TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP GIỮA KỲ MÔN XÁC SUẤT THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

**KHẢO SÁT MỘT SỐ HÀM VÀ THUẬT TOÁN ỨNG DỤNG TRONG THỐNG KÊ**

*Người hướng dẫn*: TS NGUYỄN QUỐC BÌNH

*Người thực hiện*: **BÙI PHƯƠNG NAM – 522H0002**

Lớp **: 22H50201**

Khoá  **: K26**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn đến Thầy Nguyễn Quốc Bình. Trong quá trình học cũng như là làm bài tiểu luận, nhờ có sự giúp đỡ tận tình của Thầy, Thầy đã truyền đạt cho chúng em những kiến thức bổ ích. Thầy đã hướng dẫn chúng em nhiệt tình, nhờ đó mà chúng em có thể hoàn thành bài tiểu luận này.XÁC SUẤT THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN là môn học thú vị và nhưng cũng rất phức tạp. Những kiến thức và kỹ năng về môn học này của em vẫn còn nhiều hạn chế. Do đó, bài tiểu luận của em khó tránh khỏi sai sót. Rất mong nhận được những góp ý của Thầy về bài tiểu luận này.

Một lần nữa, em xin chân thành cảm ơn!

**BÀI TẬP ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi và được sự hướng dẫn của TS NGUYỄN QUỐC BÌNH. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*Bùi Phương Nam*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

*TÓM TẮT*

Ở thời điểm hiện tại, nhu cầu về thống kê ngày càng trở nên quan trọng đối với các tổ chức doanh nghiệp. Bất kỳ tổ chức nào muốn hoạt động cũng cần có dữ liệu, và công cụ để xây dựng cũng như quản lý nó chính là thống kê. Chẳng hạn, Bộ giáo duc thường quan tâm đến phổ điểm của các học sinh qua các năm để điều chỉnh độ khó của đề thi nhằm đánh giá đúng năng lực; các nhà đầu tư quan tâm đến mức độ ổn định chính trị, ổn định kinh tế, môi trường pháp lý và số lượng/chất lượng nhân lực khi đầu tư quốc ngoại; các lãnh đạo quan tâm đến tỷ lệ nhập xuất để đánh giá mức độ phụ thuộc kinh tế của quốc gia;… Thống kê sẽ giải quyết những bài toán đó bằng nhiều bộ công cụ ra đời dưới dạng phần mềm hay tính năng của một ngôn ngữ lập trình. Do đó, đề tài này sẽ xoay quanh các thư viện thống kê đơn giản. Ngoài ra, để thấy được ứng dụng của thống kê trong các lĩnh vực khác, chẳng hạn như xử lý ảnh số, ta sẽ xét một số hàm chuyên dụng đi kèm với những cơ sở lý thuyết toán học quan trọng

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc387692905)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN i](#_Toc387692906)

[TÓM TẮT i](#_Toc387692907)

[MỤC LỤC 1](#_Toc387692908)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 1](#_Toc387692909)

[CHƯƠNG 1 – MỞ ĐẦU 1](#_Toc387692910)

[1.1 Tiểu mục cấp 1 1](#_Toc387692911)

[1.1.1 Tiểu mục cấp 2 1](#_Toc387692912)

[1.1.1.1 Tiểu mục cấp 3 1](#_Toc387692913)

[1.1.1.2 Tiểu mục cấp 3 tiếp theo. 1](#_Toc387692914)

[1.1.2 Tiểu mục cấp 2 tiếp theo 1](#_Toc387692915)

[1.2 Nội dung của chương này 1](#_Toc387692916)

[CHƯƠNG 2 – TỔNG QUAN 1](#_Toc387692917)

[1.1 Trình bày công thức toán học 1](#_Toc387692918)

[1.2 Trình bày một hình vẽ, sơ đồ 1](#_Toc387692919)

[CHƯƠNG 3 – CƠ SỞ LÝ THUYẾT / NGHIÊN CỨU THỰC NGHIỆM 1](#_Toc387692920)

[3.1 Chèn bảng: 1](#_Toc387692921)

[3.2 Viết tắt 1](#_Toc387692922)

[3.3 Trích dẫn 1](#_Toc387692923)

[3.3.1 Tài liệu tham khảo và cách trích dẫn 1](#_Toc387692924)

[3.3.2 Qui định của Khoa Công nghệ thông tin 1](#_Toc387692925)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

Hình 1.01: Ví dụ về hàm mean

Hình 1.02: Ví dụ về hàm geometric\_mean

Hình 1.03: Ví dụ về hàm harmonic\_mean

Hình 1.04: Ví dụ về hàm median

Hình 1.05: Ví dụ về hàm median\_low

Hình 1.06: Ví dụ về hàm median\_high

Hình 1.07: Ví dụ về median\_grouped

Hình 1.08: Ví dụ về mode

Hình 1.09: Ví dụ về multimode

Hình 1.10: Ví dụ về hàm quantiles

Hình 1.11: Ví dụ về hàm variance

Hình 1.12: Ví dụ về hàm pvariance

Hình 1.13: Ví dụ về hàm stdev

Hình 1.14: Ví dụ về hàm pstdev

Hình 1.15: Ví dụ về hàm covariance

Hình 1.16: Ví dụ về hàm correlation .

Hình 1.17: Ví dụ về hàm linear\_regression

CHƯƠNG 1 – CÁC HÀM THỐNG KÊ TRONG PYTHON

1.1 *Tổng quan*

Ở thời điểm hiện tại, đã có nhiều công cụ chuyên dụng dành cho các chuyên gia về thống kê như Minitab, SAS, Matlab,… Nhưng để tối giản hóa nội dung, đề tài sẽ chỉ quan tâm đến một số hàm nằm trong một module nhỏ của ngôn ngữ Python. Và nói đi thì cũng phải nói lại, mặc dù tối giản hóa, module mà đề tài hướng đến sẽ mới lạ hơn (bỏ qua các thư viện quen thuộc như Numpy, Sympy, Scipy). Cụ thể, đây sẽ là module *statistics*.

*Kham khảo thêm về module tại đường link:* [*statistics.py*](https://github.com/python/cpython/blob/3.12/Lib/statistics.py)

Trong module này, các hàm thống kê được chia thành 3 nhóm:

1.1.1 Nhóm ước lượng giá trị trung bình, trung vị của dữ liệu thống kê

|  |  |
| --- | --- |
| mean() | Tính trung bình cộng của tập dữ liệu |
| fmean() | Tương tự mean() nhưng hiệu suất nhanh hơn |
| geometric\_mean() | Tính trung bình hình học của tập dữ liệu |
| harmonic\_mean(). | Tính trung bình điều hoà của tập dữ liệu |
| median() | Tính trung vị (giá trị ở vị trí giữa) của tập dữ liệu. |
| median\_low() | Tính trung vị “dưới” của tập các dữ liệu |
| median\_high() | Tính trung vị “trên” của tập các dữ liệu. |
| median\_grouped() | Tính trung vị gom nhóm của một tập dữ liệu |
| mode() | Tìm một biến rời rạc với tần xuất cao nhất trong tập dữ liệu |
| multimode() | Tìm tất cả biến rời rạc với tần xuất cao nhất trong tập dữ liệu |
| quantiles() | Chia dữ liệu thành các khoảng có xác suất bằng nhau |

***1.1.2*** ***Nhóm tính toán độ lệch chuẩn và phương sai của dữ liệu***

|  |  |
| --- | --- |
| pstdev() | Hàm tính độ lệch chuẩn của toàn bộ dữ liệu Hàm tính độ lệch chuẩn |
| pvariance() | Hàm tính phương sai của toàn bộ dữ liệu |
| stdev(): | Hàm tính độ lệch chuẩn của mẫu dữ liệu nhỏ |
| variance(): | Hàm tính phương sai của mẫu dữ liệu nhỏ |

***1.1.3 Nhóm tính toán Covariance của tập dữ liệu***

|  |  |
| --- | --- |
| covariance() | Tính toán covariance sai mẫu giữa hai biến ngẫu nhiên |
| correlation() | Tính toán hệ số tương quan Pearson và Spearman giữa hai biến. |
| linear\_regression() | Tính toán góc hồi quy và hệ số giao của phương trình hồi quy tuyến tính đơn giản giữa hai biến |

1.2 Nội dung của chương

Chương này sẽ khảo sát chi tiết các hàm thống kê của module statistic. Bất kỳ khi nào sử dụng một hàm thống kê, người ta luôn quan tâm đến 3 vấn đề: tham số đầu vào của hàm, giá trị trả về, và cách thực hiện của hàm đó. Giải quyết được ba vấn đề này chính là chìa khóa để xác định cách giải quyết các bài toán thống kê. Trước hết, bắt đầu từ “Nhóm ước lượng giá trị trung bình, trung vị”.

1.2.1 Nhóm ước lượng giá trị trung bình, trung vị của dữ liệu thống kê

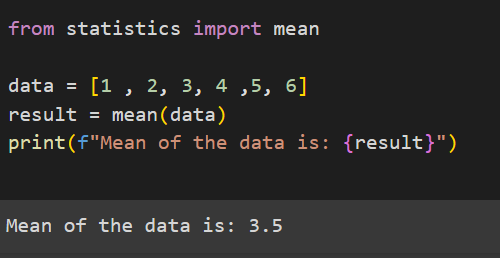
Trong thống kê có một đại lượng quan trọng là Kỳ vọng (Expectation), một đại lượng ám chỉ giá trị trung bình của dữ liệu được ghi nhận. Kỳ vọng rất có ý nghĩa thực tiễn. Nó là thước đo chuẩn để đánh giá dữ liệu vì nó cho ta biết các giá trị của dữ liệu thường gần với con số nào nhất. Công thức tổng quát của Expectation là:

Trong đó, P(X = x) là xác suất để biến ngẫu nhiên X đạt giá trị x

Thực tế, giá trị Kỳ vọng có thể là trung bình số học, trung bình hình học, hoặc trung bình điều hòa tùy thuộc vào loại dữ liệu. Trong trường hợp tập dữ liệu độc lập với xác suất P(X = x) tại mọi điểm là như nhau, ta có thể dùng trung bình số học để thay thế cho Kỳ vọng. Công thức tổng quát của trung bình số học là :

Module statistics trong python cung cấp hàm mean() giúp ta triển khai công thức trên, cụ thể hàm sẽ:

* Nhận tham 2 số đầu vào:
  + Danh sách các số thực
  + Số thực đầu tiên ta xét (mặc định là 0.0)
* trả về: một số thực. Nếu danh sách đó trống thì ném ra ngoại lệ [StatisticsError](https://docs.python.org/3/library/statistics.html#statistics.StatisticsError)



Hình 1.01: Ví dụ về hàm mean

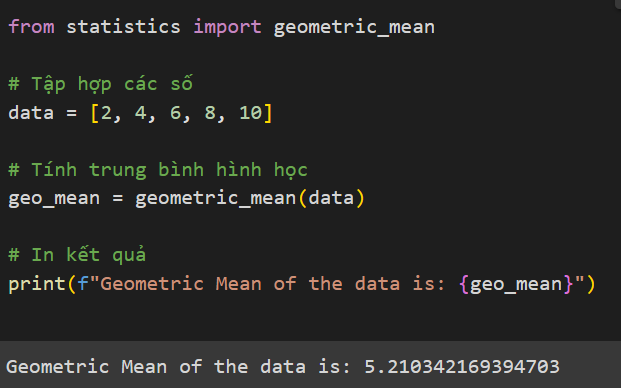
Trong ví dụ trên, ta thí nghiệm tung xúc xắc. Vì xác suất của mỗi nút là như nhau (tại một thời điểm chỉ có 1 nút được xuất hiện) nên ta có thể dùng hàm mean() để tính kỳ vọng của dữ liệu. Kết quả 3.5 cho ta biết rằng khả năng cao các nút ta nhận được rơi vào khoảng 3 – 4. Tuy nhiên, đó chỉ là lý thuyết, thực tế sẽ có nhiều yếu tố vật lý ảnh hưởng đến quá trình tung xúc xắc nên con số 3.5 chỉ mang tính chất tương đối

fmean() tương tự như mean() nhưng hiệu suất có phần nhanh hơn.

Có những trường hợp, dữ liệu sẽ là các giá trị cấp số nhân (chẳng hạn như lãi suất, số lượng tế bào vi khuẩn phân chia trong thí nghiệm ). Do đó, người ta sẽ cần dùng đến trung bình hình học để tính kỳ vọng. Ta có công thức:

Để triển khai công thức trên, ta có hàm geometric\_mean(). Hàm sẽ:

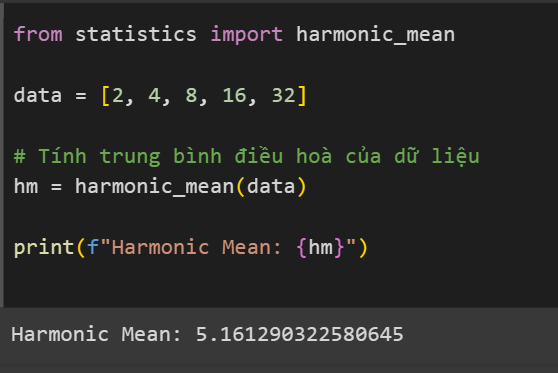
* Nhận 1 tham số đầu vào: Danh sách các số thực
* Giá trị trả về: một số thực
* Nếu danh sách đó trống thì ném ra ngoại lệ [StatisticsError](https://docs.python.org/3/library/statistics.html#statistics.StatisticsError)

*Hình 1.02: Ví dụ về hàm geometric\_mean*

Tuy nhiên, cũng có trường hợp, dữ liệu là các con số nhỏ hơn 1 ( chẳng hạn tỷ lệ vốn hóa, phần trăm). Khi đó người ta sẽ dùng trung bình điều hòa để gán cho Kỳ vọng. Công thức toán học cụ thể là:

Và để thực hiện hóa điều đó trong code, ta có hàm harmonic\_mean(). Cụ thể, hàm sẽ

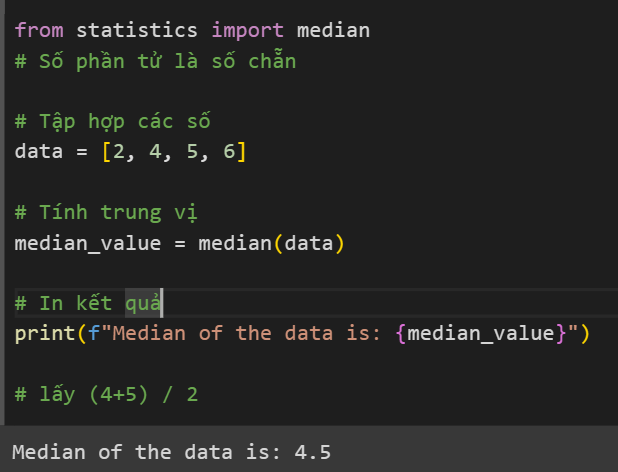
* Nhận 1 tham số đầu vào: Danh sách các số thực
* Trả về: một số thực
* Nếu danh sách đó trống thì ném ra ngoại lệ [StatisticsError](https://docs.python.org/3/library/statistics.html#statistics.StatisticsError)

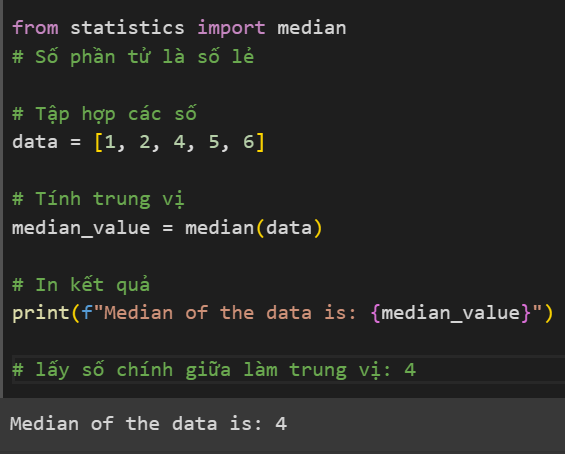


Hình 1.03: Ví dụ về hàm harmonic\_mean

Tuy nhiên, đôi khi Kỳ vọng có thể không có ý nghĩa. Chẳng hạn, ta không thể tính lương trung bình của một công ty nếu lương của các nhân viên là 5 triệu/tháng trong khi lương của giám đốc là 100 triệu/tháng, chưa kể công ty sẽ có nhiều người quản lý nên con số sẽ càng khác biệt. Trong một tập dữ liệu, sẽ có những giá trị quá lớn hoặc quá nhỏ để dùng kỳ vọng, do đó ta sẽ chọn con số chính giữa của tập dữ liệu để làm thước đo. Con số đó chính là trung vị. Trong python, ta có các hàm tính trung vị sau:

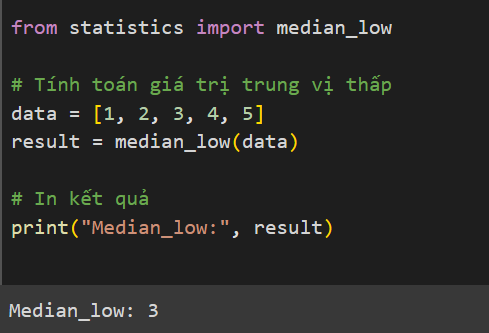
Hàm median():

* Nhận tham 1 số đầu vào: Danh sách các số thực
* trả về: một số thực/số nguyên
* Cách thực hiện: Nếu danh sách đó trống thì ném ra ngoại lệ [StatisticsError](https://docs.python.org/3/library/statistics.html#statistics.StatisticsError)



*Hình 1.04: Ví dụ về hàm median*

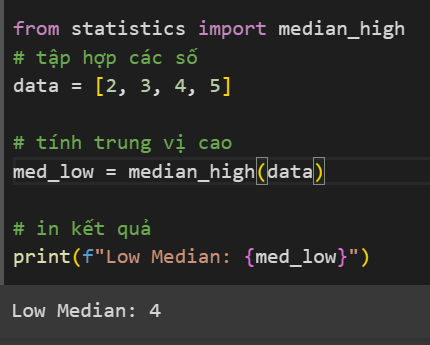
Ta còn có một phiên bản khác của median() là hàm median\_low(). Cách hoạt động gần như tương tự, nhưng nếu số hần tử là số chẵn, median\_low() sẽ trả về trung vị “thấp”, tức giá trị gần nhất thấp hơn so với trung vị trong danh sách



Hình 1.05: Ví dụ về hàm median\_low

Trong ví dụ trên, số phần tử là 4 nên trung vị của tập số là ( 3.4 + 4 ) /2 = 3.7 nhưng hàm luôn trả về số nhỏ hơn có trong danh sách là 3.4

Tương tự như median\_low(), ta cũng có hàm trả về trung vị “trên” của danh sách là median\_high() khi số phần tử là số chẵn



*Hình 1.6: Ví dụ về hàm median\_high*

Trong ví dụ trên, ố phần tử là 4 nên trung vị là (4+3)/2 = 3.5 nhưng hàm sẽ trả về giá trị lớn hơn 3.5 trong danh sách là 4

Một dạng trung vị đặc biệt khác mà ta quan tâm là trung vị gom nhóm. Đối với tập dữ liệu là biến liên tục, ta không thể tìm được chính xác giá trị ở giữa( giống như việc ta không thể tìm chính xác chiều cao trung bình của tất cả học sinh trong trường). Khi đó, giải pháp đặt ra là ta sẽ chia tập dữ liệu thành nhiều nhóm rồi tính trung vị theo công thức toán học:

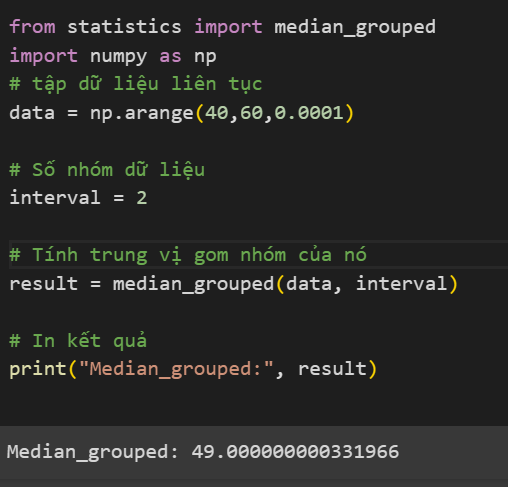
, Trong đó:

* + - L là Giới hạn dưới của nhóm thứ k trong danh sách
    - n là số phần tử trong danh sách
    - fk là tần số của trung vị của nhóm thứ k
    - cf­k-1­ là tần số tích lũy của nhóm thứ k - 1

Trong python, ta sẽ cần hàm median\_group() để tính trung vị gom nhóm. Cụ thể, hàm median\_group() sẽ:

- Nhận tham 2 số đầu vào:

* + Danh sách các số thực
  + d - Độ rộng của các nhóm
* trả về: một số thực/số nguyên. Nếu danh sách đó trống thì ném ra ngoại lệ [StatisticsError](https://docs.python.org/3/library/statistics.html#statistics.StatisticsError)

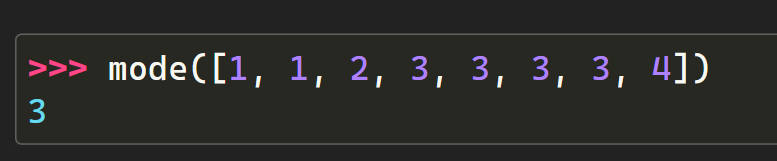


*Hình 1.07: Ví dụ về median\_grouped*

Đôi khi, việc chỉ ra một giá trị thường xuyên xuất hiện trong dữ liệu cũng là một cách đánh giá thay thế cho kỳ vọng và trung vị. Để làm điều đó, python cho ta 3 hàm:

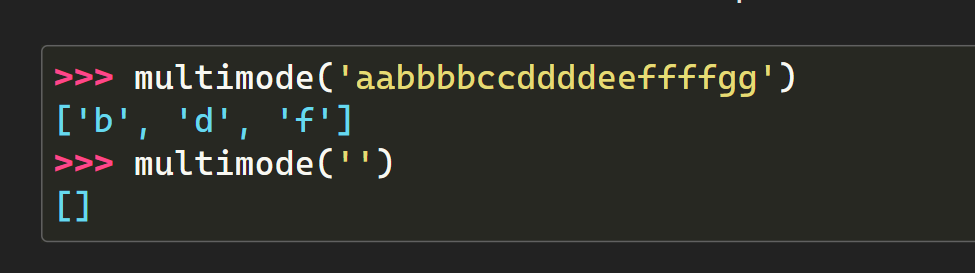
Hàm mode

* Nhận tham 1 số đầu vào: Danh sách các số thực
* trả về: một số thực/số nguyên có số lần xuất hiện lớn nhất



*Hình 1.08: Ví dụ về mode*

multimode: tương tự mode nhưng tham số nhận vào và giá trị trả về có thể rất đa dạng. ví dụ như chuỗi, ký tự,…



*Hình 1.09: Ví dụ về multimode*

Hàm quantile() là một hàm đặc biệt và không liên quan đến việc tính giá trị trung bình hay trung vị, mà nó tính các phân vị (quantile) của tập dữ liệu. Phân vị thứ n trong tập dữ liệu là giá trị cho ta biết tồn tại n phần trăm dữ liệu luôn nhỏ hơn hoặc bằng nó. Điều này rất có ý nghĩa vì không phải lúc nào dữ liệu cũng phân bố đồng đều ở mọi vị trí. Chẳng hạn như, người ta thống kê một tập thể có 99% nam giới đạt chiều cao là 6 feet. Khi đó, 6 feet chính là phân vị của phần trăm thứ 99. Ta có thể tính được phân vị bằng hàm quantile()

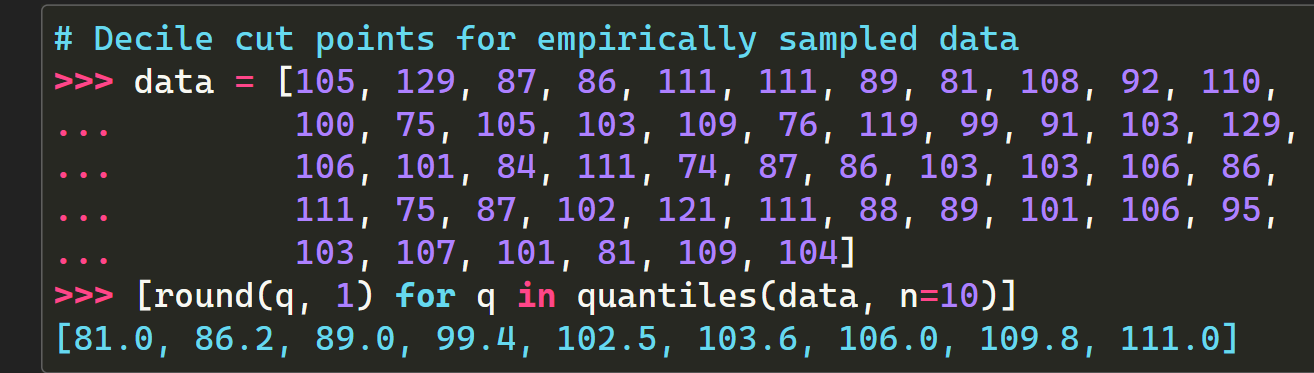
Cụ thể, hàm sẽ:

* Nhận 3 Tham số đầu vào:
  + danh sách các số thực.
  + độ rộng từng khoảng
  + Phương pháp tính phân vị. Mặc định là 'exclusive'

Hoặc có thể lựa chọn ‘inclusive'

* Giá trị trả về:

một mảng các giá trị quantile tương ứng với mỗi phần trăm.



*Hình 1.10: Ví dụ về hàm quantile*

Trong ví dụ trên, ta chia dữ liệu thành 10 khoảng. Kết quả trả về [81.0, 86.2, 89.0, 99.4, 102.5, 103.6, 106.0, 109.8, 111.0] đang hiển thị các quantile tương ứng với phần trăm 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 của tập dữ liệu.

Để giải thích cụ thể hơn, chúng ta sẽ có:

* Phần trăm 10% có phân vị là 81.0. Điều này nghĩa là có 10% số thực nhỏ hơn hoặc bằng 81.0 trong danh sách
* Phần trăm 20% có phân vị là 86.2. Tương tự, có 20% số thực nhỏ hơn hoặc bằng 86.2 trong danh sách
* Phần trăm 30% có phân vị là 89.0. Nghĩa là có 30% dữ liệu nhỏ hơn hoặc bằng 89.0
* Và cứ tiếp tục như vậy với các quantile 0.4, 0.5, 0.6, 0.7, 0.8, 0.9

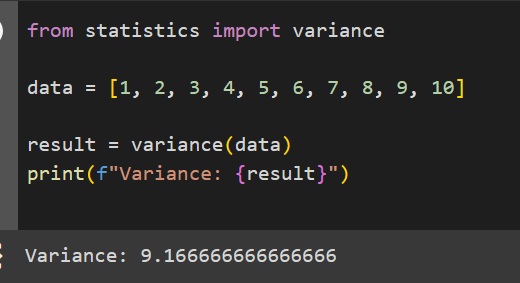
Những giá trị quantile cho ta biết các phân bố của các giá trị trong dữ liệu. Dựa vào các giá trị đó, ta có thể dự đoán xem kết quả mong đợi trong tương lại sẽ rơi vào phân bố nào nếu quy luật không thay đổi.

Tiếp theo, ta xét đến nhóm “tính độ lệch chuẩn và phương sai dư liễu”. Đôi khi, dữ liệu sẽ có những giá trị phân tán khỏi đại lượng kỳ vọng. Trong một số trường hợp, các giá trị đó có thể là phương sai (variance). Khi đó ta có công thức tổng quát:

, với

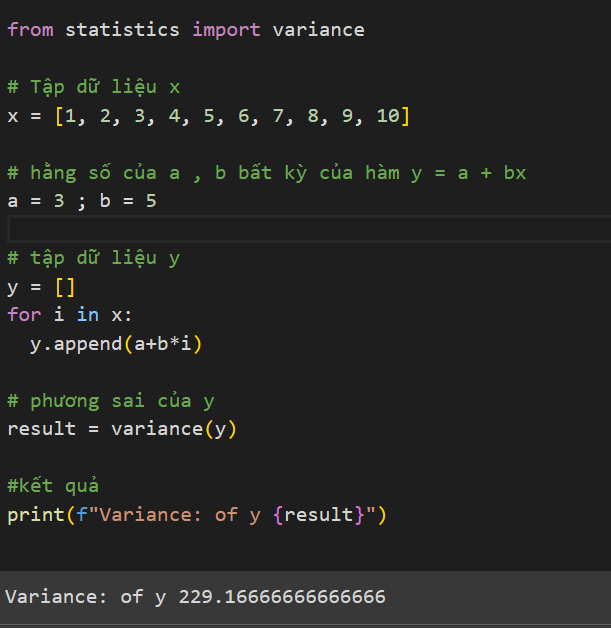
Để tính phương sai thông qua công thức trên, module statistic cung cấp cho ta hàm variance(). Cụ thể, hàm sẽ:

* Nhận 2 Tham số đầu vào:
  + danh sách các số thực.
  + µ (nếu có)
* Giá trị trả về: một số thực



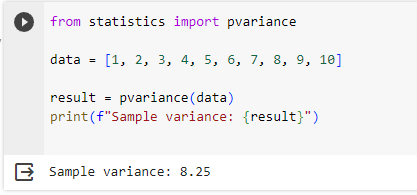
*Hình 1.11: Ví dụ về hàm variance*

Ngoài ra, variance() còn có thể tính phương sai của một tập dữ liệu được ánh xạ từ một tập dữ liệu khác thông qua một hàm tuyến tính. Thật vậy, giả sử ta có biến ngẫu nhiên Y và X thỏa đẳng thức Y = a + bX với a, b là các số thực. Thử nghiệm, với hàm variance() , ta có kết quả sau:



Không chỉ có một dữ liệu tổng thể, ngay cả tập hợp con của dữ liệu đó cũng có phương sai riêng mà người ta gọi là phương sai của mẫu. Điều này thường xảy ra khi ta đưa các giá trị thành nhóm mà tại nhóm đó tồn tại một giá trị kỳ vọng, đồng thời tất cả giá trị bị phân tán khỏi kỳ vọng ấy. Để tính được phương sai của mẫu, ta sẽ dùng đến hàm pvariance(). Hàm sẽ:

* Nhận 2 tham số:
  + Danh sách số thực
  + mu - Trung bình mẫu (nếu có)
* trả về: một số thực được tính băng công thức tương tự (1)



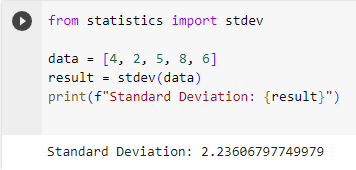
*Hình 1.12 ví dụ về hàm pvariance*

Một đại lượng khác cũng có thể biễu diễn mức độ phân tán của dữ liệu khỏi giá trị kỳ vọng là độ lêch chuẩn. Đại lượng này được dùng nhiều hơn so với phương sai vì nó dễ tiếp cận với cả người chuyên lẫn không chuyên về thống kê. Về cơ bản, độ lêch chuẩn chính là căn bâc 2 không âm của phương sai.

(1)

Trong module statistics, ta có thể sử dụng hàm [stdev()](https://docs.python.org/3/library/statistics.html#statistics.stdev), để tính độ lệch chuẩn của dữ liệu. Hàm sẽ:

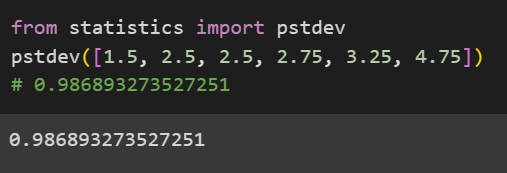
* Nhận 2 tham số:
  + Danh sách số thực
  + mu – giá trị kỳ vọng
* trả về: một số thực



*Hình 1.13 Ví dụ về stdev*

Tương tự hàm pvariance, ta cũng có một đại lượng là độ lệch mẫu dùng để biểu diễn phân tán của một tập con của dữ liệu gốc. Để tính được độ lệch mẫu, ta sẽ dùng đến hàm pstdev(). Hàm sẽ:

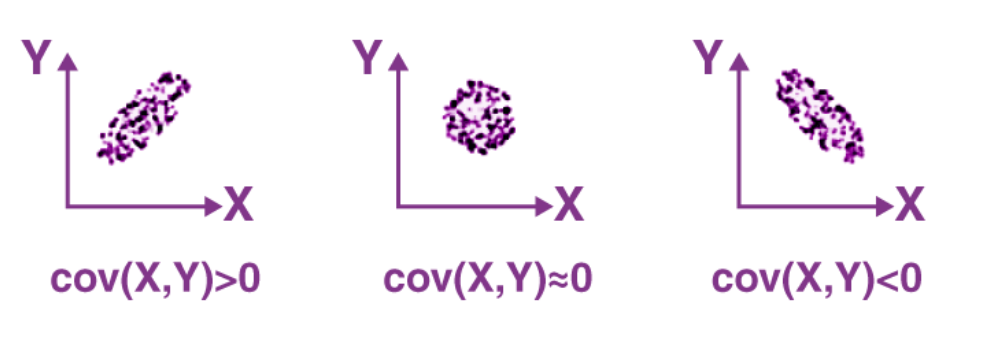
* Nhận 2 tham số:
  + Danh sách số thực
  + mu - Trung bình mẫu (nếu có)
* trả về: một số thực chính bằng kết quả tính từ công thức (1)



*Hình 1.14 Ví dụ về pstdev*

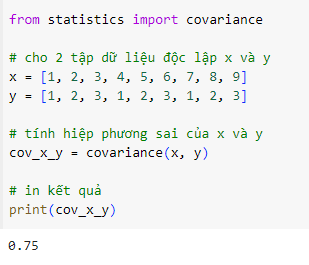
Ta tiếp đến nhóm “Tính toán quan hệ giữa 2 biến độc lập” . Có những bài toán thống kê phức tạp mà dữ liệu không chỉ có một mà có đến 2 biến liên tục/rời rạc. Nghĩa là ta phải khảo sát cả 2 tập dữ liệu độc lâp. Điều quan trọng là cần phải tìm ra mối quan hệ giữa 2 biến này để đi đến kết quả tối ưu nhất. Thông thường, giữa chúng sẽ có các đại lượng chung: hiệp phương sai, hệ số tương quan pearson, độ dốc và điểm chặn của hồi quy tuyến tính.

Đại lượng đầu tiên, Hiệp phương sai (covariance) , hiểu đơn giản là phương sai của cả 2 biến ngẫu nhiên độc lâp ứng với 2 tập dữ liệu, nhưng thay vì tính toán độ phân tán của chúng so với kỳ vọng, đại lượng sẽ đo lường mức độ của thay đổi của 2 biến so với nhau. Chẳng hạn, nếu ta có 2 biến độc lập X và Y, giá trị covariace của chúng âm thì điều đó có nghĩa là chúng sẽ thay đổi ngược hướng với nhau trên đồ thị. Ngược lại , nếu giá trị covariace của chúng dương thì điều đó có nghĩa là chúng sẽ thay đổi cùng hướng với nhau. Nếu bằng 0, sự thay đổi chúng không thay đổi quá nhiều và gần như đứng tại chỗ.



Trong python, ta có thể tính được covariance của 2 tập dữ liệu nhờ hàm covariance() của nhóm “Tính toán quan hệ giữa 2 biến độc lập”. Hàm sẽ:

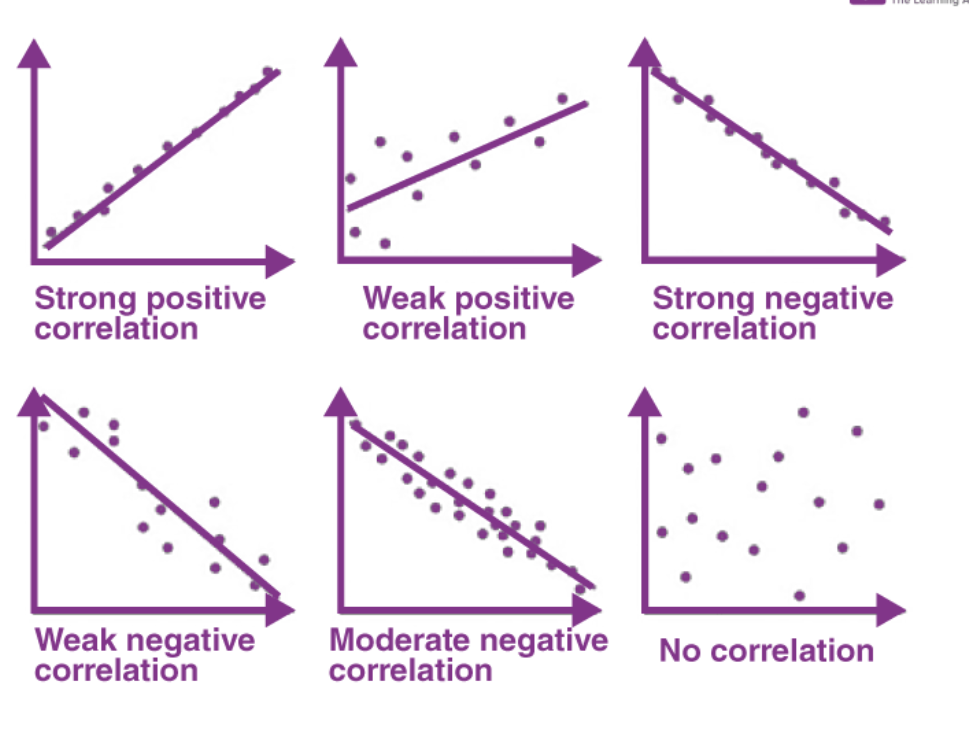
* Nhận 4 tham số:
  + 2 danh sách số thực có kích thước không hơn nhau 2 đơn vị trở lên
  + 2 trung bình mẫu ứng với 2 danh sách (nếu có)
* trả về: một số thực



***Hình 1.14: Ví dụ về hàm covariance***

Đại lượng thứ 2, hệ số tương quan Pearson,  là đại lượng mô tả cách mà một biến thay đổi trong mối quan hệ với một biến khác. Nó cho ta một đường thẳng phù hợp biểu diễn quan hệ dữa 2 biến độc lập. Hệ số tương quan sẽ có giá trị từ -1.0 đến +1.0. Trong đó:

* +1.0 đại diện cho **Mối tương quan thuận**, tức cả hai biến thay đổi theo cùng một hướng
* -1.0 đại diện cho **Mối tương quan nghịch**, tức cả hai biến thay đổi theo các hướng ngược nhau
* **0** nghĩa là không có tương quan nào cả



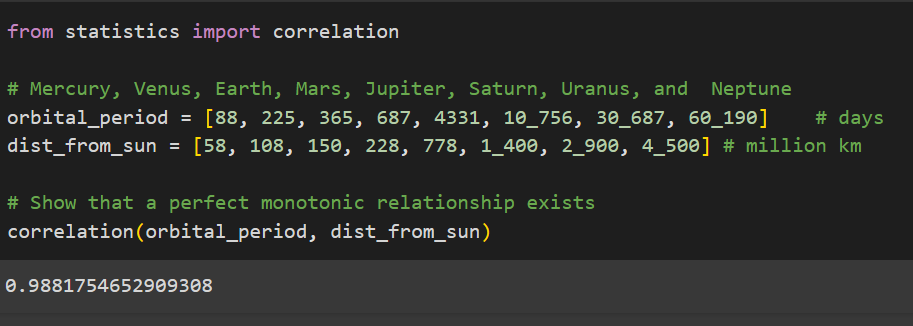
Để tính được hệ số tương quan của 2 tập dữ liệu, ta sẽ dùng hàm correlation() với 3 tham số:

* + 2 danh sách số thực có kích thước không hơn nhau 2 đơn vị trở lên
  + Phương thức tính hệ số. mặc định là ‘linear’

Hoặc có thể chọn ‘rank’

Lưu ý: một số phiên bản python không hỗ trợ nhập phương thức

Hàm sẽ trả về một số thực



Hai đại lượng cuối cùng, độ dốc và điểm chặn trong hồi quy tuyến tính, luôn đi thành từng cặp sẽ được tính thông qua hàm [linear\_regression()](https://docs.python.org/3/library/statistics.html#statistics.linear_regression). Hồi quy tuyến tính là phương pháp thống kê đặc biệt trong machine learning dùng để thiết lập mối quan hệ giữa một biến phụ thuộc với nhiều biến độc lập khác nhau. Ví dụ, ta có quan hệ giữa Chiều cao & trọng lượng với tuổi của một người được biểu diễn qua phương trình

Tuổi = 5 + Chiều cao × 10 + Trọng lượng × 13

Có rất nhiều cách biểu diễn hổi quy tuyến tính. Tuy nhiên, ta chỉ xét công thức hồi quy đơn giản sau:

Trong đó

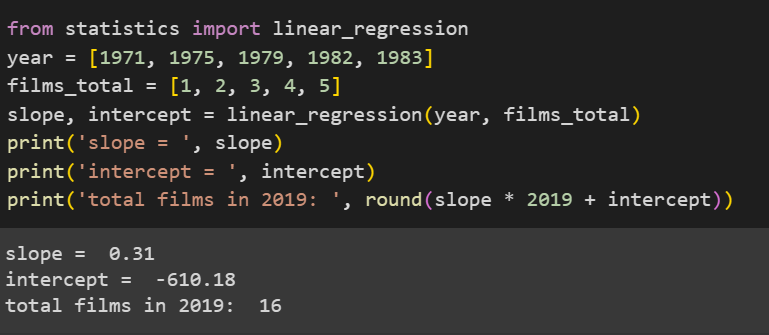
* + y là biến phụ thuộc mà chúng ta cố gắng dự đoán hoặc giải thích.
  + X là biến độc lập
  + slope là hệ số góc, thể hiện mức độ biến đổi của y tương ứng với mỗi biến độc lập x.
  + intercept là hệ số giao, là giá trị dự đoán của y khi x bằng 0.
  + Noise là sai số giữa giá trị y thực và giá trị y theo lý thuyết.

Trong python, hàm [linear\_regression()](https://docs.python.org/3/library/statistics.html#statistics.linear_regression) sẽ triển khai công thức trên. Hàm sẽ:

* Nhận 2 tham số: 2 mảng số thực có cùng kích thước
* Trả về: giá trị slope và intercept

Giả sử ta có bài toán sau: [Monty Python là một nhóm hài nổi tiếng của Anh được thành lập vào năm 1969 bởi Graham Chapman, John Cleese, Terry Gilliam, Eric Idle, Terry Jones và Michael Palin1](https://en.wikipedia.org/wiki/Monty_Python%27s_Flying_Circus). Nhóm này đã sản xuất nhiều chương trình truyền hình, phim điện ảnh và các tác phẩm khác. Biết rằng năm trong các năm 1971, 1975, 1979, 1982, 1983 , họ đã ra mắt tổng số lượng phim tương ứng là 1, 2, 3, 4, 5. Hỏi năm 2019, dự đoán họ có tổng cộng bao nhiêu bộ phim nếu tốc độ sản suất không thay đổi?

Giải: Ta không có công thức cụ thể biểu diễn quan hệ giữa năm và số lượng phim nhóm Monty Python cho ra mắt nhưng ta nhận thấy chúng tỉ lệ thuận với nhau. Ta có thể dùng hàm [linear\_regression()](https://docs.python.org/3/library/statistics.html#statistics.linear_regression) kiểm tra mối quan hệ giữa chúng

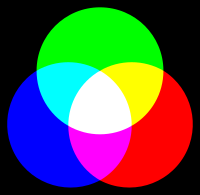


CHƯƠNG 2 – Cân bằng Histogram hình ảnh

Cân bằng histogram hình ảnh là thuật toán phổ biến làm cân bằng độ tương phản của ảnh kỹ thuật số bằng cách làm tăng độ sáng của các điểm ảnh. Thuật toán được hình thành dựa trên lý thuyết thống kê cơ bản.

2.1 Cơ sở lý thuyết

Về nguyên tắc, mỗi hình ảnh cơ bản được cấu tạo từ hàng triệu điểm ảnh với màu sắc khác nhau (giống như cơ thể con người được tạo ra từ hàng tỷ tế bào). Mỗi điểm ảnh được gọi là một pixel và màu của nó được biểu diễn dưới dạng con số 8 bit theo tiêu chuẩn của hệ thống mô hình màu RGB. Theo đó, giá trị của một pixel là (a , b, c) với a, b, c là giá trị tương ứng của ba màu cơ bản là đỏ, lục , lam, giới hạn trong khoảng (0, 255) như trong hình dưới đây:

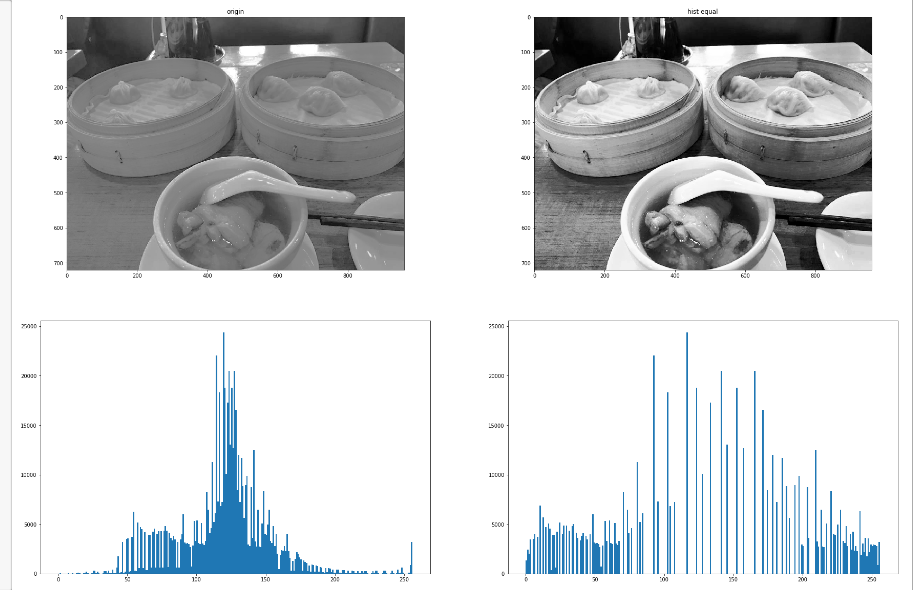


Một pixel được cho là tối nhất nếu giá trị của nó là (0,0,0), và nếu một tấm ảnh có toàn bộ pixel như vậy là một tấm ảnh đen kịt. Ngược lại, một pixel sáng nhất sẽ có giá trị là (255, 255, 255), và bức ảnh chứa toàn bộ pixel như vậy là bức ảnh toàn màu trắng.

255,255,255 )

(0 , 0 , 0 )

Trong xử lý hình ảnh, , mỗi một bức ảnh là một đối tượng có 2 thuộc tính là số lượng pixel và mức xám của pixel. Histogram(lược đồ mức xám) của một hình ảnh cho ta biết được số pixel ứng với một mức xám nhất định. Nếu ta coi số lượng pixel và mức xám của chúng là 2 biến ngẫu nhiên Y và X độc lập của một tập dữ liệu thì ta có thể xem xét phân phối của chúng trên histogram để đánh giá độ tương phản của một bức ảnh:



Hình 1.1..đồ thị histogram

Trong 2 ảnh trên, ta có 2 histogram, mỗi cái đều có trục ngang đại diện cho các giá trị mức xám của ảnh và các cột dọc màu xanh đại diện cho số lượng pixel. Có thể thấy, khi các cột tập trung lại vị trí trung tâm của trục X (hình bên trái), nó cho ta một hình ảnh khá mờ vì gần như toàn bộ pixel đều có mức xám ở tầm trung trong khi có rất ít pixel mang mức xám rất lớn hoặc rất nhỏ. Điều này có nghĩa là phân phối xác suất của X và Y là phân phối Gaussian. Ngược lại, khi các cột dọc phân bố trải đều ở mọi điểm trên trục ngang (hình bên phải), nghĩa là số lương pixel tại mọi mức xám đều rất lớn, nó cho ta một hình ảnh cực rõ nét. Nói cách khác, màu trắng và màu đen trong mọi vị trí của bức ảnh đều rõ ràng tạo độ tương phản cho bức ảnh. Phân phối xác suất của X và Y lúc này đã trở thành phân phối đều. Từ đó có thể rút ra kết luận, để hình có độ tương phản cao, ta sẽ đưa phân phối ban đầu của X và Y về dạng chuẩn, làm cho đồ thị histogram “giãn nở”. Ngược lại, nếu muốn giảm độ tương phản, ta cần làm cho đồ thị “co lại”. Thuật toán histogram có thể giải quyết được bài toán. Tuy nhiên, nó chỉ áp dụng cho ảnh có độ tương phản thấp. Đối với ảnh đã rõ nét, cân bằng histogram có thể làm cho ảnh trở nên quá sáng hoặc quá đậm.

Thuật toán bao gồm các bước:

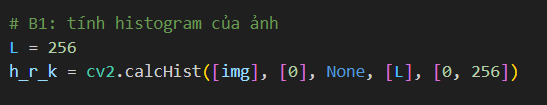
1. Tính toán hàm histogram:

gọi rk là mức xám thứ k của ảnh, nk là số pixel có giá trị r­k. Khi đó, histogram tổng quát cho mọi ảnh được xác định bởi hàm:

Trong python, ta có thể tính bằng hàm calcHist(). Hàm sẽ

Nhận 4 tham số:

* + images: Danh sách các ảnh đầu vào.
  + channels: danh sách kênh màu. Ví dụ, nếu ảnh là ảnh xám, channels=[0]. Đối với ảnh màu, channels=[0, 1, 2] ứng với 3 màu lam, lục, đỏ
  + mask: Mặt nạ ảnh. Ta không quan tâm tham số này nên mặc định là None
  + histSize: Số lượng bins (ngăn) trong histogram. Đối với ảnh xám, thường là [256]. Đối với ảnh màu, có thể là [256, 256, 256] để có histogram cho từng kênh.
  + ranges: Phạm vi giá trị để tính histogram. Đối với ảnh xám, thường là [0, 256]. Đối với ảnh màu, có thể là [0, 256, 0, 256, 0, 256] để xác định phạm vi cho từng kênh.

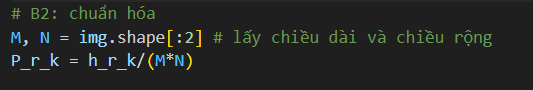


Trả về: mảng numpy chứa các giá trị histogram ứng với từng pixel

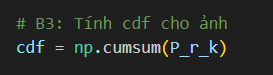
1. Chuẩn hóa h(rk) thành P(rk) cho phù hợp với hình ảnh cần xử lý

​​

Lý giải công thức trên: một tấm ảnh luôn có tổng số lượng pixel là M × N . nk là số lượng pixel có mức xám rk­ đang xét. Do đó, bản chất P(rk) chính là mật độ xác suất (pdf) của số lượng pixel đang xét

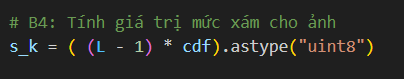


1. Tính cdf cho ảnh:



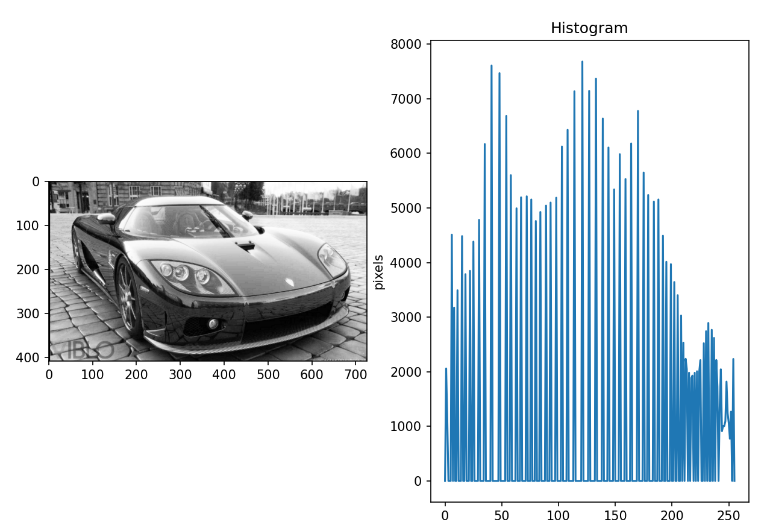
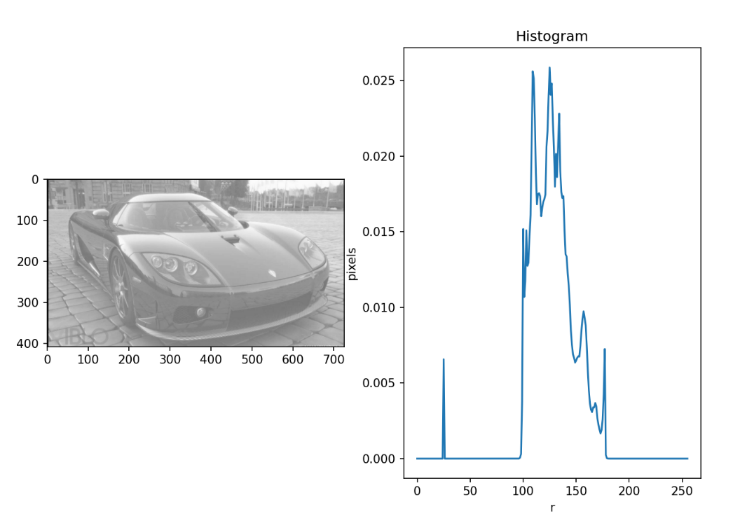
cdf của ảnh là một hàm tăng biểu diễn xác suất tích lũy của từng mức xám. Giá trị cdf giới hạn trong đoạn [0,1]

1. Tính giá trị mức xám cho ảnh mới: Sk = (L – 1) CDF



Việc nhân cdf với giá trị L – 1 nhằm chuyển đổi từ đoạn [0 , 1] thành đoạn [0 , L – 1] đại diện cho miền giá trị mức xám ứng mới. Vì bài toán yêu cầu tăng độ tương phản của ảnh nên mặc định L là 256. Trong một số trường hợp xử lý ảnh có mức xám lớn hơn 256, giá trị L sẽ lớn hơn.

1. Tạo ảnh mới từ mức xám trên

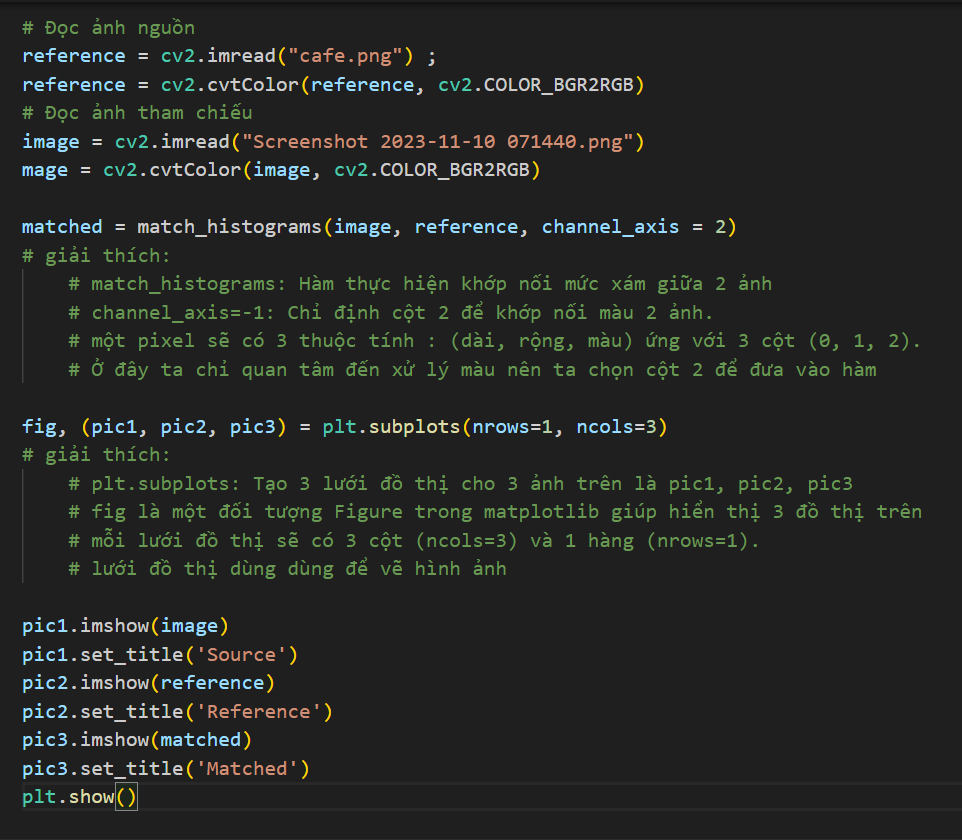
****Ban đầuSau khi chỉnh sửa

CHƯƠNG 2 – Khớp nối Histogram hình ảnh

Trong xử lý ảnh , ta cũng có một cách tương tự như Cân bằng histogram là khớp nối histogram. Tuy nhiên, bài toán sẽ có một chút thay đổi: cho một ảnh bất kỳ, yêu cầu so khớp hisogram của nó với histogram của tấm ảnh khác. Công việc này chính là khớp nối histogram. Bản chất nó chính là làm cho histogram của 2 tấm ảnh khớp với nhau. Như vậy, về cơ bản, ta sẽ cần đến 2 tấm ảnh, một tấm cần xử lý và một tấm dùng làm khuôn mẫu. Chất lượng ảnh được xử lý sẽ tốt nếu khuôn mẫu có histogram tốt, ngược lại thì có thể tệ hơn.



Matching histogram là một quá trình phức tạp gấp nhiều lần so với equalizing histogram. Do đó, để thuận tiện, ta có thể sử dụng hàm match\_histograms từ thư viện skimage.exposure để so khớp histogram của ảnh nguồn (image) với histogram của ảnh tham chiếu.



**TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

1. [**statistics**](https://docs.python.org/3/library/statistics.html)
2. **NguyenDuong -** [**Viblo**](https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRlY3)
3. [Trung Thành Nguyễn](https://viblo.asia/u/TrungThanhNguyen0502) - [Viblo](https://viblo.asia/p/xu-li-anh-thuat-toan-can-bang-histogram-anh-GrLZDOogKk0)
4. [Nguyễn Tiến Đạt](https://viblo.asia/u/nguyen.tien.datb)- [Viblo](https://viblo.asia/p/tuan-3-histogram-histogram-equalization-3P0lPnxmKox)
5. [**scikit-image.org**](https://scikit-image.org/docs/stable/auto_examples/color_exposure/plot_histogram_matching.html)