Python for Data Science
Revision and Assessment Numpy

```
import numpy as np
```

Array Preparation

```
# np.array([10,20,30,40,50])
# np.zeros((5,3))
```

```
# np.ones((5,3,6))
```

```
np.arange(2,10,0.2)
```

```
array([2. , 2.2, 2.4, 2.6, 2.8, 3. , 3.2, 3.4, 3.6, 3.8, 4. , 4.2, 4.4,
       4.6, 4.8, 5. , 5.2, 5.4, 5.6, 5.8, 6. , 6.2, 6.4, 6.6, 6.8, 7. ,
       7.2, 7.4, 7.6, 7.8, 8. , 8.2, 8.4, 8.6, 8.8, 9. , 9.2, 9.4, 9.6,
       9.8])
```

```
ar = np.linspace(2,10,60)
```

```
ar.size
```

```
60
```

```
ar.shape = (12,5,1,1)
```

```
ar.ndim
```

```
4
```

```
# np.random.randint(20,90,(10,2))
```

with replacement or without

```
np.random.choice(list("abcdefghij"), 10, replace=False)
```

```
array(['i', 'g', 'f', 'j', 'e', 'd', 'b', 'h', 'c', 'a'], dtype='<U1')
```

```
np.random.choice(['a','b','c'], 10, p=[0.7,0.2,0.1])
```

```
array(['b', 'a', 'a', 'a', 'a', 'c', 'b', 'a', 'a', 'a'], dtype='<U1')
```

```
ar = np.random.normal(70,2,10).round(1)
```

```
ar.shape
```

```
(10,)
```

```
# How can you convert single dim into 2 dim
```

```
ar1 = ar.reshape(-1,1)
```

```
ar1.shape
ar1.ndim
```

2

```
import matplotlib.pyplot as plt
```

```
x = np.linspace(-3,3,40)
y = x**2 + 1
```

We can use random.normal in order to prepare noise

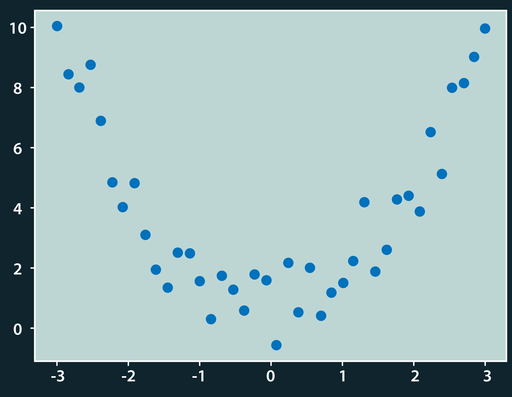
```
noise = np.random.normal(0, 0.8, 40)
```

```
yy = y + noise
```

noise prepared using random.normal can be used to disperse values

```
# plt.plot(x,y)
plt.scatter(x,yy)
```

<matplotlib.collections.PathCollection at 0x27291d740a0>




```
np.random.rand(5)
```

array([0.37379647, 0.2974247 , 0.75429811, 0.15641021, 0.70415262])

Indexing and Slicing and Axes

```
ar = np.arange(10,70).reshape(3,4,5)
```

```
ar
```

```
array([[[10, 11, 12, 13, 14],
        [15, 16, 17, 18, 19],
        [20, 21, 22, 23, 24],
        [25, 26, 27, 28, 29]],

       [[30, 31, 32, 33, 34],
        [35, 36, 37, 38, 39],
        [40, 41, 42, 43, 44],
        [45, 46, 47, 48, 49]],

       [[50, 51, 52, 53, 54],
        [55, 56, 57, 58, 59],
        [60, 61, 62, 63, 64],
        [65, 66, 67, 68, 69]]])
```

Slicing along the axis

```
ar[1][2][1:3]
```

array([41, 42])

```
ar[1,2,1:3]
```

array([41, 42])

```
ar[1][1:3]
```

array([[35, 36, 37, 38, 39],
[40, 41, 42, 43, 44]])

```
ar[1,1:3,1]
```

array([36, 41])

```
ar[1,1,3].ndim
```

0

```
ar
```

array([[[10, 11, 12, 13, 14],
[15, 16, 17, 18, 19],
[20, 21, 22, 23, 24],
[25, 26, 27, 28, 29]],

[[30, 31, 32, 33, 34],
[35, 36, 37, 38, 39],
[40, 41, 42, 43, 44],
[45, 46, 47, 48, 49]],

[[50, 51, 52, 53, 54],
[55, 56, 57, 58, 59],
[60, 61, 62, 63, 64],
[65, 66, 67, 68, 69]]])

```
arr = np.arange(24).reshape(2,3,4)
```

```
arr
```

array([[[0, 1, 2, 3],
[4, 5, 6, 7],
[8, 9, 10, 11]],

[[12, 13, 14, 15],
[16, 17, 18, 19],
[20, 21, 22, 23]]])

Summation along the axis

```
np.sum(arr, axis=0)
```

array([[12, 14, 16, 18],
[20, 22, 24, 26],
[28, 30, 32, 34]])

```
np.sum(arr, axis=1)
```

array([[12, 15, 18, 21],
[48, 51, 54, 57]])

```
np.sum(arr, axis=2)
```

array([[6, 22, 38],
[54, 70, 86]])

```
np.sum(arr, axis=1).shape
```

(2, 4)