NumPy Data Types

By default Python have these data types:

strings — used to represent text data, the text is given under quote marks. e.g. "ABCD"

integer — used to represent integer numbers. e.g. → -1, -2, -3

float — used to represent real numbers. e.g → 1.2, 142.42

boolean — used to represent True or False.

Complex — used to represent complex numbers. e.g. 1.0 + 2.0j, 1.5 + 2.5j

i - integer	m - timedelta
b - boolean	M - datatime
u - unsigned integer	O - object
f - float	S - string
c - complex float	U - unicode string
	V - fixed chunk of memory

Checking the Data Type of an Array

```
import numpy as np
arr = np.array([1,2,3,4])
print(arr.dtype)
```

Output
int64

Get the datatype of an array, which contain string -

```
import numpy as np
arr = np.array(['apple', 'banana', 'cherry'])
print(arr.dtype)
```

Output
<U6
Which refers to unicode string.

Creating Arrays With a Defined Data Type

We use the array() function to create arrays, this function can take an optional argument: dtype() that allows us to define the expected data type of the array.

Create an array with data type string.

```
import numpy as np
arr = np.array([1,2,3,4], dtype='S')
print(arr)
print(arr.dtype)
```

Output

```
[b'1' b'2' b'3' b'4']
|S1
```

2) import numpy as np
arr = np.array([1,2,3,4], dtype = 'i4')
print(arr)
print(arr.dtype)

Output
[1 2 3 4]
int32

Q What if value cannot be converted?

If a type is given in which elements can't be casted then Numpy will raise a ValueError.

ValueError: In Python ValueError is raised when the type of passed argument to a function is unexpected/incorrect.

import numpy as np
arr = np.array(['a','2','3'], dtype = 'i')
Output
This will be raising error.

Converting Data Type on Existing Arrays

The best way to change the data type of an existing array, is to make a copy of the array with the astype() method.

The astype() function creates a copy of the array and allows you to specify the data type as a parameter.

e.g

```
import numpy as np
arr = np.array([1.1, 2.1, 3.1])
newarr = arr.astype('i')
print(newarr)
print(newarr.dtype)
```

Output
[1 2 3]
int32

Change integer datatype to boolean

```
import numpy as np
arr = np.array([1,0,3])
newarr = arr.astype(bool)
print(newarr)
print(newarr.dtype)
```

```
[True False True]
bool
```