A L G O R I T H M

P R A C T I C E

Numpy库基础

Steven Tang

渡一教育
Duyi Education

NumPy 是什么？

- NumPy是Python的一种开源的数值计算扩展库。它包含很多功能：

- 创建n维数组（矩阵）
- 对数组进行函数运算
- 数值积分
- 线性代数运算
- 傅里叶变换
- 随机数产生
- ……

一大波API正在靠近！！！！

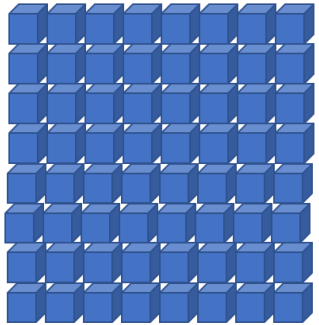
NumPy 是什么?

- NumPy是在1995年诞生的Python库Numeric的基础上建立起来的。但真正促使NumPy的发行的是Python的SciPy库。
- SciPy是2001年发行的一个类似于Matlab，Maple，Mathematica等数学计算软件的Python库，它实现里面的大多数功能。

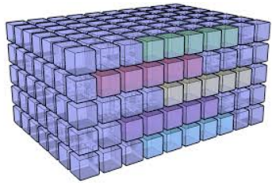
```
In [1]: import numpy

In [2]: numpy.__version__
Out[2]: '1.18.1'

In [3]:
```

高维数据

一维



二维



三维

普通python数组的问题

ndarray的创建

NumPy中的核心对象是ndarray。
ndarray可以看成数组，类似于R语言的向量或者矩阵。
NumPy里面所有的函数都是围绕ndarray展开的。

```
In [3]: import numpy as np

In [4]: a = np.array([1, 2, 3, 4])

In [5]: a = np.array((1, 2, 3, 4))

In [6]: a
Out[6]: array([1, 2, 3, 4])

In [7]: a.shape
Out[7]: (4,)

In [8]: type(a)
Out[8]: numpy.ndarray
```

```
In [9]: a[0]
Out[9]: 1

In [10]: a[0]=2

In [11]: a
Out[11]: array([2, 2, 3, 4])
```

ndarray的创建

ndarray对维数没有限制。

[ ]从内到外分别为第0轴，第1轴，第2轴。

c第0轴长度为3，第1轴长度为4。

```
In [12]: b=np.array([[1,2,3],[3,4,5]])

In [13]: b.shape
Out[13]: (2, 3)

In [14]: b[1,2]
Out[14]: 5
```

```
In [21]: c=np.array([[1,2,3],[2,3]])

In [22]: c[1]
Out[22]: [2, 3]

In [23]: c[1][1]
Out[23]: 3

In [24]: c[1][2]
Traceback (most recent call last):

  File "<ipython-input-24-e74963c1e143>", line 1, in
<module>
    c[1][2]

IndexError: list index out of range
```

ndarray的创建

ndarray对维数没有限制。

[ ]从内到外分别为第0轴，第1轴，第2轴.....

```
In [12]: b=np.array([[1,2,3],[3,4,5]])

In [13]: b.shape
Out[13]: (2, 3)

In [14]: b[1,2]
Out[14]: 5
```

```
In [21]: c=np.array([[1,2,3],[2,3]])

In [22]: c[1]
Out[22]: [2, 3]

In [23]: c[1][1]
Out[23]: 3

In [24]: c[1][2]
Traceback (most recent call last):

  File "<ipython-input-24-e74963c1e143>", line 1, in
<module>
    c[1][2]

IndexError: list index out of range
```

ndarray的创建

```
In [26]: a=np.zeros([4,3])

In [27]: a
Out[27]:
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
In [28]: b=np.ones([2,3,4])

In [29]: b
Out[29]:
array([[[1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.]],

       [[1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.]]])
```

```
In [30]: c=np.array([[1,2,3],[3,4,5],[5,6,7],
[7,8,9]])

In [31]: c.shape
Out[31]: (4, 3)

In [32]: d=np.zeros_like(c)

In [33]: d
Out[33]:
array([[0, 0, 0],
       [0, 0, 0],
       [0, 0, 0],
       [0, 0, 0]])
```

ndarray的创建(random)

```
In [78]: a=np.random.random((3,3))

In [79]: a
Out[79]:
array([[0.79875186, 0.45771437, 0.04968383],
       [0.6051633 , 0.21668759, 0.33554397],
       [0.3878375 , 0.50245591, 0.50258755]])

In [80]: b=np.random.randint(10,size=(5,5))

In [81]: b
Out[81]:
array([[4, 9, 8, 7, 9],
       [5, 9, 8, 2, 6],
       [4, 1, 4, 7, 0],
       [7, 4, 6, 0, 0],
       [4, 0, 9, 7, 2]])
```

```
In [84]: c=np.random.normal(size=[3,4])

In [85]: c
Out[85]:
array([[-0.17234932, -0.75148405, -0.36387373, -0.45728386],
       [-1.04081629,  0.18870088,  0.67632058,  1.44110169],
       [-0.45944254, -0.13865626, -0.04428486, -0.67464427]])

In [86]: d=np.random.uniform(size=[2,2])

In [87]: d
Out[87]:
array([[0.78807662, 0.43159557],
       [0.70345374, 0.13496507]])
```

ndarray的创建(arrange和linespace)

```
In [36]: a=np.arange(10)

In [37]: a
Out[37]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [38]: b=np.arange(0.1,1,0.1)

In [39]: b
Out[39]: array([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7,
0.8, 0.9])
```

```
In [40]: c=np.linspace(0,1,10)

In [41]: c
Out[41]:
array([0.        , 0.11111111, 0.22222222,
0.33333333, 0.44444444,
       0.55555556, 0.66666667, 0.77777778,
0.88888889, 1.        ])

In [42]: d=np.linspace(0,1,10,endpoint=False)

In [43]: d
Out[43]: array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6,
0.7, 0.8, 0.9])
```

数据类型的确定，dtype属性

```
In [44]: a=np.array([1,2,3,4],np.int)

In [45]: a
Out[45]: array([1, 2, 3, 4])

In [46]: a=np.array([1,2,3,4],np.float)

In [47]: a
Out[47]: array([1., 2., 3., 4.])
```

```
In [50]: b=np.array([1,2,3,4])

In [51]: b.dtype
Out[51]: dtype('int32')

In [52]: a+b
Out[52]: array([2., 4., 6., 8.])
```

np

```
In [44]: a=np.array([1,2,3,4],np.int)

In [45]: a
Out[45]: array([1, 2, 3, 4])

In [46]: a=np.array([1,2,3,4],np.float)

In [47]: a
Out[47]: array([1., 2., 3., 4.])
```

```
In [50]: b=np.array([1,2,3,4])

In [51]: b.dtype
Out[51]: dtype('int32')

In [52]: a+b
Out[52]: array([2., 4., 6., 8.])
```

数据类型的确定，dtype属性

```
In [44]: a=np.array([1,2,3,4],np.int)

In [45]: a
Out[45]: array([1, 2, 3, 4])

In [46]: a=np.array([1,2,3,4],np.float)

In [47]: a
Out[47]: array([1., 2., 3., 4.])
```

```
In [50]: b=np.array([1,2,3,4])

In [51]: b.dtype
Out[51]: dtype('int32')

In [52]: a+b
Out[52]: array([2., 4., 6., 8.])
```

数据类型的确定，dtype属性

```
In [44]: a=np.array([1,2,3,4],np.int)

In [45]: a
Out[45]: array([1, 2, 3, 4])

In [46]: a=np.array([1,2,3,4],np.float)

In [47]: a
Out[47]: array([1., 2., 3., 4.])
```

```
In [50]: b=np.array([1,2,3,4])

In [51]: b.dtype
Out[51]: dtype('int32')

In [52]: a+b
Out[52]: array([2., 4., 6., 8.])
```

numpy shape和reshape方法

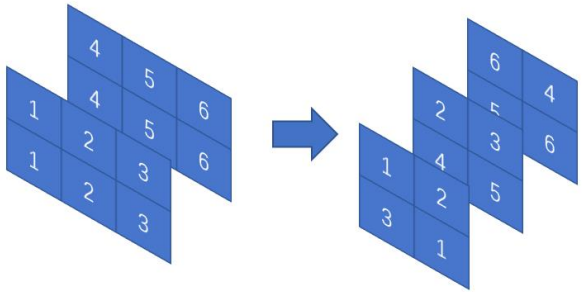
```
In [88]: a=np.array([[1,2,3],[4,5,6]])

In [89]: a.reshape(3,2)
Out[89]:
array([[1, 2],
       [3, 4],
       [5, 6]])
```

1	2	3
4	5	6

→

1	2
3	4
5	6

```
In [91]: a=np.array([[[1,2,3],[1,2,3]],[[4,5,6],[4,5,6]]])

In [92]: a.shape
Out[92]: (2, 2, 3)

In [93]: a.reshape(3,2,2)
Out[93]:
array([[[1, 2],
        [3, 1]],

       [[2, 3],
        [4, 5]],

       [[6, 4],
        [5, 6]]])
```



numpy shape和reshape方法

```
In [94]: a=np.ones([3,4])

In [95]: a.reshape(-1,6)
Out[95]:
array([[1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.]])

In [96]: a
Out[96]:
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

```
In [97]: a.reshape(-1)
Out[97]: array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
1.])

In [98]: a.reshape(4,-1)
Out[98]:
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])

In [99]: a.reshape(-1,-1)
Traceback (most recent call last):

  File "<ipython-input-99-aa43799da6cd>", line 1, in <module>
    a.reshape(-1,-1)

ValueError: can only specify one unknown dimension
```

numpy 的切片

```
In [108]: a=np.array([1,2,3,4,5])

In [109]: b=a[1:3]

In [110]: b
Out[110]: array([2, 3])

In [111]: a[1]=10

In [112]: b
Out[112]: array([10,  3])
```

ndarray通过切片产生一个新的数组b，**b和a共享同一块数据存储空间。**

```
In [113]: a=[1,2,3,4,5]

In [114]: b=a[1:3]

In [115]: a[1]=10

In [116]: b
Out[116]: [2, 3]
```

numpy 的切片

```
In [105]: ls=[[4, 9, 8, 7, 9],
     ...:       [5, 9, 8, 2, 6],
     ...:       [4, 1, 4, 7, 0],
     ...:       [7, 4, 6, 0, 0],
     ...:       [4, 0, 9, 7, 2]]

In [106]: ls[1:3]
Out[106]: [[5, 9, 8, 2, 6], [4, 1, 4, 7, 0]]

In [107]: ls[1:3,2:4]
Traceback (most recent call last):

  File "<ipython-input-107-c08d9f092322>", line 1, in
<module>
    ls[1:3,2:4]

TypeError: list indices must be integers or slices, not tuple
```

```
In [3]: a=np.array([[[1,2,3],[2,3,4]],[[3,4,5],[4,5,6]]])

In [4]: a[...,:2]
Out[4]:
array([[[1, 2],
        [2, 3]],

       [[3, 4],
        [4, 5]]])
```

```
In [100]: b[:3,:4]
Out[100]:
array([[4, 9, 8, 7],
       [5, 9, 8, 2],
       [4, 1, 4, 7]])

In [101]: b
Out[101]:
array([[4, 9, 8, 7, 9],
       [5, 9, 8, 2, 6],
       [4, 1, 4, 7, 0],
       [7, 4, 6, 0, 0],
       [4, 0, 9, 7, 2]])

In [102]: b[:3,:4]
Out[102]:
array([[4, 9, 8, 7],
       [5, 9, 8, 2],
       [4, 1, 4, 7]])
```

numpy 的运算

```
In [2]: import numpy as np

In [3]: a=np.array([1,2,3,4])

In [4]: a+1
Out[4]: array([2, 3, 4, 5])

In [5]: la=[1,2,3,4]
```

```
In [5]: la=[1,2,3,4]

In [6]: la+1
Traceback (most recent call last):

  File "<ipython-input-6-6c1eddcfa633>", line 1, in <module>
    la+1

TypeError: can only concatenate list (not "int") to list


In [7]:

In [7]: la_plus1=[i+1 for i in la]

In [8]: la_plus1
Out[8]: [2, 3, 4, 5]
```

numpy 的运算

```
In [20]: a=np.array([1,2,3,4])

In [21]: b=np.array([3,4,5,6])

In [22]: a+b
Out[22]: array([ 4,  6,  8, 10])

In [23]: b=np.array([3,4,5])

In [24]: a+b
Traceback (most recent call last):

  File "<ipython-input-24-ca730b97bf8a>", line 1, in <module>
    a+b

ValueError: operands could not be broadcast together with
shapes (4,) (3,)
```

```
In [12]: np.sin(angles)
Out[12]:
array([8.66025404e-01, 7.07106781e-01, 1.00000000e+00,
5.00000000e-01,
       1.22464680e-16])
```

numpy 的切片

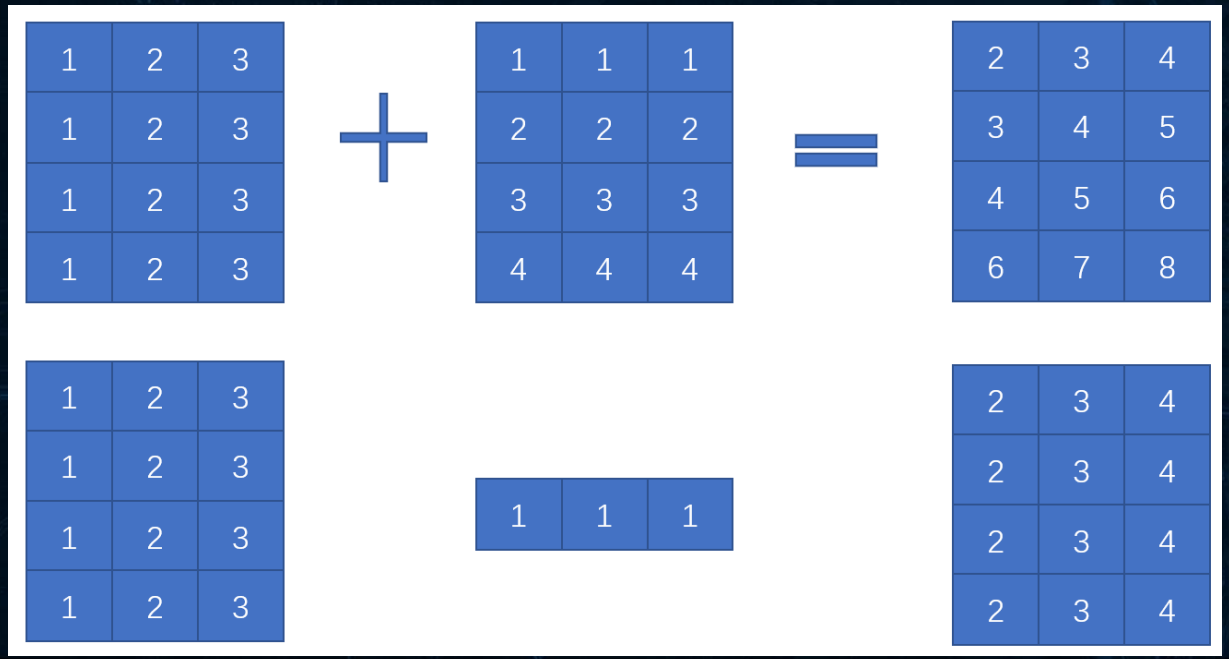
```
In [31]: np.divide(a,b)
Out[31]: array([0.33333333, 0.5       , 0.6       ,
0.66666667])

In [32]: np.add(a,b)
Out[32]: array([ 4,  6,  8, 10])

In [33]: np.subtract(a,b)
Out[33]: array([-2, -2, -2, -2])

In [34]: np.multiply(a,b)
Out[34]: array([ 3,  8, 15, 24])

In [35]: np.divide(a,b)
Out[35]: array([0.33333333, 0.5       , 0.6       ,
0.66666667])
```

numpy 数组之间的比较

```
In [9]: (np.array([1, 1, 3]) < np.array([2, 2,
1])).astype(np.int)
Out[9]: array([1, 1, 0])

In [10]: np.array([1, 1, 3]) < np.array([2, 2, 1])
Out[10]: array([ True,  True, False])

In [11]: np.int(np.array([1, 1, 3]) < np.array([2, 2, 1]))
Traceback (most recent call last):

  File "<ipython-input-11-d80d5c20d882>", line 1, in <module>
    np.int(np.array([1, 1, 3]) < np.array([2, 2, 1]))

TypeError: only size-1 arrays can be converted to Python
scalars
```

```
In [13]: b=np.array([1, 1, 3]) < np.array([2, 2, 1])

In [14]: b.astype(np.int)
Out[14]: array([1, 1, 0])
```

numpy 的广播机制（重要）



numpy 的广播机制（重要）

```
In [16]: a=np.array([[1,2,3],[1,2,3],[1,2,3],[1,2,3]])

In [17]: b=np.array([1,1,1])

In [18]: a+b
Out[18]:
array([[2, 3, 4],
       [2, 3, 4],
       [2, 3, 4],
       [2, 3, 4]])

In [19]: c=np.array([[1,1,1]])

In [20]: a+c
Out[20]:
array([[2, 3, 4],
       [2, 3, 4],
       [2, 3, 4],
       [2, 3, 4]])
```

numpy 的广播机制（重要）

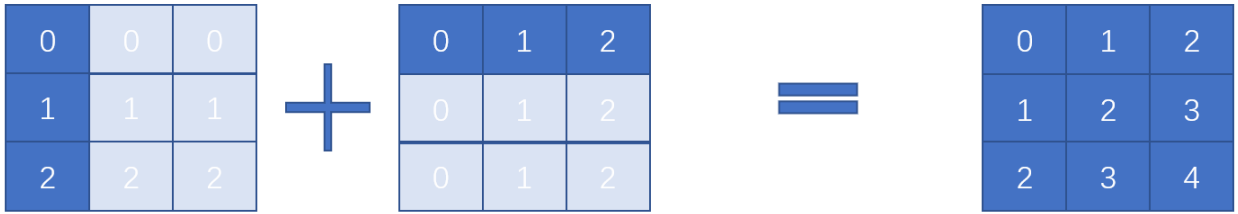
```
In [21]: a=np.array([[[1,2],[1,2],[1,2]],[[3,4],[3,4],
[3,4]]])

In [22]: a
Out[22]:
array([[[1, 2],
        [1, 2],
        [1, 2]],

       [[3, 4],
        [3, 4],
        [3, 4]]])

In [23]: a.shape
Out[23]: (2, 3, 2)

In [24]: b=np.array([[1,2,3],[1,2,3]])
```

```
In [23]: a.shape
Out[23]: (2, 3, 2)

In [24]: b=np.array([[1,2,3],[1,2,3]])

In [25]: b.shape
Out[25]: (2, 3)

In [26]: a+b
Traceback (most recent call last):

  File "<ipython-input-26-ca730b97bf8a>", line 1, in <module>
    a+b

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

```
In [27]: c=np.array([[1,2],[2,3],[3,4]])

In [28]: a+c
Out[28]:
array([[[2, 4],
        [3, 5],
        [4, 6]],

       [[4, 6],
        [5, 7],
        [6, 8]]])
```

numpy 的广播机制（重要）

```
In [29]: a=np.array([[1,2,3]])

In [30]: b=np.array([2,3,4])

In [31]: a+b
Out[31]: array([[3, 5, 7]])

In [32]: np.arange(3).reshape(3,1)+np.arange(3)
Out[32]:
array([[0, 1, 2],
       [1, 2, 3],
       [2, 3, 4]])
```



求和，平均值，方差

```
In [35]: a=np.random.normal(size=(2000,))

In [36]: np.mean(a)
Out[36]: -0.005297785391031539

In [37]: np.std(a)
Out[37]: 1.0088318547105104

In [38]: np.sum(a)
Out[38]: -10.595570782063078
```

```
In [39]: a=np.arange(12).reshape(3,4)

In [40]: a
Out[40]:
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])

In [41]: np.sum(a,axis=1)
Out[41]: array([ 6, 22, 38])

In [42]: np.sum(a,axis=0)
Out[42]: array([12, 15, 18, 21])
```

```
In [43]: np.sum(a,1,keepdims=True)
Out[43]:
array([[ 6],
       [22],
       [38]])

In [44]: np.sum(a,axis=0,keepdims=True)
Out[44]: array([[12, 15, 18, 21]])
```

numpy 维度改变

```
In [45]: a=np.array([1,2,3])

In [46]: a=a[None,:]

In [47]: a
Out[47]: array([[1, 2, 3]])

In [48]: a.shape
Out[48]: (1, 3)

In [49]: a=np.array([1,2,3])

In [50]: a=a[:,None]

In [51]: a.shape
Out[51]: (3, 1)
```

```
In [55]: a=np.array([1,2,3])

In [56]: a.shape
Out[56]: (3,)

In [57]: np.expand_dims(a,axis=0)
Out[57]: array([[1, 2, 3]])

In [58]: b=np.expand_dims(a,axis=0)

In [59]: b.shape
Out[59]: (1, 3)

In [60]: c=np.expand_dims(b,axis=2)

In [61]: c.shape
Out[61]: (1, 3, 1)
```

```
In [62]: np.squeeze(c)
Out[62]: array([1, 2, 3])

In [63]: np.squeeze(c).shape
Out[63]: (3,)

In [64]: np.squeeze(c,axis=2).shape
Out[64]: (1, 3)
```

numpy 的最大最小

```
In [1]: import numpy as np

In [2]: a=np.arange(12).reshape(3,4)

In [3]: a
Out[3]:
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])

In [4]: np.max(a)
Out[4]: 11

In [5]: np.max(a,0)
Out[5]: array([ 8,  9, 10, 11])
```

```
In [5]: np.max(a,0)
Out[5]: array([ 8,  9, 10, 11])

In [6]: b=np.random.randint(10,size=(3,4))

In [7]: b
Out[7]:
array([[7, 2, 1, 7],
       [4, 7, 6, 0],
       [7, 2, 3, 1]])

In [8]: np.maximum(a,b)
Out[8]:
array([[ 7,  2,  2,  7],
       [ 4,  7,  6,  7],
       [ 8,  9, 10, 11]])
```

```
In [13]: np.maximum(a,5)
Out[13]:
array([[ 5,  5,  5,  5],
       [ 5,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

numpy 中的argmax,argmin

```
In [2]: a=np.random.randint(10,size=(3,4,2))

In [3]: a
Out[3]:
array([[[5, 9],
        [7, 7],
        [0, 4],
        [7, 8]],

       [[0, 4],
        [3, 0],
        [1, 4],
        [8, 0]],

       [[5, 2],
        [7, 4],
        [5, 5],
        [1, 9]]])
```

```
In [4]: a.argmax(axis=0)
Out[4]:
array([[0, 0],
       [0, 0],
       [2, 2],
       [1, 2]], dtype=int64)

In [5]: a.argmax(axis=1)
Out[5]:
array([[1, 0],
       [3, 0],
       [1, 3]], dtype=int64)

In [6]: a.argmax(axis=2)
Out[6]:
array([[1, 0, 1, 1],
       [1, 0, 1, 0],
       [0, 0, 0, 1]], dtype=int64)
```

numpy 中的排序

```
In [8]: a
Out[8]:
array([[[5, 9],
        [7, 7],
        [0, 4],
        [7, 8]],

       [[0, 4],
        [0, 3],
        [1, 4],
        [0, 8]],

       [[2, 5],
        [4, 7],
        [5, 5],
        [1, 9]]])
```

```
In [10]: np.sort(a,axis=0)
Out[10]:
array([[[0, 4],
        [0, 3],
        [0, 4],
        [0, 8]],

       [[2, 5],
        [4, 7],
        [1, 4],
        [1, 8]],

       [[5, 9],
        [7, 7],
        [5, 5],
        [7, 9]]])
```

```
In [11]: np.sort(a,axis=1)
Out[11]:
array([[[0, 4],
        [5, 7],
        [7, 8],
        [7, 9]],

       [[0, 3],
        [0, 4],
        [0, 4],
        [1, 8]],

       [[1, 5],
        [2, 5],
        [4, 7],
        [5, 9]]])
```

```
In [12]: np.sort(a,axis=2)
Out[12]:
array([[[5, 9],
        [7, 7],
        [0, 4],
        [7, 8]],

       [[0, 4],
        [0, 3],
        [1, 4],
        [0, 8]],

       [[2, 5],
        [4, 7],
        [5, 5],
        [1, 9]]])
```

numpy 中的排序

```
In [18]: a
Out[18]:
array([[[2, 5],
        [2, 2],
        [6, 6],
        [8, 9]],

       [[3, 5],
        [3, 4],
        [6, 3],
        [3, 3]],

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

```
In [19]: a.argsort(axis=0)
Out[19]:
array([[[0, 2],
        [0, 0],
        [0, 2],
        [1, 1]],

       [[1, 0],
        [1, 1],
        [1, 1],
        [2, 2]],

       [[2, 1],
        [2, 2],
        [2, 0],
        [0, 0]]], dtype=int64)
```

```
In [20]: a.argsort(axis=1)
Out[20]:
array([[[0, 1],
        [1, 0],
        [2, 2],
        [3, 3]],

       [[0, 2],
        [1, 3],
        [3, 1],
        [2, 0]],

       [[3, 2],
        [1, 0],
        [2, 1],
        [0, 3]]], dtype=in
```

```
In [21]: a.argsort(axis=-1)
Out[21]:
array([[[0, 1],
        [0, 1],
        [0, 1],
        [0, 1]],

       [[0, 1],
        [0, 1],
        [1, 0],
        [0, 1]],

       [[1, 0],
        [1, 0],
        [1, 0],
        [0, 1]]], dtype=int64)
```

numpy 的矩阵操作

```
In [18]: mat=np.mat('1,2,3;4,5,6')

In [19]: mat
Out[19]:
matrix([[1, 2, 3],
        [4, 5, 6]])

In [20]: mat.shape
Out[20]: (2, 3)
```

```
In [21]: mat.T
Out[21]:
matrix([[1, 4],
        [2, 5],
        [3, 6]])

In [22]: mat.reshape(3,2)
Out[22]:
matrix([[1, 2],
        [3, 4],
        [5, 6]])
```

```
In [23]: mat2=np.mat('1;2;3')

In [24]: mat2.shape
Out[24]: (3, 1)

In [25]: mat*mat2
Out[25]:
matrix([[14],
        [32]])
```

numpy 的矩阵操作

```
In [30]: a
Out[30]:
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])

In [31]: b
Out[31]:
array([[7, 2, 1],
       [7, 4, 7],
       [6, 0, 7],
       [2, 3, 1]])

In [32]: c=np.matmul(a,b)

In [33]: c
Out[33]:
array([[ 25,  13,  24],
       [113,  49,  88],
       [201,  85, 152]])
```

```
In [36]: a
Out[36]:
array([[[ 0,  1,  2,  3],
        [ 4,  5,  6,  7],
        [ 8,  9, 10, 11]]])

In [37]: a.shape
Out[37]: (1, 3, 4)

In [38]: b
Out[38]:
array([[7, 2, 1],
       [7, 4, 7],
       [6, 0, 7],
       [2, 3, 1]])

In [39]: b.shape
Out[39]: (4, 3)

In [40]: c=np.matmul(a,b)
```

numpy 的拼接

```
In [42]: a = np.arange(3)

In [43]: a
Out[43]: array([0, 1, 2])

In [44]: b = np.arange(10, 13)

In [45]: b
Out[45]: array([10, 11, 12])

In [46]: v = np.vstack((a, b))

In [47]: v
Out[47]:
array([[ 0,  1,  2],
       [10, 11, 12]])
```

```
In [50]: h
Out[50]: array([ 0,  1,  2, 10, 11, 12])

In [52]: c=np.column_stack((a,b))

In [53]: c
Out[53]:
array([[ 0, 10],
       [ 1, 11],
       [ 2, 12]])
```

numpy 的拼接（Concatenate函数）

```
In [54]: a = np.array([[1, 2], [3, 4]])

In [55]: b = np.array([[5, 6]])

In [56]: np.concatenate((a, b), axis=0)
Out[56]:
array([[1, 2],
       [3, 4],
       [5, 6]])

In [57]: np.vstack((a,b))
Out[57]:
array([[1, 2],
       [3, 4],
       [5, 6]])
```

```
In [64]: a = np.array([[[1, 2], [3, 4]],[[1, 2], [3, 4]]])

In [65]: b=np.array([[[1, 2], [3, 4]],[[1, 2], [3, 4]],[[1,
2], [3, 4]]])

In [66]: c=np.concatenate((a,b))

In [67]: c.shape
Out[67]: (5, 2, 2)

In [69]: b=b.reshape(b.shape[1],b.shape[0],b.shape[2])

In [70]: b
Out[70]:
array([[[1, 2],
        [3, 4],
        [1, 2]],

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

In [71]: c=np.concatenate((a,b),axis=1)

In [72]: c.shape
Out[72]: (2, 5, 2)
```

numpy 的拼接

```
In [42]: a = np.arange(3)

In [43]: a
Out[43]: array([0, 1, 2])

In [44]: b = np.arange(10, 13)

In [45]: b
Out[45]: array([10, 11, 12])

In [46]: v = np.vstack((a, b))

In [47]: v
Out[47]:
array([[ 0,  1,  2],
       [10, 11, 12]])
```

```
In [50]: h
Out[50]: array([ 0,  1,  2, 10, 11, 12])

In [52]: c=np.column_stack((a,b))

In [53]: c
Out[53]:
array([[ 0, 10],
       [ 1, 11],
       [ 2, 12]])
```

numpy 的split

```
In [4]: import numpy as np

In [5]: a = np.arange(12).reshape(3, 4)

In [6]: a
Out[6]:
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])

In [7]: np.split(a, 2, axis = 1)
Out[7]:
[array([[0, 1],
        [4, 5],
        [8, 9]]),
 array([[ 2,  3],
        [ 6,  7],
        [10, 11]])]

In [8]: np.split(a, 3, axis = 0)
Out[8]: [array([[0, 1, 2, 3]]), array([[4, 5, 6, 7]]), array([[
8,  9, 10, 11]])]
```

```
In [9]: x = np.arange(9.0)

In [10]: np.split(x, [3, 5, 6, 10])
Out[10]:
[array([0., 1., 2.]),
 array([3., 4.]),
 array([5.]),
 array([6., 7., 8.]),
 array([], dtype=float64)]
```

numpy 与函数的结合

```
In [4]: def fun(x):
   ...:     return math.sqrt(x)
   ...:

In [5]: a=np.array([1,4,9])

In [6]: fun(a)
Traceback (most recent call last):

  File "<ipython-input-6-73c0118913d2>", line 1, in <module>
    fun(a)

  File "<ipython-input-4-eb8670e087f4>", line 2, in fun
    return math.sqrt(x)

TypeError: only size-1 arrays can be converted to Python
scalars
```

```
In [7]: def fun(x):
   ...:     return np.sqrt(x)
   ...:

In [8]: fun(a)
Out[8]: array([1., 2., 3.])
```

numpy 与图像处理

```
In [1]: import cv2

In [2]: im=cv2.imread('Images/000001.jpg')

In [3]: type(im)
Out[3]: numpy.ndarray

In [4]: im.shape
Out[4]: (500, 353, 3)
```