Universidad de los Andes Colombia | Departamento de Ingeniería de Sistemas y Computación

Introducción a Numpy

Introducción

NumPy es una librería (o módulo) para el lenguaje de programación Python que permite acceder a funciones y estructuras de datos **optimizadas para el procesamiento de vectores y matrices**, junto con una gran colección de funciones matemáticas de alto nivel para operar en estos tipos de datos. Más información acerca de esta librería puedes ser encontrada aquí.

La información de imágenes satelitales es almacenada en matrices multidimensionales de máximo 3 dimensiones (latitud, longitud y tiempo) cada una. En este notebook hacemos una breve introducción al entendimiento y manipulación de matrices multidimensionales.

Contenido

1. Importar la librería numpy

Para poder hacer uso de la bondades que brinda de la librería `numpy` debemos importarla usando la palabra especial `import` .

NOTA: Además de `numpy` , `import` permite importar otro tipo de librerías como `panda` y `sklearn` que permiten realizar ciencia de datos y inteligencia artificial respectivamente.

In [5]:
```
import numpy
```

Otra alternativa para importar una librería es usar el operador `as` para hacer un renombramiento. De esta forma, cada que desee usar algúna funcionalidad de la librería `numpy` , puedo referirme en su lugar a `np`

In [8]:
```
import numpy as np
```

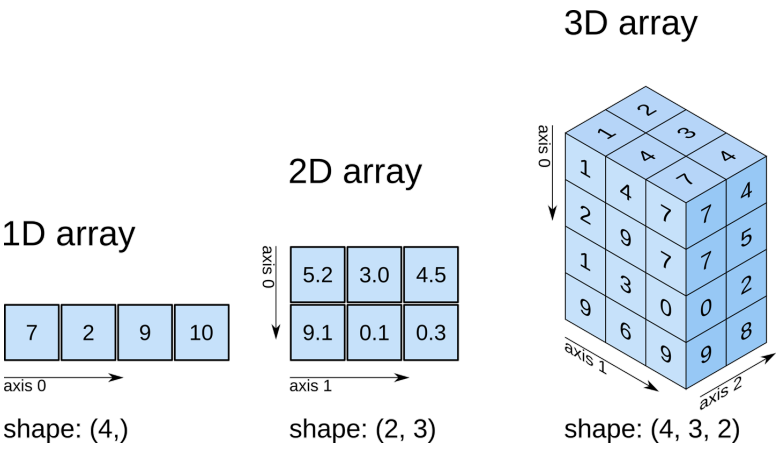
Ahora podemos acceder a todas las funcionalidades de la librería numpy usando `numpy.name_of_function` . Donde `name_of_function` es una de las funciones definidas en la librería `numpy` . Por ejemplo, usando la funcionalidad `__version__` podemos ver qué versión de la librería se encuentra cargada.

In [10]:
```
np.__version__
```

Out[10]: `'1.19.4'`

2. Arreglos

Los arreglos son colecciones de datos que pueden ser organizados en una o varias dimensiones. Por ejemplo, la información de imágenes satelitales se organiza en arreglos de 3 dimensiones: tiempo, longitud y latitud. La imagen que se muestra a continuación describe las caracterśiticas de los arreglos de 1, 2 y 3 dimensiones.



Definición de arreglos

Para crear un arreglo, usamos la funcionalidad `array` de la librería `numpy` como se muestra a continuación.

In [28]:
```
array_1d = np.array([1,2,3,4])
array_1d
```

Out[28]:
```
array([1, 2, 3, 4])
```

Los arreglos tienen la propiedad `shape` que permite validar cómo están organizados los datos en el arreglo. Es decir, permite ver sus dimensiones.

In [20]:
```
# Arreglo de 1 dimensión con 10 elementos
array_1d.shape
```

Out[20]:
```
(10,)
```

Ejemplo de un arreglo de 2 dimensiones

In [25]:
```
# Arreglo de 2 dimensiones con 3 files y 4 columnas
array_2d = np.array(
    [[1,2,3,4],
     [1,2,3,4],
     [1,2,3,4]]
)

array_2d
```

Out[25]:
```
array([[1, 2, 3, 4],
       [1, 2, 3, 4],
       [1, 2, 3, 4]])
```

Para validar la forma del arreglo de dos dimensiones usamos la propiedad `shape`

In [26]:
```
array_2d.shape
```

Out[26]:
```
(3, 4)
```

Ejemplo de un arreglo de 3 dimensiones

In [27]:
```
array_3d = np.array(
    [[[1,2,3,4],
      [1,2,3,4],
      [1,2,3,4]],

     [[5,6,7,8],
      [5,6,7,8],
      [5,6,7,8]]]
)

array_3d
```

Out[27]:
```
array([[[1, 2, 3, 4],
        [1, 2, 3, 4],
        [1, 2, 3, 4]],

       [[5, 6, 7, 8],
        [5, 6, 7, 8],
        [5, 6, 7, 8]]])
```

Para validar la forma del arreglo de tres dimensiones usamos la propiedad `shape`

In [19]:
```
array_3d.shape
```

Out[19]:
```
(2, 3, 4)
```

Operaciones aritméticas

Los arreglos creados con la librería numpy soportan una gran variedad de operaciones aritméticas. A continuación se muestra algúnas de estas operaciones. Los operaciones son realizadas elemento por elemento.

Operador	Descripción	Ejemplo
+	Suma	a + b
-	Resta	a - b
-	Negación	-a
*	Multiplicación	a * b
**	Exponente	a ** 2
/	División	a / b
//	División entera	a // b
%	Módulo	a % b

Ejemplos arreglos `a` y `b`

In [38]:
```
a = np.array(
    [[1,2,3],
     [4,5,6],
     [7,8,9]]
)

b = np.array(
    [[1,2,3],
     [4,5,6],
     [7,8,9]]
)

print('Areglo a')
print(a)
print('Areglo b')
print(b)
```

```
Areglo a
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Areglo b
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Ejemplo operación de suma (+)

In [49]:
```
# Suma de dos matrices.
a % b
```

Out[49]:
```
array([[0, 0, 0],
       [0, 0, 0],
       [0, 0, 0]])
```

> **TODO:** Realice los ejemplos de los operadores resta (`-`), multiplicación (`*`), exponente (`**`), división (`/`),

> división entera (`//`) y módulo (`%`) usando las matrices `a` y `b` definidas. Cree una nueva celda para cada ejemplo. Puede apoyarse en cualquiera de los ejemplo mostrados arriba.

Otras operaciones

Los arreglos creados en base a la librería `numpy` presentan una serie de funciones (operaciones) que permiten reducir la información contenida en los arreglos. Los operaciones básicas generan una estadística sobre todos los datos del arreglo (Ejemplo 1). Por otro lado, es posible indicar si el cálculo se desea realizar a lo largo de un eje (`axis`) en arreglos multi-dimensionales (Ejemplo 2).

Operador	Descripción	Ejemplo 1	Ejemplo 2
mean	Promedio	a.mean()	a.mean(axis=0)
std	Desvisación estándar	a.std()	a.std(axis=0)
sum	Suma	a.sum()	a.sum(axis=0)

Ejemplo operación (`sum`)

In [57]:
```
a.sum()
```

Out[57]:
```
45
```

In [55]:
```
a.sum(axis=0)
```

Out[55]:
```
array([12, 15, 18])
```

> **TODO:** Realice los ejemplos (Ejemplo 1 y Ejemplo 2) de los operadores desviación estandar (`std`) y promedio (`mean`). Cree una nueva celda para cada ejemplo. Puede apoyarse en cualquiera de los ejemplo mostrados arriba.