**数据的前端展示**

Highchart

1.1 引入Jquery

1.2 使用CDN引入Highchart

Jupyter

2.1 安装

pip install jupyter

2.2 启动

jupyter notebook

2.3 安装并引入highchart库

pip3 install charts

import charts

2.4 画图

charts.plot(series, options=options,show='inline')

**numpy的基本用法**

基本介绍

强大的ndarray对象和ufunc函数

精巧的函数

适合线性代数和随机数处理等科学计算

有效的通用多维数据，可定义任意数据类型

无缝对接数据库

>>> import numpy as np

>>> aArray = np.array([1,2,3])

>>> aArray

array([1, 2, 3])

>>> bArray = np.array([(1,2,3),(4,5,6)])

>>> bArray

array([[1, 2, 3],

[4, 5, 6]])

>>> np.arange(1,5,0.5)

array([ 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5])

>>> np.random.random((2,2))

array([[ 0.69540526, 0.82446662],

[ 0.16561937, 0.22484018]])

>>> np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None) # 从起始点->终止点->个数的一个等差数列，终止点默认是True包含的

array([ 1. , 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9])

>>> aArray.ndim

1

>>> bArray.ndim

2

基本属性

维度(dimensions)成为轴(axis)，轴的个数成为秩(rank)

· ndarray.ndim(秩)

· ndarray.shape(维度) —— shape元组的长度就是rank或者维度的个数ndim

· ndarray.size(元素总个数)

· ndarray.dtype(元素类型)

· ndarray.itemsize(元素字节大小)

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

>>> cArray.ndim # 秩（rank） ndim中的dim是英文dimension维度的缩写

2

>>> cArray.shape # 维度

(3, 3)

>>> cArray.size # 元素总个数

9

>>> cArray.dtype # 元素类型

dtype('int64')

>>> cArray.itemsize # 元素字节大小

8

赋固定值方法

>>> np.ones([2,3])

array([[ 1., 1., 1.],

[ 1., 1., 1.]])

>>> np.zeros((2,2))

array([[ 0., 0.],

[ 0., 0.]])

>>> np.fromfunction(lambda i,j:(i+1)\*(j+1),(9,9))

array([[ 1., 2., 3., 4., 5., 6., 7., 8., 9.],

[ 2., 4., 6., 8., 10., 12., 14., 16., 18.],

[ 3., 6., 9., 12., 15., 18., 21., 24., 27.],

[ 4., 8., 12., 16., 20., 24., 28., 32., 36.],

[ 5., 10., 15., 20., 25., 30., 35., 40., 45.],

[ 6., 12., 18., 24., 30., 36., 42., 48., 54.],

[ 7., 14., 21., 28., 35., 42., 49., 56., 63.],

[ 8., 16., 24., 32., 40., 48., 56., 64., 72.],

[ 9., 18., 27., 36., 45., 54., 63., 72., 81.]])

ndarray的操作

>>> aArray = np.array([(1,2,3),(4,5,6)])

>>> aArray

array([[1, 2, 3],

[4, 5, 6]])

>>> aArray[1]

array([4, 5, 6])

>>> aArray[0:2] # 选取第0行和第1行

array([[1, 2, 3],

[4, 5, 6]])

>>> aArray[:,[0,1]] # 选取所有行，以及第1、2列

array([[1, 2],

[4, 5]])

>>> aArray[1,[0,1]] # 选取第1行，以及第0、1列

array([4, 5])

>>> for row in aArray:

... print(row)

...

[1 2 3]

[4 5 6]

>>> aArray = np.array([(1,2,3),(4,5,6)])

>>> aArray.shape

(2, 3)

# 把新数组赋值给bArray, 但不改变aArray的维度

>>> bArray = aArray.reshape(3,2)

>>> bArray

array([[1, 2],

[3, 4],

[5, 6]])

>>> aArray.resize(3,2) # resize直接改变aArray的大小

>>> aArray

array([[1, 2],

[3, 4],

[5, 6]])

>>> bArray = np.array([1,3,7])

>>> cArray = np.array([3,5,8])

>>> np.vstack((bArray,cArray)) # 垂直方向上拼接

array([[1, 3, 7],

[3, 5, 8]])

>>> np.hstack((bArray,cArray)) # 水平方向上拼接

array([1, 3, 7, 3, 5, 8])

ndarray的运算

正常+-\*/

>>> aArray = np.array([(5,5,5),(5,5,5)])

>>> bArray = np.array([(2,2,2),(2,2,2)])

>>> cArray = aArray \* bArray

>>> cArray

array([[10, 10, 10],

[10, 10, 10]])

>>> aArray += bArray

>>> aArray

array([[7, 7, 7],

[7, 7, 7]])

广播

>>> a = np.array([1,2,3])

>>> b = np.array([[1,2,3],[4,5,6]])

>>> a + b

array([[2, 4, 6],

[5, 7, 9]])

求和

>>> aArray = np.array([(1,2,3),(4,5,6)])

>>> aArray.sum()

21

>>> aArray.sum(axis=0) # 按列求和

array([5, 7, 9])

>>> aArray.sum(axis=1) # 按行求和

array([ 6, 15])

各种值

>>> aArray.min() # 返回最小值

1

>>> aArray.argmax() # 返回最大值的index

5

>>> aArray.mean() # 返回平均值

3.5

>>> aArray.var() # 返回方差

2.9166666666666665

>>> aArray.std() # 返回标准差

1.707825127659933

线性代数

dot 矩阵内积

linalg.det 行列式

linalg.inv 逆矩阵

linalg.solve 多元一次方程组求根

linalg.eig 求特征值和特征向量

>>> x = np.array([[1,2],[3,4]])

>>> r1 = np.linalg.det(x) # 求行列式

>>> r1

-2.0000000000000004

>>> r2 = np.linalg.inv(x) # 求逆矩阵

>>> r2

array([[-2. , 1. ],

[ 1.5, -0.5]])

>>> r3 = np.dot(x,x)

>>> r3

array([[ 7, 10],

[15, 22]])

加载文件

文件，分隔符，数据类型，忽略第一行

data\_arr = np.loadtxt(data\_file, delimiter=",", dtype='str', skiprows=1)

替换字符

np.core.defchararray.replace(str, old, new, count=None)