**Cheatsheet**
**NumPy**

## 1. Introduction

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:

```
import numpy as np
```

## 2. Creating Arrays

A = np.array([34, 25, 16]) - One dimensional array
B = np.array([[38, 56, 43], [12, 78, 27]]) - Two dimensional array
B = np.array([[38, 56, 43], [12, 78, 27], [36, 45, 33]]) - Three dimensional array

## 3. Initial Placeholders

np.zeros(3) - 1D array of length 3 will all values as 0
np.ones((3,4)) - 3x4 array with all values 1
np.eye(5) - 5x5 array of 0 with 1 on diagonal (Identity matrix)
np.linspace(0,100,6) -Array of 6 evenly divided values from 0 to 100
np.arange(0,10,3) - Array of values from 0 to less than 10 with step 3 (eg [0,3,6,9])
np.full((2,3),8) - 2x3 array with all values 8
np.random.rand(4,5) - 4x5 array of random floats between 0 - 1
np.random.rand(6,7)*100 - 6 x 7 array of random floats between 0–100
np.random.randint(5,size=(2,3)) - 2x3 array with random ints between 0 - 4

## 4. Importing Data

np.load("newarray.npy") - Loading from disk
np.loadtxt('file.txt') - From a text file
np.genfromtxt('file.csv',delimiter=',') - From a CSV file

## 5. Exporting Data

np.save("array", a) - Writes to a disc
np.savez("array.npz", a, b) - Writes to a disc
np.savetxt('file.txt',arr,delimiter=' ') - Writes to a text file
np.savetxt('file.csv',arr,delimiter=',') - Writes to a CSV file

## 6. Inspecting Array

len(arr) - Length of an array
arr.size - Returns number of elements in *arr*
arr.shape - Return the shape of an array
arr.dtype - Returns type of elements in *arr*
arr.astype(dtype) - Convert *arr* elements to type dtype
arr.tolist() - Convert *arr* to a Python list
arr.ndim - Number of array dimensions

## 7. Operations

### Copying

np.copy(arr) - Copies arr to new memory
arr.view(dtype) - Creates view of arr elements with type dtype

### Sorting

arr.sort() - Sort an array
arr.sort(axis=0) - Sort the elements of an array's axis
np.argsort(arr) - Returns the indices of a NumPy array so that the indexed values would be sorted.

### Reshaping a array

arr.flatten() - Return a copy of the array collapsed into one dimension.
arr.T - Transposes *arr* (rows become columns and vice versa)
arr.reshape(3,4) - Reshapes *arr* to 3 rows, 4 columns without changing data
arr.resize((5,6)) - Changes *arr* shape to 5x6 and fills new values with 0
arr.ravel() - Return a contiguous flattened array

### Adding Elements

np.append(arr, values) - Append items to an array
np.insert(arr, 1, values) - Insert items in an array before 1

### Deleting Elements

np.delete(arr, 4, axis=0) - Deletes row on index 4 of arr
np.delete(arr, 3, axis=1) - Deletes column on index 3 of arr

### Combining Arrays

np.concatenate((arr1,arr2),axis=0) - Adds arr2 as rows to the end of arr1
np.concatenate((arr1,arr2),axis=1) - Adds arr2 as columns to end of arr1

### Splitting Array

np.split(arr, 4) - Splits arr into 4 sub-arrays
np.hsplit(arr, 3) - Splits arr horizontally on the 3rd index

### Indexing

arr[3] - Returns the element at index 3
arr[3,4] - Returns the 2D array element on index [3][4]
arr[2] = 4 - Assigns array element on index 2 the value 4
arr[2, 3] = 11 - Assigns array element on index [2][3] the value 11

### Slicing

arr[0:4] - Returns the elements at indices 0, 1, 2, 3 (On a 2D array: returns rows 0, 1, 2, 3)
arr[0:3, 3] - Returns the elements on rows 0, 1, 2 at column 3
arr[:3] - Returns the elements at indices 0, 1, 2
arr[:, 1] - Returns the elements at index 1 on all rows

### Subsetting

arr < 3 - Returns an array with boolean values
(arr1 < 4 ) (arr2>5) - Returns an array with boolean values
arr - Inverts a boolean array
arr[arr < 9] - Returns array elements smaller than 9

## 8. Statistics Functions

np.mean(arr,axis=0) - Returns mean along specific axis
arr.sum() - Returns sum of arr
arr.min() - Returns minimum value of arr
arr.max(axis=0) - Returns maximum value of specific axis
np.var(arr) - Returns the variance of array
np.std(arr,axis=1) - Returns the standard deviation of specific axis
arr.corrcoef() - Returns correlation coefficient of array

**9. Scalar Math**

np.add(arr, 4) - Add 4 to each array element
np.subtract(arr, 3) - Subtract 3 from each array element
np.multiply(arr, 2) - Multiply each array element by 2
np.divide(arr, 4) -Divide each array element by 4 (returns np.nan for division by zero)
np.power(arr,2) - Raise each array element to the 2nd power

---

**10. Vector Math**

np.add(arr1, arr2) - Element-wise add arr2 to arr1
np.subtract(arr1,arr2) - Element-wise subtract arr2 from arr1
np.multiply(arr1,arr2) - Element-wise multiply arr1 by arr2
np.divide(arr1,arr2) - Element-wise divide arr1 by arr2
np.power(arr1,arr2) - Element-wiseraise arr1 raised to the power of arr2
np.array _ equal(arr1,arr2) - Returns True if the arrays have the same elements and shape
np.sqrt(arr) - Square root of each element in the array
np.sin(arr) - Sine of each element in the array
np.log(arr) - Natural log of each element in the array
np.abs(arr) - Absolute value of each element in the array
np.ceil(arr) - Rounds up to the nearest int
np.floor(arr) - Rounds down to the nearest int
np.round(arr) - Rounds to the nearest int

---

## Broadcasting

**Goal:** bring arrays with different shapes into the same shape during arithmetic operations.

```
salary = np.array([2000, 4000, 8000])
salary_bump = 1.1
print(salary * salary_bump)
# [2200. 4400. 8800.]
```

- For any dimension where first array has size of one, NumPy conceptually copies its data until the size of the second array is reached.

- If dimension is completely missing for array B, it is simply copied along the missing dimension.

---