呆鸟译

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。

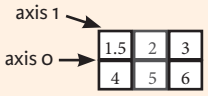
使用以下语句导入 Numpy 库：

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

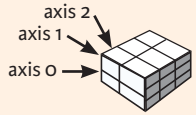
NumPy 数组

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))
>>> np.ones((2,3,4),dtype=np.int16)
>>> d = np.arange(10,25,5)

>>> np.linspace(0,2,9)

>>> e = np.full((2,2),7)
>>> f = np.eye(2)
>>> np.random.random((2,2))
>>> np.empty((3,2))
```

创建值为0数组
创建值为1数组
创建均匀间隔的数组（步进值）

创建均匀间隔的数组（样本数）

创建常数数组
创建2x2单位矩阵
创建随机值的数组
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
>>> np.float32
>>> np.complex
>>> np.bool
>>> np.object
>>> np.string_
>>> np.unicode_
```

带符号的64位整数
标准双精度浮点数
显示为128位浮点数的复数
布尔值：True值和False值
Python对象
固定长度字符串
固定长度Unicode

数组信息

```
>>> a.shape
>>> len(a)
>>> b.ndim
>>> e.size
>>> b.dtype
>>> b.dtype.name
>>> b.astype(int)
```

数组形状，几行几列
数组长度
几维数组
数组有多少元素
数据类型
数据类型的名字
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
  array([[-0.5,  0. ,  0. ],
         [-3. , -3. , -3. ]])
>>> np.subtract(a,b)
>>> b + a
  array([[ 2.5,  4. ,  6. ],
         [ 5. ,  7. ,  9. ]])
>>> np.add(b,a)
>>> a / b
  array([[ 0.66666667,  1.        ,  1.        ],
         [ 0.25      ,  0.4       ,  0.5       ]])
>>> np.divide(a,b)
>>> a * b
  array([[  1.5,   4. ,   9. ],
         [  4. ,  10. ,  18. ]])
>>> np.multiply(a,b)
>>> np.exp(b)
>>> np.sqrt(b)
>>> np.sin(a)
>>> np.cos(b)
>>> np.log(a)
>>> e.dot(f)
  array([[ 7.,  7.],
         [ 7.,  7.]])
```

减法

减法
加法

加法
除法

除法
乘法

乘法
幂
平方根
正弦
余弦
自然对数
点积

比较

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

对比值

对比值

对比数组

聚合函数

```
>>> a.sum()
>>> a.min()
>>> b.max(axis=0)
>>> b.cumsum(axis=1)
>>> a.mean()
>>> b.median()
>>> a.corrcoef()
>>> np.std(b)
```

数组汇总
数组最小值
数组最大值，按行
数组元素的累加值
平均数
中位数
相关系数
标准差

数组复制

```
>>> h = a.view()
>>> np.copy(a)
>>> h = a.copy()
```

使用同一数据创建数组视图
创建数组的副本
创建数组的深度拷贝

数组排序

```
>>> a.sort()
>>> c.sort(axis=0)
```

数组排序
以轴为依据对数组排序

子集、切片、 索引

参阅 列表

子集

```
>>> a[2]
  3
>>> b[1,2]
  6.0
```

选择索引2对应的值
选择行1列2对应的值（等同于b[1][2]

切片

```
>>> a[0:2]
  array([1, 2])
>>> b[0:2,1]
  array([ 2.,  5.])
>>> b[:1]
  array([[1.5, 2., 3.]])
>>> c[1,...]
  array([[[ 3.,  2.,  1.],
          [ 4.,  5.,  6.]]])
>>> a[ : :-1]
  array([3, 2, 1])
```

选择索引为0与1对应的值
选择第1列中第0行、第1行的值
选择第0行的所有值（等同于b[0:1,:1]
等同于 [1,:,:]
反转数组a

条件索引

```
>>> a[a<2]
  array([1])
```

选择数组a中所有小于2的值

花式索引

```
>>> b[[1, 0, 1, 0],[0, 1, 2, 0]]
  array([ 4. , 2. , 6. , 1.5])
>>> b[[1, 0, 1, 0]][:,[0,1,2,0]]
  array([[ 4. , 5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5],
         [ 4. , 5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5]])
```

选择(1,0),(0,1),(1,2) 和(0,0)所对应的值
选择矩阵的行列子集

数组操作

转置数组

```
>>> i = np.transpose(b)
>>> i.T
```

转置数组
转置数组

改变数组形状

```
>>> b.ravel()
>>> g.reshape(3,-2)
```

拉平数组
改变数组形状，但不改变数据

添加或删除值

```
>>> h.resize((2,6))
>>> np.append(h,g)
>>> np.insert(a, 1, 5)
>>> np.delete(a,[1])
```

返回形状为(2,6)的新数组
追加数据
插入数据
删除数据

合并数组

```
>>> np.concatenate((a,d),axis=0)
  array([ 1,  2,  3, 10, 15, 20])
>>> np.vstack((a,b))
  array([[ 1. ,  2. ,  3. ],
         [ 1.5,  2. ,  3. ],
         [ 4. ,  5. ,  6. ]])
>>> np.r_[e,f]
>>> np.hstack((e,f))
  array([[ 7.,  7.,  1.,  0.],
         [ 7.,  7.,  0.,  1.]])
>>> np.column_stack((a,d))
  array([[ 1, 10],
         [ 2, 15],
         [ 3, 20]])
>>> np.c_[a,d]
```

拼接数组
纵向以行的维度堆叠数组
纵向以行的维度堆叠数组
横向以列的维度堆叠数组
以列的维度创建堆叠数组
以列的维度创建堆叠数组

分割数组

```
>>> np.hsplit(a,3)
  [array([1]),array([2]),array([3])]
>>> np.vsplit(c,2)
[array([[[ 1.5,  2. ,  1. ],
         [ 4. ,  5. ,  6. ]]]),
 array([[[ 3. ,  2. ,  3. ],
         [ 4. ,  5. ,  6. ]]])]
```

纵向分割数组为3等份
横向分割数组为2等份