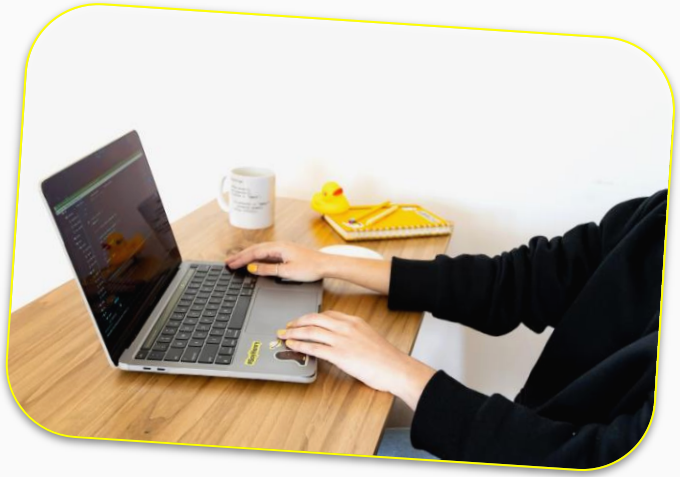
HENRY

Numpy

Data Science

Agenda



- Arrays de Numpy
- Operaciones con Arrays
- Matrices en Numpy
- Estadística con Numpy
- Máscaras

OBJETIVOS DE LA CLASE

Al finalizar esta lecture estarás en la capacidad de...

- Conocer las ventajas del uso de la librería Numpy para manejo de Arrays y Matrices en Python

Numpy

¿Qué es?

NumPy (Numerical Python) es una librería numérica de Python.

Es base de todos los cálculos científicos.

Es de código abierto, proporciona estructuras de datos matriciales y funciones matemáticas de alto nivel.

Arrays de **Numpy**

¿Qué son?

En **NumPy** se trabaja con una estructura de datos llamada array o arreglos numéricos multidimensionales.

✓ Parecidos a las listas de Python, heredan algunas propiedades como el ser mutables y poder realizar slicing.

✗ Tienen diferencias importantes: son menos pesados, más rápidos y permiten crear fácilmente arrays de N dimensiones.

Tipos de arrays

- Un array unidimensional puede ser una fila o una columna de una tabla, igual que una lista, esta se conoce como vector.
- Un array bidimensional es lo que llamamos comúnmente matriz.
- Un array de 3 dimensiones o más, es decir, una matriz de matrices, se conoce como tensor.

Crear arrays

A partir de una lista

```
>>> my_list = [0, 1, 2, 3, 4]
>>> print(np.array(my_list))
[0 1 2 3 4]
```

A partir de secuencias

```
>>> print(np.arange(start=2, stop=10, step=2))
[2 4 6 8]
```

```
>>> print(np.linspace(0, 1, 11))
[0.  0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1. ]
```

Crear arrays

Predefinidos

```
>>> print(np.zeros(4))
[0. 0. 0. 0.]
```

```
>>> print(np.ones(6))
[1. 1. 1. 1. 1. 1.]
```

```
>>> print(np.full(shape=(2, 2), fill_value=5))
[[5 5]
 [5 5]]
```

```
>>> base = np.linspace(2, 6, 4)
>>> print(np.full_like(base, np.pi))
[3.14159265 3.14159265 3.14159265 3.14159265]
```

Crear arrays

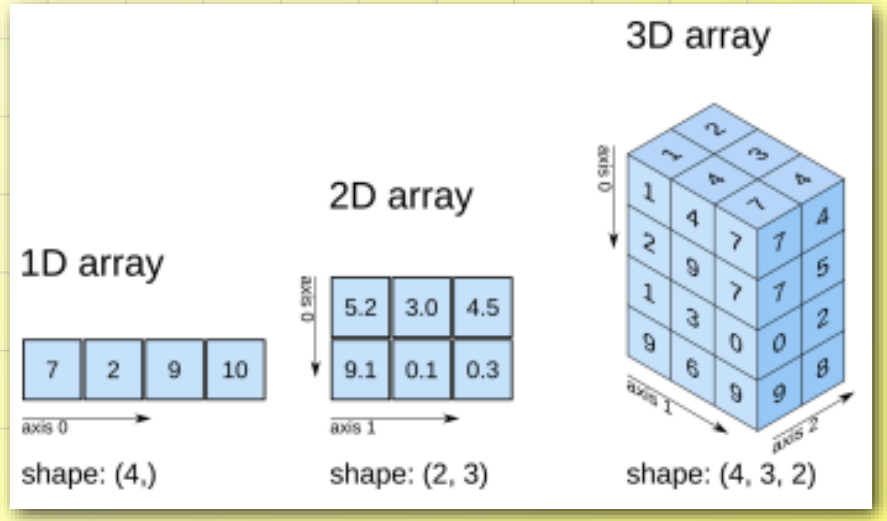
Aleatorios

```
>>> print(np.random.rand(2, 2))
[[0.62740202 0.11171536]
 [0.47526728 0.19739417]]
>>>
>>> print(np.random.uniform(low=0, high=1, size=6))
[0.7878737  0.3431897  0.77765595 0.60943181 0.30961326 0.60167083]
 >>>
>>> print(np.random.randn(2, 2))
[[ 0.91140011  1.72792052]
 [-0.84028707 -0.27378577]]
>>>
>>> print(np.random.normal(loc=0, scale=2, size=6))
[-2.36743682 -3.12673482 -1.14254395 -3.19805542 -1.11930443 -2.70161226]
```

Tamaño de arrays

```
>>> B = np.reshape(a, [3,3])
>>> print(B)
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

```
>>> a = np.arange(1,10)
>>> print(a)
[1 2 3 4 5 6 7 8 9]
```



Slicing con arrays

```
>>> matrix_cool = np.arange(9).reshape(3, 3)
>>> print(matrix_cool)
[[0 1 2]
 [3 4 5]
 [6 7 8]]
>>> print(matrix_cool[1, 2])
5
>>> print(matrix_cool[0, :])
[0 1 2]
```

```
>>> print(matrix_cool[:, 1])
[1 4 7]
>>> print(matrix_cool[:, 1:])
[[1 2]
 [4 5]
 [7 8]]
>>> print(matrix_cool[0:2, 0:2])
[[0 1]
 [3 4]]
```

Copiar arrays

```
>>> a1 = np.array([2, 4, 6])
>>> a2 = a1.copy()
>>> a1[0] = 8
>>> print(a1)
>>> print(a2)
[8 4 6]
[2 4 6]
```

Operaciones con arrays

Suma

```
>>> A = np.arange(5, 11)
>>> print(A)
[ 5  6  7  8  9 10]
>>> print(A + 10)
[15 16 17 18 19 20]
```

Resta

```
>>> B = np.full(4, 3)
>>> C = np.ones(4, dtype='int')
>>> print(B)
[3 3 3 3]
>>> print(C)
[1 1 1 1]
>>> print(B - C)
[2 2 2 2]
```

Operaciones con arrays

Shape (forma)

```
>>> a = np.array([[2,3],[2,3],[2,3]])
>>> a.shape
(3, 2)
>>> b = np.array([[1,6,5,2,7],[1,2,7,0,9]])
>>> b.shape
(2, 5)
```

Multiplicación

```
>>> np.matmul(a,b)
array([[ 5, 18, 31,  4, 41],
       [ 5, 18, 31,  4, 41],
       [ 5, 18, 31,  4, 41]])
```

Trasposición

```
array([[ 5,  5,  5],
       [18, 18, 18],
       [31, 31, 31],
       [ 4,  4,  4],
       [41, 41, 41]])
```

Operaciones con arrays

```
>>> # Aritmetica
>>> a = np.arange(4)
>>>
>>> print("a     =", a)
a     = [0 1 2 3]
>>> print("a + 5 =", a + 5)
a + 5 = [5 6 7 8]
>>> print("a - 5 =", a - 5)
a - 5 = [-5 -4 -3 -2]
>>> print("a * 2 =", a * 2)
a * 2 = [0 2 4 6]
```

Otros ejemplos

```
>>> print("a / 2 =", a / 2)
a / 2 = [ 0.   0.5  1.   1.5]
>>> print("a // 2 =", a // 2)
a // 2 = [0 0 1 1]
>>> print("-a     = ", -a)
-a     =  [ 0 -1 -2 -3]
>>> print("a ** 2 = ", a ** 2)
a ** 2 =  [0 1 4 9]
>>> print("a % 2  = ", a % 2)
a % 2  =  [0 1 0 1]
```

Estadística

```
>>> height_list = [74, 74, 72, 72, 73, 69, 69, 71, 76, 71, 73, 73, 74, 74, 69, 70, 73, 75, 78, 79, 76, 74, 76, 72, 71, 75]
>>> print(np.mean(height_list))
73.1923076923077
>>> print(np.median(height_list))
73.0
>>> print(np.std(height_list))
2.572326554954764
>>> print(np.percentile(height_list,90))
76.0
```

Otros ejemplos

```
>>> a = np.arange(4)
>>> b = np.arange(1,5)
>>>
>>> display(np.exp(a))          # exponencial
array([ 1.        ,  2.71828183,  7.3890561 , 20.08553692])
>>> display(np.log(b))          # logaritmo natural
array([ 0.        ,  0.69314718,  1.09861229,  1.38629436])
>>> display(np.sqrt(a))         # raiz cuadrada
array([ 0.        ,  1.        ,  1.41421356,  1.73205081])
>>> display(np.greater(a,b))    # superior o igual punto a punto
array([False, False, False, False], dtype=bool)
```

Máscaras

```
>>> a = np.arange(0,20).reshape(2,10)
>>> print(a)
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]]
>>> mascara = ((a % 2) == 0)
>>> print(mascara)
[[ True False  True False  True False  True False  True False]
 [ True False  True False  True False  True False  True False]]
>>> a[mascara]
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18])
```


HENRY