Phần 1 : Các hàm thống kê toán học

Trong python, một số phép toán học có thể được thực hiện một cách dễ dàng bằng cách import các mô-đun để xác định các hàm khác nhau giúp nhiệm vụ của chúng ta dễ dàng hơn đặc biệt là trong thống kê và xử lý dữ liệu.

Các hàm thống kê có các chức năng hỗ trợ các kiểu dữ liệu int, float. Decimal và Fraction.

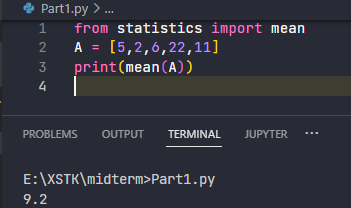
1. Hàm mean(data)

* Dùng để tính trung bình số học của một dãy số học.
* Ví dụ ta truyền vào mean() một mảng danh sách các số, hàm này sẽ trả về cho ta giá trị trung bình số học của dãy dữ liệu đư vào :
* Ví dụ : khi ta truyền vào một dãy số : 5 , 2, 6, 22, 11 thì khi sử dụng hàm mean() sẽ trả về cho ta giá trị là 9.2 có nghĩa là (5 + 2 + 6 + 22 + 11)/5 = 9.2
* Minh họa bằng code :

from statistics import mean

A = [5,2,6,22,11]

print(mean(A))



Bước 1 : thêm mô-đun statistics để gọi hàm mean()

Bước 2 : Thêm một danh sách giá trị vào A[]

Bước 3 : gọi hàm mean(A) để tính giá trị trung b

1. Hàm fmean(data,weight = None)

* Hàm fmean giống với mean() nhưng chạy nhanh hơn và luôn trả về giá trị kiểu float. Dữ liệu đầu vào có thể là một chuỗi hoặc lặp lại.
* Data là dữ liệu đưa vào, weight là trọng số, tuy nhiên khi đặt trọng số, thì số lượng phần tử weight phải bằng với số lượng phần tử của dữ liệu đưa vào. Trước đây phần trọng số không được hỗ trợ, nhưng từ phiên bản 3.11 thì python đã thêm trọng số vào fmean().
* Ví dụ khi ta cần tính tổng của một dãy số, tuy nhiên dãy số chỉ lấy một lượng phần trăm nhất định như điểm môn học, lương, chiết khấu … thì ta cần sử dụng hàm fmean().

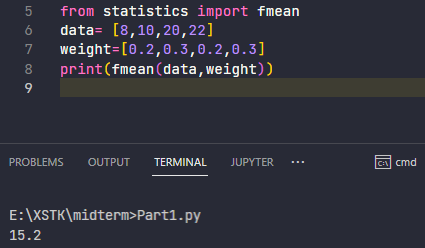
VD : cho một dãy dữ liệu gồm 8,10,20,22 nhưng các phần tử lại lấy với tỉ lệ phần trăm như sau : 0.2, 0.3,0.2,0.3 thì kết quả sẽ như sau :

from statistics import fmean

data= [8,10,20,22]

weight=[0.2,0.3,0.2,0.3]

print(fmean(data,weight))



Bước 1 : thêm mô-đun statistics để gọi hàm fmean()

Bước 2 : tạo một danh sách giá trị data[] và danh sách trọng số weight[]

Bước 3 : gọi hàm fmean(data, weight) để tính giá trị

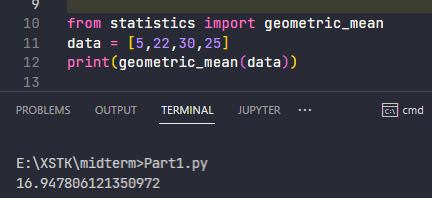
1. Hàm geometric\_mean(data)

* Hàm tính trung bình hình học, hàm geometric sẽ chuyển dữ liệu thành dạng float và tính giá trị trung bình hình học. Có nghĩa là căn bậc số lượn phần tử của tích các phần tử đó
* Hàm geometric dùng để tính xu hướng của giá trị trung tâm của dãy số.
* Ví dụ cho dãy số : 5, 22, 30, 25 tìm xu hướng trung tâm của dãy số, ta có đoạn mã như sau:

from statistics import geometric\_mean

data = [5,22,30,25]

print(geometric\_mean(data))



Bước 1 : thêm mô-đun statistics để gọi hàm geometric\_mean()

Bước 2 : tạo một danh sách giá trị data[]

Bước 3 : gọi geometric\_mean(data)để tính xu hướng của giá trị trung tâm trong dãy số

1. Hàm harmonic\_mean(data, weights=None)

Gần giống với hàm mean(), đều trả về giá trị trung bình của dữ liệu, một chuỗi hoặc có thể lặp lại của các số có giá trị thực.

Giá trị trung bình điều hòa là nghịch đảo của hàm arithmetic mean()

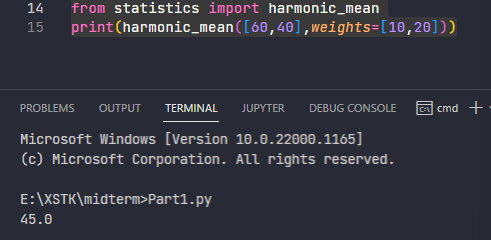
Giá trị trung bình điều hòa là thước đo trung tâm của bộ dữ liệu. Nó thường được sử dụng để lấy tỷ lệ hoặc tỷ lệ trung bình, ví dụ như vận tốc:

Giả sử một ô tô đi với vận tốc 60km/h trong 10km, sau đó giản xún còn 40km/h trong suốt 20km còn lại, tính tốc độ trung bình của ô tô

Giải :

from statistics import harmonic\_mean

print(harmonic\_mean([60,40],*weights*=[10,20]))



1. Hàm median(data)

Hàm median(data) hàm trung bị, trả về giá trị ở giữa của dữ liệu số. sử dụng phương pháp “giá trị trung bình của hai giá trị giữa”. dữ liệu có thể là một chuỗi hoặc lặp lại.

Hàm này chỉ trả về giá trị của vị trí trung tâm, và ít sự ảnh hưởng bởi sự hiện diện của các điểm ngoại lệ.

Khi số lượng phần tử là lẽ, thì dữ liệu ở vị trí chính giữa sẽ được trả về. Tuy nhiên nếu là chẵn thì hàm sẽ trả về giá trị trung bình của 2 giá trị tại 2 vị trí ở giữa.

Ví dụ : tính trung vị dãy số A = [1, 5, 7] và dãy B = [ 1, 4, 6, 7]

Giải:

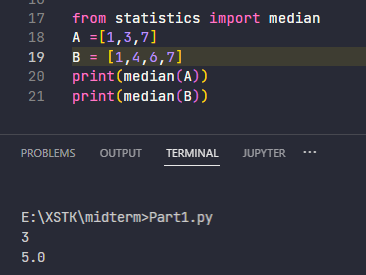
from statistics import median

A =[1,3,7]

B = [1,4,6,7]

print(median(A))

print(median(B))



1. Hàm median\_low(data)

Giống với hàm median nhưng trả về giá trị nhỏ hơn:

VD : tính trung vị thấp dãy số A = [1, 5, 7] và dãy B = [ 1, 4, 6, 7]

from statistics import median\_low

A =[1,3,7]

B = [1,4,6,7]

print(median\_low(A))

print(median\_low(B))



1. Hàm median\_high(data)

Hàm median\_high(data ) là hàm trái ngược lại với median\_low(data)

VD : tính trung vị cao dãy số A = [1, 5, 7] và dãy B = [ 1, 4, 6, 7]

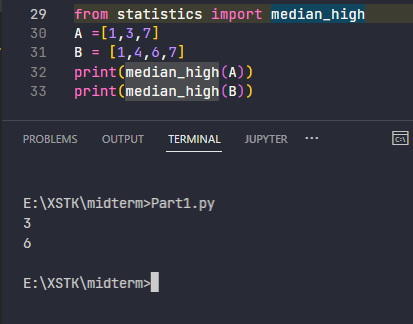
from statistics import median\_high

A =[1,3,7]

B = [1,4,6,7]

print(median\_high(A))

print(median\_high(B))



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

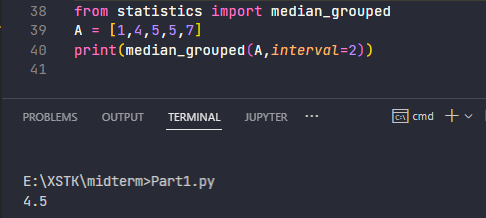
Trả về giá trị trung bình của dữ liệu liên tục được nhóm lại, được tính bằng phân vị thứ 50 sử dụng phép nội suy.

VD : cho bộ dũ liệu A = [1, 4, 5, 5, 7] và interval = 2

from statistics import median\_grouped

A = [1,4,5,5,7]

print(median\_grouped(A,*interval*=2))



1. Hàm mode(data)

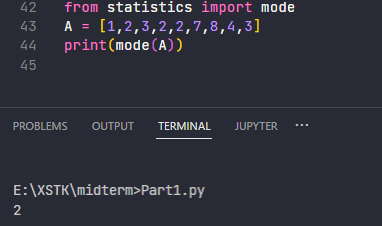
Hàm mode(data) trả về giá trị xuất hiện nhiều lần nhất trong bộ dữ liệu

Ví dụ cho dãy dữ liệu A = [1,2,3,2,2,7,8,4,3] tìm mode của bộ dữ liệu

from statistics import mode

A = [1,2,3,2,2,7,8,4,3]

print(mode(A))



1. Hàm multimode(data)

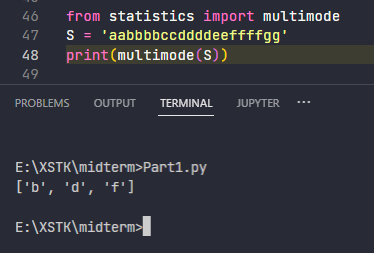
Hàm multimode(data) dùng để trả về dnah sách các giá trị thường xuyên nhất theo thứ tự lần đầu tiên chúng gặp trong bộ dữ liệu

VD : cho chuỗi S = 'aabbbbccddddeeffffgg' tìm multimode của S:

from statistics import multimode

S = 'aabbbbccddddeeffffgg'

print(multimode(S))



1. Hàm pstdev(data, mu = None)

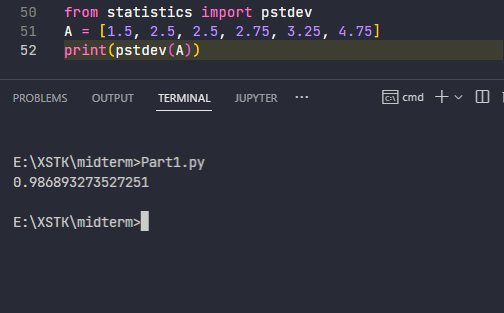
Hàm pstdev(data, mu = None) trả về độ lệch chuẩn tổng thể (có nghĩa là căn bậc hai phương sai tổng thể).

VD cho dãy A = [1.5, 2.5, 2.5, 2.75, 3.25, 4.75] tìm độ lệch chuẩn của A:

from statistics import pstdev

A = [1.5, 2.5, 2.5, 2.75, 3.25, 4.75]

print(pstdev(A))



1. Hàm pvariance(data, mu = Node)

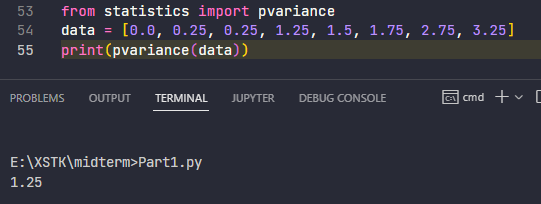
Hàm pvariance(data, mu = Node) trả về phương sai tổng thể của dữ liệu, một chuối rỗng hoặc lặp lại các số có giá trị thực. Phương sai là thước đo tính biến thiên của dữ liệu. Phương sai nhỏ thì dữ liệu được trãi rộng, còn phương sai nhỏ thì tập hơn chặt chẽ hơn xunh quanh giá trị trung bình.

VD : tìm phương sai của bộ dữ liệu data = [0.0, 0.25, 0.25, 1.25, 1.5, 1.75, 2.75, 3.25]:

from statistics import pvariance

data = [0.0, 0.25, 0.25, 1.25, 1.5, 1.75, 2.75, 3.25]

print(pvariance(data))



1. Hàm stdev(data, xbar = None)

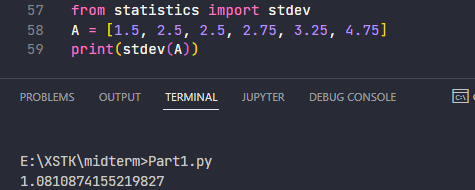
Hàm stdev(data, xbar = None) trả về độ lệch chuẩn của mẫu, bằng căn bậc hai của phương sai mẫu.

VD : tìm phương sai mẫu của dãy A = [1.5, 2.5, 2.5, 2.75, 3.25, 4.75]

from statistics import stdev

A = [1.5, 2.5, 2.5, 2.75, 3.25, 4.75]

print(stdev(A))



1. Hàm variance(data, xbar = None)

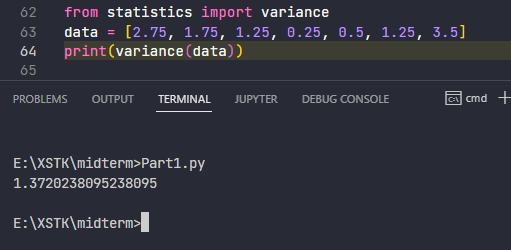
Hàm variance(data, xbar = None) trả về phương sai của mẫu, có thể lặp lại của ít nhất hai số thực. Phương sai, hay thời điểm thứ hai về giá trị trung bình, là thước đo tính biến thiên trãi rộng của bộ dữ liệu. Một phương sai lớn chỉ ra rằng dữ liệu được trải rộng; một phương sai nhỏ cho thấy nó được tập hợp chặt chẽ xung quanh giá trị trung bình.

VD : tìm phương sai mẫu của bộ dữ liệu sau A = [2.75, 1.75, 1.25, 0.25, 0.5, 1.25, 3.5]

from statistics import variance

data = [2.75, 1.75, 1.25, 0.25, 0.5, 1.25, 3.5]

print(variance(data))



1. Hàm quantiles(data, \*, n=4, method='exclusive')

Hàm số lượng tử dùng để chia dữ liệu thành n khoảng liên tục với xác suất bằng nhau. Trả về danh sách các điểm cắt tách các khoảng n – 1.

Đặt n thành 4 cho phần tử (mặc định). Đặt n = 10 cho các deciles. Đặt n = 100 cho các phân vị, cung cấp 99 điểm cắt để phân tách thành 100 nhóm có kích thước bằng nhau.

Để kết quả có ý nghĩa, số lượng điểm dữ liệu phải lớn hơn n.

Ví dụ về hàm quantiles:

Cho bộ dữ liệu data = [105, 129, 87, 86, 111, 111, 89, 81, 108, 92, 110,

100, 75, 105, 103, 109, 76, 119, 99, 91, 103, 129,

106, 101, 84, 111, 74, 87, 86, 103, 103, 106, 86,

111, 75, 87, 102, 121, 111, 88, 89, 101, 106, 95,

103, 107, 101, 81, 109, 104]

from statistics import quantiles

data = [105, 129, 87, 86, 111, 111, 89, 81, 108, 92, 110,

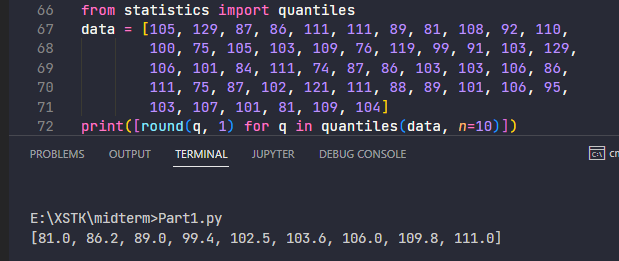
        100, 75, 105, 103, 109, 76, 119, 99, 91, 103, 129,

        106, 101, 84, 111, 74, 87, 86, 103, 103, 106, 86,

        111, 75, 87, 102, 121, 111, 88, 89, 101, 106, 95,

        103, 107, 101, 81, 109, 104]

print([round(q, 1) for q in quantiles(data, *n*=10)])



1. Hàm covariance(x, y, /)

Hàm trả về phương sai mẫu của hai đầu vào x và y. Hiệp phương sai là thước đo sự thay đổi chung của hai đầu vào.

Cả hai đầu vào phải cùng độ dài.

VD: cho x = [1, 2, 3, 4, 5, 6, 7, 8, 9]

y = [1, 2, 3, 1, 2, 3, 1, 2, 3]

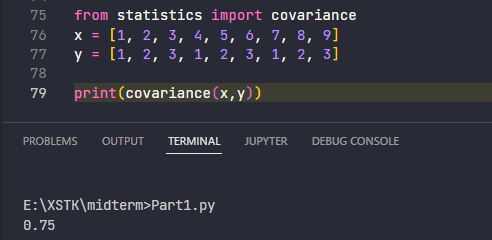
tìm phương sai mẫu của x,y

from statistics import covariance

x = [1, 2, 3, 4, 5, 6, 7, 8, 9]

y = [1, 2, 3, 1, 2, 3, 1, 2, 3]

print(covariance(x,y))



1. Hàm correlation(x, y, /)

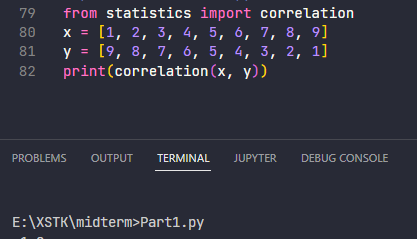
Hàm correlation dùng để trả về hệ số tương quan của Pearson cho hai đầu vào. Hệ số tương quan của Pearson nhận các giá trị từ -1 đến 1. Nó đo độ mạnh và hướng của mối quan hệ tuyến tính, trong đó +1 có nghĩa là mối quan hệ tuyến tính rất mạnh, tích cực, -1 rất mạnh, mối quan hệ tuyến tính tiêu cực và 0 không có mối quan hệ tuyến tính

from statistics import correlation

x = [1, 2, 3, 4, 5, 6, 7, 8, 9]

y = [9, 8, 7, 6, 5, 4, 3, 2, 1]

print(correlation(x, y))



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

Hồi quy tuyến tính trả về độ dốc và hệ số chặn của các tham số hồi quy tuyến tính đơn giản được ước tính bằng cách sử dụng bình phương nhỏ nhất thông thường.

Cả 2 dữ liệu phải có cùng độ dài và biến độc lập x không được là hằng số.

Ví dụ: chúng ta có thể sử dụng ngày phát hành của các bộ phim Monty Python để dự đoán số lượng tích lũy các bộ phim Monty Python sẽ được sản xuất vào năm 2019 với giả định rằng chúng đã giữ được tốc độ.

from statistics import linear\_regression

year = [1971, 1975, 1979, 1982, 1983]

films\_total = [1, 2, 3, 4, 5]

slope, intercept = linear\_regression(year, films\_total)

print(round(slope \* 2019 + intercept))

