Numpy

March 24, 2017

1 Numpy

Numpy provides us with multi-dimensional array objects which is useful for scientific computing. In numpy, rank refers to the number of dimensions of the array. The shape is a tuple of integers providing the size of the array in each dimension.

[1,2,3] is a numpy object of rank 1 and shape 3

[[1,2], [4,5], [7,8]] is a numpy object of rank 3 and shape 2.

1.1 Creating a numpy object

The following provides different ways of creating numpy objects.

```
In [1]: import numpy as np
        a = np.array([1,2,3])
        print(a)
        print(a.shape)
```

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

```
In [2]: b = np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])
        print(b)
        print(b.shape)
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
(3, 4)
```

```
In [3]: print("a[1]=", a[1])
        print("b[2,3]=", b[2,3])
```

```
a[1]= 2
b[2,3]= 12
```

```
In [4]: c = np.zeros((2,3))
        print(c)

[[ 0.  0.  0.]
 [ 0.  0.  0.]]


In [5]: d = np.ones((2,3))
        print(d)

[[ 1.  1.  1.]
 [ 1.  1.  1.]]


In [6]: e = np.eye(3)
        print(e)

[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]]


In [7]: f = np.arange(20).reshape(4,5)
        print(f)

[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]
 [15 16 17 18 19]]
```

1.2 Accessing rows and columns

```
In [8]: print(f[1,:])

[5 6 7 8 9]


In [9]: print(f[:,1])

[ 1  6 11 16]


In [10]: print(f[1:3,:])

[[ 5  6  7  8  9]
 [10 11 12 13 14]]
```

1.3 Mathematical Operations

Note that in numpy, the traditional mathematical operations happen element wise and are not matrix operations like in matlab.

```
In [11]: g = np.array([[1,2,3],[4,5,6],[7,8,9]])

In [12]: h = np.array([[21,22,23],[24,25,26],[27,28,29]])

In [13]: h+g

Out[13]: array([[22, 24, 26],
                [28, 30, 32],
                [34, 36, 38]])

In [14]: h-g

Out[14]: array([[20, 20, 20],
                [20, 20, 20],
                [20, 20, 20]])

In [15]: h*g

Out[15]: array([[ 21,  44,  69],
                [ 96, 125, 156],
                [189, 224, 261]])

In [16]: h/g

Out[16]: array([[ 21.        ,  11.        ,   7.66666667],
                [  6.        ,   5.        ,   4.33333333],
                [  3.85714286,   3.5       ,   3.22222222]])
```

Note the difference between dot product and multiplication.

```
In [17]: i = np.ones((3,3))
         j = np.array([1,2,3])
         print(i*j)

[[ 1.  2.  3.]
 [ 1.  2.  3.]
 [ 1.  2.  3.]]


In [18]: print(i.dot(j))

[ 6.  6.  6.]


In [19]: print(h)
```

```
[[21 22 23]
 [24 25 26]
 [27 28 29]]
```

```
In [20]: print(np.sum(h))
```

```
225
```

```
In [21]: print(np.sum(h, axis=0))
```

```
[72 75 78]
```

```
In [22]: print(h.T)
```

```
[[21 24 27]
 [22 25 28]
 [23 26 29]]
```

For more on numpy, refer to the official numpy tutorial here