homework#1 numpy tutorial

September 19, 2018

In [2]:
```
def quicksort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quicksort(left) + middle + quicksort(right)
```

In [3]:
```
print(quicksort([3,6,8,10,1,2,1]))
```

```
[1, 1, 2, 3, 6, 8, 10]
```

In [4]:
```
x = 3
print(type(x))
print(x)
print(x + 1)
print(x - 1)
print(x * 2)
print(x ** 2)
x += 1
print(x)
x *= 2
print(x)
y = 2.5
print(type(y))
print(y, y + 1, y * 2, y ** 2)
```

```
<class 'int'>
3
4
2
6
9
4
8
<class 'float'>
```

```
2.5 3.5 5.0 6.25
```

```
In [5]: t = True
        f = False
        print(type(t))
        print(t and f)
        print(t or f)
        print(not t)
```

```
<class 'bool'>
False
True
False
```

```
In [6]: hello = 'hello'
        world = "world"
        print(hello)
        print(len(hello))
        hw = hello + '' + world
        print(hw)
        hw12 = '%s %s %d' % (hello, world, 12)
        print(hw12)
```

```
hello
5
helloworld
hello world 12
```

```
In [7]: s = "hello"
        print(s.capitalize())
        print(s.upper())
        print(s.rjust(7))
        print(s.center(7))
        print(s.replace('l', '(ell)'))
        print(' world'.strip())
```

```
Hello
HELLO
  hello
 hello
he(ell)(ell)o
world
```

```
In [8]: xs = [3, 1, 2]
        print(xs, xs[2])
```

```
        print(xs[-1])
        xs[2] = 'foo'
        print(xs)
        xs.append('bar')
        print(xs)
        x = xs.pop()
        print(x, xs)
```

```
[3, 1, 2] 2
2
[3, 1, 'foo']
[3, 1, 'foo', 'bar']
bar [3, 1, 'foo']
```

```
In [9]: nums = list(range(5))
        print(nums)
        print(nums[2:4])
        print(nums[2:])
        print(nums[:2])
        print(nums[:])
        print(nums[:-1])
        nums[2:4] = [8, 9]
        print(nums)
```

```
[0, 1, 2, 3, 4]
[2, 3]
[2, 3, 4]
[0, 1]
[0, 1, 2, 3, 4]
[0, 1, 2, 3]
[0, 1, 8, 9, 4]
```

```
In [10]: animals = ['cat', 'dog', 'monkey']
         for animal in animals:
             print(animal)
```

```
cat
dog
monkey
```

```
In [11]: animals = ['cat', 'dog', 'monkey']
         for idx, animal in enumerate(animals):
             print('#%d: %s' % (idx + 1, animal))
```

```
#1: cat
#2: dog
#3: monkey
```

```
In [12]: nums = [0, 1, 2, 3, 4]
         squares = []
         for x in nums:
             squares.append(x ** 2)
         print(squares)
```

```
[0, 1, 4, 9, 16]
```

```
In [13]: nums = [0, 1, 2, 3, 4]
         squares = [x ** 2 for x in nums]
         print(squares)
```

```
[0, 1, 4, 9, 16]
```

```
In [14]: nums = [0, 1, 2, 3, 4]
         even_squares = [x ** 2 for x in nums if x % 2 == 0]
         print(even_squares)
```

```
[0, 4, 16]
```

```
In [15]: d = {'cat': 'cute', 'dog': 'furry'}
         print(d['cat'])
         print('cat' in d)
         d['fish'] = 'wet'
         print(d['fish'])
         # print(d['monkey'])
         print(d.get('monkey', 'N/A'))
         print(d.get('fish', 'N/A'))
         del d['fish']
         print(d.get('fish', 'N/A'))
```

```
cute
True
wet
N/A
wet
N/A
```

```
In [16]: d = {'person': 2, 'cat': 4, 'spider': 8}
         for animal in d:
             legs = d[animal]
             print('A %s has %d legs' % (animal, legs))
```

```
A person has 2 legs
A cat has 4 legs
A spider has 8 legs
```

```
In [17]: d = {'person': 2, 'cat': 4, 'spider': 8}
         for animal, legs in d.items():
             print('A %s has %d legs' % (animal, legs))
```

```
A person has 2 legs
A cat has 4 legs
A spider has 8 legs
```

```
In [18]: nums = [0, 1, 2, 3, 4]
         even_num_to_square = {x: x ** 2 for x in nums if x % 2 == 0}
         print(even_num_to_square)
```

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

```
In [19]: animals = {'cat', 'dog'}
         print('cat' in animals)
         print('fish' in animals)
         animals.add('fish')
         print('fish' in animals)
         print(len(animals))
         animals.add('cat')
         print(len(animals))
         animals.remove('cat')
         print(len(animals))
```

```
True
False
True
3
3
2
```

```
In [20]: animals = {'cat', 'dog', 'fish'}
         for idx, animal in enumerate(animals):
             print('#%d: %s' % (idx + 1, animal))
```

```
#1: cat
#2: dog
#3: fish
```

```
In [21]: from math import sqrt
         nums = {int(sqrt(x)) for x in range(30)}
         print(nums)
```

```
{0, 1, 2, 3, 4, 5}
```

```
In [22]: d = {(x, x + 1): x for x in range(10)}
         t = (5, 6)
         print(type(t))
         print(d[t])
         print(d[(1, 2)])
```

```
<class 'tuple'>
5
1
```

```
In [23]: def sign(x):
             if x > 0:
                 return 'positive'
             elif x < 0:
                 return 'negative'
             else:
                 return 'zero'

         for x in [-1, 0, 1]:
             print(sign(x))
```

```
negative
zero
positive
```

```
In [24]: def hello(name, loud=False):
             if loud:
                 print('HELLO, %s!' % name.upper())
             else:
                 print('Hello, %s' % name)

         hello('Bob')
         hello('Fred', loud=True)
```

```
Hello, Bob
HELLO, FRED!
```

```
In [25]: class Greeter(object):

             def __init__(self, name):
                 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')
        g.greet()
        g.greet(loud=True)
```

```
Hello, Fred
HELLO, FRED!
```

```
In [26]: import numpy as np
```

```
In [27]: a = np.array([1, 2, 3])
         print(type(a))
         print(a.shape)
         print(a[0], a[1], a[2])
         a[0] = 5
         print(a)

         b = np.array([[1,2,3],[4,5,6]])
         print(b.shape)
         print(b[0, 0], b[0, 1], b[1, 0])
```

```
<class 'numpy.ndarray'>
(3,)
1 2 3
[5 2 3]
(2, 3)
1 2 4
```

```
In [28]: a = np.zeros((2,2))
         print(a)

         b = np.ones((1,2))
         print(b)

         c = np.full((2,2), 7)
         print(c)

         d = np.eye(2)
         print(d)

         e = np.random.random((2,2))
         print(e)
```

```
[[0. 0.]
 [0. 0.]]
[[1. 1.]]
```

```
[[7 7]
 [7 7]]
[[1. 0.]
 [0. 1.]]
[[0.38606616 0.9801186 ]
 [0.09936632 0.44708098]]
```

```
In [29]: a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])

         b = a[:2, 1:3]
         print(b)

         print(a[0, 1])
         b[0, 0] = 77
         print(a[0, 1])
```

```
[[2 3]
 [6 7]]
2
77
```

```
In [30]: a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])

         row_r1 = a[1, :]
         row_r2 = a[1:2, :]
         print(row_r1, row_r1.shape)
         print(row_r2, row_r2.shape)

         col_r1 = a[:, 1]
         col_r2 = a[:, 1:2]
         print(col_r1, col_r1.shape)
         print(col_r2, col_r2.shape)
```

```
[5 6 7 8] (4,)
[[5 6 7 8]] (1, 4)
[ 2  6 10] (3,)
[[ 2]
 [ 6]
 [10]] (3, 1)
```

```
In [31]: a = np.array([[1,2], [3, 4], [5, 6]])

         print(a[[0, 1, 2], [0, 1, 0]])

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

```
        print(a[[0, 0], [1, 1]])

        print(np.array([a[0, 1], a[0, 1]]))
```

```
[1 4 5]
[1 4 5]
[2 2]
[2 2]
```

```
In [32]: a = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])

         print(a)


         b = np.array([0, 2, 0, 1])

         print(a[np.arange(4), b])

         a[np.arange(4), b] += 10

         print(a)
```

```
[[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
[ 1  6  7 11]
[[11  2  3]
 [ 4  5 16]
 [17  8  9]
 [10 21 12]]
```

```
In [33]: a = np.array([[1,2], [3, 4], [5, 6]])

         bool_idx = (a > 2)

         print(bool_idx)

         print(a[bool_idx])

         print(a[a > 2])
```

```
[[False False]
 [ True  True]
 [ True  True]]
[3 4 5 6]
[3 4 5 6]
```

```
In [34]: x = np.array([1, 2])
         print(x.dtype)

         x = np.array([1.0, 2.0])
         print(x.dtype)

         x = np.array([1, 2], dtype=np.int64)
         print(x.dtype)
```

```
int64
float64
int64
```

```
In [35]: x = np.array([[1,2],[3,4]], dtype=np.float64)
         y = np.array([[5,6],[7,8]], dtype=np.float64)

         print(x + y)
         print(np.add(x, y))

         print(x - y)
         print(np.subtract(x, y))

         print(x * y)
         print(np.multiply(x, y))

         print(x / y)
         print(np.divide(x, y))

         print(np.sqrt(x))
```

```
[[ 6.  8.]
 [10. 12.]]
[[ 6.  8.]
 [10. 12.]]
[[-4. -4.]
 [-4. -4.]]
[[-4. -4.]
 [-4. -4.]]
[[ 5. 12.]
 [21. 32.]]
[[ 5. 12.]
 [21. 32.]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[1.         1.41421356]
```

```
 [1.73205081 2.        ]]
```

```
In [36]: x = np.array([[1,2],[3,4]])
         y = np.array([[5,6],[7,8]])

         v = np.array([9,10])
         w = np.array([11, 12])

         print(v.dot(w))
         print(np.dot(v, w))

         print(x.dot(v))
         print(np.dot(x, v))

         print(x.dot(y))
         print(np.dot(x, y))
```

```
219
219
[29 67]
[29 67]
[[19 22]
 [43 50]]
[[19 22]
 [43 50]]
```

```
In [37]: x = np.array([[1,2],[3,4]])

         print(np.sum(x))
         print(np.sum(x, axis=0))
         print(np.sum(x, axis=1))
```

```
10
[4 6]
[3 7]
```

```
In [38]: x = np.array([[1,2], [3,4]])
         print(x)
         print(x.T)

         v = np.array([1,2,3])
         print(v)
         print(v.T)
```

```
[[1 2]
 [3 4]]
```

```
[[1 3]
 [2 4]]
[1 2 3]
[1 2 3]
```

```
In [39]: x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
         v = np.array([1, 0, 1])
         y = np.empty_like(x)

         for i in range(4):
             y[i, :] = x[i, :] + v

         print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

```
In [40]: 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))
         print(vv)

         y = x + vv

         print(y)
```

```
[[1 0 1]
 [1 0 1]
 [1 0 1]
 [1 0 1]]
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

```
In [41]: x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
         v = np.array([1, 0, 1])
         y = x + v
         print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

```
In [43]: v = np.array([1,2,3])
         w = np.array([4,5])

         print(np.reshape(v, (3, 1)) * w)

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

         print(x + v)

         print((x.T + w).T)

         print(x + np.reshape(w, (2, 1)))

         print(x * 2)
```

```
[[ 4  5]
 [ 8 10]
 [12 15]]
[[2 4 6]
 [5 7 9]]
[[ 5  6  7]
 [ 9 10 11]]
[[ 5  6  7]
 [ 9 10 11]]
[[ 2  4  6]
 [ 8 10 12]]
```

```
In [54]: from scipy.misc import imread, imsave, imresize

         img = imread('assets/cat.jpg')
         print(img.dtype, img.shape)

         img_tinted = img * [1, 0.95, 0.9]

         img_tinted = imresize(img_tinted, (300, 300))

         imsave('assets/cat_tinted.jpg', img_tinted)

         plt.imshow(np.uint8(img_tinted))
```

```
uint8 (400, 248, 3)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: DeprecationWarning: `imread` is
`imread` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``imageio.imread`` instead.
  This is separate from the ipykernel package so we can avoid doing imports until
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:8: DeprecationWarning: `imresize`
```

```
`imresize` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``skimage.transform.resize`` instead.

/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:10: DeprecationWarning: `imsave` i
`imsave` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``imageio.imwrite`` instead.
  # Remove the CWD from sys.path while we load stuff.
```

Out[54]: <matplotlib.image.AxesImage at 0xb219c00b8>

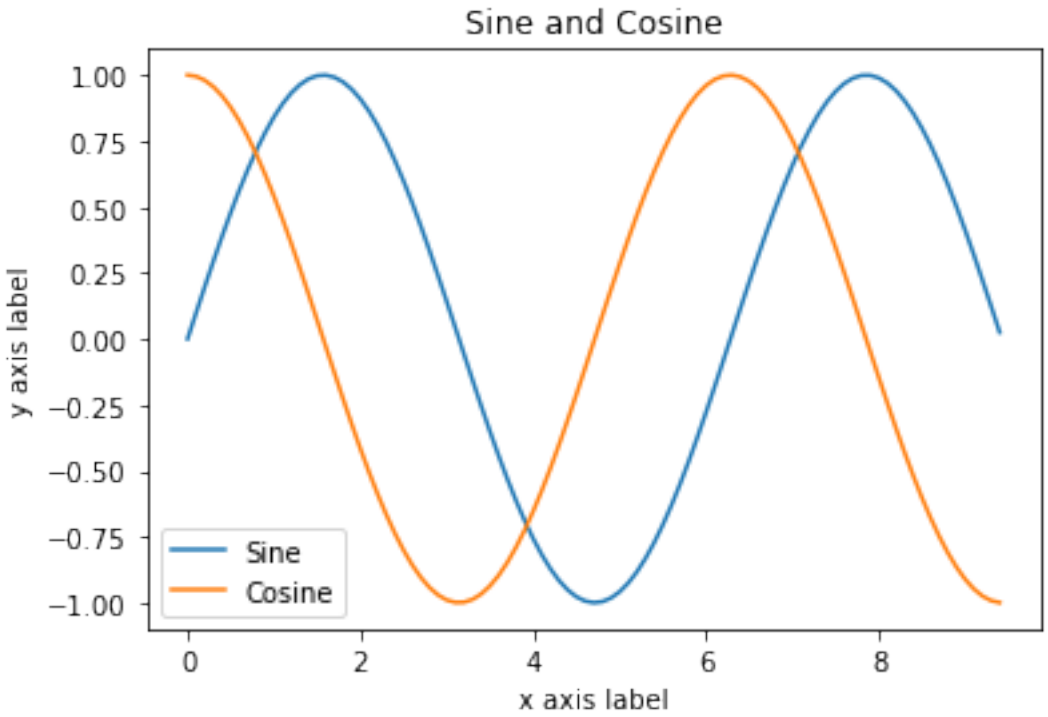
In [45]:
```
from scipy.spatial.distance import pdist, squareform

x = np.array([[0, 1], [1, 0], [2, 0]])
print(x)

d = squareform(pdist(x, 'euclidean'))
print(d)
```

```
[[0 1]
 [1 0]
 [2 0]]
[[0.         1.41421356 2.23606798]
 [1.41421356 0.         1.        ]
 [2.23606798 1.         0.        ]]
```

```
In [46]: import matplotlib.pyplot as plt

         x = np.arange(0, 3 * np.pi, 0.1)
         y = np.sin(x)

         plt.plot(x, y)
         plt.show()
```

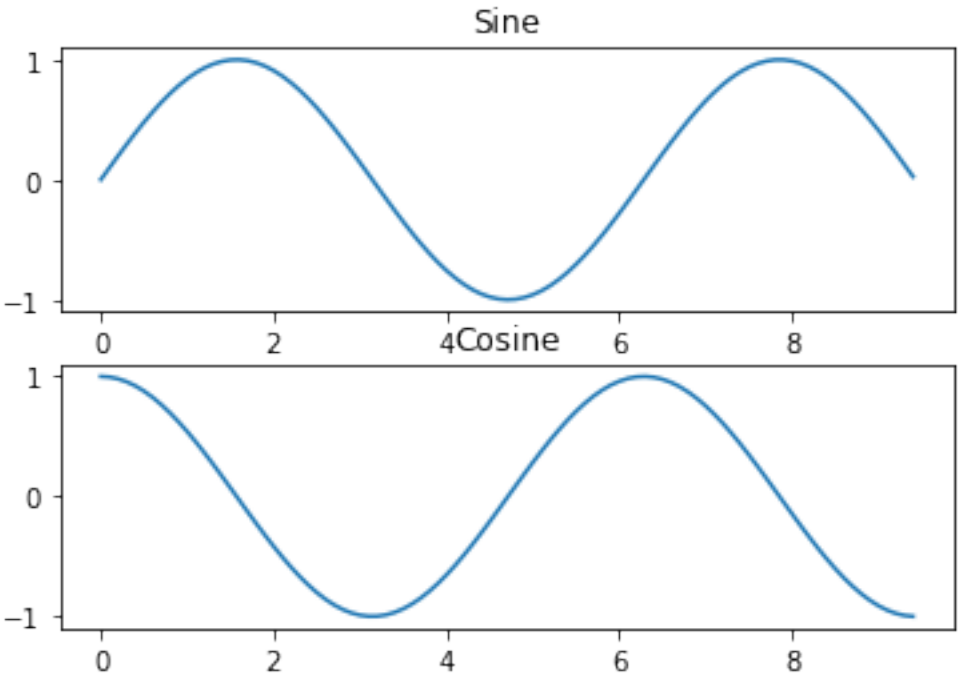
```
<Figure size 640x480 with 1 Axes>
```

```
In [47]: x = np.arange(0, 3 * np.pi, 0.1)
         y_sin = np.sin(x)
         y_cos = np.cos(x)

         plt.plot(x, y_sin)
         plt.plot(x, y_cos)
         plt.xlabel('x axis label')
         plt.ylabel('y axis label')
         plt.title('Sine and Cosine')
         plt.legend(['Sine', 'Cosine'])
         plt.show()
```




```
In [48]: x = np.arange(0, 3 * np.pi, 0.1)
         y_sin = np.sin(x)
```

```
y_cos = np.cos(x)

plt.subplot(2, 1, 1)

plt.plot(x, y_sin)
plt.title('Sine')

plt.subplot(2, 1, 2)
plt.plot(x, y_cos)
plt.title('Cosine')

plt.show()
```




```
In [50]: from scipy.misc import imread, imresize

         img = imread('assets/cat.jpg')
         img_tinted = img * [1, 0.95, 0.9]

         plt.subplot(1, 2, 1)
         plt.imshow(img)

         plt.subplot(1, 2, 2)

         plt.imshow(np.uint8(img_tinted))
         plt.show()
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:3: DeprecationWarning: `imread` is
`imread` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``imageio.imread`` instead.
  This is separate from the ipykernel package so we can avoid doing imports until
```