数组属性

NumPy 数组的维数称为秩（rank），秩就是轴的数量，即数组的维度，一维数组的秩为 1，二维数组的秩为 2，以此类推。

在 NumPy中，每一个线性的数组称为是一个轴（axis），也就是维度（dimensions）。比如说，二维数组相当于是两个一维数组，其中第一个一维数组中每个元素又是一个一维数组。所以一维数组就是 NumPy 中的轴（axis），第一个轴相当于是底层数组，第二个轴是底层数组里的数组。而轴的数量——秩，就是数组的维数。

很多时候可以声明 axis。axis=0，表示沿着第 0 轴进行操作，即对每一列进行操作；axis=1，表示沿着第1轴进行操作，即对每一行进行操作。

NumPy 的数组中比较重要 ndarray 对象属性有：

属性	说明
ndarray.ndim	秩，即轴的数量或维度的数量
ndarray.shape	数组的维度，对于矩阵，n 行 m 列
ndarray.size	数组元素的总个数，相当于 .shape 中 n*m 的值
ndarray.dtype	ndarray 对象的元素类型
ndarray.itemsize	ndarray 对象中每个元素的大小，以字节为单位
ndarray.flags	ndarray 对象的内存信息
ndarray.real	ndarray元素的实部
ndarray.imag	ndarray 元素的虚部
ndarray.data	包含实际数组元素的缓冲区，由于一般通过数组的索引获取元素，所以通常不需要使用这个属性。

dim、shape和reshape

```python
import numpy as np

a = np.arange(24)

print(a)
print(a.ndim)
```

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

```python
b = a.reshape(2, 3, 4)
print(b)
print(b.ndim)

print(b.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]]]
3
(2, 3, 4)
```

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

print('-' * 20)

a.shape = (3,2)
print(a)
```

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

dtype和itemsize

```
a1 = np.array([1,2,3,4,5], dtype=int)
print(a1.dtype)
print(a1.itemsize)

a2 = np.array([1,2,3,4,5], dtype=float)
print(a2.dtype)
print(a2.itemsize)
```

```
int32
4
float64
8
```

创建数组

numpy.empty

numpy.empty 方法用来创建一个指定形状（shape）、数据类型（dtype）且未初始化的数组：

```
numpy.empty(shape, dtype = float, order = 'C')
```

参数	描述
shape	数组形状
dtype	数据类型，可选
order	有"C"和"F"两个选项,分别代表，行优先和列优先，在计算机内存中的存储元素的顺序。

```
a = np.empty((3, 2), dtype='i4')
print(a)
```

```
[[2036429426 1701602592]
 [1953391981  168439411]
 [ 538976288 1634885968]]
```

numpy.zeros

创建指定大小的数组，数组元素以 0 来填充

```
numpy.zeros(shape, dtype = float, order = 'C')
```

参数	描述
shape	数组形状
dtype	数据类型，可选
order	'C' 用于 C 的行数组，或者 'F' 用于 FORTRAN 的列数组

```
a = np.zeros(5)
print(a)
```

```
[0.  0.  0.  0.  0.]
```

```
a = np.zeros((3, 5), dtype=int)
print(a)
```

```
[[0 0 0 0 0]
 [0 0 0 0 0]
 [0 0 0 0 0]]
```

```
a = np.zeros((3, 2), dtype=[('x', 'f4'), ('y', 'i4')])
print(a)

print('-' * 20)

a[0][0]['x'] = 1.23
a[0][0]['y'] = 1.23

print(a)
```

```
[[(0., 0) (0., 0)]
 [(0., 0) (0., 0)]
 [(0., 0) (0., 0)]]
--------------------
[[(1.23, 1) (0.  , 0)]
 [(0.  , 0) (0.  , 0)]
 [(0.  , 0) (0.  , 0)]]
```

numpy.ones

创建指定形状的数组，数组元素以 1 来填充

```
numpy.ones(shape, dtype = None, order = 'C')
```

参数	描述
shape	数组形状
dtype	数据类型，可选
order	'C' 用于 C 的行数组，或者 'F' 用于 FORTRAN 的列数组

```
a = np.ones(5)
print(a)
print(a.dtype)
```

```
[1. 1. 1. 1. 1.]
float64
```

```
a = np.ones((3, 5), dtype=int)
print(a)
```

```
[[1 1 1 1 1]
 [1 1 1 1 1]
 [1 1 1 1 1]]
```

numpy.full

创建指定形状的数组，数组元素以 fill_value 来填充

```
numpy.full(shape, fill_value, dtype=None, ordr='C')
```

参数	说明
shape	数组形状
fill_value	填充的数据
dtype	数据类型，可选
order	'C' 用于 C 的行数组，或者 'F' 用于 FORTRAN 的列数组

```
a = np.full(5, fill_value=999)
print(a)
print(a.dtype)
```

```
[999 999 999 999 999]
int32
```

numpy.eye

对角线为1其他的位置为0

参数	说明
N	行数量
M	列数量，默认等于行数量，可选
dtype	数据类型，可选
order	'C' 用于 C 的行数组，或者 'F' 用于 FORTRAN 的列数组

```
a = np.eye(10, dtype=int)
print(a)
```

```
[[1 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0 1]]
```

从已有数组创建数组

numpy.asarray

numpy.asarray 类似 numpy.array

参数	描述
a	任意形式的输入参数，可以是，列表, 列表的元组, 元组, 元组的元组, 元组的列表，多维数组
dtype	数据类型，可选
order	可选，有"C"和"F"两个选项,分别代表，行优先和列优先，在计算机内存中的存储元素的顺序。

```
data = [[1,2,3],[4,5,6],[7,8,9]]

a = np.array(data, dtype=int)
b = np.asarray(data, dtype=int)
print(a)
print(b)
```

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

```
import numpy as np

a = np.array([1,2,3,4,5,6], dtype=int)
b = np.array(a)
print(a)
print(b)

b[0] = 8

print(a)
print(b)
```

```
[1 2 3 4 5 6]
[1 2 3 4 5 6]
[1 2 3 4 5 6]
[8 2 3 4 5 6]
```

```
a = np.array([1,2,3,4,5,6], dtype=int)
b = np.asarray(a)
print(a)
print(b)

b[0] = 8

print(a)
print(b)
```

```
[1 2 3 4 5 6]
[1 2 3 4 5 6]
[8 2 3 4 5 6]
[8 2 3 4 5 6]
```

numpy.frombuffer

numpy.frombuffer 用于实现动态数组。

numpy.frombuffer 接受 buffer 输入参数，以流的形式读入转化成 ndarray 对象。

```
numpy.frombuffer(buffer, dtype = float, count = -1, offset = 0)
```

参数	描述
buffer	可以是任意对象，会以流的形式读入。
dtype	返回数组的数据类型，可选
count	读取的数据数量，默认为-1，读取所有数据。
offset	读取的起始位置，默认为0。

```
s =  b'https://www.hioier.com/'
a = np.frombuffer(s, dtype =  'S1', count=10, offset=5)

print (a)
```

```
[b':' b'/' b'/' b'w' b'w' b'w' b'.' b'h' b'i' b'o']
```

numpy.fromiter

numpy.fromiter 方法从可迭代对象中建立 ndarray 对象，返回一维数组。

```
numpy.fromiter(iterable, dtype, count=-1)
```

参数	描述
iterable	可迭代对象
dtype	返回数组的数据类型
count	读取的数据数量，默认为-1，读取所有数据

```
it = range(10)

a = np.fromiter(it, dtype=np.int_)
print(a)
print(a.dtype)
```

```
[0 1 2 3 4 5 6 7 8 9]
int32
```

从数值范围创建数组

numpy.arange

numpy 包中的使用 arange 函数创建数值范围并返回 ndarray 对象

```
numpy.arange(start, stop, step, dtype)
```

参数	描述
`start`	起始值，默认为 `0`
`stop`	终止值（不包含）
`step`	步长，默认为 `1`
`dtype`	返回 `ndarray` 的数据类型，如果没有提供，则会使用输入数据的类型。

```
a = np.arange(1, 20, 3)
print(a)
```

```
[ 1  4  7 10 13 16 19]
```

numpy.linspace

numpy.linspace 函数用于创建一个一维数组，数组是一个等差数列构成的，格式如下：

```
np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)
```

参数	描述
`start`	序列的起始值
`stop`	序列的终止值，如果 `endpoint` 为 `true`，该值包含于数列中
`num`	要生成的等步长的样本数量，默认为 `50`
`endpoint`	该值为 `true` 时，数列中包含 `stop` 值，反之不包含，默认是True。
`retstep`	如果为 True 时，生成的数组中会显示间距，反之不显示。
`dtype`	`ndarray` 的数据类型

```
a = np.linspace(10, 20, 10, endpoint=False)
print(a)
```

```
[10. 11. 12. 13. 14. 15. 16. 17. 18. 19.]
```

numpy.logspace

numpy.logspace 函数用于创建一个于等比数列。格式如下：

```
np.logspace(start, stop, num=50, endpoint=True, base=10.0, dtype=None)
```

base 参数意思是取对数的时候 log 的下标。

参数	描述
`start`	序列的起始值为：base ** start
`stop`	序列的终止值为：base ** stop。如果`endpoint`为`true`，该值包含于数列中
`num`	要生成的等步长的样本数量，默认为`50`
`endpoint`	该值为`true`时，数列中中包含`stop`值，反之不包含，默认是True。
`base`	对数 log 的底数。
`dtype`	`ndarray`的数据类型

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

a = np.logspace(1.0, 2.0, num=10)
print(a)
```

```
[ 10.          12.91549665  16.68100537  21.5443469   27.82559402
  35.93813664  46.41588834  59.94842503  77.42636827 100.        ]
```

numpy.random.rand()

原型：rand(d0,d1,...,dn$\hat{0}$,$\hat{1}$,...,$\hat{\hat{}}$)

作用：生成[0, 1)之间的随机数

In [3]:
```
a = np.random.rand()
print(a)

a = np.random.rand(3, 2)
print(a)
```

```
0.4306010087386921
[[0.1176491  0.34751026]
 [0.41961551 0.25187987]
 [0.19071042 0.88750756]]
```

numpy.random.random()

原型：`random(size=None)`

作用：生成[0, 1)之间的随机数

In [5]:
```
a = np.random.random()
print(a)

a = np.random.random(3)
print(a)
```

```
0.6531637256985003
[0.80633677 0.84124032 0.86596112]
```

numpy.random.randint()

原型：`randint(low, high=None, size=None, dtype='1')`

作用：生成随机数

参数	说明
low	包含的下限
high	不包含的上限
size	元素个数
dtype	元素类型

In [16]:
```
a = np.random.randint(1, 10, 5, dtype='i4')
print(a)

a = np.random.randint(1000000000, 10000000000, 5, dtype='i8')
print(a)
```

```
[4 2 4 9 1]
[1108719844 3671077639 1060608534 1645368802 9098830823]
```

numpy.random.randn()

原型：randn(d0,d1,...,dn`0,`1,...,``)

作用：返回一个或一组样本，具有标准正态分布

标准正态分布：又称为u分布，是以0为均值、以1为标准差的正态分布，记为N(0,1)

In [18]:
```
a = np.random.randn(3)
print(a)
```

```
[ 1.74820556 -1.18356502 -0.56816756]
```

numpy.random.normal()

原型：`normal(loc=0.0, scale=1.0, size=None)`

作用：生成高斯分布的概率密度随机数

参数	说明
loc	浮点型，此概率分布的均值（对应着整个分布的中心centre）
scale	浮点型，此概率分布的标准差（对应于分布的宽度，scale越大越矮胖，scale越小，越瘦高）
size	输出的shape，默认为None，只输出一个值

In [20]:
```
a = np.random.normal(1, 2, 5)
print(a)
```

```
[ 2.8374781   0.89452405 -1.51077621  0.6160877  -2.37839277]
```

numpy数组与Python中列表的对比

In [1]:
```
import numpy as np
import time
import random
import matplotlib.pyplot as plt
import cv2
```

In [5]:
```
li = []
for i in range(10000000):
    li.append(random.random())
```

In [6]:
```
t1 = time.time()
s = sum(li)
t2 = time.time()
print("list耗时：%f" % (t2 - t1))
```

```
list耗时：0.049900
```

In [7]:
```
a = np.array(li)
t1 = time.time()
s = np.sum(a)
t2 = time.time()
print("numpy耗时：%f" % (t2 - t1))
```

```
numpy耗时：0.012016
```