**Mean and variance**

import numpy as np

x = np.array([1,2,3,4,5])

mu = np.mean(x)

y= x-mu

#Traspose = y.T

#matrix multiplication = y.dot(y.T)

var= (y.dot(y.T))/(len(x)-1)

print(var)

**Output**

﻿the mean is: 3.0

the variance is: 2.0

**Observance:**

The mean is the average of the elements of the array i.e, the sum of all the elements divided by the total number of elements. The variance is the sum of square of difference of each element with the mean divided by total number of elements. It is calculated using a single array and its transpose

**Covariance**

import numpy as np  
x1 = np.array([2,3,1,0])   
x2 = np.array([4,1,0,2])   
mu1 =  np.mean(x1)  
mu2 =  np.mean(x2)  
y1= x1-mu1  
y2= x2-mu2  
n=len(x1)  
cov= (y1.dot(y2.T))/n  
print(cov)

**Output**

﻿ ﻿the covariance is : 0.125

**Observance:**

The variance is the sum of square of difference of each element with the mean divided by total number of elements it is calculated using a single array and its transpose. Whereas the covariance with 2 different arrays using the same formula as variance but instead of square we multiply the difference of elements of array 1 with its mean and difference of elements of array 2 with its mean divided by the length of the array 1.

**Correlation**

﻿import numpy as np

x1 = np.array([2,3,1,0])

x2 = np.array([4,1,0,2])

mu1 = np.mean(x1)

mu2 = np.mean(x2)

y1= x1-mu1

y2= x2-mu2

n=len(x1)

var1=(y1.dot(y1.T))/n

var2=(y2.dot(y2.T))/n

cov= (y1.dot(y2.T))/n

z=(var1\*var2)\*\*(0.5)

print("the correlation is :",cov/(z))

**Output**

﻿the correlation is : 0.07559289460184544

**Observance:**

The correlation is covariance divided by the square root variance of array 1 and array 2. Covariance gives how they are related and correlation gives how much they are differed

**Input**

m = int(input("Enter value of m"))

n = int(input("Enter value of n"))

mat = np.random.randint(10,size=(m,n))

print(mat)

def var(arr):

arr\_mean = np.mean(arr)

arri = []

arr\_sum = 0

i = 0

j = 0

for i in range(len(arr)):

arri.append(np.round((arr[i]-arr\_mean)\*(arr[i]-arr\_mean),3))

for j in range(len(arr)):

arr\_sum += arri[j]

arr\_var = arr\_sum/(len(arr)-1)

return np.round(arr\_var,3)

def corr(arr1,arr2):

cov = covar(arr1,arr2)

var1 = var(arr1)

var2 = var(arr2)

cor = np.round((cov/(math.sqrt(var1\*var2))),3)

return cor

def covar(arr1,arr2):

arri1 = []

i1 = 0

j1 = 0

arr1\_mean = np.mean(arr1)

arr2\_mean = np.mean(arr2)

for i1 in range(len(arr1)):

arri1.append(np.round((arr1[i1]-arr1\_mean)))

arri2 = []

i2 = 0

j2 = 0

for i2 in range(len(arr1)):

arri2.append(np.round((arr2[i2]-arr2\_mean)))

arr\_dot = np.dot(arri1,arri2)

arr\_cov = arr\_dot/(len(arr1)-1)

return np.round(arr\_cov,3)

def cov\_mat(arr):

x = []

y = []

i = j = 0

p = input("Enter the feature column number (i) to be computed: ")

q = input("Enter the feature column number (j) to be computed: ")

for i in range(m):

for j in range(i,i+1):

x.append(arr[i][j])

for i in range(m):

for j in range(j,j+1):

y.append(arr[i][j])

print("The feature column",p,":",x)

print("The feature column",q,":",y)

covm = [[covar(x,x),covar(x,y)],[covar(y,x),covar(y,y)]]

corm = [[corr(x,x),corr(x,y)],[corr(y,x),corr(y,y)]]

return covm,corm

cvm,crr = cov\_mat(mat)

print("The covariance matrix computed: ",cvm)

print("The correlation matrix computed: ",crr)

**`OUTPUT:**

Enter value of m12

Enter value of n12

[[0 7 9 4 0 8 8 6 5 8 5 9]

[9 7 9 8 5 5 9 1 5 2 7 9]

[1 1 5 3 9 0 4 3 3 0 5 3]

[2 1 3 5 2 3 8 3 0 5 3 0]

[1 1 4 4 8 8 1 2 1 2 1 7]

[8 0 0 5 8 8 8 0 1 9 8 9]

[4 2 2 9 6 4 9 8 6 0 6 0]

[5 9 7 9 7 0 4 0 7 2 4 4]

[9 3 6 1 2 1 6 7 0 1 4 2]

[0 3 3 1 6 0 7 7 4 8 6 5]

[9 4 0 0 6 8 9 2 0 0 9 9]

[9 1 8 6 5 0 1 9 3 5 2 3]]

Enter the feature column number (i) to be computed: 1

Enter the feature column number (j) to be computed: 1

The feature column 1 : [0, 7, 5, 5, 8, 8, 9, 0, 0, 8, 9, 3]

The feature column 1 : [9, 9, 3, 0, 7, 9, 0, 4, 2, 5, 9, 3]

The covariance matrix computed: [[12.909000000000001, 2.3639999999999999], [2.

3639999999999999, 12.364000000000001]]

The correlation matrix computed: [[1.002, 0.187], [0.187, 1.0]]

**Observation:**

These are very insightful matrices. Each member of the correlation matrix is the strength of dependency of two variables of the vector. These are the best ways to find the relation of each element with the other.