(1-1) Welcome to Visualization (OT)

데이터 시각화란 데이터를 그래픽 요소로 매핑하여 시각적으로 표현하는 것

시각화는 다양한 요소가 포함된 Task

- 목적 : 왜 시각화 하나요?
- 독자 : 시각화 결과는 누구를 대상으로 하나요?
- 데이터 : 어떤 데이터를 시각화할 것인가요?
- 스토리 : 어떤 흐름으로 인사이트를 전달할 것인가요
- 방법 : 전달하고자 하는 내용에 맞게 효과적인 방법을 사용하고 있나요?
- 디자인 : UI에서 만족스러운 디자인을 가지고 있나요?

이론 추천 도서

- Visualization Analysis and Design, Tamara Munzner
- Fundamentals of Data Visualization, Claus O. Wilke

개발 추천 사이트

- https://kaggle.com
- https://observablehq.com/
- https://dataviztoday.com/
- https://medium.com/nightingale
- http://ieeevis.org/year/2021/welcom

(1-2) 시각화의 요소

데이터

1. 데이터셋 관점 (global)
2. 개별 데이터의 관점 (local)

정형 데이터

- 통계적 특성과 feature 사이 관계
- 데이터 간 관계
- 데이터 간 비교

시계열 데이터

- 기온, 주가 등 정형데이터와 음성, 비디오와 같은 비정형 데이터 존재
- 시간 흐름에 따른 추세(Trend), 계절성(Seasonality), 주기성(Cycle) 등을 살핌

지리/지도 데이터

- 지도 정보와 보고자 하는 정보 간의 조화 중요 + 지도 정보를 단순화 시키는 경우도 존재

- 거리, 경로, 분포 등 다양한 실사용

관계 데이터

- 객체와 객체 간의 관계를 시각화 => Graph visualization / Network Visualization
- 객체는 Node로, 관계는 Link로 표현
- 크기, 색, 수 등으로 객체와 관계의 가중치를 표현
- 휴리스틱하게 노드 배치를 구성

계층적 데이터

- 관계 중에서도 **포함관계**가 분명한 데이터
- 네트워크 시각화로도 표현 가능
- Tree, Treemap, Sunburst 등이 대표적

수치형(numerical)

- 연속형 (continuous)
- 이산형 (discrete) : 비례 가능

범주형(categorical)

- 명목형 (norminal)
- 순서형 (ordinal) : 비례 불가능

시각화

mark : 전, 선, 면으로 이루어진 데이터 시각화

channel : 각 mark를 변경할 수 있는 요소들

- 위치, 색, 모양, 크기, 기울기 등

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

print(f'numpy version : {np.__version__}') # version check
print(f'matplotlib version : {mlt.__version__}') # version check
```

```
numpy version : 1.19.5
matplotlib version : 3.2.2
```

```
!pip install numpy==1.18.5
!pip install matplotlib==3.3.4
```

```
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp37-cp37m-manylinux1_x86_64.whl (20.1 MB)
     |████████████████████████████████| 20.1 MB 1.2 MB/s
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.19.5
    Uninstalling numpy-1.19.5:
      Successfully uninstalled numpy-1.19.5
ERROR: pip's dependency resolver does not currently take into account all the packages that are install
tensorflow 2.5.0 requires numpy~=1.19.2, but you have numpy 1.18.5 which is incompatible.
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompatible.
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 which is incompatible.
Successfully installed numpy-1.18.5
Collecting matplotlib==3.3.4
  Downloading matplotlib-3.3.4-cp37-cp37m-manylinux1_x86_64.whl (11.5 MB)
     |████████████████████████████████| 11.5 MB 6.3 MB/s
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib=
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in /usr/local/lib/python3.7/dis
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packages (from matplotlib==
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplo
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from mat
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from cycler>=0.10->matplo
Installing collected packages: matplotlib
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.2.2
    Uninstalling matplotlib-3.2.2:
      Successfully uninstalled matplotlib-3.2.2
ERROR: pip's dependency resolver does not currently take into account all the packages that are install
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 which is incompatible.
```

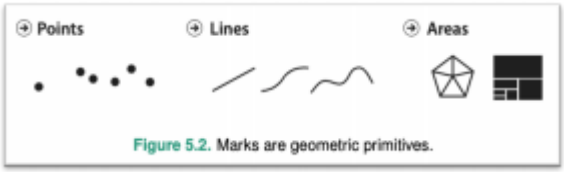


Figure 5.2. Marks are geometric primitives.

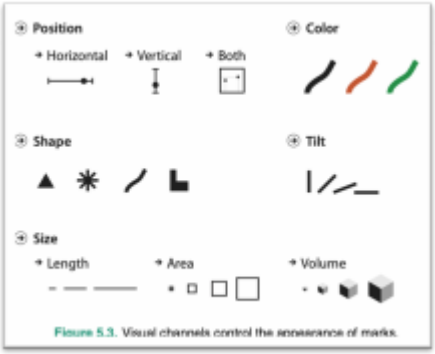
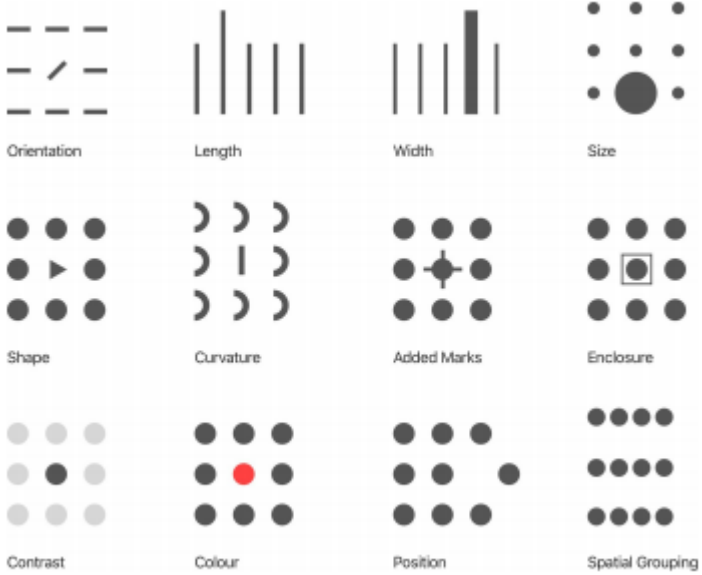


Figure 5.3. Visual channels control the appearance of marks.

전주의적(Pre-attentive Attribute) 속성 : 주의를 주지 않아도 인지하게 되는 요소

- 동시에 사용하면 인지하기 어려움
- 적절하게 사용하는 방법은 시각적 분리(visual pop-out)하는 것



(1-3) Python과 Matplotlib

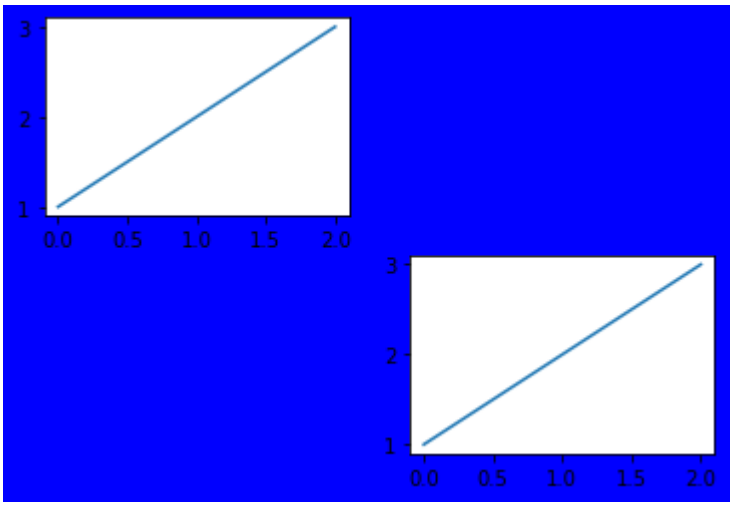
D3.js : 점, 선, 면 직접 구현해야 함

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

print(f'numpy version : {np.__version__}') # version check
print(f'matplotlib version : {mpl.__version__}') # version check
```

```
numpy version : 1.18.5
matplotlib version : 3.3.4
```

Pyplot API : 순차적 방법

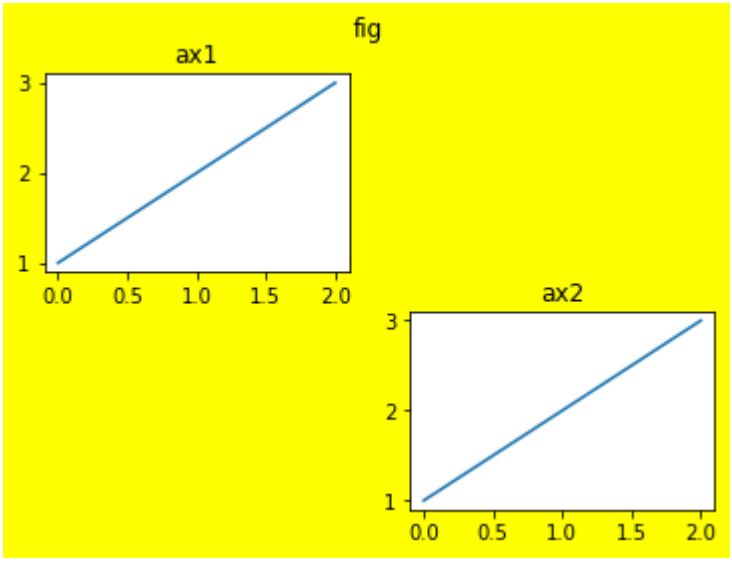
```
# %matplotlib inline 매직 키워드
fig = plt.figure()

# set_facecolor
fig.set_facecolor('blue')

# 1행 2열

ax1 = fig.add_subplot(221) # == fig.add_subplot(1,2,1) 첫번째 액자

# 그래프 순차적으로 그려짐
x = [1,2,3]
plt.plot(x)

ax2 = fig.add_subplot(224) # 두번째 액자

x = [1,2,3]
plt.plot(x)

plt.show() # ipynb 아니라면 필수코드
```



객체지향(Object-Oriented) API : 그래프에서 각 객체에 대해 직접적으로 수정해서

- plt.gcf().get_axes()로 다시 서브플롯 객체를 받아서 사용할 수도 있음

```
# %matplotlib inline 매직 키워드
fig = plt.figure()

# set_facecolor
fig.set_facecolor('yellow')

# 1행 2열
x = [1,2,3]
x = [1,2,3]

ax1 = fig.add_subplot(221) # == fig.add_subplot(1,2,1) 첫번째 액자
ax2 = fig.add_subplot(224) # 두번째 액자

ax1.plot(x)
ax2.plot(x)

ax1.set_title('ax1')
ax2.set_title('ax2')

fig.suptitle('fig') # super

plt.show() # ipynb 아니라면 필수코드
```



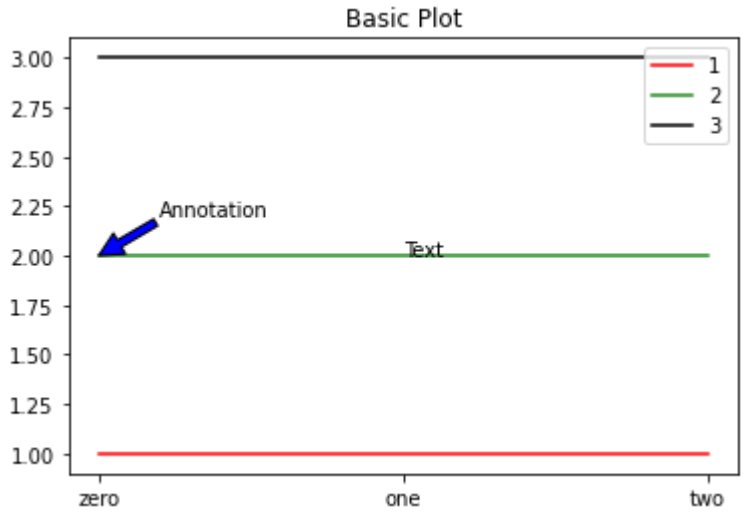
서브플롯에 여러 그래프 그리기

ax에서 데이터 변경 시 .set_{} 형태의 메서드가 많다.

set으로 설정한 정보는 .get_{} 으로 받아올 수 있다.

annotate는 화살표 추가 가능

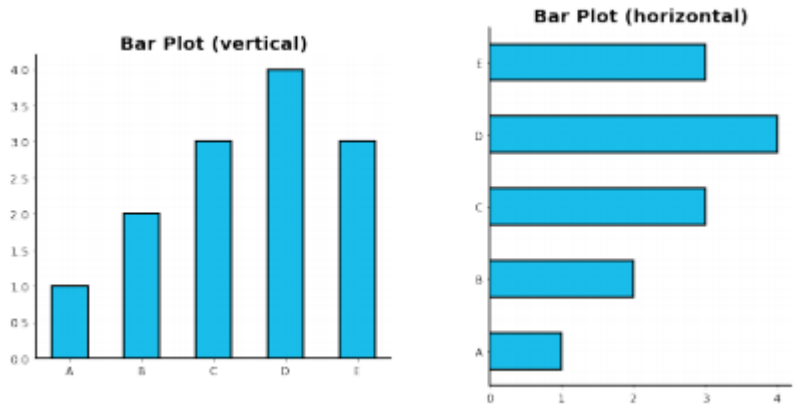
```
fig = plt.figure()
ax = fig.add_subplot(111)

# 3개의 그래프

ax.plot([1, 1, 1], color='r', label='1') # 색상 단축이름, 빠르게 그릴 때 좋음
ax.plot([2, 2, 2], color='forestgreen', label='2') # css 색상명, 외우기 어려워
ax.plot([3, 3, 3], color='#000000', label='3') # RGB 색상코드(hex code), 검정

ax.set_title('Basic Plot')
ax.set_xticks([0,1,2])
ax.set_xticklabels(['zero','one','two'])

# 텍스트 추가
ax.text(x=1, y=2, s='Text')
ax.annotate(text='Annotation', xy=(0,2), xytext=(.2,2.2), arrowprops=dict(facecolor='blue'))
# ax.annotate(s='Annotation', xy=(0,2), xytext=(.2,2.2), arrowprops=dict(facecolor='blue'))

ax.legend() # 범례 추가

plt.show()
```



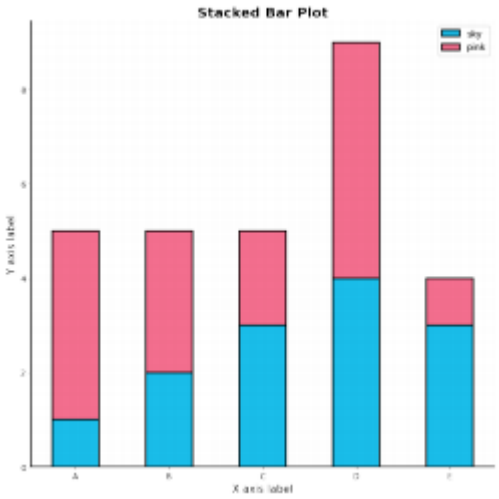
(2-1) Bar Plot 사용하기

막대 그래프 종류

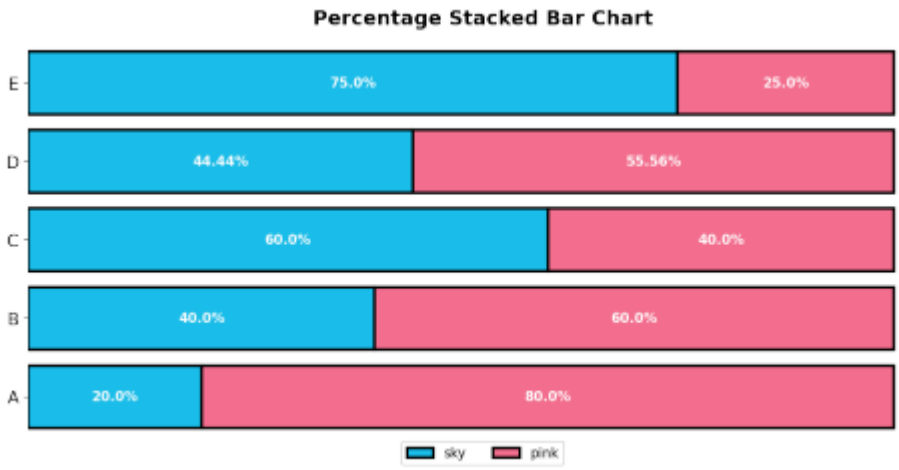
범주 많을 때 수평 막대 그래프 사용



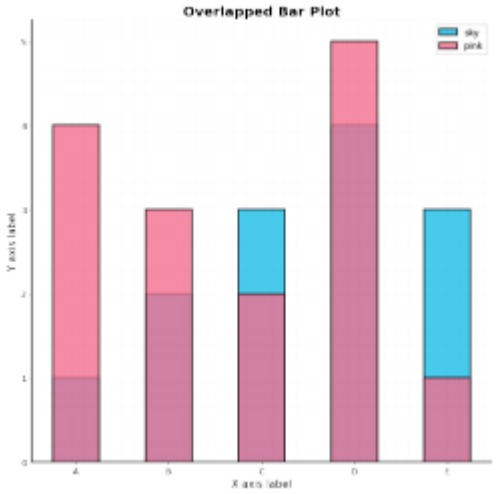
Stacked Bar Plot



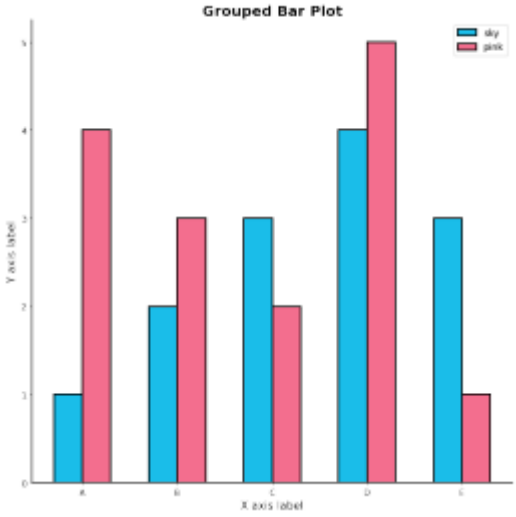
Percentage Stacked Bar Chart : 전체에서 비율을 나타내는 그래프



Overlapped Bar Plot : 비교 쉬우나 두 개의 그룹만 사용 가능



Grouped Bar Plot :



Principle of Proportion Ink : 실제 값과 그에 표현되는 그래픽으로 표현되는 잉크 양은 비례해야 함

데이터 정렬

1. 시계열 : 시간순
2. 수치형 : 크기순, 1,2,3,4
3. 순서형 : 범주의 순서대로, 초중고
4. 명목형 : 범주의 값 따라 정렬, 최대-최소

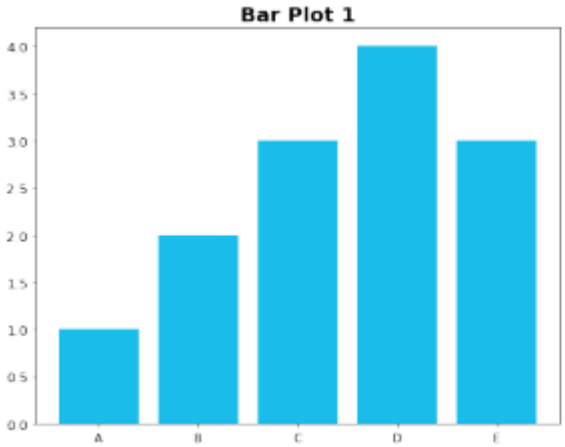
여러 가지 기준으로 정렬을 하여 패턴을 발견

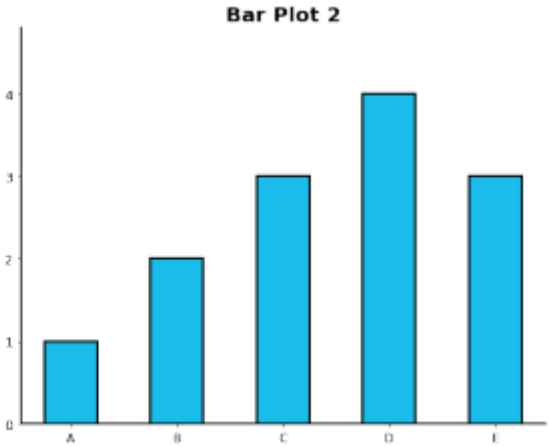
대시보드에서는 Interactive로 제공하는 것이 유용

Matplotlib techniques

- X/Y axis Limit (.set_xlim(), .set_ylime()) : 0 -> 0.5
- Spines (.spines[spine].set_visible()) : 네모박스 없애기
- Gap (width) : 막대 두께, 기본 0.8, 줄이면 가독성 증가
- Legend (.legend())

- Margins (.margins()) : 막대 테두리 조절





실습 코드

[ ] ↳ 숨겨진 셀 *46*개

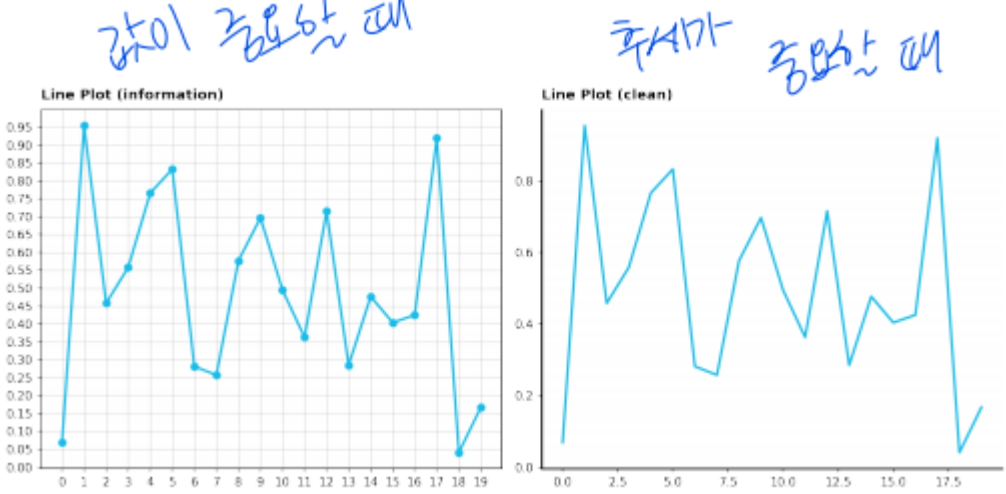
(2-2) Line Plot 사용하기

선 그래프 특징

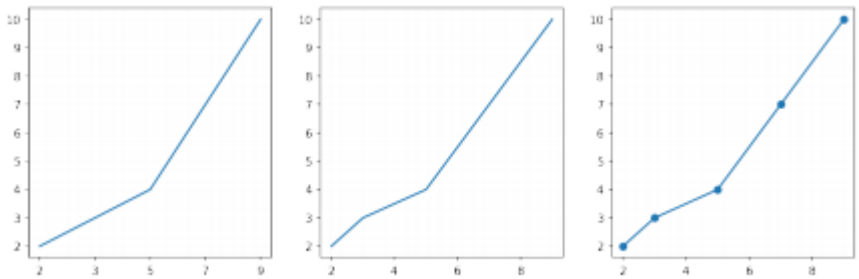
- 시간/순서에 대한 변화에 적합, 시계열 분석에 특화
- 가독성을 위해 5개 이하의 선을 사용
- 노이즈 축소를 위한 전처리로 smoothing 사용
    - moving average, 보간법 등 사용
- 잉크 비례 원칙에서 벗어난 편이다.

선 그래프의 요소

- 색상
- 마커(마커 모양, 마커 사이즈)
- 선의 종류(선의 모양, 선의 두께)



규칙적인 간격의 데이터가 아니라면 마커를 사용하자.



일반적인 분석에서 보간법 사용 지양하자.

실습코드

[ ] ↳ 숨겨진 셀 33개

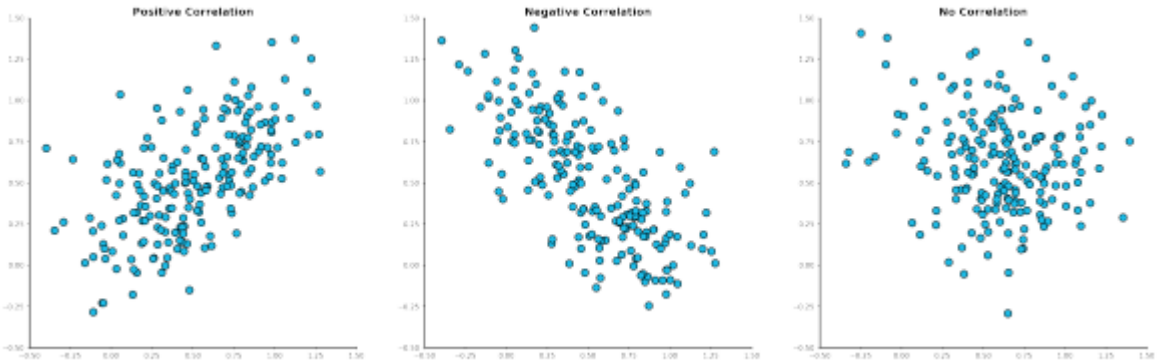
(2-3) Scatter Plot 사용하기

Scatter Plot의 요소

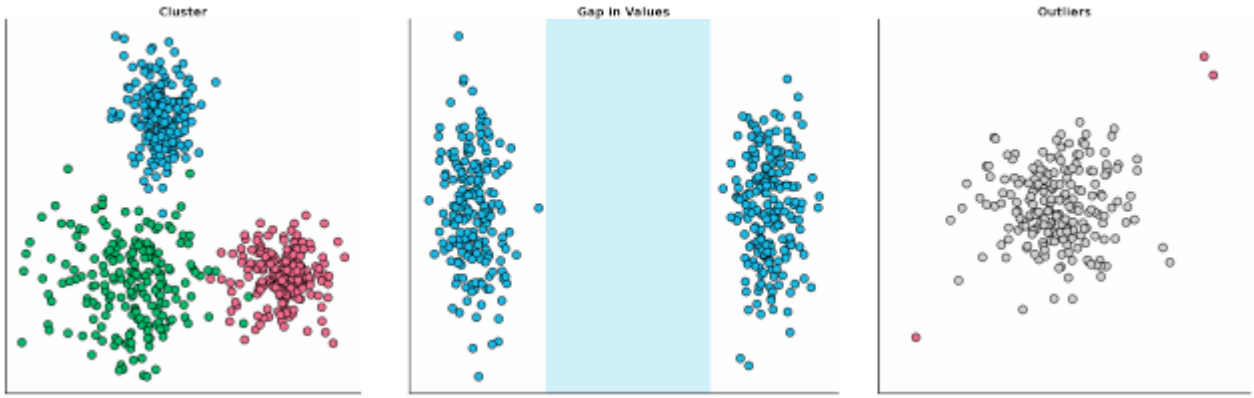
- 색 (color)
- 모양 (marker)
- 크기 (size)

Scatter plot의 목적

상관 관계 확인



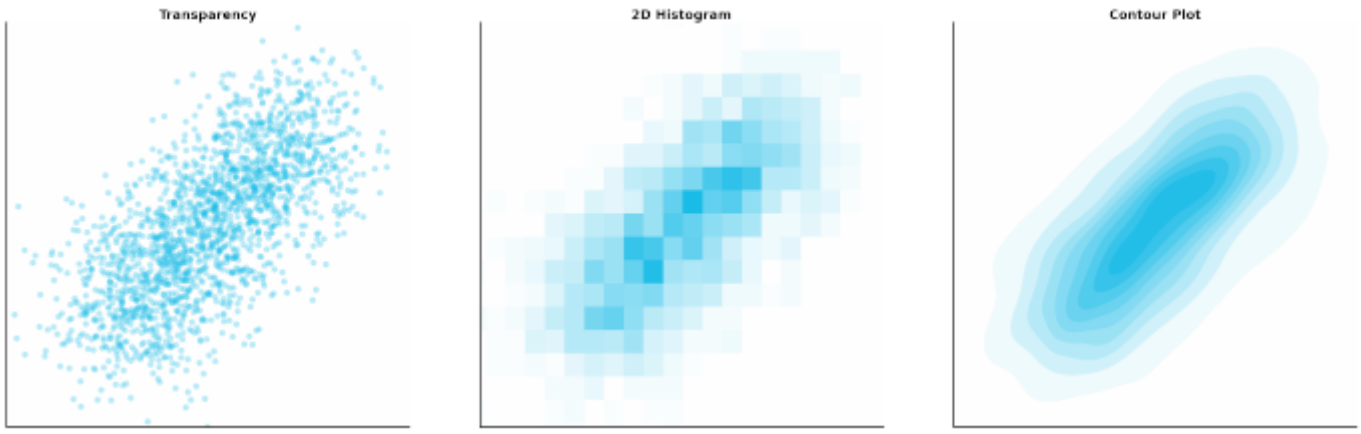
군집 / 값 사이의 차이 / 이상치



Overplotting

산점도에서 점이 많아질수록 점의 분포를 파악하기 어려움

- 투명도 조정
- 지터링 (jittering) : 점의 위치를 약간씩 변경, 효과는 미미
- 2차원 히스토그램 : 히트맵을 사용하여 깔끔한 시각화, 투명도 조정도 같이 사용하면 좋다.
- Contour plot : 분포를 등고선을 사용하여 표현, 등고선 좁으면 밀도 넓으면 낮으면 밀도 낮다.



점의 요소와 인지

색

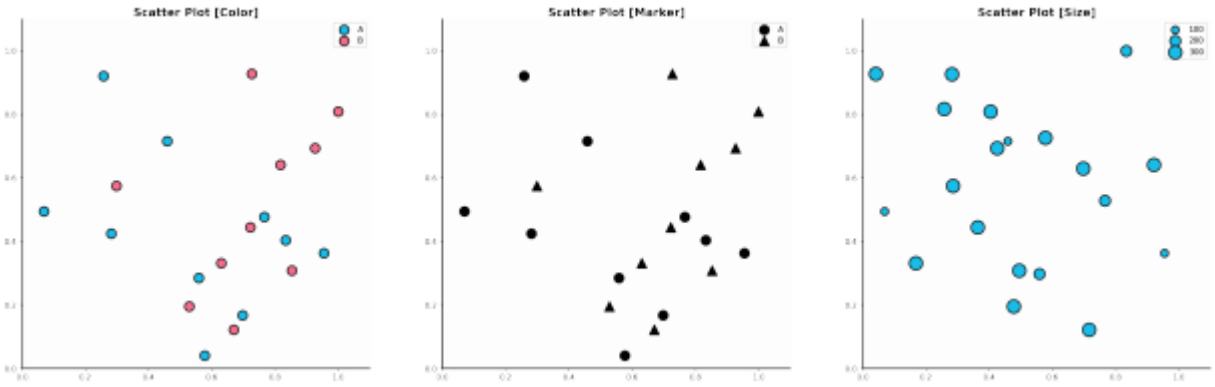
- 연속은 gradient, 이산은 개별 색상으로 사용

마커

- 거의 구별하기 힘들다 + 크기가 고르지 않음
- 동그라미, 네모 세모 모양의 잉크 비례 원칙에 따르지 않는다.

크기

- 흔히 버블 차트 (bubble chart)라고 부름
- 구별하기는 쉽지만 **오용하기 쉬움** (원의 크기 비교)
- 관계보다는 **각 점간 비율에 초점**을 둔다면 좋음
- SWOT 분석 등에 활용 가능

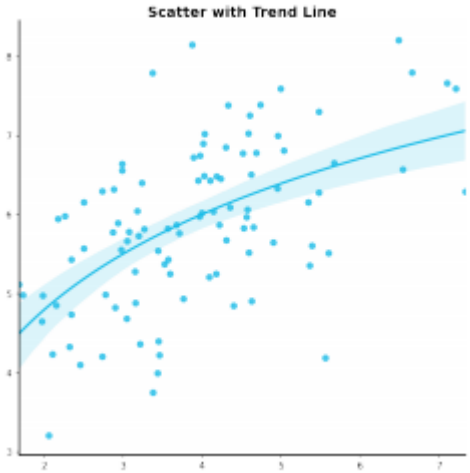


인과관계와 상관관계

인과 관계는 항상 사전 정보와 함께 가정으로 제시해야 한다.

추세선

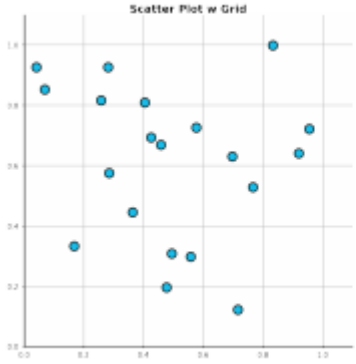
추세선으로 scatter의 패턴을 유추할 수 있으나 추세선이 2개 이상이면 가독성이 떨어진다.

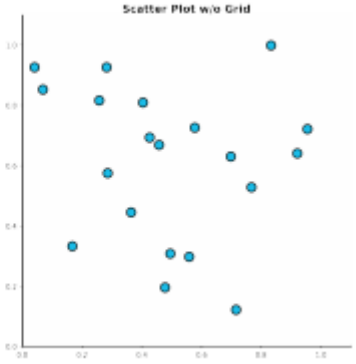


기타

산점도에서 Grid는 지양한다. 사용한다면 최소한으로 쓰고 Grid의 색은 무채색이어야 한다.

범주형이 포함된 관계에서는 heatmap 또는 bubble chart를 권장





실습 코드

↳ 숨겨진 셀 22개