파이썬을 이용한 금융데이터 분석

데이터 기초분석 실습

2019.11.12

데이터 기초분석 실습

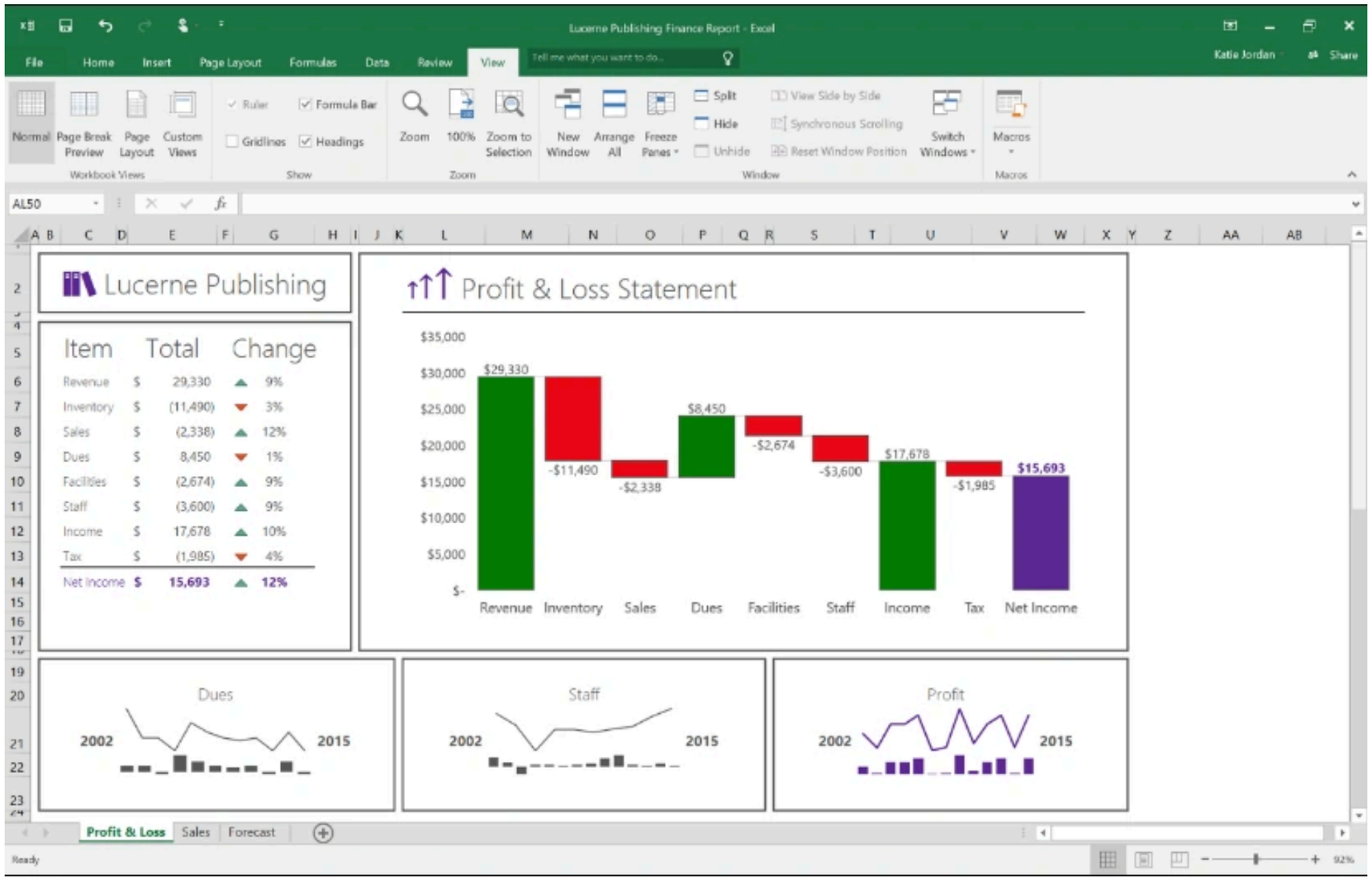
강의 목표

- 이틀이라는 짧은 기간동안 프로그래밍을 제대로 배우는 것은 **불가능**
- 그 대신에,
  - 파이썬을 이용한 데이터 분석 과정을 이해
  - 내가 해보고 싶은 데이터 분석 주제 혹은 질문
  - 파이썬 프로그래밍을 배워보고 싶다는 호기심

데이터 기초분석 실습

강의 목표

- 엑셀로는 부족한가요?
  - 간단한 통계 분석부터 그래프 그리기까지 엑셀도 좋은 분석 툴



데이터 기초분석 실습

강의 목표

- 엑셀로는 부족한가요? **네, 종종 부족합니다.**
  - 이유 1: 백만개 가량의 데이터만 처리 가능
  - 이유 2: 낮은 자유도

1,048,576 rows

Maximum **number of rows** & columns in **Excel**

By default, **Excel** supports three Worksheets in a Workbook file, and each Worksheet **can** support up to 1,048,576 **rows** and 16,384 columns of data. 2019. 3. 11.

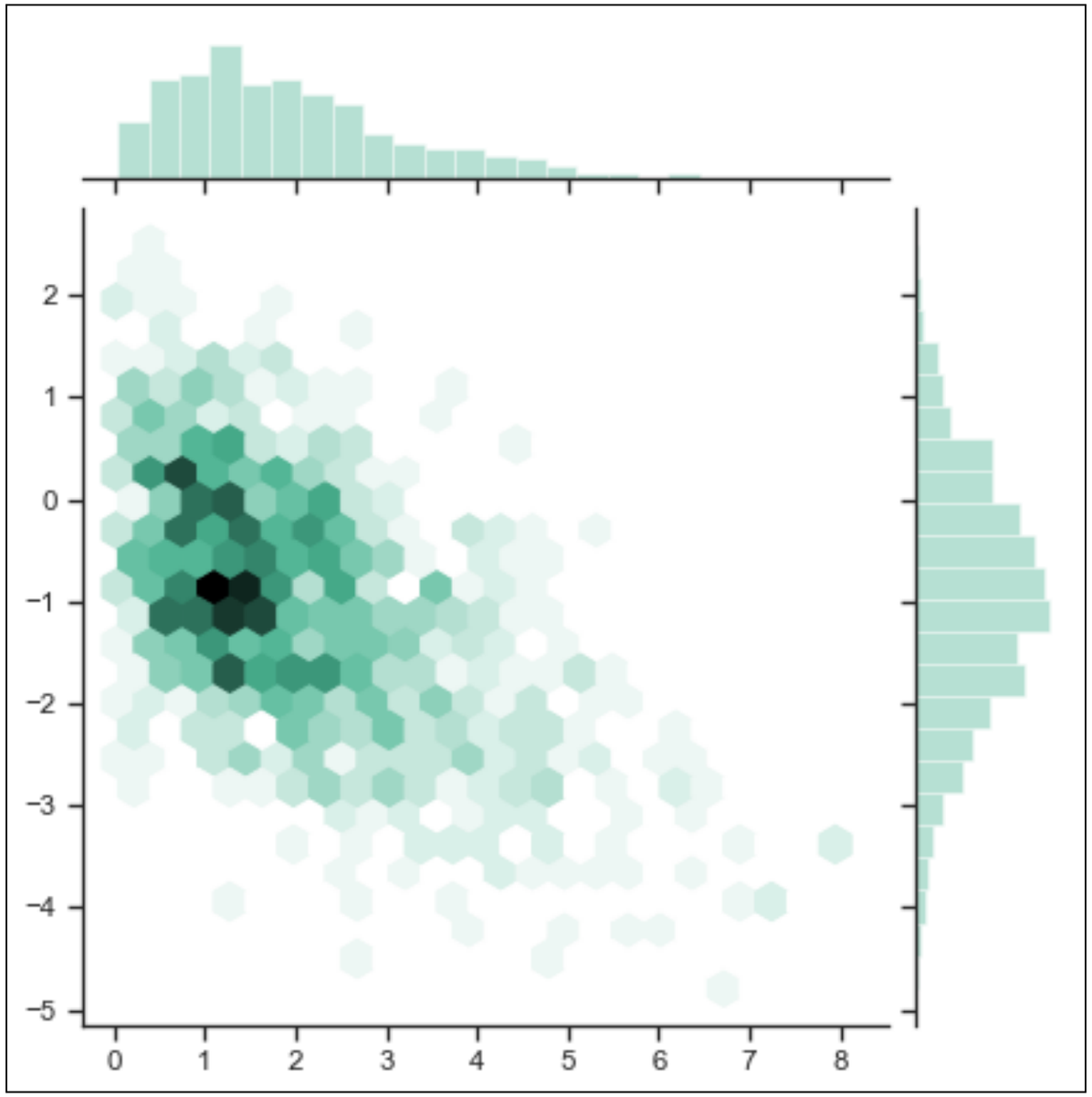
What is the maximum number of columns & rows in Excel ...

https://www.thewindowsclub.com › what-is-the-maximum-number-of-colum...

데이터 기초분석 실습

강의 목표

- 엑셀에서 이런 그래프를 그릴 수 있을까요?



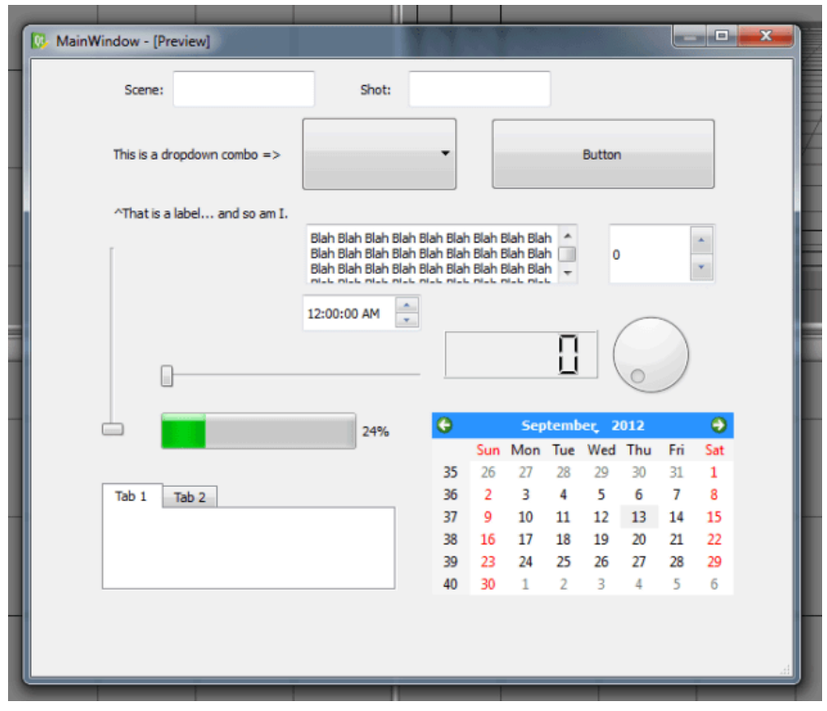
파이썬 기초

주피터 노트북 활용

- 프로그램 코드의 일반적인 작동

<코드 작성>

실행



<결과 확인>

종료 후 수정

파이썬 기초

주피터 노트북 활용

- 주피터 노트북에서의 코드 작동

In [7]:
```
df.groupby("dataset")["x"].describe()
```

<코드 작성>

Out[7]:

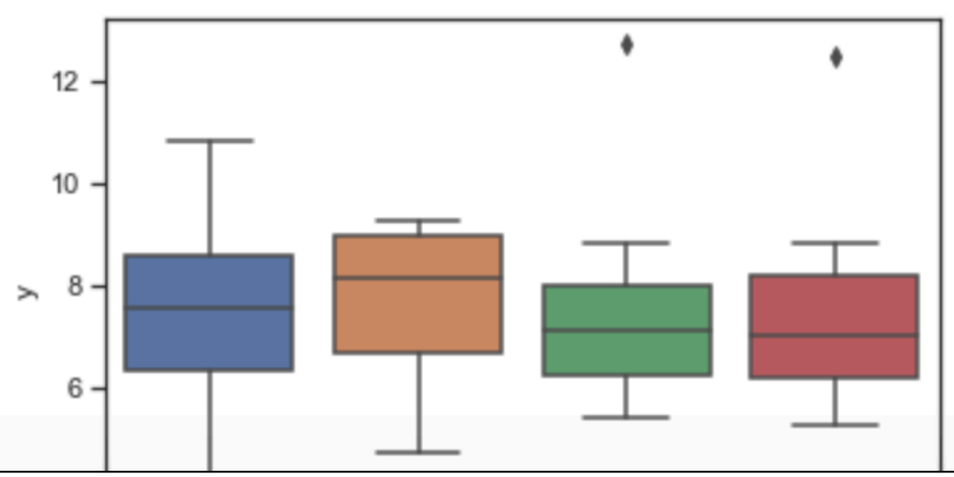
dataset	count	mean	std	min	25%	50%	75%	max
I	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0
II	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0
III	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0
IV	11.0	9.0	3.316625	8.0	8.0	8.0	8.0	19.0

<결과 확인>

In [10]:
```
sns.boxplot(x="dataset", y="y", data=df)
```

<코드 작성>

Out[10]: `<matplotlib.axes._subplots.AxesSubplot at 0x1a1cc06eb8>`

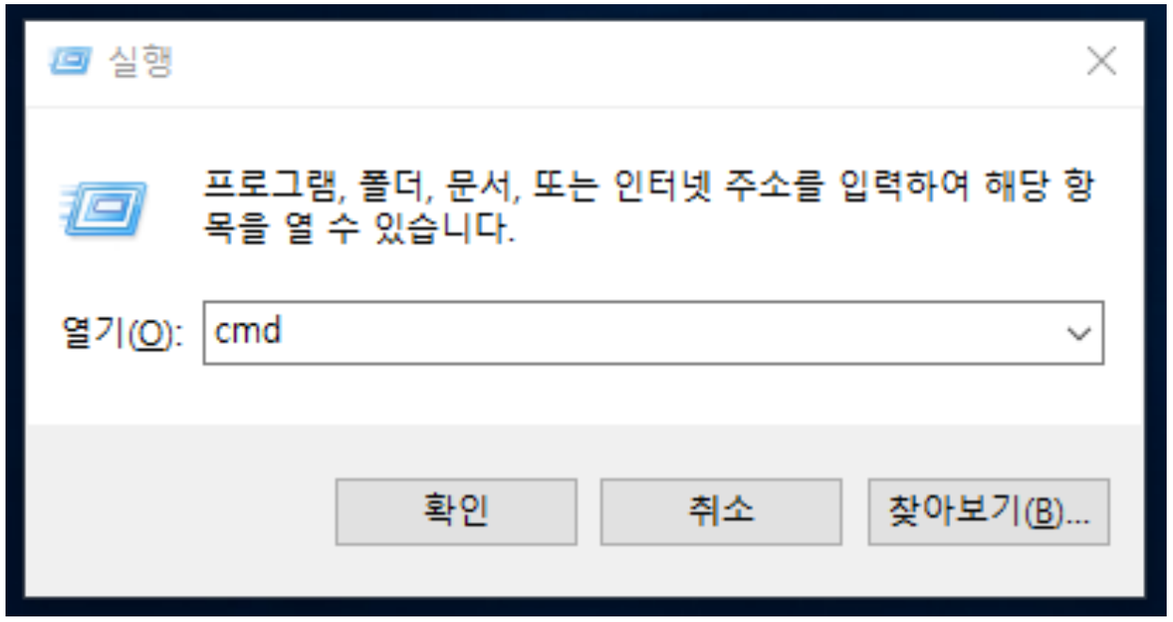


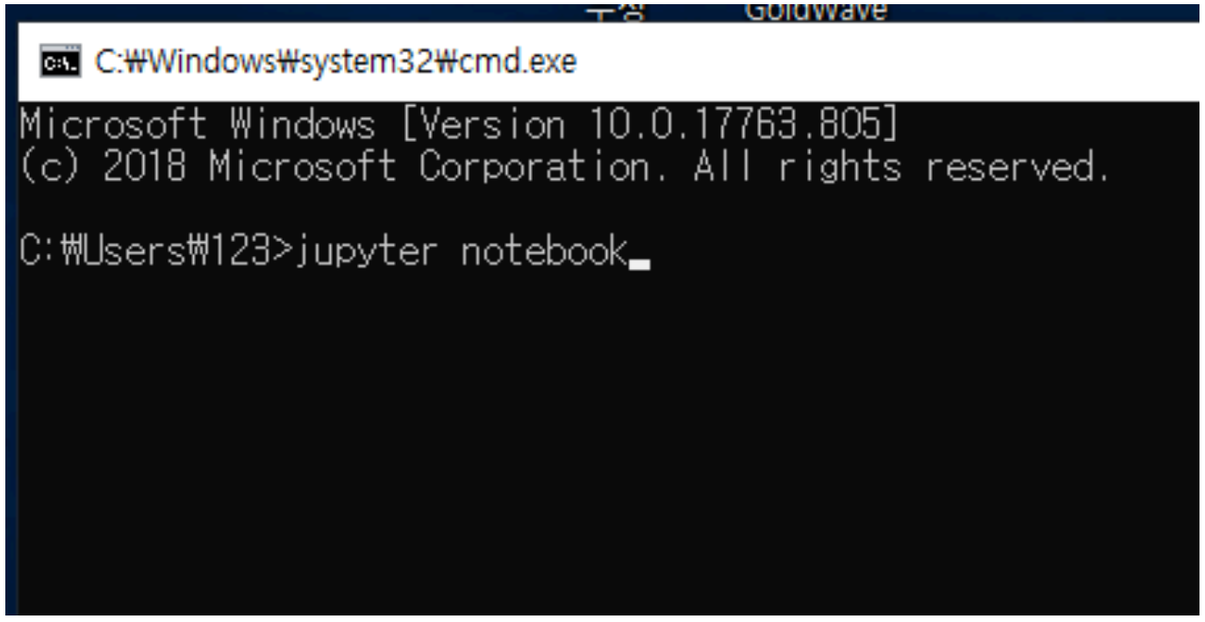
<결과 확인>

파이썬 기초

주피터 노트북 활용

- 주피터 노트북 실행하기
  - [윈도우 + R] 을 눌러 실행 창을 연 후 cmd를 입력 후 엔터
  - 명령 프롬프트에서 jupyter notebook을 입력 후 엔터

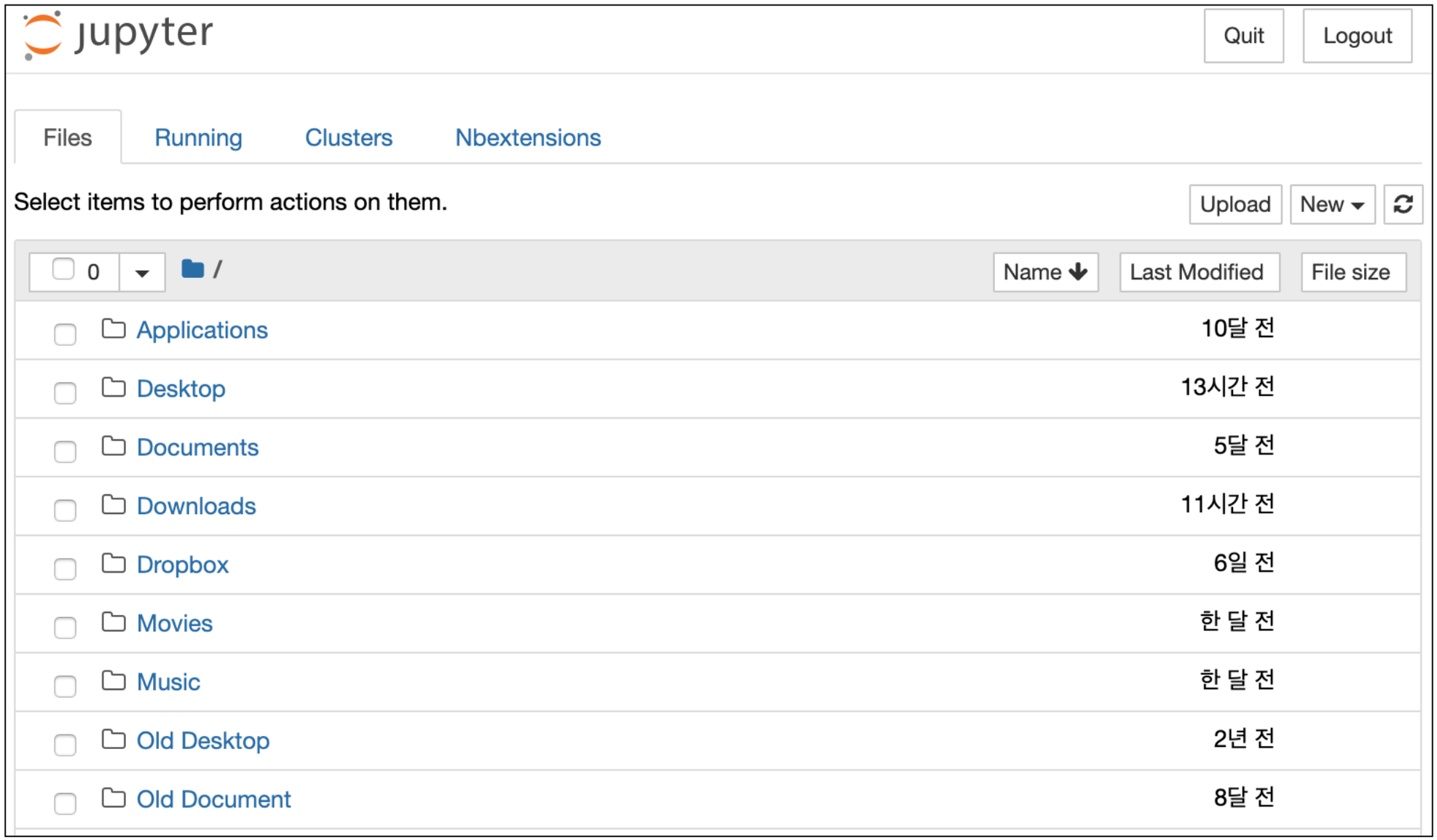




파이썬 기초

주피터 노트북 활용

- 주피터 노트북 실행하기
  - 웹 브라우저가 열린 후 아래 그림과 같은 페이지가 나타나면 성공



1. 파이썬 문법 살펴보기

1.0. 화면에 출력하기

- 코드

```
print ("안녕하세요. 오늘은 12월 1일 입니다.")
```

1. 파이썬 문법 살펴보기

1.1. 변수에 값 저장하기

- 코드

```
pi = 3.14

print (pi)
```

```
msg = “foo”

print (msg)
```

1. 파이썬 문법 살펴보기

1.2. 들여쓰기

- 코드

```
print ("이 문장은 한 번 출력됩니다.")
for i in range(5):
    print ("이 문장은 다섯 번 출력됩니다")
print ("이 문장은 어떨까요?")
```

2. 파이썬 자료형

2.1. 기본 자료형

- 코드

```
number = 1234

string = "Bank" # 따옴표에 주목
```

2. 파이썬 자료형

2.1.1. 기본 자료형의 연산 (숫자형)

- 코드

```
a = 5

b = 2

print (a + b)

print (a - b)

print (a * b)

print (a / b)
```

2. 파이썬 자료형

2.1.2. 기본 자료형의 연산 (문자형)

- 코드

```
a = "365일 함께하면 행복합니다."

b = "신용사회의 금융파트너 여신금융협회"


print (a + b)

print (a[0:4])
```

2. 파이썬 자료형

2.2. 복합 자료형

- 코드

```
aList = [120, 30, 40, 70] # 리스트

aSet = {0, 5, 4, 3, 2} # 집합

aDict = {"수학":100, "영어":90} # 딕셔너리


print (aList)

print (aSet)

print (aDict)
```

3. 데이터 기초분석

3.1. Pandas

- 코드

```
import pandas as pd


url = 'http://samplecsvs.s3.amazonaws.com/SalesJan2009.csv'
# 데이터가 저장되어있는 주소
# 2009년 1월 1개월간의 판매 데이터
data = pd.read_csv(url)
# 해당 주소의 파일(csv 형식)을 열어 변수 data에 저장
data.head() # data의 첫 5줄을 화면에 출력
```

3. 데이터 기초분석

3.1.1. 데이터 열어보기

- 코드

```
data.head(3)
```

```
data.tail(3)
```

3. 데이터 기초분석

3.1.2. 데이터 선택하기

- 코드

```
data[“Name”]
```

```
data.loc[0]
```

```
data.loc[data["Name"] == "Kim", ["Transaction_date",
"Product", "Price", "Country"]]
```

```
data.loc[data["Price"] > 5000, ["Transaction_date",
"Product", "Price", "Name"]]
```

3. 데이터 기초분석

3.1.3. 컬럼 삭제하기

- 코드

```
data.drop(["City", "State", "Account_Created",
"Last_Login"], axis=1, inplace=True)
data.head(3)
```

3. 데이터 기초분석

3.1.4. 자료형 확인하기

- 코드

```
data.dtypes
```

3. 데이터 기초분석

3.1.5. 기초 통계량 확인하기

- 코드

```
data.describe()
```

```
data.describe(include="all")
```

```
data["Latitude"].median()
```

```
data["Price"].unique()
```

```
data["Price"].value_counts()
```

3. 데이터 기초분석

3.1.6. 데이터 값 치환하기

- 코드

```
data.replace({"Price": {"13,000":"13000"}})
```

```
data["Price"].unique()
```

```
data = data.replace({"Price": {"13,000":"13000"}})
data["Price"].unique()
```

```
data["Product"].value_counts()
```

```
data["Product"].unique()
```

```
data = data.replace({"Product": {"Product3 ":"Product3"}})
data["Product"].unique()
```

3. 데이터 기초분석

3.1.7. 데이터 자료형 바꾸기

- 코드

```
data["Price"] = pd.to_numeric(data["Price"])

data.dtypes
```

3. 데이터 기초분석

3.1.8. 데이터 그룹화

- 코드

```
data.groupby("Product")["Price"].describe()
```

기초 통계량의 함정

Anscombe’s Quartet

- Anscombe는 4종류 데이터를 기초 통계량의 함정을 보여줬다

```
import seaborn as sns


df = sns.load_dataset("anscombe")
df.head()
```

	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

기초 통계량의 함정

Anscombe’s Quartet

- Anscombe는 4종류 데이터를 기초 통계량의 함정을 보여줬다

```
df.groupby("dataset").describe()
```

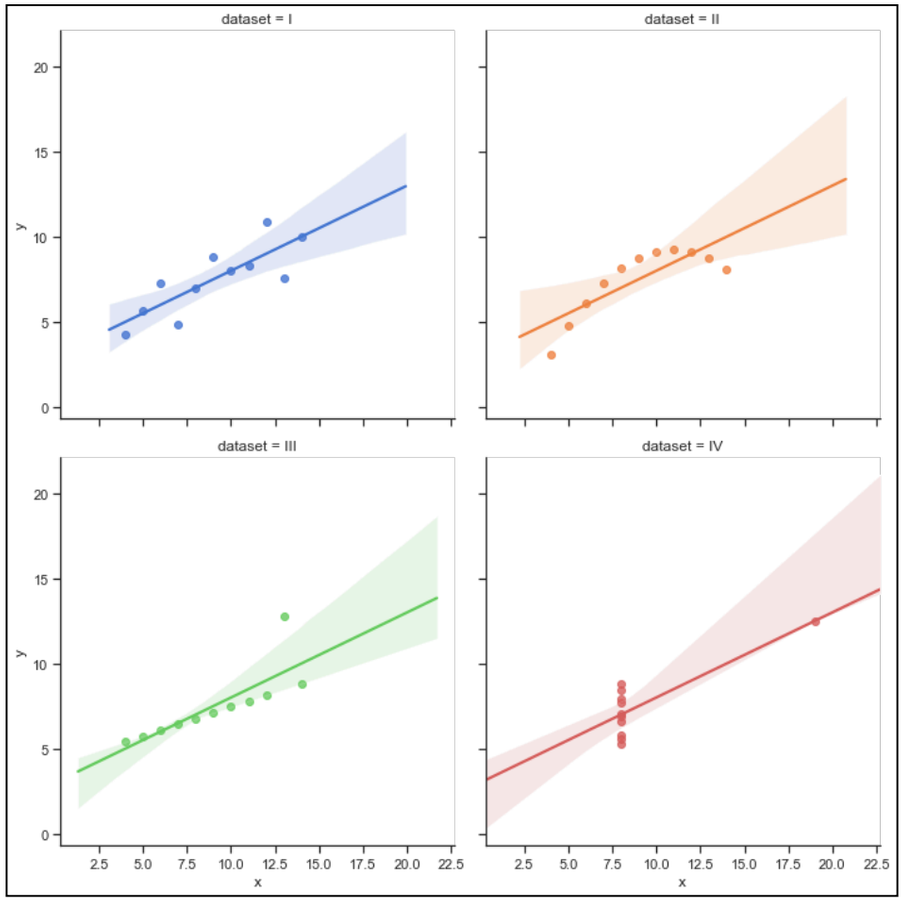
	x								y							
	count	mean	std	min	25%	50%	75%	max	count	mean	std	min	25%	50%	75%	max
dataset																
I	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0	11.0	7.500909	2.031568	4.26	6.315	7.58	8.57	10.84
II	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0	11.0	7.500909	2.031657	3.10	6.695	8.14	8.95	9.26
III	11.0	9.0	3.316625	4.0	6.5	9.0	11.5	14.0	11.0	7.500000	2.030424	5.39	6.250	7.11	7.98	12.74
IV	11.0	9.0	3.316625	8.0	8.0	8.0	8.0	19.0	11.0	7.500909	2.030579	5.25	6.170	7.04	8.19	12.50

기초 통계량의 함정

Anscombe's Quartet

- Anscombe는 4종류 데이터를 기초 통계량의 함정을 보여줬다

```
%matplotlib inline
sns.lmplot(x="x", y="y", col="dataset", hue="dataset",
           col_wrap=2, data=df, palette="muted")
```

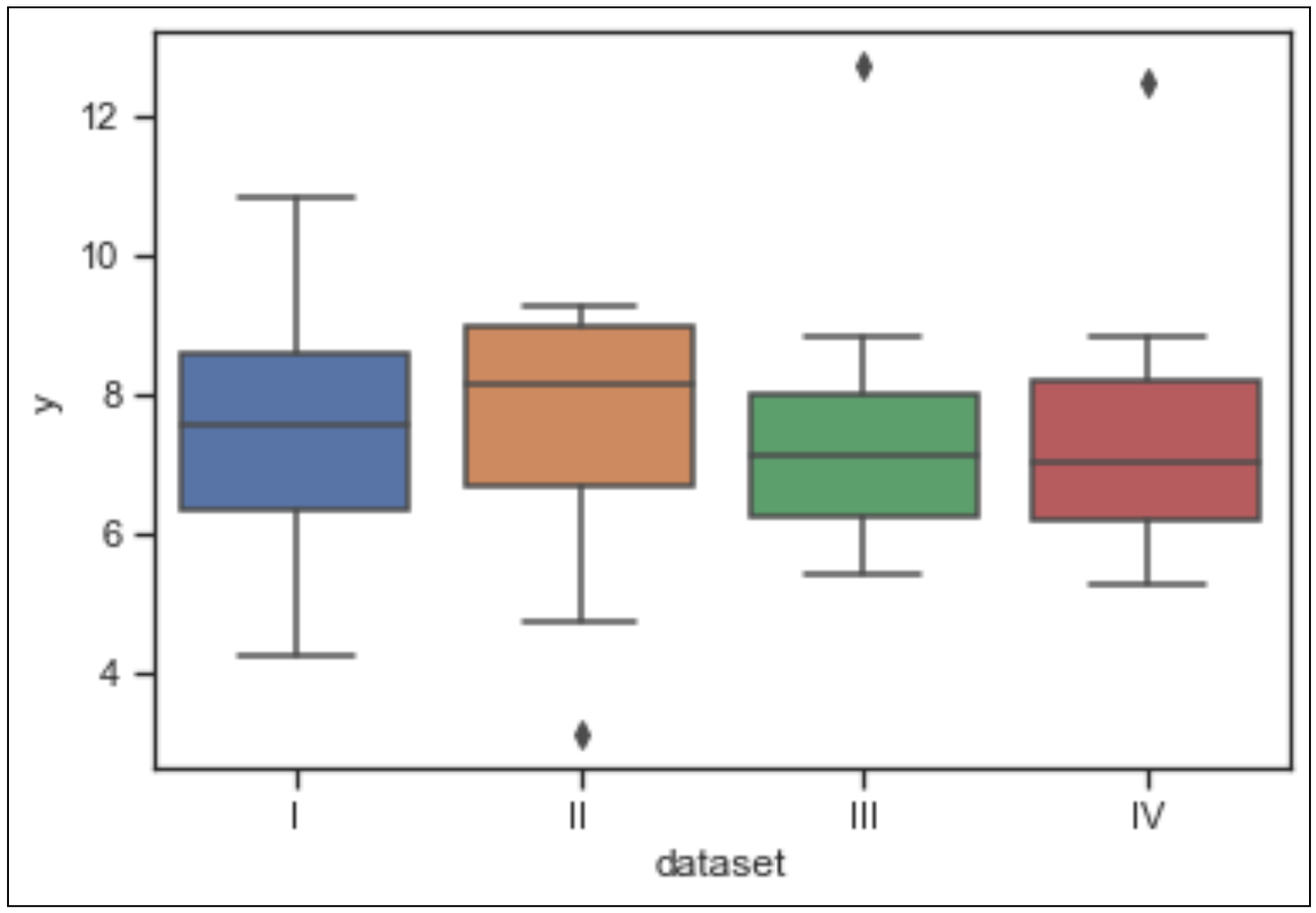


데이터 시각화

Anscombe's Quartet

- 상자 그림(box plot) 그리기

```
sns.boxplot(x="dataset", y="y", data=df)
```

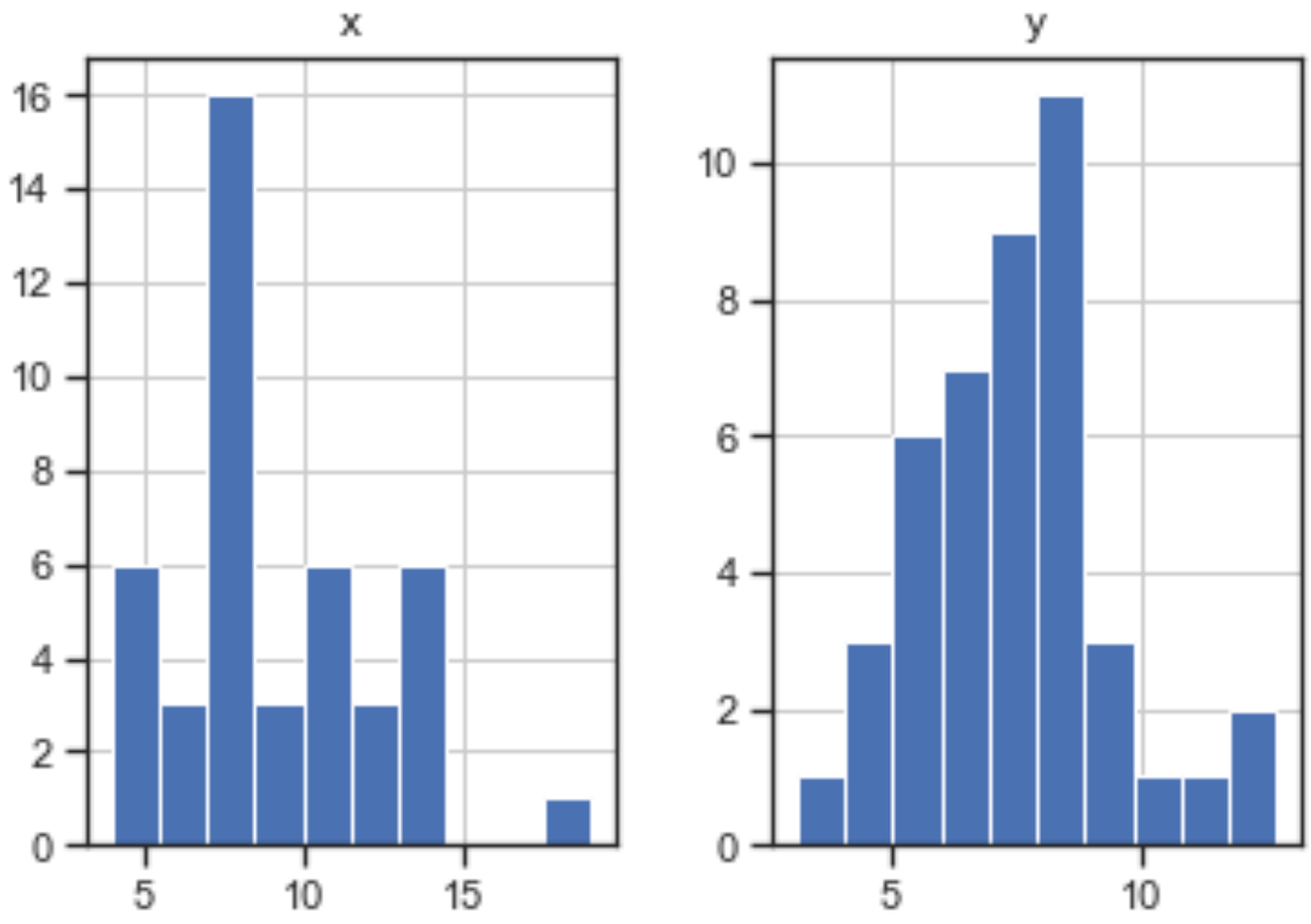


데이터 시각화

Anscombe's Quartet

- 히스토그램 그리기

```
df.hist()
```



데이터 시각화

Titanic

- 타이타닉 사고 데이터를 시각화 해보자

```
titanic = sns.load_dataset("titanic")

titanic.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alo
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	Fal
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	Fal
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	Tr
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	Fal
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	Tr

데이터 시각화

Titanic

- 타이타닉 사고 데이터를 시각화 해보자

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alo
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	Fal
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	Fal

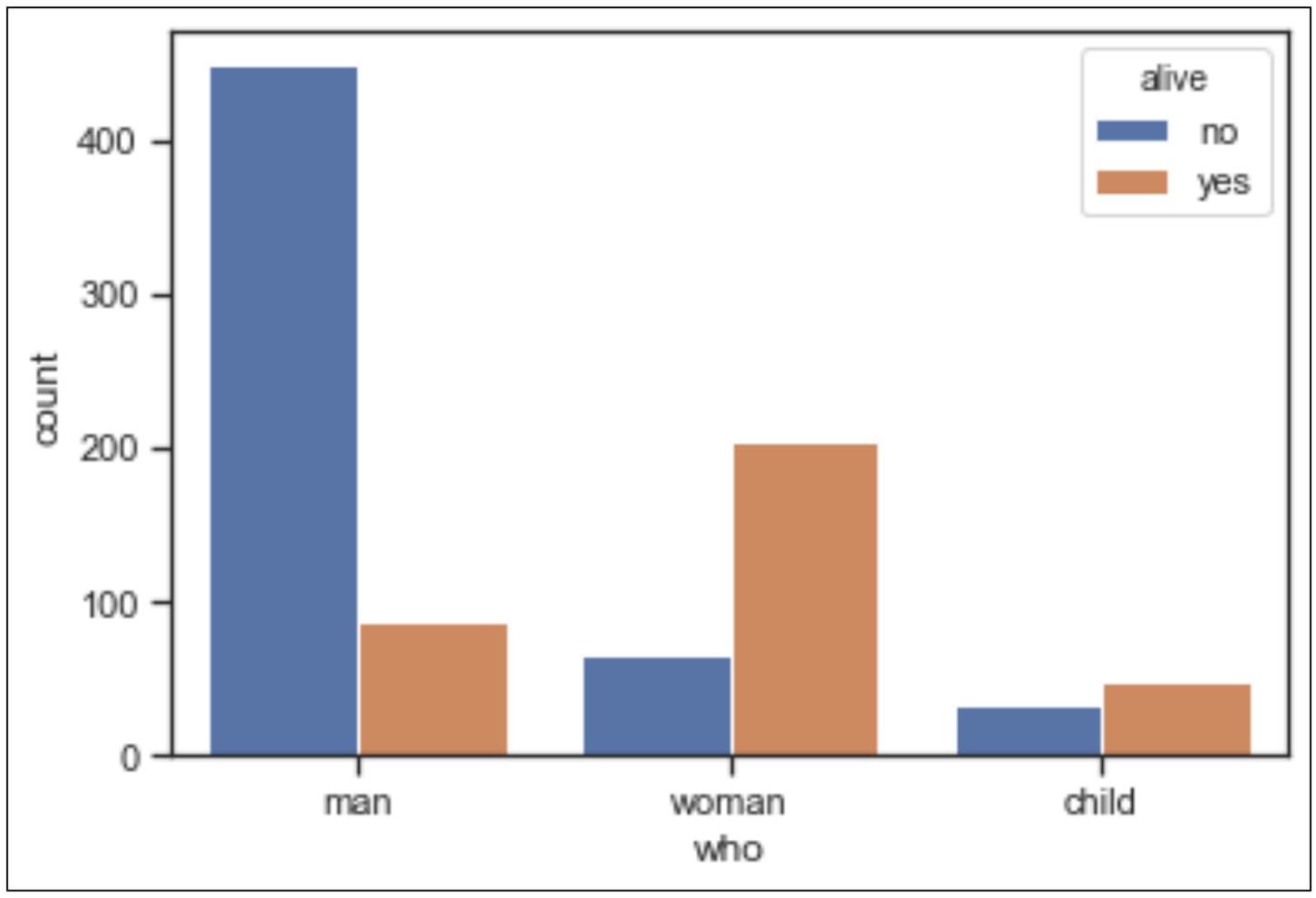
- survival: 생존 여부(0=사망, 1=생존)
- pclass: 탑승 클래스(1=1등석, 2=2등석, 3=3등석)
- sex: 성별
- age: 나이
- sibs: 동승한 형제/배우자 숫자
- parch: 동승한 부모/자녀 숫자
- fare: 티켓 가격
- embarked: 탑승지(C=Cherbourg, Q=Queenstown, S=Southampton)
- who: 성인, 남녀 구분(man, woman, child)
- deck: 갑판(A-G)

데이터 시각화

Titanic

- 타이타닉 사고 데이터를 시각화 해보자

```
sns.countplot(x="who", hue="alive", data=titanic)
```

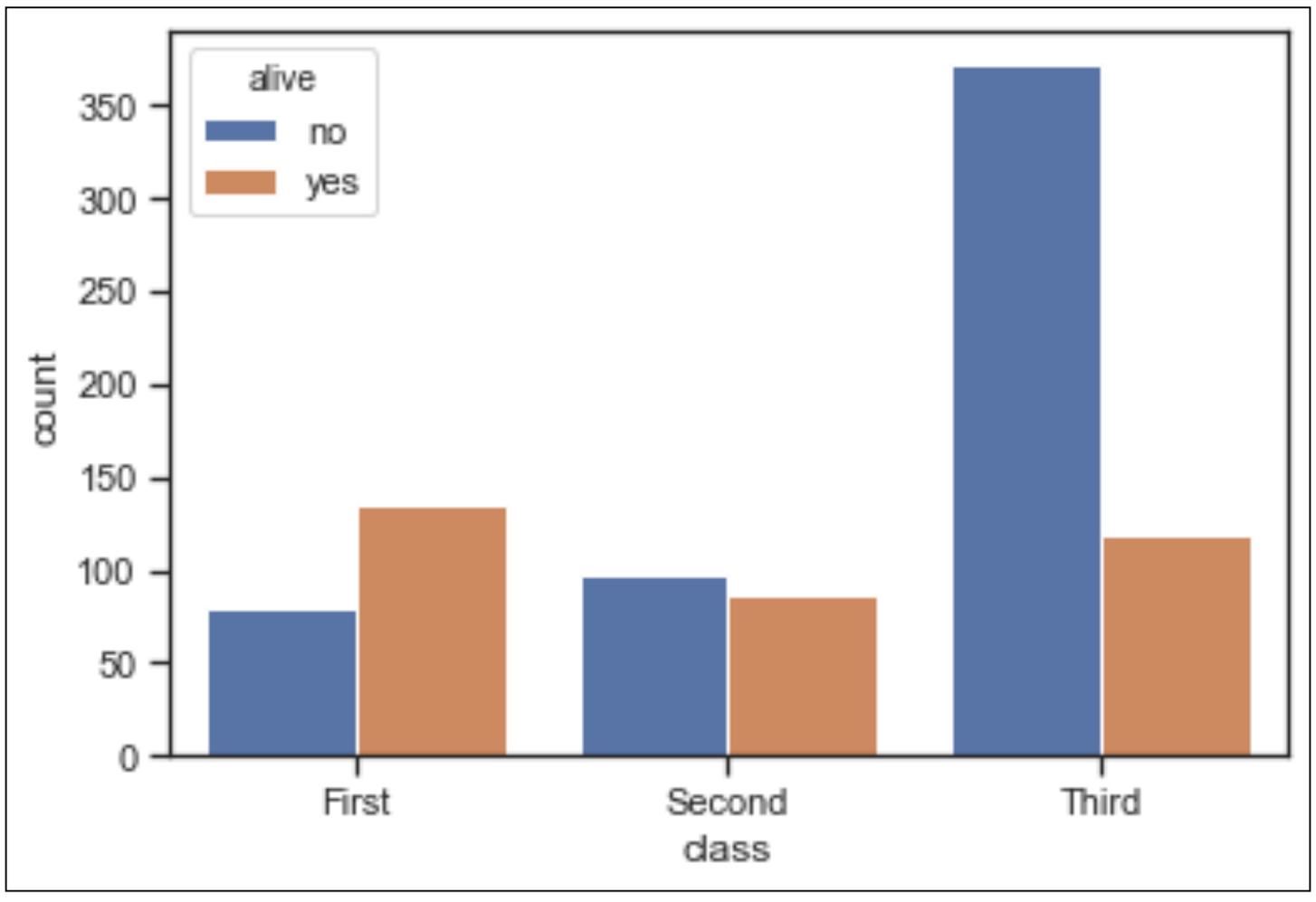


데이터 시각화

Titanic

- 타이타닉 사고 데이터를 시각화 해보자

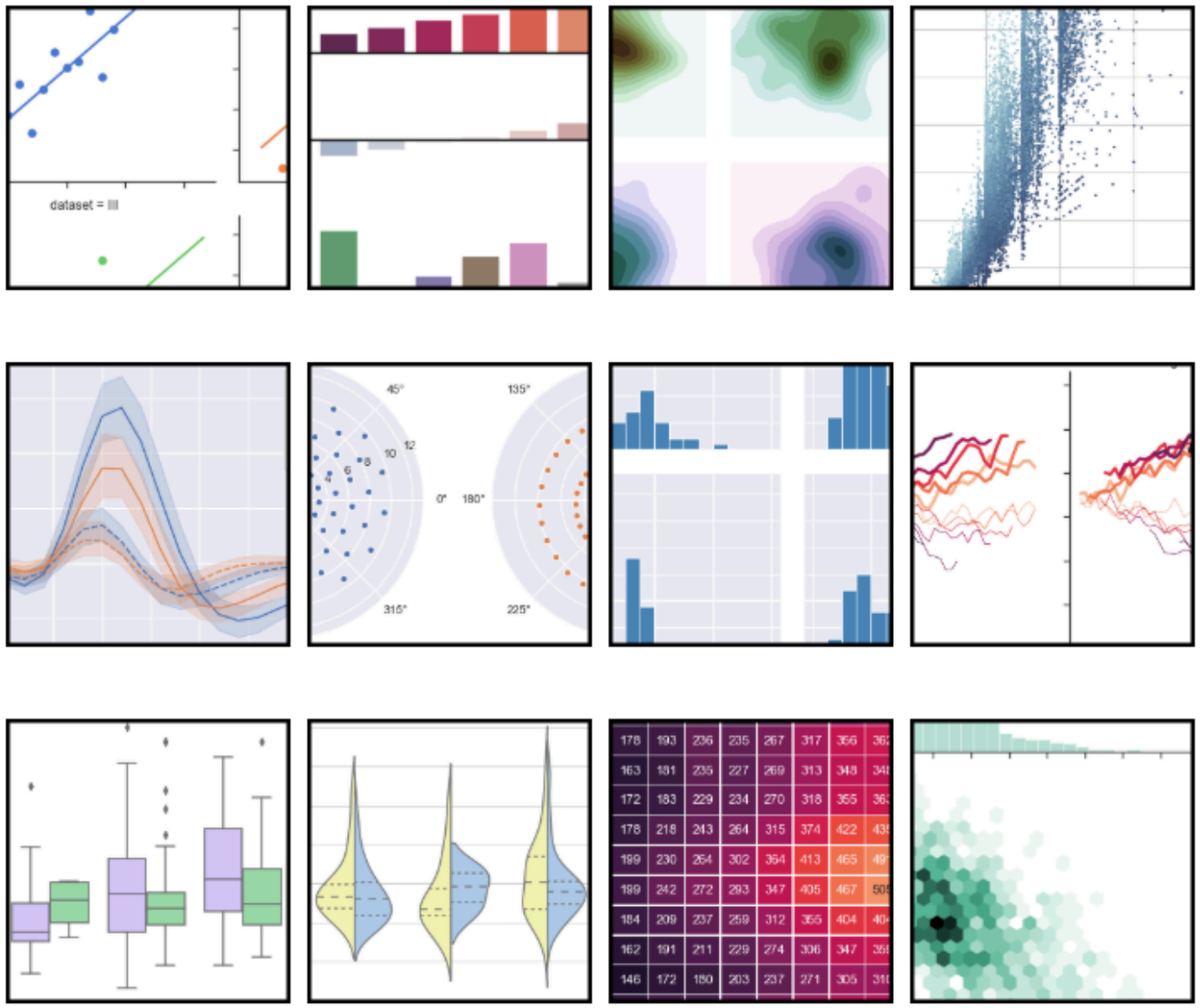
```
sns.countplot(x="class", hue="alive", data=titanic)
```



데이터 시각화

Seaborn

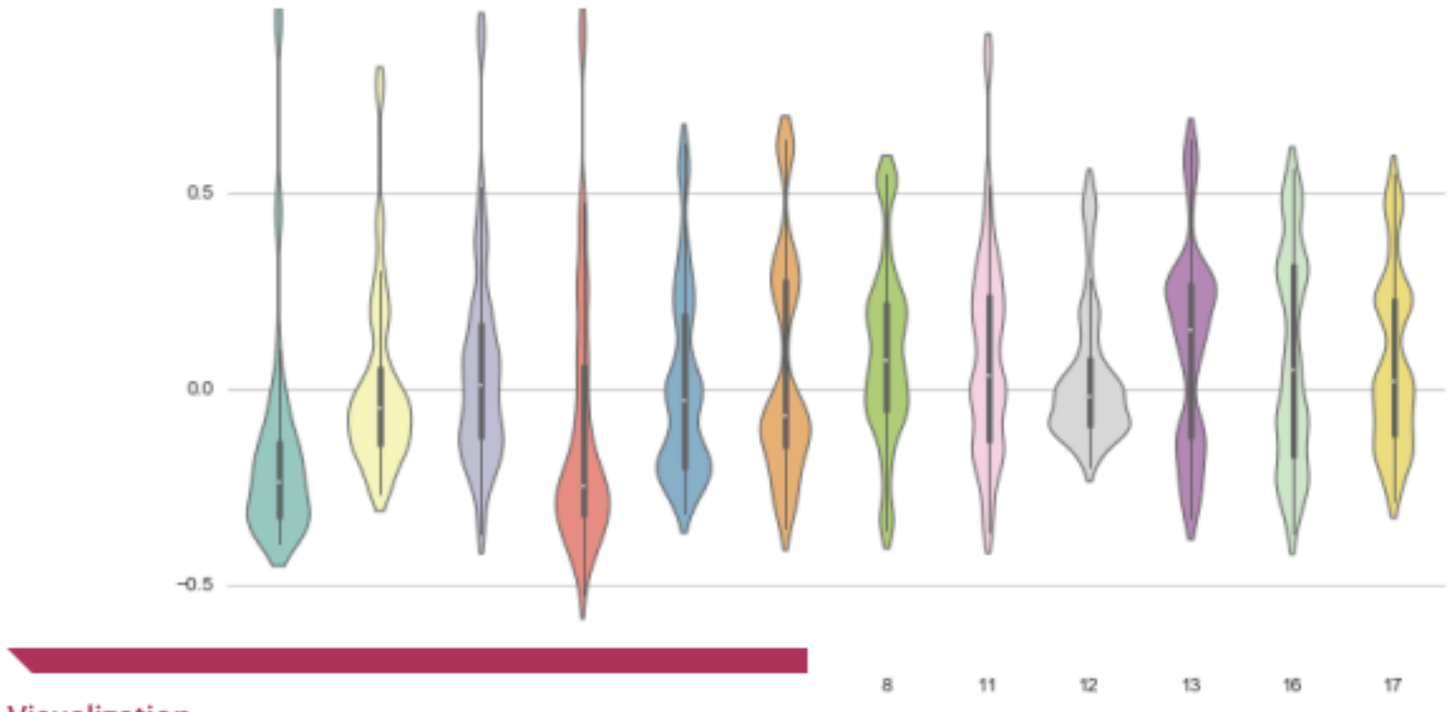
- https://seaborn.pydata.org/



데이터 시각화

그 밖에 유용한 라이브러리

- https://mode.com/blog/python-data-visualization-libraries



Visualization

June 8, 2016 • 8 minute read

10 Useful Python Data Visualization Libraries for Any Discipline

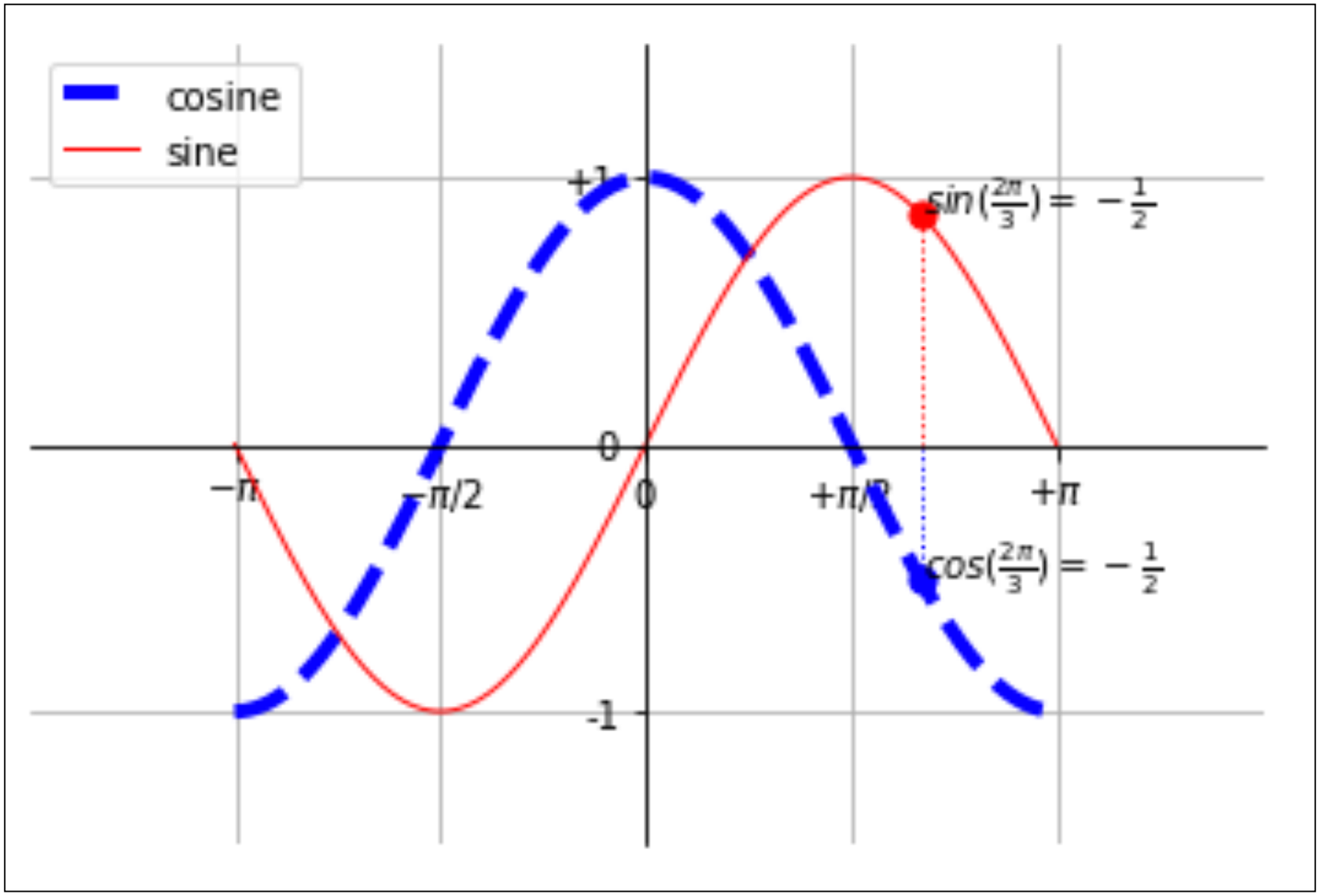
MELISSA BIERLY

CONTENT MARKETING

데이터 시각화

matplotlib

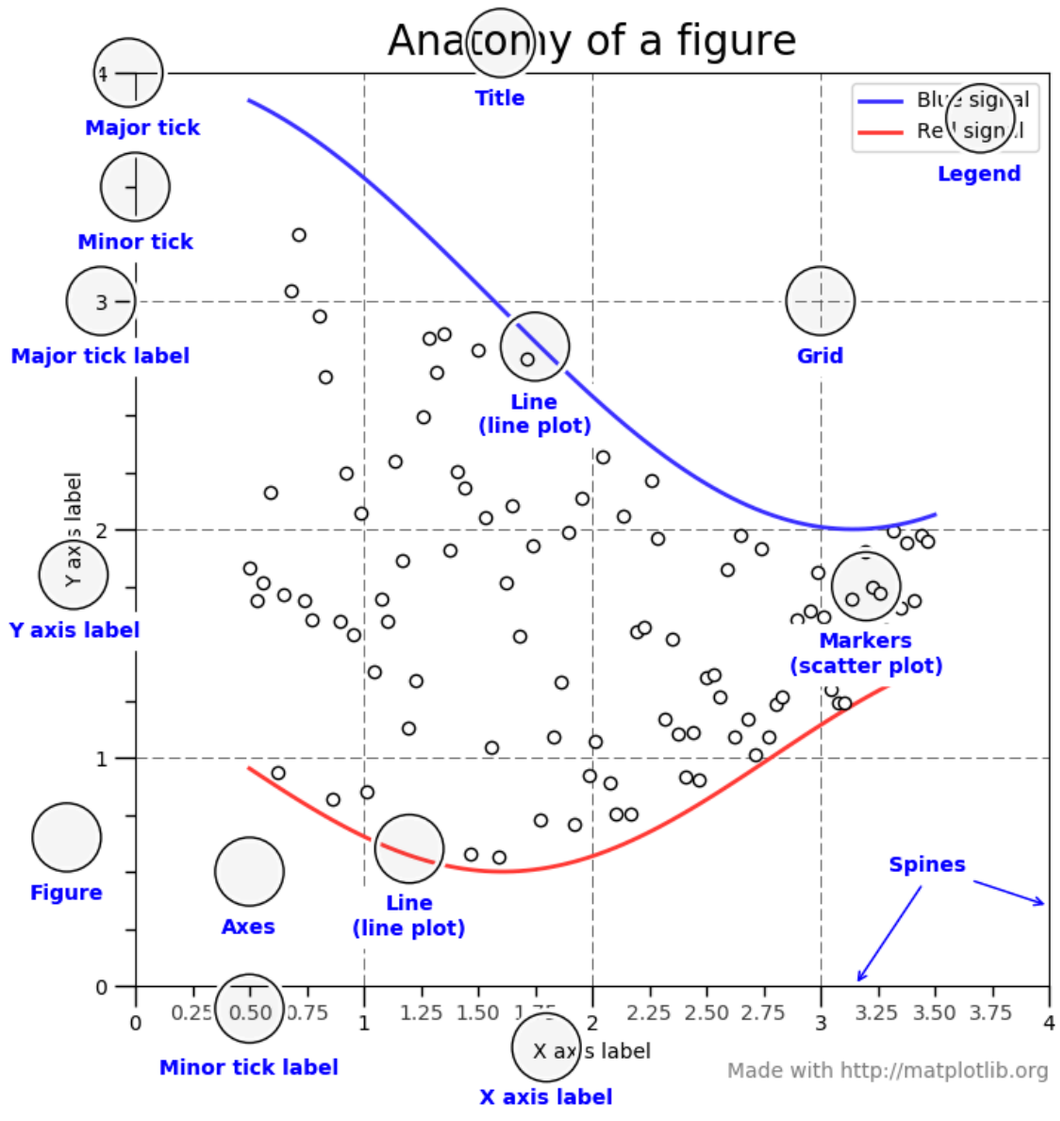
- matplotlib를 이용해서 아래 그래프를 그려보자

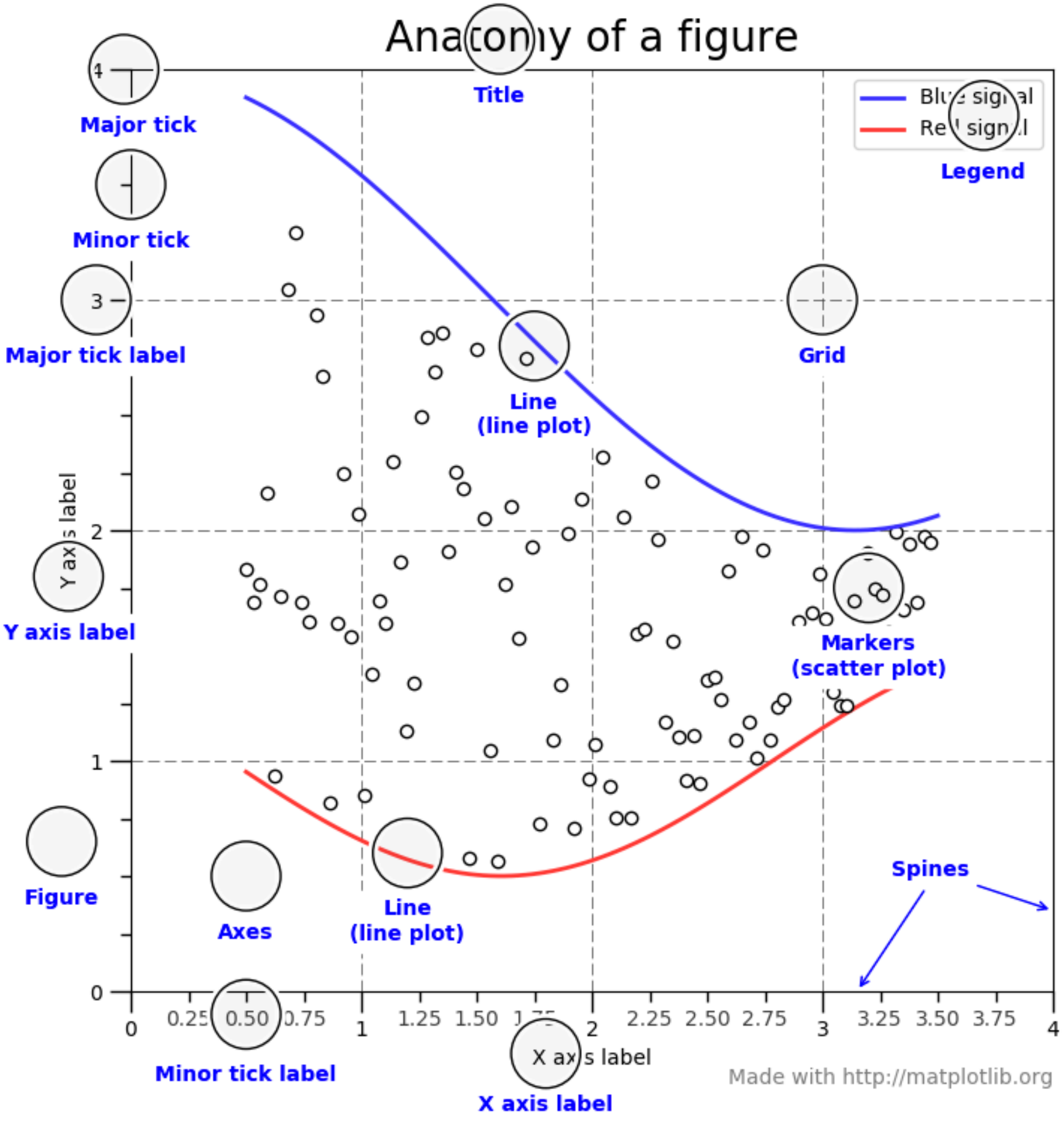


데이터 시각화

matplotlib

- 기본 개념
  - *Figure*: 전체 그래프
  - *Axes*: 그래프를 구성하는 작은 그래프
    - Axis
    - Tick
    - Legend




Anatomy of a figure
Title
Major tick
Minor tick
Major tick label
Legend
Grid
Line
(line plot)
Markers
(scatter plot)
Y axis label
Figure
Axes
Line
(line plot)
Spines
Minor tick label
X axis label
Made with http://matplotlib.org

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드

```
fig, ax = plt.subplots()

X = np.linspace(-np.pi, np.pi, 256, endpoint=True) # added
S, C = np.sin(X), np.cos(X) # added

ax.plot(X, S) # added
ax.plot(X, C) # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 선 색깔, 모양 바꾸기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=3, linestyle="--") # added
ax.plot(X, S, c="red", linewidth=1, linestyle="-") # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 가로축, 세로축 범위 설정하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--")
ax.plot(X, S, c="red", linewidth=1, linestyle="-")

ax.set_xlim(X.min() * 1.5, X.max() * 1.5) # added
ax.set_ylim(C.min() * 1.5, C.max() * 1.5) # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 가로축, 세로축 눈금 지정하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--")
ax.plot(X, S, c="red", linewidth=1, linestyle="-")
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])
ax.set_xticklabels(["-pi", "-pi/2", "0", "+pi/2", "+pi"]) # added
ax.set_yticklabels(["-1", "0", "+1"]) # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 수학 기호 표시하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--")
ax.plot(X, S, c="red", linewidth=1, linestyle="-")
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])

ax.set_xticklabels([r'$-\pi$', r'$-\frac{\pi}{2}$', r'$0$', r'$+
\frac{\pi}{2}$', r'$+\pi$']) # added
ax.set_yticklabels(["-1", "0", "+1"])

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 가로축, 세로축 위치 이동하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--")
ax.plot(X, S, c="red", linewidth=1, linestyle="-")
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])
ax.set_xticklabels([r'$-\pi$', r'$-\frac{\pi}{2}$', r'$0$', r'$+
\frac{\pi}{2}$', r'$+\pi$'])
ax.set_yticklabels(["-1", "0", "+1"])

ax.spines['right'].set_color('none') # added
ax.spines['top'].set_color('none') # added
ax.spines['bottom'].set_position(('data',0)) # added
ax.spines['left'].set_position(('data',0)) # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 범례 표시하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--", label="cosine") # added
ax.plot(X, S, c="red", linewidth=1, linestyle="-", label="sine") # added
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])
ax.set_xticklabels([r'$-\pi$', r'$-\frac{\pi}{2}$', r'$0$', r'$+\frac{\pi}
{2}$', r'$+\pi$'])
ax.set_yticklabels(["-1", "0", "+1"])

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position(('data',0))
ax.spines['left'].set_position(('data',0))

ax.legend(loc='upper left') # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 눈금 표시하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--", label="cosine") # added
ax.plot(X, S, c="red", linewidth=1, linestyle="-", label="sine") # added
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])
ax.set_xticklabels([r'$-\pi$', r'$-\frac{\pi}{2}$', r'$0$', r'$+\frac{\pi}{2}$', r'$
+\pi$'])
ax.set_yticklabels(["-1", "0", "+1"])

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position(('data',0))
ax.spines['left'].set_position(('data',0))

ax.legend(loc='upper left')
ax.grid(True) # added

plt.show()
```

데이터 시각화

matplotlib - 코사인, 사인함수 그래프 그리기

- 코드 - 특정점 강조하기

```
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
S, C = np.sin(X), np.cos(X)

fig, ax = plt.subplots()

ax.plot(X, C, c="blue", linewidth=4, linestyle="--", label="cosine")
ax.plot(X, S, c="red", linewidth=1, linestyle="-", label="sine")
ax.set_xlim(X.min() * 1.5, X.max() * 1.5)
ax.set_ylim(C.min() * 1.5, C.max() * 1.5)
ax.set_xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi])
ax.set_yticks([-1, 0, +1])
ax.set_xticklabels([r'$-\pi$', r'$-\frac{\pi}{2}$', r'$0$', r'$+\frac{\pi}{2}$', r'$+\pi$'])
ax.set_yticklabels(["-1", "0", "+1"])

ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position(('data',0))
ax.spines['left'].set_position(('data',0))

ax.legend(loc='upper left')
ax.grid(True)

t = 2 * np.pi / 3 # added
ax.plot([t, t], [0, np.cos(t)], c='blue', linewidth=1, linestyle=":") # added
ax.scatter(t, np.cos(t), 50, color='blue') # added
ax.annotate(r'$cos(\frac{2\pi}{3} )=-\frac{1} {2} $', xy=(t, np.cos(t))) # added
ax.annotate(r'$sin(\frac{2\pi}{3} )=-\frac{1} {2} $', xy=(t, np.sin(t))) # added
ax.plot([t, t],[0, np.sin(t)], c='red', linewidth=1, linestyle=":") # added
ax.scatter(t, np.sin(t), 50, c='red') # added

plt.show()
```