AI TRAINING SERIES

Vector & Matrix

Vector & matrix

CONTENTS

Python hands on

- Basic
- Numpy
- Matplot

Regression hands on

- Perceptron & linear regression

AI TRAINING SERIES

Python실습

기본문법/Numpy/Matlib

*Python기본 선언 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

변수 선언

In [2]:

```
a = 1 #int로 선언
b = 2. #float으로 선언
c = "String" #string으로 선언
print(a)
print(b)
print(c)
print(type(a))
print(type(b))
print(type(c))
```

```
1
2.0
String
<class 'int'>
<class 'float'>
<class 'str'>
```

함수 선언

```
def f(x, y):
    val = x + y
    return val
```

```
a = 1
b = 2.
d = f(a,b)
print(d)
```

```
3.0
```

익명 함수

```
f = lambda x,y : x + y
```

```
a = 1
b = 2.
d = f(a,b)
print(d)
```

```
3.0
```

*Python기본 타입 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

리스트

```
a = [1, 3, 4]
print(a)
a[0] = 9
print(a)

b = [1, 3, 'string']
print(b)
b.append(6.24)
print(b)

print(2*a)
print(b*2)
c = [a[i] + a[i] for i in range(len(a))]
print(c)
```

```
[1, 3, 4]
[9, 3, 4]
[1, 3, 'string']
[1, 3, 'string', 6.24]
[9, 3, 4, 9, 3, 4]
[1, 3, 'string', 6.24, 1, 3, 'string', 6.24]
[18, 6, 8]
```

튜플

```
a = (1, 2, 3)
print(a)
b = (1, 3, 'string')
print(b)

a[0] = 2
a.append(4)
```

```
(1, 2, 3)
(1, 3, 'string')

---------------------------------------------
-----------------------------
TypeError                                 Traceba
ck (most recent call last)
Cell In[14], line 6
      3 b = (1, 3, 'string')
      4 print(b)
----> 6 a[0] = 2
      7 a.append(4)

TypeError: 'tuple' object does not support item a
ssignment
```

딕셔너리

```
info = {'A' : 2.3, 'B' : 'C', 5 : 'D'}
print(info)

info['A'] = 5.2
print(info)

info['Hello'] = [1, 2, 3, 4, 'World.']
print(info)
```

```
{'A': 2.3, 'B': 'C', 5: 'D'}
{'A': 5.2, 'B': 'C', 5: 'D'}
{'A': 5.2, 'B': 'C', 5: 'D', 'Hello': [1, 2, 3, 4,
'World.']}
```

* Python기본 타입 튜플 (tuple) 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
thistuple = ("apple", "banana", "cherry")
print(thistuple[1])
for x in thistuple:
  print(x)
tuple1 = ("abc", 34, True, 40, "male","abc")
print(tuple1[2])
a=1,
b=1
print(type(a),type(b))
print(a,b)

#tuple 연산
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)
tuple3 = tuple1 + tuple2
tuple4=tuple2*2
print(tuple3,tuple4)
```

* Python기본 타입 튜플 (tuple) 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
#update tuple
thistuple = ("apple", "banana", "cherry")
print(thistuple)
y = ("orange",)
thistuple += y
print(thistuple)

thistuple = ("apple", "banana", "cherry")
print(thistuple)
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)
print(thistuple)

# tuple unpack
fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")
(green, yellow, *red) = fruits
print(green)
print(yellow)
print(red)

fruits = ("apple", "mango", "papaya", "pineapple", "cherry")
(green, *tropic, red) = fruits
print(green)
print(tropic)
print(red)
```

*Python기본 for-loop 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

for loop

```
items = [1, 2, 3, 4, 'Hello', 6.24]

for k in range(0, len(items)):
    print(items[k])
print('------------')
for item in items:
    print(item)
print('------------')
items = [[1,2], [3,4], [5,6]]
for item in items:
    print(item[0], item[1])
print('------------')
for item1, item2 in items:
    print(item1, item2)
print('------------')
info = {'A' : 1, 'B' : 2, 'C' : 3}
for key in info:
    print(key, info[key])
print('------------')
for key, value in info.items():
    print(key, value)
```

zip이 들어간 for loop

```
items1 = [[1,2], [3,4], [5,6]]
items2 = [['A','B'], ['C','D'], ['E','F']]
print(items1)
print(items2)
print('-----------------------------------')
for digits, characters in zip(items1, items2):
    print(digits, characters)
```

```
[[1, 2], [3, 4], [5, 6]]
[['A', 'B'], ['C', 'D'], ['E', 'F']]
-----------------------------------
[1, 2] ['A', 'B']
[3, 4] ['C', 'D']
[5, 6] ['E', 'F']
```

한 줄 for loop

```
a = []
for k in range(0,5):
    a.append(k)
print(a)
print('-----------------------------------')
a = [k for k in range(0,5)]
print(a)
print('-----------------------------------')
a = [k if (k+1)%2 else k*5+1 for k in range(0,5)]
print(a)
print('-----------------------------------')
a = [k for k in range(0,5) if k % 2 == 0]
print(a)
print('-----------------------------------')
a = {k : k*10 for k in range(0,5) }
print(a)
print('-----------------------------------')
a = [1, 3, 4]
c = [a[i] + a[i] for i in range(len(a))]
print(c)
```

```
[0, 1, 2, 3, 4]
---------------------------------
[0, 1, 2, 3, 4]
---------------------------------
[0, 6, 2, 16, 4]
---------------------------------
[0, 2, 4]
---------------------------------
{0: 0, 1: 10, 2: 20, 3: 30, 4: 40}
---------------------------------
[2, 6, 8]
```

*Python기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

enumerate

```
x = ('apple', 'banana', 'cherry')
y = enumerate(x)
print(list(y))
print('-----------------------------------')
for entry in enumerate(['A', 'B', 'C']):
    print(entry)
print('-----------------------------------')
for i, letter in enumerate(['A', 'B', 'C']):
    print(i, letter)
print('-----------------------------------')
for i, letter in enumerate(['A', 'B', 'C'], start=101):
    print(i, letter)
```

```
[(0, 'apple'), (1, 'banana'), (2, 'cherry')]
-----------------------------------
(0, 'A')
(1, 'B')
(2, 'C')
-----------------------------------
0 A
1 B
2 C
-----------------------------------
101 A
102 B
103 C
```

파일 쓰기/읽기

쓰기

```
filename = 'readme.txt'
file = open(filename, 'w')
file.write("Hello, World!")
file.close()
```

```
filename = 'readme.txt'
file = open(filename, 'r')
content = file.read()
print(content)
file.close()
```

```
Hello, World!
```

```
filename = 'readme.txt'
with open(filename, 'w') as file:
    file.write("Hello, World!")
```

```
filename = 'readme.txt'
with open(filename, 'r') as file:
    content = file.read()
    print(content)
```

```
Hello, World!
```

* Python기본 String format 실습

- 일반적으로 문자열은 ‘문자열 + 문자열’의 형태로 조합가능.

```
'str' + 'ing'
```

```
'string'
```

- 문자열과 수치를 조합하는 방법으로 일반적으로 ‘format’ 사용.
- 문자열 내부의 ‘{0}’, ‘{1}’, ‘{2}’ 부분을 ‘format’의 x,y,z변수로 대치하는 방법임.
- 또다른 방법으로 ‘수동 문자열 포멧팅’이라고 하는 f-string을 사용 할 수 있음.
- 자세한 내용은 아래 경로 참조

https://docs.python.org/ko/3.11/library/string.html#formatstrings

* Python기본 String format실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
print('{0} and {1}'.format('spam', 'eggs'))
print('{1} and {0}'.format('spam', 'eggs'))
print('This {food} is {adjective}.'.format(
      food='spam', adjective='absolutely horrible'))
print('The story of {0}, {1}, and {other}.'.format('Bill', 'Manfred',
                                                   other='Georg'))


s = 'coffee'
n = 5
result1 = f'저는 {s}를 좋아합니다. 하루 {n}잔 마셔요.'
print(result1)
```

Python lib

NumPy

Example

A 2-dimensional array of size 2 x 3, composed of 4-byte integer elements:

```
>>> x = np.array([[1, 2, 3], [4, 5, 6]], np.int32)
>>> type(x)
<class 'numpy.ndarray'>
>>> x.shape
(2, 3)
>>> x.dtype
dtype('int32')
```

The array can be indexed using Python container-like syntax:

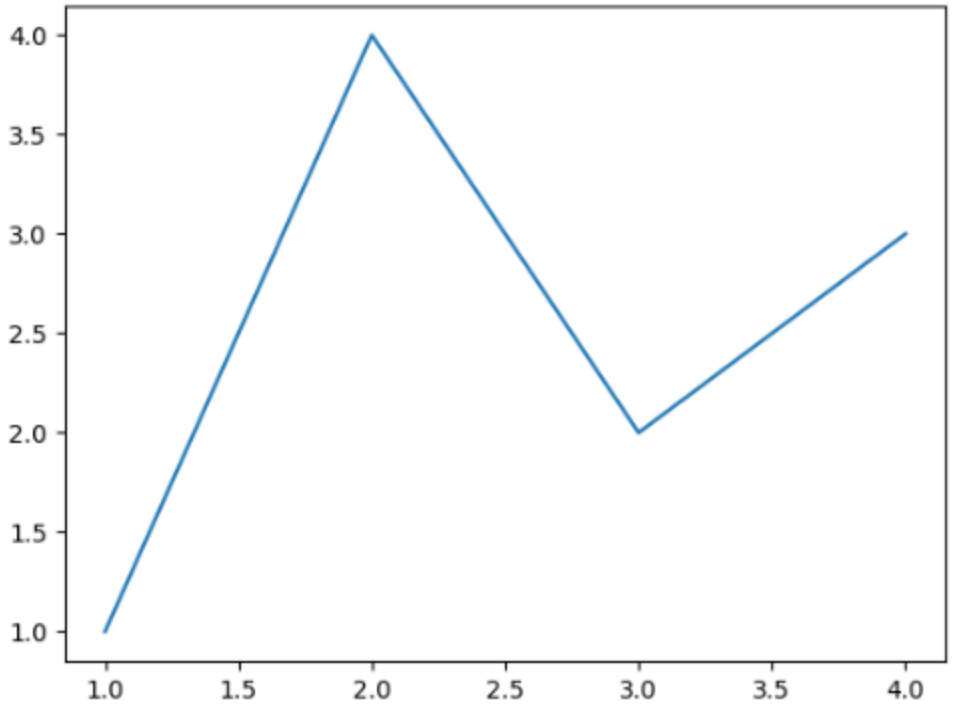
```
>>> # The element of x in the *second* row, *third* column, namely,
>>> x[1, 2]
6
```

For example slicing can produce views of the array:

```
>>> y = x[:,1]
>>> y
array([2, 5], dtype=int32)
>>> y[0] = 9 # this also changes the corresponding element in x
>>> y
array([9, 5], dtype=int32)
>>> x
array([[1, 9, 3],
       [4, 5, 6]], dtype=int32)
```

https://numpy.org/doc/stable/reference/

matplotlib



*Python numpy 에서의 벡터

- 벡터는 보통 numpy에서 제공하는 np.array(list형)으로 정의
- Python에서 제공하는 list와는 약간 다르게 사용
- 요소의 참조는 대괄호를 사용하여 [0]부터 참조
- np.arrange(a,b)문을 이용하여 a부터 b-1까지의 1차원 배열을 만들 수 있음.
- 비슷한 type으로 튜플(tuple)이 있으나 내부원소를 수정할 수 없으며 추가, 삭제만 가능

*Numpy기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

numpy 배열

```
import numpy as np
a = np.array([1,2,3,4])
print(a)
print(a + a)
```

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

일반 배열

```
b = [1,2,3,4]
print(b + b)
```

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

이중 배열(행렬)

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

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

삼중 배열(행렬)

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

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

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

배열의 모양(Shape)

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

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

Norm

```
import numpy as np
from numpy import linalg as LA
c = np.array([[ 1, 2, 3],
              [-1, 1, 4]])
print(LA.norm(c, axis=0))
print(LA.norm(c, axis=1))
print(LA.norm(c, ord=1, axis=1))
print(LA.norm(c, ord=2, axis=1))
```

```
[1.41421356 2.23606798 5.        ]
[3.74165739 4.24264069]
[6. 6.]
[3.74165739 4.24264069]
```

- L1 놈: 벡터의 원소의 절대값의 합
- L2 놈: 유클리디안 거리로 계산된 벡터의 길이
- 무한 놈: 벡터의 원소 중 절대값이 가장 큰 값

*Numpy기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

전치연산(Transpose)

```
a = np.array([[1],[2],[3],[4]])
print(a) #shape = (4,1)
print(a.T) #shape = (1,4)
print(a.T.reshape(-1,4))
print(a.shape)
print(a.T.reshape(-1,4).T.shape)
```

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

```
a = np.array([1,2,3,4])
b= a.reshape(4,-1)
print(a)
print(a.reshape(2,-1))
print(a.shape,",",b.shape,",",np.array([[1,2,3,4]]).shape)
```

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

Reshape

```
a = np.array([1,2,3,4,5,6])
print(a.reshape(3,2))
print(a.shape)
b=a.reshape(3,-1)
print(b)
print(b.shape)
c=a.reshape(-1,2)
print(c)
print(c.shape)
```

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

```
a = np.array([1,2,3,4])
print(a)
print(a.T)
b=a.reshape(4,-1)
print(b.shape)
print(b)
print(b.T.shape)
```

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

*Numpy기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
# Download the image from the openvino_notebooks storage
image_filename = download_file(
    "https://storage.openvinotoolkit.org/repositories/openvino_notebooks/data/data/image/empty_road_mapillary.jpg",
    directory="data"
)

# The segmentation network expects images in BGR format.
image = cv2.imread(str(image_filename))

rgb_image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image_h, image_w, _ = image.shape

# N,C,H,W = batch size, number of channels, height, width.
N, C, H, W = input_layer_ir.shape

# OpenCV resize expects the destination size as (width, height).
resized_image = cv2.resize(image, (W, H))

# Reshape to the network input shape.
input_image = np.expand_dims(
    resized_image.transpose(2, 0, 1), 0
)
plt.imshow(rgb_image)
```

* Numpy기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

배열 인덱싱

```
a = np.array([10,20,30,40,50,60])
print(a)
b = a[[4,2,0]]
print(b)
idx = np.arange(0, len(a))
print(idx)
np.random.shuffle(idx)
print(idx)
print(a[idx])
```

```
[10 20 30 40 50 60]
[50 30 10]
[0 1 2 3 4 5]
[2 1 4 5 3 0]
[30 20 50 60 40 10]
```

```
import numpy as np
c = np.array([ 1, 2, 3, 4, 5, 6])
print(c[0])
print(c[5])
print(c[-1])
print(c[-2])
print(c[-6])
print(c[0:2])
print(c[2:5])
print(c[:2])
print(c[4:])
print(c[-2:])
print(c[:2] , c[2:])
print(c[:4] , c[4:])
print(np.arange(5,10))
```

* Numpy기본 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
word = 'Python'
print(word[0])   # character in position 0    'P'
print(word[5])  # character in position 5    'n'
print(word[-1])  # last character    'n'
print(word[-2]) # second-last character   'o'
print(word[-6])# 'P'
print(word[0:2])  # characters from position 0 (included) to 2 (excluded)    'Py'
print(word[2:5])  # characters from position 2 (included) to 5 (excluded)    'tho'
print(word[:2])   # character from the beginning to position 2 (excluded)    'Py'
print(word[4:])   # characters from position 4 (included) to the end   'on'
print(word[-2:])  # characters from the second-last (included) to the end   'on'
print(word[:2] + word[2:])  # 'Python'
print(word[:4] + word[4:])  # 'Python'
```

*Numpy matrix 실습

Example (Matrix-Vector Multiplication)

Write code for the follwoing problem.

$$\begin{bmatrix} 1 & 4 & 2 & 0 \\ 9 & 5 & 0 & 0 \\ 4 & 0 & 2 & 4 \\ 6 & 1 & 8 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = ?$$

```python
A = np.array([[1, 4, 2, 0], [9, 5, 0, 0], [4, 0, 2, 4], [6, 1, 8, 3]])
x = np.array([1,2,3,4])
b = np.array([0,0,0,0])
n = 4
for i in range(0, n):
    val = 0.0
    for j in range(0,n):
        # TODO 2
        val += A[i,j] * x[j]
    b[i] = val

print("calculate=", b)

b = np.dot(A,x)
print("dot=",b)

b = np.matmul(A,x)
print("matmul=",b)

b =  A@x
print("A@x=",b)

b =  A*x
print("A*x=",b)
```

```
calculate= [15 19 26 44]
dot= [15 19 26 44]
matmul= [15 19 26 44]
A@x= [15 19 26 44]
A*x= [[ 1  8  6  0]
 [ 9 10  0  0]
 [ 4  0  6 16]
 [ 6  2 24 12]]
```

*Numpy matrix 실습

Example (Soution of the Linear System)

1. Determine if this linear system has a solution.
2. Write code to find a solution, x, of the following linear system code using `numpy`.

$$\begin{bmatrix} 1 & 4 & 2 & 0 \\ 9 & 5 & 0 & 0 \\ 4 & 0 & 2 & 4 \\ 6 & 1 & 8 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 15 \\ 19 \\ 26 \\ 44 \end{bmatrix}$$

```
import numpy as np
A = np.array([[1, 4, 2, 0], [9, 5, 0, 0], [4, 0, 2, 4], [6, 1, 8, 3]])
b = np.array([15,19,26,44])

print("det=",np.linalg.det(A))

x = np.linalg.solve(A,b)
print("solver =",x)

x = np.dot(np.linalg.inv(A),b)
print("inverse1 =",x)

tmp_b=np.dot(A.T,b)
tmp_T=np.dot(A.T,A)
tmp_inv=np.linalg.inv(np.dot(A.T,A))
x = np.dot(tmp_inv,tmp_b)
print("inverse2 =",x)
```

```
det= 853.9999999999995
solver = [1. 2. 3. 4.]
inverse1 = [1. 2. 3. 4.]
inverse2 = [1. 2. 3. 4.]
```

* Numpy기본 실습 : 고유값

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

$$1)\ A=\begin{bmatrix}2 & -1\\ -1 & 2\end{bmatrix} \qquad \begin{bmatrix}2-\lambda & -1\\ -1 & 2-\lambda\end{bmatrix} \overset{det}{\Rightarrow} (2-\lambda)(2-\lambda)-1=0 \rightarrow\ \ \lambda=1,3\ \ v=\begin{bmatrix}1\\1\end{bmatrix},\begin{bmatrix}1\\-1\end{bmatrix}$$

```python
from numpy import linalg as LA
import numpy as np
import matplotlib.pylab as plt
```

```python
A = np.array([[2, -1], [-1, 2]])
eigenvalues, eigenvectors = LA.eig(A)
x=eigenvectors
lamda=eigenvalues
y1=A@x[:,0]
y2=A@x[:,1]
print("A:₩n",A,"₩neigen value0:",lamda[0],
      "₩neigen vector0:₩n",x[:,0],"₩nAx:₩n",y1,
      "₩neigen value1:",lamda[1],
      "₩neigen vector1:₩n",x[:,1],"₩nAx:₩n",y2)
print("₩n--------randowm array-----₩n")

x_test=np.random.rand(2,1)
y_test=A@x_test
print("test vector:₩n",x_test,"₩nAx_test:₩n",y_test)

plt.subplot(3,3,1)
plt.plot([0,x[0,0]],[0,x[1,0]], 'go-', label='eigenVector0', linewidth=2)
plt.plot([0,y1[0]],[0,y1[1]], 'rs-', label='eigenVector0 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[0]}')
```

```python
plt.subplot(3,3,2)
plt.plot([0,x[0,1]],[0,x[1,1]], 'go-', label='eigenVector1', linewidth=2)
plt.plot([0,y2[0]],[0,y2[1]], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[1]}')

plt.subplot(3,3,3)
plt.plot([0,x_test[0,0]],[0,x_test[1,0]], 'go-', label='eigenVector1',linewidth=2)
plt.plot([0,y_test[0,0]],[0,y_test[1,0]], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title('random')

plt.show()
```

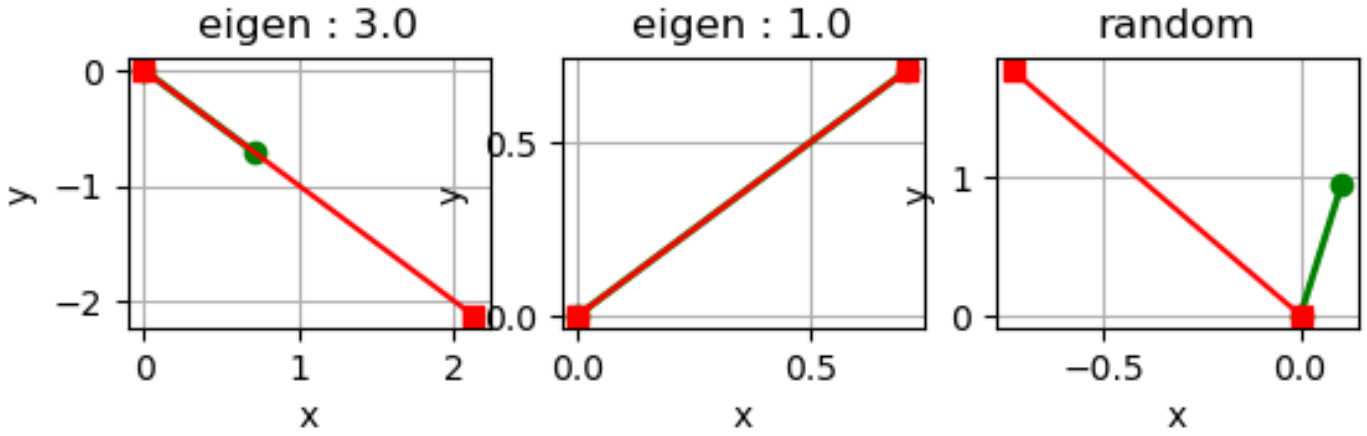
* Numpy기본 실습 : 고유값

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

$$1)\ A = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \qquad \begin{bmatrix} 2-\lambda & -1 \\ -1 & 2-\lambda \end{bmatrix} \overset{det}{\Rightarrow} (2-\lambda)(2-\lambda) - 1 = 0 \rightarrow \ \lambda = 1, 3 \ \ v = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

```
A:
 [[ 2 -1]
 [-1  2]]
eigen value0: 3.0
eigen vector0:
 [ 0.70710678 -0.70710678]
Ax:
 [ 2.12132034 -2.12132034]
eigen value1: 1.0
eigen vector1:
 [0.70710678 0.70710678]
Ax:
 [0.70710678 0.70710678]
```

```
--------randowm array-----

test vector:
 [[0.10349652]
 [0.9364154 ]]
Ax_test:
 [[-0.72942236]
 [ 1.76933428]]
```



* Numpy기본 실습 : 고유값

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

$$2)\ A=\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix} \qquad \begin{bmatrix}2-\lambda & 0\\ 0 & 2-\lambda\end{bmatrix} \overset{det}{\Rightarrow} (2-\lambda)(2-\lambda)=0 \rightarrow \ \lambda=2 \quad v=\begin{bmatrix}1\\0\end{bmatrix},\begin{bmatrix}0\\1\end{bmatrix},\cdots$$

```python
A = np.array([[2, 0], [0, 2]])
eigenvalues, eigenvectors = LA.eig(A)
x=eigenvectors
lamda=eigenvalues
y1=A@x[:,0]
y2=A@x[:,1]
print("A:₩n",A,"₩neigen value0:",lamda[0],
      "₩neigen vector0:₩n",x[:,0],"₩nAx:₩n",y1,
      "₩neigen value1:",lamda[1],
      "₩neigen vector1:₩n",x[:,1],"₩nAx:₩n",y2)
print("₩n--------randowm array-----₩n")


x_test=np.random.rand(2,1)
y_test=A@x_test
print("test vector:₩n",x_test,"₩nAx_test:₩n",y_test)

plt.subplot(3,3,1)
plt.plot([0,x[0,0]],[0,x[1,0]], 'go-', label='eigenVector0', linewidth=2)
plt.plot([0,y1[0]],[0,y1[1]], 'rs-', label='eigenVector0 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[0]}')
```

```python
plt.subplot(3,3,2)
plt.plot([0,x[0,1]],[0,x[1,1]], 'go-', label='eigenVector1', linewidth=2)
plt.plot([0,y2[0]],[0,y2[1]], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[1]}')

plt.subplot(3,3,3)
plt.plot([0,x_test[0,0]],[0,x_test[1,0]], 'go-', label='eigenVector1', linewidth=2)
plt.plot([0,y_test[0,0]],[0,y_test[1,0]], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title('random')
plt.show()
```

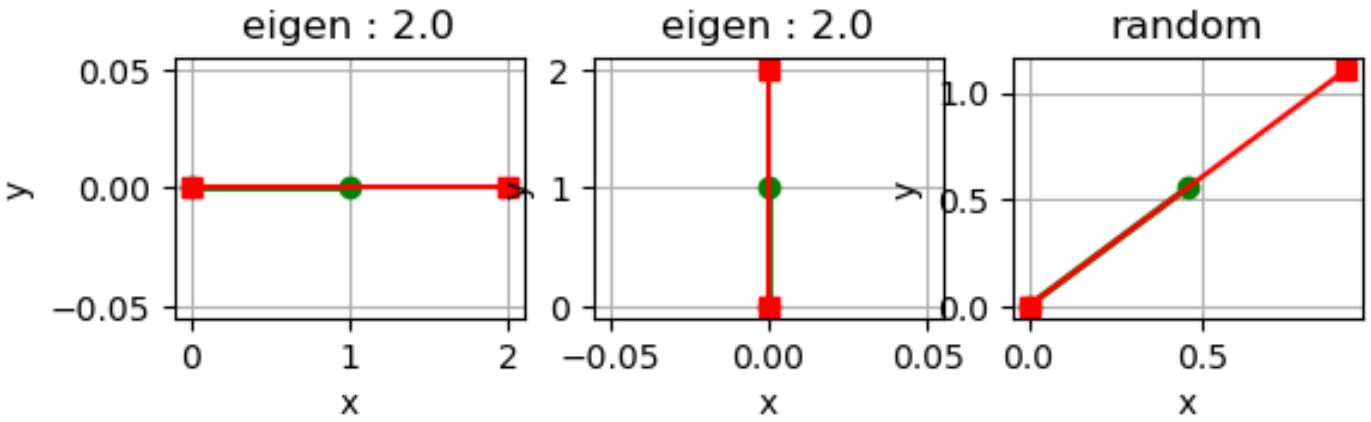
* Numpy기본 실습 : 고유값

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

2) $A = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ $\begin{bmatrix} 2-\lambda & 0 \\ 0 & 2-\lambda \end{bmatrix} \overset{det}{\Rightarrow} (2-\lambda)(2-\lambda) = 0 \rightarrow \ \lambda = 2 \ \ v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \cdots$

```
A:
 [[2 0]
 [0 2]]
eigen value0: 2.0
eigen vector0:
 [1. 0.]
Ax:
 [2. 0.]
eigen value1: 2.0
eigen vector1:
 [0. 1.]
Ax:
 [0. 2.]
```

```
--------randowm array-----

test vector:
 [[0.45753573]
 [0.55387363]]
Ax_test:
 [[0.91507147]
 [1.10774725]]
```



* Numpy기본 실습 : 고유값

3) $A = \begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix}$ $\begin{bmatrix} -\lambda & -2 \\ 2 & -\lambda \end{bmatrix}$ $\overset{det}{\Rightarrow}$ $\lambda^2 + 4 = 0$

$$\begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix} \rightarrow c \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

```python
A = np.array([[0, -2], [2, 0]])
eigenvalues, eigenvectors = LA.eig(A)
x=eigenvectors
lamda=eigenvalues
y1=A@x[:,0]
y2=A@x[:,1]

print("A:₩n",A,"₩neigen value0:",np.round(lamda[0],2),
      "₩neigen vector0:₩n",x[:,0],"₩nAx:₩n",y1,
      "₩neigen value1:",np.round(lamda[1],2),
      "₩neigen vector1:₩n",x[:,1],"₩nAx:₩n",y2)
print("₩n--------randowm array-----₩n")

x_test=np.random.rand(2,1)
y_test=A@x_test
print("test vector:₩n",x_test,"₩nAx_test:₩n",y_test)

plt.subplot(3,3,1)
plt.plot([0,x[0,0]],[0,np.abs(x[1,0])], 'go-', label='eigenVector0', linewidth=2)
plt.plot([0,y1[0]],[0,np.abs(y1[1])], 'rs-', label='eigenVector0 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[0]:.1f}')
```

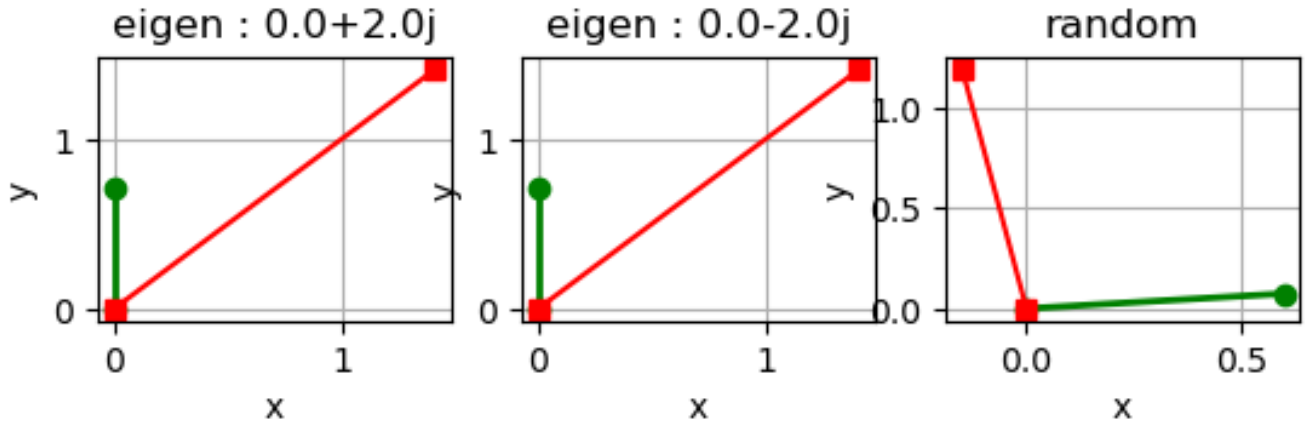
```python
plt.subplot(3,3,2)
plt.plot([0,x[0,1]],[0,np.abs(x[1,1])], 'go-', label='eigenVector1', linewidth=2)
plt.plot([0,y2[0]],[0,np.abs(y2[1])], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'eigen : {lamda[1]:.1f}')

plt.subplot(3,3,3)
plt.plot([0,x_test[0,0]],[0,x_test[1,0]], 'go-', label='eigenVector1', linewidth=2)
plt.plot([0,y_test[0,0]],[0,y_test[1,0]], 'rs-', label='eigenVector1 result')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title('random')
plt.show()
```

* Numpy기본 실습 : 고유값

3) $A = \begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix}$ $\begin{bmatrix} -\lambda & -2 \\ 2 & -\lambda \end{bmatrix}$ $\overset{det}{\Rightarrow}$ $\lambda^2 + 4 = 0$ $\begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix} \rightarrow c \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

```
A:
 [[ 0 -2]
 [ 2  0]]
eigen value0: 2j
eigen vector0:
 [ 0.         -0.70710678j -0.70710678+0.j        ]
Ax:
 [1.41421356+0.j         0.         -1.41421356j]
eigen value1: -2j
eigen vector1:
 [ 0.         +0.70710678j -0.70710678-0.j        ]
Ax:
 [1.41421356+0.j         0.         +1.41421356j]
```

```
--------randowm array-----

test vector:
 [[0.59580948]
 [0.07439715]]
Ax_test:
 [[-0.14879429]
 [ 1.19161895]]
```

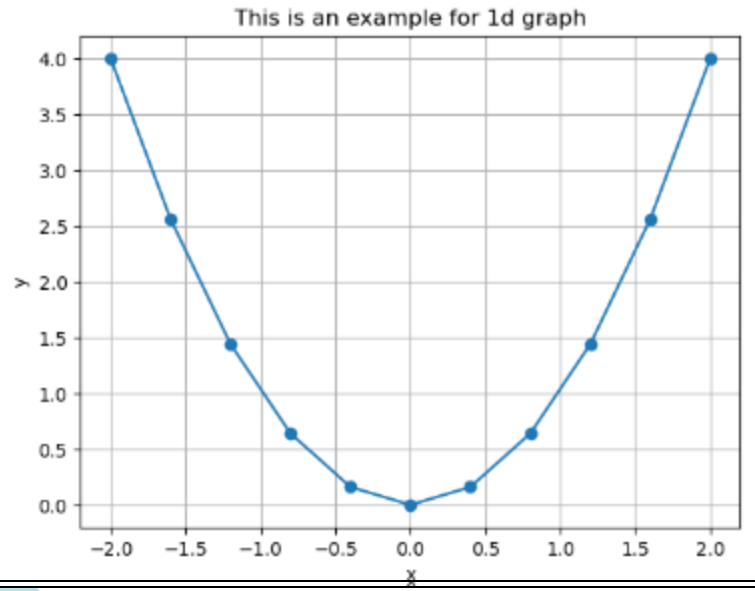


*Matplotlib실습 : line plot, surface

```
import numpy as np
import matplotlib.pylab as plt
```

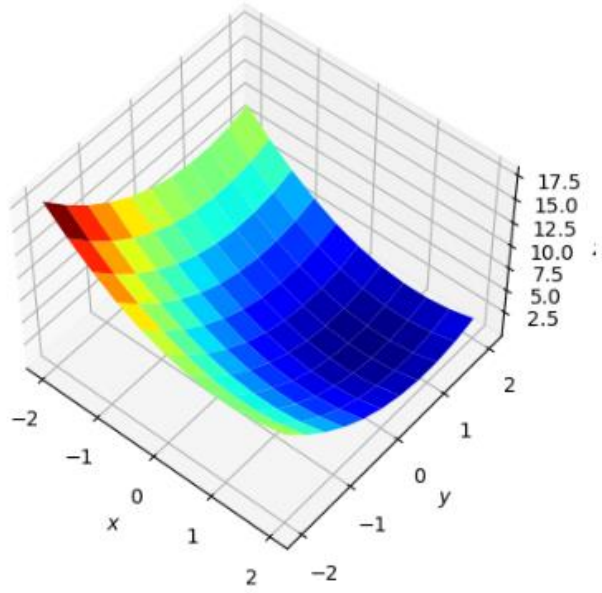
```
x = np.linspace(-2, 2, 11)
f = lambda x: x ** 2
fx = f(x)
print(x)
print(fx)
```

```
plt.plot(x, fx, '-o')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title('This is an example for 1d graph')
plt.show()
```




```
import numpy as np
import matplotlib.pylab as plt
x = np.linspace(-2,2, 11)
y = np.linspace(-2,2, 11)

print(x)
print(y)

x,y = np.meshgrid(x,y)
print(x)
print(y)

f = lambda x,y : (x-1)**2 + (y-1)**2
z = f(x,y)
print(z)
```

```
from mpl_toolkits.mplot3d import Axes3D

ax = plt.axes(projection='3d', elev=50, azim=-50)
ax.plot_surface(x, y, z, cmap=plt.cm.jet)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')

plt.show()
```



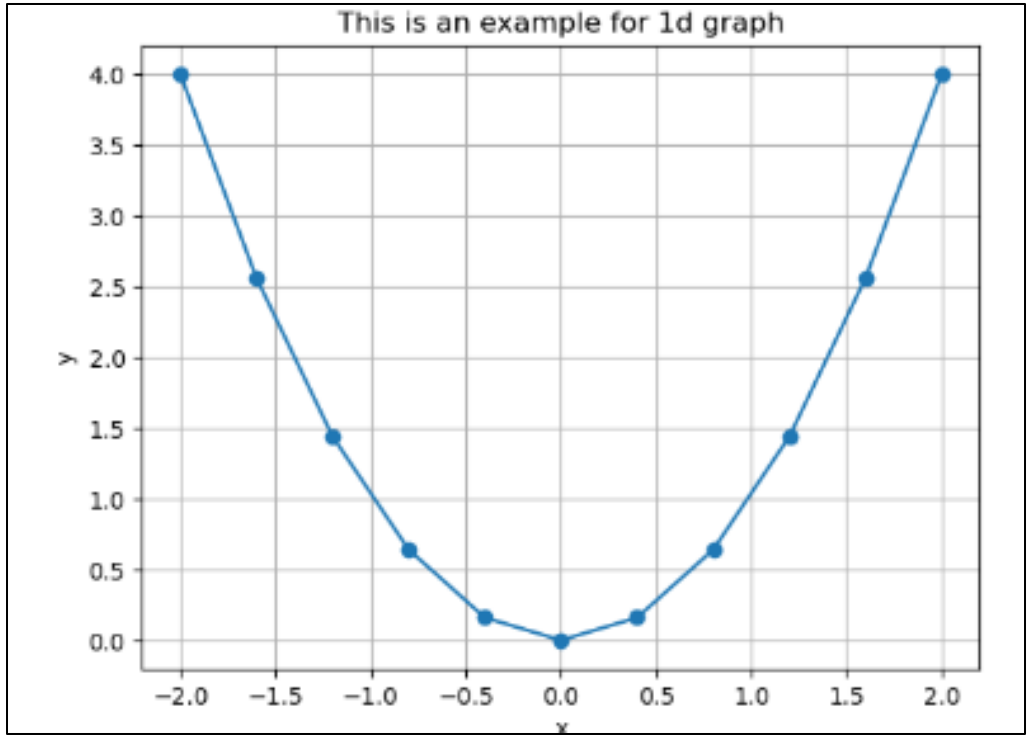
http://chebe163.caltech.edu/2018w/handouts/intro_to_latex.html

*Matplotlib실습 : line plot, surface

```
import numpy as np
import matplotlib.pylab as plt
```

```
x = np.linspace(-2, 2, 11)
f = lambda x: x ** 2
fx = f(x)
print(x)
print(fx)
```

```
plt.plot(x, fx, '-o')
plt.grid()
plt.xlabel('x')
plt.ylabel('y')
plt.title('This is an example for 1d graph')
plt.show()
```

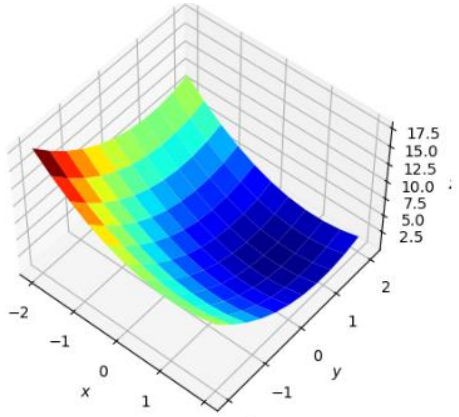


http://chebe163.caltech.edu/2018w/handouts/intro_to_latex.html

*Matplotlib실습 : line plot, surface

```
import numpy as np
import matplotlib.pylab as plt
x = np.linspace(-2,2, 11)
y = np.linspace(-2,2, 11)

print(x)
print(y)

x,y = np.meshgrid(x,y)
print(x)
print(y)

f = lambda x,y : (x-1)**2 + (y-1)**2
z = f(x,y)
print(z)
```

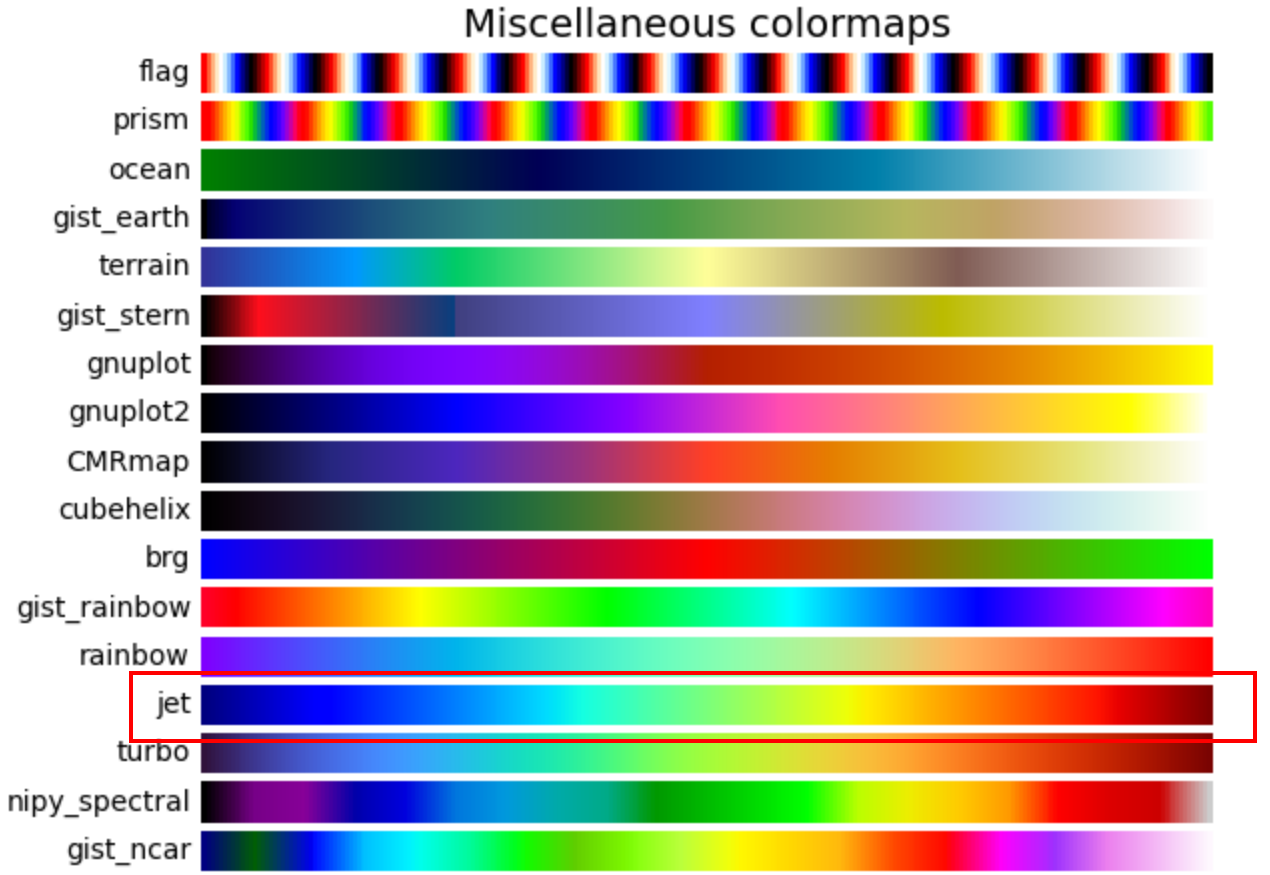
```
from mpl_toolkits.mplot3d import Axes3D

ax = plt.axes(projection='3d', elev=50, azim=-50)
ax.plot_surface(x, y, z, cmap=plt.cm.jet)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')

plt.show()
```



http://chebe163.caltech.edu/2018w/handouts/intro_to_latex.html

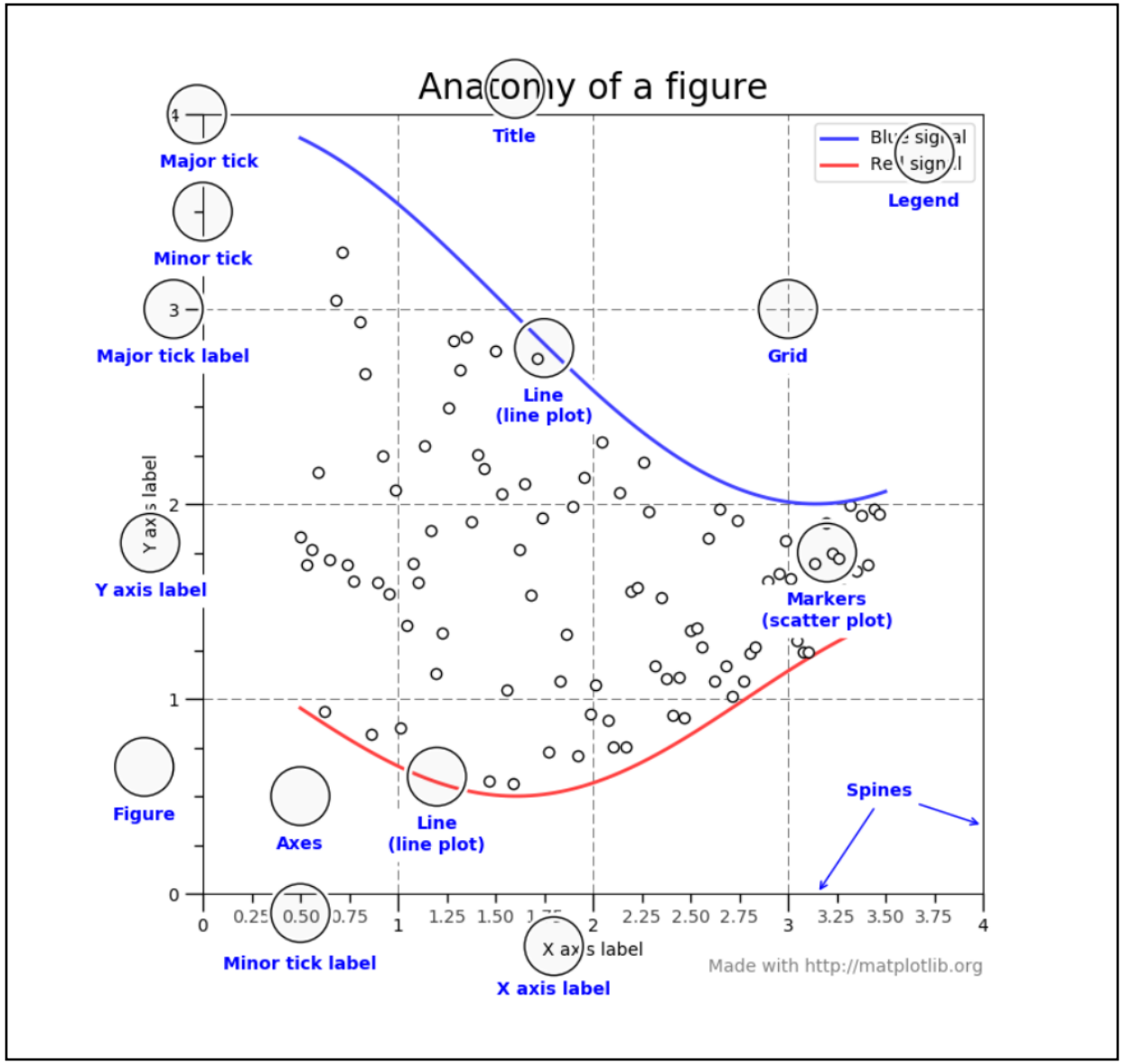
*Matplotlib실습 : cmap

```
import matplotlib.pyplot as plt
import numpy as np

cmaps = [('Perceptually Uniform Sequential', [
            'viridis', 'plasma', 'inferno', 'magma', 'cividis']),
         ('Sequential', [
            'Greys', 'Purples', 'Blues', 'Greens', 'Oranges', 'Reds',
            'YlOrBr', 'YlOrRd', 'OrRd', 'PuRd', 'RdPu', 'BuPu',
            'GnBu', 'PuBu', 'YlGnBu', 'PuBuGn', 'BuGn', 'YlGn']),
         ('Sequential (2)', [
            'binary', 'gist_yarg', 'gist_gray', 'gray', 'bone', 'pink',
            'spring', 'summer', 'autumn', 'winter', 'cool', 'Wistia',
            'hot', 'afmhot', 'gist_heat', 'copper']),
         ('Diverging', [
            'PiYG', 'PRGn', 'BrBG', 'PuOr', 'RdGy', 'RdBu',
            'RdYlBu', 'RdYlGn', 'Spectral', 'coolwarm', 'bwr', 'seismic']),
         ('Cyclic', ['twilight', 'twilight_shifted', 'hsv']),
         ('Qualitative', [
            'Pastel1', 'Pastel2', 'Paired', 'Accent',
            'Dark2', 'Set1', 'Set2', 'Set3',
            'tab10', 'tab20', 'tab20b', 'tab20c']),
         ('Miscellaneous', [
            'flag', 'prism', 'ocean', 'gist_earth', 'terrain', 'gist_stern',
            'gnuplot', 'gnuplot2', 'CMRmap', 'cubehelix', 'brg',
            'gist_rainbow', 'rainbow', 'jet', 'turbo', 'nipy_spectral',
            'gist_ncar'])]

gradient = np.linspace(0, 1, 256)
gradient = np.vstack((gradient, gradient))


def plot_color_gradients(cmap_category, cmap_list):
    # Create figure and adjust figure height to number of colormaps
    nrows = len(cmap_list)
    figh = 0.35 + 0.15 + (nrows + (nrows-1)*0.1)*0.22
    fig, axs = plt.subplots(nrows=nrows, figsize=(6.4, figh))
    fig.subplots_adjust(top=1-.35/figh, bottom=.15/figh, left=0.2, right=0.99)

    axs[0].set_title(f"{cmap_category} colormaps", fontsize=14)

    for ax, cmap_name in zip(axs, cmap_list):
        ax.imshow(gradient, aspect='auto', cmap=cmap_name)
        ax.text(-.01, .5, cmap_name, va='center', ha='right', fontsize=10,
                transform=ax.transAxes)

    # Turn off *all* ticks & spines, not just the ones with colormaps.
    for ax in axs:
        ax.set_axis_off()


for cmap_category, cmap_list in cmaps:
    plot_color_gradients(cmap_category, cmap_list)
```



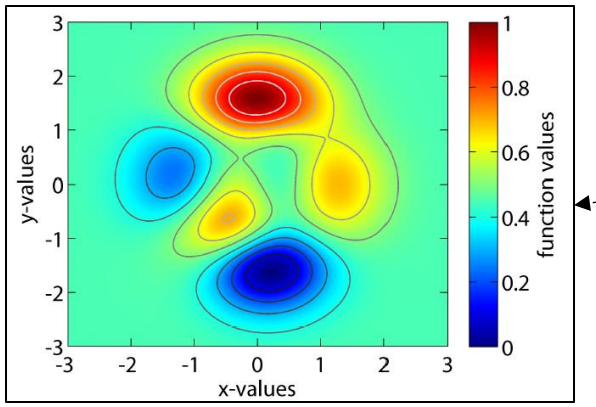
https://matplotlib.org/stable/gallery/color/colormap_reference.html

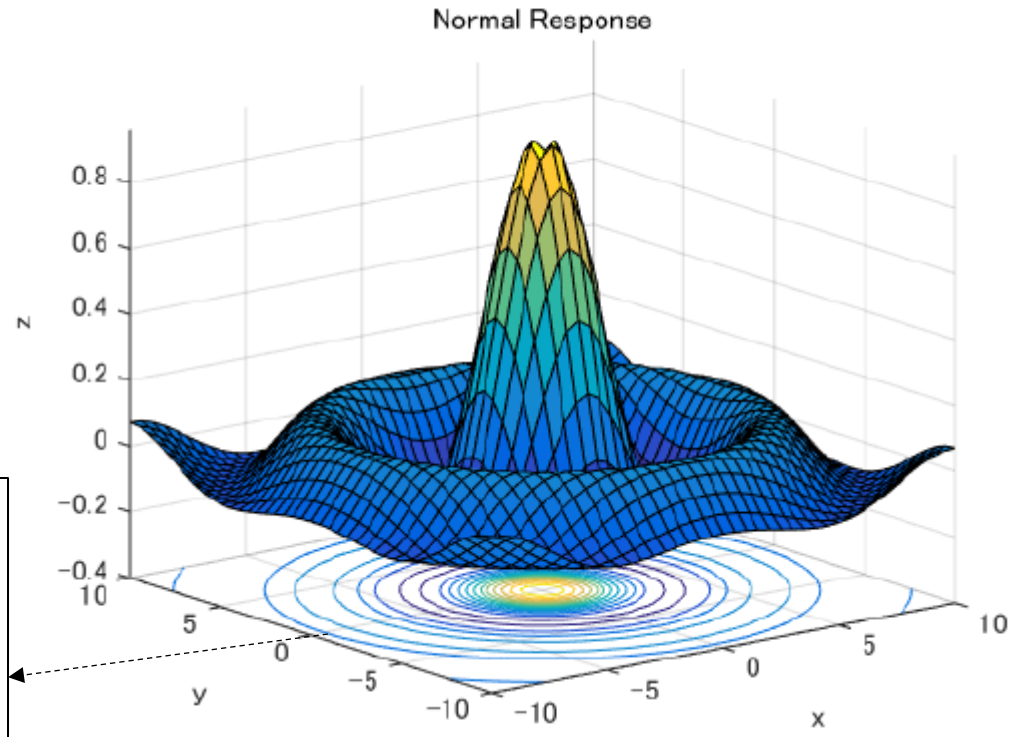
*Matplotlib실습 : matplot해부도



*Matplotlib실습 : contour

$$f(x, y) = c$$





*Matplotlib실습 : contour, quiver

```
ax = plt.axes()
ax.contour(x, y, z, levels=np.linspace(0, 20, 20), cmap=plt.cm.jet)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```

*Matplotlib실습 : contour, quiver

```
grad_f_x = lambda x, y: 2 * (x-1)
grad_f_y = lambda x, y: 2 * (y-1)
```

```
dz_dx = grad_f_x(x,y)
dz_dy = grad_f_y(x,y)
```

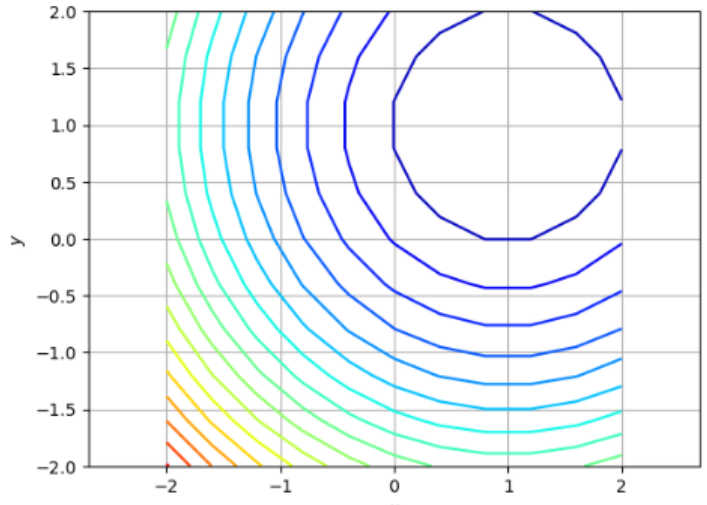
```
ax = plt.axes()
ax.quiver(x, y, -dz_dx, -dz_dy)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```

*Matplotlib실습 : contour, quiver

```
ax = plt.axes()
ax.contour(x, y, z, levels=np.linspace(0, 10, 20), cmap=plt.cm.jet)
ax.quiver(x, y, -dz_dx, -dz_dy)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```

*Matplotlib실습 : contour, quiver

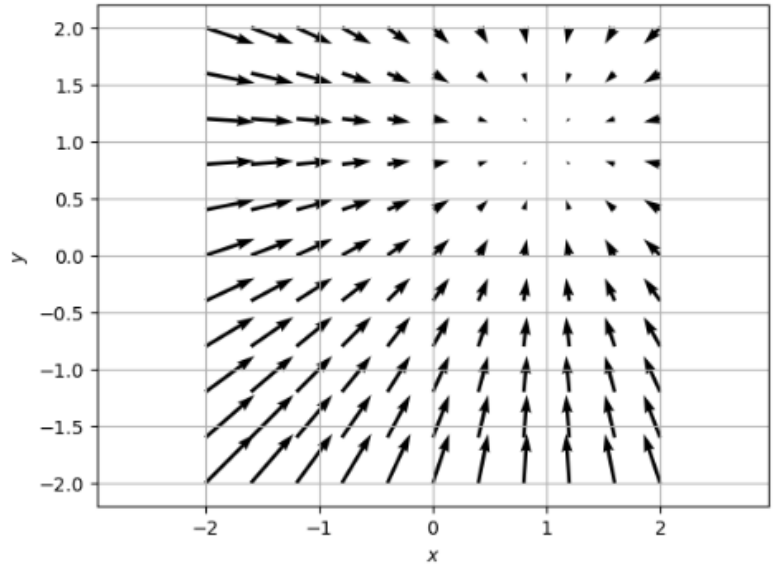
```
ax = plt.axes()
ax.contour(x, y, z, levels=np.linspace(0, 20, 20), cmap=plt.cm.jet)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```



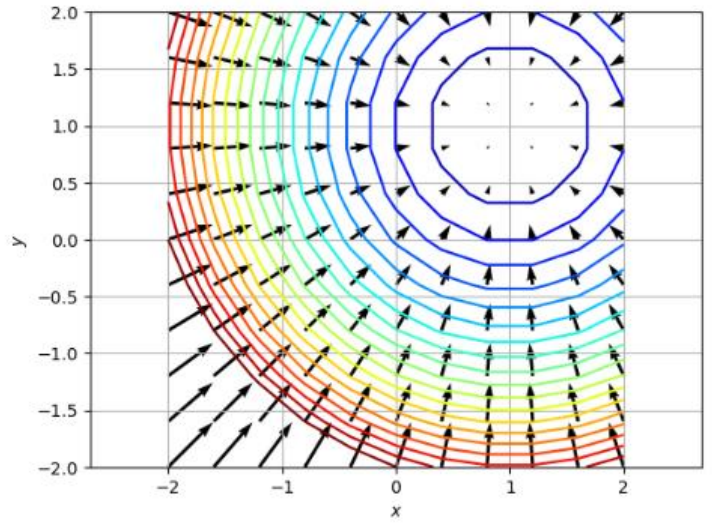

```
grad_f_x = lambda x, y: 2 * (x-1)
grad_f_y = lambda x, y: 2 * (y-1)
```

```
dz_dx = grad_f_x(x,y)
dz_dy = grad_f_y(x,y)
```

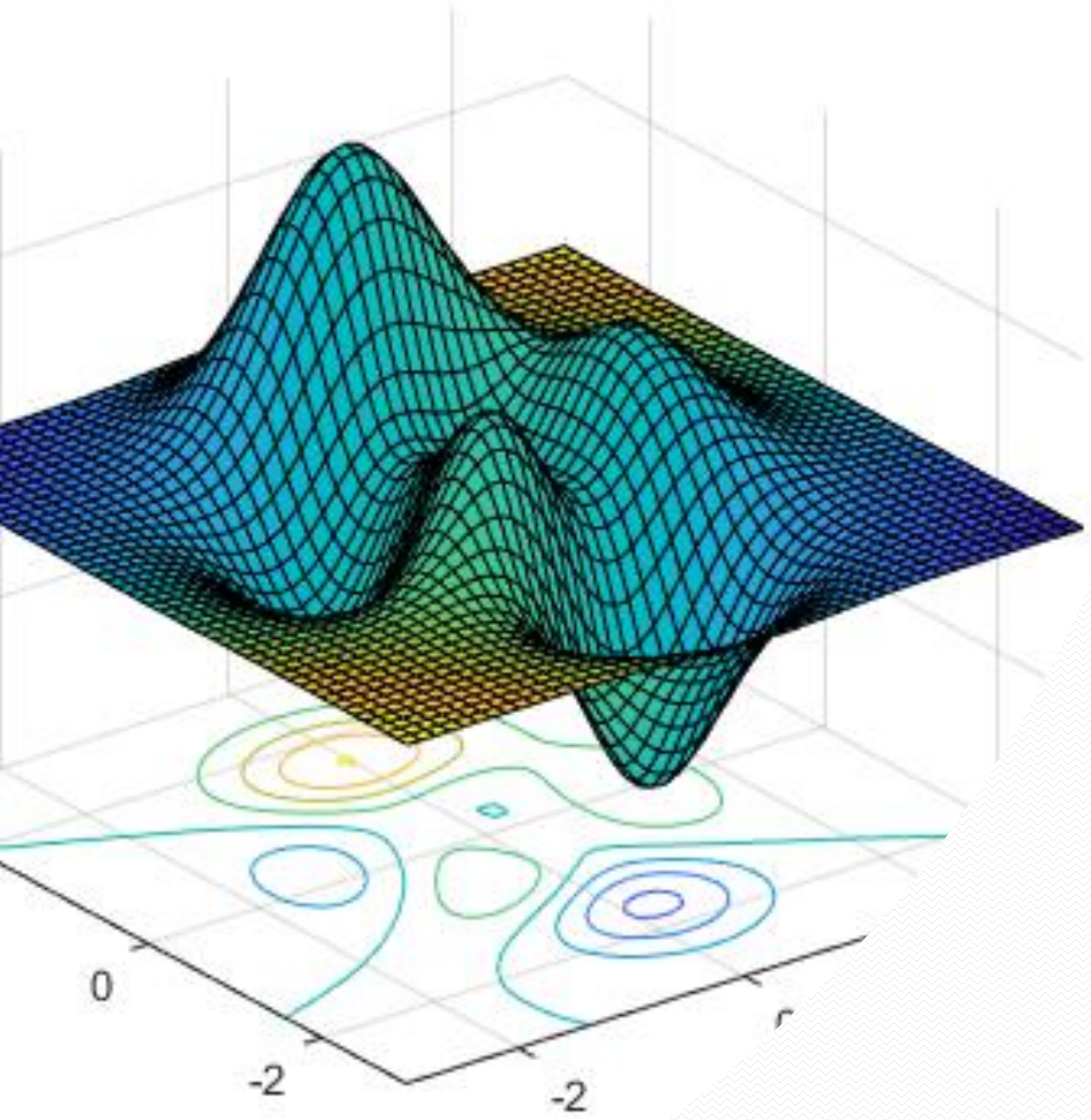
```
ax = plt.axes()
ax.quiver(x, y, -dz_dx, -dz_dy)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```




```
ax = plt.axes()
ax.contour(x, y, z, levels=np.linspace(0, 10, 20), cmap=plt.cm.jet)
ax.quiver(x, y, -dz_dx, -dz_dy)
ax.grid()
ax.axis('equal')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
plt.show()
```

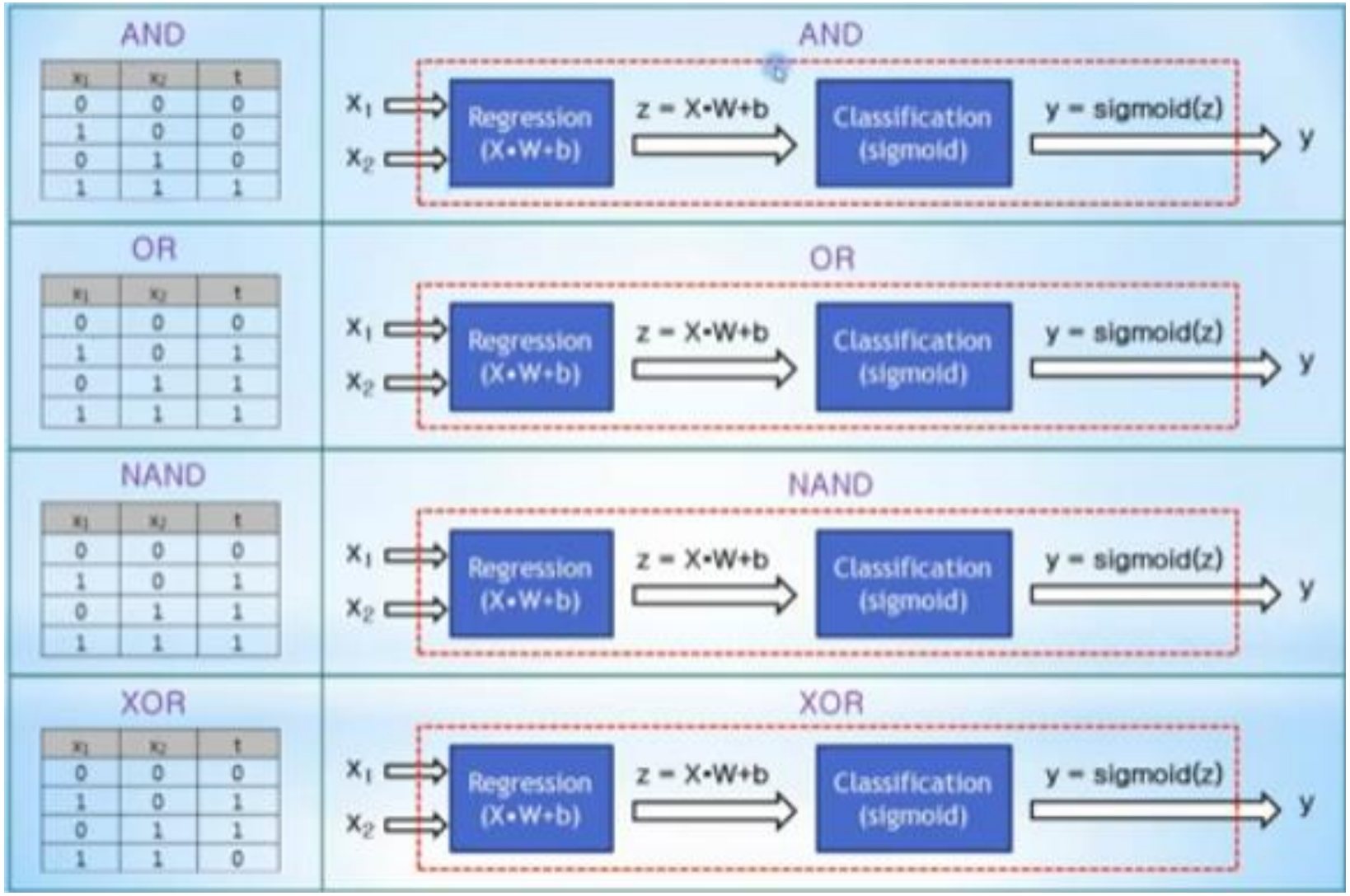


Session Break



Hands on

*perceptron 실습



*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```python
import numpy as np
```

```python
def sigmoid(x):
    return 1/(1+np.exp(-x))
```

```python
def numerical_derivative(f,x):
    delta_x=1e-4
    gradf=np.zeros_like(x)

    it = np.nditer(x,flags=['multi_index'],op_flags=['readwrite'])

    while not it.finished:
        idx=it.multi_index
        tmp_val=x[idx]
        x[idx]=float(tmp_val)+delta_x
        fx1=f(x)

        x[idx]=float(tmp_val)-delta_x
        fx2=f(x)
        gradf[idx]=(fx1-fx2)/(2*delta_x)

        x[idx]=tmp_val
        it.iternext()
    return gradf
```

*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
class logicGate:
    def __init__(self,gate_name,xdata,tdata,learning_rate=0.01,threshold=0.5):
        self.name=gate_name

        self.__xdata=xdata.reshape(4,2)
        self.__tdata=tdata.reshape(4,1)

        self.__w=np.random.rand(2,1)
        self.__b=np.random.rand(1)

        self.__learning_rate=learning_rate
        self.__threshold=threshold

    def __loss_func(self):
        delta=1e-7

        z=np.dot(self.__xdata, self.__w)+self.__b
        y=sigmoid(z)

        return -np.sum(self.__tdata*np.log(y+delta)+(1-self.__tdata)*np.log((1-y)+delta))
```

*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
21      def err_val(self):
22          delta=1e-7
23
24          z=np.dot(self.__xdata, self.__w)+self.__b
25          y=sigmoid(z)
26
27          return -np.sum(self.__tdata*np.log(y+delta)+(1-self.__tdata)*np.log((1-y)+delta))
28      def train(self):
29
30          f=lambda x : self.__loss_func()
31
32          print("init error : ",self.err_val())
33
34          for stp in range(20000):
35              self.__w -= self.__learning_rate * numerical_derivative(f,self.__w)
36              self.__b -= self.__learning_rate * numerical_derivative(f,self.__b)
37
38              if (stp%2000 == 0):
39                  print("step : ", stp, "error : ",self.err_val())
40
```

*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

```
def predict(self, input_data):

    z=np.dot(input_data,self.__w) + self.__b
    y=sigmoid(z)
    #print(z,y,np.shape(self.__w))

    if y[0]>self.__threshold:
        result = 1
    else:
        result =0
    #print("weighting :", self.__w," b :",self.__b)

    return y, result
```

*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

OR

```
1  xdata=np.array([[0,0],[0,1],[1,0],[1,1]])
2  tdata=np.array([[1,1,1,0]])
3
4  AND_gate= logicGate("AND_GATE",xdata,tdata,)
5  AND_gate.train()
6
7  for in_data in xdata:
8      (sig_val,logic_val)=AND_gate.predict(in_data)
9      print(in_data , " : ", logic_val)
```

```
init error :  3.111693807533499
step :  0 error :  3.103007938531621
step :  2000 error :  0.6905547757007772
step :  4000 error :  0.4009223215621792
step :  6000 error :  0.2802724329082963
step :  8000 error :  0.2144597243739002
step :  10000 error :  0.17323727210991532
step :  12000 error :  0.1450875701742919
step :  14000 error :  0.12468737175336034
step :  16000 error :  0.10924616862852524
step :  18000 error :  0.09716393958711722
[0 0]  :  1
[0 1]  :  1
[1 0]  :  1
[1 1]  :  0
```

*perceptron 실습

***아래 내용을 jupyter notebook에 typing하고 결과를 확인해봅시다**

NAND

```
1  xdata=np.array([[0,0],[0,1],[1,0],[1,1]])
2  tdata=np.array([[1,0,0,0]])
3
4  AND_gate= logicGate("AND_GATE",xdata,tdata,)
5  AND_gate.train()
6
7  for in_data in xdata:
8      (sig_val,logic_val)=AND_gate.predict(in_data)
9      print(in_data , " : ", logic_val)
```

```
init error :  3.920868024440878
step :  0 error :  3.86137591365093
step :  2000 error :  0.44775219185853654
step :  4000 error :  0.23752357339586552
step :  6000 error :  0.1593974560072097
step :  8000 error :  0.11933534712918722
step :  10000 error :  0.09514053602806942
step :  12000 error :  0.07900127306672228
step :  14000 error :  0.0674915301705854 7
step :  16000 error :  0.05887982370436922
step :  18000 error :  0.05219944367370667
[0 0]  :  1
[0 1]  :  0
[1 0]  :  0
[1 1]  :  0
```

THANK YOU