numpy: Indexing

In [1]:

```
import numpy as np
```

In [2]:

```
a = np.random.rand(3,4,5)
a.shape
```

Out[2]:

```
(3, 4, 5)
```

What's the result of this?

In [3]:

```
a[0].shape
```

Out[3]:

```
(4, 5)
```

And this?

In [4]:

```
a[...,2].shape
```

Out[4]:

```
(3, 4)
```

In [5]:

```
a[1,0,3]
```

Out[5]:

```
0.025588609438720655
```

Like all other things in Python, numpy indexes from 0.

In [6]:

```
a[3,2,2].shape
```

```
---------------------------------------------------------------------
-------
IndexError                                Traceback (most recent cal
l last)
<ipython-input-6-4c22dfd164ed> in <module>()
      1 #keep
----> 2 a[3,2,2].shape

IndexError: index 3 is out of bounds for axis 0 with size 3
```

In [7]:

```
a[:,2].shape
```

Out[7]:

```
(3, 5)
```

Indexing into numpy arrays usually results in a so-called *view*.

In [8]:

```
a = np.zeros((4,4))
a
```

Out[8]:

```
array([[ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.]])
```

Let's call `b` the top-left 2×2 submatrix.

In [9]:

```
b = a[:2,:2]
b
```

Out[9]:

```
array([[ 0.,  0.],
       [ 0.,  0.]])
```

What happens if we change `b`?

In [10]:

```
b[1,0] = 5
b
```

Out[10]:

```
array([[ 0.,  0.],
       [ 5.,  0.]])
```

In [12]:

```
print(a)
```

```
[[ 0.  0.  0.  0.]
 [ 5.  0.  0.  0.]
 [ 0.  0.  0.  0.]
 [ 0.  0.  0.  0.]]
```

To decouple `b` from `a`, use `.copy()`.

In [13]:

```
b = b.copy()
b[1,1] = 7
print(a)
```

```
[[ 0.  0.  0.  0.]
 [ 5.  0.  0.  0.]
 [ 0.  0.  0.  0.]
 [ 0.  0.  0.  0.]]
```

You can also index with other arrays:

In [14]:

```
a = np.random.rand(4,4)
a
```

Out[14]:

```
array([[ 0.94747406,  0.89080192,  0.46799144,  0.54340544],
       [ 0.54409333,  0.27586608,  0.60682897,  0.61962813],
       [ 0.06203009,  0.7958913 ,  0.93468584,  0.88864481],
       [ 0.98627827,  0.73442815,  0.90304704,  0.18186312]])
```

In [15]:

```
i = np.array([0,2])
a[i]
```

Out[15]:

```
array([[ 0.94747406,  0.89080192,  0.46799144,  0.54340544],
       [ 0.06203009,  0.7958913 ,  0.93468584,  0.88864481]])
```

And with conditionals:

In [16]:

```
a>0.5
```

Out[16]:

```
array([[ True,  True, False,  True],
       [ True, False,  True,  True],
       [False,  True,  True,  True],
       [ True,  True,  True, False]], dtype=bool)
```

In [17]:

```
a[a>0.5]
```

Out[17]:

```
array([ 0.94747406,  0.89080192,  0.54340544,  0.54409333,  0.606828
97,
        0.61962813,  0.7958913 ,  0.93468584,  0.88864481,  0.986278
27,
        0.73442815,  0.90304704])
```