链滴

Python 数据科学 (1)——NumPy(1.Arrays 基础篇)

作者：hsxian

原文链接：https://ld246.com/article/1555740478053

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

Python数据科学中有两个很重要的扩展程序库，numpy和pandas。本文将首先介绍numpy的基本用，本文假设你已经具有了一定的python基础，故而不会特别介绍python的语法。

NumPy(Numerical Python) 是 Python 语言的一个扩展程序库，支持大量的维度数组与矩阵运算，外也针对数组运算提供大量的数学函数库。这种工具可用来存储和处理大型矩阵，比Python自身的套列表（nested list structure)结构要高效的多（该结构也可以用来表示矩阵（matrix））。numpy含以下特性：

1. 一个强大的N维数组对象Array；
2. 比较成熟的（广播）函数库；
3. 用于整合C/C++和Fortran代码的工具包；
4. 实用的线性代数、傅里叶变换和随机数生成函数。

numpy和稀疏矩阵运算包scipy配合使用更加方便。

在你的计算机上安装最简单的方式是pip install numpy,对于python3可能需使用pip3，前提是你已安装了python环境和对应的pip工具。本文强烈建议学习者使用IPython交互式 shell其在命令行下有大的提示功能，对应安装。或者使用可直接运行python代码的Jupyter交互式笔记本，对应安装。本使用ipython工具作为演示，只要在shell里输入ipython即可，如下所示：

```
~$ ipython
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.4.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
```

紧接着，我们只需输入import numpy as np 即可导入numpy模块。

numpy数组

python的整数不单纯

由于python本身的动态特性，不需要像静态语言那般需要首先声明变量和规定其对应的变量类型(如在中，使用变量i之前需要声明int i = 0;)，这意味着我们可给同一变量赋予不同的值（如x = 0; x = 'python',两次赋值都是合法的）。我们知道标准的python实现是用c语言编写的，python中每个变量都一个巧妙封装的c结构体变量。所以，就一个具体的长整型来说不是一个单纯的长整型，其在c中对应结构体：

```
struct _longobject {
    long ob_refcnt ; //一个帮助Python静默处理内存分配和释放的引用计数
    PyTypeObject * ob_type ; //它编码变量的类型
    size_t ob_size ; //指定以下数据成员的大小
    long ob_digit [ 1 ]; //它包含我们期望Python变量表示的实际整数值。
};
```

这意味着python的数据结构天然的比c语言或其他相似的静态语言数据结构要多出许多而外的开销。此，也可以推断出，python的l列表也不是单纯的列表，二是索引了其他对象的列表。所以，即使我输入[True, "2", 3.0, 4]也是合法的。这有点类似于c#或java中的var ls = new List<object>{true, "2", 3.0, 4};，同样的在c中空指针void *p;也是个万能的存在,你可以用它指向任何对象。在灵活性方面无挑剔，但是就资源和效率方面来说实在让人感到堪忧。作为固定类型的numpy数组可以解决我们的担。

Python中的固定类型数组

1. array

array在python3.3以后才合法,。在ipython中，输入以下内容：

```
In [1]: import array
In [2]: L = list(range(10))
In [3]: A = array.array('i', L)     # i是类型标识符，用于指定数组为整性
In [4]: A
Out[4]: array('i', [0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

2. numpy

在ipython中，使用np.array即可创建数组

```
In [1]: import numpy as np
In [2]: np.array([1, 4, 2, 5, 3])
Out[2]: array([1, 4, 2, 5, 3])
```

可以使用dtpye来指定具体类型np.array([1, 2, 3, 4], dtype='float32')。作为重点：numpy数组和普python数组不同，它可以是多维的，这意味着它可以用来进行矩阵运算。

```
In [1]: np.array([range(i, i + 3) for i in [2, 4, 6]])
Out[1]:
array([[2, 3, 4],
       [4, 5, 6],
       [6, 7, 8]])
```

使用numpy原生借口创建大型数组，效率会有更好的支持。

创建初始值为0的数组：

```
In [1]: np.zeros(10, dtype=int)
Out[1]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

创建初始值为1的多维数组：

```
In [1]: np.ones((3, 5), dtype=float)
Out[1]:
array([[ 1.,  1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.,  1.]])
```

创建自定义初始值的多维数组：

```
In [1]: np.full((3, 5), 3.14)
Out[1]:
array([[ 3.14,  3.14,  3.14,  3.14,  3.14],
       [ 3.14,  3.14,  3.14,  3.14,  3.14],
       [ 3.14,  3.14,  3.14,  3.14,  3.14]])
```

也可以像普通数组那样创建线性序列：

```
In [1]: np.arange(0, 20, 2)
Out[1]: array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18])
```

创建均分序列：

```
In [1]: np.linspace(0, 1, 5)
Out[1]: array([ 0.  ,  0.25,  0.5 ,  0.75,  1.  ])
```

创建随机数组：

```
In [1]: np.random.random((3, 3))
Out[1]:
array([[ 0.68306984,  0.88296926,  0.25726865],
       [ 0.58618715,  0.64491405,  0.79462547],
       [ 0.64385013,  0.95813808,  0.37981454]])
```

创建正态分布的数组：

```
In [1]: np.random.normal(0, 1, (3, 3))
Out[1]:
array([[ 1.19851917,  0.72038886, -1.42196799],
       [ 1.26959716,  3.40746064, -1.18897577],
       [ 0.98618923,  2.2003404 , -0.25561642]])
```

创建整形随机数组：

```
In [1]: np.random.randint(0, 10, (3, 3))
Out[1]:
array([[1, 3, 1],
       [8, 2, 9],
       [3, 8, 6]])
```

创建单位矩阵：

```
In [1]: np.eye(3)
Out[1]:
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
```

创建空数组（初始值为内存值，即只创建，但不改变内存分配得到的值）：

```
In [31]: np.empty(3)
Out[31]: array([ 1.,  1.,  1.])
```

numpy中的标准数据类型

数据类型	描述
bool_	存储为字节的布尔值（True或False）
int_	默认整数类型（与C相同long;通常为int64或int32）
intc	与C相同int（通常int32或int64）
intp	用于索引的整数（与C相同ssize_t;通常为int32或者int64）
int8	字节（-128到127）
int16	整数（-32768至32767）
int32	整数（-2147483648至2147483647）
int64	整数（-9223372036854775808至9223372036854775807）
uint8	无符号整数（0到255）
uint16	无符号整数（0到65535）
uint32	无符号整数（0到4294967295）
uint64	无符号整数（0到18446744073709551615）
float_	简写float64。
float16	半精度浮点数：符号位，5位指数，10位尾数
float32	单精度浮点数：符号位，8位指数，23位尾数
float64	双精度浮点数：符号位，11位指数，52位尾数
complex_	简写complex128。
complex64	复数，由两个32位浮点数表示
complex128	复数，由两个64位浮点数表示

我们可以这样使用np.zeros(10, dtype=np.int16)或者np.zeros(10, dtype='int16')。当然，除此之

，也可以自定义更多高级的数据类型，例如指定其所占的字节数大小，有兴趣或者需要的请自行查阅。