03_hello_numpy_?????????

September 8, 2020

1 Python Numpy Tutorial #1

NumPy

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

List

[2]:
```
a = [1, 2, 3]
```

[3]:
```
print(a)
```

```
[1, 2, 3]
```

Numpy

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

[5]:
```
print(a)
```

```
[1 2 3]
```

[6]:
```
a.shape
```

[6]:
```
(3,)
```

[8]:
```
b = np.array([[1, 2, 3],[4, 5, 6]])
print(b.shape)
print(b[0, 0], b[0, 1], b[1, 0])
```

```
(2, 3)
1 2 4
```

Axis/axes

- the nth coordinate to index an array in Numpy.
- multidimensional arrays can have one index per axis.

[9]:
```
a = np.array([[1, 2], [3, 4]])
print(a)
```

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

Axis

[10]:
```
np.mean(a)
```

[10]:
```
2.5
```

Axis 0 ()

[11]:
```
np.mean(a, axis=0)
```

[11]:
```
array([2., 3.])
```

[12]:
```
np.sum(a, axis=0)
```

[12]:
```
array([4, 6])
```

Axis 1 ()

[13]:
```
np.mean(a, axis=1)
```

[13]:
```
array([1.5, 3.5])
```

[15]:
```
np.sum(a, axis=1)
```

[15]:
```
array([3, 7])
```

Broadcast

(without explicit for-loop)

[16]:
```
A = np.array([[56.0, 0.0, 4.0, 68.0],
              [1.2, 104.0, 52.0, 8.0],
              [1.8, 135.0, 99.0, 0.9]])
print(A)
```

```
[[ 56.    0.    4.   68. ]
 [  1.2 104.   52.    8. ]
 [  1.8 135.   99.    0.9]]
```

[17]:
```
cal = A.sum(axis=0)
print(cal)
```

```
[ 59.  239.  155.   76.9]
```

[19]:
```
percentage = 100*A / cal
print(percentage)
```

```
[[94.91525424  0.          2.58064516 88.42652796]
 [ 2.03389831 43.51464435 33.5483871  10.40312094]
 [ 3.05084746 56.48535565 63.87096774  1.17035111]]
```

Stack

[20]:
```
a = np.array([1, 2, 3, 4])
b = np.array([5, 6, 7, 8])
```

-

[21]:
```
c = np.vstack((a,b))
print(c)
print(c.shape)
```

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

-

[22]:
```
d = np.hstack((a,b))
print(d)
```

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

[23]:
```
x = np.array([-2.0, 1.2, 3.7])
y = x > 0
print(y)
```

```
[False  True  True]
```

[24]:
```
y = y.astype(np.int)
print(y)
```

```
[0 1 1]
```

Noise

[25]:
```
x = np.arange(0, 6, 0.1)
y = np.sin(x)
```

[26]:
```
y
```

[26]:
```
array([ 0.        ,  0.09983342,  0.19866933,  0.29552021,  0.38941834,
        0.47942554,  0.56464247,  0.64421769,  0.71735609,  0.78332691,
        0.84147098,  0.89120736,  0.93203909,  0.96355819,  0.98544973,
        0.99749499,  0.9995736 ,  0.99166481,  0.97384763,  0.94630009,
        0.90929743,  0.86320937,  0.8084964 ,  0.74570521,  0.67546318,
        0.59847214,  0.51550137,  0.42737988,  0.33498815,  0.23924933,
        0.14112001,  0.04158066, -0.05837414, -0.15774569, -0.2555411 ,
       -0.35078323, -0.44252044, -0.52983614, -0.61185789, -0.68776616,
       -0.7568025 , -0.81827711, -0.87157577, -0.91616594, -0.95160207,
```

```
       -0.97753012, -0.993691   , -0.99992326, -0.99616461, -0.98245261,
       -0.95892427, -0.92581468, -0.88345466, -0.83226744, -0.77276449,
       -0.70554033, -0.63126664, -0.55068554, -0.46460218, -0.37387666])
```

[27]:
```
signal_length = y.shape[0]
print(signal_length)
```

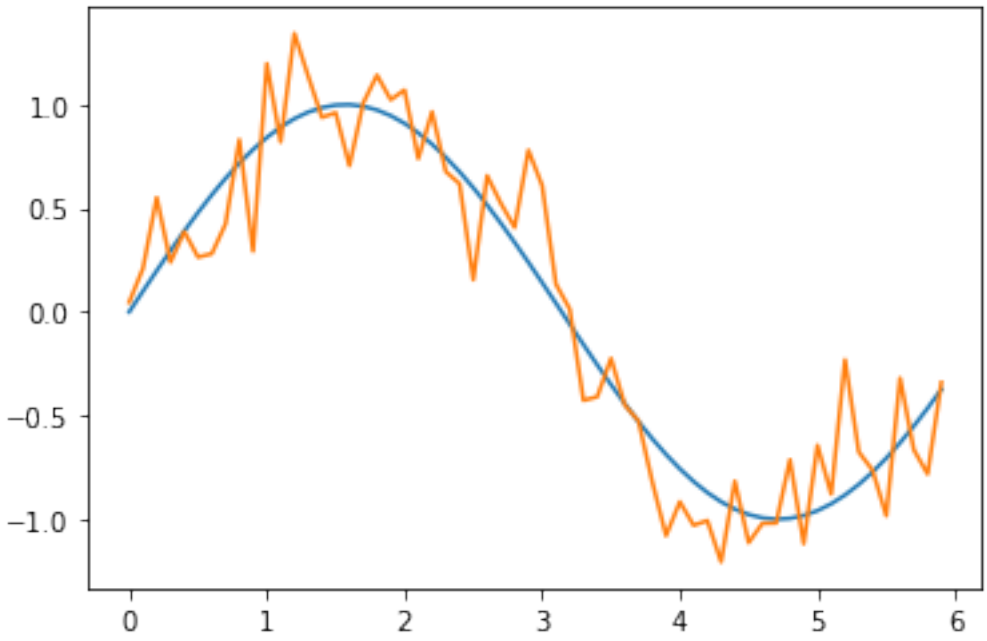
```
60
```

- np.random.normal : random samples from a normal (Gaussian) distrubution

[29]:
```
noise = np.random.normal(0, 1, signal_length)
y2 = y + 0.2* noise
```

[30]:
```
plt.plot(x, y)
plt.plot(x, y2)
```

[30]:
```
[<matplotlib.lines.Line2D at 0x7f3b950d1cf8>]
```



[31]:
```
A = np.random.rand(4, 3, 2)
print(A)
```

```
[[[0.74797859 0.60616772]
  [0.8547254  0.4611236 ]
  [0.01206711 0.17158532]]
```

```
[[0.38129509 0.62130143]
 [0.75295755 0.71576835]
 [0.85387774 0.36354404]]

[[0.8963715  0.75617475]
 [0.97953046 0.95151118]
 [0.94908524 0.64696353]]

[[0.10395606 0.83102202]
 [0.94935734 0.43014359]
 [0.63066627 0.55101232]]]
```

[32]:
```
A = A.reshape(12, 2)
print(A)
```

```
[[0.74797859 0.60616772]
 [0.8547254  0.4611236 ]
 [0.01206711 0.17158532]
 [0.38129509 0.62130143]
 [0.75295755 0.71576835]
 [0.85387774 0.36354404]
 [0.8963715  0.75617475]
 [0.97953046 0.95151118]
 [0.94908524 0.64696353]
 [0.10395606 0.83102202]
 [0.94935734 0.43014359]
 [0.63066627 0.55101232]]
```

1 ravel() vs.reshape(0 vs.flatten() * 1

[33]:
```
B = A.ravel()
print(B.shape)
```

```
(24,)
```

[35]:
```
C = A.flatten()
print(C.shape)
```

```
(24,)
```

reshape

[53]:
```
D = A.reshape(-1)
print(D.shape)
```

```
(24,)
```

Random
np.random.choice

[38]:
```
np.random.seed(0)
p = np.array([0.1, 0.0, 0.7, 0.2])
```

[39]:
```
for i in range(10):
  index = np.random.choice([0, 1, 2, 3], p = p.ravel())
  print(i, index)
```

```
0 2
1 2
2 2
3 2
4 2
5 2
6 2
7 3
8 3
9 2
```

This means that you will pick the index according to the distribution: (= 0) = 0.1, (= 1) = 0.0, (= 2) = 0.7, (= 3) = 0.2

[40]:
```
v = [0,0,0,0]
ntest = 1000
for i in range(ntest):
  idx = np.random.choice([0, 1, 2, 3], p = p.ravel())
  v[idx] += 1
```

np.random.permutation

[41]:
```
m1 = np.random.permutation(5)
print(m1)
```

```
[2 1 4 0 3]
```

- array shuffle (1)

[42]:
```
X = np.array([10,20,30,40,50])
shuffled_X = X[m1]
```

[43]:
```
print(X)
print(shuffled_X)
```

```
[10 20 30 40 50]
[30 20 50 10 40]
```

- array shuffle (2)

[44]:
```
x1 = [2, 4, 8, 10, 20]
x2 = [0.2, 0.4, -0.8, 1.0, -2.0]
X = np.transpose(np.vstack((x1, x2)))
```

[45]:
```
print(X)
print(X.shape)
```

```
[[ 2.   0.2]
 [ 4.   0.4]
 [ 8.  -0.8]
 [10.   1. ]
 [20.  -2. ]]
(5, 2)
```

[48]:
```
permutation = np.random.permutation(5)
```

[49]:
```
X_shuffle= X[permutation]
```

[50]:
```
print(X_shuffle)
```

```
[[ 8.  -0.8]
 [10.   1. ]
 [20.  -2. ]
 [ 4.   0.4]
 [ 2.   0.2]]
```