NumPy copy()

Summary: in this tutorial, you'll learn how to use the NumPy `copy()` method to create a copy of an array rather than a view.

Introduction to the NumPy copy() method

When you slice (https://www.pythontutorial.net/python-numpy/numpy-array-slicing/) an array, you get a subarray. The subarray is a view of the original array. In other words, if you change elements in the subarray, the change will be reflected in the original array. For example:

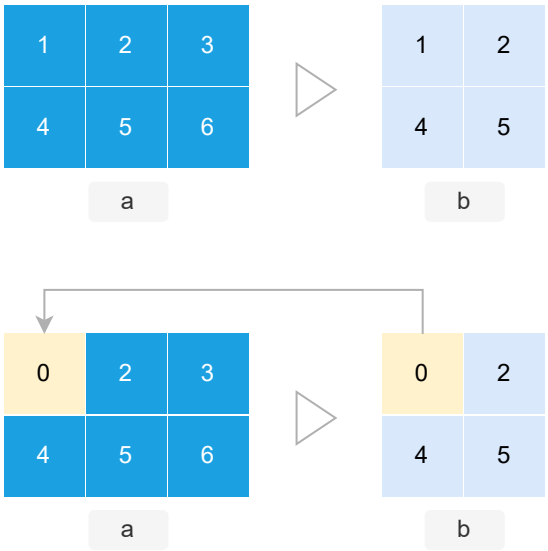
```
import numpy as np

a = np.array([
    [1, 2, 3],
    [4, 5, 6]
])

b = a[0:, 0:2]
print(b)
```

```
b[0, 0] = 0
print(b)
print(a)
```

How it works.



First, create a 2D array:

```
a = np.array([
    [1, 2, 3],
    [4, 5, 6]
])
```

Second, slice the array a and assign the subarray to the variable b:

```
b = a[0:, 0:2]
```

The variable b is:

```
[[1 2]
 [4 5]]
```

Third, change the element at index [0,0] in the subarray b to zero and display the variable b:

```
b[0, 0] = 0
print(b)
```

```
[[0 2]
 [4 5]]
```

Since b is a view of array a, the change is also reflected in array a:

```
print(a)
```

```
[[0 2 3]
 [4 5 6]]
```

The reason numpy creates a view instead of a new array is that it doesn't have to copy data therefore improving performance.

However, if you want a copy of an array rather than a view, you can use `copy()` method. For example:

```
import numpy as np

a = np.array([
    [1, 2, 3],
    [4, 5, 6]
])

# make a copy
b = a[0:, 0:2].copy()
print(b)

b[0, 0] = 0
print(b)
```

```
print(a)
```

In this example:

First, call the `copy()` method of array a to make a copy of a subarray and assign it to the variable b.

Second, change the element at index [0,0] of the array b, because both arrays are independent, the change doesn't affect array a.

Summary

- When you slice an array, you'll get a view of the array.
- Use the `copy()` method to make a copy of an array rather than a view.