22684251-trinhduonghoan

August 19, 2024

Họ Tên : Trịnh Dương Hoan
MSSV : 22684251

[1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

0.1 Numpy

[3]:
```
# Khai báo thư viện
import numpy as np
from time import time
```

0.1.1 Create an array

Array function

[13]:
```
# Tạo một danh sách các giá trị float
list_of_values = [20., 2., 5.]

# Chuyển đổi danh sách thành một mảng numpy 1D với kiểu dữ liệu float
x = np.array(list_of_values)

# Tạo một danh sách chứa các giá trị được bọc trong danh sách con
more_values = [[[20], [2], [5]]]

# Chuyển đổi danh sách thành một mảng numpy 3D
y = np.array(more_values)

# In ra mảng x
print(x)  # Mảng 1D: [20.  2.  5.]

# In ra mảng y
print(y)
```

```
[20.  2.  5.]
[[[20]
```

```
  [ 2]
  [ 5]]]
```

Using zero, one

[12]:
```
zeros = np.zeros((20, 2), dtype=np.int32)
print(zeros)  # In ra mảng chứa toàn bộ giá trị 0

# Tạo một mảng numpy chứa toàn bộ giá trị 1 với kích thước 5x2x1 và kiểu dữ␣
 ↪liệu np.float32
ones = np.ones((5, 2, 1), dtype=np.float32)
print(ones)  # In ra mảng chứa toàn bộ giá trị 1
```

```
[[0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]
 [0 0]]
[[[1.]
  [1.]]

 [[1.]
  [1.]]

 [[1.]
  [1.]]

 [[1.]
  [1.]]

 [[1.]
  [1.]]]
```

Using zeros_like, ones_like

[10]:
```
x = np.array([5, 5])  # Tạo một mảng numpy với các giá trị [5, 5]
zeros = np.zeros_like(x)  # Tạo một mảng zeros có cùng kích thước và kiểu dữ␣
 ↪liệu với mảng x
ones = np.ones_like(x, dtype = np.float32)  # Tạo một mảng ones có cùng kích␣
 ↪thước và kiểu dữ liệu với mảng x
print(zeros)  # In ra mảng zeros
print(ones)  # In ra mảng ones
```

```
[0 0]
[1. 1.]
```

Using the function arange(start,stop, step)

[12]:
```
x = np.arange(5)  # Tạo một mảng numpy chứa các giá trị từ 0 đến 4 (0, 1, 2, 3,␣
 ↪4)
y = np.arange(2, 5)  # Tạo một mảng numpy chứa các giá trị từ 2 đến 4 (2, 3, 4)
z = np.arange(2, 5, 2)  # Tạo một mảng numpy chứa các giá trị từ 2 đến dưới 5,␣
 ↪với bước nhảy là 2 (2, 4)

print(x)  # In ra mảng x
print(y)  # In ra mảng y
print(z)  # In ra mảng z
```

```
[0 1 2 3 4]
[2 3 4]
[2 4]
```

Using eye

[14]:
```
x = np.eye(5)  # Tạo một ma trận đơn vị 5x5 (ma trận vuông với đường chéo chính␣
 ↪là 1 và các phần tử khác là 0)
print(x)  # In ra ma trận đơn vị
```

```
[[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
```

Using rand function

[15]:
```
x = np.random.rand(3,5)  # Tạo một mảng numpy 3x5 chứa các giá trị ngẫu nhiên␣
 ↪từ 0 đến 1
print(x)  # In ra mảng x
```

```
[[0.38219481 0.61616395 0.75552471 0.12444059 0.20382667]
 [0.032308   0.42418014 0.41994891 0.49965754 0.08499076]
 [0.51963127 0.76379537 0.26882404 0.18881617 0.29498512]]
```

Using full

[16]:
```
x = np.full((2,2), 7)  # Tạo một mảng numpy 2x2 chứa các giá trị 7
print(x)  # In ra mảng x
```

```
[[7 7]
 [7 7]]
```

[17]:
```
x = np.ones((2,2)) * 7  # Tạo một mảng numpy 2x2 chứa các giá trị 7
print(x)  # In ra mảng x
```

```
[[7. 7.]
 [7. 7.]]
```

0.1.2 Attributes

dtype

[19]:
```
x = np.array([2., 5., 3.])  # Tạo một mảng numpy chứa các giá trị [2., 5., 3.]
y = np.array([2., 5., 3], dtype=np.int32)  # Tạo một mảng numpy chứa các giá
  ↪trị [2, 5, 3] với kiểu dữ liệu là int32
print(x.dtype)  # In ra kiểu dữ liệu của mảng x
print(y.dtype)  # In ra kiểu dữ liệu của mảng y
```

```
float64
int32
```

shape

[21]:
```
x = np.array([2.,5.,3.])  # Tạo một mảng numpy chứa các giá trị [2., 5., 3.]
y = np.ones((2,4,1,2,3))  # Tạo một mảng numpy chứa các giá trị 1 có kích thước
  ↪(2,4,1,2,3)
print(x.shape)  # In ra kích thước của mảng x
print(y.shape)  # In ra kích thước của mảng y
```

```
(3,)
(2, 4, 1, 2, 3)
```

0.1.3 Changing the shape of arrays

using expand_dims

[26]:
```
x = np.full((2,2,3), 7)  # Tạo một mảng numpy 2x2x3 chứa các giá trị 7
print(x.shape)  # In ra kích thước của mảng x
x = np.expand_dims(x, axis=0)  # Tăng kích thước của mảng x lên 1 chiều
assert x.shape == (1,2,2,3)  # Kiểm tra xem kích thước của mảng x có đúng không
print(x)  # In ra mảng x
```

```
(2, 2, 3)
```

using the reshape function

[29]:
```
x = np.arange((10))  # Tạo một mảng numpy chứa các giá trị từ 0 đến 9
y = np.reshape(x, (2,5))  # Đổi kích thước của mảng x thành (2,5)
assert y.shape == (2,5)  # Kiểm tra xem kích thước của mảng y có đúng không
```

[47]:
```
x = np.arange((20))  # Tạo một mảng numpy chứa các giá trị từ 0 đến 19
y = np.reshape(x, (2,-1,2))  # Đổi kích thước của mảng x thành (2,5,2)
assert y.shape == (2,5,2)  # Kiểm tra xem kích thước của mảng y có đúng không
print(x) # In ra mảng x
print(y) # In ra mảng y
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
[[[ 0  1]
  [ 2  3]
  [ 4  5]
  [ 6  7]
  [ 8  9]]

 [[10 11]
  [12 13]
  [14 15]
  [16 17]
  [18 19]]]
```

ValueError

[ ]:
```
x = np.arange((20))  # Tạo một mảng numpy chứa các giá trị từ 0 đến 19
y = np.reshape(x, (2,-1,-1))  # Test lỗi
```

Using squeeze

[42]:
```
x = np.full((20,1,1), 5)  # Tạo một mảng numpy 20x1x1 chứa các giá trị 5
y = np.squeeze(x)  # Giảm kích thước của mảng x lên 1 chiều
assert y.shape == (20,)  # Kiểm tra xem kích thước của mảng y có đúng không
print(x) # In ra mảng x
print(y) # In ra mảng y
```

```
[[[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]
```

```
 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]]
[5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5]
```

[43]:
```python
x = np.full((20,1,1), 5)  # Tạo một mảng numpy 20x1x1 chứa các giá trị 5
y = np.squeeze(x, axis=1)  # Giảm kích thước của mảng x lên 1 chiều
assert y.shape == (20,1)  # Kiểm tra xem kích thước của mảng y có đúng không
print(x) # In ra mảng x
print(y) # In ra mảng y
```

```
[[[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]
```

```
 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]

 [[5]]]
[[5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]
 [5]]
```

0.1.4 Elements of arrays

index notation

[44]:
```
x = np.arange(20)  # Tạo một mảng numpy chứa các giá trị từ 0 đến 19
element = x[10]  # Lấy phần tử thứ 10 trong mảng x
print(element)  # In ra phần tử thứ 10 trong mảng x
```

```
10
```

item function

[45]:
```
x = np.arange(20)  # Tạo một mảng numpy chứa các giá trị từ 0 đến 19
element = x.item(10)  # Lấy phần tử thứ 10 trong mảng x
print(element)  # In ra phần tử thứ 10 trong mảng x
```

```
10
```

Slice notation

[48]:
```
x = np.arange(20)  # Tạo một mảng numpy chứa các giá trị từ 0 đến 19
element = x[10:15]  # Lấy các phần tử từ 10 đến 14 trong mảng x
print(element)  # In ra các phần tử từ 10 đến 14 trong mảng x
```

```
[10 11 12 13 14]
```

[50]:
```
more_element = x[10:-7]  # Lấy các phần tử từ 10 đến len(x) -8 trong mảng x
assert np.array_equal(more_element, x[10:13])
print(more_element)  # In ra các phần tử từ 10 đến len(x) -8 trong mảng x
```

```
[10 11 12]
```

[51]:
```
array = x[::3]  # Lấy các phần tử từ 0 đến len(x) -1 với bước nhảy là 3
print(array)  # In ra các phần tử từ 0 đến len(x) -1 với bước nhảy là 3
```

```
[ 0  3  6  9 12 15 18]
```

concatenate function

[53]:
```
x = np.full((5,2), 3)  # Tạo một mảng numpy 5x2 chứa các giá trị 3
y = np.full((5,1), 4)  # Tạo một mảng numpy 5x1 chứa các giá trị 4
z = np.concatenate([x,y], axis=1)  # Nối hai mảng x và y theo trục 1
print(z) # In ra mảng z
assert z.shape == (5,3)  # Kiểm tra xem kích thước của mảng z có đúng không
```

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

0.1.5 NumPy math

sum and subtraction

[57]:
```
x = np.full((4,2,3), 8)  # Tạo một mảng numpy 4x2x3 chứa các giá trị 8
y = np.ones_like(x)  # Tạo một mảng numpy có cùng kích thước và kiểu dữ liệu␣
 ↪với mảng x

array_sum = x + y  # Cộng hai mảng x và y
assert np.array_equal(array_sum, np.ones_like(x)* 9)  # Kiểm tra xem mảng␣
 ↪array_sum có đúng không
print(array_sum)  # In ra mảng array_sum

arrray_sub = x - y  # Trừ hai mảng x và y
assert np.array_equal(arrray_sub, np.ones_like(x) * 7)  # Kiểm tra xem mảng␣
 ↪arrray_sub có đúng không
print(arrray_sub)  # In ra mảng arrray_sub
```

```
[[[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]]
[[[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]]
```

broadcasting

[14]:
```
x = np.full((4,2,3), 8)  # Tạo một mảng numpy 4x2x3 chứa các giá trị 8
y = 1  # Tạo một số nguyên y

array_sum = x + y  # Cộng hai mảng x và y
assert np.array_equal(array_sum, np.ones_like(x)* 9)  # Kiểm tra xem mảng␣
 ↪array_sum có đúng không
```

```
print(array_sum)  # In ra mảng array_sum

arrray_sub = x - y  # Trừ hai mảng x và y
assert np.array_equal(arrray_sub, np.ones_like(x) * 7)  # Kiểm tra xem mảng␣
 ↪arrray_sub có đúng không
print(arrray_sub)  # In ra mảng arrray_sub
```

```
[[[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]

 [[9 9 9]
  [9 9 9]]]
[[[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]

 [[7 7 7]
  [7 7 7]]]
```

ValueError

[ ]:
```
x = np.full((4,2,3), 8)  # Tạo một mảng numpy 4x2x3 chứa các giá trị 8
y = np.full((4,3), 3)  # Tạo một mảng numpy 4x3 chứa các giá trị 3

array_sum = x + y  # ValueError

y2 = np.full((4),3)  # Tạo một mảng numpy 4 chứa các giá trị 3

array_sum = x + y2  # ValueError

y3 = np.ones(4)  # Tạo một mảng numpy 4 chứa các giá trị 1
y3 = np.expand_dims(1, 0)  # Tạo một mảng numpy 4x1 chứa các giá trị 1

array_sum = x + y3  # it works !
```

multiplication

[59]:
```
x1 = np.full((4,2,3), 8)  # Tạo một matrix numpy 4x2x3 chứa các giá trị 8
x2 = np.full((3,3),7)  # Tạo một matrix numpy 3x3 chứa các giá trị 7
y = np.eye(3)  # Tạo một ma trận đơn vị 3x3
mul = np.matmul(x1, y)  # Nhân hai matrix x1 và y
assert np.array_equal(mul, x1)  # Kiểm tra xem ma trận mul có đúng không
print(mul)  # In ra matrix mul
```

```
[[[8. 8. 8.]
  [8. 8. 8.]]

 [[8. 8. 8.]
  [8. 8. 8.]]

 [[8. 8. 8.]
  [8. 8. 8.]]

 [[8. 8. 8.]
  [8. 8. 8.]]]
```

[60]:
```
mul = np.matmul(x2,y)  # Nhân hai matrix x2 và y
assert np.array_equal(mul, x2)  # Kiểm tra xem ma trận mul có đúng không
print(mul)  # In ra matrix mul
```

```
[[7. 7. 7.]
 [7. 7. 7.]
 [7. 7. 7.]]
```

0.1.6 Conditions

where

[62]:
```
x = np.arange(5)  # Tạo một mảng numpy chứa các giá trị từ 0 đến 4
y = np.where(x < 2,0,255)  # Thay thế các phần tử < 2 trong mảng x bằng 0 và␣
↪các phần tử >= 2 trong mảng x bằng 255
print(y)  # In ra mảng y
```

```
[  0   0 255 255 255]
```

[67]:
```
x = np.random.rand(4,640, 480, 3)  # Tạo một mảng numpy chứa các giá trị ngẫu␣
↪nhiên từ 0 đến 1
batch, height, width, channels = x.shape  # Lấy kích thước của mảng x
start = time()
y = np.ones_like(x)  # Tạo một mảng numpy có cùng kích thước và kiểu dữ liệu␣
↪với mảng x
for b in range(batch):
    for h in range(height):
        for w in range(width):
          for c in range(channels) :
```

```
            y[b,h,w,c] =  0 if x[b,h,w,c] < 0.05 else 255  # Thay thế các phần
 ↪tử < 0.05 trong mảng x bằng 0 và các phần tử >= 0.05 trong mảng x bằng 255
duration = time() - start
print(duration)
```

```
3.00933837890625
```

[68]:
```
x = np.random.rand(4,640, 480, 3)  # Tạo một mảng numpy chứa các giá trị ngẫu
 ↪nhiên từ 0 đến 1
start = time()
y = np.where(x < 0.05, 0, 255)
duration = time() - start
print(duration)
```

```
0.06139802932739258
```

=> cách 2 tối ưu hơn cách 1

0.1.7 NumPy Input/Output

loadtxt

[8]:
```
data = np.loadtxt("/content/drive/MyDrive/XuLiAnh/Lab_Numpy/test.txt", dtype=np.
 ↪float32, delimiter=',')
print(data)
```

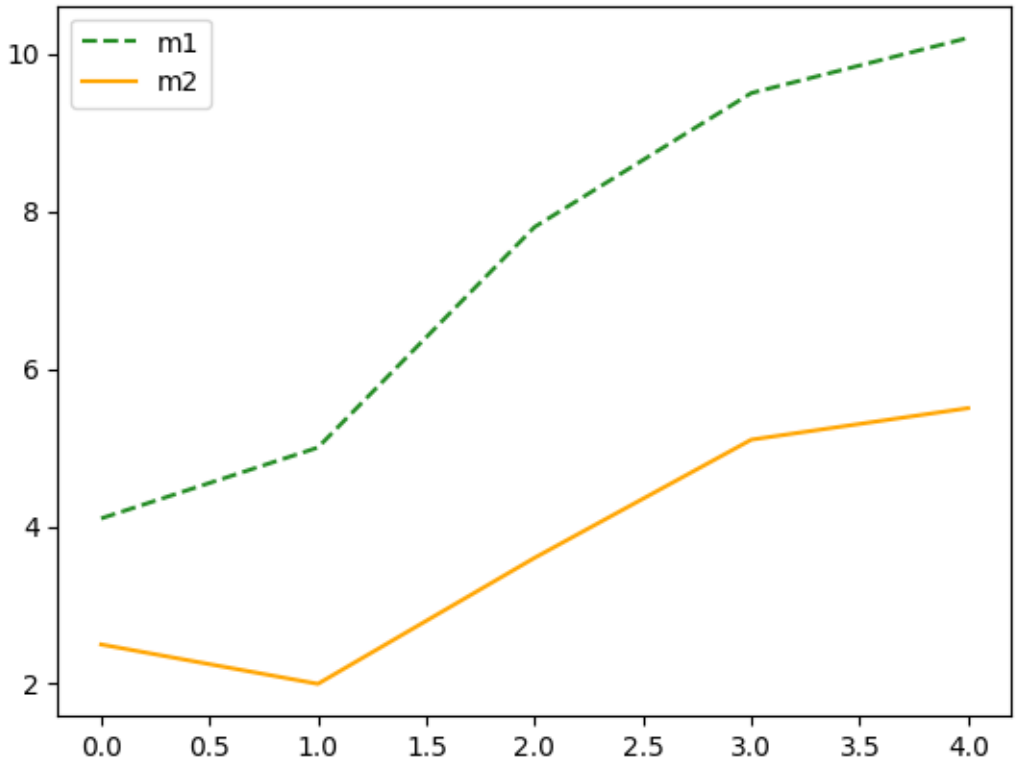
```
[[1. 2. 3.]
 [4. 5. 6.]]
```

0.2 Matplotlib

[10]:
```
import matplotlib.pyplot as plt # Khai báo thư viện
```

[11]:
```
labels = ['7', '8', '9', '10','11'] # Tạo danh sách nhãn
x = np.arange(5) # Tạo mảng x chứa các giá trị từ 0 đến 4
m1 = np.array([4.1, 5.0, 7.8, 9.5, 10.2])  # Tạo mảng m1 chứa các giá trị
m2 = np.array([2.5, 2.0, 3.6, 5.1, 5.5])  # Tạo mảng m2 chứa các giá trị

plt.plot(x,m1, color = "forestgreen", label = "m1", linestyle = 'dashed') # Vẽ
 ↪đồ thị cho m1
plt.plot(x,m2, color = "orange", label = "m2") # Vẽ đồ thị cho m2
plt.legend()  # Hiển thị chú thích
plt.savefig('chart.png')  # Lưu ảnh
plt.show()  # Hiển thị biểu đồ
```


m1
m2
10
8
6
4
2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

[ ]: