Python内建的filter()函数用于过滤序列。

和map()类似，filter()也接收一个函数和一个序列。和map()不同的时，filter()把传入的函数依次作用于每个元素，然后根据返回值是True还是False决定保留还是丢弃该元素。

例如，在一个list中，删掉偶数，只保留奇数，可以这么写：

[?](https://www.jb51.net/article/64999.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5 | **def** is\_odd(n):  **return** n **%** 2 **==** 1    filter(is\_odd, [1, 2, 4, 5, 6, 9, 10, 15])  # 结果: [1, 5, 9, 15] |

把一个序列中的空字符串删掉，可以这么写：

[?](https://www.jb51.net/article/64999.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5 | **def** not\_empty(s):  **return** s **and** s.strip()    filter(not\_empty, ['A', '', 'B', None, 'C', ' '])  # 结果: ['A', 'B', 'C'] |

可见用filter()这个高阶函数，关键在于正确实现一个“筛选”函数。

.ndim ：维度   
.shape ：各维度的尺度 （2，5）   
.size ：元素的个数 10   
.dtype ：元素的类型 dtype(‘int32’)   
.itemsize ：每个元素的大小，以字节为单位 ，每个元素占4个字节   
ndarray数组的创建   
np.arange(n) ; 元素从0到n-1的ndarray类型   
np.ones(shape): 生成全1   
np.zeros((shape)， ddtype = np.int32) ： 生成int32型的全0   
np.full(shape, val): 生成全为val   
np.eye(n) : 生成单位矩阵

np.ones\_like(a) : 按数组a的形状生成全1的数组   
np.zeros\_like(a): 同理   
np.full\_like (a, val) : 同理

np.linspace（1,10,4）： 根据起止数据等间距地生成数组   
np.linspace（1,10,4, endpoint = False）：endpoint 表示10是否作为生成的元素   
np.concatenate():

* 数组的维度变换

.reshape(shape) : 不改变当前数组，依shape生成   
.resize(shape) : 改变当前数组，依shape生成   
.swapaxes(ax1, ax2) : 将两个维度调换   
.flatten() : 对数组进行降维，返回折叠后的一位数组

* 数组的类型变换

数据类型的转换 ：a.astype(new\_type) : eg, a.astype (np.float)   
数组向列表的转换： a.tolist()   
数组的索引和切片

* 一维数组切片

a = np.array ([9, 8, 7, 6, 5, ])   
a[1:4:2] –> array([8, 6]) ： a[起始编号：终止编号（不含）： 步长]

* 多维数组索引

a = np.arange(24).reshape((2, 3, 4))   
a[1, 2, 3] 表示 3个维度上的编号， 各个维度的编号用逗号分隔

* 多维数组切片

a [：，：，：：2 ] 缺省时，表示从第0个元素开始，到最后一个元素   
数组的运算   
np.abs(a) np.fabs(a) : 取各元素的绝对值   
np.sqrt(a) : 计算各元素的平方根   
np.square(a): 计算各元素的平方   
np.log(a) np.log10(a) np.log2(a) : 计算各元素的自然对数、10、2为底的对数   
np.ceil(a) np.floor(a) : 计算各元素的ceiling 值， floor值（ceiling向上取整，floor向下取整）   
np.rint(a) : 各元素 四舍五入   
np.modf(a) : 将数组各元素的小数和整数部分以两个独立数组形式返回   
np.exp(a) : 计算各元素的指数值   
np.sign(a) : 计算各元素的符号值 1（+），0，-1（-）   
.   
np.maximum(a, b) np.fmax() : 比较（或者计算）元素级的最大值   
np.minimum(a, b) np.fmin() : 取最小值   
np.mod(a, b) : 元素级的模运算   
np.copysign(a, b) : 将b中各元素的符号赋值给数组a的对应元素

* 数据的CSV文件存取

CSV (Comma-Separated Value,逗号分隔值) 只能存储一维和二维数组

np.savetxt(frame, array, fmt=’% .18e’, delimiter = None): frame是文件、字符串等，可以是.gz .bz2的压缩文件； array 表示存入的数组； fmt 表示元素的格式 eg： %d % .2f % .18e ; delimiter： 分割字符串，默认是空格   
eg： np.savetxt(‘a.csv’, a, fmt=%d, delimiter = ‘,’ )

np.loadtxt(frame, dtype=np.float, delimiter = None, unpack = False) : frame是文件、字符串等，可以是.gz .bz2的压缩文件； dtype：数据类型，读取的数据以此类型存储； delimiter: 分割字符串，默认是空格; unpack: 如果为True， 读入属性将分别写入不同变量。   
多维数据的存取   
a.tofile(frame, sep=’’, format=’%s’ ) : frame: 文件、字符串； sep: 数据分割字符串，如果是空串，写入文件为二进制 ； format:： 写入数据的格式   
eg: a = np.arange(100).reshape(5, 10, 2)   
a.tofile(“b.dat”, sep=”,”, format=’%d’)

np.fromfile(frame, dtype = float, count=-1, sep=’’)： frame： 文件、字符串 ； dtype： 读取的数据以此类型存储； count：读入元素个数， -1表示读入整个文件； sep: 数据分割字符串，如果是空串，写入文件为二进制

PS: a.tofile() 和np.fromfile（）要配合使用，要知道数据的类型和维度。

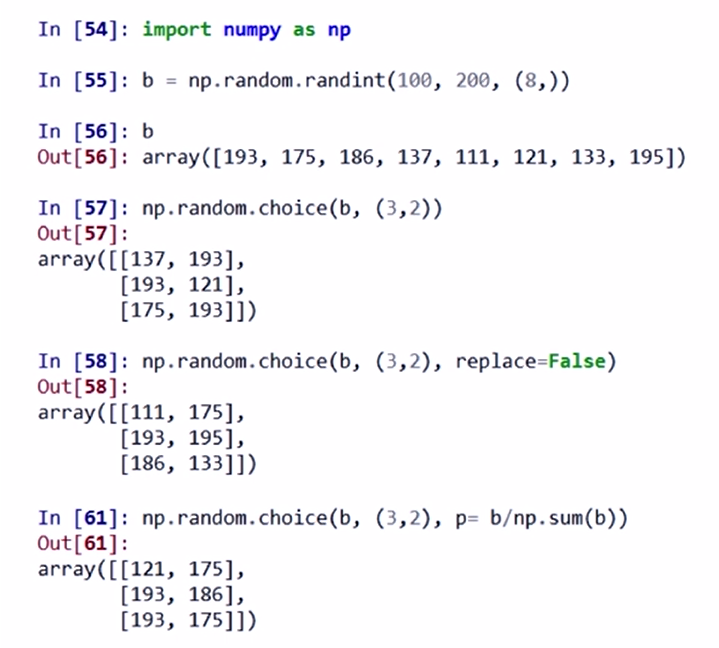
np.save(frame, array) : frame: 文件名，以.npy为扩展名，压缩扩展名为.npz ； array为数组变量   
np.load(fname) : frame: 文件名，以.npy为扩展名，压缩扩展名为

np.save() 和np.load() 使用时，不用自己考虑数据类型和维度。

* numpy随机数函数

numpy 的random子库

rand(d0, d1, …,dn) : 各元素是[0, 1）的浮点数，服从均匀分布   
randn(d0, d1, …,dn)：标准正态分布   
randint(low， high,（ shape）): 依shape创建随机整数或整数数组，范围是[ low, high）   
seed(s) ： 随机数种子

shuffle(a) : 根据数组a的第一轴进行随机排列，改变数组a   
permutation(a) : 根据数组a的第一轴进行随机排列， 但是不改变原数组，将生成新数组   
choice(a[, size, replace, p]) : 从一维数组a中以概率p抽取元素， 形成size形状新数组，replace表示是否可以重用元素，默认为False。   
eg：    
replace = False时，选取过的元素将不会再选取

uniform(low, high, size) : 产生均匀分布的数组，起始值为low，high为结束值，size为形状   
normal(loc, scale, size) : 产生正态分布的数组， loc为均值，scale为标准差，size为形状   
poisson(lam, size) : 产生泊松分布的数组， lam随机事件发生概率，size为形状   
eg: a = np.random.uniform(0, 10, (3, 4)) a = np.random.normal(10, 5, (3, 4))

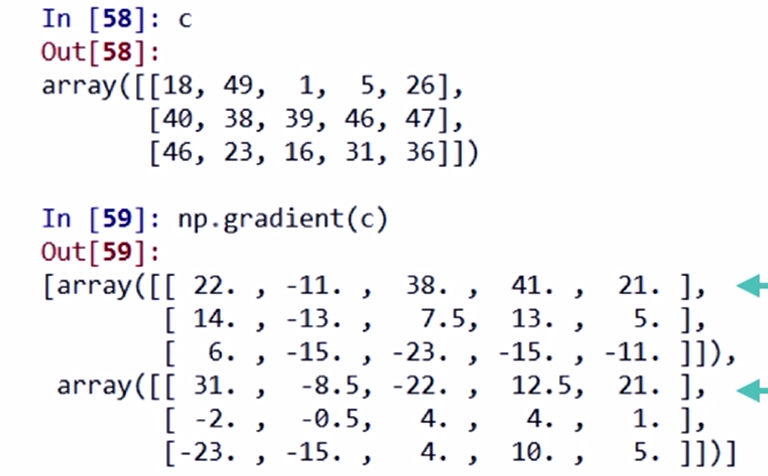
* numpy的统计函数

sum(a, axis = None) : 依给定轴axis计算数组a相关元素之和，axis为整数或者元组   
mean(a, axis = None) : 同理，计算平均值   
average(a, axis =None, weights=None) : 依给定轴axis计算数组a相关元素的加权平均值   
std（a, axis = None） ：同理，计算标准差   
var（a, axis = None）: 计算方差   
eg： np.mean(a, axis =1) ： 对数组a的第二维度的数据进行求平均   
a = np.arange(15).reshape(3, 5)   
np.average(a, axis =0, weights =[10, 5, 1]) : 对a第一各维度加权求平均，weights中为权重，注意要和a的第一维匹配

min(a) max(a) : 计算数组a的最小值和最大值   
argmin(a) argmax(a) : 计算数组a的最小、最大值的下标（注：是一维的下标）   
unravel\_index(index, shape) : 根据shape将一维下标index转成多维下标   
ptp(a) : 计算数组a最大值和最小值的差   
median(a) : 计算数组a中元素的中位数（中值）   
eg：a = [[15, 14, 13],   
[12, 11, 10] ]   
np.argmax(a) –> 0   
np.unravel\_index( np.argmax(a), a.shape) –> (0,0)

* numpy的梯度函数

np.gradient(a) ： 计算数组a中元素的梯度，f为多维时，返回每个维度的梯度   
离散梯度： xy坐标轴连续三个x轴坐标对应的y轴值：a, b, c 其中b的梯度是（c-a）/2   
而c的梯度是： (c-b)/1

当为二维数组时，np.gradient(a) 得出两个数组，第一个数组对应最外层维度的梯度，第二个数组对应第二层维度的梯度。   


* 图像的表示和变换

PIL， python image library 库   
from PIL import Image   
Image是PIL库中代表一个图像的类（对象）

im = np.array(Image.open(“.jpg”))

im = Image.fromarray(b.astype(‘uint8’)) # 生成   
im.save(“路径.jpg”) # 保存

im = np.array(Image.open(“.jpg”).convert(‘L’)) # convert(‘L’)表示转为灰度图

sys.argv #命令行参数List，第一个元素是程序本身路径 sys.exit(n) #退出程序，正常退出时exit(0) sys.version #获取Python解释程序的版本信息 sys.maxint #最大的Int值 sys.path #返回模块的搜索路径，初始化时使用PYTHONPATH环境变量的值 sys.platform #返回操作系统平台名称 sys.stdin #输入相关 sys.stdout #输出相关 sys.stderror #错误相关

os.getcwd() #获取当前工作目录，即当前python脚本工作的目录路径 os.chdir("dirname") #改变当前脚本工作目录；相当于shell下cd os.curdir #返回当前目录: ('.') os.pardir #获取当前目录的父目录字符串名：('..') os.makedirs('dir1/dir2') #可生成多层递归目录 os.removedirs('dirname1') #若目录为空，则删除，并递归到上一级目录，如若也为空，则删除，依此类推 os.mkdir('dirname') #生成单级目录；相当于shell中mkdir dirname os.rmdir('dirname') #删除单级空目录，若目录不为空则无法删除，报错；相当于shell中rmdir dirname os.listdir('dirname') #列出指定目录下的所有文件和子目录，包括隐藏文件，并以列表方式打印 os.remove() #删除一个文件 os.rename("oldname","new") #重命名文件/目录 os.stat('path/filename') #获取文件/目录信息 os.sep #操作系统特定的路径分隔符，win下为"\\",Linux下为"/" os.linesep #当前平台使用的行终止符，win下为"\t\n",Linux下为"\n" os.pathsep #用于分割文件路径的字符串 os.name #字符串指示当前使用平台。win->'nt'; Linux->'posix' os.system("bash command") 运行shell命令，直接显示 os.environ #获取系统环境变量 os.path.abspath(path) #返回path规范化的绝对路径 os.path.split(path) #将path分割成目录和文件名二元组返回 os.path.dirname(path) #返回path的目录。其实就是os.path.split(path)的第一个元素 os.path.basename(path) #返回path最后的文件名。如何path以／或\结尾，那么就会返回空值。即os.path.split(path)的第二个元素 os.path.exists(path) #如果path存在，返回True；如果path不存在，返回False os.path.isabs(path) #如果path是绝对路径，返回True os.path.isfile(path) #如果path是一个存在的文件，返回True。否则返回False os.path.isdir(path) #如果path是一个存在的目录，则返回True。否则返回False os.path.join(path1[, path2[, ...]]) #将多个路径组合后返回，第一个绝对路径之前的参数将被忽略 os.path.getatime(path) #返回path所指向的文件或者目录的最后存取时间 os.path.getmtime(path) #返回path所指向的文件或者目录的最后修改时间 os.getlogin() #获取当前登录用户

np.tile（A,raps）此函数将A重复输出

.get() .iteritems() sorted()

torch.max()[0]， 只返回最大值的每个数

troch.max()[1]， 只返回最大值的每个索引

torch.max()[1].data 只返回variable中的数据部分（去掉Variable containing:）

torch.max()[1].data.numpy() 把数据转化成numpy ndarry

torch.max()[1].data.numpy().squeeze() 把数据条目中维度为1 的删除掉