vncoder.vn

Bài 3: Kiểu dữ liệu

4-5 phút

NumPy hỗ trợ nhiều kiểu dữ liệu hơn nhiều so với Python. Bảng sau đây cho thấy các kiểu dữ liệu vô hướng khác nhau được xác định trong NumPy.

1. **bool_ :** Boolean (Đúng hoặc Sai) được lưu trữ dưới dạng byte
2. **int_ :** Kiểu số nguyên mặc định (giống C long; thường là int64 hoặc int32)
3. **intc :** Giống hệt với int C (thường là int32 hoặc int64)
4. **intp :** Số nguyên được sử dụng để lập chỉ số (giống như C ssize_t; thông thường là int32 hoặc int64)
5. **int8 :** Byte (-128 đến 127)
6. **int16 :** Số nguyên (-32768 đến 32767)
7. **int32 :** Số nguyên (-2147483648 đến 2147483647)
8. **int64 :** Số nguyên (-9223372036854775808 đến 9223372036854775807)
9. **uint8 :** Số nguyên không dấu (0 đến 255)
10. **uint16 :** Số nguyên không dấu (0 đến 65535)
11. **uint32 :** Số nguyên không dấu (0 đến 4294967295)
12. **uint64 :** Số nguyên không dấu (0 đến 18446744073709551615)
13. **float_ :** Viết tắt cho float64

14. **float16 :** Nửa chính xác float: bit dấu, số mũ 5 bit, phần định trị 10 bit
15. **float32 :** số thực chính xác đơn: bit dấu, số mũ 8 bit, phần định trị 23 bit
16. **float64 :** số thực chính xác kép: bit dấu, số mũ 11 bit, phần định trị 52 bit
17. **complex_ :** Viết tắt cho complex128
18. **complex64**
19. **complex128**

    Các kiểu số NumPy là các thể hiện của các đối tượng dtype (kiểu dữ liệu), mỗi đối tượng có các đặc điểm riêng biệt. Các kiểu có sẵn như np.bool_, np.float32, v.v.

Đối tượng kiểu dữ liệu (dtype) :

Một đối tượng kiểu dữ liệu mô tả diễn giải khối bộ nhớ cố định tương ứng với một mảng, tùy thuộc vào các khía cạnh sau

- Loại dữ liệu (integer, float or Python object)
- Kích thước của dữ liệu
- Thứ tự Byte (little-endian hoặc big-endian)
- Trong trường hợp kiểu có cấu trúc, tên của các trường, kiểu dữ liệu của từng trường và một phần của khối bộ nhớ được lấy bởi từng trường.
- Nếu kiểu dữ liệu là một mảng con, hình dạng và kiểu dữ liệu của nó

Thứ tự byte được quyết định bằng cách thêm tiền tố '<' hoặc '>' vào kiểu dữ liệu. '<' có nghĩa là mã hóa có giá trị nhỏ (ít quan trọng nhất được lưu trữ ở địa chỉ nhỏ nhất). '>' có nghĩa là mã

hóa là big-endian (byte quan trọng nhất được lưu trữ ở địa chỉ nhỏ nhất).

Một đối tượng dtype được xây dựng bằng cú pháp sau:

```
numpy.dtype(object, align, copy)
```

Tham số là :

- **Object** − Được chuyển đổi thành đối tượng kiểu dữ liệu
- **Align** −Nếu đúng, thêm phần đệm vào trường để làm cho trường tương tự như C-struct
- **Copy** − Tạo một bản sao mới của đối tượng dtype. Nếu sai, kết quả là tham chiếu đến đối tượng kiểu dữ liệu nội trang

Ví dụ 1:

```
# using array-scalar type
import numpy as np
dt = np.dtype(np.int32)
print dt
```

Kết quả :

Ví dụ 2 :

```
#int8, int16, int32, int64 can be replaced by
equivalent string 'i1', 'i2','i4', etc.
import numpy as np

dt = np.dtype('i4')
print dt
```

Kết quả :

Ví dụ 3 :

```
# using endian notation
import numpy as np
dt = np.dtype('>i4')
print dt
```

Kết quả :

Các ví dụ sau đây cho thấy việc sử dụng kiểu dữ liệu có cấu trúc. Ở đây, tên trường và kiểu dữ liệu vô hướng tương ứng sẽ được khai báo.

Ví dụ 4 :

```
# first create structured data type
import numpy as np
dt = np.dtype([('age',np.int8)])
print dt
```

Kết quả :

Ví dụ 5 :

```
# now apply it to ndarray object
import numpy as np

dt = np.dtype([('age',np.int8)])
a = np.array([(10,),(20,),(30,)], dtype = dt)
print a
```

Kết quả :

Ví dụ 6 :

```
# file name can be used to access content of
age column
import numpy as np

dt = np.dtype([('age',np.int8)])
a = np.array([(10,),(20,),(30,)], dtype = dt)
print a['age']
```

Kết quả :

Ví dụ 7 :

Các ví dụ sau đây xác định kiểu dữ liệu có cấu trúc được gọi là **student** với trường chuỗi là 'name', trường số nguyên 'age' và trường float 'mark'. Loại dtype này được áp dụng cho đối tượng ndarray.

```
import numpy as np
student = np.dtype([('name','S20'), ('age',
'i1'), ('marks', 'f4')])
print student
```

kết quả :

```
[('name', 'S20'), ('age', 'i1'), ('marks',
'<f4')])
```

Ví dụ 8 :

```
import numpy as np

student = np.dtype([('name','S20'), ('age',
'i1'), ('marks', 'f4')])
a = np.array([('abc', 21, 50),('xyz', 18, 75)],
```

```
dtype = student)
print a
```

kết quả :

```
[('abc', 21, 50.0), ('xyz', 18, 75.0)]
```

Mỗi kiểu dữ liệu dựng sẵn có một mã ký tự nhận dạng riêng biệt:

- **'b'** − boolean
- **'i'** − (signed) integer (số nguyên)
- **'u'** − unsigned integer (số nguyên không dấu)
- **'f'** − floating-point (số thực)
- **'c'** − complex-floating point
- **'m'** − timedelta
- **'M'** − datetime
- **'O'** − (Python) objects
- **'S', 'a'** − (byte-)string
- **'U'** − Unicode
- **'V'** − raw data (void)