**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN**

****

**BÀI TẬP LỚN**

**PHÂN TÍCH XỬ LÝ DỮ LIỆU**

**SÀN THƯƠNG MẠI ĐIỆN TỬ MERCADO LIVRE**

NGÀNH: KHOA HỌC MÁY TÍNH

CHUYÊN NGÀNH: TRÍ TUỆ NHÂN TẠO VÀ KHOA HỌC DỮ LIỆU

SINH VIÊN: **VŨ QUANG PHÚC**

MÃ SINH VIÊN: **12522079**

MÃ LỚP: **124221**

HƯỚNG DẪN: **TS. HOÀNG QUỐC VIỆT**

**HƯNG YÊN – 2024**

**NHẬN XÉT**

**Nhận xét của giáo viên hướng dẫn**

**GIẢNG VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

**LỜI CAM ĐOAN**

Em xin cam đoan bài tập lớn môn Lập trình Python nâng cao có tên là “Phân tích xử lý dữ liệu sàn thương mại điện tử Mercado Livre” là kết quả thực hiện của bản thân em dưới sự hướng dẫn của thầy Hoàng Quốc Việt

Những phần sử dụng tài liệu tham khảo trong bài tập lớn đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong bài tập lớn và chương trình xây dựng được hoàn toàn là kết quả do bản thân em thực hiện.

Nếu vi phạm lời cam đoan này, em xin chịu hoàn toàn trách nhiệm trước khoa và nhà trường.

*Hưng Yên, ngày … tháng*…*năm*…

Sinh viên

…………………………………..

**LỜI CẢM ƠN**

Để có thể hoàn thành bài tập lớn này, lời đầu tiên em xin phép gửi lời cảm ơn tới bộ môn Khoa học máy tính, Khoa Công nghệ thông tin – Trường Đại học Sư phạm Kỹ thuật Hưng Yên đã tạo điều kiện thuận lợi cho em thực hiện bài tập lớn môn học này.

Đặc biệt em xin chân thành cảm ơn thầy Hoàng Quốc Việt đã rất tận tình hướng dẫn, chỉ bảo em trong suốt thời gian thực hiện bài tập lớn vừa qua.

Em cũng xin chân thành cảm ơn tất cả các Thầy, các Cô trong Trường đã tận tình giảng dạy, trang bị cho em những kiến thức cần thiết, quý báu để giúp em thực hiện được bài tập lớn này.

Mặc dù em đã có cố gắng, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hy vọng sẽ nhận được những ý kiến nhận xét, góp ý của các Thầy cô về những kết quả triển khai trong bài tập lớn.

Em xin trân trọng cảm ơn!

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 7](#_Toc165580055)

[1.1 Lý do chọn đề tài 7](#_Toc165580056)

[1.2 Mục tiêu của đề tài 7](#_Toc165580057)

[1.2.1 Mục tiêu tổng quát 7](#_Toc165580058)

[1.2.2 Mục tiêu cụ thể 7](#_Toc165580059)

[1.3 Giới hạn và phạm vi của đề tài 8](#_Toc165580060)

[1.3.1 Đối tượng nghiên cứu 8](#_Toc165580061)

[1.3.2 Phạm vi 8](#_Toc165580062)

[1.4 Nội dung thực hiện 8](#_Toc165580063)

[1.5 Phương pháp tiếp cận 8](#_Toc165580064)

[CHƯƠNG 2: KHẢO SÁT VÀ PHÂN TÍCH THIẾT KẾ HỆ THỐNG 9](#_Toc165580065)

[2.1 Phát biểu yêu cầu 9](#_Toc165580066)

[2.1.1 Danh sách các yêu cầu 10](#_Toc165580067)

[2.1.2 Biểu đồ ca sử dụng 14](#_Toc165580068)

[2.1.3 Phân rã usecase mua hàng 14](#_Toc165580069)

[2.1.4 Phân rã usecase chức năng nhận hàng 16](#_Toc165580070)

[2.1.5 Phân rã usecase Thêm vào giỏ hàng 18](#_Toc165580071)

[2.1.6 Phân rã usecase chức năng tìm kiếm sản phẩm 19](#_Toc165580072)

[2.1.7 Phân rã usecase đánh giá sản phẩm 21](#_Toc165580073)

[2.1.8 Phân rã usecase Đăng kí tài khoản 22](#_Toc165580074)

[2.1.9 Phân rã chức năng cập nhật thông tin khách hàng 24](#_Toc165580075)

[2.1.10 Phân rã usecase đăng kí cửa hàng 26](#_Toc165580076)

[2.1.11 Phân rã usecase Tạo Dashboard 28](#_Toc165580077)

[2.1.12 Phân rã usecase thêm sản phẩm 29](#_Toc165580078)

[2.1.13 Phân rã usecase xóa sản phẩm 31](#_Toc165580079)

[2.1.14 Phân rã usecase cập nhật sản phẩm 33](#_Toc165580080)

[2.1.15 Phân rã usecase xác nhận đơn hàng 34](#_Toc165580081)

[2.1.16 Phân rã usecase kiểm duyệt sản phẩm 36](#_Toc165580082)

[2.1.17 Đặc tả ca usecase thêm nhân viên 38](#_Toc165580083)

[2.2 Biểu đồ lớp thực thể 40](#_Toc165580084)

[2.2.1 Danh sách các lớp đối tượng 40](#_Toc165580085)

[2.2.2 Chi tiết hóa các lớp đối tượng 41](#_Toc165580086)

[2.2.3 Mô hình hóa các lớp đối tượng 51](#_Toc165580087)

[2.3 Thiết kế CSDL 52](#_Toc165580088)

[2.3.1 Lược đồ CSDL 52](#_Toc165580089)

[2.3.2 Mô tả chi tiết từng bảng trong CSDL 52](#_Toc165580090)

[CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG WINDOWS FORMS 62](#_Toc165580091)

[3.1 Triển khai các chức năng nghiệp vụ 62](#_Toc165580092)

[3.1.1 Giao diện các sản phẩm mới nhất 62](#_Toc165580093)

[3.1.2 Giao diện chi tiết sản phẩm 63](#_Toc165580094)

[3.1.3 Chức năng tìm kiếm sàn phẩm 64](#_Toc165580095)

[3.1.4 Giao diện Giỏ hàng 65](#_Toc165580096)

[3.1.5 Giao diện dặt hàng 66](#_Toc165580097)

[3.1.6 Giao diện thông tin khách hàng 67](#_Toc165580098)

[3.1.7 Giao diện các đơn hàng đã đặt 67](#_Toc165580099)

[3.1.8 Giao diện các sản phẩm của shop 68](#_Toc165580100)

[3.1.9 Giao diện cập nhật sản phẩm 68](#_Toc165580101)

[3.1.10 Chức năng quản lý sản phẩm 69](#_Toc165580102)

[3.2 Triển khai các chức năng thống kê, báo cáo 69](#_Toc165580103)

[3.2.1 Giao diện Dashboard 69](#_Toc165580104)

[3.3 Kiểm thử và triển khai ứng dụng 70](#_Toc165580105)

[3.3.1 Kiểm thử 70](#_Toc165580106)

[3.3.2 Đóng gói ứng dụng 70](#_Toc165580107)

[3.3.3 Triển khai ứng dụng 70](#_Toc165580108)

[KẾT LUẬN 71](#_Toc165580109)

[TÀI LIỆU THAM KHẢO 73](#_Toc165580110)

**DANH MỤC HÌNH ẢNH**

No table of figures entries found.

**DANH MỤC BẢNG BIỂU**

[Bảng 1.1 Danh sách các features trong dataset 7](#_Toc168582787)

# GIỚI THIỆU BÀI TOÁN

## Bài toán

Hiện nay các sàn thương mại điện tử đang là xu thế của toàn cầu về quá trình mua bán giữa người tiêu dùng và doanh nghiệp. Nó không những công nghệ hóa các giai đoạn, thao tác của quá trình vận chuyển hàng hóa đến tay người tiêu dùng. Là 1 trong những mắt xích không thể thiếu trong thời đại công nghệ phát triển như hiện nay, việc mua bán đã trở lên dễ dàng hơn bao giờ hết chỉ với 1 chiếc smartphone có kết nối internet là chúng ta đã có thể mua được hàng hóa, dịch vụ ở khắp mọi nơi trên thế giới. Và thị trường Brazil là 1 trong số các thị trường đã và đang ứng dụng các công nghệ của sàn thương mại điện tử rất tốt để nâng cao trải nghiệm của người dùng trong thời đại cách mạng AI hiện nay và Olist Store là 1 trong những sàn thương mại điện tử có tiềm năng lớn ở Brazil thời điểm hiện tại, luôn là top đầu trong ngành thương mại điện tử Brazil.

Bằng cách phân tích dữ liệu của Olist Store, chúng ta có thể có được 1 cái nhìn tổng quan về xu hướng mua sắm, các danh mục sản phẩm được ưa chuộng của người tiêu dùng Brazil. Các vấn đề mà thương mại điện tử tại Brazil nói chung và sàn thương mại Olist Store nói riêng, bộ dữ liệu sẽ cho chúng ra 1 góc nhìn tổng quan nhất về tình hình kinh tế của Brazil trong lĩnh vực này

Cho nên vấn đề đặt ra là chúng ta cần phải hiểu được bộ dữ liệu này và từ đó đưa ra được những thông tin đáng chú ý, cốt lõi mang tính chiến lược thông qua các ví dụ trực quan bằng biểu đồ, hình vẽ,.. bên cạnh đó thì việc chỉ ra các sự thiếu sót, không đồng nhất, mất mát dữ liệu .. cũng 1 phần đưa ra được rất nhiều thôn tin có ý nghĩa cho quá trình phân tích. Bộ dữ liệu sẽ giúp chúng ta hiểu thêm về xu hướng, mức độ gia tăng về giá cả lẫn khác hàng, sản phẩm qua các năm,.. Từ đó đưa ra được những đánh giá khách quan và giải pháp cũng như vấn đề mà doanh nghiệp đang gặp phải(nếu có). Lược bỏ các thông số, nội dung không có ý nghĩa nhiều, tập trung vào các nội dung trọng tâm, mang tích chiến lược từ đó có thể đưa ra giải pháp cho doanh nghiệp hoặc vấn đề mà chúng ta đang băn khoăn.

Để có thể hiểu và làm tốt quá trình phân tích này chúng ta cần các kiến thức liên quan đến nghiệp vụ của 1 sàn thương mại điện tử, các công cụ phục vụ cho việc phân tích dữ liệu như IPython, jupyter notebook or jupyter lab, pandas, seaborn,...

## Trình bày dữ liệu của bài toán

Dữ liệu của bài toán đang ở link sau: Link Github

Bộ dữ liệu này do 1 sàn thương mại điện tử Olist Store bên Brazil đã công bố vào 3 năm trước, bộ dữ liệu ghi lại lịch sử giao dịch, đặt hàng của các khách hàng, người bán đã có giao dịch xáy ra trong khoảng thời gian từ 10/2016 - 9/2018.

Mỗi 1 bản ghi đại diện do 1 chi tiết giao dịch trên sàn lưu thông tin về người bán, người mua, sản phẩm, giá cả và số lượng. Các thông tin này sẽ tồn tại ở dạng ID là những mã đã được hash có cùng độ dài 32 kí tự. Do đó sẽ có rất nhiều các bản ghi có trùng mã đơn hàng, các bản ghi đó là các chi tiết đơn hàng thuộc cùng 1 đơn hàng.

Các dữ liệu liên quan đến thông tin cá nhân của các khách hàng và cả người mua, sản phẩm đều không được công bố, vì bộ dữ liệu này sẽ tập trung miêu tả giao dịch nên sẽ công bố các thông tin các quan trọng hơn như ngày giờ giao dịch, số tiền, danh mục hàng hóa, thành phố của sàn phẩm, khách hàng, phí vận chuyển, đánh giá,..

Do được chính Olist Store công bố nên dữ liệu gần như ở dạng thô và chưa qua xử lý, bộ dữ liệu thô này gồm tổng cộng 9 dataset nhỏ, liên kết với nhau thông qua khóa chính và khó ngoại, tương tự như các bảng trong database. Bộ dữ liệu này bao gồm không dưới 30 cột và có nhiều cột khá khó để phân tích. Để quá trình phân tích trở nên dễ hiểu thì bộ datasets này cần phải được gộp lại thành 1 dataset lớn hơn và đồng nhất hơn. Quá trình gộp có thể được tìm thấy [tại đây.](https://github.com/vuquangphucliar/Olist_store_analysis) Vì quá trinh gộp dữ liệu khá đơn giản, ta chỉ cần gộp chúng lại dựa trên db\_schema đã được cung cấo, loại bỏ các cột không cần thiết, và hạn chế xử lý dữ liệu ở bước này. Phần xử lý dữ liệu sẽ được trình bày sau.

Phần mô tả dữ liệu bên dưới sẽ sơ lược qua ngắn gọn về các features có trong bộ dữ liệu đã được xử lý. Chi tiết mô tả về bộ dữ liệu có thể được tìm thấy tại đây.

* Bộ dữ liệu này gồm \_\_\_ cột và \_\_\_\_ hàng
* Gồm \_\_ biến định tính và \_\_ biến định lượng:

Các features có trong dataset:

Bảng 1.1 Danh sách các features trong dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Feature** | **Not Null** | **Type** | **Desctiption** |
| 1 | Order\_id | 97254 | object | Id của đơn hàng |
| 2 | Customer\_id | 97254 | object | Id của khách hàng |
| 3 | Quantity | 97254 | int | Số lượng sản phẩm đã mua |
| 4 | Price\_MRP | 97254 | float(64) | Giá bán lẻ |
| 5 | Payment | 97254 | float(64 | Giá thanh toán |
| 6 | Timestamp | 97254 | object | Thời gian mua |
| 7 | Rating | 97254 | int | Đánh giá |
| 8 | Category | 97254 | object | Danh mục sản phẩm |
| 9 | Product\_id | 97254 | object | Id sản phẩm |
| 10 | Payment\_type | 97254 | object | Phương thức thanh toán |
| 11 | Order\_status | 97254 | object | Trạng thái đơn hàng |
| 12 | Weight | 97254 | int | Trọng lượng sàn phẩm |
| 13 | Length | 97254 | int | Độ dài sàn phẩm |
| 14 | Width | 97254 | int | Chiều rộng sàn phẩm |
| 15 | Height | 97254 | int | Chiều cao sàn phẩm |
| 16 | Customer\_state | 97254 | object | Bang của khách hàng |
| 17 | Seller\_id | 97254 | object | Id người bán |
| 18 | seller\_state | 97254 | object | Bang của người bán |

Thống kê cơ bản các biến định lượng trong dataset:

Mô tả cơ bản các biến định tính trong dataset:

* rating: chứa các số từ 1-5 đại diện cho số sao khách hàng đánh giá
* category: chứa tên của 71 danh mục trên sàn được giao dịch
* payment\_type: gồm 5 phương thức thanh toán, credit card, debit card,..
* order\_status: gồm 7 trạng thái của đơn hàng invoiced, delivered, processing,..
* customer\_state: gồm các tên viết tắt của 27 bang tại Brazil
* seller\_state: tương tự như customer\_state( tên đầy đủ được gắn tại link trên)
* payment\_installments: gồm các số từ 1-24 đại diện cho 12 tháng.

## Tiền xử lý dữ liệu

## Thống kê dữ liệu

## Trực quan hóa dữ liệu

# CƠ SỞ LÝ THUYẾT

## Numpy

NumPy (Numeric Python) là một thư viện Python mã nguồn mở được sử dụng trong hầu hết các lĩnh vực khoa học và kỹ thuật. Đây là tiêu chuẩn chung để làm việc với dữ liệu số bằng Python và là tiêu chuẩn cốt lõi của hệ sinh thái Python và PyData. Numpy hỗ trợ mạnh mẽ việc tính toán với ma trận, vector và các hàm đại số tuyến tính cơ bản. Điều này làm cho nó trở thành một công cụ quan trọng trong việc hiện thực các thuật toán Machine Learning.Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng “core Python” đơn thuần.

1. **Cài đặt thư viện Numpy**

- Mở Command Prompt và gõ lệnh: pip install numpy

1. **Các thao tác với Numpy**

1. Khai báo thư viện

*import numpy as np*

2. Khởi tạo mảng

- Khởi tạo mảng một chiều với kiểu dữ liệu các phần tử là Integer

*arr = np.array([1,3,4,5,6], dtype = int)*

- Khởi tạo mảng một chiều với kiểu dữ liệu mặc định

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

*print(arr)*

OUTPUT:

[1 3 4 5 6]

- Khởi tạo mảng hai chiều

*arr1 = np.array([(4,5,6), (1,2,3)], dtype = int)*

*print(arr1)*

OUTPUT:

[[4 5 6]

[1 2 3]]

- Khởi tạo mảng ba chiều

*arr2 = np.array(([(2,4,0,6), (4,7,5,6)],*

*[(0,3,2,1), (9,4,5,6)],*

*[(5,8,6,4), (1,4,6,8)]), dtype = int)*

*print(arr2)*

OUTPUT:

[[[2 4 0 6]

[4 7 5 6]]

[[0 3 2 1]

[9 4 5 6]]

[[5 8 6 4]

[1 4 6 8]]]

- Khởi tạo với các hàm có sẵn

*np.zeros((3,4), dtype = int): Tạo mảng hai chiều các phần tử 0 với kích thước 3x4.*

*np.ones((2,3,4), dtype = int): mảng 3 chiều các phần tử 1 với kích thước 2x3x4.*

*np.arange(1,7,2): Tạo mảng với các phần tử từ 1 - 6 với bước nhảy là 2.*

*np.full((2,3),5): Tạo mảng 2 chiều các phần tử 5 với kích thước 2x3.*

*np.eye(4, dtype=int): Tạo ma trận đơn vị với kích thước là 4x4.*

*np.random.random((2,3)): Ma trận các phần tử ngẫu nhiên với kích thước 2x3.*

1. **Thao tác với mảng**

* dtype: Kiểu dữ liệu của phần tử trong mảng.
* shape: Kích thước của mảng.
* size: Số phần tử trong mảng.
* ndim: Số chiều của mảng.

*print("Kiểu dữ liệu của phần tử trong mảng:", arr2.dtype)*

*print("Kích thước của mảng:", arr2.shape)*

*print("Số phần tử trong mảng:", arr2.size)*

*print("Số chiều của mảng:", arr2.ndim)*

Output:

Kiểu dữ liệu của phần tử trong mảng: int32

Kích thước của mảng: (3, 2, 4)

Số phần tử trong mảng: 24

Số chiều của mảng: 3

* Truy cập phần tử trong mảng

Các phần tử trong mảng được đánh số từ 0 trở đi

arr[i]: Truy cập tới phần tử thứ i của mảng 1 chiều.

arr1[i,j]: Truy cập tới phần tử hàng i, cột j của mảng 2 chiều.

arr2[n,i,j]: Truy cập tới phần tử chiều n, hàng i, cột j của mảng 3 chiều.

arr[a:b]: Truy cập tới các phần tử từ a đến b-1 trong mảng 1 chiều.

arr1[:,:i]: Truy cập tới phần tử từ cột 0 đến cột i-1

*print("arr[2]=", arr[2])*

*print("arr1[1:2]=", arr1[1,2])*

*print("arr2[1,2,3]=", arr2[1,1,3])*

*print("arr[0:3]=", arr[0:3])*

*print("arr1[:,:1]=", arr1[:,:2])​*

Output

arr[2]= 4

arr1[1:2]= 3

arr2[1,2,3]= 6

arr[0:3]= [1 3 4]

arr1[:,:1]= [[4 5]

[1 2]]

- Các hàm thống kê

arr.max() hoặc np.max(arr): Lấy giá trị lớn nhất của mảng arr.

arr.min() hoặc np.min(arr): Lấy giá trị nhỏ nhất của mảng arr.

arr.sum() hoặc np.sum(arr): Tổng tất cả các phần tử trong mảng arr.

arr.mean() hoặc np.mean(arr):

np.median(arr): Trả về giá trị trung vị của mảng arr.

*print("Giá trị lớn nhất của mảng arr là:", np.max(arr))*

*print("Giá trị nhỏ nhất của mảng arr là:", np.min(arr))*

*print("Tổng tất cả các phần tử của mảng arr là:", np.sum(arr))*

*print("Trung bình cộng tất cả các phần tử của mảng arr là:", np.mean(arr))*

*print("Giá trị trung vị của mảng arr là:", np.median(arr))*

Output:

Giá trị lớn nhất của mảng arr là: 6

Giá trị nhỏ nhất của mảng arr là: 1

Tổng tất cả các phần tử của mảng arr là: 19

Trung bình cộng tất cả các phần tử của mảng arr là: 3.8

Giá trị trung vị của mảng arr là: 4.0

## Pandas

Pandas là một thư viện Python cung cấp các cấu trúc dữ liệu nhanh, mạnh mẽ, linh hoạt và mang hàm ý. Tên thư viện được bắt nguồn từ “panel data” (bảng dữ liệu). Pandas được thiết kế để làm việc dễ dàng và trực quan với dữ liệu có cấu trúc (dạng bảng, đa chiều, có tiềm năng không đồng nhất) và dữ liệu chuỗi thời gian1. Thư viện này thường được sử dụng để thao tác, phân tích và dọn dẹp dữ liệu. Pandas cung cấp nhiều cấu trúc dữ liệu và phép tính hỗ trợ thao tác dữ liệu số và dữ liệu thời gian2. Nó là một công cụ toàn diện cho xử lý dữ liệu trong Python, phù hợp cho nhiều loại tác vụ phân tích và quản lý dữ liệu. Thư viện pandas trong python là một thư viện mã nguồn mở, hỗ trợ đắc lực trong thao tác dữ liệu. Đây cũng là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ của ngôn ngữ lập trình python. Thư viện này được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu. Thư viện này sử dụng một cấu trúc dữ liệu riêng là Dataframe. Pandas cung cấp rất nhiều chức năng xử lý và làm việc trên cấu trúc dữ liệu này. Chính sự linh hoạt và hiệu quả đã khiến cho pandas được sử dụng rộng rãi.

**Vì sao bạn nên chọn pandas?**

* Pandas rất phù hợp với nhiều loại dữ liệu khác nhau:
* Dữ liệu dạng bảng với các cột được nhập không đồng nhất, như trong bảng SQL hoặc bảng tính Excel.
* Dữ liệu chuỗi thời gian theo thứ tự và không có thứ tự (không nhất thiết phải có tần số cố định).
* Dữ liệu ma trận tùy ý (được nhập đồng nhất hoặc không đồng nhất) với nhãn hàng và cột.
* Bất kỳ hình thức khác của các bộ dữ liệu quan sát / thống kê. Dữ liệu thực sự không cần phải được dán nhãn vào cấu trúc dữ liệu pandas.
* Pandas được xây dựng dựa trên NumPy. Hai cấu trúc dữ liệu chính của pandas là Series (1 chiều) và DataFrame (2 chiều) xử lý được phần lớn các trường hợp điển hình trong tài chính, thống kê, khoa học xã hội và nhiều lĩnh vực kỹ thuật.

**Ưu điểm của pandas:**

* Dễ dàng xử lý dữ liệu mất mát, được biểu thị dưới dạng NaN, trong dữ liệu dấu phẩy động cũng như dấu phẩy tĩnh theo ý người dùng mong muốn: bỏ qua hoặc chuyển sang 0
* Khả năng thay đổi kích thước: các cột có thể được chèn và xóa khỏi DataFrame và các đối tượng chiều cao hơn
* Căn chỉnh dữ liệu tự động và rõ ràng: các đối tượng có thể được căn chỉnh rõ ràng với một bộ nhãn hoặc người dùng chỉ cần bỏ qua các nhãn và để Series, DataFrame, v.v. tự động căn chỉnh dữ liệu cho bạn trong các tính toán
* Chức năng group by mạnh mẽ, linh hoạt để thực hiện các hoạt động kết hợp phân tách áp dụng trên các tập dữ liệu, cho cả dữ liệu tổng hợp và chuyển đổi
* Dễ dàng chuyển đổi dữ liệu rời rạc (ragged), chỉ mục khác nhau (differently-indexed) trong các cấu trúc dữ liệu khác của Python và NumPy thành các đối tượng DataFrame
* Cắt lát (slicing) thông minh dựa trên nhãn, lập chỉ mục ưa thích (fancy indexing) và tập hợp lại (subsetting) các tập dữ liệu lớn
* Gộp (merging) và nối (joining) các tập dữ liệu trực quan
* Linh hoạt trong định hình lại (reshaping) và xoay (pivoting) các tập dữ liệu
* Dán nhãn phân cấp (hierarchical) của các trục (có thể có nhiều nhãn trên mỗi đánh dấu)
* Các công cụ IO mạnh mẽ để tải dữ liệu từ các tệp phẳng (flat file) như CSV và delimited, tệp Excel, cơ sở dữ liệu và lưu / tải dữ liệu từ định dạng HDF5 cực nhanh
* Chức năng theo chuỗi thời gian (time series) cụ thể: tạo phạm vi ngày và chuyển đổi tần số, thống kê cửa sổ di chuyển, dịch chuyển ngày và độ trễ.
* Tích hợp tốt với các thư viện khác của python như SciPy, Matplotlib, Plotly, v.v.
* Hiệu suất tốt

**Cài đặt thư viện Pandas**

* Sử dụng pip và gõ lệnh: p*ip install pandas*
* Hoặc bằng Anaconda, dùng lệnh*: conda install pandas*
* Khai báo thư viện Pandas: *import pandas as pd*

(Bạn không nên thay đổi từ pd bằng từ khác vì các tài liệu hướng dẫn đều ngầm quy ước như vậy.)

**Thao tác với cấu trúc dữ liệu cơ bản**

Pandas có 2 cấu trúc dữ liệu cơ bản là:

* Series (1 chiều)
* DataFrame (2 chiều).

Panel (3 chiều) từng là một cấu trúc dữ liệu trong pandas trước khi bị gỡ bỏ từ phiên bản 0.25. Bạn có thể tham khảo về panel ở phiên bản 0.24.

**1. Series**

*Series([data, index, dtype, name, copy, . . . ])*

Series là mảng một chiều giống như mảng Numpy, hay như một cột của một bảng, nhưng nó bao gồm thêm một bảng đánh label. Series có thể được khởi tạo thông qua NumPy, kiểu Dict hoặc các dữ liệu vô hướng bình thường. Series có nhiều thuộc tính như index, array, values, dtype, v.v. Bạn có thể thực hiện chuyển đổi Series sang dạng dtype xác định, tạo bảng copy, trả về dạng bool của một thành phần, chuyển Series từ DatetimeIndex sang PeriodIndex, v.v.

- Một số ví dụ về thao tác với Series:

Tạo Series

Ví dụ 1: Không truyền index

*import pandas as pd*

*s = pd.Series([0,1,2,3])*

*print(s)*

Output:

0 0

1 1

2 2

3 3

dtype: int64

pandas sẽ mặc định truyền indextừ 0 đến len(data)-1.

Ví dụ 2: Có truyền index

*import pandas as pd*

*s = pd.Series([0,1,2,3], index=["a","b","c","d"])*

*print(s)*

Output:

a 0

b 1

c 2

d 3

dtype: int64

Ví dụ 3: Tạo Series từ dict

*import pandas as pd*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*print(ser)*

Output:

a -1.3

c NaN

b 11.7

d 2.0

e NaN

f 10.0

dtype: float64

Chúng ta tạo dict có index a, b, d, f, g. Sau đó tạo Series từ dict data này nhưng các index c và e không có trong dict nên dữ liệu tại các index này bị thiếu (missing data). pandas hiển thị NaN để báo các dữ liệu này bị trống.

Ví dụ 4: Tạo Series từ Scalar

Nếu dữ liệu là một giá trị scalar, index phải được cung cấp. Giá trị sẽ được lặp lại để phù hợp với độ dài của index.

*import pandas as pd*

*ser = pd.Series(5, index=[1, 2, 3, 4, 5])*

*print(ser)*

Output:

1 5

2 5

3 5

4 5

5 5

dtype: int64

Truy cập dữ liệu từ Series với index và vị trí

Truy cập dữ liệu của Series tương tự với ndarray trong NumPy.

Ví dụ 5: Lấy dữ liệu tại index cụ thể

*import pandas as pd*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*print(ser['d'])*

*print(ser['c'])*

Output:

2.0

Nan

Ví dụ 6: Lấy dữ liệu từ đầu đến vị trí index cụ thể

*import pandas as pd*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*print(ser[:'d'])*

Output:

a -1.3

c NaN

b 11.7

d 2.0

dtype: float64

Ví dụ 7: Lấy dữ liệu theo vị trí: 2 dữ liệu đầu

*import pandas as pd*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*print(ser[:2])*

Output:

a -1.3

c NaN

dtype: float64

Ví dụ 8: Lấy 3 dữ liệu cuối

*import pandas as pd*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*print(ser[-3:])*

Output:

d 2.0

e NaN

f 10.0

dtype: float64

- Chuyển đổi sang dạng khác

Ví dụ 9: Lấy dạng array của Series bằng numpy.asarray

*import pandas as pd*

*import numpy as np*

*data = {'a' : -1.3, 'b' : 11.7, 'd' : 2.0, 'f': 10, 'g': 5}*

*ser = pd.Series(data,index=['a','c','b','d','e','f'])*

*a = np.asarray(ser)*

*print(a)*

Output:

[-1.3 nan 11.7 2. nan 10. ]

2. DataFrame

*DataFrame([data, index, columns, dtype, copy])*

Dataframe là cấu trúc dữ liệu được gắn nhãn hai chiều với các cột và hàng như bảng tính (spreadsheet) hoặc bảng (table). Giống như Series, DataFrame có thể chứa bất kỳ loại dữ liệu nào. Một điều quan trọng cần làm nổi bật là tất cả các cột trong khung dữ liệu là series Pandas. Vì vậy, một DataFrame là sự kết hợp của nhiều Series đóng vai trò như các cột! DataFrame được sử dụng rộng rãi và là một trong những cấu trúc dữ liệu quan trọng nhất.

Chúng ta hãy bắt đầu với tạo DataFrame

Ví dụ 1: Tạo DataFrame từ dict các Series 1

*import pandas as pd*

*# tạo dict từ các series*

*s = {'một': pd.Series([1., 2., 3., 5.], index=['a', 'b', 'c', 'e']),*

*'hai': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd'])}*

*# tại DataFrame từ dict*

*df = pd.DataFrame(s)*

*print(df)*

Output:

một hai

a 1.0 1.0

b 2.0 2.0

c 3.0 3.0

d NaN 4.0

e 5.0 NaN

Ví dụ 2: Tạo DataFrame từ dict các Series 2

*import pandas as pd*

*# tạo các series*

*s = {'một': pd.Series([1., 2., 3., 5.], index=['a', 'b', 'c', 'e']),*

*'hai': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd'])}*

*# tạo DataFrame từ dict theo các index được chọn*

*df = pd.DataFrame(s, index=['a','c','d'])*

*print(df)*

Output:

một hai

a 1.0 1.0

c 3.0 3.0

d NaN 4.0

- Các thao tác chọn, thêm, xóa cột

Ví dụ 3: Chọn cột (column selection)

*import pandas as pd*

*s = {'một': pd.Series([1., 2., 3., 5.], index=['a', 'b', 'c', 'e']),*

*'hai': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd']),*

*'ba': pd.Series([9., -1.3, 3.5, 41.1], index=['a', 'b', 'c', 'd'])}*

*df = pd.DataFrame(s)*

*# chọn cột hai*

*df\_hai = df['hai']*

*print(df\_hai)*

Output:

a 1.0

b 2.0

c 3.0

d 4.0

e NaN

Name: hai, dtype: float64

Ví dụ 4: Một số cách thêm cột (column addition)

*import pandas as pd*

*s = {'một': pd.Series([1., 2., 3., 5.], index=['a', 'b', 'c', 'e']),*

*'hai': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd']),*

*'ba': pd.Series([9., -1.3, 3.5, 41.1], index=['a', 'b', 'c', 'd'])}*

*df = pd.DataFrame(s)*

*# thêm cột bốn với giá trị mỗi ô theo công thức*

*df['bốn'] = df['hai'] - df['ba']*

*# thêm cột với giá trị vô hướng (scalar value)*

*df['Chuẩn'] = 'OK'*

*# thêm cột không cùng số lượng index với DataFrame*

*df['Khác'] = df['hai'][:3]*

*# thêm cột True/False theo điều kiện*

*df['KT'] = df['một'] == 3.0*

*# dùng hàm insert. Cột "chèn" bên dưới sẽ ở vị trí 2 (tính từ 0), có giá trị bằng cột một*

*df.insert(2, 'chèn', df['một'])*

*print(df)*

Output:

một hai chèn ba bốn Chuẩn Khác KT

a 1.0 1.0 1.0 9.0 -8.0 OK 1.0 False

b 2.0 2.0 2.0 -1.3 3.3 OK 2.0 False

c 3.0 3.0 3.0 3.5 -0.5 OK 3.0 True

d NaN 4.0 NaN 41.1 -37.1 OK NaN False

e 5.0 NaN 5.0 NaN NaN OK NaN False

Ví dụ 5: Xóa cột (column deletion)

Có thể xóa cột bằng lệnh def hoặc hàm pop

*import pandas as pd*

*s = {'một': pd.Series([1., 2., 3., 5.], index=['a', 'b', 'c', 'e']),*

*'hai': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd']),*

*'ba': pd.Series([9., -1.3, 3.5, 41.1], index=['a', 'b', 'c', 'd'])}*

*df = pd.DataFrame(s)*

*# xóa cột hai*

*del df['hai']*

*# pop cột ba với dict tv\_ba*

*tv\_ba = df.pop('ba')*

*print( df)*

Output:

một

a 1.0

b 2.0

c 3.0

d NaN

e 5.0

## Matplotlib

Matplotlib là một thư viện vẽ đồ thị cho ngôn ngữ lập trình Python và được mở rộng từ thư viện toán học số học NumPy. Nó cung cấp một API hướng đối tượng để nhúng các biểu đồ vào ứng dụng sử dụng các toolkit GUI đa năng như Tkinter, wxPython, Qt hoặc GTK. Matplotlib cho phép bạn tạo và hiển thị các biểu đồ, hình ảnh và các hình vẽ khác, rất hữu ích cho việc trực quan hóa dữ liệu trong Python. Một phần quan trọng của Matplotlib là Pyplot, một module cung cấp các hàm đơn giản để thêm các thành phần plot như đường, hình ảnh, văn bản và nhiều hơn nữa vào các khung vẽ (axes).

Matplotlib là một trong những thư viện Python phổ biến nhất được sử dụng để trực quan hóa dữ liệu. Nó là một thư viện đa nền tảng để tạo các đồ thị 2D từ dữ liệu trong các mảng. Matplotlib được viết bằng Python và sử dụng NumPy, phần mở rộng toán học của Python. Nó cung cấp một API hướng đối tượng giúp nhúng các plot trong các ứng dụng và sử dụng bộ công cụ GUI Python như PyQt, WxPythonotTkinter. Ngoài ra có thể được sử dụng trong Python và IPython shell, Jupyter Notebook và các máy chủ web.

Matplotlib có giao diện được đặt tên là Pylab, được thiết kế giống với MATLAB - ngôn ngữ lập trình độc quyền được phát triển bởi MathWorks. Matplotlib cùng với NumPy có thể được coi là mã nguồn mở tương đương với MATLAB.

Matplotlib ban đầu được viết bởi John D. Hunter vào năm 2003. Phiên bản ổn định hiện tại là 2.2.0 được phát hành vào tháng 1 năm 2018.

**1. Matplotlib dùng để làm gì? :**

Để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu của bạn và Matplotlib là một trong những giải pháp như vậy cho người dùng Python. Nó là một thư viện vẽ đồ thị rất mạnh mẽ hữu ích cho những người làm việc với Python và NumPy. Module được sử dụng nhiều nhất của Matplotib là Pyplot cung cấp giao diện như MATLAB nhưng thay vào đó, nó sử dụng Python và nó là nguồn mở.

Để cài đặt Matplotlib nếu bạn có Anaconda chỉ cần gõ conda install matplotlib hoặc sử dụng tools pip pip install matplotlib

**2. Khái niệm chung**

Một Matplotlib figure có thể được phân loại thành nhiều phần như dưới đây:

Figure: Như một cái cửa sổ chứa tất cả những gì bạn sẽ vẽ trên đó.

Axes: Thành phần chính của một figure là các axes (những khung nhỏ hơn để vẽ hình lên đó). Một figure có thể chứa một hoặc nhiều axes. Nói cách khác, figure chỉ là khung chứa, chính các axes mới thật sự là nơi các hình vẽ được vẽ lên.

Axis: Chúng là dòng số giống như các đối tượng và đảm nhiệm việc tạo các giới hạn biểu đồ.

Artist: Mọi thứ mà bạn có thể nhìn thấy trên figure là một artist như Text objects, Line2D objects, collection objects. Hầu hết các Artists được gắn với Axes.

**3. Ưu điểm :**

Matplotlib là một thư viện giống như GNUplot. Ưu điểm chính so với GNUplot đó là Matplotlib là một module của Python. Do mức độ phổ biến của python ngày càng tăng, nên matplotlib cũng nhận được sự quan tâm tương tự.

Một lý do khác cho sự hấp dẫn của Matplotlib nằm ở chỗ nó được xem là một sự lựa chọn hoàn hảo thay thế cho MATLAB, nếu nó được sử dụng kết hợp với Numpy và Scipy. Trong khi MATLAB đắt đỏ và mã nguồn đóng, Matplotlib lại miễn phí và mã nguồn mở. Nó cũng là ngôn ngữ hướng đối tượng. Hơn nữa, nó có thể được sử dụng với bộ công cụ GUI mục đích chung như wxPython, Qt, và GTK+. Cũng có một thủ tục "pylab", được thiết kế để giống với MATLAB. Điều này có thể làm cho những người quen sử dụng MATLAB dễ dàng chuyển sang dùng matplotlib.

Matplotlib có thể được sử dụng để tạo ra những figures đủ chất lượng cho một loạt các định dạng hardcopy và môi trường tương tác trên nền tảng.

Một đặc điểm khác của matplotlib là tốc độ lĩnh hội, có nghĩa là người dùng thường đạt được tiến bộ nhanh chóng sau khi bắt đầu. Các trang web chính thức có thể nói những điều sau đây: "matplotlib cố gắng làm những điều khó khăn, phức tạp trở lên dễ dàng nhất có thể. Bạn có thể tạo ra các hình vẽ, histograms, phổ, biểu đồ thanh, errorcharts, scatterplots, vv, với chỉ một vài dòng mã."

Matplotlib có một số interfaces để tương tác với thư viện matplotlib: Object-Oriented API, The Scripting Interface (pyplot), The MATLAB Interface (pylab). Pyplot và pylab đều là lightweight interfaces, tuy nhiên Pyplot cung cấp một giao diện thủ tục các thư viện vẽ hướng đối tượng trong matplotlib. Các lệnh vẽ của nó được thiết kế tương tự với Matlab cả về cách đặt tên và ý nghĩa các đối số. Cách thiết kế này đã giúp cho việc sử dụng pyplot dễ dàng và dễ hiểu hơn vì vậy trong các bài viết về Matplotlib, tôi sẽ sử dụng giao diện pyplot thay vì hai giao diện còn lại. Nếu chúng ta muốn can thiệp sâu hơn, với nhiều tùy chỉnh hơn thì Object-Oriented API sẽ là lựa chọn thích hợp.

Để cài đặt Matplotlib nếu bạn có Anaconda chỉ cần gõ conda install matplotlib hoặc sử dụng tools pip

*pip install matplotlib*

**Khái niệm chung**

Một Matplotlib figure có thể được phân loại thành nhiều phần như dưới đây:

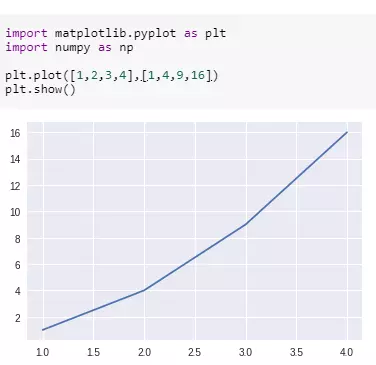
* Figure: Như một cái cửa sổ chứa tất cả những gì bạn sẽ vẽ trên đó.
* Axes: Thành phần chính của một figure là các axes (những khung nhỏ hơn để vẽ hình lên đó). Một figure có thể chứa một hoặc nhiều axes. Nói cách khác, figure chỉ là khung chứa, chính các axes mới thật sự là nơi các hình vẽ được vẽ lên.
* Axis: Chúng là dòng số giống như các đối tượng và đảm nhiệm việc tạo các giới hạn biểu đồ.
* Artist: Mọi thứ mà bạn có thể nhìn thấy trên figure là một artist như Text objects, Line2D objects, collection objects. Hầu hết các Artists được gắn với Axes.
* Bắt đầu với Pyplot
* Pyplot là một module của Matplotlib cung cấp các hàm đơn giản để thêm các thành phần plot như lines, images, text, v.v. vào các axes trong figure.

**Tạo một biểu đồ đơn giản**

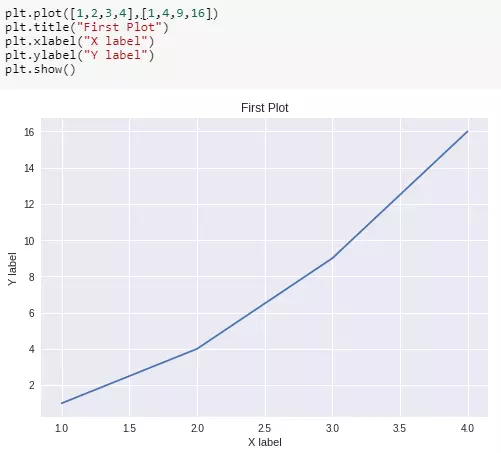
*import matplotlib.pyplot as plt*

*import numpy as np*

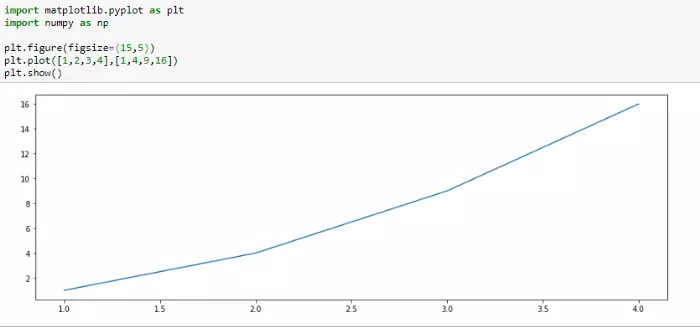
Ở đây chúng ta import Matplotlib’s Pyplot module và thư viện Numpy vì hầu hết các dữ liệu mà ta sẽ làm việc sẽ chỉ ở dạng mảng.



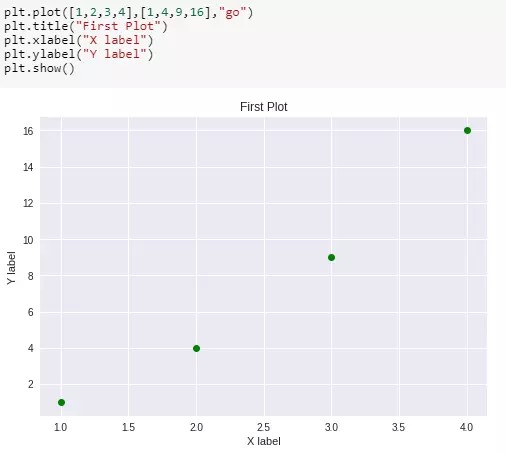
Chúng ta chuyển hai mảng làm đối số đầu vào cho phương thức plot() và sử dụng phương thức show() để gọi biểu đồ được yêu cầu. Ở đây lưu ý rằng mảng đầu tiên xuất hiện trên trục x và mảng thứ hai xuất hiện trên trục y của biểu đồ. Bây giờ, biểu đồ đầu tiên của chúng ta đã sẵn sàng, chúng ta hãy thêm tiêu đề và đặt tên trục x và trục y bằng cách sử dụng các phương thức title(), xlabel() và ylabel().



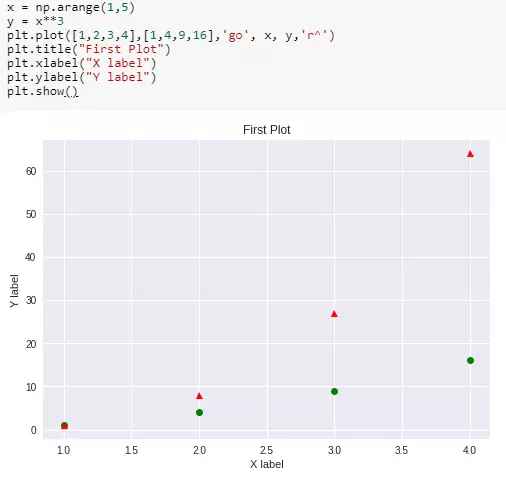
Chúng ta cũng có thể chỉ định kích thước của hình bằng cách sử dụng phương thức figure() và truyền các giá trị dưới dạng một tuple về độ dài của các hàng và cột cho đối số figsize



Với mỗi đối số X và Y, bạn cũng có thể chuyển một đối số thứ ba tùy chọn dưới dạng một chuỗi cho biết màu sắc và loại đường của biểu đồ. Định dạng mặc định là b- có nghĩa là một đường màu xanh lam đặc. Trong hình dưới đây, mình sử dụng go có nghĩa là vòng tròn màu xanh lá cây. Tương tự như vậy, chúng ta có thể thực hiện nhiều kết hợp như vậy để định dạng biểu đồ của mình.



Chúng ta cũng có thể vẽ nhiều bộ dữ liệu bằng cách chuyển vào nhiều bộ đối số của trục X và Y trong phương thức plot() như bên dưới.

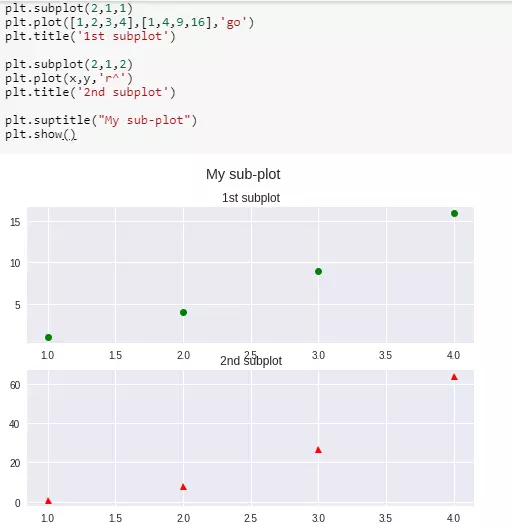


**Nhiều biểu đồ trong 1 figure**

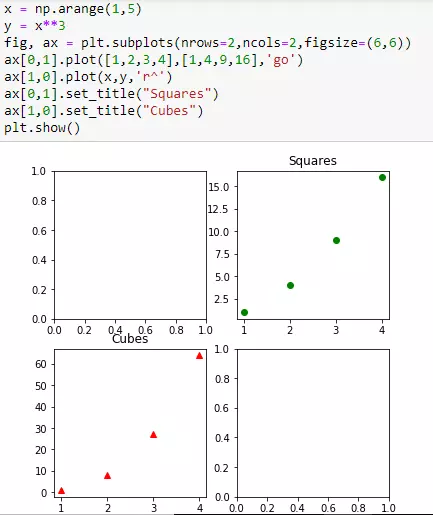
Chúng ta có thể sử dụng phương thức subplot() để thêm nhiều plots trong một hình. Trong hình ảnh bên dưới, mình đã sử dụng phương pháp này để phân tách hai biểu đồ mà đã vẽ trên cùng một trục trong ví dụ trước. Phương thức subplot() có ba đối số: nrows, ncols và index. Chúng chỉ ra số lượng hàng, số cột và số index của sub-plot. Ví dụ, mình muốn tạo hai sub-plot trong một hình sao cho nó nằm trên một hàng và trên hai cột và do đó ta chuyển các đối số (1,2,1) và (1,2,2) trong phương thức subplot(). Lưu ý rằng ta đã sử dụng riêng phương thức title() cho cả các subplots. Ta sử dụng phương thức suptitle() để tạo một tiêu đề tập trung cho hình.



Nếu ta muốn các sub-plots thành hai hàng và một cột, chúng ta có thể truyền các đối số (2,1,1) và (2,1,2)



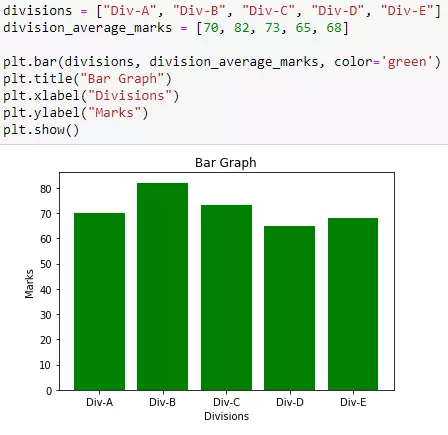
Cách tạo ra subplots trên đây trở nên hơi tẻ nhạt khi chúng ta muốn có nhiều subplots trong hình. Một cách thuận tiện hơn là sử dụng phương thức subpltots(). Lưu ý sự khác biệt của các giá trị trong cả hai phương thức. Phương thức này lấy hai đối số nrows và ncols làm số lượng hàng và số cột tương ứng. Phương thức này tạo ra hai đối tượng: figure và axes mà chúng ta lưu trữ trong các biến fig và ax có thể được sử dụng để thay đổi các thuộc tính mức figure và axes tương ứng. Lưu ý rằng các tên biến này được chọn tùy ý.



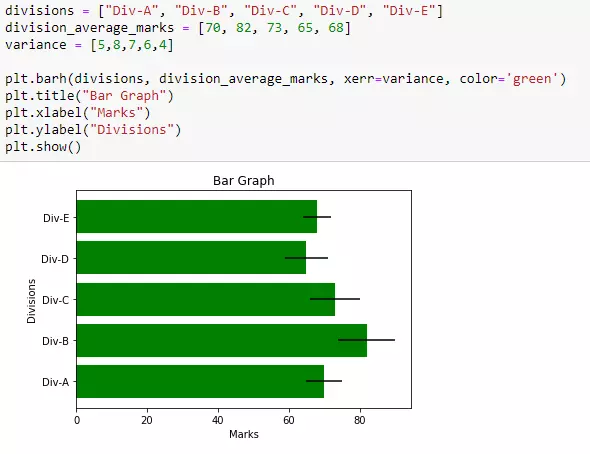
**Tạo các loại biểu đồ khác nhau với Pyplot**

1. Biểu đồ thanh

Biểu đồ thanh là một trong những loại biểu đồ phổ biến nhất và được sử dụng để hiển thị dữ liệu được liên kết với các biến phân loại. Pyplot cung cấp một phương thức bar() để tạo các biểu đồ thanh có các đối số: biến phân loại, giá trị và màu sắc của chúng (nếu bạn muốn chỉ định bất kỳ)



Để tạo biểu đồ thanh ngang sử dụng phương thức barh() Ngoài ra, chúng ta có thể truyền đối số (với giá trị của nó) xerr or yerr (trong trường hợp biểu đồ thanh dọc ở trên) để mô tả phương sai trong dữ liệu của chúng ta như sau:



Để tạo các biểu đồ thanh xếp chồng theo chiều ngang, ta sử dụng phương thức bar() hai lần và chuyển các đối số trong đó ta đề cập đến index và width của biểu đồ thanh để xếp chúng theo chiều ngang. Ngoài ra, chú ý việc sử dụng hai phương thức legend() được sử dụng để hiển thị chú giải của biểu đồ và xticks() để gắn nhãn trục x dựa trên vị trí của các thanh.



Tương tự, để xếp theo chiều dọc các biểu đồ thanh với nhau, chúng ta có thể sử dụng đối số bottom và đề cập đến biểu đồ thanh mà chúng ta muốn xếp chồng bên dưới làm giá trị của nó.



2. Biểu đồ tròn

Một loại biểu đồ cơ bản nữa là biểu đồ Pie có thể được tạo bằng phương thức pie() Chúng ta cũng có thể chuyển các đối số để tùy chỉnh biểu đồ Pie của mình để hiển thị shadow, explode một phần của nó.

# GIẢI PHÁP

## Mã nguồn mô tả dữ liệu

## Mã nguồn tiền sử lý dữ liệu

## Mã nguồn thống kê dữ liệu

## Mã nguồn trực quan hóa dữ liệu

# KẾT LUẬN

**Kết quả đạt được**

Sau khi hoàn thành xong bài tập lớn phân tích và xử lý dữ liệu sàn thương mại điện tử Mercado Livre với Python đã giúp em học thêm nhiều kiến thức hơn về Python, và các thông tin cơ bản về tình hình kinh tế, xu hướng tiêu dùng của Brazil. Quan trọng hơn hết vẫn là các kiến thức liên quan đến quá trình phân tích, xử lý 1 bộ dữ liệu khi lần đầu chạm vào. Về kiến thức python em đã đạt được:

* Cách sử dụng IPython, Jupyter notebook
* Các thư viện phục vụ cho việc phân tích và trực quan dữ liệu: numpy, pandas,...

Về quá trình các bước phân tích 1 bộ dữ liệu mới như sau:

* Tìm hiểu về nghiệp vụ của dataset( dữ liệu lấy từ đâu, dữ liệu mô tả về lĩnh vực gì, vấn đề mà doanh nghiệp đang phải đối mặt( hoặc vấn đề chung của lĩnh vực này), bộ dữ liệu đã cho có khả năng đưa ra các insight có ích giải quyết được vấn đề không, nếu không thì có có thể cho biết điều gì,..).
* Khai thác, phân tích bộ dữ liệu, (chia bộ dữ liệu thành 2 loại biến, định tính và định lượng, phân tích độ phân phối, tương quan, xu hướng tập trung của dữ liệu, cố gắng hiểu được bộ dữ liệu, xem nó đang mô tả điều gì) .
* Lấy insight trong dataset( đặt câu hỏi và kết hợp với trực quan cho dữ liệu, tập trung vào những câu hỏi mang giá trị cao, trực quan mang tính so sánh cao).
* Biến insight thành actionable insight( chọn lọc các câu hỏi để đi đến giải pháp, các đề xuất, giải pháp phải tính hiện thực cao và có khả năng là solution cho vấn đề, bài toán).

Các điều cần chú ý về kĩ thuật khi phân tích 1 bộ dữ liệu (Pov):

* Phân tích luôn luôn đi với trực quan, không phân tích mọi thứ lan man.
* Dữ liệu hashed, id không có khả năng trực quan đơn lẻ, nên dùng để đếm.
* Dữ liệu nào có thể số hóa thì nên số hóa (phải kèm theo quy tắc số hóa)
* Nếu dữ liệu có số lượng unique value cao, nên dùng top để so sánh.
* Dữ liệu trực quan phải có sự so sánh, tương phản, tương cao.
* Chọn biểu đồ thích hợp dự trên số lượng feature, tính chất feature,..
* Dữ liệu ngày tháng thường thống kê theo tháng, năm, mùa,.
* Chú thích sao cho người xem không cần phải đối chiếu qua trục để lấy value.

**Hạn chế của đề tài**

Tuy đề tài này đã được hoàn thành 1 cách chu đáo nhưng đâu đó vẫn có những điểm bất thường hoặc cũng có thể là sai sót. Một phần do đây là đề tài đầu tiên nên không thể chuẩn bị kiến thức 1 cách hoàn thiện được, cộng thêm thời gian tìm tòi nghiên cứu khá lâu để có thể ra được kết quả như hiện tại. Các hạn chế của đề tài:

* Thời gian chuẩn bị lâu
* Dữ liệu thiếu chính xác
* Đội ngũ còn hạn chế về nghiệp vụ chuyên môn, chưa có trải nghiệm thực tế.

**Hướng phát triển**

Đề tài này đã trở thành 1 bước đệm vững chắc cho em để có thể tiếp tục khám phá, phân tích các bộ dữ liệu mới sau này 1 cách dễ dàng hơn, giúp em trang bị các kiến thức cần thiết để có thể học được các kiến thức nâng cao về sau như Machine learning, Data Analysis.

# TÀI LIỆU THAM KHẢO

|  |  |  |
| --- | --- | --- |
| [1] |  | K. CNTT, Phân tích thiết kế phần mềm, Hưng Yên: Đại học SPKT H. |
| [2] |  | K. CNTT, Cơ sở dữ liệu, Hưng Yên: Đại học SPKT Hưng Yên, 2012. |
| [3] |  | K. CNTT, Lập trình ứng dụng windows forms, Hưng Yên: Đại học SPKT Hưng Yên, 2019. |
| [4] |  | W3schools, "W3schools," [Online]. Available: https://www.w3schools.com/. |
| [5] |  | K. CNTT, Kiểm thử phần mềm, Hưng Yên: Đại học SPKT Hưng Yên, 2016. |