Tasca2A

February 3, 2021

1 Estructura de una Matriu

Anem a practicar i a familiaritzar-nos amb l'estructure de Matrius, dimensio, forma, vectorització i Broadcasting .

[1]:
```
import numpy as np
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
```

1.0.1 Nivell 1

Treballem els conceptes de l'estructura d'una matriu, dimensió, eixos i la vectorització que ens permet reduir l'ús de for loops en operacions aritmètiques o matemàtiques.

Exercici 1

Crea un np.array de una dimension, que inclogui l'almenys 8 nombres sencers, data type int64. Mostra la dimensió i la forma de la matriu.

[2]:
```
## Random vector
x = np.random.randint(10, size = 8, dtype = "int64")
print (x)
```

```
[6 2 5 9 1 6 4 5]
```

[3]:
```
## Dimensió
dim = x.ndim
print (dim)
```

```
1
```

[4]:
```
## Forma
shape = x.shape
print (shape)
```

```
(8,)
```

Exercici 2

De la matriu de l'exercici 1, calcula el valor mitjà dels valors introduïts i resta la mitjana resultant de cada un dels valors de la matriu.

[5]:
```
## Mitja
mean = x.mean()
print (mean)
```

```
4.75
```

[6]:
```
## Vector transformat
x_trans = x - mean
print (x_trans)
```

```
[ 1.25 -2.75  0.25  4.25 -3.75  1.25 -0.75  0.25]
```

Exercici 3

Crea una matriu bidimensional amb una forma de 5 x 5. Extreu el valor màxim de la matriu, i els valors màxims de cadascun dels seus eixos.

[7]:
```
## Random Matrix
matriu = np.random.randint(10, size = (5,5), dtype = "int64")
print (matriu)
```

```
[[2 8 9 2 5]
 [0 4 1 9 9]
 [6 3 4 8 0]
 [3 7 7 7 7]
 [7 3 4 9 9]]
```

[8]:
```
## Maxim de la Matriu
maxim = matriu.max()
print (maxim)
```

```
9
```

[9]:
```
## Maxims x files
max_f = matriu.max(axis = 1)
print (max_f)
```

```
[9 9 8 7 9]
```

[10]:
```
## Maxims per columnes
max_c = matriu.max(axis = 0)
print (max_c)
```

```
[7 8 9 9 9]
```

1.0.2 Nivell 2

Treballem els conceptes de l'estructura d'una matriu, Broadcasting, indexació, Mask.

Exercici 4

Mostreu-me amb exemples de diferents matrius, la regla fonamental de Broadcasting que diu : “les matrius es poden transmetre / broadcast entre si si les seves dimensions coincideixen o si una de les matrius té una mida d’1”.

Exemple 1:

[11]:
```
x = np.arange(4)
print ("shape:", x.shape)
print (x)
```

```
shape: (4,)
[0 1 2 3]
```

[12]:
```
y = np.ones(5)
print ("shape:", y.shape)
print (y)
```

```
shape: (5,)
[1. 1. 1. 1. 1.]
```

[13]:
```
## Dimensions error
x + y
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-13-0a0b4de110ab> in <module>
      1 ## Dimensions error
----> 2 x + y

ValueError: operands could not be broadcast together with shapes (4,) (5,)
```

[14]:
```
xx = x.reshape(4,1)
print ("shape:", xx.shape)
print (xx)
```

```
shape: (4, 1)
[[0]
 [1]
 [2]
 [3]]
```

[15]:
```
print ("shape:", (xx+y).shape)
print (xx + y)
```

```
shape: (4, 5)
[[1. 1. 1. 1. 1.]
 [2. 2. 2. 2. 2.]
```

```
 [3. 3. 3. 3. 3.]
 [4. 4. 4. 4. 4.]]
```

Exemple 2:

[16]:
```
x = np.arange(4)
print ("shape:", x.shape)
print (x)
```

```
shape: (4,)
[0 1 2 3]
```

[17]:
```
z = np.ones((3,4))
print ("shape:", z.shape)
print (z)
```

```
shape: (3, 4)
[[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]
```

[18]:
```
print ("shape:", (x+z).shape)
print (x + z)
```

```
shape: (3, 4)
[[1. 2. 3. 4.]
 [1. 2. 3. 4.]
 [1. 2. 3. 4.]]
```

Exemple 3:

[19]:
```
a = np.array([0.0, 10.0, 20.0, 30.0])
print ("shape:", a.shape)
print (a)
```

```
shape: (4,)
[ 0. 10. 20. 30.]
```

[20]:
```
b = np.array([1.0, 2.0, 3.0])
print ("shape:", b.shape)
print (b)
```

```
shape: (3,)
[1. 2. 3.]
```

[21]:
```
c = a[:, np.newaxis] + b
print ("shape:", c.shape)
print (c)
```

```
shape: (4, 3)
[[ 1.  2.  3.]
 [11. 12. 13.]
 [21. 22. 23.]
 [31. 32. 33.]]
```

Exercici 5

Utilitza la Indexació per extreure els valors d'una columna i una fila de la matriu. I suma els seus valors.

[22]:
```
## Random Matrix
matriu = np.random.randint(1, 5, size = (5,5), dtype = "int64")
print (matriu)
```

```
[[2 3 4 4 1]
 [3 2 4 3 4]
 [4 3 4 4 2]
 [4 1 3 2 1]
 [2 4 1 4 3]]
```

[23]:
```
## Segona fila
mat_f = matriu[1,:]
print (mat_f)
```

```
[3 2 4 3 4]
```

[24]:
```
## Tercera columna
mat_c = matriu[:,2]
print (mat_c)
```

```
[4 4 4 3 1]
```

[25]:
```
## Suma
suma = mat_f + mat_c
print (suma)
```

```
[7 6 8 6 5]
```

[26]:
```
## Suma Total
print (sum(suma))
```

```
32
```

Exercici 6

Mask la matriu anterior, realitzeu un càlcul booleà vectoritzat, agafant cada element i comprovant si es divideix uniformement per quatre. Això retorna una matriu de mask de la mateixa forma amb els resultats elementals del càlcul.

[27]:
```
## Matriu mask
mat_mask = matriu % 4 == 0
print (mat_mask)
```

```
[[False False  True  True False]
 [False False  True False  True]
 [ True False  True  True False]
 [ True False False False False]
 [False  True False  True False]]
```

Exercici 7

A continuació, utilitzeu aquesta màscara per indexar a la matriu de números original. Això fa que la matriu perdi la seva forma original, reduint-la a una dimensió, però encara obteniu les dades que esteu cercant.

[28]:
```
## Vector de 4
vec = matriu[mat_mask]
```

1.0.3 Nivell 3

Manipulació d'imatges amb Matplotlib.

Carregareu qualsevol imatge (jpg, png ..) amb Matplotlib. adoneu-vos que les imatges RGB (Red, Green, Blue) són realment només amplades × alçades × 3 matrius (tres canals Vermell, Verd i Blau), una per cada color de nombres enters int8, manipuleu aquests bytes i torneu a utilitzar Matplotlib per desar la imatge modificada un cop hàgiu acabat.

Ajuda:Importeu, import matplotlib.image as mpimg. estudieu el metodde mpimg.imread(()

Exercici 8

- Mostreu-me a veure que passa quan eliminem el canal G Verd o B Blau. Hauries de utizar la indexacion per selecciones el canal que voleu anul · lar.
- Utilitzar el mètode, mpimg.imsave () de la llibreria importada, per guardar les imatges modificades i que haureu de pujar al vostre repositori a github.

[29]:
```
## Read Image
img = mpimg.imread('Jirafa3.jpg')
print(img.shape)
```

```
(547, 539, 3)
```

[30]:
```
## Plot Images

fig, axs = plt.subplots(2, 2, figsize=(15, 15));

for ax in axs.flat:
    ax.axis("off")
```

```
## Normal
axs[0,0].imshow(img);

## Read Channel Only
img_red = img[:, :, 0]
axs[0,1].imshow(img_red, cmap = "Reds_r");

## Green Channel Only
img_green = img[:, :, 1]
axs[1,0].imshow(img_green, cmap = "Greens_r");

## Blue Channel Only
img_blue = img[:, :, 2]
axs[1,1].imshow(img_blue, cmap = "Blues_r");

##
```

Si eliminamos dos canales y nos quedamos con uno, el imshow que obtenemos muestra la intensidad de ese canal. En esta figura observamos la imagen original y como estan distribuidos los rojos, los verdes y los azules en la imagen.

[31]:
```
## Save Img
mpimg.imsave('Jirafa_R.png', img_red, cmap = "Reds_r")
mpimg.imsave('Jirafa_G.png', img_green, cmap = "Greens_r")
mpimg.imsave('Jirafa_B.png', img_blue, cmap = "Blues_r")
```