

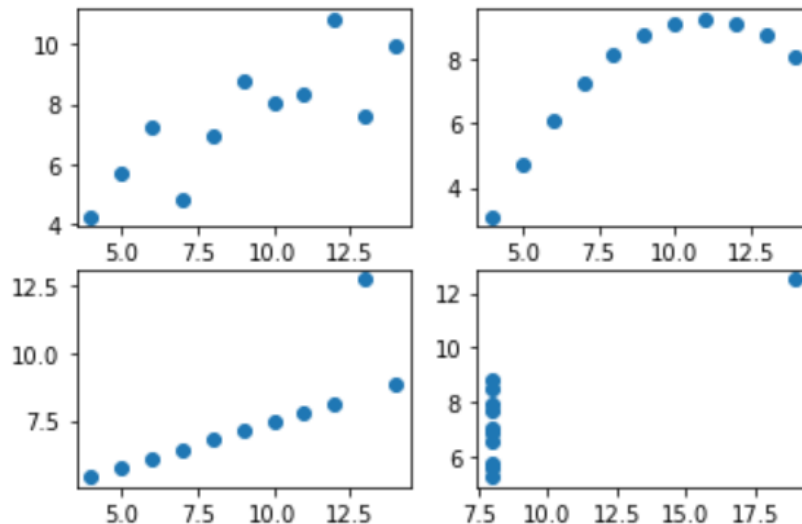
그래프 그리기

- 데이터 시각화가 필요한 이유
- matplotlib 라이브러리
- seaborn 라이브러리 자유자재로 사용하기
- 데이터프레임과 시리즈로 그래프 그리기

■ 데이터 시각화가 필요한 이유

● 앤스콤 4분할 그래프

- 데이터 시각화의 전형적인 사례
- 시각화 하지 않고 수치만 확인할 때 발생할 수 있는 문제점을 보여줌
(by 프랭크 앤스콤)
- 4개의 데이터 그룹으로 구성되며, 평균 / 분산 / 상관관계 / 회귀선이 같음
➔ 데이터 그룹 I, II, III, IV 의 데이터는 모두 같을 것이라는 착각
시각화 하면 서로 다른 데이터 패턴을 가지고 있다는 점을 알 수 있음



■ 데이터 시각화가 필요한 이유

● 앤스콤 4분할 그래프

- conda install seaborn

```
(base) C:\Users\WGGoReb>conda install seaborn
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
## Package Plan ##
```

```
environment location: C:\Users\WGGoReb\miniconda3
```

```
added / updated specs:
- seaborn
```

The following packages will be downloaded:

package	build	
icc_rt-2019.0.0	h0cc432a_1	6.0 MB
scipy-1.5.2	py38h14eb087_0	11.9 MB
seaborn-0.11.1	pyhd3eb1b0_0	212 KB
Total:		18.1 MB

■ 데이터 시각화가 필요한 이유

- 앤스콤 데이터 집합 불러온 후 그래프 그리기
 - 앤스콤 데이터 집합 불러오기

```
import seaborn as sns
anscombe = sns.load_dataset("anscombe")
print(anscombe)
print(type(anscombe))
```

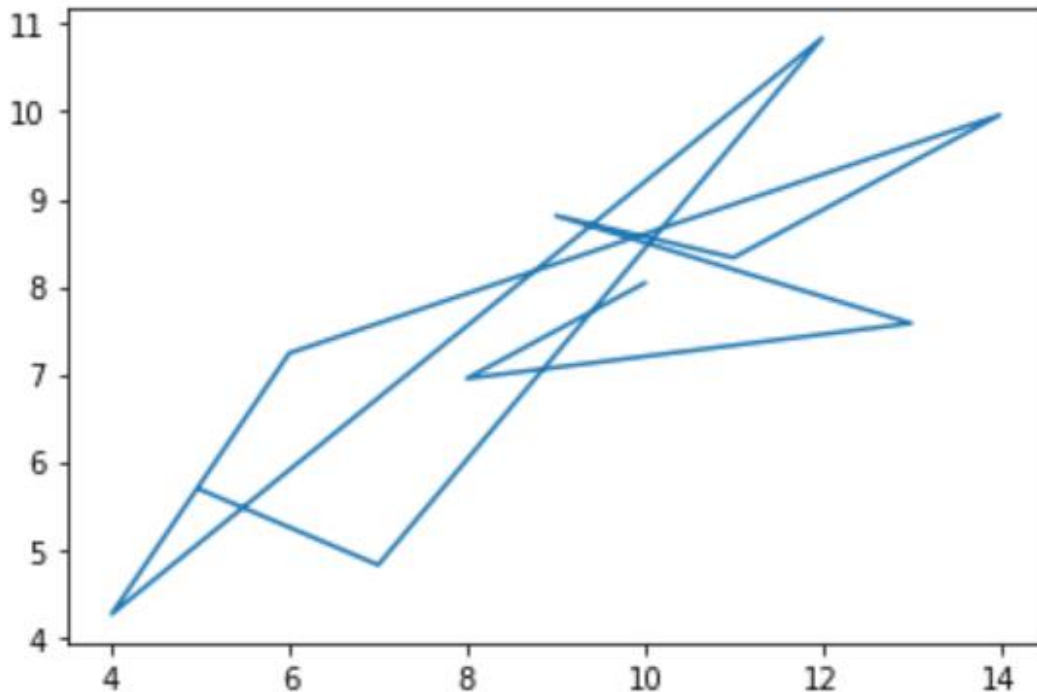
```
   dataset    x     y
0         I  10.0  8.04
1         I   8.0  6.95
2         I  13.0  7.58
3         I   9.0  8.81
4         I  11.0  8.33
<class 'pandas.core.frame.DataFrame'>
```

■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기

- matplotlib 라이브러리로 그래프 그리기 (1번 데이터)

```
import matplotlib.pyplot as plt
dataset_1 = anscombe[anscombe['dataset'] == 'I']
plt.plot(dataset_1['x'], dataset_1['y'])
plt.show()
```



■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기

- matplotlib 라이브러리로 그래프 그리기 (모든 데이터 - I ~ IV)

```
dataset_1 = anscombe[anscombe['dataset'] == 'I']  
dataset_2 = anscombe[anscombe['dataset'] == 'II']  
dataset_3 = anscombe[anscombe['dataset'] == 'III']  
dataset_4 = anscombe[anscombe['dataset'] == 'IV']  
  
fig = plt.figure() # 그래프 기본 틀 생성  
axes1 = fig.add_subplot(2, 2, 1) # row, col, index  
axes2 = fig.add_subplot(2, 2, 2)  
axes3 = fig.add_subplot(2, 2, 3)  
axes4 = fig.add_subplot(2, 2, 4)
```

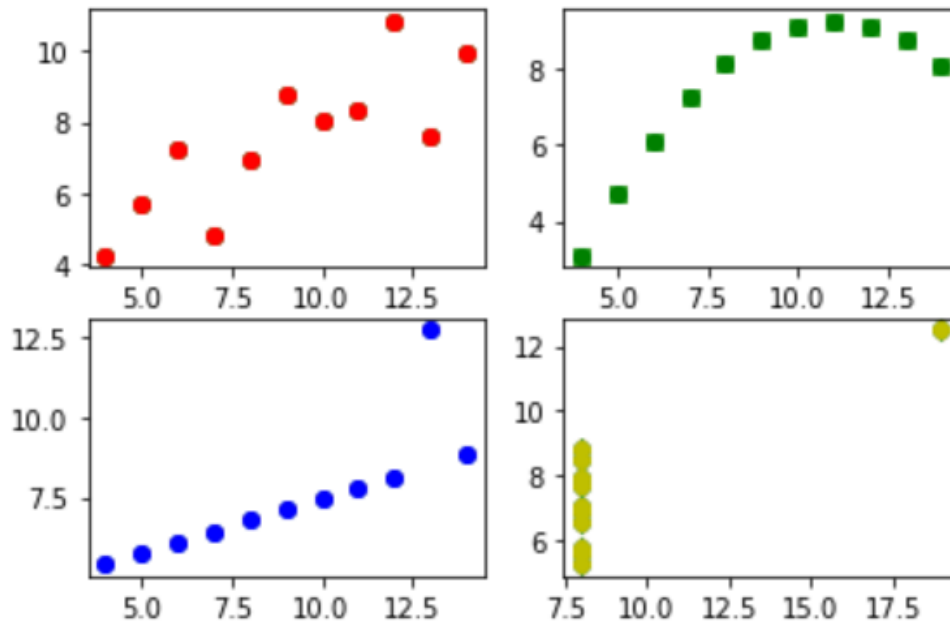
■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기

- matplotlib 라이브러리로 그래프 그리기 (모든 데이터 - I ~ IV)

```
axes1.plot(dataset_1['x'], dataset_1['y'], 'ro') # red circle  
axes2.plot(dataset_2['x'], dataset_2['y'], 'go') # green circle  
axes3.plot(dataset_3['x'], dataset_3['y'], 'bo') # blue circle  
axes4.plot(dataset_4['x'], dataset_4['y'], 'yo') # yellow circle
```









fig



■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기














- 그래프 색상

약자	의미	표현
b	blue	
c	cyan	
g	green	
k	black	
m	magenta	
r	red	
w	white	
y	yellow	

■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기





- 그래프 마커

약자	의미	표현	약자	의미	표현
.	point		D	diamond	
,	pixel		d	thin diamond	
o	circle		*	star	
s	square		+	plus	
p	pentagon		x	x	
1	tri_down		v	triangle down	
2	tri_up		^	triangle up	
3	tri_left		<	triangle left	
4	tri_right		>	triangle left	
h	hexagon1		H	hexagon2	

■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기

- 그래프 선 스타일

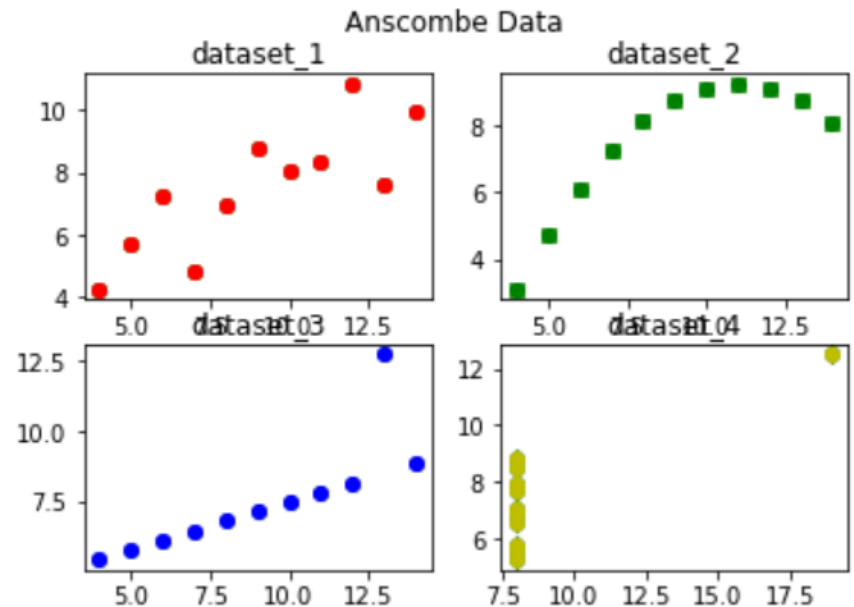
약자	의미	표현
-	solid	
--	dashed	
-.	dash-dot	
:	dotted	

■ 데이터 시각화가 필요한 이유

● 앤스콤 데이터 집합 불러온 후 그래프 그리기

- sub 그래프 및 figure 제목 지정

```
axes1.set_title("dataset_1")  
axes2.set_title("dataset_2")  
axes3.set_title("dataset_3")  
axes4.set_title("dataset_4")  
  
fig.suptitle("Anscombe Data")  
  
fig
```

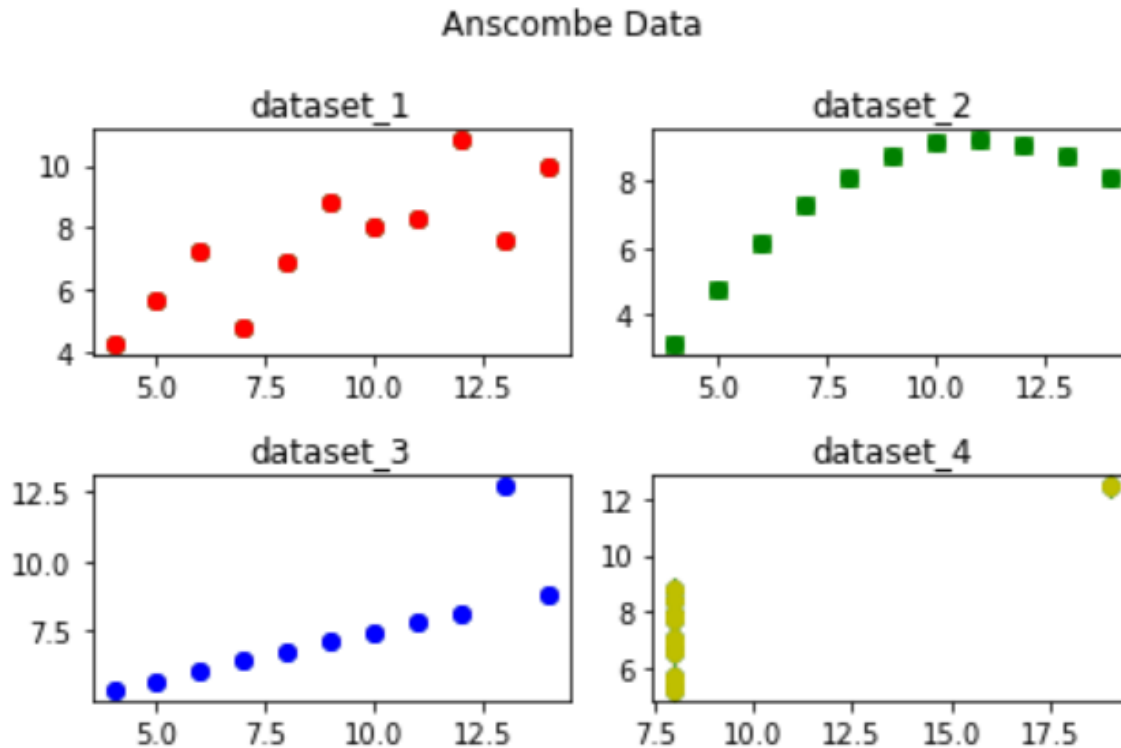


■ 데이터 시각화가 필요한 이유

- 앤스콤 데이터 집합 불러온 후 그래프 그리기
 - 레이아웃 최적화

```
fig.tight_layout()
```

```
fig
```



■ matplotlib 라이브러리

● 기초 그래프 그리기

- seaborn 라이브러리의 tips dataset 사용

➔ 식당에서 팁을 지불한 손님 정보 데이터

```
tips = sns.load_dataset("tips")  
print(tips.head())  
print(type(tips))
```

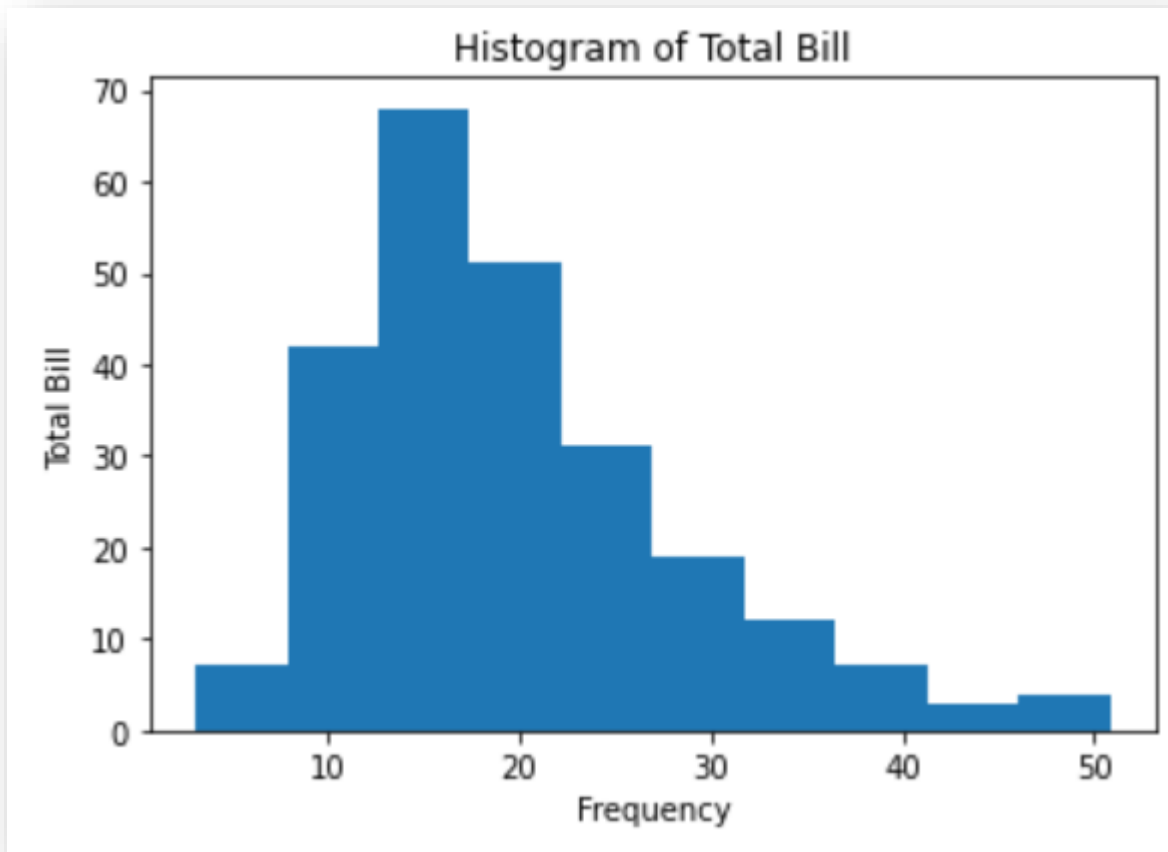
```
   total_bill  tip  sex smoker  day  time  size  
0      16.99  1.01 Female    No  Sun  Dinner     2  
1      10.34  1.66  Male    No  Sun  Dinner     3  
2      21.01  3.50  Male    No  Sun  Dinner     3  
3      23.68  3.31  Male    No  Sun  Dinner     2  
4      24.59  3.61 Female    No  Sun  Dinner     4  
<class 'pandas.core.frame.DataFrame'>
```

■ matplotlib 라이브러리

● 기초 그래프 그리기 - 히스토그램

- 데이터의 분포와 빈도를 살펴보는 용도로 자주 사용
- 'total_bill', 'tip' 등의 열 : 변수

➔ 변수를 하나만 사용해서 그린 그래프 : 일변량 그래프

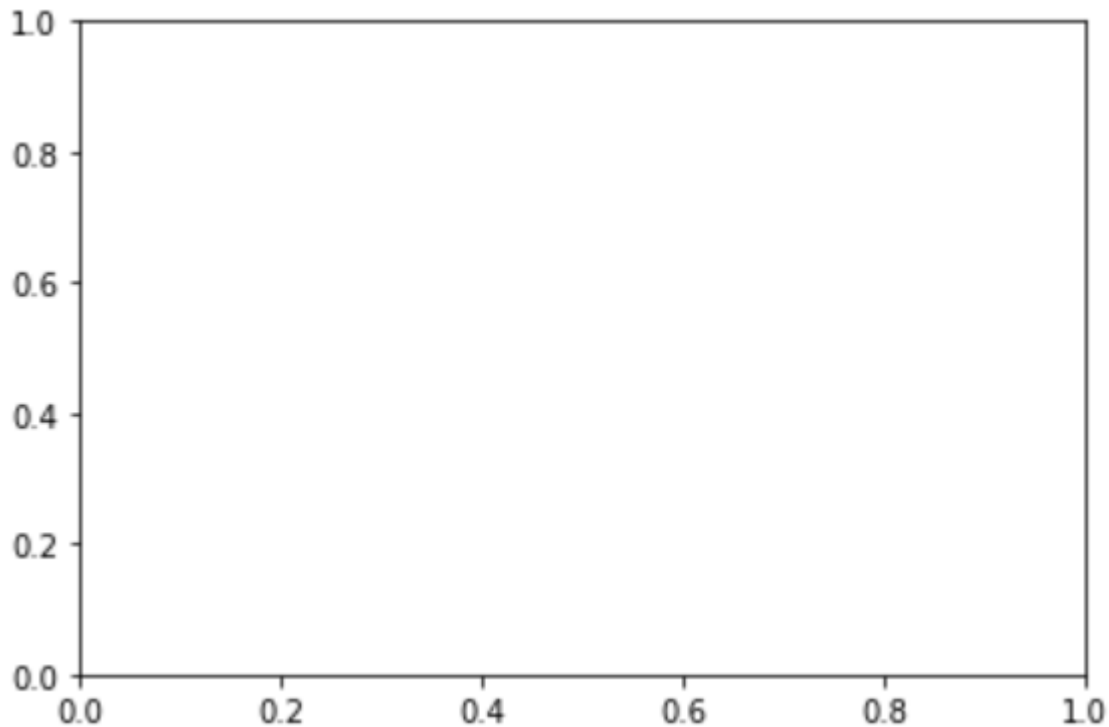


■ matplotlib 라이브러리

● 기초 그래프 그리기 - 히스토그램

- 기본 틀(figure)을 준비하고 그래프 격자 구성

```
fig = plt.figure()  
axes1 = fig.add_subplot(1, 1, 1)
```



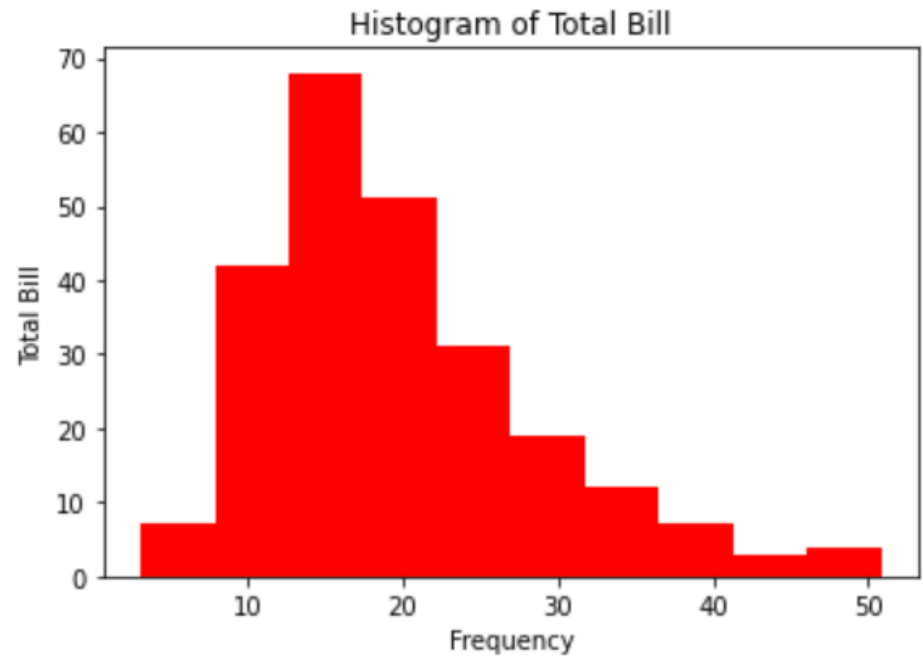
■ matplotlib 라이브러리

● 기초 그래프 그리기 - 히스토그램

- hist 메소드에 데이터 전달
- bins 옵션으로 x축 간격 조정

```
axes1.hist(tips['total_bill'], bins=10, color='red')  
axes1.set_title('Histogram of Total Bill')  
axes1.set_xlabel('Frequency')  
axes1.set_ylabel('Total Bill')
```

fig

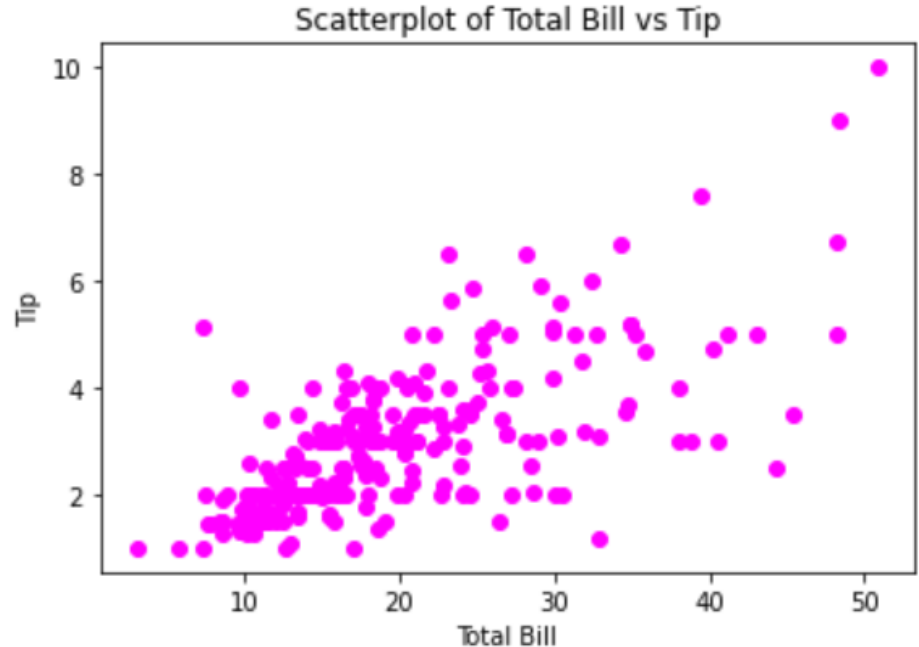


■ matplotlib 라이브러리

● 기초 그래프 그리기 - 산점도 그래프

- 변수 2개를 사용해서 만드는 그래프로 '이변량 그래프' 라고 부름
- 'total_bill' 데이터에 따른 'tip' 데이터 분포 표현

```
scatter_plot = plt.figure()
axes1 = scatter_plot.add_subplot(1, 1, 1)
axes1.scatter(tips['total_bill'], tips['tip'], c='magenta')
axes1.set_title('Scatterplot of Total Bill vs Tip')
axes1.set_xlabel('Total Bill')
axes1.set_ylabel('Tip')
```



■ matplotlib 라이브러리

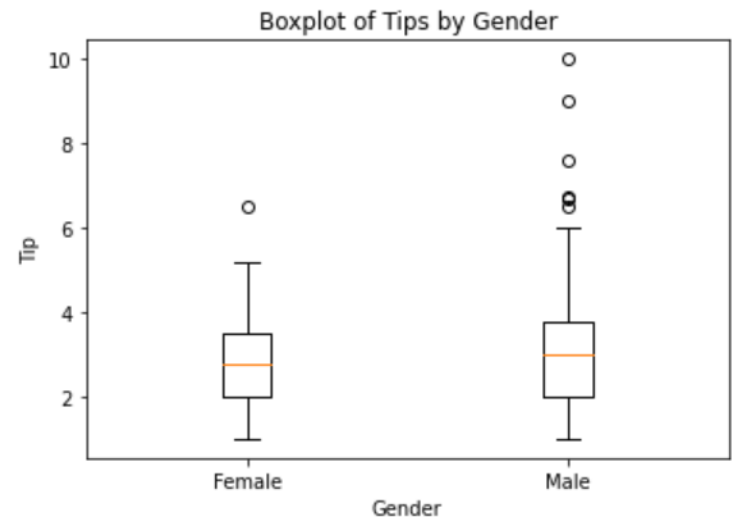
● 기초 그래프 그리기 – 박스 그래프

- 이산형 변수와 연속형 변수를 함께 사용하는 그래프
- 이산형 변수 : 성별, 지역, 학교 등 명확하게 구분되는 범주 값
- 연속형 변수 : 비율, 가격, 나이 등 명확하게 구분하지 못하는 범위 값

```
boxplot = plt.figure()
axes1 = boxplot.add_subplot(1, 1, 1)

axes1.boxplot(
    [tips[tips['sex'] == 'Female']['tip'],
     tips[tips['sex'] == 'Male']['tip']],
    labels=['Female', 'Male'])

axes1.set_xlabel('Gender')
axes1.set_ylabel('Tip')
axes1.set_title('Boxplot of Tips by Gender')
```



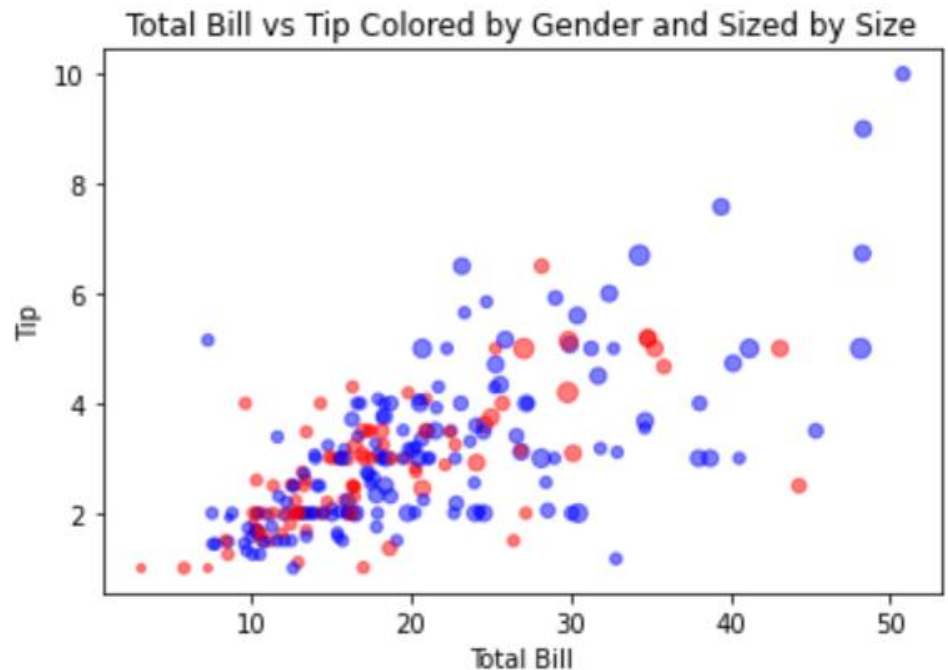
■ matplotlib 라이브러리

● 다변량 그래프 그리기

- 3개 이상의 변수를 사용한 그래프

ex) 'total_bill' 데이터에 따른 'tip' 데이터 분포 표현 + '성별' + '인원'

- x축 : 'total_bill'
- y축 : 'tip'
- 마커의 크기(s) : '인원'
- 마커의 색상(c) : '성별'



■ matplotlib 라이브러리

● 다변량 그래프 그리기

- 성별에 따라 마커의 색상을 변경하기 위해 함수 작성

```
def recode_gender(gender):  
    if gender == 'Female':  
        return 'red'  
    else:  
        return 'blue'
```

- apply 함수를 사용하여 데이터를 변경한 후 'color' 열에 추가

```
tips['color'] = tips['sex'].apply(recode_gender)  
tips.head()
```

	total_bill	tip	sex	smoker	day	time	size	color
0	16.99	1.01	Female	No	Sun	Dinner	2	red
1	10.34	1.66	Male	No	Sun	Dinner	3	blue
2	21.01	3.50	Male	No	Sun	Dinner	3	blue
3	23.68	3.31	Male	No	Sun	Dinner	2	blue
4	24.59	3.61	Female	No	Sun	Dinner	4	red

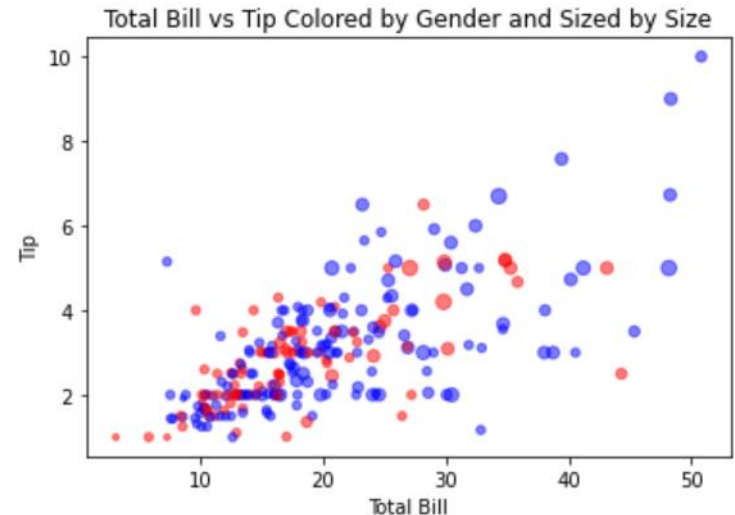
■ matplotlib 라이브러리

● 다변량 그래프 그리기

- 산점도 그래프로 표현

```
scatter_plot = plt.figure()
axes1 = scatter_plot.add_subplot(1, 1, 1)
axes1.scatter(
    x=tips['total_bill'],
    y=tips['tip'],
    s=tips['size'] * 10,
    c=tips['color'],
    alpha=0.5)

axes1.set_title('Total Bill vs Tip Colored by Gender and Sized by Size')
axes1.set_xlabel('Total Bill')
axes1.set_ylabel('Tip')
```



■ Seaborn 라이브러리 자유자재로 사용하기

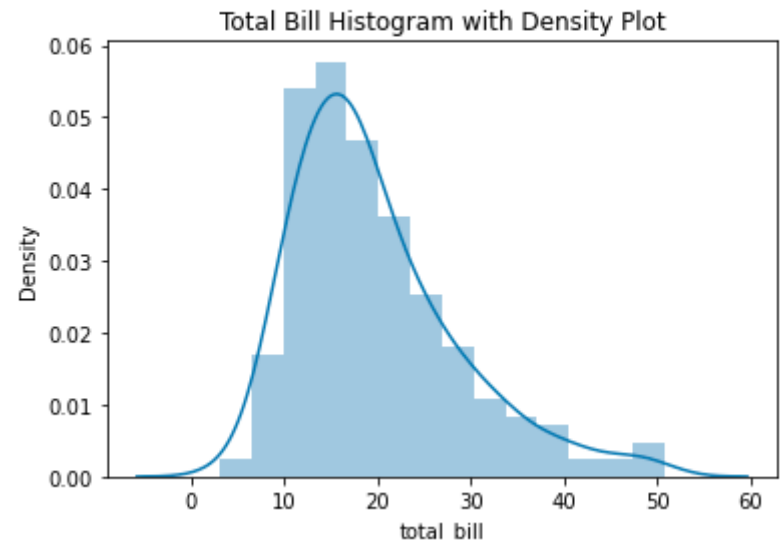
● 단변량 그래프 그리기

- 히스토그램으로 표현

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

ax = plt.subplots()
ax = sns.distplot(tips['total_bill'])
ax.set_title('Total Bill Histogram with Density Plot')
```

Text(0.5, 1.0, 'Total Bill Histogram with Density Plot')



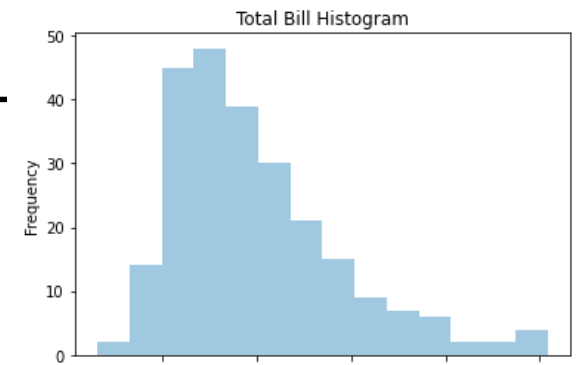
■ Seaborn 라이브러리 자유자재로 사용하기

● 단변량 그래프 그리기

- 히스토그램으로 표현

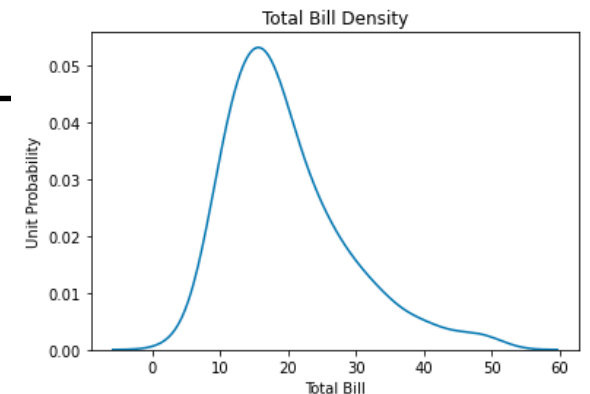
```
ax = plt.subplots()
ax = sns.distplot(tips['total_bill'], kde = False)
ax.set_title('Total Bill Histogram')
ax.set_xlabel('Total Bill')
ax.set_ylabel('Frequency')
```

Text(0, 0.5, 'Frequency')



```
ax = plt.subplots()
ax = sns.distplot(tips['total_bill'], hist = False)
ax.set_title('Total Bill Density')
ax.set_xlabel('Total Bill')
ax.set_ylabel('Unit Probability')
```

Text(0, 0.5, 'Unit Probability')



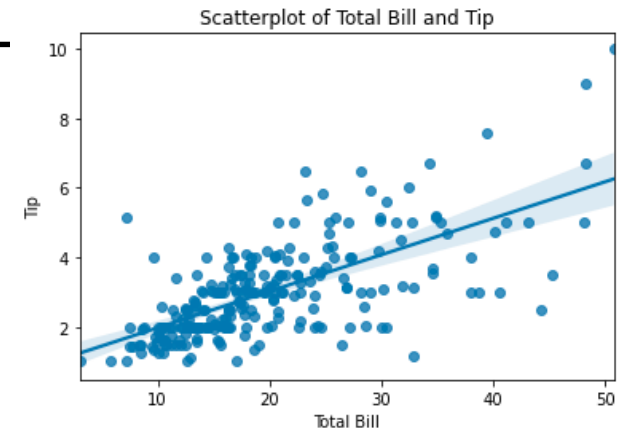
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 산점도로 표현

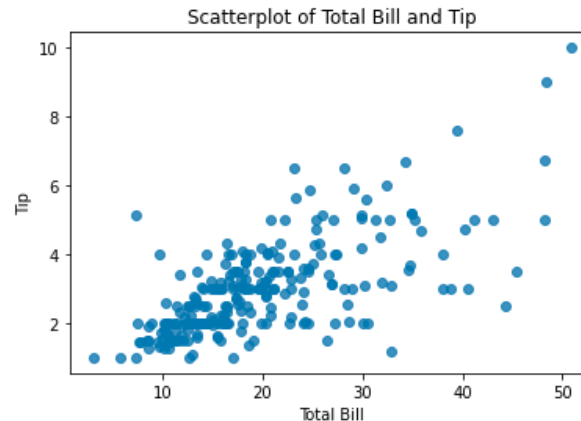
```
ax = plt.subplots()
ax = sns.regplot(x='total_bill', y='tip', data=tips)
ax.set_title('Scatterplot of Total Bill and Tip')
ax.set_xlabel('Total Bill')
ax.set_ylabel('Tip')
```

Text(0, 0.5, 'Tip')



```
ax = sns.regplot(x='total_bill', y='tip',
data=tips, fit_reg=False)
```

Text(0, 0.5, 'Tip')



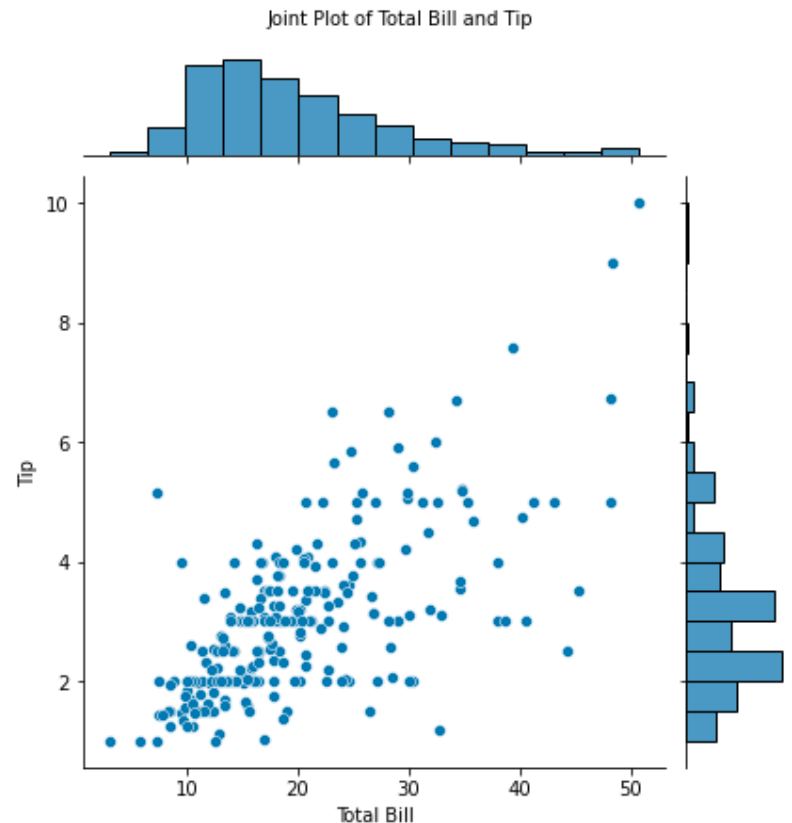
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 산점도와 히스토그램 한번에 표현

```
joint = sns.jointplot(x='total_bill', y='tip', data=tips)
joint.set_axis_labels(xlabel='Total Bill', ylabel='Tip')
joint.fig.suptitle('Joint Plot of Total Bill and Tip', fontsize=10,  
y=1.03)
```

Text(0.5, 1.03, 'Joint Plot of Total Bill and Tip')



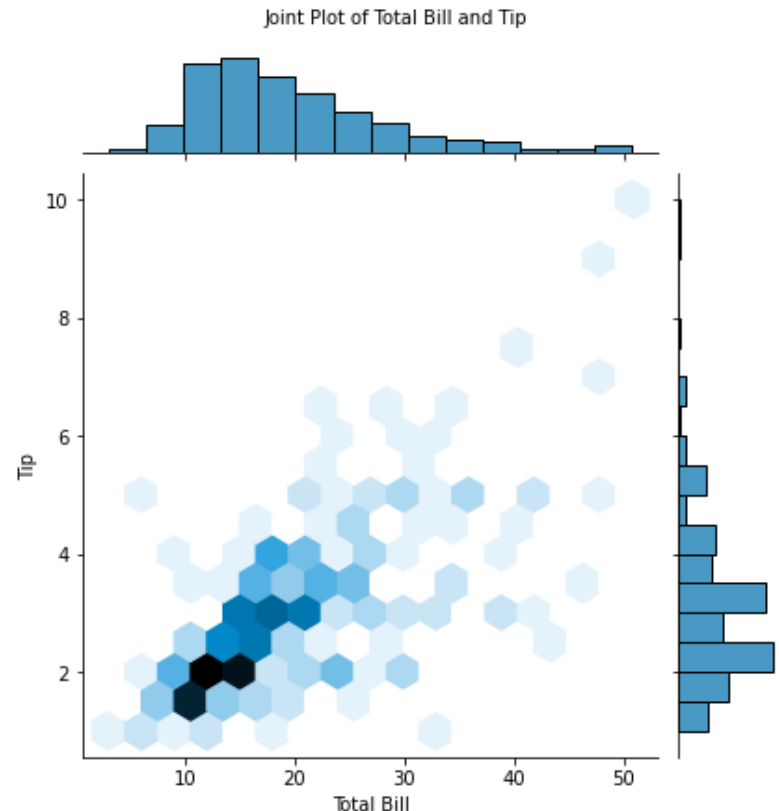
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 산점도와 히스토그램 한번에 표현

```
joint = sns.jointplot(x='total_bill', y='tip', data=tips, kind='hex')
joint.set_axis_labels(xlabel='Total Bill', ylabel='Tip')
joint.fig.suptitle('Joint Plot of Total Bill and Tip', fontsize=10,  
                  y=1.03)
```

Text(0.5, 1.03, 'Joint Plot of Total Bill and Tip')



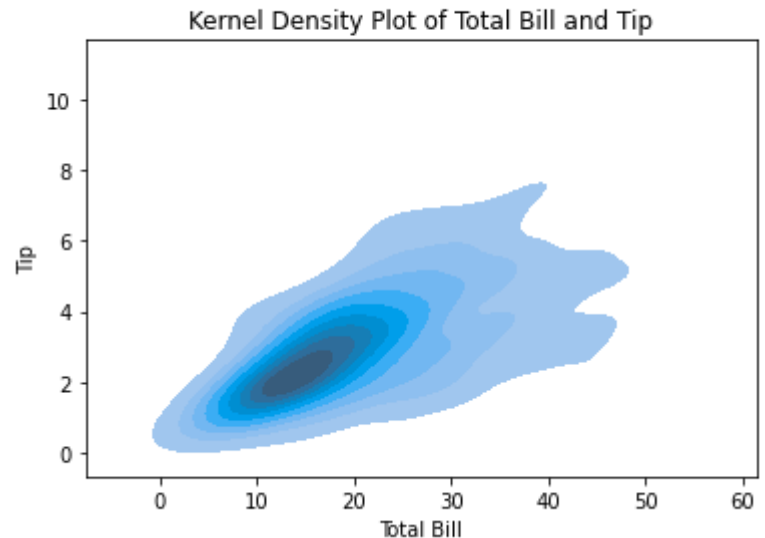
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 이차원 밀집도 표현

```
ax = plt.subplots()
ax = sns.kdeplot(x=tips['total_bill'], y=tips['tip'], shade=True)
ax.set_title('Kernel Density Plot of Total Bill and Tip')
ax.set_xlabel('Total Bill')
ax.set_ylabel('Tip')
```

Text(0, 0.5, 'Tip')



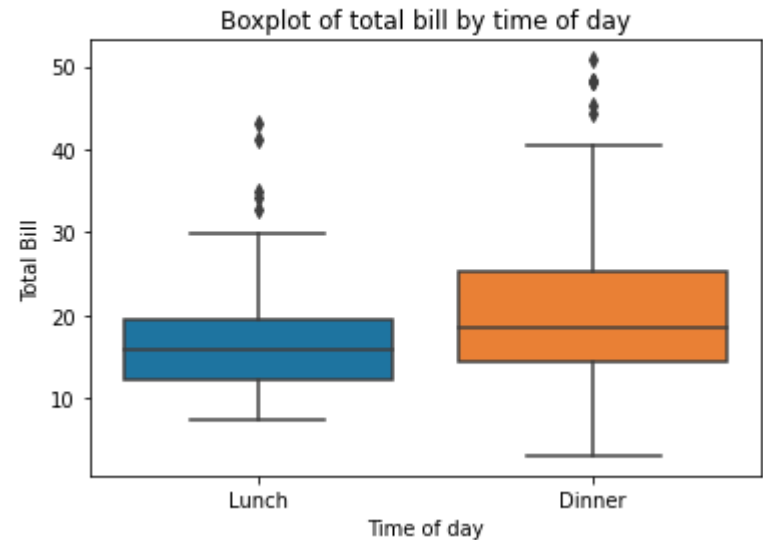
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 박스 그래프 표현

```
ax = plt.subplots()
ax = sns.boxplot(x='time', y='total_bill', data=tips)
ax.set_title('Boxplot of total bill by time of day')
ax.set_xlabel('Time of day')
ax.set_ylabel('Total Bill')
```

Text(0, 0.5, 'Total Bill')



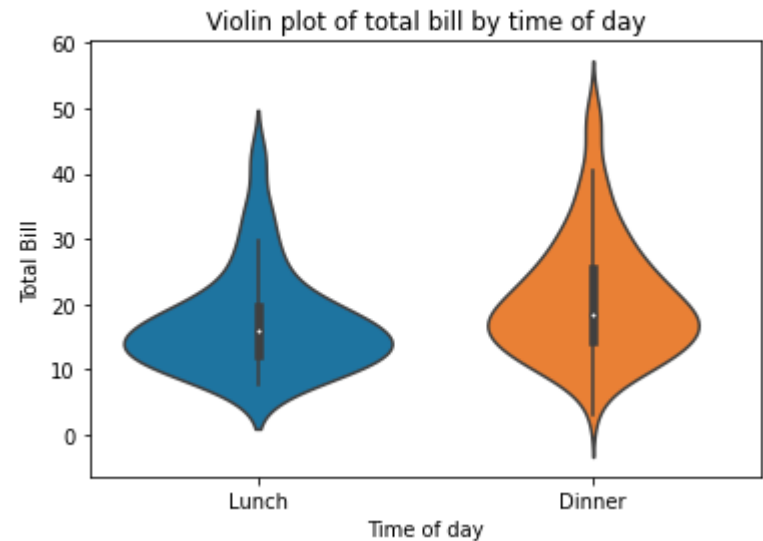
■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 박스 그래프 변형 표현

```
ax = plt.subplots()
ax = sns.violinplot(x='time', y='total_bill', data=tips)
ax.set_title('Violin plot of total bill by time of day')
ax.set_xlabel('Time of day')
ax.set_ylabel('Total Bill')
```

Text(0, 0.5, 'Total Bill')

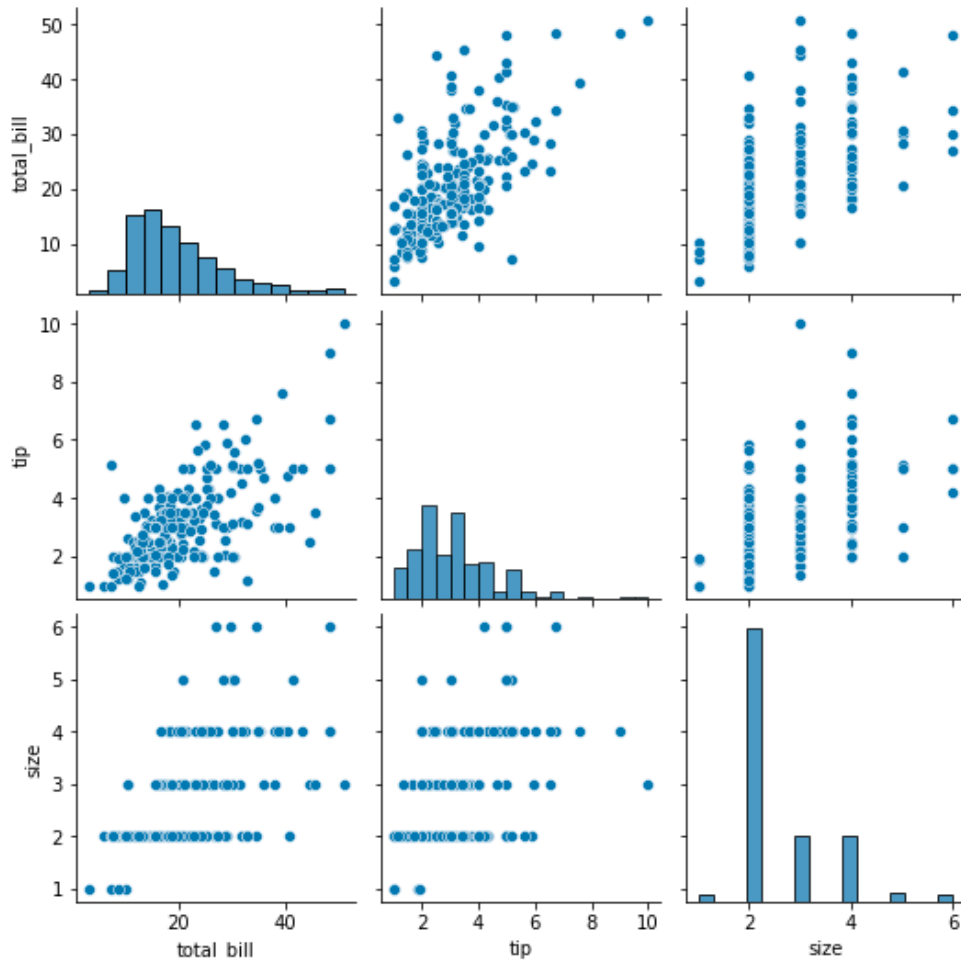


■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 관계 그래프 표현

```
fig = sns.pairplot(tips)
```



데이터에 존재하는 데이터
간의 관계 그래프를 한번에
표현

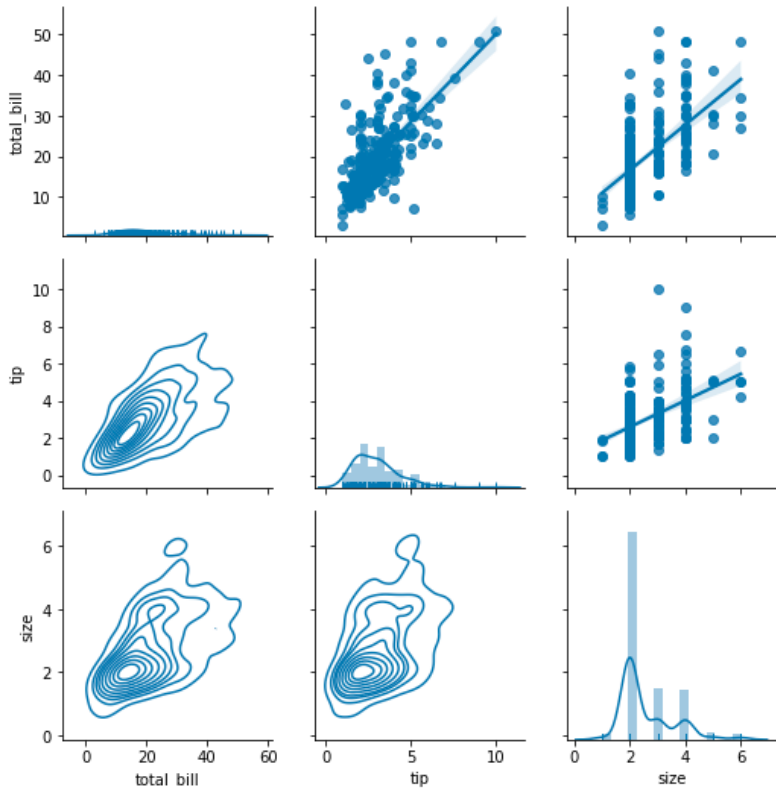
가운데 대각선을 기준으로
중복된 정보가 표현되는 단
점이 있음

■ Seaborn 라이브러리 자유자재로 사용하기

● 이변량 그래프 그리기

- 관계 그래프 표현

```
pair_grid = sns.PairGrid(tips)
pair_grid = pair_grid.map_upper(sns.regplot)
pair_grid = pair_grid.map_lower(sns.kdeplot)
pair_grid = pair_grid.map_diag(sns.distplot, rug=True)
plt.show()
```

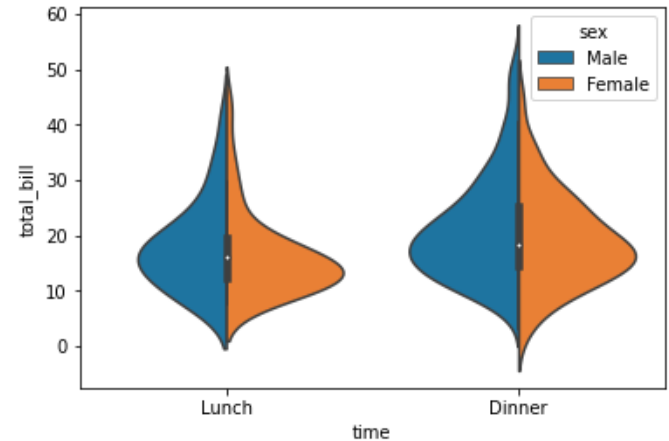


■ Seaborn 라이브러리 자유자재로 사용하기

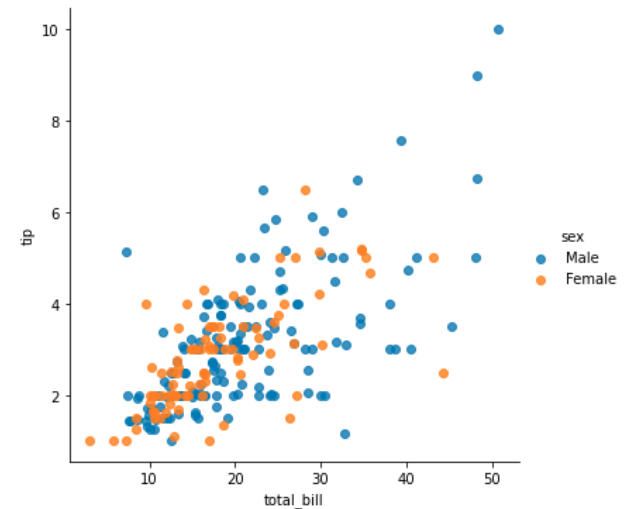
● 다변량 그래프 그리기

- 색상 추가로 표현

```
ax = plt.subplots()
ax = sns.violinplot(x='time',
y='total_bill', hue='sex', data=tips,
split=True)
```



```
scatter = sns.lmplot(x='total_bill',
y='tip', data=tips, hue='sex',
fit_reg=False)
```

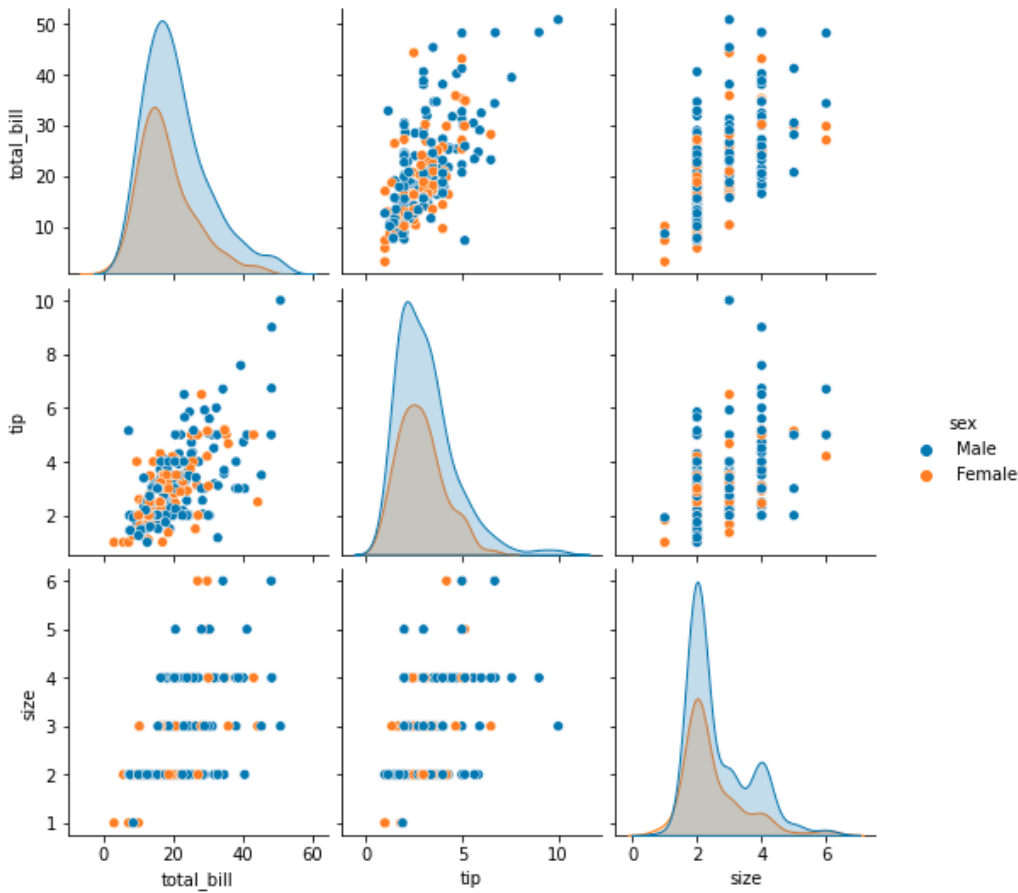


■ Seaborn 라이브러리 자유자재로 사용하기

● 다변량 그래프 그리기

- 색상 추가로 표현

```
fig = sns.pairplot(tips, hue='sex')
```



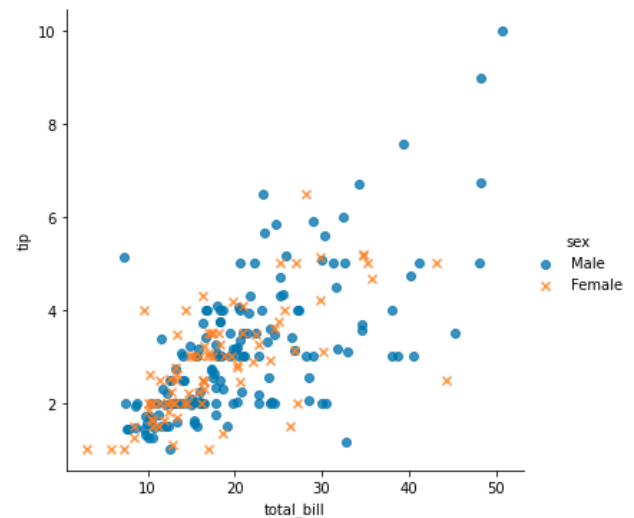
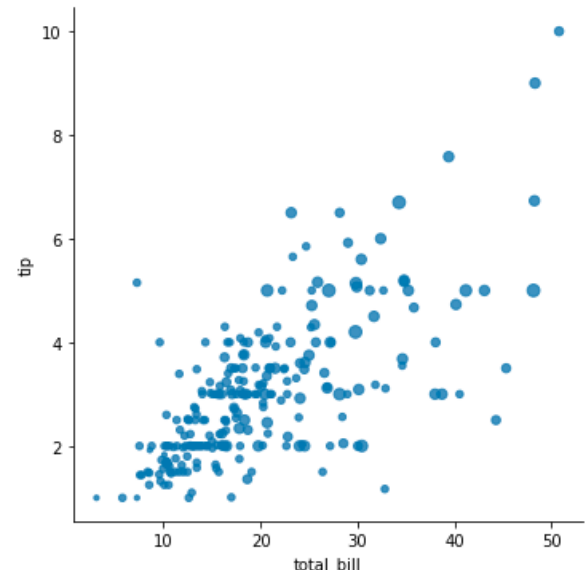
■ Seaborn 라이브러리 자유자재로 사용하기

● 다변량 그래프 그리기

- 크기와 모양으로 표현

```
sns.lmplot( x='total_bill', y='tip',  
data=tips, fit_reg=False,  
scatter_kws={'s': tips['size']*10})
```

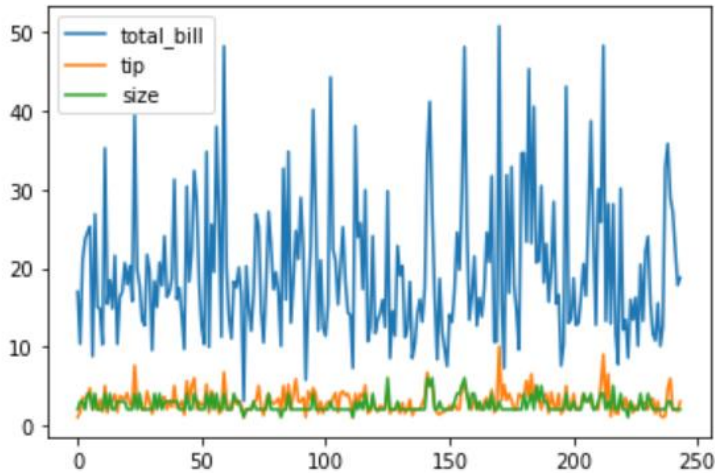
```
sns.lmplot( x='total_bill', y='tip',  
data=tips, fit_reg=False, hue='sex',  
markers=['o', 'x'])
```



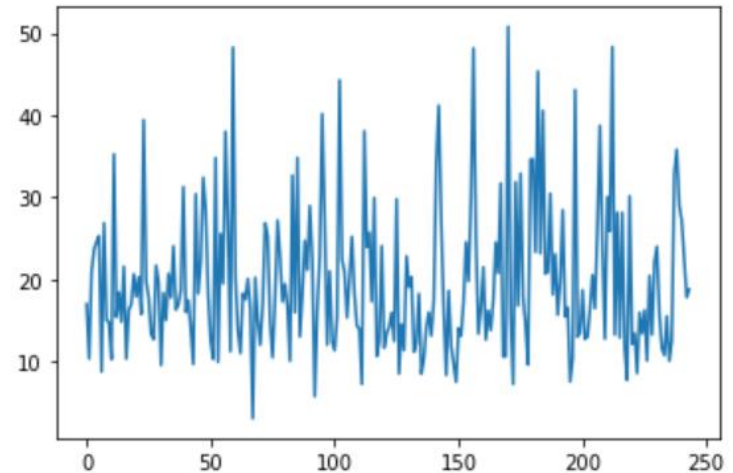
■ 데이터프레임과 시리즈로 그래프 그리기

- 데이터프레임 또는 시리즈의 plot 메소드 사용

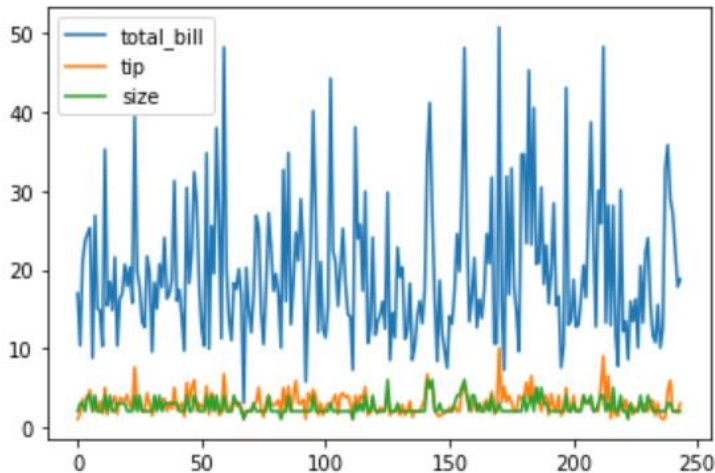
```
tips.plot()
```



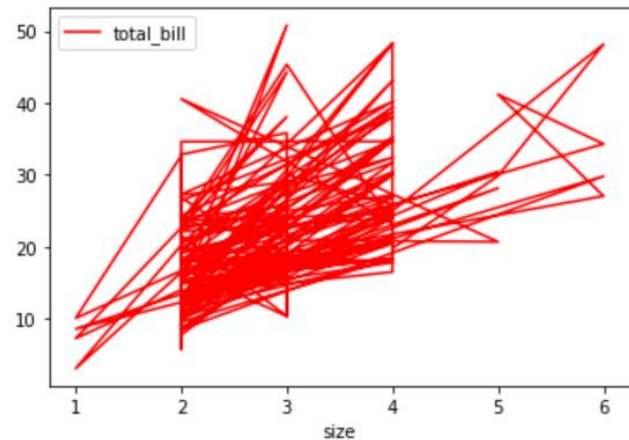
```
tips['total_bill'].plot()
```



```
tips[['total_bill', 'tip']].plot()
```



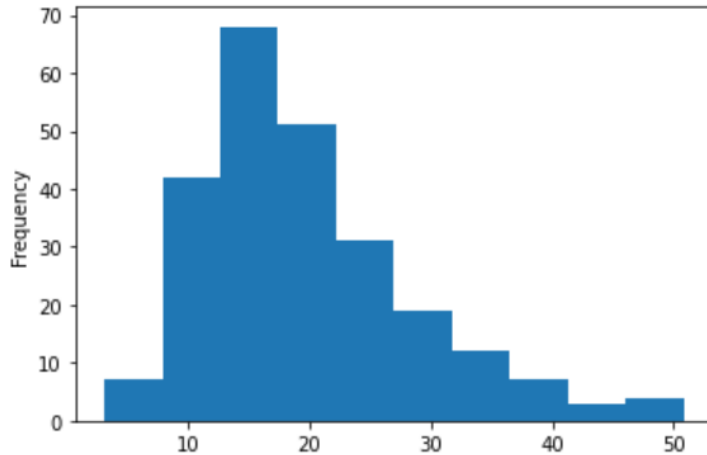
```
tips[['size', 'total_bill']].plot(  
    x='size', y='total_bill', c='red')
```



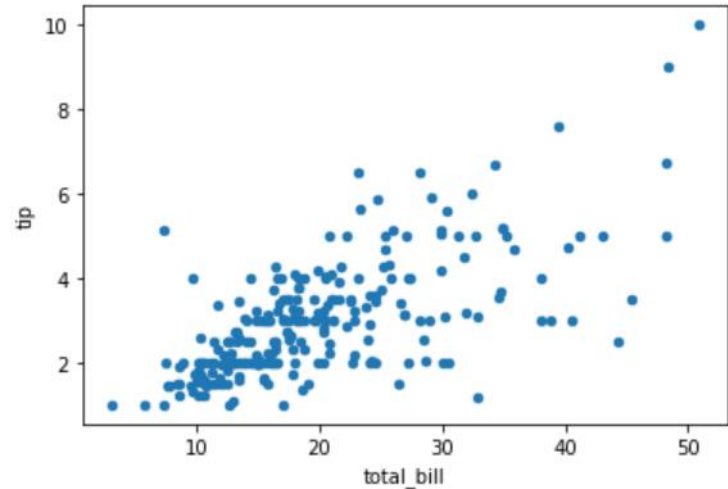
■ 데이터프레임과 시리즈로 그래프 그리기

- 데이터프레임 또는 시리즈의 plot 속성에 정의된 여러 메소드 사용

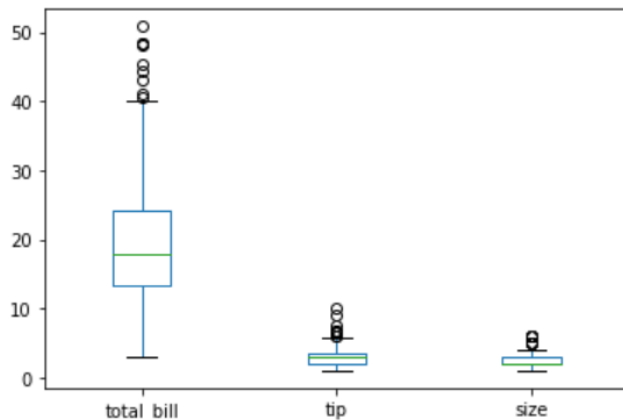
```
tips['total_bill'].plot.hist()
```



```
tips.plot.scatter(x='total_bill', y='tip')
```



```
tips.plot.box()
```



```
tips.plot.hexbin(x='total_bill', y='tip', gridsize=10)
```

