebook (.ipynb) cho các ví dụ mã lệnh thực hành và trực quan hóa dữ liệu. Mỗi chương có thể bao gồm cả hai loại file này

Markdown (.md): Dùng để viết nội dung lý thuyết, giải thích khái niệm, hướng dẫn từng bước, và cung cấp tài liệu tham khảo.

Jupyter Notebook (.ipynb): Dùng để triển khai mã lệnh Python, thực hiện các phép toán, và tạo biểu đồ trực quan. Người đọc có thể chạy mã trực tiếp trong Jupyter Notebook và nhìn thấy kết quả tức thì.

Tích Hợp Mã Nguồn, Đồ Thị, và Hình Ảnh Vào Tài Liệu:

Mã nguồn: Các đoạn mã Python sẽ được tích hợp trực tiếp vào trong các notebook, giúp người học không chỉ đọc lý thuyết mà còn thực hành ngay lập tức. Những ví dụ mã nguồn này giúp giải thích các khái niệm lý thuyết và thực hành tạo các biểu đồ.

Đồ thị và hình ảnh: Hình ảnh trực quan, bao gồm các biểu đồ, đồ thị, và ảnh minh họa, sẽ được nhúng vào tài liệu để làm rõ các khái niệm và kết quả của mã lệnh.

## CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

### 3.1. Chuẩn bị công cụ và môi trường làm việc

#### 3.1.1. Cài đặt Python

Cài đặt Python từ <https://www.python.org>

Chạy tệp tải về và làm theo các bước hướng dẫn để cài đặt Python trên máy tính của bạn. Khi cài đặt, nhớ chọn tùy chọn **Add Python to PATH** để dễ dàng sử dụng Python từ dòng lệnh.

Kiểm tra cài đặt Python Mở terminal:

**python –version**



Hình 3.1: Logo Python

Chú Thích: Python là ngôn ngữ lập trình chính được sử dụng trong dự án. Việc cài đặt Python đảm bảo môi trường cần thiết để chạy mã lệnh và thực hiện các bài tập lập trình.

#### 3.1.2. Cài đặt Jupyter Book:

Sau khi cài đặt Python, mở terminal trên Visual studio code và cài đặt Jupyter Book bằng pip:

```
PS D:\110122219> pip install -U jupyter-book
```

Kiểm tra cài đặt Jupyter Book bằng cách chạy lệnh sau:

```
PS D:\110122219> jupyter-book --version
```



Hình 3.2: Logo Jupyter Book

Chú thích: Jupyter Book là công cụ dùng để biên soạn và xuất bản sách, hỗ trợ tích hợp nội dung lý thuyết, mã nguồn, và trực quan hóa dữ liệu trong cùng một tài liệu.

### 3.1.3. Cài đặt các thư viện hỗ trợ (Matplotlib, Seaborn, Plotly):

```
PS D:\110122219> pip install matplotlib
```

```
PS D:\110122219> pip install seaborn
```

```
PS D:\110122219> pip install plotly
```

Chú thích: Các thư viện này cung cấp công cụ mạnh mẽ để trực quan hóa dữ liệu, tạo biểu đồ và đồ thị từ cơ bản đến nâng cao.

### 3.1.4. Cài đặt Anaconda để quản lý môi trường làm việc:

Tải và cài đặt Anaconda từ <https://www.anaconda.com>

Tạo môi trường mới với Python 3.10:

```
conda create --name book_env python=3.10
```

```
conda activate book_env
```

[Download & Install](#)



Hình 3.3: Logo Anaconda

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

Chú thích: Anaconda giúp quản lý môi trường Python, cài đặt thư viện, và hỗ trợ làm việc với Jupyter Notebook một cách dễ dàng, hiệu quả.

### 3.1.5. Cài đặt Visual Studio Code để soạn thảo nội dung sách:

Tải Visual Studio Code từ <https://code.visualstudio.com>

Cài đặt các tiện ích mở rộng như Python và Markdown All-in-One.



Hình 3.4: Logo Visual Studio Code

Chú thích: Visual Studio Code là môi trường phát triển tích hợp (IDE) hỗ trợ việc soạn thảo nội dung, quản lý mã nguồn, và chỉnh sửa tệp Markdown hoặc Notebook thuận tiện.

## 3.2. Xây dựng cấu trúc và nội dung sách

Tạo thư mục dự án Jupyter Book

|                |              |
|----------------|--------------|
| chapter5.ipynb | chapter1.md  |
| chapter6.ipynb | chapter2.md  |
| chapter7.ipynb | chapter3.md  |
| chapter8.ipynb | chapter4.md  |
| chapter9.ipynb | chapter10.md |

Hình 3.5: Thư mục dự án

Thư mục dự án tổ chức nội dung Jupyter Book một cách rõ ràng, hỗ trợ biên dịch sách, dễ dàng bảo trì, mở rộng, và tạo điều kiện thuận lợi cho cộng tác hiệu quả.

Tạo tệp \_config.yml để cấu hình sách:

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

```
mybook > !_config.yml
1  # Book settings
2  # Learn more at https://jupyterbook.org/customize/config.html
3
4  title: My sample book
5  author: The Jupyter Book Community
6  logo: logo.png
7
8  # Force re-execution of notebooks on each build.
9  # See https://jupyterbook.org/content/execute.html
10 execute:
11   | execute_notebooks: force
12
13 # Define the name of the latex output file for PDF builds
14 latex:
15   | latex_documents:
16     |   targetname: book.tex
17
18 # Add a bibtex file so that we can create citations
19 bibtex_bibfiles:
20   | - references.bib
21
22 # Information about where the book exists on the web
23 repository:
24   | url: https://github.com/executablebooks/jupyter-book # Online location of your book
25   | path_to_book: docs # Optional path to your book, relative to the repository root
26   | branch: master # Which branch of the repository should be used when creating links (optional)
```

Hình 3.6: \_config.yml cấu hình sách

Tệp \_config.yml cấu hình các thiết lập chính cho Jupyter Book, bao gồm tiêu đề, tác giả, mục lục, giao diện, tính năng mở rộng, và định dạng xuất bản, giúp kiểm soát cách sách hiển thị và hoạt động.

Tạo tệp \_toc.yml để định nghĩa cấu trúc nội dung.

```
> !_toc.yml
# Table of contents
# Learn more at https://jupyterbook.org/customize/toc.html

format: jb-book
root: intro
chapters:
- file: thu_muc
- file: chapter1
- file: chapter2
- file: chapter3
- file: chapter4
- file: chapter5
- file: chapter6
- file: chapter7
- file: chapter8
- file: chapter9
- file: chapter10
```

Hình 3.7: \_toc.yml Cấu trúc nội dung

Tệp \_toc.yml xác định cấu trúc nội dung Jupyter Book, bao gồm thứ tự các chương, mục, và các tệp liên kết để tổ chức sách một cách hợp lý.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

### 3.3. Tích hợp mã nguồn và trực quan hóa dữ liệu

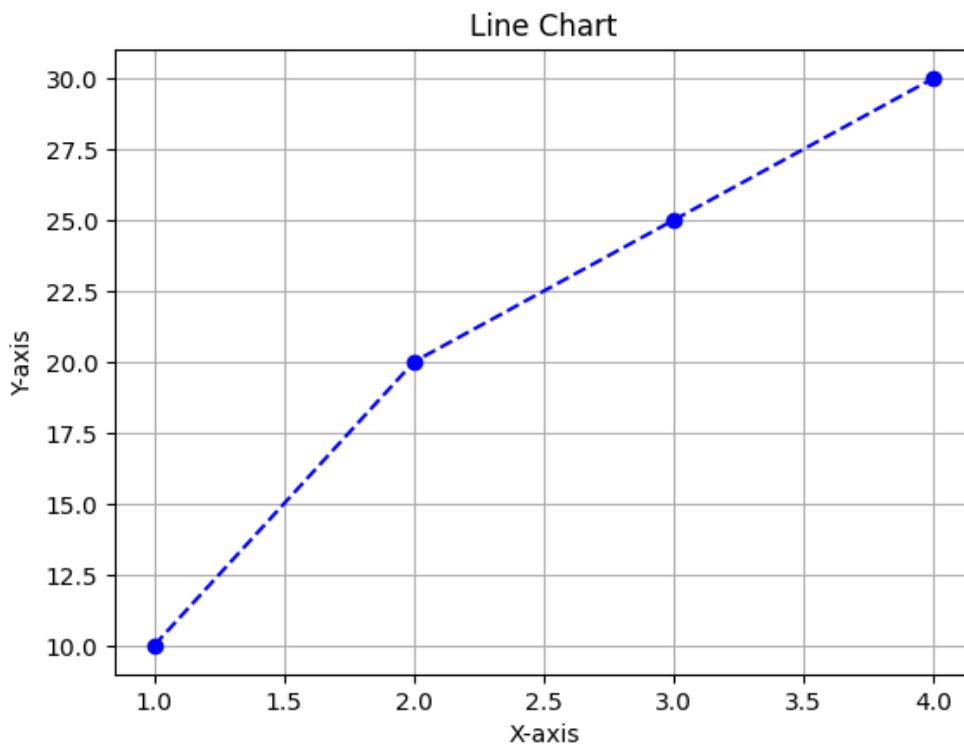
Tạo biểu đồ với Matplotlib, Seaborn, và Plotly:

Matplotlib: Biểu đồ đường thẳng, tinh, cơ bản.

```
import matplotlib.pyplot as plt

# Dữ liệu
x = [1, 2, 3]
y = [4, 5, 6]

# Vẽ biểu đồ
plt.plot(x, y, marker='o', linestyle='-', color='blue', label='Line Plot')
plt.title('Biểu đồ Đường thẳng với Matplotlib')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)
plt.show()
```



Hình 3.8: Biểu đồ đường Matplotlib

Chú thích: Biểu đồ đường Matplotlib hướng dẫn cách tạo biểu đồ đường cơ bản và tùy chỉnh trong Matplotlib.

### Tạo bộ cục nhiều biểu đồ (Subplots)

```
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]

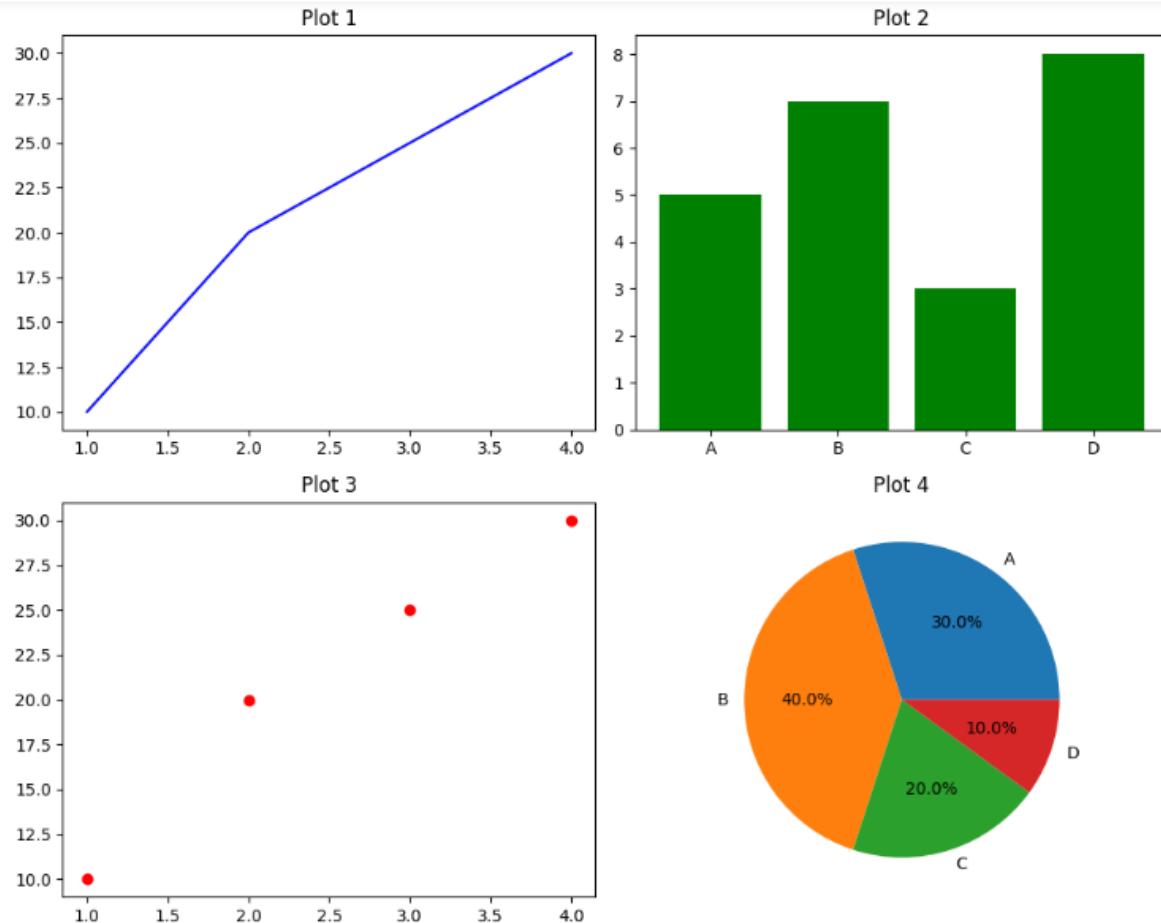
fig, axs = plt.subplots(2, 2, figsize=(10, 8))
axs[0, 0].plot(x, y, color='blue') # Biểu đồ đường
axs[0, 0].set_title("Plot 1")

categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]
axs[0, 1].bar(categories, values, color='green') # Biểu đồ cột
axs[0, 1].set_title("Plot 2")

axs[1, 0].scatter(x, y, color='red') # Biểu đồ tán xạ
axs[1, 0].set_title("Plot 3")

sizes = [30, 40, 20, 10]
labels = ['A', 'B', 'C', 'D']
axs[1, 1].pie(sizes, labels=labels, autopct='%1.1f%%') # Biểu đồ hình tròn
axs[1, 1].set_title("Plot 4")

plt.tight_layout()
plt.show()
```



Hình 3.9: Tạo bộ cục nhiều biểu đồ

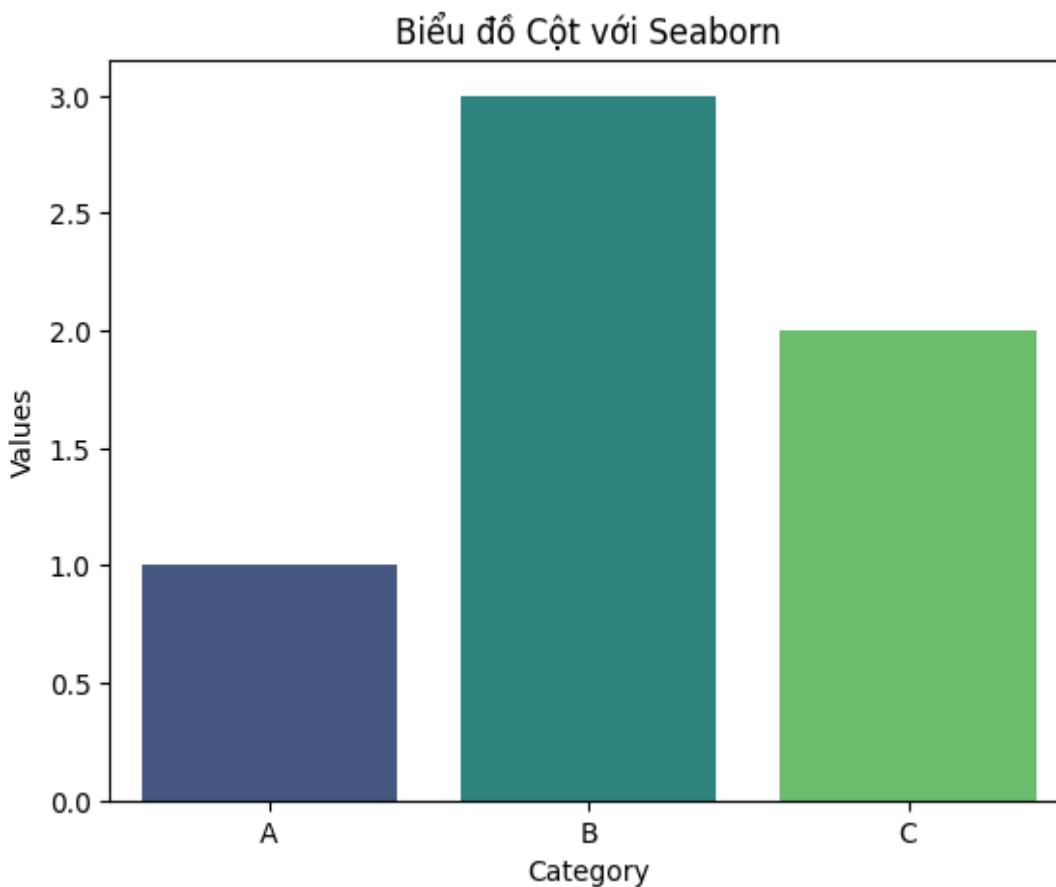
Seaborn: Biểu đồ cột, trực quan và màu sắc dễ nhìn.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Dữ liệu
data = {
    "Category": ["A", "B", "C"],
    "Values": [1, 3, 2]
}

# Tạo DataFrame từ dữ liệu
import pandas as pd
df = pd.DataFrame(data)

# Vẽ biểu đồ cột
sns.barplot(data=df, x="Category", y="Values", palette="viridis")
plt.title('Biểu đồ Cột với Seaborn')
plt.show()
```

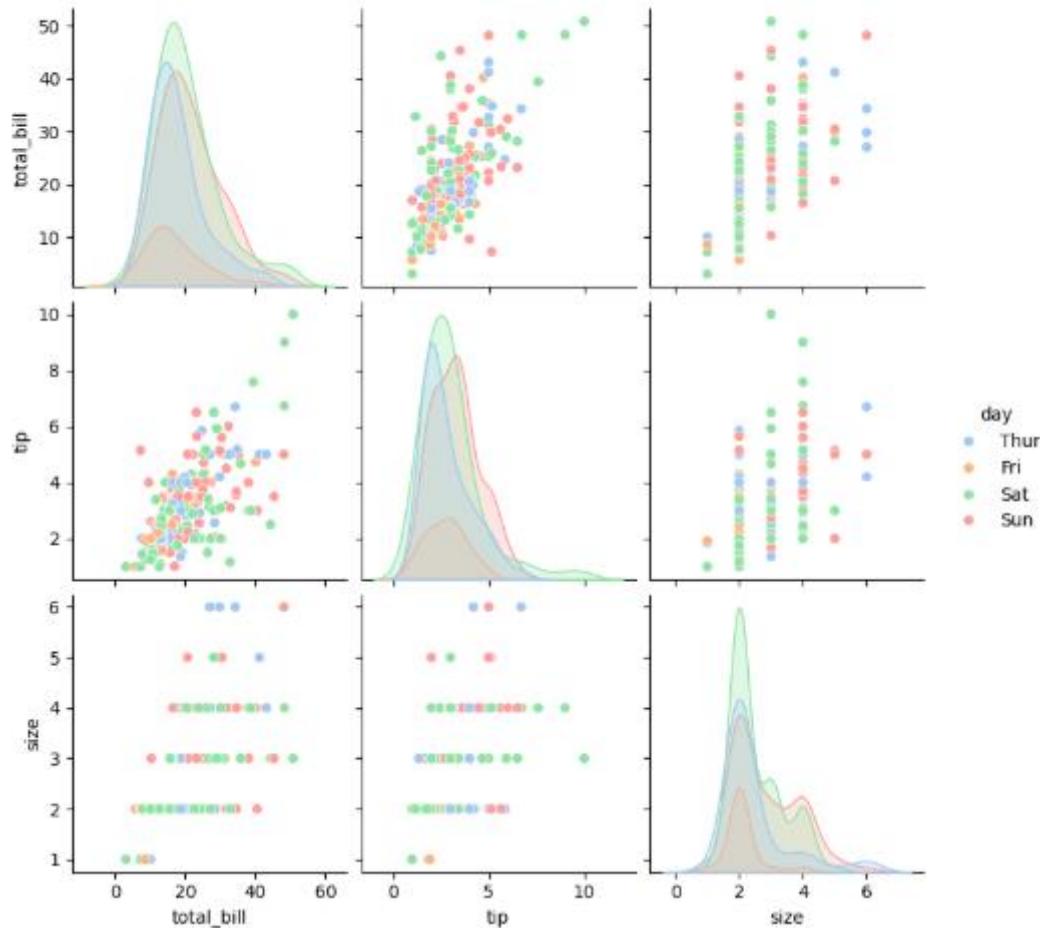


Hình 3.10: Biểu đồ cột Seaborn

Chú thích: Biểu đồ cột Seaborn minh họa cách tạo và tùy chỉnh biểu đồ cột sử dụng Seaborn, với dữ liệu trực quan hơn.

Biểu đồ ghép (Pair Plot) Hiển thị mối quan hệ giữa tất cả các cặp biến.

```
sns.pairplot(tips, hue="day", palette="pastel")
plt.show()
```



Hình 3.11: Biểu đồ ghép

Chú thích: Mỗi ô trong biểu đồ đại diện cho một cặp biến số. Đường chéo chính chứa các biểu đồ phân phối, còn các ô khác là các biểu đồ tán xạ giữa hai biến.

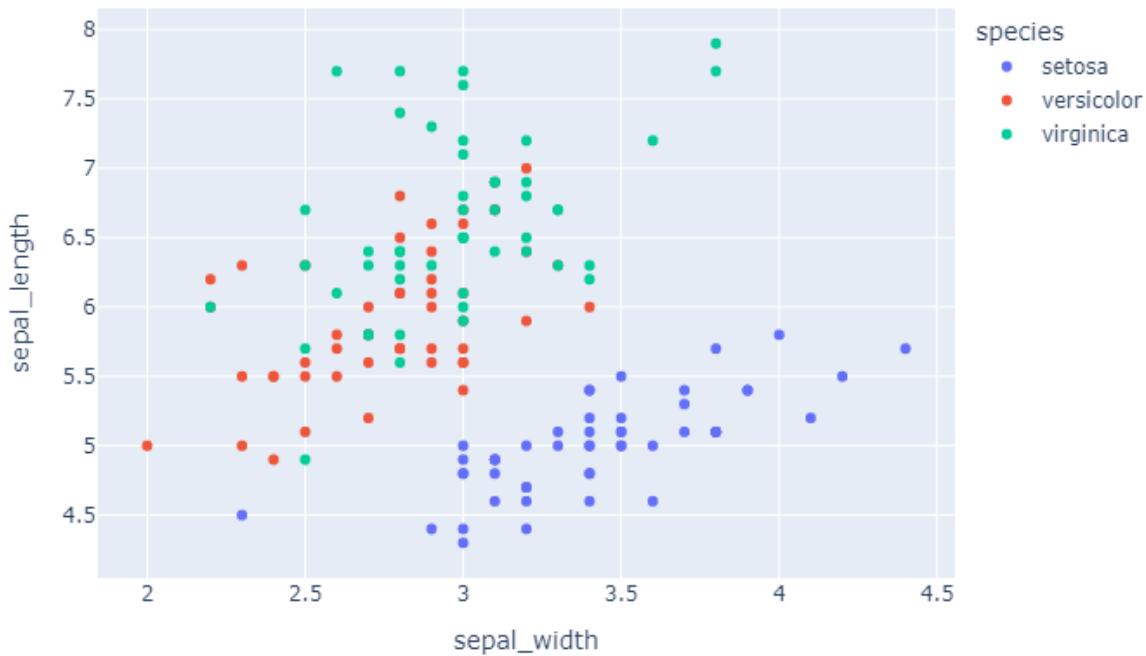
Plotly: Biểu đồ đường tương tác, cho phép zoom, pan, và xuất dữ liệu.:

```
import plotly.express as px

df = px.data.iris()
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species",
                  title="Biểu đồ Phân tán")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

Biểu đồ Phân tán



Hình 3.11: Biểu đồ tương tác Plotly

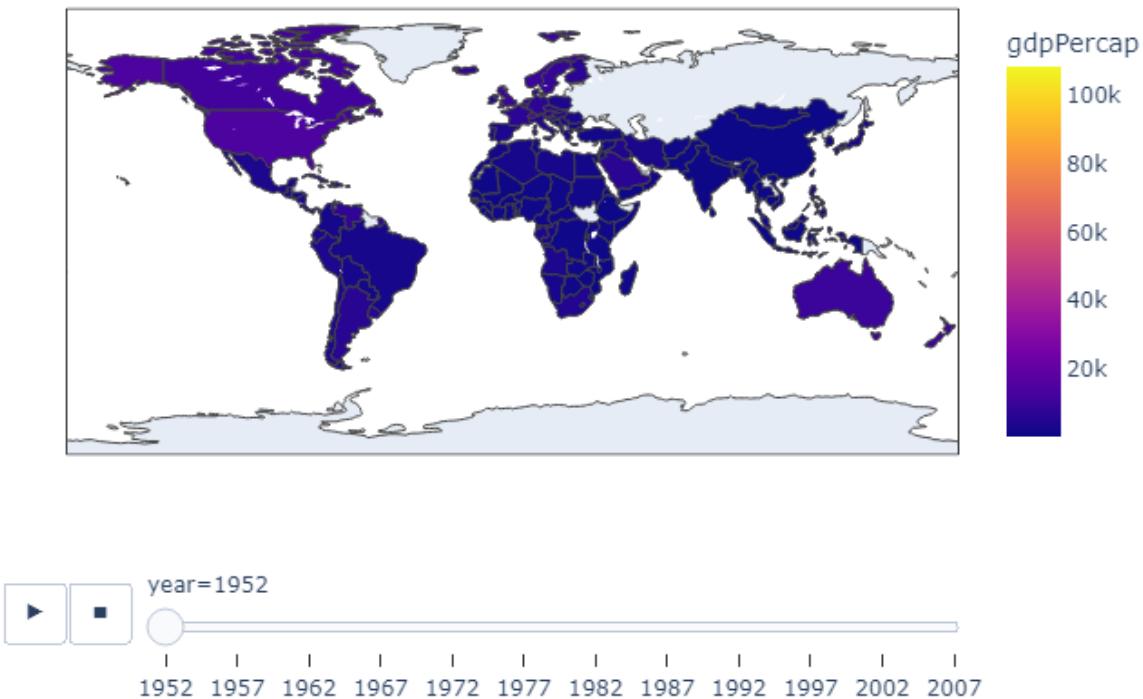
Chú Thích: Biểu đồ tương tác Plotly : Tạo các biểu đồ tương tác như biểu đồ đường, biểu đồ cột trong Plotly.

Biểu đồ bản đồ (Choropleth Map):

```
fig = px.choropleth(px.data.gapminder(),
                     locations="iso_alpha",
                     color="gdpPercap",
                     hover_name="country",
                     animation_frame="year",
                     title="Biểu đồ Bản đồ")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

Biểu đồ Bản đồ



Hình 3.12: Biểu đồ bản đồ

Chú thích: px.choropleth(): Tạo biểu đồ bản đồ, trong đó các quốc gia được tô màu theo GDP.

### 3.4. Xây dựng và xuất bản sách

Sử dụng lệnh jupyter-book build

**jupyter-book build my-book/**

Kiểm tra nội dung sách ở thư mục \_build/html.

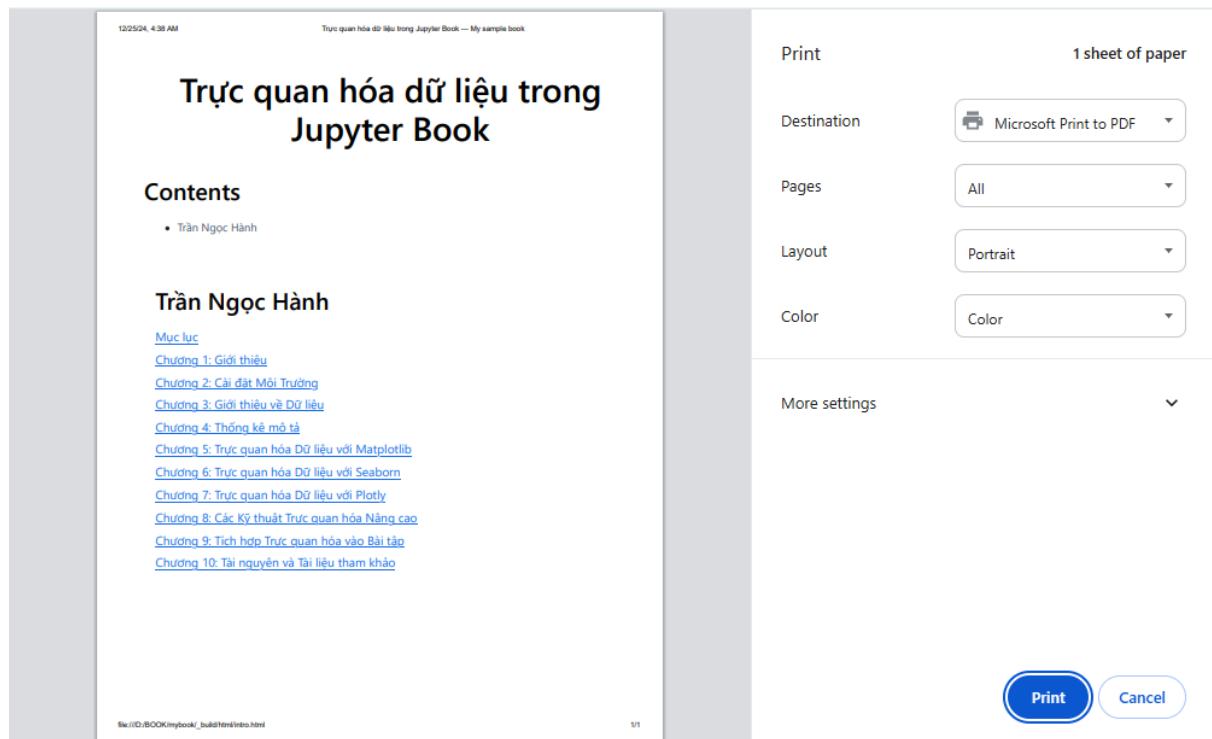
---

```
=====  
Finished generating HTML for book.  
Your book's HTML pages are here:  
    mybook\_\_build\html\  
You can look at your book by opening this file in a browser:  
    mybook\_\_build\html\index.html  
Or paste this line directly into your browser bar:  
    file:///D:/BOOK/mybook\_\_build\html\index.html  
=====
```

Hình 3.13: Kiểm tra nội dung sách

Chú thích: Kiểm tra nội dung sách: Đánh giá và chỉnh sửa nội dung hoàn chỉnh của sách trước khi xuất bản.

Xuất bản sách sang định dạng PDF:



Hình 3.14: Xuất bản sách sang định dạng PDF

Chú thích: Xuất bản sách sang định dạng PDF giúp dễ dàng phân phối, bảo toàn định dạng nội dung, hỗ trợ in ấn và lưu trữ lâu dài.

## CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

### 4.1. Kết quả đạt được

Tạo tài liệu hướng dẫn hoàn chỉnh: Đã biên soạn và xuất bản một tài liệu hướng dẫn chi tiết, bao gồm các chương mục từ lý thuyết cơ bản đến thực hành nâng cao và tài liệu được biên soạn dưới dạng một Jupyter Book, cung cấp giao diện trực quan và tích hợp các ví dụ mã lệnh thực thi trực tiếp.

Cấu trúc nội dung chặt chẽ, logic: Các chương được tổ chức rõ ràng, bắt đầu từ khái niệm cơ bản (cách cài đặt Python, giới thiệu các thư viện) đến các kỹ thuật trực quan hóa phức tạp (biểu đồ động, tích hợp Dashboards) và Tài liệu hỗ trợ cả người mới bắt đầu và những người có kinh nghiệm trong lập trình dữ liệu.

Thực hiện trực quan hóa dữ liệu bằng Python: Đã tạo ra các biểu đồ mẫu minh họa cho từng thư viện:

Matplotlib: Biểu đồ đường, biểu đồ cột, biểu đồ phân tán.

Seaborn: Biểu đồ hộp, heatmap, biểu đồ violin.

Plotly: Biểu đồ tương tác, biểu đồ 3D.

Bokeh: Biểu đồ tương tác nâng cao.

Tích hợp các ví dụ với dữ liệu thực tế để người học dễ hình dung và thực hành.

Hướng dẫn xuất bản và chia sẻ Jupyter Book: Đã hướng dẫn chi tiết cách xuất bản Jupyter Book lên các nền tảng như GitHub Pages và xuất tài liệu dưới dạng HTML hoặc PDF và đảm bảo tài liệu có thể dễ dàng truy cập và chia sẻ với cộng đồng.

### 4.2. Đánh giá hiệu năng

Hiệu năng của tài liệu: Tài liệu được tối ưu hóa để tải nhanh trên trình duyệt nhờ việc sử dụng định dạng HTML và người dùng có thể chạy các đoạn mã trực tiếp trong tài liệu mà không cần cài đặt thêm công cụ, nhờ tích hợp với Binder.

Hiệu năng của các thư viện trực quan hóa đã thử nghiệm các thư viện trực quan hóa trên tập dữ liệu lớn và đánh giá thời gian xử lý, tốc độ hiển thị biểu đồ:

Matplotlib và Seaborn: Tốc độ nhanh với dữ liệu nhỏ và trung bình.

Plotly: Cung cấp các biểu đồ tương tác mạnh mẽ, tuy nhiên tốc độ giảm khi xử lý dữ liệu lớn.

Bokeh: Hiệu suất tốt, đặc biệt khi sử dụng với ứng dụng web.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

### 4.3. Trải nghiệm người dùng

Dễ sử dụng: Giao diện của Jupyter Book thân thiện, người dùng có thể dễ dàng điều hướng giữa các chương mục và các đoạn mã được viết rõ ràng, kèm theo giải thích chi tiết, giúp người học dễ dàng tiếp cận.

Phản hồi từ thử nghiệm thực tế: Trong quá trình thử nghiệm với một nhóm sinh viên và lập trình viên, tài liệu nhận được phản hồi tích cực về tính trực quan và dễ hiểu và người dùng đánh giá cao phần ví dụ minh họa thực tế và hướng dẫn từng bước cụ thể.

### 4.4. Giao diện chức năng nghiên cứu

Giao diện của Jupyter Book: trang chủ hiển thị tổng quan nội dung của tài liệu, bao gồm mục lục và hướng dẫn sử dụng.

Chương mục chi tiết: Các trang nội dung được trình bày với tiêu đề rõ ràng, tích hợp hình ảnh và biểu đồ minh họa.

Chạy mã trực tiếp: Mỗi ví dụ mã đều có thể chạy trực tiếp thông qua Binder, cho phép người dùng tương tác với mã mà không cần cài đặt.

Biểu đồ minh họa: Tích hợp nhiều biểu đồ tương tác giúp người dùng dễ dàng tùy chỉnh và khám phá dữ liệu.

Xuất bản và chia sẻ: Tài liệu được triển khai trên GitHub Pages, với các liên kết dễ truy cập và hỗ trợ tải xuống ở định dạng PDF.

### 4.5. Những khó khăn và cách khắc phục

Khó khăn:

Việc tối ưu hóa hiệu năng khi sử dụng dữ liệu lớn với các thư viện trực quan hóa.

Một số tính năng của Jupyter Book đòi hỏi cấu hình phức tạp, đặc biệt là khi xuất bản trên GitHub Pages

Cách khắc phục:

Sử dụng các kỹ thuật giảm kích thước dữ liệu và tối ưu hóa mã nguồn.

Tham khảo tài liệu chính thức của Jupyter Book và các diễn đàn hỗ trợ để giải quyết các vấn đề kỹ thuật.

### 4.6. Một số nội dung của tài liệu đã biên soạn

Tham khảo phần phụ lục

---

## CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1. Kết luận

**Mục tiêu đạt được:** Tài liệu hướng dẫn lập trình trực quan hóa dữ liệu bằng Python đã giúp người học hiểu và thực hành các kỹ thuật trực quan hóa dữ liệu với các thư viện như Matplotlib, Seaborn, và Plotly. Các khái niệm từ cơ bản đến nâng cao đã được giải thích rõ ràng và minh họa qua các ví dụ thực tế

**Hiệu quả:** Qua quá trình triển khai, tài liệu đã hỗ trợ người học nắm vững các kỹ thuật trực quan hóa và áp dụng chúng vào các dự án thực tế. Phản hồi tích cực từ người học cho thấy tài liệu là công cụ hữu ích cho việc học và thực hành.

**Đóng góp:** Tài liệu góp phần nâng cao khả năng sử dụng các công cụ Python trong việc phân tích và trực quan hóa dữ liệu, hỗ trợ việc đưa ra quyết định trong nghiên cứu và công việc.

### 5.2. Hướng phát triển

**Mở rộng ví dụ thực tế:** Trong tương lai, tài liệu có thể bổ sung thêm các bài toán thực tế, như phân tích dữ liệu lớn hoặc trực quan hóa dữ liệu từ các nguồn không gian (ví dụ: dữ liệu không gian địa lý).

**Cập nhật công nghệ mới:** Liên tục cập nhật các thư viện và công cụ mới trong Python để tài liệu luôn phản ánh các xu hướng mới trong lĩnh vực trực quan hóa dữ liệu, như việc tích hợp với các công cụ AI và machine learning.

**Phát triển phiên bản nâng cao:** Cung cấp thêm các kỹ thuật nâng cao trong việc trực quan hóa dữ liệu, như trực quan hóa dữ liệu động, 3D hoặc trực quan hóa dữ liệu phức tạp.

**Tăng tính tương tác:** Phát triển thêm các ví dụ về biểu đồ tương tác (interactive visualizations) để người học có thể trực tiếp thao tác và trải nghiệm dữ liệu.

## **DANH MỤC TÀI LIỆU THAM KHẢO**

- [1] P. Jupyter, Jupyter Book Documentation, Hoa Kỳ: (tổ chức phi lợi nhuận) và cộng đồng mã nguồn mở., 2018.
- [2] P. Jupyter, Jupyter Book Tutorial, Hoa Kỳ: tài liệu này đã được phát triển song song với Jupyter Book., 2018.
- [3] P. Jupyter, Creating Interactive Data Science Content, Hoa Kỳ: tài liệu này đã được xuất bản để hướng dẫn người dùng về việc tạo nội dung tương tác trong Jupyter Book., 2020.
- [4] R. Python, Data Visualization with Matplotlib and Seaborn, Hoa Kỳ: bài viết này được xuất bản trên Real Python., 2020.
- [5] Plotly, Plotly Python Documentation, Hoa Kỳ: Tài liệu chính thức của Plotly liên tục được cập nhật, nhưng Plotly bản ổn định đầu tiên được phát hành vào năm 2013., 2013.
- [6] T. D. Science, Jupyter Book for Data Science, Hoa Kỳ: bài viết này được xuất bản trên Towards Data Science., 2021.

Tìm hiểu Jupyter Book và biên soạn tài liệu hướng dẫn lập trình trực quan hóa dữ liệu trên Python

---

## PHỤ LỤC

# Mục lục

- [Chương 1: Giới thiệu](#)
  - [1.1: Mục Tiêu Của Cuốn Sách](#)
  - [1.2: Lợi ích của trực quan hóa dữ liệu](#)
  - [1.3: Vì sao chọn python cho trực quan hóa dữ liệu](#)
  - [1.4: Tổng quan về các chương](#)
  - [1.5: Cách sử dụng cuốn sách](#)
  - [1.6 Các lĩnh vực ứng dụng của trực quan hóa dữ liệu](#)
  - [1.7 Thực hành và ứng dụng](#)
  - [1.8 So sánh các công cụ trực quan hóa](#)
  - [1.9 Tương lai của trực quan hóa dữ liệu](#)
  - [1.10 Tổng kết chương 1](#)
- [Chương 2: Cài đặt Môi trường](#)
  - [2.1 Lựa chọn môi trường phát triển](#)
  - [2.2 Cài đặt Python](#)
  - [2.3 Cài đặt Jupyter Notebook](#)
  - [2.4 Cài đặt các thư viện cần thiết](#)
  - [2.5 Kiểm tra môi trường](#)
  - [2.6 Một số lưu ý](#)
  - [2.7 Tổng kết chương 2](#)
- [Chương 3: Giới thiệu về Dữ liệu](#)
  - [3.1 Dữ liệu là gì?](#)
  - [3.2 Các nguồn dữ liệu phổ biến](#)
  - [3.3 Cách làm việc với dữ liệu trong Python](#)
  - [3.4 Làm sạch dữ liệu](#)
  - [3.5 Tổng hợp dữ liệu](#)
  - [3.6 Xuất dữ liệu](#)
  - [3.7 Các loại dữ liệu nâng cao](#)
  - [3.8 Các công cụ và phần mềm hỗ trợ làm việc với dữ liệu](#)

- [3.9 Các phương pháp phân tích dữ liệu](#)
- [3.10 Thực hành nâng cao](#)
- [3.11 Các thách thức trong làm việc với dữ liệu](#)
- [3.12 Tương lai của dữ liệu](#)
- [3.13 Thực hành](#)
- [3.14 Tổng kết chương 3](#)
- [Chương 4: Thống kê mô tả](#)
  - [4.1 Thống kê mô tả là gì?](#)
  - [4.2 Các chỉ số trung tâm](#)
  - [4.3 Các chỉ số phân tán](#)
  - [4.4 Hình dạng phân phối](#)
  - [4.5 Tóm tắt dữ liệu bằng Pandas](#)
  - [4.6 Trực quan hóa thống kê mô tả](#)
  - [4.7 Thực hành](#)
  - [4.8 Thực tiễn ứng dụng của thống kê mô tả](#)
  - [4.9 Những sai lầm thường gặp](#)
  - [4.10 So sánh với thống kê suy diễn](#)
  - [4.11 Công cụ và thư viện](#)
  - [4.12 Nghiên cứu điển hình](#)
  - [4.13 Tổng kết chương 4](#)
- [Chương 5: Trực quan hóa Dữ liệu với Matplotlib](#)
  - [5.1 Giới thiệu về Matplotlib](#)
  - [5.2 Biểu đồ cơ bản với Matplotlib](#)
  - [5.3 Tùy chỉnh biểu đồ](#)
  - [5.4 Sử dụng Object-Oriented API](#)
  - [5.5 Trực quan hóa nhiều biểu đồ](#)
  - [5.6 Lưu biểu đồ](#)
  - [5.7 Thực hành](#)
  - [5.8 Tổng kết chương 5](#)
- [Chương 6: Trực quan hóa Dữ liệu với Seaborn](#)
  - [6.1 Giới thiệu về Seaborn](#)
  - [6.2 Các biểu đồ cơ bản với Seaborn](#)

- [6.3 Các biểu đồ nâng cao](#)
- [6.4 Tùy chỉnh và cài đặt giao diện](#)
- [6.5 Lưu biểu đồ](#)
- [6.6 Thực hành](#)
- [6.7 Tổng kết chương 6](#)
- [Chương 7: Trực quan hóa Dữ liệu với Plotly](#)
  - [7.1 Giới thiệu về Plotly](#)
  - [7.2 Tạo Biểu Đồ Cơ Bản với Plotly](#)
  - [7.3 Các Biểu Đồ Tương Tác](#)
  - [7.4 Tùy chỉnh Biểu Đồ](#)
  - [7.5 Tạo Biểu Đồ Nâng Cao](#)
  - [7.6 Xuất và Chia Sẻ Biểu Đồ](#)
  - [7.7 Các Ví Dụ Thực Tế](#)
  - [7.8 Tài liệu tham khảo](#)
  - [7.9 Tổng kết chương 7](#)
- [Chương 8: Các Kỹ thuật Trực quan hóa Nâng cao](#)
  - [8.1 Giới thiệu về Kỹ thuật Trực quan hóa Nâng cao](#)
  - [8.2 Trực quan hóa Dữ liệu 3D](#)
  - [8.3 Heatmaps và Biểu đồ Ma trận](#)
  - [8.4 Network Graphs \(Biểu đồ Mạng\)](#)
  - [8.5 Tùy chỉnh và Kết hợp Biểu đồ](#)
  - [8.6 Lưu trữ và Chia sẻ Biểu đồ](#)
  - [8.7 Bài tập Thực hành](#)
  - [8.8 Tổng kết chương 8](#)
- [Chương 9: Tích hợp Trực quan hóa vào Bài tập](#)
  - [9.1 Giới thiệu về Tích hợp Trực quan hóa](#)
  - [9.2 Tích hợp Biểu đồ vào Báo cáo Word hoặc PDF](#)
  - [9.3 Tích hợp Biểu đồ vào Jupyter Notebook](#)
  - [9.4 Xây dựng Báo cáo Trực quan](#)
  - [9.5 Ứng dụng Trực quan hóa trong Thuyết trình](#)
  - [9.6 Bài tập Thực hành](#)
  - [9.7 Hướng dẫn Tích hợp vào Các Dự án Lớn](#)

- [9.8 Kết luận](#)
- [9.9 Tổng kết chương 9](#)
- [Chương 10: Tài nguyên và Tài liệu tham khảo](#)
  - [10.1 Sách về Dữ liệu hóa học trực quan](#)
  - [10.2 Khóa học trực tuyến](#)
  - [10.3 Trang web và Blog](#)
  - [10.4 Công cụ và Thư viện](#)
  - [10.5 Các công cụ hỗ trợ khác](#)
  - [10.6 Tổng Kết Chương 10](#)

# Chương 1: Giới thiệu

## Contents

- Giới thiệu
- 1.1 Mục Tiêu của Cuốn Sách
- 1.2 Lợi ích của trực quan hóa dữ liệu
- 1.3 Vì sao chọn python cho trực quan hóa dữ liệu
- 1.4 Tổng quan về các chương
- 1.5 Cách sử dụng cuốn sách
- 1.6 Các lĩnh vực ứng dụng của trực quan hóa dữ liệu
- 1.7 Thực hành và ứng dụng
- 1.8 So sánh các công cụ trực quan hóa
- 1.9 Tương lai của trực quan hóa dữ liệu
- 1.10 Tổng kết chương 1

## Giới thiệu

**Trực quan hóa dữ liệu (Data Visualization)** là một trong những kỹ năng quan trọng nhất trong phân tích dữ liệu và khoa học dữ liệu. Việc trình bày dữ liệu qua các biểu đồ, đồ thị không chỉ giúp dễ dàng nhận diện xu hướng và mô hình mà còn tạo điều kiện truyền tải thông tin hiệu quả đến các đối tượng khác nhau.

## 1.1 Mục Tiêu của Cuốn Sách

Cuốn sách này được thiết kế nhằm:

- **Cung cấp nền tảng vững chắc** về lập trình Python liên quan đến trực quan hóa dữ liệu.
- **Hướng dẫn từng bước** sử dụng các thư viện phổ biến nhất như Matplotlib, Seaborn và Plotly.
- **Thực hành trực tiếp** với các ví dụ minh họa, bài tập thực tế, và ứng dụng vào các dự án cụ thể.
- **Tạo động lực** cho người học tiếp tục khám phá và nâng cao kỹ năng trực quan hóa dữ liệu. Cuốn sách phù hợp với cả **người mới bắt đầu** và **người đã có kiến thức cơ bản** về Python và phân tích dữ liệu.

## 1.2 Lợi ích của trực quan hóa dữ liệu

- **Hiểu rõ dữ liệu:** Biểu đồ và đồ thị giúp xác định các xu hướng, mẫu, và bất thường trong dữ liệu dễ dàng hơn.
- **Ra quyết định chính xác:** Thông tin trình bày rõ ràng sẽ hỗ trợ việc ra quyết định dựa trên dữ liệu thay vì cảm tính.
- **Giao tiếp hiệu quả:** Trực quan hóa là công cụ mạnh mẽ để truyền tải ý tưởng hoặc kết quả nghiên cứu đến đồng nghiệp, lãnh đạo hoặc công chúng.

## 1.3 Vì sao chọn python cho trực quan hóa dữ liệu

Python là một trong những ngôn ngữ lập trình phổ biến nhất hiện nay trong lĩnh vực phân tích dữ liệu. Các lý do chính để chọn Python:

- **Hệ sinh thái phong phú:** Python cung cấp nhiều thư viện mạnh mẽ như Matplotlib, Seaborn, Plotly, Bokeh, Altair, v.v.
- **Tính dễ học:** Python có cú pháp đơn giản, dễ tiếp cận, phù hợp cho cả người mới bắt đầu.
- **Tính linh hoạt:** Ngoài trực quan hóa dữ liệu, Python còn hỗ trợ thu thập, xử lý, và phân tích dữ liệu từ nhiều nguồn.
- **Cộng đồng lớn:** Python có cộng đồng lập trình viên rộng lớn, dễ dàng tìm kiếm tài liệu, hướng dẫn, và hỗ trợ.

## 1.4 Tổng quan về các chương

Cuốn sách được chia thành 10 chương như sau:

- **Giới thiệu:** Tổng quan về trực quan hóa dữ liệu và Python.
- **Cài đặt Môi trường:** Hướng dẫn thiết lập môi trường làm việc với Python và các thư viện cần thiết.
- **Giới thiệu về Dữ liệu:** Các khái niệm cơ bản về dữ liệu và cách xử lý dữ liệu trong Python.
- **Thống kê mô tả:** Trực quan hóa dữ liệu thống kê cơ bản.
- **Matplotlib:** Thư viện nền tảng cho trực quan hóa dữ liệu.
- **Seaborn:** Trực quan hóa nâng cao với Seaborn.
- **Plotly:** Tạo biểu đồ tương tác với Plotly.
- **Kỹ thuật nâng cao:** Các phương pháp trực quan hóa dữ liệu phức tạp hơn.
- **Tích hợp vào bài tập:** Áp dụng kỹ năng trực quan hóa trong các bài tập thực tế.
- **Tài nguyên và Tài liệu tham khảo:** Tài liệu và nguồn học thêm.

## 1.5 Cách sử dụng cuốn sách

- **Học tuần tự:** Đọc từng chương từ đầu đến cuối để hiểu toàn bộ nội dung.
- **Thực hành ngay:** Làm theo các ví dụ trong sách để thực hành trực tiếp.
- **Quay lại khi cần:** Sử dụng cuốn sách như tài liệu tham khảo bất cứ lúc nào trong quá trình học hoặc làm việc.

## 1.6 Các lĩnh vực ứng dụng của trực quan hóa dữ liệu

- **Kinh doanh:** Trực quan hóa báo cáo bán hàng, phân tích thị trường.
- **Khoa học và nghiên cứu:** Phân tích dữ liệu khoa học hoặc kết quả nghiên cứu.
- **Chính phủ và xã hội:** Theo dõi chỉ số kinh tế, biểu đồ dân số.
- **Truyền thông:** Trình bày dữ liệu dưới dạng đồ họa để tiếp cận công chúng.

## 1.7 Thực hành và Ứng dụng

- **Ứng dụng thực tế:** Cung cấp các ví dụ thực tế về cách trực quan hóa dữ liệu trong các lĩnh vực khác nhau như y tế, tài chính, và giáo dục.
- **Cảnh báo về các lỗi thường gặp:** Thảo luận về những lỗi phổ biến trong trực quan hóa dữ liệu và cách tránh chúng, như việc sử dụng màu sắc không phù hợp hoặc biểu đồ gây hiểu lầm.

## 1.8 So sánh các công cụ trực quan hóa

- **Matplotlib:** Thư viện cơ bản, mạnh mẽ cho việc tạo biểu đồ tĩnh.
- **Seaborn:** Xây dựng trên Matplotlib, cung cấp các biểu đồ đẹp hơn và dễ sử dụng hơn.
- **Plotly:** Tạo biểu đồ tương tác, phù hợp cho các ứng dụng web.

## 1.9 Tương lai của trực quan hóa dữ liệu

- **Xu hướng mới:** Sự phát triển của AI trong việc tạo ra các biểu đồ tự động và sự gia tăng của thực tế ảo trong trực quan hóa dữ liệu.

## 1.10 Tổng kết chương 1

- **Nội dung:** Trong chương 1 bạn sẽ được hướng dẫn chi tiết và rõ ràng, bạn sẽ sớm nắm vững các kỹ thuật trực quan hóa dữ liệu, sẵn sàng áp dụng chúng vào công việc thực tế hoặc nghiên cứu học thuật.

# Chương 2: Cài đặt Môi Trường

## Contents

- Cài đặt Môi Trường
- 2.1 Lựa chọn môi trường phát triển
- 2.2 Cài đặt Python
- 2.3 Cài đặt Jupyter Notebook
- 2.4 Cài đặt các thư viện cần thiết
- 2.5 Kiểm tra môi trường
- 2.6 Một số lưu ý
- 2.7 Tổng kết chương 2

## Cài đặt Môi Trường

Trước khi bắt đầu làm việc với trực quan hóa dữ liệu, chúng ta cần thiết lập môi trường làm việc. Điều này bao gồm cài đặt Python, các công cụ hỗ trợ, và các thư viện cần thiết. Chương này sẽ hướng dẫn bạn từng bước.

## 2.1 Lựa chọn môi trường phát triển

Python là ngôn ngữ phổ biến với nhiều môi trường phát triển khác nhau. Dưới đây là các công cụ thường được sử dụng:

### Jupyter Notebook:

- Là công cụ lý tưởng cho trực quan hóa dữ liệu.
- Hỗ trợ kết hợp mã nguồn Python, biểu đồ, và ghi chú vào cùng một tài liệu.

- **Tính năng nổi bật:** Hỗ trợ trực quan hóa dữ liệu và chia sẻ tài liệu dễ dàng.
- **Cách sử dụng:** Hướng dẫn cách tạo và lưu trữ notebook.

### Google Colab:

- Là phiên bản dựa trên web của Jupyter Notebook, miễn phí và không cần cài đặt.
- Thích hợp cho những người không muốn cài đặt Python trên máy tính.
- **Tính năng nổi bật:** Tích hợp Google Drive để lưu trữ và chia sẻ.
- **Hướng dẫn:** Cách sử dụng GPU miễn phí cho các tác vụ tính toán nặng.

### Integrated Development Environment (IDE):

- VS Code hoặc PyCharm: Lý tưởng để làm việc với các dự án lớn.
- Hỗ trợ quản lý mã nguồn và tích hợp nhiều công cụ.
- **VS Code:**
  - **Tính năng mở rộng:** Cách cài đặt các tiện ích mở rộng cho Python.
  - **Tích hợp Git:** Hướng dẫn sử dụng Git trong VS Code.
- **PyCharm:**
  - **Tính năng nổi bật:** Hỗ trợ kiểm tra mã và gợi ý mã thông minh.

## 2.2 Cài đặt Python

### Tải Python

- Truy cập [Trang Chủ Python](#) và tải phiên bản mới nhất.
- **Lựa chọn phiên bản:** Hướng dẫn chọn phiên bản Python phù hợp (3.x).

### Cài đặt Python

- Trong quá trình cài đặt, đảm bảo chọn tùy chọn “**Add Python to PATH**”.
- Hoàn tất cài đặt và kiểm tra phiên bản bằng lệnh:

```
python --version
```

- **Cách kiểm tra cài đặt:** Hướng dẫn kiểm tra cài đặt Python và pip.

## 2.3 Cài đặt Jupyter Notebook

### Sử dụng pip

- Cài đặt Jupyter Notebook thông qua Python:

```
pip install notebook
```

- Sau đó khởi chạy Notebook bằng lệnh:

```
jupyter notebook
```

- **Cách khởi động Jupyter:** Hướng dẫn mở Jupyter Notebook từ terminal.

### Sử dụng Anaconda

- [Tải Anaconda](#) và cài đặt.
- Mở Anaconda Navigator và khởi chạy Jupyter Notebook.
- **Quản lý môi trường:** Hướng dẫn tạo và quản lý môi trường ảo trong Anaconda.

## 2.4 Cài đặt các thư viện cần thiết

Các thư viện Python phổ biến để trực quan hóa dữ liệu:

- **Matplotlib:** Nền tảng cho các thư viện khác.
  - **Ví dụ sử dụng:** Cung cấp một ví dụ đơn giản về cách tạo biểu đồ.
- **Seaborn:** Trực quan hóa nâng cao, dễ sử dụng.
  - **Tính năng nổi bật:** Hướng dẫn sử dụng Seaborn để tạo biểu đồ thống kê.
- **Plotly:** Tạo biểu đồ tương tác.
  - **Tính năng tương tác:** Hướng dẫn tạo biểu đồ tương tác với Plotly.

\*\* Cách cài đặt:\*\*

- Sử dụng lệnh sau trong terminal hoặc Jupyter Notebook:

```
pip install matplotlib seaborn plotly
```

- Hoặc nếu bạn sử dụng Anaconda:

```
conda install matplotlib seaborn plotly
```

## 2.5 Kiểm tra môi trường

- Trước khi bắt đầu, hãy kiểm tra xem mọi thứ đã được cài đặt chính xác hay chưa. Chạy đoạn mã dưới đây trong Jupyter Notebook:

```
import sys
import matplotlib
import seaborn
import plotly

print("Python version:", sys.version)
print("Matplotlib version:", matplotlib.__version__)
print("Seaborn version:", seaborn.__version__)
print("Plotly version:", plotly.__version__)
```

- Nếu tất cả các phiên bản được hiển thị mà không có lỗi, bạn đã sẵn sàng!
- **Giải thích kết quả:** Hướng dẫn cách đọc và hiểu kết quả kiểm tra.

## 2.6 Một số lưu ý

- **Quản lý thư viện:** Sử dụng virtualenv hoặc conda để tạo môi trường làm việc độc lập, tránh xung đột thư viện.
- **Cập nhật thư viện:** Thường xuyên cập nhật các thư viện để sử dụng các tính năng mới nhất:

```
pip install --upgrade matplotlib seaborn plotly
```

- **Xử lý lỗi:** Nếu gặp vấn đề cài đặt, kiểm tra lỗi và tìm giải pháp trên [Stack Overflow](#) hoặc trang GitHub của thư viện.

## 2.7 Tổng kết chương 2

**Nội dung:** Trong chương 2 này giúp các bạn hoàn tất thiết lập môi trường! Hãy chuyển sang tiếp chương 3 để bắt đầu tìm hiểu về dữ liệu.

# Chương 3: Giới thiệu về Dữ liệu

## Contents

- Giới thiệu về Dữ liệu
- 3.1 Dữ liệu là gì?
- 3.2 Các nguồn dữ liệu phổ biến
- 3.3 Cách làm việc với dữ liệu trong Python
- 3.4 Làm sạch dữ liệu
- 3.5 Tổng hợp dữ liệu
- 3.6 Xuất dữ liệu
- 3.7 Các loại dữ liệu nâng cao
- 3.8 Các công cụ và phần mềm hỗ trợ làm việc với dữ liệu
- 3.9 Các phương pháp phân tích dữ liệu
- 3.10 Thực hành nâng cao
- 3.11 Các thách thức trong làm việc với dữ liệu
- 3.12 Tương lai của dữ liệu
- 3.13 Thực hành
- 3.14 Tổng kết chương 3

## Giới thiệu về Dữ liệu

Dữ liệu là yếu tố trung tâm trong mọi dự án phân tích và trực quan hóa. Chương này sẽ giúp bạn hiểu các khái niệm cơ bản về dữ liệu, cách làm việc với dữ liệu trong Python và chuẩn bị dữ liệu cho trực quan hóa.

## 3.1 Dữ liệu là gì?

Dữ liệu là tập hợp các giá trị được thu thập để nghiên cứu hoặc phân tích. Dữ liệu có thể tồn tại dưới nhiều dạng khác nhau, như:

**Dữ liệu số (Numerical):** Số liệu có thể đo lường (ví dụ: chiều cao, cân nặng).

**Dữ liệu phân loại (Categorical):** Dữ liệu thể hiện các nhóm hoặc loại (ví dụ: giới tính, màu sắc).

**Dữ liệu theo thời gian (Time-series):** Dữ liệu được thu thập theo thời gian (ví dụ: giá cổ phiếu hàng ngày).

## 3.2 Các nguồn dữ liệu phổ biến

Dữ liệu có thể được lấy từ nhiều nguồn khác nhau, bao gồm:

**Tệp tin:** CSV, Excel, JSON, v.v. **Cơ sở dữ liệu:** MySQL, PostgreSQL, MongoDB, v.v. **API:** Các dịch vụ cung cấp dữ liệu qua giao thức HTTP (ví dụ: Google Maps API, Twitter API). **Web scraping:** Thu thập dữ liệu từ các trang web.

## 3.3 Cách làm việc với dữ liệu trong Python

Python cung cấp nhiều thư viện mạnh mẽ để xử lý và phân tích dữ liệu. Trong chương này, chúng ta sẽ tập trung vào hai thư viện chính: **Pandas** và **NumPy**.

**3.3.1 Thư viện Pandas** Pandas là thư viện phổ biến nhất để thao tác với dữ liệu dạng bảng (DataFrame).

**Cài đặt Pandas:**

```
pip install pandas
```

**Đọc dữ liệu từ tệp CSV:**

```
import pandas as pd

# Đọc dữ liệu từ tệp CSV
data = pd.read_csv('example.csv')

# Hiển thị 5 dòng đầu tiên
print(data.head())
```

**Giải thích:**

- pd.read\_csv('example.csv'): Đọc dữ liệu từ tệp example.csv và lưu vào biến data dưới dạng DataFrame.
- data.head(): Hiển thị 5 dòng đầu tiên của DataFrame, giúp bạn xem sơ qua dữ liệu.

**Một số thao tác cơ bản với Pandas:**

```
# Hiển thị thông tin tổng quan về dữ liệu
print(data.info())

# Hiển thị thống kê cơ bản của dữ liệu
print(data.describe())

# Lọc dữ liệu theo điều kiện
filtered_data = data[data['column_name'] > 50]
```

**Giải thích:**

- data.info(): Hiển thị thông tin tổng quan về DataFrame như số lượng dòng, cột và kiểu dữ liệu.
- data.describe(): Tính toán các thống kê cơ bản (mean, std, min, max) cho dữ liệu số trong DataFrame.
- data[data['column\_name'] > 50]: Lọc các dòng có giá trị trong cột column\_name lớn hơn 50.

**3.3.2 Thư viện NumPy** NumPy hỗ trợ làm việc với các mảng số học hiệu quả.**Cài đặt NumPy:**

```
pip install numpy
```

**Tạo và thao tác với mảng:**

```
import numpy as np

# Tạo mảng
array = np.array([1, 2, 3, 4, 5])

# Tính toán cơ bản
print("Mean:", np.mean(array))
print("Standard Deviation:", np.std(array))
```

### Giải thích:

- np.array([1, 2, 3, 4, 5]): Tạo một mảng NumPy từ danh sách Python.
- np.mean(array): Tính giá trị trung bình của mảng.
- np.std(array): Tính độ lệch chuẩn của mảng.

## 3.4 Làm sạch dữ liệu

Dữ liệu thực tế thường không hoàn hảo và cần được làm sạch trước khi trực quan hóa.

### Các bước làm sạch dữ liệu:

#### 1. Xử lý giá trị thiếu:

- Xóa các dòng hoặc cột có giá trị thiếu:

```
data = data.dropna()
```

- Điền giá trị thay thế:

```
data['column_name'] = data['column_name'].fillna(0)
```

#### 2. Xử lý dữ liệu không hợp lệ:

- Loại bỏ các giá trị không hợp lệ hoặc ngoại lệ (outliers).

#### 3. Chuyển đổi kiểu dữ liệu:

- Chuyển đổi cột sang kiểu số hoặc chuỗi phù hợp:

```
data['column_name'] = data['column_name'].astype(float)
```

#### 4. Chuẩn hóa dữ liệu:

- Đổi tên cột để dễ sử dụng:

```
data.rename(columns={'OldName': 'NewName'}, inplace=True)
```

## 3.5 Tổng hợp dữ liệu

Việc tổng hợp giúp rút ra thông tin ý nghĩa từ dữ liệu. Pandas hỗ trợ các phương pháp sau:

- Nhóm dữ liệu:**

```
grouped = data.groupby('category_column').mean()
print(grouped)
```

#### Giải thích:

- data.groupby('category\_column'): Nhóm dữ liệu theo cột category\_column.
- .mean(): Tính giá trị trung bình của các nhóm.

#### Tạo Pivot Table:

```
pivot_table = data.pivot_table(values='value_column', index='category_column', columns
print(pivot_table)
```

#### Giải thích:

- pivot\_table(): Tạo bảng tổng hợp dữ liệu, sử dụng category\_column làm chỉ mục và other\_column làm các cột.
- aggfunc='sum': Tính tổng giá trị cho mỗi nhóm.

## 3.6 Xuất dữ liệu

Sau khi làm việc với dữ liệu, bạn có thể xuất kết quả ra các định dạng khác nhau:

```
# Xuất dữ liệu ra CSV  
data.to_csv('output.csv', index=False)  
  
# Xuất dữ liệu ra Excel  
data.to_excel('output.xlsx', index=False)
```

**Giải thích:**

- `data.to_csv('output.csv')`: Lưu DataFrame vào tệp CSV.
- `data.to_excel('output.xlsx')`: Lưu DataFrame vào tệp Excel.

## 3.7 Các loại dữ liệu nâng cao

- **Dữ liệu phi cấu trúc (Unstructured Data)**: Giới thiệu về dữ liệu không có cấu trúc rõ ràng như văn bản, hình ảnh, video.
- **Dữ liệu bán cấu trúc (Semi-structured Data)**: Ví dụ về dữ liệu như XML, JSON mà có cấu trúc nhưng không hoàn toàn theo định dạng bảng.

## 3.8 Các công cụ và phần mềm hỗ trợ làm việc với dữ liệu

- Giới thiệu về các công cụ như Jupyter Notebook, Google Colab, và các phần mềm như Tableau, Power BI cho việc phân tích và trực quan hóa dữ liệu.

## 3.9 Các phương pháp phân tích dữ liệu

- **Phân tích mô tả (Descriptive Analysis)**: Cách sử dụng thống kê mô tả để tóm tắt dữ liệu.
- **Phân tích dự đoán (Predictive Analysis)**: Giới thiệu về các mô hình học máy cơ bản để dự đoán xu hướng tương lai từ dữ liệu hiện tại.

## 3.10 Thực hành nâng cao

- Cung cấp một số bài tập thực hành nâng cao hơn, chẳng hạn như:
  - Sử dụng Pandas để thực hiện phân tích dữ liệu từ một tệp CSV lớn.
  - Tạo biểu đồ trực quan hóa dữ liệu bằng Matplotlib hoặc Seaborn từ dữ liệu đã xử lý.

## 3.11 Các thách thức trong làm việc với dữ liệu

- Thảo luận về các vấn đề thường gặp như dữ liệu không đầy đủ, dữ liệu không chính xác, và cách giải quyết chúng.

## 3.12 Tương lai của dữ liệu

- Một cái nhìn tổng quan về xu hướng tương lai trong lĩnh vực dữ liệu, như Big Data, Machine Learning, và AI.

## 3.13 Thực hành

Hãy thử các bài tập sau để làm quen với dữ liệu:

1. Đọc tệp data.csv và hiển thị 5 dòng đầu tiên.
2. Kiểm tra xem tệp có giá trị thiếu không và xử lý chúng.
3. Tạo một Pivot Table để tính trung bình một cột theo từng nhóm.

## 3.14 Tổng kết chương 3

**Nội dung:** Trong chương 3 này đã giới thiệu các khái niệm cơ bản và công cụ làm việc với dữ liệu trong Python. Trong chương 4, chúng ta sẽ tìm hiểu về thống kê mô tả và cách trực quan hóa các đặc điểm cơ bản của dữ liệu.

# Chương 4: Thống kê mô tả

## Contents

- Thống kê mô tả
- 4.1 Thống kê mô tả là gì?
- 4.2 Các chỉ số trung tâm
- 4.3 Các chỉ số phân tán
- 4.4 Hình dạng phân phối
- 4.5 Tóm tắt dữ liệu bằng Pandas
- 4.6 Trực quan hóa thống kê mô tả
- 4.7 Thực hành
- 4.8 Thực tiễn ứng dụng của thống kê mô tả
- 4.9 Những sai lầm thường gặp
- 4.10 So sánh với thống kê suy diễn
- 4.11 Công cụ và thư viện
- 4.12 Nghiên cứu điển hình
- 4.13 Tổng kết chương 4

## Thống kê mô tả

Thống kê mô tả là bước đầu tiên trong phân tích dữ liệu, tập trung vào việc tóm tắt và mô tả các đặc điểm cơ bản của dữ liệu. Việc sử dụng thống kê mô tả giúp hiểu rõ hơn về cấu trúc và xu hướng của dữ liệu trước khi tiến hành các phân tích sâu hơn.

# 4.1 Thống kê mô tả là gì?

Thống kê mô tả bao gồm:

- **Các chỉ số trung tâm** (Central Tendency): Trung bình, trung vị, mode.
- **Các chỉ số phân tán** (Dispersion): Độ lệch chuẩn, phương sai, khoảng giá trị.
- **Các chỉ số hình dạng phân phối**: Độ lệch (Skewness) và độ nhọn (Kurtosis).

## 4.2 Các chỉ số trung tâm

**4.2.1 Mean (Trung bình):** Là giá trị trung bình cộng của tất cả các giá trị trong tập dữ liệu.

- Cách tính:

```
import numpy as np

data = [1, 2, 3, 4, 5]
mean = np.mean(data)
print("Mean:", mean)
```

**Giải thích:**

- np.mean(data): Tính giá trị trung bình của tập dữ liệu. Trong ví dụ này, giá trị trung bình là  $(1 + 2 + 3 + 4 + 5) / 5 = 3$ .

**4.2.2 Median (Trung vị):**

Là giá trị nằm ở giữa tập dữ liệu khi sắp xếp theo thứ tự tăng dần.

- Cách tính:

```
median = np.median(data)
print("Median:", median)
```

**Giải thích:**

- np.median(data): Trả về giá trị ở giữa danh sách sau khi sắp xếp. Trong ví dụ trên, giá trị trung vị là 3.

### 4.2.3 Mode (Mode - Giá trị xuất hiện nhiều nhất):

- Cách tính:

```
from scipy import stats

mode = stats.mode(data)
print("Mode:", mode)
```

#### Giải thích:

- stats.mode(data): Tính giá trị xuất hiện nhiều nhất. Nếu tất cả các giá trị có số lần xuất hiện bằng nhau, mode sẽ trả về giá trị đầu tiên.

## 4.3 Các chỉ số phân tán

### 4.3.1 Range (Khoảng giá trị):

Hiệu số giữa giá trị lớn nhất và nhỏ nhất.

- Cách tính:

```
range_value = np.ptp(data)
print("Range:", range_value)
```

#### Giải thích:

- np.ptp(data): Trả về giá trị chênh lệch giữa giá trị lớn nhất và nhỏ nhất trong dữ liệu. Ví dụ:  $(5 - 1) = 4$ .

### 4.3.2 Variance (Phương sai):

Đo lường mức độ phân tán của dữ liệu.

- Cách tính:

```
variance = np.var(data)
print("Variance:", variance)
```

#### Giải thích:

- np.var(data): Tính phương sai của tập dữ liệu, đo lường sự biến thiên của dữ liệu so với giá trị trung bình.

### 4.3.3 Standard Deviation (Độ lệch chuẩn):

Là căn bậc hai của phương sai, đo lường độ phân tán quanh giá trị trung bình.

- Cách tính:

```
std_dev = np.std(data)
print("Standard Deviation:", std_dev)
```

### Giải thích:

- np.std(data): Tính độ lệch chuẩn của tập dữ liệu, cho biết mức độ phân tán

## 4.4 Hình dạng phân phối

### 4.4.1 Skewness (Độ lệch):

Cho biết sự không đối xứng của dữ liệu. Skewness > 0: lệch phải, Skewness < 0: lệch trái.

- Cách tính:

```
skewness = stats.skew(data)
print("Skewness:", skewness)
```

### Giải thích:

- stats.skew(data): Tính độ lệch phân phối của dữ liệu. Nếu giá trị skewness > 0, dữ liệu lệch phải, nếu skewness < 0, dữ liệu lệch trái.

### 4.4.2 Kurtosis (Độ nhọn):

Đo lường “độ nhọn” của phân phối.

- Cách tính:

```
kurtosis = stats.kurtosis(data)
print("Kurtosis:", kurtosis)
```

**Giải thích:**

- stats.kurtosis(data): Tính độ nhọn của phân phối. Giá trị kurtosis cao cho thấy phân phối có đỉnh nhọn hơn bình thường.

## 4.5 Tóm tắt dữ liệu bằng Pandas

Pandas cung cấp một phương thức describe() để nhanh chóng tính toán các chỉ số thống kê mô tả cho tất cả các cột số trong DataFrame.

```
import pandas as pd

# Tạo DataFrame
data = {'Age': [25, 30, 35, 40, 45], 'Salary': [50000, 60000, 75000, 80000, 90000]}
df = pd.DataFrame(data)

# Tóm tắt dữ liệu
print(df.describe())
```

**Giải thích:**

- df.describe(): Tính toán các chỉ số thống kê mô tả như mean, std, min, max và các percentiles (25%, 50%, 75%) cho tất cả các cột số trong DataFrame.

## 4.6 Trực quan hóa thống kê mô tả

Trực quan hóa các chỉ số thống kê mô tả giúp nhận diện xu hướng và sự phân bố dữ liệu.

### 4.6.1 Histogram (Biểu đồ tần suất):

Biểu đồ tần suất cho biết phân phối của một biến.

```
import matplotlib.pyplot as plt

plt.hist(data, bins=5, color='blue', edgecolor='black')
plt.title("Histogram")
plt.xlabel("Values")
plt.ylabel("Frequency")
plt.show()
```

**Giải thích:**

- plt.hist(data): Vẽ biểu đồ tần suất cho dữ liệu, chia thành 5 bins (ngắn). Trục x là giá trị, và trục y là tần suất xuất hiện của giá trị đó.

**4.6.2 Boxplot (Biểu đồ hộp):**

Boxplot giúp bạn nhận diện các giá trị ngoại lai (outliers) và phân bố dữ liệu.

```
plt.boxplot(data)
plt.title("Boxplot")
plt.show()
```

**Giải thích:**

- plt.boxplot(data): Vẽ biểu đồ hộp, giúp trực quan hóa các chỉ số thống kê như trung vị, phần tư (quartiles) và ngoại lai (outliers).

**4.6.3 Violin Plot:** Violin Plot kết hợp cả Histogram và Boxplot, giúp bạn thấy rõ phân phối và các đặc điểm thống kê của dữ liệu.

```
import seaborn as sns

sns.violinplot(data=data)
plt.title("Violin Plot")
plt.show()
```

**Giải thích:**

- sns.violinplot(data): Vẽ Violin Plot, cho bạn thấy cả phân phối và các phần tử cơ bản của dữ liệu.

## 4.7 Thực hành

- **Bài tập 1:** Tạo một tập dữ liệu gồm 10 số ngẫu nhiên. Tính toán và hiển thị: Mean, Median, Mode, Variance, Standard Deviation, Skewness, và Kurtosis.
- **Bài tập 2:** Sử dụng dữ liệu về chiều cao và cân nặng của 50 người, tạo biểu đồ Histogram và Boxplot để phân tích sự phân bố.

## 4.8 Thực tiễn ứng dụng của thống kê mô tả

Thống kê mô tả có nhiều ứng dụng trong thực tế, bao gồm:

- **Phân tích thị trường:** Sử dụng để hiểu rõ hơn về hành vi của người tiêu dùng.
- **Y tế:** Giúp phân tích dữ liệu bệnh nhân và xu hướng sức khỏe.
- **Giáo dục:** Đánh giá kết quả học tập và hiệu suất của học sinh.

## 4.9 Những sai lầm thường gặp

Khi sử dụng thống kê mô tả, có một số sai lầm phổ biến mà người dùng cần lưu ý:

- **Chỉ dựa vào trung bình:** Trung bình có thể bị ảnh hưởng bởi các giá trị ngoại lai.
- **Không xem xét độ phân tán:** Chỉ nhìn vào các chỉ số trung tâm mà không xem xét độ phân tán có thể dẫn đến hiểu lầm về dữ liệu.

## 4.10 So sánh với thống kê suy diễn

Thống kê mô tả chỉ tóm tắt và mô tả dữ liệu hiện có, trong khi thống kê suy diễn sử dụng mẫu dữ liệu để đưa ra kết luận về toàn bộ quần thể.

## 4.11 Công cụ và thư viện

Một số công cụ và thư viện phổ biến để thực hiện thống kê mô tả trong Python bao gồm:

- **NumPy:** Thư viện cơ bản cho các phép toán số học.
- **Pandas:** Cung cấp các phương thức mạnh mẽ để phân tích và tóm tắt dữ liệu.

- **SciPy:** Hỗ trợ các phép toán thống kê nâng cao.

## 4.12 Nghiên cứu điển hình

- **Nghiên cứu về sức khỏe:** Phân tích dữ liệu từ một nghiên cứu y tế để xác định các yếu tố ảnh hưởng đến sức khỏe cộng đồng.
- **Phân tích dữ liệu bán hàng:** Sử dụng thống kê mô tả để hiểu rõ hơn về doanh thu và hành vi mua sắm của khách hàng.

## 4.13 Tổng kết chương 4

**Nội dung:** Trong chương 4 này đã giúp bạn nắm vững các chỉ số thống kê mô tả và cách áp dụng chúng vào dữ liệu thực tế. Trong chương 5 chúng ta sẽ bắt đầu với trực quan hóa dữ liệu bằng Matplotlib.

# Chương 5: Trực quan hóa Dữ liệu với Matplotlib

## Contents

- Trực quan hóa Dữ liệu với Matplotlib
- 5.1 Giới thiệu về Matplotlib
- 5.2 Biểu đồ cơ bản với Matplotlib
- 5.3 Tùy chỉnh biểu đồ
- 5.4 Sử dụng Object-Oriented API
- 5.5 Trực quan hóa nhiều biểu đồ
- 5.6 Lưu biểu đồ
- 5.7 Thực hành
- 5.8 Tổng kết chương 5

## Trực quan hóa Dữ liệu với Matplotlib

Matplotlib là một thư viện mạnh mẽ và phổ biến để trực quan hóa dữ liệu trong Python. Nó cung cấp các công cụ linh hoạt để tạo ra các loại biểu đồ khác nhau như biểu đồ đường, cột, tán xạ, và nhiều loại khác. Trong chương này, bạn sẽ học cách sử dụng Matplotlib để trực quan hóa dữ liệu.

## 5.1 Giới thiệu về Matplotlib

### 5.1.1 Cài đặt Matplotlib

Để cài đặt Matplotlib, sử dụng lệnh:

```
pip install matplotlib
```

### 5.1.2 Nhập Matplotlib và cấu trúc cơ bản

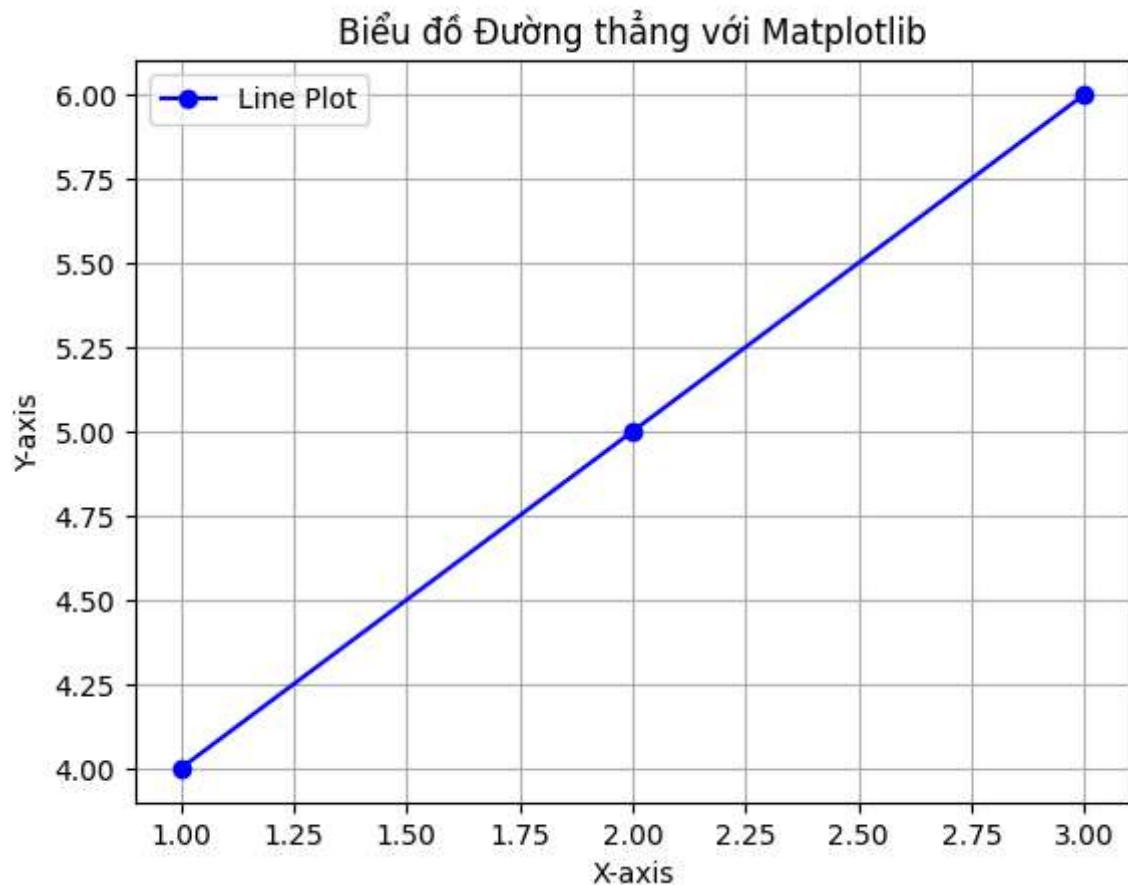
- Matplotlib được tổ chức theo hai cách sử dụng:

**Pyplot API:** Giao diện đơn giản và phổ biến. **Object-Oriented API:** Linh hoạt hơn, cho phép tùy chỉnh sâu. Biểu đồ cơ bản đường thẳng

```
import matplotlib.pyplot as plt

# Dữ liệu
x = [1, 2, 3]
y = [4, 5, 6]

# Vẽ biểu đồ
plt.plot(x, y, marker='o', linestyle='-', color='blue', label='Line Plot')
plt.title('Biểu đồ Đường thẳng với Matplotlib')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid(True)
plt.show()
```

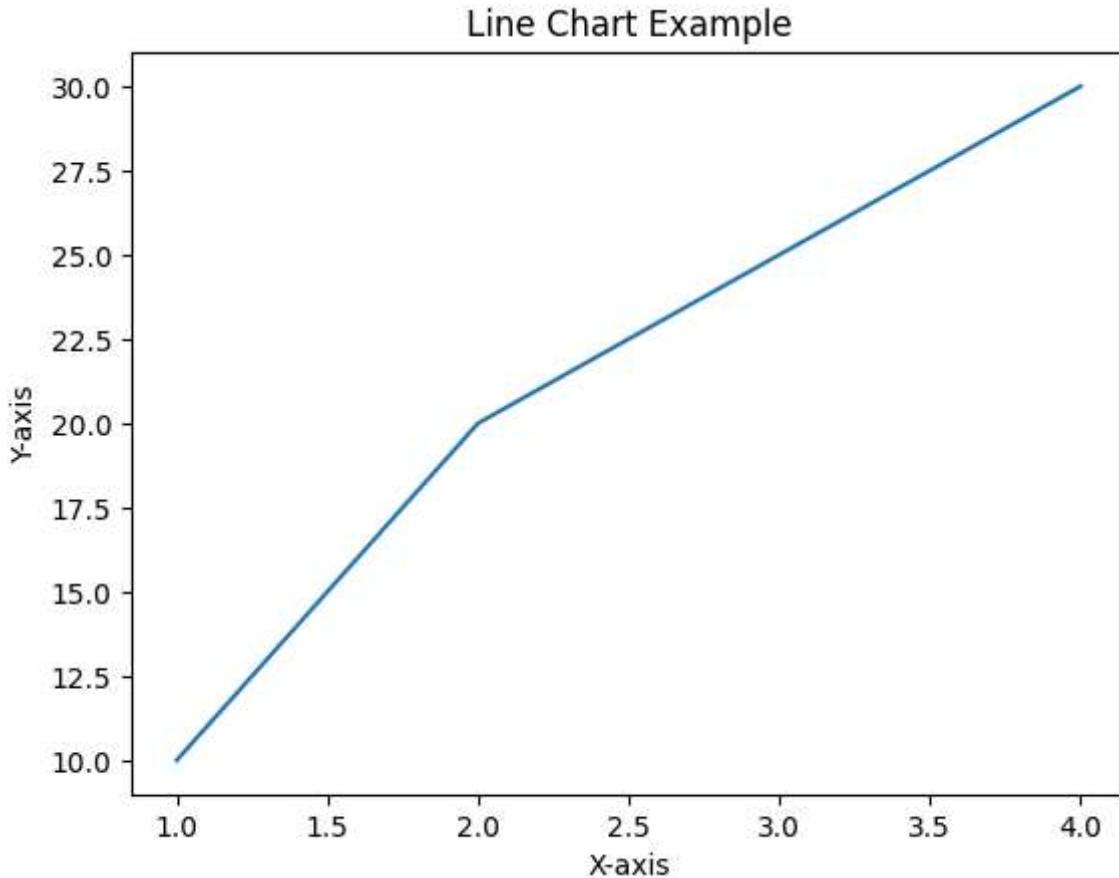


**Chú thích:** Thêm dấu chấm tròn (marker='o') tại các điểm dữ liệu.

Ví dụ cơ bản với Pyplot:

```
import matplotlib.pyplot as plt

# Tạo biểu đồ đường
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
plt.plot(x, y)
plt.title("Line Chart Example")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```



**Chú thích:**

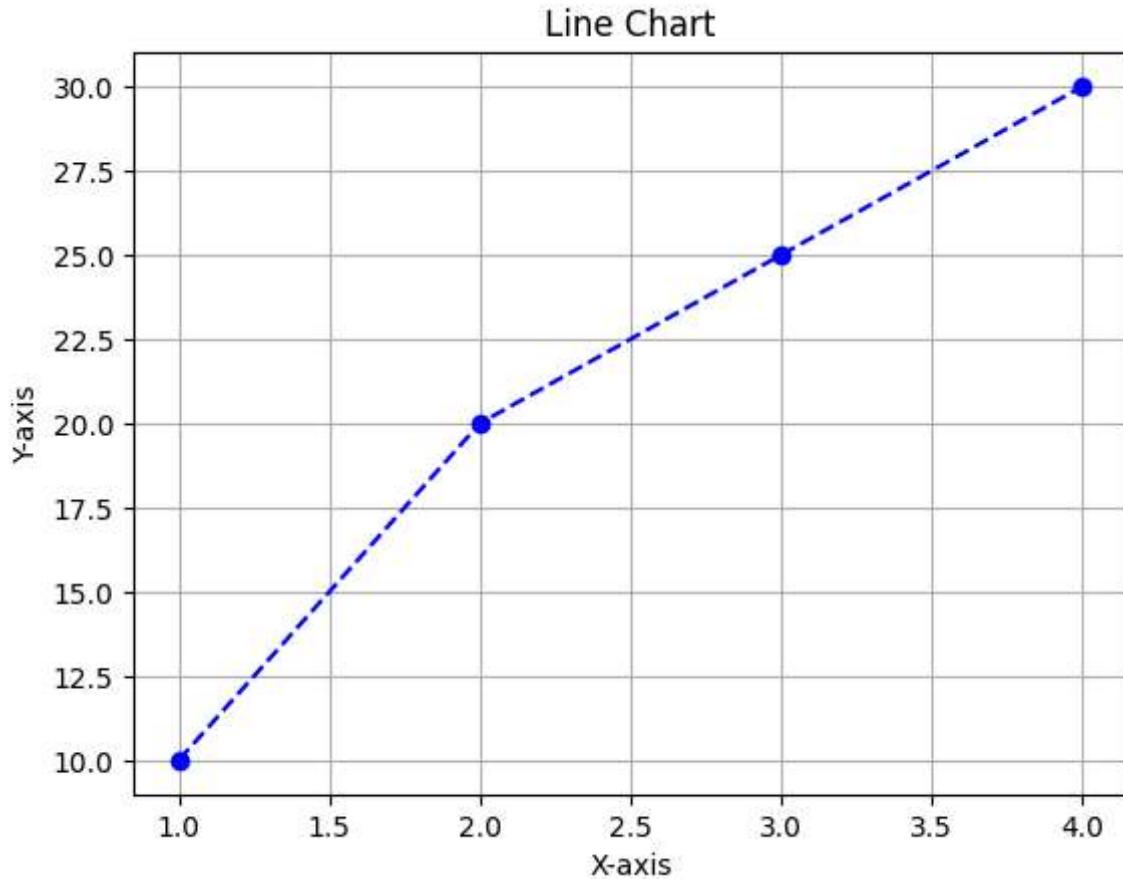
- plt.plot(x, y) vẽ một đường nối giữa các điểm (x, y).
- plt.title và plt.xlabel, plt.ylabel thêm tiêu đề và nhãn trục.

## 5.2 Biểu đồ cơ bản với Matplotlib

### 5.2.1 Biểu đồ đường (Line Chart)

Dùng để hiển thị xu hướng theo thời gian hoặc tuần tự.

```
plt.plot(x, y, color='blue', linestyle='--', marker='o')
plt.title("Line Chart")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.grid(True)
plt.show()
```



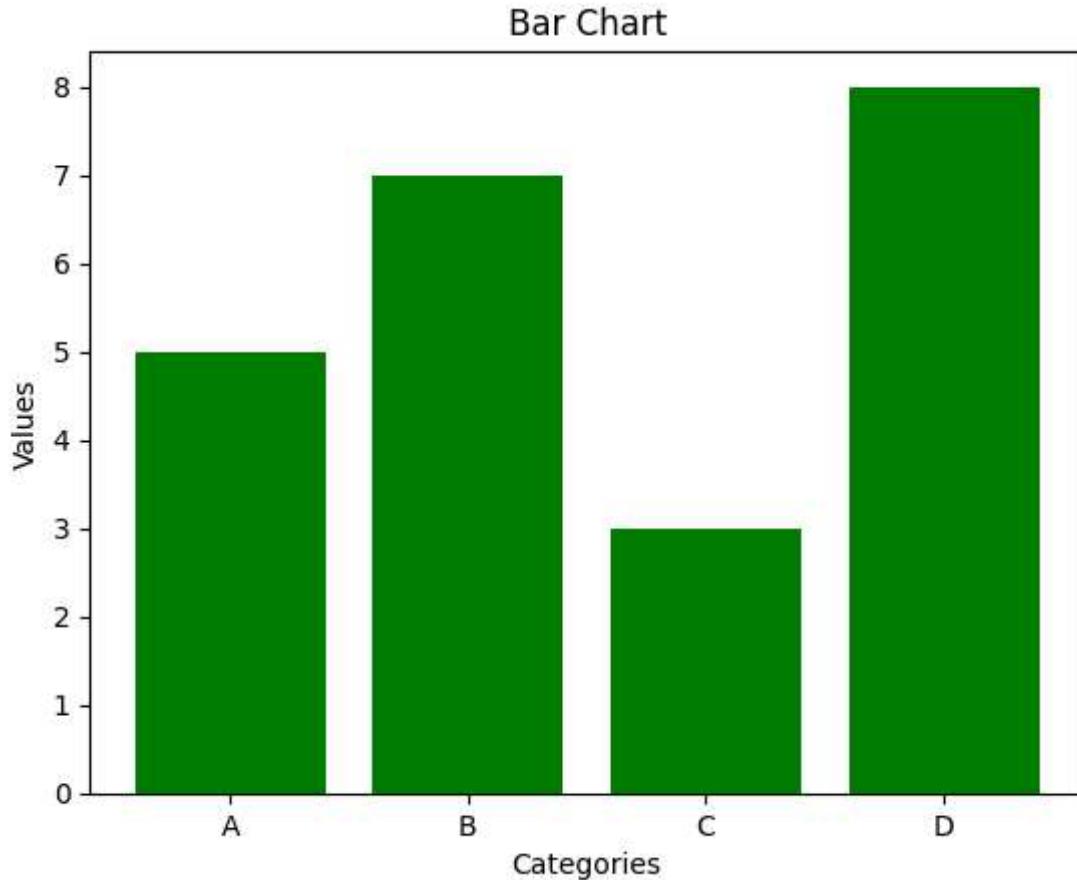
#### Chú thích:

- color='blue': Đặt màu xanh cho đường.
- linestyle='--': Kẻ đường nét đứt.
- marker='o': Thêm các điểm đánh dấu (marker) trên mỗi điểm dữ liệu.
- plt.grid(True): Thêm lưới để biểu đồ dễ đọc hơn.

### 5.2.2 Biểu đồ cột (Bar Chart)

Dùng để so sánh các nhóm.

```
categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]
plt.bar(categories, values, color='green')
plt.title("Bar Chart")
plt.xlabel("Categories")
plt.ylabel("Values")
plt.show()
```

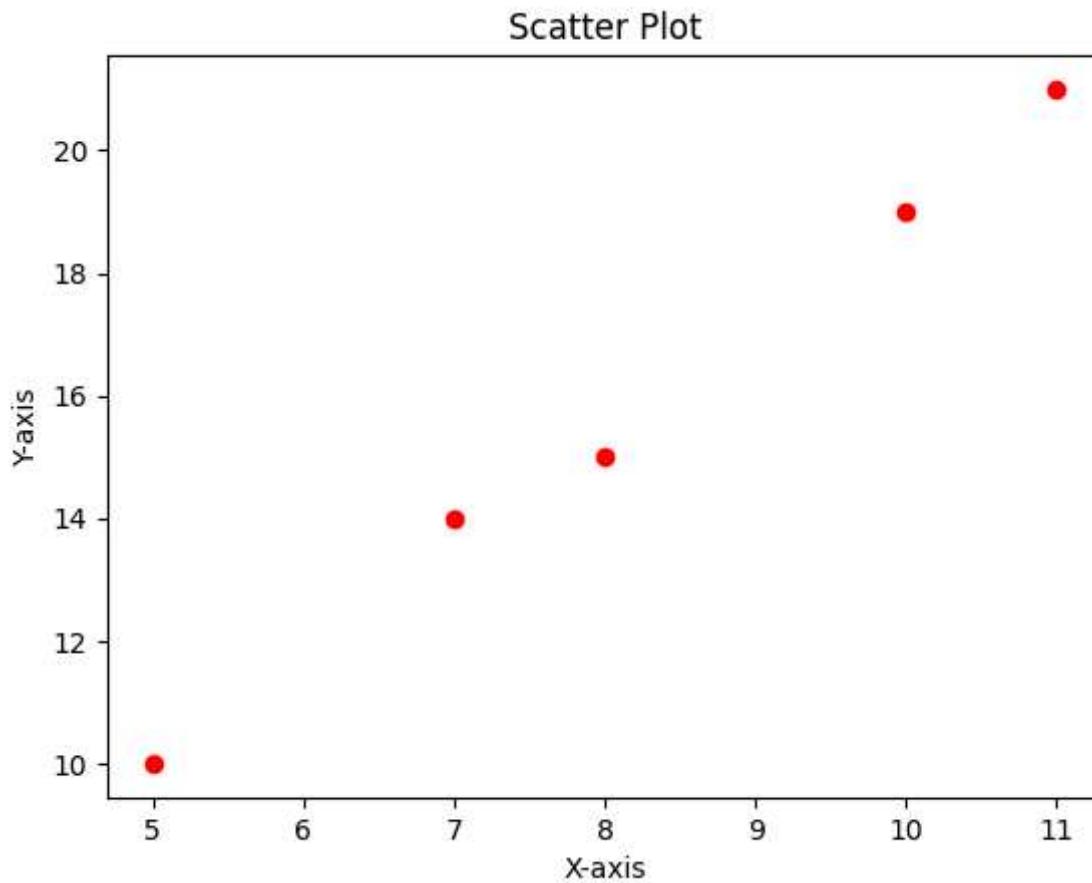


### Chú thích:

- plt.bar vẽ các cột tương ứng với categories và giá trị values.
- color='green' đặt màu xanh lá cây cho các cột.

**5.2.3 Biểu đồ tán xạ (Scatter Plot)** Dùng để hiển thị mối quan hệ giữa hai biến.

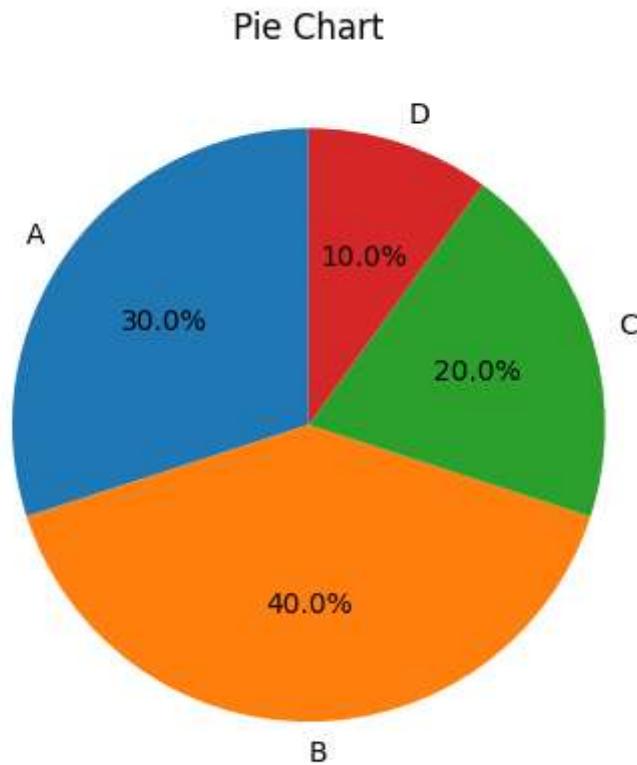
```
x = [5, 7, 8, 10, 11]
y = [10, 14, 15, 19, 21]
plt.scatter(x, y, color='red')
plt.title("Scatter Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

**Chú thích:**

- plt.scatter vẽ các điểm rời rạc tương ứng với dữ liệu.
- Dùng khi bạn muốn phân tích mối quan hệ giữa hai biến (x, y).

**5.2.4 Biểu đồ hình tròn (Pie Chart)** Dùng để biểu diễn tỷ lệ phần trăm của các phần tử.

```
labels = ['A', 'B', 'C', 'D']
sizes = [30, 40, 20, 10]
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title("Pie Chart")
plt.show()
```



#### Chú thích:

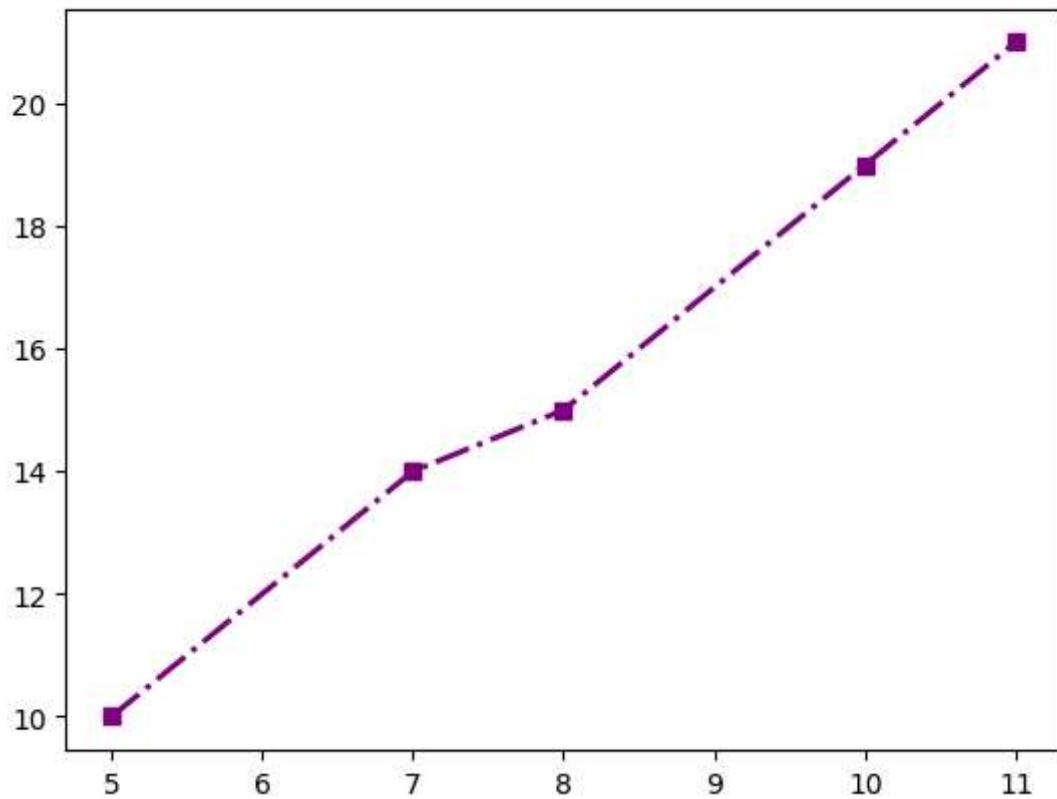
- autopct='%.1f%%': Hiển thị tỷ lệ phần trăm trên từng phần tử.
- startangle=90: Xoay biểu đồ để bắt đầu từ góc 90 độ.

## 5.3 Tùy chỉnh biểu đồ

### 5.3.1 Tùy chỉnh màu sắc và kiểu dáng

```
plt.plot(x, y, color='purple', linestyle='-.', marker='s', linewidth=2)
plt.title("Customized Line Chart")
plt.show()
```

### Customized Line Chart



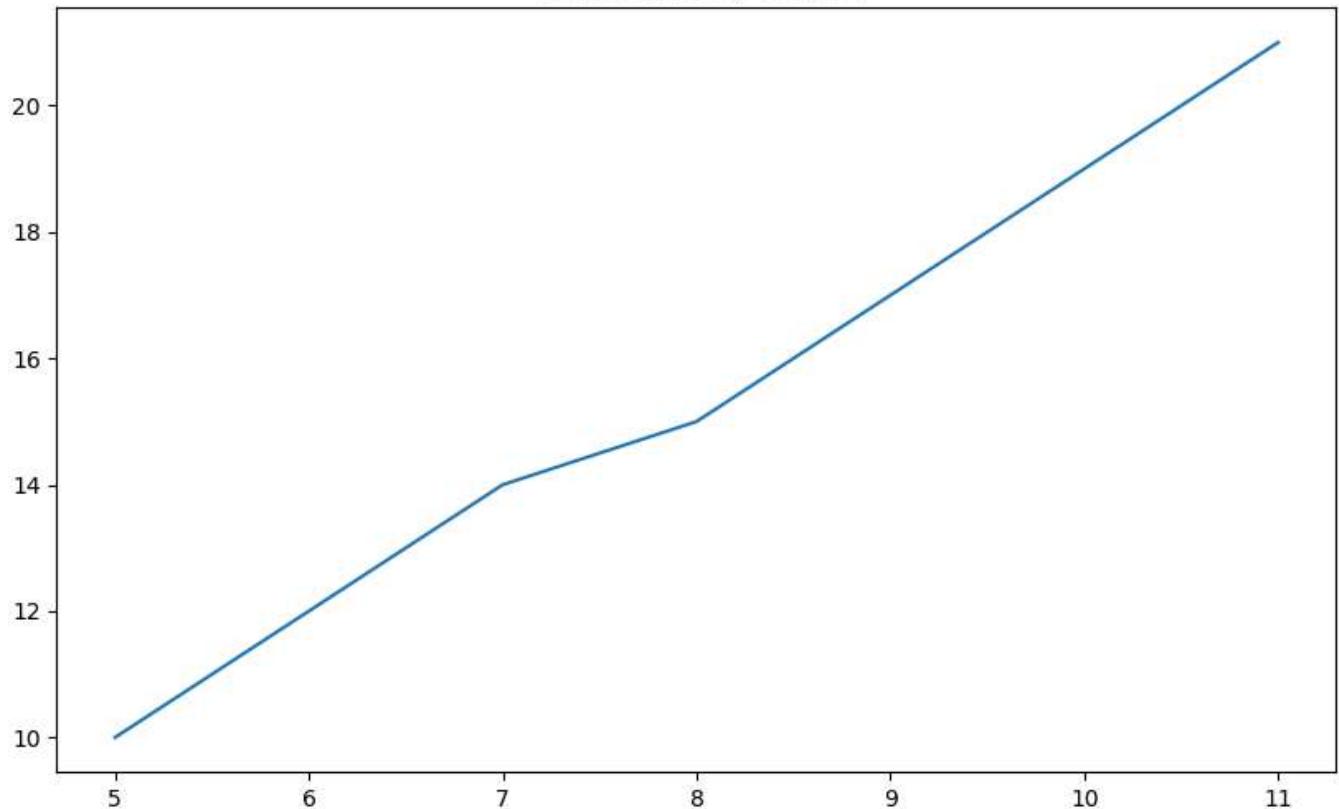
#### Chú thích:

- linewidth=2: Đặt độ dày cho đường.
- color, linestyle, và marker cho phép tùy chỉnh màu sắc, kiểu đường, và kiểu điểm đánh dấu.

#### 5.3.2 Tùy chỉnh kích thước biểu đồ

```
plt.figure(figsize=(10, 6))
plt.plot(x, y)
plt.title("Customized Chart", fontsize=16)
plt.show()
```

## Customized Chart

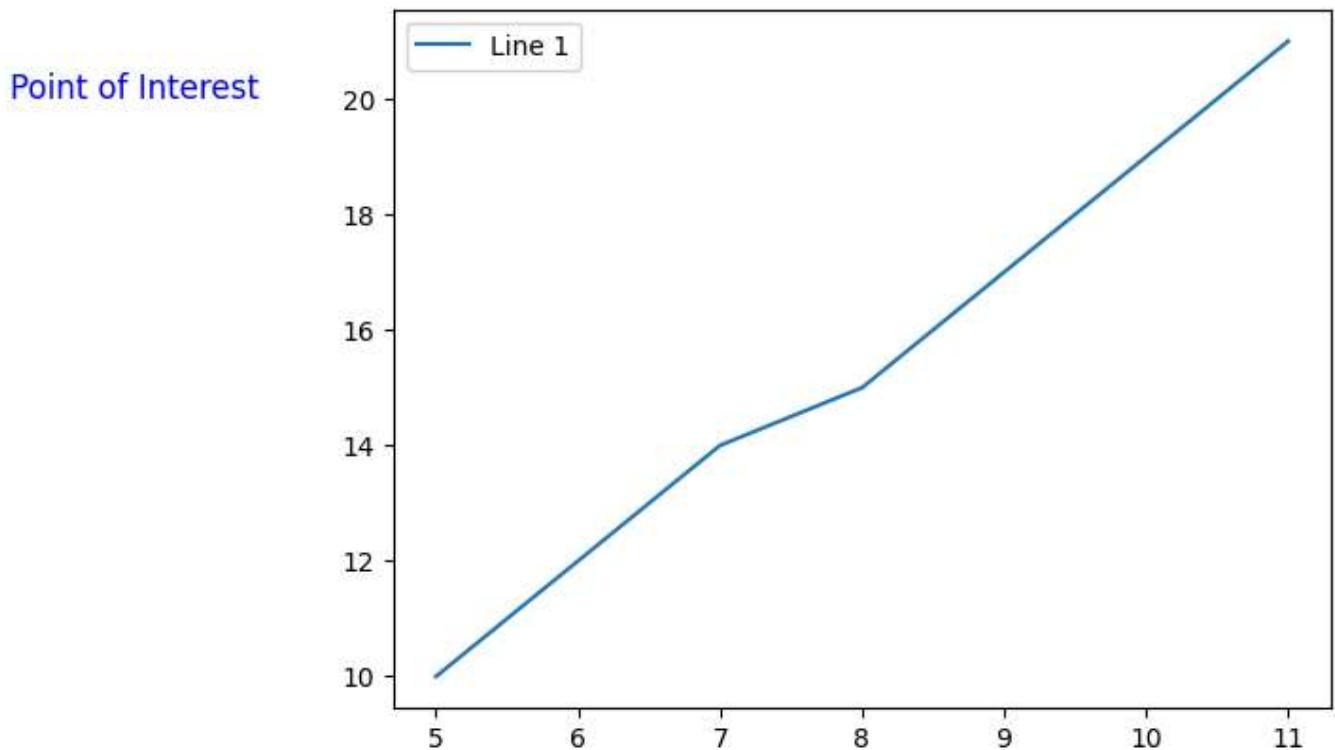


### Chú thích:

- figsize=(10, 6): Đặt kích thước biểu đồ (chiều ngang 10, chiều cao 6).
- fontsize=16: Đặt kích cỡ cho tiêu đề.

### 5.3.3 Thêm chú thích và văn bản

```
plt.plot(x, y)
plt.text(2, 20, 'Point of Interest', fontsize=12, color='blue')
plt.legend(['Line 1'], loc='upper left')
plt.show()
```



### Chú thích:

- plt.text: Thêm văn bản tại vị trí chỉ định.
- plt.legend: Hiển thị chú giải ở góc trên bên trái.

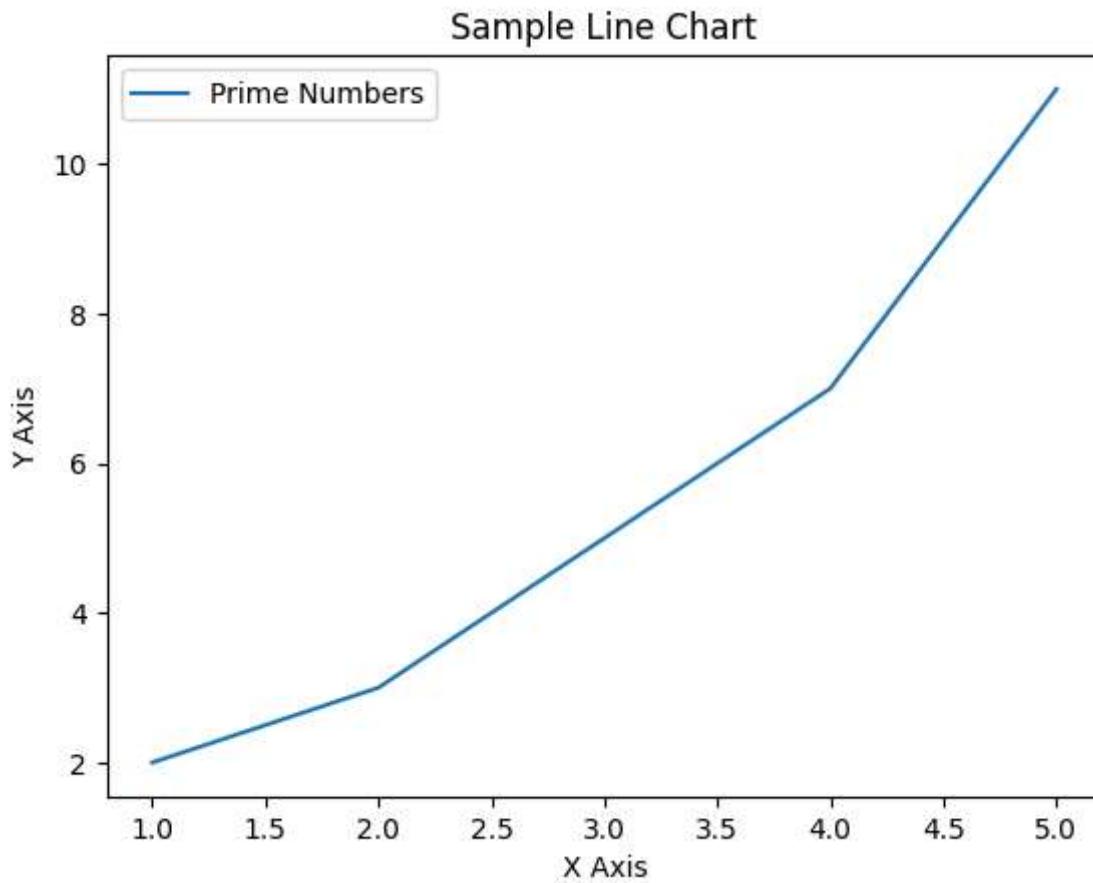
## 5.4 Sử dụng Object-Oriented API

### 5.4.1 Biểu đồ đường với Matplotlib

- Đây là biểu đồ đường API

```
import matplotlib.pyplot as plt

class LineChart:
    def __init__(self, title, x_label, y_label):
        self.fig, self.ax = plt.subplots()
        self.ax.set_title(title)
        self.ax.set_xlabel(x_label)
        self.ax.set_ylabel(y_label)
    def plot(self, x_data, y_data, label):
        self.ax.plot(x_data, y_data, label=label)
        self.ax.legend()
    def show(self):
        plt.show()
if __name__ == "__main__":
    # Sample data
    x = [1, 2, 3, 4, 5]
    y = [2, 3, 5, 7, 11]
    chart = LineChart("Sample Line Chart", "X Axis", "Y Axis")
    chart.plot(x, y, label="Prime Numbers")
    chart.show()
```



### Chú thích:

Lớp LineChart : Lớp này bao gồm chức năng tạo biểu đồ đường.

- `__init__`: Phương pháp : Khởi tạo hình và trục, đặt tiêu đề và nhãn cho các trục.
- `plot`: Lấy dữ liệu x và y cùng với nhãn để vẽ đường thẳng trên các trục.
- `show`: Hiển thị biểu đồ.

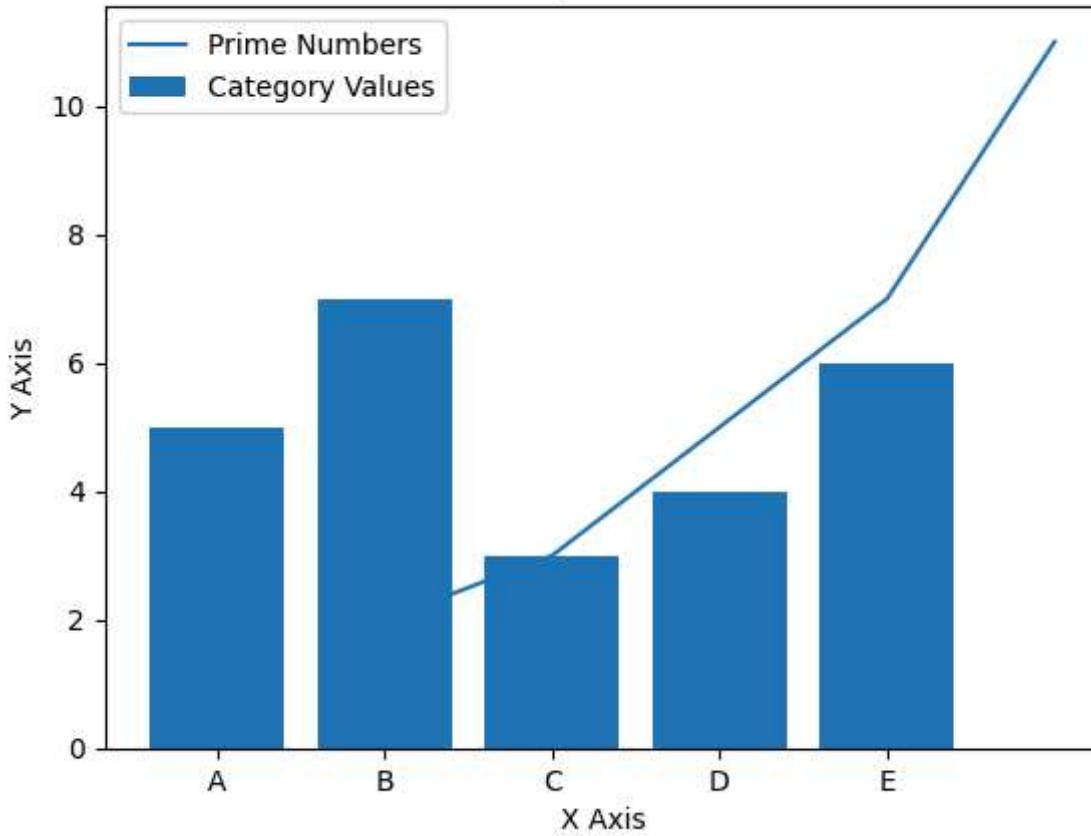
#### 5.4.2 Thêm phương pháp biểu đồ thanh

```
import matplotlib.pyplot as plt

class Chart:
    def __init__(self, title, x_label, y_label):
        self.fig, self.ax = plt.subplots()
        self.ax.set_title(title)
        self.ax.set_xlabel(x_label)
        self.ax.set_ylabel(y_label)
    def plot_line(self, x_data, y_data, label):
        self.ax.plot(x_data, y_data, label=label)
        self.ax.legend()
    def plot_bar(self, x_data, y_data, label):
        self.ax.bar(x_data, y_data, label=label)
        self.ax.legend()

    def show(self):
        plt.show()
if __name__ == "__main__":
    # Sample data for line chart
    x_line = [1, 2, 3, 4, 5]
    y_line = [2, 3, 5, 7, 11]
    x_bar = ['A', 'B', 'C', 'D', 'E']
    y_bar = [5, 7, 3, 4, 6]
    chart = Chart("Sample Chart", "X Axis", "Y Axis")
    chart.plot_line(x_line, y_line, label="Prime Numbers")
    chart.plot_bar(x_bar, y_bar, label="Category Values")
    chart.show()
```

### Sample Chart



**Chú thích:** Lớp biểu đồ : Lớp này hiện hỗ trợ cả biểu đồ đường và biểu đồ thanh.

**plot\_bar** Phương pháp : Phương pháp mới này sử dụng dữ liệu x và y cùng với nhãn để tạo biểu đồ thanh trên các trục. Ví dụ sử dụng :

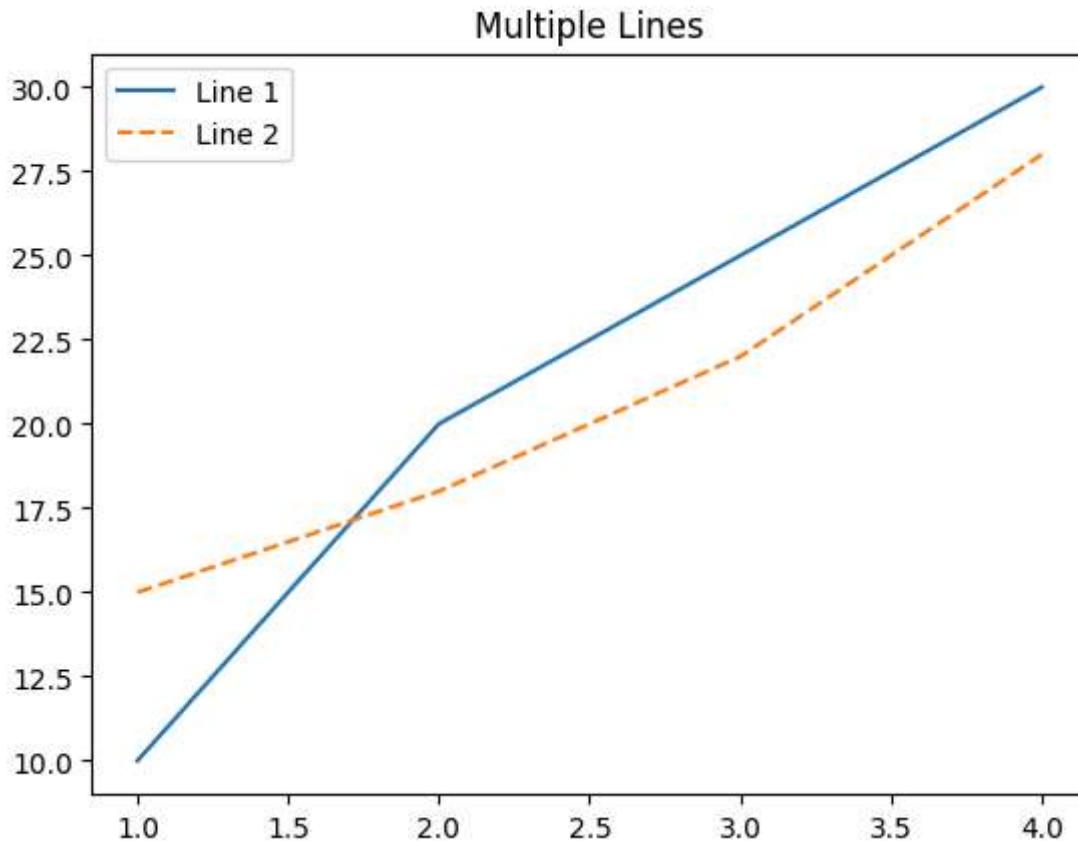
- Dữ liệu mẫu được xác định cho cả biểu đồ đường và biểu đồ thanh.
- Một thể hiện của Chart được tạo ra với tiêu đề và nhãn trục.
- Phương pháp này `plot_line` được gọi để thêm dữ liệu đường vào biểu đồ.
- Phương pháp này `plot_bar` được gọi để thêm dữ liệu thanh vào cùng một biểu đồ.
- Cuối cùng, `show` phương thức này được gọi để hiển thị biểu đồ kết hợp.

## 5.5 Trực quan hóa nhiều biểu đồ

### 5.5.1 Nhiều biểu đồ trong cùng một hình

```
x = [1, 2, 3, 4] # Chỉ giữ lại 4 giá trị
y1 = [10, 20, 25, 30]
y2 = [15, 18, 22, 28]

plt.plot(x, y1, label='Line 1')
plt.plot(x, y2, label='Line 2', linestyle='--')
plt.title("Multiple Lines")
plt.legend()
plt.show()
```



### Chú thích:

- Vẽ nhiều đường trên cùng một biểu đồ, mỗi đường có chú thích riêng.

### 5.5.2 Tạo bố cục nhiều biểu đồ (Subplots)

```

x = [1, 2, 3, 4]
y = [10, 20, 25, 30]

fig, axs = plt.subplots(2, 2, figsize=(10, 8))
axs[0, 0].plot(x, y, color='blue') # Biểu đồ đường
axs[0, 0].set_title("Plot 1")

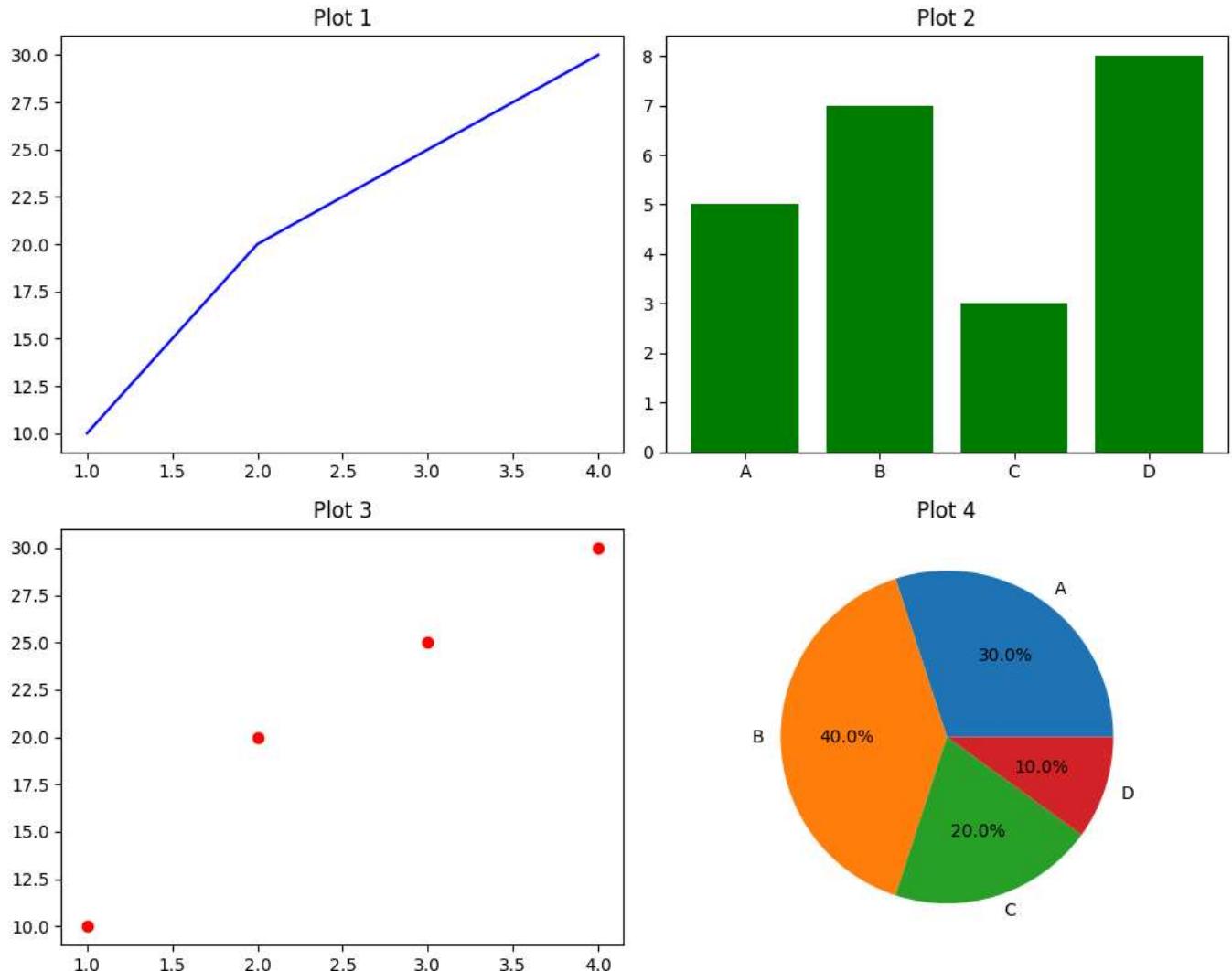
categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]
axs[0, 1].bar(categories, values, color='green') # Biểu đồ cột
axs[0, 1].set_title("Plot 2")

axs[1, 0].scatter(x, y, color='red') # Biểu đồ tán xạ
axs[1, 0].set_title("Plot 3")

sizes = [30, 40, 20, 10]
labels = ['A', 'B', 'C', 'D']
axs[1, 1].pie(sizes, labels=labels, autopct='%1.1f%%') # Biểu đồ hình tròn
axs[1, 1].set_title("Plot 4")

plt.tight_layout()
plt.show()

```



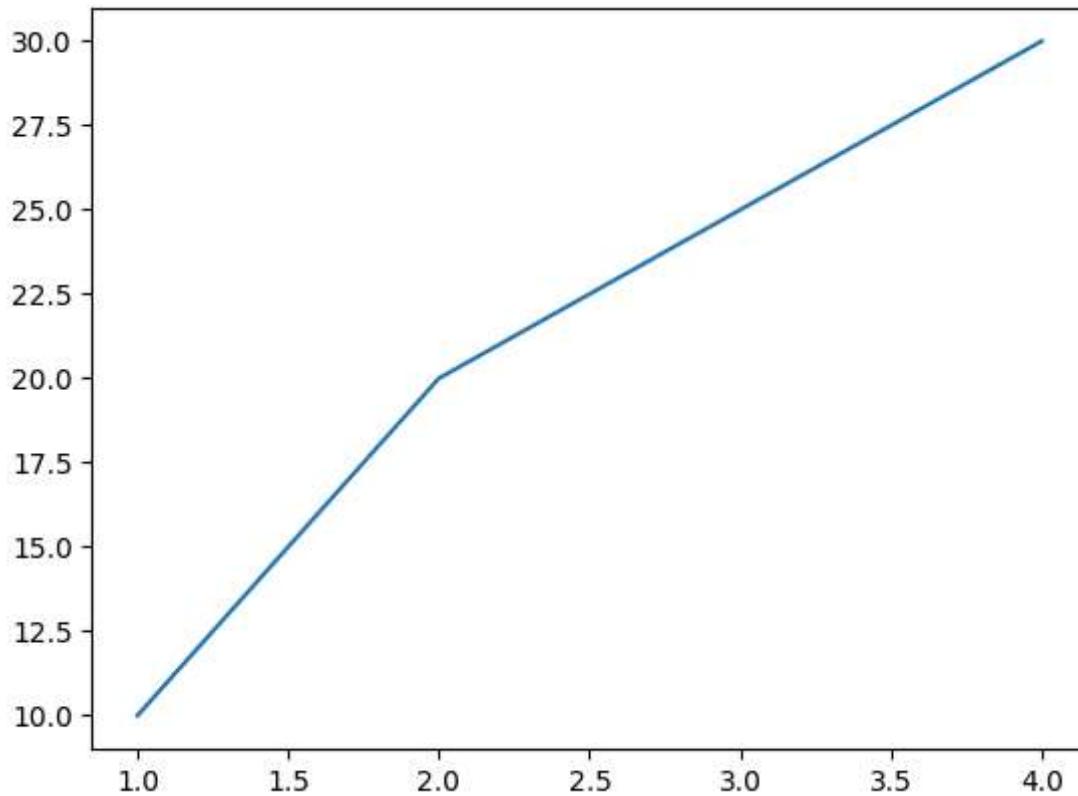
**Chú thích:**

- plt.subplots(2, 2): Tạo một bố cục 2x2 chứa 4 biểu đồ.

## 5.6 Lưu biểu đồ

Matplotlib cho phép lưu biểu đồ dưới nhiều định dạng.

```
plt.plot(x, y)
plt.savefig("line_chart.png", dpi=300, bbox_inches='tight')
plt.show()
```

**Chú thích:**

- dpi=300: Tăng độ phân giải cho hình ảnh.
- bbox\_inches='tight': Đảm bảo hình ảnh không bị cắt khi lưu.

## 5.7 Thực hành

- **Bài tập 1:** Sử dụng dữ liệu doanh số bán hàng hàng tháng để tạo biểu đồ đường và biểu đồ cột.
- **Bài tập 2:** Dùng dữ liệu về số lượng học sinh theo từng khoa để tạo biểu đồ hình tròn.
- **Bài tập 3:** Tạo một bố cục  $2 \times 2$ , hiển thị 4 biểu đồ khác nhau (đường, cột, tán xạ, hình tròn) dựa trên dữ liệu bạn tự tạo.

## 5.8 Tổng kết chương 5

**Nội dung:** Trong Chương 5 đã cung cấp nền tảng vững chắc về cách sử dụng Matplotlib để trực quan hóa dữ liệu. Trong chương 6 chúng ta sẽ tiếp tục với Seaborn, một thư viện nâng cao được xây dựng trên Matplotlib.

# Chương 6: Trực quan hóa Dữ liệu với Seaborn

## Contents

- Trực quan hóa Dữ liệu với Seaborn
- 6.1 Giới thiệu về Seaborn
- 6.2 Các biểu đồ cơ bản với Seaborn
- 6.3 Các biểu đồ nâng cao
- 6.4 Tùy chỉnh và cài đặt giao diện
- 6.5 Lưu biểu đồ
- 6.6 Thực hành
- 6.7 Tổng kết chương 6

## Trực quan hóa Dữ liệu với Seaborn

Seaborn là một thư viện trực quan hóa dữ liệu mạnh mẽ, được xây dựng trên Matplotlib. Thư viện này cung cấp các biểu đồ nâng cao và giao diện đơn giản để làm việc với dữ liệu dạng bảng, đặc biệt khi tích hợp với pandas.

## 6.1 Giới thiệu về Seaborn

### 6.1.1 Cài đặt Seaborn

Để cài đặt Seaborn, sử dụng lệnh:

```
pip install seaborn
```

## 6.1.2 Nhập thư viện và dữ liệu mẫu

Seaborn đi kèm với nhiều tập dữ liệu mẫu (datasets). Bạn có thể tải và sử dụng chúng để thực hành.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Tải dữ liệu mẫu
tips = sns.load_dataset("tips")
print(tips.head())
```

```
import plotly.io as pio
pio.renderers.default = 'vscode'
```

## 6.2 Các biểu đồ cơ bản với Seaborn

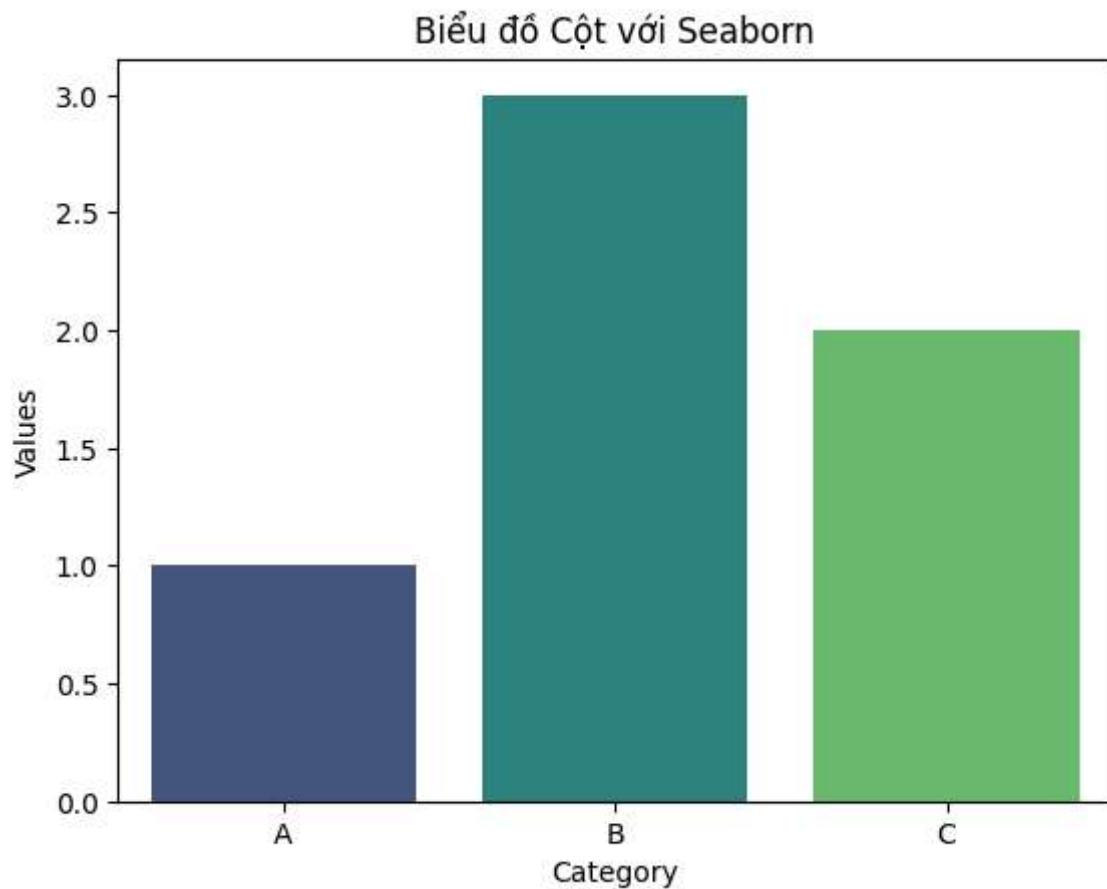
### 6.2.1 Biểu đồ cột và phân phối

Biểu đồ cột:

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import warnings

warnings.filterwarnings("ignore", category=FutureWarning)

data = {
    "Category": ["A", "B", "C"],
    "Values": [1, 3, 2]
}
df = pd.DataFrame(data)
sns.barplot(data=df, x="Category", y="Values", palette="viridis")
plt.title('Biểu đồ Cột với Seaborn')
plt.show()
```



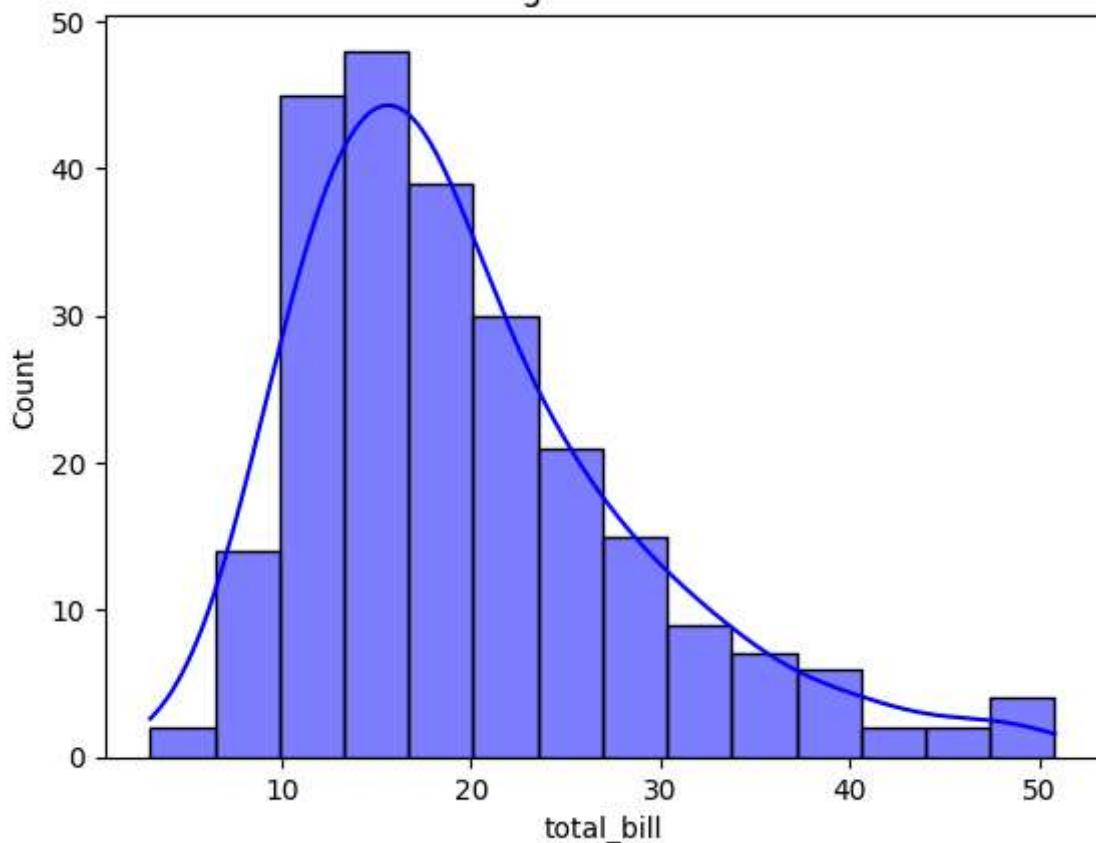
Biểu đồ phân phối:

```
import seaborn as sns
import matplotlib.pyplot as plt

tips = sns.load_dataset("tips")

sns.histplot(tips['total_bill'], kde=True, color='blue')
plt.title("Histogram with KDE")
plt.show()
```

## Histogram with KDE



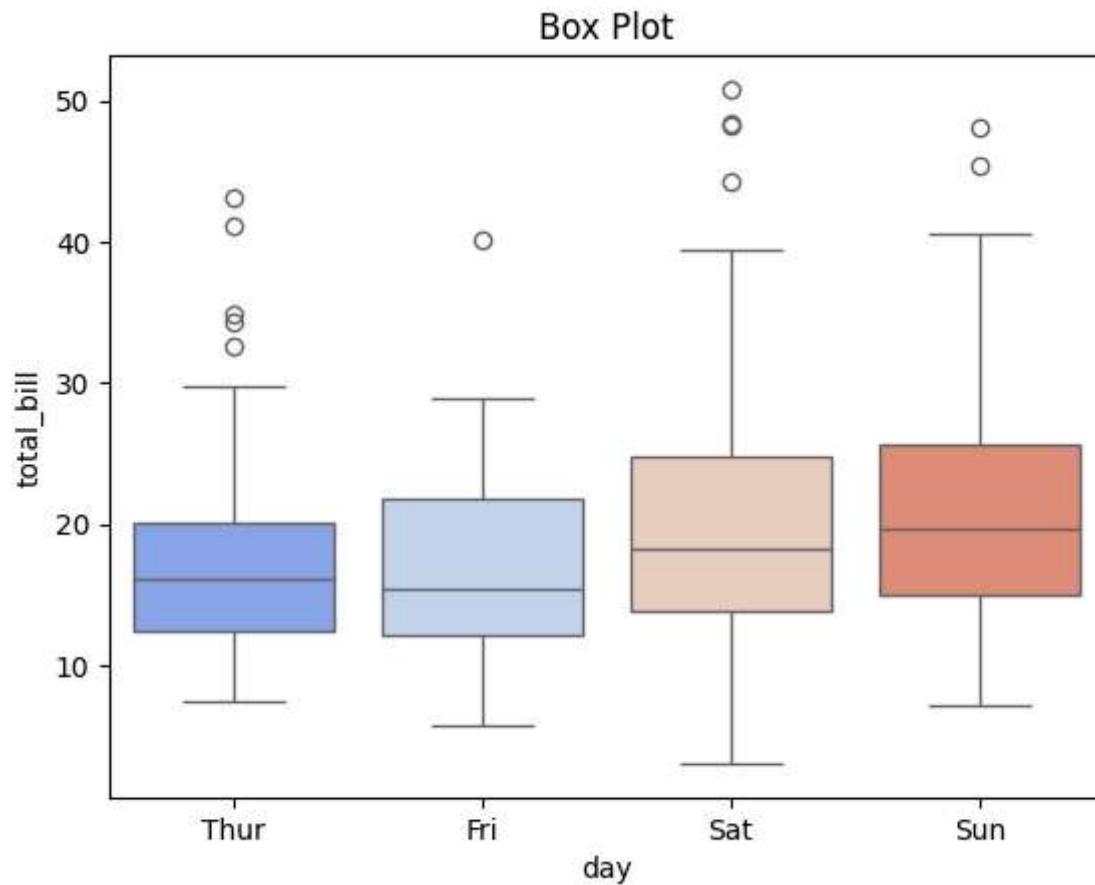
### Chú thích:

- Biểu đồ hiển thị phân phối hóa đơn total\_bill trong tập dữ liệu. Đường KDE cho thấy mật độ xác suất của dữ liệu.

#### 6.2.2 Biểu đồ hộp (Box Plot)

Dùng để hiển thị phân phối và phát hiện giá trị ngoại lai.

```
sns.boxplot(x="day", y="total_bill", data=tips, hue="day", palette="coolwarm", dodge=True)
plt.legend([],[], frameon=False) #Ẩn legend nếu không cần
plt.title("Box Plot")
plt.show()
```

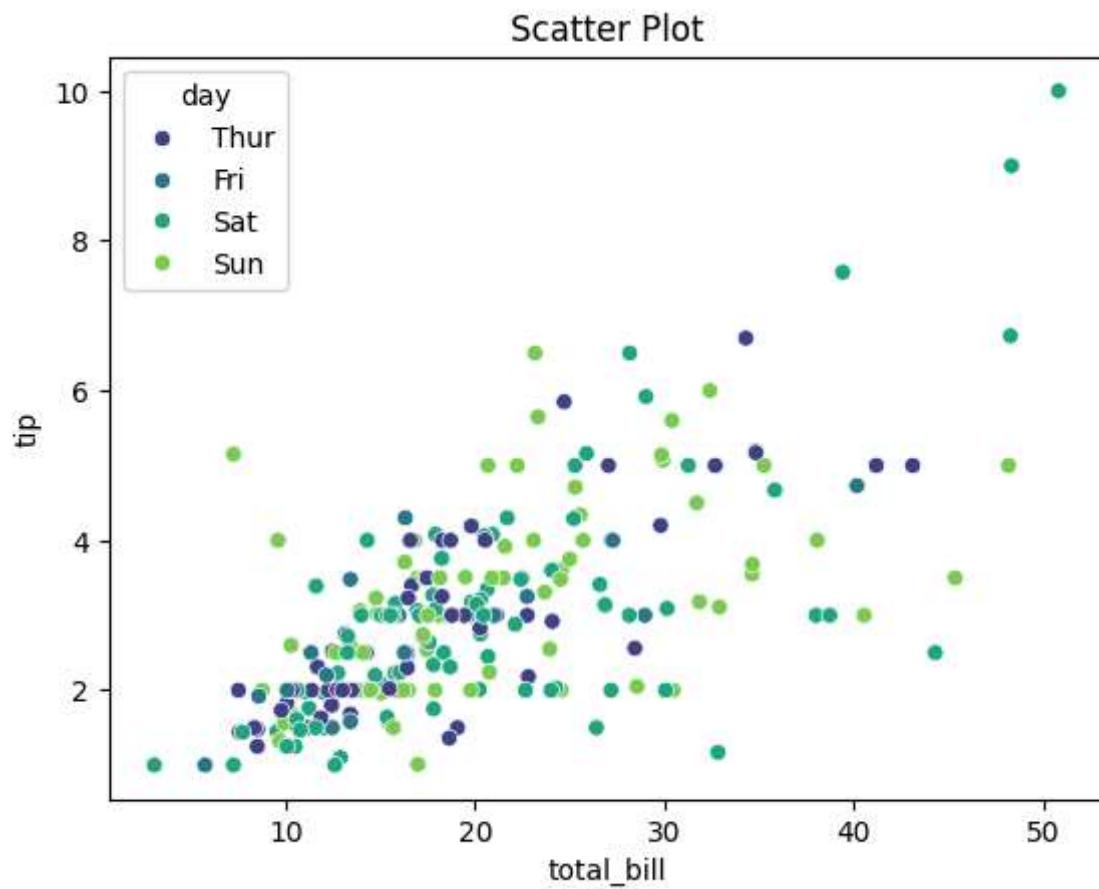


#### Chú thích:

- Biểu đồ cho thấy phân phối giá trị total\_bill theo các ngày trong tuần (day). Các giá trị ngoài dải (outliers) được biểu thị bằng các dấu chấm.

**6.2.3 Biểu đồ tán xạ (Scatter Plot)** Dùng để hiển thị mối quan hệ giữa hai biến.

```
sns.scatterplot(x="total_bill", y="tip", hue="day", data=tips, palette="viridis")
plt.title("Scatter Plot")
plt.show()
```

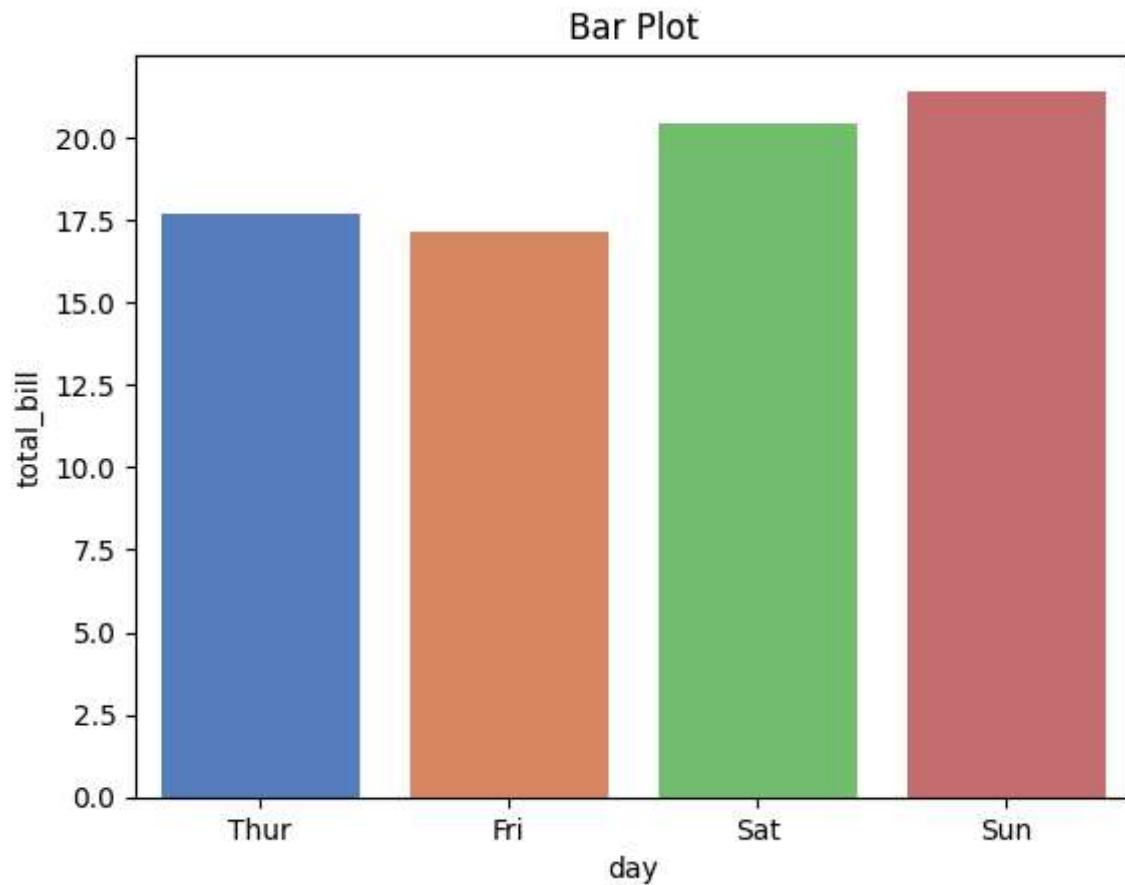


#### Chú thích:

- Biểu đồ này cho thấy mối quan hệ giữa tổng hóa đơn (total\_bill) và tiền tip (tip) theo từng ngày. Các màu sắc khác nhau đại diện cho các ngày trong tuần.

**6.2.4 Biểu đồ thanh (Bar Plot)** Dùng để hiển thị giá trị trung bình của một biến theo nhóm.

```
sns.barplot(x="day", y="total_bill", data=tips, errorbar=None, palette="muted", hue="day")
plt.title("Bar Plot")
plt.show()
```



#### Chú thích:

- Biểu đồ này cho thấy tổng hóa đơn trung bình (total\_bill) theo từng ngày trong tuần. Các thanh đại diện cho giá trị trung bình của từng nhóm.

## 6.3 Các biểu đồ nâng cao

**6.3.1 Biểu đồ ma trận tương quan (Heatmap)** Dùng để trực quan hóa mối tương quan giữa các biến số.

```

import seaborn as sns
import matplotlib.pyplot as plt

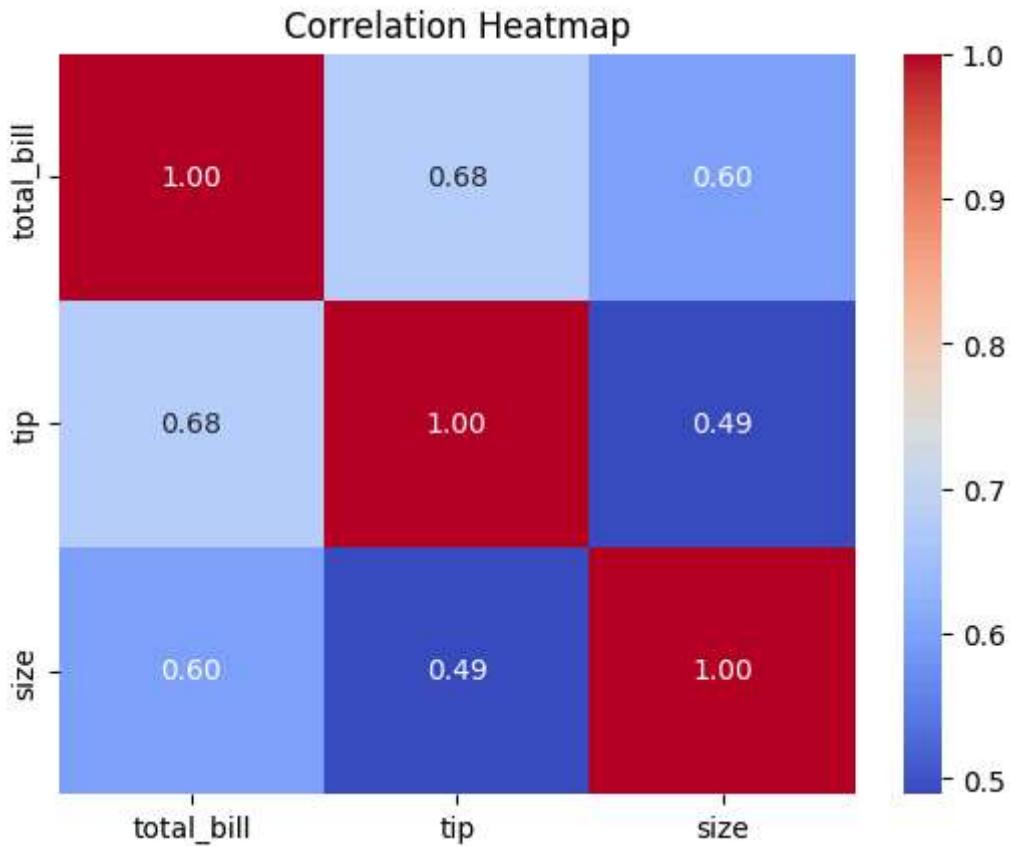
tips = sns.load_dataset("tips")

numeric_tips = tips.select_dtypes(include='number')

correlation = numeric_tips.corr()

sns.heatmap(correlation, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()

```



### Chú thích:

- Biểu đồ này hiển thị mối quan hệ tuyến tính giữa các cặp biến số trong dữ liệu. Giá trị gần 1 hoặc -1 cho thấy mối tương quan mạnh, trong khi giá trị gần 0 cho thấy mối tương quan yếu.

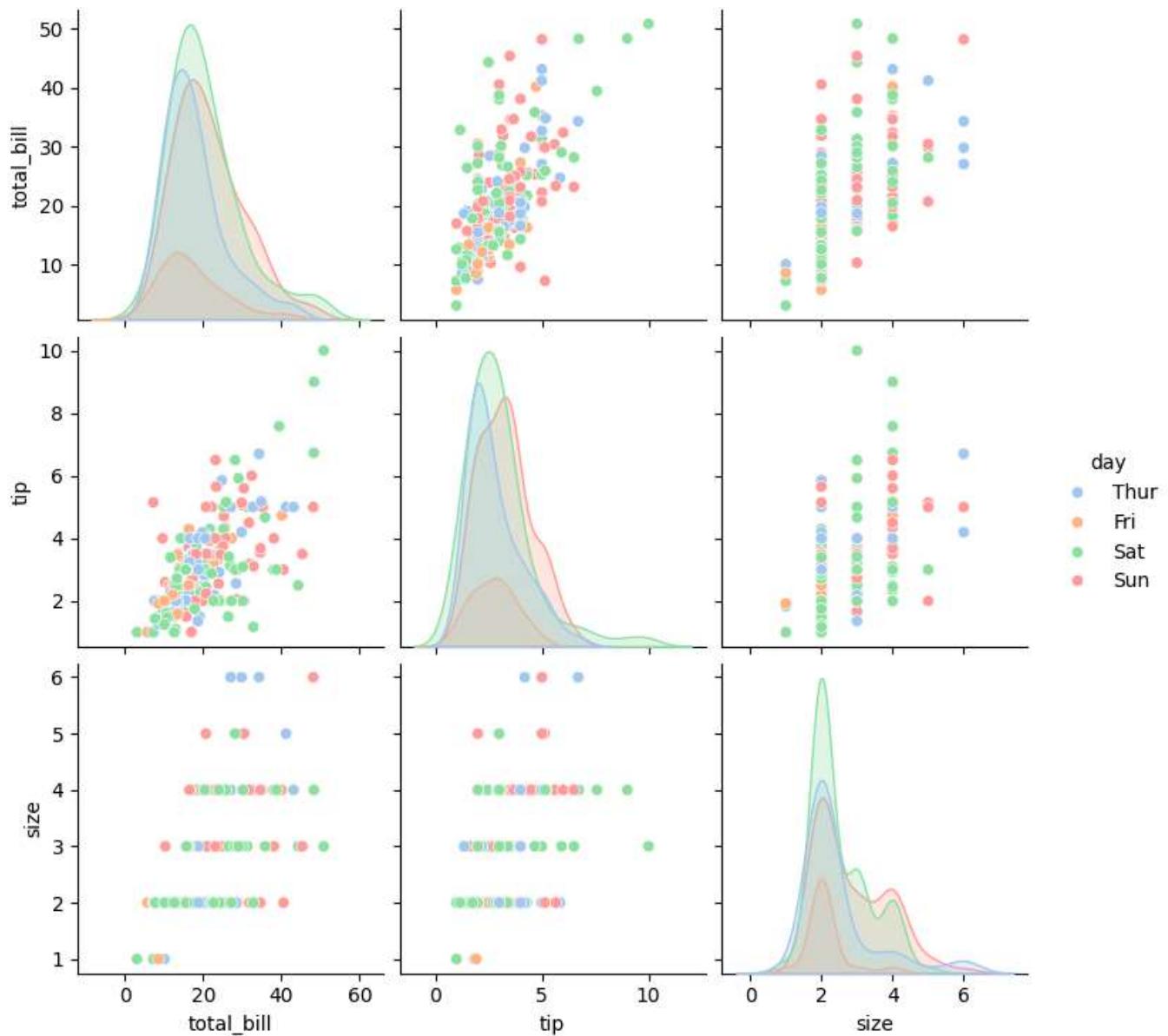
### 6.3.2 Biểu đồ ghép (Pair Plot)

Hiển thị mối quan hệ giữa tất cả các cặp biến.

```

sns.pairplot(tips, hue="day", palette="pastel")
plt.show()

```

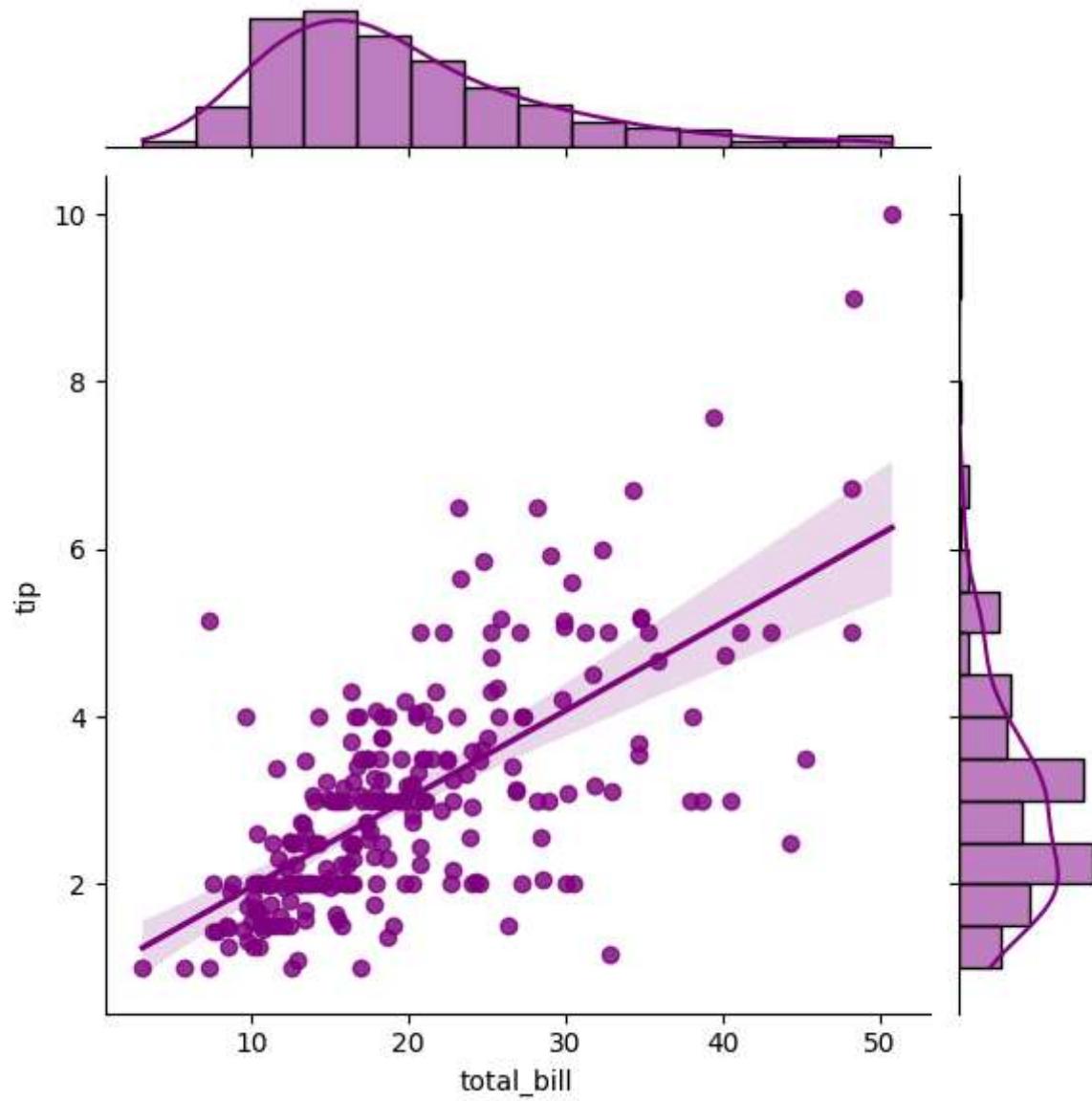


### Chú thích:

- Mỗi ô trong biểu đồ đại diện cho một cặp biến số. Đường chéo chính chứa các biểu đồ phân phối, còn các ô khác là các biểu đồ tán xạ giữa hai biến.

**6.3.3 Biểu đồ phân phối dạng khớp (Joint Plot)** Hiển thị mối quan hệ giữa hai biến cùng với biểu đồ phân phối của chúng.

```
sns.jointplot(x="total_bill", y="tip", data=tips, kind="reg", color="purple")
plt.show()
```



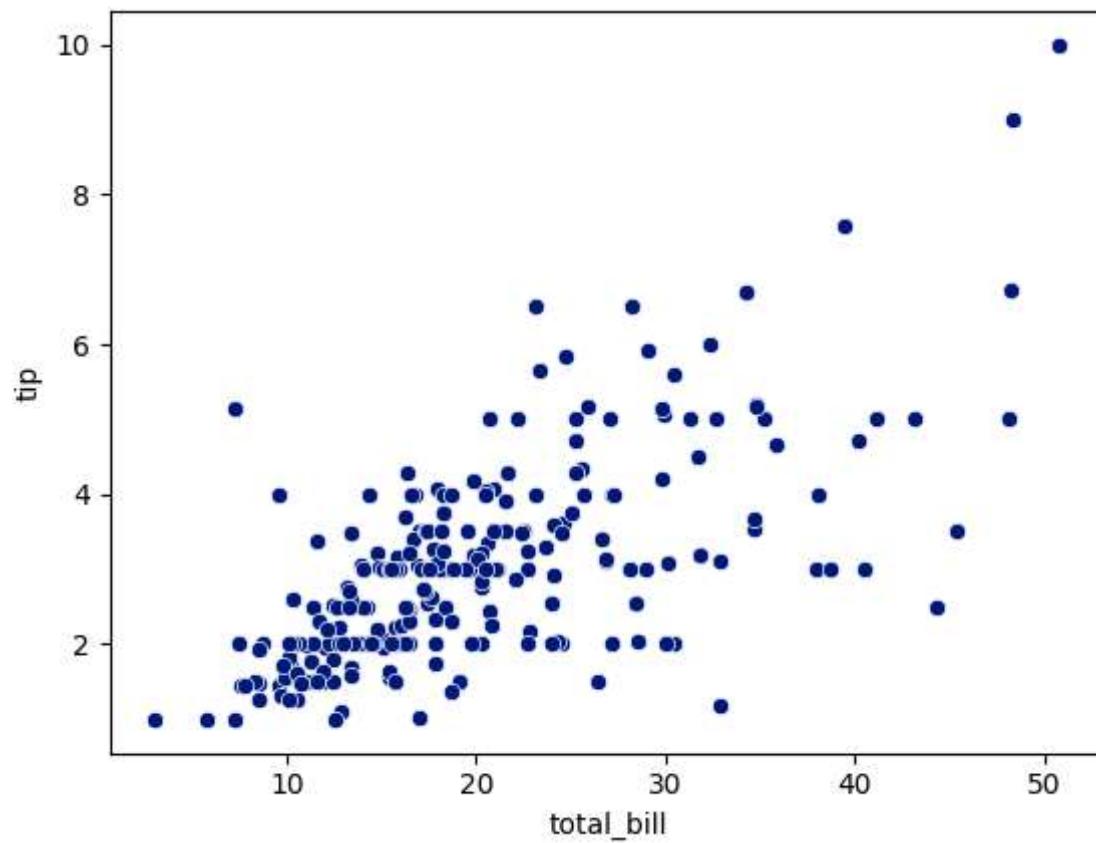
**Chú thích:**

- Biểu đồ này không chỉ hiển thị mối quan hệ giữa hai biến mà còn hiển thị các phân phối biên (marginal distributions). Đường hồi quy (regression) chỉ ra xu hướng tổng quát.

## 6.4 Tùy chỉnh và cài đặt giao diện

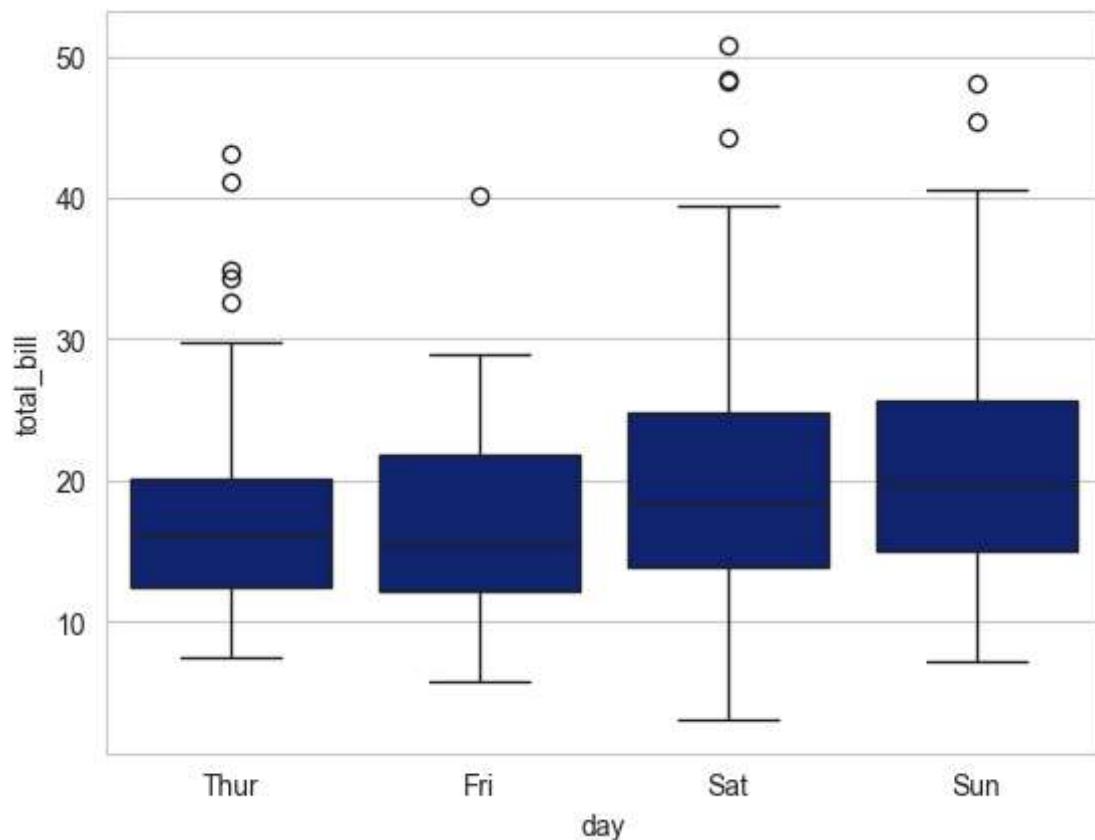
**6.4.1 Tùy chỉnh màu sắc** Seaborn cung cấp các bảng màu sẵn có.

```
sns.set_palette("dark")
sns.scatterplot(x="total_bill", y="tip", data=tips)
plt.show()
```



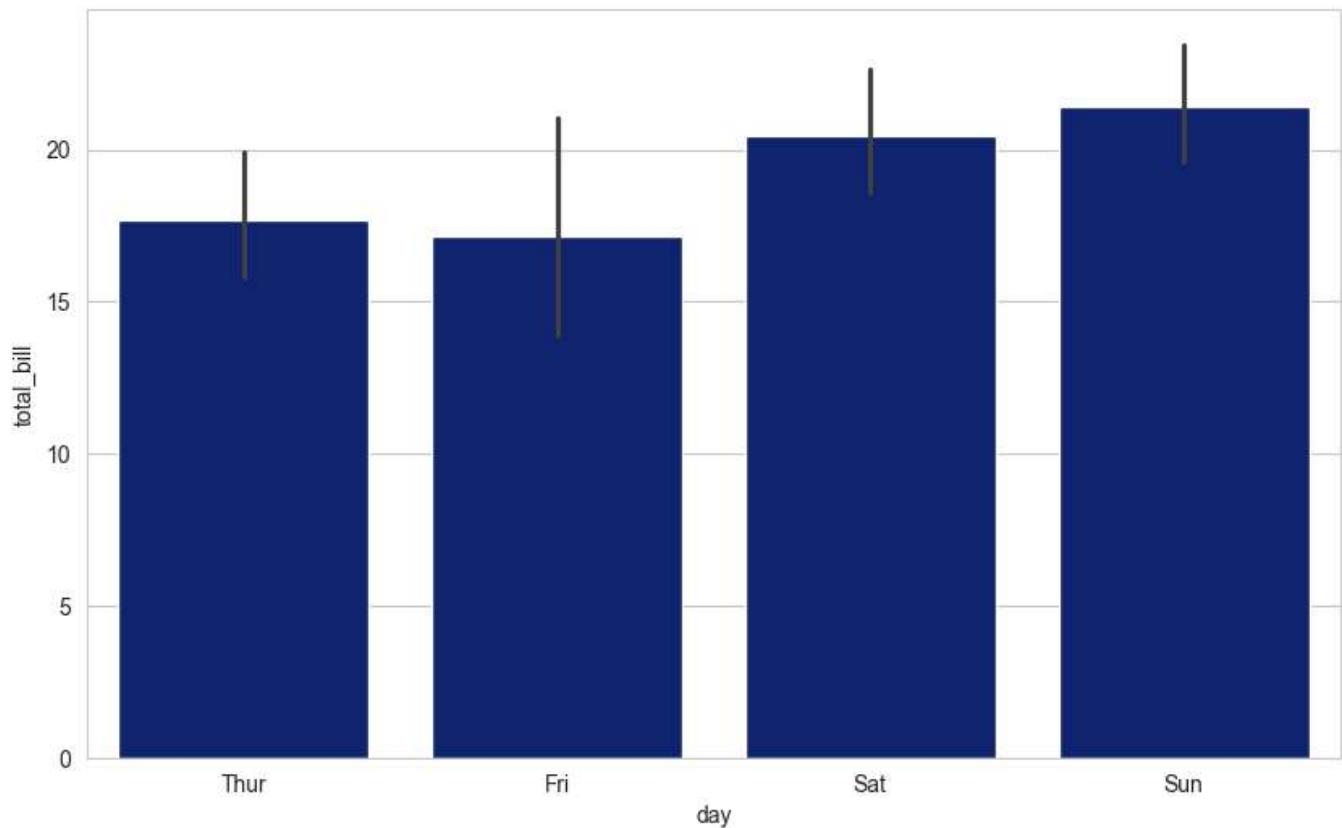
**6.4.2 Thay đổi phong cách** Bạn có thể thay đổi phong cách để biểu đồ trông chuyên nghiệp hơn.

```
sns.set_style("whitegrid")
sns.boxplot(x="day", y="total_bill", data=tips)
plt.show()
```



#### 6.4.3 Điều chỉnh kích thước biểu đồ

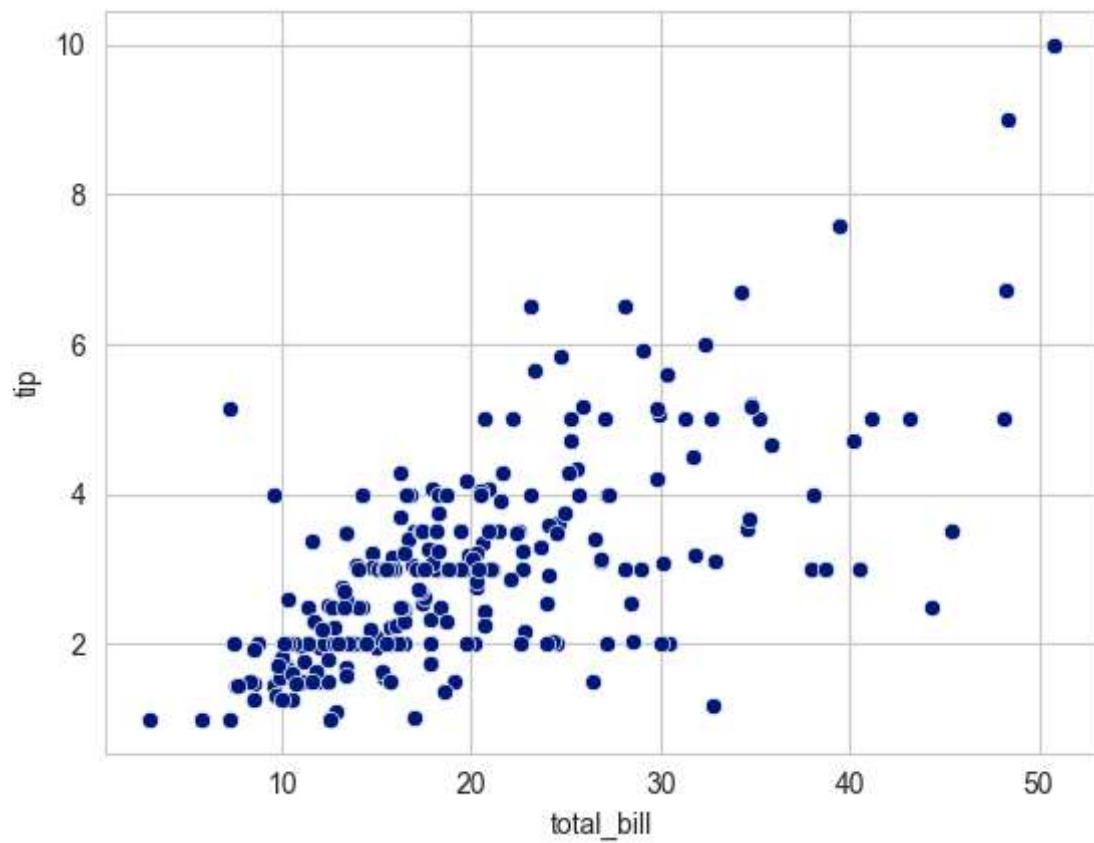
```
plt.figure(figsize=(10, 6))
sns.barplot(x="day", y="total_bill", data=tips)
plt.show()
```



## 6.5 Lưu biểu đồ

Seaborn hoạt động trên nền Matplotlib, nên bạn có thể lưu biểu đồ giống như Matplotlib.

```
sns.scatterplot(x="total_bill", y="tip", data=tips)
plt.savefig("scatter_plot.png", dpi=300, bbox_inches="tight")
plt.show()
```



## 6.6 Thực hành

**Bài tập 1:** Sử dụng tập dữ liệu tips để tạo:

- Biểu đồ phân phối với sns.histplot.
- Biểu đồ hộp nhóm theo day.

**Bài tập 2:** Tải dữ liệu mẫu iris và tạo:

- Biểu đồ ghép (Pair Plot) với sns.pairplot.
- Biểu đồ nhiệt mối tương quan giữa các biến.

**Bài tập 3:** Với một tập dữ liệu của riêng bạn, hãy:

- Tạo biểu đồ thanh hiển thị giá trị trung bình của một biến theo nhóm.
- Tùy chỉnh màu sắc và phong cách biểu đồ.

## 6.7 Tổng kết chương 6

**Nội dung:** Trong chương 6 đã cung cấp cách sử dụng Seaborn để trực quan hóa dữ liệu một cách đơn giản nhưng chuyên nghiệp. Trong chương 7 bạn sẽ học cách làm việc với Plotly để tạo các biểu đồ tương tác mạnh mẽ.

# Chương 7: Trực quan hóa Dữ liệu với Plotly

## Contents

- Trực quan hóa Dữ liệu với Plotly
- 7.1 Giới thiệu về Plotly
- 7.2 Tạo Biểu Đồ Cơ Bản với Plotly
- 7.3 Các Biểu Đồ Tương Tác
- 7.4 Tùy chỉnh Biểu Đồ
- 7.5 Tạo Biểu Đồ Nâng Cao
- 7.6 Xuất và Chia Sẻ Biểu Đồ
- 7.7 Các Ví Dụ Thực Tế
- 7.8 Tài liệu tham khảo
- 7.9 Tổng kết chương 7

## Trực quan hóa Dữ liệu với Plotly

Chương này sẽ hướng dẫn cách sử dụng thư viện Plotly để tạo các biểu đồ tương tác trong Python, từ những biểu đồ cơ bản đến nâng cao. Nội dung chương có thể được tổ chức thành các phần sau:

### 7.1 Giới thiệu về Plotly

Plotly là một thư viện mã nguồn mở giúp tạo ra các biểu đồ tương tác đẹp mắt và dễ sử dụng trong Python. Nó hỗ trợ nhiều loại biểu đồ khác nhau và cho phép xuất biểu đồ dưới dạng

HTML hoặc hình ảnh.

### Lợi ích của việc sử dụng Plotly:

- **Tính tương tác cao:** Người dùng có thể tương tác với biểu đồ (zoom, di chuyển, hover, click).
- **Hỗ trợ nhiều loại biểu đồ:** Bao gồm biểu đồ đường, cột, phân tán, hộp, bản đồ, biểu đồ 3D, và nhiều loại khác.
- **Khả năng xuất biểu đồ:** Plotly hỗ trợ xuất biểu đồ dưới dạng HTML hoặc hình ảnh.

### Cách cài đặt và thiết lập:

```
pip install plotly
```

### Để xuất biểu đồ dưới dạng hình ảnh, bạn cần cài thêm kaleido:

```
pip install kaleido
```

## 7.2 Tạo Biểu Đồ Cơ Bản với Plotly

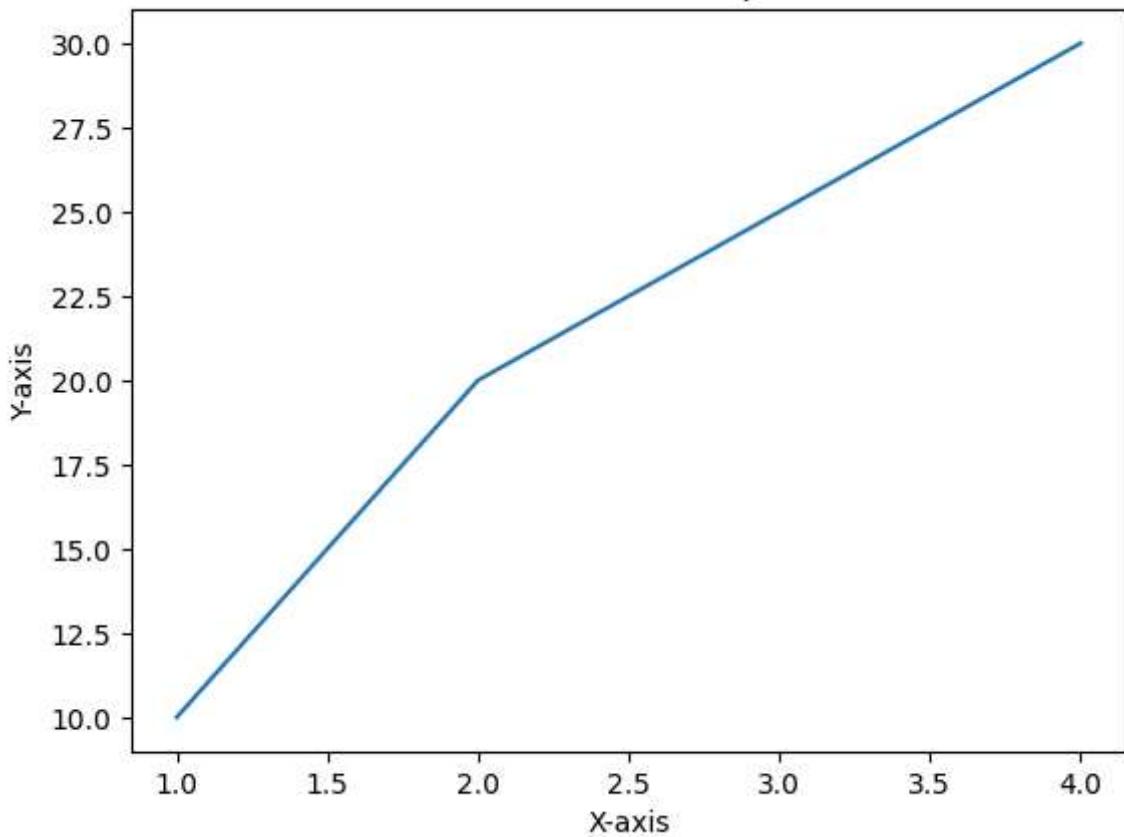
### 7.2.1 Biểu đồ đường (Line Chart):

- Biểu đồ đường giúp bạn trực quan hóa sự thay đổi của một biến theo thời gian hoặc các giá trị liên tục.

```
import matplotlib.pyplot as plt

# Tạo biểu đồ đường
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
plt.plot(x, y)
plt.title("Line Chart Example")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

### Line Chart Example



**Chú thích:**

- go.Figure(): Tạo một đối tượng Figure.
- go.Scatter(): Dùng để vẽ biểu đồ đường. mode='lines+markers' giúp vẽ cả đường và các điểm đánh dấu trên đó.
- fig.update\_layout(): Cập nhật các thông số như tiêu đề và nhãn trục.

**7.2.2 Biểu đồ cột (Bar Chart):**

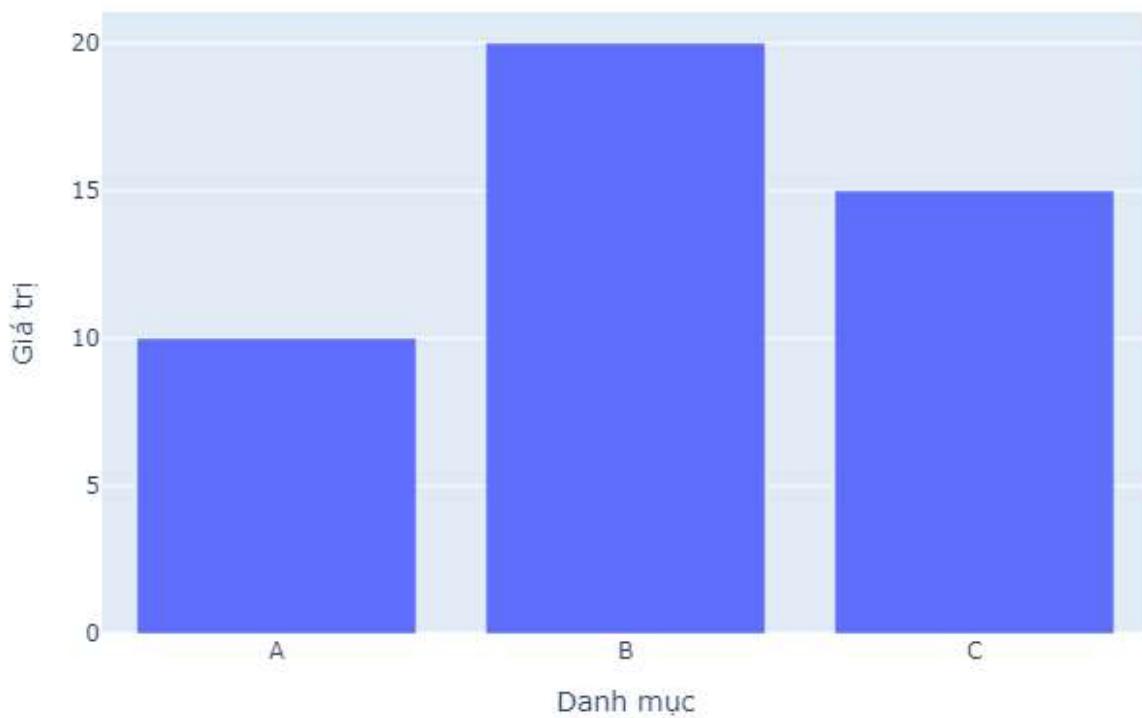
- Biểu đồ cột giúp bạn so sánh giá trị giữa các danh mục.

```
import plotly.graph_objects as go

fig = go.Figure(data=go.Bar(x=['A', 'B', 'C'], y=[10, 20, 15]))
fig.update_layout(title="Biểu đồ Cột", xaxis_title="Danh mục", yaxis_title="Giá trị")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

## Biểu đồ Cột



### Chú thích:

- go.Bar(): Dùng để vẽ biểu đồ cột. Trục x là các danh mục và trục y là các giá trị tương ứng.

## 7.3 Các Biểu Đồ Tương Tác

Plotly nổi bật với khả năng tạo các biểu đồ tương tác, nơi người dùng có thể zoom in/out, di chuyển, và hover để xem thông tin chi tiết.

### 7.3.1 Biểu đồ phân tán (Scatter Plot):

Biểu đồ phân tán giúp bạn tìm mối quan hệ giữa hai biến.

```

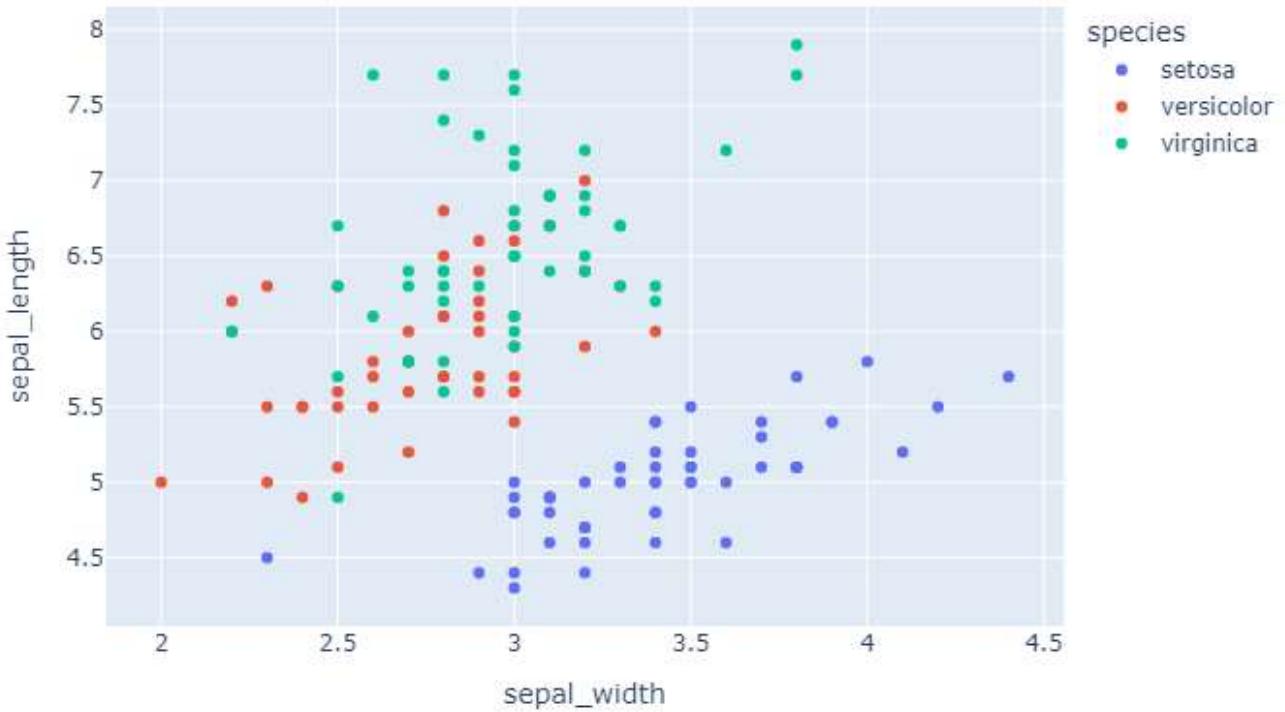
import plotly.express as px

df = px.data.iris()
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species",
                  title="Biểu đồ Phân tán")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")

```

Biểu đồ Phân tán



### Chú thích:

- px.scatter(): Dùng để tạo biểu đồ phân tán, trong đó các điểm được tô màu theo từng nhóm (species).

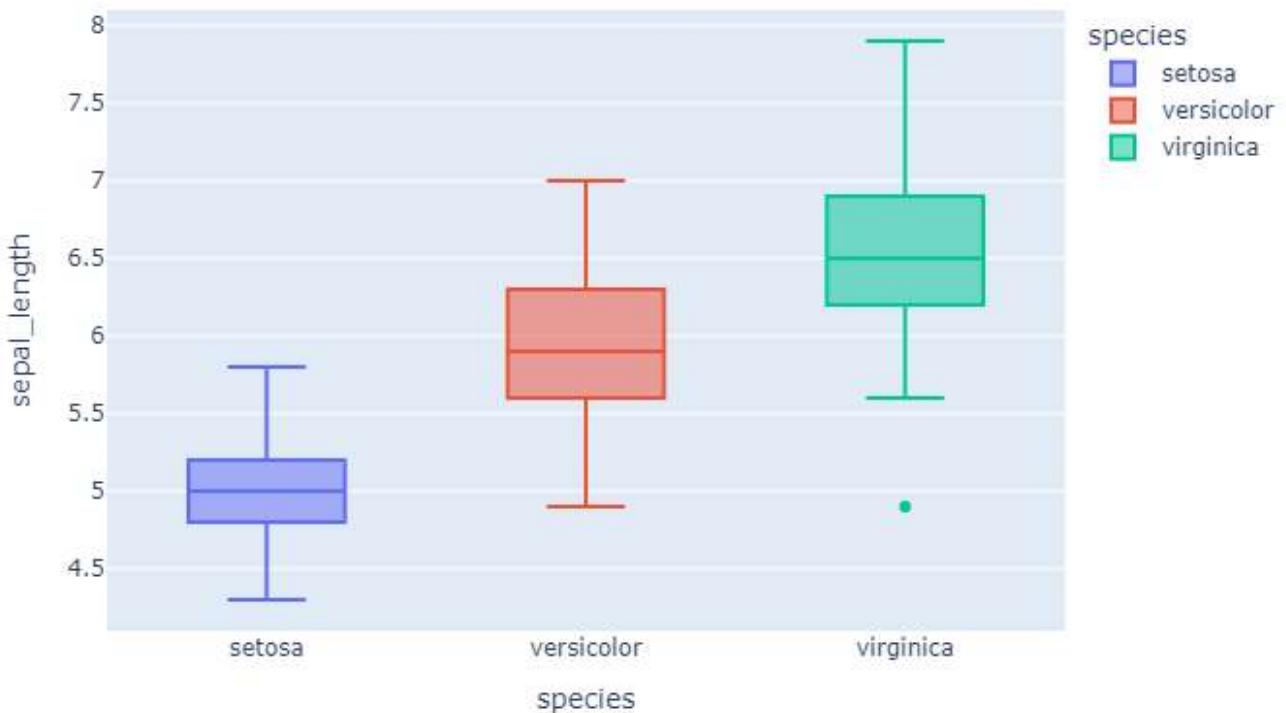
### 7.3.2 Biểu đồ hộp (Box Plot):

Biểu đồ hộp giúp bạn quan sát sự phân bố của dữ liệu và phát hiện các giá trị ngoại lai.

```
fig = px.box(df, x="species", y="sepal_length", color="species",
              title="Biểu đồ Hộp")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

Biểu đồ Hộp



### Chú thích:

- px.box(): Tạo biểu đồ hộp cho dữ liệu, cho thấy các phần tử cơ bản như trung vị, phần tư, và giá trị ngoại lai.

## 7.4 Tùy chỉnh Biểu Đồ

### Thay đổi màu sắc, kích thước, và kiểu dáng

Plotly cho phép bạn dễ dàng tùy chỉnh các yếu tố của biểu đồ như màu sắc, kích thước, và kiểu dáng.

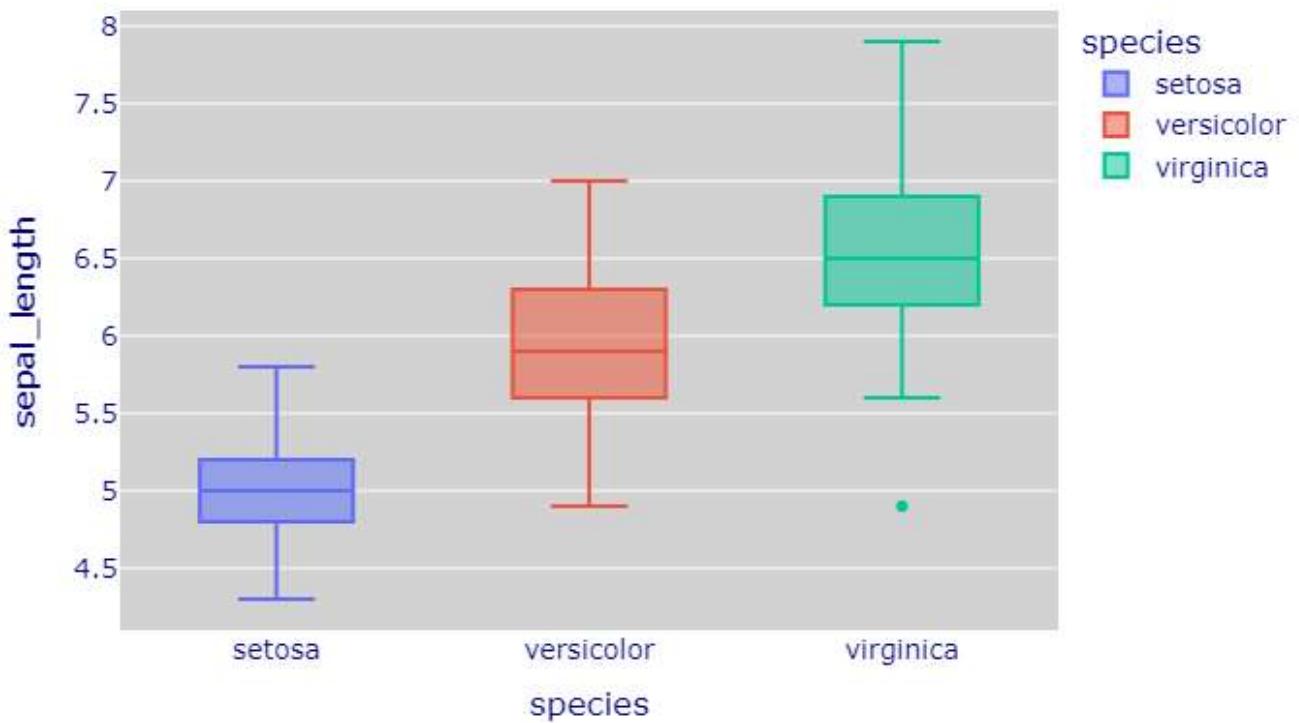
```

fig.update_layout(
    title="Tùy chỉnh Biểu đồ",
    font=dict(size=14, color="darkblue"),
    plot_bgcolor="lightgray"
)
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")

```

## Tùy chỉnh Biểu đồ



### Chú thích:

- `update_layout()`: Tùy chỉnh giao diện của biểu đồ, bao gồm màu sắc nền, phông chữ, và tiêu đề.

## 7.5 Tạo Biểu Đồ Nâng Cao

Plotly cũng hỗ trợ các loại biểu đồ nâng cao, ví dụ như biểu đồ bản đồ và biểu đồ 3D.

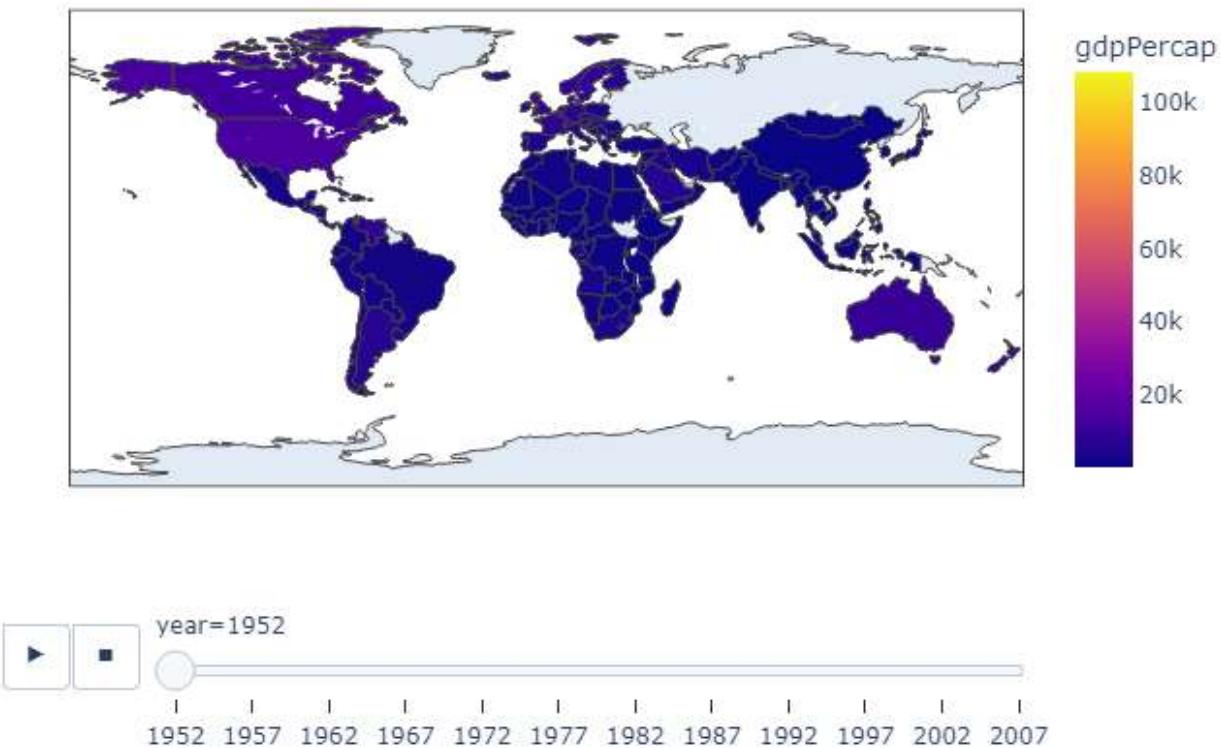
### 7.5.1 Biểu đồ bản đồ (Choropleth Map):

Biểu đồ bản đồ giúp bạn trực quan hóa dữ liệu theo vị trí địa lý.

```
fig = px.choropleth(px.data.gapminder(),
                     locations="iso_alpha",
                     color="gdpPercap",
                     hover_name="country",
                     animation_frame="year",
                     title="Biểu đồ Bản đồ")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

Biểu đồ Bản đồ



### Chú thích:

- px.choropleth(): Tạo biểu đồ bản đồ, trong đó các quốc gia được tô màu theo GDP.

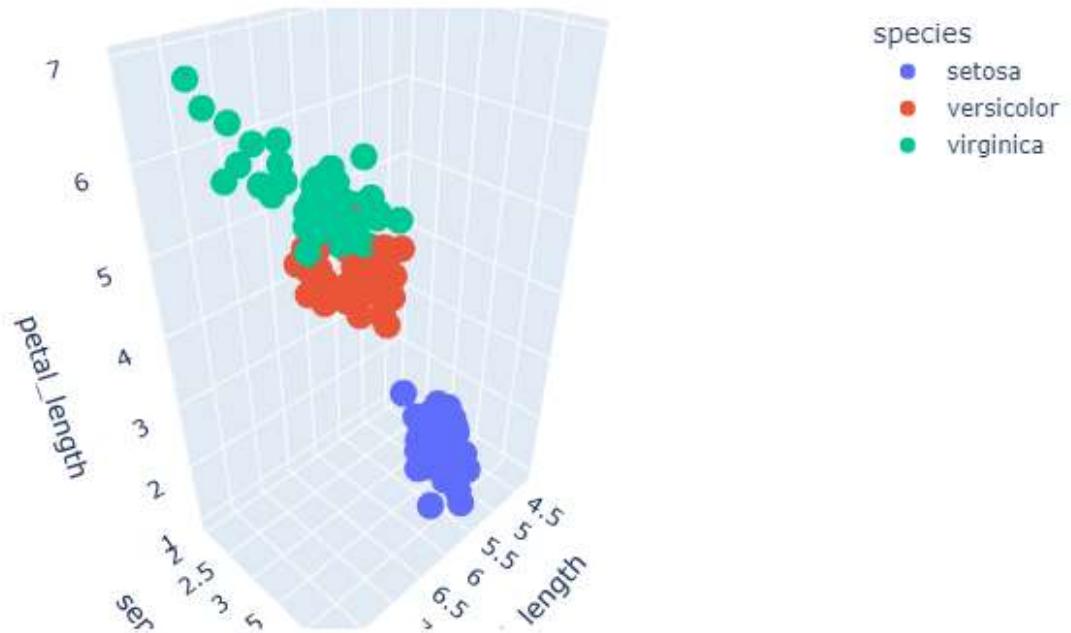
### 7.5.2 Biểu đồ 3D (3D Scatter):

Biểu đồ 3D giúp bạn quan sát mối quan hệ giữa ba biến.

```
fig = px.scatter_3d(df, x="sepal_length", y="sepal_width", z="petal_length",
                     color="species", title="Biểu đồ 3D")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

Biểu đồ 3D

**Chú thích:**

- px.scatter\_3d(): Tạo biểu đồ phân tán 3D với ba trục không gian.

## 7.6 Xuất và Chia Sẻ Biểu Đồ

### Lưu biểu đồ dưới dạng HTML:

- Lưu biểu đồ dưới dạng ảnh

```
fig.write_image("chart.png")
```

- Hiển thị trong Jupyter Book

```
from IPython.display import Image  
Image("chart.png")
```

## 7.7 Các Ví Dụ Thực Tế

- Phân tích dữ liệu doanh thu bán hàng.
- Theo dõi dữ liệu môi trường.
- Trực quan hóa dữ liệu thống kê trong lĩnh vực y tế.

## 7.8 Tài liệu tham khảo

- Liệt kê các tài liệu tham khảo:
- [Trang chủ Plotly](#)
- Tài liệu Python chính thức của Plotly.

## 7.9 Tổng kết chương 7

**Nội dung:** Trong chương 7 này chúng ta đã làm biểu đồ từ cơ bản đến nâng cao chỉ với những đoạn code đơn giản, trong chương 8 chúng ta sẽ đến các Kỹ thuật Trực quan hóa Nâng cao.

# Chương 8: Các Kỹ thuật Trực quan hóa Nâng cao

## Contents

- Các Kỹ thuật Trực quan hóa Nâng cao
  - 8.1 Giới thiệu về Kỹ thuật Trực quan hóa Nâng cao
  - 8.2 Trực quan hóa Dữ liệu 3D
  - 8.4 Network Graphs (Biểu đồ Mạng)
  - 8.5 Tùy chỉnh và Kết hợp Biểu đồ
  - 8.6 Lưu trữ và Chia sẻ Biểu đồ
  - 8.7 Bài tập Thực hành
  - 8.8 Tổng kết chương 8

## Các Kỹ thuật Trực quan hóa Nâng cao

Chương này sẽ giúp bạn tìm hiểu các kỹ thuật trực quan hóa nâng cao, cho phép khai thác sâu hơn các công cụ và phương pháp để phân tích và hiển thị dữ liệu phức tạp. Nội dung của chương được chia thành các phần sau:

## 8.1 Giới thiệu về Kỹ thuật Trực quan hóa Nâng cao

**Mục tiêu:** Hiểu lý do và bối cảnh sử dụng các kỹ thuật trực quan hóa nâng cao.

**Lý thuyết:**

- Trực quan hóa nâng cao giúp ta phân tích các dữ liệu lớn và phức tạp, từ đó nhận diện các mối quan hệ, xu hướng hoặc bất thường mà các phương pháp phân tích truyền thống không thể làm được.
- Các công cụ và thư viện như Plotly, Seaborn, Bokeh, Dash rất thích hợp cho việc tạo ra các đồ thị nâng cao với khả năng tương tác và tính linh hoạt cao.
- Các loại đồ thị nâng cao bao gồm các đồ thị 3D, Heatmaps, và Network Graphs. Những công cụ này giúp hiển thị các mối quan hệ trong dữ liệu từ nhiều góc độ khác nhau.

## 8.2 Trực quan hóa Dữ liệu 3D

**Ứng dụng:** Biểu đồ 3D giúp hiển thị các mối quan hệ giữa ba hoặc nhiều biến số. Đây là một công cụ quan trọng khi bạn cần phân tích các dữ liệu không gian hoặc mối quan hệ đa chiều.

**Công cụ sử dụng:** Plotly, Matplotlib (Axes 3D).

**Ví dụ mã code:**

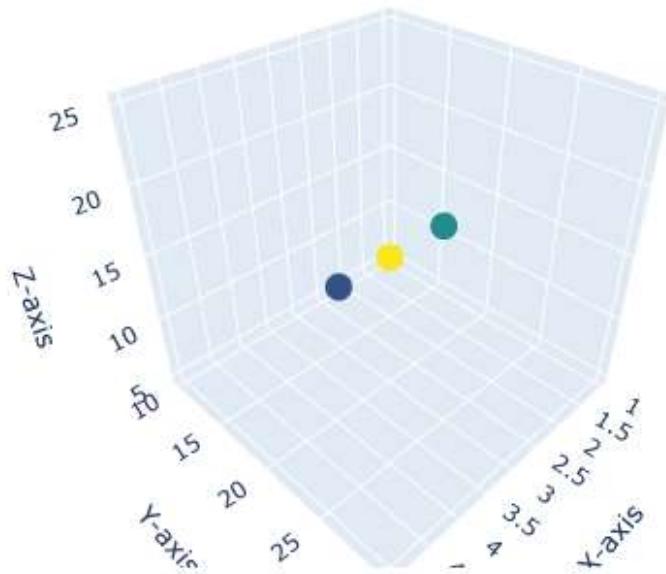
```
import plotly.graph_objects as go

x = [1, 2, 3, 4, 5]
y = [10, 20, 15, 25, 30]
z = [5, 15, 10, 20, 25]

fig = go.Figure(data=[go.Scatter3d(x=x, y=y, z=z, mode='markers',
                                     marker=dict(size=8, color=z, colorscale='Viridis')])
fig.update_layout(title="Biểu đồ 3D", scene=dict(xaxis_title="X-axis", yaxis_title="Y-axis"))
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

## Biểu đồ 3D



## 8.3 Heatmaps và Biểu đồ Ma trận

**Ứng dụng:** Heatmaps rất hữu ích trong việc phân tích mối tương quan giữa các biến số trong dữ liệu ma trận, ví dụ như các dữ liệu hành vi người dùng hoặc số liệu kinh tế theo tháng, năm.

**Công cụ sử dụng:** Seaborn, Plotly.

**Ví dụ mã code:**

```

import seaborn as sns
import matplotlib.pyplot as plt
from IPython.display import Image

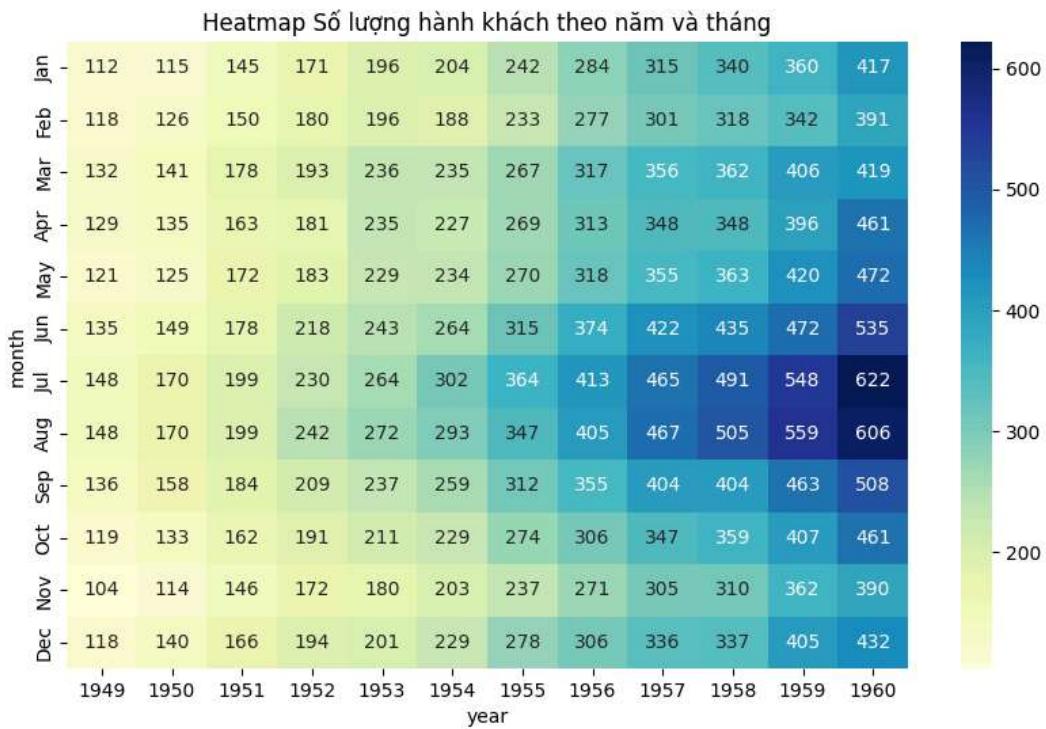
# Tải dữ liệu và tạo bảng pivot
data = sns.load_dataset("flights")
pivot_table = data.pivot(index="month", columns="year", values="passengers")

# Tạo heatmap
plt.figure(figsize=(10, 6)) # Tạo kích thước cho hình vẽ
sns.heatmap(pivot_table, annot=True, fmt="d", cmap="YlGnBu")
plt.title("Heatmap Số lượng hành khách theo năm và tháng")

# Lưu biểu đồ dưới dạng hình ảnh
plt.savefig("chart.png")

# Hiển thị ảnh đã lưu
plt.close() # Đóng biểu đồ sau khi lưu để không bị hiển thị lại
Image("chart.png")

```



### Chú thích:

- sns.heatmap tạo một heatmap với bảng màu được chọn, cùng với giá trị số hành khách hiển thị trên từng ô.
- plt.figure(figsize=(10, 6)) thay đổi kích thước của biểu đồ.
- plt.title("Heatmap...") tạo tiêu đề cho biểu đồ.

## 8.4 Network Graphs (Biểu đồ Mạng)

**Ứng dụng:** Biểu đồ mạng dùng để thể hiện các mối liên kết hoặc mạng lưới, ví dụ như mối quan hệ trong mạng xã hội hoặc trong các hệ thống dữ liệu phức tạp.

**Công cụ sử dụng:** NetworkX, Plotly.

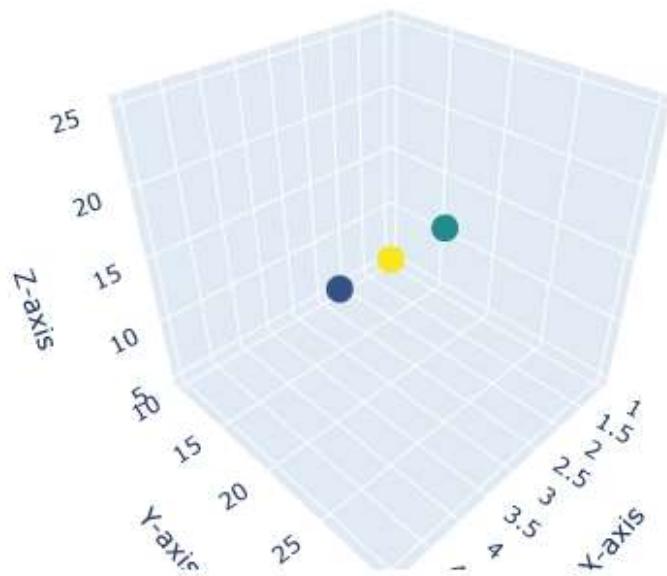
**Ví dụ mã code:**

```
import networkx as nx
import matplotlib.pyplot as plt

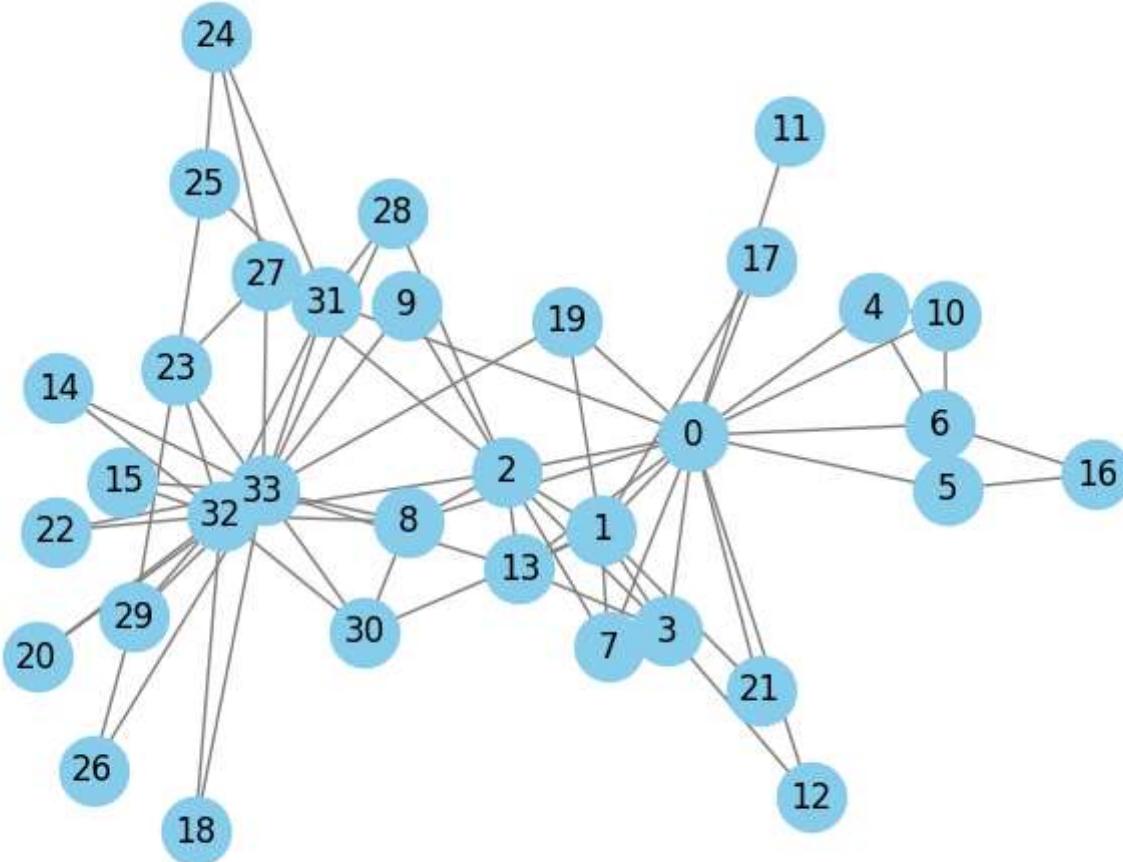
G = nx.karate_club_graph()
nx.draw(G, with_labels=True, node_color='skyblue', node_size=600, edge_color='gray')
plt.title("Network Graph - Karate Club")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")
```

## Biểu đồ 3D



## Network Graph - Karate Club



**Chú thích:**

- nx.draw() dùng để vẽ đồ thị. Các tham số điều chỉnh giao diện của đồ thị:
- with\_labels=True giúp hiển thị các nhãn của các nút.
- node\_color='skyblue' màu sắc của các nút.
- node\_size=600 điều chỉnh kích thước của các nút.
- edge\_color='gray' xác định màu sắc của các cạnh nối giữa các nút.

## 8.5 Tùy chỉnh và Kết hợp Biểu đồ

**Ứng dụng:** Kết hợp các loại biểu đồ khác nhau trong cùng một hình giúp bạn phân tích dữ liệu từ nhiều góc độ. Ví dụ: kết hợp biểu đồ đường và biểu đồ cột để minh họa dữ liệu theo danh mục và theo thời gian

**Ví dụ mã code:**

```

import plotly.graph_objects as go

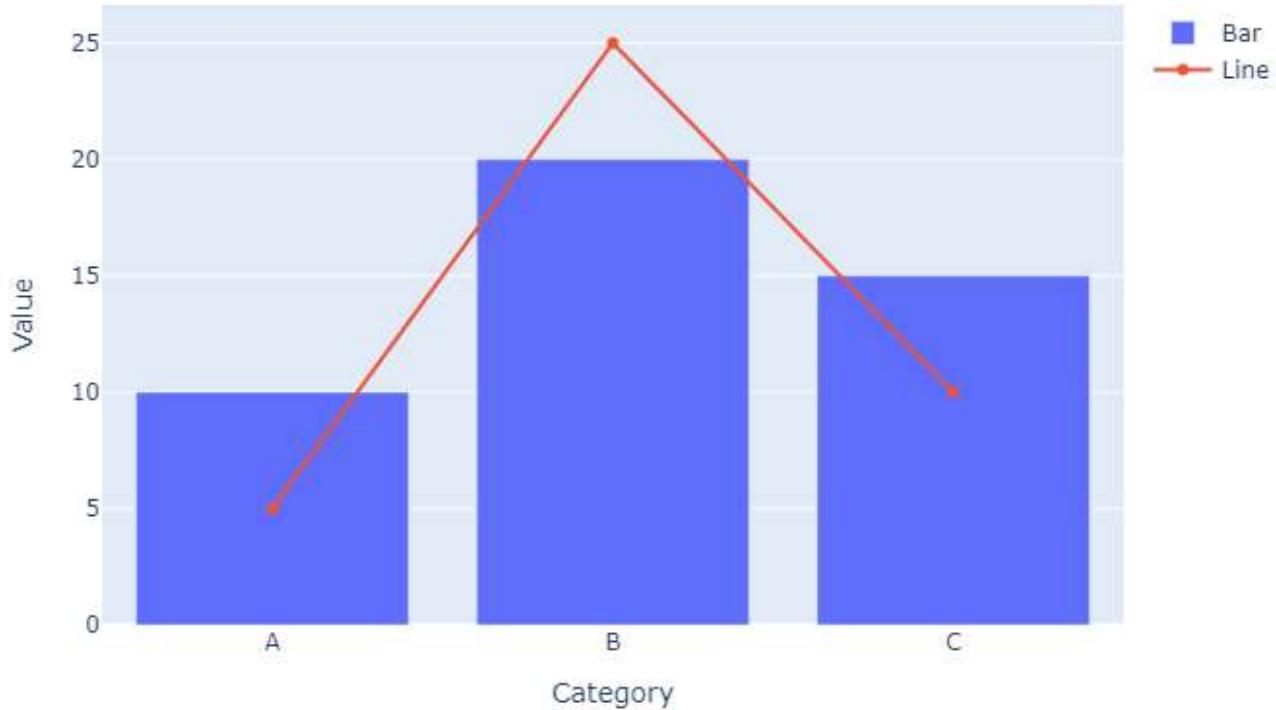
fig = go.Figure()
fig.add_trace(go.Bar(x=["A", "B", "C"], y=[10, 20, 15], name="Bar"))
fig.add_trace(go.Scatter(x=["A", "B", "C"], y=[5, 25, 10], mode="lines+markers", name="Line"))

fig.update_layout(title="Kết hợp Biểu đồ", xaxis_title="Category", yaxis_title="Value")
fig.write_image("chart.png")

from IPython.display import Image
Image("chart.png")

```

Kết hợp Biểu đồ



## Chú thích

- `fig = go.Figure()` tạo một đối tượng đồ thị mới để chứa các yếu tố đồ họa.
- `fig.add_trace(go.Bar(x=["A", "B", "C"], y=[10, 20, 15], name="Bar"))`: Thêm một biểu đồ cột, nơi các danh mục "A", "B", "C" sẽ có giá trị tương ứng là 10, 20 và 15.
- `name="Bar"` giúp gán tên cho biểu đồ cột trong phần chú thích.
- `fig.add_trace(go.Scatter(x=["A", "B", "C"], y=[5, 25, 10], mode="lines+markers", name="Line"))`: Thêm một biểu đồ đường, với các điểm dữ liệu tại "A", "B", "C" có giá trị 5,

25, 10 tương ứng. Các điểm này sẽ được nối với nhau thành một đường, và các điểm cũng được đánh dấu bằng các ký tự (markers).

- mode="lines+markers" kết hợp cả đường và dấu chấm.

## 8.6 Lưu trữ và Chia sẻ Biểu đồ

**Lưu trữ biểu đồ:** Biểu đồ có thể được lưu dưới các định dạng phổ biến như PNG, HTML, PDF hoặc SVG, giúp bạn dễ dàng chia sẻ hoặc nhúng vào các báo cáo, trang web, hoặc tài liệu.

**Chia sẻ trên nền tảng web:**

Plotly cung cấp khả năng chia sẻ biểu đồ trực tuyến, cho phép nhúng biểu đồ vào các website hoặc chia sẻ qua các nền tảng như Dash.

## 8.7 Bài tập Thực hành

**Bài tập 1:** Tạo một biểu đồ 3D minh họa dữ liệu thời gian.

**Bài tập 2:** Tạo heatmap thể hiện mối tương quan giữa các biến số trong bộ dữ liệu.

**Bài tập 3:** Xây dựng một Dashboard nhỏ kết hợp hai biểu đồ (cột và đường).

## 8.8 Tổng kết chương 8

- **Nội dung:** Trong chương 8 này, chúng ta đã khám phá các kỹ thuật trực quan hóa nâng cao, bao gồm biểu đồ 3D, heatmaps và biểu đồ mạng, cùng với cách tùy chỉnh và kết hợp các biểu đồ để phân tích dữ liệu phức tạp. Chương 9 sẽ tiếp tục với việc tích hợp các kỹ thuật này vào các bài tập thực tế, giúp bạn hiểu rõ hơn cách áp dụng trực quan hóa vào các bài toán thực tế.

# Chương 9: Tích hợp Trực quan hóa vào Bài tập

## Contents

- Tích hợp Trực quan hóa vào Bài tập
- 9.1 Giới thiệu về Tích hợp Trực quan hóa
- 9.2 Tích hợp Biểu đồ vào Báo cáo Word hoặc PDF
- 9.3 Tích hợp Biểu đồ vào Jupyter Notebook
- 9.4 Xây dựng Báo cáo Trực quan
- 9.5 Ứng dụng Trực quan hóa trong Thuyết trình
- 9.6 Bài tập Thực hành
- 9.7 Hướng dẫn Tích hợp vào Các Dự án Lớn
- 9.8 Kết luận
- 9.9 Tổng kết chương 9

## Tích hợp Trực quan hóa vào Bài tập

Trong chương này, bạn sẽ học cách sử dụng trực quan hóa dữ liệu để nâng cao chất lượng bài tập và dự án. Chúng ta sẽ tìm hiểu các phương pháp tích hợp trực quan hóa vào các tài liệu báo cáo, Jupyter Notebooks, bài thuyết trình, và các dự án lớn.

## 9.1 Giới thiệu về Tích hợp Trực quan hóa

- **Mục tiêu:** Giúp người học hiểu cách sử dụng trực quan hóa dữ liệu để nâng cao chất lượng bài tập và dự án.

- **Lý thuyết:**
- **Vai trò của trực quan hóa trong trình bày dữ liệu:** Trực quan hóa giúp làm nổi bật các xu hướng và thông tin ẩn trong dữ liệu, giúp người xem dễ dàng tiếp cận và hiểu thông tin. Các loại biểu đồ như cột, đường, scatter plot, và pie chart giúp người đọc nhanh chóng nắm bắt các thông tin quan trọng.
- **Kết hợp biểu đồ vào bài tập thống kê:** Trong các bài tập thống kê hoặc báo cáo phân tích, việc sử dụng biểu đồ giúp làm cho kết quả phân tích trở nên sinh động và dễ hiểu hơn. Biểu đồ cột có thể dùng để so sánh giá trị giữa các nhóm, biểu đồ đường để thể hiện xu hướng theo thời gian, v.v.
- **Lựa chọn loại biểu đồ phù hợp:** Việc chọn loại biểu đồ phải phù hợp với mục tiêu của bài tập, ví dụ biểu đồ cột để so sánh các nhóm, hoặc biểu đồ đường để thể hiện sự thay đổi theo thời gian.

## 9.2 Tích hợp Biểu đồ vào Báo cáo Word hoặc PDF

- **Phương pháp:**
- **Sử dụng biểu đồ đã lưu (PNG, SVG, PDF) trong các báo cáo:** Sau khi tạo biểu đồ, bạn có thể lưu chúng dưới dạng ảnh (PNG, SVG) hoặc PDF, sau đó chèn chúng vào tài liệu Word hoặc PDF để trình bày kết quả phân tích.
- **Tích hợp biểu đồ tương tác (HTML) vào tài liệu:** Bạn có thể nhúng các biểu đồ tương tác (như từ Plotly) vào tài liệu HTML hoặc báo cáo web để người đọc có thể tương tác với chúng.
- **Ví dụ mã:**

```
import plotly.graph_objects as go

fig = go.Figure(data=go.Scatter(x=[1, 2, 3], y=[10, 20, 15], mode='lines+markers'))
fig.write_image("scatter_plot.png") # Lưu ảnh
```

Sau khi lưu biểu đồ dưới dạng ảnh, bạn có thể mở tài liệu Word và chèn ảnh vào báo cáo.

## 9.3 Tích hợp Biểu đồ vào Jupyter Notebook

- **Công cụ sử dụng:** Plotly, Matplotlib, Seaborn.

- **Ví dụ mã:**

```
import plotly.express as px
import pandas as pd
import plotly.io as pio

df = pd.DataFrame({"X": [1, 2, 3], "Y": [10, 20, 15]})
fig = px.bar(df, x="X", y="Y", title="Biểu đồ Cột trong Jupyter Notebook")
pio.write_html(fig, file='plot.html', auto_open=True)
```

Biểu đồ sẽ được hiển thị ngay trong notebook, hỗ trợ trực quan hóa dữ liệu trong quá trình phân tích.

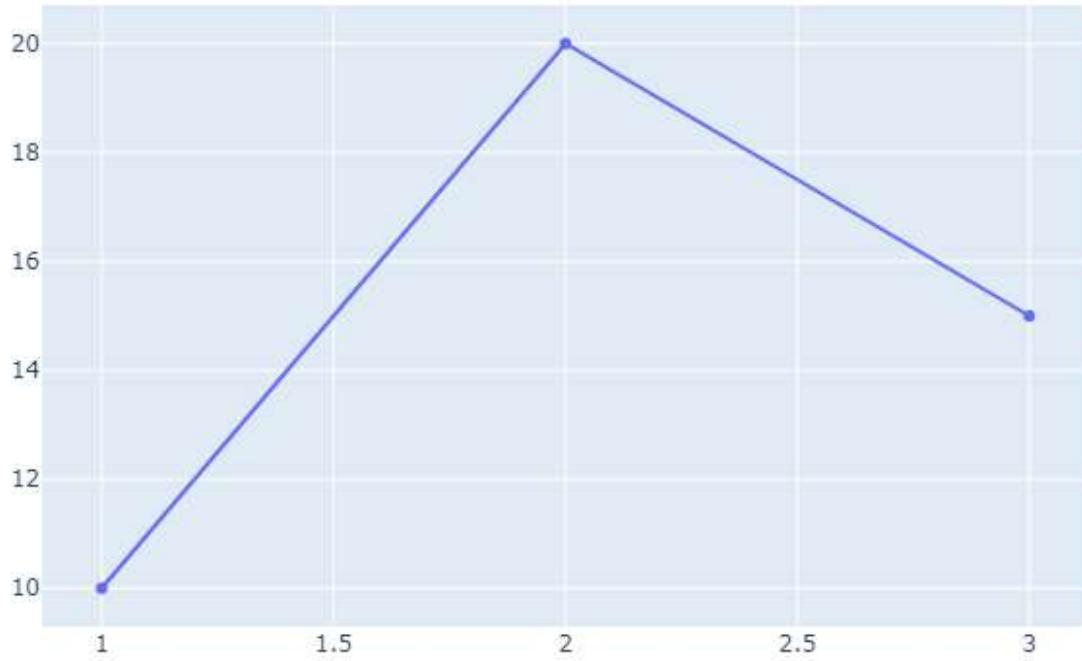
## 9.4 Xây dựng Báo cáo Trực quan

**Ứng dụng:** Tích hợp trực quan hóa vào các báo cáo với Markdown và Jupyter Notebook.

- **Xuất báo cáo dưới dạng HTML hoặc PDF:** Sau khi hoàn thành báo cáo trực quan, bạn có thể xuất nó thành HTML hoặc PDF để chia sẻ hoặc in ấn.
- **Ví dụ Markdown trong Jupyter Notebook:**

### Báo cáo Phân tích

- **Dữ liệu:** Số lượng khách hàng theo tháng.
- **Biểu đồ:**



Markdown cho phép bạn dễ dàng tạo nội dung báo cáo và nhúng hình ảnh của biểu đồ trực quan hóa.

## 9.5 Ứng dụng Trực quan hóa trong Thuyết trình

- **Công cụ sử dụng:** PowerPoint, Google Slides, Jupyter Slide Show.
- **Ví dụ:**
- **Nhúng biểu đồ Plotly vào trang trình bày Google Slides:** Bạn có thể lưu biểu đồ dưới dạng HTML và nhúng vào Google Slides.
- **Tạo slide tương tác từ Jupyter Notebook:**

```
jupyter nbconvert my_notebook.ipynb --to slides --post serve
```

Đây là một cách tuyệt vời để chia sẻ báo cáo hoặc phân tích dữ liệu với nhóm hoặc trong các buổi thuyết trình.

## 9.6 Bài tập Thực hành

**Bài tập 1:** Tạo một báo cáo dữ liệu bán hàng với biểu đồ cột và biểu đồ đường.

- **Yêu cầu:** Dùng dữ liệu bán hàng theo tháng, tạo biểu đồ cột để so sánh doanh thu từng tháng và biểu đồ đường để thể hiện xu hướng theo thời gian.

**Bài tập 2:** Chèn một biểu đồ trực quan hóa vào bài tập thuyết trình (PowerPoint hoặc Google Slides).

- **Yêu cầu:** Sử dụng một biểu đồ của bạn (ví dụ, từ Plotly) và nhúng nó vào một slide PowerPoint hoặc Google Slides.

## 9.7 Hướng dẫn Tích hợp vào Các Dự án Lớn

**Ứng dụng:**

- **Sử dụng trực quan hóa trong báo cáo cuối kỳ, dự án nhóm, hoặc nghiên cứu cá nhân:** Bạn có thể ứng dụng trực quan hóa trong các báo cáo nghiên cứu, bài thuyết trình dự án nhóm hoặc dự án cá nhân.
- **Lập kế hoạch:** Thu thập dữ liệu → Phân tích → Trực quan hóa → Báo cáo.

**Ví dụ thực tế:**

### 1. Phân tích dữ liệu doanh số bán hàng qua các khu vực với biểu đồ heatmap:

- Sử dụng biểu đồ heatmap để trực quan hóa phân bổ doanh thu ở các khu vực khác nhau.

### 2. So sánh hiệu suất giữa các đội nhóm qua biểu đồ cột:

- Tạo biểu đồ cột để so sánh hiệu suất công việc của các đội nhóm trong công ty.

## 9.8 Kết luận

**Tóm tắt các bước tích hợp trực quan hóa vào bài tập:**

- Trực quan hóa giúp người học hiểu rõ hơn về dữ liệu và dễ dàng truyền đạt kết quả phân tích. Các biểu đồ giúp bài tập trở nên sinh động và dễ tiếp cận.

### Đề xuất tài nguyên mở rộng:

- **Sách:** "Python for Data Analysis" của Wes McKinney.
- **Khóa học:** Các khóa học trực tuyến về Plotly, Dash, hoặc báo cáo dữ liệu.
- **Công cụ hỗ trợ:** Matplotlib, Seaborn, Plotly, Dash, Streamlit.

## 9.9 Tổng kết chương 9

**Nội dung:** Trong chương 9 này, bạn đã học cách tích hợp trực quan hóa dữ liệu vào các bài tập thực tế và dự án lớn. Việc sử dụng trực quan hóa sẽ giúp bạn truyền đạt kết quả phân tích hiệu quả hơn và làm cho bài tập trở nên trực quan và dễ hiểu hơn. Chương 10 sẽ giới thiệu các tài nguyên và tài liệu tham khảo để tiếp tục học hỏi và áp dụng các kỹ thuật đã học vào các dự án thực tế.

# Chương 10: Tài nguyên và Tài liệu tham khảo

## Contents

- Tài nguyên và Tài liệu tham khảo
- 10.1 Sách về Dữ liệu hóa học trực quan
- 10.2 Khóa học trực tuyến
- 10.3 Trang web và Blog
- 10.4 Công cụ và Thư viện
- 10.5 Các công cụ hỗ trợ khác
- 10.6 Tổng Kết Chương 10

## Tài nguyên và Tài liệu tham khảo

Chương này cung cấp danh sách các tài nguyên hữu ích để người học có thể tiếp tục nghiên cứu và học hỏi về trực quan hóa dữ liệu, bao gồm sách, khóa học, trang web và các công cụ hỗ trợ. Mục tiêu là giúp người học có thể tự học và nâng cao kỹ năng trực quan hóa dữ liệu của mình sau khi hoàn thành chương trình học.

## 10.1 Sách về Dữ liệu hóa học trực quan

“Python để phân tích dữ liệu” – Wes McKinney

- **Mô tả :** Cuốn sách này cung cấp một cái nhìn tổng quát và chi tiết về cách sử dụng Python để phân tích dữ liệu và trực quan hóa. Tác giả là người phát triển thư viện Pandas, giúp độc giả hiểu cách sử dụng Pandas và các công cụ khác để phân tích và trực quan hóa dữ liệu.

- **Link tham khảo :** [Python for Data Analysis](#)

### “Trực quan hóa dữ liệu bằng Python” – Kyran Dale

- **Mô tả :** Cuốn sách này hướng dẫn cách tạo các biểu đồ và đồ họa đẹp mắt, dễ hiểu bằng Python, tập trung vào công việc sử dụng Matplotlib và Seaborn.
- **Link tham khảo :** [Data Visualization with Python](#)

### “Trình bày trực quan thông tin định lượng” – Edward Tufte

- **Mô tả :** Một cuốn sách kinh điển trong lĩnh vực trực quan hóa dữ liệu. Tufte giới thiệu các cơ sở nguyên lý về cách trình bày và minh họa dữ liệu một cách trực quan và rõ ràng.
- **Link tham khảo :** [Hiển thị trực quan thông tin định lượng](#)

### “Kể chuyện bằng dữ liệu” – Cole Nussbaumer Knaflic

- **Mô tả :** Cuốn sách này giúp người đọc hiểu cách kể chuyện bằng thông tin dữ liệu qua các biểu đồ, từ đó làm cho dữ liệu trở nên dễ hiểu và hấp dẫn hơn.
- **Link tham khảo :** [Kể chuyện bằng dữ liệu](#)

## 10.2 Khóa học trực tuyến

### Coursera: Trực quan hóa dữ liệu bằng Python (Đại học Michigan)

- **Mô tả :** Khóa học này hướng dẫn cách sử dụng thư viện Python để trực tiếp hóa dữ liệu. Nó bao gồm các thư viện như Matplotlib, Seaborn và Plotly.
- **Link tham khảo :** [Data Visualization with Python - Coursera](#)

### Udemy: Python cho trực quan hóa dữ liệu

- **Mô tả :** Khóa học này giúp người học làm quen với các thư viện như Matplotlib, Seaborn và Plotly. Đây là một khóa học tuyệt vời cho những ai muốn thực hiện trực quan hóa dữ liệu trong Python.
- **Link tham khảo :** [Python for Data Visualization - Udemy](#)

### DataCamp: Trực quan hóa dữ liệu với Plotly trong Python

- **Mô tả :** Khóa học này dạy cách sử dụng Plotly để tạo các biểu đồ tương tác, bao gồm các biểu đồ cột, đường và biểu đồ phân tán.

- **Link tham khảo :** [Data Visualization with Plotly - DataCamp](#)

## 10.3 Trang web và Blog

### Tài liệu Plotly

- **Mô tả :** Tài liệu chính thức của Plotly, cung cấp chi tiết hướng dẫn chi tiết về cách sử dụng thư viện này để tạo các biểu đồ tương thích trong Python.
- **Link tham khảo :** [Plotly Documentation](#)

### Tài liệu Seaborn

- **Mô tả :** Tài liệu chính thức của Seaborn, thư viện trực tuyến mạnh mẽ được xây dựng trên Matplotlib.
- **Link tham khảo :** [Seaborn Documentation](#)

### Hướng tới Khoa học dữ liệu (Trung bình)

- **Mô tả :** Một blog nổi bật về dữ liệu và máy học. Các bài viết tại đây cung cấp các hướng dẫn, mẹo và bài viết về trực quan hóa dữ liệu và các kỹ thuật phân tích dữ liệu.
- **Link tham khảo :** [Towards Data Science](#)

### Danh mục trực quan hóa dữ liệu

- **Mô tả :** Một trang web rất hữu ích cho việc lựa chọn loại biểu đồ phù hợp với dữ liệu. Trang web này cung cấp nhiều loại biểu đồ khác nhau cùng ví dụ minh họa.
- **Link tham khảo :** [The Data Visualisation Catalogue](#)

## 10.4 Công cụ và Thư viện

### Cốt truyện

- **Mô tả :** Một thư viện Python mạnh mẽ giúp tạo ra các biểu đồ tương tác, rất hữu ích trong việc trực tiếp hóa dữ liệu phức hợp.
- **Link tham khảo :** [Plotly](#)

### Matplotlib

- **Mô tả :** Một thư viện Python phổ biến để tạo các biểu đồ tĩnh. Dù ít tính năng tương tác hơn Plotly, nhưng Matplotlib vẫn rất mạnh và dễ sử dụng.
- **Link tham khảo :** [Matplotlib](#)

## Seaborn

- **Mô tả :** Một thư viện trực quan hóa dựa trên Matplotlib, cung cấp các chức năng cao cấp và dễ dàng tạo ra các biểu đồ đẹp mắt.
- **Link tham khảo :** [Seaborn](#)

## Dash của Plotly

- **Mô tả :** Dash là một framework mạnh mẽ giúp tạo các ứng dụng web tương tác từ các biểu đồ Plotly.
- **Link tham khảo :** [Dash](#)

## Dòng chảy

- **Mô tả :** Một thư viện Python giúp tạo ra các ứng dụng web tương tác nhanh chóng mà không cần phải biết nhiều về HTML, CSS hay JavaScript.
- **Link tham khảo :** [Streamlit](#)

# 10.5 Các công cụ hỗ trợ khác

## Sổ tay Jupyter

- **Mô tả :** Một công cụ tuyệt vời để tạo báo cáo trực tiếp bằng Python mã hóa. Bạn có thể dễ dàng tạo các biểu đồ và tài liệu mô tả kèm theo trong một môi trường phân tích.
- **Link tham khảo :** [Jupyter Notebooks](#)

## Google Colab

- **Mô tả :** Một dịch vụ miễn phí giúp bạn thực hiện trực tuyến Python notebook. Colab hỗ trợ hầu hết các thư viện như Plotly, Matplotlib, và Seaborn, rất hữu ích cho việc thực hành trực quan hóa dữ liệu.
- **Link tham khảo :** [Google Colab](#)

## Bảng

- **Mô tả :** Một công cụ phân tích và trực tuyến hóa dữ liệu rất mạnh, phù hợp với các nhà phân tích và chuyên gia dữ liệu không muốn viết mã.
- **Link tham khảo :** [Tableau](#)

## 10.6 Tổng Kết Chương 10

**Nội dung:** Chương 10 này đã cung cấp cho bạn những tài nguyên quan trọng để tiếp tục nâng cao kỹ năng trực quan hóa dữ liệu của mình. Những tài nguyên này giúp bạn tiếp cận những công cụ mạnh mẽ và học hỏi từ các chuyên gia trong ngành. Hãy tiếp tục khám phá và thực hành, và đừng sẵn sàng tìm thêm tài liệu và khóa học để nâng cao kiến thức của mình.