차트 옵션

크기 제어, 축 레이블, 제목, 범례, 그리드

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

# 임의의 데이터 생성
x = np.linspace(0, 2 * np.pi, 100)
y1 = np.sin(x)
y2 = np.cos(x)

# Figure와 Axes 객체 생성 (figsize를 통해 크기 지정)
fig, ax = plt.subplots(figsize=(10, 3))

# ylim, xlim을 통해 표기할 범위 지정 가능
plt.ylim(-0.75, 1)     # y축 0~1 값으로 표시
plt.xlim(0,10)         # x축 0~10 값으로 표시

# 그래프 그리기 (label 인자 지정)
ax.plot(x, y1, label='Sine', color='blue')
ax.plot(x, y2, label='Cosine', color='red', linestyle='--')
# Pandas DataFrame의 경우, 자체적으로 plot 메소드를 통해 그릴 수도 있음

# 제목과 축 레이블 설정
ax.set_title('Sine and Cosine Waves', fontsize=16, fontweight='bold')
ax.set_xlabel('Radian', fontsize=12)
ax.set_ylabel('Value', fontsize=12)
# fontsize, fontweight, color, linestyle 등 파라미터를 통해 제어 가능

# 범례 표시 (loc='best'는 최적의 위치에 자동으로 배치)
# .plot에 label 값이 입력되어 있어야 범례 표시가 이루어짐
ax.legend(loc='upper right', fontsize=10)

# 그리드 추가
ax.grid(True)

plt.show()
```

서브플롯 (Subplots)

- `plt.tight_layout()`: 서브플롯들이 겹치지 않도록 자동으로 간격을 조정해주는 유용한 함수

subplot의 속성 순서

```
plt.subplot(row, column, index)
# row : 고정된 행 수
```

```
# column : 고정된 열 수
# index : 순서
```

세로로 그리기

```
# 2행 1열로 그리기 (세로로 2개 나열)
plt.figure(figsize = (8,8))            # 전체 그래프 크기 지정

# 첫번째 그래프 (2행 1열 중 1번째)
plt.subplot(2,1,1)
plt.plot(x, y1, label='Sine', color='blue')
plt.grid()

# 두번째 그래프 (2행 1열 중 2번째)
plt.subplot(2,1,2)
plt.plot(x, y2, label='Cosine', color='red', linestyle='--')

plt.tight_layout()                     # 그래프간 간격을 적절히 맞추기
plt.show()
```

가로로 그리기

```
# 1행 2열로 그리기 (가로 2개 나열)
plt.figure(figsize = (12,8))            # 전체 그래프 크기 지정

# 첫번째 그래프
plt.subplot(1,2,1)
plt.plot(x, y1, label='Sine', color='blue')
plt.grid()

# 두번째 그래프
plt.subplot(1,2,2)
plt.plot(x, y2, label='Cosine', color='red', linestyle='--')

plt.tight_layout()                     # 그래프간 간격을 적절히 맞추기
plt.show()
```

인덱스로 접근하기

- `Figure` 객체와 `nrows` x `ncols` 크기의 `Axes` 객체 배열을 반환하는 것을 이용하여, 각 `Axes` 객체에 인덱스로 접근해서 작성할 수도 있다.

```
# 2x2 형태의 서브플롯 생성
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(10, 8))

# 전체 제목 설정
```

```
fig.suptitle('Various Plots in Subplots', fontsize=16)

# 첫 번째 서브플롯 (0, 0)
axes[0, 0].plot(x, y1, color='green')
axes[0, 0].set_title('Sine Wave')

# 두 번째 서브플롯 (0, 1)
axes[0, 1].plot(x, y2, color='orange')
axes[0, 1].set_title('Cosine Wave')

# 세 번째 서브플롯 (1, 0) - 히스토그램
data = np.random.randn(1000)
axes[1, 0].hist(data, bins=30, color='purple')
axes[1, 0].set_title('Histogram')

# 네 번째 서브플롯 (1, 1) - 산점도
x_scatter = np.random.rand(50)
y_scatter = np.random.rand(50)
axes[1, 1].scatter(x_scatter, y_scatter, color='magenta')
axes[1, 1].set_title('Scatter Plot')

# 레이아웃 자동 조정
plt.tight_layout(rect=[0, 0, 1, 0.96]) # suptitle과 겹치지 않도록 조정

plt.show()
```

여러 그래프 겹치기

- 동일한 축을 공유하는 여러 종류의 그래프를 함께 그려 풍부한 정보를 전달
- **주의사항**
  - 두 번째 그래프를 그릴 때, 축을 공유해야 함
  - `twinx()`: y축 스케일이 다를 경우, 보조 y축 생성을 위해 사용하는 함수

```
import seaborn as sns
tips = sns.load_dataset('tips')

# 1. 동일한 Axes에 겹쳐 그리기 (KDE + Rug plot)
fig, ax = plt.subplots()
# seaborn 0.9.0에서는 data 직접 전달로 구현하려면 x, y 값 전부 지정되어야 오류가 발생하지 않음 → Series로 전달하는게 안전
# sns.kdeplot(data=tips, x="total_bill", ax=ax, color='blue', label='KDE')
# sns.rugplot(data=tips, x="total_bill", ax=ax, color='black', label='Rug')
sns.kdeplot(tips["total_bill"], ax=ax, color='blue', label='KDE')  # 밀도 함수 작성
sns.rugplot(tips["total_bill"], ax=ax, color='black')              # 분포를 파악하기 위한 러그 플롯
ax.legend()
ax.set_title('KDE and Rug Plot of Total Bill')
plt.show()

# 2. 보조 Y축 사용하기 (twinx)
fig, ax1 = plt.subplots(figsize=(8, 5))
```

```
# 첫 번째 Y축 (ax1) - 바 차트
sales_data = {'month': ['Jan', 'Feb', 'Mar', 'Apr'], 'sales': [100, 120, 150,
130]}
ax1.bar(sales_data['month'], sales_data['sales'], color='skyblue', label='Sales')
ax1.set_xlabel('Month')
ax1.set_ylabel('Sales (units)', color='skyblue')
ax1.tick_params(axis='y', labelcolor='skyblue')

# 두 번째 Y축 (ax2) - 라인 플롯
ax2 = ax1.twinx()
growth_data = {'growth': [0.05, 0.2, 0.1, -0.05]}
ax2.plot(sales_data['month'], growth_data['growth'], color='red', marker='o',
label='Growth Rate')
ax2.set_ylabel('Growth Rate', color='red')
ax2.tick_params(axis='y', labelcolor='red')
ax2.axhline(0, color='gray', linestyle='--') # 성장률 0 기준선

fig.suptitle('Sales and Growth Rate')
fig.tight_layout()
plt.show()
```

- **결과 해석**:
  - 첫 번째 예시는 커널 밀도 추정(KDE) 그래프 아래에 각 데이터의 위치를 나타내는 러그 플롯(rug plot)을 겹쳐 그려 분포를 더 상세하게 보여줍니다.
  - 두 번째 예시는 월별 판매량(바 차트)과 전월 대비 성장률(라인 플롯)을 하나의 그래프에 함께 보여줍니다. 두 데이터의 단위와 스케일이 다르기 때문에, `twinx()`를 사용하여 왼쪽과 오른쪽에 각각의 y축을 만들어 표현했습니다.

수평선, 수직선, 텍스트 추가

```
plt.axhline(40, color = 'grey', linestyle = '--') # 수평선 y = 40 에서 그림
plt.axvline(10, color = 'red', linestyle = '--')  # 수직선 x = 10 에서 그림
plt.text(5, 41, '40')                             # x, y 위치 = (5, 41) 위치에 텍스
트("40") 띄움
```