Tasca 4

May 13, 2021

1 IT Academy - Data Science amb Python

1.1 Tasca 4: Programació numèrica amb NumPy

[1]:
```
import numpy as np
from numpy import random
```

1.1.1 Exercici 1

[2]:
```
arr1 = random.randint(100, size=(6))
arr2 = random.randint(100, size = (3, 3))
```

[3]:
```
def arr_summary(arr):
    print(arr)
    if arr.ndim == 1:
        print("Mitja: ", round(np.mean(arr),2))
        print("Mitjana: ", np.median(arr))
        print("Variància: ", round(np.var(arr),2))
        print("Desviació standard: ", round(np.std(arr), 2))
    else:
        print("La matriu introduïda te més d'una dimensió.")
```

[4]:
```
arr_summary(arr1)
```

```
[76 86 52 36  7 42]
Mitja:  49.83
Mitjana:  47.0
Variància:  680.81
Desviació standard:  26.09
```

[5]:
```
arr_summary(arr2)
```

```
[[ 6 32 59]
 [43 21  2]
 [20  8  5]]
La matriu introduïda te més d'una dimensió.
```

1.1.2 Exercici 2

[6]:
```
def random_array(size):
    return np.random.randint(101, size = size)
```

[7]:
```
arr3 = random_array((3, 3, 3))
print(arr3)
```

```
[[[ 56  48  42]
  [  8  23  55]
  [100  31  71]]

 [[ 62  98 100]
  [ 90  53   8]
  [ 15  49  69]]

 [[ 37  83  43]
  [ 22  95  90]
  [ 43  89  94]]]
```

1.1.3 Exercici 3

[8]:
```
def total_sum(arr):
    if arr.ndim == 2:
        print("Total per files: ", np.sum(arr, axis = 1))
        print("Total per columnes: ", np.sum(arr, axis = 0))
```

[9]:
```
arr4 = random_array((3,3))
print(arr4)
```

```
[[47  4  2]
 [31 72 15]
 [38 92 12]]
```

[10]:
```
total_sum(arr4)
```

```
Total per files:  [ 53 118 142]
Total per columnes:  [116 168  29]
```

1.1.4 Exercici 4

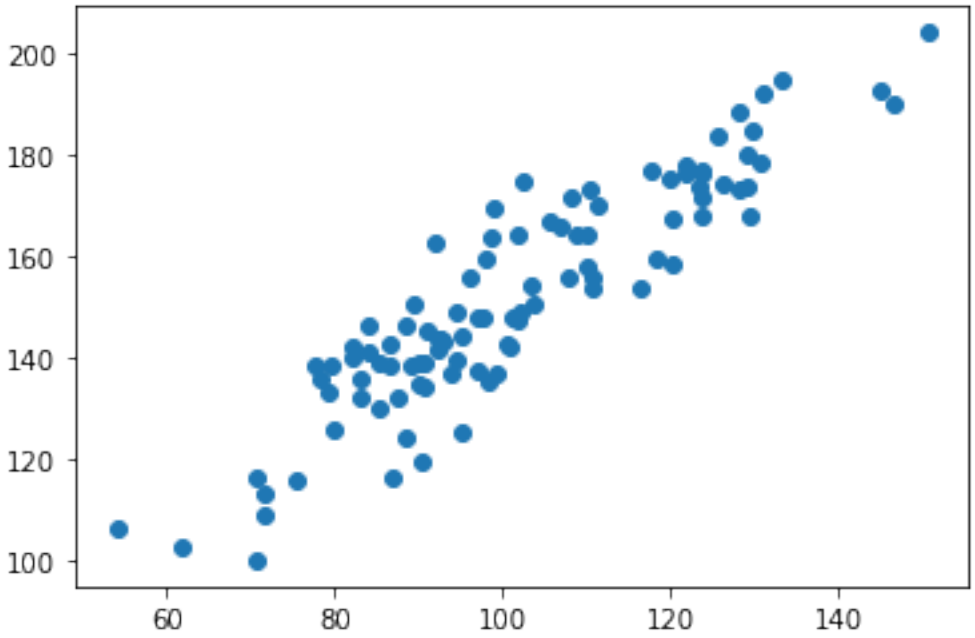
Coeficient de Correlació de Pearson El Coeficient de Correlació de Pearson, també conegut com el Coeficient de Correlació Producte-Moment de Pearson, és un índex que mesura la relació lineal entre dues variables quantitatives.

El Coeficient de Correlació ( ) entre dues variables aleatòries X i Y és el quocient de la seva covariància pel producte de les seves desviacions estàndard.

0. Preparació de les dades

[11]:
```
from matplotlib import pyplot
from math import sqrt

# Creació del Generador
rng = random.default_rng()

# Creació dels conjunts de dades de X i de Y
data1 = 20 * rng.standard_normal(100) + 100
data2 = data1 + (10 * rng.standard_normal(100) + 50)

# Resum estadístic de les dades
print('data1: mean=%.3f stdv=%.3f' % (np.mean(data1), np.std(data1)))
print('data2: mean=%.3f stdv=%.3f' % (np.mean(data2), np.std(data2)))

# Visualització de les dades
pyplot.scatter(data1, data2)
pyplot.show()
```

```
data1: mean=101.564 stdv=19.211
data2: mean=151.865 stdv=21.922
```



1. Covariància entre X i Y La covariància és una mesura de dispersió conjunta de dues variables estadístiques. La covariància mesura la variació total de dues variables aleatòries respecte els seus valors esperats.

Amb aquesta mesura només es pot mesurar la direcció de la relació (és a dir, si les variables

tendeixen a moure's en tàndem o mostren una relació inversa). No obstant això, no indica la força de la relació, ni la dependència entre les variables.

[12]:
```
x_mean = np.mean(data1)
y_mean = np.mean(data2)
n = data1.size

x_distance = [xi - x_mean for xi in data1]
y_distance = [yi - y_mean for yi in data2]
cov_xy = sum(np.multiply(x_distance, y_distance)) / (n-1)
print("Covariància entre X i Y: %.3f" % cov_xy)
```

```
Covariància entre X i Y: 388.420
```

2. Desviació estàndard La desviació estàndard ( ), també coneguda com a desviació tipus o desviació típica, és una mesura que quantifica quanta dispersió hi ha respecte a la mitjana.

Una desviació tipus baixa indica que els punts de dades tendeixen a ser propers a la mitjana, mentre que una desviació tipus alta indica que les dades s'estenen al llarg d'un gran rang de valors.

La desviació estándard d'un conjunt de dades és l'arrell quadrada de la seva variància. Una propietat útil de la desviació tipus és que, a diferència de la variància, està expressada en les mateixes unitats que les de les dades.

[13]:
```
x_distance_squared = [(xi - x_mean)**2 for xi in data1]
x_variance = np.sum(x_distance_squared) / n
x_std = sqrt(x_variance)
print("Desviació estàndard de X: %.3f" % x_std)
```

```
Desviació estàndard de X: 19.211
```

[14]:
```
y_distance_squared = [(yi - y_mean)**2 for yi in data2]
y_variance = np.sum(y_distance_squared) / n
y_std = sqrt(y_variance)
print("Desviació estàndard de Y: %.3f" % y_std)
```

```
Desviació estàndard de Y: 21.922
```

3. Càlcul de la correlació de Pearson La correlació mesura la força de la relació entre variables. La correlació és la mesura escalada de la covariància. No té cap dimensió, és a dir, el coeficient de correlació és sempre un valor pur i no es mesura en cap unitat.

El seu valor, amb un rang de [-1, 1], pot interpretar-se de la següent manera: * +1.0 - Correlació positiva completa * +0.8 - Correlació positiva forta * +0.6 - Correlació positiva moderada * 0.0 - Sense cap correlació * -0.6 - Correlació negativa moderada * -0.8 - Correlació negativa forta * -1.0 - Correlació negativa completa

El rendiment d'alguns algoritmes pot deteriorar-se si dues o més variables estan estretament relacionades, anomenades multicol · linealitat. Per exemple, el model de regressió lineal múltiple assumeix que no existeix multicol · linealitat perfecta en el model.

[15]:
```
coefcorr = cov_xy / (x_std * y_std)
print("Coeficient de Correlació entre X i Y: %.3f" % coefcorr)
```

```
Coeficient de Correlació entre X i Y: 0.922
```

4. Creació de la funció

[16]:
```
def coefcorr(x, y):

    x_distance = [xi - np.mean(x) for xi in x]
    y_distance = [yi - np.mean(y) for yi in y]
    cov_xy = sum(np.multiply(x_distance, y_distance)) / (x.size-1)

    x_distance_square = np.power(x_distance, 2)
    x_std = sqrt(np.sum(x_distance_square / x.size))

    y_distance_square = np.power(y_distance, 2)
    y_std = sqrt(np.sum(y_distance_square/ y.size))

    return cov_xy / (x_std * y_std)
```

[17]:
```
coefcorr(data1, data2)
```

[17]:
```
0.9222953166030119
```

5. Comprobació

[18]:
```
np.corrcoef(data1, data2)
```

[18]:
```
array([[1.        , 0.91307236],
       [0.91307236, 1.        ]])
```