(32-B) Data Visualization with Corona Data

- 2019_nCoV_data 파일 (~ 800 lines, 8 attributes) 에 있는 Data로 코로나 바이러스에 관한 infographic (인포그래픽, 정보시각화) 을 수행하려 한다
- World Wide Data (2020년 3월 10일전까지)

	Sno	Date	Province/State	Country	Last Update	Confirmed	Deaths	Recovered
0	1	01/22/2020 12:00:00	Anhui	China	01/22/2020 12:00:00	1.0	0.0	0.0
1	2	01/22/2020 12:00:00	Beijing	China	01/22/2020 12:00:00	14.0	0.0	0.0
2	3	01/22/2020 12:00:00	Chongqing	China	01/22/2020 12:00:00	6.0	0.0	0.0
3	4	01/22/2020 12:00:00	Fujian	China	01/22/2020 12:00:00	1.0	0.0	0.0
4	5	01/22/2020 12:00:00	Gansu	China	01/22/2020 12:00:00	0.0	0.0	0.0
5	6	01/22/2020 12:00:00	Guangdong	China	01/22/2020 12:00:00	26.0	0.0	0.0
6	7	01/22/2020 12:00:00	Guangxi	China	01/22/2020 12:00:00	2.0	0.0	0.0
7	8	01/22/2020 12:00:00	Guizhou	China	01/22/2020 12:00:00	1.0	0.0	0.0
8	9	01/22/2020 12:00:00	Hainan	China	01/22/2020 12:00:00	4.0	0.0	0.0
9	10	01/22/2020 12:00:00	Hebei	China	01/22/2020 12:00:00	1.0	0.0	0.0
10	11	01/22/2020 12:00:00	Heilongjiang	China	01/22/2020 12:00:00	0.0	0.0	0.0
11	12	01/22/2020 12:00:00	Henan	China	01/22/2020 12:00:00	5.0	0.0	0.0
12	13	01/22/2020 12:00:00	Hong Kong	China	01/22/2020 12:00:00	0.0	0.0	0.0
13	14	01/22/2020 12:00:00	Hubei	China	01/22/2020 12:00:00	444.0	0.0	0.0
14	15	01/22/2020 12:00:00	Hunan	China	01/22/2020 12:00:00	4.0	0.0	0.0
15	16	01/22/2020 12:00:00	Inner Mongolia	China	01/22/2020 12:00:00	0.0	0.0	0.0

- **Practice 1: 중국에서 확진자가 많은 10개 지역의 확진자 수를 정보시각화를 통해 알아보라**
- **Practice 2: 사망자가 있는 곳에서 사망/회복 비율을 정보시각화를 통해 알아보라**

Practice 1: 중국에서 확진자가 많은 10개 지역의 확진자 수는? [1/2]

참고 1: 앞장의 CSV file을 DataFrame df 로 index를 'Sno'로 읽는다.

참고 2: 중국에서 확진자 수가 많은 지역을 10개 추출한다.

```
df_china = df[df['Country']=='China']
df_china_state_confirmed = df_china.groupby('Province/State')['Confirmed'].sum()
df_china_state_confirmed_sorted = df_china_state_confirmed.sort_values(ascending=False)
df_china_top_10 = df_china_state_confirmed_sorted[:10]
df_china_top_10 = df_china_top_10.reset_index()
```

Province/State	Confirmed
Anhui	1
Beijing	14
Chongqing	6
Fujian	1
Gansu	0
Guangdong	26
Guangxi	2
Guizhou	1
Hainan	4

지역별 확진자 수 (China_State_confirmed)

Province/State	
Hubei	444.0
Guangdong	26.0
Beijing	14.0
Zhejiang	10.0
Shanghai	9.0
Chongqing	6.0
Sichuan	5.0
Henan	5.0
Tianjin	4.0
Hunan	4.0
Hainan	4.0
Guangxi	2.0
Liaoning	2.0
Jiangxi	2.0
Shandong	2.0
Fujian	1.0
Guizhou	1.0
Hebei	1.0

Practice 1: 중국에서 확진자가 많은 10개 지역의 확진자 수는? [2/2]

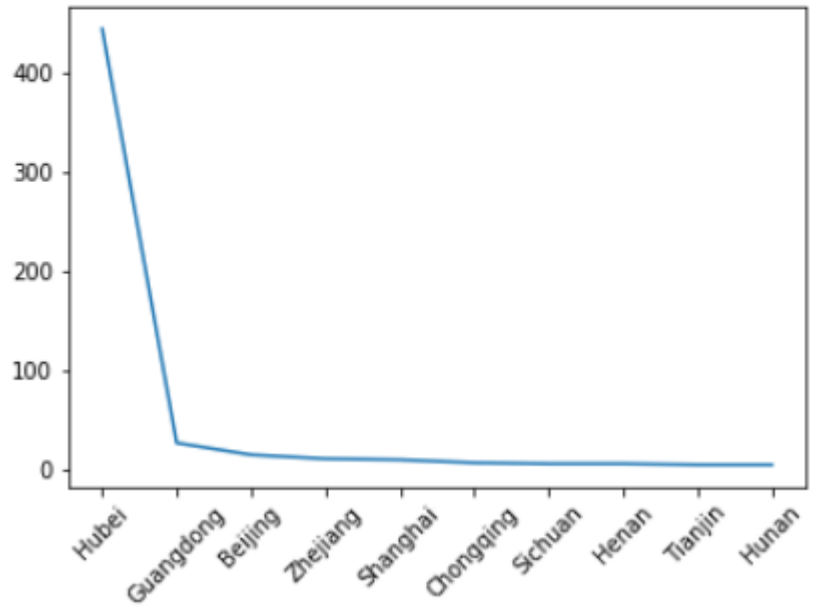
참고 3: 지역별 확진자수 top10 data로 plot를 해본다

지역별 확진자 수 (China_State_confirmed)

x	y
Province/State	
Hubei	444.0
Guangdong	26.0
Beijing	14.0
Zhejiang	10.0
Shanghai	9.0
Chongqing	6.0
Sichuan	5.0
Henan	5.0
Tianjin	4.0
Hunan	4.0
Hainan	4.0
Guangxi	2.0
Liaoning	2.0
Jiangxi	2.0
Shandong	2.0
Fujian	1.0
Guizhou	1.0
Hebei	1.0

```
x = China_top_10['Province/State']
y = China_top_10['Confirmed']
plt.plot(x, y)
plt.xticks(rotation=45)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9], <a list of 10 Text
```



위의 Plot도 한방법이지만 최대한 정보를 잘 표현하는 차트로 변경해보라! 그리고 본인이 선택한 그래프를 왜 선택했는지 이유도 쓰시오. 예를 들어 막대 그래프를 선택했다면 그 이유를 쓴다.

Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [1/6]

참고1: 사망자가 0이 아닌 지역의 데이터만 추출한다.

```
death = df[df['Deaths'] > 0].groupby('Province/State')[['Confirmed', 'Deaths', 'Recovered']].sum()
```

Death DataFrame

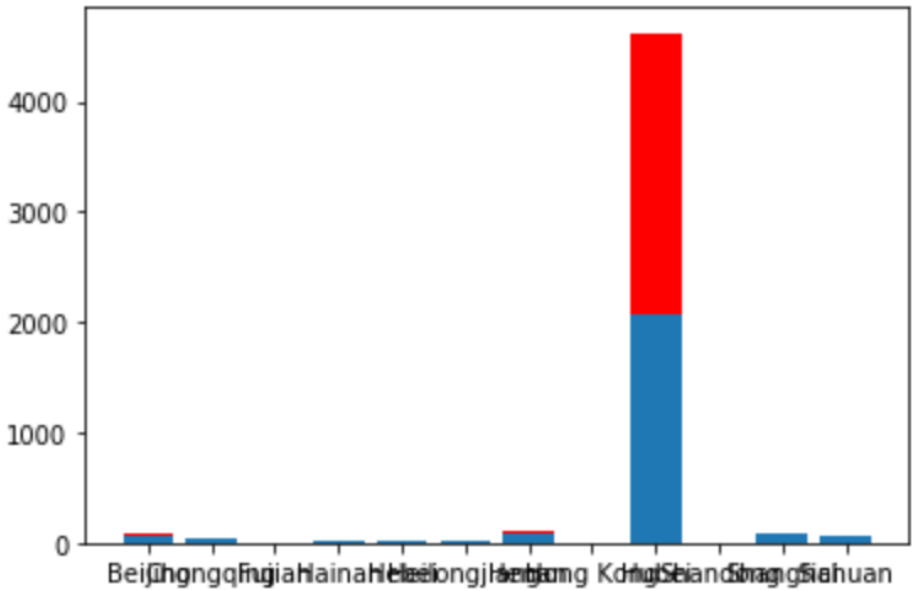
Province/State	Confirmed	Deaths	Recovered
Beijing	1356	9	73
Chongqing	1503	8	34
Fujian	18	1	0
Hainan	531	9	15
Hebei	836	13	13
Heilongjiang	788	17	10
Henan	4065	19	88
Hong Kong	18	1	0
Hubei	77732	2546	2075
Shandong	87	1	0
Shanghai	1432	11	74
Sichuan	1564	7	55

Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [2/6]

참고2: 사망자가 0이 아닌 곳의 데이터에서 사망자와 회복자를 bar chart로 그려본다

```
index = death.index
y_death = death['Deaths'].values
y_confirmed = death['Confirmed'].values
y_recover = death['Recovered'].values
plt.bar(index, y_death, color='red', bottom=y_recover)
plt.bar(index, y_recover)
```

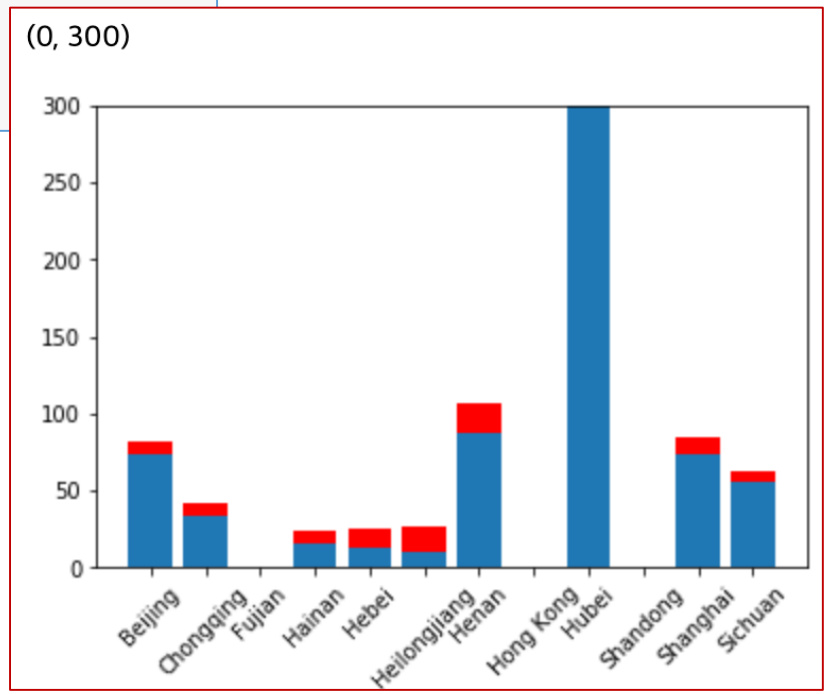
지역 이름을 X 좌표로 사용



Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [3/6]

참고3: 앞 페이지 Bar chart에서 x좌표가 잘 안보이는 문제와 데이터 양의 차이로 Bar chart가 clear하지 않은 문제를 해결한다.

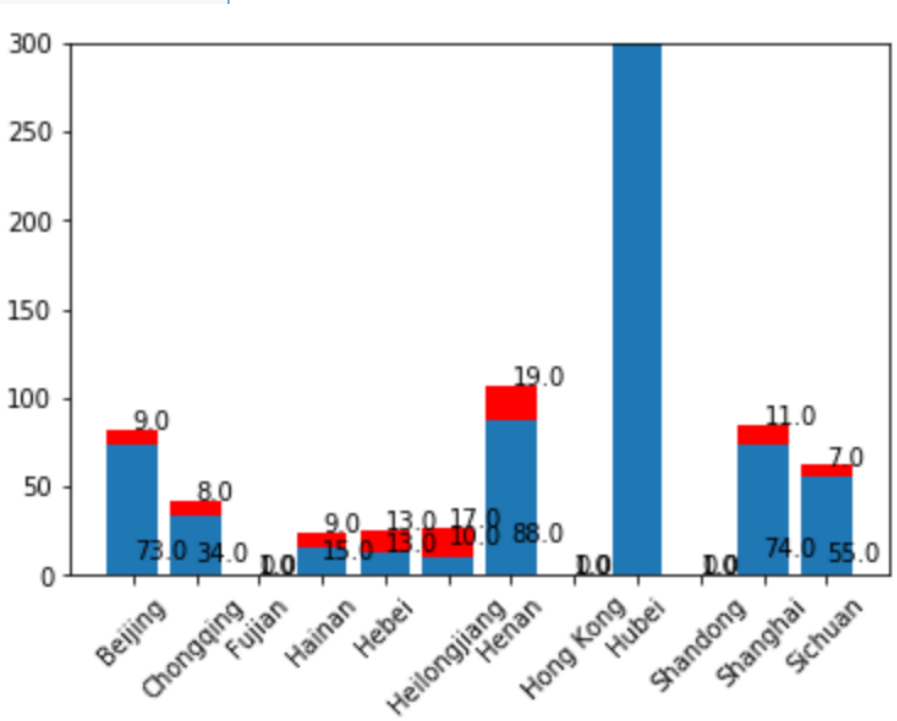
```
index = death.index
y_death = death['Deaths'].values
y_confirmed = death['Confirmed'].values
y_recover = death['Recovered'].values
plt.bar(index, y_death, color='red', bottom=y_recover)
plt.bar(index, y_recover)
plt.xticks(rotation=45)
plt.ylim((0,300))
```



Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [4/6]

참고4: 앞 페이지 Bar chart에서 데이터 양이 clear하지 않은 문제를 숫자를 붙여서 해결해 본다

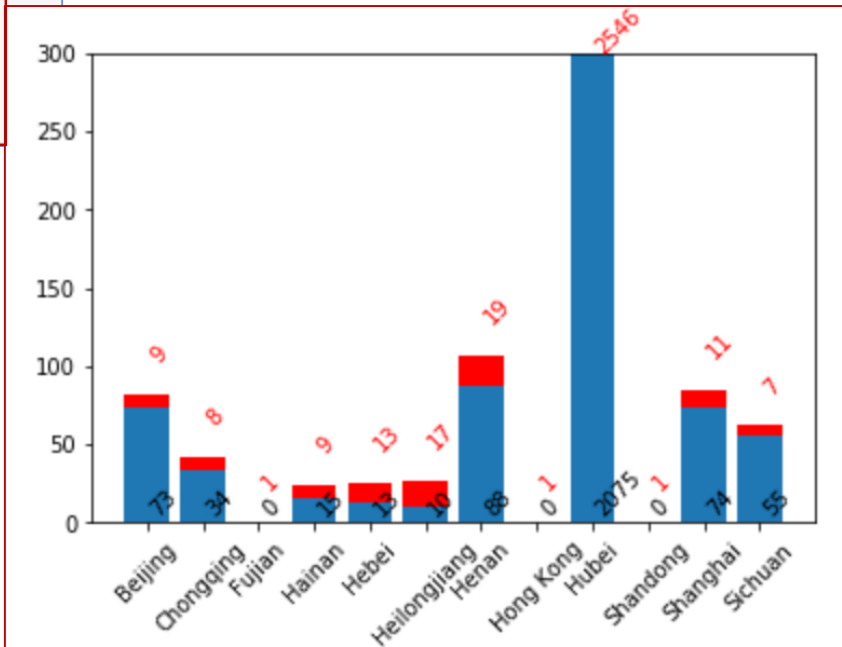
```
index = death.index
y_death = death['Deaths'].values
y_confirmed = death['Confirmed'].values
y_recover = death['Recovered'].values
plt.bar(index, y_death, color='red', bottom=y_recover)
plt.bar(index, y_recover)
plt.xticks(rotation=45)
plt.ylim((0,300))
for (a,b,c) in zip(index, y_death, y_recover):
    plt.text(a,b+c, b)
    plt.text(a,b,c)
```



Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [5/6]

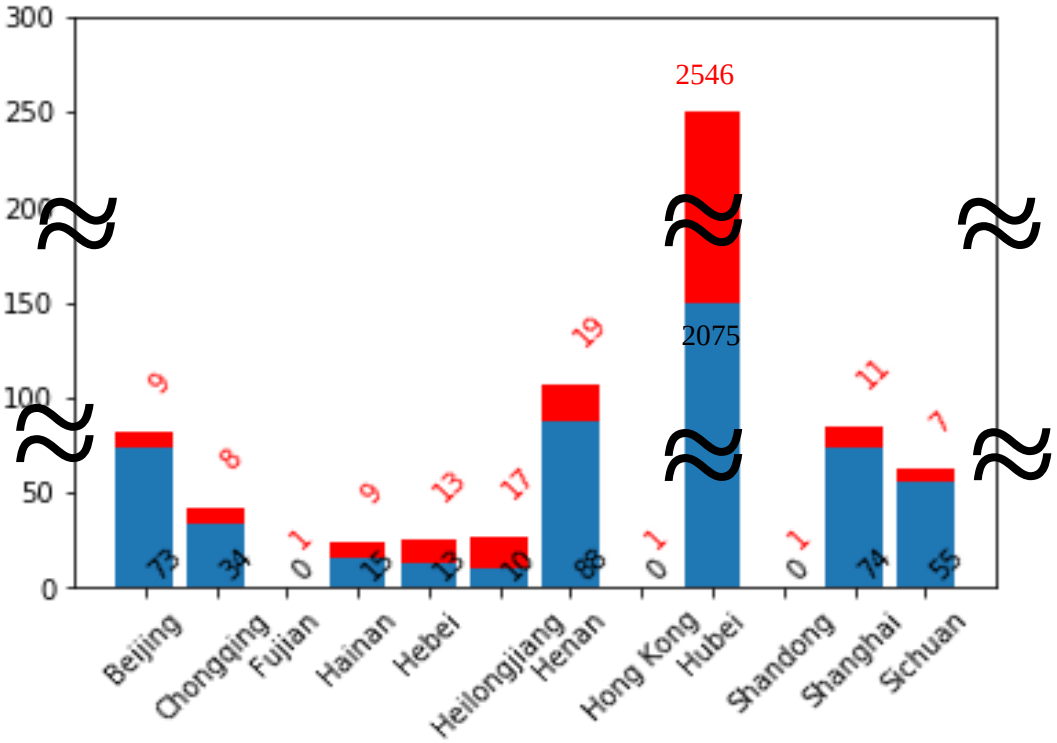
참고5: 앞 페이지 Bar chart에서 숫자들을 명확히 하는 작업을 해본다

```
index = death.index
y_death = death['Deaths'].values
y_confirmed = death['Confirmed'].values
y_recover = death['Recovered'].values
plt.bar(index, y_death, color='red', bottom=y_recover)
plt.bar(index, y_recover)
plt.xticks(rotation=45)
plt.ylim((0,300))
for (a,b,c) in zip(index, y_death, y_recover):
    if b <= 300:
        plt.text(a, b+c+20, str(int(b)), rotation=45, color='red')
    else:
        plt.text(a, 300, str(int(b)), rotation=45, color='red')
    plt.text(a, 5, str(int(c)), rotation=45)
```

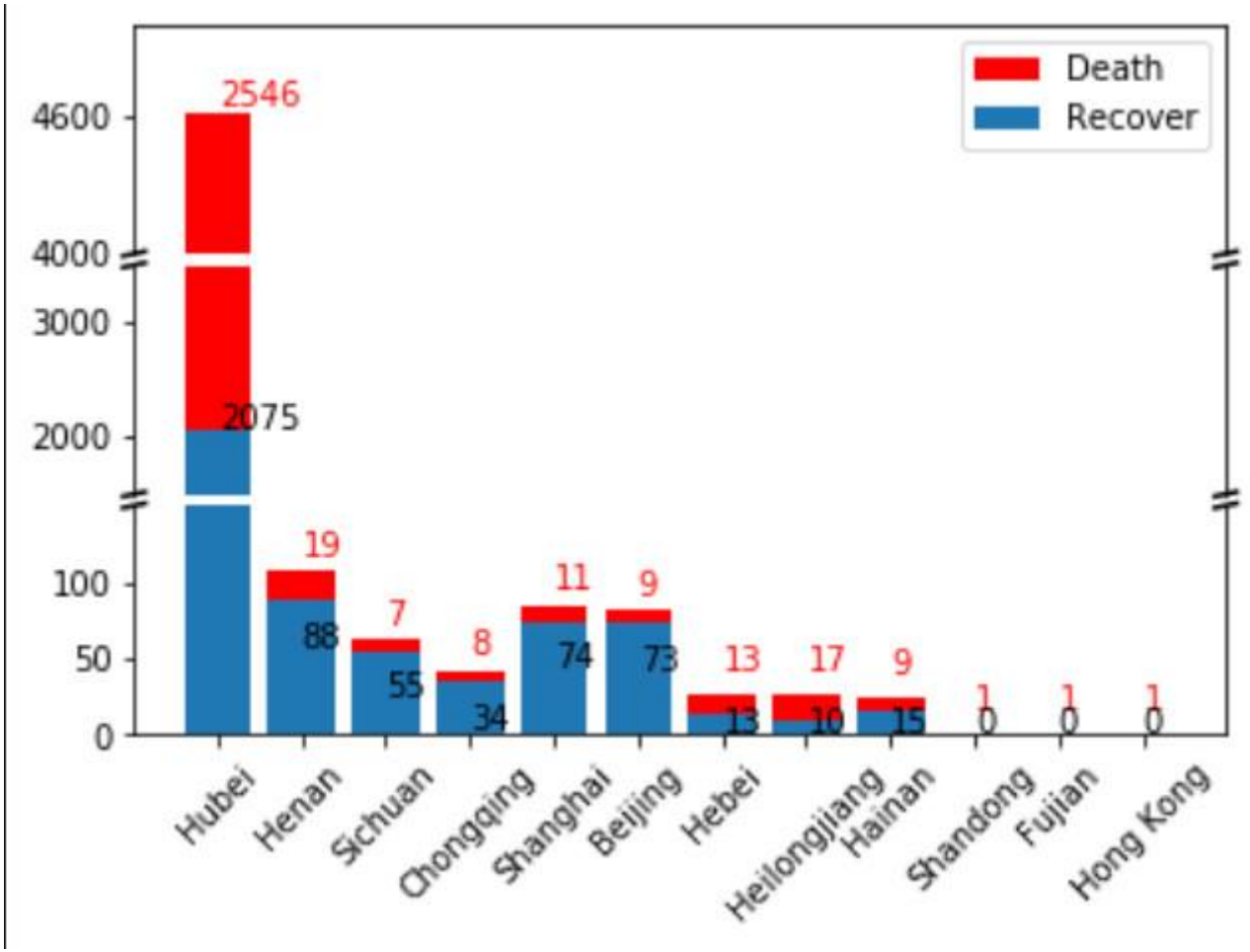


Practice 2: 사망자가 있는 곳에서 사망/회복 비율은? [6/6]

- 아래처럼 ~ 표시로 중간 생략을 한 그래프를 그리시오
- 하나의 차트로 그릴 수 없으면, 여러 개의 차트를 그리고 합쳐도 됨
  - Figure와 subplot을 이용하라
- Google search를 적극 이용하여 해결책을 검색해본다



참고 차트


Death
Recover
4600
4000
3000
2000
100
50
0
2546
2075
19
88
7
55
8
34
11
74
9
73
13
13
17
10
9
15
1
0
1
0
1
0
Hubei
Henan
Sichuan
Chongqing
Shanghai
Beijing
Hebei
Heilongjiang
Hainan
Shandong
Fujian
Hong Kong