Python-numpy常用方法总结

import numpy as np

1. multiply

```
[in] x1 = [1,2,3]
     x2 = [4,5,6]
     np.multiply(x1,x2)

[out] [4,10,18]
```

multiply函数得到的结构是对应位置上面元素的乘积

2. std标准差, var方差

```
[in] b = [1,3,5,6]
     np.var(b)
     np.std(b)
     ll = [[1,2,3,4,5,6],[3,4,5,6,7,8]]
     np.var(ll[0])
     np.var(ll,0) # 第二个参数为0,表示按列求方差
     np.var(ll,1) # 。。。。。,表示按行求方差

[out] 3.6875
      1.920286436967152
      2.9166666666666665
      [1. 1. 1. 1. 1. 1. ]
      [2.916666667 2.9166666667]
```

3. mean

```
[in] b = [1,3,5,6]
     ll = [[1,2,3,4,5,6],[3,4,5,6,7,8]]
     np.mean(b)
     np.mean(ll) #求全部元素均值
     np.mean(11,0) #按列求均值
     np.mean(11,1) #按行求均值

[out] 3.75
      4.5
      [2. 3. 4. 5. 6. 7.]
      [3.5 3.5]
```

mean函数得向量的均值

4. sum

```
[in]  x = [[0,1,2],[2,1,0]]
      b = [1,3,5,6]
      np.sum(b)
      np.sum(x)

[out] 15
      6
```

sum求向量的和。也可以求矩阵所以元素的和

5. cov()

```
[in]  b = [1,3,5,6]
      print cov(b)

[out] array(4.91666667)
```

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$$

$$S = \frac{1}{N-1}\sum_{i=1}^{N} (X_i - \overline{X})^2$$

$$C = \frac{1}{N-1}\sum_{i=1}^{N} (X_i - \overline{X})(Y_i - \overline{Y})$$

cov求的是样本协方差

- 公式一 样本均值
- 公式二 样本方差
- 公式三 样本协方差

```
[in]  X = [[0,1,2],[2,1,0]]
      print cov(x)

[out]  array([[ 1., -1.],
       [-1.,  1.]])
```

cov的参数是矩阵，输出结果也是矩阵！输出的矩阵为协方差矩阵！

6. corrcoef 该函数得到相关系数矩阵

```
[in] vc = [1,2,39,0,8]
     vb = [1,2,38,0,8]
     np.corrcoef(vc,vb)

[out] [[1.          0.99998623]
       [0.99998623  1.        ]]
```

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y},$$

其中，μ是数学期望，cov表示协方差，σ_X和σ_Y是标准差。

因为$\mu_X = E(X)$, $\sigma_X^2 = E(X^2) - E^2(X)$, 同样地，对于$Y$，可以写成

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\ \sqrt{E(Y^2) - E^2(Y)}}。$$

相关系数公式：

7. vdot向量的点积

```
# vdot 返回两向量的点积
[in]  l1 = [1,2,3]
      l2 = [4,5,6]
      np.vdot(l1,l2)
[out] 32
```

对应位置相乘求和

8. mat

```
[in] l1 = [1,2,3]
     l2 = [4,5,6]
     ll = [l1,l2]
     np.vdot(l1,l2)
     np.mat(ll)

[out] 32
      [[1 2 3]
       [4 5 6]]
```

mat函数把列表转换成矩阵形式。

9. shape

```
#矩阵有一个shape属性，是一个(行,列)形式的元组
[in]  a = np.array([[1,2,3],[4,5,6]])
      a.shape

[out] (2,3)
```

10.ones

```
#返回按要求的矩阵
[in]  ones = np.ones((2,1))
      ones

[out] array([[1.],
       [1.]])
```

ones返回指定行列数的全意矩阵

11. range

```
[in]  for i in range(3):
            print i
      test = [1,2,3,4]
      test[:]
      test[2:3]

      for i in range(2,5):
        print i

[out]  0
       1
       2
       [1,2,3,4]
       [3]
       2
       3
       4
```

range用于循环中,参数为一个整数的话，可循环遍历小于该参数的值。两个参数，则循环遍历两个整数之间的值。
test[:]则表示获取test列表中的所有元素 test[2:3] 则表示获取从第2个位置到第3个位置间的元素

12.strptime

```
import time
from datetime import datetime,date

dd = datetime.strptime('2019-11-08T10:53:49.875z',"%Y-%m-%dT%H:%M:%S.%fZ")
time.mktime(dd.timetuple())

[out]1573181629.0
```

13. tuple和数组转换成字符串

```
[in]  tuple = (1,2,3)
      tuple[len(tuple)-1]
      tuple[-1]
      9.99.__repr__()
      str(9.99)

[out]  3
       3
       '9.99'
       '9.99'
```

tuple是一个元组，访问元素的时候，可以通过[index]这种方式访问。
访问最后一个元素的时候，可以通过[-1]访问.
另外，数字转换成字符串有两种方式：

```
__repr__()
str()
```

14. transpose和.T

```
[in]  aa = [[1],[2],[3]]
      aa = np.mat(aa)#将列表转换成矩阵，并存放在aa中
      aa.transpose()#将矩阵进行转置
      aa.T#将矩阵进行转置

[out]  matrix([[1],
        [2],
        [3]])
       matrix([[1, 2, 3]])
       matrix([[1, 2, 3]])
```

15. zeros() ones()

```
[in]  np.zeros((2,1))
      np.ones((2,3))

[out] [[ 0.]
      [ 0.]]
      [[ 1.  1.  1.]
      [ 1.  1.  1.]]
```

zeros返回指定行列全零矩阵
ones返回指定行列的全一矩阵

16. 列表 数组 linspace

```
#列表和数组的区别
#列表：[1,2,3,4]
#数组：[1 2 3 4]
```
数组中间元素没有分隔符,列表逗号分隔

```
[in]  np.linspace(0,3,6)
array([0. , 0.6, 1.2, 1.8, 2.4, 3. ])
```

17. argsort排序索引

```
[in] ary=np.array(np.zeros(4))
     ary[0]=0.1
     ary[1]= 0.6
     ary[2]= 0.5
     ary[3]= 0.7
     #有-号，降序排列
     #无-号，升序排列
     sortindex = np.argsort(-ary)
     for id in sortindex:
        print '索引: ',id
     for i in ary:
        print i

[out] 索引: 3
      索引: 1
      索引: 2
      索引: 0
      0.1
      0.6
      0.5
      0.7
```

18. [:,:]矩阵元素切片

```
#矩阵元素获取
[in]  ll = [[1,2,3],[4,5,6],[7,8,9]]
#获取第二行第0个元素
      np.mat(ll)[2:0]
#第一个冒号代表获取行的起止行号
#第二个冒号代表获取列的起止行号
      np.mat(ll)[:,:]

[out]  7
        matrix([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 9]])
```

19. diag

```
#构建对角矩阵
#diag()参数为列表即可
[in]  dd = [1,2,3]
      dilogg = np.diag(dd)

[out] print("diag = ",dilogg)
      diag= [[1 0 0]
      [0 2 0]
      [0 0 3]]
```

20. sorted排序

```
[in]  ll = [8,0,3,6,1,0,5,3,8,9]
      sort(ll, reverse = True) #降序
      sort(ll, reverse = False) #升序

[out] [9, 8, 8, 6, 5, 3, 3, 1, 0, 0]
      [0, 0, 1, 3, 3, 5, 6, 8, 8, 9]
```

21. random.rand

```
[in]  rr = np.random.rand(3,3)

[out] rr
        array([[0.30313948, 0.11521827, 0.78194459],
       [0.83063256, 0.21187216, 0.4652472 ],
       [0.31186867, 0.29639689, 0.97625607]])
# 获取3*3个0-1之间的数字
```

21. arange

```
[in]  delta = 0.25
      x = arange(-3.0,3.0,delta)

[out] array([-3.  , -2.75, -2.5 , -2.25, -2.  , -1.75,        -1.5 , -1.25, -1.  ,
       -0.75, -0.5 , -0.25,  0.  ,  0.25,  0.5 ,  0.75,  1.  ,  1.25,
        1.5 ,  1.75,  2.  ,  2.25,  2.5 ,  2.75])
```

22.nonzero

```
[in]  x = [[1,0,0,0,2],[0,0,3,0,0]]
      nz=np.nonzero(x)
[out] nz

     (array([0, 0, 1], dtype=int64), array([0, 4, 2], dtype=int64))
```

23. chr函数,获取指定的字符

```
[in]  for i in range(65,70):
        print(str(chr(i)))

[out] A
      B
      C
      D
      E
```