Muhammad Danial

muhammaddanialarain@gmail.com

Notebook 2

This is the second notebook on numpy in this notebook we covered some advance functions of numpy.

Numpy array Vs Python Lists

- speed

```
# speed
# list
a=[i for i in range(10000000)]
b=[i for i in range(10000000,20000000)]

c=[]
import time
start=time.time()
for i in range(len(a)):
    c.append(a[i]+b[i])
print(time.time() - start)
```

```
4.414110898971558
```

```
# numpy
import numpy as np
a=np.arange(10000000)
b=np.arange(10000000,20000000)
start=time.time()
c=a+b
print(time.time() - start)
```

```
0.1560802459716797
```

- Memory

```
# memory of list
a=[i for i in range(10000000)]
import sys
print("list size : ",sys.getsizeof(a))
# memory of numpy array
b=np.arange(10000000)
```

```
print("size of numpy array : ",sys.getsizeof(b))
```

```
list size :  89095160
size of numpy array :  40000112
```

```
# convenience
# numpy is better than python list
```

Advanced Indexing

- Normal Indexing and slicing

```
a=np.arange(12).reshape(4,3)
a
```

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

```
a[1,2]
```

```
5
```

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

```
array([[4, 5],
       [7, 8]])
```

- Fancy Indexing

```
a
```

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

```
a[[0,2,3]] # inside list provide index of rows
```

```
array([[ 0,  1,  2],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

```
a[:,[0,2]]
```

```
array([[ 0,  2],
       [ 3,  5],
       [ 6,  8],
       [ 9, 11]])
```

- Boolean Indexing

```
a=np.random.randint(1,100,24).reshape(6,4)
a
```

```
array([[31, 93, 68, 15],
       [99, 32, 25, 63],
       [59,  8, 17, 86],
       [71, 97, 22, 50],
       [35, 17, 92, 43],
       [21, 72, 47, 60]])
```

- find all numbers greater than 50

```
a[a>50]
```

```
array([93, 68, 99, 63, 59, 86, 71, 97, 92, 72, 60])
```

- find out even number

```
a[a%2==0]
```

```
array([68, 32,  8, 86, 22, 50, 92, 72, 60])
```

- find all numbers greater than 50 and are odd

```
a[(a > 50) & (a % 2 != 0)]
```

```
array([93, 99, 63, 59, 71, 97])
```

```
# find all numbers not divisible by 7
a[a%7==0]
```

```
array([98])
```

BroadCasting

- The term broadcasting describes how numpy treats arrays with different shapes during arithmetic operations.
- The smaller array is "broadcast" across the larger array so that they have compatible shapes.

```
# same shape
a=np.arange(6).reshape(2,3)
b=np.arange(6,12).reshape(2,3)
print(a)
print("++++++++++++")
print(b)
print("=======")
print(a+b)
```

```
[[0 1 2]
 [3 4 5]]
++++++++++++
```

```
[[ 6  7  8]
 [ 9 10 11]]
=======
[[ 6  8 10]
 [12 14 16]]
```

```
# different shapes
a=np.arange(6).reshape(2,3)
b=np.arange(3).reshape(1,3)
print(a)
print("=======")
print(b)
print("---------------")
print(a+b)
```

```
[[0 1 2]
 [3 4 5]]
=======
[[0 1 2]]
---------------
[[0 2 4]
 [3 5 7]]
```

```
a=np.arange(12).reshape(4,3)
b=np.arange(3)

print(a)
print("+"*20)
print(b)
print("="*20)
print(a+b)
```

```
[[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]
 [ 9 10 11]]
++++++++++++++++++++
[0 1 2]
====================
[[ 0  2  4]
 [ 3  5  7]
 [ 6  8 10]
 [ 9 11 13]]
```

```
a=np.arange(12).reshape(3,4)
b=np.arange(3)

print(a)
print("+"*20)
print(b)
print("="*20)
```

```
print(a+b)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
++++++++++++++++++++
[0 1 2]
====================
```

```
---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
~\AppData\Local\Temp/ipykernel_8128/23792332.py in <module>
      7 print("="*20)
      8
----> 9 print(a+b)

ValueError: operands could not be broadcast together with shapes (3,4)
(3,)
```

```
a=np.arange(3).reshape(1,3)
b=np.arange(3).reshape(3,1)

print(a)
print("+"*20)
print(b)
print("="*20)
print(a+b)
```

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

```
a=np.arange(3).reshape(1,3)
b=np.arange(4).reshape(4,1)

print(a)
print("+"*20)
print(b)
print("="*20)
print(a+b)
```

```
[[0 1 2]]
++++++++++++++++++++
[[0]
 [1]
 [2]
 [3]]
====================
[[0 1 2]
 [1 2 3]
 [2 3 4]
 [3 4 5]]
```

```
a=np.array([1])
b=np.arange(4).reshape(2,2)

print(a)
print(b)

print(a+b)
```

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

working with mathematical formulas

```
# sigmoid
def sigmoid(array):
    return 1/(1+np.exp(-(array)))
sigmoid(np.arange(10))
```

```
array([0.5       , 0.73105858, 0.88079708, 0.95257413, 0.98201379,
       0.99330715, 0.99752738, 0.99908895, 0.99966465, 0.99987661])
```

```
# mean squared error
actual=np.random.randint(1,50,25)
predicted=np.random.randint(1,50,25)

print(actual)
print("============================")
print(predicted)

def mse(actual,predicted):
    return np.mean((actual-predicted)**2)
print("------------------------------")
mse(actual,predicted)
```

```
[45 36 42  3 31 27 35  9 13  8 49 11 38 18 14 20 12 32 28 39  6 31 29
28
```

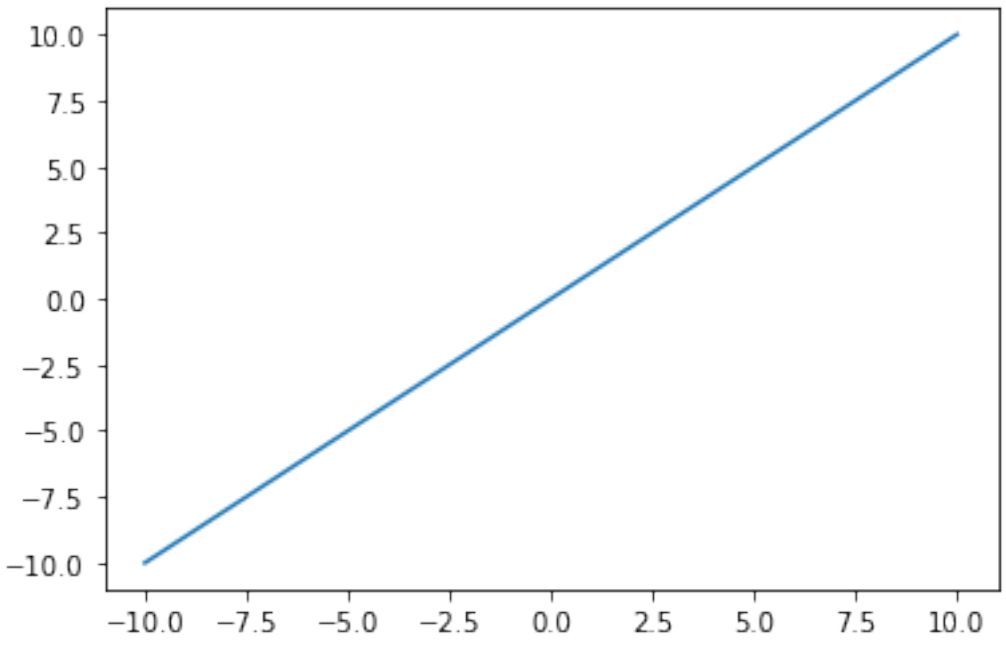
```
 39]
============================
[48  7 14 14  2 18 46 17 49  8 12 12 36  1 30 49 14 43  1 44 16 11 33
44
 34]
------------------------------

343.76
```

working with missing values

```
# working with missing values --> np.nan
a=np.array([1,2,3,4,np.nan,6])
print(a)

[ 1.  2.  3.  4. nan  6.]

a[~np.isnan(a)]

array([1., 2., 3., 4., 6.])
```

plotting graphs

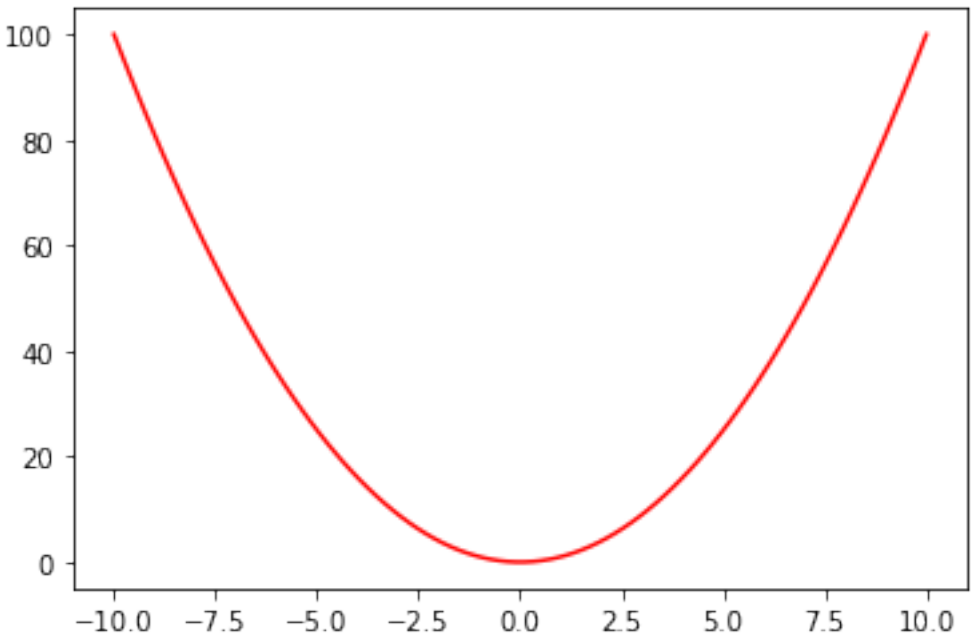
```
# plotting a 2D plot
# x=y
import matplotlib.pyplot as plt
x=np.linspace(-10,10,100)
y=x
plt.plot(x,y)

[<matplotlib.lines.Line2D at 0x1fe80eab3a0>]
```



- square plot

```
x=np.linspace(-10,10,100)
y=x**2
plt.plot(x,y,color='red')
```

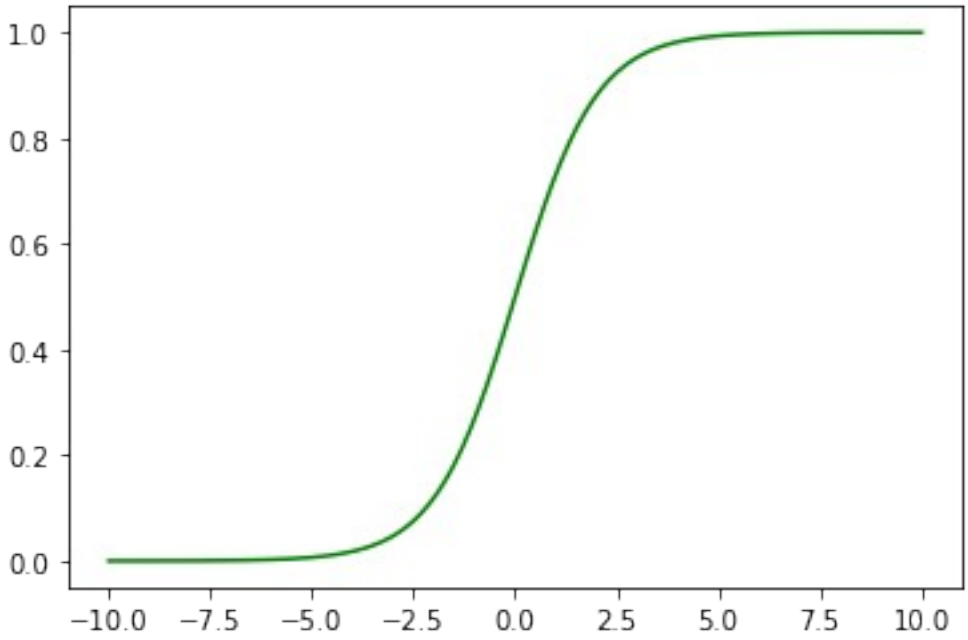
```
[<matplotlib.lines.Line2D at 0x1fe818976d0>]
```



- sigmoid function plot

```
x=np.linspace(-10,10,100)
y=1/(1+np.exp(-(x)))
plt.plot(x,y,color='green')
```

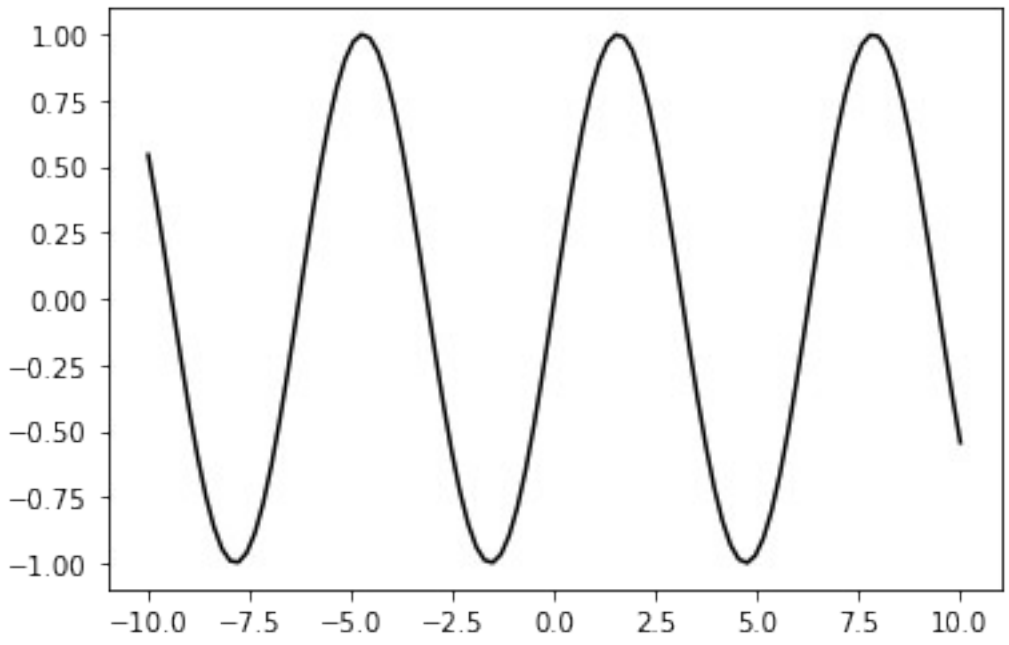
```
[<matplotlib.lines.Line2D at 0x1fe8191baf0>]
```



- y =sin(x)

```
x=np.linspace(-10,10,100)
y=np.sin(x)
plt.plot(x,y,color='black')
```

```
[<matplotlib.lines.Line2D at 0x1fe81a720d0>]
```

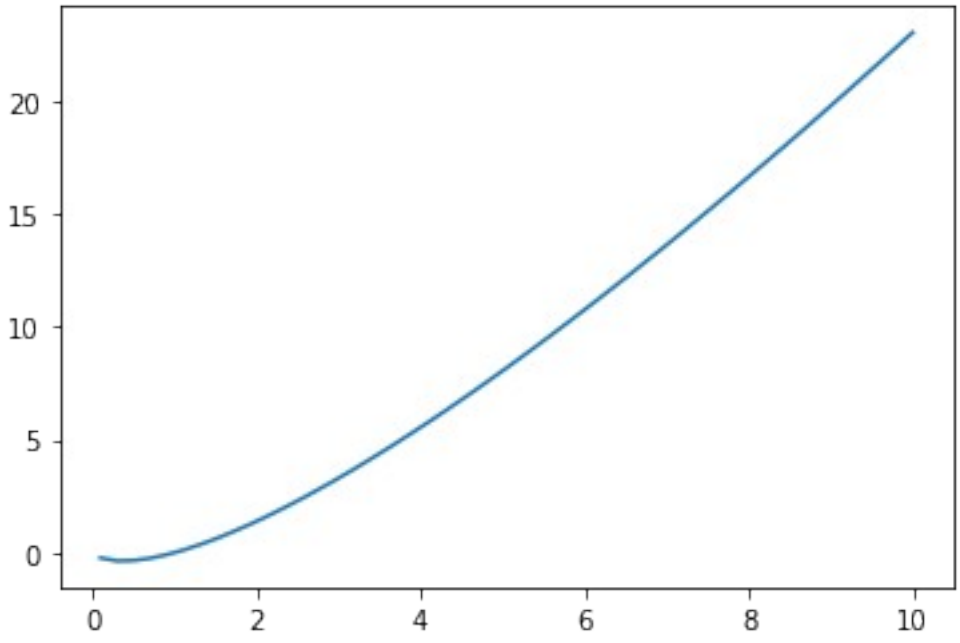


- y=xlog(x)

```
x=np.linspace(-10,10,100)
y=x*(np.log(x))
plt.plot(x,y)
```

```
C:\Users\DELL\AppData\Local\Temp/ipykernel_8128/353753872.py:2:
RuntimeWarning: invalid value encountered in log
  y=x*(np.log(x))
```

```
[<matplotlib.lines.Line2D at 0x1fe81adf550>]
```


20
15
10
5
0
0
2
4
6
8
10