NumPy 陣列的邏輯運算

簡報閱讀 範例與作業 問題討論 學習心得(完成)

重要知識點

- 正確的使用 NumPy 中的比較與邏輯
- 掌握 NumPy 陣列的遮罩特性與使用
- 知道 NumPy 陣列與 Python 列表的用
  異

比較運算是用來判斷數值之間的比較關係，例如：相等、不相等、大於小於等等的關係；邏輯運算適用於布林值的組合，例如 AND 或 OR 之類的運算。

比較運算符	描述
= =	判斷2個數值是否相等
!=	判斷2個數值是否不相等
>	判斷左數是否大於右數
<	判斷左數是否大於右數
>=	判斷左數是否大於右數
<=	判斷左數是否大於右數

邏輯運算符	描述
and	與，只有在 x 和 y 都是 True 的情況下，才返回 True，否則返回 False
or	或，只要 x 或者 y 有一個為True，就返回 True，否則返回 False
not	非，返回相反的結果

陣列中的比較運算

在 NumPy 陣列中也有比較運算，一樣會符合「對齊」與「廣播」的特性：

```
1  import numpy as np
2
3  a = np.array( [20,30,40,50] )
4  b = np.arange( 4 )
5
6  print(a > b) # [ True  True  True  True]
7  print(a < b) # [False False False False]
8  print(a == b) # [False False False False]
9  print(a != b) # [ True  True  True  True]
```

需要特別注意的是，在 NumPy 陣列中沒有邏輯運算，可以使用位元運算或是邏輯運算的函式方法取代：

```
import numpy as np

a = np.array( [True, True, False, False] )
b = np.array( [True, False, True, False]  )

print(a and b)
Traceback (most recent call last):
  ...
ValueError: The truth value of an array with m
```

```
import numpy as np

a = np.array( [True, True, False, False] )
b = np.array( [True, False, True, False]  )

print(a & b) # [ True False False False]
print(a | b) # [ True  True  True False]
```

邏輯運算與比較運算共同性

邏輯運算與比較運算都會產生一組有布林所組成的陣列：

```
a = np.array( [20,30,40,50] )
b = np.arange( 4 )

print(a > b) # [ True  True  True  True]
print(a < b) # [False False False False]
print(a == b) # [False False False False]
print(a != b) # [ True  True  True  True]
```

```
import numpy as np

a = np.array( [True, True, False, False] )
b = np.array( [True, False, True, False]  )

print(a & b) # [ True False False False]
print(a | b) # [ True  True  True False]
```

利用布林值作為篩選的條件：遮罩

可以用一組 True/False 做為每一個位置的篩選條件，這種方法稱為遮罩（Mask）：

```
import numpy as np

a = np.array( [10, 20, 30, 40] )

print(a[ [True, True, True, True] ])
# [10 20 30 40]
print(a[ [True, False, True, False] ])
# [10 30]
print(a[ [False, False, False, False] ])
# []
```

從比較/邏輯運算到遮罩特性

元素：

```
import numpy as np

a = np.array( [10, 20, 30, 40] )

print(a > 20)
# [False False  True  True]
print(a[ [False, False, True, True] ])
# [30 40]
print(a[ a > 20 ])
# [30 40]
```

遮罩特性背後的強大之處

「遮罩」是陣列當中最重要的特性之一，用於篩選出符合條件的元素。以「找出 >3 的元素」這個例子來看，兩種截然不同的做法：

```
a = np.arange( 4 )
print(a[a > 1])
# array([2 3])
```

```
a = np.arange( 4 )
b = []
for i in a:
  if i > 1:
    b.append(i)
print(b)
# [2, 3]
```

綜合這兩天的介紹，我們可以知道在陣列算符合「對齊」、「廣播」以及「遮罩」三個特性。這三種特性都符合矩陣以整組為單位的運算，而非一個

補充：any() 和 all()

除了用邏輯跟比較運算產生由 True 或 False 組成布林陣列之外，還有一些常用的方法可以用：

```
import numpy as np

print(np.any([True, True, True]))
# True
print(np.any([True, False, False]))
# True
print(np.any([False, False, False]))
# False
```

```
import numpy as np

print(np.all([True, True, True]))
# True
print(np.all([True, False, False]))
# False
print(np.all([False, False, False]))
# False
```

陣列的比較與陣列元素的比較

陣列的比較指的是兩個變數之間的比較，會是一個布林的結果；而陣列元素的比較是指陣列中的元素兩兩對齊比較，會回傳一組布林的結果。

NumPy 陣列邏輯函式 - 陣列內容

isinf()、isposinf()、isneginf()	判別元素是否為無限數、正無限數、負無限數，若是的話回傳 True，否則回傳 False。
isnat()	isnat() 的 nat (NaT) 是 not a time 的意思，用來判別陣列元素是否為日期時間。若非日期時間 (包括 datetime 或 timedelta) 的話回傳 True，若是的話則回傳 False。

- numpy.nan 與 numpy.NAN 都是 NumPy 常數，代表 Not a Number，不過在官方文件中建議統一使用小寫的 nan。
- 判斷無限數的函式有 isinf()、isposinf()、isneginf()，分別用來判斷判斷陣列元素是否為正無限數或負無限數、是否為正無限數、是否為負無限數。
- NumPy 內建常數 (Constants) 來示範，無限數相關的常數如右表：

常數	說明	別名
np.inf	正無限數	np.Inf, np.Infinity, np.PINF, np.infty
np.Inf	正無限數	
np.Infinity	正無限數	
np.PINF	正無限數	
np.infty	正無限數	
np.NINF	負無限數	

- isnat() 的 nat (NaT) 是 not a time 的意思，用來判別陣列元素是否為日期時間。若非日期時間 (包括 datetime 或 timedelta) 的話回傳 True，若是的話則回傳 False。

NumPy 陣列邏輯函式 - 陣列型別測

isreal()	判斷輸入的陣列元素是否為實數 。
iscomplex()	判斷輸入的陣列元素是否為複數。
isrealobj()	判斷整個陣列物件是否為實數物件。
iscomplexobj()	判斷整個陣列物件是否為複數物件。

NumPy 陣列邏輯函式 - 陣列比較

- 使用 np.array_equal()、np.array_equiv() 比較 2 個陣列是否相同。
- 兩個函式不同之處在於 array_equal() 需要形狀完全一樣且元素值皆相同才為 True。
- 說明如下：

函式	說明
np.array_equal()	若兩個陣列形狀與元素值均相同，回傳 True
np.array_equiv()	兩個陣列形狀元素值均相同，回傳 True； 如果兩個陣列維度不同的話，須符合廣播規則，且元素值均相同，則回傳 True

- 比較等於/不等於、大於/大於或等於、小於/小於或等於，可以使用右表函式：
- 表中的函式均可以比較兩個形狀相同的陣列，或是比較符合廣播規則的兩個陣列，若元素值相同的話就回傳 True。比較時均是 element-wise 的比較。

函式	說明
np.equal()	等於
np.not_equal()	不等於
np.greater()	大於
np.greater_equal()	大於或等於
np.less()	小於
np.less_equal()	小於或等於

- 邏輯比較函式都是 element-wise，比較 2 個陣列元素。如果 2 個陣列的形狀不同的話，必須符合廣播 (broadcasting) 規則。
- 邏輯操作與對應的函式如下表：

Logical operation	函式
AND	np.logical_and()
OR	np.logical_or()
NOT	np.logical_not()
XOR	np.logical_xor()

NumPy 陣列邏輯函式 - Truth值測試

使用 np.all() 進行 Truth 值測試

```
np.all([-1, 4, 0])
```

```
False
```

np.all([-1, 4, 0])

使用 np.any() 進行 Truth 值測試

```
np.any([[True, False], [False, False]], axis=0)
```

```
array([ True, False])
```

np.any([[True, False], [False, False]], axis=0)

- 可以依軸 (axis) 進行比較，兩個函式不同的地方在於 np.all() 是 AND 邏輯的比較，而 np.any() 是 OR 邏輯的比較。

True：True、NaN、正無限值、負無限值。

知識點回顧

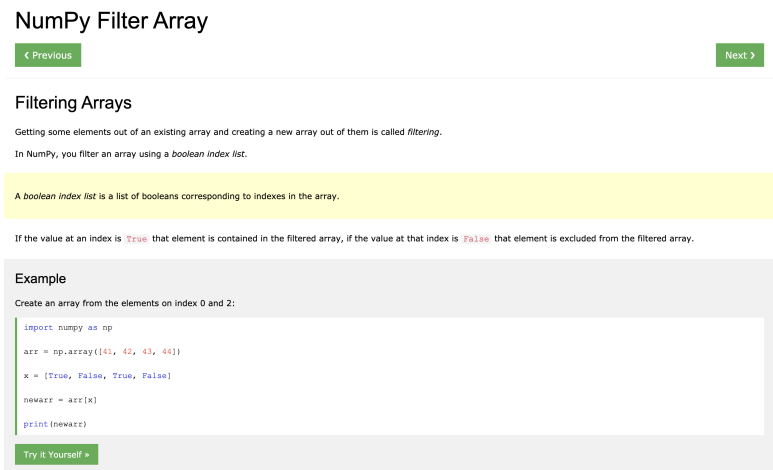
- 正確的使用 NumPy 中的比較與邏輯運算
- 掌握 NumPy 陣列的遮罩特性與使用
- 知道 NumPy 陣列與 Python 列表的用法差異

參考資料

NumPy Filter Array

網站：[w3schools]

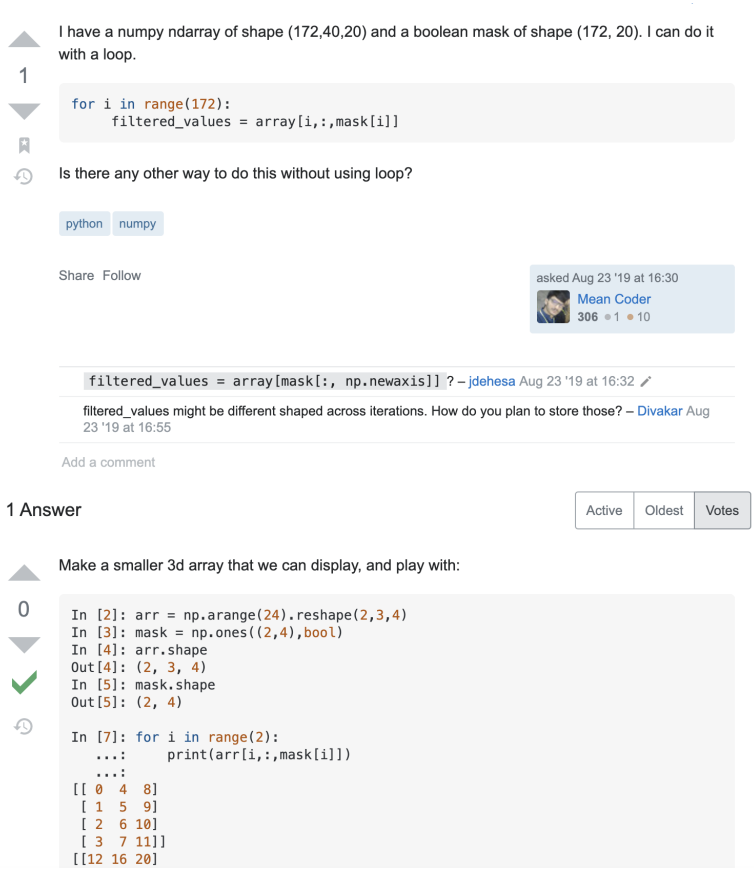
更基本的遮罩入門教學，如果對於該用法不太熟悉的話建議可以看這篇。



How to filter values of numpy ndarray based on a boolean mask?

網站：[stackoverflow]

話可以參考。



I have a numpy ndarray of shape (172,40,20) and a boolean mask of shape (172, 20). I can do it with a loop.

```
for i in range(172):
     filtered_values = array[i,:,mask[i]]
```

Is there any other way to do this without using loop?

python numpy

Share Follow

asked Aug 23 '19 at 16:30
Mean Coder
306 ● 1 ● 10

`filtered_values = array[mask[:, np.newaxis]]` ? – jdehesa Aug 23 '19 at 16:32

filtered_values might be different shaped across iterations. How do you plan to store those? – Divakar Aug 23 '19 at 16:55

Add a comment

1 Answer

Active Oldest Votes

Make a smaller 3d array that we can display, and play with:

```
In [2]: arr = np.arange(24).reshape(2,3,4)
In [3]: mask = np.ones((2,4),bool)
In [4]: arr.shape
Out[4]: (2, 3, 4)
In [5]: mask.shape
Out[5]: (2, 4)

In [7]: for i in range(2):
   ...:     print(arr[i,:,mask[i]])
   ...:
[[ 0  4  8]
 [ 1  5  9]
 [ 2  6 10]
 [ 3  7 11]]
[[12 16 20]
```

下一步：閱讀範例與完成作業