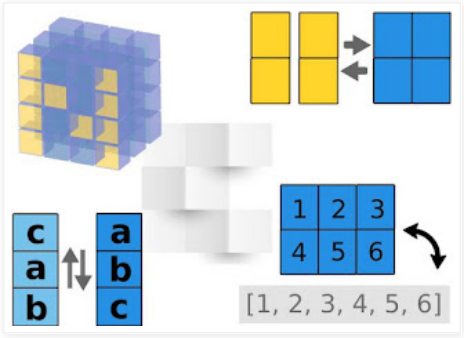
Python 3 para impacientes

"Simple es mejor que complejo" (Tim Peters)

sábado, 16 de noviembre de 2019

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 diferenciando las copias por valor y por referencia; para ordenar arrays de diferentes dimensiones con algunas opciones avanzadas y, finalmente, métodos para unir varios arrays en uno o dividir uno en varios.

Convertir

asarray()

Convertir listas en arrays Numpy.

```
# Convertir dos listas a arrays con distintas dimensiones:

lista1 = [1, 2, 3, 4, 5]
lista2 = [[1, 2, 3], [4, 5, 6]]
a = np.asarray(lista1)
b = np.asarray(lista2)
print(a)

# [1 2 3 4 5]

print(b)

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

tolist()

Convertir arrays en listas Python.

```
# Convertir dos arrays con dimensiones diferentes a listas:

a = np.array([1, 2, 3, 4, 5])
b = np.array([[1, 2, 3], [4, 5, 6]])
lista1 = a.tolist()
lista2 = b.tolist()
print(lista1)

# [1, 2, 3, 4, 5]

print(lista2)
```

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

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

Copiar arrays

copy()

Copiar un array (por valor).

Crea una copia de un array en otro área de la memoria.

```
a = np.array([1, 2, 3, 4, 5])
copia = a.copy()
print(copia)

# [1 2 3 4 5]
```

array2 = array1

Copiar un array (por referencia).

La copia por referencia se realiza asignando la variable de un array a otra. Después de la asignación ambas variables compartirán los mismos datos en el mismo área de memoria así como los cambios que se produzcan.

```
a = np.array([[1, 2, 3], [4, 5, 6]])
b = a
print(b)

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

a[0, 1] = -1

print(a[0, 1])

# -1

print(b[0, 1])

# -1
```

Ordenar arrays

sort()

Ordenar arrays de distintas dimensiones y por diferentes criterios.

```
# Ordenar un vector de modo ascendente:

a = np.array([5, 3, 1, 4, 2])
a.sort()
print(a)

# [1 2 3 4 5]

# Ordenar un vector de modo descendente:

a = np.array([5, 3, 1, 4, 2])
a[::-1].sort()
print(a)

# [5 4 3 2 1]

# Ordenar un array bidimensional por filas (eje 0):

a = np.array([[1, 2, 1], [3, 1, 2], [2, 3, 3]])
a.sort(axis=0)
print(a)

# [[1 1 1]
#  [2 2 2]
#  [3 3 3]]
```

que permiten obtener de distintos modos números a...

Archivo

noviembre 2019 (3)

python.org

pypi.org

Sitios

- ActivePython
- Anaconda
- Bpython
- Django
- Flask
- Ipython
- IronPython
- Matplotlib
- MicroPython
- Numpy
- Pandas
- Pillow
- PortablePython
- PyBrain
- PyCharm
- PyDev
- PyGame
- Pypi
- PyPy
- Pyramid
- Python.org
- PyTorch
- SciPy.org
- Spyder
- Tensorflow
- TurboGears

```
# Ordenar un array bidimensional por columnas (eje 1):

a = np.array([[1, 2, 1], [3, 1, 2], [2, 3, 3]])
a.sort(axis=1)
print(a)

# [[1 1 2]
#  [1 2 3]
#  [2 3 3]]

# Ordenar un array 2D por filas en modo descendente (eje 0):

a = np.array([[1, 2, 1], [3, 1, 2], [2, 3, 3]])
a[::-1, :].sort(axis=0)
print(a)

# [[3 3 3]
#  [2 2 2]
#  [1 1 1]]

# Ordenar un array 3D por (eje 2):

a = np.array([[[4, 2], [3, 1]], [[5, 0], [0, 1]]])
print(a)

# [[[4 2]
#   [3 1]]
#
#  [[5 0]
#   [0 1]]]

a.sort(axis=2)
print(a)

# [[[2 4]
#   [1 3]]
#
#  [[0 5]
#   [0 1]]]

# Ordenar un array por uno de sus campos:

tipo = [('id', int), ('nombre', 'S10')]
valores = [(2, 'Marta'), (3, 'Pablo'), (1, 'Carmen')]
a = np.array(valores, dtype=tipo)
print(a)

# [(2, b'Marta') (3, b'Pablo') (1, b'Carmen')]

a.sort(order='id')
print(a)

[(1, b'Carmen') (2, b'Marta') (3, b'Pablo')]
```

argsort()

Ordenar un array y obtener los índices de los elementos.

```
# Ordenar un vector y obtener los índices de sus elementos:

a = np.array([5, 3, 1, 4, 2])
indices = np.argsort(a)
print(indices)

# [2 4 1 3 0]

# Ordenar un array 2D por filas (eje 0) y obtener los índices:

a = np.array([[1, 4, 3], [2, 3, 6]])
indices = np.argsort(a)
print(indices)

# [[0 2 1]
#  [0 1 2]]
```

partition()

Reorganizar elementos de un array.

Reorganizar los elementos menores a la izquierda del elemento seleccionado y el resto a la derecha.

```
a = np.array([9, 8, 7, 6, 5, 4, 3, 2])
a.partition(3)  # Hace referencia a 6
print(a)

# [3 2 4 5 6 8 7 9]
```

argpartition()

Reorganizar los índices de un array.

Obtener los índices de los elementos si se situaran los menores a la izquierda del elemento seleccionado y a la derecha el resto.

```
a = np.array([9, 8, 7, 6, 5, 4, 3, 2])
indices = a.argpartition(3)  # Hace referencia a 6
print(indices)

# [6 7 5 4 3 1 2 0]
```

searchsorted()

Obtener de un array ordenado el índice donde después de insertar un valor seguiría manteniéndose el orden dentro dicho array.

```
a = np.array([1,3,6,7,8,9])
indice = a.searchsorted(2)
print(indice)

# 1
```

Obtener de un array ordenado los índices donde después de insertar los valores de una lista seguiría manteniéndose el orden dentro de dicho array.

```
a = np.array([1,3,6,7,8,9])
indices = a.searchsorted([5, 2])
print(indices)

# [2 1]
```

Unir y dividir arrays

concatenate()

Concatenar o unir varios arrays en uno.

```
# Concatenar o unir dos arrays por filas (eje 0):

a = np.zeros((2, 2))
b = np.ones((2, 2))
c = np.concatenate((a, b), axis=0)
print(a)

# [[0. 0.]
#  [0. 0.]]

print(b)

# [[1. 1.]
#  [1. 1.]]

print(c)

# [[0. 0.]
#  [0. 0.]
```

```
#  [1. 1.]
#  [1. 1.]]
```

split()

Dividir un array en varios arrays por filas o columnas.

```
# Dividir un array en dos arrays por columnas (eje 1):

a = np.array([[10, 20], [30, 40]])
b, c = np.split(a, 2, axis=1)
print(a)

# [[10 20]
#  [30 40]]

print(b)

# [[10]
#  [30]]

print(c)

# [[20]
#  [40]]
```

vsplit()

Dividir un array en varios arrays por filas.

```
# Dividir un array en 3 arrays por filas (eje 0):

a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]], dtype=np.int8)
print(a)

# [[1 2 3]
#  [4 5 6]
#  [7 8 9]]

b, c, d = np.vsplit(a, 3)
print(b)

# [[1 2 3]]

print(c)

# [[4 5 6]]

print(d)

# [[7 8 9]]
```

hsplit()

Dividir un array en varios arrays por columnas.

```
# Dividir un array en 3 arrays por columnas (eje 1):

a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]], dtype=np.int8)
print(a)

# [[1 2 3]
#  [4 5 6]
#  [7 8 9]]

b, c, d = np.hsplit(a, 3)
print(b)

# [[1]
#  [4]
#  [7]]

print(c)
```

```
# [[2]
#  [5]
#  [8]]

print(d)

# [[3]
#  [6]
#  [9]]
```

array_split()

Dividir un array en varios con un tamaño similar.

```
# Dividir un array en 3 arrays de tamaño similar por columnas:

a = np.array([[1, 2, 3, 4], [5, 6, 7, 8]], dtype=np.int8)
print(a)

# [[1 2 3 4]
#  [5 6 7 8]]

b, c, d = np.array_split(a, 3, axis=1)
print(b)

# [[1 2]
#  [5 6]]

print(c)

# [[3]
#  [7]]

print(d)

# [[4]
#  [8]]
```

Publicado por Pherkad en 3:21

Etiquetas: Numpy

Entrada más reciente Inicio Entrada antigua