**Numpy**

The NumPy library is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Use the following import convention:

```python
>>> import numpy as np
```

**NumPy Arrays**

<table>
<thead>
<tr>
<th>1D array</th>
<th>2D array</th>
<th>3D array</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="1D array" /></td>
<td><img src="image2.png" alt="2D array" /></td>
<td><img src="image3.png" alt="3D array" /></td>
</tr>
</tbody>
</table>

**Creating Arrays**

```python
>>> np.array([1, 2, 3])
>>> np.array([1.0, 2.0, 3.0])
>>> np.array([1+2j, 3+4j])
```

**Array Mathematics**

**Arithmetic Operations**

```python
>>> g = a + b # Addition
>>> a = np.array([1.0, 2.0, 3.0])
>>> b = np.array([4.0, 5.0, 6.0])
>>> g = a + b
```

**Data Types**

```python
>>> np.info(np.ndarray.dtype)
>>> np.genfromtxt( , delimiter= )
>>> np.loadtxt( )
>>> np.save( , a)
>>> np.empty((3,2))
>>> np.linspace( , 7)
```

**Array Manipulation**

**Sorting Arrays**

```python
>>> a = np.array([3, 1, 2])
>>> a.sort() # Sort an array
```

**Subsetting, Slicing, Indexing**

```python
>>> a[0, 0] # Access the element of an array's axis
```

**Inspecting Your Array**

```python
>>> np.info(np.ndarray.dtype)
>>> np.array_equal(a, b) # Array equality
```

**Saving & Loading On Disk**

```python
>>> np.save('my_array.npy', a) # Save array to disk
>>> a = np.load('my_array.npy') # Load array from disk
```

**Copying Arrays**

```python
>>> h = a.copy() # Create a deep copy of the array
>>> h is a # False
>>> h is np.empty_like(a) # True
>>> h = a # shallow copy
```

---

Learn Data Skills Online at www.DataCamp.com