# 2020 D&A

# BASIC SESSION

4. 시각화

D&A

# Contents



D&A

## 01 시각화의 목적

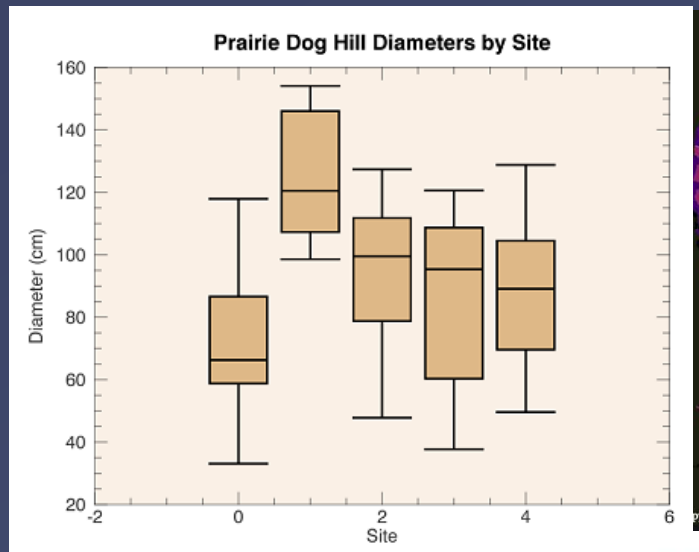
# 시각화(Visualization)?

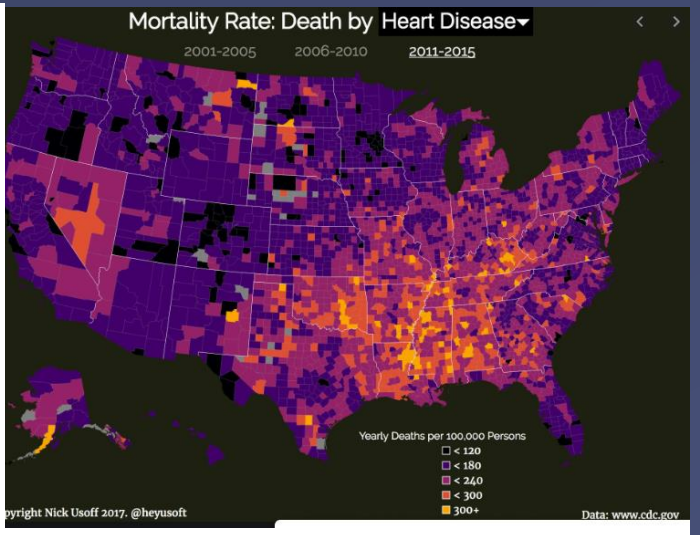
(사전적 정의)
데이터의 분석결과를 이해하기 쉽도록 시각화 도구를 통해 전달하는 것

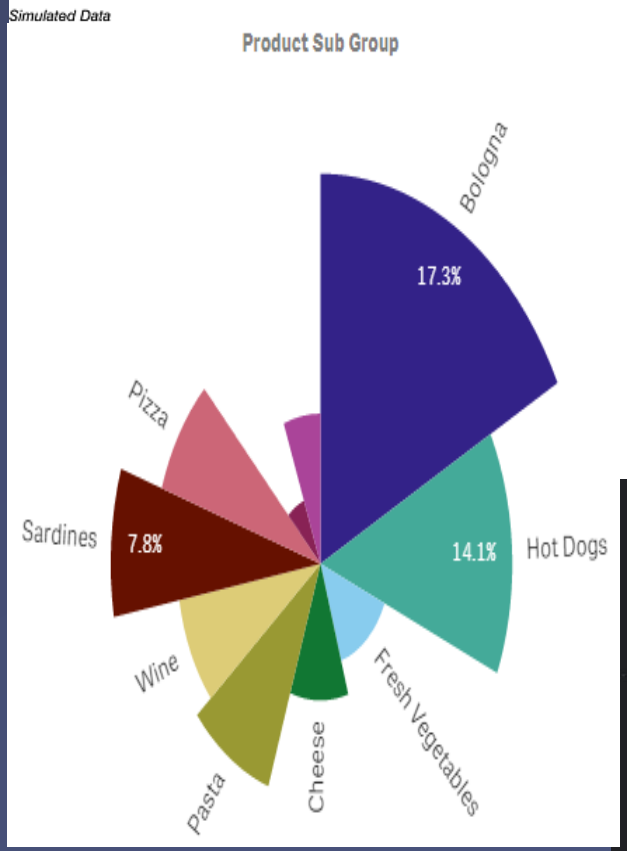
EDA(Exportary Data Analytics)를 하기 위함

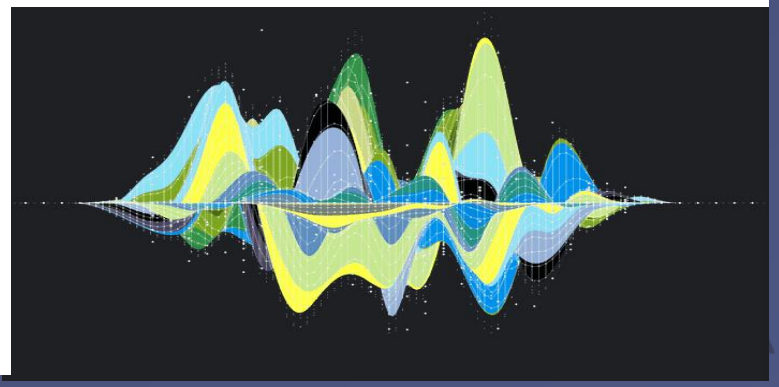
결측치와 이상치 처리를 할 때 특히 유용함

데이터 분석 결과를 공유할 때 유용함



# Matplotlib이란?

파이썬에서 시각화를 할 때 가장 흔하게 쓰이는 시각화 라이브러리
실선, 막대, 히스토그램, 박스 도표, 산점도, 파이, 히트맵 등 다양한 그래프를 지원함

둘 중 어떻게 써도 상관없지만
%matplotlib inline 은 써줘야 함!

```
In [2]: import matplotlib.pylab as plt
        %matplotlib inline
```

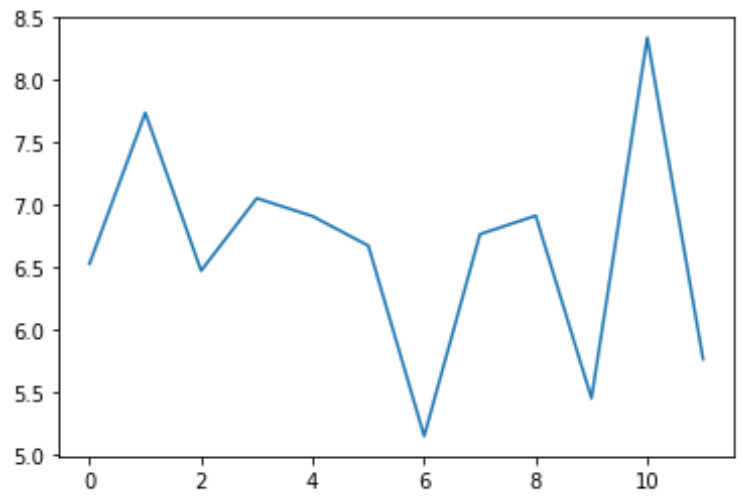
```
In [3]: import matplotlib.pyplot as plt
        %matplotlib inline
```

# 02 Matplotlib - plot(실선 그래프)

변화율을 나타낼 때 유용하게 사용

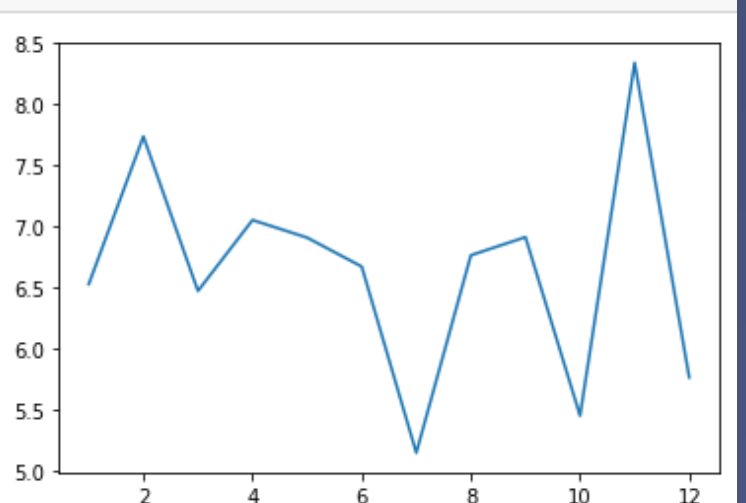
## plt.plot(y)
## -> y값만 지정되면 x값은 0부터 자동으로 지정됨

```
In [6]: y = df['첫회시청률']

        plt.plot(y)
        plt.show() -> 그래프를 출력하는 메소드
```




## X값과 y값 따로 지정 가능
## plt.plot(x,y)

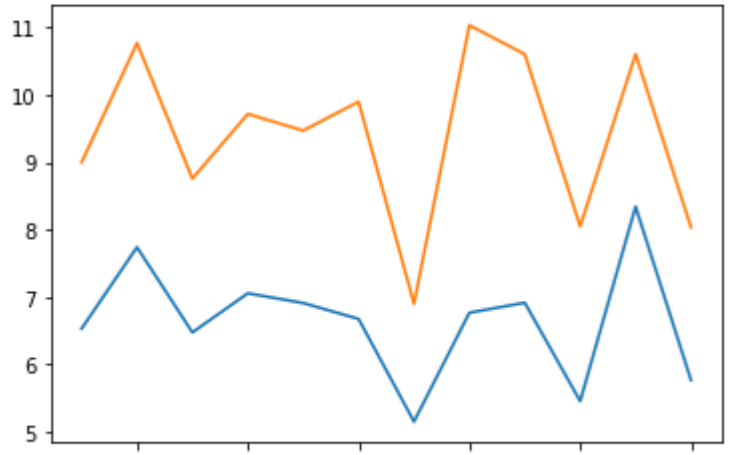
```
In [6]: x = df['방영월']
        y = df['첫회시청률']

        plt.plot(x,y)
        plt.show()
```



# 02 Matplotlib - plot(실선 그래프)

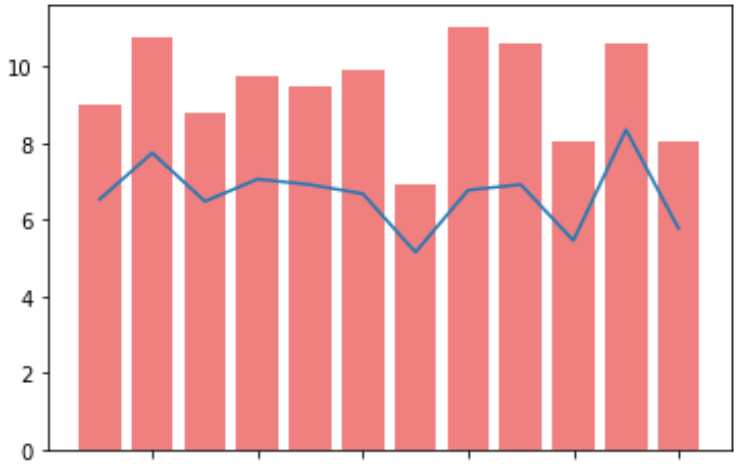
## 한 그래프에 여러 개의 선 그리기

```
In [8]: x1 = df['방영월'] ; x2 = df['방영월']
        y1 = df['첫회시청률'] ; y2 = df['마지막시청률']

        plt.plot(x1,y1)
        plt.plot(x2,y2)
        plt.show()
```




## 한 그래프에 선과 막대 동시에 그리기

```
In [9]: x1 = df['방영월'] ; x2 = df['방영월']
        y1 = df['첫회시청률'] ; y2 = df['마지막시청률']

        plt.plot(x1,y1)
        plt.bar(x2,y2,color='lightcoral')
        plt.show()
```

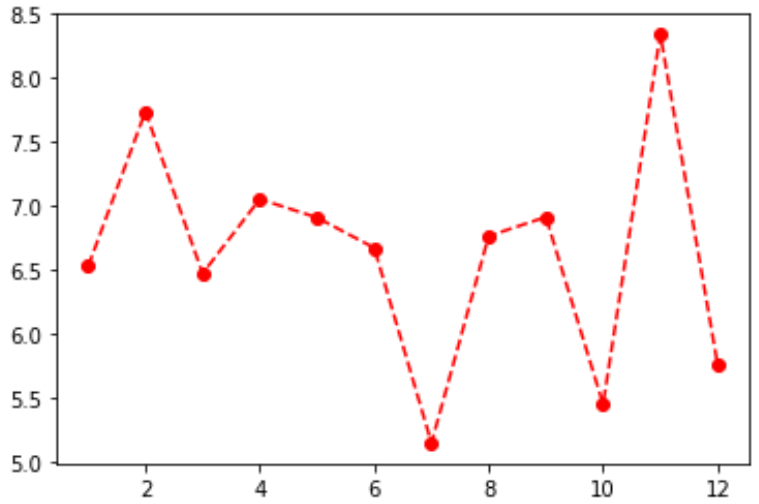


# 02 Matplotlib - plot(실선 그래프)

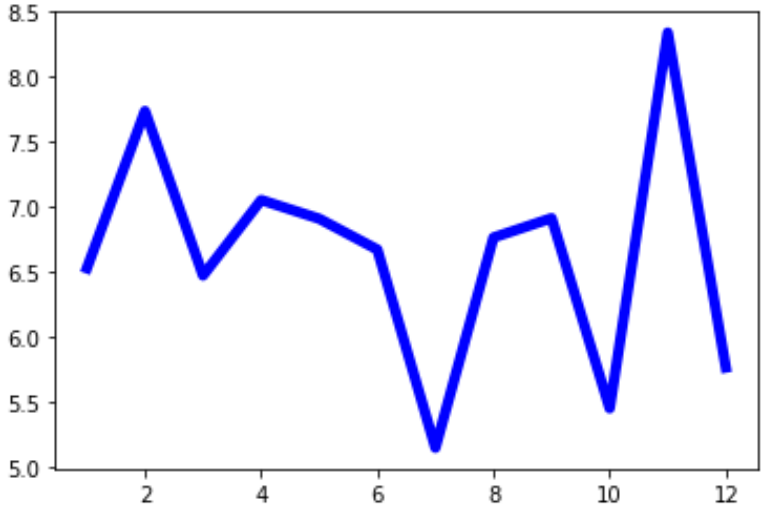
선 스타일을 다양하게 지정할 수 있음
그 외의 여러가지 파라미터를 이용해 꾸밀 수 있음

| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| Alpha | 투명도 조절 | alpha = 0.2 |
| Color | 색 지정 | color = 'r' |
| Label | 라인의 라벨 지정 | label='첫번째 라인 ' |
| Marker | 마커스타일 지정 | marker = 'o' |
| Linestyle | 선스타일 지정 | linestyle = '.' |
| Linewidth | 선의 두께 | linewidth = 2 |

```
In [10]: x = df['방영월']
         y = df['첫회시청률']

         plt.plot(x,y,color='r',marker='o',linestyle='--')
         plt.show()
```




```
In [11]: x = df['방영월']
         y = df['첫회시청률']

         plt.plot(x,y,color='blue',linewidth='5')
         plt.show()
```



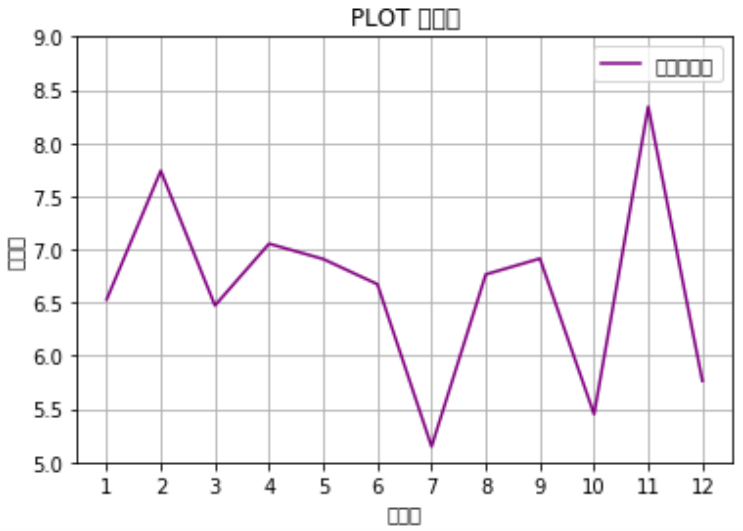

D&A

## 02 Matplotlib - plot(실선 그래프)

### 한글폰트가 깨지는 현상 발생!!

```
In [13]: x = df['방영월']
         y = df['첫회시청률']

         plt.plot(x,y,color='purple',label='첫회시청률')
         plt.title('PLOT 그래프')
         plt.ylim(5,9)
         plt.xticks([1,2,3,4,5,6,7,8,9,10,11,12])
         plt.xlabel('방영월')
         plt.ylabel('시청률')
         plt.grid(True)
         plt.legend(loc=0)
         plt.show()
```

```
nt font.
  font.set_text(s, 0, flags=flags)
```




### ※ 거의 모든 그래프에서 사용가능!

| 메소드 | 의미 |
|---|---|
| plt.title() | 그래프의 제목 지정 |
| plt.grid() | 그래프의 그리드 지정 |
| plt.legend() | 그래프의 범례 지정 |
| plt.xlim() | 그래프에 표시할 x범위 지정 |
| plt.ylim() | 그래프에 표시할 y범위 지정 |
| plt.xticks() | 그래프의 x축 틱을 직접 지정 |
| plt.yticks() | 그래프의 y축 틱을 직접 지정 |
| plt.xlabel() | 그래프의 x축 값의 이름 지정 |
| plt.ylabel() | 그래프의 y축 값의 이름 지정 |

### -> matplotlib에서는 한글을 지원하지 않아서 아래 코드로 해결함!

```
In [14]: plt.rc('font',family='malgun gothic')
```

### -> 마이너스 부호 또한 깨지기 때문에 아래 코드로 해결함!

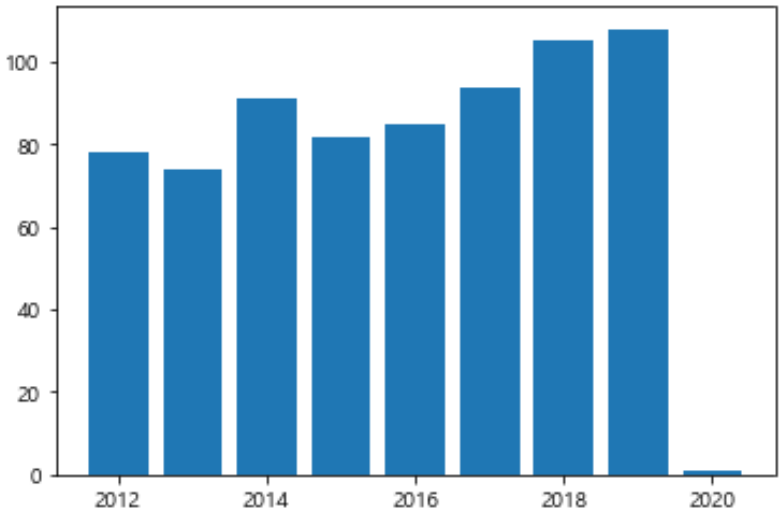
```
In [2]: plt.rc('axes', unicode_minus=False)
```

# 02 Matplotlib - bar(막대 그래프)

추세보다는 분포를 나타낼 때 유용하게 사용

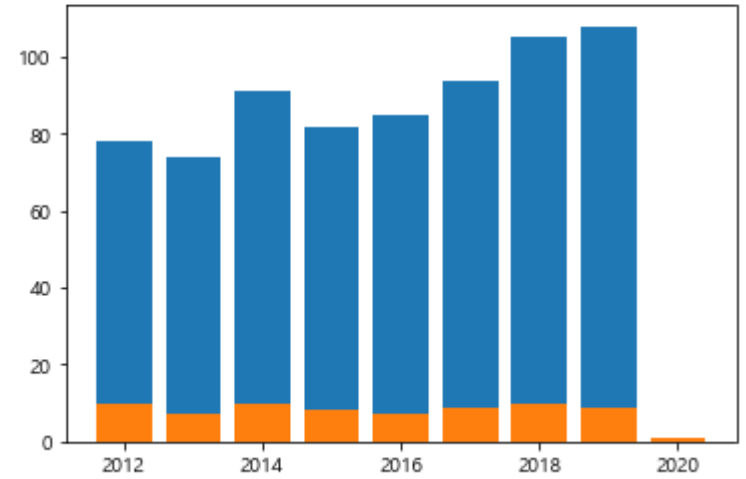
막대형태 그래프는 x값과 y값이
필수로 들어가야함

```
In [16]: x = df['방영년']
         y = df['드라마수']

         plt.bar(x,y)
         plt.show()
```




bar를 여러 개 그릴 땐 겹쳐져서 그려짐

```
In [17]: x1 = df['방영년'] ; x2 = df['방영년']
         y1 = df['드라마수'] ; y2 = df['방송사수']

         plt.bar(x1,y1)
         plt.bar(x2,y2)
         plt.show()
```

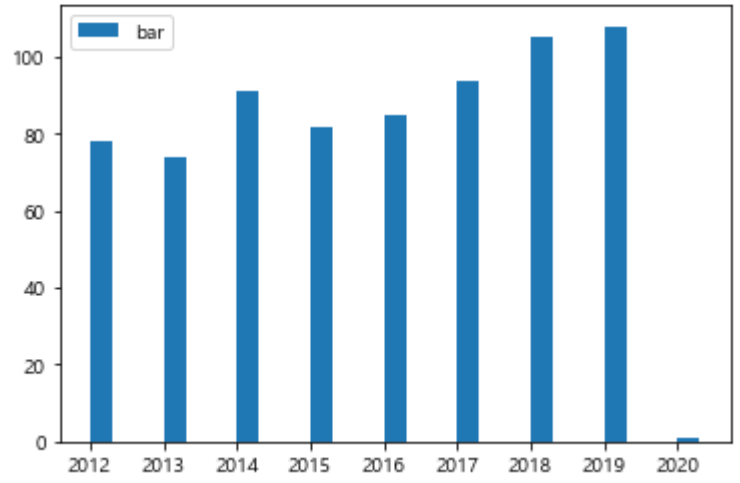



D&A

# 02 Matplotlib - bar(막대 그래프)

## plot그래프와 마찬가지로 다양한 형태로 bar그래프를 조작할 수 있음

```
In [18]: x = df['방영년']
         y = df['드라마수']

         plt.bar(x,y,align='edge',width=0.3,label='bar')
         plt.legend()
         plt.xticks([2012,2013,2014,2015,2016,2017,2018,2019,2020])
         plt.show()
```

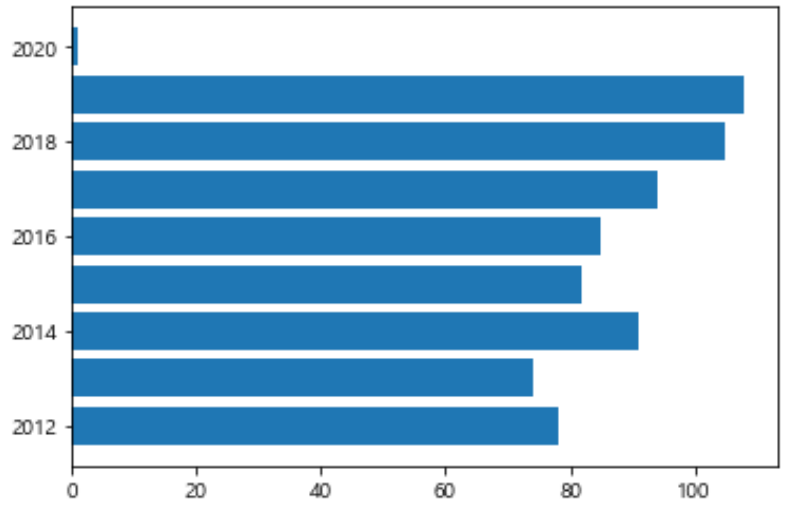



| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| alpha | 투명도 조절 | alpha = 0.2 |
| color | 색 지정 | color = 'r' |
| label | 막대의 라벨 지정 | label = '첫번째 막대 ' |
| width | 막대의 두께 | width = 2 |
| align | 축을 기준으로 막대가 그려지는 위치 | align='edge' |

# 02 Matplotlib - barh(눕힌 막대 그래프)

bar그래프를 눕히고 싶을 때에는 barh그래프를 사용
들어가는 파라미터는 bar와 거의 동일

```
In [19]: x = df['방영년']
         y = df['드라마수']

         plt.barh(x,y)
         plt.show()
```




| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| alpha | 투명도 조절 | alpha = 0.2 |
| color | 색 지정 | color = 'r' |
| label | 라인의 라벨 지정 | label = '첫번째 막대 ' |
| height | 막대의 두께 | height = 2 |
| align | 축을 기준으로 막대가 그려지는 위치 | align='edge' |

D&A

# 02 Matplotlib - bar(막대 그래프)

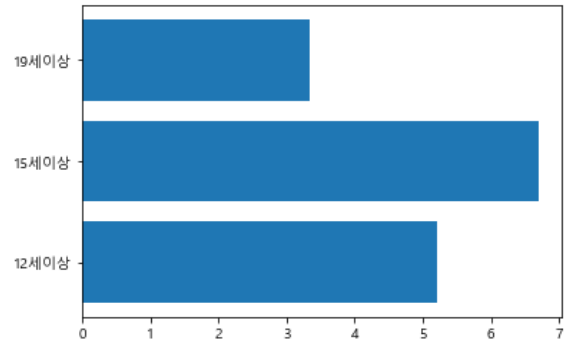
## sort_values()를 사용해 값에 따라 오름차순, 내림차순으로 정렬 가능!

```
In [38]: df = data.groupby('시청연령')['첫회시청률'].agg('mean').reset_index()
```

Out [38]:

| | 시청연령 | 첫회시청률 |
|---|---|---|
| 0 | 12세이상 | 5.220000 |
| 1 | 15세이상 | 6.705551 |
| 2 | 19세이상 | 3.334000 |

```
In [41]: x = df['시청연령']
         y = df['첫회시청률']

         plt.barh(x,y)
         plt.show()
```



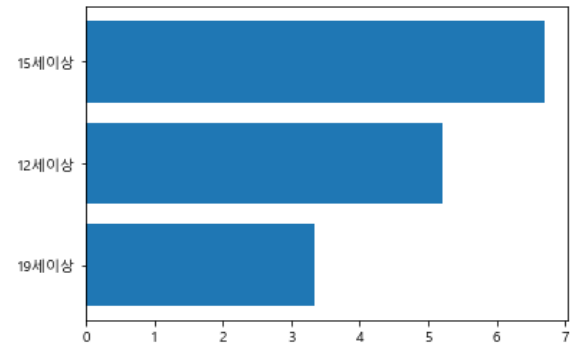

```
In [42]: df = df.sort_values('첫회시청률') ; df
```

-> 정렬할 값

Out [42]:

| | 시청연령 | 첫회시청률 |
|---|---|---|
| 2 | 19세이상 | 3.334000 |
| 0 | 12세이상 | 5.220000 |
| 1 | 15세이상 | 6.705551 |

```
In [43]: x = df['시청연령']
         y = df['첫회시청률']

         plt.barh(x,y)
         plt.show()
```



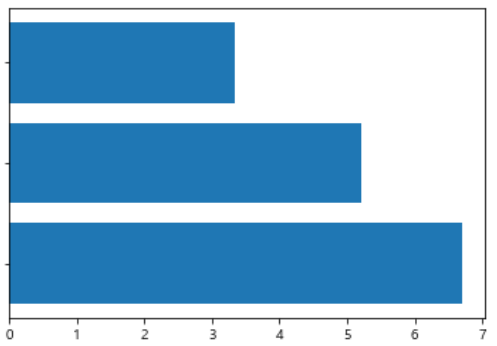

```
In [45]: df = df.sort_values('첫회시청률',ascending=False) ; df
```

True가 오름차순
(디폴트값)

Out [45]:

| | 시청연령 | 첫회시청률 |
|---|---|---|
| 1 | 15세이상 | 6.705551 |
| 0 | 12세이상 | 5.220000 |
| 2 | 19세이상 | 3.334000 |

```
In [46]: x = df['시청연령']
         y = df['첫회시청률']

         plt.barh(x,y)
         plt.show()
```



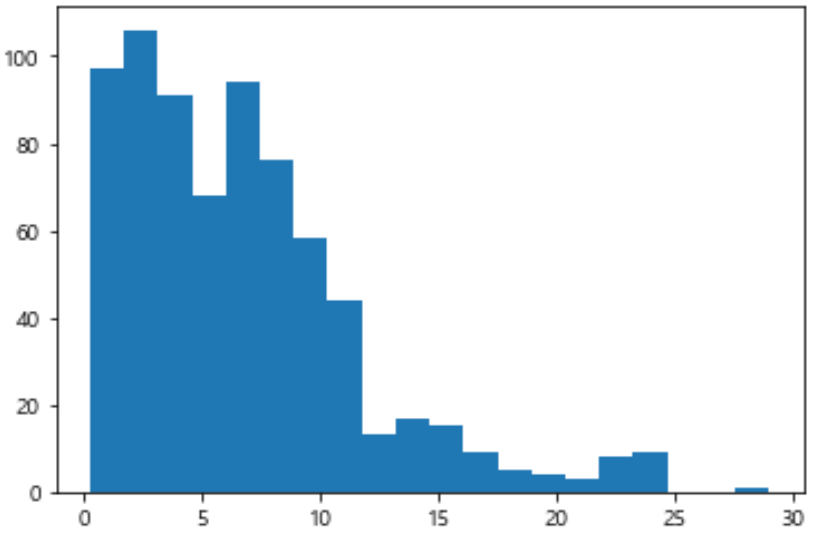

D&A

# 02 Matplotlib - histogram(히스토그램)

연속값의 분포를 나타낼 때 사용

히스토그램은 특정 값들의 빈도수를 나타내줌
파라미터 값은 비슷함

```
In [70]: plt.hist(data['첫회시청률'],bins=20)
         plt.show()
```




| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| alpha | 투명도 조절 | alpha = 0.2 |
| facecolor | 색 지정 | facecolor = 'r' |
| bins | 막대의 개수 | bins=5 |
| density | y의 값을 밀도로 변경 | density = True |
| edgecolor | 테두리 색 지정 | edgecolor='w' |

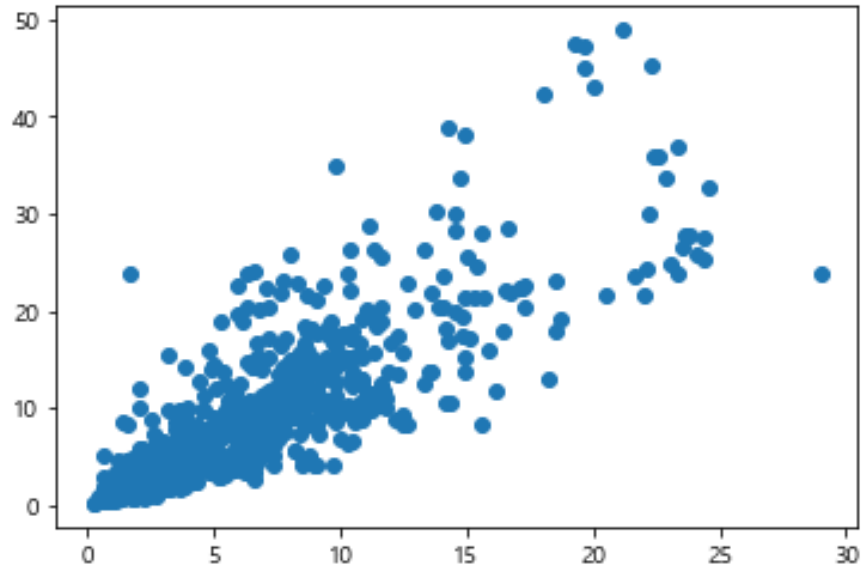
# 02 Matplotlib - scatter(산점도)

x와 y값에 따라 어느 곳에 데이터가 분포하는지를
나타낼 때 유용함

x와 y값에 따라 점을 찍어줌
scatter도 여러가지 색이나 크기 등
형태를 지정할 수 있음

| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| alpha | 투명도 조절 | alpha = 0.2 |
| color | 색 지정 | color = 'r' |

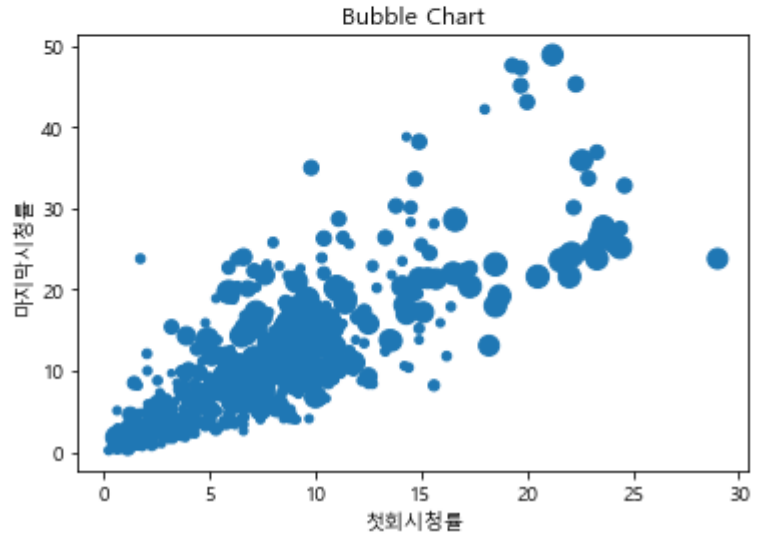
```
In [20]: x = data['첫회시청률']
         y = data['마지막시청률']

         plt.scatter(x,y)
         plt.show()
```




D&A

# 02 Matplotlib - bubble chart(버블차트)

```
In [21]: x = data['첫회시청률']
         y = data['마지막시청률']
         z = data['몇부작']

         plt.title('Bubble Chart')
         plt.scatter(x,y,s=z)
         plt.xlabel('첫회시청률')
         plt.ylabel('마지막시청률')
         plt.show()
```




**※ 버블차트란?**
**bar를 산점도로 표현한 형태**
**원의 크기로 양을 표현할 수 있어**
**3차원적인 표현이라고 할 수 있음**

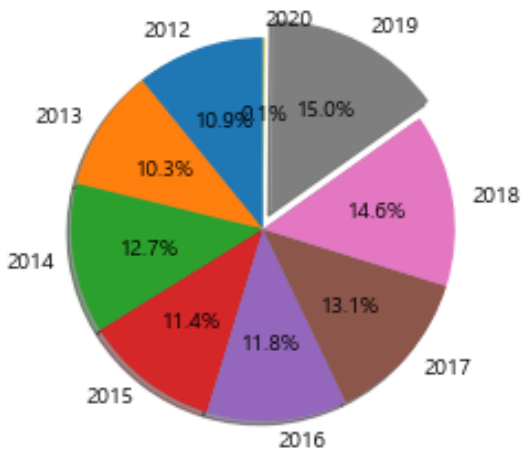
| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| alpha | 투명도 조절 | alpha = 0.2 |
| color | 색 지정 | color = 'r' |
| size(s) | 점의 크기 지정 | s = [1.3.6.2] |

D&A

# 02 Matplotlib - pie(원형 그래프)

값들을 100을 기준으로 값을 정해 한 원 안에 파이 모양으로 영역을 나눠주는 그래프
한 값이 전체 데이터에서 어느 정도의 비율을 차지하고 있는지에 대해 알기 쉬움

```
In [23]: labels = df['방영년']
         sizes = df['드라마']
         explode = [0,0,0,0,0,0,0,0.1,0]
         plt.pie(sizes, explode=explode, labels=labels,
                 autopct='%1.1f%%', shadow=True, startangle=90)
         plt.show()
```




| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| sizes | 도표의 값 | sizes=[17,20,18] |
| labels | 값의 이름 | labels=['보현', '영경','지섭'] |
| explode | 파이조각 돌출 정도 | explode=[0,0,0.1] |
| colors | 각 값의 색 | colors=['r','b','w'] |
| autopct | 값의 소수표현의 형식 지정 | autopct= '%1.1f%%' |
| shadow | 파이도표에 그림자 지정 | shadow=True |
| startangle | 가장 첫 값을 어느 각도에서부터 시작할지 지정 | startangle=0 |

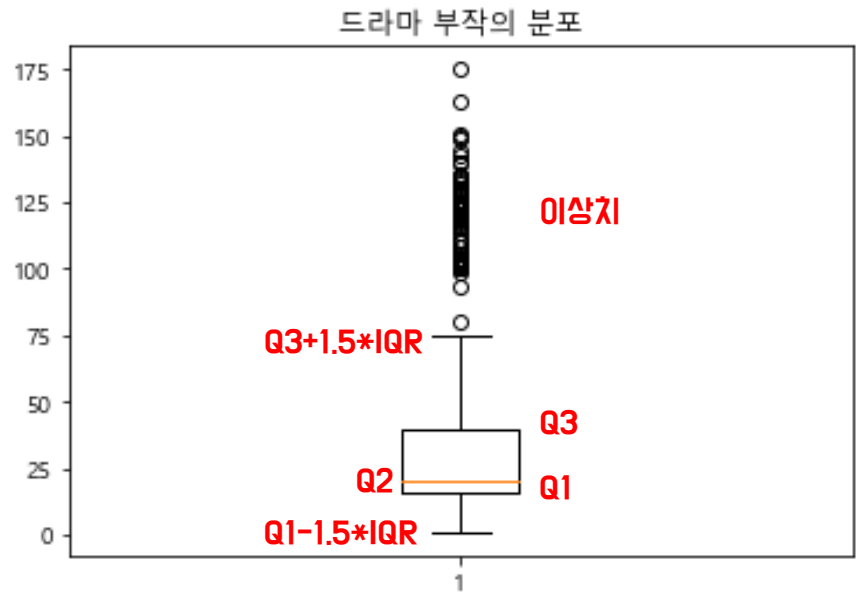
D&A

# 02 Matplotlib - boxplot(박스 도표)

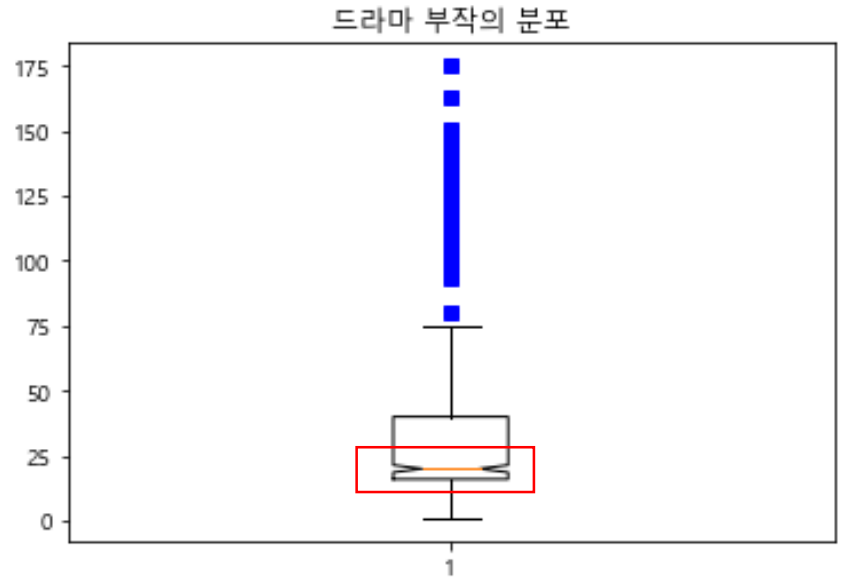
데이터의 분포에 대해 알기 쉬운 그래프
특히, 이상치의 분포를 알아보기 쉬움

| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| notch | 중앙값 부분을 v모양으로 꺾음 | notch = 1 |
| sym | 이상치 표현 모양 지정 | sym = 'bs' |

```
In [24]: plt.boxplot(data['몇부작'])
         plt.title('드라마 부작의 분포')
         plt.show()
```




```
In [25]: plt.boxplot(data['몇부작'],notch=1,sym='bs')
         plt.title('드라마 부작의 분포')
         plt.show()
```



# 이상치(Outlier) 처리의 방법?

실제 raw 데이터를 받아보면 굉장히 더러움

이상치를 처리할 때 주의사항은 결측치와 동일한 방법으로 섣불리 처리하면 안된다는 것!

처리 방법

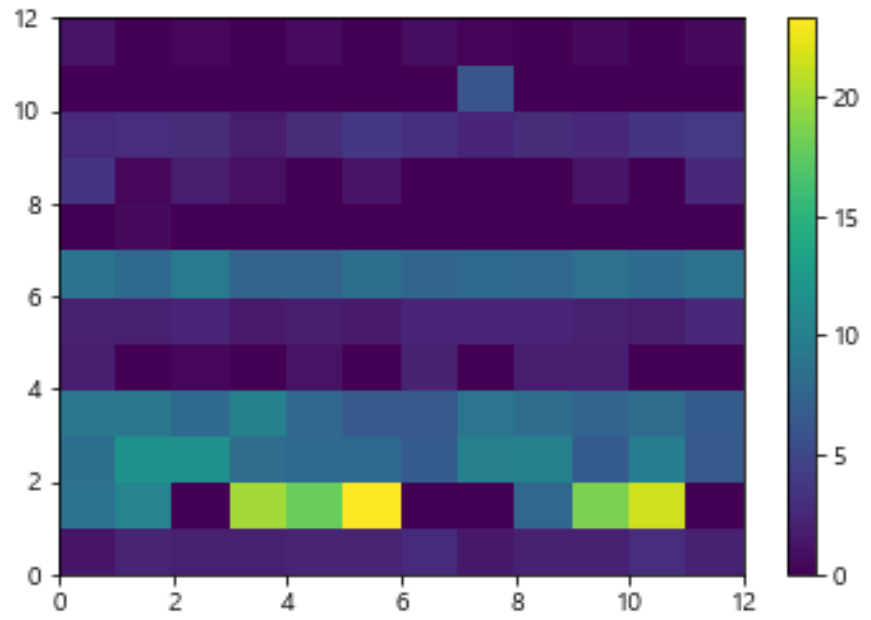
1) 하한값과 상한값 결정 후 이상치를 그 값으로 대체함
2) 제거

# 02 Matplotlib - heatmap(히트맵)

히트맵은 전체적인 데이터에서 어떤 데이터 끼리 상관관계가 있는지 알아볼 때 사용함

※ 히트맵은 seaborn 사용을 추천

```
In [27]: plt.pcolor(df)
         plt.colorbar() -> 옆의 정도를 나타내는 바
         plt.show()
```



# Seaborn이란?

시각화 라이브러리이며, matplotlib에서 x와 y라고 표현하는 것을 x와 hue라고 표현하는 것이 차이점

범례를 따로 지정하지 않아도 보여준다는 특징이 있음

sns.set() 을 실행시키면, 배경스타일이 seaborn에서 제공하는 회색 배경의 그리드를 그려줌

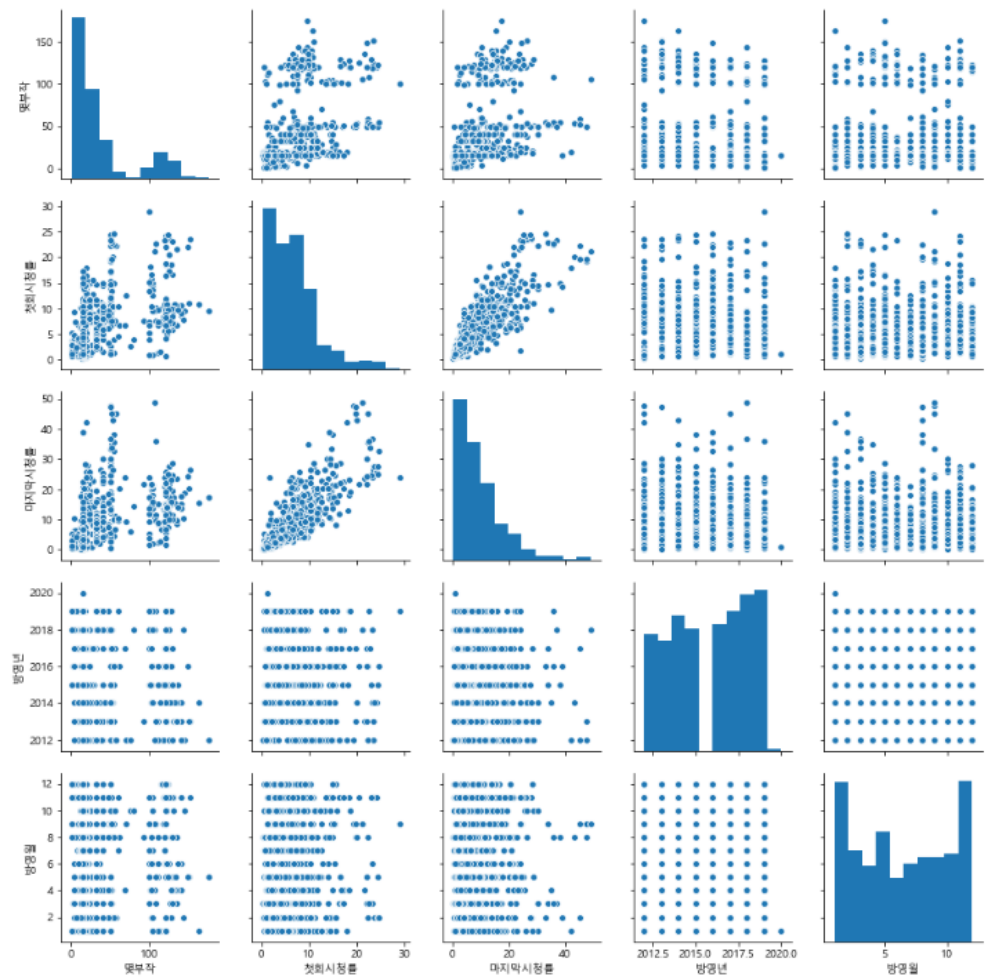
Import seaborn as sns

```
In [28]: import seaborn as sns
```

# 03 Seaborn - pairplot

## pairplot은 여러 열을 각각의 상관관계를 보여줌

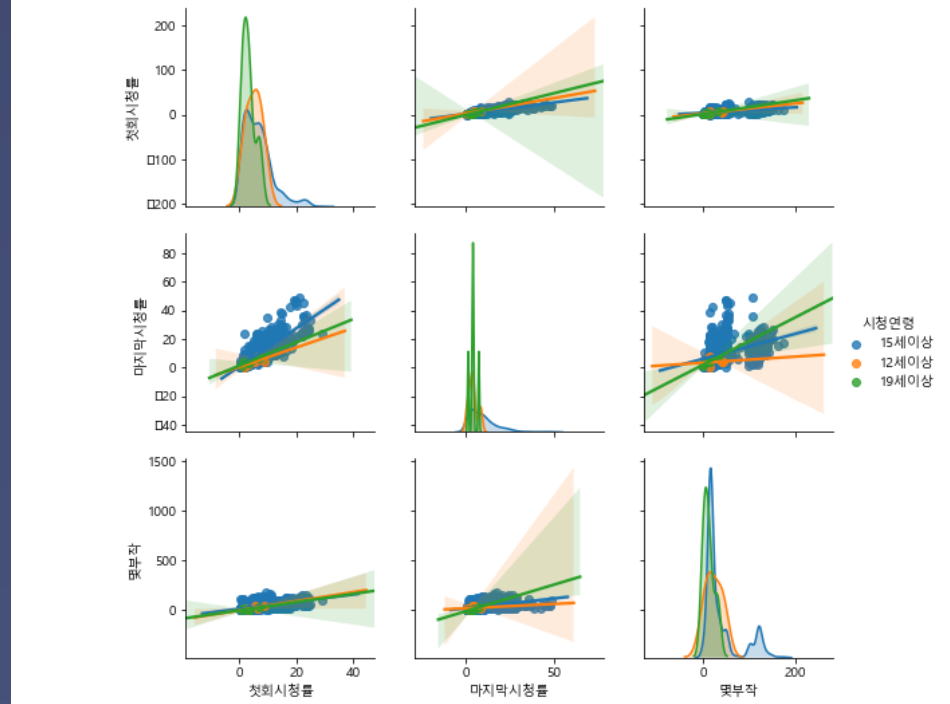
```
In [29]: sns.pairplot(data)
         plt.show()
```




| 파라미터 종류 | 의미 | 예시 |
|---|---|---|
| vars | 비교할 열 지정 | vars = ['방영월','방영년'] |
| kind | 비교하는 방법 | kind = 'reg' |
| hue | 구분할 값 지정 | hue='시청연령' |
| size | 각 그래프의 크기 지정 | size = 4 |

```
In [30]: sns.pairplot(data,vars=['첫회시청률','마지막시청률','몇부작'],hue='시청연령',kind='reg')
         plt.show()
```

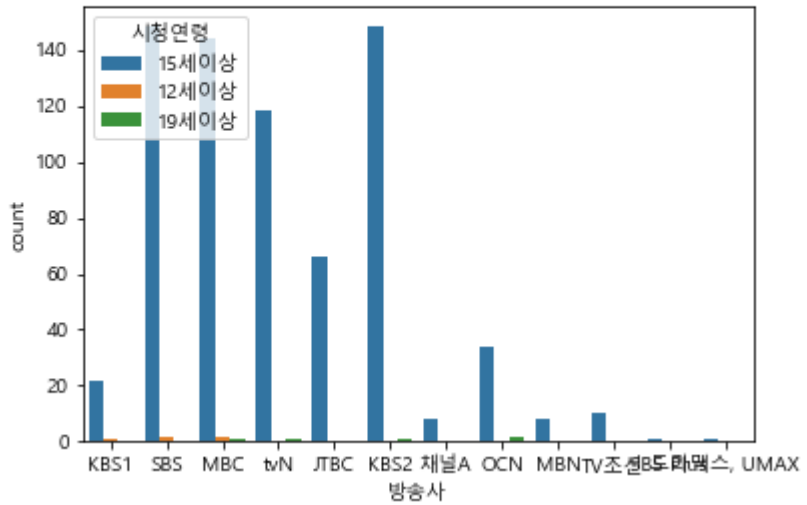
```
C:₩Users₩Bohyeon Kim₩Anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.py:211:
ont.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩Bohyeon Kim₩Anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.py:180:
ont.
  font.set_text(s, 0, flags=flags)
```



# 03 Seaborn - count plot

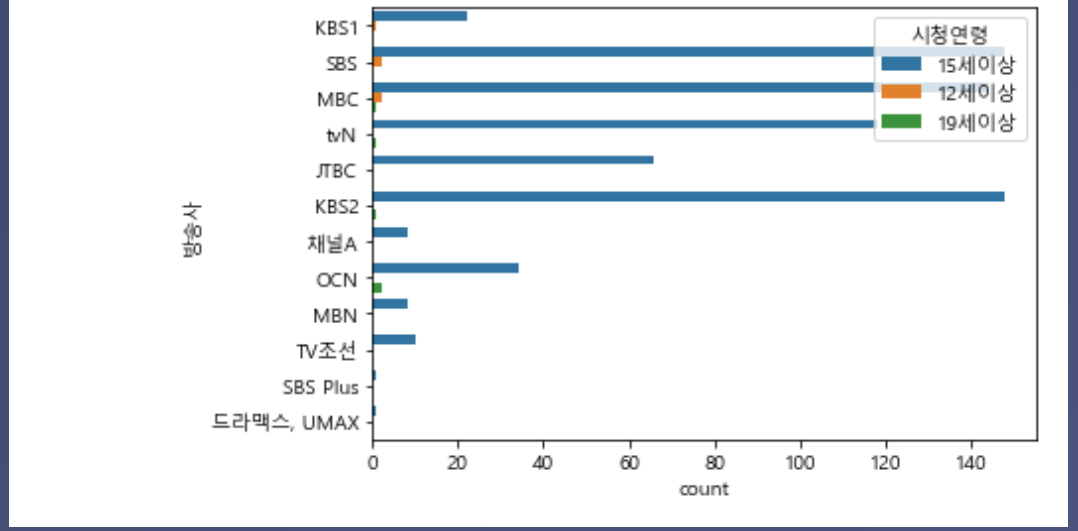
countplot은 특정 열을 값을 세어주는 막대그래프

```
In [31]: sns.countplot(data=data, x='방송사', hue='시청연령')
         plt.show()
```




가로로 눕힌 그래프를 그리려면, x를 y로 변경해면 됨

```
In [32]: sns.countplot(data=data, y='방송사', hue='시청연령')
         plt.show()
```



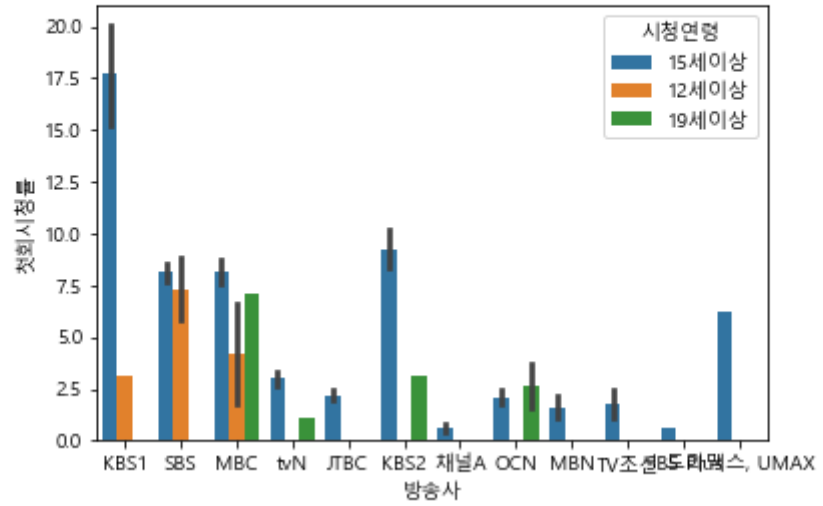

D&A

# 03 Seaborn - barplot, boxplot

## barplot에서 막대는 각각 값의 평균을 의미하고, 선들은 편차를 의미함

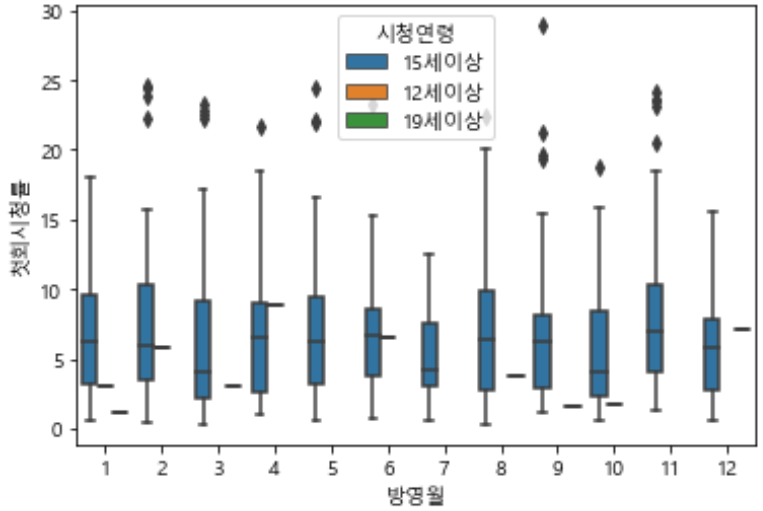
### sns.barplot(data=data, x='x값', y='y값', hue='계산할 값')

```
In [33]: sns.barplot(data=data, x='방송사', y='첫회시청률', hue='시청연령')
         plt.show()
```




### sns.boxplot(data=data, x='x값', y='y값', hue='계산할 값')

```
In [34]: sns.boxplot(x="방영월", y="첫회시청률", hue="시청연령", data=data)
         plt.show()
```

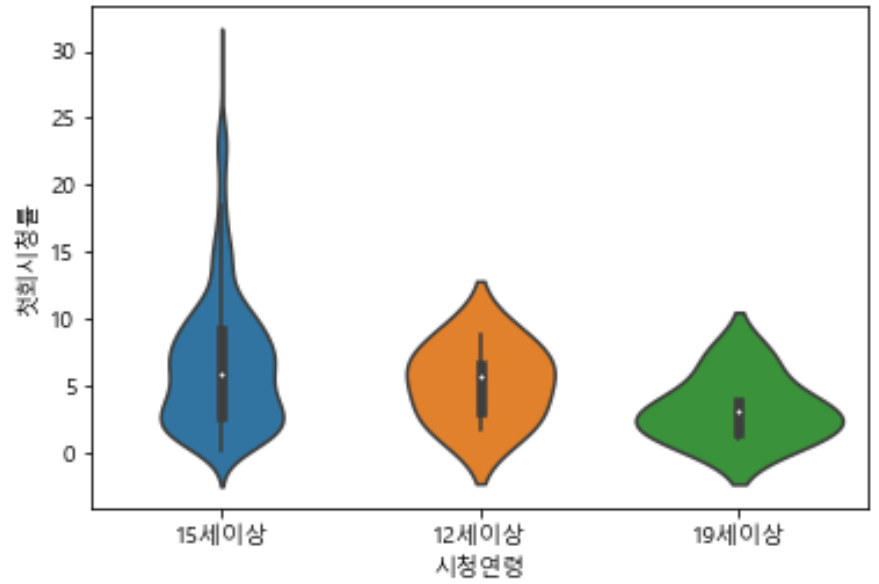



D&A

# 03 Seaborn - violin plot, heatmap

Violin plot은 각 x값에 대한 y의 분포를
나타내는 그래프
한 플랏의 양쪽 값을 다르게 설정할 수 있음

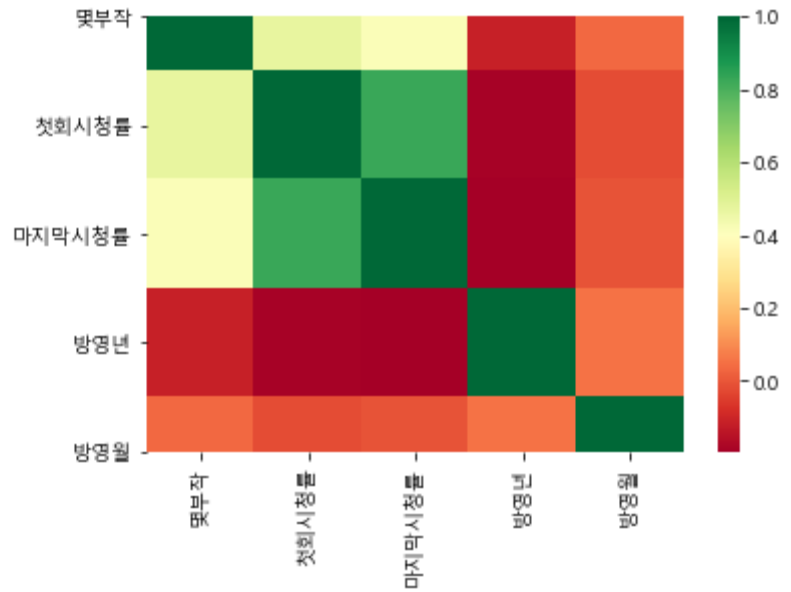
```
In [35]: sns.violinplot(x="시청연령", y='첫회시청률', data=data)
         plt.show()
```




Seaborn에서의 heatmap은
데이터의 상관계수를 주로 입력값으로 사용
annot=True를 추가하면, 정확한 색의 값이 적힘

```
In [36]: sns.heatmap(data.corr(), cmap='RdYlGn')
         plt.show()
```

-> corr()는 상관계수를 구해주는 메소드

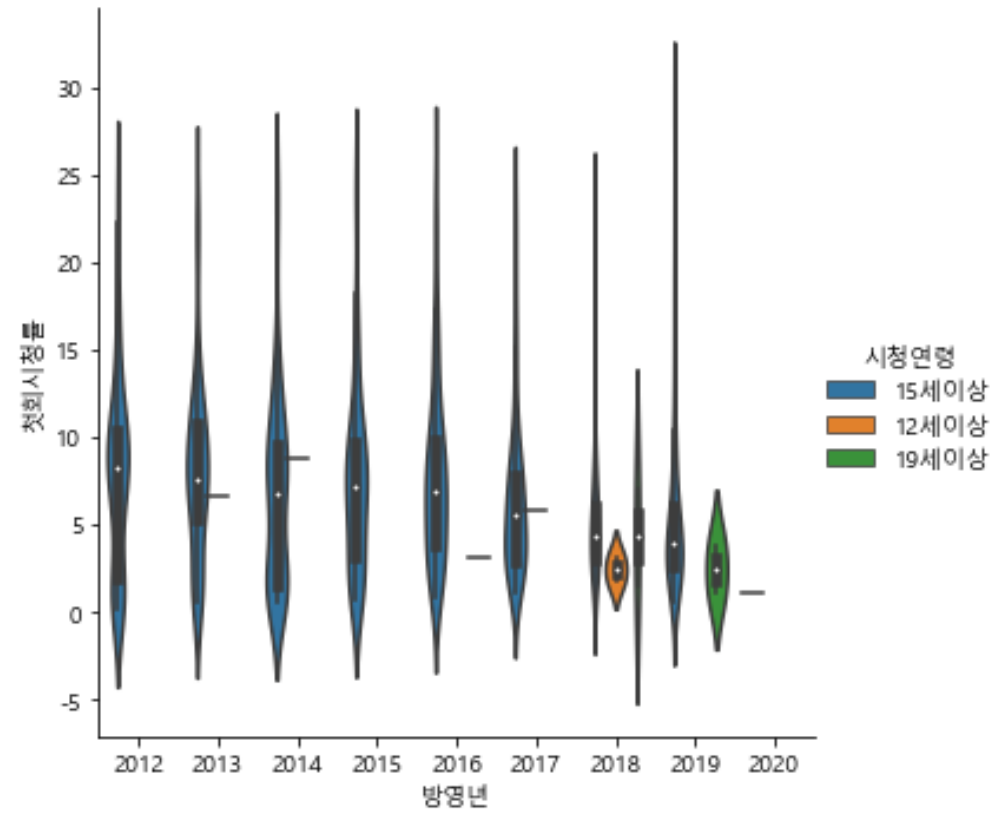



D&A

# 03 Seaborn - catplot

3개 이상의 카테고리를 대상으로 분포 변화를 파악할 수 있는 그래프
kind의 종류에는 violin, count등이 있고 기본은 산점도

```
In [87]: sns.catplot(x='방영년',y='첫회시청률',hue='시청연령',kind='violin',data=data)
         plt.show()
```



# 04 추가내용 - Subplot

그래프를 동시에 여러 개를 띄울 때 사용

그래프는 기본적으로 작은 사이즈이기 때문에 크게 키워서 자세히 볼 수 있음

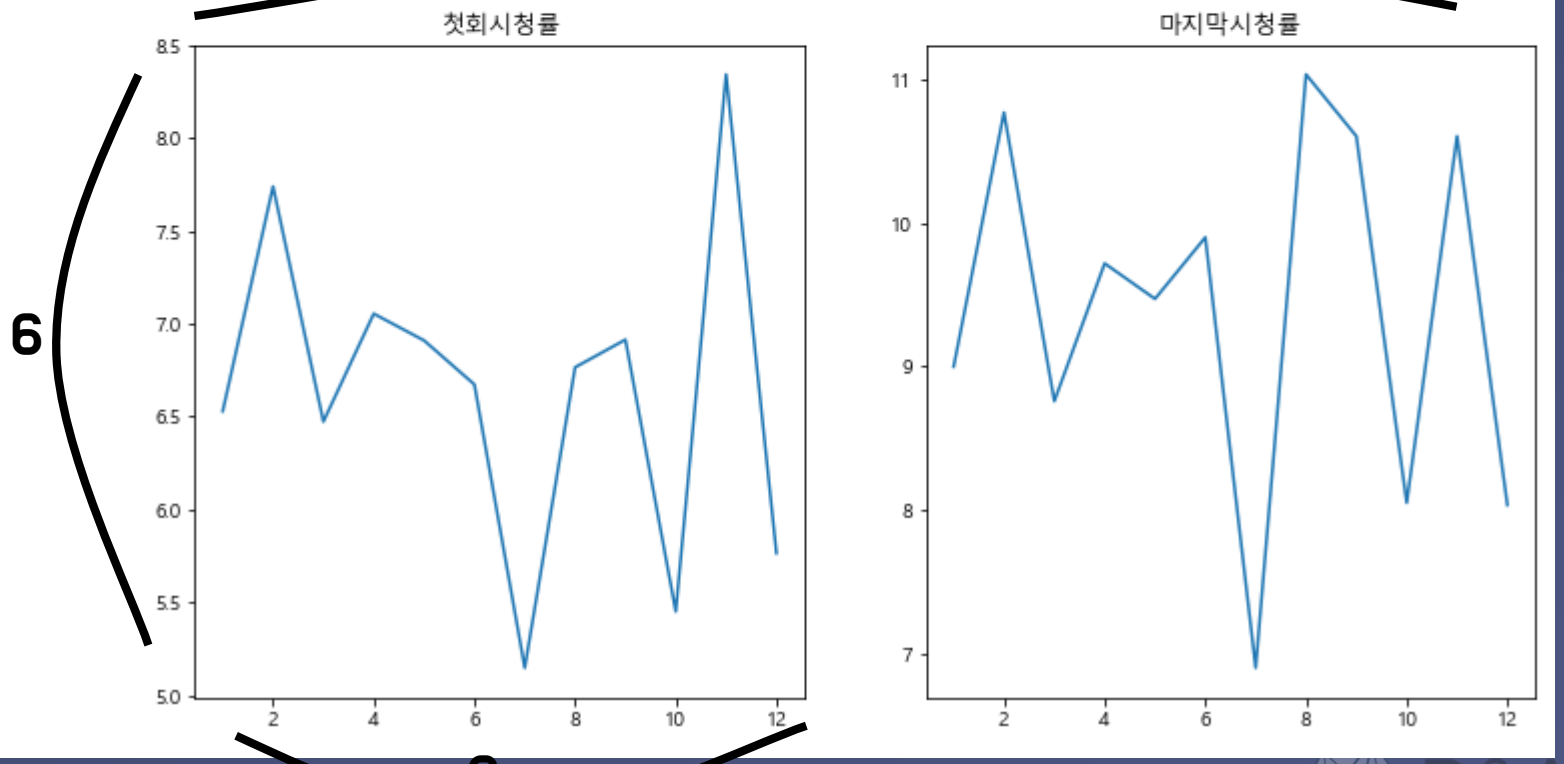
```
In [38]: x1 = df['방영월'] ; x2 = df['방영월']
         y1 = df['첫회시청률'] ; y2 = df['마지막시청률']

         plt.figure(figsize=(12,6))

         plt.subplot(1,2,1)
         plt.title('첫회시청률')
         plt.plot(x1,y1)

         plt.subplot(1,2,2)
         plt.title('마지막시청률')
         plt.plot(x2,y2)
         plt.show()
```

<- plt.figure() 그래프를 그리는 판을 만듦
figsize=(a,b) 가로 a 세로 b

<- Subplot
(행의 개수, 열의 개수, 그래프의 순서)

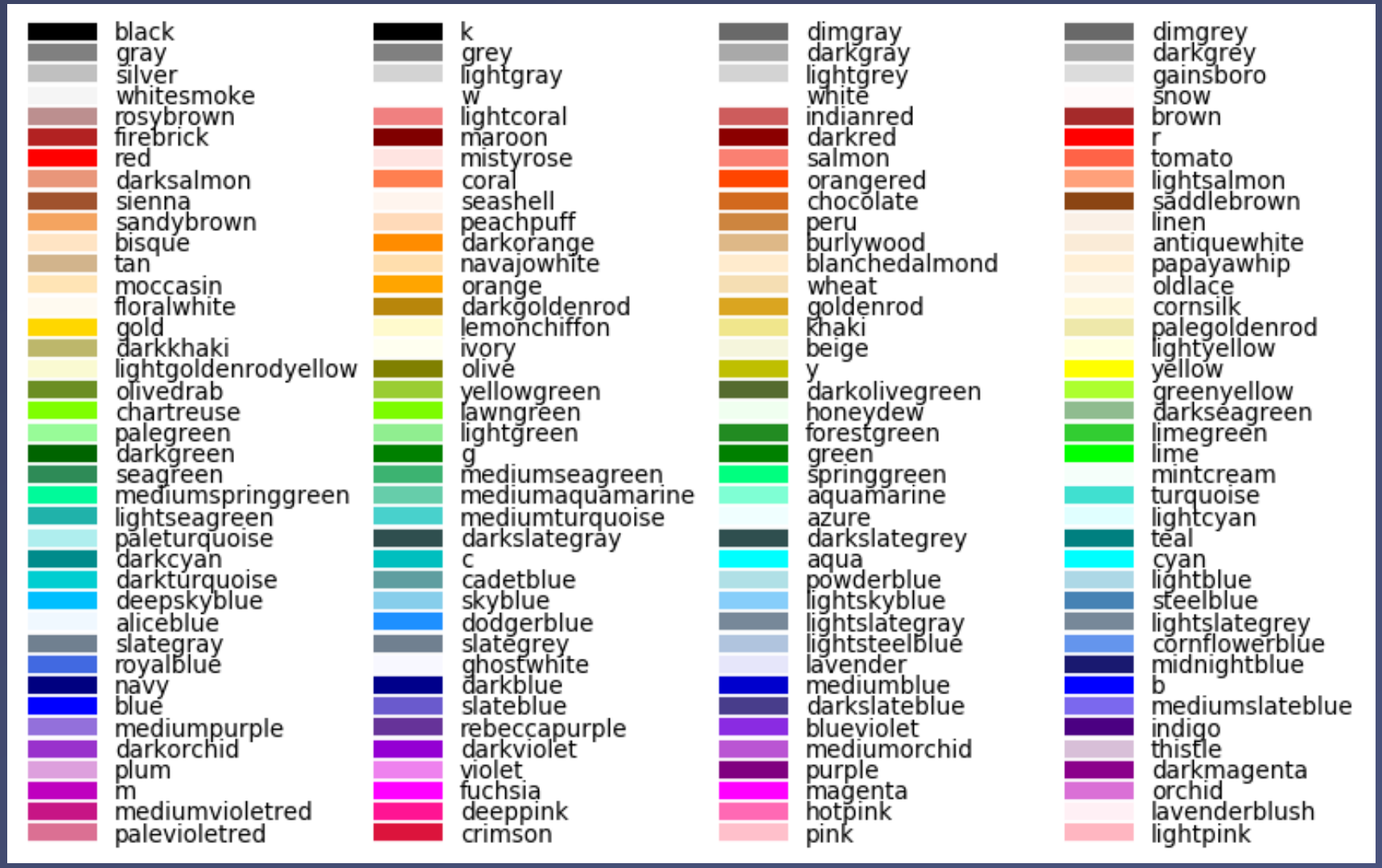
12

6




6

# 04 추가내용 - Python에서 쓰이는 color의 종류



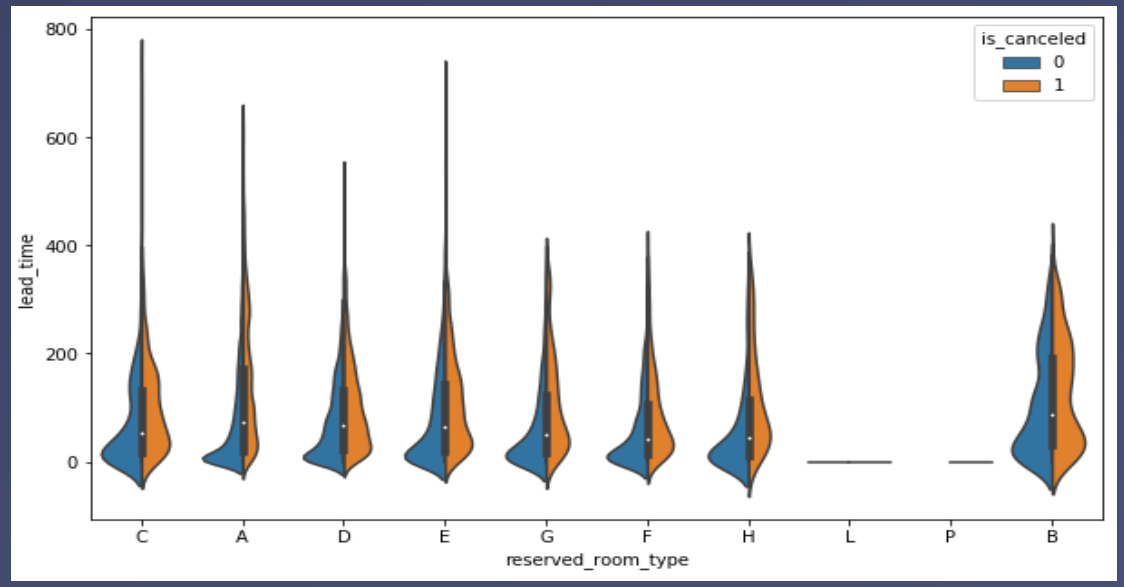

※ seaborn에서 쓰이는 pallete(여러 색 조합)도 존재함

## 과제

1. 월별로 예약건수의 추이를 나타내는 그래프를 그리시오.

2. stays in week nights, adults, babies, adr, required_car_parking_spaces, total_of_special_requests의 이상치를 그래프를 통해 찾아보고 이상처처리를 진행한 후, 처리방안에 대한 이유를 작성하시오.

3. distribution_chance의 각 범주별 booking_changes의 합계의 비율을 가장 잘 나타내는 그래프를 그리고 이유를 작성하시오.




4. 위의 그래프를 똑같이 따라서 그리고 이 그래프를 통해 알 수 있는 데이터의 특징을 생각해보시오.

5. 특정 고객이 주차공간을 원하는 지 알아보고 싶다. 이 때, 예측에 용이한 데이터 분석을 진행하시오.