**数组对象**

Numpy中常用的数据结构是ndarray格式(又叫array)

**ndim属性**

表示数组维度

array对象.ndim

**shape属性**

表示数组的形状

array对象.shape

**dtype属性**

表示数组中元素的类型

array对象.dtype

**size属性**

表示数组中元素的总数目

array对象.size

**创建数组**

**根据数据类型创建数组**

import numpy as np#导入numpy库

array\_id=np.array([1,2,3])#基于列表创建数组

**根据指定数值创建数组**

import numpy as np#导入numpy库

array\_ones=np.ones((2,3))#输出形状为(2,3)的数组，其元素全为1

array\_zeros=np.ones((2,3))#输出形状为(2,3)的数组，其元素全为0

array\_empty=np.empty((2,3))#输出形状为(2,3)的数组，其元素全是随机的

**根据指定数值范围创建数组**

import numpy as np#导入numpy库

array\_arange=np.arange(1,30,5)#arange和range的写法一样，都是一维数组

#可以用reshape转换形状

new\_arr=array\_arange.reshape(2,3)#将一维数组变为形状为(2,3)的数组

**访问数组元素**

**使用整数索引访问元素**

#一维

arr\_one=np.array([1,2,3])

arr\_one[2]#输出索引为3的元素：3

#二维

arr\_two=np.array([[1,2,3],[4,5,6]])

arr\_two[1]#输出第二行是元素：[4,5,6]

arr\_two[1,2]#输出第二行第三列的元素：6

**使用花式索引或布尔索引访问元素**

**花式索引(两组方括号)**

#一维

arr\_one=np.array([1,2,3,4,5,6,7,8,9])

arr\_one[[2,5,8]]#输出下标为2和5和8的元素组成的数组：[3,6,9]

#二维

arr\_two=np.array([[1,2,3],[4,5,6],[7,8,9]])

arr\_two[[0,2]]#输出3一行和第三行的数组：[[1,2,3],[7,8,9]]

arr\_two[[0,2],[1,2]]#输出下标为(0,1)和(2,2)的元素组成的数组：[2,9]

**布尔索引**

arr\_two=np.array([[1,2,3],[4,5,6],[7,8,9]])

arr\_two>5#输出[[False,False,False],[False,False,True],[True,True,True]]

arr\_two[arr\_two>5]#输出为True的元素：[6,7,8,9]

**使用切片访问元素**

**一维切片**

import numpy as np

array\_ones=np.array([10.20,30,40,50,60])

array\_ones[1:4]#访问索引为1，2，3的元素(是左闭右开的范围)

array\_ones[:4]#访问前四个元素

array\_ones[:-1]#访问除末尾元素以外的元素

array\_ones[:]#访问全部元素

array\_ones[::2]#访问从开头到末尾,步长为2获取的元素

**二维切片**

import numpy as np

arr\_two=np.array([[1,2,3],[4,5,6],[7,8,9]])

arr\_two[:2]#访问前两行数据

arr\_two[:2，0:1]#访问前两行，第一列数据

arr\_two[:2，1]#访问前两行，第二1列数据

**数组运算**

**形状相同的数组间运算**

import numpy as np

arr1=np.array([[1,2,3],[4,5,6],[7,8,9]])

arr2=np.array([[1,2,3],[4,5,6],[7,8,9]])

arr3=arr1+arr2#[[2,4,6],[8,10,12],[14,16,18]]

#其实就是线性代数的同位置相加减乘除

**形状不同的数组间运算**

import numpy as np

arr1=np.array([[1],[2],[3]])#形状为(3,)

arr2=np.array([1,2,3])#形状为(,3)

arr3=arr1+arr2#arr1和arr2按行或按列复制自己的元素来匹配成相同的形状,再运算

#arr3为[[2,3,4],[3,4,5],[4,5,6]]形状为(3,3)

**数组与标量的运算**

import numpy as np

arr1=np.array([[1,2,3],[4,5,6],[7,8,9]])

num=1

arr2=arr1+num#每个元素都加1：[[2,3,4],[5,6,7],[8,9,10]]

#数组与标量的运算:其实是数组中每一个元素都与该标量进行加减乘除

**矩阵点乘**

np.dot(np矩阵1, np矩阵2)

//与

np矩阵1.dot(np矩阵2)

//等价

**数组操作**

**排序**

**值的排序**

sort(axis=-1,kind=None,order=None)#从大到小排序

- axis : 表示排序的轴编号，默认为-1，表示沿着最后一个轴排序

- kind : 表示排序的算法，默认为'quicksort'(快速排序)

- order: 表示按哪个字段排序

#二维数组排序

import numpy as np

arr1=np.array([[6,8,7],[3,1,9],[0,5,2]])

arr1.sort()#在自己的行中进行排序：[[6,7,8],

# [1,3,9],

# [0,2,5]]

arr2=np.array([[6,8,7],[3,1,9],[0,5,2]])

arr2.sort(axis=0)#在行列同时进行排序：[[0 1 2],

# [3 5 7],

# [6 8 9]]

sorted(arr1,reverse=True)#降序

**索引的排序**

s=np.array([1,2,3,4,3,1,2,2,4,6,7,2,4,8,4,5])

np.argsort(s)#返回的是数据中从小到大的索引值

**检索数组元素**

np.all(arr>0)#检索arr的元素是否全部大于0

np.any(arr>0)#检索arr的元素至少有一个大于0

np.where(arr>3,1,-1)#满足条件赋值为1,不满足条件赋值为-1，返回长度与arr一样

np.extract(arr>3,arr)#只输出满足条件的数据

**元素唯一化**

np.unique(arr)#输出arr中不重复的元素

**数组的转置**

**T属性(二维转置)**

arr1=np.arange([[1,2,3],[4,5,6]])#形状为(2,3)

arr2=arr1.T#转置将行列转换

#arr2为[[1,4],[2,5],[3,6]],形状为(3,2)

**swapaxes()方法(三维转置)**

import numpy as np

arr = np.arange(16).reshape(2,2,4) #创建一个2\*2\*4的三维数组

# array([[[ 0, 1, 2, 3],

# [ 4, 5, 6, 7]],

#

# [[ 8, 9, 10, 11],

# [12, 13, 14, 15]]])

arr.swapaxes(1,0)

# array([[[ 0, 1, 2, 3],

# [ 8, 9, 10, 11]],

#

# [[ 4, 5, 6, 7],

# [12, 13, 14, 15]]])

**transpose()方法**

import numpy as np

arr = np.arange(16).reshape(2,2,4) #创建一个2\*2\*4的三维数组

# array([[[ 0, 1, 2, 3],

# [ 4, 5, 6, 7]],

#

# [[ 8, 9, 10, 11],

# [12, 13, 14, 15]]])

brr = arr.transpose(1,2,0)

# array([[[ 0, 8],

# [ 1, 9],

# [ 2, 10],

# [ 3, 11]],

#

# [[ 4, 12],

# [ 5, 13],

# [ 6, 14],

# [ 7, 15]]])

**随机数**

**随机矩阵**

np.random.randint(start, end, size = (n, m))#返回start和end之间的任意整数组成的矩阵(n,m)

**生成指定个数的[0,1)均匀分布随机点**

#返回一个或一组服从[0,1)均匀分布的随机样本值

np.random.rand(个数)

#返回服从[0,1)均匀分布的随机样本值矩阵

np.random.rand(行数,列数)

**从正态分布中抽取随机样本**

numpy.random.normal(loc=中心点, scale=宽度, size=随机样本个数)

**创建等差数列**

x\_data =np.linspace(起点,终点,个数,endpoint)#：endpoint(可选):True则包含stop；False则不包含stop