

□□

```
import matplotlib.pyplot as plt
```

□□□

□□ `plt.scatter(x,y)` □□□□□□

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

x = [0.13,0.22,0.39,0.59,0.68,0.74,0.93]
y = [0.75,0.34,0.44,0.52,0.80,0.25,0.65]

plt.figure(figsize = (8,6))
## □□□□□
plt.scatter(x,y,marker = 's',s = 50)

## □□□□□□□□
for x,y in zip(x,y):
    plt.annotate('%s,%s'%(x,y),xy=(x,y),xytext = {0,-15},textcoords = 'offset
points')

plt.show()
```

□□□

□□ `plt.hist()` □□□□□

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

data = np.random.normal(0,20,1000)
bins = np.arange(-100,100,5)

plt.hist(data,bins = bins)
plt.show()
```

□□□

- □□ `plt.bar(x,y)` □□□□□
- □□ `plt.barh(x,y)` □□□□□□□□

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

np.random.seed(0)
x = np.arange(5)
```

```

y = np.random.randn(5)
fig, axes = plt.subplots(ncols = 2)
## 
v_bars = axes[0].bar(x,y,color = 'red',alpha = 0.5)
h_bars = axes[1].barh(x,y,color = 'red',alpha = 0.5)

```

3D

```
from mpl_toolkits.mplot3d import Axes3d
```

3d

```
ax = plt.gca(projection = '3d')
```

```

import numpy as np
import matplotlib.pyplot as plt

t = np.linspace(0,10,100)
x = np.sin(t)
y = np.cos(t)
z = t

## 
ax = plt.gca(projection = '3d')
## 
ax.plot(x,y,z,label = '3D')
ax.set_xlabel('x label')
ax.set_ylabel('y label')
ax.set_zlabel('z label')
ax.legend()

plt.show()

```

API

- `plt.plot(x,y)` : `x` `y`
 - `label` `label`
 - `color`
 - `linestyle`
 - `linewidth`
 - `alpha`
 - `marker`
- `plt.subplot(nrows,ncols,index)` :
 - `nrows`
 - `ncols`
 - `index`
 - `==plt.plot() ==`

- `plt.subplots()` : figure 객체와 axes 객체

- `kwargs`

- `nrows`
- `ncols`
- `sharex(y)`

- `axes` 객체

- `plot()`
- `set_title()`
- `set_xlabel()`
- `set_xlim()`

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

np.random.seed(0)
x = np.arange(5)
y = np.random.randn(5)
fig, axes = plt.subplots(ncols=2)

axes[0].plot(x, y, label='1')
axes[1].plot(y, x, label='2')

plt.show()
```

- `plt.show()` : `plt` 객체

- `plt.figure(figsize=(20,8),dpi=80)` : `fig` 객체, `figsize`

- `fig` 객체

- `fig.add_subplot()` 객체

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

# Figure 객체
fig = plt.figure(figsize=(8, 4), dpi=100)

# Subplot
ax1 = fig.add_subplot(1, 2, 1)
ax2 = fig.add_subplot(1, 2, 2)

# Data
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)

# Plot
ax1.plot(x, y1, label='Sine')
```

```

ax1.set_title('Sine Wave')
ax1.legend()

ax2.plot(x, y2, label='Cosine', color='orange')
ax2.set_title('Cosine Wave')
ax2.legend()

# 显示
plt.show()

```

- `pyplot.savefig("保存路径")` :保存svg格式的图片
- `pyplot.xticks()` :设置x(y)轴的范围

- `xticks(ticks, labels, **kwargs)`

- `ticks`:x轴上的刻度值
- `labels`:与`ticks`对应的label值
- `**kwargs`:其他参数
 - `rotation`:旋转角度

- ```
import matplotlib.pyplot as plt
设置x轴
bar_label = ['bar1', 'bar2', 'bar3']
x_pos = list(range(len(bar_label)))
plt.xticks(x_pos, bar_label)
```

- `font_manager`和`fontproperties`模块
- `pyplot.x(y)label()` :设置x(y)轴label
  - `font_size`:字体大小
- `pyplot.title()` :设置标题
  - 设置位置
- `pyplot.grid(alpha)` :设置网格
  - `alpha`:透明度
- `pyplot.legend()` :设置图例
  - `loc`:图例位置
- `plt.annotate(text, xy, xytext=None, arrowprops=None, **kwargs)`
  - `text`:标注文本
  - `xy`:标注点(x,y)
  - `xytext`:标注文本位置(xy)
  - `arrowprops`:标注箭头属性

- `**kwargs` : `kwargs`를 넘겨줄 수 있음

- `import matplotlib.pyplot as plt`

```
plt.annotate('Local Max', xy=(1.57, 1), xytext=(3, 0.5),
 arrowprops=dict(facecolor='red', shrink=0.05),
 fontsize=12, color='blue')
```

- `plt.fill_between(x, y1, y2=0, **kwargs)` : `y1`과 `y2` 사이의 영역을 채움

- `x` : x 값의 배열

- `y1` : y 값의 배열

- `y2` : y 값의 배열 (기본값은 0) 또는 `y=0`로 지정 가능

- `**kwargs` : `color` (색상), `alpha` (투명도), `label` (라벨) 등

- `plt.xlim()` : x축의 범위를 설정

- `import matplotlib.pyplot as plt`  
`# x축 범위를 1-10로 설정`  
`plt.xlim(1,10)`

- `plt.vlines(x,ymin,ymax,colors,linestyles)` : 수직 선 그리기

- `x` : x 값의 배열

- `ymin` : y의 최소값

- `ymax` : y의 최대값

- `colors` : 색상

- `linestyles` : 선 스타일

- `import matplotlib.pyplot as plt`

```
x = [1,2,3,4,5]
y = [2,3,1,4,7]
plt.plot(x,y)

x=3에서 y=0부터 y=5까지
plt.vlines(x = 3,ymin = 0,ymax = 5,colors = 'r',linestyles='dash')
plt.show()
```

- `zip(x,y)` : `x`와 `y`를 쌍으로 묶어서 반환