③

# Numpy

⇒ It is a multi dimentional array library.

| List | NumPy |
|---|---|
| * Very Slow | * very fast |
| → Variable type | → fixed type |
| → More memory needed for storing each element. | → Less memory needed for storing each element. |
| → NumPy uses scattered memory | → Numpy uses Contiguous memory |

⇒ Applications of NumPy?

- → MATLAB Replacement
- → Plotting (Matplotlib)
- → Backend (Pandas)

★ Initializing an array

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

→ A list

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

→ [[1, 2 3]
 [4 5 6]]

* Get Dimension

a.ndim → 1

b.ndim → 2

* Get Shape

a.shape → (3)

b.shape → (3,2)

* Get Type

a.dtype → int32

* Spacifing type while initialization

a = np.array([1,2,3], dtype='int16')

* Get Size

a.itemsize → 2 → in byte

* Get number of items

a.size → 3

b.size → 6

* Get total size

a.nbytes → 6

* Accessing /Changing specific elements, rows , columns etc...

a = np.array([[1,2,3,4,5,6,7],[8,9,10,11,12,13,14]])

* a[1,5] ⇒ 13

* a[0,:] ⇒ [1,2,3,4,5,6,7]

* a[:,2] ⇒ [3,10]

* a[0, 1:6:2] → [2,4,6]

1 → Start
6 → Stop
2 → Step

## ★ Initializing different types of Arrays

**np.zeros((2,3))** → shape

```
[[0 0 0]
 [0 0 0]]
```

**np.ones((2,3))**

```
[[1 , 1 , 1],
 [1 ,1  1]]
```

**np.full ((2,3), 27)**

```
[[27 ,27 ,27],
 [27 ,27 , 27]]
```

**np.random.rand(2,3)** → two parameter

```
[[0.8 ,0.7  , 0.3]
 [0.1 , 0.2  ,0.34]]
```

→ random number between 0 & 1

**np.random.random_sample((2,3))** → shape

**np.random.randint(4, 8, size=(2,3))** → End → Start

```
[[6  5  4],
 [4  7  5]]
```

np.identity(3)

```
→ [[1 0 0],
   [0 1 0],
   [0 0 1]]
```

arr = np.array([[1,2,3]])

r1 = np.repeat ~~concat~~ (arr, 3, axis=0)

```
→ [[1,2,3]
   [1,2,3]
   [1,2,3]]
```

⇒ Be careful when copying array

b = a.copy()

★ Mathematics (basic)

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

a+2

→ [3,4,5,6]

a*2

→ [2,4,6,8]

a ** 2

→ [1,4,9,16]

★ **Linear Algebra**

np.matmul(a,b)

↳ Matrix multiplication of a&b

np.linalg.det(a)

↳ Determinant of a

★ **Statistics**

np.min(a)

↳ Smallest element in array a

np.max(a)

↳ Largest element in array a

⇒ Axis can be added to above function as an argument.

np.Sum(a)

↳ Sum of all element in a

## ★ Reorganizing Arrays

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

b = np.reshape((2,3))
      \
       -> [[1,2],
           [3,4],
           [5,6]]

v1 = np.array([1,2,3,4])
v2 = np.array([[5,6,7,8])

np.vstack([v1 , v2])
      \
       -> [[1,2,3,4],
           [5,6,7,8]]

np.hstack((v1,v2))
      \
       -> [1,2,3,4,5,6,7,8]
```

## ★ Load data from file

```
a = np.genfromtext('data.txt', delimiter=',')
```

→ This will load all data as float

⇒ To change dtype.

```
a = a.astype('int32')
```

## * Boolean masking & advanced indexing

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

a>=3
  → [[False , False True],
     [True , True ,True]]

a[a>=3]
  → [3,4,5,6]

np.any(a>5, axis=0)
  → [False True True]

np.all (a>=3, axis=0)
  → [False, False, True)
```

$$a = \begin{bmatrix} 1 & \boxed{2} & \boxed{3} \\ 4 & 5 & 6 \\ 7 & \boxed{8} & \boxed{9} \end{bmatrix}$$

```
a [[0,2],1:]
  → [2 3
     8 9]
```