TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TIỂU LUẬN MÔN XÁC SUẤT THỐNG KÊ**

**ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

**MIDTERM ESSAY**

*Người hướng dẫn*: **TRẦN LƯƠNG QUỐC ĐẠI**

*Người thực hiện*: **ĐẶNG THÀNH NHÂN– 522H0006**

Lớp **: 22H50201**

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**



**BÀI TIỂU LUẬN MÔN XÁC SUẤT THỐNG KÊ**

**ỨNG DỤNG CHO CÔNG NGHỆ THÔNG TIN**

**MIDTERM ESSAY**

Người hướng dẫn: **TRẦN LƯƠNG QUỐC ĐẠI**

Người thực hiện: **ĐẶNG THÀNH NHÂN**

Lớp **: 22H50201**

Khoá  **: 26**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

* Lời đầu tiên em xin cảm ơn sâu sắc đến thầy Trần Lương Quốc Đại đã dành thời gian để hỗ trợ và giúp đỡ em trong quá trình học tập và hoàn thành bài tiểu luận. Những lời khuyên và nhận xét đã giúp em hiểu rõ hơn và hoàn thành bài tiểu luận này.
* Xin cảm ơn khoa Công Nghệ Thông Tin đã giao cho em bài tiểu luận này. Em tin chắc rằng bài tiểu luận này sẽ giúp em rèn luyện, tiếp thu và phát triển những kĩ năng để phục vụ cho công việc tương lai của mình.
* Do chưa có nhiều kinh nghiệm làm tiểu luận cũng như những hạn chế về kiến thức nên trong bài tiểu luận chắc chắn sẽ có những sai sót. Rất mong nhận được những nhận xét và ý kiến đóng góp từ thầy/cô để em có thể hoàn thiện hơn về kĩ năng này.
* Lời cuối cùng em chân thành cảm ơn thầy/cô đã đọc và cho những đánh giá. Chúc thầy/cô thầy thật nhiều sức khỏe, thành công và hạnh phúc.

**ĐỒ ÁN 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 thầy Trần Lương Quốc Đại. 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*

*(ký tên và ghi rõ họ tên)*



*Đặng Thành Nhân*

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 tiểu luận nghiên cứu về các thư viện thống kê và khảo sát cân bằng lược đồ và thuật toán xử lí ảnh cơ bản Histogram thông qua ngôn ngữ Python. Bài tiểu luận được thực hiện bởi sinh viên khoa Công Nghệ Thông Tin của trường Đại Học Tôn Đức Thắng. Bài tiểu luận được tham khảo từ nguồn website khoa cung cấp và nhiều nguồn khác trên internet. Thông qua bài tiểu luận có thể hiểu thêm về thuật toán xử lí ảnh cơ bản Histogram và thư viện thống kê trong ngôn ngữ Python.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc150457253)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc150457254)

[TÓM TẮT iv](#_Toc150457255)

[MỤC LỤC 1](#_Toc150457256)

[CHƯƠNG 1 – THƯ VIỆN THỐNG KÊ TRONG PYTHON 3](#_Toc150457257)

[1.1 Tổng quan về thư viện thống kê trong Python 3](#_Toc150457258)

[1.1.1 Trung bình và phương pháp tìm vị trí trung tâm 3](#_Toc150457259)

[1.1.2 Đo lường về sự phân tán 4](#_Toc150457260)

[1.1.3 Thống kê cho hai dữ liệu đầu vào 4](#_Toc150457261)

[1.1.4 Lớp phân phối chuẩn 4](#_Toc150457262)

[1.1.4.1 Các thuộc tính của NormalDist (chỉ đọc- không thể thay đổi) 4](#_Toc150457263)

[1.1.4.2 Các phương thức của NormalDist (chỉ đọc- không thể thay đổi) 4](#_Toc150457264)

[1.2 Giải thích ý nghĩa của từng hàm 5](#_Toc150457265)

[1.2.1 Trung bình và phương pháp tìm vị trí trung tâm 5](#_Toc150457266)

[1.2.2 Đo lường về sự phân tán 16](#_Toc150457267)

[1.2.3 Thống kê cho hai dữ liệu đầu vào 21](#_Toc150457268)

[1.2.4 Lớp phân phối chuẩn 24](#_Toc150457269)

[1.2.4.1 Các thuộc tính của NormalDist (chỉ đọc- không thể thay đổi) 24](#_Toc150457270)

[1.2.4.2 Các phương thức của NormalDist (chỉ đọc- không thể thay đổi) 24](#_Toc150457271)

[CHƯƠNG 2 – GIẢI THUẬT CÂN BẰNG LƯỢC ĐỒ (HISTOGRAM EQUALIZATION) 29](#_Toc150457272)

[2.1 Tổng quan về phương pháp Histogram Equalization 29](#_Toc150457273)

[2.1.1 Bài toán và mục tiêu 29](#_Toc150457274)

[2.1.2 Ràng buộc và điều kiện 29](#_Toc150457275)

[2.1.3 Phương pháp và thuật toán 29](#_Toc150457276)

[2.1.4 Nhận xét và phân tích 29](#_Toc150457277)

[2.2 Hình ảnh minh họa 30](#_Toc150457278)

[CHƯƠNG 3 – GIẢI THUẬT CÂN BẰNG LƯỢC ĐỒ (HISTOGRAM MATCHING) 31](#_Toc150457279)

[3.1 Tổng quan về phương pháp Histogram Matching 31](#_Toc150457280)

[3.1.1 Bài toán và mục tiêu 31](#_Toc150457281)

[3.1.2 Ràng buộc và điều kiện 31](#_Toc150457282)

[3.1.3 Phương pháp và thuật toán 31](#_Toc150457283)

[3.1.4 Nhận xét và phân tích 31](#_Toc150457284)

[3.2 Hình ảnh minh họa 32](#_Toc150457285)

[32](#_Toc150457286)

[CHƯƠNG 4 – THỰC HIỆN CHƯƠNG TRÌNH 33](#_Toc150457287)

[4.1 Giải thuật cân bằng lược đồ (Histogram Equalization) 33](#_Toc150457288)

[4.1.1 Thực thi chương trình 33](#_Toc150457289)

[4.1.3 Hướng dẫn xây dựng và chạy chương trình 34](#_Toc150457290)

[4.1.4 Thực nghiệm và kết luận 34](#_Toc150457291)

[4.2 Giải thuật cân bằng lược đồ (Histogram Matching) 34](#_Toc150457292)

[4.2.1 Thực thi chương trình 34](#_Toc150457293)

[4.2.3 Hướng dẫn xây dựng và chạy chương trình 36](#_Toc150457294)

[4.2.4 Thực nghiệm và kết luận 36](#_Toc150457295)

CHƯƠNG 1 – THƯ VIỆN THỐNG KÊ TRONG PYTHON

* 1. Tổng quan về thư viện thống kê trong Python

1.1.1 Trung bình và phương pháp tìm vị trí trung tâm

* **mean()**: Hàm tính trung bình cộng dữ liệu.
* **fmean()**: Hàm tính trung bình cộng dữ liệu với khả năng đặt trọng số tùy chọn.
* **fmean()**: Hàm tính trung bình cộng dữ liệu với khả năng đặt trọng số tùy chọn.
* **geomectric\_mean()**: Hàm tính trung bình hình học của dữ liệu.
* **harmonic\_mean()**: Hàm tính trung bình điều hòa của dữ liệu.
* **median()**: Hàm tính trung vị (giá trị ở giữa) của dữ liệu.
* **median\_low()**: Hàm tính trung vị thấp của dữ liệu.
* **median\_high()**: Hàm tính trung vị cao của dữ liệu.
* **median\_grouped()**: Hàm tính trung vị, hoặc phân vị thứ 50, của dữ liệu đã được nhóm lại.
* **mode()**: Hàm trả về duy nhất một giá trị mode (giá trị phổ biến nhất) của dữ liệu rời rạc hoặc dữ liệu danh nghĩa.
* **multimode()**: Hàm trả về một danh sách chứa giá trị mode (giá trị phổ biến nhất) của dữ liệu rời rạc hoặc dữ liệu danh nghĩa.
* **quantiles()**: Hàm chia dữ liệu thành các khoảng có xác xuất bằng nhau.

1.1.2 Đo lường về sự phân tán

* **pstdev()**: Hàm tính độ lệch chuẩn của quần thể dữ liệu (population).
* **pvariance()**: Hàm tính phương sai của quần thể dữ liệu (population).
* **stdev()**: Hàm tính độ lệch chuẩn của dữ liệu mẫu (sample).
* **variance()**: Đây là hàm tính phương sai của dữ liệu mẫu (sample).

1.1.3 Thống kê cho hai dữ liệu đầu vào

* **covariance():** Hiệp phương sai mẫu của hai biến.
* **correlation():** Hệ số tương quan Pearson và Spearman.
* **linear\_regression**(): Độ dốc và điểm cắt của phương trình hồi quy tuyến tính đơn giản.

1.1.4 Lớp phân phối chuẩn

1.1.4.1 Các thuộc tính của NormalDist (chỉ đọc- không thể thay đổi)

* **mean:** Thuộc tính trả về giá trị trung bình số học.
* **median:** Thuộc tính trả về giá trị trung vị.
* **mode:** Thuộc tính trả về giá trị có tần suất xuất hiện hiều nhất.
* **stdev:** Thuộc tính trả về độ lệch chuẩn.
* **variance:** Thuộc tínhrả về phương sai.

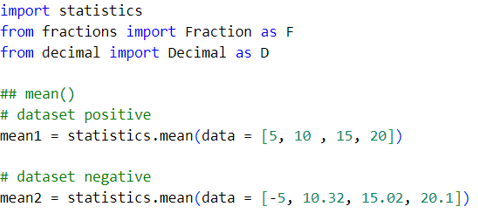
1.1.4.2 Các phương thức của NormalDist (chỉ đọc- không thể thay đổi)

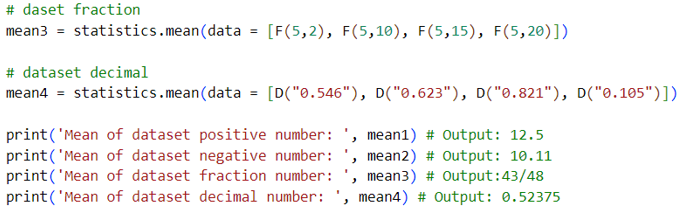
* **from\_sample():** Phương thức tạo đối tượng phân phối chuẩn.
* **samples():** Phương thức tạo mẫu ngẫu nhiên.
* **pdf():** Phương thức tính hàm phân phối xác suất.
* **cdf():** Phương thức tính hàm phân phối tích lũy.
* **inv\_cdf():** Phương thức tính giá trị ngược của hàm phân phối tích lũy.
* **overlap():** Tính độ trùng lặp giữa 2 phân phối chuẩn.
* **quantiles():** Dùng để tính quantiles của phân phối chuẩn.
* **zcore():** Dùng để tính khoảng cách giữa x và giá trị trung bình của phân phối chuẩn theo đơn vị của độ lệch chuẩn.

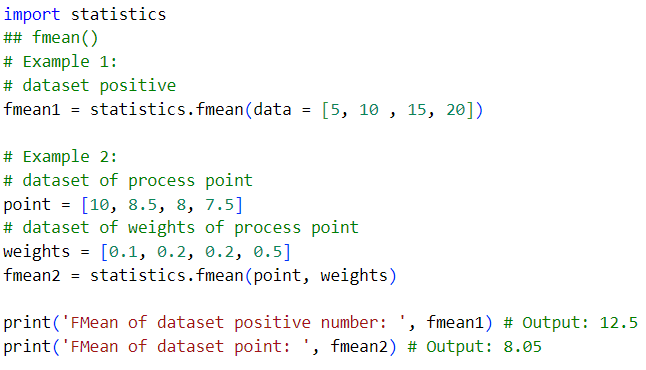
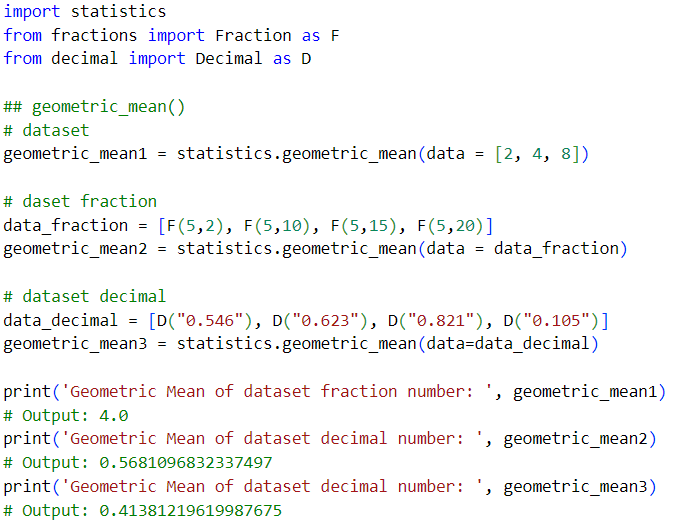
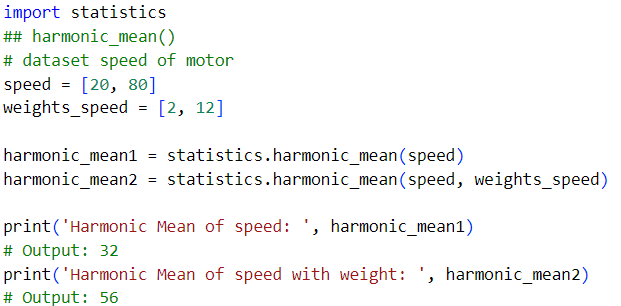
1.2 Giải thích ý nghĩa của từng hàm

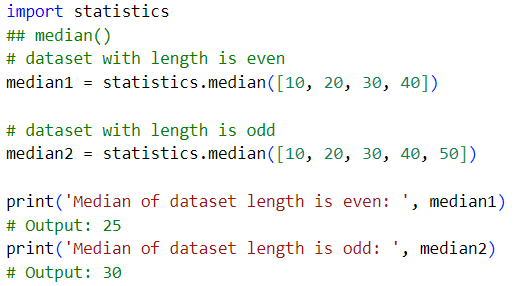
1.2.1 Trung bình và phương pháp tìm vị trí trung tâm

* **Hàm mean():**
* **Công dụng:** Dùng để tính giá trị trung bình (arithmetic mean) từ dữ liệu đầu vào. Thường được dùng để xác định được xu hướng trung tâm của dữ liệu.
* **Cú pháp sử dụng hàm:** statistics.mean(data).
* **Tham số đầu vào:** Tham số đầu vào hàm mean() là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là giá trị trung bình của tham số đầu vào data.
* **Ví dụ:**
* Hàm mean nhận dữ liệu đầu vào từ các mảng số dương, số âm, phân số, số thập phân và cho ra kết quả trung bình tương ứng với mỗi mảng.

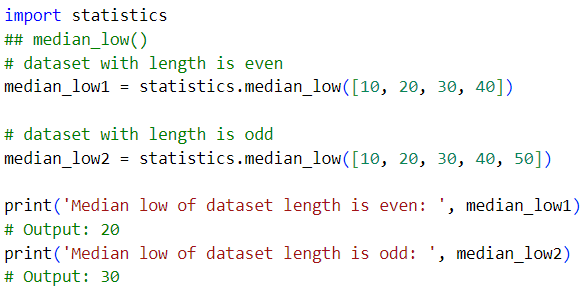
****

****

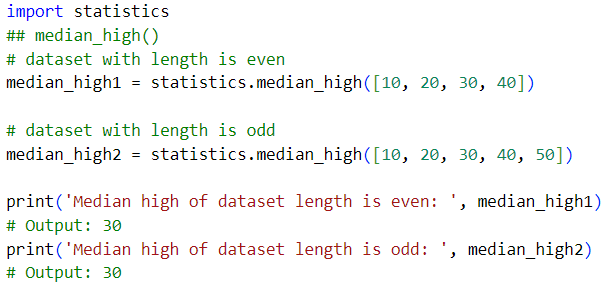
* **Hàm fmean():**
* **Công dụng**: Hàm fmean() có chức năng tương tự với hàm mean() và là một phiên bản nhanh hơn của hàm mean(). Hàm fmean() cho kết quả chính xác hơn hàm mean() với việc chuyển đổi dữ liệu sang kiểu dữ liệu thực (float) trước khi tính toán.
* **Cú pháp sử dụng hàm:** statistics.fmean(data) hoặc statistics.fmean(data, weights).
* **Tham số đầu vào:**
* **data**: Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **weights**: Là một tập dữ liệu số (numeric) tương tự như data, wights chứa các trọng số tương ứng với từng phần tử trong data. Tham số này là tùy chọn và mặc định là None.
* **Tham số đầu ra:** Là một giá trị số thực (float) chứa giá trị trung bình của giá trị đầu vào data theo tỉ lệ tương ưng weights.
* **Ví dụ:**
* Hàm fmean nhận dữ liệu đầu vào và trả về kết quả trung bình của mảng.
* Tính điểm trung bình môn học của 1 một học sinh theo quy chế: điểm quá trình 1 là 10%, quá trình 2 là 20%, giữa kì là 20%, cuối kỳ là 50%.
* **Hàm geomectric\_mean():**
* **Công dụng:** Hàm tính trung bình hình học chỉ ra giá trị trung tâm hoặc giá trị điển hình của dữ liệu đầu vào.
* **Cú pháp sử dụng hàm:** statistics.geometric\_mean(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Trong đó tất cả giá trị phải là số dương. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một giá trị rỗng (null), chứa một phần tử âm hoặc bằng 0 thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra**: Là một giá trị số thực (float) chứa giá trị trung bình hình học của tham số đầu vào data.
* **Ví dụ:**
* Hàm geometric\_mean nhận dữ liệu đầu vào là số nguyên, phân số, số thập phân và cho ra kết quả trung bình hình học tương ứng.
* **Hàm harmonic\_mean():**
* **Công dụng:** Hàm harmonic\_mean() là hàm tính giá trị trung bình điều hòa dữ liệu.
* **Cú pháp sử dụng hàm:** harmonic\_mean(data, weights).
* **Tham số đầu vào:**
* **data**: Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Trong đó tất cả giá trị phải là số dương. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data của hàm harmonic\_mean là một giá trị rỗng (null) hoặc chứa một phần tử là số âm thì hàm sẽ trả về lỗi StatisticsError. Nếu tham số đầu vào data chứa một phần tử là số 0 thì kết quả trả về sẽ là 0.
* **weights**: Có kiểu dữ liệu tương tự như data, weights chứa các trọng số tương ứng với từng phần tử trong data. Tham số này là tùy chọn và mặc định là None.
* **Tham số đầu ra:** Là một giá trị số thực (float) chứa giá trị trung bình điều hòa của giá trị đầu vào data theo tỉ lệ tương ứng weights.
* **Ví dụ:**
* Giả sử một chiếc xe máy đang đi trong 2 đoạn với tốc độ lần lượt là 20, 80. Trong trường hợp này giá trị weightsđược bỏ trống nên sẽ được mặc định là None.
* Giả sử một chiếc xe máy đang đi trong đường hẹp nên duy trì tốc độ 20 km/h trong 2km. Sau khi đi ra khỏi đường hẹp nên xe tăng tốc độ lên 80km/h trong 12km tiếp theo. Trường hợp này weights là độ dài của 2 đoạn đường.
* **Hàm median():**
* **Công dụng:** Hàm median() cho biết giá trị ỡ giữa sau sắp xếp của dữ liệu đầu vào của n phần trong tham số đầu vào. Với n (là độ dài của tham số đầu vào):
* Nếu n là một giá trị lẻ thì giá trị trung vị nằm ở ví giữa của tập tham số đầu vào.
* Nếu n là một giá trị chẵn thì giá trị trung vị sẽ bằng với trung bình cộng của 2 phần tử ở giữa.
* **Cú pháp sử dụng hàm:** statistics.median(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vàodata là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) chứa giá trị trung vị của tham số đầu vào data.
* **Ví dụ:**
* Hàm median nhận tham số đầu vào lần lượt là các mảng có độ dài chẵn và lẽ sau đó tính vị trung vị tương ứng của từng mảng.

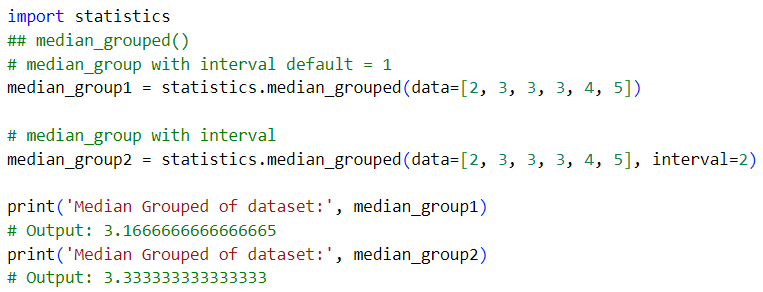
****

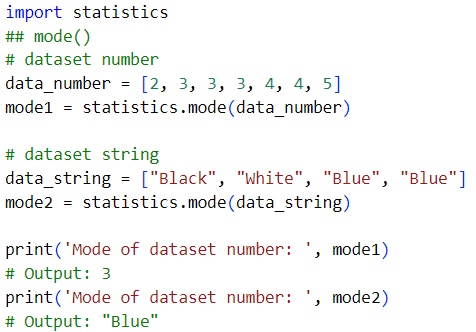
* **Hàm median\_low():**
* **Công dụng:** Hàm median\_low() dùng để tính trung vị thấp của n phần tử có trong dữ liệu truyền vào. Với n (là độ dài của tham số đầu vào):
* Nếu n là một giá trị lẻ thì giá trị trung vị thấp nằm ở ví giữa của tập tham số đầu vào.
* Nếu n là một giá trị chẵn thì giá trị trung vị thấp sẽ là giá trị nhỏ hơn giữa 2 phần tử nằm giữa mảng.
* **Cú pháp sử dụng hàm:** statistics.median\_low(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vàodata là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) chứa giá trị trung vị thấp của tham số đầu vào data.
* **Ví dụ:**
* Hàm median\_low nhận tham số đầu vào lần lượt là các mảng có độ dài chẵn và lẽ sau đó tính vị trung vị tương ứng của từng mảng.

****

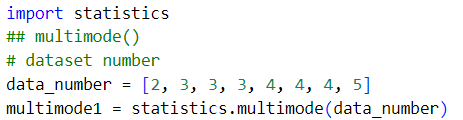
* **Hàm median\_high():**
* **Công dụng:** Dùng để tính trung vị thấp của n phần tử có trong dữ liệu truyền vào. Với n (là độ dài của tham số đầu vào):
* Nếu n là một giá trị lẻ thì giá trị trung vị cao nằm ở ví giữa của tập tham số đầu vào.
* Nếu n là một giá trị chẵn thì giá trị trung vị cao sẽ là giá trị lớn hơn giữa 2 phần tử nằm giữa mảng.
* **Cú pháp sử dụng hàm:** statistics.median\_high(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vàodata là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) chứa giá trị trung vị cao của tham số đầu vào data.
* **Ví dụ:**
* Hàm median\_high nhận tham số đầu vào lần lượt là các mảng có độ dài chẵn và lẽ sau đó tính vị trung vị tương ứng của từng mảng.

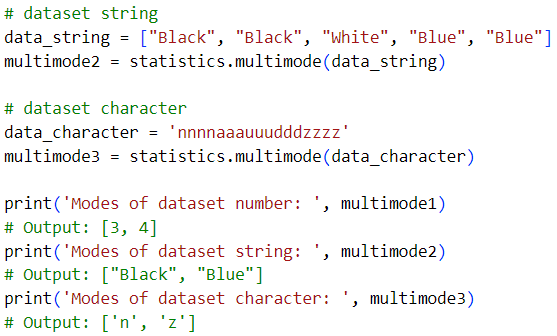
****

* **Hàm median\_grouped():**
* **Công dụng:** Hàm median\_grouped() dùng để tính trung vị(median) của tham số đầu vào khi chúng đã được nhóm thành các khoảng (interval). Phương pháp này thường được sử dụng khi dữ liệu liên tục và không đơn lẻ.
* **Cú pháp sử dụng hàm:** statistics.median\_grouped(data, interval).
* **Tham số đầu vào:**
* **data:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Trong đó tất cả giá trị phải là số dương. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một giá trị rỗng thì hàm sẽ trả về lỗi StatisticsError.
* **interval:** là một số nguyên dương. Tham số này là tùy chọn và mặc định là 1.
* **Tham số đầu ra**: Là một giá trị số thực (float) chứa giá trị trung vị của tham số đầu vào datađã được nhóm lại hoặc tính ở phân vị thứ 50.
* **Ví dụ:**
* ****Hàm median\_group nhận tham số đầu vào tham số đầu vào và trả về giá trị trung vị theo khoảng.
* **Hàm mode():**
* **Công dụng:** Hàm mode() dùng để tính giá trị mode của một tập tham số đầu vào. Mode là giá trị xuất hiện với tần suất nhiều nhất trong tập tham số đầu vào. Thường được sử dụng để xác định giá trị phổ biến nhất trong tập dữ liệu.
* **Cú pháp sử dụng hàm:** statistics.mode(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vàodata là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) hoặc chuỗi(string) chứa giá trị xuất hiện với tần suất nhiều nhất trong tham số đầu vào data.
* **Ví dụ:**
* Hàm mode nhận vào tập dữ liệu vào trả về giá trị có tần số xuất hiện nhiều nhất trong đó.

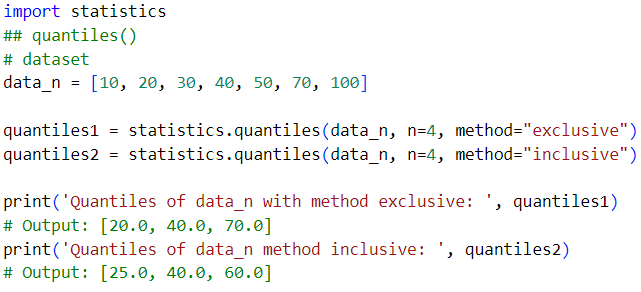
****

* **Hàm multimode():**
* **Công dụng:** Khác với hàm mode() chỉ trả về một giá trị với tần suất xuất hiện nhiều nhất trong tham số đầu vào hàm multimode() trả về một tập các các giá trị có tần suất xuất hiện nhiều nhất trong tập tham số đầu vào.
* **Cú pháp sử dụng hàm:** statistics.multimode(data).
* **Tham số đầu vào:** Là một tập dữ liệu số (numberic) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vàodata là một giá trị rỗng (null) thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một tập các giá trị số thực (float) hoặc chuỗi(string) xuất hiện với tần suất nhiều nhất trong tham số đầu vào data.
* **Ví dụ:**
* Hàm multimode nhận vào tập dữ liệu vào trả về các giá trị có tần số xuất hiện niều nhất trong đó.



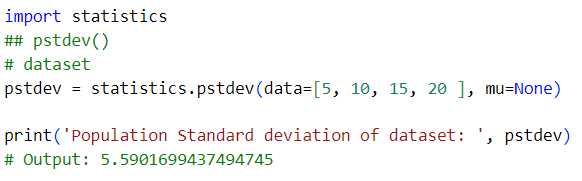


* **Hàm quantiles():**
* **Công dụng:** Hàm quantiles() dùng để tính các giá trị phân vị (quantiles) của tham số đầu vào.
* **Cú pháp sử dụng hàm:** statistics.quantiles(data, n, method).
* **Tham số đầu vào:**
* **data** : Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống.
* **n**: Số lượng khoảng quantiles muốn chia dữ liệu thành. Giá trị mặc định là 4, tương đương với việc chia dữ liệu thành 4 khoảng tương đương với việc tính các phân vị 25%, 50%, 75%.
* **method**: Phương pháp sử dụng để tính quantiles. Có hai phương pháp được hỗ trợ: ‘inclusive’ và ‘exclusive’. Phương pháp inclusive tính quantiles bằng cách lấy giá trị trung bình của hai giá trị gần nhất khi chia dữ liệu thành các khoảng. Phương pháp ‘exclusive’ tính quantiles bằng cách lấy giá trị gần nhất khi chia dữ liệu thành các khoảng. Giá trị mặc định là ‘exclusive’.
* Nếu tham số đầu vàon của hàm quantiles() lớn hơn 1 hoặc data chỉ có 1 phần tử hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra**: Là một tập các giá trị số thực (float) chứa giá trị phân vị tính được từ tham số đầu vào data.
* **Ví dụ:**
* Hàm quantiles() nhận tham số đầu vào data, n , method . Hàm sẽ chia dữ liệu đầu vào data thành 4 khoảng và tính giá trị phân vị cho các khoảng đó.

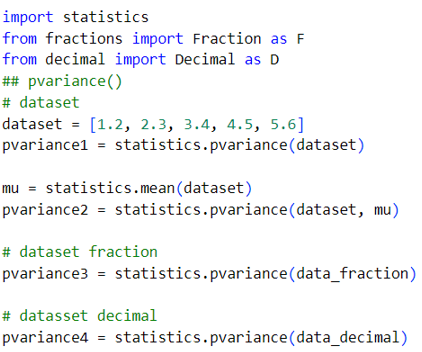


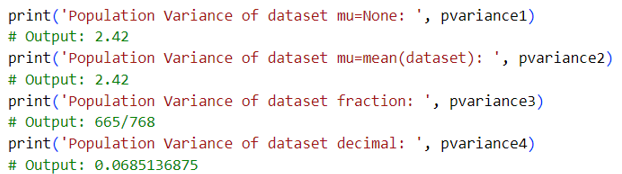
1.2.2 Đo lường về sự phân tán

* **Hàm pstdev():**
* **Công dụng:** Hàm pstdev() dùng để tính độ lệch chuẩn mẫu của tham số đầu vào. Thường được dùng để đo mức độ phân tán và biến thiên của dữ liệu xung quanh giá trị trung bình.
* **Cú pháp sử dụng hàm:** statistics.pstdev(data, mu).
* **Tham số đầu vào:**
* **data:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một tập rỗng thì hàm sẽ trả về lỗi StatisticsError.
* **mu**: Là giá trị trung bình của tổng quần thể. Giá trị mặc định là None.
* **Tham số đầu ra:** Là một giá trị số thực (float) biểu diễn độ lệch chuẩn của tham số đầu vào data.
* **Ví dụ:**
* Giá trị độ lệch chuẩn sẽ bằng căn bậc 2 của phương sai.

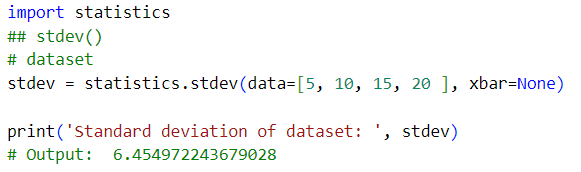
****

* **Hàm pvariance():**
* **Công dụng:** Hàm pvariance() dùng để tính phương sai mẫu của tham số đầu vào. Thường được dùng để đo mức độ phân tán và biến thiên của dữ liệu xung quanh giá trị trung bình.
* **Cú pháp sử dụng hàm:** statistics.pvariance(data, mu).
* **Tham số đầu vào:**
* **data** : Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một tập rỗng thì hàm sẽ trả về lỗi StatisticsError.
* **mu**: Thường là giá trị trung bình (mean) của tổng quần thể. Giá trị mặc định là None. Nếu giá trị mu là None thì mu sẽ được gán bằng giá trị trung bình (mean) của tham số đầu vào.
* **Tham số đầu ra:** Là một giá trị số thực (float) biểu diễn độ lệch chuẩn của tham số đầu vào data.
* **Ví dụ:**
* Hàm pvariance nhận tham số đầu vào lần lượt là số nguyên, phân số, số thập phân và trả về các giá trị phương sai tương ứng.

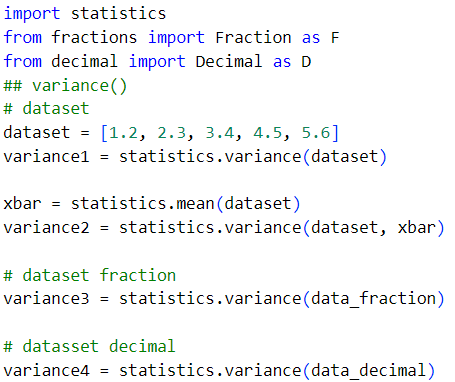


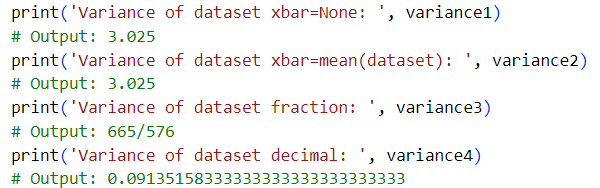


* **Hàm stdev():**
* **Công dụng:** Hàm stdev() có chức năng như hàm pstdev(). Hàm stdev tính độ lệch chuẩn dựa trên tổng thể các giá trị thay vì dựa trên một mẫu như hàm pstdev().
* **Cú pháp sử dụng hàm:** statistics.stdev(data, xbar)
* **Tham số đầu vào:**
* **data:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data là một tập rỗng thì hàm sẽ trả về lỗi StatisticsError.
* **xbar:** Thường là giá trị trung bình của tổng quần thể. Giá trị mặc định là None.
* **Ví dụ:**
* Giá trị độ lệch chuẩn sẽ bằng căn bậc 2 của phương sai.

****

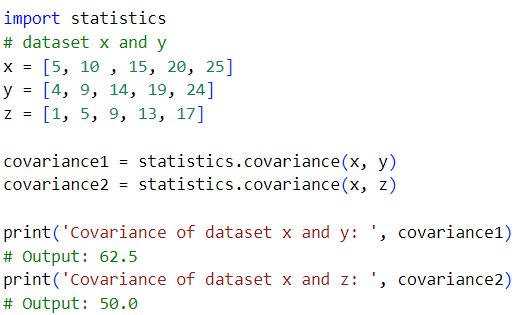
* **Hàm variance():**
* **Công dụng:** Hàm variance() có chức năng như hàm pvariance(). Hàm variance tính độ lệch chuẩn dựa trên tổng thể các giá trị thay vì dựa trên một mẫu như hàm pvariance().
* **Cú pháp sử dụng hàm:** statistics.variance(data, xbar).
* **Tham số đầu vào:**
* **data:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Tham số này là bắt buộc và không thể bỏ trống. Nếu tham số đầu vào data à một tập rỗng thì hàm sẽ trả về lỗi StatisticsError.
* **xbar:** Thường là giá trị trung bình (mean) của tổng quần thể. Giá trị mặc định là None. Nếu giá trị mu là None thì mu sẽ được gán bằng giá trị trung bình (mean) của tham số đầu vào.
* **Ví dụ:**
* Hàm pvariance nhận tham số đầu vào lần lượt là số nguyên, phân số, số thập phân và trả về các giá trị phương sai tương ứng



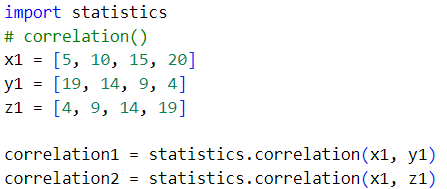


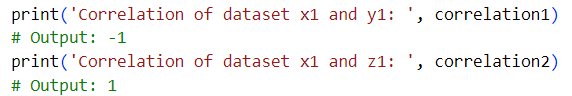
1.2.3 Thống kê cho hai dữ liệu đầu vào

* **Hàm covariance():**
* **Công dụng:** Dùng để tính hiệp phương sai (covariance) của 2 tham số đầu vào. Thường dùng để đo độ biến đổi đồng thời của 2 tham số đầu vào.
* **Cú pháp sử dụng hàm:** statistics.stdev(x, y)
* **Tham số đầu vào: x và y** là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Nếu tham 2 tham số đầu vào không cùng độ dài thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) thể hiện giá trị hiệp phương sai của 2 tham số đầu vào.
* **Ví dụ:**
* Hàm covariance nhận 2 tham số đầu vào và trả về hiệp phương sai.

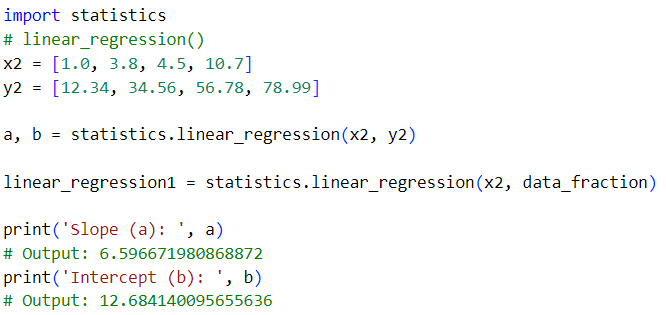
****

* **Hàm correlation():**
* **Công dụng:** Dùng để tính hệ số tương quan (correlation coefficient) giữa 2 tham số đầu vào.
* **Cú pháp sử dụng hàm:** statistics.scorrelation(x, y)
* **Tham số đầu vào: x và y** là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Nếu tham 2 tham số đầu vào không cùng độ dài thì hàm sẽ trả về lỗi StatisticsError.
* **Tham số đầu ra:** Là một giá trị số thực (float) thể hiện giá trị tương quan giữa 2 tham số đầu vào x và y chạy trong khoảng từ -1 đến 1:
* Nếu gần bằng 1 thì thể hiện chúng có mối quan hệ tuyến tính mạnh và biến đổi cùng chiều.
* Nếu gần bằng -1 thì thể hiện chúng có mối quan hệ tuyến tính yếu và biến đổi ngược chiều.
* Nếu gần bằng 0 thì không có mối quan hệ tuyến tính giữa 2 tham số đầu vào.
* **Ví dụ:**
* Hàm correlation nhận tham số đầu vào và tính độ tương quan giữa 2 tham số.

****

****

* **Hàm linear\_regression**():
* **Công dụng:** Dùng để thực hiện phân tích hồi quy tuyến tính giữa hai tập hợp dữ liệu x và y. Phân tích hồi quy tuyến tính tìm ra một đường thẳng gần nhất biểu diễn mối quan hệ tuyến tính giữa hai biến.
* **Cú pháp sử dụng hàm:** statistics.linear\_regression(x, y, proportional).
* **Tham số đầu vào:**
* **x và y:** Là một tập dữ liệu số (numeric) như: mảng (array), danh sách (list), tuple. Nếu tham 2 tham số đầu vào không cùng độ dài thì hàm sẽ trả về lỗi StatisticsError.
* **proportional:** Là một tham số tùy chọn có kiểu dữ liệu boolean. Nếu proportional = True hàm sẽ thực hiện hồi quy tuyến tính theo tỷ lệ bằng cách tìm đường thẳng đi qua góc tọa độ (0, 0). Nếu proportional = False hàm sẽ tìm đường thẳng tối ưu nhất đi qua toàn bộ tham số đầu vào.
* **Tham số đầu ra:** Hàm trả về một tuple trong đó chứa 2 tham số hệ số góc (slope) và điểm cắt trục y của đường hồi quy (intercept).
* **Ví dụ:**
* Hàm linear\_regression nhận 2 tham số đầu vào và trả về kết quả hệ số góc của đường thẳng hồi quy và đường thẳng hồi quy ới trục y.

****

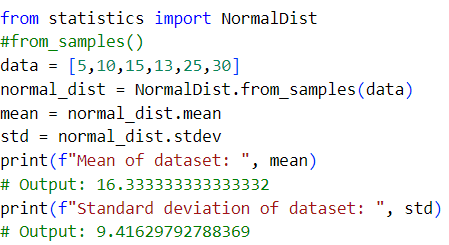
1.2.4 Lớp phân phối chuẩn

1.2.4.1 Các thuộc tính của NormalDist (chỉ đọc- không thể thay đổi)

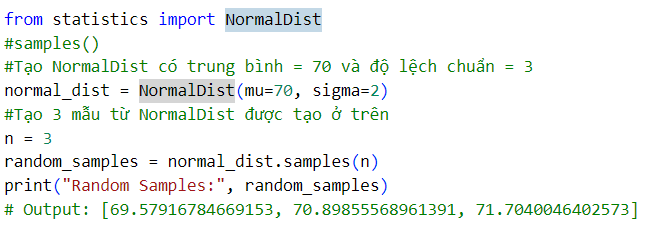
* **mean:** Thuộc tính trả về giá trị trung bình số học.
* **median:** Thuộc tính trả về giá trị trung vị.
* **mode:** Thuộc tính trả về giá trị có tần suất xuất hiện hiều nhất.
* **stdev:** Thuộc tính trả về độ lệch chuẩn.
* **variance:** Thuộc tínhrả về phương sai.

1.2.4.2 Các phương thức của NormalDist (chỉ đọc- không thể thay đổi)

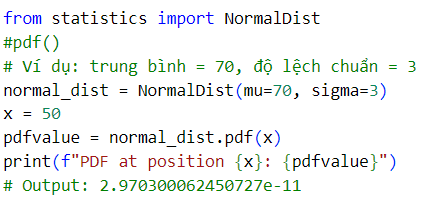
* **Phương thức from\_sample():**
* **Công dụng:** Dùng để tạo đối tượng phân phối chuẩn dựa trên phân phối chuẩn.
* **Ý nghĩa:** Cho phép tính toán thống kê phân phối bằng các thuộc tính có ở trên.
* **Ví dụ:**

****

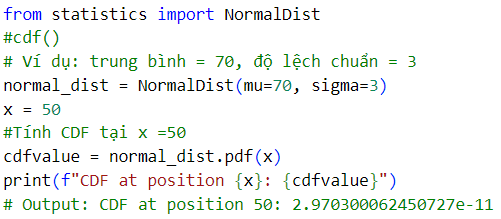
* **samples():**
* **Công dụng:** Tạo ra n phân phối ngãu nhiên từ phân phối chuẩn dựa trên dữ liệu đã có.
* **Ý nghĩa:** Cho phép tính toán thống kê phân phối bằng các thuộc tính có ở trên.
* **Ví dụ:**

****

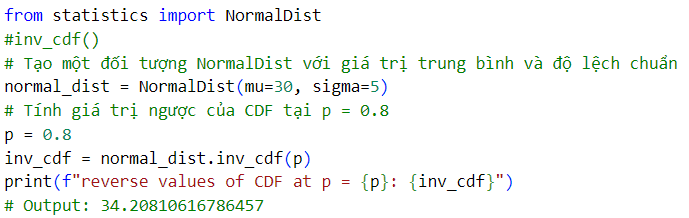
* **pdf():**
* **Công dụng:** Dùng để tính xác xuất mật độ phân phối chuẩn cụ thể một giá trị.
* **Tham số đầu vào:**  Là một giá trị muốn tính pdf ở đó.
* **Tham số đầu ra:** Giá trị phân phối xác xuất của biến truyền vào.
* **Ví dụ:**

****

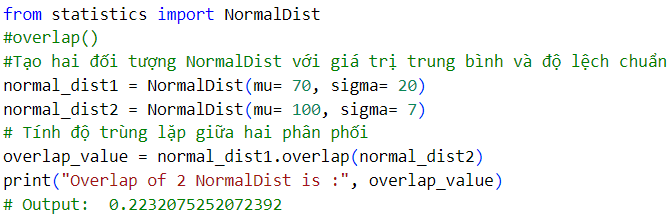
* **cdf():**
* **Công dụng:** Dùng để tính phân phối tích lũy chuẩn cụ thể một giá trị.
* **Tham số đầu vào:**  Là một giá trị muốn tính pdf ở đó.
* **Tham số đầu ra:** Giá trị phân phối tích lũy của biến truyền vào.
* **Ví dụ:**

****

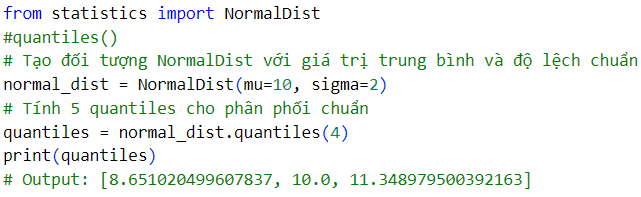
* **inv\_cdf():**
* **Công dụng:** Dùng để tính giá trị ngược của hàm phân phối tích lũy truyền vào.
* **Tham số đầu vào:**  Là một giá trị xác xuất p (float).
* **Tham số đầu ra:** Trả về một giá trị 0 – 1 cho diện tích giao giữa 2 hàm mật độ xác xuất.
* **Ví dụ:**

****

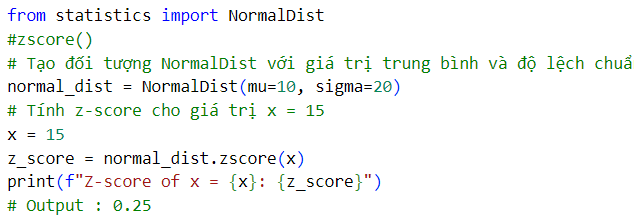
* **overlap():**
* **Công dụng:** Dùng để tính giá trị trùng lặp giữa 2 phân phối chuẩn truyền vào.
* **Tham số đầu vào:**  Là một hàm NormalDist khac2 muốn so sánhh trùng lặp.
* **Tham số đầu ra:** Trả về một giá trị 0 – 1 đại diện cho độ trùng lặp. Càng gần 1 thì 2 hàm càng có độ trùng lặp cao.
* **Ví dụ:**



* **quantiles():**
* **Công dụng:** Dùng để tính giá trị quantiles của phân phối chuẩn truyền vào.
* **Tham số đầu vào:**  Là n số lương quantiles cần tính.
* **Tham số đầu ra:** Trả về một danh sách chứa các giá trị quantiles.
* **Ví dụ:**



* **zcore():**
* **Công dụng:** Dùng để tính khoảng cách giữa x và giá trị trung bình của phân phối theo đơn vị của độ lệch chuẩn.
* **Tham số đầu vào:**  Là một số nguyên hoặc thực.
* **Tham số đầu ra:** Trả về khoảng cách giữa giá trị x và giá trị trung bình phân phối theo đơn vị độ lệch chuẩn.
* **Ví dụ:**



CHƯƠNG 2 – GIẢI THUẬT CÂN BẰNG LƯỢC ĐỒ (HISTOGRAM EQUALIZATION)

2.1 Tổng quan về phương pháp Histogram Equalization

2.1.1 Bài toán và mục tiêu

* Bài toán: Histogram Equalization là một kỹ thuật xử lí hình ảnh bằng cách cân bằng lại độ tương phản và phân bố cường độ sáng của ảnh.
* Mục tiêu: Làm cân đối độ sáng trên toàn ảnh, giúp cải thiện chất lượng ảnh và làm rõ các chi tiết trên ảnh.

2.1.2 Ràng buộc và điều kiện

* Ràng buộc: Phương pháp Histogram chỉ nhận ảnh đầu vào là ảnh xám (grayscale image), không tiếp nhận ảnh màu.
* Điều kiện: Ảnh cần có đủ số lượng pixel và mức độ khác nhau để cân bằng Histogram. Ảnh quá nhỏ sẽ không đủ dữ liệu.

2.1.3 Phương pháp và thuật toán

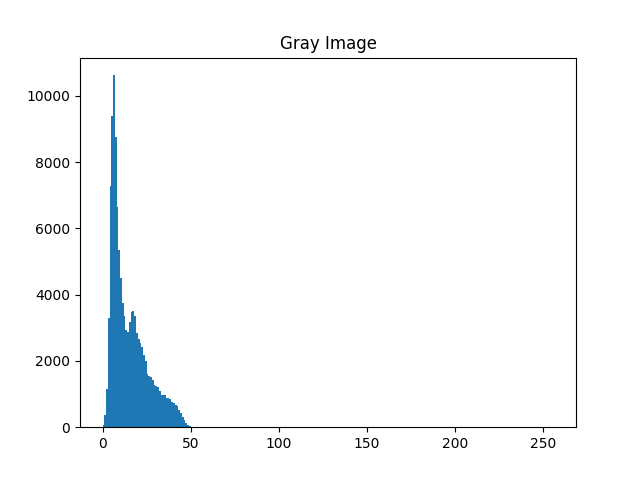
* Bước 1: Tính và chuẩn hóa histogram của ảnh gốc bằng cách đếm tần suất của các pixel.
* Bước 2: Tính hàm phân phối tích lũy (CDF) từ histogram bằng cách cộng tất cả các pixel ở từng mức xàm từ (0-255).
* Bước 3: Chuẩn hóa CDF để đảm bảo rằng các giá trị trong CDF nằm trong khoảng từ (0-255).
* Bước 4: Sử dụng CDF đã chuẩn hóa để ánh xạ lại mức sáng của từng pixel trong ảnh đầu vào để tạo ra ảnh cân bằng histogram.

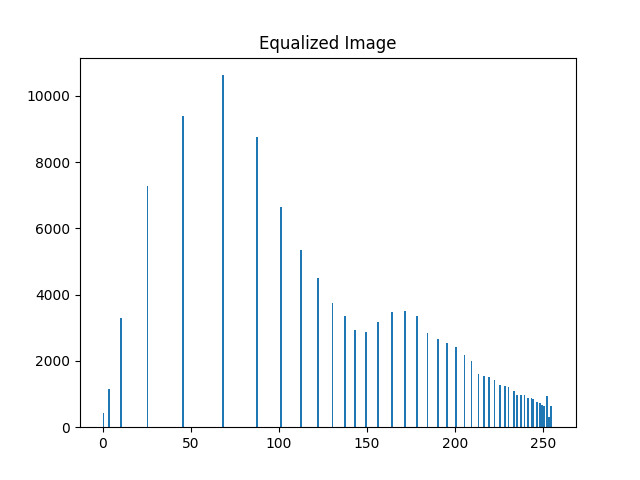
2.1.4 Nhận xét và phân tích

* Phương pháp Histogram Equalization là một phương pháp đơn giản, dễ thực hiện và hiệu quả trong việc cải thiện chất lượng hình ảnh. Tuy nhiên có một số nhược điểm cần cân nhắc khi áp dụng vào thực tế như: tạo ra hiệu ứng không tự nhiên cho một số loại ảnh và có thể làm tăng nhiễu cho ảnh có nhiễu ban đầu.

2.2 Hình ảnh minh họa

**Ảnh và lược đồ Histogram trước khi cân bằng**



**Ảnh và lược đồ Histogram sau khi cân bằng**

CHƯƠNG 3 – GIẢI THUẬT CÂN BẰNG LƯỢC ĐỒ (HISTOGRAM MATCHING)

3.1 Tổng quan về phương pháp Histogram Matching

3.1.1 Bài toán và mục tiêu

* Bài toán: Histogram Matching là một kỹ thuật xử lí hình ảnh bằng cách cập nhật độ tương phản và phân bố cường độ sáng của ảnh theo một ảnh tham chiếu.
* Mục tiêu: Cân bằng chất lượng hình ảnh của một ảnh theo ảnh tham chiếu.

3.1.2 Ràng buộc và điều kiện

* Ràng buộc: phương pháp Histogram Matching yêu cầu ảnh đầu vào và ảnh tham kênh màu để so sánh histogram.
* Điều kiện: Ảnh cần có đủ số lượng pixel và mức độ khác nhau để cân bằng Histogram. Ảnh quá nhỏ sẽ không đủ dữ liệu.

3.1.3 Phương pháp và thuật toán

* Bước 1: Tính và chuẩn hóa histogram của ảnh gốc và ảnh tham chiếu bằng cách đếm số pixel tương ứng với mỗi mức xám.
* Bước 2: Tính hàm phân phối tích lũy (CDF) từ 2 histogram đã chuẩn hóa bằng cách cộng tất cả các pixel ở từng mức xàm từ (0-255).
* Bước 3: Xây dựng bảng tra mapping bằng cách ánh xạ từ CDF của ảnh tham chiếu sang CDF của ảnh gốc.
* Bước 4: Sử dụng bảng tra mapping để ánh xạ lại mức sáng của từng pixel trong ảnh đầu vào để tạo ra ảnh cân bằng matching.

3.1.4 Nhận xét và phân tích

* Phương pháp Histogram Matching là một phương pháp đơn giản, dễ thực hiện và có thể thay đổi phân bố histogram của một ảnh linh hoạt dựa trên một ảnh tham chiếu. Phương pháp Histogram Matching chỉ xử lí được các thuộc tính toàn cục của ảnh, không xử lý được từng vùng ảnh.

3.2 Hình ảnh minh họa

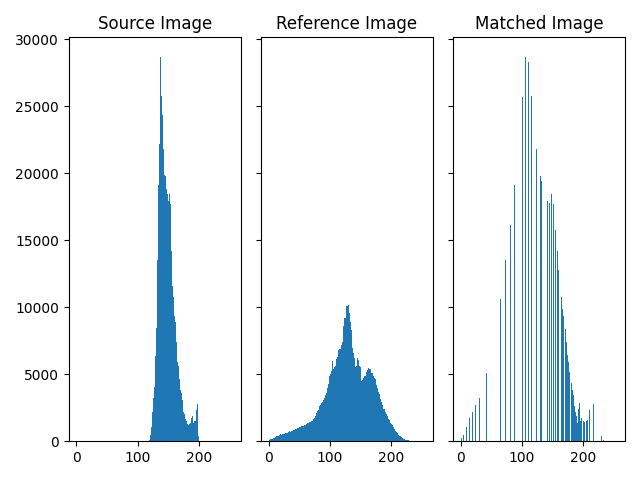




***Source Image***

***Reference Image***

***Matched Image***



CHƯƠNG 4 – THỰC HIỆN CHƯƠNG TRÌNH

4.1 Giải thuật cân bằng lược đồ (Histogram Equalization)

* + 1. Thực thi chương trình
* Dòng 1-3: Import thư viện OpenCV, thư viện Numpy và thư viện Mathplotlib.
* Dòng 6: Đọc ảnh đầu vào từ file “image.jpg” và lưu vào biến “image”.
* Dòng 9: Chuyển ảnh sang kênh màu xám và lưu vào biến “gray\_image”.
* Dòng 13: Lấy kích thước của ảnh bằng cách truy cập thuộc tính “shape” của ảnh xám. Biến “height” lưu chiều cao và biến “width” lưu chiều rộng của ảnh.
* Dòng 16: Khởi tạo một mảng “gray\_hist” gồm 256 phần tử.
* Dòng 17-20: Duyệt qua từng pixel của ảnh xám “gray\_image” đếm tần số xuất hiện của mỗi giá trị cường độ trong ảnh xám và lưu vào mảng “gray\_hist”
* Dòng 23: Chuẩn hóa giá trị histogram bằng cách chia cho tổng số pixel (height \* width).
* Dòng 26: Khởi tạo một mảng “cdf” gồm 256 phần tử.
* Dòng 27: Gán giá trị pixel đầu tiên trong mảng “gray\_hist” cho mảng “cdf”
* Dòng 28-29: Tính hàm phân phối tích lũy (CDF) của ảnh xám bằng cách cộng tất cả các pixel ở từng mức xám tương ứng trong mảng “gray\_hsit”.
* Dòng 32: Chuẩn hóa hàm phân phối tích lũy (CDF) của ảnh xám bằng cách lấy mảng cdf nhân với 255.
* Dòng 35: Tạo ra 1 ảnh có cùng kích thước với ảnh xám để chứa ảnh xám sau khi cân bằng.
* Dòng 36-39: Duyệt qua từng pixel trong ảnh trong ảnh xám, ánh xạ các giá trị pixel mới cho ảnh xám “gray\_image” từ mảng phân phối tích lũy đã chuẩn hóa (normalized\_cdf).
* Dòng 42-43: Hiển thị ảnh gốc và ảnh đã cân bằng bằng cửa sổ hiển thị của OpenCV. Chương trình sẽ hiển thị 2 ảnh song song, một cửa sổ cho ảnh gốc và một cửa sổ cho ảnh đã cân bằng.
* Dòng 46-54: Vẽ lược đồ histogram của ảnh gốc và ảnh đã cân bằng để so sánh.
* Dòng 57: Nhất một nút bất kì để thoát khỏi cửa sổ ảnh.
* Dòng 60: Đóng tất cả cửa sổ đang hiển thị

4.1.3 Hướng dẫn xây dựng và chạy chương trình

* Bước 1: Đảm bảo đã cài đặt thư viện OpenCV, NumPy và Mathplotlib trên máy tính.
* Bước 2: Tạo một file python mới để lưu đoạn code mẫu trên vào.
* Bước 3: Chuyển file ảnh bạn muốn cân bằng vào cùng thư mục với mã code python và đặt tên file ảnh là “image.jpg”
* Bước 4: Lưu và chạy mã code python trong file “522H006\_Part2.py”.
* Bước 5: Nhấn một phím bất kì để đóng các cửa sổ sau khi xem kết quả thực thi.

4.1.4 Thực nghiệm và kết luận

* Phương pháp Histogram Equalization là một phương pháp đơn giản, dễ thực hiện và hiệu quả trong việc cải thiện chất lượng hình ảnh.
* Tuy nhiên trong một số trường hợp phương pháp này có thể gây mất thông tin về độ sáng tổng thể của ảnh, tăng nhiều và làm mờ một số chi tiết của ảnh.
* Cần xem xét kĩ lưỡng và chọn phương pháp phù hợp cho từng loại ảnh khác nhau.

4.2 Giải thuật cân bằng lược đồ (Histogram Matching)

4.2.1 Thực thi chương trình

* Dòng 1-3: Import thư viện OpenCV, thư viện Numpy và thư viện Mathplotlib.
* Dòng 6-7: Đọc ảnh đầu gốc từ file “source\_image.jpg” và ảnh tham chiếu” reference\_image”. Và chuyển 2 ảnh sang kênh màu xám.
* Dòng 10-11: Lấy kích thước của ảnh gốc và ảnh tham chiếu bằng cách truy cập thuộc tính “shape”.
* Dòng 14: Khởi tạo một mảng “source\_hist” gồm 256 phần tử.
* Dòng 15-18: Duyệt qua từng pixel của ảnh gốc “source\_image” đếm tần số xuất hiện của mỗi giá trị cường độ trong ảnh xám và lưu vào mảng “source\_hist”
* Dòng 20: Khởi tạo một mảng “reference\_hist” gồm 256 phần tử.
* Dòng 21-24: Duyệt qua từng pixel của ảnh gốc “source\_image” đếm tần số xuất hiện của mỗi giá trị cường độ trong ảnh xám và lưu vào mảng “source\_hist”
* Dòng 27-28: Chuẩn hóa giá trị histogram của cả 2 ảnh bằng cách chia cho tổng số pixel (height \* width).
* Dòng 31: Khởi tạo một mảng “cdf\_source” gồm 256 phần tử.
* Dòng 32: Gán giá trị pixel đầu tiên trong mảng “source\_hist” cho mảng “cdf\_source”.
* Dòng 33-34: Tính hàm phân phối tích lũy (CDF) của ảnh xám bằng cách cộng tất cả các pixel ở từng mức xám tương ứng trong mảng “source\_hsit”.
* Dòng 36: Khởi tạo một mảng “cdf\_reference” gồm 256 phần tử.
* Dòng 37: Gán giá trị pixel đầu tiên trong mảng “reference\_hist” cho mảng “cdf\_reference”.
* Dòng 38-39: Tính hàm phân phối tích lũy (CDF) của ảnh xám bằng cách cộng tất cả các pixel ở từng mức xám tương ứng trong mảng “reference\_hsit”.
* Dòng 42-51: Tạo bảng mapping để tìm ra giá trị tương xứng giữa 2 ảnh và lưu vào bảng mapping.
* Dòng 54: Tạo ra 1 ảnh có cùng kích thước với ảnh xám để chứa ảnh xám sau khi cân bằng.
* Dòng 55-58: Duyệt qua từng pixel trong ảnh trong ảnh gốc, ánh xạ các giá trị pixel mới cho ảnh gốc “source\_image” từ bảng mapping.
* Dòng 61-63: Hiển thị ảnh gốc, ảnh tham chiếu và ảnh đã cân bằng bằng cửa sổ hiển thị của OpenCV. Chương trình sẽ hiển thị 3 ảnh song song.
* Dòng 65-79: Vẽ lược đồ histogram của ảnh gốc, ảnh tham chiếu và ảnh đã cân bằng để so sánh.
* Dòng 82: Nhất một nút bất kì để thoát khỏi cửa sổ ảnh.
* Dòng 85: Đóng tất cả cửa sổ đang hiển thị.

4.2.3 Hướng dẫn xây dựng và chạy chương trình

* Bước 1: Đảm bảo đã cài đặt thư viện OpenCV, NumPy và Mathplotlib trên máy tính.
* Bước 2: Tạo một file python mới để lưu đoạn code mẫu trên vào.
* Bước 3: Chuyển file ảnh bạn muốn cân bằng vào cùng thư mục với mã code python và đặt tên file ảnh gốc là là “source\_image.jpg” và file ảnh cần tham chiếu là “reference\_image”.
* Bước 4: Lưu chạy mã code python từ file “522H0006\_Part3.py”.
* Bước 5: Nhấn một phím bất kì để đóng các cửa sổ sau khi xem kết quả thực thi.

4.2.4 Thực nghiệm và kết luận

* Phương pháp Histogram Matching là một phương pháp đơn giản, dễ thực hiện và có thể thay đổi phân bố histogram của một ảnh linh hoạt dựa trên một ảnh tham chiếu.
* Tuy nhiên trong một số trường hợp phương pháp này có thể gây hiện tượng màu bị lệch khỏi vùng thực tế và không thích hợp cho ảnh có nhiều màu sắc phức tạp mất thông tin về độ sáng tổng thể của ảnh, tăng nhiều và làm mờ một số chi tiết của ảnh.
* Cần xem xét kỹ lưỡng và chọn phương pháp phù hợp cho từng loại ảnh khác nhau.

**TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

1. Thư viện Numpy trong Python: <https://viblo.asia/p/tim-hieu-ve-thu-vien-numpy-trong-pythonphan-1-Do7542QXZM6?fbclid=IwAR0kA0hfi0Vz93EVYNpVRK0luGW-x1YG-HGhqcOvGZMvFi2tLgsa89akahU>
2. Thuật toán cân bằng lược đồ (Histogram Equalization): <https://viblo.asia/p/tuan-3-histogram-histogram-equalization-3P0lPnxmKox>

<https://viblo.asia/p/xu-li-anh-thuat-toan-can-bang-histogram-anh-GrLZDOogKk0>

**Tiếng Anh**

1. Thư viện OpenCV trong python: <https://www.mygreatlearning.com/blog/opencv-tutorial-in-python/?fbclid=IwAR344lqpjfiHIsN6GjhcPGZdSQXYxX5jaH34HXW6zfKId70TrivCD4XY-2o>
2. Thư viện Numpy trong Python: <https://numpy.org/doc/stable/user/absolute_beginners.html>
3. Thư viện Mathplotlib trong Python: <https://www.geeksforgeeks.org/matplotlib-pyplot-hist-in-python/?fbclid=IwAR3OkXawckx2TATzNJILRUW8f8KiOUkYfAO-S-BN8-aDsob2jm4_EYYyY9w>
4. Thư viện thống kê toán học trong python: <https://docs.python.org/3/library/statistics.html>
5. Thuật toán cân bằng lược đồ (Histogram Equalization): <https://en.wikipedia.org/wiki/Histogram_equalization>
6. Thuật toán cân bằng lược đồ (Histogram Matching): <https://en.wikipedia.org/wiki/Histogram_matching>