vncoder.vn

Bài 9: Advanced Indexing - Numpy trong Python

4-5 phút

Có thể thực hiện lựa chọn từ ndarray là một chuỗi không phải tuple, ndarray kiểu dữ liệu số nguyên hoặc Boolean, hoặc một bộ dữ liệu có ít nhất một phần tử là đối tượng chuỗi. Advanced indexing luôn trả về một bản sao của dữ liệu. Ngược lại, việc cắt chỉ hiển thị một khung nhìn.

Có hai kiểu Advanced indexing - Integer và Boolean.

1. Integer Indexing

Cơ chế này giúp chọn bất kỳ mục tùy ý nào trong một mảng dựa trên chỉ mục Thứ nguyên của nó. Mỗi mảng số nguyên đại diện cho số chỉ mục trong thứ nguyên đó. Khi chỉ mục bao gồm nhiều mảng số nguyên như kích thước của mảng đích, nó sẽ trở nên đơn giản.

Trong ví dụ sau, một phần tử của cột được chỉ định từ mỗi hàng của đối tượng ndarray được chọn. Do đó, chỉ mục hàng chứa tất cả các số hàng và chỉ mục cột chỉ định phần tử được chọn.

Ví dụ 1 :

```
import numpy as np

x = np.array([[1, 2], [3, 4], [5, 6]])
y = x[[0,1,2], [0,1,0]]
```

```
print y
```

Kết quả :

Vùng chọn bao gồm các phần tử tại (0,0), (1,1) và (2,0) từ mảng đầu tiên.

Trong ví dụ sau, các phần tử được đặt ở các góc của mảng 4X3 được chọn. Các chỉ số hàng được lựa chọn là [0, 0] và [3,3] trong khi các chỉ số cột là [0,2] và [0,2].

Ví dụ 2 :

```
import numpy as np
x = np.array([[ 0,  1,  2],[ 3,  4,  5],[ 6,
7,  8],[ 9, 10, 11]])

print 'Our array is:'
print x
print '\n'

rows = np.array([[0,0],[3,3]])
cols = np.array([[0,2],[0,2]])
y = x[rows,cols]

print 'The corner elements of this array are:'
print y
```

Kết quả :

```
Our array is:
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]
```

```
The corner elements of this array are:
[[ 0  2]
 [ 9 11]]
```

kết quả là một ndarray chứa các phần tử góc.

index cơ bản và nâng cao có thể được kết hợp bằng cách sử dụng slice (:) hoặc dấu chấm lửng (…) với một mảng chỉ mục. Ví dụ sau sử dụng lát cắt cho hàng và index nâng cao cho cột. Kết quả là như nhau khi lát cắt được sử dụng cho cả hai. Nhưng index nâng cao dẫn đến bản sao và có thể có bố cục bộ nhớ khác nhau.

Ví dụ 3

```
import numpy as np
x = np.array([[ 0,  1,  2],[ 3,  4,  5],[ 6,
7,  8],[ 9, 10, 11]])

print 'Our array is:'
print x
print '\n'

# slicing
z = x[1:4,1:3]

print 'After slicing, our array becomes:'
print z
print '\n'

# using advanced index for column
y = x[1:4,[1,2]]

print 'Slicing using advanced index for
```

```
column:'
print y
```

```
Our array is:
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]

After slicing, our array becomes:
[[ 4  5]
 [ 7  8]
 [10 11]]

Slicing using advanced index for column:
[[ 4  5]
 [ 7  8]
 [10 11]]
```

2. Boolean Array Indexing

Loại index nâng cao này được sử dụng khi đối tượng kết quả có nghĩa là kết quả của các phép toán Boolean, chẳng hạn như các toán tử so sánh.

Ví dụ 1 :

```
import numpy as np
x = np.array([[ 0,  1,  2],[ 3,  4,  5],[ 6,
7,  8],[ 9, 10, 11]])

print 'Our array is:'
print x
print '\n'
```

```
# Now we will print the items greater than 5
print 'The items greater than 5 are:'
print x[x > 5]
```

Kết quả :

```
Our array is:
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]

The items greater than 5 are:
[ 6  7  8  9 10 11]
```

Ví dụ 2 :

Trong ví dụ này, các phần tử NaN (Không phải là Số) được bỏ qua bằng cách sử dụng ~ (toán tử bổ sung).

```
import numpy as np
a = np.array([np.nan, 1,2,np.nan,3,4,5])
print a[~np.isnan(a)]
```

Kết quả :

Ví dụ 3 :

Ví dụ sau đây cho thấy cách lọc ra các phần tử không phức tạp khỏi một mảng.

```
import numpy as np
a = np.array([1, 2+6j, 5, 3.5+5j])
print a[np.iscomplex(a)]
```

Kết quả :