專題

NUMPY模組之基本概念

什麼是Numpy

Numpy 是 Python 的一個重要模組，主要用於資料處理上。Python 處理龐大資料時，其串列 list 的效能表現並不理想，而 Numpy 具備平行處理的能力，可以將數個操作動作一次套用在大型陣列上。此外 Python 其餘重量級的資料科學相關套件（例如：Pandas、SciPy、Scikit-learn 等）都幾乎是奠基在 Numpy 的基礎上。因此學會 Numpy 對於往後學習其他資料科學相關套件打好堅實的基礎。

建立一維陣列

Numpy 所建立的陣列資料型態為ndarray (n-dimension array) , 其中 n 代表維度

我們使用 array( ) 建立一維陣列

```
 8  import numpy as np
 9
10  x = np.array([6,7,8])
11
12  print(x)
13  print(type(x))
```

```
[6 7 8]
<class 'numpy.ndarray'>
```

跟串列不同的地方在於各元素之間沒有用逗號隔開

建立一維陣列

我們使用索引輸出特定元素：

```
 8  import numpy as np
 9
10  x = np.array([6,7,8])
11
12  print(x)
13  print(x[0])
14  print(x[1])
15  print(x[2])
```

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

也可修改元素：

```
 8  import numpy as np
 9
10  x = np.array([6,7,8])
11  print(x)
12
13  x[1] = 20
14  print(x)
```

```
[6 7 8]
[ 6 20  8]
```

建立一維陣列

認識 ndarray 的一些屬性：

```
 8  import numpy as np
 9
10  x = np.array([6,7,8])
11
12  print(x.ndim)      # 輸出陣列維度
13  print(x.shape)     # 輸出陣列外型
14  print(x.size)      # 輸出陣列元素個數
```

```
1
(3,)
3
```

一維陣列的運算

```
 8  import numpy as np
 9
10  x = np.array([6, 7, 8])
11  y = x+5   # 陣列與整數相加
12  print(y)
13
14  y = np.array([-2, 1, 2.5]) #兩個一維陣列的運算
15  z1 = x+y
16  z2 = x*y
17  z3 = x/y
18
19  print(z1)
20  print(z2)
21  print(z3)
```

```
[11 12 13]
[ 4.   8.  10.5]
[-12.   7.  20.]
[-3.   7.   3.2]
```

一維陣列的運算

```
 8  import numpy as np
 9
10  x = np.array([6, 7, 8])
11  y = np.array([2, 9, 8])
12
13  z1 = x>y     # 關係運算子
14  z2 = x<=y
15  print(z1)
16  print(z2)
```

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

陣列的結合與加入元素

使用 concatenate( ) 將陣列結合

使用 insert(陣列 , 索引 , 元素) 加入元素

使用 delete( ) 刪除元素

```
[6 7 8 2 9 8]
[ 6  7  8  9 10]
[6 7 3 8]
[6 8]
[7]
```

```python
import numpy as np

x = np.array([6, 7, 8])
y = np.array([2, 9, 8])
z = np.concatenate((x,y))
print(z)

z = np.concatenate((x, [9, 10]))
print(z)

z = np.insert(x, 2, 3)
print(z)

z1 = np.delete(x, 1)
z2 = np.delete(x, [0, 2])
print(z1)
print(z2)
```

建立二維陣列

兩種建立二維陣列的方法

```
 8  import numpy as np
 9
10  x = [3, 6, 9]
11  y = [4, 7, 10]
12  z = np.array([x, y], ndmin = 2)
13
14  print(z)
15
16  a = np.array([[3, 6, 9], [4, 7, 10]])
17  print(a)
```

```
[[ 3  6  9]
 [ 4  7 10]]
[[ 3  6  9]
 [ 4  7 10]]
```

建立二維陣列

輸出特定元素：

```
8  import numpy as np
9
10 x = [3, 6, 9]
11 y = [4, 7, 10]
12 z = np.array([x, y], ndmin = 2)
13
14 print(z[1][2])
15 print(z[0,1])
```

```
10
6
```

建立元素都是0的陣列 及 都是 1 的陣列：

```
8  import numpy as np
9
10 x = np.zeros(3, dtype = int)
11 print(x)
12
13 y = np.zeros((2,5), dtype = int)
14 print(y)
15
16 x = np.ones(3, dtype = int)
17 print(x)
18
19 y = np.ones((2,5), dtype = int)
20 print(y)
```

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

輸出子陣列

概念與 python 串列擷取子串列的手法相同

```
 8  import numpy as np
 9
10  x = np.array([4, 5, 6, 7, 8, 9])
11
12  print(x[2:])
13  print(x[1:4])
14  print(x[0:4:2])
```

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

```
 8  import numpy as np
 9
10  x = np.array([4, 5, 6, 7, 8, 9])
11
12  print(x[-2])
13  print(x[::-1])
14  print(x[3::-1])
15  print(x[3:-5:-1])
```

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

輸出子陣列

使用布林值輸出子陣列

```
 8  import numpy as np
 9
10  x = np.array([4, 5, 6, 7, 8, 9])
11  y = (x%2 == 0)
12  print(y)
13  print(x[y])
14  x[y]=0
15  print(x)
```

```
[ True False  True False  True False]
[4 6 8]
[0 5 0 7 0 9]
```

建立三維陣列

建立 2x3x6 的三維陣列

```
 8  import numpy as np
 9
10  x = [4, 5, 6, 7, 8, 9]
11  y = [2, 4, 6, 8, 10,12]
12  z = [1, 5, 9, 13,17,21]
13  z1=np.array([x, y, z])
14  z2=np.array([z1, z1])
15
16  print(z2)
17  print(z2[0,0,1])
18  print(z2[1,1,1])
19  print(z2[0,2,3])
20  print(z2[1,1,5])
```

```
[[[ 4  5  6  7  8  9]
  [ 2  4  6  8 10 12]
  [ 1  5  9 13 17 21]]

 [[ 4  5  6  7  8  9]
  [ 2  4  6  8 10 12]
  [ 1  5  9 13 17 21]]]
5
4
13
```

陣列分割

```
 8  import numpy as np
 9
10  x = np.arange(16).reshape(4,4) #reshape設定陣列規模
11  print(x)
12
13  y1, y2 = np.hsplit(x,2) #hsplit將陣列x水平分成2個陣列
14  print(y1)
15  print(y2)
16
17  z1, z2 = np.vsplit(x,2) #hsplit將陣列x鉛分成2個陣列
18
19  print(z1)
20  print(z2)
```

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

陣列合併

```
8  import numpy as np
9
10 x1 = np.arange(9).reshape(3,3) #reshape設定陣列規模
11 x2 = np.arange(9,18).reshape(3,3)
12 x = np.vstack((x1,x2)) # 鉛直方向合併
13 print(x)
14
15 x = np.hstack((x1,x2)) # 水平方向合併
16 print(x)
```

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

作業一

(1) 建立 1~50之 10x5 陣列

```
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]
 [26 27 28 29 30]
 [31 32 33 34 35]
 [36 37 38 39 40]
 [41 42 43 44 45]
 [46 47 48 49 50]]
```

(2) 使用ones( ) 建立 2x4 陣列, 陣列元素皆為 8

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

(3) 建立 0~19 的一維陣列 , 輸出兩個一維陣列各位自蒐集偶數與奇數

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

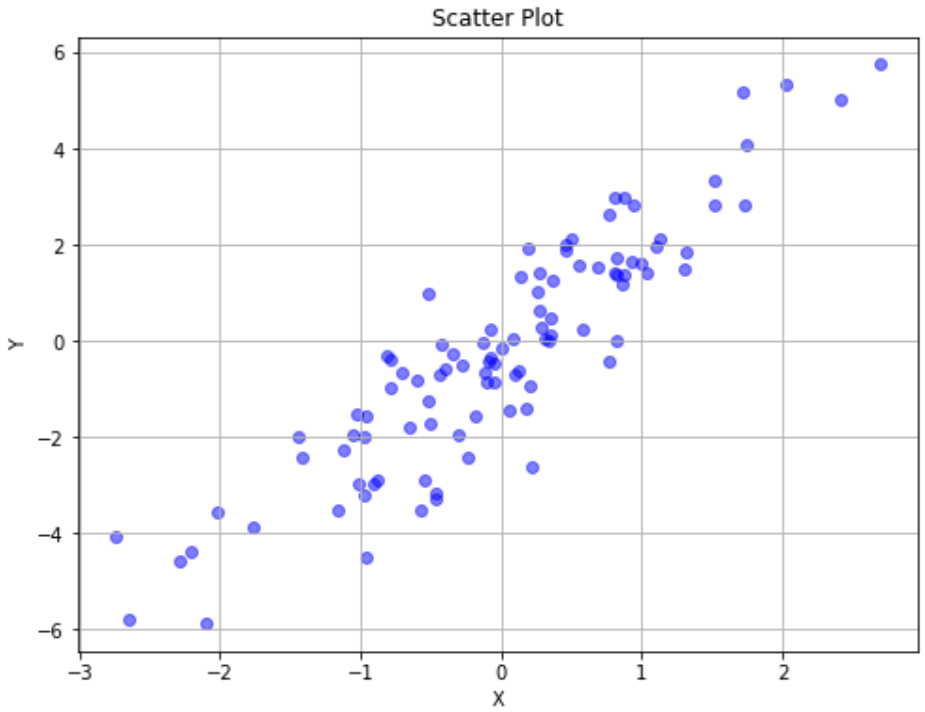
作業一

(4) 建立下列三維陣列

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

 [[3 4 5 6]
  [6 7 8 9]]]
```

實例：繪製散佈圖

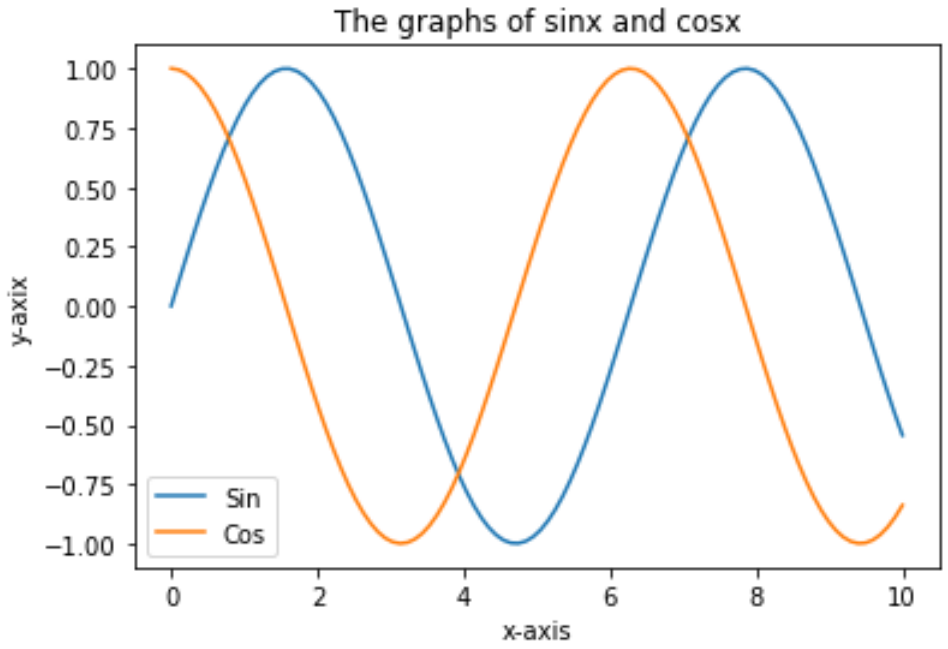
```python
import numpy as np
import matplotlib.pyplot as plt

# 生成隨機數據

x = np.random.randn(100)  # 生成 100 個符合標準常態分佈的隨機數字的一維陣列

y = 2 * x + np.random.randn(100)  # 生成 y 與 x 線性相關的隨機數，但加入了一些雜訊

# 繪製散布圖
plt.figure(figsize=(8, 6))
plt.scatter(x, y, color='blue', alpha=0.5)  # alpha 設置透明度，使重疊的點更加明顯
plt.title('Scatter Plot')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True)
plt.show()
```



實例: Numpy內建三角函數

繪製正弦函數和餘弦函數

```
 8  import numpy as np
 9  import matplotlib.pyplot as plt
10
11  # 0~10之間生成100個數據
12  x = np.linspace(0, 10, 100)
13  y1 = np.sin(x)
14  y2 = np.cos(x)
15
16  # 繪製折線圖
17  plt.plot(x, y1, label='Sin')
18  plt.plot(x, y2, label='Cos')
19  plt.xlabel('x-axis')
20  plt.ylabel('y-axix')
21  plt.title('The graphs of sinx and cosx')
22  plt.legend()
23  plt.show()
```



陣列資料排序

一維陣列排序 : sort( )

```
 8  import numpy as np
 9
10  x = np.array([5,2,8])
11  y = np.sort(x)
12
13  print(x)
14  print(y)
```

```
[5 2 8]
[2 5 8]
```

陣列資料排序

二維陣列排序

```
 8  import numpy as np
 9
10  x = np.array([[5,2,8,1,4],
11                [9,5,6,2,0]])
12  y = np.sort(x, axis=0)
13  z = np.sort(x, axis=1)
14
15  print(x)
16  print(y)
17  print(z)
```

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

向量內積

人工智慧中的卷積運算，即為內積運算的延伸，目的在取得圖像特徵

```
 8  import numpy as np
 9
10  x = np.array([5,2,8,1,4])
11  y = np.array([9,5,6,2,0])
12
13  z1 = np.dot(x,y)
14  z2 = np.inner(x,y)
15
16  print(z1)
17  print(z2)
```

```
105
105
```

向量外積

機器學習常見的運算

```
 8  import numpy as np
 9
10  x = np.array([3,2,4])
11  y = np.array([1,3,0])
12
13  z = np.cross(x,y)
14
15  print(z)
```

```
[-12   4   7]
```

練習

計算下列兩個向量的內積與外積

15. $\mathbf{v}_1 = \begin{bmatrix} 7 \\ 1 \\ -6 \end{bmatrix}, \mathbf{v}_2 = \begin{bmatrix} -5 \\ 3 \\ 0 \end{bmatrix}$

16. $\mathbf{v}_1 = \begin{bmatrix} 3 \\ 0 \\ 2 \end{bmatrix}, \mathbf{v}_2 = \begin{bmatrix} -2 \\ 0 \\ 3 \end{bmatrix}$

矩陣的行列式

一區域作線性變換時 (水平伸縮 / 垂直伸縮 / 放大縮小) 面積變動的比例

```
 8  import numpy as np
 9
10  x = np.array([[3,2,4],
11                [1,3,0],
12                [0,2,1]])
13
14  d = np.linalg.det(x)
15  print(d)
```

```
15.0
```

解一元二次方程式

輸入一元二次方程式 $ax^2 + bx + c = 0$ 的係數 a , b , c

```
 8  import numpy as np
 9
10  x = np.array([1,3,2])
11
12  y = np.roots(x)
13
14  print(y)
```

```
[-2. -1.]
```

解線性聯立方程組

$$\begin{cases} x + 2y &= 7 \\ 2x - y &= 4 \end{cases}$$

```
 8  import numpy as np
 9
10  coe = np.array([[1,2],[2,-1]])
11  con = np.array([7,4])
12
13  ans = np.linalg.solve(coe,con)
14
15  print(ans)
```

```
[3. 2.]
```

練習

求解線性聯立方程組

$$
\begin{aligned}
x_1 + 3x_2 - 5x_3 &= 4 \\
x_1 + 4x_2 - 8x_3 &= 7 \\
-3x_1 - 7x_2 + 9x_3 &= -6
\end{aligned}
$$