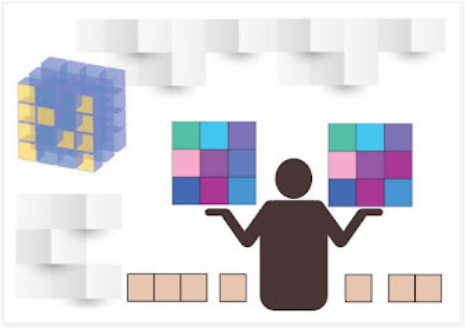
Python 3 para impacientes

"Simple es mejor que complejo" (Tim Peters)

viernes, 27 de diciembre de 2019

Comparar arrays en Numpy



En Numpy hay funciones que facilitan la tarea de comparar los elementos de un array con un valor o con los valores de otro array para saber si son mayores, iguales, menores, distintos, etc., para verificar si todos los valores o alguno son **True** (o tienen un valor distinto de **0**), para obtener los valores máximos o mínimos y funciones para evaluar el resultado de operaciones a nivel binario.

Funciones para comparar arrays

greater(), less(), equal(), not_equal(), ...

Estos métodos comparan los elementos de un array con un valor o con los elementos de otro array, elemento a elemento, y devuelven un array con valores booleanos (**True** y False) según sea el resultado de la comparación: mayor que, mayor o igual que, menor que, menor o igual que, igual que o distinto con los métodos **greater()**, **greater_equal()**, **less()**, **less_equal()**, **equal()** y **not_equal()**, respectivamente.

```
# Devuelve True si el elemento del array a es mayor que 2.

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

# [False False  True]

# Devuelve True si el elemento del array a es menor o
# igual que el del array b.

a = np.array([1, 2, 3])
b = np.array([0, 2, 7])
c = np.less_equal(a, b)
print(c)

# [False  True  True]

# Devuelve True si el elemento del array es igual que el del array b.

a = np.array([1, 2, 3])
b = np.array([0, 2, 7])
c = np.equal(a, b)
print(c)

# [False  True False]
```

all()

Buscar

Buscar

Python para impacientes

Python
IPython
EasyGUI
Tkinter
JupyterLab
Numpy

Anexos

Guía urgente de MySQL
Guía rápida de SQLite3

Entradas + populares

Dar color a las salidas en la consola
En Python para dar color a las salidas en la consola (o en la terminal de texto) existen varias posibilidades. Hay un método basado ...

Instalación de Python, paso a paso
Instalación de Python 3.6 A finales de 2016 se produjo el lanzamiento de Python 3.6 . El propósito de esta entrada es mostrar, pas...

Variables de control en Tkinter
Variables de control Las variables de control son objetos especiales que se asocian a los widgets para almacenar sus valore...

Añadir, consultar, modificar y suprimir elementos en Numpy
Acceder a los elementos de un array. [], [,], ... Acceder a un elemento de un array. Para acceder a un elemento se utiliz...

Cálculo con arrays Numpy
Numpy ofrece todo lo necesario para obtener un buen rendimiento cuando se trata de hacer cálculos con arrays. Por como está concebid...

Operaciones con fechas y horas. Calendarios
Los módulos datetime y calendar amplían las posibilidades del módulo time que provee funciones para manipular expresiones de ti...

Tkinter: interfaces gráficas en Python
Introducción Con Python hay muchas posibilidades para programar una interfaz gráfica de usuario (GUI) pero Tkinter es fácil d...

Convertir, copiar, ordenar, unir y dividir arrays Numpy
Esta entrada trata sobre algunos métodos que se utilizan en Numpy para convertir listas en arrays y viceversa; para copiar arrays d...

Tkinter: Tipos de ventanas
Ventanas de aplicación y de diálogos En la entrada anterior tratamos los distintos gestores de geometría que se utilizan para di...

El módulo random
El módulo random de la librería estándar de Python incluye un conjunto de funciones

Devuelve **True** si todos los elementos del array son **True** o tienen valores que son distinto de **0**.

```
a = np.array([True, True, False])
estado = a.all()
print(estado)

# False

b = np.array([5, 4, 4])
estado = b.all()
print(estado)

# True
```

any()

Devuelve **True** si alguno de los elementos del array es **True** o un valor distinto de **0**.

```
a = np.array([True, True, False])
estado = a.any()
print(estado)

# True

b = np.array([0, 0, 0])
estado = b.any()
print(estado)

# False
```

Funciones para comparar arrays y obtener máximos o mínimos

maximum()

Obtener valores máximos.

```
# Comparar dos arrays y obtener valores máximos.

a = np.array([0, 9, 3, 7])
b = np.array([0, 2, 7, 4])
c = np.maximum(a, b)
print(c)

# [0 9 7 7]

# Comparar un array con un valor y obtener valores máximos.

a = np.array([0, 9, 3, 7])
c = np.maximum(a, 5)
print(c)

# [5 9 5 7]
```

minimum()

Obtener valores mínimos.

```
# Comparar dos arrays y obtener valores mínimos.

a = np.array([0, 9, 3, 7])
b = np.array([0, 2, 7, 4])
c = np.minimum(a, b)
print(c)

# [0 2 3 4]

# Comparar un array con valor y obtener valores mínimos.

a = np.array([0, 9, 3, 7])
c = np.minimum(a, 5)
print(c)
```

que permiten obtener de distintos modos números a...

```
# [0 5 3 5]
```

Funciones para evaluar operaciones a nivel binario

logical_and(), logical_or(), logical_xor(), logical_not()

Estos métodos operan a nivel binario los elementos de un array con un valor o con los elementos de otro array, elemento a elemento, y devuelven un array con valores booleanos (**True** y **False**) según sea el resultado de la operación **0** o distinto de **0** con los métodos **logical_and()**, **logical_or()**, **logical_xor()**, **logical_not()**, entre otros.

```
# Devuelve False si el resultado del producto lógico (AND) es 0.

a = np.array([0, 0, 1, 0])
b = np.array([0, 0, 1, 0])
c = np.logical_and(a, b)
print(c)

# [False False  True False]

# Devuelve False si el resultado de la suma lógica (OR) es 0.

a = np.array([0, 2, 3, 7])
b = np.array([0, 2, 7, 4])
c = np.logical_or(a, b)
print(c)

# [False  True  True  True]

# Operación:
# 000 010 011 111
# 000 010 111 100
# --- --- --- ---
# 000 010 111 111
```

Publicado por Pherkad en 4:26

Etiquetas: Numpy

Entrada más reciente Inicio Entrada antigua