

2023 semi PROJECT

통신 데이터를 활용한
서비스 패턴 분석 및 우선순위 기반 광고 추천 시스템



목차

- 01** 프로젝트 선정 배경 및 목표
- 02** 팀 구성 및 역할
- 03** 수행 절차 및 방법
- 04** 데이터 전처리
- 05** 현황 분석 및 데이터 시각화
- 06** 시스템 모델링
- 07** 결론 및 향후 과제
- 08** 프로젝트 소감

01

프로젝트 선정 배경 및 목표

프로젝트 선정 배경

다양한 공공 데이터 중 분석이 용이하고 일상 생활과 밀접한 관련이 있는 통신 기반 데이터를 통하여 실제 적용 가능한 결과를 도출하고자 선정함.

통신 데이터에는 기지국과 스마트폰 기반으로 수집된 다양한 데이터가 존재하고 특히 다양한 서비스의 사용일수가 존재함.

단순한 데이터 분석에서 벗어나 다양한 Feature를 활용한 개인화된 서비스 패턴 분석 및 광고 추천 시스템을 구현하여 실제 사용이 가능하고 즉시 업무에 사용할 수 있는 실용성을 추구함.

프로젝트 목표

[시스템 설계]

- 전처리를 통해 구축된 Data를 로컬 서버 저장 및 웹 서비스와 연동
- 고객 리소스 감소 및 신속하고 지속적인 최신화가 가능하도록 제작

[시스템 1. 고객별 맞춤형 광고 알고리즘]

- 개인화(자치구, 성별, 연령대) 설정 및 타겟 선정
- 타겟 고객(개인화) 입력 시 7개 서비스 대한 광고 우선순위 추천

[시스템 2. 서비스별 타겟 고객 및 광고 추천 시스템]

- 기업 입장에서 광고 희망 분야 입력 시 타겟 고객 추천
- 타겟 고객 추천에 더해 개인화된 서비스별 상관관계 통해 타 광고 우선순위도 추천

프로젝트 사용 시스템

프로그램	 [Excel]	 [Jupyter notebook]	 [VScode]
언어 및 포맷	 [Python]	 [HTML]	 [CSS]
패키지	 [NumPy]	 [Pandas]	
라이브러리	 [Matplotlib]	 [Seaborn]	 [Folium]
프레임워크			

02

팀 구성 및 역할

최현민	정혜원	김수연	김 현	최수정
조장	부조장	팀원	팀원	팀원
<ul style="list-style-type: none">- 프로젝트 총괄- 코드 및 PPT 종합- 기획서 작성- 업무수행흐름도 작성- 시각화 코드 작성- 가중치 코드 작성	<ul style="list-style-type: none">- 서비스 개요 작성- 데이터 전처리 및 시각화 작업- 가중치 산정- 추천 시스템 서비스 구현- Django 웹서비스 구현- 추천 시스템 설명파트 PPT 제작 및 발표	<ul style="list-style-type: none">- 데이터 전처리- 혼황분석 시각화- 서비스별 추세분석- 가중치 반영 코드 작성- 가중치 부분 PPT 제작 및 발표	<ul style="list-style-type: none">- 기획안 작성- 데이터 전처리 및 혼황분석, 시각화- 추천시스템 데이터 모델링- Django 웹 서비스 구축- Django 파트 PPT 제작 및 발표	<ul style="list-style-type: none">- 데이터 전처리- 혼황분석 시각화- 서비스별 추세분석- 데이터 시각화 PPT 제작 및 발표

개인 역할 상세

김현 조원

역 할	내 용
기획안 작성	기획안을 수정하고 WBS를 토대로 기획안을 작성함
데이터 전처리	기존 1월부터 9월까지의 데이터를 총합하고 시각화 전에 데이터를 전 처리함
현황 분석 및 시각화	Bar plot, 지도 그래프, 히트맵 시각화 및 시각화된 데이터들로 현황을 분석함
추천시스템 데이터 모델링	시나리오1과 시나리오2에 해당하는 데이터
Django 웹서비스 구축	Django로 해당하는 서비스를 웹사이트에서 구현할수 있도록 코드를 만들고 연동을 함
Django 파트 PPT제작 및 발표	담당했던 Django부분에 해당하는 PPT를 제작하고 발표를 함

03

수행 절차 및 방법

전체 수행 일정

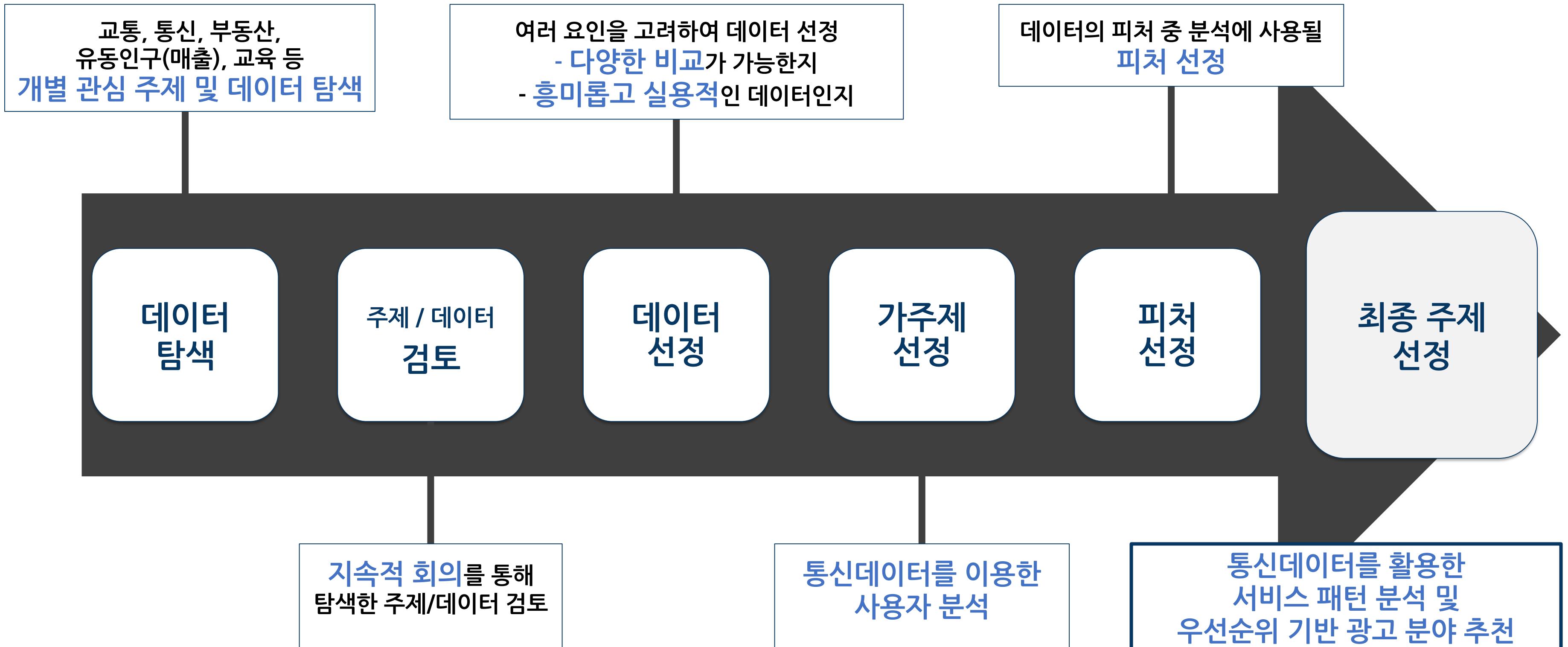
전체 프로젝트 수행 기간	시작일	종료일	기간(일)
	2023.11.14	2023.12.04	21

구분	일자	시작일	종료일	기간(일)
1	주제 선정	2023.11.14	2023.11.15	2
2	데이터 수집 및 탐색적 분석	2023.11.14	2023.11.16	3
3	현황 분석 및 데이터 시각화	2023.11.15	2023.11.21	7
4	데이터 모델링(추천 시스템)	2023.11.20	2023.11.30	11
5	프로젝트 발표 준비	2023.11.29	2023.12.04	6

상세 수행 일정

구분	공정	완료율	기간(일)	시작 날짜	완료 날짜	담당자	1일	2일	3일	4일	5일	6일	7일	8일	9일	10일	11일	12일	13일	14일	15일	16일	17일	18일	19일	20일	21일
							11/14	11/15	11/16	11/17	11/18	11/19	11/20	11/21	11/22	11/23	11/24	11/25	11/26	11/27	11/28	11/29	11/30	12/1	12/2	12/3	12/4
프로젝트 관리	프로젝트 수행계획 수립	100%	2	2023-11-14	2023-11-15	전인원																					
	프로젝트 착수 보고(Kick-off)	100%	2	2023-11-16	2023-11-17	팀장																					
	중간보고 및 확인	0%	1	2023-11-27	2023-11-27	팀장																					
	종료보고 및 확인	0%	1	2023-12-04	2023-12-04	팀장																					
프로젝트 수행	1. 주제 선정																										
	주제 탐색 리서치	100%	1	2023-11-14	2023-11-14	전인원																					
	세부 요구사항 정의	100%	1	2023-11-15	2023-11-15	전인원																					
	2. 데이터 수집 및 탐색적 분석																										
	주제 관련 상세정보 리서치	100%	1	2023-11-14	2023-11-14	전인원																					
	서울시 공공데이터 수집 및 탐색적 분석	100%	2	2023-11-14	2023-11-15	전인원																					
	서울시 통신데이터 수집 및 탐색적 분석	100%	2	2023-11-14	2023-11-15	전인원																					
	현황 분석 피쳐 선별	100%	2	2023-11-15	2023-11-16	전인원																					
	모델링 피쳐 선별	100%	2	2023-11-15	2023-11-16	전인원																					
	3. 현황 분석 및 데이터 시각화																										
	자치구별 연령대 및 성별 전처리	100%	6	2023-11-15	2023-11-20	개인별																					
	현황분석 시각화(bar plot) - 인구수(자치구, 성별, 연령대)	100%	6	2023-11-15	2023-11-20	개인별																					
	현황분석 시각화(지도) - 자치구별 평균 연령대(종합, 남성, 여성)	100%	6	2023-11-15	2023-11-20	개인별																					
	현황분석 시각화(히트맵) - 서비스별 상관관계(연령대)	100%	6	2023-11-15	2023-11-20	개인별																					
	시각화 코드 리뷰 및 종합	100%	1	2023-11-20	2023-11-20	전인원																					
	개요 작성	100%	1	2023-11-20	2023-11-20	최현민, 정혜원																					
	외부 검토(강사님)	100%	1	2023-11-21	2023-11-21	전인원																					
	4. 데이터 모델링(추천 시스템)																										
	시스템 1: 고객별 맞춤형 광고 알고리즘	100%	5	2023-11-20	2023-11-24	전인원																					
	시스템 2: 서비스별 타겟 고객 및 광고 추천 시스템	100%	5	2023-11-20	2023-11-24	전인원																					
	서비스별 추세분석	100%	5	2023-11-24	2023-11-28	김수연, 최수정, 최현민																					
	웹 서비스 구축	100%	5	2023-11-24	2023-11-28	김현, 정혜원																					
	자체 코드리뷰 1차	100%	1	2023-11-24	2023-11-24	전인원																					
	자체 코드리뷰 2차	100%	2	2023-11-27	2023-11-28	전인원																					
	코드 종합	100%	8	2023-11-22	2023-11-29	최현민																					
	개요 작성(파트별)	100%	9	2023-11-22	2023-11-30	전인원																					
	5. 프로젝트 발표 준비																										
	프로젝트 포트폴리오 작성(파트별)	100%	5	2023-11-29	2023-12-03	전인원																					
	프로젝트 발표용 PPT 제작	100%	4	2023-11-30	2023-12-03	전인원																					
	포트폴리오 및 발표용 PPT 종합	100%	3	2023-12-01	2023-12-03	최수정																					
	발표 연습	100%	3	2023-12-01	2023-12-03	전인원																					
	프로젝트 발표	100%	1	2023-12-04	2023-12-04	전인원																					

주제 및 데이터 선정 과정



데이터 출처 및 소개

[서울 시민생활 데이터] (출처 : 서울 열린데이터 광장)

- ▶ 서울시와 SK텔레콤이 공공빅데이터와 통신데이터 [가명결합](#)을 통해 추정한 행정동 단위 성, 연령별 1인가구와 [서울시민의 생활특성 정보](#)
- ▶ 2022년 1월부터 2023년 9월까지 월별 데이터가 존재하며 그 중 2023년 1월부터 2023년 9월까지의 데이터만 사용하여 프로젝트를 진행함

내 용	
공간단위	25개 자치구, 426개 행정동
작성정보	10개 관심집단수, 29개 통신정보
작성단위	성, 연령, 1인가구, 시민전체
작성주기	월 단위
데이터 사용 범위	2023년 1월 ~ 9월 데이터

사용피쳐	설 명
자치구	25개 자치구
행정동	426개 행정동
성별	남, 여
연령대	20세부터 75세까지 5살 단위
총인구수	자치구, 행정동, 성별, 연령대에 따른 총인구수
서비스 사용일수	게임
	금융
	배달
	쇼핑
	동영상/방송
	유튜브
	넷플릭스

- 최근 3개월의 월 평균 해당 서비스 사용량
- 사용량은 각 서비스 이용일수의 합으로 정의
- 집계 방식: 최근 3개월 총 게임 서비스 사용일수의 합 / 3개월

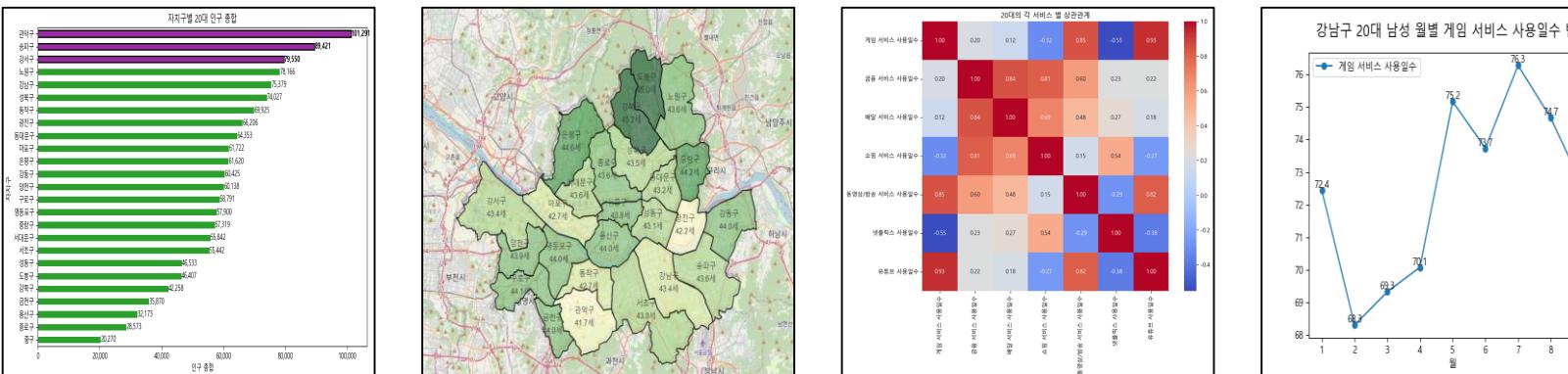
- 최근 3개월의 월 평균 해당 서비스 사용일수
- 산출방법: 최근 3개월 총 유튜브 사용일수/3개월
- 해당 날짜의 사용 여부는 서비스의 접속 여부로 계산하며 원천자료를 Z-score로 변환하여 제공

04

데이터 전처리

Feature 엔지니어링

자치구	성별	월	연령대	총인구수	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
강남구	F	1	20	39603.42	1734.02	3667.23	677.89	5974.04	924.27	-1.9	5.78
강남구	F	2	20	39592.44	1649.39	3487.39	651.63	5642.07	859.61	-1.92	5.88
강남구	F	3	20	39594.47	1693.72	3654.84	671.47	5843.9	905.19	-1.13	5.58
강남구	F	4	20	39611.88	1735.03	3935.58	725.22	6107.88	1227.61	-1.63	5.68
강남구	F	5	20	39597.88	1859.37	4467.36	798.2	6832.55	1619.83	-2.05	5.82
강남구	F	6	20	39597.16	1833.17	4637.32	755.05	6793.74	1583.65	-5.75	5.71
강남구	F	7	20	39600	1861.88	4889.95	722.33	7089.12	1653.58	-5.59	5.56
강남구	F	8	20	39636	1787.67	5043.21	702.18	7125.21	1713.2	-5.08	5.55
강남구	F	9	20	39694	1714.87	5178.22	693.52	7232.24	1596.58	-5.15	5.61
강남구	F	1	30	50923.01	1706.59	4763.75	694.29	8349.38	1111.6	-4.31	1.32



자치구	성별	연령대	월	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
강남구	F	20	1	-0.622545	0.147355	0.656832	0.277327	0.145569	0.817553	0.979137

▶ 데이터 종합

2023년 1월~9월 데이터를 1개의 파일로 종합

▶ 그룹화 및 개인화

- 그룹화 : 자치구, 성별, 연령대로 구분 > 특성을 나타내는 최소단위로 지정
- 개인화 : 총 300개 단위(행정동 > 자치구 / 성별 > 남,녀 / 연령대 > 10세 단위)

▶ 사용 피쳐 선정

- 원 데이터 중 사용 피쳐 선정 및 추출(자치구, 성별, 연령대, 총인구수, 서비스 7개)
- 사용일수(가나다 순) / 영상서비스(동영상/방송, 넷플릭스, 유튜브) 순

▶ 단위 통일

- 원 데이터 상 z-value(넷플릭스, 유튜브 서비스 사용일수)와의 통일을 위해
나머지 서비스 또한 StandardScaler를 이용해서 정규화 실행

▶ 현황분석 및 시각화

- 자치구별 인구 총합 > 바 플롯
- 자치구별 평균 연령대 > 지도 표시
- 서비스별 시계열(월) 분석 > 꺾은선 그래프
- 성별, 연령대 별 서비스 간의 상관관계 분석 > 히트맵

▶ 가중치 산정 및 반영

개인화 데이터 내 7개 서비스의 데이터 사용일수에 따른 가중치를 산정하여
z-value 값에 가중치 반영하여 최종 순위 산정

Code_데이터 전처리

1. 데이터 불러오기

2023년 월별 데이터 병합

```
# 파일 경로
# file_path = "/content/drive/MyDrive/data/2023.{}월_29개 통신정보.xlsx" # 코랩용
file_path = "./data/2023.{}월_29개 통신정보.xlsx" # 로컬용

# 1월부터 9월까지의 데이터를 읽어오는 리스트 생성
dataframes = [pd.read_excel(file_path.format(month)) for month in range(1, 10)]

# 각 데이터프레임을 하나로 합치기
combined_df = pd.concat(dataframes)

# 성별 전처리
combined_df['성별'] = combined_df['성별'].replace({1:'M', 2:'F'}) # 1->M 2->F로 변경
# 숫자 데이터만 포함하는 열 선택
# numeric_df = combined_df.select_dtypes(include=[np.number])

# 동일한 위치의 셀들을 뺏어 평균을 계산, 계층형 인덱스 사용 안함
average_df = combined_df.groupby(['자치구', '행정동', '성별', '연령대']).mean().reset_index()
```

1) 연령대 10단위 지정 함수

```
def merge_age(age):
    return int((age // 10) * 10)
```

2) 필요 col만 고르기

```
average_df_col = average_df[
    ['자치구', '성별', '연령대', '총인구수',
     '게임 서비스 사용일수', '금융 서비스 사용일수', '배달 서비스 사용일수', '쇼핑 서비스 사용일수', # 가나다순
     '동영상/방송 서비스 사용일수', '넷플릭스 사용일수', '유튜브 사용일수']] # 여가관련(3개)
```

3) 데이터 성별 구분

```
average_df_col_M = average_df_col[average_df_col['성별'] == 'M'] #남자
average_df_col_F = average_df_col[average_df_col['성별'] == 'F'] #여자
```

2. 자치구 평균 연령대

1) 남녀 통합

```
data = pd.read_excel('./data/average.xlsx')
# data = average_df_col.copy()

data['자치구 연령대 총합'] = data['연령대'] * data['총인구수']
total = data.groupby(['자치구'])['총인구수'].sum()

average_age_total = pd.DataFrame()
average_age_total['평균 연령대'] = data.groupby(['자치구'])['자치구 연령대 총합'].sum() / total
total = np.ceil(total).astype(int) # 총인구수 정수올림

average_age_total['총인구수'] = total
average_age_total = average_age_total.reset_index()
```

2) 남녀 구분

```
data = pd.read_excel('./data/average.xlsx')
# data = average_df_col.copy()

data['자치구 성별 연령대 총합'] = data['연령대'] * data['총인구수']
total = data.groupby(['자치구', '성별'])['총인구수'].sum()

average_age_sex = pd.DataFrame()
average_age_sex['평균 연령대'] = data.groupby(['자치구', '성별'])['자치구 성별 연령대 총합'].sum() / total
total = np.ceil(total).astype(int) # 총인구수 정수올림

average_age_sex['총인구수'] = total
average_age_sex = average_age_sex.reset_index()
```

Code_데이터 전처리

3. 자치구 연령대별 총인구수

1) 남녀 통합

```
# data = pd.read_excel('./average.xlsx')
data = average_df_col.copy()

age_most_region_total = pd.DataFrame()
age_most_region_total['총인구수'] = data.groupby(['자치구', '연령대'])['총인구수'].sum().apply(np.ceil).astype(int)
age_most_region_total = age_most_region_total.reset_index().sort_values(by=['자치구', '연령대'])

# 연령대 10단위 묶음
age_most_region_total['연령대'] = age_most_region_total['연령대'].apply(merge_age)
age_most_region_total_10 = age_most_region_total.groupby(['자치구', '연령대'])['총인구수'].sum().reset_index()
```

2) 남녀 구분

```
# data = pd.read_excel('./average.xlsx')
data = average_df_col

age_most_region_sex = pd.DataFrame()
age_most_region_sex['총인구수'] = data.groupby(['자치구', '성별', '연령대'])['총인구수'].sum().apply(np.ceil).astype(int)
age_most_region_sex = age_most_region_sex.reset_index().sort_values(by=['자치구', '성별', '연령대'])
```

연령대 10단위 묶음

```
age_most_region_sex['연령대'] = age_most_region_sex['연령대'].apply(merge_age)
age_most_region_sex_10 = age_most_region_sex.groupby(['자치구', '성별', '연령대'])['총인구수'].sum().reset_index()

age_most_region_sex_10
# age_most_region_sex_10.to_excel('./age_most_region_sex_10.xlsx')
```

4. 연령대 별 총인구수 확인

1) 총인구수 대비 연령대 비율 확인 후 10% 미만의 데이터(70대)는 사용하지 않음

```
data = age_most_region_total_10
df_pop = pd.DataFrame()
df_pop['총인구수'] = data.groupby('연령대')['총인구수'].sum()
df_pop['비율'] = df_pop['총인구수'] / df_pop['총인구수'].sum() * 100
df_pop['총인구수'] = df_pop['총인구수'].map('{:.0f}'.format)
```

총인구수 col 삭제

```
average_df_col_drop_pop = average_df_col.drop(['총인구수'], axis=1)
# average_df_col_drop_pop['연령대'] = average_df_col_drop_pop['연령대'].apply(merge_age)
# average_df_col_drop_pop = average_df_col_drop_pop.groupby(['자치구', '성별', '연령대']).mean()
```

가중치 산정 및 반영

자치구	성별	연령대	월	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
0 강남구	F	20	1	1734.02	3667.23	677.89	5974.04	924.27	-1.90	5.78
1 강남구	F	20	2	1649.39	3487.39	651.63	5642.07	859.61	-1.92	5.88
2 강남구	F	20	3	1693.72	3654.84	671.47	5843.90	905.19	-1.13	5.58
3 강남구	F	20	4	1735.03	3935.58	725.22	6107.88	1227.61	-1.63	5.68
4 강남구	F	20	5	1859.37	4467.36	798.20	6832.55	1619.83	-2.05	5.82

[가중치 전처리]

▶ 가중치를 주는 이유?

단순 z-value에 기반한 추천이 아닌 [가중치를 부여](#)하여 보다 더 정확한 추천시스템을 개발하기 위함

▶ 과정

1. raw 파일에 '월' column 추가
2. 1~9월 파일을 하나의 dataframe으로 병합
3. 성별을 M,F로 변환하고, 연령대를 10단위로 변경, 사용 column만 지정하는 전처리 함수 생성
4. [자치구, 성별, 연령대, 월] 별로 묶어서 각 서비스 사용일수의 합계 구하기

자치구	성별	연령대	월	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
0 강남구	F	20	1	0.983431	0.847129	0.953657	0.916877	0.688411	0.566225	1.016611
1 강남구	F	20	2	0.935434	0.805586	0.916714	0.865927	0.640251	0.572185	1.034200
2 강남구	F	20	3	0.960575	0.844267	0.944625	0.896904	0.674200	0.336755	0.981434
3 강남구	F	20	4	0.984004	0.909118	1.020241	0.937418	0.914344	0.485762	0.999023
4 강남구	F	20	5	1.054522	1.031959	1.122909	1.048639	1.206475	0.610927	1.023647

[가중치 산정]

▶ 가중치 산정 방식

가중치 = 각 서비스의 해당 월 사용일수 / 월별 총 평균(1~9월) 사용일수

▶ 과정

1. 새로운 df 생성 -> for문을 이용하여 가중치를 구한 후 삽입
2. 각 [자치구, 성별, 연령대]의 월별 총 평균 구하기
3. 서비스 별로 해당 월 사용일수 / 월별 총 평균을 구해서 가중치 생성

가중치 산정 및 반영

자치구	성별	연령대	월	게임 서비스 사용일수_x	금융 서비스 사용일수_x	배달 서비스 사용일수_x	쇼핑 서비스 사용일수_x	동영상/방송 서비스 사용일수_x	넷플릭스 사용일수_x	유튜브 사용일수_x	게임 서비스 사용일수_y
강남구	F	20	1	0.983431	0.847129	0.953657	0.916877	0.688411	0.566225	1.016611	-0.633034
강남구	F	20	2	0.935434	0.805586	0.916714	0.865927	0.640251	0.572185	1.034200	-0.633034

[서비스_X : 가중치, 서비스_Y : 기존 Z_value]

자치구	성별	연령대	월	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
강남구	F	20	1	-0.622545	0.147355	0.656832	0.277327	0.145569	0.817553	0.979137
강남구	F	20	2	-0.592161	0.140129	0.631388	0.261917	0.135385	0.826159	0.996077

[가중치 적용 후의 Z_value]

[가중치 반영]

▶ 방식

가중치 = 새로 산출한 가중치 * 원래 구해 놓은 z-value

▶ 과정

1. 가중치 df와 z-value df를 merge 함수를 이용하여 병합
2. 새로운 df 생성
3. 병합된 df의 [자치구, 성별, 연령대, 월] column을 새로운 df에 삽입
4. 각 서비스별 z-value와 가중치를 곱한 column 생성

Code_가중치 전처리 / 산정 / 반영

```
# 가중치 전처리

# 성별 변환
def preprocessing(df):
    df['성별'].replace({1:'M', 2:'F'}, inplace=True)

# 10단위 연령대로 적용
df['연령대'] = df['연령대']//10 * 10

df = df.reset_index()

# 사용할 columns
df = df[['자치구','성별','연령대','월','게임 서비스 사용일수','금융
서비스 사용일수','배달 서비스 사용일수','쇼핑 서비스 사용일수','동
영상/방송 서비스 사용일수','넷플릭스 사용일수','유튜브 사용일수']]
return df

month = preprocessing(combined_df)
month = month.groupby(['자치구', '성별', '연령대', '월']).sum().reset_index().sort_values(by=['자치구', '성별', '연령대', '월'])
```

```
# 가중치 산정

weight = pd.DataFrame()
service_list = ['게임 서비스 사용일수','금융 서비스 사용일수',
'배달 서비스 사용일수','쇼핑 서비스 사용일수', '동영상/방송 서비스
사용일수','넷플릭스 사용일수','유튜브 사용일수']

# 월별 총 평균을 월 데이터로 나눠주기
for i in range(300):
    case = month.iloc[0 + 9*i : 9 + 9*i]
    region = case.loc[9*i, '자치구']
    sex = case.loc[9*i, '성별']
    age = case.loc[9*i, '연령대']
    for name in service_list:
        df = case[f'{name}']
        case[f'{name}'] = df / df.mean()

# 새로운 데이터프레임으로 생성
weight = pd.concat([weight, case])
```

```
# 가중치 반영

z_value = region_sex_age_services
merge_left = pd.merge(weight, z_value, how='left', on=
['자치구','연령대','성별'])

# 새로운 데이터프레임 생성
service_weight = pd.DataFrame()
service_weight[['자치구','성별','연령대','월']] = merge_left[
['자치구','성별','연령대','월']]
service_list = ['게임 서비스 사용일수','금융 서비스 사용일수',
'배달 서비스 사용일수','쇼핑 서비스 사용일수', '동영상/방송 서비스
사용일수','넷플릭스 사용일수','유튜브 사용일수']

for service in service_list:
    service_weight[f'{service}'] = merge_left[f'{service}_x'] *
merge_left[f'{service}_y']
```

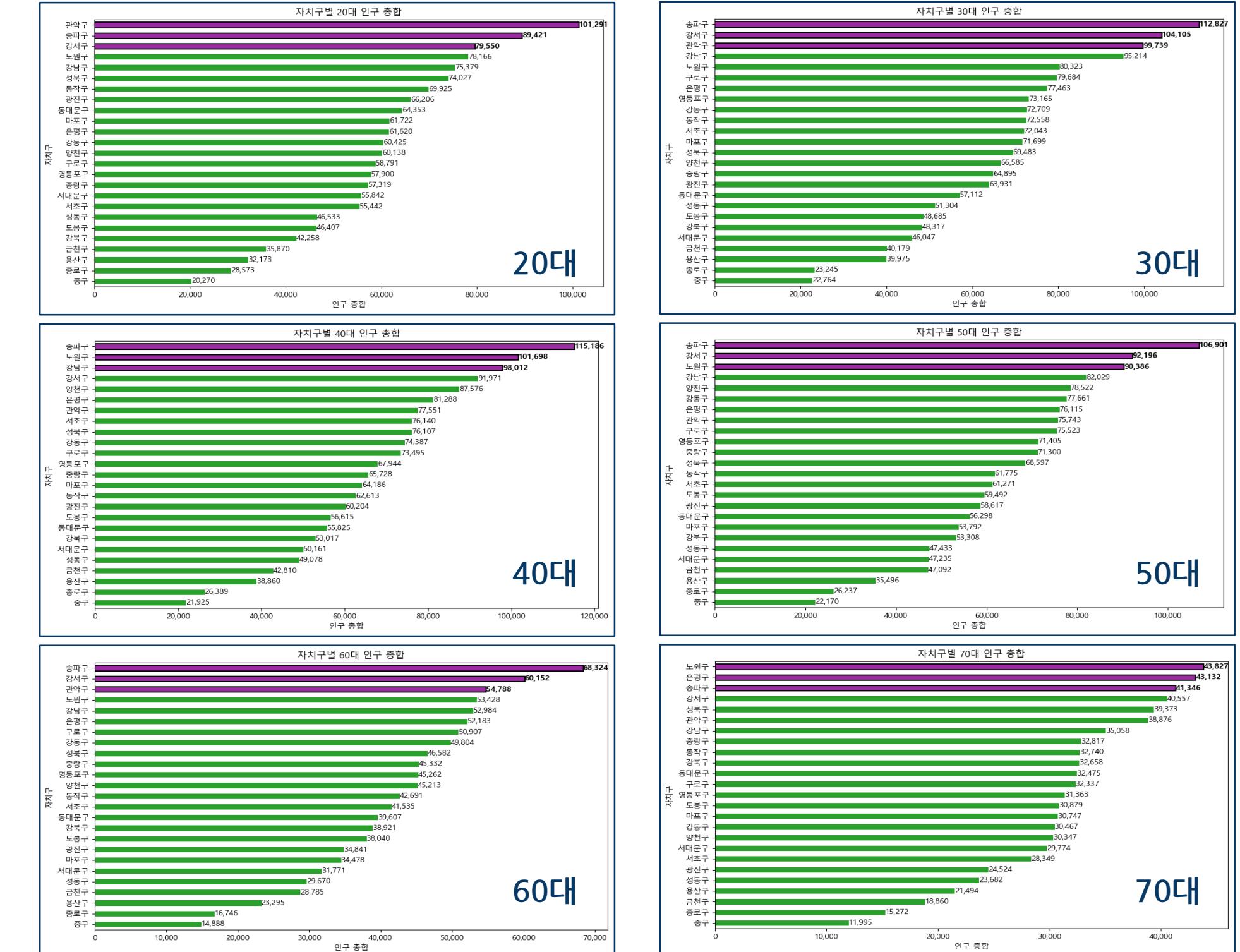
05

현황 분석 및 데이터 시각화

자치구별 연령대 인구 현황 그래프_종합

- ▶ **관악구** : 20대가 가장 많이 거주하는 자치구임.
- 상대적으로 저렴한 주거비용, 강남 지역의 출퇴근 용이한 입지적인 요인들로 인한 것으로 예상됨.
- ▶ **송파구** : 경제 활동 인구(30~60대)가 가장 많이 거주하는 자치구임.
- 학군 및 입지적인 요인, 주택 공급량 등의 이유로 예상.
- 주로 자녀를 두고 직장 생활을 하는 경제 활동 인구가 선택하기 좋은 거주지로 예상됨.
- ▶ **노원구** : 70대 노령인구가 많은 자치구임.
- 퇴직 후 서울 외곽의 조용한 위치, 집값 등의 이유가 영향을 미쳤을 것으로 예상됨.

종합	1순위	2순위	3순위
20대	관악구	송파구	강서구
30대	송파구	강서구	관악구
40대	송파구	노원구	강남구
50대	송파구	강서구	강서구
60대	송파구	강서구	관악구
70대	노원구	은평구	송파구



자치구별 연령대 인구 현황 그래프_남성

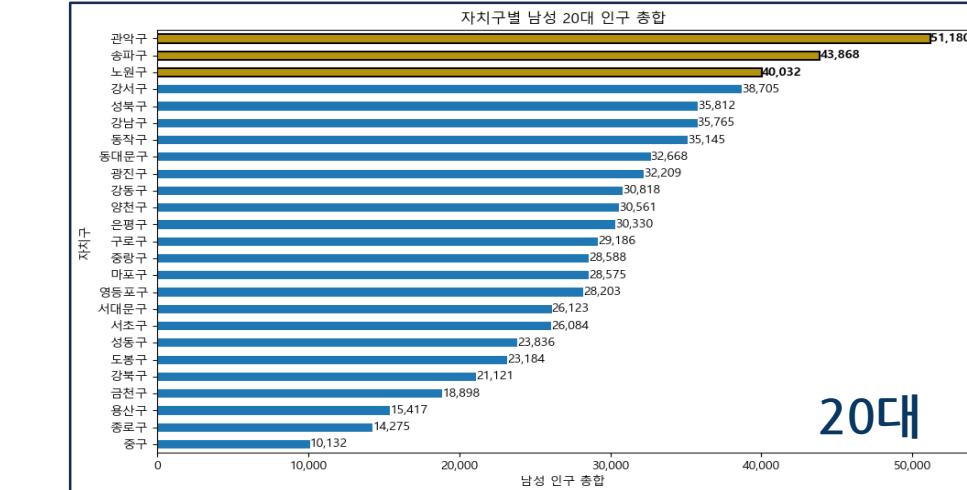
▶ 노원구

- 남성의 거주 선호도가 종합 그래프와 비교 시 더 짧은 층에서 보임.

▶ 그 외 지역

- 남녀 종합 분포와 비교 시 연령대별 거주지 선호도 큰 차이 없음.

남성	1순위	2순위	3순위
20대	관악구	송파구	노원구
30대	송파구	관악구	강서구
40대	송파구	노원구	강남구
50대	송파구	노원구	강서구
60대	송파구	강서구	관악구
70대	은평구	송파구	관악구

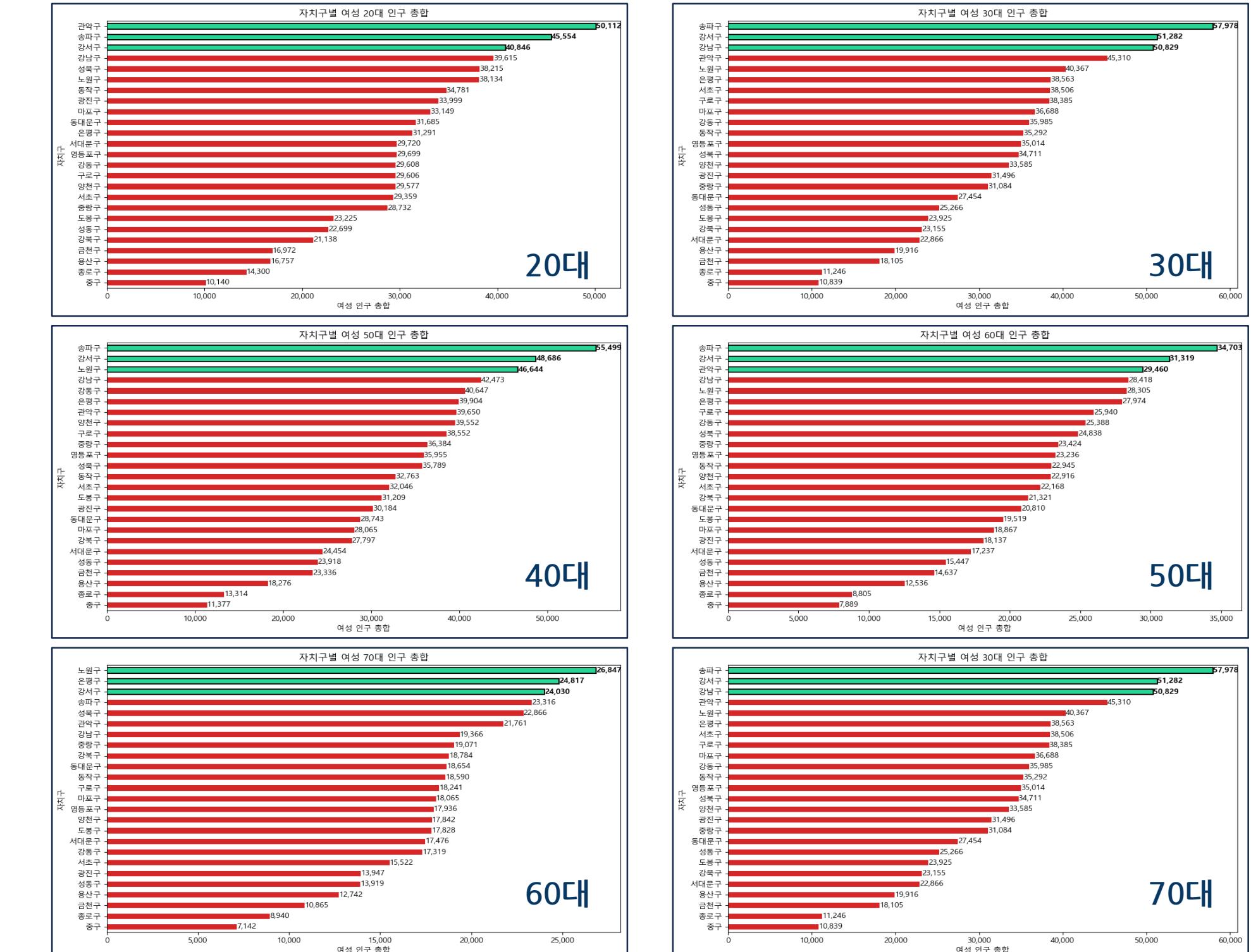


자치구별 연령대 인구 현황 그래프_여성

▶ 전체 자치구

- 남녀 종합 분포와 비교 시 연령대별 거주지 선호도 큰 차이 없음.

여성	1순위	2순위	3순위
20대	관악구	송파구	강서구
30대	송파구	강서구	강남구
40대	송파구	강서구	강남구
50대	송파구	강서구	노원구
60대	송파구	강서구	관악구
70대	노원구	은평구	강서구



Code_자치구별 연령대 인구 현황 그라프(Bar plot)

```

import pandas as pd

import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
# 데이터 로드
data_total = age_most_region_total_10
data_sex = age_most_region_sex_10
# 종합 그래프 생성 및 파일로 저장
for age in range(20,71,10):
    age_total_group = data_total[(data_total['연령대'] == age)]
    age_total_by_region = age_total_group.groupby('자치구')['총인구수'].sum()
    age_total_by_region_sorted = age_total_by_region.sort_values() # (ascending=False)
    plt.figure(figsize=(10, 6)), # 예를 들어, 가로 길이를 14인치로 늘림
    bars = age_total_by_region_sorted.plot(kind='barh', color='tab:green')
    gender_text = '종합'
    plt.title(f'자치구별 {age}대 인구 총합')
    plt.xlabel('인구 총합')
    plt.ylabel('자치구')
    plt.xticks(rotation=0)
    plt.tight_layout() # 레이아웃 조정

```

```

# x축 라벨에 천 단위 구분기호 추가
plt.gca().xaxis.set_major_formatter(ticker.FuncFormatter(lambda x, p: format(int(x), ',')))

for i, bar in enumerate(bars.patches):
    if i > 21: # 상위 3개에 대해서만
        bar.set_edgecolor('black') # 바 외곽선 색 변경
        bar.set linewidth(1.5) # 바 외곽선 굵기 변경
        bar.set facecolor('#9A2BA1') # 상위 3개 바 색 변경
        plt.text(bar.get_width() + 0.1,
                 bar.get_y() + bar.get_height() / 2, # y 위치를 바의 중간으로 설정
                 f'{bar.get_width():,.0f}', # 천 단위 구분자 포함
                 ha='left', va='center', fontweight='bold', color='black')
    else:
        plt.text(bar.get_width() + 0.1, bar.get_y() + bar.get_height() / 2,
                 f'{bar.get_width():,.0f}',
                 ha='left', va='center')

# 파일저장
# filename_total = f'자치구별 {age}대_인구_총합.png'
# plt.savefig(filename_total)
plt.show();

```

Code_자치구별 연령대 인구 현황 그라프(Bar plot)

```
# 성별과 연령대별로 그래프 생성 및 파일로 저장
# 이중 for문으로 남녀 성별 한번에 그래프 해결!
for gender in ['M', 'F']:
    for age in range(20,71,10):
        # 해당 성별과 연령대를 필터링
        age_group = data_sex[(data_sex['성별'] == gender) & (data_sex['연령대'] == age)]

        # 자치구별로 그룹화하여 '성별 인구 총합'을 합산
        age_sex_by_region = age_group.groupby('자치구')['총인구수'].sum()

        # 내림차순으로 정렬
        age_sex_by_region_sorted = age_sex_by_region.sort_values() #(ascending=)
sort_values()함수 사용
        # 그래프 그리기
        plt.figure(figsize=(10, 6)),
        bars = age_sex_by_region_sorted.plot(kind='barh', color='tab:blue' if gender == 'M'
else 'tab:red')
        gender_text = '남성' if gender == 'M' else '여성'
        plt.title(f'자치구별 {gender_text} {age}대 인구 총합') #표 제목
        plt.xlabel(f'{gender_text} 인구 총합') #x라벨
        plt.ylabel('자치구') #y라벨
        plt.xticks(rotation=0)#x축 회전 0
        plt.tight_layout() # 레이아웃 조정
```

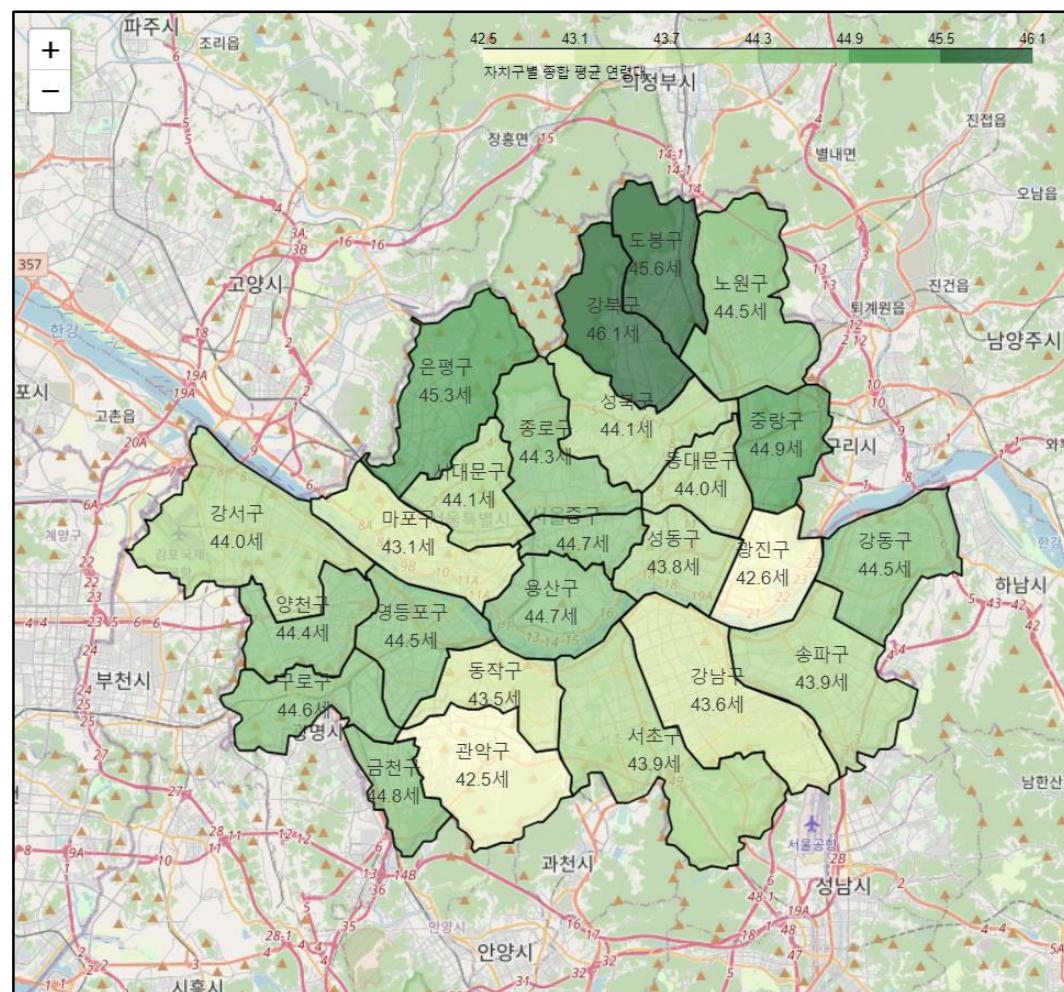
```
# x축 라벨에 천 단위 구분기호 추가
plt.gca().xaxis.set_major_formatter(ticker.FuncFormatter(lambda x, p: format(int(x), ',')))

for i, bar in enumerate(bars.patches):
    if i > 21: # 상위 3개에 대해서만
        bar.set_edgecolor('black') # 바 외곽선 색 변경
        bar.set_linewidth(1.5) # 바 외곽선 굵기 변경
        bar.set_facecolor('#B5930D' if gender == 'M' else '#27D694') # 상위 3개 바 색 변경
        plt.text(bar.get_width() + 0.1,
                 bar.get_y() + bar.get_height() / 2, # y 위치를 바의 중간으로 설정
                 f'{bar.get_width():,.0f}', # 천 단위 구분자 포함
                 ha='left', va='center', fontweight='bold', color='black')
    else:
        plt.text(bar.get_width() + 0.1, bar.get_y() + bar.get_height() / 2,
                 f'{bar.get_width():,.0f}',
                 ha='left', va='center')

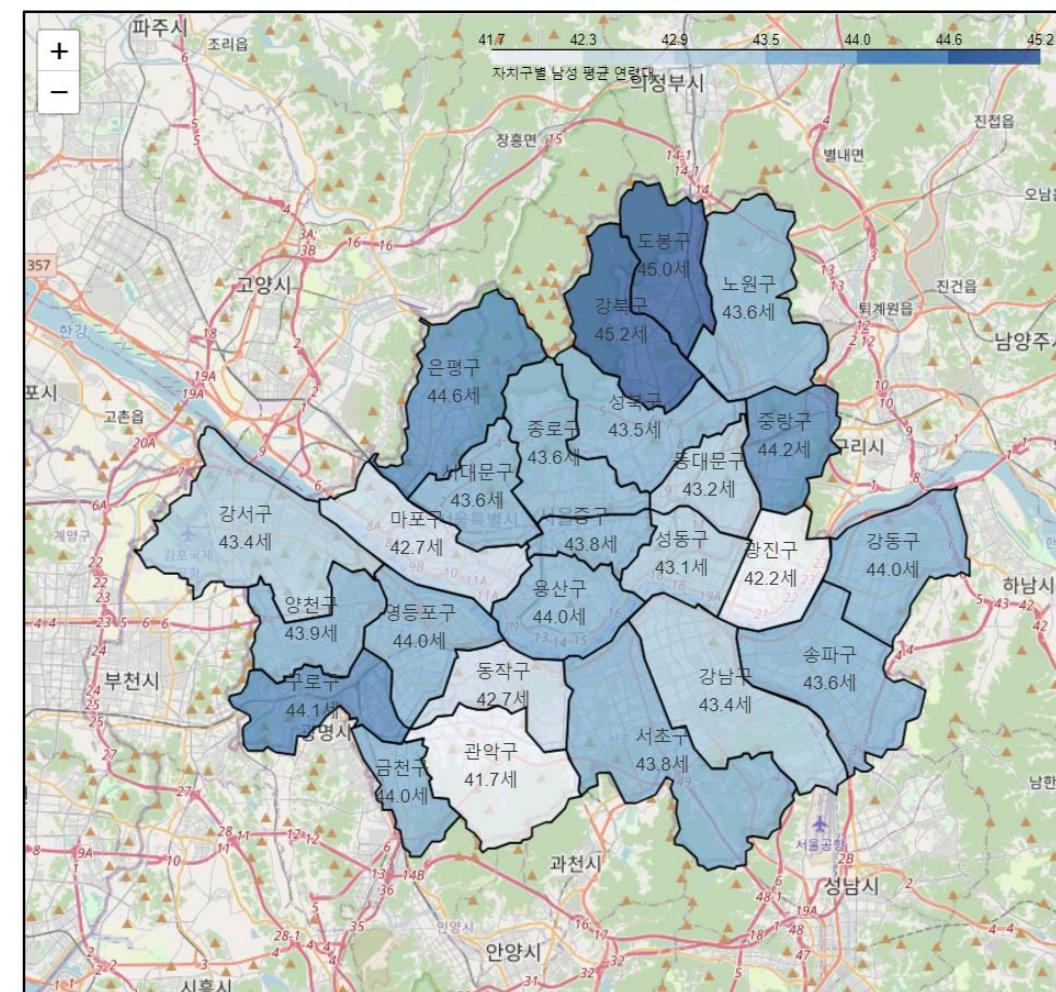
# 파일로 저장
# filename_sex = f'자치구별_{gender_text}_{age}대_인구_총합.png'
# plt.savefig(filename_sex)
plt.show();
1; # 마지막에 의미 없는 표현식 추가, 필요없는 문구 출력 방지
```

자치구별 평균 연령대

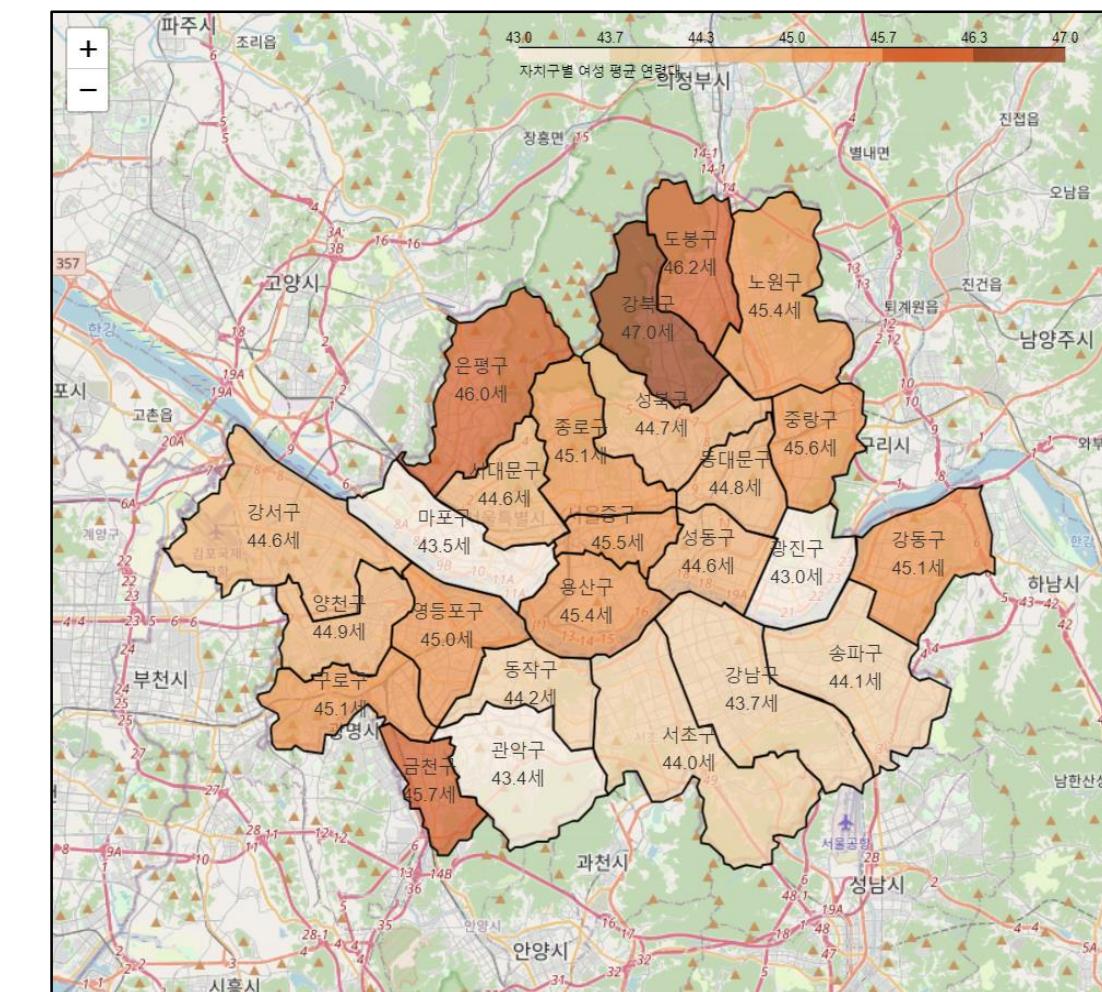
성별과 상관 없이 평균 연령대는
약 42~47세로 큰 차이가 없으며
유의미한 데이터로 볼 수 없음



[종합]



[남성]



[여성]

Code_자치구별 평균 연령대(지도 시각화)

```
# json 사용
import folium
import json

import remale_data = average_age_sex[average_age_sex['성별'] == 'M']
male_data['평균 연령대'] = male_data['평균 연령대'].round(1)

female_data = average_age_sex[average_age_sex['성별'] == 'F']
female_data['평균 연령대'] = female_data['평균 연령대'].round(1)

# 종합 지도 생성
seoul_map_total = folium.Map(
    location=[37.566345, 126.977893],
    zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth_total',
    data=average_age_total,
    columns=['자치구', '평균 연령대'],
    key_on='feature.properties.name',
    fill_color='YlGn',
    fill_opacity=0.7,
    line_opacity=1,
    line_weight=1.5,
    line_color='#000',
    legend_name='자치구별 종합 평균 연령대',
).add_to(seoul_map_total)
```

```
# 남성 지도 생성
seoul_map_male = folium.Map(
    location=[37.566345, 126.977893],
    zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth_male',
    data=male_data,
    columns=['자치구', '평균 연령대'],
    key_on='feature.properties.name',
    fill_color='Blues',
    fill_opacity=0.7,
    line_opacity=1,
    line_weight=1.5,
    line_color='#000',
    legend_name='자치구별 남성 평균 연령대',
).add_to(seoul_map_male)
```

```
# 여성 지도 생성
seoul_map_female = folium.Map(
    location=[37.566345, 126.977893],
    zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth_female',
    data=female_data,
    columns=['자치구', '평균 연령대'],
    key_on='feature.properties.name',
    fill_color='Oranges',
    fill_opacity=0.7,
    line_opacity=1,
    line_weight=1.5,
    line_color='#000',
    legend_name='자치구별 여성 평균 연령대',
).add_to(seoul_map_female)
```

Code_자치구별 평균 연령대(지도 시각화)

```
# 자치구명과 평균연령 표시
coordinates = {'종로구': (37.59491732, 126.9773213),
               '중구': (37.56014356, 126.9959681),
               '용산구': (37.53138497, 126.979907),
               '성동구': (37.55102969, 127.0410585),
               '광진구': (37.54670608, 127.0857435),
               '동대문구': (37.58195655, 127.0548481),
               '중랑구': (37.59780259, 127.0928803),
               '성북구': (37.6057019, 127.0175795),
               '강북구': (37.64347391, 127.011189),
               '도봉구': (37.66910208, 127.0323688),
               '노원구': (37.65251105, 127.0750347),
               '은평구': (37.61921128, 126.9270229),
               '서대문구': (37.57778531, 126.9390631),
               '마포구': (37.55931349, 126.90827),
               '양천구': (37.52478941, 126.8554777),
               '강서구': (37.56123543, 126.822807),
               '구로구': (37.49440543, 126.8563006),
               '금천구': (37.46056756, 126.9008202),
               '영등포구': (37.52230829, 126.9101695),
               '동작구': (37.49887688, 126.9516415),
               '관악구': (37.46737569, 126.9453372),
               '서초구': (37.47329547, 127.0312203),
               '강남구': (37.49664389, 127.0629852),
               '송파구': (37.50561924, 127.115295),
               '강동구': (37.55045024, 127.1470118)}
```

```
# 남성 지도 생성
seoul_map_male = folium.Map(
    location=[37.566345, 126.977893],
    zoom_start=10.5)
# 종합 지도에 각 자치구별 평균 연령대 마커 추가
for index, row in average_age_total.iterrows():
    district_name = row['자치구']
    age_avg = row['평균 연령대']
    if district_name in coordinates:
        folium.Marker(
            location=coordinates[district_name],
            icon=DivIcon(
                icon_size=(150,36),
                icon_anchor=(75,20),
                html=f'

<{district_name}<br>{age_avg}세</div>',
            )
        ).add_to(seoul_map_male);


```

```
# 남성 지도에 각 자치구별 평균 연령대 마커 추가
for index, row in male_data.iterrows():
    district_name = row['자치구']
    age_avg = row['평균 연령대']
    if district_name in coordinates:
        folium.Marker(
            location=coordinates[district_name],
            icon=DivIcon(
                icon_size=(150,36),
                icon_anchor=(75,20),
                html=f'

<{district_name}<br>{age_avg}세</div>',
            )
        ).add_to(seoul_map_male);


```

Code_자치구별 평균 연령대(지도 시각화)

```
# 여성 지도 생성
seoul_map_female = folium.Map(
    location=[37.566345, 126.977893],
    zoom_start=10.5)
folium.Choropleth(
    geo_data=geo_json,
    name='choropleth_female',
    data=female_data,
    columns=['자치구', '평균 연령대'],
    key_on='feature.properties'
)

# 여성 지도에 각 자치구별 평균 연령대 마커 추가
for index, row in female_data.iterrows():
    district_name = row['자치구']
    age_avg = row['평균 연령대']
    if district_name in coordinates:
        folium.Marker(
            location=coordinates[district_name],
            icon=DivIcon(
                icon_size=(150,36),
                icon_anchor=(75,20),
                html=f'

{district_name}<br>{age_avg}세</div>',
            )
        ).add_to(seoul_map_female);

).add_to(seoul_map_female);


```

```
folium.Choropleth(
    geo_data=geo_json,
    name='choropleth_male',
    data=male_data,
    columns=['자치구', '평균 연령대'],
    key_on='feature.properties.name',
    fill_color='Blues',
    fill_opacity=0.7,
    line_opacity=1,
    line_weight=1.5,
    line_color='#000',
    legend_name='자치구별 남성 평균 연령대',
).add_to(seoul_map_male)

).name',
fill_color='Oranges',
fill_opacity=0.7,
line_opacity=1,
line_weight=1.5,
line_color='#000',
legend_name='자치구별 여성 평균 연령대',
).add_to(seoul_map_female)
```

```
# 여성 지도에 각 자치구별 평균 연령대 마커 추가
for index, row in female_data.iterrows():
    district_name = row['자치구']
    age_avg = row['평균 연령대']
    if district_name in coordinates:
        folium.Marker(
            location=coordinates[district_name],
            icon=DivIcon(
                icon_size=(150,36),
                icon_anchor=(75,20),
                html=f'

{district_name}<br>{age_avg}세</div>',
            )
        ).add_to(seoul_map_female);

).add_to(seoul_map_female);

# 지도 생성
seoul_map_total
seoul_map_male
seoul_map_female

# 지도 출력
# seoul_map_total.save('seoul_map_total.html')
# seoul_map_male.save('seoul_map_male.html')
# seoul_map_female.save('seoul_map_female.html')


```

연령대별 서비스 상관관계(1)

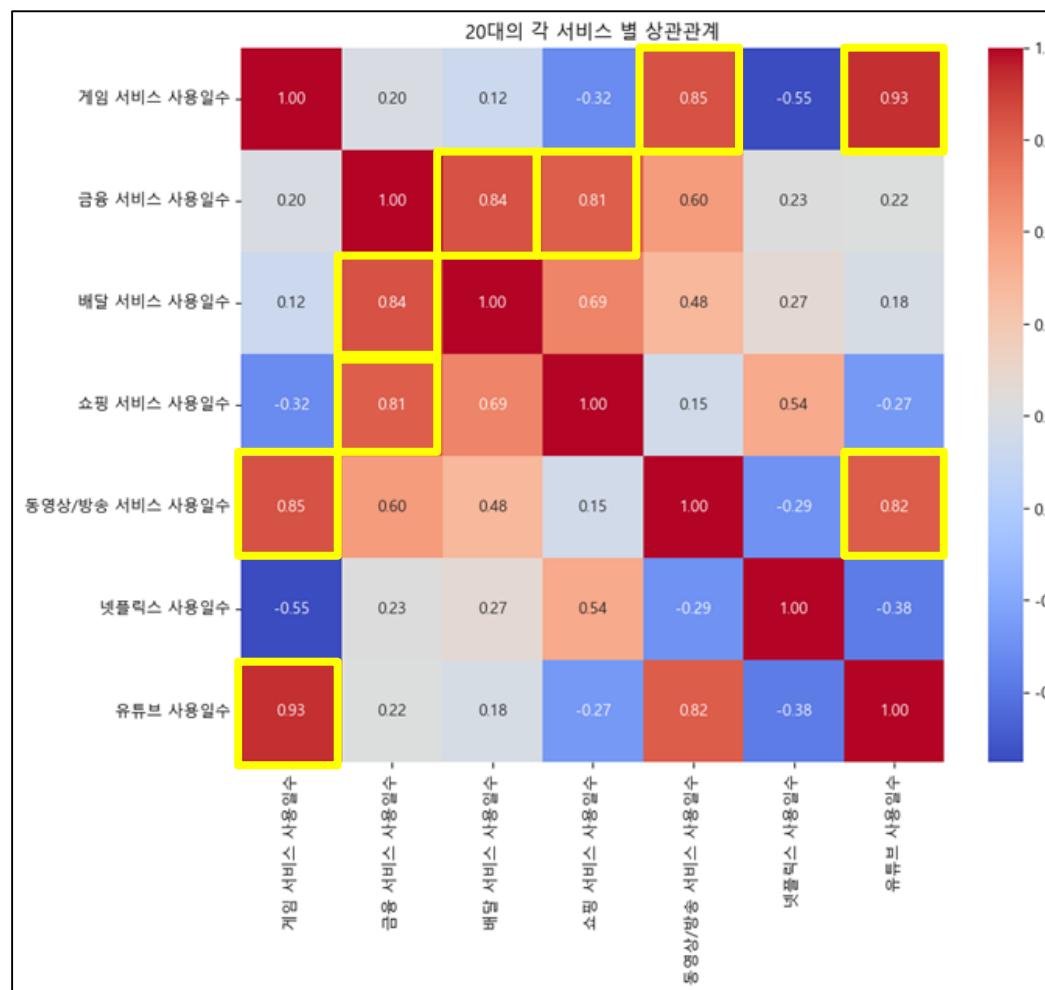
상관계수가 “0.7 이상”으로 Feature들 간의 상관 관계가 높음을 알 수 있다.

게임	동영상/방송, 유튜브
금융	쇼핑, 배달
배달	금융
쇼핑	금융, 배달
동영상/방송	게임, 유튜브
유튜브	게임, 동영상/방송
넷플릭스	높은 상관관계 없음

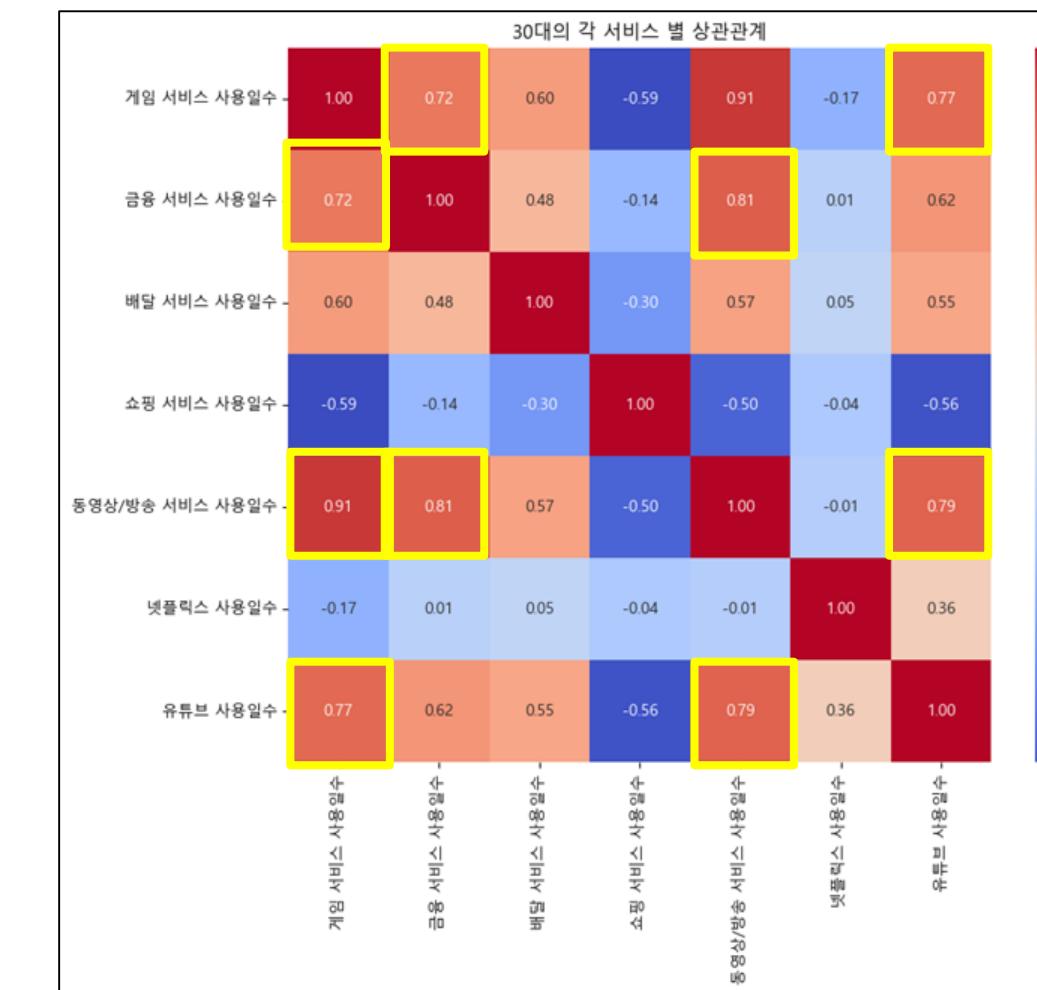
게임	금융, 동영상/방송, 유튜브
금융	게임
배달	금융
쇼핑	모든 서비스 음의 상관관계
동영상/방송	게임, 금융, 유튜브
유튜브	게임, 동영상/방송
넷플릭스	음의 상관관계

게임	동영상/방송, 유튜브, 금융
금융	게임, 동영상/방송, 유튜브
동영상/방송	게임, 금융, 유튜브
유튜브	게임, 금융, 동영상/방송
배달	유의미한 상관관계 없음
쇼핑	
넷플릭스	

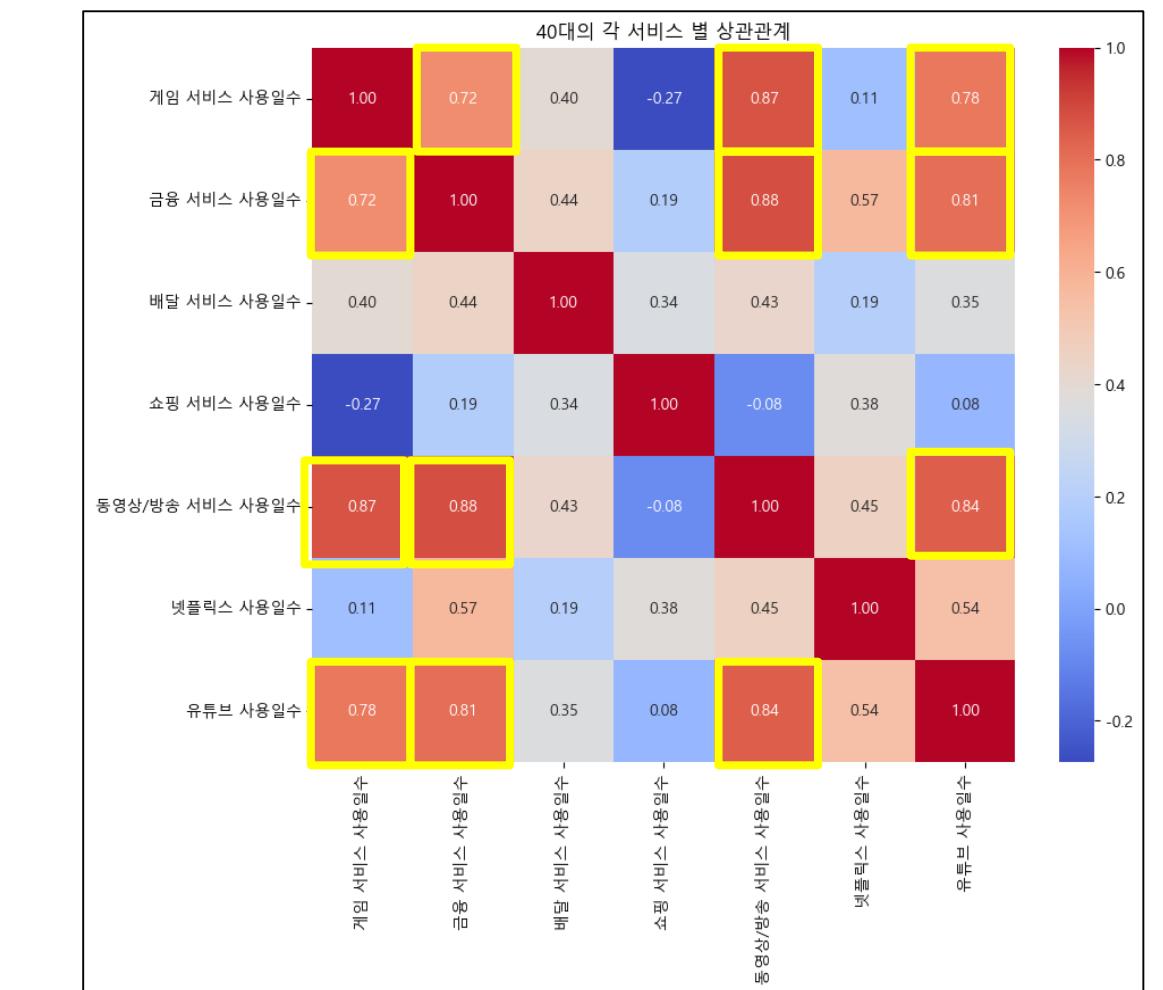
[20대]



[30대]



[40대]



연령대별 서비스 상관관계(2)

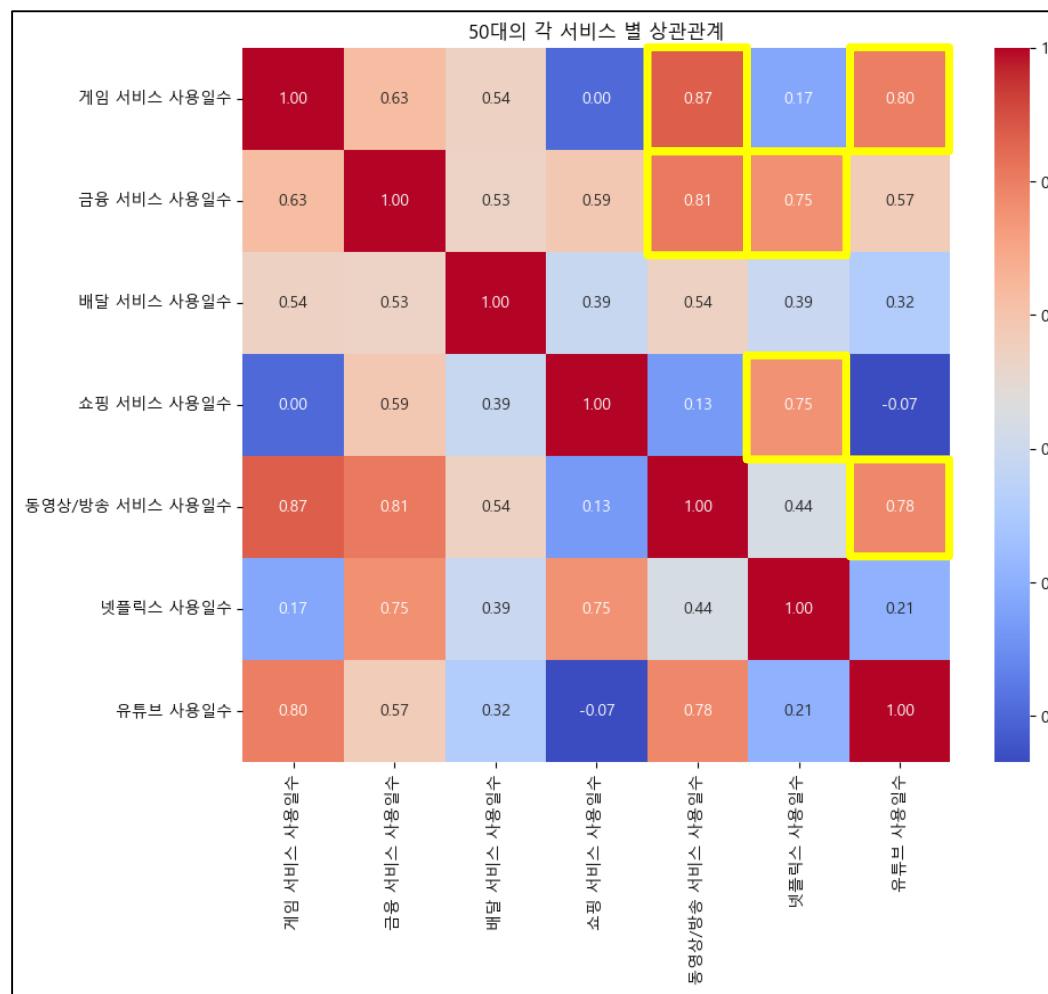
상관계수가 “0.7 이상”으로 Feature들 간의 상관 관계가 높음을 알 수 있다.

게임	동영상/방송, 유튜브
금융	동영상/방송, 넷플릭스
배달	높은 상관관계 없음
쇼핑	넷플릭스
동영상/방송	유튜브
유튜브	게임, 동영상/방송
넷플릭스	금융, 쇼핑

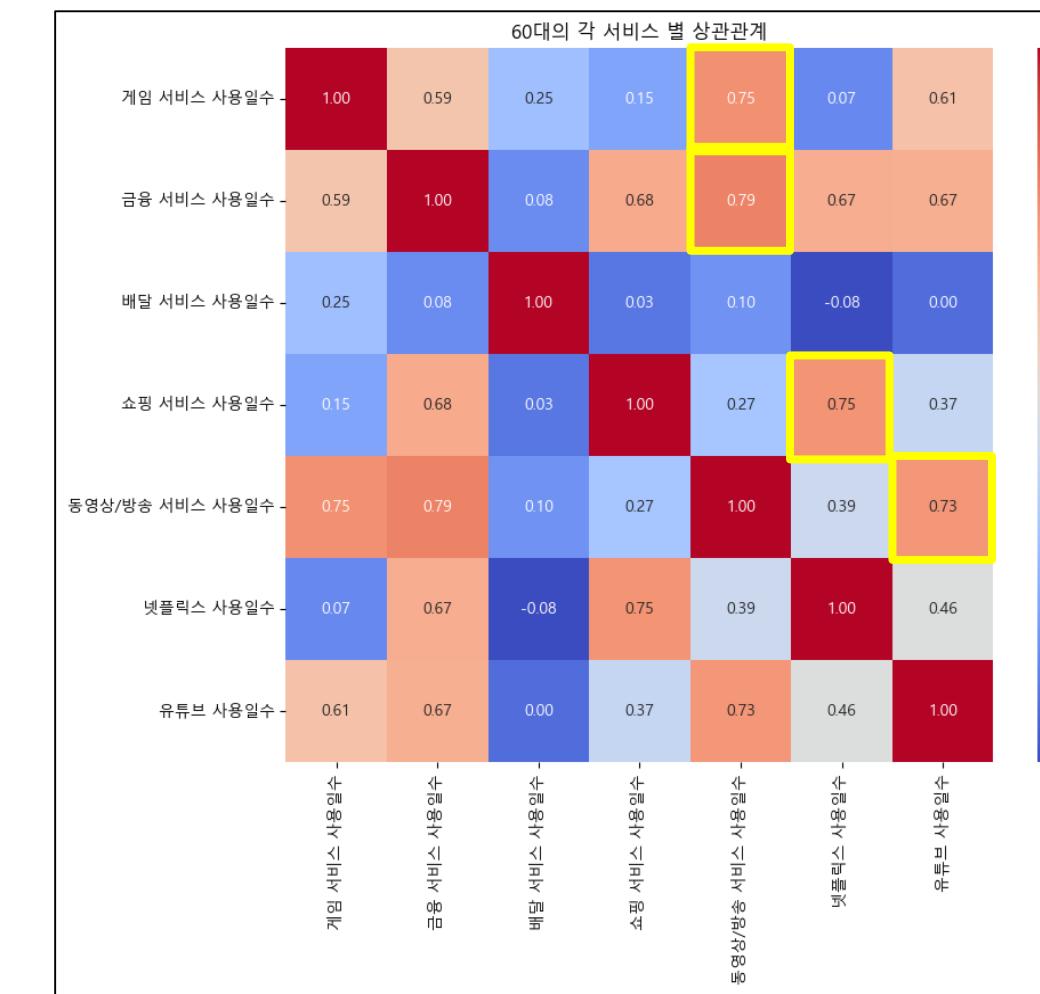
게임	동영상/방송
금융	동영상/방송
배달	낮은 상관관계
쇼핑	넷플릭스
동영상/방송	유튜브
유튜브	동영상/방송
넷플릭스	쇼핑

게임	금융, 동영상/방송, 유튜브
금융	쇼핑, 동영상/방송
배달	모든 서비스 음의 상관관계
쇼핑	유튜브
동영상/방송	게임, 금융
유튜브	쇼핑, 게임, 금융
넷플릭스	높은 상관관계 없음

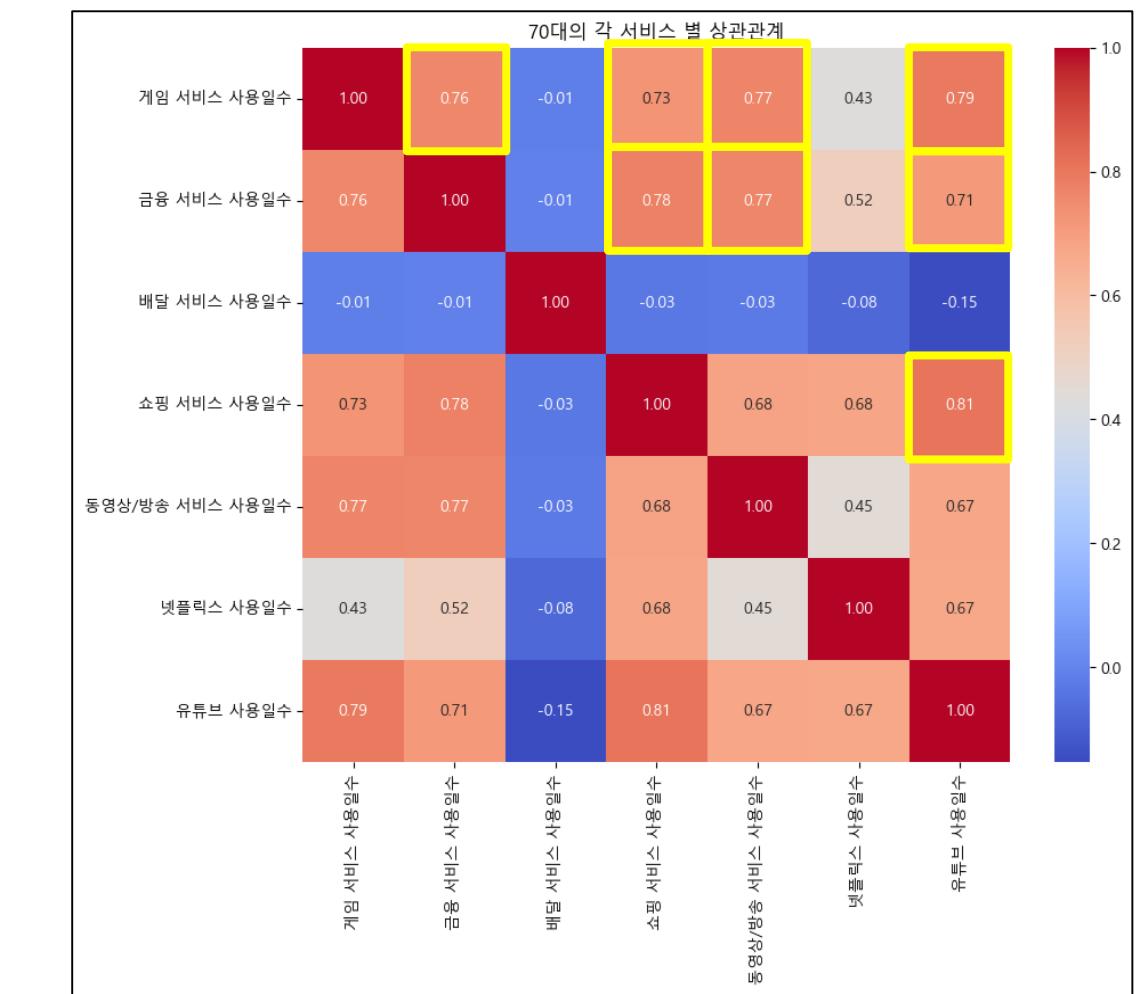
[50대]



[60대]



[70대]



Code_ 연령대별 서비스 상관관계(히트맵)

```
average_df_col_hit = average_df_col
average_df_col_hit['10단위 연령대'] = average_df_col_hit['연령대'].apply(merge_age)

# 10단위 연령대 컬럼 생성
from sklearn.preprocessing import StandardScaler
average_df_col_hit
```

```
for i in range(2, 8):
    age = average_df_col_hit[average_df_col['10단위 연령대'] == i * 10]

    scaler = StandardScaler()

    sv = age[['게임 서비스 사용일수', '금융 서비스 사용일수', '배달 서비스 사용일수',
              '쇼핑 서비스 사용일수', '동영상/방송 서비스 사용일수', '넷플릭스 사용일수', '유튜브 사용일수']]
    scaler.fit(sv)

    # 상관계수
    corr_mat = sv.corr()

    # plot
    plt.figure(figsize=(10, 8))
    sns.heatmap(corr_mat, annot=True, cmap='coolwarm', fmt='.2f')
    plt.title('{0}0대의 각 서비스 별 상관관계'.format(i))
    plt.show()
```

06

시스템 모델링

추천 서비스 시스템

1. 고객별 맞춤형 광고 알고리즘

“월별 서비스 트렌드 기반” 추천 시스템

2. 서비스별 타겟 고객 및 광고 송출 서비스 추천

서비스 사용일수에 따른 “Top N recommendation” 추천 시스템

추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

고객의 “**자치구, 성별, 연령대**”를 입력하면 월별로
고객 개개인의 “**광고 알고리즘에 뜰 서비스**”를
“**서비스 사용일수 높은 순**”으로 추천

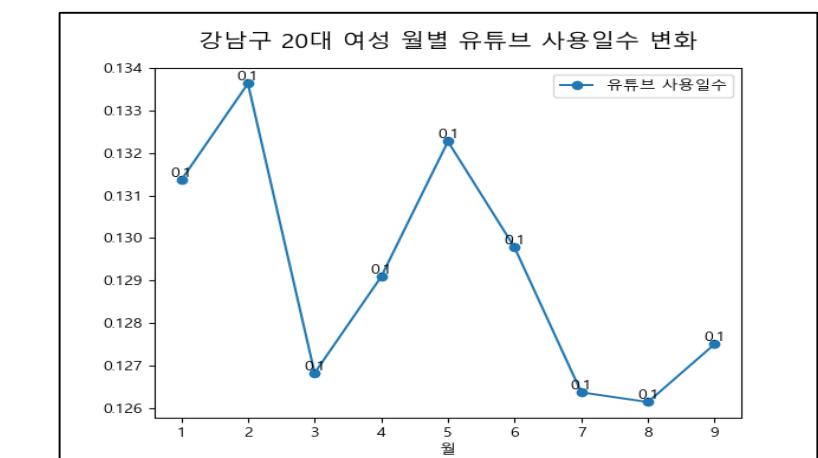
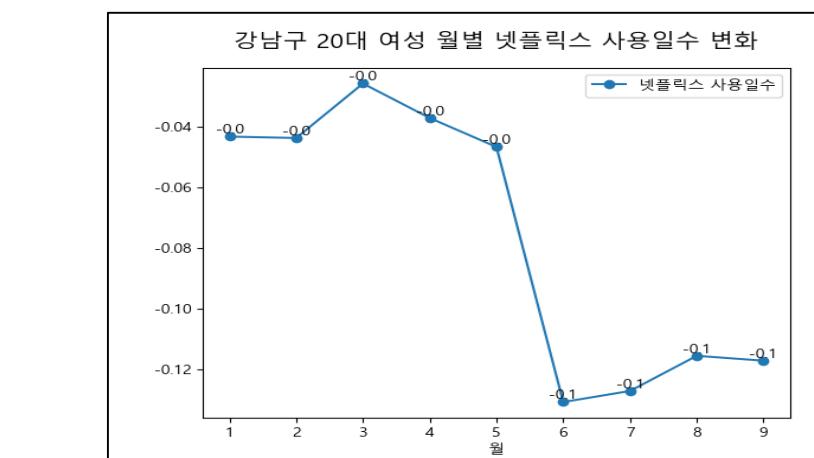
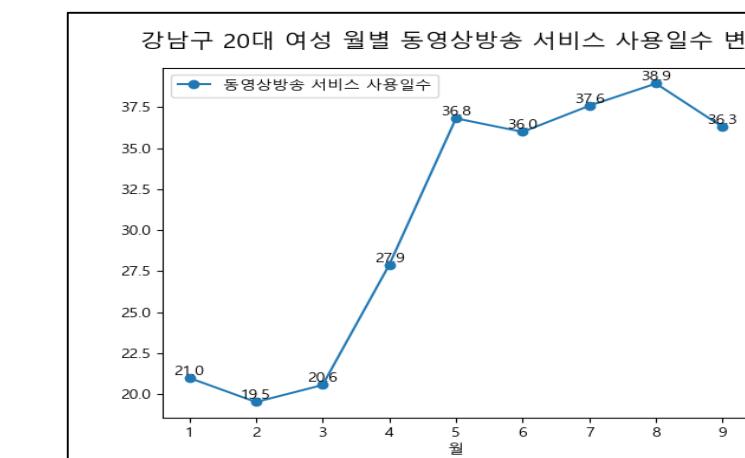
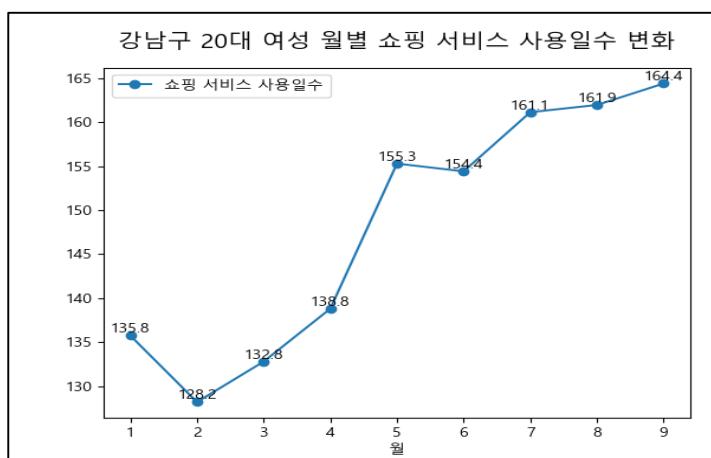
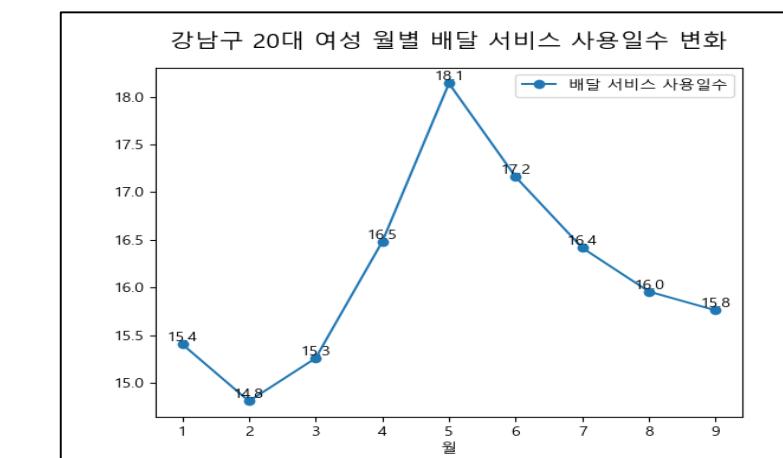
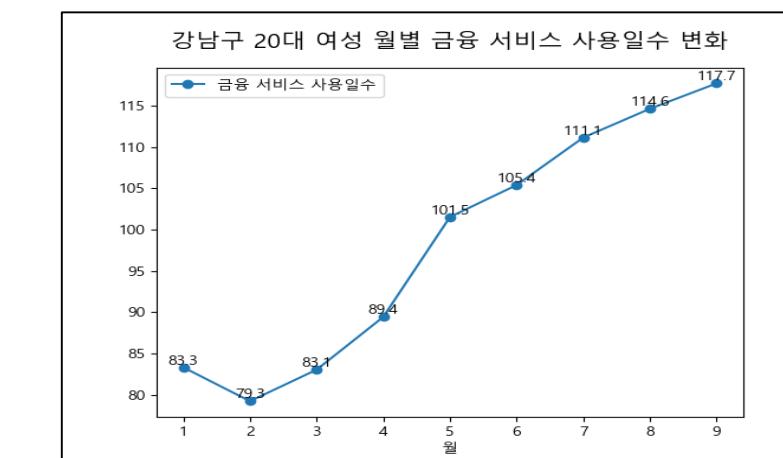
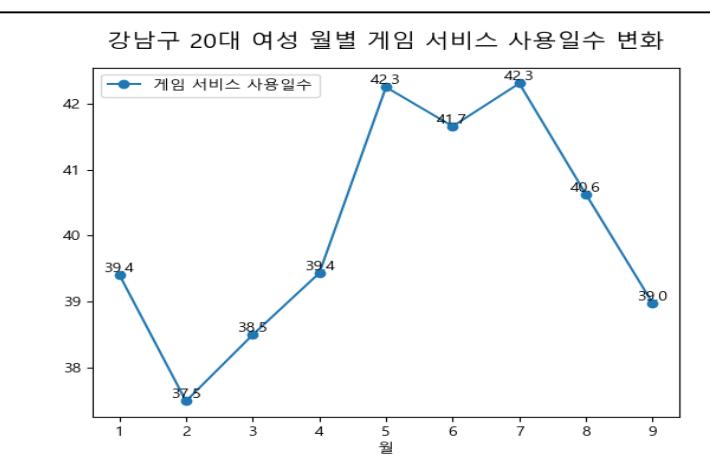


추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

고객의 “자치구, 성별, 연령대” 뿐만 아니라

“월별 서비스 사용일수의 분포가 상이”하게 나타남.

따라서 **“고객별, 월별 맞춤형 광고 서비스 추천”**을 진행하는 것이 **최종 목적**이다.



추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

“가중치가 부여된 월별 서비스별 사용일수”를 이용해

자치구	연령대	성별	월	게임 서비스 사용일수	금융 서비스 사용일수	배달 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	넷플릭스 사용일수	유튜브 사용일수
강남구	20	F	1	-0.622544839	0.147354753	0.656832439	0.277327412	0.145568574	0.817553247	0.979137384
강남구	20	F	2	-0.592161124	0.140128515	0.631388164	0.261916671	0.1353849	0.826159071	0.996077477
강남구	20	F	3	-0.608076403	0.146856905	0.650611866	0.271286041	0.142563555	0.486229036	0.945257198
강남구	20	F	4	-0.622907448	0.158137455	0.702692209	0.283540544	0.193343327	0.701374628	0.962197291
강남구	20	F	5	-0.667547778	0.179505166	0.773405203	0.317181238	0.2551163	0.882096924	0.985913421
강남구	20	F	6	-0.658141499	0.186334412	0.731595588	0.315379597	0.249418105	2.4741743	0.967279319
강남구	20	F	7	-0.668448914	0.196485461	0.699891982	0.329091754	0.260431781	2.405327711	0.941869179
강남구	20	F	8	-0.641806169	0.202643675	0.68036791	0.330767127	0.269821676	2.185879208	0.94017517
강남구	20	F	9	-0.61566964	0.208068578	0.671976918	0.335735683	0.251454524	2.215999591	0.950339226

자치구, 성별, 연령대에 따라 300가지 경우의 수
각각의 경우의 수마다 1~9월까지 월별 데이터
300가지 경우의 수 * 1~9월까지 월별 데이터 = 행 기준 2700개의 데이터
(가중치 부여, 정규화 처리된 사용일수 데이터)

행별 사용
서비스 우선순위를
내림차순으로 정렬

특정 자치구, 연령대, 성별의 고객이 “가장 많이 사용하는 서비스 순”으로 추천을 진행함

자치구	연령대	성별	월	1	2	3	4	5	6	7
강남구	20	F	1	유튜브	넷플릭스	배달 서비스	쇼핑 서비스	금융 서비스	동영상/방송 서비스	게임 서비스
강남구	20	F	2	유튜브	넷플릭스	배달 서비스	쇼핑 서비스	금융 서비스	동영상/방송 서비스	게임 서비스
강남구	20	F	3	유튜브	배달 서비스	넷플릭스	쇼핑 서비스	금융 서비스	동영상/방송 서비스	게임 서비스
강남구	20	F	4	유튜브	배달 서비스	넷플릭스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스
강남구	20	F	5	유튜브	넷플릭스	배달 서비스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스
강남구	20	F	6	넷플릭스	유튜브	배달 서비스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스
강남구	20	F	7	넷플릭스	유튜브	배달 서비스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스
강남구	20	F	8	넷플릭스	유튜브	배달 서비스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스
강남구	20	F	9	넷플릭스	유튜브	배달 서비스	쇼핑 서비스	동영상/방송 서비스	금융 서비스	게임 서비스

자치구, 성별, 연령대, 월에 따라 서비스 사용의 우선순위가 다르게 나옴

Code_추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

```

service_columns = ['게임 서비스 사용일수', '금융 서비스 사용일수', '쇼핑 서비스 사용일수',
'동영상/방송 서비스 사용일수', '유튜브 사용일수', '넷플릭스 사용일수', '배달 서비스 사용일수']

# 데이터 프레임 생성
df = pd.DataFrame()

# 데이터 프레임의 컬럼 생성
df['자치구'] = data['자치구']
df['연령대'] = data['연령대']
df['성별'] = data['성별']
df['1'] = 'service 1'
df['2'] = 'service 2'
df['3'] = 'service 3'
df['4'] = 'service 4'
df['5'] = 'service 5'
df['6'] = 'service 6'
df['7'] = 'service 7'

```

```

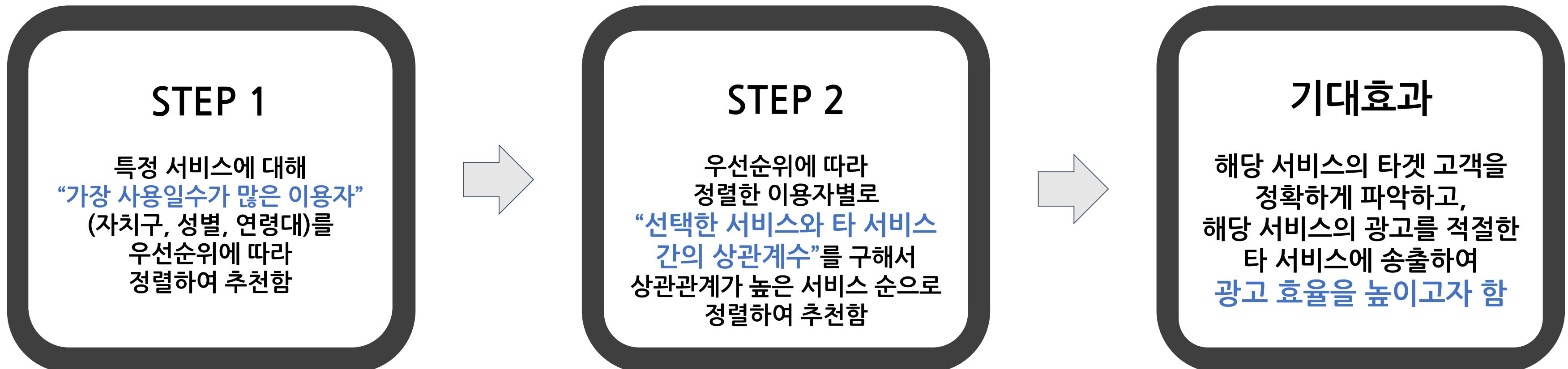
for i in range(2700): # 자치구, 연령대, 성별에 따라 300가지 경우의 수 * 1~9월
    row = data.loc[[i]] # 월별 가중치 처리한 엑셀 파일 data
    priority = []
    for col in service_columns:
        priority.append([float(row[col]), col]) # 가중치 값과 col명을 함께 append
    priority = pd.DataFrame(priority)
    priority = priority.sort_values(by = 0, ascending=False) # 가중치로 내림차순 정렬
    priority = priority.reset_index(drop=True)

    # 가중치 값이 가장 높은 col명에서 '사용일수'를 제외하고
    서비스명만 반환될 수 있도록 하고 df의 i행의 '1'컬럼에 서비스명을 넣는다.
    df.loc[i, '1'] = priority.loc[0][1][-5]
    df.loc[i, '2'] = priority.loc[1][1][-5]
    df.loc[i, '3'] = priority.loc[2][1][-5]
    df.loc[i, '4'] = priority.loc[3][1][-5]
    df.loc[i, '5'] = priority.loc[4][1][-5]
    df.loc[i, '6'] = priority.loc[5][1][-5]
    df.loc[i, '7'] = priority.loc[6][1][-5]

```

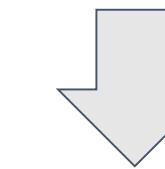
추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

“특정 서비스”를 선택하면 서비스 사용일수가 가장 많은 고객순으로 자치구, 연령대, 성별을 제공함
또한 고객 군집별로 선택한 서비스와
“가장 상관관계가 높은 타 서비스를 내림차순으로 정렬”하여 제공함

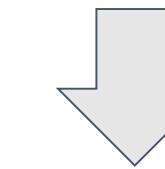


추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

“게임회사” 가 새로 나온 게임 서비스의 광고를 하고 싶어함



게임 서비스 광고의 “타겟 고객”的 자치구, 성별, 연령대 정보를 알고 싶음



게임 서비스 광고를 “어느 서비스 분야의 플랫폼에 노출” 시켜야 할지 추천 받고 싶음

[추천 시스템 2의 최종 목적]

게임 서비스에 대해 상위 10개 군집의 “타겟 고객의 자치구, 성별, 연령대 정보”를 추천,

타겟 고객 군집별로 “게임 광고를 어떤 서비스 분야의 플랫폼에 노출시켜야 하는지”를 추천

추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

z-value로 처리된 서비스별 사용일수 데이터이며 자치구, 연령대, 성별 기준으로 group by 되어 있음

자치구	연령대	성별	게임 서비스 사용일수	금융 서비스 사용일수	쇼핑 서비스 사용일수	동영상/방송 서비스 사용일수	유튜브 사용일수	넷플릭스 사용일수	배달 서비스 사용일수	
강남구	20 F		-0.633033757	0.173946102	0.302469563		0.21145586	0.963138407	1.443865969	0.688751364
강남구	20 M		1.650808048	0.360857321	-0.003575118		1.018430968	1.556589964	1.194777907	0.866319541
강남구	30 F		-0.606889006	0.720859842	1.163397038		0.569296833	0.436702112	1.188535405	0.765111366
강남구	30 M		1.048269362	1.24932228	0.703659484		1.308372543	0.90659428	1.086819347	1.330061666
강남구	40 F		-0.433757399	0.43737117	1.036516516		0.258879854	0.019427397	0.613735633	0.138156925
강남구	40 M		0.833763242	1.125689054	0.713507908		1.140649389	0.382513991	0.725488656	0.25989101
강남구	50 F		-0.768392424	-0.216493856	0.131137535		-0.564196815	-0.330177612	-0.186283811	-0.781135052

각 서비스 사용일수 컬럼을 기준으로 내림차순 정렬한 데이터프레임을 각각 생성 (타겟 고객 선정)

게임 서비스 사용일수가 가장 많은 1순위 타겟 고객 군집이다.

자치구	연령대	성별	게임 서비스 사용일수
금천구	20 M		2.17357074
강북구	20 M		2.12635986
양천구	20 M		2.120855492
구로구	20 M		2.097964627
도봉구	20 M		2.07640089
노원구	20 M		2.074481255
중랑구	20 M		2.063684575
강서구	20 M		2.049125461
은평구	20 M		2.033658574
관악구	20 M		1.958004551
종로구	20 M		1.945960629

자치구	연령대	성별	금융 서비스 사용일수
영등포구	30 M		1.621496967
강동구	30 M		1.537967227
강서구	30 M		1.508727694
노원구	30 M		1.481817842
구로구	30 M		1.458323034
동대문구	30 M		1.428503453
성북구	30 M		1.422332046
중랑구	30 M		1.417785342
성동구	30 M		1.415258833
광진구	30 M		1.405234904
송파구	30 M		1.403015862

자치구	연령대	성별	유튜브 사용일수
용산구	20 M		1.71318251
종로구	20 M		1.689288497
광진구	20 M		1.681327355
관악구	20 M		1.679407647
성북구	20 M		1.673840494
성동구	20 M		1.671333581
동대문구	20 M		1.66795796
마포구	20 M		1.646592638
동작구	20 M		1.644468961
서대문구	20 M		1.636899827
구로구	20 M		1.631798888

Code_ 추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

```
# 서비스별로 dataframe 생성
```

```
game = pd.DataFrame()
game = data.groupby(['자치구', '연령대', '성별'])['게임 서비스 사용일수'].mean()
game = pd.DataFrame(game).reset_index().sort_values(by='게임 서비스 사용일수',
ascending=False) # 게임 서비스 사용일수에 대해 내림차순 정렬 진행
game = game.reset_index(drop=True)

# 게임 서비스와 상관관계가 높은 타 서비스를 우선순위에 따라 저장하기 위한 컬럼 생성
game['1'] = 'service1'
game['2'] = 'service2'
game['3'] = 'service3'
game['4'] = 'service4'
game['5'] = 'service5'
game['6'] = 'service6'

video = pd.DataFrame()
video = data.groupby(['자치구', '연령대', '성별'])['동영상/방송 서비스 사용일수'].mean()
video = pd.DataFrame(video).reset_index().sort_values(by='동영상/방송 서비스 사용일수',
ascending=False)
video = video.reset_index(drop=True)
video['1'] = 'service1'
video['2'] = 'service2'
video['3'] = 'service3'
video['4'] = 'service4'
video['5'] = 'service5'
video['6'] = 'service6'
```

```
finance = pd.DataFrame()
```

```
finance = data.groupby(['자치구', '연령대', '성별'])['금융 서비스 사용일수'].mean()
finance = pd.DataFrame(finance).reset_index().sort_values(by='금융 서비스 사용일수',
ascending=False)
```

```
finance = finance.reset_index(drop=True)
finance['1'] = 'service1'
```

```
finance['2'] = 'service2'
```

```
finance['3'] = 'service3'
```

```
finance['4'] = 'service4'
```

```
finance['5'] = 'service5'
```

```
finance['6'] = 'service6'
```

```
shopping = pd.DataFrame()
```

```
shopping = data.groupby(['자치구', '연령대', '성별'])['쇼핑 서비스 사용일수'].mean()
shopping = pd.DataFrame(shopping).reset_index().sort_values(by='쇼핑 서비스 사용일수',
ascending=False)
```

```
shopping = shopping.reset_index(drop=True)
shopping['1'] = 'service1'
```

```
shopping['2'] = 'service2'
```

```
shopping['3'] = 'service3'
```

```
shopping['4'] = 'service4'
```

```
shopping['5'] = 'service5'
```

```
shopping['6'] = 'service6'
```

```
youtube = pd.DataFrame()
```

```
youtube = data.groupby(['자치구', '연령대', '성별'])['유튜브 사용일수'].mean()
youtube = pd.DataFrame(youtube).reset_index().sort_values(by='유튜브 사용일수',
ascending=False)
```

```
youtube = youtube.reset_index(drop=True)
```

```
youtube['1'] = 'service1'
```

```
youtube['2'] = 'service2'
```

```
youtube['3'] = 'service3'
```

```
youtube['4'] = 'service4'
```

```
youtube['5'] = 'service5'
```

```
youtube['6'] = 'service6'
```

```
netflix = pd.DataFrame()
```

```
netflix = data.groupby(['자치구', '연령대', '성별'])['넷플릭스 사용일수'].mean()
netflix = pd.DataFrame(netflix).reset_index().sort_values(by='넷플릭스 사용일수',
ascending=False)
```

```
netflix = netflix.reset_index(drop=True)
```

```
netflix['1'] = 'service1'
```

```
netflix['2'] = 'service2'
```

```
netflix['3'] = 'service3'
```

```
netflix['4'] = 'service4'
```

```
netflix['5'] = 'service5'
```

```
netflix['6'] = 'service6'
```

```
delivery = pd.DataFrame()
```

```
delivery = data.groupby(['자치구', '연령대', '성별'])['배달 서비스 사용일수'].mean()
delivery = pd.DataFrame(delivery).reset_index().sort_values(by='배달 서비스 사용일수',
ascending=False)
```

```
delivery = delivery.reset_index(drop=True)
```

```
delivery['1'] = 'service1'
```

```
delivery['2'] = 'service2'
```

```
delivery['3'] = 'service3'
```

```
delivery['4'] = 'service4'
```

```
delivery['5'] = 'service5'
```

```
delivery['6'] = 'service6'
```

추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

타겟 고객별로 “선택한 서비스와 다른 서비스들 간의 상관계수”를 구해 게임 광고를 송출할 타 서비스 플랫폼 추천

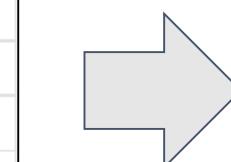
자치구	연령대	성별	게임 서비스 사용일수	1	2	3	4	5	6
금천구	20 M		2.17357074	유튜브	동영상/방송 서비스	쇼핑 서비스	넷플릭스	금융 서비스	배달 서비스
강북구	20 M		2.12635986	금융 서비스	배달 서비스	쇼핑 서비스	동영상/방송 서비스	넷플릭스	유튜브
양천구	20 M		2.120855492	동영상/방송 서비스	금융 서비스	배달 서비스	쇼핑 서비스	유튜브	넷플릭스
구로구	20 M		2.097964627	유튜브	금융 서비스	동영상/방송 서비스	배달 서비스	쇼핑 서비스	넷플릭스
도봉구	20 M		2.07640089	금융 서비스	유튜브	쇼핑 서비스	동영상/방송 서비스	배달 서비스	넷플릭스
노원구	20 M		2.074481255	동영상/방송 서비스	유튜브	쇼핑 서비스	배달 서비스	금융 서비스	넷플릭스
중랑구	20 M		2.063684575	동영상/방송 서비스	쇼핑 서비스	금융 서비스	배달 서비스	유튜브	넷플릭스
강서구	20 M		2.049125461	유튜브	쇼핑 서비스	배달 서비스	동영상/방송 서비스	넷플릭스	금융 서비스
은평구	20 M		2.033658574	배달 서비스	유튜브	쇼핑 서비스	동영상/방송 서비스	금융 서비스	넷플릭스
관악구	20 M		1.958004551	동영상/방송 서비스	금융 서비스	유튜브	넷플릭스	쇼핑 서비스	배달 서비스
종로구	20 M		1.945960629	동영상/방송 서비스	유튜브	쇼핑 서비스	넷플릭스	금융 서비스	배달 서비스

게임 서비스를 가장 많이 이용하는
고객 군집(금천구 20대 남성)에 대해
해당 군집의 게임 서비스 사용은
유튜브와 가장 큰 상관관계를 띠고 있다.

따라서, 게임 광고의 1순위 타겟 고객은
“금천구 20대 남성”이며,

그들에 대해

“게임 광고를 유튜브에 광고하는 것”이
가장 효과적



모든 서비스별로 동일한 과정을 반복

자치구	연령대	성별	금융 서비스 사용일수	1	2	3	4	5	6
영등포구	30 M		1.621496967	넷플릭스	유튜브	쇼핑 서비스	동영상/방송 서비스	게임 서비스	배달 서비스
강동구	30 M		1.537967227	쇼핑 서비스	동영상/방송 서비스	유튜브	넷플릭스	배달 서비스	게임 서비스
강서구	30 M		1.508727694	쇼핑 서비스	동영상/방송 서비스	넷플릭스	유튜브	게임 서비스	배달 서비스
노원구	30 M		1.481817842	쇼핑 서비스	유튜브	동영상/방송 서비스	넷플릭스	게임 서비스	배달 서비스
구로구	30 M		1.458323034	넷플릭스	쇼핑 서비스	동영상/방송 서비스	유튜브	배달 서비스	게임 서비스
동대문구	30 M		1.428503453	쇼핑 서비스	넷플릭스	동영상/방송 서비스	유튜브	게임 서비스	배달 서비스
성북구	30 M		1.422332046	쇼핑 서비스	동영상/방송 서비스	게임 서비스	넷플릭스	유튜브	배달 서비스
중랑구	30 M		1.417785342	쇼핑 서비스	넷플릭스	유튜브	동영상/방송 서비스	배달 서비스	게임 서비스
성동구	30 M		1.415258833	쇼핑 서비스	동영상/방송 서비스	넷플릭스	유튜브	게임 서비스	배달 서비스
광진구	30 M		1.405234904	동영상/방송 서비스	쇼핑 서비스	넷플릭스	게임 서비스	유튜브	배달 서비스
송파구	30 M		1.403015862	쇼핑 서비스	배달 서비스	동영상/방송 서비스	게임 서비스	유튜브	넷플릭스

Code_ 추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

```
# 서비스별로 기술된 코드들을 반복한다.

# 연령대 합치는 함수 작성 (20대, 30대,...)
def merge_age(age):
    return age//10 * 10

# 10단위 연령대 컬럼 생성
average['연령대'] = average['연령대'].apply(merge_age)
average = average.groupby(['자치구', '행정동', '성별', '연령대'])[
    ['게임 서비스 사용일수', '금융 서비스 사용일수', '쇼핑 서비스 사용일수', '넷플릭스 사용일수',
    '동영상/방송 서비스 사용일수', '배달 서비스 사용일수', '유튜브 사용일수']
].mean()
average = average.reset_index()
average = average.sort_values(by=['자치구', '성별', '연령대'])

from sklearn.preprocessing import StandardScaler

# average
for index, row in region_age_sex.iterrows(): # 행들 차례대로 읽기
    df = pd.DataFrame()
    for i in range(5088): # 자치구, 성별, 연령대 같은 행들을 모아 데이터프레임 생성
        if (average.loc[i]['자치구'] == row['자치구']) & (average.loc[i]['성별'] == row['성별']) &
        (average.loc[i]['연령대'] == row['연령대']):
            df = pd.concat([df, average.loc[i:i]], ignore_index=True)
```

```
scaler = StandardScaler()

# 자치구, 성별, 연령대가 같은 데이터들에 대해 서비스 사용일수를 이용해 상관계수 구하기
sv = df[['게임 서비스 사용일수', '금융 서비스 사용일수', '쇼핑 서비스 사용일수', '넷플릭스 사용일수',
    '동영상/방송 서비스 사용일수', '배달 서비스 사용일수', '유튜브 사용일수']]
scaler.fit(sv)

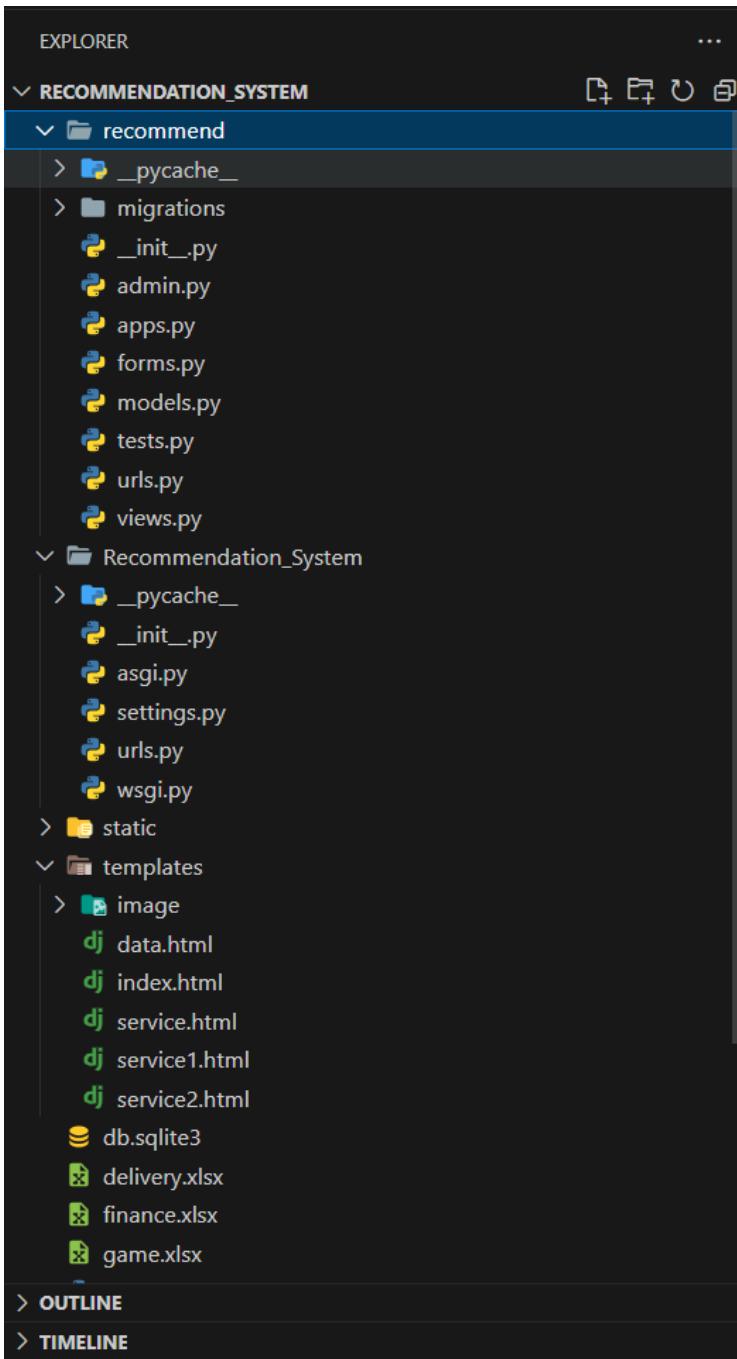
# 상관계수
corr_mat = sv.corr()
corr_mat = corr_mat.loc[['게임 서비스 사용일수']][['금융 서비스 사용일수', '쇼핑 서비스 사용일수',
    '동영상/방송 서비스 사용일수', '넷플릭스 사용일수', '유튜브 사용일수', '배달 서비스 사용일수']]
corr_mat = corr_mat.loc['게임 서비스 사용일수'].sort_values(ascending=False)
corr_mat = pd.DataFrame(corr_mat)

# corr_mat

for i in range(300):
    if (game.loc[i]['자치구'] == row['자치구']) & (game.loc[i]['연령대'] == row['연령대']) &
    (game.loc[i]['성별'] == row['성별']):
        # game.loc[i]
        corr_mat.index.values[0]
        game.loc[i, '1'] = corr_mat.index.values[0][-5]
        game.loc[i, '2'] = corr_mat.index.values[1][-5]
        game.loc[i, '3'] = corr_mat.index.values[2][-5]
        game.loc[i, '4'] = corr_mat.index.values[3][-5]
        game.loc[i, '5'] = corr_mat.index.values[4][-5]
        game.loc[i, '6'] = corr_mat.index.values[5][-5]
```

Django 서비스 구현

Django Project 폴더 구성



[프로젝트/앱 파일명]

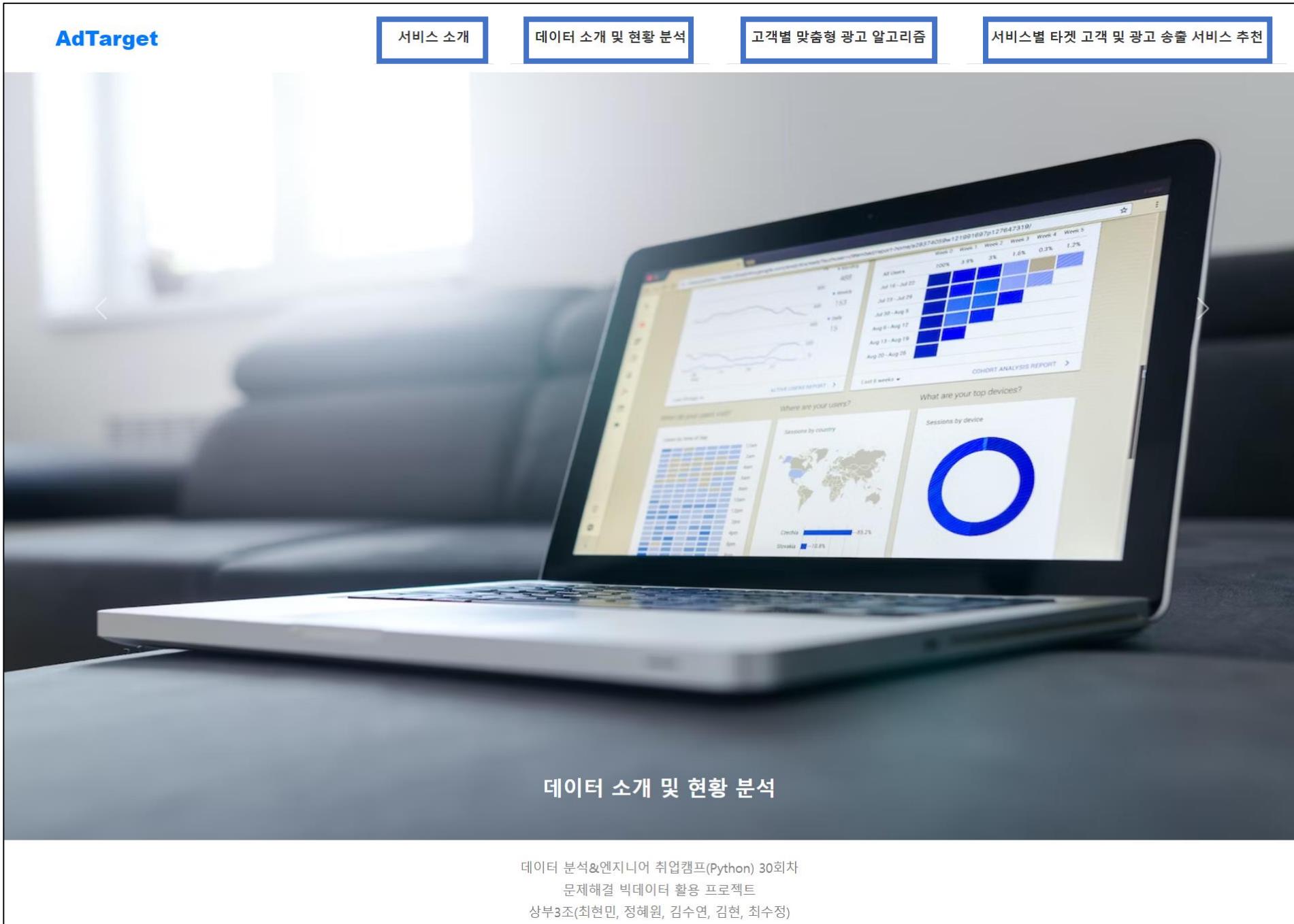
- ▶ Project : RECOMMENDATION_SYSTEM
- ▶ App : recommend

[파일 소개]

- ▶ Templates :
 - data.html : 데이터 소개 및 현황 분석
 - index.html : 메인 화면
 - service.html : 서비스 소개
 - service1.html : 고객별 맞춤형 광고 알고리즘 서비스
 - service2.html : 서비스별 타겟 고객 및 광고 송출 서비스

Django 서비스 구현

메인 화면



[주요 파트]

- ▶ 서비스 소개
- ▶ 데이터 소개 및 현황 분석
- ▶ 고객별 맞춤형 광고 알고리즘
- ▶ 서비스별 타겟 고객 별 광고 송출 서비스 추천

[기능 소개]

- ▶ 좌우로 화면 전환 가능
- ▶ 사진 클릭 시 관련 서비스 페이지 이동
- ▶ 사용자 원하는 정보 쉽게 확인 가능

Django 서비스 구현

Code_Index.html

```
<div class="carousel-item">
  <a href="service2">
    
  </a>
  <div class="carousel-caption d-none d-md-block">
    <h4><strong>서비스별 타겟 및 광고 송출 서비스 추천
      </strong></h4>
    </div>
  </div>
```

- ▶ 이용
- ▶ 이미지를 누르면 해당 페이지로 이동할수있게 구현
- ▶ “data” = data.html
- “service1” = service1.html
- “service2” = service2.html

Django 서비스 구현

서비스 소개

AdTarget

서비스 소개

데이터 소개 및 현황 분석

고객별 맞춤형 광고 알고리즘

서비스별 타겟 고객 및 광고 송출 서비스 추천

당신의 비즈니스를 위한 최적화된 솔루션, [AdTarget](#)을 소개합니다.

AdTarget은 현황 분석을 통해 국내 시장의 트렌드를 파악하고, 이를 기반으로 여러분의 광고 전략을 세우는데 도움을 줄 것입니다. 우리의 특별한 맞춤형 광고 알고리즘은 고객의 자치구, 연령대, 성별에 따라 각 개인에게 가장 적합한 광고를 제공합니다. 여러분의 광고는 게임부터 금융, 쇼핑, 동영상/방송, 유튜브, 넷플릭스, 배달 서비스에 이르기까지 다양한 서비스를 포함합니다. 또한, 우리는 각 서비스의 이용 수가 가장 많은 타겟 고객의 자치구, 연령대, 성별을 우선순위로 정렬하여, 최적의 광고 송출 타겟 고객을 추천해드립니다.

이제, [AdTarget](#)과 함께 효율적인 광고 전략을 구현해보세요.

Our Services

데이터 소개 및 현황 분석

고객별 맞춤형 광고 알고리즘

서비스별 타겟 고객 및 광고 송출 서비스 추천

데이터 분석&엔지니어 졸업캠프(Python) 30회차
문제해결 빅데이터 활용 프로젝트
상부3조(최현민, 정혜원, 김수연, 김현, 최수정)

[서비스 이름 및 소개]

- ▶ 서비스 이름 : "AdTarget"
 - "Advertisement target"의 축약
- ▶ 서비스 소개 문구를 통해 소비자가 서비스 파악 가능

[Our Services]

- ▶ 제공 서비스 종류 소개
- ▶ 버튼 클릭 시 해당 서비스 이동 기능 제공

Django 서비스 구현

Code_views.py

```
def index(request):
    return render(request, 'index.html')

def service(request):
    return render(request, 'service.html')

def data(request):
    return render(request, 'data.html')
```

▶ 각각 페이지 렌더링 함수

Django 서비스 구현

데이터 소개 및 현황 분석

The screenshot shows the AdTarget service interface. At the top, there are tabs: 'AdTarget' (highlighted in blue), '서비스 소개' (Service Introduction), '데이터 소개 및 현황 분석' (Data Introduction and Status Analysis, highlighted in blue), '고객별 맞춤형 광고 알고리즘' (Customer-specific advertising algorithm), and '서비스별 타겟 고객 및 광고 송출 서비스 추천' (Service-specific target customer and advertisement delivery service recommendation). The main content area has two sections:

- 데이터 소개**: This section contains text about the service's purpose, data sources, and analysis methods. It states that the service uses data from various sources like population density, age groups, gender, and service usage to analyze trends and correlations.
- 현황 분석**: This section lists five analysis items: '자치구 연령대별 인구 총합' (Total population by age group per district), '자치구 연령대별 남성 인구 총합' (Total male population by age group per district), '자치구 연령대별 여성 인구 총합' (Total female population by age group per district), '서비스별 상관관계' (Correlation between services), and '자치구별 평균연령' (Average age by district).

At the bottom, there is a footer note: '데이터 분석&엔지니어 취업캠프(Python) 30회차 문제해결 빅데이터 활용 프로젝트 상부3조(최현민, 정혜원, 김수연, 김현, 최수정)'.

[데이터 소개]

- ▶ 현재 사용 중 데이터 출처 확인 가능
- ▶ 현황 분석의 원천 데이터 확인 가능
- ▶ 투명성 보장 및 데이터 분석 결과 신뢰성 판단에 도움

[현황 분석]

- ▶ 버튼 클릭 시 해당 현황에 해당하는 그림 표출

Django 서비스 구현

현황 분석



[현황 분석]

▶ 표출 내용

- 자치구 연령대별 인구 총합
- 자치구 연령대별 남성 인구 총합
- 자치구 연령대별 여성 인구 총합
- 서비스별 상관관계
- 자치구별 평균 연령

▶ 버튼 클릭 시 시각화 자료(그래프 및 히트맵) 출력

▶ 각 자료 하단에 관련 정보 정리한 분석 결과 포함

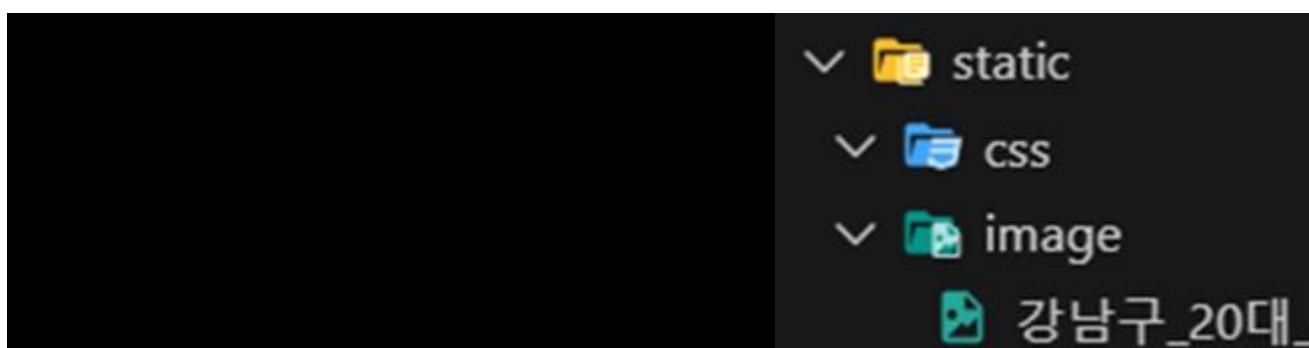
Django 서비스 구현

이미지(static 파일) 파일 불러오기

```
STATIC_URL = 'static/'  
# static 파일 요청이 오는 URL  
STATICFILES_DIRS = [BASE_DIR/'static']  
# 이 경로에서 static 파일을 찾는다.
```

[settings.py]

- ▶ django에서 image, css, js 파일은 static 파일에 저장



[static 파일 생성]

- ▶ django에서 image, css, js 파일은 static 파일에 저장

```

```

[data.html]

- ▶ img 정적 파일 주소를 동적으로 가져옴

```
{% load static %}  
<!DOCTYPE html>
```

- ▶ html 파일의 맨 위에 static module import 실행

Django 서비스 구현

서비스별 시나리오

[시나리오 1]

고객별 맞춤형 광고 알고리즘 서비스

강남구에 거주하는 20대 남성 고객과
구로구에 거주하는 40대 여성 고객이 있다.

이 고객들에게 광고 제공 시,
고객들이 많이 사용하는 서비스를
광고할수록 광고 효율이 올라가며
또한, 계절 또는 트렌드의 영향으로
월별로 사용하는 서비스의 우선순위가
달라질 수 있기 때문에
월별 분석 또한 필요하다.

[시나리오 2]

서비스별 타겟 고객 및 광고 송출 서비스

배달 서비스 회사인 ‘배달의민족’에서
광고 진행을 희망한다.

광고를 상위 타겟 고객들에 한해서
예산을 더 많이 투입해서 진행하고 싶다.

또한 상위 타겟 고객 군집에는
광고를 어느 서비스에서 송출할건지
또한 세분화시키려고 한다.

Django 서비스 구현

추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

The screenshot shows a Django-based web application interface for 'AdTarget'. The top navigation bar includes links for '서비스 소개', '데이터 소개 및 현황 분석', '고객별 맞춤형 광고 알고리즘' (highlighted in blue), and '서비스별 타겟 고객 및 광고 송출 서비스 추천'. The main content area has a light blue header titled '고객별 맞춤형 광고 알고리즘'. Below it is a box containing text about the system's purpose: '고객의 자치구, 연령대, 성별에 따라 고객 개개인의 광고 알고리즘에 넣을 서비스를 우선순위에 따라 추천해 드리는 시스템입니다. 또한, 해당 고객의 자치구, 연령대, 성별 정보에 따라 2023년 1월부터 9월까지 해당 고객 군집의 서비스별 사용일수 추이를 보여드립니다.' A form below allows users to input their location, gender, and age group, with dropdown menus for '자치구' (Gangnam-gu), '성별' (Male), and '연령대' (20s). A 'Submit' button is at the bottom. At the very bottom of the page, there is footer text: '데이터 분석&엔지니어 취업캠프(Python) 30회차', '문제해결 빅데이터 활용 프로젝트', and '상부3조(최현민, 정혜원, 김수연, 김현, 최수정)'.

[고객별 맞춤형 광고 알고리즘]

- ▶ 고객 거주 자치구, 연령대, 성별 입력
- ▶ 광고 알고리즘에 의해 우선순위에 따라 추천
- ▶ 2023 1~9월 해당 고객 군집의 서비스별 사용일수 시각화 자료(추이 그래프)를 나타내 줌

Django 서비스 구현

결과 화면_추천 서비스 시스템 [1. 고객별 맞춤형 광고 알고리즘]

강남구에 거주하는 20대의 남성이 가장 많이 사용하는 월별 서비스의 순위입니다.

해당 고객의 사용 서비스 우선순위에 따라 해당 고객에게 나타날 광고 서비스의 우선순위는 아래와 같습니다.

#	1순위	2순위	3순위	4순위	5순위	6순위	7순위
1월	게임 서비스	유튜브	배달 서비스	동영상/방송 서비스	넷플릭스	금융 서비스	쇼핑 서비스
2월	유튜브	게임 서비스	배달 서비스	넷플릭스	동영상/방송 서비스	금융 서비스	쇼핑 서비스
3월	게임 서비스	유튜브	넷플릭스	배달 서비스	동영상/방송 서비스	금융 서비스	쇼핑 서비스
4월	게임 서비스	유튜브	넷플릭스	동영상/방송 서비스	배달 서비스	금융 서비스	쇼핑 서비스
5월	게임 서비스	유튜브	동영상/방송 서비스	넷플릭스	배달 서비스	금융 서비스	쇼핑 서비스
6월	게임 서비스	넷플릭스	유튜브	동영상/방송 서비스	배달 서비스	금융 서비스	쇼핑 서비스
7월	게임 서비스	넷플릭스	유튜브	동영상/방송 서비스	배달 서비스	금융 서비스	쇼핑 서비스
8월	게임 서비스	유튜브	넷플릭스	동영상/방송 서비스	배달 서비스	금융 서비스	쇼핑 서비스
9월	게임 서비스	넷플릭스	유튜브	동영상/방송 서비스	배달 서비스	금융 서비스	쇼핑 서비스

데이터 분석 & 엔지니어 쿠얼캠프(Python) 30회차
문제해결 빅데이터 활용 프로젝트
상부3조(최혁민, 정혜원, 김수연, 김현, 최수정)

[결과 화면]

- ▶ 광고 알고리즘 결과 화면
- ▶ 2023년 1~9월 데이터 기반의 순위별 우선순위 보여줌
- ▶ 각 서비스별 해당 기간 동안의 변화 그래프 출력
- ▶ 개인화된 어떤 광고 서비스가 가장 효과적인지 결정에

도움을 줌

Django 서비스 구현

Code_views.py

```
df = pd.read_excel("./zvalue_month.xlsx")
```

[Z-value 처리 파일 불러오기]

```
def service1(request):
    if request.method == 'POST':
```

[웹 요청(request)을 인자로 받아 처리]

- ▶ if request.method == 'POST'는 사용자의 요청이 POST 방식인지를 확인

```
자치구 = request.POST.get('자치구')
성별 = request.POST.get('성별').upper()
연령대 = int(request.POST.get('연령대'))
```

[사용자로부터 입력 받는 부분]

- ▶ 사용자가 웹 페이지에서 입력한 '자치구', '성별', '연령대' 값을 받아옴
- ▶ 이 과정에서 '연령대' 값은 숫자 변환 필요

```
target_data = df[(df['자치구'] == 자치구)
& (df['성별'] == 성별) & (df['연령대'] == 연령대)]
```

[사용자가 입력한 정보에 따라 데이터를 필터링하는 부분]

- ▶ 사용자가 입력한 '자치구', '성별', '연령대'에 해당하는 데이터를 데이터프레임에서 찾음

```
return render(request, 'service1.html',
{'data': data, 'service_data': service_data,
'자치구': 자치구, '연령대': 연령대, '성별':성별,
'services': services, 'first': first})
```

[데이터를 화면에 출력하는 부분]

- ▶ 필터링된 데이터를 사용자에게 보여주는 작업을 수행
- ▶ 사용자가 입력한 '자치구', '성별', '연령대'에 대한 정보와 함께, 해당 정보에 따른 서비스 우선순위 데이터를 화면에 출력

Django 서비스 구현

추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

The screenshot shows a Django-based web application interface for 'AdTarget'. The top navigation bar includes links for '서비스 소개', '데이터 소개 및 현황 분석', '고객별 맞춤형 광고 알고리즘', and '서비스별 타겟 고객 및 광고 송출 서비스 추천'. The main content area has a light blue header with the text '서비스별 타겟 고객 및 광고 송출 서비스'. Below this is a descriptive box containing text about the system's purpose: '게임, 금융, 쇼핑, 동영상/방송, 유튜브, 넷플릭스, 배달 서비스 광고주분들을 위해 서비스의 이용수가 가장 많은 타겟 고객의 자치구, 연령대, 성별을 우선순위 순으로 정렬해드리며, 맞춤형 광고 송출 대상 서비스를 추천해드리는 시스템입니다.' A large input form is centered, prompting users to '서비스의 종류를 선택해주세요.' with a dropdown menu currently showing '배달 서비스' and a 'Submit' button. At the bottom of the page, there is footer information: '데이터 분석&엔지니어 취업캠프(Python) 30회차', '문제해결 빅데이터 활용 프로젝트', and '상부3조(최현민, 정혜원, 김수연, 김현, 최수정)'.

[서비스별 타겟 고객 및 광고 송출 서비스 추천]

- ▶ 7개 서비스 중 선택한 서비스에 대해 해당 서비스 이용수가 가장 많은 타겟 고객의 자치구, 연령대, 성별 우선순위 정렬한 정보 제공
- ▶ 타겟 고객별로 선택한 서비스와 상관관계 높은 타 서비스 순으로 맞춤형 광고 송출 대상 서비스 추천하는 시스템임.

Django 서비스 구현

결과 화면_추천 서비스 시스템 [2. 서비스별 타겟 고객 및 광고 송출 서비스 추천]

<p>배달 서비스를 가장 많이 이용하는 고객은 강북구 남성 30대입니다.</p> <p>해당 고객에 대해 배달 서비스와 상관관계가 가장 높은 타서비스는 게임 서비스입니다.</p> <p>배달 서비스의 타겟 고객 우선순위와 타겟 고객별 광고 송출 서비스 추천 우선순위는 아래와 같습니다.</p>	
타겟 고객 우선순위	배달 서비스 타겟 고객별 광고 송출 서비스 추천
1순위 강북구 남성 30대	1순위: 게임 서비스 2순위: 금융 서비스 3순위: 쇼핑 서비스 4순위: 유튜브 5순위: 동영상/방송 서비스 6순위: 넷플릭스
2순위 관악구 남성 30대	1순위: 게임 서비스 2순위: 금융 서비스 3순위: 쇼핑 서비스 4순위: 유튜브 5순위: 동영상/방송 서비스 6순위: 넷플릭스
3순위 종로구 남성 30대	1순위: 게임 서비스 2순위: 금융 서비스 3순위: 쇼핑 서비스 4순위: 유튜브 5순위: 동영상/방송 서비스 6순위: 넷플릭스

[결과 화면]

- ▶ 사용자 입력 특정서비스에 대해 타겟 고객 추천
- ▶ 추천 타겟 고객이 선호하는 타 서비스 또한 상관관계를 통해 추천 기능 제공
- ▶ 광고주에게 타겟 고객 선정에 대한 효율적인 의사결정에 도움

Django 서비스 구현

Code_views.py

```
def service2(request):
    if request.method == 'POST':
        service_files = {
            '배달 서비스': "./delivery.xlsx",
            '금융 서비스': "./finance.xlsx",
            '게임 서비스': "./game.xlsx",
            '넷플릭스': "./netflix.xlsx",
            '쇼핑 서비스': "./shopping.xlsx",
            '유튜브': "./youtube.xlsx",
            '동영상/방송 서비스': "./video.xlsx"
        }
```

[서비스와 파일 경로 연결]

- ▶ 함수 내부에 클라이언트가 요청한 서비스와 해당 서비스에 대한 파일 경로를 매칭시키는 딕셔너리를 생성
- ▶ 클라이언트의 요청에 맞는 파일 경로를 가져올 수 있음.

```
sorted_target = df.groupby(['자치구', '성별', '연령대'])
[f'{서비스} 사용일수'].mean().sort_values(ascending=False)
target_services = df.iloc[:, 5:]
```

[서비스 사용일수가 많은 순서대로 정렬]

- ▶ 불러온 데이터를 '자치구', '성별', '연령대'로 그룹화
- ▶ 각 그룹의 서비스 사용일수 평균을 계산한 후 내림차순으로 정렬
- ▶ target_services에 어떤 서비스를 타겟 고객에게 추천할지 결정할 수 있음

```
recommended_targets = sorted_target[:10].index.tolist()
```

[상위 10개의 타겟을 추천]

- ▶ 정렬된 결과에서 상위 10개를 선택하여 추천 대상으로 설정

Django 서비스 구현

Code_views.py

```
targets = []
other_services = []
    for i, target in enumerate(recommended_targets, 1):
        targets.append(f"{i}순위 {target[0]} {'남성' if
target[1] == 'M' else '여성'} {target[2]}대")
        for i, service in
enumerate(target_services.values[0], 1):
            other_services.append(f"{i}순위: {service}")
```

[상위 그룹과 관련 서비스를 순위별로 정리하여 출력]

- ▶ targets 리스트 : 사용자 그룹(자치구, 성별, 연령대)을 서비스 사용일수가 많은 순서대로 순위를 매겨 저장
- ▶ enumerate(recommended_targets, 1) : 추천천 대상 리스트를 순회, 각 대상의 순위(i)와 정보(target)를 반환. 정보는 "순위 자치구 성별 연령대" 형태의 문자열로 변환되어 targets 리스트에 추가.
- ▶ other_services 리스트 : 대상 서비스를 순위별로 저장
- ▶ target_services.values[0] : 서비스 사용일수 데이터의 첫 번째 행 (사용일수가 가장 많은 서비스)을 의미 이를 순회하면서, 각 서비스의 순위(i)와 이름(service)을 반환 정보는 "순위: 서비스 이름" 형태의 문자열로 변환되어 other_services 리스트에 추가

```
return render(request, 'service2.html', {'top_service':
top_service, 'first_customer':first_customer, 'top':top,
'first': first, 'others':others, '서비스': 서비스, 'targets':
targets, 'other_services': other_services})
```

[데이터를 화면에 출력하는 부분]

- ▶ 필요한 데이터를 'service2.html' 템플릿에 전달하며 페이지를 렌더링합니다

07

결론 및 향후 과제

[결론]

- ▶ 기업 입장에서 원하는 타겟 고객의 선호도를 기반으로 광고를 송출하여 광고 효율을 높임
- ▶ 특정 서비스 기업에서 타겟 고객층을 유의미한 수준으로 선별하고 상관관계 분석을 통하여 광고 효율 증대와 불필요한 광고 비용 감소
- ▶ 개별 데이터 수집 등의 고객 대상으로 자동화된 광고 추천 및 송출 가능

[향후 과제]

- ▶ 월별 데이터셋을 늘려 분석 기간을 확장, 시계열 분석 통한 월별, 계절별 변동 파악 및 추세 예측을 통한 알고리즘 개선
- ▶ 데이터셋에 존재한 다양한 이동 데이터 사용한 동선 기반의 광고 송출 시스템 구축 가능

08

프로젝트 후 소감

	최현민	정혜원	김수연	김현	최수정
좋은점	직접 프로젝트를 수행하며 머리 속의 이론과 내용이 정리되어 좋았다. 조장으로써 프로젝트 기간동안 많은 부분을 조율하고 진행하고 일부분은 독단적으로 밀어 붙인 내용도 있었는데 조원 분들이 다들 잘따라와 주시고 맑은 파트에 대해 시간과 노력을 다해주어서 리더십을 더욱 기를 수 있었다.	거의 모든 프로젝트 과정을 팀원들이 같이 해보고 회의를 자주 진행하는 방식으로 진행을 했는데 이러한 방식이 프로젝트의 전 과정을 이해하고 진행하는데 도움이 되었던 것 같습니다. 또한 회의를 통해서 각 파트의 부족한 부분들에 대한 피드백을 받아 더 완성도 있는 프로젝트를 만들 수 있었던 것 같습니다.	수업에서 배운 머신 러닝 기법들을 실제 프로젝트에 적용해보며 추후 실무에 도움이 될 것 같아서 좋았고, 팀원들과 함께 피드백을 주고 받으며 문제를 정의하고 해결해 나가는 과정이 재밌었습니다.	프로젝트를 통해 추상적이었던 개념들을 실제로 적용해 볼 수 있다는 점이 좋았다. 이론적인 내용들을 실제 데이터에 적용하면서, 그 결과를 분석하고 해석하는 과정이 재미있었다.	팀프로젝트를 통해 내가 부족한 부분도 알수 있었고, 막히는 부분도 팀원 분들과 소통하면서 해결해 나갈수 있어서 좋았습니다. 내 코드 뿐만 아니라 다른 팀원들의 코드도 공유하면서 간결하게 코드를 짜는 법을 배울수 있었습니다.
아쉬운점	나의 실력을 깨닫게 되었고, 부족한 점에 대해서 알게되어 더욱 정진 해야겠다는 생각이 들었다. 좋은 데이터 셋이니 이동관련도 추후에라도 활용해서 결과물을 만들어내면 좋을 것 같다.	아무래도 세미 프로젝트이다 보니 시간적인 제약이 있어서 시간/거리 관련 피처들을 추천 시스템에 적용하여 구현하지 못한 점이 아쉽습니다.	딥러닝 기법 LSTM을 이용해서 시계열 분석까지 해보고 싶었지만 시간 상의 여유가 없었고, 개념을 정확히 이해하지 못해서 시계열 분석을 수행하지 못한 점이 아쉽습니다.	제한된 시간으로 인해 이동 시간대까지는 다루지 못하게 아쉬웠다. 이후 기회가 된다면, 팀원들과 같이 해보면 좋을 것 같다.	첫 프로젝트이다보니 부족한 점이 많았습니다. 팀원분들의 코드를 공부하면서 진행했는데 데이터 시각화 위주로 참여하였습니다. 다음 프로젝트에선 좀 더 공부하여 다양한 파트에 참여해보고 싶습니다.
느낀점	좋은 조원분들을 만난 것 같다. 프로젝트의 진행에 있어 각자의 장점을 살려 진행하는 것도 업무에서 중요한 부분이라고 생각한다. 이번 프로젝트에서 나또한 많이 배웠고, 다른 조원분에게도 나의 경험이 많이 전해졌으면 좋겠다.	데이터와 관련하여 처음 해보는 팀 프로젝트라 부족한 점도 있겠지만 팀원분들과 협력하여 좋은 결과물을 만들어낸 것 같아서 뿌듯하고 이론으로 배웠던 내용들을 직접 데이터를 다루면서 적용해보니까 이해도 더 잘 되고 심화된 학습을 할 수 있었던 것 같습니다.	프로젝트 실습 전에는 모든 것이 막막했고, 잘 해내지 못할 거라는 막연한 불안감이 있었는데, 막상 해보니 프로젝트가 잘 진행되는 것을 보며 약간은 자신감이 생긴 것 같습니다.	프로젝트를 진행하면서 어려운 과제들을 마주하게 되었었는데, 이를 해결하고 과정에서 큰 성취감과 성장의 기회를 느낄 수 있었다.	프로젝트를 통해 한번 더 복습 할 수 있었던 기회가 되었고 같고, 처음하는 팀 프로젝트로 흐름이나 진행과정을 경험해 볼수 있어서 좋았습니다. 혼자라면 할 수 없었겠지만, 팀원분들 덕분에 부족한 부분을 알 수 있었고, 많이 배울 수 있었습니다.

Q&A

THANKS