Python 과학 프로그래밍 기초
7. NumPy (3)

한림대학교 소프트웨어융합대학
박섭형

2021년 1학기

NumPy 다차원 배열 변수의 Assignment, Shallow Copy, Deep Copy

NumPy 다차원 배열 변수의 Assignment

[1]:
```
import numpy as np
x = np.arange(5)
y = x
print(id(x), id(y))
```

```
2281193574928 2281193574928
```

[2]:
```
print("x = ", x)
print("y = ", y)
x[3] = 10
print("x = ", x)
print("y = ", y)
```

```
x =  [0 1 2 3 4]
y =  [0 1 2 3 4]
x =  [ 0  1  2 10  4]
y =  [ 0  1  2 10  4]
```

NumPy 다차원 배열 변수의 View는 shallow copy

[3]:
```
x = np.arange(5)
print("x = ", x)
```

```
y = x[1:4]
print("y = ", y)
print(id(x), id(y))
```

```
x =  [0 1 2 3 4]
y =  [1 2 3]
2281197865120 2281197892592
```

[4]:
```
y[1] = 10
print("y = ", y)
print("x = ", x)
```

```
y =  [ 1 10  3]
x =  [ 0  1 10  3  4]
```

[5]:
```
x[3] = 20
print("y = ", y)
print("x = ", x)
```

```
y =  [ 1 10 20]
x =  [ 0  1 10 20  4]
```

[6]:
```
xx = x.view()
print(x)
print(xx)
```

```
[ 0  1 10 20  4]
[ 0  1 10 20  4]
```

[7]:
```
print(id(x), id(xx))
```

```
2281197865120 2281197893872
```

[8]:
```
x[3] = 30
print("x = ", x)
```

```
print("xx = ", xx)
```

```
x =  [ 0  1 10 30  4]
xx =  [ 0  1 10 30  4]
```

[9]:
```
a = np.arange(24).reshape(2,3,4)
a
```

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

       [[12, 13, 14, 15],
        [16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

[10]:
```
b = a[:,:,::2]
print(b.shape)
b
```

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

[10]:
```
array([[[ 0,  2],
        [ 4,  6],
        [ 8, 10]],

       [[12, 14],
        [16, 18],
        [20, 22]]])
```

[11]:
```
b[0, 2, 0] = 88
b
```

[11]:
```
array([[[ 0,  2],
        [ 4,  6],
        [88, 10]],

       [[12, 14],
```

```
           [16, 18],
           [20, 22]]])
```

[12]:
```
a
```

[12]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7],
        [88,  9, 10, 11]],

       [[12, 13, 14, 15],
        [16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

[13]:
```
a[1,2,2] = 220
a
```

[13]:
```
array([[[  0,   1,   2,   3],
        [  4,   5,   6,   7],
        [ 88,   9,  10,  11]],

       [[ 12,  13,  14,  15],
        [ 16,  17,  18,  19],
        [ 20,  21, 220,  23]]])
```

[14]:
```
b
```

[14]:
```
array([[[  0,   2],
        [  4,   6],
        [ 88,  10]],

       [[ 12,  14],
        [ 16,  18],
        [ 20, 220]]])
```

NumPy 다차원 배열 변수의 Deep Copy: ndarray.copy

[15]:
```
import numpy as np
x = np.arange(4)
y = np.copy(x)
```

```
z = x.copy()
print("x = ", x)
print("y = ", y)
print("z = ", z)
print(id(x), id(y), id(z))
```

```
x =  [0 1 2 3]
y =  [0 1 2 3]
z =  [0 1 2 3]
2281197991056 2281197990176 2281197892592
```

[16]:
```
x[1] = 10
print("x = ", x)
print("y = ", y)
print("z = ", z)
```

```
x =  [ 0 10  2  3]
y =  [0 1 2 3]
z =  [0 1 2 3]
```

NumPy 다차원 배열의 재구성

NumPy 다차원 배열의 shape 변경

[17]:
```
np.arange(6).reshape(2,3)
```

[17]:
```
array([[0, 1, 2],
       [3, 4, 5]])
```

[18]:
```
np.arange(6).reshape(2,-1)
```

[18]:
```
array([[0, 1, 2],
       [3, 4, 5]])
```

[19]:
```
np.arange(6).reshape(-1,3)
```

[19]:
```
array([[0, 1, 2],
       [3, 4, 5]])
```

[20]:
```
np.arange(24).reshape(3,2,4)
```

[20]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7]],

       [[ 8,  9, 10, 11],
        [12, 13, 14, 15]],

       [[16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

[21]:
```
np.arange(24).reshape(3,2,-1)
```

[21]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7]],

       [[ 8,  9, 10, 11],
        [12, 13, 14, 15]],

       [[16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

[22]:
```
np.arange(24).reshape(3,-1,4)
```

[22]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7]],

       [[ 8,  9, 10, 11],
        [12, 13, 14, 15]],

       [[16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

[23]:
```
np.arange(24).reshape(-1,2,4)
```

[23]:
```
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7]],

       [[ 8,  9, 10, 11],
        [12, 13, 14, 15]],
```

```
       [[16, 17, 18, 19],
        [20, 21, 22, 23]]])
```

NumPy 다차원 배열의 shape 변경

[24]:
```
a = np.arange(6).reshape(2,3)
a.reshape(6)
```

[24]:
```
array([0, 1, 2, 3, 4, 5])
```

[25]:
```
a = np.arange(6).reshape(2,3)
a.reshape(6,)
```

[25]:
```
array([0, 1, 2, 3, 4, 5])
```

[26]:
```
a = np.arange(6).reshape(2,3)
a.reshape(-1)
```

[26]:
```
array([0, 1, 2, 3, 4, 5])
```

[27]:
```
a = np.arange(6).reshape(2,3)
a.reshape(-1,)
```

[27]:
```
array([0, 1, 2, 3, 4, 5])
```

[28]:
```
a = np.arange(24).reshape(3, 2, 4)
a.reshape(-1,)
```

[28]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23])
```

[29]:
```
a = np.arange(24).reshape(3, 2, 4)
a.reshape(4,6)
```

[29]:
```
array([[ 0,  1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23]])
```

[30]:
```
a = np.arange(24).reshape(3, 2, 4)
a.reshape(4,-1)
```

[30]:
```
array([[ 0,  1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23]])
```

[31]:
```
a = np.arange(24).reshape(3, 2, 4)
a.reshape(-1,6)
```

[31]:
```
array([[ 0,  1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10, 11],
       [12, 13, 14, 15, 16, 17],
       [18, 19, 20, 21, 22, 23]])
```

[32]:
```
a = np.arange(24).reshape(3, 2, 4)
a.reshape(4, 3, 2)
```

[32]:
```
array([[[ 0,  1],
        [ 2,  3],
        [ 4,  5]],

       [[ 6,  7],
        [ 8,  9],
        [10, 11]],

       [[12, 13],
        [14, 15],
        [16, 17]],

       [[18, 19],
        [20, 21],
        [22, 23]]])
```

[33]:
```
a = np.arange(24).reshape(3, 2, 4)
b = a.reshape(2,-1)
b[1,3] = 100

print("Address of a:", id(a))
print("Address of b:", id(b))
```

```
print(a)
print(b)
print(a[1,1,3])
```

```
Address of a: 2281198029440
Address of b: 2281198095696
[[[  0   1   2   3]
  [  4   5   6   7]]

 [[  8   9  10  11]
  [ 12  13  14 100]]

 [[ 16  17  18  19]
  [ 20  21  22  23]]]
[[  0   1   2   3   4   5   6   7   8   9  10  11]
 [ 12  13  14 100  16  17  18  19  20  21  22  23]]
100
```

ravel()과 flatten()

- ravel(): 원 배열의 view를 반환
- flatten(): 원 배열의 복사본을 반환

[34]:
```
a = np.arange(6).reshape(2,3)
b = a.ravel()
print(b)
c = a.flatten()
print(c)
```

```
[0 1 2 3 4 5]
[0 1 2 3 4 5]
```

[35]:
```
a[0,0] = 10
print("a =", a)
print("b =", b)
print("c =", c)
```

```
a = [[10  1  2]
 [ 3  4  5]]
b = [10  1  2  3  4  5]
c = [0 1 2 3 4 5]
```

tile()

[36]:
```
np.tile(np.arange(3), 3)
```

[36]:
```
array([0, 1, 2, 0, 1, 2, 0, 1, 2])
```

[37]:
```
np.tile(np.arange(3), (3,2))
```

[37]:
```
array([[0, 1, 2, 0, 1, 2],
       [0, 1, 2, 0, 1, 2],
       [0, 1, 2, 0, 1, 2]])
```

[38]:
```
np.tile(np.arange(6).reshape(2,3), 3)
```

[38]:
```
array([[0, 1, 2, 0, 1, 2, 0, 1, 2],
       [3, 4, 5, 3, 4, 5, 3, 4, 5]])
```

[39]:
```
np.tile(np.arange(6).reshape(2,3), (3,2))
```

[39]:
```
array([[0, 1, 2, 0, 1, 2],
       [3, 4, 5, 3, 4, 5],
       [0, 1, 2, 0, 1, 2],
       [3, 4, 5, 3, 4, 5],
       [0, 1, 2, 0, 1, 2],
       [3, 4, 5, 3, 4, 5]])
```

[40]:
```
a = np.arange(6).reshape(2,3)
b = np.tile(a, (3,2))
a[0,0] = 10
print(b)
```

```
[[0 1 2 0 1 2]
 [3 4 5 3 4 5]
 [0 1 2 0 1 2]
 [3 4 5 3 4 5]
 [0 1 2 0 1 2]
 [3 4 5 3 4 5]]
```

stack(), vstack(), hstack()

[41]:
```
a = np.arange(3)
b = np.arange(3, 6)
c = np.stack((a,b))
print(c)
print(c.shape)
```

```
[[0 1 2]
 [3 4 5]]
(2, 3)
```

[42]:
```
d = np.arange(10, 16).reshape(2,3)
e = np.stack((c,d), axis=0)
print(e)
print(e.shape)
```

```
[[[ 0  1  2]
  [ 3  4  5]]

 [[10 11 12]
  [13 14 15]]]
(2, 2, 3)
```

[43]:
```
f = np.stack((c,d), axis=1)
print(f)
print(f.shape)
```

```
[[[ 0  1  2]
  [10 11 12]]

 [[ 3  4  5]
  [13 14 15]]]
(2, 2, 3)
```

[44]:
```
g = np.stack((c,d), axis=2)
print(c)
```

```
print(d)
print(g)
print(g.shape)
```

```
[[0 1 2]
 [3 4 5]]
[[10 11 12]
 [13 14 15]]
[[[ 0 10]
  [ 1 11]
  [ 2 12]]

 [[ 3 13]
  [ 4 14]
  [ 5 15]]]
(2, 3, 2)
```

[45]:
```
h1 = np.hstack((np.array([1,2]),np.array([3,4])))
print(h1)
print(h1.shape)
```

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

[46]:
```
h2 = np.hstack((np.array([1,2, 3]),np.array([4,5])))
print(h2)
print(h2.shape)
```

```
[1 2 3 4 5]
(5,)
```

[47]:
```
h3 = np.hstack((np.array([1,2, 3]),
                np.array([4,5]),
               np.array([6, 7, 8, 9])))
print(h3)
print(h3.shape)
```

```
[1 2 3 4 5 6 7 8 9]
(9,)
```

[48]:
```
h4 = np.hstack(([1,2],[3,4]))
print(h4)
print(h4.shape)
```

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

[49]:
```
h5 = np.hstack((c,d))
print(c)
print(d)
print(c.shape)
print(d.shape)
print(h5)
print(h5.shape)
```

```
[[0 1 2]
 [3 4 5]]
[[10 11 12]
 [13 14 15]]
(2, 3)
(2, 3)
[[ 0  1  2 10 11 12]
 [ 3  4  5 13 14 15]]
(2, 6)
```

[50]:
```
aa = np.arange(24).reshape(2,3,4)
bb = np.arange(32).reshape(2,4,4)

h6 = np.hstack((aa,bb))
print(aa.shape)
print(bb.shape)
print(h6.shape)
print(h6)
```

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

 [[12 13 14 15]
  [16 17 18 19]
  [20 21 22 23]
  [16 17 18 19]
  [20 21 22 23]
  [24 25 26 27]
  [28 29 30 31]]]
```

[51]:
```
aa = np.arange(3)
bb = np.arange(3,6)
cc = np.arange(6,9)

h7 = np.vstack((aa,bb,cc))
print(h7)
print(h7.shape)
```

```
[[0 1 2]
 [3 4 5]
 [6 7 8]]
(3, 3)
```

[52]:
```
aa = np.arange(24).reshape(2,3,4)
bb = np.arange(24, 60).reshape(3,3,4)
h8 = np.vstack((aa,bb))
print(h8)
```

```
print(h8.shape)
```

```
[[[ 0  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 51]
  [52 53 54 55]
  [56 57 58 59]]]
(5, 3, 4)
```