【Python】numpy——矩阵matrix

```
import numpy as np
```

一、创建矩阵

```
import numpy as np
x = np.matrix([[1,2,3], [4,5,6]])
y = np.matrix([1,2,3,4,5,6])
```

```
In [1]: import numpy as np

        x = np.matrix([[1,2,3], [4,5,6]])
        x

Out[1]: matrix([[1, 2, 3],
                [4, 5, 6]])

In [3]: y = np.matrix([1,2,3,4,5,6])
        y

Out[3]: matrix([[1, 2, 3, 4, 5, 6]])
```


二、矩阵转置

```
>>> x = np.matrix([[1,2,3],[4,5,6]])
>>> x
matrix([[1, 2, 3],
        [4, 5, 6]])
>>> x.T
matrix([[1, 4],
        [2, 5],
        [3, 6]])
>>>
>>> y = np.matrix([1,2,3,4,5,6])
>>> y
matrix([[1, 2, 3, 4, 5, 6]])
>>> y.T
matrix([[1],
        [2],
        [3],
        [4],
        [5],
        [6]])
```


三、查看矩阵特征

1.平均值 matrix .mean()

```
import numpy as np
x = np.matrix([[1,2,3],[4,5,6]])


x.mean()              # 所有元素平均值
x.mean(axis=0)        # 纵向平均值
x.mean(axis=1)        # 横向平均值

x.mean(axis=0).shape  # 纵向平均值数组形状
```

2.求和 matrix.sum(), 最值 matrix.max()、matrix.min()

```
x.sum()

x.max(axis=1)
x.argmax(axis=1)
```

3.对角线元素 matrix.diagonal(),非0元素下标 matrix.nonzero()

```
>>> x
matrix([[1, 2, 3],
        [4, 5, 6]])
>>>
>>> x.diagonal()
matrix([[1, 5]])
>>>
>>> x.nonzero()
(array([0, 0, 0, 1, 1, 1], dtype=int64), array([0, 1, 2, 0, 1, 2], dtype=int64))
```


x.nonzero()返回非0元素下标：行下标数组和列下标数组

四、矩阵乘法

> C[i,j]=Σ第i行对应元素*第j列对应元素

```
>>> x
matrix([[1, 2, 3],
        [4, 5, 6]])
>>> y
matrix([[1, 2],
        [3, 4],
        [5, 6]])
>>> x*y
matrix([[22, 28],
        [49, 64]])
```

五、相关系数矩阵

> - 相关系数矩阵是一个对称矩阵，其中对角线上的元素都是1，表示自相关系数。
> - 非对角线元素表示互相关系数，每个元素的绝对值都小于等于1，反应变量变化趋势的相似程度。
> - 例如，如果的相关系数矩阵中非对角线元素大于0，表示两个信号正相关，其中一个信号变大时另一个信号也变大，变化方向一致，或者说-的变化对另一个信号的影响是“正面”的或者积极的。
> - 相关系数的绝对值越大，表示两个信号互相影响的程度越大。

In [24]:
```
import numpy as np
np.corrcoef([1,2,3,4], [4,3,2,1])  #负相关，变化方向相反

#    x   y
# x  1  -1
# y -1   1
```

Out[24]:
```
array([[ 1., -1.],
       [-1.,  1.]])
```

In [25]:
```
np.corrcoef([1,2,3,4], [8,3,2,1])  #负相关，变化方向相反
```

Out[25]:
```
array([[ 1.        , -0.91350028],
       [-0.91350028,  1.        ]])
```

In [26]:
```
np.corrcoef([1,2,3,4], [1,2,3,4])  #正相关，变化方向一致
```

Out[26]:
```
array([[1., 1.],
       [1., 1.]])
```

In [27]:
```
np.corrcoef([1,2,3,4], [1,2,3,40]) #正相关，变化趋势接近
```

Out[27]:
```
array([[1.        , 0.8010362],
       [0.8010362, 1.       ]])
```


六、统计函数

垂直堆叠矩阵

```
x = [-2.1, -1,  4.3]
y = [3,  1.1,  0.12]
X = np.vstack((x,y))                    # 垂直堆叠矩阵
X
```

```
array([[-2.1 , -1.  ,  4.3 ],
       [ 3.  ,  1.1 ,  0.12]])
```


1.一维(方差、标准差)

```
import numpy as np

print(np.cov([1, 1, 1, 1, 1]))          # 方差
print(np.std([1, 1, 1, 1, 1]))          # 标准差
```

```
0.0
0.0
```


2.二维(方差、标准差、协方差)

```
In [29]:
x = [-2.1, -1,  4.3]
y = [3,  1.1,  0.12]
X = np.vstack((x,y))                    # 垂直堆叠矩阵
X

Out[29]:
array([[-2.1 , -1.  ,  4.3 ],
       [ 3.  ,  1.1 ,  0.12]])

In [35]:
np.cov(X)                               # 协方差

Out[35]:
array([[11.71      , -4.286     ],
       [-4.286     ,  2.14413333]])

In [34]:
np.cov(x, y)

Out[34]:
array([[11.71      , -4.286     ],
       [-4.286     ,  2.14413333]])

In [37]:
print(np.std(X))                        # 标准差

2.2071223094538484

In [38]:
print(np.std(X, axis=1))

[2.79404128 1.19558447]

In [39]:
print(np.cov(x))                        # 方差

11.709999999999999
```