영상처리프로그래밍 : NumPy (2)

한림대학교 소프트웨어융합대학
박섭형

2022년 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))
```

```
2136487264592 2136487264592
```

[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]
2136487265072 2136508277648
```

[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))
```

```
2136487265072 2136508278416
```

[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]
2136508277168 2136508279376 2136487264592
```

[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: 2136508382448
```

```
Address of b: 2136508381584
[[[  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]
```

Broadcasting

- Shape이 다른 두 개의 ndarray들을 이용해서 산술 연산을 할 때 numpy가 ndarrays를 처리하는 방법

Shape이 같은 두 ndarray의 산술 연산

- 원소끼리 연산이 이루어짐

[36]:
```
a = np.array([1, 2, 3])
b = np.array([10, 20, 30])
a + b
```

[36]:
```
array([11, 22, 33])
```

[37]:
```
a * b
```

[37]:
```
array([10, 40, 90])
```

[38]:
```
a = np.arange(12).reshape(3,4)
a
```

[38]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

[39]:
```
b = np.arange(10, 22).reshape(3,4)
b
```

[39]:
```
array([[10, 11, 12, 13],
       [14, 15, 16, 17],
       [18, 19, 20, 21]])
```

[40]:
```
a + b
```

[40]:
```
array([[10, 12, 14, 16],
       [18, 20, 22, 24],
       [26, 28, 30, 32]])
```

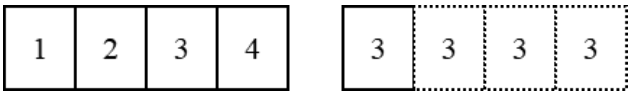
[41]:
```
a = np.array([1, 2, 3])
b = np.array([1, 2])
a + b
```

```
␣
↪---------------------------------------------------------------------------

       ValueError                                Traceback (most recent call␣
↪last)

       C:\Users\Public\Documents\ESTsoft\CreatorTemp/ipykernel_34148/3439319059.
↪py in <module>
         1 a = np.array([1, 2, 3])
         2 b = np.array([1, 2])
   ----> 3 a + b


       ValueError: operands could not be broadcast together with shapes (3,)␣
↪(2,)
```

ndarray와 상수의 산술 연산

[42]:
```
x = np.arange(1,5)
print(x)
print(x + 3)
```

```
[1 2 3 4]
[4 5 6 7]
```



[43]:
```
print(x * 3)
```

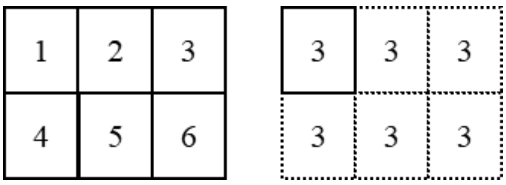
```
[ 3  6  9 12]
```

[44]:
```
print(x / 3)
```

```
[0.33333333 0.66666667 1.         1.33333333]
```

[45]:
```
x = np.arange(1,7).reshape(2,3)
print(x + 3)
```

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



[46]:
```
print(3 * x)
```

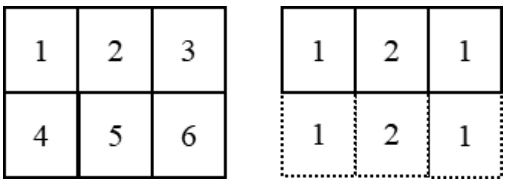
```
[[ 3  6  9]
 [12 15 18]]
```

Shape이 서로 다른 두 ndarray 사이의 산술 연산

axis의 원소의 갯수가 같은 두 ndarray 사이의 산술 연산

[47]:
```
a = np.arange(1,7).reshape(2,3)
b = np.array([[1, 2, 1]])
print(a + b)
```

```
[[2 4 4]
 [5 7 7]]
```



[48]:
```
a = np.arange(1,7).reshape(2,3)
b = np.array([1, 2, 1])
print(a + b)
```

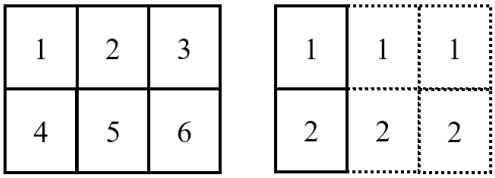
```
[[2 4 4]
 [5 7 7]]
```

[49]:
```
a = np.arange(1,7).reshape(2,3)
b = np.array([1, 2])
print(a + b)
```

```
␣
↪---------------------------------------------------------------------------

       ValueError                                Traceback (most recent call␣
↪last)

       C:\Users\Public\Documents\ESTsoft\CreatorTemp/ipykernel_34148/1824696652.
↪py in <module>
         1 a = np.arange(1,7).reshape(2,3)
         2 b = np.array([1, 2])
   ----> 3 print(a + b)

       ValueError: operands could not be broadcast together with shapes (2,3)␣
↪(2,)
```

[50]:
```
a = np.arange(1,7).reshape(2,3)
b = np.array([[1], [2]])
print(a + b)
```

```
[[2 3 4]
 [6 7 8]]
```



[51]:
```
a = np.arange(1,7).reshape(2,3)
b = np.array([[1, 1], [2, 2]])
print(a + b)
```

```
␣
↪---------------------------------------------------------------------------

ValueError                                Traceback (most recent call␣
↪last)

C:\Users\Public\Documents\ESTsoft\CreatorTemp/ipykernel_34148/2081393599.
↪py in <module>
      1 a = np.arange(1,7).reshape(2,3)
      2 b = np.array([[1, 1], [2, 2]])
----> 3 print(a + b)

ValueError: operands could not be broadcast together with shapes (2,3)␣
↪(2,2)
```

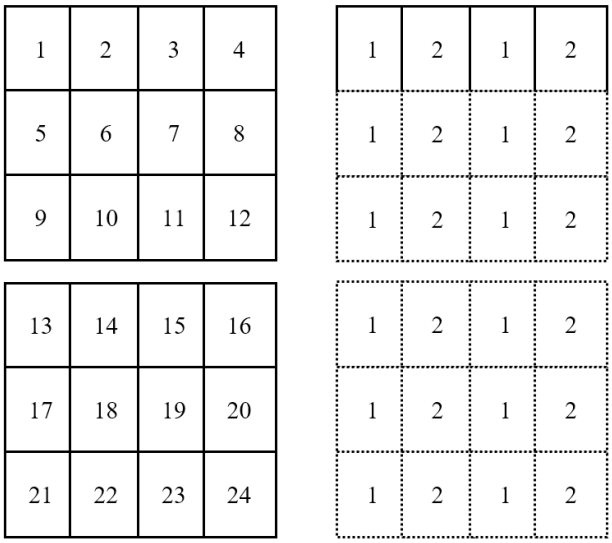
[52]:
```
a = np.arange(1,9).reshape(2,4)
b = np.array([[1, 1], [2, 2]])
print(a + b)
```

```
␣
↪---------------------------------------------------------------------------

ValueError                                Traceback (most recent call␣
↪last)

C:\Users\Public\Documents\ESTsoft\CreatorTemp/ipykernel_34148/3663828647.
↪py in <module>
```

```
      1 a = np.arange(1,9).reshape(2,4)
      2 b = np.array([[1, 1], [2, 2]])
----> 3 print(a + b)

ValueError: operands could not be broadcast together with shapes (2,4)␣
↪(2,2)
```

[53]:
```
a = np.arange(1,25).reshape(2,3,4)
b = np.array([1, 2, 1, 2])
print(a + b)
```

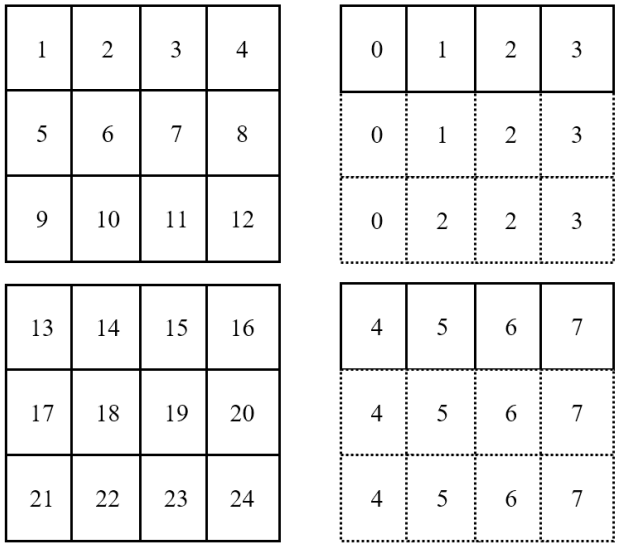
```
[[[ 2  4  4  6]
  [ 6  8  8 10]
  [10 12 12 14]]

 [[14 16 16 18]
  [18 20 20 22]
  [22 24 24 26]]]
```



[54]:
```
a = np.arange(1, 25).reshape(2,3,4)
b = np.arange(8).reshape(2,1,4)
```

```
print(a)
print(b)
print(a + b)
```

```
[[[ 1  2  3  4]
  [ 5  6  7  8]
  [ 9 10 11 12]]

 [[13 14 15 16]
  [17 18 19 20]
  [21 22 23 24]]]
[[[0 1 2 3]]

 [[4 5 6 7]]]
[[[ 1  3  5  7]
  [ 5  7  9 11]
  [ 9 11 13 15]]

 [[17 19 21 23]
  [21 23 25 27]
  [25 27 29 31]]]
```



[55]:
```
a = np.array([1,2,3]).reshape(3,1)
b = np.array([1,2,1]).reshape(1,3)
print(a)
print(b)
print(a + b)
```

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