Big Data

Matplotlib

Enjoy your data analysis

강사 김은영

스마트인재개발원
Smart Human Resources Development

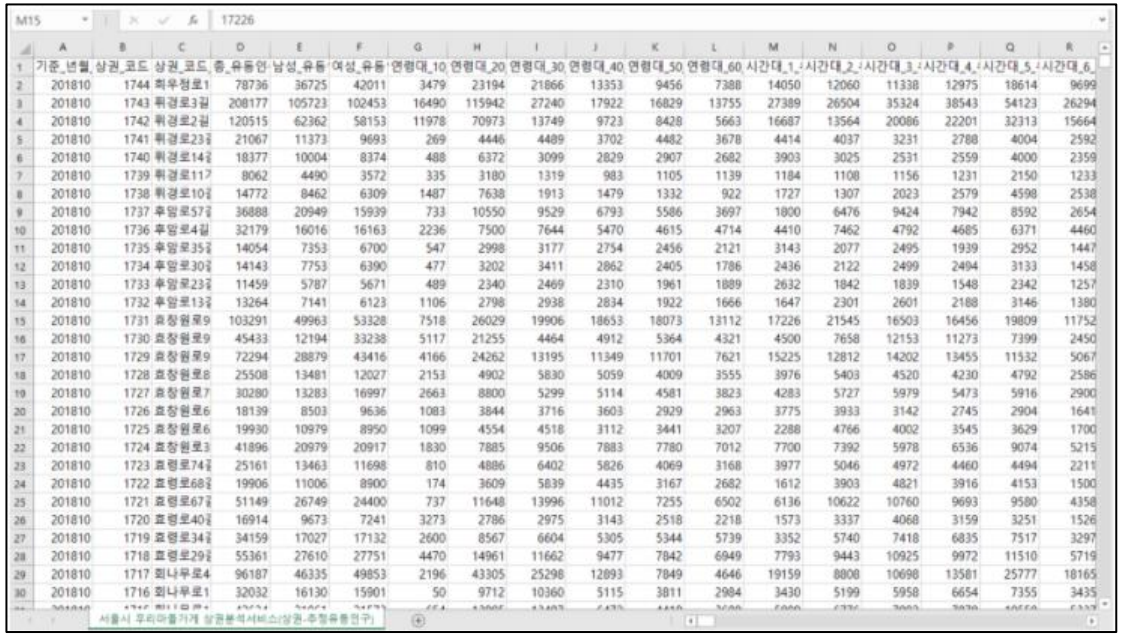
목차

데이터 시각화

- 광범위하게 분산된 방대한 양의 자료를 한눈에 볼 수 있도록 도표나 차트 등으로 정리하는 것
- 시각화를 통해 데이터의 특징을 쉽게 파악 가능
- 분석 결과를 상대방에게 효과적으로 전달 가능

Matplotlib

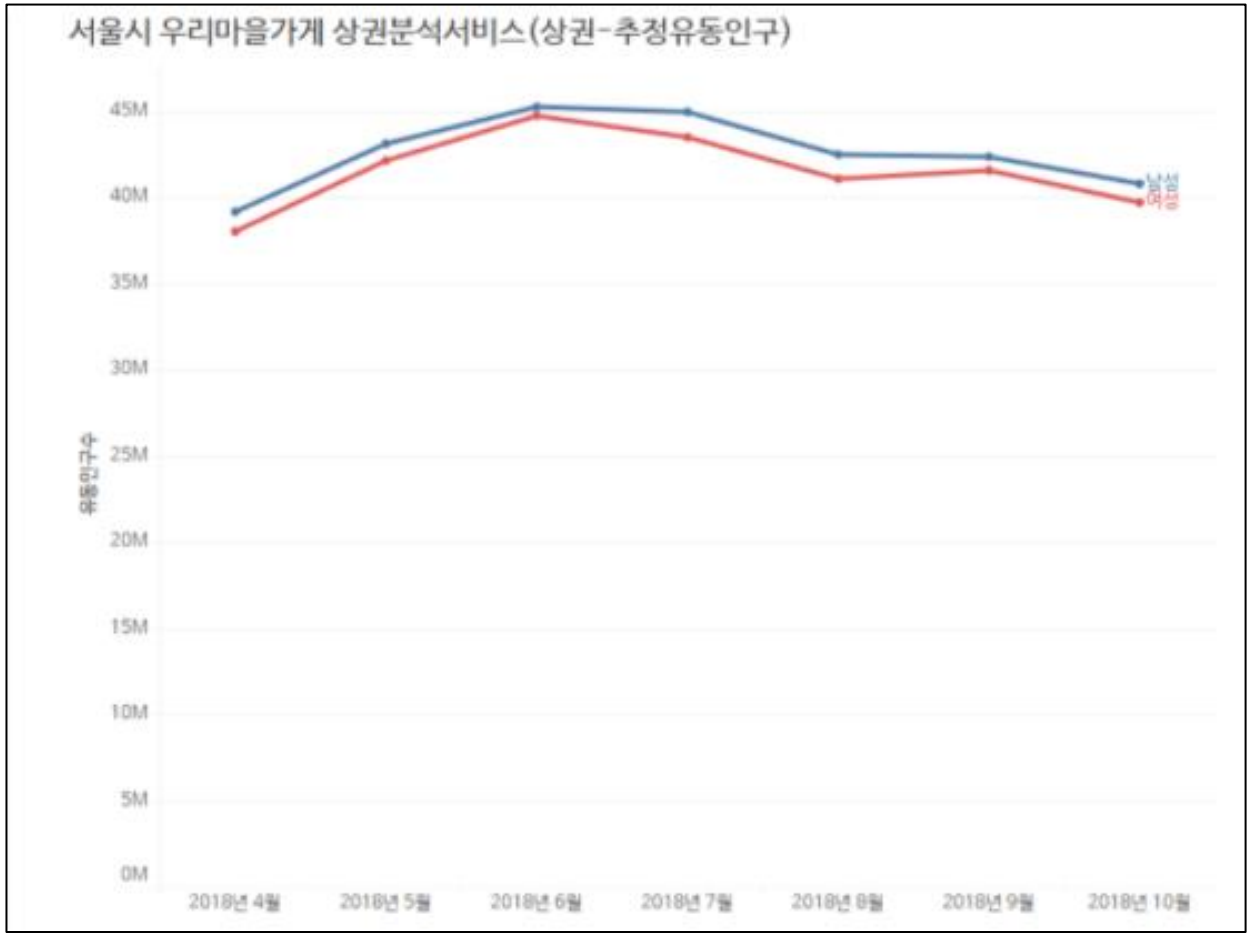
Enjoy your data analysis



M15 | 17226

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	기준_년월	상권_코드	상권_코드_	총_유동인	남성_유동	여성_유동	연령대_10	연령대_20	연령대_30	연령대_40	연령대_50	연령대_60	시간대_1	시간대_2	시간대_3	시간대_4	시간대_5	시간대_6
2	201810	1744	희우정로1	78736	36725	42011	3479	23194	21866	13353	9456	7388	14050	12060	11338	12975	18614	9699
3	201810	1743	월드컵로3길	208177	105723	102453	16490	115942	27240	17922	16829	13755	27389	26504	35324	38543	54123	26294
4	201810	1742	월드컵로2길	120515	62362	58153	11978	70973	13749	9723	8428	5663	16687	13564	20086	22201	32313	15664
5	201810	1741	월드컵로23길	21067	11373	9693	269	4446	4489	3702	4482	3678	4414	4037	3231	2788	4004	2592
6	201810	1740	월드컵로14길	18377	10004	8374	488	6372	3099	2829	2907	2682	3903	3025	2531	2559	4000	2359
7	201810	1739	월드컵로117	8062	4490	3572	335	3180	1319	983	1105	1139	1184	1108	1156	1231	2150	1233
8	201810	1738	월드컵로10길	14772	8462	6309	1487	7638	1913	1479	1332	922	1727	1307	2023	2579	4598	2538
9	201810	1737	후암로57길	36888	20949	15939	733	10550	9529	6793	5586	3697	1800	6476	9424	7942	8592	2654
10	201810	1736	후암로4길	32179	16016	16163	2236	7500	7644	5470	4615	4714	4410	7462	4792	4685	6371	4460
11	201810	1735	후암로35길	14054	7353	6700	547	2998	3177	2754	2456	2121	3143	2077	2495	1939	2952	1447
12	201810	1734	후암로30길	14143	7753	6390	477	3202	3411	2862	2405	1786	2436	2122	2499	2494	3133	1458
13	201810	1733	후암로23길	11459	5787	5671	489	2340	2469	2310	1961	1889	2632	1842	1839	1548	2342	1257
14	201810	1732	후암로13길	13264	7141	6123	1106	2798	2938	2834	1922	1666	1647	2301	2601	2188	3146	1380
15	201810	1731	효창원로9	103291	49963	53328	7518	26029	19906	18653	18073	13112	17226	21545	16503	16456	19809	11752
16	201810	1730	효창원로9	45433	12194	33238	5117	21255	4464	4912	5364	4321	4500	7658	12153	11273	7399	2450
17	201810	1729	효창원로9	72294	28879	43416	4166	24262	13195	11349	11701	7621	15225	12812	14202	13455	11532	5067
18	201810	1728	효창원로8	25508	13481	12027	2153	4902	5830	5059	4009	3555	3976	5403	4520	4230	4792	2586
19	201810	1727	효창원로7	30280	13283	16997	2663	8800	5299	5114	4581	3823	4283	5727	5979	5473	5916	2900
20	201810	1726	효창원로6	18139	8503	9636	1083	3844	3716	3603	2929	2963	3775	3933	3142	2745	2904	1641
21	201810	1725	효창원로6	19930	10979	8950	1099	4554	4518	3112	3441	3207	2288	4766	4002	3545	3629	1700
22	201810	1724	효창원로3	41896	20979	20917	1830	7885	9506	7883	7780	7012	7700	7392	5978	6536	9074	5215
23	201810	1723	효령로74길	25161	13463	11698	810	4886	6402	5826	4069	3168	3977	5046	4972	4460	4494	2211
24	201810	1722	효령로68길	19906	11006	8900	174	3609	5839	4435	3167	2682	1612	3903	4821	3916	4153	1500
25	201810	1721	효령로67길	51149	26749	24400	737	11648	13996	11012	7255	6502	6136	10622	10760	9693	9580	4358
26	201810	1720	효령로40길	16914	9673	7241	3273	2786	2975	3143	2518	2218	1573	3337	4068	3159	3251	1526
27	201810	1719	효령로34길	34159	17027	17132	2600	8567	6604	5305	5344	5739	3352	5740	7418	6835	7517	3297
28	201810	1718	효령로29길	55361	27610	27751	4470	14961	11662	9477	7842	6949	7793	9443	10925	9972	11510	5719
29	201810	1717	회나무로4	96187	46335	49853	2196	43305	25298	12893	7849	4646	19159	8808	10698	13581	25777	18165
30	201810	1716	회나무로1	32032	16130	15901	50	9712	10360	5115	3811	2984	3430	5199	5958	6654	7355	3435

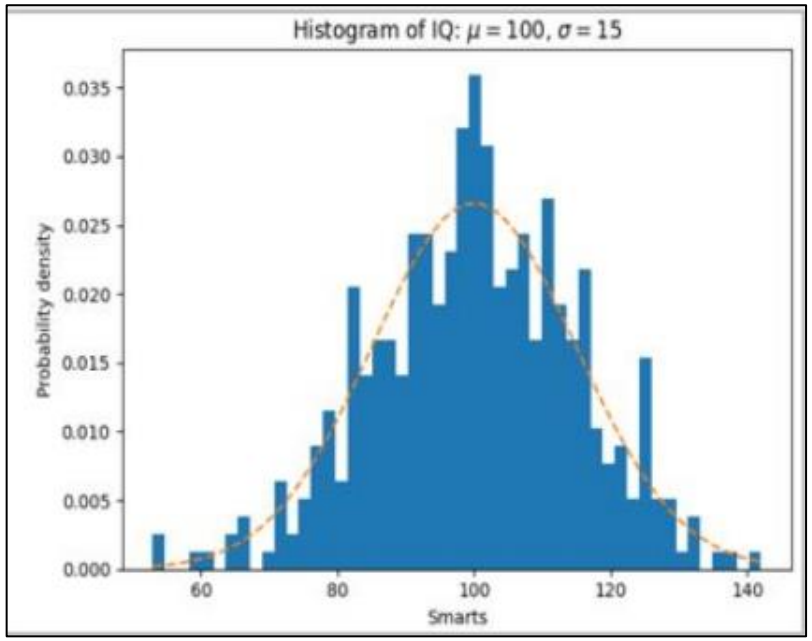
서울시 우리마을가게 상권분석서비스(상권-추정유동인구)



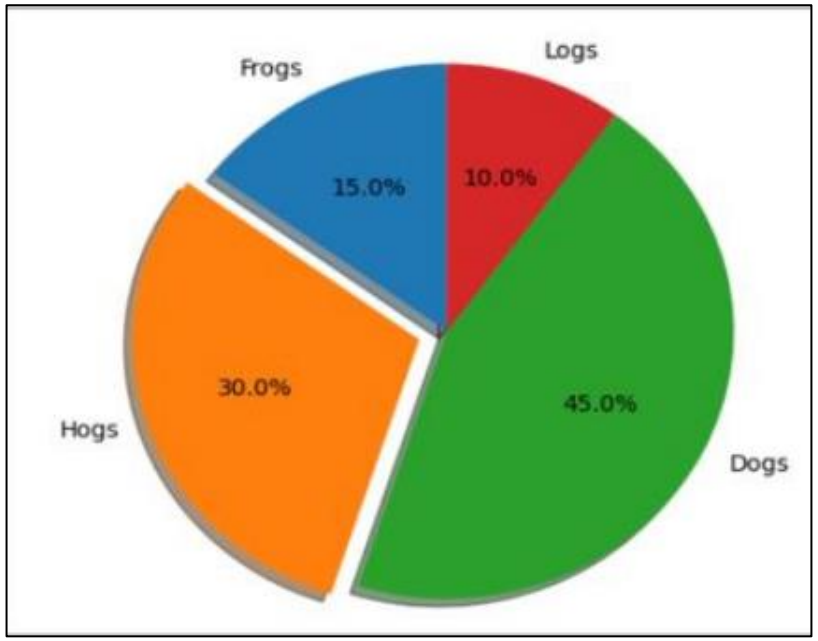
Matplotlib

Enjoy your data analysis

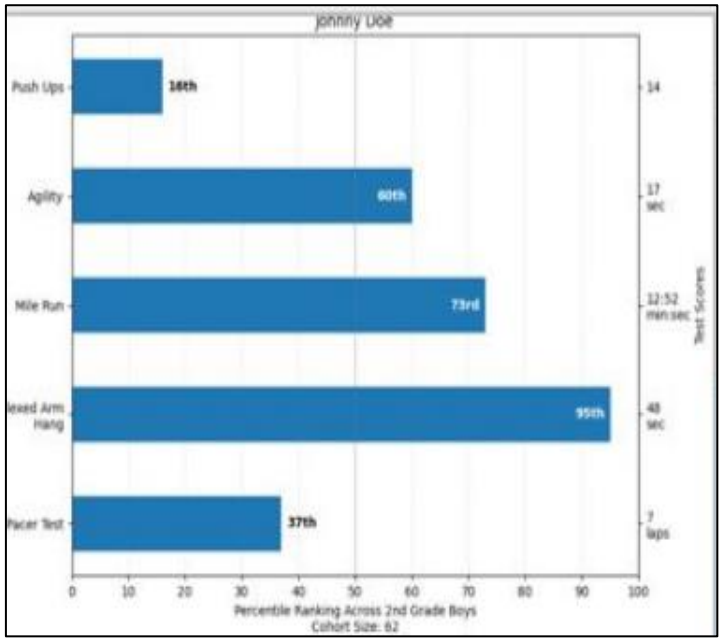
Chart 종류 - 1차원



Histogram

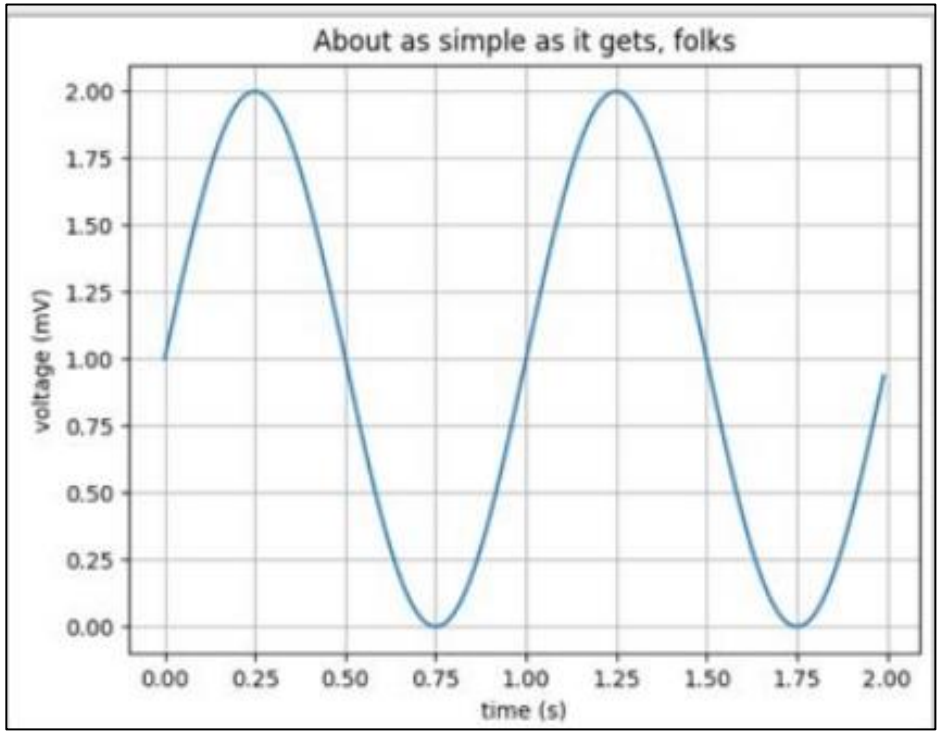


Pie Chart

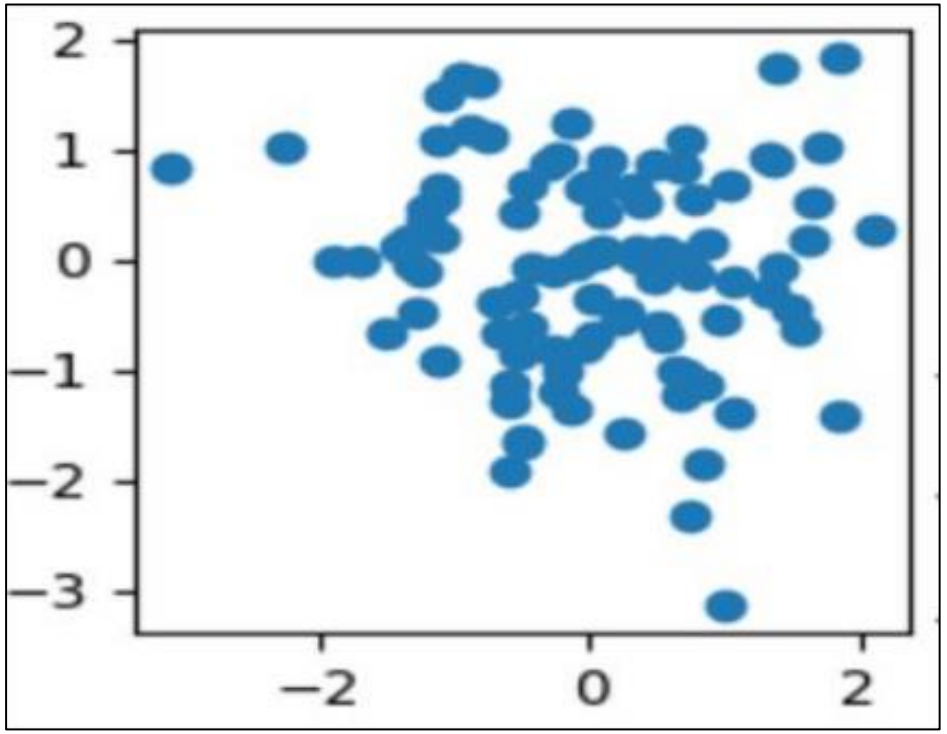


Bar Chart

Chart 종류 - 2차원



Line Plot



Scatter Plot

Matplotlib 이해

- Python에서 데이터를 관리하고 시각화 할 수 있는 대표적인 패키지
- pyplot과 pylab이란 sub패키지

	pyplot	pylab
기능	시각화	시각화 + Numpy
특징	비대화형 (간단한 정보만 입력)	대화형 (비교적 많은 정보를 입력)

1 Matplotlib 사용하기

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
```

- Matplotlib의 서브 패키지 pyplot를 import하고 앞으로 plt라는 이름으로 부름
- Seaborn 이라는 시각화 모듈을 import 하고 앞으로 sns라는 이름으로 부름

2 Line plot

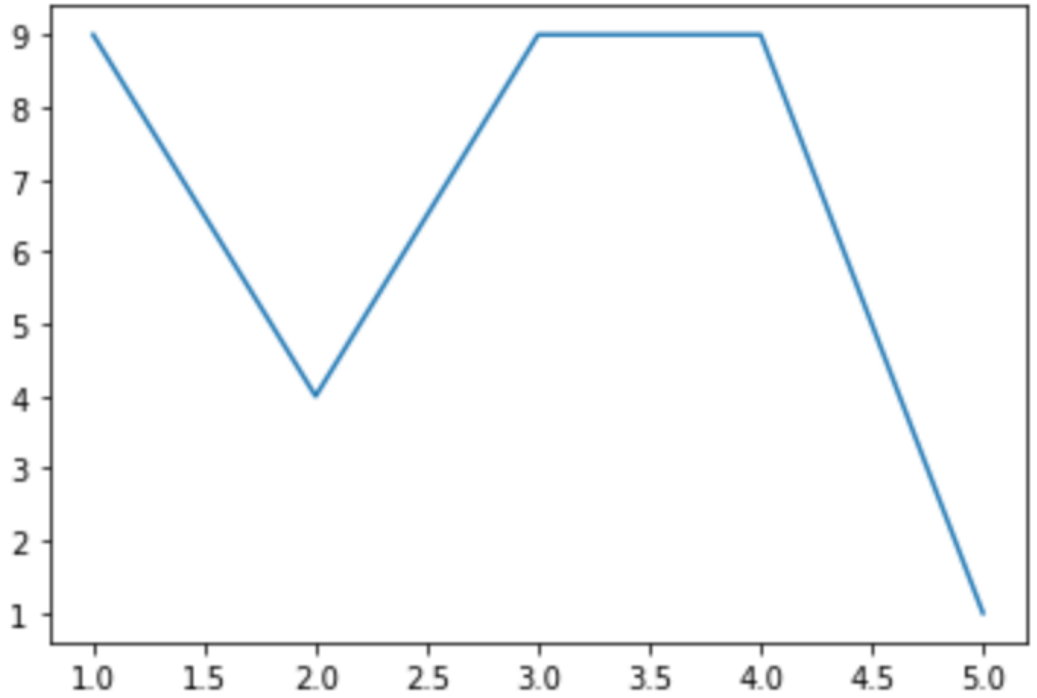
Line plot 그려보기

```
# 배열 생성하여 데이터 준비
x = np.arange(1,6)
np.random.seed(3)
y = np.random.randint(1,10,size=5)
print(x)
print(y)
```

```
[1 2 3 4 5]
[9 4 9 9 1]
```

Line plot 그려보기

```
# 선 그래프 그리기
plt.plot(x,y)
plt.show()
```



Line plot - style option

스타일 옵션 종류	마커 종류 정의	약자
color	선 색깔	c
linewidth	선 굵기	lw
linestyle	선 스타일	ls
marker	마커 종류	
markersize	마커 크기	ms
markeredgecolor	마커 선 색깔	mec
markeredgewidth	마커 선 굵기	mew
markerfacecolor	마커 내부 색깔	mfc

Line plot - line style & color

Character	Description
'-'	Solid line style
'--'	Dashed line style
'-.'	Dash-dot lint style
':'	Dotted line style

Character	Color	Character	Color
'b'	Blue	'm'	Magenta
'g'	Green	'y'	yellow
'r'	Red	'k'	Black
'c'	cyan	'w'	white

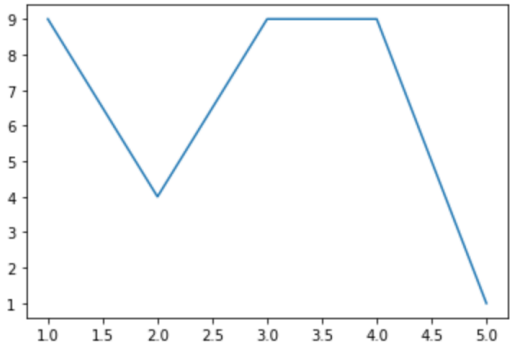
Line plot - marker

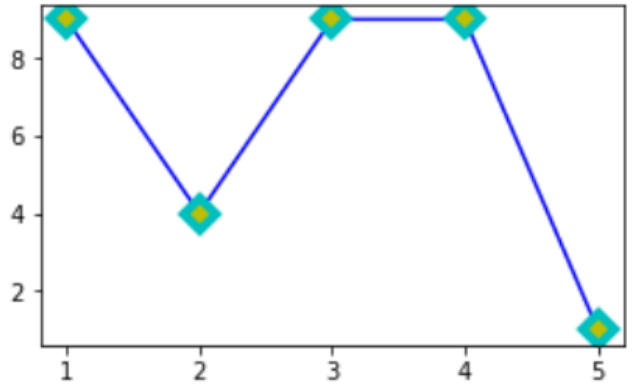
Character	Description	Character	Description
'.'	Point marker	'1'	Tri_down marker
'o'	Circle marker	'2'	Tri_up marker
'v'	Triangle_down marker	'3'	Tri_left marker
'^'	Triangle_up marker	'4'	Tri_right marker
'<'	Triangle_left marker	'*'	Star marker
'>'	Triangle_right marker	'h','H'	Hexagon1,2 marker
's'	Square marker	'+'	Plus marker
'p'	Pentagon marker	'D'	Diamond marker
'|'	V line marker	'_'	H line marker

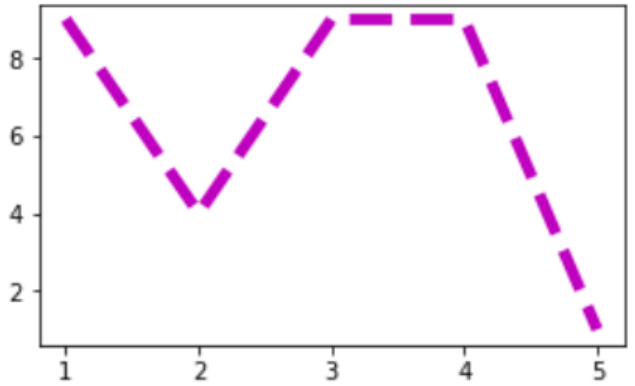
Line plot - line style 적용하기

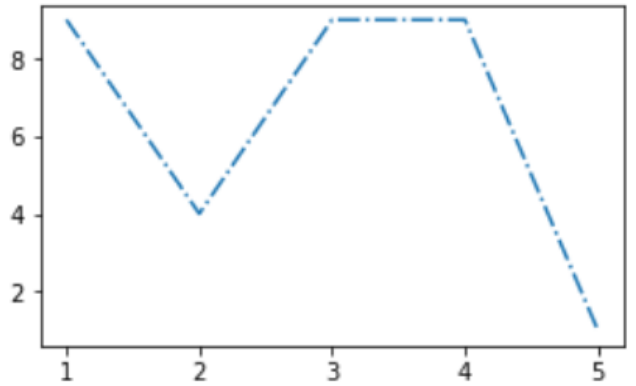
보다 명확하고 이해하기 쉬운 시각화 결과를 제공하기 위한 스타일 옵션

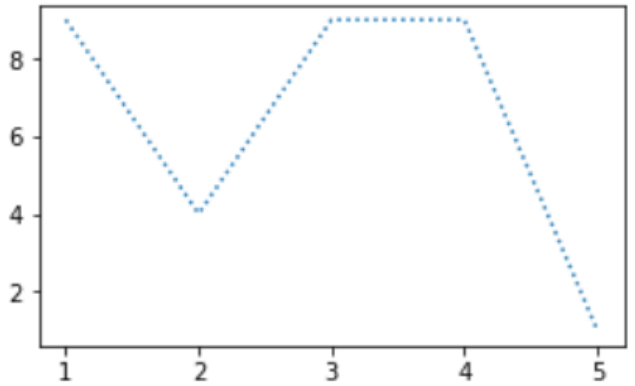
```
# 선 그래프 그리기
plt.plot(x,y)
plt.show()
```











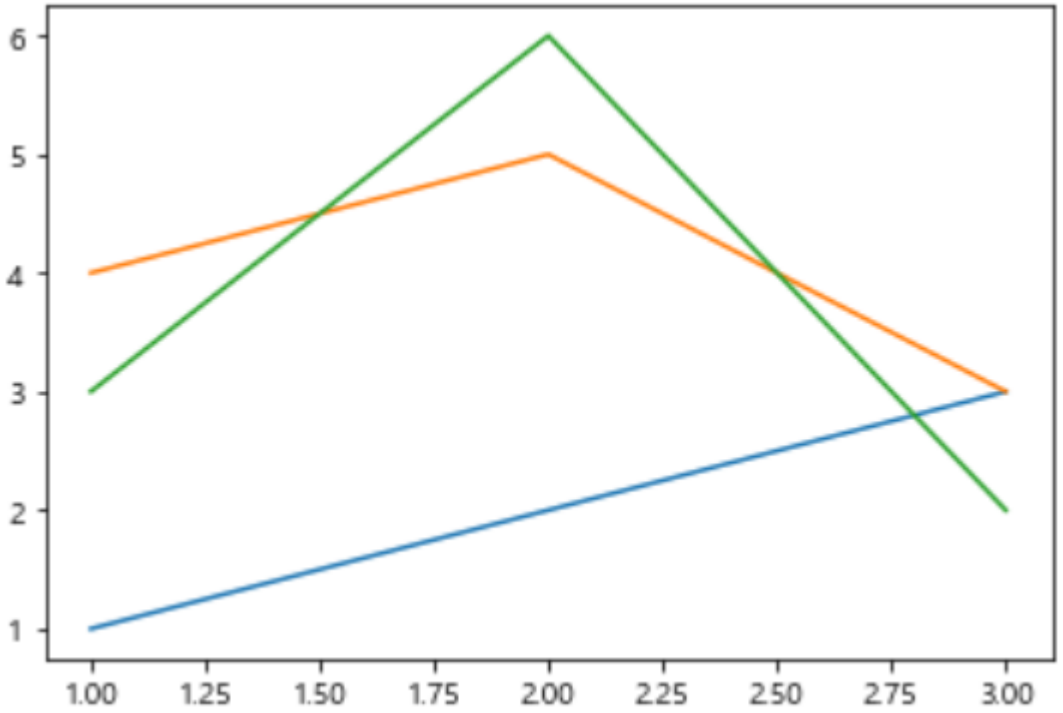
Matplotlib

Enjoy your data analysis

show() 사용

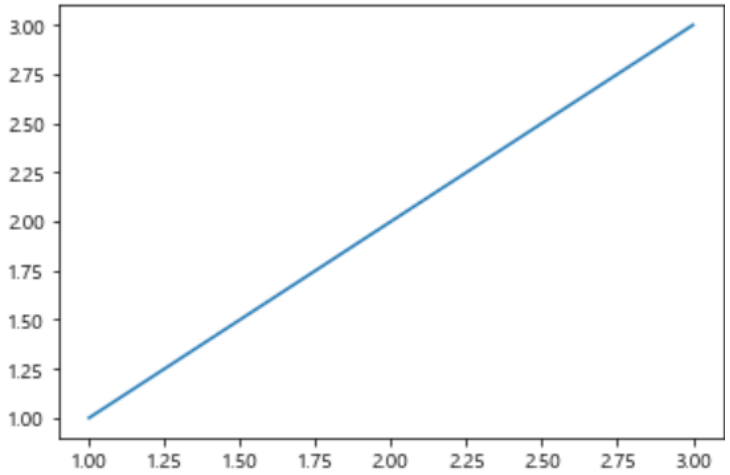
작성된 모든 figure를 표시하는 기능

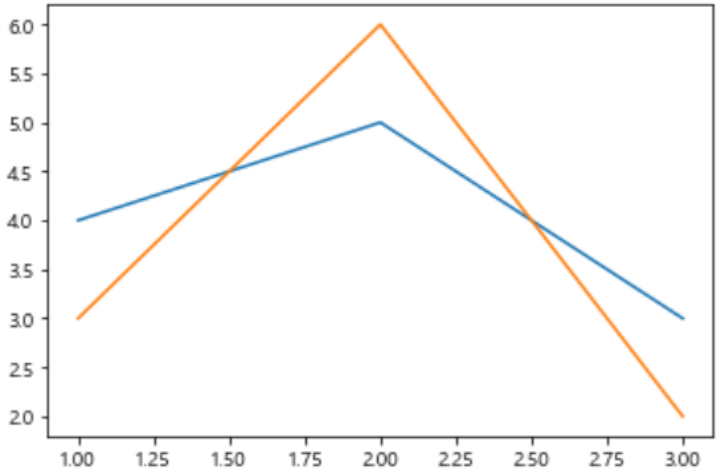
```
# 여러 그래프를 한창에 그리기
x = np.arange(1,4)
y1 = np.array([1,2,3])
y2 = np.array([4,5,3])
y3 = np.array([3,6,2])
plt.plot(x,y1)
plt.plot(x,y2)
plt.plot(x,y3)
plt.show()
```



show() 사용

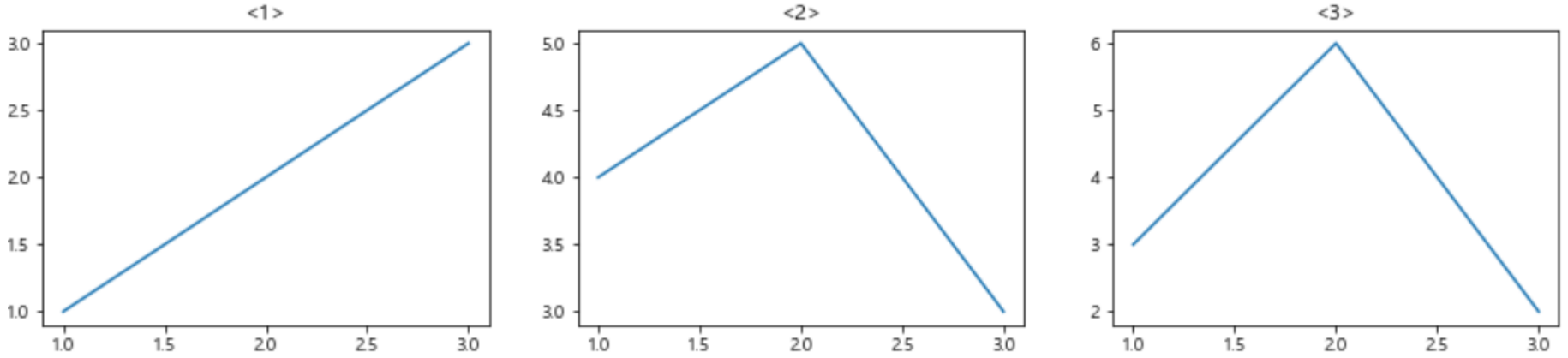
```
# 여러 창으로 나누어 그래프 그리기
plt.plot(x,y1)
plt.show()
plt.plot(x,y2)
plt.plot(x,y3)
plt.show()
```





subplot() - 한 창에 칸 나눠서 그래프 여러 개 표시하기

```
plt.figure(figsize=(15,3))
plt.subplot(1,3,1)
plt.title('<1>')
plt.plot(x,y1)
plt.subplot(1,3,2)
plt.title('<2>')
plt.plot(x,y2)
plt.subplot(1,3,3)
plt.title('<3>')
plt.plot(x,y3)
plt.show()
```



예제1

국가통계포털의 장래_인구변동 데이터를 이용하여
lineplot으로 출생아수, 사망자수 시각화

```
# 1) 데이터 불러오기
data = pd.read_csv('./data/장래_인구변동_KOSIS.csv',
                   encoding = 'euc-kr', index_col = '인구변동요인별')
data
```

인구변동요인별	2020	2030	2040	2050	2060	2070
인구(천명)	51836	51199	50193	47359	42617	37656
출생아수(천명)	275	305	286	236	181	196
사망자수(천명)	308	408	527	680	741	702

예제1

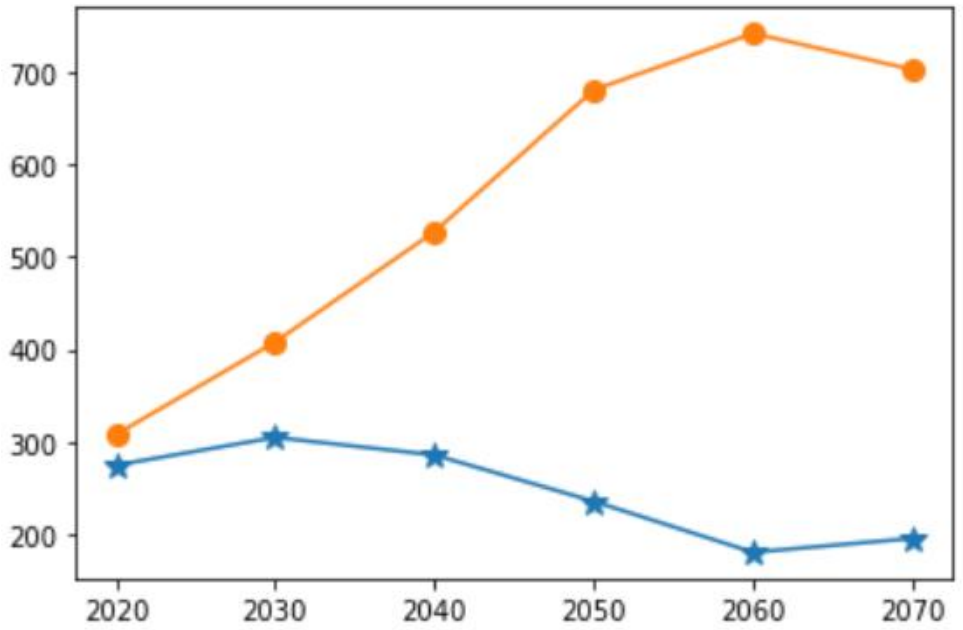
```
# 2) 데이터 살펴보기
data.info()
data.index
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 3 entries, 인구(천명) to 사망자수(천명)
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   2020    3 non-null      int64
 1   2030    3 non-null      int64
 2   2040    3 non-null      int64
 3   2050    3 non-null      int64
 4   2060    3 non-null      int64
 5   2070    3 non-null      int64
dtypes: int64(6)
memory usage: 168.0+ bytes
```

```
Index(['인구(천명)', '출생아수(천명)', '사망자수(천명)'], dtype='object', name
='인구변동요인별')
```

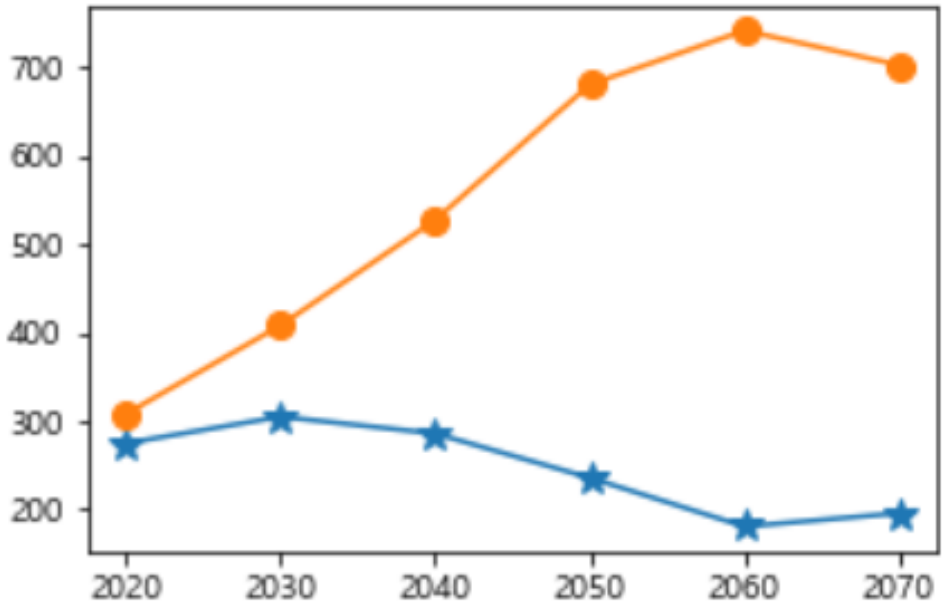
예제1

```
# 3) 출생아수와 사망자수 선그래프 시각화
# x, y 축 데이터 설정
x = range(2020,2071,10)
y1  =  data.iloc[1] # 출생아 수
y2 = data.iloc[2] # 사망자 수

plt.figure(figsize = (6,4))
plt.plot(x, y1, marker='*', ms = 10)
plt.plot(x, y2, marker='o', ms = 8)
plt.show()
```



예제1

보다 명확하고 이해하기 쉬운 시각화 결과를 제공하기 위한 그래프 옵션 적용해보기



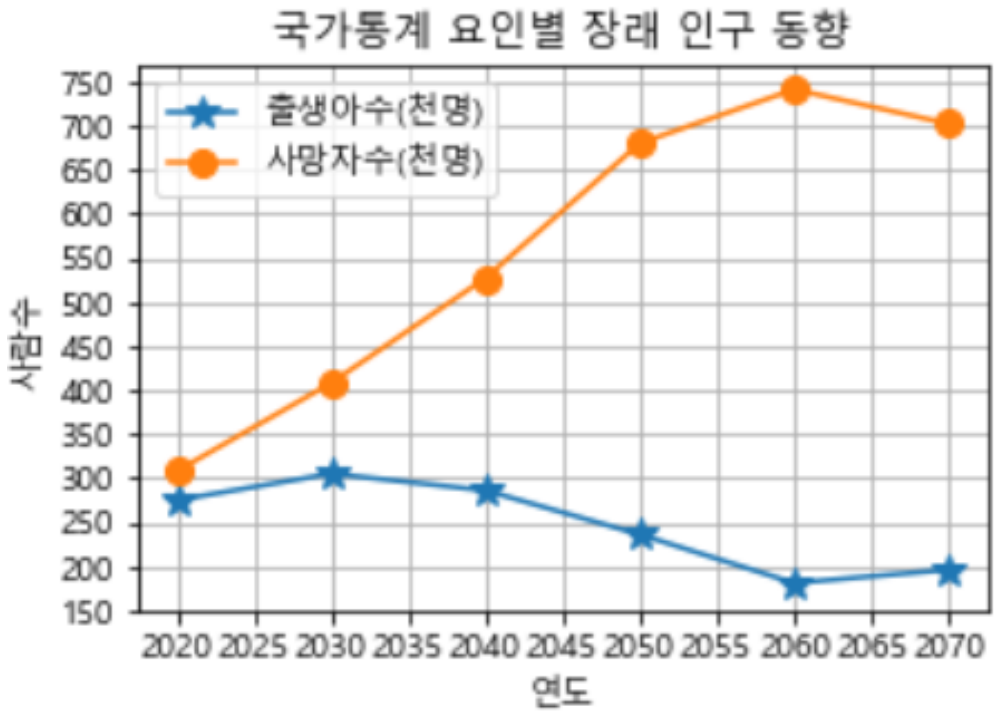


Chart option

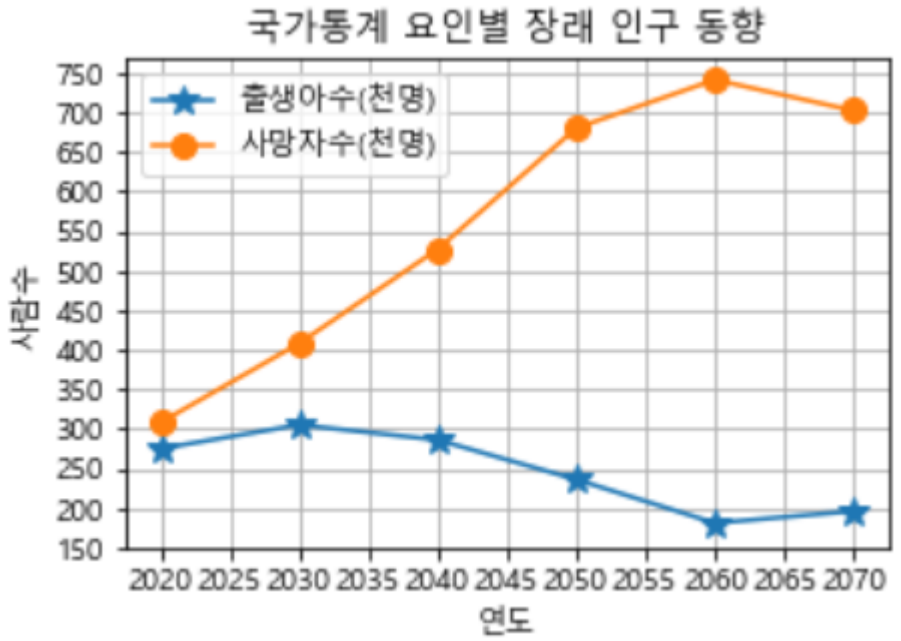
plot 옵션	정의
xlim, ylim	x 축 범위, y축 범위
grid	격자눈금
legend	범례
xlabel, ylabel	x축 타이틀, y축 타이틀
title	그래프 제목
xticks, yticks	x축 눈금 조정, y축 눈금 조정

Matplotlib

Enjoy your data analysis

예제1

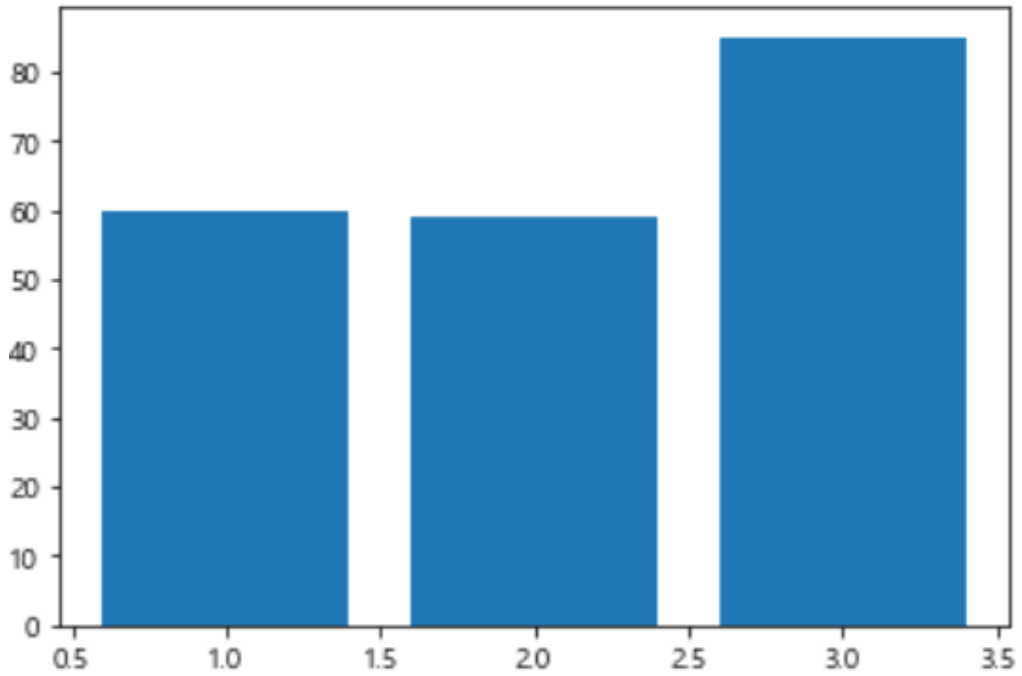
```
plt.plot(x, y1, marker='*', ms = 10,
         label = '출생0수(천명)')
plt.plot(x, y2, marker='o', ms = 8,
         label = '사망자수(천명)')
plt.title('국가통계 요인별 장래 인구 동향')
plt.xlabel('연도')
plt.ylabel('사람수')
plt.grid()
plt.xticks(range(2020,2071,5))
plt.yticks(range(150,800,50))
# plt.xlim('2020','2050')
# plt.ylim(200,550)
plt.legend()
plt.show()
```



3 Bar chart

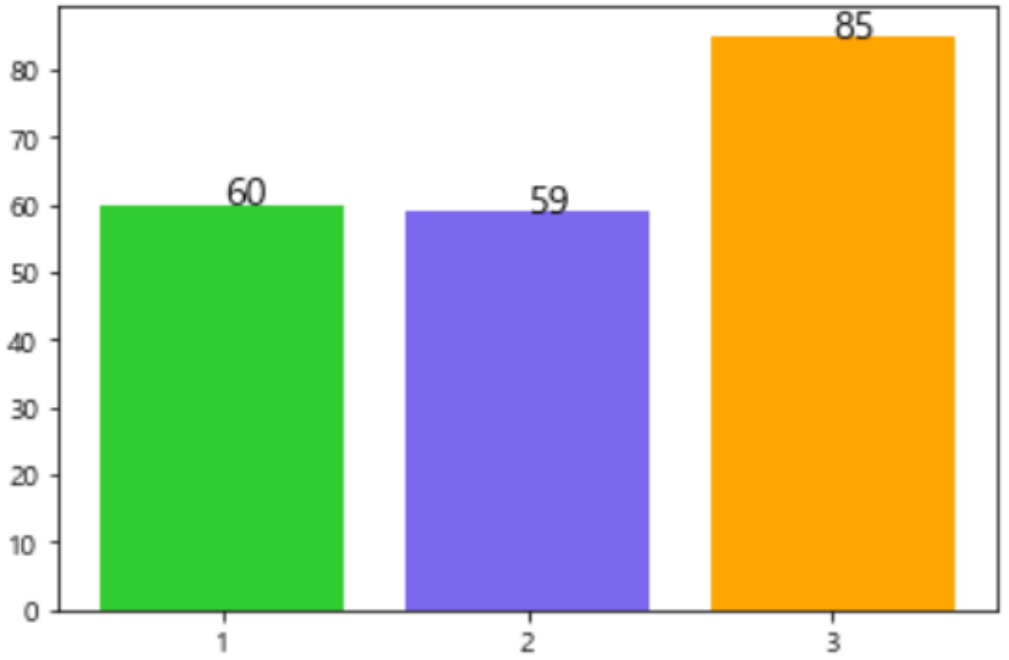
Bar chart 그려보기

```
# 데이터 준비하여 막대 그래프 그려보기
np.random.seed(6)
bar_x = np.arange(1,4)
bar_y = np.random.randint(50,
                          100,
                          size=3)

plt.bar(bar_x,bar_y)
plt.show()
```



Bar chart 옵션 추가 - 막대 색상 변경, 텍스트 추가

```
# 그래프 옵션 추가
# 막대그래프 색상 변경, 텍스트 출력
plt.bar(bar_x,bar_y,
            color=['limegreen’,
                        'mediumslateblue’,
                        'orange’])

for i in range(len(bar_x)):
    plt.text(bar_x[i], bar_y[i], f ' {bar_y[i]} ' ,
                fontdict={ ' color ' : ' black ' ,
                                ' size ' :14})

plt.xticks(range(1,4))
plt.show()
```



예제2

국가통계포털 경제활동 인구 데이터 활용 데이터 분석 및 시각화

```
# 1) 데이터 불러오기
economi = pd.read_csv('./data/시도_성별_경제활동인구_총괄_KOSIS_2022.csv',
                      encoding = 'euc-kr')
economi.head()
```

	행정구역(시도)	성별	15세이상인구 (천명)	경제활동인구 (천명)	취업자 (천명)	실업자 (천명)	비경제활동인구 (천명)	경제활동참가율(%)	고용률(%)	실업률(%)
0	서울특별시	남자	4007	2849	2760	89	1158	71.1	68.9	3.1
1	서울특별시	여자	4440	2418	2347	72	2022	54.5	NaN	3.0
2	부산광역시	남자	1406	973	946	27	433	69.2	67.3	2.7
3	부산광역시	여자	1528	768	745	23	760	50.3	48.8	3.0
4	대구광역시	남자	1001	708	691	17	294	70.6	69.0	2.3

예제2

2) 데이터 정보 확인
economi.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34 entries, 0 to 33
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   행정구역(시도)      34 non-null     object
 1   성별              34 non-null     object
 2   15세이상인구 (천명)  34 non-null     int64
 3   경제활동인구 (천명)   34 non-null     int64
 4   취업자 (천명)       34 non-null     int64
 5   실업자 (천명)       34 non-null     int64
 6   비경제활동인구 (천명)  34 non-null     int64
 7   경제활동참가율(%)    34 non-null     float64
 8   고용률(%)         24 non-null     float64
 9   실업률(%)         28 non-null     float64
dtypes: float64(3), int64(5), object(2)
memory usage: 2.8+ KB
```

null값 확인
economi.isnull().sum()

```
행정구역(시도)          0
성별                 0
15세이상인구 (천명)      0
경제활동인구 (천명)       0
취업자 (천명)          0
실업자 (천명)          0
비경제활동인구 (천명)      0
경제활동참가율(%)        0
고용률(%)            10
실업률(%)             6
dtype: int64
```

고용률
생산가능연령인 15세 이상 인구 중에서 특정시점에 취업하고 있는 사람의 비율

실업률
경제활동참가자 중에서 실업상태에 있는 사람의 비율

round(): 소수점 반올림
round(3.362, 1)
결과: 3.4

예제2

```
# 3) 고용률(%) 전처리하기
# 특성공학-> 기존의 컬럼을 활용 데이터 생성
# 고용률 = 취업자/15세이상인구

display(economi['고용률(%)'].head())

emp_pt =  economi['취업자 (천명)']/economi['15세이상인구 (천명)']
emp_pt = round(emp_pt,3)*100
economi['고용률(%)'] = emp_pt

display(economi['고용률(%)'].head())
```

```
0    68.9
1     NaN
2    67.3
3    48.8
4    69.0
Name: 고용률(%), dtype: float64
```

```
0    68.9
1    52.9
2    67.3
3    48.8
4    69.0
Name: 고용률(%), dtype: float64
```

예제2

```
# 4) 실업률(%) 전처리하기
# 실업률 = 실업자/경제활동인구

display(economi['실업률(%)'].tail())

unemp_pt =  economi['실업자 (천명)']/economi['경제활동인구 (천명)']
unemp_pt = round(emp_pt,3)*100
economi[' 실업률(%)'] = unemp_pt

display(economi[' 실업률(%)']. tail())
```

```
29    2.1
30    2.6
31    NaN
32    2.3
33    1.0
Name: 실업률(%), dtype: float64
```

```
29    2.1
30    2.6
31    2.6
32    2.2
33    1.0
Name: 실업률(%), dtype: float64
```

예제2

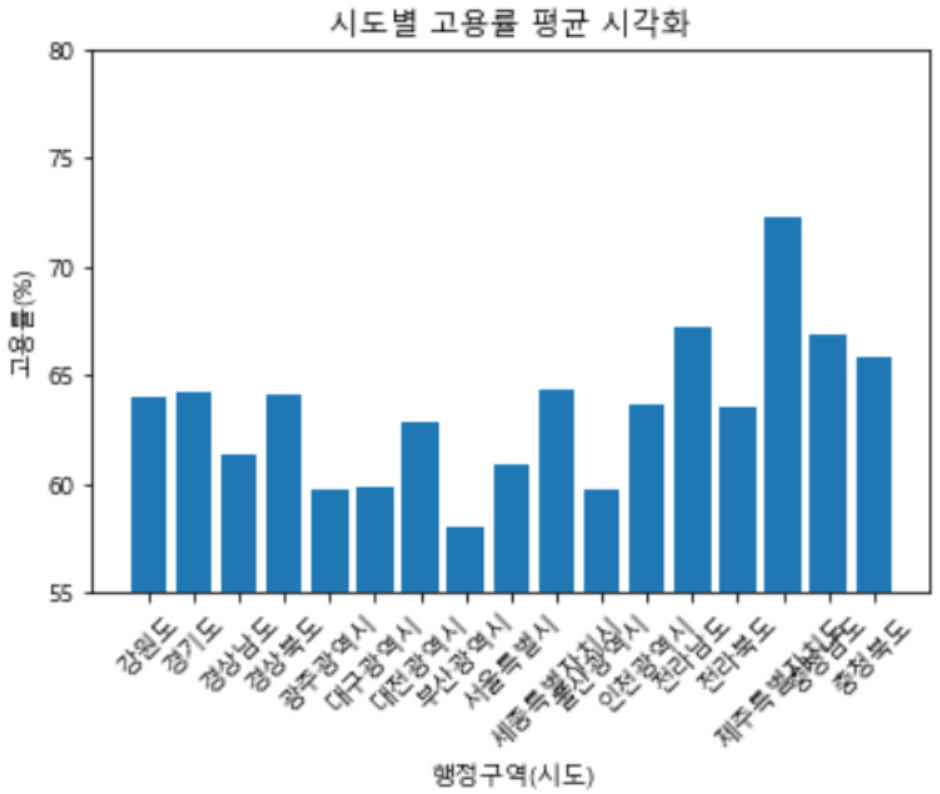
```
# 5) 시도별 고용률 평균 막대 그래프로 시각화하고 고용률이 가장 높은 지역 확인

state_emp = economi[['행정구역(시도)','고용률(%)']].groupby('행정구역(시도)',
                                                    as_index = False).mean()

state_emp.head()
```

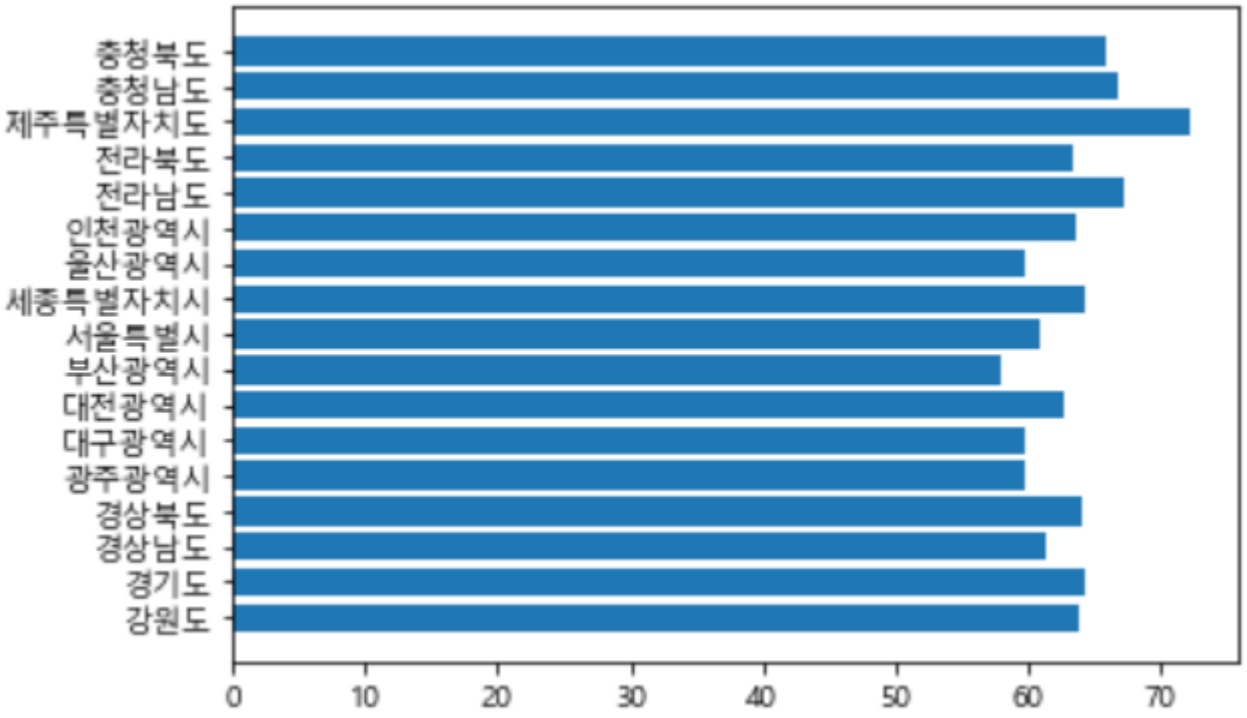
	행정구역(시도)	고용률(%)
0	강원도	63.95
1	경기도	64.25
2	경상남도	61.40
3	경상북도	64.10
4	광주광역시	59.80

Matplotlib

Enjoy your data analysis

예제2



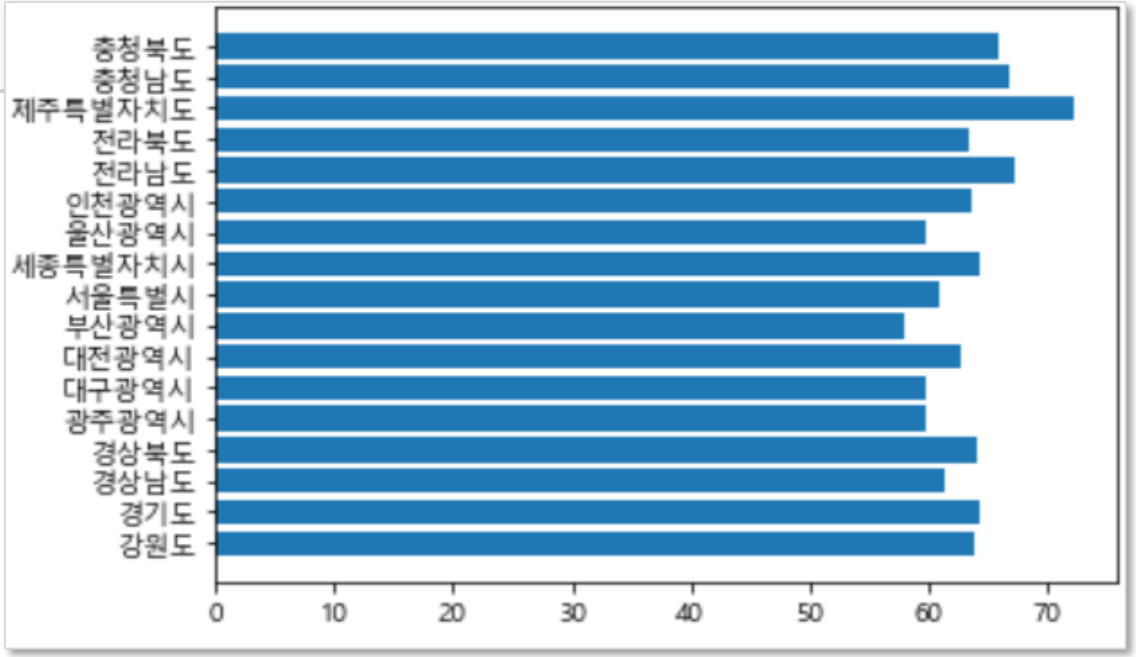


예제2

```
# 5) 시도별 고용률 평균 막대 그래프로 시각화하고 고용률이 가장 높은 지역 확인
# Vertical Bar (수직 막대그래프)
plt.figure(figsize =(8,4))
plt.bar(state_emp['행정구역(시도)'],state_emp['고용률(%)'])
plt.xticks(rotation = 45)
plt.title('시도별 고용률 평균 시각화')
plt.xlabel('행정구역(시도)')
plt.ylabel('고용률(%)')
# ylim 크기 55~80 조절해서 확인
plt.ylim(55,80)
plt.show()
```

예제2

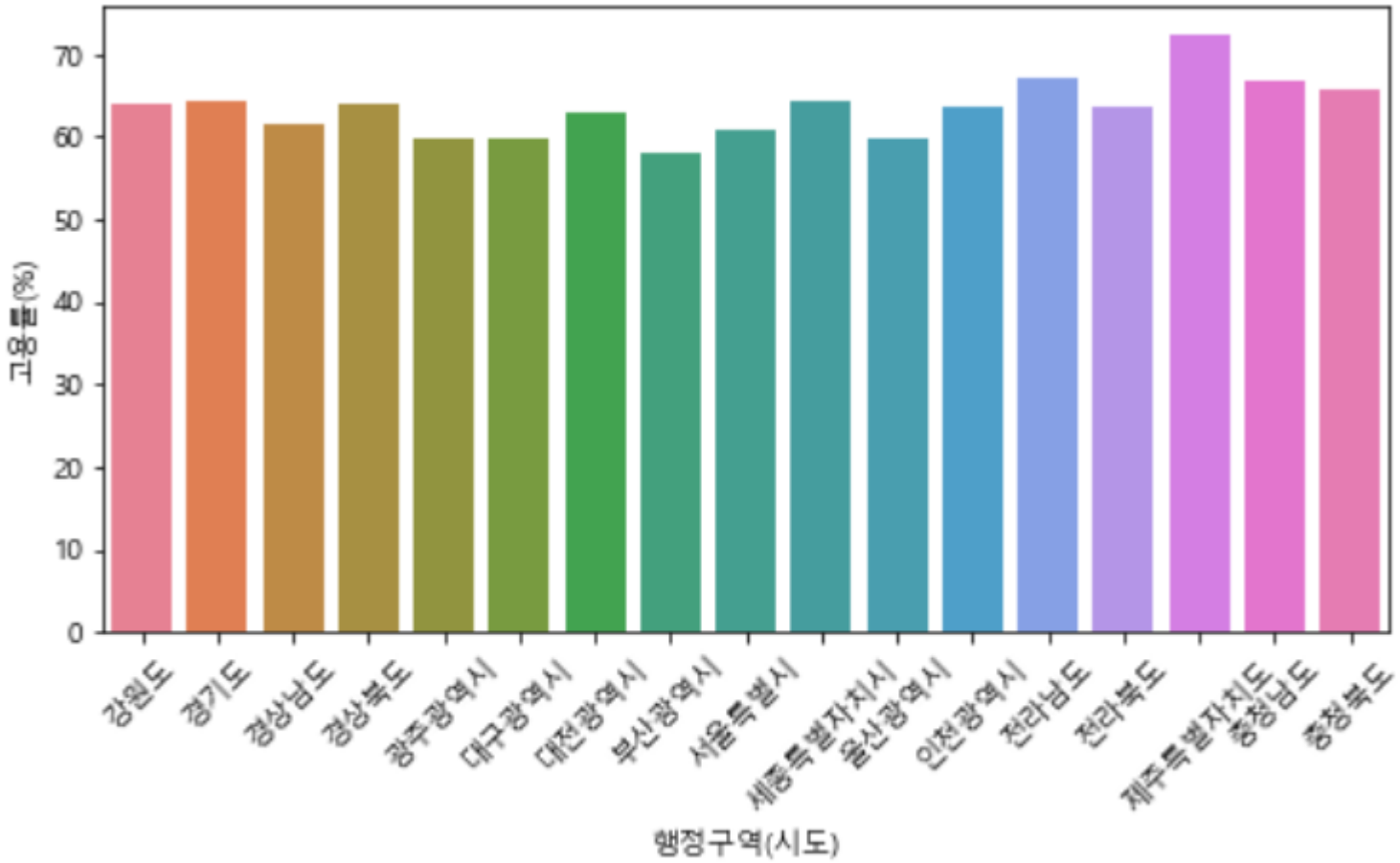
```
# 5) 시도별 고용률 평균 막대 그래프로 시각화하고 고용률이 가장 높은 지역 확인
# Horizontal Bar (수평 막대그래프)
plt.figure(figsize =(10,4))
plt.barh(state_emp['행정구역(시도)'],state_emp['고용률(%)'])
plt.show()
```



Matplotlib

Enjoy your data analysis

예제2



예제2

```
# 5) 시도별 고용률 평균 막대 그래프로 시각화하고 고용률이 가장 높은 지역 확인
# seaborn 라이브러리 이용 막대그래프 그리기

plt.figure(figsize =(8,4))
sns.barplot(x = state_emp['행정구역(시도)'], y = state_emp['고용률(%)'],
                    palette="husl")
# 그래프에 표시는 팔레트 색상 개수는 xlabel 수만큼 자동으로 설정
plt.xticks(rotation = 45)
plt.show()
```

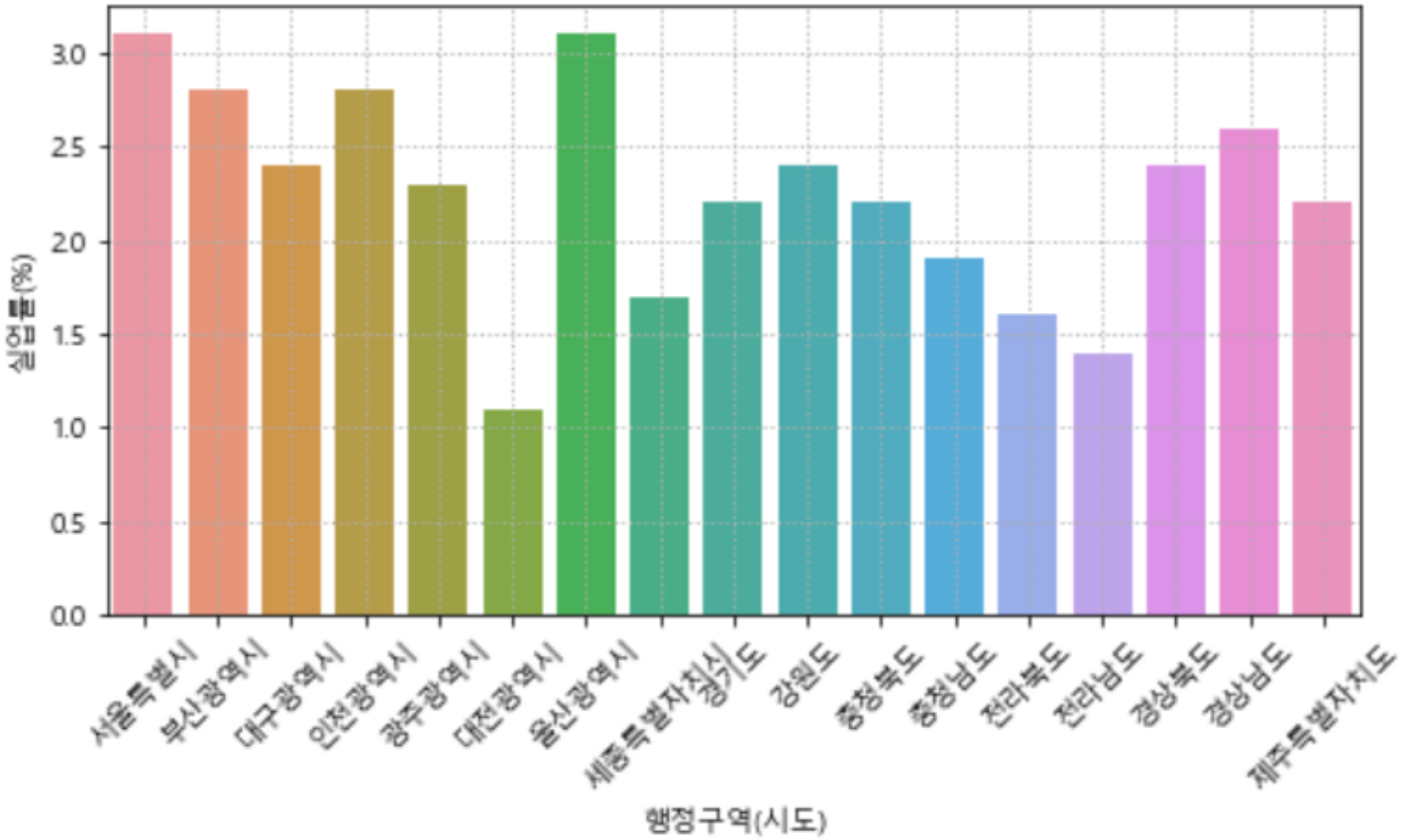
예제2

```
# 6) 시도별 남자의 실업률을 시각화하고 가장 높은 곳은 어디인지 확인하기

man = economi.query("성별 == '남자'")
man_unemp = man[['행정구역(시도)','실업률(%)']]
man_unemp.head()
```

	행정구역(시도)	실업률(%)
0	서울특별시	3.1
2	부산광역시	2.8
4	대구광역시	2.4
6	인천광역시	2.8
8	광주광역시	2.3

예제2

6) 시도별 남자의 실업률을 시각화하고 가장 높은 곳은 어디인지 확인하기



예제2

```
# 6) 시도별 남자의 실업률을 시각화하고 가장 높은 곳은 어디인지 확인하기

plt.figure(figsize = (8,4))
sns.barplot(x = man_unemp['행정구역(시도)'], y = man_unemp['실업률(%)'])
plt.grid(ls=':', lw = 1)
plt.xticks(rotation=45)
plt.show()
```

4 Scatter plot

Pie chart

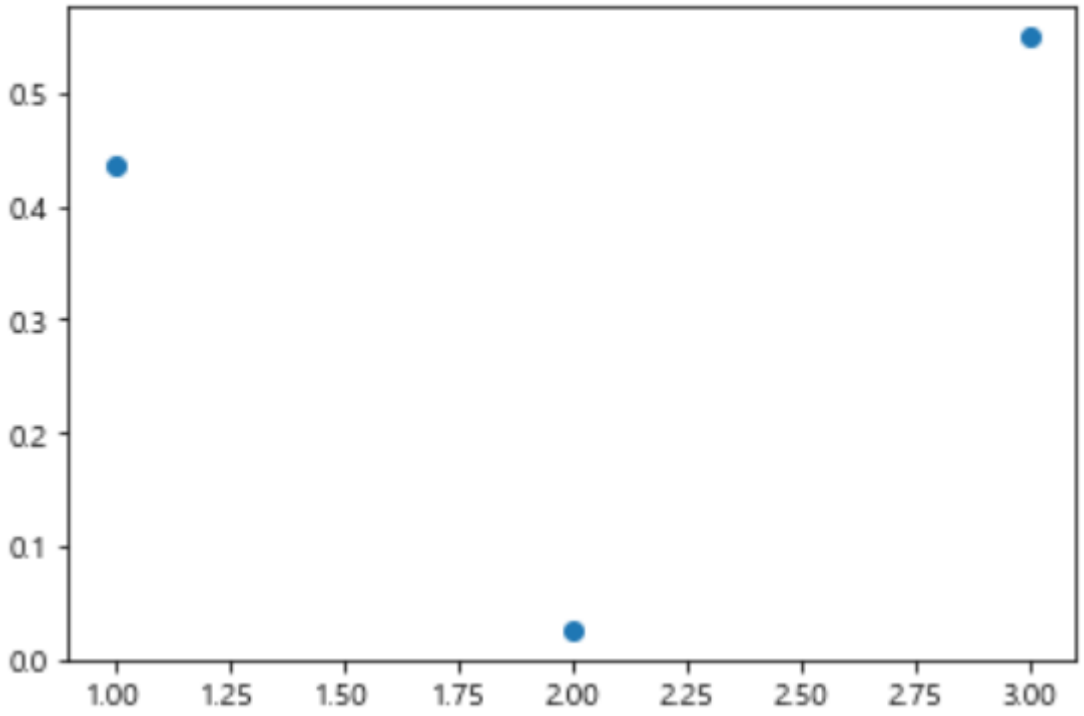
Matplotlib

Enjoy your data analysis

Scatter plot

```
np.random.seed(2)
x = np.arange(1,4)
y = np.random.rand(3)
print(x,y)

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

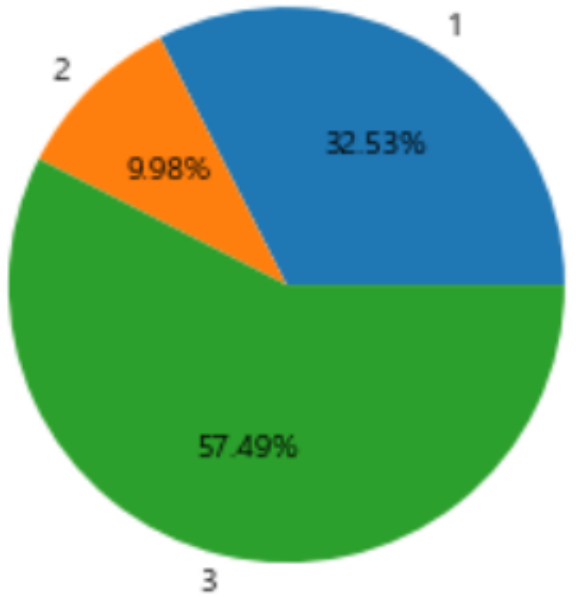
```
[1 2 3] [0.4359949  0.02592623 0.54966248]
```



Pie chart

```
for i in y:
   print(f'{i/y.sum()*100:.2f}')

plt.pie(y, labels = x, autopct= '%.2f%%')
plt.show()
# %% 문자열 조합 : 이스케이프 코드 => % 문자열 출력
```

```
32.53
9.98
57.49
```



5 종합 최종예제

전국교통사고 2017 데이터 활용 분석 및 시각화

전국교통사고 2017 데이터 활용 분석 및 시각화

1) 데이터 불러오기

2) 데이터 정보 확인하기

3) 요일별 교통사고 시각화

4) 차대차 사건 중 죽거나 다친 사람이 많은 발생지 시도를 알아보고 시각화 해보자

5) 교통사고가 가장 많이 발생하는 시간대를 알아보고 시각화 해보자

6) 광주지역 법규위반 사항별 사고 건수를 파이그래프로 시각화해보자

Matplotlib

Enjoy your data analysis

데이터 불러오기

```
acc_data = pd.read_csv('./data/Traffic_Accident_2017.csv',
                       encoding ='euc-kr')

acc_data.head(2)
```

	발생년	발생년월일시	발생분	주야	요일	사망자수	사상자수	중상자수	경상자수	부상신고자수	발생지시도	발생지시군구	사고유형_대분류	사고유형_중분류	사고유형	법규위반_대분류	법규위반	도로형태_대분류	도로형태	당사자종별_1당_대분류	당사자종별_1당	당사자종별_2당_대분류	당사자종별_2당	발생위치X_UTMK	발생위치Y_UTMK	경도	위도
0	2017	2017010101	15	야간	일	1	2	1	0	0	전남	장성군	차대차	기타	기타	운전자법규위반	안전운전의무불이행	단일로	기타단일로	승용차	승용차	승용차	승용차	933501	1700129	126.768634	35.294464
1	2017	2017010102	28	야간	일	1	1	0	0	0	서울	송파구	차대사람	횡단중	횡단중	운전자법규위반	안전운전의무불이행	교차로	교차로부근	승용차	승용차	보행자	보행자	967570	1944453	127.133107	37.498741

Matplotlib

Enjoy your data analysis

데이터 정보 확인하기

```
print(acc_data.shape)
acc_data.info()
```

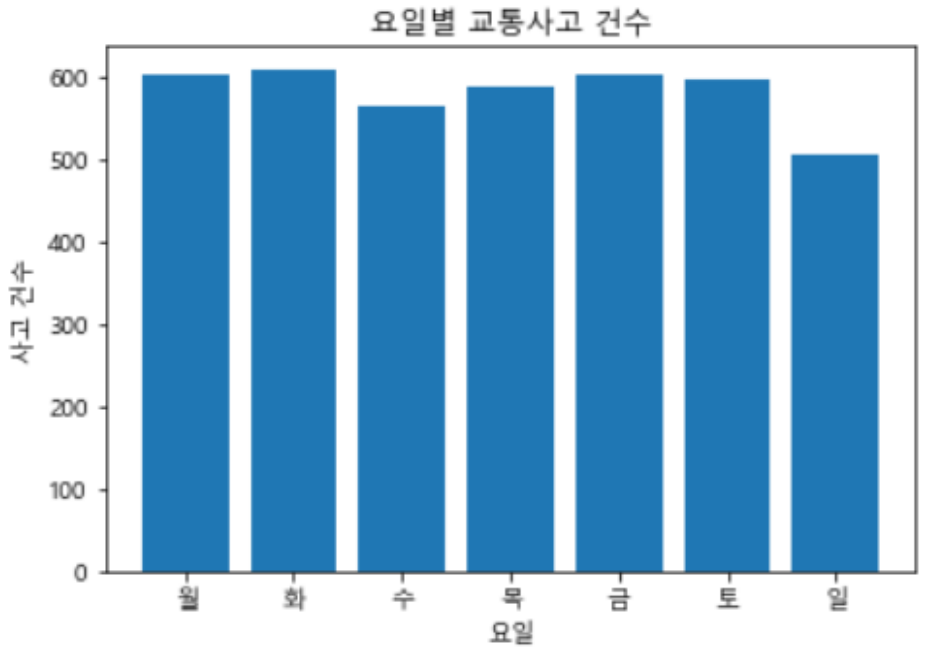
```
(4065, 27)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4065 entries, 0 to 4064
Data columns (total 27 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   발생년            4065 non-null   int64
 1   발생년월일시        4065 non-null   int64
 2   발생분            4065 non-null   int64
 3   주야              4065 non-null   object
 4   요일              4065 non-null   object
 5   사망자수           4065 non-null   int64
 6   사상자수           4065 non-null   int64
 7   중상자수           4065 non-null   int64
 8   경상자수           4065 non-null   int64
 9   부상신고자수         4065 non-null   int64
 10  발생지시도          4065 non-null   object
 11  발생지시군구         4065 non-null   object
 12  사고유형_대분류       4065 non-null   object
 13  사고유형_중분류       4065 non-null   object
 14  사고유형            4065 non-null   object
 15  법규위반_대분류       4065 non-null   object
 16  법규위반            4065 non-null   object
 17  도로형태_대분류       4065 non-null   object
 18  도로형태            4065 non-null   object
 19  당사자종별_1당_대분류  4065 non-null   object
 20  당사자종별_1당       4065 non-null   object
 21  당사자종별_2당_대분류  4065 non-null   object
 22  당사자종별_2당       4065 non-null   object
 23  발생위치X_UTMK     4065 non-null   int64
 24  발생위치Y_UTMK     4065 non-null   int64
 25  경도              4065 non-null   float64
 26  위도              4065 non-null   float64
dtypes: float64(2), int64(10), object(15)
memory usage: 857.6+ KB
```

요일별 교통사고 시각화

```
tmp = acc_data['요일'].value_counts()
tmp = tmp[['월','화','수','목','금','토','일']]
tmp
```

```
월    603
화    608
수    565
목    586
금    603
토    596
일    504
Name: 요일, dtype: int64
```

```
plt.bar(tmp.index, tmp)
plt.title('요일별 교통사고 건수')
plt.ylabel('사고 건수')
plt.xlabel('요일')
plt.show()
```



차대차 사건 중 죽거나 다친 사람이 많은 발생지 시도를 알아보고 시각화 해보자

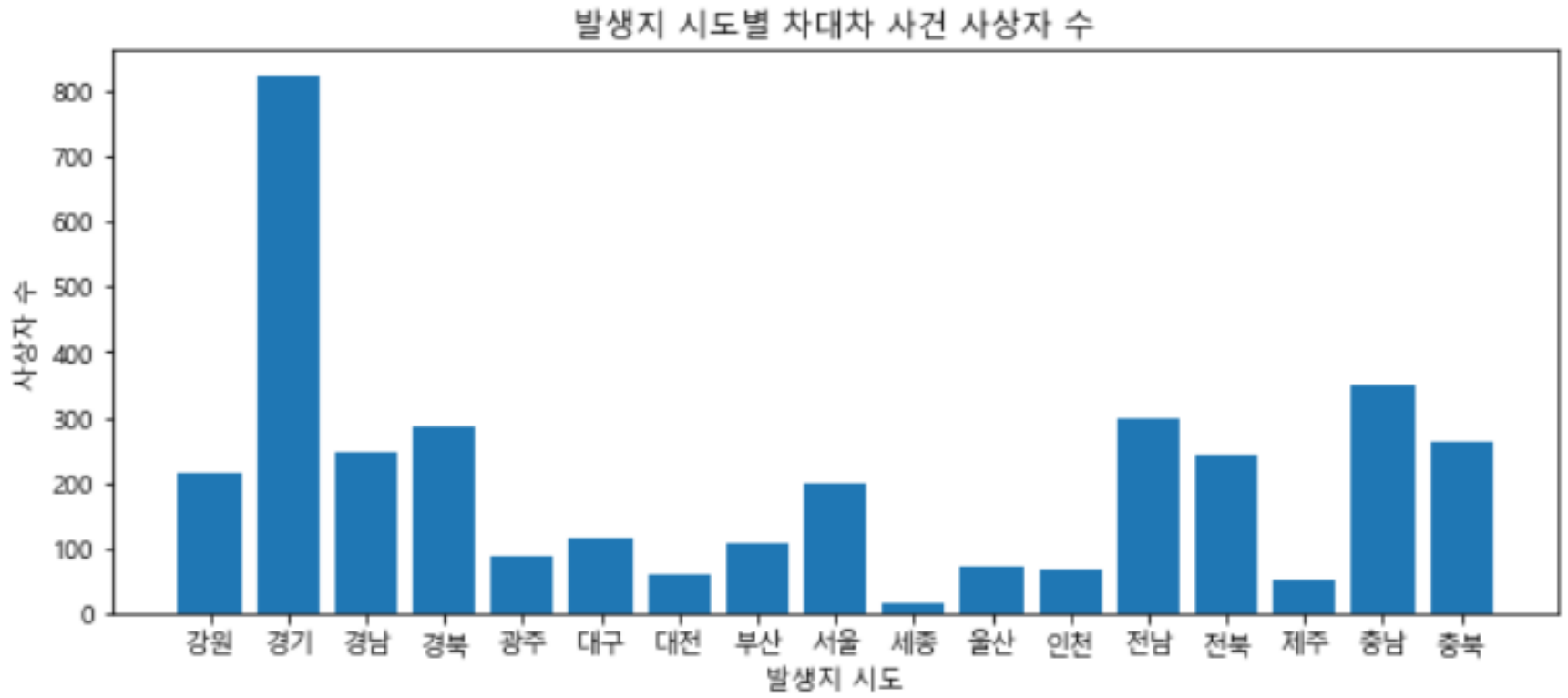
```
# acc_data[acc_data['사고유형_대분류'] =='차대차']
car_acc = acc_data.query("사고유형_대분류 == '차대차'")
state_car_acc = car_acc[['발생지시도','사상자수']].groupby('발생지시도',
                                                    as_index = False).sum()
state_car_acc.head(5)
```

	발생지시도	사상자수
0	강원	214
1	경기	824
2	경남	248
3	경북	287
4	광주	87

Matplotlib

Enjoy your data analysis

차대차 사건 중 죽거나 다친 사람이 많은 발생지 시도를 알아보고 시각화 해보자



Matplotlib

Enjoy your data analysis

차대차 사건 중 죽거나 다친 사람이 많은 발생지 시도를 알아보고 시각화 해보자

```
# 시각화
plt.figure(figsize = (10,4))

plt.bar(state_car_acc['발생지시도'], state_car_acc['사상자수'])
plt.title('발생지 시도별 차대차 사건 사상자 수')
plt.ylabel('사상자 수')
plt.xlabel('발생지 시도')
plt.savefig('data/요일별_교통사고_건수.png') # 이미지 저장
plt.show()
```

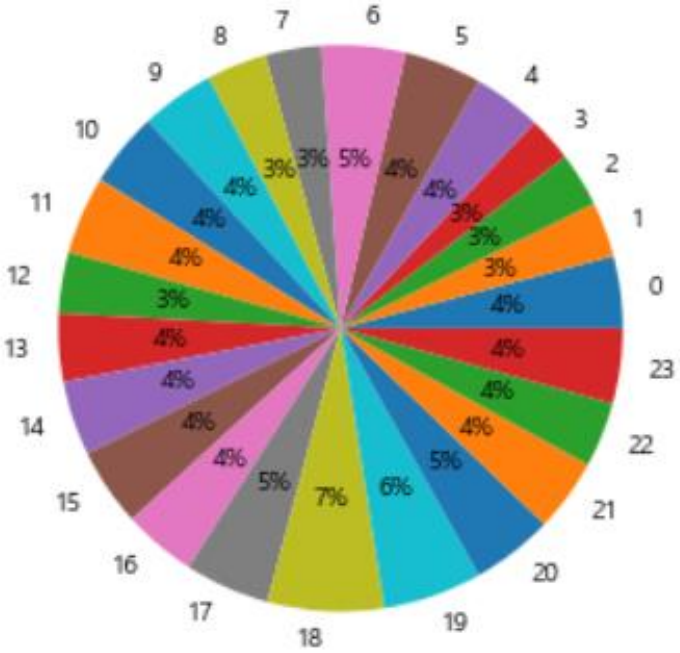
교통사고가 가장 많이 발생하는 시간대를 알아보고 시각화 해보자

```
acc_time = acc_data['발생년월일시']%100
acc_time_cnt = acc_time.value_counts().sort_index()
acc_time_cnt
```

```
0      167
1      128
2      126
3      105
4      159
5      180
      ...
```

교통사고가 가장 많이 발생하는 시간대를 알아보고 시각화 해보자

```
# 파이플롯 시각화
plt.figure(figsize = (5,5))
plt.pie(acc_time_cnt, labels = acc_time_cnt.index,
        autopct = '%1.0f%%')
plt.show()
```



시간대 간격이 짧고 구분 카테고리가 많아져서
발생 비율의 크고 작음의 의미 파악이 어려움,
카테고리를 줄여서 보다 명확한 관찰을 해보자

교통사고가 가장 많이 발생하는 시간대를 알아보고 시각화 해보자

```
# 사소한 관찰의 오류를 줄여보기 위해 카테고리를 구간화(binning) 시켜보자
# 0~2,3~5,6~8,9~11,12~14,15~17, 18~20,21~23 => 24구간을 8구간으로

bins = [-1,2,5,8,11,14,17,20,23]
labels = ['0~2','3~5','6~8','9~11','12~14','15~17', '18~20','21~23']
acc_time2= pd.cut(acc_time, bins = bins, labels = labels)
acc_time2_cnt = acc_time2.value_counts()
acc_time2_cnt
```

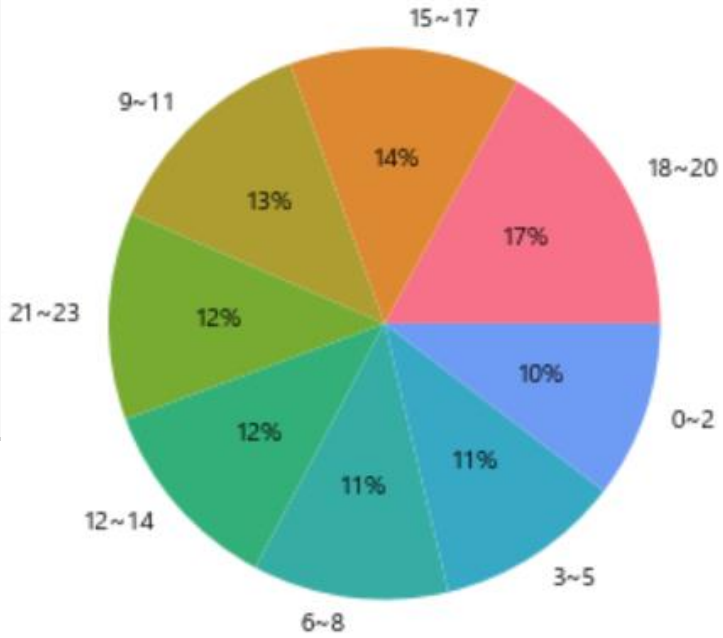
```
18~20    692
15~17    550
9~11     526
21~23    494
12~14    472
6~8      466
3~5      444
0~2      421
Name: 발생년월일시, dtype: int64
```

Matplotlib

Enjoy your data analysis

교통사고가 가장 많이 발생하는 시간대를 알아보고 시각화 해보자

```
colors = sns.color_palette("husl",10)

plt.figure(figsize = (5,5))
plt.pie(acc_time2_cnt, labels = acc_time2_cnt.index,
            autopct = '%1.0f%%', colors =  colors)
plt.show()
```



광주지역 법규위반 사항별 사고 건수를 파이그래프로 시각화해보자

```
gj_data = acc_data.query('발생지시도 =="광주"')
gj_data['법규위반'].value_counts()
```

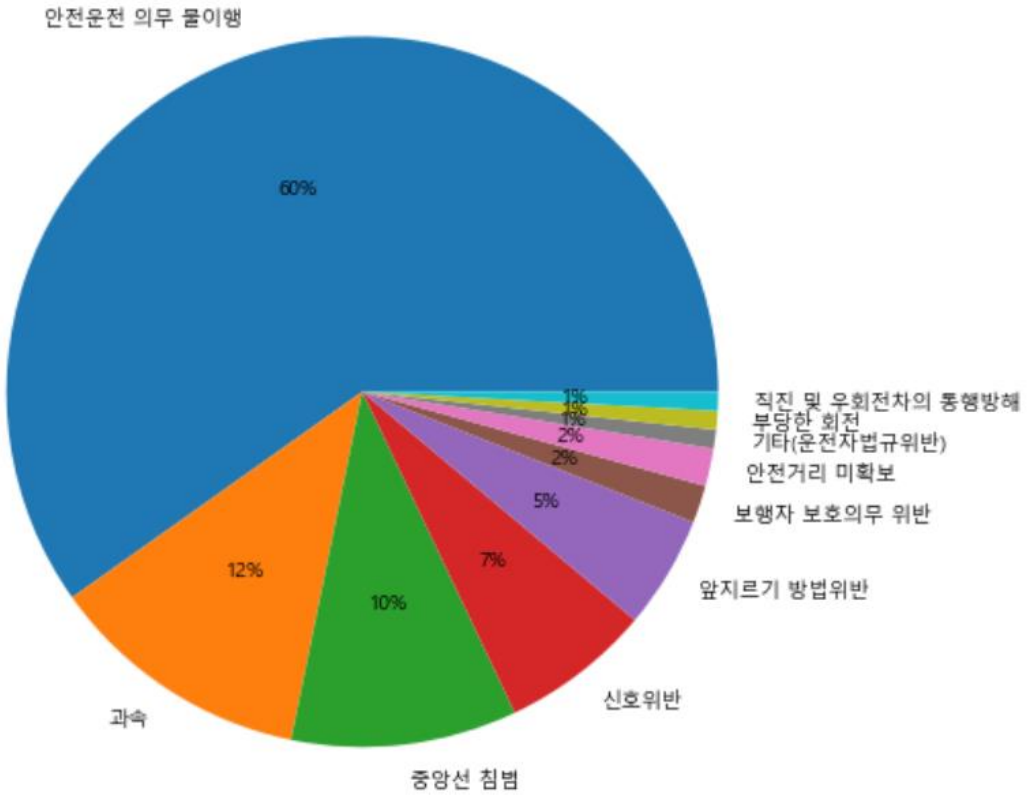
```
안전운전 의무 불이행          70
신호위반                    14
과속                        12
중앙선 침범                   8
보행자 보호의무 위반            6
안전거리 미확보                 2
기타(운전자법규위반)            2
앞지르기 방법위반               1
부당한 회전                   1
직진 및 우회전차의 통행방해       1
Name: 법규위반, dtype: int64
```

Matplotlib

Enjoy your data analysis

광주지역 법규위반 사항별 사고 건수를 파이그래프로 시각화해보자

```
plt.figure(figsize = (8,8))
plt.pie(gj_data['법규위반'].value_counts(),
        labels = gj_data['법규위반'].unique(),
        autopct = '%1.0f%%')
plt.show()
```



Big Data

수고하셨습니다.

Enjoy your data analysis