**实验任务5：numpy进阶**

**【实验目的】**

机器学习模型：在编写机器学习算法时，需要对矩阵进行各种数值计算。例如矩阵乘法、换位、加法等。NumPy提供了一个非常好的库，用于简单(在编写代码方面)和快速(在速度方面)计算。NumPy数组用于存储训练数据和机器学习模型的参数。

图像处理和计算机影像学：计算机中的图像表示为多维数字数组。NumPy成为同样情况下最自然的选择。实际上，NumPy提供了一些优秀的库函数来快速处理图像。例如，镜像图像、按特定角度旋转图像等。

**【实验原理】**

NumPy数组在创建时具有固定的大小，与Python的原生数组对象（可以动态增长）不同。 更改ndarray的大小将创建一个新数组并删除原来的数组。

NumPy数组中的元素都需要具有相同的数据类型，因此在内存中的大小相同。 例外情况：Python的原生数组里包含了NumPy的对象的时候，这种情况下就允许不同大小元素的数组。

NumPy数组有助于对大量数据进行高级数学和其他类型的操作。 通常，这些操作的执行效率更高，比使用Python原生数组的代码更少。

越来越多的基于Python的科学和数学软件包使用NumPy数组; 虽然这些工具通常都支持Python的原生数组作为参数，但它们在处理之前会还是会将输入的数组转换为NumPy的数组，而且也通常输出为NumPy数组。 换句话说，为了高效地使用当今科学/数学基于Python的工具（大部分的科学计算工具），你只知道如何使用Python的原生数组类型是不够的 - 还需要知道如何使用NumPy数组。

**【实验环境】**

Linux Ubuntu 16.04

Python 3.6.1

Jupyter NoteBook

**【实验内容】**

Numpy对于执行各种数学任务非常有用，如数值积分，微分，内插，外推等，本实验要求掌握使用Numpy处理矩阵维度运算，按照需求快速生成矩阵。熟练使用内置Numpy的基本方法。

**【实验步骤】**

1.打开终端模拟器，切换到data目录下，下载实验所需数据。

cd /data

wget http://172.18.3.177:32600/allfiles/iris/iris.data

Copy

2.在终端输入ipython notebook --ip='127.0.0.1'，在浏览器中会打开下面界面。

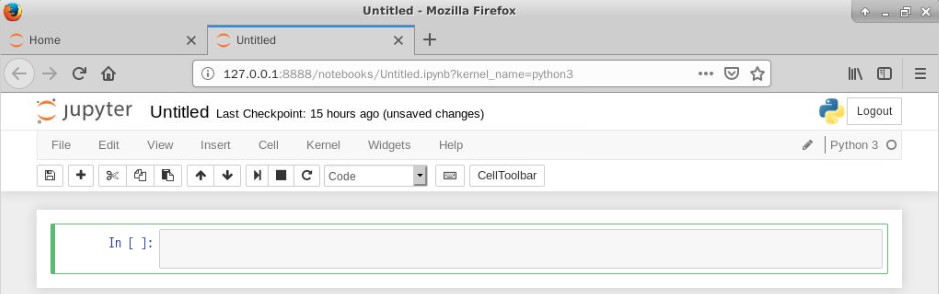
ipython notebook --ip='127.0.0.1'

Copy



点击New，在其下拉框中选择Python3，新建一个ipynb文件，用于编写并执行代码。





3.导入numpy，取别名为np并打印其版本号

import numpy as np

print(np.\_\_version\_\_) # print numpy model's version

# 创建从0到9的一维数字数组

arr = np.arange(10)

print(arr)

Copy

点击运行按钮：



执行结果如下：



4.创建一个Numpy数组元素值全为True(真)的数组。

np.full((3, 3), True, dtype=bool)

Copy

or

np.ones((3,3), dtype=bool)

Copy

执行结果如下：（要求：在/data/目录下，创建文件check，并将此步骤的运行结果保存至check中。）



5.用numpy数组中的另一个值替换满足条件的元素项，将arr中的所有奇数替换为-1。

arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

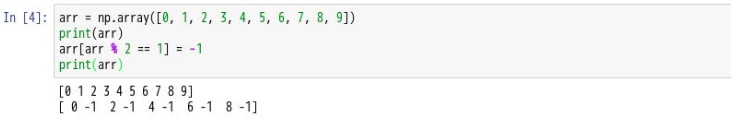
print(arr)

arr[arr % 2 == 1] = -1

print(arr)

Copy

执行结果如下：



6.将arr中的所有奇数替换成-1，而不改变arr。

arr = np.arange(10)

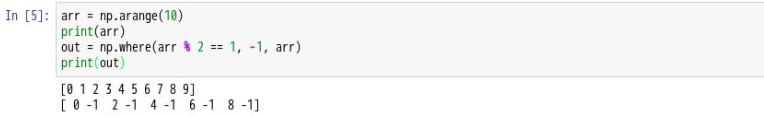
print(arr)

out = np.where(arr % 2 == 1, -1, arr)

print(out)

Copy

执行结果如下：



7.垂直叠加和水平叠加两个形状相同的数组。

a = np.arange(10).reshape(2,-1)

b = np.repeat(1, 10).reshape(2,-1)

# Method 1:

np.concatenate([a, b], axis=0)

# Method 2:

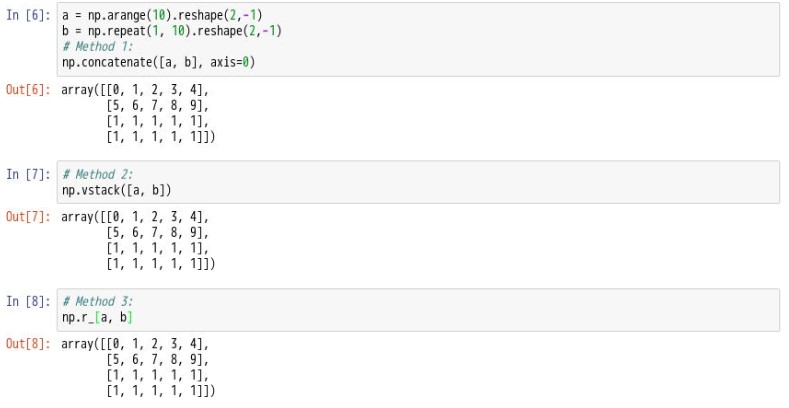
np.vstack([a, b])

# Method 3:

np.r\_[a, b]

Copy

执行结果如下：



8.水平叠加两个数组。

a = np.arange(10).reshape(2,-1)

b = np.repeat(1, 10).reshape(2,-1)

# Method 1:

np.concatenate([a, b], axis=1)

# Method 2:

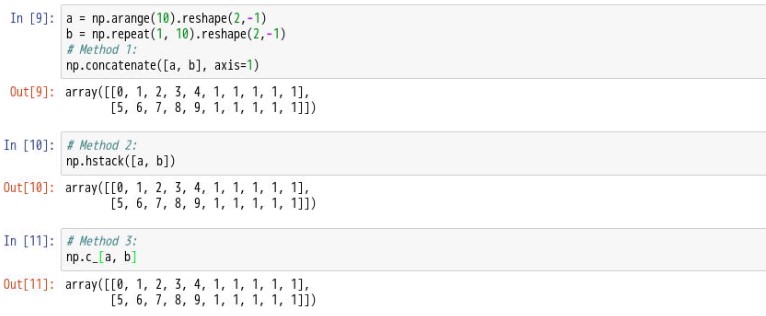
np.hstack([a, b])

# Method 3:

np.c\_[a, b]

Copy

执行结果如下：



9.从numpy数组中提取给定范围内的所有数字，获取5到10之间的所有值。

a = np.arange(15)

# Method 1

index = np.where((a >= 5) & (a<= 10))

a[index]

# Method 2:

index = np.where(np.logical\_and(a>=5, a<=10))

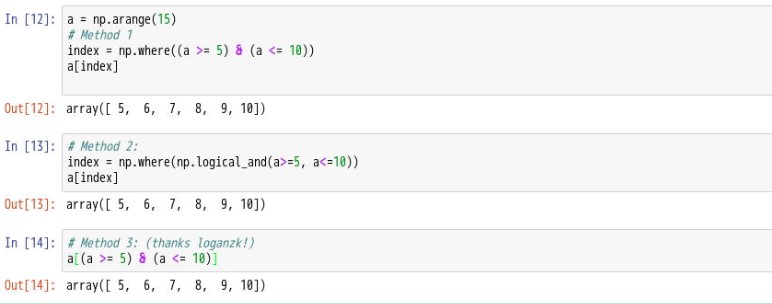
a[index]

# Method 3: (thanks loganzk!)

a[(a >= 5) & (a<= 10)]

Copy

执行结果如下：



10.创建一个形状为5x3的二维数组，以包含5到10之间的随机十进制数。（要求：在/data/目录下，创建文件check1，并将此步骤的运行结果保存至check1中。）

# Solution Method 1:

rand\_arr = np.random.randint(low=5, high=10, size=(5,3)) + np.random.random((5,3))

print(rand\_arr)

Copy

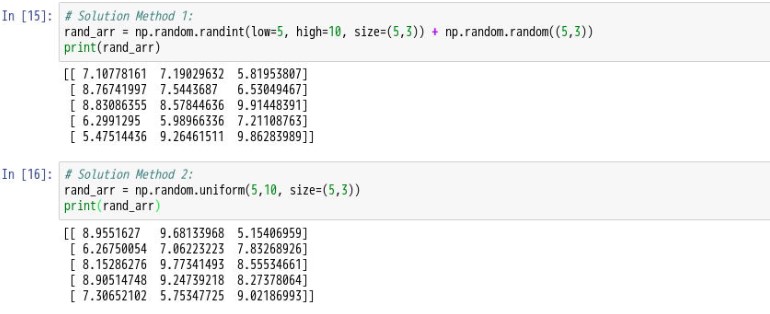
# Solution Method 2:

rand\_arr = np.random.uniform(5,10, size=(5,3))

print(rand\_arr)

Copy

执行结果如下：



11.限制numpy数组中打印的个数。将numpy数组a中打印的项数限制为最多6个元素。

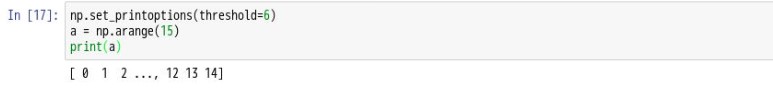
np.set\_printoptions(threshold=6)

a = np.arange(15)

print(a)

Copy

执行结果如下：



12.求出鸢尾属植物萼片长度的平均值、中位数和标准差(第1列)。

outfile = r'/data/iris.data'

sepalLength = np.loadtxt(outfile, dtype=float, delimiter=',', skiprows=1, usecols=[0])

print(sepalLength[0:10])

print(np.mean(sepalLength))

print(np.median(sepalLength))

print(np.std(sepalLength))

Copy

执行结果如下：



13.创建一种标准化形式的鸢尾属植物萼片长度，其值正好介于0和1之间，这样最小值为0，最大值为1（第1列，sepallength）。

sepalLength = np.loadtxt(outfile, dtype=float, delimiter=',', skiprows=1, usecols=[0])

# Method1

aMax = np.amax(sepalLength)

aMin = np.amin(sepalLength)

x = (sepalLength - aMin) / (aMax - aMin)

print(x[0:10])

Copy

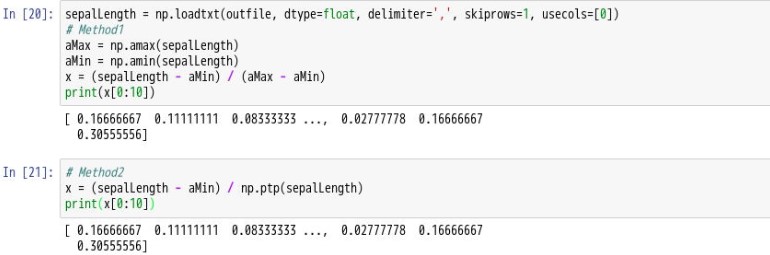
# Method2

x = (sepalLength - aMin) / np.ptp(sepalLength)

print(x[0:10])

Copy

执行结果如下：



14.在数组中随机位置插入值，例子：在iris\_2d数据集中的20个随机位置插入np.nan值。

# Method1

iris\_data = np.loadtxt(outfile, dtype=object, delimiter=',', skiprows=1)

i, j = iris\_data.shape

np.random.seed(20200621)

iris\_data[np.random.randint(i, size=20), np.random.randint(j, size=20)] = np.nan

print(iris\_data[0:10])

Copy

# Method2

iris\_data = np.loadtxt(outfile, dtype=object, delimiter=',', skiprows=1)

i, j = iris\_data.shape

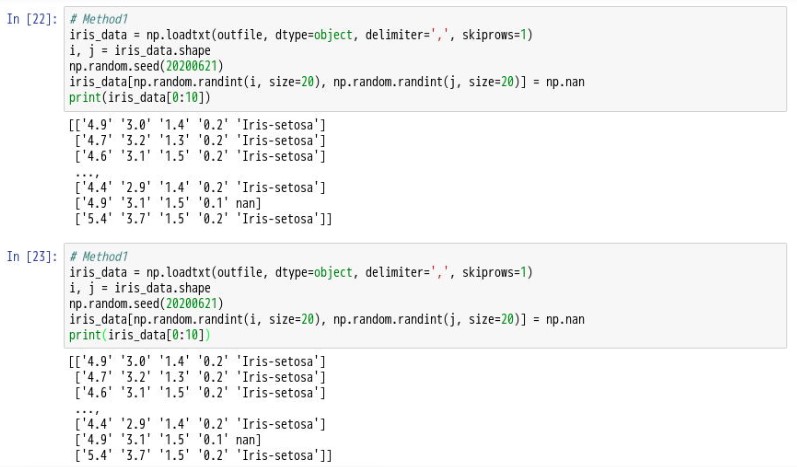
np.random.seed(20200620)

iris\_data[np.random.choice(i, size=20), np.random.choice(j, size=20)] = np.nan

print(iris\_data[0:10])

Copy

执行结果如下：



15.找出iris\_data是否有任何缺失值。

iris\_data = np.loadtxt(outfile, dtype=float, delimiter=',', skiprows=1, usecols=[0, 1, 2, 3])

x = np.isnan(iris\_data)

print(np.any(x)) # False

Copy

执行结果如下：



16.找出所有的缺失值，用0填充。

iris\_data = np.loadtxt(outfile, dtype=float, delimiter=',', skiprows=1, usecols=[0, 1, 2, 3])

i, j = iris\_data.shape

np.random.seed(20200621)

iris\_data[np.random.randint(i, size=20), np.random.randint(j, size=20)] = np.nan

iris\_data[np.isnan(iris\_data)] = 0

print(iris\_data[0:10])

Copy

执行结果如下：



使用random模块是随机值，故结果不唯一。

17.根据sepallength列对花萼长度数据集进行排序。

outfile = r'/data/iris.data'

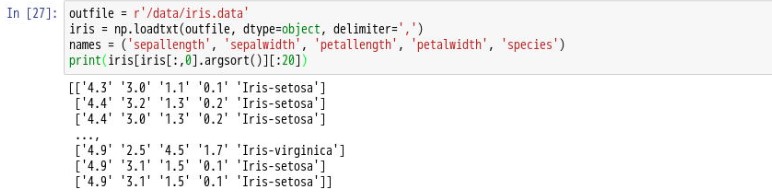
iris = np.loadtxt(outfile, dtype=object, delimiter=',')

names = ('sepallength', 'sepalwidth', 'petallength', 'petalwidth', 'species')

print(iris[iris[:,0].argsort()][:20])

Copy

执行结果如下：（要求：在/data/目录下，创建文件check2，并将此步骤的运行结果保存至check2中。）



至此，实验就结束了。