

# Tổng quan về thư viện Numpy

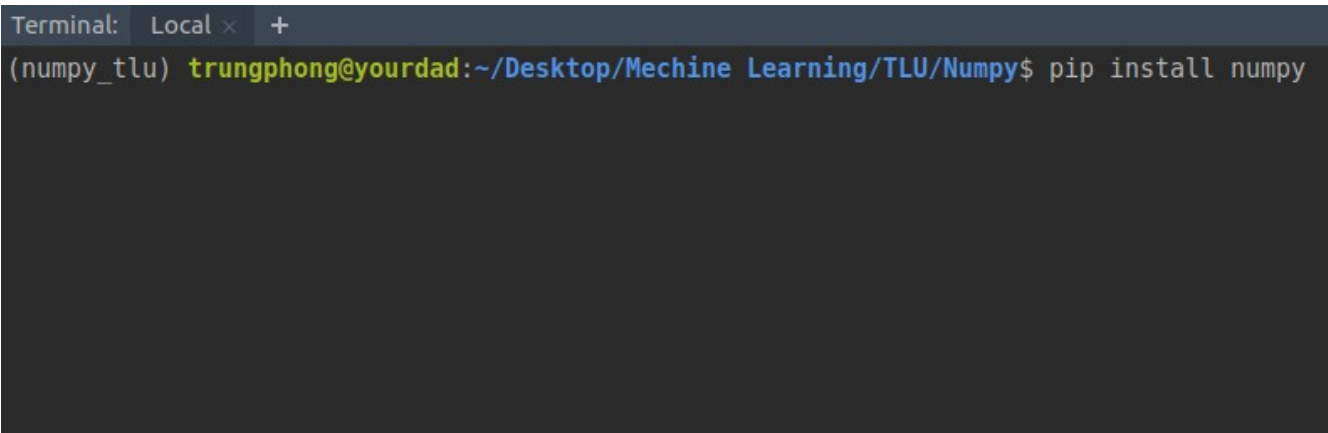
## (Vietnamese Language)

### 1. Giới thiệu (Introduction):

- Numpy(NP) là một mô-đun mã nguồn mở cho python giúp bạn rất hữu ích khi tính toán một ma trận có kích thước lớn với nhau và tối ưu hóa được thời gian tính toán hay những thao tác lên mảng, ma trận.
- NP được áp dụng tính toán trong môn đại số tuyến tính và đặc biệt là ứng dụng vào Machine Learning(ML).

### 2. Cách cài đặt numpy (Install numpy):

- **pip install numpy**



```
Terminal: Local x +  
(numpy_tlu) trungphong@yourdad:~/Desktop/Machine Learning/TLU/Numpy$ pip install numpy
```

Đây là terminal của phần mềm pycharm đã được activate môi trường. Hiện tại mình đang install numpy.

- **import numpy as np** (Khai báo để ta có thể bắt đầu sử dụng các hàm số của NP) .

```
import numpy as np|
```

Ở đây numpy đã được import vào và giờ chỉ cần sử dụng nó dưới dạng viết tắt np. để gọi ra các hàm.

- Lưu ý:

- + Cài đặt môi trường và activate môi trường (Hướng dẫn cài đặt môi trường sẽ có trong một bài viết khác).
- + Kiểm tra xem trong môi trường cài đặt đã có NP hay chưa.

### 3. Các hàm và cách sử dụng của thư viện numpy (Using function in numpy) :

Ở đây chúng ta sẽ tập chung nói đến những hàm thường dùng nhất trong ML. Mình sẽ giải thích từng tham số quan trọng được truyền vào và kiểu dữ liệu trả về của nó.

- **np.array(object, dtype=None, copy=True, order='K', subok=False, ndmin=0)**

- **object**: là một mảng, có thể là mảng hoặc chuỗi lồng nhau.

- **dtype** : kiểu dữ liệu cho mảng

- Kiểu trả về sẽ là một ndarray thỏa mãn các tham số truyền vào (Return).

**Ví dụ:** Ta sẽ khai báo một mảng biến một mảng bình thường thành 1 ndarray.

- Ta sẽ dùng x ở đây là một ndarray với các giá trị là lần lượt là 1,2,3

```
x = np.array([1,2,3],dtype=int)
print(x)
print(type(x))
```

- Ở đây ta sẽ in ra giá trị của x và kiểu dữ liệu của nó.

```
(numpy_tlu) trungphong@yourdad:~/Desktop/Machine Learning/TLU/Numpy$ python3 main.py
[1 2 3]
<class 'numpy.ndarray'>
```

- Như đã thấy thì kiểu dữ liệu của biến x này là 1 numpy.ndarray

- Khởi tạo mảng toàn giá trị 1 hoặc 0:

**np.ones(shape, dtype=None, order='C')**

- **shape** : ở đây là kích thước của mảng.

**Ví dụ** : 2,(2,3)....

- **dtype** : kiểu dữ liệu của mảng.

- Kiểu trả sẽ là ndarray với tất cả phần tử là 1.

**np.zeros(shape, dtype=float, order='C')**

- Tương tự giống với **np.ones( )** tuy nhiên tất cả phần tử của nó sẽ là 0.

**Ví dụ:** Tạo một vector ngang với kích thước là 1x5 và giá trị từng phần tử của nó sẽ là 1.

```
x = np.ones(5,dtype=int)
print(x)
print(type(x))
```

```
[1 1 1 1 1]
<class 'numpy.ndarray'>
```

**Ví dụ:** Tạo một ma trận với kích thước 2x3 với toàn phần tử 0.

```
x = np.zeros((2,3), dtype=int)
print(x)
print(type(x))
```

```
[[0 0 0]
 [0 0 0]]
<class 'numpy.ndarray'>
```

- **np.arange(start=0, stop, step=1, dtype=None)**

- Hàm này tạo ra một ma trận với cấp số cộng với phần tử bắt đầu từ start → end-1 với bước nhảy sẽ là step. có thể hiểu đơn giản start=start+step với (start <= end-1)

- **start** : giá trị bắt đầu của hàm số cộng (Có thể hiểu start là giá trị chặn dưới của hàm).

- **stop** : giá trị kết thúc của hàm số cộng (Có thể hiểu là giá trị chặn trên của hàm).

- **step** : cấp số cộng với số trong dãy hiện tại tạo ra số tiếp theo.

Sau đây là những ví dụ về mảng cấp số cộng. Lưu ý nếu hàm chỉ truyền một tham số thì nó sẽ là **stop** và chạy từ 0 → **stop** - 1. Nếu 2 tham số sẽ là **start** và **stop**.

**Ví dụ:** Tạo ra một mảng cấp số cộng từ 0 → 5.

```
x = np.arange(5, dtype=int)
print(x)
print(type(x))
```

```
[0 1 2 3 4]
<class 'numpy.ndarray'>
```

**Ví dụ**: Tạo một mảng cấp số cộng từ 1 → 10 với cấp số cộng là 2.

```
x = np.arange(start=1, stop=10, step=2, dtype=int)
print(x)
print(type(x))
```

```
[1 3 5 7 9]
<class 'numpy.ndarray'>
```

- `np.loadtxt(fname, dtype=<class'float'>, delimiter=None)`
  - Hàm này dùng để đọc dữ liệu từ file text.
  - **fname** : Đường dẫn đến file.
  - **delimiter** : cách để nhận biết hai số khác nhau trên cùng một dòng được ngăn cách nhau bởi dấu gì.
  - Kiểu dữ liệu trả về là một mảng ndarray từ file được đọc.

**Ví dụ**: Đọc dữ liệu từ một file dữ liệu trên một dòng ngăn cách nhau bởi dấu phẩy.

```
data = np.loadtxt(fname = path, delimiter=",")
print(data)
print(type(data))
```

path ở đây là đường dẫn đến file data1.txt

```
[[1. 2.]
 [3. 4.]
 [5. 6.]]
<class 'numpy.ndarray'>
```

```
1, 2
3, 4
5 ,6
```

Các giá trị trong file text

- **np.dot(a, b, out=None)**
  - Đây là một hàm phép nhân giữa hai ma trận a và b.
  - **a, b** : là một ma trận hoặc vector.
  - Nếu **a** vs **b** là một vector, kết quả trả về sẽ trả về tích vô hướng của chúng.
  - Nếu **a** vs **b** là một ma trận hai chiều, kết quả trả về sẽ là phép nhân giữa hai ma trận.
  - Nếu **a** là một ma trận N chiều và **b** là 1 vector, kết quả trả về sẽ là tổng cột cuối của a giữa b.
  - Nếu **a** là một ma trận N chiều và **b** là một ma trận M thì kết quả trả về của nó sẽ được tính: **dot(a,b) [i,j,k,m] = sum(a[i, j, :] \* b[k, :, m])**.

**Ví dụ1:** Nhân hai vector với nhau.

```
X = np.dot([1, 2], [3, 4])  
print(X)
```

```
11
```

**Ví dụ2:** Nhân hai ma trận.

```
A = [[1, 0], [0, 1]]  
B = [[3, 4], [5, 6]]  
  
X = np.dot(A, B)  
print(X)
```

```
[[3 4]  
 [5 6]]
```

- **np.eye(N, M=None, k=0, dtype=<class 'float'>, order='C')**
  - Tạo ra một ma trận đơn vị với kích thước là **N** x **M**.
  - **N** số hàng của ma trận đơn vị được tạo ra.
  - **M** là số cột của ma trận đơn vị được tạo ra, nếu không nhập **M** thì mặc định **M=N**.
  - **k** là vị trí cột bắt đầu của ma trận. Mặc định nó sẽ tại vị trí 0.

Ví dụ1: Tạo một ma trận đơn vị với kích thước 5x5.

```
X = np.eye(5)
print(X)
```

```
[[1.  0.  0.  0.  0.]
 [0.  1.  0.  0.  0.]
 [0.  0.  1.  0.  0.]
 [0.  0.  0.  1.  0.]
 [0.  0.  0.  0.  1.]
```

Ví dụ2: Tạo một ma trận đơn vị với index bắt đầu là 2.

```
X = np.eye(5, k=2, dtype=int)
print(X)
```

```
[[0 0 1 0 0]
 [0 0 0 1 0]
 [0 0 0 0 1]
 [0 0 0 0 0]
 [0 0 0 0 0]]
```

- **np.diag(v, k=0)**

- Trả về một vector với các phần tử là đường chéo chính, trong một số trường hợp sẽ là đường chéo của các phần tử khác với ma trận là 2-D. Nếu là một vector nó sẽ trả về một ma trận 2-D với các phần tử của vector lần lượt nằm trên đường chéo chính của ma trận.

- **v** : là tham số ma trận hoặc vector truyền vào.

- **k** : là vị trí bắt đầu của đường chéo, mặc định nó sẽ là 0.



Ví dụ1: Ta lấy đường chéo của 1 mảng 2-D .

```
data = np.loadtxt(fname = _path_, delimiter=",", dtype=int)
# data
# 1, 2, 3
# 4, 5, 6
# 7, 8, 9

X = np.diag(data)
print(X)
```

```
[1 5 9]
```

Ví dụ2: Ta biến một vector thành đường chéo của một ma trận 2-D.

```
data = np.loadtxt(fname = _path_, delimiter=",", dtype=int)
# data
# 1, 2, 3
X = np.diag(data)
print(X)
```

```
[[1 0 0]
 [0 2 0]
 [0 0 3]]
```

- **np.transpose(a, axes=None)**
  - Đây là hàm giúp chúng ta chuyển vị ma trận.
  - **a** : truyền vào một ma trận hoặc vector.
  - **axes** : Mặc định là thay đổi kích thước ma trận, nếu không nó sẽ hoán vị các cột với các giá trị đã cho.

**Ví dụ**: Hoán vị một ma trận 2x3.

```
data = np.loadtxt(fname=_path,delimiter=",",dtype=int)
# data
# 1, 2, 3
# 4, 5, 6
X = np.transpose(data)
print(X)
```

```
[[1 4]
 [2 5]
 [3 6]]
```

- **np.reshape(a, newshape, order='C')**

- Hàm này giúp chúng ta thay đổi kích thước của mảng mà không ảnh hưởng đến giá trị của các phần tử trong nó.
- **a** : Ma trận hoặc vector được truyền vào.
- **newshape**: điều chỉnh kích thước lại theo yêu cầu.

Ví dụ1: Biến một vector có kích thước 1 x 6 thành ma trận 3 x 2.

```
data = np.loadtxt(fname=_path,delimiter=",",dtype=int)
# data
# 1, 2, 3, 4, 5, 6
X = np.reshape(data, (3, 2))
print(X)
```

```
[[1 2]
 [3 4]
 [5 6]]
```

Ví dụ2: Biến một ma trận 2x3 thành một vector 1x6.

```
data = np.loadtxt(fname=_path,delimiter=",",dtype=int)
# data
# 1, 2, 3
# 4, 5, 6
X = np.reshape(data, 6)
print(X)
```

```
[1 2 3 4 5 6]
```

Sẽ có sự thiếu sót trong bản này và mong các bạn góp ý kiến thêm hoặc có gì chưa đúng mong được nhận được ý kiến phản hồi từ các bạn.

#### 4. Nguồn tham khảo (Reference source):

- Khóa học free trên youtube của **TheEngineeringWorld** (Course tutorial free of **TheEngineeringWorld**) : <https://bitly.vn/3vgn>
- Trang web về numpy (You can refer to search numpy in website): <https://www.numpy.org>  
(<https://docs.scipy.org/doc/numpy/reference/generated/>)
- Machine Learning cơ bản của anh Vũ Khắc Tiệp nói về numpy: <https://bitly.vn/3vii>

**Mình sẽ bổ sung thêm những gì cần thiết hơn nữa ở phiên bản sau rất mong sự góp ý của bạn**