



Python 培训第三期

N u m p y 初 步

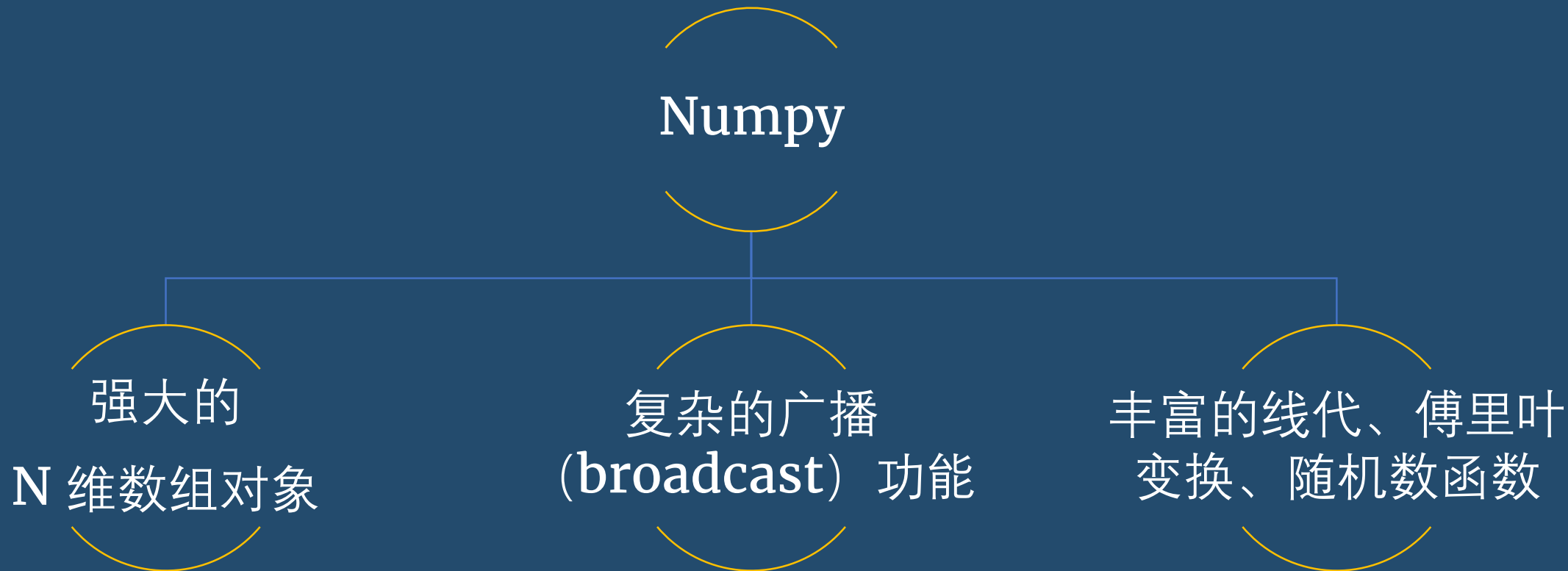
By 李亚军

CONTENT

- Intro and Installation
- The Basics
- Array Creation
- Basic Operations
- Indexing, Slicing and Iterating
- Shape Manipulation
- Copies and Views

Intro and Installation

Numpy 是 Python 中用于科学计算的基础包。



Intro and Installation

- 和其他第三方包安装方式一样，可以采用 **conda** 或者 **pip** 方式安装。
- **conda** 方式

```
$ conda install numpy
```

- **pip** 方式

```
$ pip install numpy
```

The Basics

- **Numpy** 中主要的对象就是 **ndarray** 对象，其中的元素都具有相同类型，用从 0 开始的下标来索引。在 **numpy** 中，**dimensions** 被叫做 *axes*，**axes** 的数量叫做 *rank*。

<code>ndarray.ndim</code>	axes (dimensions) 的个数，在 Python 里，就是指 <i>rank</i>
---------------------------	--

<code>ndarray.shape</code>	array 的维度， shape 的长度就是 rank 或者 ndim
----------------------------	---

<code>ndarray.size</code>	array 里元素的个数，等于 shape 中的元素的乘积
---------------------------	---

<code>ndarray.dtype</code>	array 里元素的类型。例如 <code>numpy.int16</code> ， <code>numpy.float32</code> 等
----------------------------	--

The Basics

array	number of axes	rank	ndim	shape	size
<code>[1, 2, 1]</code>	1	1	1	<code>(3,)</code>	3
<code>[[1., 0., 0.], [0., 1., 2.]]</code>	2	2	2	<code>(2, 3)</code>	6

Array Creation – list 转成 array

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

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

```
In [3]: a
```

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

```
In [4]: a.dtype
```

```
Out[4]: dtype('int32')
```

```
In [5]: b = np.array([1.2, 3.5, 5.1])
```

```
In [6]: b.dtype
```

```
Out[6]: dtype('float64')
```

```
In [7]: c = np.array([(1.5,2,3), (4,5,6)])
```

```
In [8]: c
```

```
Out[8]: array([[ 1.5,  2. ,  3. ], [ 4. ,  5. ,  6. ]])
```

Array Creation – 创建特定数组

```
In [9]: np.zeros( (3,4) )
Out[9]: array([[ 0.,  0.,  0.,  0.],
               [ 0.,  0.,  0.,  0.],
               [ 0.,  0.,  0.,  0.]])

In [10]: np.ones( (2,3,4), dtype=np.int16 )
Out[10]: 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]]], dtype=int16)

In [11]: np.empty( (2,3) )
Out[11]: array([[ 8.78509627e-315,  8.78509643e-315,  8.78430964e-315],
               [ 1.54766813e-311,  1.54766515e-311,  1.54766515e-311]])

In [12]: np.arange( 10, 30, 5 )
Out[12]: array([10, 15, 20, 25])

In [13]: np.linspace( 0, 2, 9 )
Out[13]: array([ 0. ,  0.25,  0.5 ,  0.75,  1. ,  1.25,  1.5 ,  1.75,  2. ])
```


Basic Operations – * VS dot

* 表示点乘，即对应元素相乘，而 dot 是真正的矩阵乘法。

```
In [22]: A = np.array( [[1,1], [0,1]] )
```

```
In [23]: B = np.array( [[2,0], [3,4]] )
```

```
In [24]: A * B
```

```
Out[24]: array([[2, 0], [0, 4]])
```

```
In [25]: A.dot(B)
```

```
Out[25]: array([[5, 4], [3, 4]])
```

Basic Operations

下面列出 `numpy` 中一些常用的方法。

方法	说明
<code>np.sum()</code>	求和
<code>np.min()/max()</code>	最大/最小值
<code>np.argmax()/argmin()</code>	最大/最小值的所在位置
<code>np.sqrt()</code>	2次根
<code>np.exp()</code>	指数
<code>np.all()</code>	判断数组中的元素是否全部为 True

Indexing, Slicing and Iterating

对于一维数组

```
In [26]: a = np.arange(10)**3
In [27]: a
Out[27]: array([ 0,  1,  8, 27, 64, 125, 216, 343, 512, 729], dtype=int32)
In [28]: a[2]
Out[28]: 8
In [29]: a[2:5]
Out[29]: array([ 8, 27, 64], dtype=int32)
In [30]: a[:6:2] = -1000
In [31]: a
Out[31]: array([-1000,  1, -1000, 27, -1000, 125, 216, 343, 512, 729], dtype=int32)
In [32]: a[ : :-1]
Out[32]: array([ 729, 512, 343, 216, 125, -1000, 27, -1000, 1, -1000], dtype=int32)
In [33]: a
Out[33]: array([-1000,  1, -1000, 27, -1000, 125, 216, 343, 512, 729], dtype=int32)
```

Indexing, Slicing and Iterating

```
In [40]: b = np.arange(15).reshape(3, 5)
In [41]: b
Out[41]: array([[ 0,  1,  2,  3,  4], [ 5,  6,  7,  8,  9], [10, 11, 12, 13, 14]])
In [42]: b[2,3]
Out[42]: 13
In [43]: b[0:3, 1]
Out[43]: array([ 1,  6, 11])
In [44]: b[1:3, :]
Out[44]: array([[ 5,  6,  7,  8,  9], [10, 11, 12, 13, 14]])
In [45]: b[-1]
Out[45]: array([10, 11, 12, 13, 14])
In [46]: for row in b:
.....:     print(row)
.....:
[0 1 2 3 4]
[5 6 7 8 9]
[10 11 12 13 14]
```

Shape Manipulation

- 数组的 shape 并不是一成不变的，可以使用一些方法来改变。

方法

说明

`np.reshape()`

改变数组的 shape，并返回改变后的结果

`np.resize()`

直接在原数组上改变 shape

`np.ravel()`

拉平数组

`np.vstack()`

在垂直方向上拼接两个数组

`np.hstack()`

在水平方向上拼接两个数组

Copies and Views

- Numpy 关于 copy 有三种情况：完全不复制、视图 (*view*) 或者叫 浅复制 (*shadow copy*) 和 深复制 (*deep copy*)。
- 完全不复制

```
In [58]: a = np.arange(12)
In [59]: b = a
In [60]: b is a
Out[60]: True
In [61]: b.shape = 3, 4
In [62]: a.shape
Out[62]: (3, 4)
```

Copies and Views – View

```
In [63]: s = a[ : , 1:3]
```

```
In [65]: s[:] = 10
```

```
In [66]: a
```

```
Out[66]:
```

```
array([[ 0, 10, 10, 3],  
       [ 4, 10, 10, 7],  
       [ 8, 10, 10, 11]])
```

```
In [67]: s.flags.owndata
```

```
Out[67]: False
```

```
In [68]: a.flags.owndata
```

```
Out[68]: True
```

```
In [69]: s is a
```

```
Out[69]: False
```

Copies and Views – Deep Copy

```
In [72]: d = a.copy()
In [73]: d is a
Out[73]: False
In [74]: d.base is a
Out[74]: False
In [75]: d[0, 0] = 9999
In [76]: a
Out[76]:
array([[ 0, 10, 10, 3],
       [ 4, 10, 10, 7],
       [ 8, 10, 10, 11]])
In [77]: d
Out[77]:
array([[9999, 10, 10, 3],
       [ 4, 10, 10, 7],
       [ 8, 10, 10, 11]])
```


THANK YOU
FOR LISTENING

