CS 231n

• python - numpy tutorial

Data types:

1. Numbers    integer, float    no "x++"
2. Booleans    and , or , not , !=
3. Strings    hello = 'hello'  world = 'world'
    hw = hello + ' ' + world
    hw12 = '%s %s %d' % ( hello, world, 12)
    s. capitalized( ) / s.upper( ) / s.just / s.center /
    s. replace('l', '(ell)') / ' world'.strip( )

Containers:

1. List    array, resizeable, different types of data
    l . append ; l.pop( ) → last element

slicing

nums = list (range(5))  // nums = [0 1 2 3 4]

nums[2:4] → [2 3 ~~4~~] exclusive

[2:] → [2 3 4]

[:2] → [0 1]

```
for num in nums:
        print (num)
```

enumerate

```
for idx, num in enumerate(nums):
        print('#%d: %s' % (idx+1, num))
```

```
square = [x**2 ~~[illegible]~~ for x in nums]
even_square = [x **2  for x in nums ~~[illegible]~~ if x%2 ==0]
```

2. Dictionaries 大括号

```
d = {'cat':'cute', 'dog':'furry'}   # map
print(d['cat'])   # cute
~~del d[~~
d['fish']='wet'   # add another entry
d.get('monkey', 'N/A')
    check if 'monkey' is an existed key
    yes: return the value of this key
    no: return the 2nd argument
del d['fish']
```

Loops

```
for animal in d:
    legs= ~~[illegible]~~ d[animal]
    print(legs)

for animal, legs in d.items():
    print('A %s has %d legs' %(animal, legs))
```

```
nums = [0, 1, 2, 3, 4]
even_num_to_square = {x: x ** 2 for x in nums if x%2==0}
```

$\Downarrow$

{0:0, 2:4, 4:16}

3. Sets unordered, distinct elements, 大括号

```
animals = {'cat', 'fish'}
animals.add('dog')
print('fish' in animals) # 'fish'是否在animals里
len(animals) # 3
animals.remove('dog')
```

Loop: unordered loop

```
animals = {'cat', 'dog', 'fish'}
for idx, animal in enumerate(animals):
        print('#%d: %s' % (idx+1, animal))
```

无序的输出

```
nums = {int(sqrt(x)) for x in range(30)}
        ⇓
    {0, 1, 2, 3, 4} # unordered
```

4. Tuples

like list, but can be elements ~~of~~ of sets and as keys ~~[illegible]~~ of dics

```
d = {(x, x+1): x for x in range(10)} #dic, key: tuple
t = (5, 6) # tuple
d[t] -> 5
d[(1, 2)] -> 1
```

('apple', 'banana', 'orange')

Function def

def sign(x):

Classes

```
class Greeter(object):

    def _init_(self, name):   # 构造函数
        self.name = name

    def greet(self, loud=False):
      if loud:
          print(' HELLO, %s!" %self.name.upper())
      else:
          print('Hello, %s' % self.name)

g = Greeter('Fred') # 构造一个class
g.greet() # 调用 class 的函数
g.greet( loud = True
```

## Numpy

import numpy as np

### Array

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

a.shape # tuple of array dimensions

↓ (3,) : 1 D array

(n, m) : 2D array; <u>n</u> is the number of rows and <u>m</u> is the number of columns

(n, m, k) : 3D

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

$$\begin{matrix} [1, & 2, & 3] \\ [4, & 5, & 6] \end{matrix} \quad \underbrace{}_{2\times 3}$$

b[0, 0] # 1

b[1, 0] # 4

np.Zeros((2, 2)) $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

np.ones ((1, 2)) $[1 \quad 1]$

np.full ((2, 2), 7) $\begin{bmatrix} 7 & 7 \\ 7 & 7 \end{bmatrix}$

np. eye(2) $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

np.random.random((2, 2)) $\begin{bmatrix} - & - \\ - & - \end{bmatrix}$

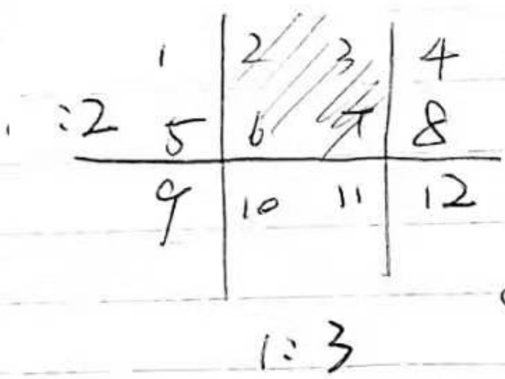
np.sum( , axis )

0 : 按~~列~~行相加

1 : 按列相加

Array indexing

a = np.array([[1,2,3,4],[5,6,7,8][9,10, 11,12]])



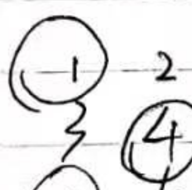
b = a[:2, 1:3]

Slicing Array

b's change will cause a to change, too

a[1, :] → [5, 6, 7, 8]   (4, )

a[1:2, :] → [[5, 6, 7, 8]]   (1,4)



a[[0, 1, 2], [0, 1, 0]]

[1, 4, 5]

np.array([a[0,0], a[1,1], a[2,0]])

Boolean array indexing

```
a=np.array([[1,2],[3,4],[5,6]])
bool_idx = (a>2) # F F
                   T T
                   T T

a[bool_idx] # [3, 4 5 6]
  a[a>2]       [3 4 5 6]
```

Datatypes

```
x= np.array([1,2])
x.dtype  # int 64
x= np.array ([1.0, 2.0])
x. dtype   # float 64
x= np. array ([1,2], dtype=np.int64)
x.dtype # int64
x=np.array([[1, 2],[3, 4]], dtype=np.float64)
y=np.array([[5, 6],[7, 8]], dtype =np. float64)
 x+y                  } same
 np.add ~~[x+y]~~ (x,y)
 x-y
 np.subtract (x,y) subtract
        multiply(x,y)          divide(x,y)
                               sqrt(x)
```

```
V = np.array([9, 10])
W = np.array([11, 12])

V.dot(W)
np.dot(V, W)   => 219
```

dot => 1D array -> dot produ
=> matrix -> matrix multiplicati

```
v.T #transpose
```

Broadcasting

```
x = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12]])
v = np.array([1,0,1])
vv = np.tile(v,(4,1)) # [[1 0 1]
                         [1 0 1]
                         [1 0 1]
                         [1 0 1]]
```

① stack 4 copies of v on top of each other

```
y = x + vv
```

② y = x + v => same

```
v = np.array([1,2,3])
w = np.array([4,5])
np.reshape(v,(3,1)) * w
```

$$\underbrace{\text{np.reshape}(v,(3,1))}$$

$$\begin{bmatrix}1\\2\\3\end{bmatrix}\begin{bmatrix}4 & 5\end{bmatrix} \quad \begin{bmatrix}4 & 5\\8 & 10\\12 & 15\end{bmatrix}$$

## Image operations

```
img = imread('assets/cat.jpg')
print (img.dtype, img.shape)
                    └>(400, 248, 3)
img-tinted = img * [1, 0.95, 0.9] #会改变 RGB.
img_tinted = imresize (img_tinted, (300, 300))
imsave ('assets/cat_tinted.jpg', img_tinted)
```

## Matplotlib

```
import matplotlib.pyplot as plt

x = np.arange(0, 3*np.pi, 0.1)
y = np.sin(x)
plt.plot(x, y)
plt.show()
```