

Note_Numpy

2018年8月13日 19:36

1. 模块的导入

- a. import numpy: 直接导入模块
- b. import numpy as np: 导入模块后改名
- c. from numpy import array: 从模块中导入其中的array方法
- d. from numpy import array as ar: 从模块中导入方法并改名

2. 创建数组

```
ar1 = np.array([2, 3, 4]) # 通过列表创建数组
ar2 = np.array([(1.2, 9, 2.0),(7, 6, 5)]) # 通过元组创建数组
ar3 = np.zeros((2, 3)) # 通过元组创建零矩阵 (2x3)
ar4 = np.identity(3) # 生成单位矩阵(3x3)
ar5 = np.random.random(size=(2, 3)) # 生成每个元素都在[0,1]之间的随机矩阵(2x3)
ar6 = np.arange(5, 20, 3) # 生成等距序列, 参数为起点, 终点, 步长, 右不包含
ar7 = np.linspace(5, 20, 3) # 生成等距序列, 参数为起点, 终点, 元素个数, 右包含
```

3. 访问数组属性

```
print(ar2.shape) # 返回矩阵的规格
print(ar2.ndim) # 返回矩阵的秩
print(ar2.size) # 返回矩阵元素总数
print(ar2.dtype.name) # 返回矩阵元素的数据类型
print(type(ar2)) # 查看整个数组对象的类型
```

4. 通过索引和切片访问数组元素

```
def f(x, y):
    return 10*x+y
ar8 = np.fromfunction(f, (4,3), dtype = int)
# 此处是通过调用函数创建复杂矩阵。x是行索引, y是列索引。索引都从0开始。

print(ar8[1, 2]) # 返回行索引为1, 列索引为2的元素
print(ar8[0:2, :]) # 返回矩阵前2行, 前面 0:2 表示行索引切片, 后面 : 表示全部列索引
print(ar8[:, 1]) # 返回矩阵第 1列, 结果会以横置的列表显示
print(ar8[-1]) # 返回矩阵最后一行
```

5. 通过迭代器访问数组元素

```
for row in ar8:
    print(row)
```

```
# 输出每一行
```

```
for element in ar8.flat:
```

```
    print (element)
```

```
# 输出每一个元素, 注意 flat 方法是用于遍历整个数组的迭代器
```

6. 数组的运算

```
ar9 = np.array([[2, 1], [1, 2]]) # 创建一个2x2矩阵
```

```
ar10 = np.array([[1, 2], [3, 4]]) # 创建另一个2x2矩阵
```

```
ar11 = np.array([[1, 2, 3], [2, 3, 4], [3, 4, 5]])
```

```
print (ar9 - ar10) # 矩阵加减法
```

```
print (ar9 ** 3) # 矩阵每个元素求幂
```

```
print (3 * ar10) # 矩阵的数乘
```

```
print (ar9 * ar10) # 矩阵每个元素求积
```

```
print (np.dot(ar9, ar10)) # 矩阵的左乘
```

```
print (ar10.T) # 矩阵转置
```

```
print (np.linalg.inv(ar10)) # 矩阵求逆
```

```
# 数组内部元素求和
```

```
print (ar10)
```

```
print (ar10.sum()) # 数组内部元素求和
```

```
print (ar10.sum(axis=0)) # 返回行求和 (每行相同列坐标的元素相加)
```

```
print (ar10.sum(axis=1)) # 返回列求和 (每列相同行坐标的元素相加)
```

```
# 数组内部元素求最大值
```

```
print (ar10)
```

```
print (ar10.max()) # 返回最大值
```

```
print (ar10.max(axis=0)) # 返回最大值所在的行
```

```
print (ar10.max(axis=1)) # 返回最大值所在的列
```

```
# 数组内部元素按行列累计求和
```

```
print (ar11)
```

```
print (ar11.cumsum()) # 每个元素累计求和
```

```
print (ar11.cumsum(axis=1)) # 行不动, 每列累计求和
```

```
print (ar11.cumsum(axis=0)) # 列不动, 每行累计求和
```

7. 通用函数

```
import math
```

```
print (ar9)
print (np.exp(ar9)) # 对每个元素求自然底的指数函数
print (math.exp(2)) # 验算
```

```
print (ar9)
print (np.sin(ar9)) # 对每个元素求 sin(弧度)
print (math.sin(2)) # 验算
```

```
print (ar9)
print (np.sqrt(ar9)) # 对每个元素求平方根
print (math.sqrt(2)) # 验算
```

```
print (ar9)
print (np.add(ar9,ar10)) # 求和函数
```

8. 数组的合并与分割

```
# 合并(堆叠)
ar12 = np.vstack((ar9, ar10)) # 数组的垂直堆叠 vertical stack
ar13 = np.hstack((ar9, ar10)) # 数组的水平堆叠 horizontal stack
print (ar12)
print (ar13)

# 分割
print (ar13)
print (np.vsplit(ar13, 2)) # 将数组垂直分割（即水平切割）成两个数组

print (ar12)
print (np.hsplit(ar12, 2)) # 将数组水平分割（即垂直切割）成两个数组
```

9. numpy 的其他方法

```
print (np.empty([3, 4], dtype=int)) # 返回一个3x4的数组
print (np.all(ar13)) # 判断是否 all 元素均为True, 返回 True or False
print (np.any(ar13)) # 判断是否 any 元素为True, 返回 True or False
print (np.average(ar13)) # 返回所有元素计算的平均值
print (np.std(ar13)) # 返回所有元素计算的标准差
print (np.var(ar13)) # 返回所有元素计算的方差
print (np.cov(ar13)) # 返回协方差矩阵, 可以附加权重参数
print (np.nonzero(ar13)) # 返回所有非零元素的位置
print (np.sort(ar13)) # 对数组内数据进行升序排序 (按行)
print (np.reshape(ar13,(1,8))) # 转换数组的规模但不改变其中的数据
```

```
print (np.eye(6)) # 生成 6X6 的单位矩阵 identity 也可以实现  
print (np.transpose(ar13)) # 转置与 .T 相同
```