SANJANA DEVI

AI/ML Student | Intern @ CodroidHub Private Limited | First-Year Engineering Student.

# NumPy Basics – Beginner's Guide

By Sanjana Devi | AI/ML Student | Intern

## 1. Introduction to NumPy

NumPy (Numerical Python) is a powerful library for numerical computing. It allows for fast operations on arrays and matrices using vectorized code. - Fast and memory-efficient - Core library for scientific computing - Used widely in AI/ML, data analysis, etc.

## Why use NumPy?

Python lists are excellent, general-purpose containers. They can be “heterogeneous”, meaning that they can contain elements of a variety of types, and they are quite fast when used to perform individual operations on a handful of elements.

Depending on the characteristics of the data and the types of operations that need to be performed, other containers may be more appropriate; by exploiting these characteristics, we can improve speed, reduce memory consumption, and offer a high-level syntax for performing a variety of common processing tasks. NumPy shines when there are large quantities of “homogeneous” (same-type) data to be processed on the CPU.

## 2. Installation & Import

Use pip to install NumPy and import it with an alias:

In [1]:

```
pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\dell\anaconda3\lib\site-packages (2.1.3)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:

```python
import numpy as np
```

## 3. NumPy Arrays

Create arrays using different functions:

In [3]:

```python
np.array([1, 2, 3])
np.zeros((2, 3))
np.ones((2, 2))
np.arange(0, 10, 2)
np.linspace(0, 1, 5)
```

Out[3]:

```
array([0.  , 0.25, 0.5 , 0.75, 1.  ])
```

## 4. Array Indexing and Slicing

In [4]:

```python
arr = np.array([10, 20, 30, 40])
print(arr[0])
print(arr[-1])
print(arr[1:3])
```

```
10
40
[20 30]
```

## 5. Array Shape and Reshaping

In [ ]:
```
a = np.array([[1, 2], [3, 4]])
print(a.shape)
print(a.ndim)
print(a.reshape(1, 4))
print(a.flatten())
```

## 6. Array Operations

In [ ]:
```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
print(a + b)
print(a * b)
print(a > 2)
```

## 7. NumPy Functions

In [ ]:
```
arr = np.array([1, 2, 3, 4, 5])
print(np.sum(arr))
print(np.mean(arr))
print(np.min(arr))
print(np.max(arr))
print(np.std(arr))
```

## 8. Random Module in NumPy

In [ ]:
```
np.random.rand(2, 2)
np.random.randint(1, 10)
np.random.randn(3)
np.random.seed(42)
```

## 9. Array Manipulation

In [ ]:
```
a = np.array([1, 2])
b = np.array([3, 4])
print(np.concatenate([a, b]))
print(np.vstack([a, b]))
print(np.hstack([a, b]))
```

## 10. Copy vs View

In [ ]:
```
a = np.array([1, 2, 3])
b = a.view()
c = a.copy()
a[0] = 99
print(b)
print(c)
```

## 11. Useful NumPy Utilities

In [ ]:
```
np.unique([1, 2, 2, 3])
np.where(np.array([1, 2, 3, 4]) > 2)
np.isnan([1, np.nan])
np.isinf([1, np.inf])
```

## 12. Real-World Examples

In [ ]:

```
data = np.array([10, 20, 30, 40])
print("Mean:", np.mean(data))
print("Std Dev:", np.std(data))
```

In [ ]:

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
print(np.dot(A, B))
```

## 13. Summary Table

| Operation | Function | Example |
|---|---|---|
| Create Array | `np.array()` | `np.array([1, 2])` |
| Zeros | `np.zeros()` | `np.zeros((2,2))` |
| Random Int | `np.random.randint()` | `np.random.randint(1, 5)` |
| Mean | `np.mean()` | `np.mean(arr)` |
| Sum | `np.sum()` | `np.sum(arr)` |
| Shape | `arr.shape` | `arr.shape` |
| Reshape | `arr.reshape()` | `arr.reshape(2,3)` |
| Sort | `np.sort()` | `np.sort(arr)` |

**Connect @**
Mail (Sanjana): sanjanadevibihana@gmail.com
Contact: +91-6283762268

SANJANA DEVI

AI/ML Student | Intern @ CodroidHub Private Limited | First-Year Engineering Student.

SANJANA DEVI