Universidad Nacional de San Agustín de Arequipa

Escuela Profesional de Ingeniería de Telecomunicaciones

Ingeniero Renzo Bolivar - Docente DAIE

Curso : Computación 2

NUMPY

Descripción:

En este notebook se explicarán *NumPy* la librería de cálculo numérico uno de los paquetes centrales para la computación numérica en Python.

Objetivos:

1. Familiarizarce con las matrices en NumPy
2. Creación de matrices en Numpy
3. Comprobar la facilidad de uso en NumPy

Contenido:

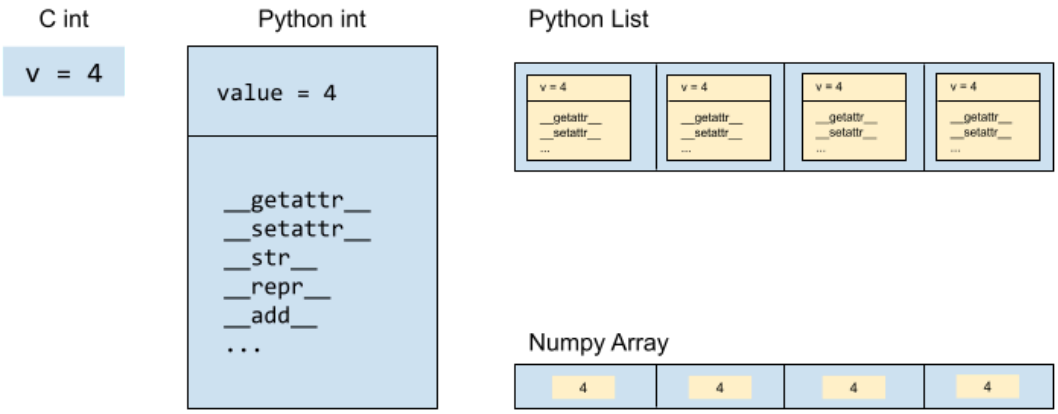
1. Numpy: Librería de cálculo numérico

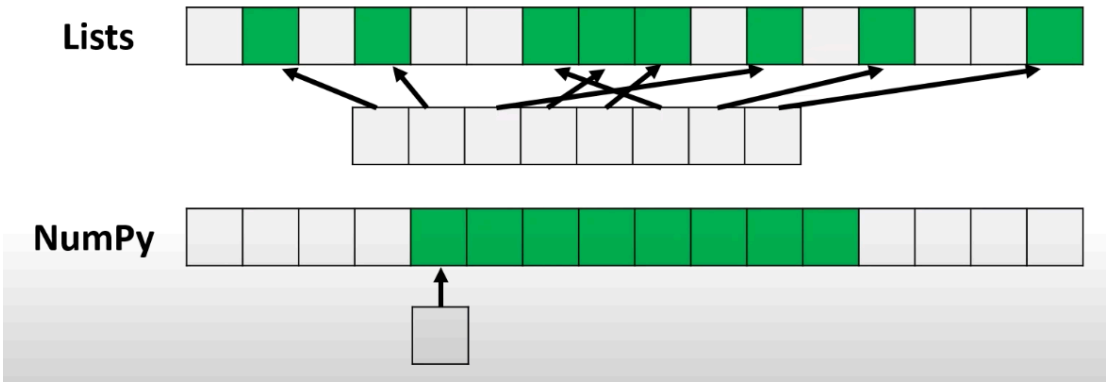
NumPy (Numerical Python) es uno de los paquetes centrales para la computación numérica en Python. Pandas, Matplotlib, Statmodels y muchas otras librerías científicas dependen de NumPy.

Las principales contribuciones de NumPy son:

- Computación numérica eficiente con código "C"
- Recopilaciones eficientes con operaciones vectoriales
- Un Aplicativo (API) de álgebra lineal integrada y natural
- Un Aplicativo (API) en C para conectar NumPy con librerías escritas en C, C++ o FORTRAN.

Desarrollemos la eficiencia. En Python, **todo es un objeto**, lo que significa que incluso los simples **enteros (int)** son también objetos, con toda la maquinaria necesaria para hacer que el objeto funcione. Los llamamos "Boxed Ints". En cambio, `NumPy` utiliza tipos numéricos primitivos (floats, ints) que hacen que el almacenamiento y el cálculo sean eficientes.



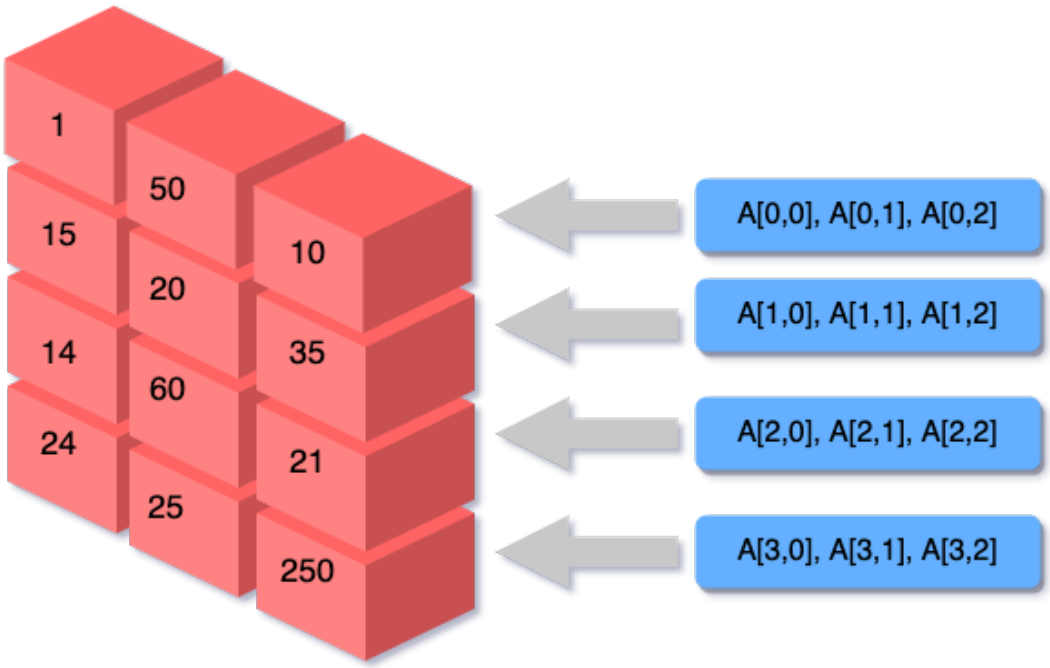


Utilización de NumPy:

- En Matemáticas (Reemplaza al MATLAB)
- Genera Gráficos a partir de Matrices Numpy
- Backend (Pandas, Connect 4 Games, Fotografía Digital)
- Machine Learning.

NumPy Array o el arreglo de matrices de NumPy es un potente objeto de matriz N-dimensional que tiene forma de filas y columnas, en la que tenemos varios elementos que están almacenados en sus respectivas ubicaciones de memoria.

En la siguiente imagen, es una **matriz bidimensiona**l porque tiene filas y columnas, como puedes ver tiene cuatro filas y tres columnas, por lo que se convierte en una matriz bidimensional. En el caso de que solo tuviera una fila entonces habría sido una matriz unidimensional.



2. Numpy: Funciones básicas

Matriz unidimensional

Con la primera instrucción `import numpy as np` le estamos indicando a nuestro programa de Python que de ahora en adelante `np` será la referencia para todo lo referente a NumPy.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Matriz bidimensional

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Porque usar NumPy

Porque usar NumPy en vez de utilizar las listas propias que ofrece Python para manejar estos datos, la primera razón es que NumPy ocupa menos memoria en comparación a las lista de Python, a su vez es bastante rápido en términos de ejecución. Lo que se observa en el siguiente ejemplo:

Cantidad de memoria asignada

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Velocidad

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Matrices Vacias

En ocasiones se requiere crear matrices vacías, esto se refiere a que se requieren marcadores de posición iniciales, que luego pueden ser rellenados. Se puede inicializar matrices con unos o ceros, pero también puedes hacer matrices que se llenan con valores espaciados uniformemente, valores constantes o aleatorios.

Algunas de las instrucciones para crear este tipo de matrices son las siguientes:

Crear una matriz en donde todos los valores sean igual a 1

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
# Crear una matriz de unos - 3 filas 4 columnas
```

Crear una matriz en donde todos los valores sean igual a 0

In [ ]:
```
# Crear una matriz de ceros - 3 filas 4 columnas
```

Crear una matriz de números aleatorios

In [ ]:
```
# Crear una matriz con valores aleatorios
```

Crear una matriz vacía

In [ ]:
```
# Crear una matriz vacía
```

Crear una matriz que contenga un solo valor en todas las posiciones

In [ ]:
```
# Crear una matriz con un solo valor
```

Crear una matriz con rangos de números

In [ ]:
```
# Crear una matriz con valores espaciados uniformemente
```

In [ ]:
```

```

Crear una matriz identidad

In [ ]:
```
# Crear una matriz identidad
```

3. Inspeccionar matrices de NumPy

Dimensión de una matriz

In [ ]:
```
# Conocer las dimensiones de una matriz
```

Tipo de Datos de una matriz

In [ ]:
```
# Conocer el tipo de los datos
```

Tamaño y forma de una matriz

In [ ]:
```
# Conocer el tamaño y forma de la matriz
```

Cambio de forma de una matriz

In [ ]:
```
# Cambio de forma de una matriz
```

Seleccionar elemento de una matriz

In [ ]:
```
# Extraer un solo valor de la matriz - el valor ubicado en la fila 0 colu
```

In [ ]:
```
# Extraer los valores de todas las filas ubicados en la columna 3
```

4. Operaciones Matemáticas entre matrices de NumPy

Valor mínimo, máximo y la suma de una matriz

In [ ]:
```
# Encontrar el mínimo, máximo y la suma
```

Raiz Cuadrada y Desviación de una matriz

In [ ]:
```
# Calcular la raíz cuadrada y la desviación estándar
```

Suma, resta, multiplicación y división de dos matrices

In [ ]:
```
# Calcular la suma, resta, multiplicación y división de dos matrices
```

EJEMPLOS

___Ejemplo 1:___ Cree una matriz de 5x5 con valores 1,2,3,4 justo debajo de la diagonal:

In [ ]:
```
Z = np.diag(1+np.arange(4),k=-1)
print(Z)
```

___Ejemplo 2:___ Crea un vector aleatorio de tamaño 30 y encuentra el valor medio :

In [ ]:
```
Z = np.random.random(30)
m = Z.mean()
print(m)
```

___Ejemplo 3:___ ¿Cómo ordenar una matriz por la enésima columna?:

In [ ]:
```
Z = np.random.randint(0,10,(3,3))
print(Z)
print(Z[Z[:,1].argsort()])
```

___Ejemplo 4:___ ¿Cómo agregar un borde (lleno de 0) alrededor de una matriz existente?

In [ ]:
```
Z = np.ones((5,5))
Z = np.pad(Z, pad_width=1, mode='constant', constant_values=0)
print(Z)
```

EJERCICIOS

Ejercicio 01

- Crear un array 7x8 lleno de ceros de tipo entero.
- Crear un array 7x8 lleno de ceros salvo la primera fila que serán todo unos.
- Crear un array 7x8 lleno de ceros salvo la última fila que será el rango entre 5 y 8.

Ejercicio 02

- Crea un vector de 10 elementos, siendo los impares 1 y los pares 2.
- Crea un «tablero de ajedrez» 8x8, con 1 en las casillas negras y 0 en las casillas blancas.

Ejercicio 03

- Crea una matriz aleatoria 7x7 y halla los valores mínimo y máximo.
- Normaliza la matriz anterior entre 0 y 1.

Ejercicio 04

De la siguiente Matriz

1	1	1	1	1
1	0	0	0	1
1	0	9	0	1
1	0	0	0	1
1	1	1	1	1

- Luego invertir lo valores de 9 y 1 de la Matriz anterior

Ejercicio 05

De la siguiente Matriz:

- Extraer la matrices de cada color

Ejercicio 06

De la siguiente Matriz:

255	34	45	16	24	99
200	50	67	15	56	88
165	67	34	26	87	70
120	100	45	45	56	67
45	225	35	67	50	44
36	255	24	78	77	23

0	0	0
0	255	0
0	0	0

Se debe **buscar dentro de la matriz de color verde el valor = 50**, luego reemplazar con la pequeña matriz de color celeste quedando la matriz de la siguiente manera:

0	0	0	16	24	99
0	255	0	15	56	88
0	0	0	26	87	70
120	100	45	0	0	0
45	225	35	0	255	0
36	255	24	0	0	0

El algoritmo solución debe poder **ubicar el valor = 50** en cualquier posición y hacer el reemplazo.

BIBLIOGRAFÍA

[1] Documentación de NumPy http://docs.scipy.org/doc/numpy

[2] Travis Oliphant, "Guide to NumPy"
http://csc.ucdavis.edu/~chaos/courses/nlp/Software/NumPyBook.pdf

[3] SciPy Lecture Notes http://scipy-lectures.github.io

[4] Nicolas Rougier, "100 NumPy exercises"
http://www.loria.fr/~rougier/teaching/numpy.100/index.html