TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN GIỮA KỲ MÔN XÁC SUẤT VÀ THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

**XÁC SUẤT VÀ THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

*Người hướng dẫn*: **ThS NGUYỄN QUỐC BÌNH**

*Người thực hiện*: **MAI NGUYỄN PHƯƠNG TRANG**

Lớp **: 22050201**

Khoá  **: 26**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN GIỮA KỲ MÔN XÁC SUẤT VÀ THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

**XÁC SUẤT VÀ THỐNG KÊ ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

Người hướng dẫn: **ThS NGUYỄN QUỐC BÌNH**

Người thực hiện: **MAI NGUYỄN PHƯƠNG TRANG**

Lớp **: 22050201**

Khoá  **: 26**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Lời đầu tiên em xin gửi lời cảm ơn chân thành đến thầy, cô trong Khoa Công nghệ thông tin thuộc trường Đại học Tôn Đức Thắng đã tạo điều kiện cho chúng em được học môn Xác suất và thống kê ứng dụng cho công nghệ thông tin. Cùng với lòng biết ơn sâu sắc, em xin đặc biệt gửi lời cảm ơn đến thầy Nguyễn Quốc Bình - Người đã trực tiếp giảng dạy và hướng dẫn em trong môn học, cũng như tiểu luận này trong suốt thời gian qua.

Đối với em, môn học này vô cùng cần thiết với công việc của em trong tương lai. Với sự tâm huyết giảng dạy của thầy và lượng kiến thức to lớn, bổ ích của thầy truyền tải, em sẽ cố gắng hết sức hoàn thành bài tiểu luận này một cách tốt nhất, tuy nhiên trong quá trình làm bài khó có thể tránh khỏi những sai sót, cũng như non nớt về kinh nghiệm làm tiểu luận nên mong thầy xem xét và bỏ qua. Em mong muốn nhận được những góp ý quý giá của thầy để em có thể rút kinh nghiệm và hoàn thiện bản thân trong tương lai.

Một lần nữa, em xin cảm ơn và gửi đến quý thầy, cô lời chúc tốt đẹp nhất.

**TIỂU LUẬN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của ThS 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 luận văn 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 luận văn 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 03 tháng 11 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Mai Nguyễn Phương Trang*

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

Bài báo cáo này là bài tiểu luận giữa kì môn Xác suất và thống kê ứng dụng cho công nghệ thông tin khoá 2022 HK1 - 2023. Tiểu luận giải thích những hàm giải quyết bài toán thống kê bằng thư viện Statistics của ngôn ngữ lập trình Python, giải thích về thuật toán cân bằng biểu đồ (Histogram equalization algorithm) và thuật toán đối chiếu biểu đồ (Histogram matching algorithm).

Nội dung gồm 3 phần, cụ thể:

* Phần 1: Thư viện Statistics:
* Tổng quan về thư viện.
* Các hàm trong thư viện, giải thích về tham số đầu vào, dữ liệu trả về và cách dùng.
* Ví dụ thông qua code bằng ngôn ngữ lập trình Python.
* Phần 2: Thuận toán cân bằng biểu đồ:
* Ngữ cảnh áp dụng, điều kiện áp dụng, thuật toán áp dụng.
* Ví dụ.
* Phân tích ví dụ.
* Phần 3: Thuật toán đối chiếu biểu đồ:
* Ngữ cảnh áp dụng, điều kiện áp dụng, thuật toán áp dụng.
* Ví dụ.
* Phân tích ví dụ.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc26582)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc9363)

[TÓM TẮT iv](#_Toc20495)

[MỤC LỤC 1](#_Toc18099)

[CHƯƠNG 1 – Thư viện Statistics trong Python 6](#_Toc23784)

[1.1 Tổng quan về thư viện Statistics trong Python 6](#_Toc15257)

[1.2 Các hàm của thư viện Statistics 6](#_Toc19591)

[1.2.1](#_Toc8170) *[Các hàm về giá trị trung tâm và phương pháp tính giá trị đại diện:](#_Toc8170)* [6](#_Toc8170)

[1.2.1.1 Hàm mean(data) - giá trị trung bình cộng: 6](#_Toc11183)

[1.2.1.2 Hàm fmean(data, weights) - giá trị trung bình cộng và có trọng số: 7](#_Toc27974)

[1.2.1.3 Hàm geometric\_mean(data) - giá trị trung bình nhân: 9](#_Toc21951)

[1.2.1.4 Hàm harmonic\_mean(data, weights) 11](#_Toc16587)

[1.2.1.5 Hàm median(data) - giá trị trung vị: 12](#_Toc19885)

[1.2.1.6 Hàm median\_low(data): 13](#_Toc19105)

[1.2.1.7 Hàm median\_high(data): 14](#_Toc32747)

[1.2.1.8 Hàm median\_grouped(data, interval): 15](#_Toc22223)

[1.2.1.9 Hàm mode(data) - yếu vị của bộ số liệu: 16](#_Toc15440)

[1.2.1.10 Hàm multimode(data): 17](#_Toc8691)

[1.2.1.11 Hàm quantiles(data, n, method): 18](#_Toc28803)

[1.2.2 Các hàm tính độ lệch của dữ liệu và xu hướng phân tán của bộ dữ liệu: 20](#_Toc154)

[1.2.2.1 Không gian mẫu (Sample space): 20](#_Toc11892)

[1.2.2.2 Tổng thể (Population): 23](#_Toc22070)

[1.2.3 Thống kê đo lường mức độ tương quan giữa hai biến: 25](#_Toc10217)

[1.2.3.1 Hàm covariance(x, y) - hiệp phương sai: 25](#_Toc279)

[1.2.3.2 Hàm correlation(x, y) - Hệ số tương quan Pearson, kí hiệu: wps: 26](#_Toc20266)

[1.2.3.3 Hàm linear\_regression(x, y, proportional = False): 28](#_Toc19147)

[1.2.4 Phân phối xác suất (phân phối chuẩn - đồ thị dạng chuông) từ dữ liệu ngẫu nhiên: 30](#_Toc3661)

[1.2.4.1. Hàm normal(mu = 0.0, sigma = 1.0) 30](#_Toc10442)

[1.2.4.2. Hàm from\_samples(data): 31](#_Toc20920)

[CHƯƠNG 2 – Thuật toán cân bằng biểu đồ(histogram equalization) 33](#_Toc14187)

[2.1 Tổng quan: 33](#_Toc18382)

[2.2 Ưu điểm và nhược điểm của thuật toán: 33](#_Toc26978)

[2.2.1 Ưu điểm của thuật toán: 33](#_Toc1761)

[2.2.2 Nhược điểm của thuật toán: 33](#_Toc30816)

[2.3 Ràng buộc: 34](#_Toc5089)

[2.4 Giải thuật: 34](#_Toc18318)

[2.4.1 Tính biểu đồ histogram: 34](#_Toc22374)

[2.4.2 Cân bằng Histogram: 38](#_Toc12021)

[2.5 Ví dụ: 42](#_Toc15351)

[CHƯƠNG 3 – Thuật toán đối chiếu (histogram matching algorithm) 44](#_Toc7954)

[3.1 Tổng quan: 44](#_Toc18380)

[3.2 Ưu điểm và nhược điểm của thuật toán: 44](#_Toc17963)

[3.2.1 Ưu điểm của thuật toán: 44](#_Toc21850)

[3.2.2 Nhược điểm của thuật toán: 44](#_Toc27930)

[3.3 Ràng buộc: 45](#_Toc26976)

[3.4 Giải thuật 45](#_Toc12205)

[3.4.1 Tính biểu đồ histogram của ảnh gốc và ảnh tham chiếu: 45](#_Toc23128)

[3.4.2 So sánh và so khớp ảnh gốc với ảnh tham chiếu: 45](#_Toc5326)

[Chương 4 - Hiện thực 48](#_Toc23673)

[4.1 Thuật toán cân bằng histogram: 48](#_Toc27910)

[4.1.1 Thêm thư viện: 48](#_Toc24228)

[4.1.2 Xử lý hình ảnh đầu vào: 48](#_Toc21030)

[4.1.3 Định nghĩa hàm tính histogram: 49](#_Toc18429)

[4.1.4 Định nghĩa hàm cân bằng histogram](#_Toc14610)*[:](#_Toc14610)* [50](#_Toc14610)

[4.1.5 Định nghĩa hàm vẽ biểu đồ và hiển thị hình ảnh 51](#_Toc25337)

[4.1.6 Chạy các hàm đã định nghĩa trong main: 52](#_Toc20224)

[4.1.7 Kết quả: 52](#_Toc27174)

[4.2 Thuật toán đối chiếu histogram: 53](#_Toc26034)

[4.2.1 Thêm thư viện 53](#_Toc3072)

[4.2.2 Xử lí ảnh gốc, ảnh tham chiếu 53](#_Toc9862)

[4.2.3 Định nghĩa hàm tính histogram của ảnh gốc và ảnh tham chiếu 54](#_Toc8791)

[4.2.4 Định nghĩa hàm matching histogram 55](#_Toc13641)

[4.2.5 Chạy các hàm đã định nghĩa trong main 56](#_Toc6995)

[4.2.6 Kết quả 57](#_Toc22459)

[TÀI LIỆU THAM KHẢO 58](#_Toc6265)

[Tiếng Việt 58](#_Toc581)

[Tiếng Anh 58](#_Toc13834)

CHƯƠNG 1 – Thư viện Statistics trong Python

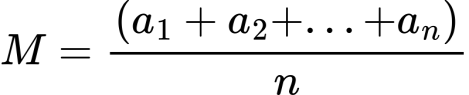
1. Tổng quan về thư viện Statistics trong Python

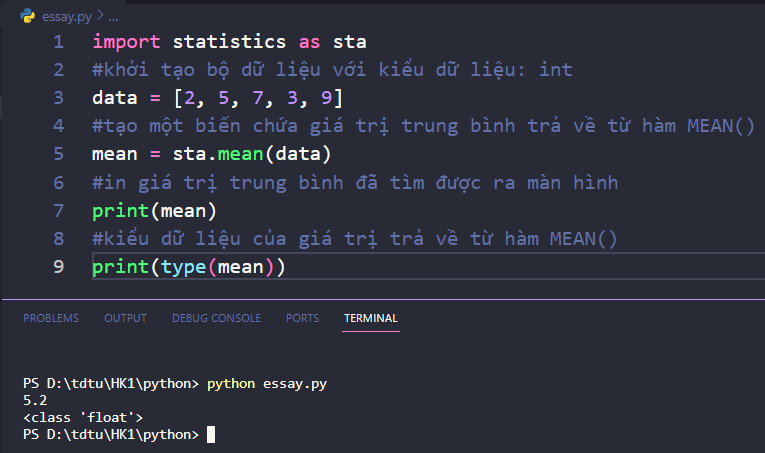
* Thư viện cung cấp những hàm dùng để giải quyết những bài toán về thống kê của bộ dữ liệu số (giá trị của bộ số liệu thuộc tập Real - số thực). Mục đích của thư viện dùng để cung cấp các thông tin cần thiết để vẽ đồ thị (giá trị trung bình, trung vị, phương sai, độ lệch chuẩn) và phân tích, đánh giá bộ số liệu một cách khoa học.
* Thư viện chỉ hỗ trợ tính toán bộ số liệu với kiểu dữ liệu: int (integer - số nguyên), float (số dấu phẩy động), decimal (số thập phân) và fraction (phấn số). Nếu bộ dữ liệu đầu vào được truyền vào dưới dạng hỗn hợp, thì ta dùng map() để xử lí số liệu đầu vào, điều đó sẽ tránh sinh ra lỗi hoặc kết quả không chính xác khi sử dụng một vài hàm trong thư viện.
* Để sử dụng thư viện ta cần nhập thư viện vào chương trình của mình:

ANH_1

Hình 1 - Thêm thư viện Statistics vào chương trình

1. Các hàm của thư viện Statistics
2. *Các hàm về giá trị trung tâm và phương pháp tính giá trị đại diện:*
3. Hàm mean(data) - giá trị trung bình cộng:

* *Giá trị trả về của hàm*: giá trị trung bình của bộ dữ liệu “data” (tham số truyền vào), giá trị trả về đa số là float (có thể trả về int), do trung bình cộng của một tập dữ liệu luôn là số thực.
* *Kiểu dữ liệu của tham số*: Data có thể là tập dữ liệu với kiểu dữ liệu đã đề cập ban đầu (int, float, decimal, fraction) và tập dữ liệu đó dưới dạng: [list](#List)[(1)](#List), [numpy array](#numpy_array)[(2)](#numpy_array), [pandas series](#pandas_series)[(3)](#pandas_series), [tuple](#tuple)[(4)](#tuple), [set](#set)[(5)](#set).
* *Giải thuật của hàm*: cộng tất cả các số của bộ dữ liệu và chia cho tổng số có trong bộ dữ liệu. Có thể gọi thân thuộc hơn là: Trung bình cộng, đây là thước đo giá trị trung bình của bộ dữ liệu.
* Nếu bộ dữ liệu “data” truyền vào là rỗng, thì ngoại lệ (exception) sẽ in lỗi ra màn hình hiển thị.
* Công thức toán học:
* M: giá trị trung bình cộng
* a1, a2, an: giá trị biến
* n: số lượng giá trị biến
* *Ví dụ*:

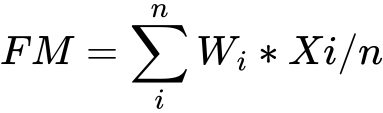


Hình 2 - Ví dụ về hàm mean(data)

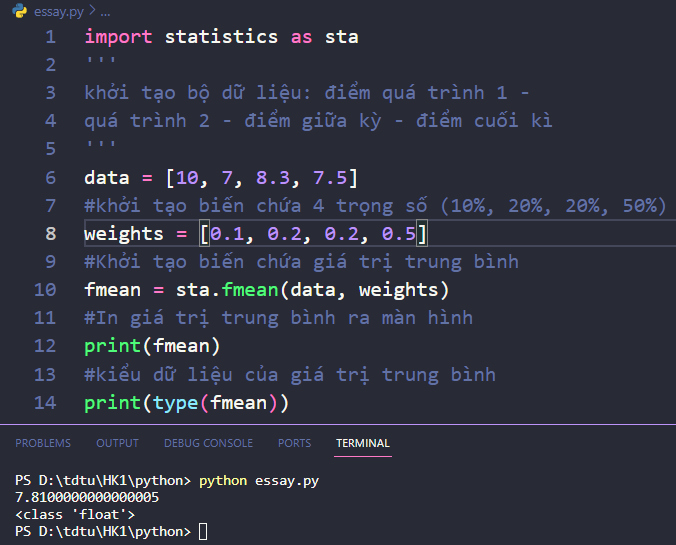
1. Hàm fmean(data, weights) - giá trị trung bình cộng và có trọng số:

\* Giátrị trung bình (mean) dễ bị ảnh hưởng bởi các giá trị có giá trị quá thấp hoặc quá cao, sẽ làm giá trị trung bình không phải là giá trị đại diện cho bộ dữ liệu. Nên thư viện hỗ trợ thêm hàm fmean(), với fmean ta sẽ tính được giá trị trung bình chính xác hơn vì có thêm tham số là [trọng số](#trọng_số)[(6)](#trọng_số).

* *Giá trị trả về của hàm*: trả về giá trị trung bình của bộ số liệu, giá trị trả về luôn có KDL là floats.
* *Giải thuật hàm:* Nhân từng trọng số tại vị trí ứng với vị trí của số trong bộ dữ liệu data. Sau đó cộng lại các kết quả vừa tính rồi chia cho số lượng dữ liệu có trong data. Vì có trọng số, nên kết quả có thể là giá trị đại diện cho bộ số liệu.
* Nếu bộ dữ liệu “data” truyền vào là rỗng, thì ngoại lệ (exception) sẽ in lỗi ra màn hình hiển thị; nếu trọng số “weights” truyền vào không đủ hoặc thừa số lượng so với dữ liệu trong data, thì ngoại lệ (exception) sẽ in lỗi ra màn hình.



* *Công thức toán học*:
* FM: giá trị trung bình cộng có trọng số
* W: trọng số
* X: giá trị biến
* n: số lượng giá trị biến
* *Ví dụ*:



Hình 3 - Ví dụ về hàm fmean(data, weights)



Hình 4 - Ví dụ về các trường hợp của trọng số trong hàm fmean

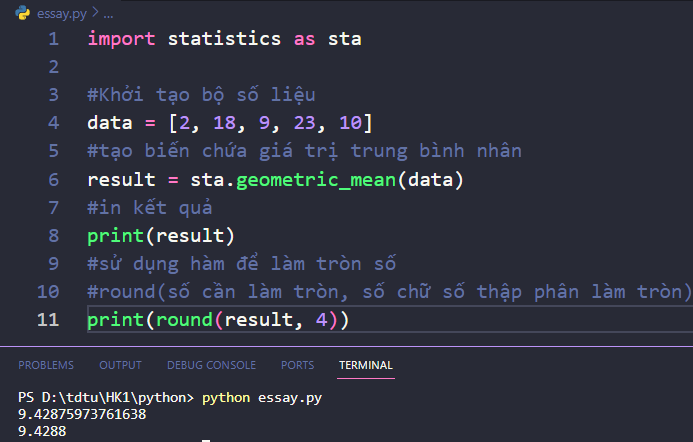
→ Từ hình 4: ta thấy được việc set “weights = None” hay để default trong hàm fmean() thì kết quả trả về tương tự với hàm mean().

* *Kiểu dữ liệu của tham số*: đối với data [(tương tự của hàm M)](#KDL_DATA), còn đối với weights (trọng số) thường là kiểu dữ liệu floats (hoặc int) và độ dài của trọng số phải bằng độ dài của số lượng giá trị có trong data (trọng số phải khác 0), nếu cho weights = none thì hàm fmean(data, weights = None) sẽ trả về giá trị tương tự khi sử dụng hàm mean(data).

1. Hàm geometric\_mean(data) - giá trị trung bình nhân:

\*Khác với hàm mean hay hàm fmean, hàm geometric mean tính giá trị trung bình thông qua phép nhân thay vì phép cộng. Đây chính là giá trị đại diện cho bộ dữ liệu.

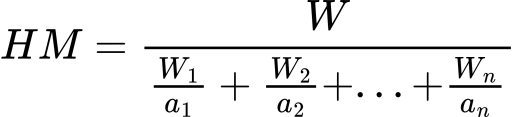
* *Giá trị trả về của hàm*: giá trị trung bình nhân và kết quả luôn có kiểu dữ liệu floats.
* *Kiểu dữ liệu của tham số*: [data tương tự như hàm M](#KDL_DATA) (nhưng hàm sẽ chuyển đổi data thành floats).
* *Giải thuật hàm*: căn bậc n của tích n số (với n là số lượng số liệu có trong data).
* *wpsCông thức toán học*:
* GM: trung bình nhân
* a1, a2, a3, an: giá trị biến
* n: số lượng giá trị biến
* Nếu bộ dữ liệu “data” truyền vào là rỗng hoặc chứa giá trị = 0 hoặc giá trị âm, thì ngoại lệ (exception) sẽ in lỗi ra màn hình hiển thị.
* Ví dụ:



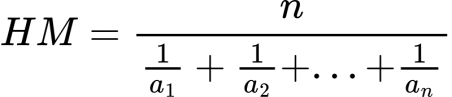
Hình 5 - Ví dụ về hàm geometric\_mean(data)

1. Hàm harmonic\_mean(data, weights)

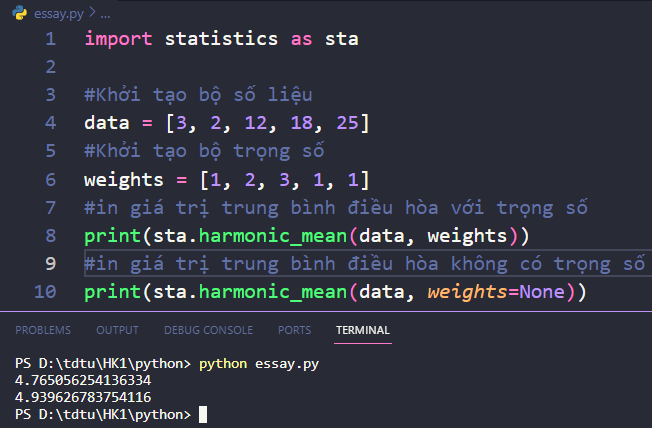
\* Đây là hàm tính trung bình điều hòa, thường được dùng để tìm giá trị trung bình của các biến được biểu diễn bởi tỉ số của hai giá trị có 2 đơn vị đo khác nhau như tốc độ (km/h),…

* *Giá trị trả về của hàm*: giá trị trung bình điều hòa của bộ số liệu, với kiểu dữ liệu floats.
* *Kiểu dữ liệu của tham số*: [đối với data tương tự của hàm M](#KDL_DATA), đối với weights thường có KDL là floats và cần phải nhập số lượng weights bằng với số lượng giá trị có trong data (weights phải khác 0).
* *Giải thuật của hàm*: nghịch đảo của trung bình cộng các giá trị nghịch đảo của dãy số.
* *Công thức toán học*:
* HM: harmonic mean
* a1, a2, a3, an: giá trị biến
* n: số lượng giá trị biến
* W1, W2, W3, Wn: giá trị trọng số
* W: tổng giá trị các trọng số

\* Nếu weights = None (không có trọng số) thì công thức toán học:

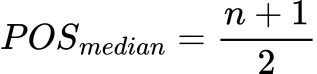


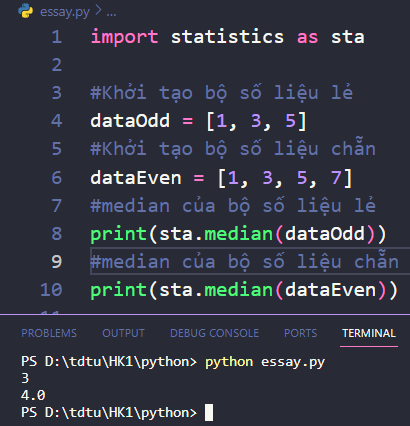
* Nếu bộ dữ liệu “data” truyền vào là rỗng, thì ngoại lệ (exception) sẽ in lỗi ra màn hình hiển thị; nếu trọng số “weights” truyền vào không đủ hoặc thừa số lượng so với dữ liệu trong data, thì ngoại lệ (exception) sẽ in lỗi ra màn hình.
* *Ví dụ:*

**

Hình 6 - Ví dụ về hàm haromonic\_mean(data, weights)

1. Hàm median(data) - giá trị trung vị:

* *Giá trị trả về của hàm*: giá trị trung vị của bộ số liệu, thường được gọi là giá trị trung bình giữa hai phần, với kiểu dữ liệu thường gặp là floats.
* *Kiểu dữ liệu của tham số*: [data tương tự như hàm m](#KDL_DATA)ean, đặc biệt là giá trị của các số liệu trong data phải tuần tự.
* *Giải thuật của hàm*: nếu số lượng giá trị của bộ số liệu là lẻ thì lấy giá trị của số ở vị trí giữa (ví dụ: bộ số liệu bao gồm: 1, 3, 5 thì median = 3), ngược lại lấy giá trị trung bình của cặp số ở vị trí giữa (ví dụ: bộ số liệu 1, 3, 5, 7 thì median = 4, vị trí giữa là ở giữa số 3 và 5, thì ta lấy giá trị ở giữa 3 và 5 là 4), với giá trị trung bình không nằm trong bộ số liệu thì phương thức này không phù hợp với việc tìm giá trị đại diện thực tế. Có thể nói rằng median là giá trị nằm ở vị trí trung tâm của dãy số tăng dần.
* *Công thức toán học để tìm vị trí của median trong dãy số*:
* POSmedian: vị trí của median trong dãy số
* n: số lượng giá trị có trong dãy số
* *Ví dụ*:



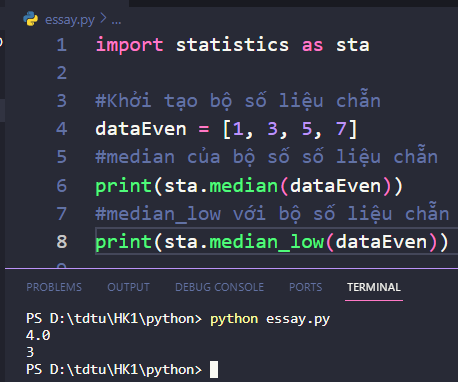
Hình 7 - Ví dụ về hàm median(data)

* Nếu bộ dữ liệu “data” truyền vào là rỗng, thì ngoại lệ (exception: StatisticsError) sẽ in lỗi ra màn hình hiển thị.

1. Hàm median\_low(data):

\*Hàm median\_low tương tự hàm median nhưng khác ở giải thuật với số lượng giá trị của bộ số liệu chẵn. Hàm sẽ chọn giá trị nhỏ hơn trong cặp số ở vị trí trung tâm thay vì tính trung bình cộng. Như vậy hàm sẽ khắc phục tình trạng: giá trị trung vị không nằm trong bộ số liệu.

* *Ví dụ*:

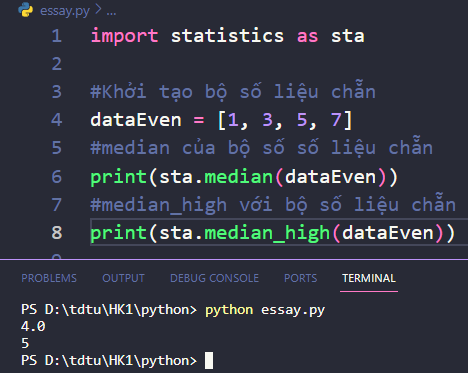


Hình 8 - Ví dụ về hàm median\_low(data)

1. Hàm median\_high(data):

\*Hàm median\_high cũng tương tự như hàm median, nhưng khác ở giải thuật với số lượng giá trị của bộ số liệu chẵn. Hàm sẽ chọn giá trị lớn hơn trong cặp số ở vị trí trung tâm thay vì tính trung bình cộng. Như vậy hàm sẽ khắc phục tình trạng: giá trị trung vị không nằm trong bộ số liệu.

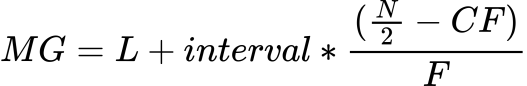
* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.
* *Ví* dụ:

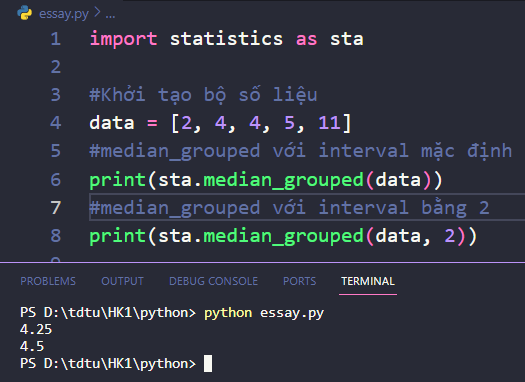


Hình 9 - Ví dụ về hàm median\_high(data)

1. Hàm median\_grouped(data, interval):

\* Các hàm median(), median\_low(), median\_high() tính trung vị cho dữ liệu rời rạc, vậy hàm median\_grouped() dùng để tính trung vị cho dữ liệu liên tục.

* *Giá trị trả về của hàm:* giá trị trung vị của dữ liệu, kiểu dữ liệu thường là floats.
* *Kiểu dữ liệu của tham số:* đối với kiểu dữ liệu của data thường là int, float, string… được truyền vào dưới dạng [list](#List)[(1)](#List), [numpy array](#numpy_array)[(2)](#numpy_array), [pandas series](#pandas_series)[(3)](#pandas_series), [tuple](#tuple)[(4)](#tuple), [set](#set)[(5)](#set); đối với kiểu dữ liệu của interval thường là floats hoặc int, interval mặc định là 1.
* *Công thức toán học:*
* MG: medium grouped
* L: giới hạn dưới trong khoảng trung vị của tập dữ liệu
* Interval: bước nhảy
* N: số lượng phần tử trong tập dữ liệu
* CF: giá trị thứ (N - 1) / 2 trong tập dữ liệu
* F: số lượng giá trị trong tập dữ liệu bằng CF
* *Ví dụ*:



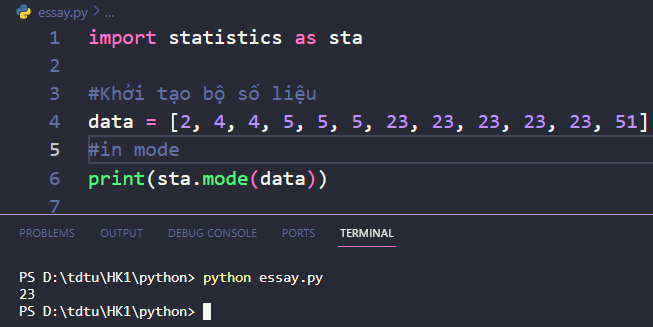
Hình 10 - Ví dụ của hàm median\_grouped(data)

* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.

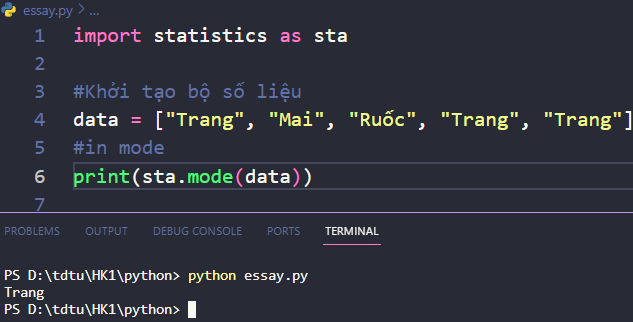
1. Hàm mode(data) - yếu vị của bộ số liệu:

\*Mode hay còn gọi là yếu vị, đây chính là hàm dùng để tìm số được lặp lại nhiều nhất trong bộ số liệu, ứng dụng trong việc tìm ra giá trị phổ biến nhất.

* *Giá trị trả về của hàm:* trả về số được lặp lại nhiều nhất, kiểu dữ liệu của mode phụ thuộc vào kiểu dữ liệu của data.
* *Kiểu dữ liệu của tham số:* [tương tự như hàm M](#KDL_DATA)ean, đặc biệt hàm có thể sử dụng cho kiểu dữ liệu là strings.
* *Giải thuật:* đếm các phần tử có giá trị bằng nhau, yếu vị là phần tử có số lần lặp lại nhiều nhất.
* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.
* *Ví dụ:*



Hình 11 - Ví dụ về hàm mode(data)



Hình 12 - Ví dụ về hàm mode(data) với strings

1. Hàm multimode(data):

\*Với hàm mode(data) sẽ chỉ trả về giá trị của phần tử có số lượng nhiều nhất đầu tiên, những phần tử khác được lặp lại với số lượng tương tự sẽ không được trả về. Để giải quyết về đề này, ta dùng hàm multimode(data).Với các thông số sẽ tương tự hàm mode, *nhưng khác ở giá trị trả về*: tất cả phần tử có số lượng lặp lại nhiều nhất, kiểu dữ liệu phụ thuộc vào kiểu dữ liệu của data.

* *Ví dụ:*

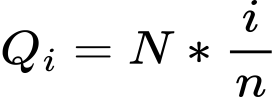
**

Hình 13 - Ví dụ về hàm multimode(data)

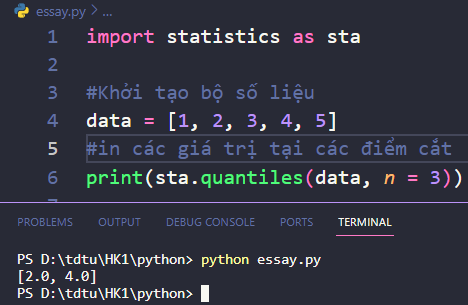
1. Hàm quantiles(data, n, method):

\* Quantiles là các điểm cắt, cắt data bởi (n - 1) điểm cắt thành n đoạn có xác suất bằng nhau.

* *Giá trị trả về của hàm:* trả về (n - 1) giá trị tại điểm cắt, với kiểu dữ liệu là floats.
* *Kiểu dữ liệu của tham số:* [data tương tự như hàm M](#KDL_DATA) và dữ liệu trong data phải tuần tự, với n có kiểu dữ liệu là int (n > 1) và n thường là 4, 10, 100, method mặc định là exclusive.
* *Công thức*: số n được truyền vào là số phần muốn chia, sẽ có n - 1 điểm cắt. Giá trị tại mỗi điểm cắt được tính theo công thức:



* Qi: điểm cắt thứ i (i < n)
* N: số lượng phần tử trong tập dữ liệu
* n: phần dữ liệu được chia
* *Ví dụ*:



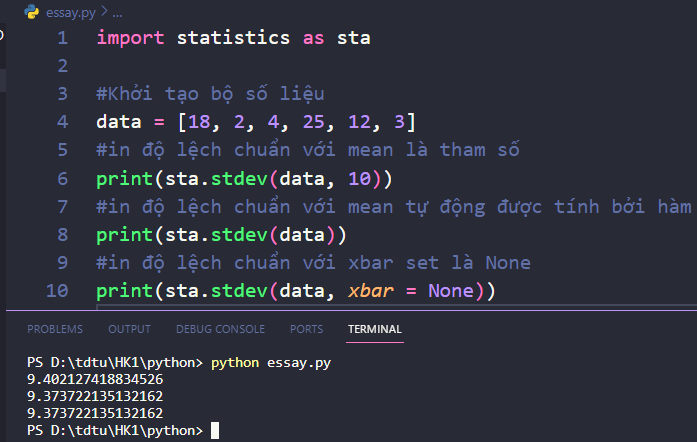
Hình 14 - Ví dụ về hàm quantiles(data, n)

**\*Chú thích:**

1. *List*: là một trong 5 kiểu dữ liệu chuẩn trong python, là dạng dữ liệu cho phép lưu trữ nhiều kiểu dữ liệu khác nhau và list có thể xem là mảng tuần tự.
2. *Numpy array*: là mảng có n chiều theo dạng ma trận, các phần tử phải có cùng kiểu dữ liệu với nhau (thường là số).
3. *Pandas series*: là đối tượng giống mảng một chiều, có thể chứa nhiều kiểu dữ liệu trong mảng. Nó được xem là một cấu trúc dữ liệu với 2 mảng: một để lưu chỉ mục, còn lại lưu dữ liệu.
4. *Tuple*: dùng để lưu trữ một tập hợp các giá trị tuần tự và bất biến. Các giá trị trong tuple có kiểu dữ liệu: số, chuỗi, danh sách, và thậm chí là các tuple khác.
5. *Set*: dùng để lưu trữ một tập hợp các giá trị tuần tự và không lặp lại. Các đối tượng trong set có thể là bất kỳ kiểu dữ liệu nào, bao gồm số, chuỗi, danh sách, và thậm chí là các set khác.
6. *Trọng số*: gọi chính xác hơn là trung bình trọng số, tức là một giá trị trung bình được dùng để tính toán, giá trị đó là mức độ quan trọng, tần suất sử dụng của số liệu.
7. Các hàm tính độ lệch của dữ liệu và xu hướng phân tán của bộ dữ liệu:
8. Không gian mẫu (Sample space):

**\* Độ lệch chuẩn (standard deviation) - kí hiệu: s - stdev(data, xbar = None):**

* *Giá trị trả về của hàm*: độ lệch chuẩn của mẫu, kiểu dữ liệu là floats.
* *Kiểu dữ liệu của tham số*: data là một tập hợp được gọi là không gian mẫu, với các giá trị tuần tự và kiểu dữ liệu có thể là int, floats, decimal, fractions,…; xbar là giá trị trung bình mà người dùng nhập vào, nếu không truyền hoặc set None thì python sẽ tự động tính (theo hàm Mean).
* *Công thức toán học: *
* s: độ lệch chuẩn của mẫu
* Xi: giá trị của điểm i trong tập dữ liệu
* n: tổng số quan sát trong tập dữ liệu
* N: được tính trong tổng xích ma của xi
* : giá trị của tập dữ liệu
* *Ví dụ*:

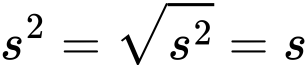


Hình 15 - Ví dụ về hàm stdev(data, xbar)

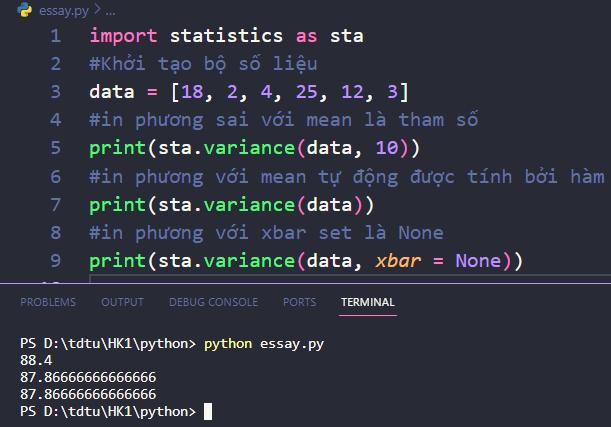
→Vậy option: xbar là default hoặc cài đặt bằng None, hàm sẽ tự tính mean và cho kết quả như nhau, còn khi cài đặt với mean theo ý mình thì quá trình tính sẽ thay mean của bộ dữ liệu bằng giá trị đầu vào, từ đó cho một kết quả khác.

* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.

**\*Phương sai (variance) - kí hiệu : C:/Users/PHUONG TRANG/AppData/Local/Temp/wps.DjqEiTwps- variance(data, xbar = None):**

Phương sai là thước đo độ biến thiên của dữ liệu. Một phương sai có giá trị nhỏ cho thấy các giá trị của phần tử biến thiên xung quanh giá trị trung bình. Phương sai có thể được tính bằng cách bình phương độ lệch chuẩn, thông thường người ta sẽ tính phương sai trước rồi lấy căn bậc hai để tìm được độ lệch chuẩn: 

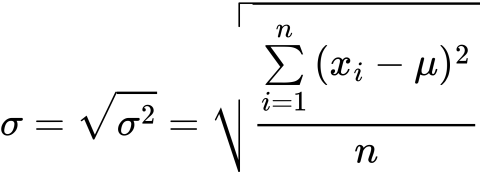
* *Giá trị trả về của hàm*: Phương sai của mẫu (bình phương độ lệch chuẩn), KDL thường là floats.
* *Kiểu dữ liệu của tham số*: data là một tập hợp được gọi là không gian mẫu, kiểu dữ liệu có thể là int, floats, decimal, fractions,…; xbar là giá trị trung bình (M) mà người dùng nhập vào, nếu không truyền hoặc set None thì hàm tự động tính (theo công thức của hàm Mean).
* *Công thức toán học*: **
* s2: độ lệch chuẩn của mẫu
* Xi: giá trị của điểm i trong tập dữ liệu
* n: tổng số quan sát trong tập dữ liệu
* N: được tính trong tổng xích ma của xi
* : giá trị của tập dữ liệu
* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.
* *Ví dụ:*



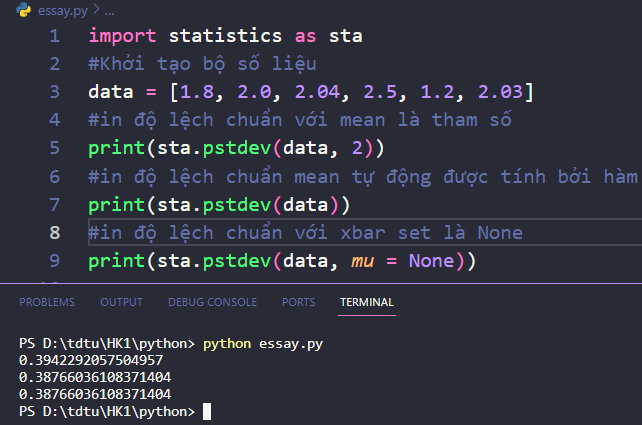
Hình 16 - Ví dụ về hàm variance(data, xbar)

1. Tổng thể (Population):

**\*Độ lệch chuẩn của tổng thể - Kí hiệu: wps/std(X) - pstdev(data, mu = None):**

* *Giá trị trả về của hàm*: độ lệch chuẩn của tổng thể, kiểu dữ liệu là floats.
* *Kiểu dữ liệu của tham số*: data là một tập hợp được gọi là không gian mẫu, với các giá trị tuần tự và kiểu dữ liệu có thể là int, floats, decimal, fractions,…; mu là giá trị trung bình mà người dùng nhập vào hoặc không nhập thì nó là giá trị trung bình của tổng thể.
* *Công thức toán học:*

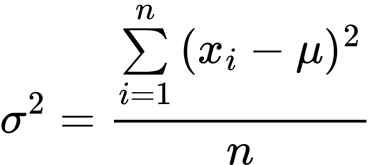
* wps: độ lệch chuẩn của tổng thể
* xi: giá trị của điểm i trong tổng thể
* n: tổng số quan sát trong tổng thể
* wps: giá trị trung bình của tổng thể
* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.
* *Ví dụ*:

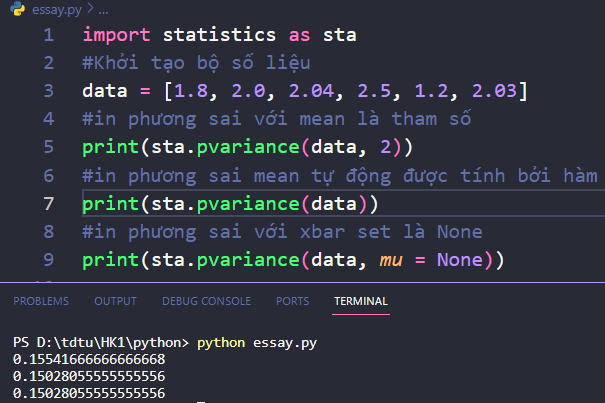


Hình 17 - Ví dụ về hàm pstdev(data, mu)

**\*Phương sai của tổng thể - Kí hiệu: C:/Users/PHUONG TRANG/AppData/Local/Temp/wps.ejxjSnwps/var(X) - pvariance(data, mu = None):**

Phương sai là thước đo độ biến thiên của dữ liệu. Một phương sai có giá trị nhỏ cho thấy các giá trị của phần tử biến thiên xung quanh giá trị trung bình. Phương sai có thể được tính bằng cách bình phương độ lệch chuẩn, thông thường người ta sẽ tính phương sai trước rồi lấy căn bậc hai để tìm được độ lệch chuẩn: 

* *Giá trị trả về của hàm*: Phương sai của tổng thể (bình phương độ lệch chuẩn), kiểu dữ liệu thường là floats.
* *Kiểu dữ liệu của tham số*: data là một tập hợp được gọi là tổng thể(thuộc và nhỏ hơn không gian mẫu) không rỗng và có thể lặp lại dữ liệu, kiểu dữ liệu có thể là int, floats, decimal, fractions,…; mu là giá trị trung bình (Mean) mà người dùng nhập vào, nếu không truyền hoặc set None thì hàm tự động tính (theo công thức của hàm Mean).
* Nếu nhập dữ liệu rỗng, ngoại lệ (exceptions: StatisticsError) sẽ in lỗi ra màn hình.
* *Công thức toán học:*
* C:/Users/PHUONG TRANG/AppData/Local/Temp/wps.OgLirrwps: phương sai của tổng thể
* xi: giá trị của điểm i trong tổng thể
* n: tổng số quan sát trong tổng thể
* wps: giá trị trung bình của tổng thể
* *Ví dụ:*

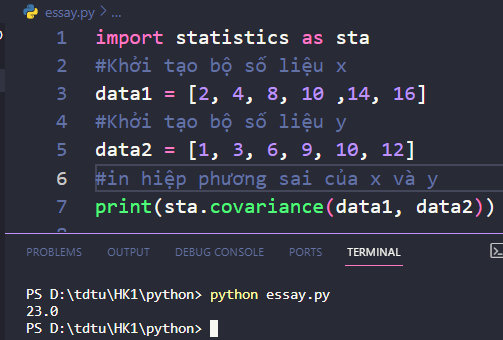


Hình 18 - Ví dụ về hàm pvariance(data, mu)

1. Thống kê đo lường mức độ tương quan giữa hai biến:
2. Hàm covariance(x, y) - hiệp phương sai:

\*Hiệp phương sai là một đại lượng thống kê đo lường mức độ tương quan tuyến tính giữa hai biến - kí hiệu: cov.

* *Giá trị trả về của hàm:* hiệp phương sai của hai tập hợp, kiểu dữ liệu thường là floats.
* cov > 0: 2 tập dữ liệu có cùng xu hướng
* cov < 0: 2 tập dữ liệu có xu hướng ngược chiều nhau.
* cov = 0: 2 tập dữ liệu không có quan hệ tuyến tính nào.
* *Kiểu dữ liệu của tham số*: x và y là một tập hợp với kiểu dữ liệu có thể là int, floats, decimal, fractions,…Quan trọng tập x và y phải cùng độ dài.
* *Công thức toán học*: 
* covx,y: hiệp phương sai của 2 tập hợp x và y
* xi,yi: giá trị ở vị trí i tương ứng với tập x và y
* : giá trị trung bình của tập x và y
* n: độ dài của tập giá trị
* *Ví dụ:*

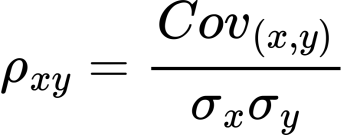


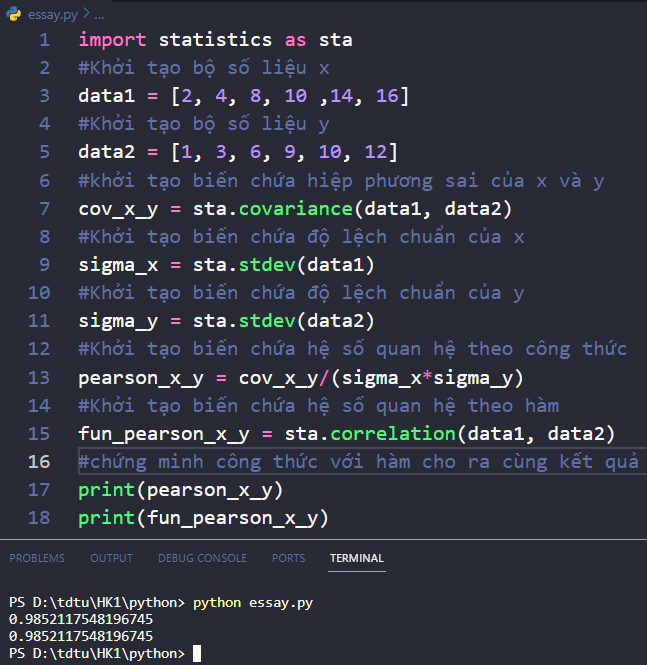
Hình 19 - Ví dụ của hàm covariance(x, y)

* Nếu nhập x hoặc y rỗng hay số lượng dữ liệu của 2 tập hợp không bằng nhau, thì ngoại lệ (exception: StatisticsError) sẽ in lỗi ra màn hình hiển thị.

1. Hàm correlation(x, y) - Hệ số tương quan Pearson, kí hiệu: wps:

\*Hàm trên tính toán hệ số tương quan giữa hai tập hợp, để chỉ ra mối quan hệ chặt chẽ giữa hai tập hợp, hoặc không có mối quan hệ giữa chúng.

* *Giá trị trả về của hàm*: hệ số tương quan (hệ số thuộc khoảng (-1, +1)), KDL là floats.
* *Kiểu dữ liệu của tham số*: Hàm có 2 tham số bắt buộc là 2 tập hợp x và y, trong đó x và y phải cùng độ dài và không nhỏ hơn 2. Chúng thường có kiểu dữ liệu là int, floats, decimal, fractions,…
* *Công thức toán học*:
* wps: hệ số tương quan Pearson
* Cov(x,y): Hiệp phương sai của x và y
* wps: độ lệch chuẩn của x
* wps: độ lệch chuẩn của y
* *Ví dụ:*

:

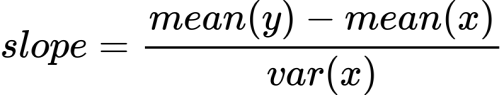
Hình 20 - Ví dụ về hàm correlation(x, y)

1. Hàm linear\_regression(x, y, proportional = False):

\*Hàm trên dùng để tính độ dốc và độ bù của một đường hồi quy tuyến tính dựa trên tham số x và y.

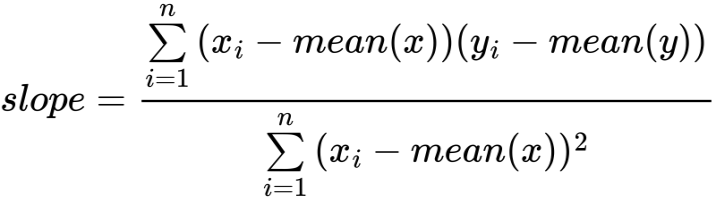
* *Giá trị trả về của hàm*:hàm sẽ trả về 2 giá trị lần lượt là độ dốc và độ bù, kiểu dữ liệu là floats.
* *Kiểu dữ liệu của tham số*: Hàm có 2 tham số bắt buộc là 2 tập hợp x và y, trong đó x và y phải cùng độ dài và không nhỏ hơn 2.Chúng thường có kiểu dữ liệu là int, floats, decimal, fractions,…Ngoài ra, tham số proportional là option giữa True và False. Với True, độ dốc và độ bù sẽ phụ thuộc vào tỷ lệ với phạm vi dữ liệu.
* *Công thức toán học*:

\*Với proportional = True:

wps

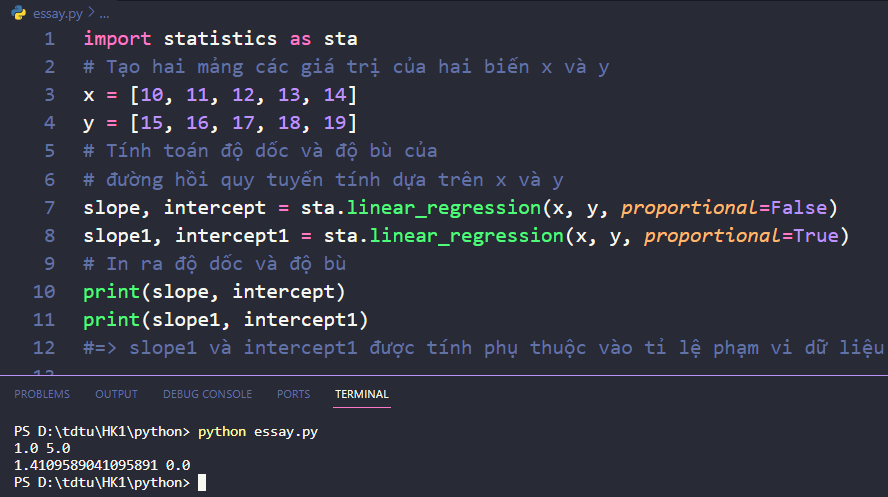
* Slope: độ dốc
* Intercept: độ bù
* Mean(y), mean(x): giá trị trung bình của tập hợp x và y.
* Var(x): phương sai của x

\*Với proportional = False:



C:/Users/PHUONG TRANG/AppData/Local/Temp/wps.AQFZHcwps

* Slope: độ dốc
* Intercept: độ bù
* Mean(y), mean(x): giá trị trung bình của tập hợp x và y.
* *Ví dụ:*



Hình 21 - Ví dụ về hàm linear\_regression(x, y, proportional)

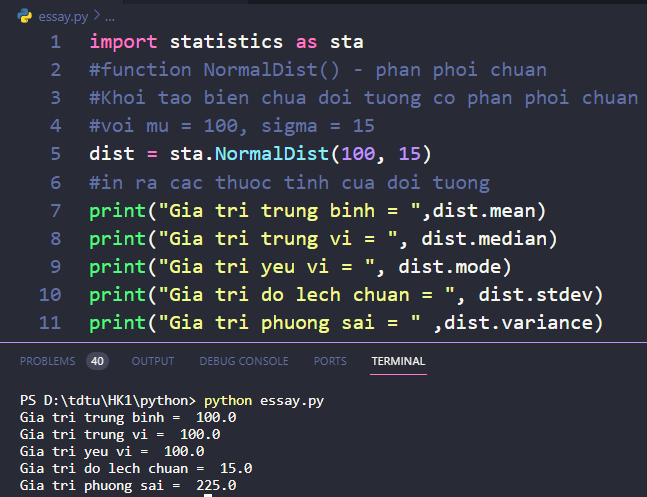
* Nếu nhập x hoặc y rỗng hay số lượng dữ liệu của 2 tập hợp không bằng nhau, thì ngoại lệ (exception: StatisticsError) sẽ in lỗi ra màn hình hiển thị.

1. Phân phối xác suất (phân phối chuẩn - đồ thị dạng chuông) từ dữ liệu ngẫu nhiên:
2. Hàm normal(mu = 0.0, sigma = 1.0)

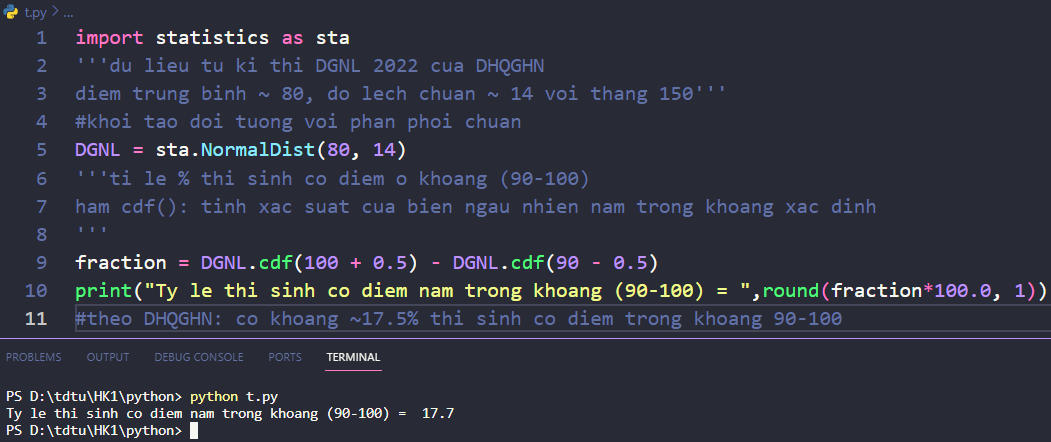
\*Phân phối chuẩn sẽ xử lí giá trị trung bình và độ lệch chuẩn của dữ liệu để tạo ra mẫu dữ liệu ngẫu nhiên, mô hình hóa dữ liệu thực tế hay tính các xác suất và thống kê liên quan đến phân phối chuẩn.

* *Giá trị trả về từ hàm*: đối tượng phân phối chuẩn mới đại diện cho mu và sigma gồm có 5 thuộc tính lần lượt là mean (giá trị trung bình), median(trung vị) , mode(yếu vị), stdev(độ lệch chuẩn) và variance(phương sai); các thuộc tính trên thuộc vào phân phối chuẩn của trung bình và độ lệch chuẩn đã truyền ban đầu.
* *Kiểu dữ liệu của tham số*: mu và sigma thường có KDL là floats (sigma > 0).

* *Ví dụ*:



Hình 22 - Ví dụ về hàm NormalDist(mu, sigma)



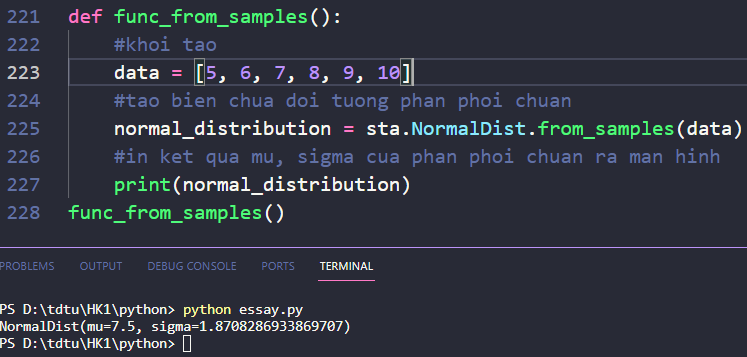
Hình 23 - Ví dụ về ứng dụng của hàm NormalDist(mu, sigma)

\*Tuy không có dữ liệu cụ thể, nhưng vẫn có thể tính gần đúng tỉ lệ phân bố xác suất dựa vào hàm NormalDist so với bộ dữ liệu gốc.

1. Hàm from\_samples(data):

\*Hàm dùng để tạo một đối tượng phân phối xác suất mới (phân phối chuẩn) từ mẫu (sample).

* *Giá trị trả về từ hàm:* với đối tượng phân phối xác suất sẽ có thể trả về các thuộc tính: name(tên phân phối xác suất), mu, sigma, mean (giá trị trung bình), std\_dev(độ lệch chuẩn), median (trung vị), mode (yếu vị), min(giá trị nhỏ nhất), max(giá trị lớn nhất).
* *Kiểu dữ liệu của tham số:* data là một danh sách các giá trị dữ liệu được dùng cho đối tượng phân phối xác suất.
* *Ví dụ:*



Hình 24 - Ví dụ hàm from\_samples(data)

CHƯƠNG 2 – Thuật toán cân bằng biểu đồ(histogram equalization)

1. **Tổng quan**:

Thuật toán cân bằng biểu đồ là một thuật toán dùng để điều chỉnh độ tương phản của ảnh (điều chỉnh cường độ màu sắc tại các điểm ảnh) giúp dễ phân biệt các đối tượng trong ảnh. Thuật toán sẽ có vài nhược điểm như khi tăng độ tương phản thì các điểm màu pixel sẽ trở nên bão hòa dẫn đến tình trạng mất vài chi tiết trong ảnh. Để khắc phục tình trạng này nhà phát triển đã mở rộng thêm thuật toán: *Histogram equalization tự nhiên:* giảm thiểu tình trạng mất chi tiết;thuật toán *Histogram equalization cục bộ:* xử lí các vùng cục bộ và sẽ không có rủi ro về mất chi tiết. Tuy nhiên, dưới đây chỉ phân tích thuật toán cân bằng biểu đồ cơ bản (histogram equalization).

1. **Ưu điểm và nhược điểm của thuật toán:**
2. Ưu điểm của thuật toán:

* Cải thiện độ tương phản: các mức xám với hình ảnh có độ tương phản thấp thường tập trung ở các mức tối và sáng khiến các chi tiết khó nhìn rõ, khi áp dụng thuật toán cân bằng các giá trị màu dàn đều từ giá trị [0,255] nên các chi tiết trong ảnh dễ nhìn thấy hơn.
* Giảm nhiễu ảnh: nhiễu ảnh là những điểm mờ có trong ảnh, điều này làm giảm chất lượng của hình ảnh và khiến người nhìn khó phân biệt các chi tiết do các nét bị điểm mờ làm ảnh hưởng, khi áp dụng thuật toán cân bằng làm tăng cường độ tương phản làm nổi bật chi tiết lên nên che giấu được các điểm nhiễu của ảnh.
* Tăng cường khả năng nhận dạng đối tượng trong hình ảnh: Khi áp dụng thuật toán, các chi tiết được làm nổi bật nên dễ dàng phân biệt với nhau, hoặc so với nền.

1. Nhược điểm của thuật toán:

* Mất các chi tiết nhỏ: với ảnh có nhiều chi tiết quá nhỏ thì khi áp dụng thuật toán chúng sẽ khó nhìn thấy hơn ban đầu, do thuật toán thường tăng những mức xám trung bình.
* Thay đổi màu sắc: histogram equalization có thể làm thay đổi màu sắc của ảnh, khác so với ảnh ban đầu. Do thuật toán điều hòa và phân bố đều các giá trị màu, khiến màu sắc của ảnh bị chênh lệch nhiều.

1. **Ràng buộc:**

* Hình ảnh đầu vào phải đạt tiêu chuẩn về độ phân giải (nếu độ phân giải quá thấp, thuật toán sẽ không đem lại kết quả hiệu quả, trường hợp xấu hơn hình ảnh sẽ biến mất).
* Hình ảnh đầu vào chưa có độ tương phản quá cao (nếu độ tương phản quá cao, thuật toán sẽ khiến cho hình ảnh hầu như mất đi sắc độ - bão hòa màu sắc, dẫn đến hình ảnh bị mất đi nhiều chi tiết và cho ra hình ảnh đầu ra không đúng mục đích).
* Khi sử dụng thuật toán, phải chấp nhận những rủi ra mà thuật toán mang đến: một vài chi tiết bị thiếu mất, thay đổi màu sắc cũng như các góc cạnh bị làm mờ - có thể nói rằng chất lượng hình ảnh có thể giảm.

1. **Giải thuật:**
2. Tính biểu đồ histogram:

\*Histogram: là biểu đồ thể hiện số lượng điểm ảnh và mức độ sáng tối của ảnh, trong đó chiều ngang (x) thể hiện cường độ ánh sáng, chiều dọc (y) thể hiện số lượng ảnh có độ sáng.

\*Sử dụng thư viện OpenCV(1) được dùng trong python:

* *Bước 1:* Từ hình ảnh xám lấy ra các giá trị màu tại các điểm pixel.
* Đầu tiên, tìm kích thước của ảnh bằng hàm img.shape; với shape[0] = height (chiều cao của ảnh), shape[1] = width (chiều rộng của ảnh).
* Tiếp theo, dùng hàm img[hi, wi] để tìm giá trị tại điểm i. Từ đó ta có ma trận heightxwidth giá trị màu của ảnh màu xám và các giá trị này thuộc [0, 255].
* VD: Với ảnh có kích thước 3x3 thì sẽ có 9 ô pixel biểu diễn giá trị màu.

|  |  |  |
| --- | --- | --- |
| 5 | 6 | 1 |
| 5 | 3 | 2 |
| 6 | 3 | 3 |

Bảng 1 - Bảng dữ liệu màu xám 3x3

* *Bước 2:* Từ dữ liệu trên, tính phân phối tích luỹ của giá trị màu (PDF).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ri | 1 | 2 | 3 | 5 | 6 |
| h(ri) | 1 | 1 | 3 | 2 | 2 |
| hi/n =P(ri) | 1/9 | 1/9 | 1/3 | 2/9 | 2/9 |

Bảng 2 - Bảng chuẩn hóa dữ liệu màu

Bảng 3 - Biểu đồ mức xám

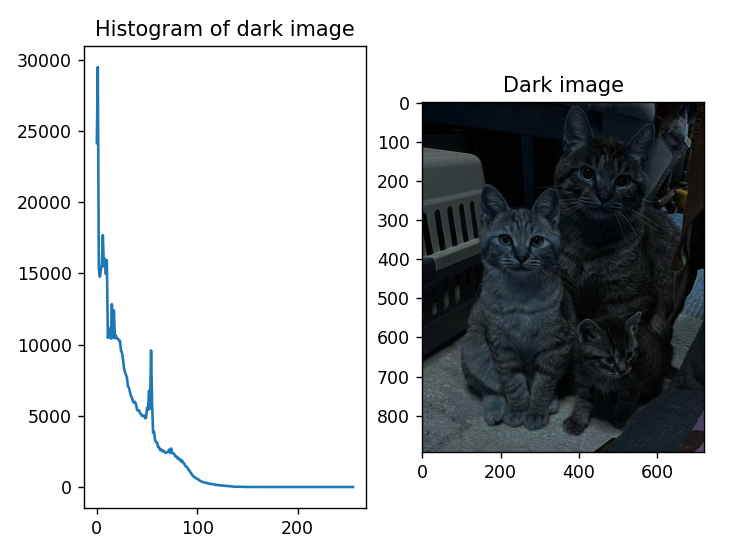
* ri: mức xám của ảnh 3x3
* ni: số điểm ảnh có giá trị ri
* Với biểu đồ mức xám chưa chuẩn hóa: h(ri) = ni với i [0,L), L là số mức xám L thuộc [0,255].
* Với biểu đồ mức xám chuẩn hóa: P(ri) = h(ri)/MN , M = height (chiều dài của ảnh), N = width (chiều rộng của ảnh),

\*Đầu tiên: từ bảng 1, ta sẽ lấy ra giá trị màu của tất cả pixel có thể có, gọi giá trị màu là ri. Tiếp theo, đếm số lần lặp lại của giá trị màu đã tìm ni, gọi số lần lặp lại của ri là h(ri) (chưa chuẩn hóa). Từ đó, tính xác suất xuất hiện của giá trị màu pixel, gọi xác suất của pixel là p(ri) (đã chuẩn hóa). Theo ví dụ trên ta có: Từ đó, ta có công thức tính p(ri) = ri/MN (MN: tổng số pixel).

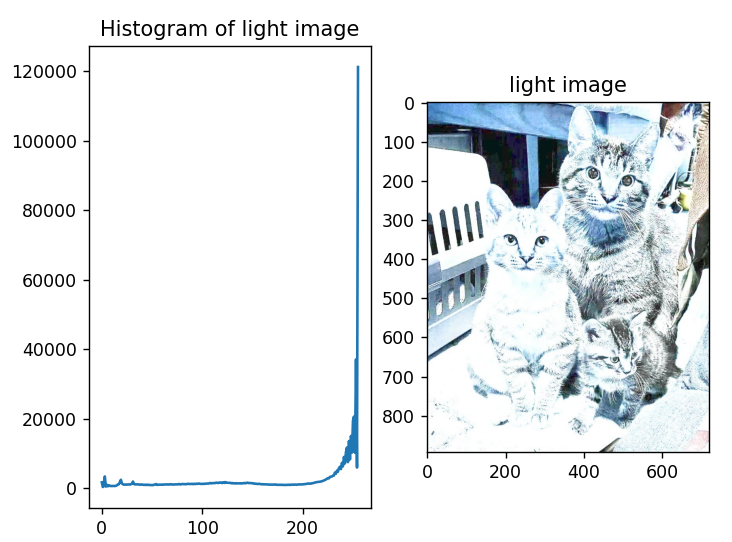
* *Bước 3:* Vẽ histogram từ các giá trị trên:

Bảng 4 - Lược đồ histogram

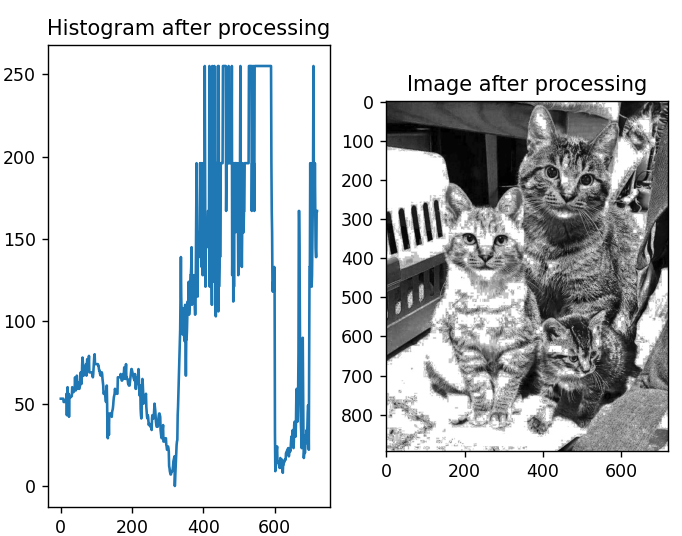
Ví dụ rõ hơn về biểu đồ histogram:



Hình 25 - Ví dụ về ảnh màu tối



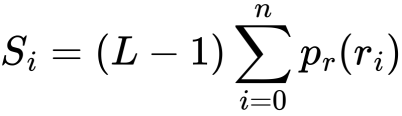
Hình 26 - Ví dụ về ảnh có màu sáng



Hình 27 - Ví dụ về ảnh có độ tương phản hài hòa

=> Từ đó, biểu đồ histogram phân bố sang phải thì ảnh có màu sáng (giá trị pixel tiến tới 255), phân bố sang trái thì ảnh có màu tối(giá trị pixel tiến tới 0), phân bố đều sẽ là ảnh cân bằng.

1. Cân bằng Histogram:

* Bước 1: Dựa vào các giá trị đã tính ở trên, ta cân bằng giá trị màu:
* Công thức (1): 
* Công thức khác (2): 

ri: giá trị sau màu tại i sau khi cân bằng.

ti: giá trị màu tại i trước khi cân bằng

min: giá trị nhỏ nhất trong các giá trị màu

max: giá trị lớn nhất trong các giá trị màu

L: giá trị cường độ có thể có (256 với ảnh màu xám)

\*Thuật toán sẽ áp dụng công thức (1) để hiện thực.

* Ví dụ:

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 6 | 1 | 4 |
| 5 | 3 | 2 | 2 |
| 6 | 3 | 3 | 8 |
| 8 | 4 | 3 | 5 |

Bảng 5 - Giá trị màu của ảnh 4x4 với mức xám L = 8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ri | 1 | 2 | 3 | 4 | 5 | 6 | 8 |
| h(ri) | 1 | 2 | 4 | 2 | 3 | 2 | 2 |
| P(ri) | 1/16 | 2/16 | 4/16 | 2/16 | 3/16 | 2/16 | 2/16 |

Bảng 6 - Ví dụ về chuẩn hóa giá trị màu của ảnh 4x4

Từ bảng giá trị, vẽ biểu đồ phân bố giá trị màu

Bảng 7 - Biểu đồ giá trị màu trước khi chuyển đổi

=> Sau khi tính được sự phân bố của các giá trị màu, chúng tập trung ở phần giữa từ đó ta thấy được hình ảnh có độ tương phản thấp. Để cải thiện, ta áp dụng công thức trên và vẽ lại biểu đồ.

|  |  |  |  |
| --- | --- | --- | --- |
| S1 | (5-1)/(8-1)\*7 = 4 | S9 | (6-1)/(8-1)\*7 = 5 |
| S2 | (6-1)/(8-1)\*7= 5 | S10 | (3-1)/(8-1)\*7 = 2 |
| S3 | (1-1)/(8-1)\*7 = 0 | S11 | (3-1)/(8-1)\*7 = 2 |
| S4 | (4-1)/(8-1)\*7 = 3 | S12 | (8-1)/(8-1)\*7 = 7 |
| S5 | (5-1)/(8-1)\*7 = 4 | S13 | (8-1)/(8-1)\*7 = 7 |
| S6 | (3-1)/(8-1)\*7 = 2 | S14 | (4-1)/(8-1)\*7 = 3 |
| S7 | (2-1)/(8-1)\*7 = 1 | S15 | (3-1)/(8-1)\*7 = 2 |
| S8 | (2-1)/(8-1)\*255 = 1 | S16 | (5-1)/(8-1)\*7 = 4 |

Bảng 8 - Ví dụ về bảng giá trị chuyển đổi

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 5 | 0 | 3 |
| 4 | 2 | 1 | 1 |
| 5 | 2 | 2 | 7 |
| 7 | 3 | 2 | 4 |

Bảng 9 - Bảng giá trị sau khi chuyển đổi

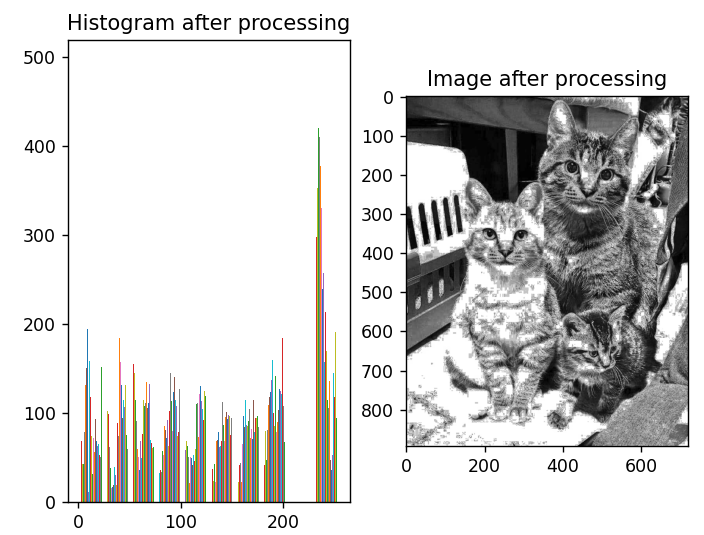
\*Chú ý: làm tròn giá trị cân bằng: do các giá trị màu trên pixel là số nguyên.

* *Bước 2*: Vẽ lại biểu đồ cân bằng Histogram dựa trên các giá trị đã tính ở bước cân bằng histogram:

Bảng 10 - Biểu đồ giá trị màu sau khi chuyển đổi

=> Sau khi áp dụng công thức, biểu đồ đã phân bố đều các giá trị màu từ đó ảnh có độ tương phản cao hơn ban đầu.

* Vẽ lược đồ histogram đã cân bằng: Để hiện thực lược đồ: hist(mảng giá trị), hàm hist() thuộc thư viện matplotlib.pyplot (hàm chuyên dùng để vẽ đồ thị).
* Thay đổi giá trị màu ban đầu thành giá trị màu đã cân bằng: thay đổi giá trị màu của hình ảnh ta dùng hàm imwrite(img path, histogram), hàm thuộc thư viện OpenCV.



Hình 28 - Ví dụ về biểu đồ histogram đã được cân bằng

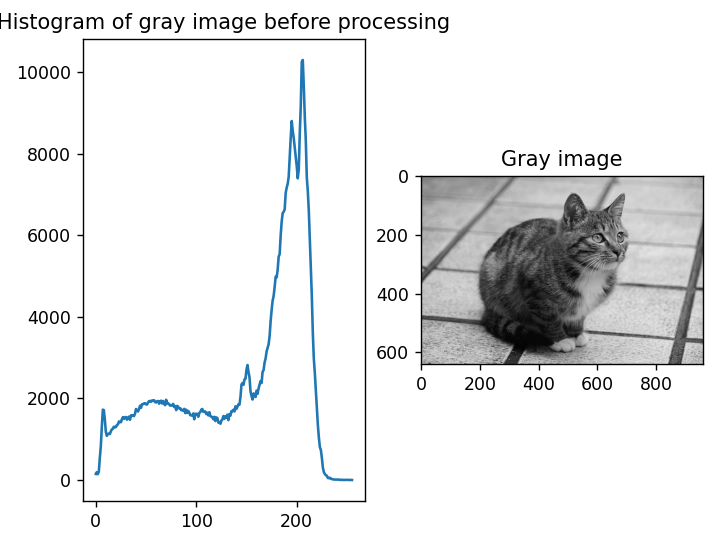
=> Ta thấy biểu đổ có lược đồ phân phối tích luỹ trải đều, cho thấy hình ảnh đã được cân bằng, nếu phân tích với số liệu lớn hơn sẽ rõ ràng hơn.

**\*Chú thích:**

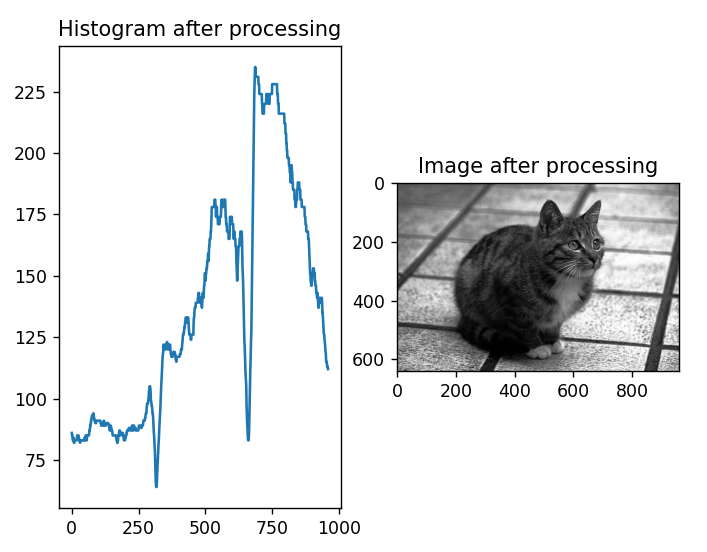
1. OpenCV: là thư viện trong python, được dùng để hỗ trợ xử lý cho thị giác máy tính: xử lí ảnh, nhận dạng hình ảnh, nhận dạng khuôn mặt, đối tượng, theo dõi chuyển động,…
2. Rgb: là viết tắt của red, green, blue. Đây là 3 màu gốc trong hệ thống màu
3. Thang màu xám: là hệ thống màu sắc chỉ bao gồm các màu sắc xám, đen, trắng, có giá trị từ 0 đến 255 (0: màu đen, 255: màu trắng).
4. **Ví dụ:**



Hình 29 - Ảnh gốc và histogram



Hình 30 - Ảnh chuyển xám và histogram



Hình 31 - Ảnh sau xử lí và histogram

=> Histogram đã cân bằng độ tương phản, sẽ có những giá trị nằm ở khoảng trung bình của đoạn [0,255] sẽ tăng lên so với histogram của ảnh gốc.

CHƯƠNG 3 – Thuật toán đối chiếu (histogram matching algorithm)

1. **Tổng quan:**

Thuật toán đối chiếu histogram là một kỹ thuật xử lí ảnh được dùng để điều chỉnh độ tương phản một hình ảnh sao cho histogram của nó giống với histogram của ảnh tham chiếu. Thuật toán này có những ưu điểm như đơn giản, dễ triển khai, nhanh chóng và có nhiều ứng dụng khác nhau trong xử lí ảnh, nhược điểm thì có thể không hiệu quả với các mẫu có kích thước quá lớn hay có độ nhiễu quá cao.

1. **Ưu điểm và nhược điểm của thuật toán:**
2. Ưu điểm của thuật toán:

* Cải thiện độ tương phản của ảnh: do thuật toán điều chỉnh cường độ của các điểm ảnh sao cho phân bố cường độ của ảnh giống với phân bố cường độ của ảnh tham chiếu. Nên độ tương phản được nâng cao chất lượng.
* Giảm nhiễu: thuật toán matching histogram cũng có thể chỉnh giảm nhiễu. Tình trạng nhiễu của hình ảnh là do phân bố cường độ ánh sáng không đồng đều, mà thuật toán trên sẽ dàn đều các điểm ảnh từ đó ảnh cũng sẽ giảm độ nhiễu.

1. Nhược điểm của thuật toán:

* Phụ thuộc vào ảnh mẫu: Do ảnh cần chỉnh sửa sẽ dựa trên ảnh được tham chiếu để thay đổi, nên nếu ảnh mẫu có độ tương phản kém, thì ảnh sau khi hiện thực thuật toán cũng có độ tương phản kém.
* Có thể làm mất chi tiết trong ảnh: giải thuật của histogram matching có thể khiến các chi tiết trong ảnh bị mờ hoặc mất đi.
* Có thể tăng độ nhiễu: Những ảnh có độ nhiễu cao, khả năng tăng độ nhiễu thông qua thuật toán càng cao hơn.

1. **Ràng buộc:**

* Kích thước mẫu không được quá lớn: thuật toán so khớp phải so sánh từng pixel trong mẫu với từng pixel trong văn bản, nên nếu mẫu có kích thước quá lớn thì thuật toán sẽ có thể không đạt được hiệu quả tối đa.
* Độ nhiễu không được quá cao: tương tự ràng buộc đầu tiên, do phải so sánh trên từng pixel nên nếu độ nhiễu quá cao thì các pixel trong mẫu và văn bản không thể khớp với nhau.

1. **Giải thuật**
2. Tính biểu đồ histogram của ảnh gốc và ảnh tham chiếu:

Hiện thực như tính biểu đồ histogram của thuật toán equalization histogram[[1]](#histogram).

1. So sánh và so khớp ảnh gốc với ảnh tham chiếu:

\*Dựa vào histogram của ảnh gốc và histogram của ảnh tham chiếu, ta cân bằng giá trị của histogram tương tự như thuật toán phía trên, rồi đem chúng so sánh:

Tại tần suất suất hiện h(ri) của ảnh gốc = giá trị màu đã cân bằng của ảnh đã tham chiếu thì ta đổi giá trị màu tại vị trí xuất hiện giá trị màu của ảnh tham chiếu bằng giá trị màu thực tế của ảnh tham chiếu.

Gọi ai là giá trị màu tại điểm đã so khớp với ảnh tham chiếu, ta có công thức:

wps

Ví dụ:

|  |  |
| --- | --- |
| Giá trị màu đã cân bằng  của ảnh tham chiếu  nk | Giá trị màu thực tế  mk |
| 1 | 80 |
| 3 | 100 |
| 5 | 90 |
| 6 |
| 6 |
| 7 | 30 |
| 7 |
| 7 |

Bảng 11 - Bảng giá trị histogram của ảnh tham chiếu

|  |  |
| --- | --- |
| Mức xám | Tần suất xuất hiện của giá trị  màu tại mức xám của ảnh gốc  h(ri) |
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 1 |
| 4 | 2 |
| 5 | 4 |
| 6 | 6 |
| 7 | 7 |

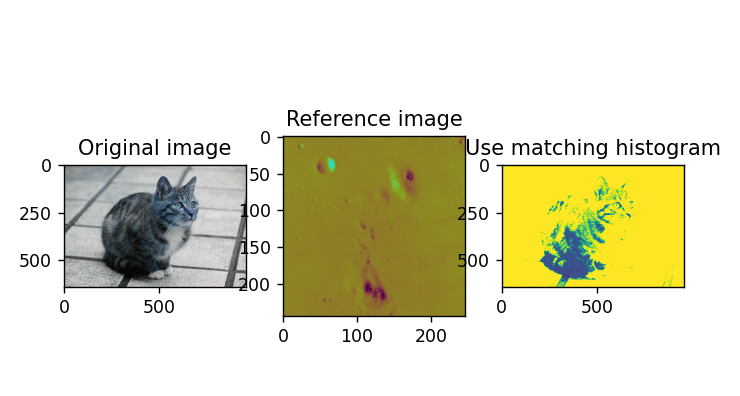
Bảng 12 - Bảng giá trị histogram của ảnh gốc

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mức xám | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Giá trị mới tại mức xám | 0 | 0 | 0 | 80 | 80 | 100 | 90 | 30 |

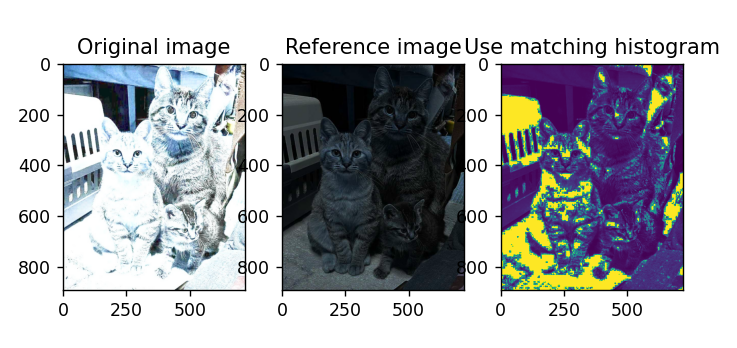
Bảng 13 - Bảng kết quả sau khi so sánh giá trị tại các pixel của ảnh gốc và ảnh tham chiếu

Bảng 14 - Biểu đồ histogram sau khi thực hiện so khớp

Ví dụ:



Hình 32 -Ví dụ về thuật toán matching histogram

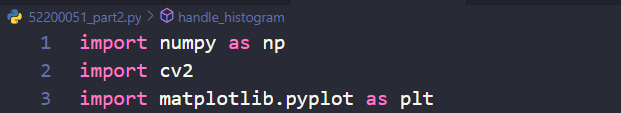


Hình 33 - Ví dụ về thuật toán matching histogram

Chương 4 - Hiện thực

1. Thuật toán cân bằng histogram:
2. Thêm thư viện:

Thêm thư viện OpenCV (cv2) để đọc ảnh, matplotlib (matplotlib.pyplot) để vẽ đồ thị phân bố điểm màu và thư viện numpy để phụ trợ trong việc xử lí các thao tác tính toán.



Hình 34 - Thêm thư viện

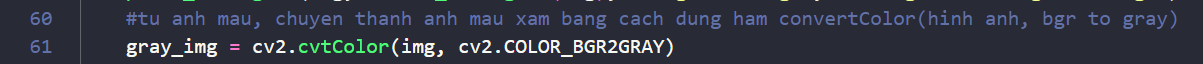
1. Xử lý hình ảnh đầu vào:

* Bước 1: đọc ảnh đầu vào ở phần chương trình main bằng phương thức thuộc thư viện OpenCV: imread (path image, x).
* Path image: địa chỉ của hình ảnh cần xử lý cân bằng (“input\_img.jpg”, “cat.png”, …).
* Với x có hai giá trị là 0 và 1; 0 đọc ảnh xám (grayscale image), 1 đọc ảnh màu (colorful image).



Hình 35 - Đọc ảnh đầu vào

* Bước 2: xử lí hình ảnh màu (ảnh đầu vào) thành ảnh có mức xám cùng với phương thức của thư viện OpenCV(cv2): cvtColor(image, method).
* Image: hình ảnh cần chuyển đổi.
* Method: phương thức muốn chuyển đổi, ví dụ: rgb (red-green-blue) sang grayscale (xám) → “COLOR\_BGR2GRAY”.

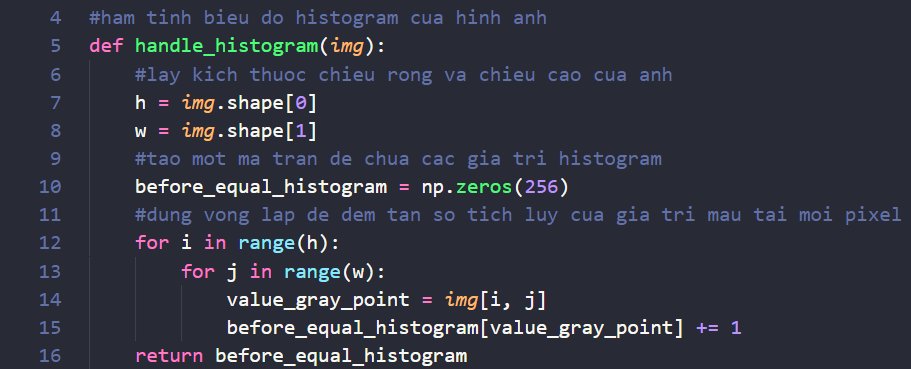


Hình 36 - Chuyển đổi ảnh sang mức xám

1. Định nghĩa hàm tính histogram:

\*Hàm có duy nhất một tham số: handle\_histogram(img) - img: hình ảnh dùng để tính histogram.

* Bước 1: lấy kích thước của ảnh (để tính tối đa số pixel có thể có trong ảnh):
* Phương thức shape thuộc thư viện numpy.
* Shape có ba thuộc tính: width (chiều rộng), height (chiều cao), depth (độ sâu) - với độ sâu = 1: ảnh xám, độ sâu = 3: ảnh màu).
* Ảnh sẽ xử lý trên mảng đa chiều (2 chiều nếu không tính độ sâu).
* Bước 2: tạo mảng rỗng có giá trị thuộc [0,255] để lưu tần suất xuất hiện giá trị màu tại tất cả các điểm pixel của ảnh:
* Dùng phương thức: zeros(n): tạo ra mảng có n phần tử có giá trị bằng 0.
* Dùng [0] \* kích thước: tương tự như zeros(n), cách này cũng sẽ tạo ra mảng có kích thước n và có tất cả các giá trị bằng 0.
* Bước 3: dùng vòng lặp trong vòng lặp để lưu giá trị tần suất lặp lại của giá trị màu theo dòng và cột (tương tự ma trận 2 chiều).
* Bước 4: Trả về histogram chứa các giá trị màu thuộc [0,255].



Hình 37 - Hàm tính histogram của ảnh

1. ***Định nghĩa hàm cân bằng histogram:***

\*Hàm có 2 tham số: equal\_histogram(img, histogram) - img: ảnh cần cân bằng, histogram: chứa histogram của ảnh chưa cân bằng.

* Bước 1: lấy kích thước của ảnh (để tính tối đa số pixel có thể có trong ảnh) - tương tự như ở hàm handle\_histogram(img).
* Bước 2: tạo mảng rỗng có kích thước 257 để lưu trữ các giá trị sau cân bằng - tương tự như ở hàm handle\_histogram(img).
* Bước 3: dùng vòng lặp để tính tổng số lần xuất hiện của các giá trị [0, 255] trong histogram. Mỗi số lần xuất hiện được lưu theo thứ tự tương ứng trong mảng.
* Bước 4: loại bỏ giá trị màu bằng 0 ra khỏi mảng (các giá trị này không sử dụng đến khi thực hiện quá trình cân bằng).
* Bước 5: tính giá trị tối đa và tối thiểu bằng hàm max() và min()) của numpy
* Bước 6: cân bằng giá trị tối đa dựa trên công thức đã nêu ở phần giải thuật của chương 2: công thức: 
* Bước 7: dùng vòng lặp để thay đổi giá trị ban đầu thành giá trị đã cân bằng, sau đó trả về hình ảnh đã cân bằng.

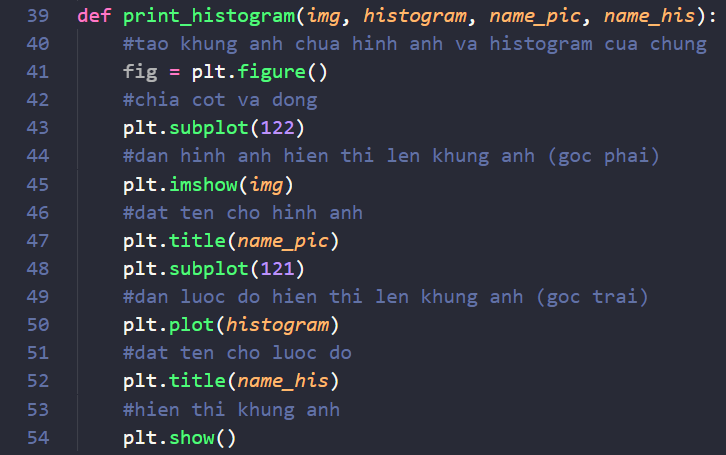


Hình 38 - Hàm cân bằng histogram của ảnh

1. Định nghĩa hàm vẽ biểu đồ và hiển thị hình ảnh

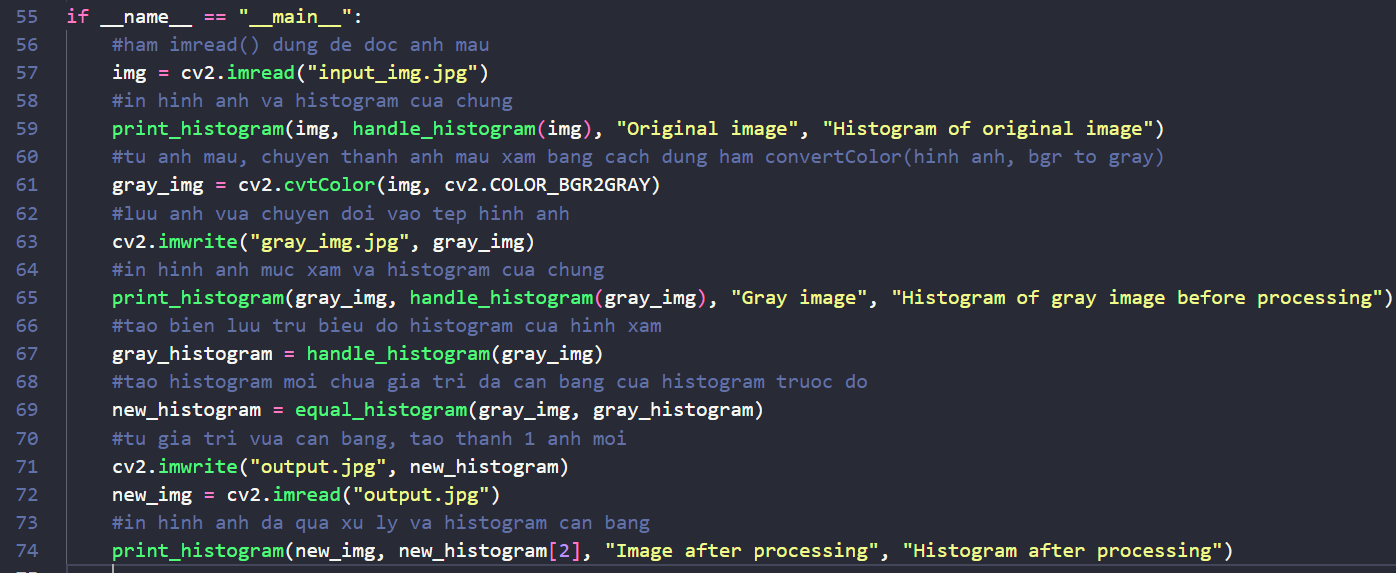
\*Hàm gồm 4 tham số: print\_histogram(img, histogram, name\_picture, name\_histogram) - img: hình ảnh cần để hiển thị, histogram: chứa histogram của hình ảnh, name\_picture: tên của hình ảnh, name\_histogram: tên của histogram.

* Bước 1: tạo khung hình trống: phương thức figure() của thư viện matplotlib
* Bước 2: chia khung ảnh thành hai phần bằng nhau: subplot() của thư viện matplotlib - 122: 12 là 1 khung ảnh chia làm 2, và 2 là chọn phần ở bên phải
* Một phần hiển thị ảnh: subplot(122) sau đó hiển thị hình ảnh imshow(img) và hiển thị tên của ảnh title(name\_picture).
* Một phần hiển thị histogram: subplot(121) để chọn sang phần bên trái và hiển thị biểu đồ plot(histogram) và hiển thị tên của biểu đồ title(name\_histogram).
* Bước 3: thể hiện toàn bộ những thành phần lên khung trống ban đầu đã tạo: dùng hàm show()



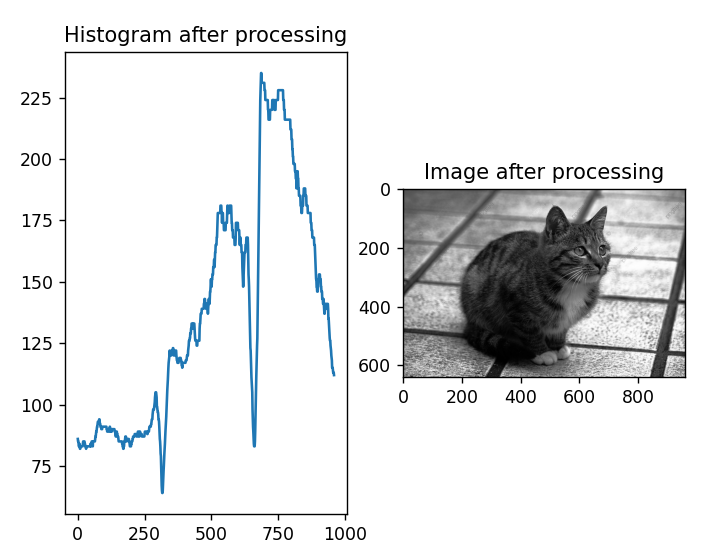
Hình 39 - Hàm hiển thị hình ảnh và histogram của ảnh

1. Chạy các hàm đã định nghĩa trong main:



Hình 40 - Code trong hàm main

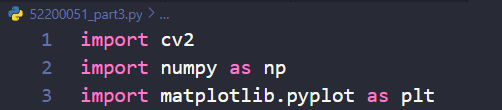
1. **Kết quả:**



Hình 41 - Kết quả sau khi xử lí

1. Thuật toán đối chiếu histogram:
2. Thêm thư viện

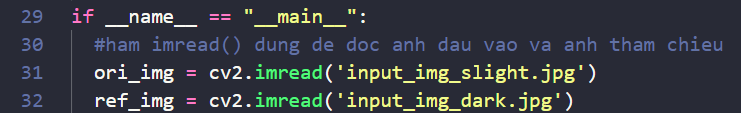
Thêm thư viện OpenCV (cv2) để đọc ảnh, matplotlib (matplotlib.pyplot) để vẽ đồ thị phân bố điểm màu và thư viện numpy để phụ trợ trong việc xử lí các thao tác tính toán.



Hình 42 - Thêm thư viện

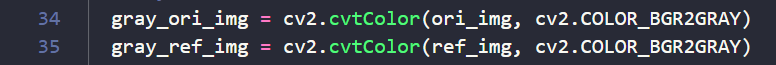
1. Xử lí ảnh gốc, ảnh tham chiếu

* Bước 1: đọc ảnh đầu vào ở phần chương trình main bằng phương thức thuộc thư viện OpenCV: imread (path image, x).
* Path image: địa chỉ của hình ảnh cần xử lý cân bằng (“input\_img.jpg”, “cat.png”, …).
* Với x có hai giá trị là 0 và 1; 0 đọc ảnh xám (grayscale image), 1 đọc ảnh màu (colorful image).



Hình 43 - Đọc ảnh đầu vào

* Bước 2: xử lí hình ảnh màu (ảnh đầu vào) thành ảnh có mức xám cùng với phương thức của thư viện OpenCV(cv2): cvtColor(image, method).
* Image: hình ảnh cần chuyển đổi.
* Method: phương thức muốn chuyển đổi, ví dụ: rgb (red-green-blue) sang grayscale (xám) → “COLOR\_BGR2GRAY”.

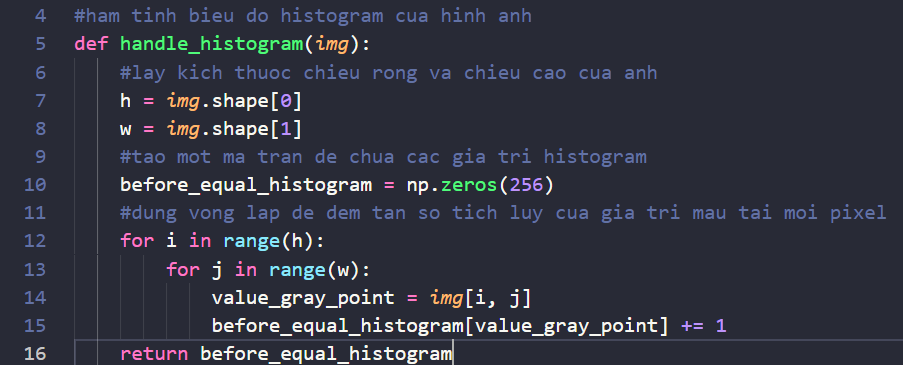


Hình 44 - Chuyển đổi ảnh đầu vào thành ảnh mức xám

1. Định nghĩa hàm tính histogram của ảnh gốc và ảnh tham chiếu

\*Hàm có duy nhất một tham số: handle\_histogram(img) - img: hình ảnh dùng để tính histogram.

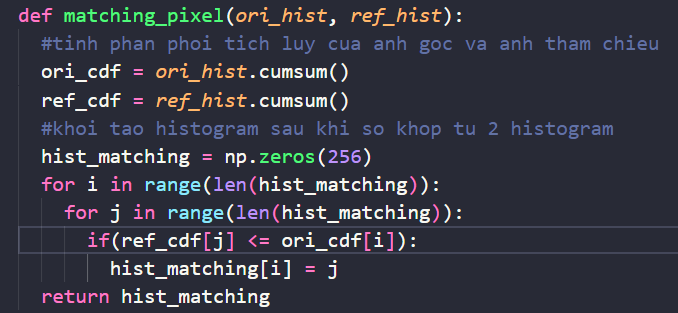
* Bước 1: lấy kích thước của ảnh (để tính tối đa số pixel có thể có trong ảnh):
* Phương thức shape thuộc thư viện numpy.
* Shape có ba thuộc tính: width (chiều rộng), height (chiều cao), depth (độ sâu) - với độ sâu = 1: ảnh xám, độ sâu = 3: ảnh màu).
* Ảnh sẽ xử lý trên mảng đa chiều (2 chiều nếu không tính độ sâu).
* Bước 2: tạo mảng rỗng có giá trị thuộc [0,255] để lưu tần suất xuất hiện giá trị màu tại tất cả các điểm pixel của ảnh:
* Dùng phương thức: zeros(n): tạo ra mảng có n phần tử có giá trị bằng 0.
* Dùng [0] \* kích thước: tương tự như zeros(n), cách này cũng sẽ tạo ra mảng có kích thước n và có tất cả các giá trị bằng 0.
* Bước 3: dùng vòng lặp trong vòng lặp để lưu giá trị tần suất lặp lại của giá trị màu theo dòng và cột (tương tự ma trận 2 chiều).
* Bước 4: Trả về histogram chứa các giá trị màu thuộc [0,255].



Hình 45 - Hàm tính histogram của ảnh

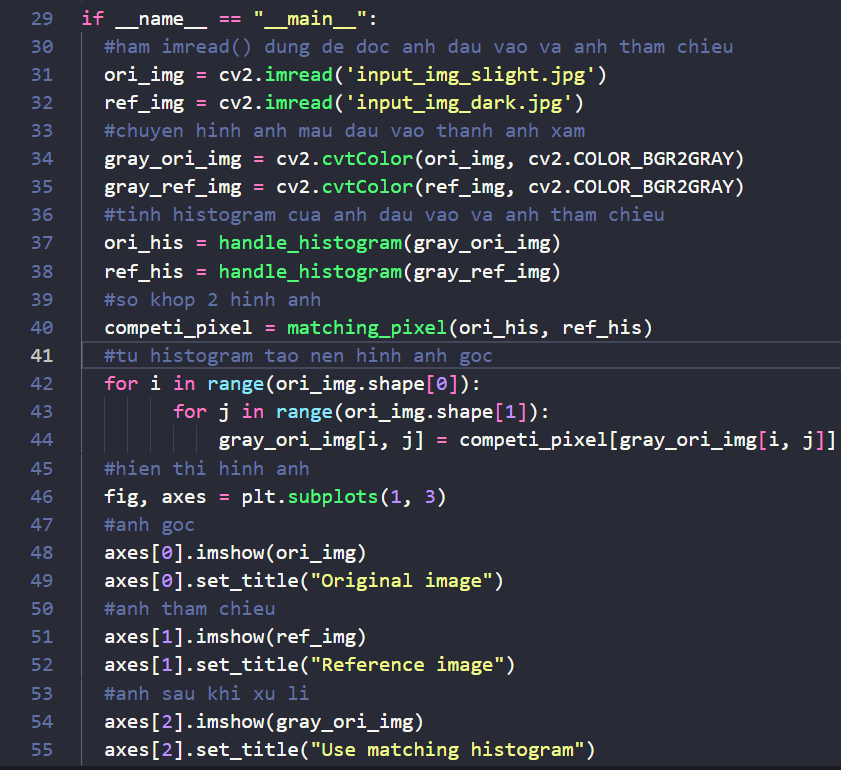
1. Định nghĩa hàm matching histogram

* Bước 1: tính phân phối tích lũy của ảnh gốc và ảnh tham chiếu: sử dụng hàm cumsum(): đây là hàm trả về tổng tích lũy của một mảng(theo bài là tổng tích lũy của histogram của ảnh).
* Bước 2: tạo mảng rỗng có giá trị thuộc [0,255] để lưu tần suất xuất hiện giá trị màu tại tất cả các điểm pixel của ảnh:
* Dùng phương thức: zeros(n): tạo ra mảng có n phần tử có giá trị bằng 0.
* Dùng [0] \* kích thước: tương tự như zeros(n), cách này cũng sẽ tạo ra mảng có kích thước n và có tất cả các giá trị bằng 0.
* Bước 3: So sánh từng gái trị đã cân bằng của ảnh tham chiếu với giá trị màu của nguồn.
* Bước 4: trả về histogram đã so sánh.

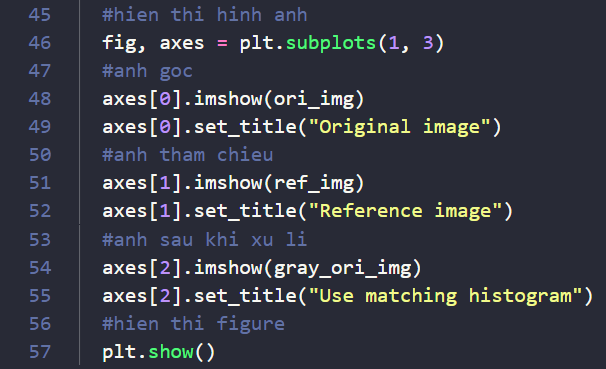


Hình 46 - Hàm matching histogram của ảnh

1. Chạy các hàm đã định nghĩa trong main

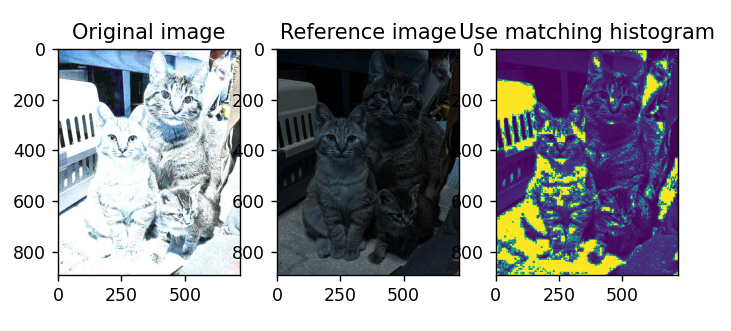


Hình 47 - Chạy hàm trong chương trình main



Hình 48 - Chạy hàm trong chương trình main

1. Kết quả



Hình 49 - Kết quả sau khi xử lí

TÀI LIỆU THAM KHẢO

Tiếng Việt

Education, E. O. (Director). (2021). *Xử lý ảnh || Chương 2. Cải thiện ảnh || Các phép toán trên điểm ảnh || Kỹ thuật Histogram* [Motion Picture].

Nguyễn, T. T. (2019, 11 22). *Xử lí ảnh: thuật toán cân bằng histogram ảnh*. Retrieved from VIBLO: https://viblo.asia/p/xu-li-anh-thuat-toan-can-bang-histogram-anh-GrLZDOogKk0

SPSS. (n.d.). *Phân tích SPSS*. Retrieved from Giá trị trung bình, trung vị, tứ phân vị: định nghĩa và phân biệt: https://phantichspss.com/gia-tri-trung-binh-trung-vi-tu-phan-vi-dinh-nghia-va-phan-biet.html

Tiếng Anh

Balram, N. (2006, 1). *Effect of Histogram Equalization*. Retrieved from ResearchGate: https://www.researchgate.net/figure/Effect-of-Histogram-Equalization-The-image-on-the-top-left-has-a-histogram-with-a-large\_fig1\_229036250

Python Software Foundation. (2023, 11 08). *Docs - Python.* Retrieved from statistics — Mathematical statistics functions: https://docs.python.org/3/library/statistics.html

TheAILearner. (2019, 10 04). *Histogram Matching (Specification)*. Retrieved from TheAILearner: https://theailearner.com/2019/04/10/histogram-matching-specification/

Tutorials Point. (n.d.). *Tutorials Point*. Retrieved from Histogram Equalization: https://www.tutorialspoint.com/dip/histogram\_equalization.htm