Intro NumPy

May 15, 2020

1 Intro a NumPy

1.1 Strutture dati in Python

- variabili
- tuple
- list
- dict
- array

1.1.1 Variabili

[1]:
```
# in Python gli oggetti sono tipizzati ma i tipi non sono dichiarati
stringa = "ciao"
num_str = str(10)
num = 5
```

[2]:
```
# infatti durante le operazioni questi tipi vengono verificati
num_str + num
```

```
␣
↪---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call␣
↪last)

<ipython-input-2-ff71c1883499> in <module>
  1 # infatti durante le operazioni questi tipi vengono verificati
----> 2 num_str + num


TypeError: must be str, not int
```

1.2 Tuple

[3]:
```
# le tuple sono insiemi di variabili, anche eterogenee
tupla_1 = (1, 'uno', 1.0)
tupla_1
```

[3]:
```
(1, 'uno', 1.0)
```

[4]:
```
# le tuple non sono modificabili e sono perfette come valori di ritorno
def make_tuple(a, b, c):
    return (a+1, str(b)+"__", c**c)

make_tuple(1, 'uno', 3.14)
```

[4]:
```
(2, 'uno__', 36.33783888017471)
```

1.3 List

[5]:
```
# le liste sono sicuramente le strutture dati piu' usate
list_1 = [1,2,3,4]
list_1
```

[5]:
```
[1, 2, 3, 4]
```

[6]:
```
# sono indicizzate e modificabili, partendo da 0
list_1[2] = 30
list_1
```

[6]:
```
[1, 2, 30, 4]
```

[7]:
```
# le liste sono contenitori eterogeneii
lista_spesa = [ (1, 'tonno'), (5, 'mele'), (3.5, 'etti di carne') ]
lista_spesa[1]
```

[7]:
```
(5, 'mele')
```

[8]:
```
# gli indici sono flessibili
lista_spesa[-1]
```

[8]:
```
(3.5, 'etti di carne')
```

[9]:
```
# dall'indice 1 incluso al 3 escluso
lista_spesa[1:3]
```

[9]:
```
[(5, 'mele'), (3.5, 'etti di carne')]
```

1.4 Dict

[10]:
```
# i dizionari sono delle mappe chiave valore
dizionario = {"uno": 1.0, "due": 2.0, "tre": 3.0}
dizionario['due']
dizionario['444'] = 4.0
dizionario.items()
```

[10]:
```
dict_items([('uno', 1.0), ('due', 2.0), ('tre', 3.0), ('444', 4.0)])
```

1.5 Array

https://docs.python.org/3.6/library/array.html

[11]:
```
# array e' una sequenza di tipi primitivi omogenea, di piu' basso livello␣
 ↪rispetto a list
from array import array
# 'l' sta per intero senza segno (vedi tabella nella doc)
arr = array('l', [1, 2, 3, 4, 5])
arr
```

[11]:
```
array('l', [1, 2, 3, 4, 5])
```

[12]:
```
arr.append(6)
arr[-1]
```

[12]:
```
6
```

[13]:
```
arr.append("7")
```

```
      ␣
↪---------------------------------------------------------------------------

       TypeError                                 Traceback (most recent call␣
↪last)

       <ipython-input-13-fba0f1324044> in <module>
   ----> 1 arr.append("7")


       TypeError: an integer is required (got type str)
```

2 NumPy

Vettori multidimensionali

[14]:
```
# la convenzione
import numpy as np
```

[15]:
```
np.array([1,2,3,4,5])
```

[15]:
```
array([1, 2, 3, 4, 5])
```

[16]:
```
np.arange(1,6)
```

[16]:
```
array([1, 2, 3, 4, 5])
```

[17]:
```
np.arange(5) * 2
```

[17]:
```
array([0, 2, 4, 6, 8])
```

[18]:
```
np.arange(5) + np.ones(5)
```

[18]:
```
array([1., 2., 3., 4., 5.])
```

2.1 Matrici

[19]:
```
# 1 dim
np.arange(12)
```

[19]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

[20]:
```
# 2 dim
np.arange(12).reshape(2,-1)
```

[20]:
```
array([[ 0,  1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10, 11]])
```

[21]:
```
# 3 dim
np.arange(12).reshape((2, 6, -1))
```

[21]:
```
array([[[ 0],
        [ 1],
        [ 2],
        [ 3],
        [ 4],
        [ 5]],
```

```
       [[ 6],
        [ 7],
        [ 8],
        [ 9],
        [10],
        [11]]])
```

2.2 Indici

[22]:
```
vect = np.arange(12).reshape((2, 3, -1))
```

[23]:
```
vect[0]
```

[23]:
```
array([[0, 1],
       [2, 3],
       [4, 5]])
```

[24]:
```
vect[0,1]
```

[24]:
```
array([2, 3])
```

[25]:
```
vect[0,1,1]
```

[25]:
```
3
```

[26]:
```
#[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
line = np.arange(10,20)
indx = [1,3,5]
line[indx]
```

[26]:
```
array([11, 13, 15])
```

[27]:
```
combinazioni = np.array([
    [0,0,0],
    [1,0,0],
    [0,1,0],
    [0,0,1],
])
scelte = np.array([
    [0,1],
    [2,3]
])
combinazioni[scelte]
```

[27]:
```
array([[[0, 0, 0],
        [1, 0, 0]],
```

```
       [[0, 1, 0],
        [0, 0, 1]]])
```

2.3 Maschere

[28]:
```
vect = np.arange(8)
```

[29]:
```
pari = vect%2 == 0
pari
```

[29]:
```
array([ True, False,  True, False,  True, False,  True, False])
```

[30]:
```
dispari = ~pari
dispari
```

[30]:
```
array([False,  True, False,  True, False,  True, False,  True])
```

[31]:
```
vect[dispari]
```

[31]:
```
array([1, 3, 5, 7])
```

2.4 Operazioni

[32]:
```
rg = np.random.default_rng(1)
tabellone = rg.random((1000,1000))
tabellone.shape
```

[32]:
```
(1000, 1000)
```

[33]:
```
tabellone[100,100]
```

[33]:
```
0.9646722103138076
```

[34]:
```
# remember: no for loop!
tab_sin = np.sin(tabellone) * 100
tab_sin[10:15, 10:15]
```

[34]:
```
array([[37.53340799, 19.48878668, 70.81411574,  5.34161961, 65.56816237],
       [70.78193449, 27.25544945, 49.36273104, 58.43269833, 47.27355616],
       [78.09655655, 57.06861549, 44.42274159, 69.79370635, 80.85967781],
       [56.73496676, 35.12566524, 38.73326575,  7.12311146, 69.46902214],
       [48.66615077, 54.44557383,  4.49580625, 78.36746532, 83.49673521]])
```

[ ]:

[ ]: