Python 数据科学 速查表

Numpy 基础

NumPy

Numpy 是 Python 数据科学计算的核心库，提供了高性能的多维数组对象及处理数组的工具。

使用以下语句导入 Numpy 库：

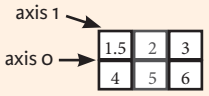
```
>>> import numpy as np
```

NumPy 数组

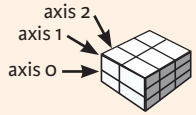
1维数组



2维数组



3维数组



创建数组

```
>>> a = np.array([1,2,3])
>>> b = np.array([(1.5,2,3), (4,5,6)], dtype = float)
>>> c = np.array([[(1.5,2,3), (4,5,6)], [(3,2,1), (4,5,6)]],
                 dtype = float)
```

初始化占位符

```
>>> np.zeros((3,4))                    创建值为0数组
>>> np.ones((2,3,4),dtype=np.int16)    创建值为1数组
>>> d = np.arange(10,25,5)             创建均匀间隔的数组（步进值）

>>> np.linspace(0,2,9)                 创建均匀间隔的数组（样本数）

>>> e = np.full((2,2),7)               创建常数数组
>>> f = np.eye(2)                      创建2x2单位矩阵
>>> np.random.random((2,2))            创建随机值的数组
>>> np.empty((3,2))                    创建空数组
```

输入/输出

保存与载入磁盘上的文件

```
>>> np.save('my_array', a)
>>> np.savez('array.npz', a, b)
>>> np.load('my_array.npy')
```

保存与载入文本文件

```
>>> np.loadtxt("myfile.txt")
>>> np.genfromtxt("my_file.csv", delimiter=',')
>>> np.savetxt("myarray.txt", a, delimiter=" ")
```

数据类型

```
>>> np.int64          带符号的64位整数
>>> np.float32        标准双精度浮点数
>>> np.complex        显示为128位浮点数的复数
>>> np.bool           布尔值：True值和False值
>>> np.object         Python对象
>>> np.string_        固定长度字符串
>>> np.unicode_       固定长度Unicode
```

数组信息

```
>>> a.shape            数组形状，几行几列
>>> len(a)             数组长度
>>> b.ndim             几维数组
>>> e.size             数组有多少元素
>>> b.dtype            数据类型
>>> b.dtype.name       数据类型的名字
>>> b.astype(int)      数据类型转换
```

调用帮助

```
>>> np.info(np.ndarray.dtype)
```

数组计算

算数运算

```
>>> g = a - b                          减法
  array([[-0.5,  0. ,  0. ],
         [-3. , -3. , -3. ]])
>>> np.subtract(a,b)                   减法
>>> b + a                              加法
  array([[ 2.5,  4. ,  6. ],
         [ 5. ,  7. ,  9. ]])
>>> np.add(b,a)                        加法
>>> a / b                              除法
  array([[ 0.66666667,  1.        ,  1.        ],
         [ 0.25      ,  0.4       ,  0.5       ]])
>>> np.divide(a,b)                     除法
>>> a * b                              乘法
  array([[  1.5,   4. ,   9. ],
         [  4. ,  10. ,  18. ]])
>>> np.multiply(a,b)                   乘法
>>> np.exp(b)                          幂
>>> np.sqrt(b)                         平方根
>>> np.sin(a)                          正弦
>>> np.cos(b)                          余弦
>>> np.log(a)                          自然对数
>>> e.dot(f)                           点积
  array([[ 7.,  7.],
         [ 7.,  7.]])
```

比较

```
>>> a == b                             对比值
  array([[False,  True,  True],
         [False, False, False]], dtype=bool)
>>> a < 2                              对比值
  array([True, False, False], dtype=bool)
>>> np.array_equal(a, b)               对比数组
```

聚合函数

```
>>> a.sum()                    数组汇总
>>> a.min()                    数组最小值
>>> b.max(axis=0)              数组最大值，按行
>>> b.cumsum(axis=1)           数组元素的累加值
>>> a.mean()                   平均数
>>> b.median()                 中位数
>>> a.corrcoef()               相关系数
>>> np.std(b)                  标准差
```

数组复制

```
>>> h = a.view()               使用同一数据创建数组视图
>>> np.copy(a)                 创建数组的副本
>>> h = a.copy()               创建数组的深度拷贝
```

数组排序

```
>>> a.sort()                   数组排序
>>> c.sort(axis=0)             以轴为依据对数组排序
```

子集、切片、 索引

参阅 列表

子集

```
>>> a[2]                                选择索引2对应的值
  3
>>> b[1,2]                              选择行1列2对应的值（等同于b[1][2]
  6.0
```

切片

```
>>> a[0:2]                              选择索引为0与1对应的值
  array([1, 2])
>>> b[0:2,1]                            选择第1列中第0行、第1行的值
  array([ 2.,  5.])
>>> b[:1]                               选择第0行的所有值（等同于b[0:1,:1]
  array([[1.5, 2., 3.]])
>>> c[1,...]                            等同于 [1,:,:]
  array([[[ 3.,  2.,  1.],
          [ 4.,  5.,  6.]]])
>>> a[ : :-1]                           反转数组a
  array([3, 2, 1])
```

条件索引

```
>>> a[a<2]                              选择数组a中所有小于2的值
  array([1])
```

花式索引

```
>>> b[[1, 0, 1, 0],[0, 1, 2, 0]]       选择(1,0),(0,1),(1,2) 和(0,0)所对应的值
  array([ 4. , 2. , 6. , 1.5])
>>> b[[1, 0, 1, 0]][:,[0,1,2,0]]       选择矩阵的行列子集
  array([[ 4. , 5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5],
         [ 4. , 5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5]])
```

数组操作

转置数组

```
>>> i = np.transpose(b)                转置数组
>>> i.T                                转置数组
```

改变数组形状

```
>>> b.ravel()                          拉平数组
>>> g.reshape(3,-2)                    改变数组形状，但不改变数据
```

添加或删除值

```
>>> h.resize((2,6))                    返回形状为(2,6)的新数组
>>> np.append(h,g)                     追加数据
>>> np.insert(a, 1, 5)                 插入数据
>>> np.delete(a,[1])                   删除数据
```

合并数组

```
>>> np.concatenate((a,d),axis=0)       拼接数组
  array([ 1,  2,  3, 10, 15, 20])
>>> np.vstack((a,b))                   纵向以行的维度堆叠数组
  array([[ 1. ,  2. ,  3. ],
         [ 1.5,  2. ,  3. ],
         [ 4. ,  5. ,  6. ]])
>>> np.r_[e,f]                         纵向以行的维度堆叠数组
>>> np.hstack((e,f))                   横向以列的维度堆叠数组
  array([[ 7.,  7.,  1.,  0.],
         [ 7.,  7.,  0.,  1.]])
>>> np.column_stack((a,d))             以列的维度创建堆叠数组
  array([[ 1, 10],
         [ 2, 15],
         [ 3, 20]])
>>> np.c_[a,d]                         以列的维度创建堆叠数组
```

分割数组

```
>>> np.hsplit(a,3)                     纵向分割数组为3等份
  [array([1]),array([2]),array([3])]
>>> np.vsplit(c,2)                     横向分割数组为2等份
[array([[[ 1.5,  2. ,  1. ],
         [ 4. ,  5. ,  6. ]]]),
 array([[[ 3.,  2.,  3.],
         [ 4.,  5.,  6.]]])]
```

原文作者 DataCamp Learn Python for Data Science Interactively