Numpy 基础

```
import numpy as np
```

```
np.set_printoptions(precision=4)
```

```
my_list = list(range(10000000)) # python列表
my_arr = np.arange(10000000) # numpy ndarray
```

```
%%time
for _ in range(10):
    my_list2 = [x*2 for x in my_list]
```

```
Wall time: 14 s
```

```
%%time
for _ in range(10):
    my_arr2 = my_arr * 2
```

```
Wall time: 352 ms
```

1. Numpy的ndarray:多维数组

```
data = np.random.randn(2,3)
data
```

```
array([[-0.89741638,  0.23687626, -0.07568737],
       [-1.13064414,  0.67084946, -0.22382149]])
```

```
data * 10
```

```
array([[ -8.97416378,   2.36876256,  -0.75687366],
       [-11.30644135,   6.70849459,  -2.23821495]])
```

```
data + data
```

```
array([[-1.79483276,  0.47375251, -0.15137473],
       [-2.26128827,  1.34169892, -0.44764299]])
```

```
data.shape
```

```
(2, 3)
```

```
data.dtype
```

```
dtype('float64')
```

1.1 创建ndarray

```
data1 = [6,7.5,8,0.1]
arr1 = np.array(data1)
arr1
```

```
array([6. , 7.5, 8. , 0.1])
```

```
data2 = [[1,2,3,4],[5,6,7,8]]
arr2 = np.array(data2)
arr2
```

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

```
arr2.shape
```

```
(2, 4)
```

```
arr2.ndim
```

```
2
```

```
arr2.dtype
```

```
dtype('int32')
```

```
np.zeros(10)
```

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

```
np.ones((3,5) )
```

```
array([[1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.]])
```

```
np.empty((3,6,2))
```

```
array([[[ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000, -2.01007741e-310]],

       [[ 0.00000000e+000,  0.00000000e+000],
```

```
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000]],

       [[ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000],
        [ 0.00000000e+000,  0.00000000e+000]]])
```

```
np.arange(15).dtype
```

```
dtype('int32')
```

1.2 ndarray 的数据类型

```
arr1 = np.array([1,2,3],dtype=np.float64)
arr2 = np.array([1,2,3],dtype=np.int32)
```

```
arr1.dtype
```

```
dtype('float64')
```

```
arr2.dtype
```

```
dtype('int32')
```

```
arr1 = np.array([1,2,3.])
```

```
arr1.dtype
```

```
dtype('float64')
```

1.3 Numpy数组的运算

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

```
array([[1., 2., 3.],
       [4., 5., 6.]])
```

```
arr * arr
```

```
array([[ 1.,  4.,  9.],
       [16., 25., 36.]])
```

```
arr - arr
```

```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

```
1/arr
```

```
array([[1.        , 0.5       , 0.33333333],
       [0.25      , 0.2       , 0.16666667]])
```

```
arr ** 0.5
```

```
array([[1.        , 1.41421356, 1.73205081],
       [2.        , 2.23606798, 2.44948974]])
```

```
arr2 = np.array([[0,4,1],[7,2,12.]])
```

```
arr2
```

```
array([[ 0.,  4.,  1.],
       [ 7.,  2., 12.]])
```

```
arr2 > arr1
```

```
array([[False,  True, False],
       [ True, False,  True]])
```

```
(arr2 > arr1).dtype
```

```
dtype('bool')
```

1.3.1 基本的索引和切片

```
arr = np.arange(10)
arr
```

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

```
arr[5]
```

```
5
```

```
arr[5:8]
```

```
array([5, 6, 7])
```

```
arr[5:8] = 12
```

```
arr
```

```
array([ 0,  1,  2,  3,  4, 12, 12, 12,  8,  9])
```

```
arr_slice = arr[5:8]
arr_slice
```

```
array([12, 12, 12])
```

```
arr_slice[2]=12345
```

```
arr_slice
```

```
array([   12,    12, 12345])
```

```
arr
```

```
array([    0,     1,     2,     3,     4,    12,    12, 12345,     8,
           9])
```

```
_arr_slice = arr[1:3].copy()
_arr_slice
```

```
array([1, 2])
```

```
_arr_slice[1] = 12345
_arr_slice
```

```
array([    1, 12345])
```

```
arr
```

```
array([    0,     1,     2,     3,     4,    12,    12, 12345,     8,
           9])
```

```
arr2d = np.array([[1,2,3],[4,5,6],[7,8,9]])
arr2d
```

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

```
arr2d[2]
```

```
array([7, 8, 9])
```

```
arr2d[0][2]
```

```
3
```

```
arr2d[0,2]
```

```
3
```

1.3.2 切片索引

```
arr2d
```

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

```
arr2d[:2]
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

```
arr2d[:2,:]
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

```
arr2d[:2,1:]
```

```
array([[2, 3],
       [5, 6]])
```

```
arr2d[1,:2]
```

```
array([4, 5])
```

```
arr2d[:,:1]
```

```
array([[1],
       [4],
       [7]])
```

```
arr2d[:2,1:] = 0
```

```
arr2d
```

```
array([[1, 0, 0],
       [4, 0, 0],
       [7, 8, 9]])
```

1.3.3 布尔型索引

```
names = np.array(['王大锤','赵铁棍','尼古拉斯','王大锤','尼古拉斯','赵铁棍','赵铁棍'])
```

```
data = np.random.randn(7,4)
data
```

```
array([[-1.70372314, -0.79805994, -0.62431594, -0.56289262],
       [ 0.746297  ,  0.005503  , -0.13108016,  0.24249904],
       [-0.99479306, -2.88504282,  1.11190286, -0.35070307],
       [-1.245134  , -1.51065238, -0.24547209, -0.69549517],
       [ 1.08401323,  0.82411223, -0.91761727,  0.20578439],
       [ 0.41281562,  0.18059028, -1.13506974, -0.71858969],
       [-0.48698305,  0.55083981,  1.07509611, -0.30973693]])
```

```
names == '王大锤'
```

```
array([ True, False, False,  True, False, False, False])
```

```
data[names=='王大锤']  # data[[ True, False, False,  True, False, False, False]]
```

```
array([[-1.70372314, -0.79805994, -0.62431594, -0.56289262],
       [-1.245134  , -1.51065238, -0.24547209, -0.69549517]])
```

```
data[names=='王大锤',2:]
```

```
array([[-0.62431594, -0.56289262],
       [-0.24547209, -0.69549517]])
```

```
data[names=='王大锤',3]
```

```
array([-0.56289262, -0.69549517])
```

```
names != '王大锤'
```

```
array([False,  True,  True, False,  True,  True,  True])
```

```
data[~(names!='王大锤')] # & and , \ or , ~  not
```

```
array([[-1.70372314, -0.79805994, -0.62431594, -0.56289262],
       [-1.245134  , -1.51065238, -0.24547209, -0.69549517]])
```

```
cond = (names=='王大锤')
data[~cond]
```

```
array([[ 0.746297  ,  0.005503  , -0.13108016,  0.24249904],
       [-0.99479306, -2.88504282,  1.11190286, -0.35070307],
       [ 1.08401323,  0.82411223, -0.91761727,  0.20578439],
       [ 0.41281562,  0.18059028, -1.13506974, -0.71858969],
       [-0.48698305,  0.55083981,  1.07509611, -0.30973693]])
```

```
mask = (names =='王大锤')| (names=='尼古拉斯')
mask
```

```
array([ True, False,  True,  True,  True, False, False])
```

```
data[~mask]
```

```
array([[ 0.746297  ,  0.005503  , -0.13108016,  0.24249904],
       [ 0.41281562,  0.18059028, -1.13506974, -0.71858969],
       [-0.48698305,  0.55083981,  1.07509611, -0.30973693]])
```

```
data
```

```
array([[-1.70372314, -0.79805994, -0.62431594, -0.56289262],
       [ 0.746297  ,  0.005503  , -0.13108016,  0.24249904],
       [-0.99479306, -2.88504282,  1.11190286, -0.35070307],
       [-1.245134  , -1.51065238, -0.24547209, -0.69549517],
       [ 1.08401323,  0.82411223, -0.91761727,  0.20578439],
       [ 0.41281562,  0.18059028, -1.13506974, -0.71858969],
       [-0.48698305,  0.55083981,  1.07509611, -0.30973693]])
```

```
data[data<0]
```

```
array([-1.70372314, -0.79805994, -0.62431594, -0.56289262, -0.13108016,
       -0.99479306, -2.88504282, -0.35070307, -1.245134  , -1.51065238,
       -0.24547209, -0.69549517, -0.91761727, -1.13506974, -0.71858969,
       -0.48698305, -0.30973693])
```

```
data < 0
```

```
array([[ True,  True,  True,  True],
       [False, False,  True, False],
       [ True,  True, False,  True],
       [ True,  True,  True,  True],
       [False, False,  True, False],
       [False, False,  True,  True],
       [ True, False, False,  True]])
```

```
data[data<0]=0
```

```
data
```

```
array([[0.        , 0.        , 0.        , 0.        ],
       [0.746297  , 0.005503  , 0.        , 0.24249904],
       [0.        , 0.        , 1.11190286, 0.        ],
       [0.        , 0.        , 0.        , 0.        ],
       [1.08401323, 0.82411223, 0.        , 0.20578439],
       [0.41281562, 0.18059028, 0.        , 0.        ],
       [0.        , 0.55083981, 1.07509611, 0.        ]])
```

```
data[names!='赵铁棍'] = 7
```

```
names!='赵铁棍'
```

```
array([ True, False,  True,  True,  True, False, False])
```

```
data
```

```
array([[7.0000e+00, 7.0000e+00, 7.0000e+00, 7.0000e+00],
       [7.4630e-01, 5.5030e-03, 0.0000e+00, 2.4250e-01],
       [7.0000e+00, 7.0000e+00, 7.0000e+00, 7.0000e+00],
       [7.0000e+00, 7.0000e+00, 7.0000e+00, 7.0000e+00],
       [7.0000e+00, 7.0000e+00, 7.0000e+00, 7.0000e+00],
       [4.1282e-01, 1.8059e-01, 0.0000e+00, 0.0000e+00],
       [0.0000e+00, 5.5084e-01, 1.0751e+00, 0.0000e+00]])
```

1.3.4 花式索引

```
arr = np.empty((8,4)) # zeros
arr
```

```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

```
for i in range(8):
    arr[i] = i
arr
```

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

```
arr[[4,3,0,6]]
```

```
array([[4., 4., 4., 4.],
       [3., 3., 3., 3.],
       [0., 0., 0., 0.],
       [6., 6., 6., 6.]])
```

```
arr[[-3,-5,-7]]
```

```
array([[5., 5., 5., 5.],
       [3., 3., 3., 3.],
       [1., 1., 1., 1.]])
```

```
arr = np.arange(32).reshape(8,4)
arr
```

```
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],
       [24, 25, 26, 27],
       [28, 29, 30, 31]])
```

```
arr[[1,5,7,2],[0,3,1,2]]  # [1,0],[5,3],[7,1],[2,2]  # arr[]
```

```
array([ 4, 23, 29, 10])
```

```
arr[[1,5,7,2]][:]  # arr[][]
```

```
array([[ 4,  5,  6,  7],
       [20, 21, 22, 23],
       [28, 29, 30, 31],
       [ 8,  9, 10, 11]])
```

```
arr[[1, 5, 7, 2]][:, [0, 3, 1, 2]]  # 1572行，0312列
```

```
array([[ 4,  7,  5,  6],
       [20, 23, 21, 22],
       [28, 31, 29, 30],
       [ 8, 11,  9, 10]])
```

1.3.5 数组转置

```
arr = np.arange(15).reshape(3,5)
arr
```

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

```
arr.T # 转置
```

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

```
np.dot(arr,arr.T) # 矩阵乘
```

```
array([[ 30,  80, 130],
       [ 80, 255, 430],
       [130, 430, 730]])
```

```
arr.dot(arr.T)
```

```
array([[ 30,  80, 130],
       [ 80, 255, 430],
       [130, 430, 730]])
```

```
arr@arr.T
```

```
array([[ 30,  80, 130],
       [ 80, 255, 430],
       [130, 430, 730]])
```

```
arr
```

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

```
arr.swapaxes(1,0)
```

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

2 通用函数 ufunc

```
arr = np.arange(10)
arr
```

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

```
np.sqrt(arr)
```

```
array([0.    , 1.    , 1.4142, 1.7321, 2.    , 2.2361, 2.4495, 2.6458,
       2.8284, 3.    ])
```

```
np.exp(arr)
```

```
array([1.0000e+00, 2.7183e+00, 7.3891e+00, 2.0086e+01, 5.4598e+01,
       1.4841e+02, 4.0343e+02, 1.0966e+03, 2.9810e+03, 8.1031e+03])
```

```
x = np.random.randn(8)
```

```
x
```

```
array([-0.2153,  0.783 , -0.674 ,  0.9593, -0.1803, -0.1002,  0.4618,
        0.3593])
```

```
y = np.random.randn(8)
y
```

```
array([-0.0534,  0.6388,  0.8168,  0.1522, -0.4421, -1.5189,  0.0968,
       -0.1472])
```

```
np.maximum(x,y)
```

```
array([-0.0534,  0.783 ,  0.8168,  0.9593, -0.1803, -0.1002,  0.4618,
        0.3593])
```

```
arr = np.random.randn(6) * 5
```

```
arr
```

```
array([ 5.8542,  7.8812, -1.6873,  0.5241,  8.0797,  1.057 ])
```

```
full,part = np.modf(arr)
```

```
full
```

```
array([ 0.8542,  0.8812, -0.6873,  0.5241,  0.0797,  0.057 ])
```

```
part
```

```
array([ 5.,  7., -1.,  0.,  8.,  1.])
```

3 利用数组进行数据处理

3.1 将条件逻辑表述为数组运算

```
xarr = np.array([1.1,1.2,1.3,1.4,1.5])
yarr = np.array([2.1,2.2,2.3,2.4,2.5])
cond = np.array([True,False,True,True,False])
```

```
list(zip(xarr, yarr, cond))
```

```
[(1.1, 2.1, True),
 (1.2, 2.2, False),
 (1.3, 2.3, True),
 (1.4, 2.4, True),
 (1.5, 2.5, False)]
```

```
result = [(x if c else y) for x, y, c in zip(xarr, yarr, cond)]
result
```

```
[1.1, 2.2, 1.3, 1.4, 2.5]
```

```
result = np.where(cond,xarr,yarr)
result
```

```
array([1.1, 2.2, 1.3, 1.4, 2.5])
```

```
arr = np.random.randn(4,4)
arr
```

```
array([[ 0.7921,  0.2906,  0.5878,  0.4929],
       [-0.6937,  0.7684, -0.3898, -0.0877],
       [-1.0897, -0.3084, -0.0351, -0.4095],
       [-0.8873,  2.4287, -0.259 ,  1.9385]])
```

```
arr > 0
```

```
array([[ True,  True,  True,  True],
       [False,  True, False, False],
       [False, False, False, False],
       [False,  True, False,  True]])
```

```
np.where(arr>0,2,-2)
```

```
array([[ 2,  2,  2,  2],
       [-2,  2, -2, -2],
       [-2, -2, -2, -2],
       [-2,  2, -2,  2]])
```

```
np.where(arr > 0, 2, arr)
```

```
array([[ 2.    ,  2.    ,  2.    ,  2.    ],
       [-0.6937,  2.    , -0.3898, -0.0877],
       [-1.0897, -0.3084, -0.0351, -0.4095],
       [-0.8873,  2.    , -0.259 ,  2.    ]])
```

3.2 数学和统计的方法

```
arr = np.random.randn(5,4)
arr
```

```
array([[ 1.6356, -0.1086,  0.6767,  0.4326],
       [ 0.5958, -0.4759, -0.3581, -1.2847],
       [ 0.0377, -0.2427,  0.1643,  0.8455],
       [ 0.3261,  0.8959,  0.1746, -0.4852],
       [ 0.727 ,  0.9231, -1.0086, -0.1029]])
```

```
arr.mean()
```

```
0.1684117990449233
```

```
np.mean(arr)
```

```
0.1684117990449233
```

```
arr.sum()
```

```
3.368235980898466
```

```
arr.mean(axis=1) # 指定轴 0 - 列，1 - 行。按行求平均值
```

```
array([ 0.6591, -0.3807,  0.2012,  0.2278,  0.1346])
```

```
arr.sum(axis=0) # 按列求总计
```

```
array([ 3.3222,  0.9918, -0.351 , -0.5947])
```

```
# sum 求和；mean 均值；std/var 标准差/方差；
# min/max 最小值/最大值  cumsum累加 cumprod 累乘
```

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

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

```
arr.cumsum(axis=0)
```

```
array([[ 0,  1,  2],
       [ 3,  5,  7],
       [ 9, 12, 15],
       [18, 22, 26]], dtype=int32)
```

```
arr.cumprod(axis=1)
```

```
array([[  0,   0,   0],
       [  3,  12,  60],
       [  6,  42, 336],
       [  9,  90, 990]], dtype=int32)
```

```
arr.cumsum()
```

```
array([ 0,  1,  3,  6, 10, 15, 21, 28, 36, 45, 55, 66], dtype=int32)
```

```
```