



Introduction To Data Analysis

데이터 분석 입문

Lecture 30. pandas로 연령별 인구구조 유사지역 분석

인공지능소프트웨어학과 강환수 교수

학습개요

- ✓ 연령별 인구 데이터를 pandas로 읽어와 전처리 및 기본 연산
- ✓ 연령별 인구구조 분석을 위한 관심 지역 선정과 유사한 연령별 인구구조 지역을 찾기 위한 계산
- ✓ 관심지역과 유사한 연령별 인구구조 지역 5개 데이터 시각화



학습목표

- ✓ 연령별 인구 데이터 pandas로 읽어와 기본 전처리와 기본 연산을 수행할 수 있다.
- ✓ 연령별 인구구조 분석을 위한 관심 지역을 선정해 DataFrame에 저장할 수 있다
- ✓ 관심지역과 유사한 연령별 인구구조 지역을 찾기 위한 계산을 수행할 수 있다.
- ✓ 관심지역과 유사한 연령별 인구구조 지역 5개의 데이터 시각화를 할 수 있다.

LESSON 01

연령별 인구구조 DataFrame 저장과 전처리



알고리즘(Algorithm) 설계하기

Step 1) 데이터를 읽어온다.

- ✓ ① 행정구역의 총 인구수가 0인 행을 제거한다.
- ✓ ② 전체 데이터를 총 인구수로 나누어 비율로 변환한다.
- ✓ ③ 총 인구수와 연령 구간 인구수를 삭제한다.

Step 2) 궁금한 지역의 이름을 입력 받는다.

Step 3) 궁금한 지역의 인구 구조를 저장한다.

Step 4) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역을 찾는다.

- ✓ ① 전국의 모든 지역 중 한 곳(B)를 선택한다.
- ✓ ③ 100세 이상 인구수에 해당하는 값까지 선정된 지역과의 차이의 제곱을 모두 더한다.
- ✓ ④ 전국의 모든 지역에 대해 반복하며 그 차이가 가장 작은 지역을 찾는다.

Step 5) 가장 비슷한 곳의 인구 구조와 궁금한 지역의 인구 구조를 시각화한다.

Step 1) 데이터를 읽어 오기 (1/2)

첫 열이 index(행 제목)가 되도록: index_col=0

```
import pandas as pd
```

```
df = pd.read_csv('age.csv', encoding='cp949', index_col=0)  
df.head()
```

✓ 0.5s

Python

| | 2023년 09월_계_ 총인구수 | 2023년 09월_계_ 연령구간 인구수 | 2023 년09 월_계 _0세 | 2023 년09 월_계 _1세 | 2023 년09 월_계 _2세 | 2023 년09 월_계 _3세 | 2023 년09 월_계 _4세 | 2023 년09 월_계 _5세 | 2023 년09 월_계 _6세 | 2023 년09 월_계 _7세 | ... | 2023 년09 월_계 _91 세 | 2023 년09 월_계 _92 세 | 2023 년09 월_계 _93 세 | 2023 년09 월_계 _94 세 | 2023 년09 월_계 _95 세 |
|--|-------------------------|--------------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|-----|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|
| 행정구역 | | | | | | | | | | | | | | | | |
| 서울특별시 (1100000000) | 9,407,540 | 9,407,540 | 38,101 | 41,599 | 43,518 | 44,893 | 48,097 | 51,107 | 56,215 | 64,176 | ... | 8,984 | 6,319 | 5,441 | 4,430 | 3,430 |
| 서울특별시 종로구 (1111000000) | 139,945 | 139,945 | 397 | 470 | 463 | 524 | 516 | 578 | 670 | 778 | ... | 172 | 146 | 110 | 93 | 86 |
| 서울특별시 종로구 청운 효자동 (1111051500) | 11,391 | 11,391 | 35 | 45 | 46 | 47 | 50 | 47 | 76 | 86 | ... | 14 | 14 | 7 | 8 | 7 |

데이터를 확인해 보면
숫자 사이에 쉼표(.)가 있습니다.
자료형을 확인해 보겠습니다.

Step 1) 데이터를 읽어 오기 (2/2)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)
df.dtypes
```

```
2021년11월_계_총인구수      object
2021년11월_계_연령구간인구수  object
2021년11월_계_0세          object
2021년11월_계_1세          object
2021년11월_계_2세          object
...
2021년11월_계_96세          object
2021년11월_계_97세          object
2021년11월_계_98세          object
2021년11월_계_99세          object
2021년11월_계_100세 이상    object
Length: 103, dtype: object
```

정수형 자료형이 아니라 객체(object) 입니다.
정수형 자료형으로 변환하기 전에
숫자 사이에 있는 쉼표(,)부터 제거하겠습니다.

코마 제거

```
import pandas as pd
```

```
df = pd.read_csv('age.csv', encoding='cp949', index_col=0)
```

```
df = df.replace(' ', '', regex=True)
```

```
df.head()
```

replace() 함수를 이용하면 전체 데이터 프레임에서
심표(.)를 한 번에 "으로 바꿀 수가 있습니다.

| | 2021년 11월_계 _총인구 수 | 2021년 11월_계 _간인구 수 | 2021 년11 월_계 _0세 | 2021 년11 월_계 _1세 | 2021 년11 월_계 _2세 | 2021 년11 월_계 _3세 | 2021 년11 월_계 _4세 | 2021 년11 월_계 _5세 | 2021 년11 월_계 _6세 | 2021 년11 월_계 _7세 | ... | 2021 년11 월_계 _91 세 | 2021 년11 월_계 _92 세 | 2021 년11 월_계 _93 세 | 2021 년11 월_계 _94 세 | 2021 년11 월_계 _95 세 | 2021 년11 월_계 _96 세 | 2021 년11 월_계 _97 세 | 2021 년11 월_계 _98 세 | 2021 년11 월_계 _99 세 | 20 년 ... |
|--|-----------------------------|-----------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|-----|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|----------------|
| 행정구역 | | | | | | | | | | | | | | | | | | | | | |
| 서울특별시 (1100000000) | 9520880 | 9520880 | 43492 | 45054 | 49318 | 51995 | 56210 | 63757 | 67763 | 66950 | ... | 8147 | 6779 | 5328 | 4048 | 2587 | 1862 | 1368 | 1274 | 816 | 20 |
| 서울특별시 종로구 (1111000000) | 145073 | 145073 | 486 | 475 | 557 | 576 | 664 | 773 | 848 | 859 | ... | 183 | 139 | | | | | | | | |
| 서울특별시 종로구 청운 효자동 (1111051500) | 12006 | 12006 | 43 | 48 | 58 | 54 | 80 | 74 | 96 | 93 | ... | 10 | 10 | | | | | | | | |
| 서울특별시 종로구 사직 동 (1111053000) | 9367 | 9367 | 38 | 27 | 40 | 46 | 55 | 67 | 74 | 71 | ... | 15 | 19 | 6 | 7 | 8 | 9 | 2 | 3 | 1 | |
| 서울특별시 종로구 삼청 동 (1111054000) | 2467 | 2467 | 7 | 5 | 3 | 10 | 14 | 6 | 17 | 8 | ... | 7 | 2 | 4 | 3 | 1 | 0 | 2 | 0 | 2 | |

5 rows x 103 columns

regex 옵션은 regular expression의 약자로,
정규 표현식으로 문자열이 완전히 일치하지 않더라도
문자열의 일부분만 치환하고 싶을 경우 True로 설정해줍니다.

이제 정수형 자료형으로
변환해 보겠습니다.

정수로 변환

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df.dtypes
```

```
2021년11월_계_총인구수      int64
2021년11월_계_연령구간인구수  int64
2021년11월_계_0세          int64
2021년11월_계_1세          int64
2021년11월_계_2세          int64
...
2021년11월_계_96세          int64
2021년11월_계_97세          int64
2021년11월_계_98세          int64
2021년11월_계_99세          int64
2021년11월_계_100세 이상    int64
Length: 103, dtype: object
```

객체(object)에서 정수형 자료형 "int64"로 변경되었습니다.

— 행정구역의 총 인구수가 0인 행 제거

```
# 전체 코드
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

# 콤마로 제거하고 정수로 변환
df = df.replace(',', '', regex=True)
df = df.apply(pd.to_numeric)

# 총 인구수가 0인 행을 삭제
df = df[df[df.columns[0]] != 0]
```

행정구역 이름이 '~~~출장소' 등으로 되어 있으면서
총 인구수가 0이 아닌 지역만을 추출해서 다시 df에 저장한다.

인구비율 계산

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

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

# 콤마로 제거하고 정수로 변환
df = df.replace(',', '', regex=True)
df = df.apply(pd.to_numeric)

# 총 인구수가 0인 행을 삭제
df = df[df.columns[0]] != 0]

# df = df.div(df['2023년09월_계_총인구수'], axis='index')
df = df.div(df[df.columns[0]], axis='index')
df.head(3)
```

㉔ 전체 데이터를 총 인구수로 나누어 비율로 변환한다.

✓ 0.5s Python

| | 2023년09월_계_총인구수 | 2023년09월_계_연령구간인구수 | 2023년09월_계_0세 | 2023년09월_계_1세 | 2023년09월_계_2세 | 2023년09월_계_3세 | 2023년09월_계_4세 | 2023년09월_계_5세 | 2023년09월_계_6세 | 2023년09월_계_7세 | 2023년09월_계_... | 2023년09월_계_91세 | 2023년09월_계_92세 |
|------------------------------|-----------------|--------------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|----------------|----------------|----------------|
| 행정구역 | | | | | | | | | | | | | |
| 서울특별시 (1100000000) | 1.0 | 1.0 | 0.004050 | 0.004422 | 0.004626 | 0.004772 | 0.005113 | 0.005433 | 0.005976 | 0.006822 | ... | 0.000955 | 0.000672 |
| 서울특별시 종로구 (1111000000) | 1.0 | 1.0 | 0.002837 | 0.003358 | 0.003308 | 0.003744 | 0.003687 | 0.004130 | 0.004788 | 0.005559 | ... | 0.001229 | 0.001043 |
| 서울특별시 종로구 청운효자동 (1111051500) | 1.0 | 1.0 | 0.003073 | 0.003950 | 0.004038 | 0.004126 | 0.004389 | 0.004126 | 0.006672 | 0.007550 | ... | 0.001229 | 0.001229 |

3 rows x 103 columns

필요 없는 열 제거

```
import pandas as pd
```

```
df = pd.read_csv('age.csv', encoding='cp949', index_col = 0)
df = df.replace(',', '', regex=True)
```

```
df = df.apply(pd.to_numeric)
```

```
# 총 인구가 0인 행을 삭제
```

```
df = df[df.columns[0]] != 0]
```

```
df = df.div(df['2023년09월_계_총인구수'], axis='index')
```

```
del df['2023년09월_계_총인구수'], df['2023년09월_계_연령구간인구수']
```

```
df.head(3)
```

✓ 0.5s

Python

⑥ 총 인구수와 연령 구간 인구수를 삭제한다.

| | 2023년 09월_계 _0세 | 2023년 09월_계 _1세 | 2023년 09월_계 _2세 | 2023년 09월_계 _3세 | 2023년 09월_계 _4세 | 2023년 09월_계 _5세 | 2023년 09월_계 _6세 | 2023년 09월_계 _7세 | 2023년 09월_계 _8세 | 2023년 09월_계 _9세 | ... | 2023년 09월_계 _91세 | 2023년 09월_계 _92세 |
|--|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----|------------------------|------------------------|
| 행정구역 | | | | | | | | | | | | | |
| 서울특별시 (1100000000) | 0.004050 | 0.004422 | 0.004626 | 0.004772 | 0.005113 | 0.005433 | 0.005976 | 0.006822 | 0.007074 | 0.007078 | ... | 0.000955 | 0.001229 |
| 서울특별시 종로구 (1111000000) | 0.002837 | 0.003358 | 0.003308 | 0.003744 | 0.003687 | 0.004130 | 0.004788 | 0.005559 | 0.005681 | 0.005981 | ... | 0.001229 | 0.001229 |
| 서울특별시 종로구 청운 효자동 (1111051500) | 0.003073 | 0.003950 | 0.004038 | 0.004126 | 0.004389 | 0.004126 | 0.006672 | 0.007550 | 0.007550 | 0.007462 | ... | 0.001229 | 0.001229 |

3 rows x 101 columns

LESSON 02

pandas로 연령별 인구구조 유사지역 시각화



— 관심 지역의 인구구조 저장 (1/2)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col = 0)
df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)

# 총 인구가 0인 행을 삭제
df = df[df.columns[0]] != 0]

df = df.div(df['2023년09월_계_총인구수'], axis='index')
del df['2023년09월_계_총인구수'], df['2023년09월_계_연령구간인구수']

name = input('원하는 지역의 이름을 입력해주세요 : ') #2. 지역 이름 입력
a = df.index.str.contains(name) #3. 해당 행을 찾아서 해당 지역의 인구 구조를 저장
df2 = df[a]
df2
```

✓ 11.9s

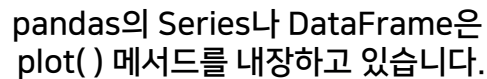
Python

| | 2023년 09월_계 _0세 | 2023년 09월_계 _1세 | 2023년 09월_계 _2세 | 2023년 09월_계 _3세 | 2023년 09월_계 _4세 | 2023년 09월_계 _5세 | 2023년 09월_계 _6세 | 2023년 09월_계 _7세 | 2023년 09월_계 _8세 | 2023년 09월_계 _9세 | ... | 2023년 09월_계 _91세 | 2023년 09월_계 _92세 |
|---------------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----|------------------------|------------------------|
| 행정구역 | | | | | | | | | | | | | |
| 서울특별시 구로구 신도 림동 (1153051000) | 0.006206 | 0.007016 | 0.007128 | 0.007296 | 0.008582 | 0.007631 | 0.008442 | 0.010259 | 0.011489 | 0.010846 | ... | 0.000559 | 0.000559 |

df.index.str.contains() 함수는
데이터 프레임의 인덱스 문자열로부터,
원하는 문자열이 포함된 행을 찾아냅니다.

```
name = input('원하는 지역의 이름을 입력해주세요 : ') #2. 지역 이름 입력
```

```
# plt.rc('font', family='Malgun Gothic')
plt.rcParams['font.family'] = 'Malgun Gothic'
df2.T.plot()
plt.show()
```



— 관심 지역과 유사 인구구조 지역 시각화 (1/4)

```
# A의 인구 비율에서 B의 인구 비율을 뺀다.
x = df.sub(df2.iloc[0], axis='columns') # axis=1 이라고 적어도 됩니다.
x.head(3)
```

| | 2021년11 월_계_0 세 | 2021년11 월_계_1 세 | 2021년11 월_계_2 세 | 2021년11 월_계_3 세 | 2021년11 월_계_4 세 | 2021년11 월_계_5 세 | 2021년11 월_계_6 세 | 2021년11 월_계_7 세 | 2021년11 월_계_8 세 | 2021년11 월_계_9 세 | ... | 2021년 11월_계 _91세 | 2021년 11월_계 _92세 | 2021년 11월_계 _93세 |
|--|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----|------------------------|------------------------|------------------------|
| 행정구역 | | | | | | | | | | | | | | |
| 서울특별시 (1100000000) | -0.003051 | -0.002775 | -0.003690 | -0.002992 | -0.003022 | -0.003786 | -0.004728 | -0.004174 | -0.003284 | -0.003071 | ... | 0.000466 | 0.000267 | 0.000198 |
| 서울특별시 종로구 (1111000000) | -0.004269 | -0.004233 | -0.005031 | -0.004482 | -0.004349 | -0.005154 | -0.006000 | -0.005284 | -0.004664 | -0.004193 | ... | 0.000872 | 0.000513 | 0.000424 |
| 서울특별시 종로구 청운 효자동 (1111051500) | -0.004037 | -0.003510 | -0.004039 | -0.003955 | -0.002262 | -0.004319 | -0.003849 | -0.003460 | -0.003486 | -0.000877 | ... | 0.000444 | 0.000388 | 0.000471 |

3 rows × 101 columns

— 관심 지역과 유사 인구구조 지역 시각화 (2/4)

```
import numpy as np

y = np.power(x, 2)
z = y.sum(axis='columns')
z
```

✓ 0.0s

행정구역

| | |
|-------------------------------|----------|
| 서울특별시 (1100000000) | 0.000551 |
| 서울특별시 종로구 (1111000000) | 0.001098 |
| 서울특별시 종로구 청운효자동 (1111051500) | 0.000512 |
| 서울특별시 종로구 사직동 (1111053000) | 0.000730 |
| 서울특별시 종로구 삼청동 (1111054000) | 0.001673 |
| ... | |
| 제주특별자치도 서귀포시 서홍동 (5013058000) | 0.000644 |
| 제주특별자치도 서귀포시 대륜동 (5013059000) | 0.000337 |
| 제주특별자치도 서귀포시 대천동 (5013060000) | 0.000328 |
| 제주특별자치도 서귀포시 중문동 (5013061000) | 0.000305 |
| 제주특별자치도 서귀포시 예래동 (5013062000) | 0.002119 |

Length: 3862, dtype: float64

— 관심 지역과 유사 인구구조 지역 시각화 (3/4)

```
i = z.sort_values().index[:6]  
i
```

✓ 0.0s

```
Index(['서울특별시 구로구 신도림동(1153051000)', '경기도 하남시 (4145000000)',  
      '경기도 남양주시 별내동(4136057000)', '서울특별시 영등포구 문래동(1156060500)',  
      '충청남도 아산시 (4420000000)', '서울특별시 영등포구 신길제7동(1156069000)'],  
      dtype='object', name='행정구역')
```

sort_values() 함수를 활용하여 오름차순으로 정렬하고,
index[:6]를 활용하여 차이가 0인 관심지역과 가장 작은 지역 5곳을 찾습니다.

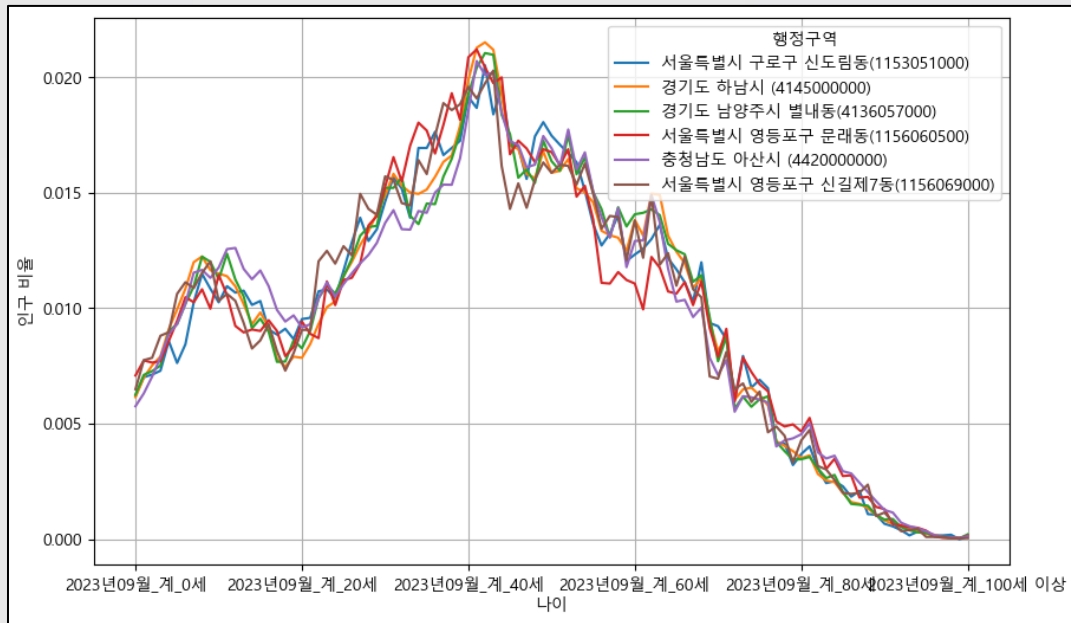
— 관심 지역과 유사 인구구조 지역 시각화 (4/4)

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

plt.rc('font', family='Malgun Gothic')
plt.rcParams['axes.unicode_minus'] = False
df.loc[i].T.plot(figsize=(10, 6))
plt.xlabel('나이')
plt.ylabel('인구 비율')
plt.grid(True)
plt.show()
```

loc은 location의 약자로 인덱스를 기준으로
행 데이터를 읽기 위해서 사용됩니다.

[참고] iloc은 행 번호를 기준으로
행 데이터를 읽기 위해서 사용됩니다.



전체 코드

```
# 전체 코드
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

# 콤마로 제거하고 정수로 변환
df = df.replace(',', '', regex=True)
df = df.apply(pd.to_numeric)

# 총 인구가 0인 행을 삭제
df = df[df.columns[0]] != 0]

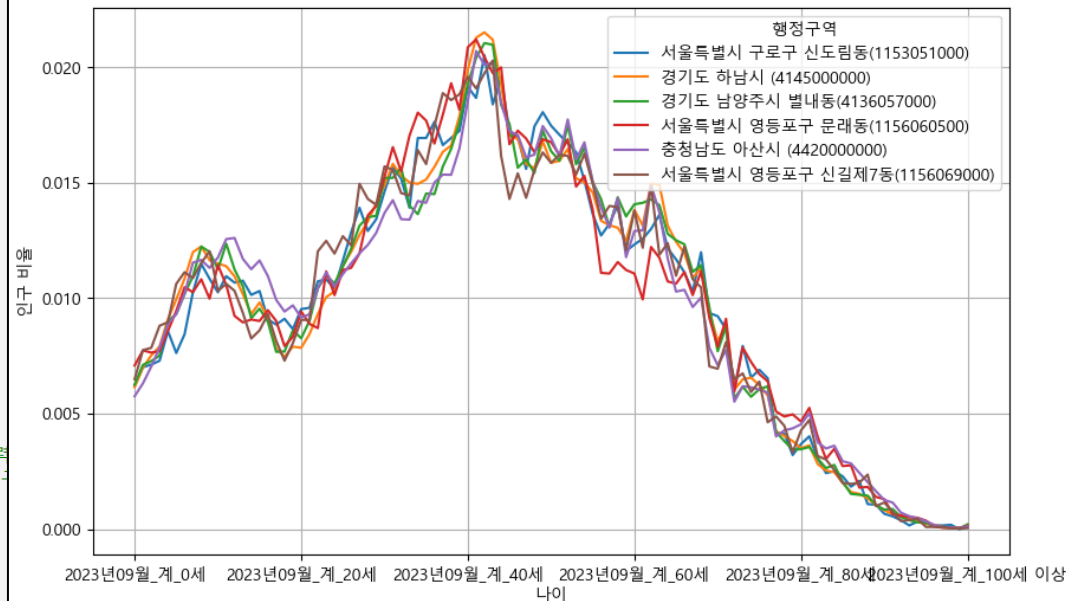
# df = df.div(df['2023년09월_계_총인구수'], axis='index')
df = df.div(df[df.columns[0]], axis='index')
del df['2023년09월_계_총인구수'], df['2023년09월_계_연령구간인구수']

name = input('원하는 지역의 이름을 입력해주세요 : ') # 원하는 지역 이름 입력
df2 = df[df.index.str.contains(name)] #3. 해당 행을 찾아서 해당 지역의 인

# 특정 지역의 인구를 빼고
x = df.sub(df2.iloc[0], axis='columns')
y = np.power(x, 2) # 차의 제곱을 연산
z = y.sum(axis='columns') # 결과는 시리즈로 인덱스가 행정 구역
i = z.sort_values().index[:6] # 차의 제곱이 가장 작은 6개 선택

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.rcParams['axes.unicode_minus'] = False

df.columns = range(101)
df.loc[i].T.plot(figsize=(10, 6))
plt.xlabel('나이')
plt.ylabel('인구 비율')
plt.grid(True)
plt.show()
```



스타일 설정, 다른 지역 유사 검색

```
# 전체 코드
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

# 콤마로 제거하고 정수로 변환
df = df.replace(',', '', regex=True)
df = df.apply(pd.to_numeric)

# 총 인구가 0인 행을 삭제
df = df[df.columns[0]] != 0]

# df = df.div(df['2023년09월_계_총인구수'], axis='index')
df = df.div(df[df.columns[0]], axis='index')
del df['2023년09월_계_총인구수'], df['2023년09월_계_연령구간인구수']

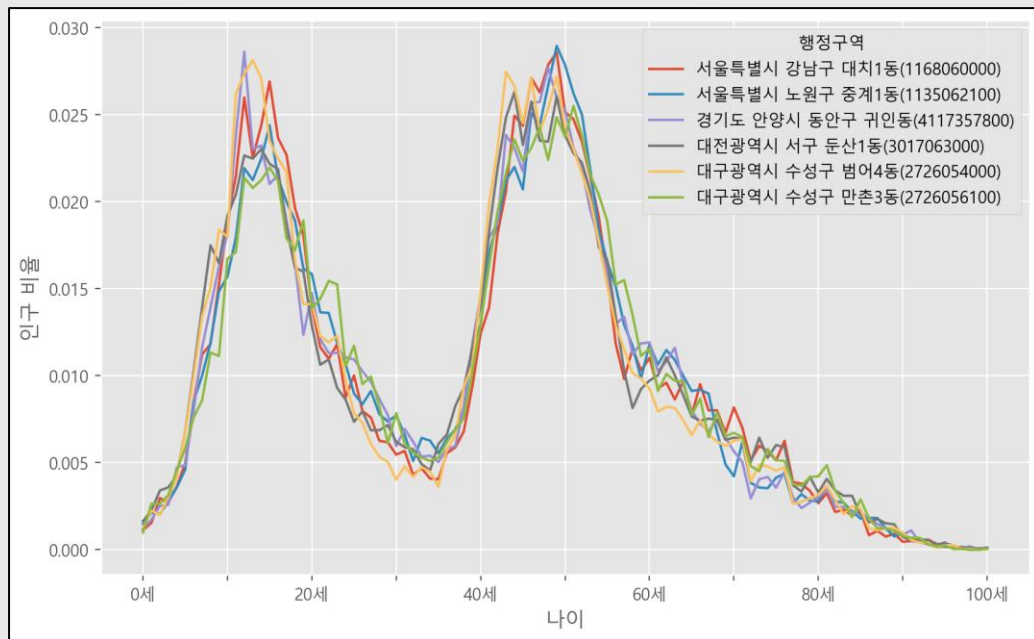
name = input('원하는 지역의 이름을 입력해주세요 : ') # 원하는 지역 이름 입력
df2 = df[df.index.str.contains(name)] #3. 해당 행을 찾아서 해당 지역의 인구 구조를 저장

# 특정 지역의 인구를 빼고
x = df.sub(df2.iloc[0], axis='columns')
y = np.power(x, 2) # 차의 제곱을 연산
z = y.sum(axis='columns') # 결과는 시리즈로 인덱스가 행정 구역
i = z.sort_values().index[:6] # 차의 제곱이 가장 작은 6개 선택

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.rcParams['axes.unicode_minus'] = False

plt.style.use('ggplot')
df.columns = [f'{i}세' for i in range(101)]
df.loc[i].T.plot(figsize=(10, 6))
plt.xlabel('나이')
plt.ylabel('인구 비율')
plt.grid(True)

plt.xticks(range(0, 101, 10))
plt.show()
```



SUMMARY

학습정리



연령별 인구 데이터를 pandas로 읽어와 전처리 및 기본 연산

```
df = pd.read_csv('age.csv', encoding='cp949', index_col=0)
df = df.replace(',', '', regex=True)
df = df.apply(pd.to_numeric)
df = df[df[df.columns[0]] != 0]
df = df.div(df[df.columns[0]], axis='index')
```

연령별 인구구조 분석을 위한 관심 지역 선정

```
name = input('원하는 지역의 이름을 입력해주세요 : ') #2. 지역 이름 입력
a = df.index.str.contains(name) #3. 해당 행을 찾아서 해당 지역의 인구 구조를 저장
df2 = df[a]
```

관심 지역과 유사한 연령별 인구구조 지역을 찾기 위한 계산

```
x = df.sub(df2.iloc[0], axis='columns')
y = np.power(x, 2)
z = y.sum(axis='columns')
```

관심지역과 유사한 연령별 인구구조 지역 5개 데이터 시각화

```
i = z.sort_values().index[:6] # 차의 제공이 가장 작은 6개 선택
df.loc[i].T.plot(figsize=(10, 6))
```

