2.1. 데이터 EDA

2.1.1. 데이터 수집

데이터 명	데이터 형식	데이터 내용	출처
서울교통공사_역주소 및 전화번호_20220314	csv(정형)	지하철역 도로명주소	서울 열린 데이터 광장
서울시 병의원 위치 정보	csv(정형)	병의원 좌표, 도로명주소	서울 열린 데이터 광장
행정동별 주민등록인구 연령별 통계	csv(정형)	연령별 주민등록인구 통계	공공데이터포털
데이터 동별 1인가구	xls(정형)	서울시 동별 1인, 2인, 3인, 4인 통계	서울 열린 데이터 광장
서울시 주택 보급률	csv(정형)	주택 보급률, 면적, 보급실수	공공데이터포털
서울시 건축물대장 법정동 코드정보	csv(정형)	전국 행정동, 법정동 코드정보	서울 열린 데이터 광장
행정동별 서울 생활인구(202111)	csv(정형)	서울시 행정동별 시간대구분 생활인구	공공데이터포털
report	txt(비정형)	주차장 면적, 수	서울 열린 데이터 광장
2019~2022년도 역세권 청년주택 공급현황 및 계획	정형	역세권 청년주택 공급현황	역세권청년주택
LSMD_ADM_SECT_UMD_11	shp(정형)	서울시 행정동 구분 지도 데이터	국가공간정보포털

2.1.2. 데이터 전처리

2.1.2.1. 수집 데이터 Excel 활용 불필요한 컬럼 제거.
행·열 정렬, 함수를 사용하여 중복된 인덱스 하나로 통합.

2.1.2.2. ‘Google Colaboratory’ 활용하여 2.1.2.1. 과정을 거친 데이터 로드 후 변환.

2.1.2.3. 데이터 전처리

- 서울교통공사_역주소 및 전화번호_20220314.csv -

```
alarm = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/지하철역 위치.shp.csv', encoding = 'cp949')
alarm['행정동'] = alarm['지번주소'].str[6:]  # 지번주소의 6번째 이후의 데이터를 만들어서 저장
alarm['행정동'] = alarm['행정동'].str.lstrip()  # 앞 공백을 제거
alarm['행정동'].str.split(" ").head()  # 공백을 기준으로 분할
dong = alarm['행정동'].str.split(" ", expand=True)  # expand = True 데이터 프레임 형태로 저장
alarm['행정동'] = dong[1]  # dong의 두번째 열을 행정동으로 지정
dataframe = pd.DataFrame(alarm)
dataframe = dataframe.drop([dataframe.index[268], dataframe.index[269],
                            dataframe.index[270], dataframe.index[271],
                            dataframe.index[272], dataframe.index[273],
                            dataframe.index[274]])  # 경기도 제거
dataframe_n = pd.DataFrame(dataframe['행정동'].value_counts())  # '행정동' 컬럼을 기준으로 각 행의 빈도수 나타내기
dataframe_n = dataframe_n.reset_index()
dataframe_n.rename(columns = {'index':'행정동명'}, inplace = True)  # 컬럼 이름 재설정
dataframe_n.rename(columns = {'행정동':'역 개수'}, inplace = True)  # 컬럼 이름 재설정
dataframe_n # 최종 데이터
```

서울교통공사_역주소 및 전화번호_20220314

=> dataframe_n

- 서울시 병의원 위치 정보.csv -

```
alarm_1 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/행정동별 서울시 병의원 정보.csv', encoding = 'cp949')
alarm_1 = pd.DataFrame(alarm_1)
alarm_1 = alarm_1[ : 225]  # 병원 개수 10개 미만인 행 삭제
alarm_1 = alarm_1.drop(index = 1) # 결측치 존재하는 행 번호 지정 후 제거
alarm_1.rename(columns = {'index':'행정동명'}, inplace = True)  # 컬럼 이름 재설정
alarm_1.rename(columns = {'입력주소':'병원 개수'}, inplace = True)  # 컬럼 이름 재설정
alarm_1
```

서울시 병의원 위치 정보

=> alarm_1

- 행정동별 주민등록인구 연령별 통계.csv -

```
alarm_2 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/행정동별 주민등록인구 연령별 통계.csv', encoding = 'cp949')
alarm_2 = pd.DataFrame(alarm_2)
alarm_2.rename(columns = {'동':'행정동명'}, inplace = True)  # 컬럼명 변경
alarm_2 = alarm_2.drop(['기간', '계'], axis = 1)  # 필요없는 컬럼 제거
alarm_2['청년인구 수'] = alarm_2['20~24세'] + alarm_2['25~29세'] + alarm_2['30~34세'] + alarm_2['35~39세']  # 열 생성 & 연산
alarm_2 = alarm_2.drop(columns = ['20~24세', '25~29세', '30~34세', '35~39세'], axis = 1)  # 필요없는 컬럼 제거
alarm_2
```

행정동별 주민등록인구 연령별 통계

=> alarm_2

- 데이터 동별 1인가구.xls -

```
alarm_3 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/행정동별 1인,2인가구 현황.csv', encoding = 'utf-8')
alarm_3 = alarm_3.drop(columns = ['2인가구'], axis = 1)  # 필요없는 컬럼 제거
alarm_3.rename(columns = {'행정동':'행정동명'}, inplace = True)  # 컬럼명 변경
alarm_3
```

데이터 동별 1인가구

=> alarm_3

- 서울시 주택 보급률.csv -

```
alarm_4 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/행정동별 주택 보급률.csv', encoding = 'cp949')
div = alarm_4.loc[:, ['주택보급률']]
div = div / 100
alarm_4.loc[:,'주택보급률(%)'] = div  # 주택보급률 백분율 컬럼 추가(인덱싱)
alarm_4 = alarm_4.drop(columns = ['주택보급률','일반가구수', '주택수', '기간'])  # 기존 주택보급률 컬럼 제거
alarm_4 = alarm_4.rename(columns = {'동':'행정동명'})  # 컬럼명 변경
alarm_4
```

서울시 주택 보급률

=> alarm_4

- 서울시 건축물대장 법정동 코드정보.csv + 행정동별 서울 생활인구(202111).csv -

- 행정동을 기준으로 하는 분석 데이터 통합 과정 -

```
alarm_5 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/서울시 건축물대장 법정동 코드정보.csv', encoding = 'utf-8')
alarm_5
```

```
alarm_5 = alarm_5.astype({'행정동코드':'str'})  # 행정동코드 문자형 변환
alarm_5 = alarm_5.astype({'시군구코드':'str'})  # 시군구코드 문자형 변환
alarm_5.dtypes  # 데이터프레임 데이터 형태 확인
```

```
시군구코드    object
법정동코드     int64
행정동코드    object
시도명       object
시군구명      object
법정동명      object
행정동명      object
dtype: object
```

행정동별 서울 생활인구(202111).csv에 있는 8자리 행정동코드를 기준으로 두 데이터프레임을 병합하기 위해 형변환 진행.

```
[10] alarm_5['code'] = alarm_5['시군구코드'] + alarm_5['행정동코드']  # 시군구코드, 행정동코드 문자열 붙이기
alarm_5
```

	시군구코드	법정동코드	행정동코드	시도명	시군구명	법정동명	행정동명	code
0	11740	10300	526	서울특별시	강동구	상일동	상일제2동	11740526
1	11740	10300	525	서울특별시	강동구	상일동	상일제1동	11740525
2	11290	10300	620	서울특별시	성북구	돈암동	정릉제1동	11290620
3	11530	11200	800	서울특별시	구로구	항동	항동	11530800
4	11305	10300	635	서울특별시	강북구	수유동	수유3동	11305635
...	...	...	...	...	...	...	...	...
1232	11110	14500	540	서울특별시	종로구	송현동	삼청동	11110540
1233	11110	18400	550	서울특별시	종로구	부암동	부암동	11110550
1234	11230	11000	740	서울특별시	동대문구	이문동	이문제1동	11230740
1235	11440	12100	670	서울특별시	마포구	동교동	동교동	11440670
1236	11440	12200	680	서울특별시	마포구	합정동	합정동	11440680

1237 rows × 8 columns

```
alarm_5 = alarm_5.astype({'code':'int'})  # 위에서 붙인 컬럼'code' 정수형 변환
alarm_5.dtypes
```

```
alarm_6 = pd.read_csv('/content/gdrive/My Drive/datasciencebasis/행정동별 서울 생활인구(202111).csv',
                      encoding = 'utf-8')  # 행정동 서울 생활인구
alarm_6.rename(columns={'행정동코드':'code'}, inplace=True)  # alarm_5, alarm_6 병합 위해서 컬럼명 재설정

alarm_7 = pd.merge(alarm_6, alarm_5, how = 'left', on='code')  # 컬럼 'code' 기준으로 병합
alarm_7 = alarm_7.drop(columns = ['남자20세부터24세생활인구수', '남자25세부터29세생활인구수', '남자30세부터34세생활인구수',
                                  '남자35세부터39세생활인구수', '여자20세부터24세생활인구수', '여자25세부터29세생활인구수',
                                  '여자30세부터34세생활인구수', '여자35세부터39세생활인구수', '시도명'])  # 연령별 컬럼 제거
alarm_7.drop_duplicates(['행정동명'])  # 행정동명 중복행 제거
```

	기준일ID	시간대구분	code	총생활인구수	시군구코드	법정동코드	행정동코드	시군구명	법정동명	행정동명
0	20211101	0	11470680	22796.1407	11470	10100	680	양천구	신정동	신정7동
1	20211101	0	11440730	37070.1563	11440	12600	730	마포구	중동	성산제2동
3	20211101	0	11200590	17952.5176	11200	10900	590	성동구	금호동1가	금호1가동
4	20211101	0	11680656	30585.8034	11680	11800	656	강남구	도곡동	도곡제2동
5	20211101	0	11680730	16468.2665	11680	11400	730	강남구	일원동	일원제1동
...	...	...	...	...	...	...	...	...	...	...
768	20211101	0	11230710	18470.7047	11230	10800	710	동대문구	회기동	회기동
769	20211101	0	11500640	22468.1467	11500	10900	640	강서구	방화동	방화제2동
771	20211101	0	11380650	11016.0757	11380	10100	650	은평구	수색동	수색동
772	20211101	0	11710566	14836.3812	11710	11100	566	송파구	방이동	오륜동
773	20211101	0	11560670	6608.6909	11560	13300	670	영등포구	대림동	신길제5동

424 rows × 10 columns

```
alarm_7.dtypes
time_list = [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21] 생활인구의 활동시간을 9 - 21시로 지정하는 리스트 생성 후
today_alarm_7 = alarm_7[alarm_7['시간대구분'].isin(time_list)]  데이터프레임에 적용
today_alarm_7.drop(columns = ['시군구코드', '법정동코드', '행정동코드', '법정동명', 'code'])  # 필수 컬럼 제외 후 제거
```

```
ID_list = [20211106, 20111107, 20111113, 20111114,
           20111120, 20111121, 20111127, 20111128]  # 평일 9 - 21시에 활동하는 동별 생활인구 수 추출
today_alarm_7[~today_alarm_7['기준일ID'].isin(ID_list)]
t_work = today_alarm_7[~today_alarm_7['기준일ID'].isin(ID_list)]
t_work
```

	기준일ID	시간대구분	code	총생활인구수	시군구코드	법정동코드	행정동코드	시군구명	법정동명	행정동명
6975	20211101	9	11215840	20841.5609	11215	10600	840	광진구	노유동	자양제3동
6976	20211101	9	11215840	20841.5609	11215	10500	840	광진구	자양동	자양제3동
6977	20211101	9	11680730	18710.1746	11680	11400	730	강남구	일원동	일원제1동
6978	20211101	9	11680730	18710.1746	11680	11500	730	강남구	수서동	일원제1동
6979	20211101	9	11500560	17654.0771	11500	10300	560	강서구	화곡동	화곡제3동
...	...	...	...	...	...	...	...	...	...	...
556445	20211130	21	11140520	15262.4543	11140	11400	520	중구	태평로2가	소공동
556446	20211130	21	11140520	15262.4543	11140	11500	520	중구	남대문로2가	소공동
556447	20211130	21	11140520	15262.4543	11140	11100	520	중구	소공동	소공동
556448	20211130	21	11740700	29675.2626	11740	10600	700	강동구	둔촌동	둔촌제2동
556449	20211130	21	11740700	29675.2626	11740	10500	700	강동구	길동	둔촌제2동

292175 rows × 10 columns

```
ID_list = [20211106, 20111107, 20111113, 20111114,
           20111120, 20111121, 20111127, 20111128]  # 주말 9 - 21시에 활동하는 동별 생활인구 수 추출
today_alarm_7[today_alarm_7['기준일ID'].isin(ID_list)]
t_holi = today_alarm_7[today_alarm_7['기준일ID'].isin(ID_list)]
t_holi
```

	기준일ID	시간대구분	code	총생활인구수	시군구코드	법정동코드	행정동코드	시군구명	법정동명	행정동명
99975	20211106	9	11110515	17667.0659	11110	10500	515	종로구	창성동	청운효자동
99976	20211106	9	11110515	17667.0659	11110	10300	515	종로구	궁정동	청운효자동
99977	20211106	9	11110515	17667.0659	11110	11900	515	종로구	세종로	청운효자동
99978	20211106	9	11110515	17667.0659	11110	10800	515	종로구	통인동	청운효자동
99979	20211106	9	11110515	17667.0659	11110	10400	515	종로구	효자동	청운효자동
...	...	...	...	...	...	...	...	...	...	...
110045	20211106	21	11740660	28523.8167	11740	10800	660	강동구	성내동	성내제3동
110046	20211106	21	11740685	55704.4865	11740	10500	685	강동구	길동	길동
110047	20211106	21	11740690	4488.6801	11740	10600	690	강동구	둔촌동	둔촌제1동
110048	20211106	21	11740700	26133.3066	11740	10600	700	강동구	둔촌동	둔촌제2동
110049	20211106	21	11740700	26133.3066	11740	10500	700	강동구	길동	둔촌제2동

10075 rows × 10 columns

```
alarm_8 = alarm_7[(alarm_7['행정동명'] == '제기동') | (alarm_7['행정동명'] == '홍제제1동') | (alarm_7['행정동명'] == '홍제제2동'
alarm_8['행정동명'] = alarm_8['행정동명'].str.replace('홍제제1동', '홍제1동')
alarm_8['행정동명'] = alarm_8['행정동명'].str.replace('홍제제2동', '홍제2동')
alarm_8['행정동명'] = alarm_8['행정동명'].str.replace('홍제제3동', '홍제3동')
alarm_8
```

ex) '홍제1동'을 '홍제제1동'으로 지정된 경우
'홍제1동'으로 통일하기 위해 전처리

```
alarm_9 = alarm_7[(alarm_7['행정동명'] != '제기동') & (alarm_7['행정동명'] != '홍제제1동') & (alarm_7['행정동명'] != '홍제제2동'
alarm_9['행정동명'] = alarm_9['행정동명'].str.replace('제', '')
alarm_10 = pd.concat([alarm_9, alarm_8], axis = 0)
alarm_10

alarm_10 = alarm_10.drop(['code', '시군구코드', '법정동코드', '행정동코드','법정동명'], axis = 1)  # 생활시간(9 - 21) 데이터 추출
time_list = [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21]
alarm_10 = alarm_10[alarm_10['시간대구분'].isin(time_list)]
alarm_10
```

	기준일ID	시간대구분	총생활인구수	시군구명	행정동명
6975	20211101	9	20841.5609	광진구	자양3동
6976	20211101	9	20841.5609	광진구	자양3동
6977	20211101	9	18710.1746	강남구	일원1동
6978	20211101	9	18710.1746	강남구	일원1동
6979	20211101	9	17654.0771	강서구	화곡3동
...	...	...	...	...	...
555473	20211130	20	31044.8733	동대문구	제기동
555853	20211130	21	10829.6227	서대문구	홍제2동
555858	20211130	21	31551.3058	동대문구	제기동
556005	20211130	21	16913.6564	서대문구	홍제3동
556250	20211130	21	24522.5805	서대문구	홍제1동

302250 rows × 5 columns

```
### 평일 9 - 21시에 활동하는 동별 생활인구 수 추출
final = pd.DataFrame(alarm_10)
ID_list = [20211106, 20111107, 20111113, 20111114, 20111120, 20111121, 20111127, 20111128]
final[~final['기준일ID'].isin(ID_list)]
final_week = final[~final['기준일ID'].isin(ID_list)]
# '행정동명' 컬럼을 기준으로 총생활인구수의 평균 집계
final_weekday = pd.DataFrame(final_week.groupby(final_week['행정동명'])['총생활인구수'].mean())
final_weekday = final_weekday.reset_index()  # 행정동별 평일 9 - 21시 총생활인구수
final_weekday = final_weekday.rename(columns = {'총생활인구수':'평일 총생활인구수'})
final_weekday
```

```
### 주말 9 - 21시에 활동하는 동별 생활인구 수 추출
ID_list = [20211106, 20111107, 20111113, 20111114, 20111120, 20111121, 20111127, 20111128]
final[final['기준일ID'].isin(ID_list)]
final_holi = final[final['기준일ID'].isin(ID_list)]
final_holi = pd.DataFrame(final.groupby(final['행정동명'])['총생활인구수'].mean())
final_holiday = final_holi.reset_index()  # 행정동별 주말 9 - 21시 총생활인구수
final_holiday = final_holiday.rename(columns = {'총생활인구수':'주말 총생활인구수'})
final_holiday
```

- 전처리 데이터 통합 => 최종 분석 데이터

```
### 9 - 21시 행정동별 데이터 병합
final_dat = pd.merge(final_holiday, final_weekday, how = 'left', on='행정동명')
final_dat
final_dat1 = pd.merge(final_dat, alarm_2, how = 'left', on = '행정동명')  # pop_age.csv
final_dat2 = pd.merge(final_dat1, alarm_3, how = 'left', on = '행정동명')  # pop_youth.csv
final_dat3 = pd.merge(final_dat2, alarm_4, how = 'left', on = '행정동명')  # housing.csv
final_dat3.drop(columns = {'자치구_x','자치구_y'}, inplace = True)
final_dat3 = final_dat3[['자치구', '행정동명', '평일 총생활인구수',
                         '주말 총생활인구수', '청년인구 수', '1인가구', '주택보급률(%)']]
final_dat3
```

```
final_dat3[final_dat3.duplicated(['행정동명'])==True]  # 중복 데이터 확인
```

	자치구	행정동명	평일 총생활인구수	주말 총생활인구수	청년인구 수	1인가구	주택보급률(%)
245	강남구	신사동	36706.853724	36665.332604	9712.5	7282.0	0.772
246	관악구	신사동	36706.853724	36665.332604	9712.5	2047.0	0.337
247	강남구	신사동	36706.853724	36665.332604	9712.5	2047.0	0.772
248	관악구	신사동	36706.853724	36665.332604	4371.5	7282.0	0.337
249	강남구	신사동	36706.853724	36665.332604	4371.5	7282.0	0.772
250	관악구	신사동	36706.853724	36665.332604	4371.5	2047.0	0.337
251	강남구	신사동	36706.853724	36665.332604	4371.5	2047.0	0.772

```
### 중복 행 삭제
final_dat33 = final_dat3.drop([247, 248, 249, 250, 251])
final_dat33[final_dat33.duplicated(['행정동명'])==True]  # 중복 행 삭제 확인
```

	자치구	행정동명	평일 총생활인구수	주말 총생활인구수	청년인구 수	1인가구	주택보급률(%)
245	강남구	신사동	36706.853724	36665.332604	9712.5	7282.0	0.772
246	관악구	신사동	36706.853724	36665.332604	9712.5	2047.0	0.337

```
### '자치구' 컬럼과 병합
gu = final_dat3.loc[:, ['자치구', '행정동명']]
gu = pd.DataFrame(gu)
gu
final_dat5 = pd.merge(gu, final_dat33, how = 'left', on = '행정동명')
final_dat5 = final_dat5.groupby('행정동명').mean()
final_dat5
```

행정동명	평일 총생활인구수	주말 총생활인구수	청년인구 수	1인가구	주택보급률(%)
가락1동	26125.190719	26120.413194	8344.0	1255.0	1.019
가락2동	25993.547328	25947.167555	9153.0	2272.0	0.908
가락본동	41818.437201	41624.432934	8682.0	3173.0	0.795
가리봉동	11824.105384	11764.873715	2791.0	2849.0	0.521
가산동	81435.798440	80162.635242	12228.5	9775.0	0.439
...	...	...	...	...	...
효창동	9989.714368	9980.813432	3374.5	1082.0	0.883
후암동	12678.979160	12671.304860	5238.0	2800.0	0.736
휘경1동	17092.518970	17118.852358	5597.0	3336.0	0.651
휘경2동	18469.509247	18424.607232	8350.5	4785.0	0.623
흑석동	31487.727133	31408.086541	9361.0	4562.0	0.717

423 rows × 5 columns

```
### '역 개수', '병원 개수' 컬럼과 병합
final_dat5 = pd.merge(final_dat33, dataframe_n, how = 'left', on='행정동명')  # 행정동명을 기준으로 병합 => 역 개수
final_dat6 = pd.merge(final_dat5, alarm_1, how = 'left', on = '행정동명')  # 행정동명을 기준으로 병합 => 병원 개수
final_dat6
```

```
# final_dat6[final_dat6['행정동명'].str.contains('*')]  # 특정 단어를 포함하는 인덱스 찾기
```

```
final_dat6 = final_dat6.drop(147, axis = 0)
final_dat6 = final_dat6.drop(355, axis = 0)
final_dat6 = final_dat6.drop(358, axis = 0)
```

```
final_dat6.loc[170, '역 개수'] = 6
final_dat6.loc[171, '역 개수'] = 6
final_dat6.loc[172, '역 개수'] = 6
final_dat6.loc[173, '역 개수'] = 6
final_dat6.loc[174, '역 개수'] = 6
final_dat6.loc[175, '역 개수'] = 6
final_dat6.loc[176, '역 개수'] = 6
final_dat6.loc[177, '역 개수'] = 6
final_dat6.loc[164, '역 개수'] = 6
final_dat6.loc[165, '역 개수'] = 6
final_dat6.loc[236, '역 개수'] = 6
final_dat6.loc[0, '역 개수'] = 5
final_dat6.loc[1, '역 개수'] = 5
final_dat6.loc[2, '역 개수'] = 5
final_dat6.loc[135, '역 개수'] = 5
final_dat6.loc[136, '역 개수'] = 5
final_dat6.loc[156, '역 개수'] = 5
final_dat6.loc[157, '역 개수'] = 5
final_dat6.loc[158, '역 개수'] = 5
final_dat6.loc[159, '역 개수'] = 5
final_dat6.loc[160, '역 개수'] = 5
final_dat6.loc[191, '역 개수'] = 5
final_dat6.loc[192, '역 개수'] = 5
final_dat6.loc[193, '역 개수'] = 5
final_dat6.loc[194, '역 개수'] = 5
final_dat6.loc[25, '역 개수'] = 4
final_dat6.loc[26, '역 개수'] = 4
final_dat6.loc[242, '역 개수'] = 4
final_dat6.loc[243, '역 개수'] = 4
final_dat6.loc[334, '역 개수'] = 4
final_dat6.loc[335, '역 개수'] = 4
final_dat6.loc[332, '역 개수'] = 4
final_dat6.loc[333, '역 개수'] = 4
final_dat6.loc[336, '역 개수'] = 4
final_dat6.loc[337, '역 개수'] = 4
final_dat6.loc[400, '역 개수'] = 4
final_dat6.loc[401, '역 개수'] = 4
final_dat6.loc[31, '역 개수'] = 3
final_dat6.loc[32, '역 개수'] = 3
final_dat6.loc[33, '역 개수'] = 3
final_dat6.loc[34, '역 개수'] = 3
final_dat6.loc[35, '역 개수'] = 3
final_dat6.loc[58, '역 개수'] = 3
final_dat6.loc[59, '역 개수'] = 3
final_dat6.loc[90, '역 개수'] = 3
final_dat6.loc[91, '역 개수'] = 3
final_dat6.loc[102, '역 개수'] = 3
final_dat6.loc[103, '역 개수'] = 3
final_dat6.loc[104, '역 개수'] = 3
final_dat6.loc[105, '역 개수'] = 3
final_dat6.loc[106, '역 개수'] = 3
final_dat6.loc[107, '역 개수'] = 3
final_dat6.loc[130, '역 개수'] = 3
final_dat6.loc[131, '역 개수'] = 3
final_dat6.loc[132, '역 개수'] = 3
final_dat6.loc[133, '역 개수'] = 3
final_dat6.loc[134, '역 개수'] = 3
final_dat6.loc[140, '역 개수'] = 3
final_dat6.loc[141, '역 개수'] = 3
final_dat6.loc[142, '역 개수'] = 3
final_dat6.loc[324, '역 개수'] = 3
final_dat6.loc[366, '역 개수'] = 3
final_dat6.loc[310, '역 개수'] = 3
final_dat6.loc[47, '역 개수'] = 3
final_dat6.loc[402, '역 개수'] = 3
final_dat6.loc[386, '역 개수'] = 3
final_dat6.loc[387, '역 개수'] = 3
final_dat6.loc[207, '역 개수'] = 3
final_dat6.loc[256, '역 개수'] = 3
final_dat6.loc[257, '역 개수'] = 3
final_dat6.loc[258, '역 개수'] = 3
final_dat6.loc[259, '역 개수'] = 3
final_dat6.loc[260, '역 개수'] = 3
```

```
final_dat6.loc[261, '역 개수'] = 3
final_dat6.loc[380, '역 개수'] = 3
final_dat6.loc[381, '역 개수'] = 3
final_dat6.loc[382, '역 개수'] = 3
final_dat6.loc[409, '역 개수'] = 3
final_dat6.loc[410, '역 개수'] = 3
final_dat6.loc[411, '역 개수'] = 3
final_dat6.loc[412, '역 개수'] = 3
final_dat6.loc[413, '역 개수'] = 3
final_dat6.loc[414, '역 개수'] = 3
final_dat6.loc[415, '역 개수'] = 3
```

```
### 결측치 중앙값으로 대체
final_dat6.loc[final_dat6['청년인구 수'] != final_dat6['청년인구 수'], '청년인구 수'] = final_dat6['청년인구 수'].median()
final_dat6.loc[final_dat6['1인가구'] != final_dat6['1인가구'], '1인가구'] = final_dat6['1인가구'].median()
final_dat6.loc[final_dat6['주택보급률(%)'] != final_dat6['주택보급률(%)'], '주택보급률(%)'] = final_dat6['주택보급률(%)'].median()
final_dat6.loc[final_dat6['역 개수'] != final_dat6['역 개수'], '역 개수'] = final_dat6['역 개수'].median()
final_dat6.loc[final_dat6['병원 개수'] != final_dat6['병원 개수'], '병원 개수'] = final_dat6['병원 개수'].median()
```

```
# final_dat6[final_dat6[''].isnull()] # 결측치 없음 확인
```

행정동명	주차장 수	자치구	평일 총생활인구수	주말 총생활인구수	청년인구 수	1인가구	주택보급률(%)	역 개수	병원 개수

```
final_dat6
```

	자치구	행정동명	평일 총생활인구수	주말 총생활인구수	청년인구 수	1인가구	주택보급률(%)	역 개수	병원 개수
0	송파구	가락1동	26125.190719	26120.413194	8344.0	1255.0	1.019	5.0	36.0
1	송파구	가락2동	25993.547328	25947.167555	9153.0	2272.0	0.908	5.0	36.0
2	송파구	가락본동	41818.437201	41624.432934	8682.0	3173.0	0.795	5.0	36.0
3	구로구	가리봉동	11824.105384	11764.873715	2791.0	2849.0	0.521	3.0	13.0
4	금천구	가산동	81435.798440	80162.635242	12228.5	9775.0	0.439	1.0	76.0
...	...	...	...	...	...	...	...	...	...
420	용산구	효창동	9989.714368	9980.813432	3374.5	1082.0	0.883	1.0	36.0
421	용산구	후암동	12678.979160	12671.304860	5238.0	2800.0	0.736	3.0	14.0
422	동대문구	휘경1동	17092.518970	17118.852358	5597.0	3336.0	0.651	3.0	36.0
423	동대문구	휘경2동	18469.509247	18424.607232	8350.5	4785.0	0.623	3.0	36.0
424	동작구	흑석동	31487.727133	31408.086541	9361.0	4562.0	0.717	3.0	49.0

422 rows × 9 columns

3.1. 데이터 분석

3.1.1. 최종 분석 데이터 정규화

3.1.1.1. 최종 분석 데이터인 'final_dat6'에서 정규화 할 속성들을 추출해서 'scaler_mm' 데이터 프레임 생성.

- **MinMax Scaler(정규화)**

MinMax(Normalization)

$$X = \frac{x - x_{min}}{x_{max} - x_{min}}$$

데이터들의 비율을 그대로 유지하는 상태에서 범위를 [0, 1]로 축소함.
데이터의 전체적인 분포는 변화지 않음.

```
import sklearn
from sklearn.preprocessing import *

scaler_mm = MinMaxScaler().fit_transform(final_dat6.loc[:, ['평일 총생활인구수', '주말 총생활인구수', '청년인구 수', '1인가구',
                                                        '주택보급률(%)', '역 개수', '병원 개수']])
scaler_mm = pd.DataFrame(scaler_mm)
scaler_mm.columns = ['평일 총생활인구수', '주말 총생활인구수', '청년인구 수', '1인가구', '주택보급률(%)', '역 개수', '병원 개수']  # 열 이름 재설정
scaler_mm.columns = ['ST_WEEK', 'ST_HOLI', 'ST_YOUT', 'ST_SOLO', 'ST_HOUS', 'ST_SUBW', 'ST_HOSP']  # 열 이름 변경
scaler_mm
```

	ST_WEEK	ST_HOLI	ST_YOUT	ST_SOLO	ST_HOUS	ST_SUBW	ST_HOSP
0	0.172275	0.174004	0.420536	0.093008	0.891278	0.8	0.038981
1	0.171255	0.172647	0.461346	0.168739	0.758662	0.8	0.038981
2	0.293857	0.295411	0.437587	0.235833	0.623656	0.8	0.038981
3	0.061478	0.061591	0.140414	0.211706	0.296296	0.4	0.004498
4	0.600790	0.597190	0.616491	0.727456	0.198327	0.0	0.098951
...	...	...	...	...	...	...	...
417	0.047266	0.047620	0.169849	0.080125	0.728793	0.0	0.038981
418	0.068101	0.068689	0.263854	0.208057	0.553166	0.4	0.005997
419	0.102295	0.103516	0.281963	0.247971	0.451613	0.4	0.038981
420	0.112963	0.113741	0.420864	0.355872	0.418160	0.4	0.038981
421	0.213821	0.215410	0.471839	0.339266	0.530466	0.4	0.058471

422 rows × 7 columns

3.1.2. 최종 데이터 점수화

```
score = scaler_mm['ST_WEEK'] + scaler_mm['ST_HOLI'] + scaler_mm['ST_YOUT'] + scaler_mm['ST_SOLO'] + scaler_mm['ST_HOUS'] + scale
score = pd.DataFrame(score)
score.columns = ['SCORE']

hjd = final_dat6.drop(columns = ['평일 총생활인구수', '주말 총생활인구수', '청년인구 수', '1인가구', '주택보급률(%)', '역 개수',

score_final = pd.concat([scaler_mm, score], axis = 1)
score_final = pd.concat([hjd, score_final], axis = 1)
score_final = score_final.sort_values('SCORE', ascending = False)
score_final

# score_final.to_csv('/content/gdrive/My Drive/datasciencebasis/점수 데이터.csv', index = False, encoding = 'cp949')

##############결과###############
#################################
## 1순위: 영등포구 여의도동
## 2순위: 관악구 신사동
## 3순위: 영등포구 양평2동
## 4순위: 서초구 서초2동
## 5순위: 노원구 상계5동
#################################
#################################
```

3.1.2.1. 분석 결과

행정동별 역세권 청년주택 우선 입지 조건은

속성(평일 총생활인구수, 주말 총생활인구수, 청년인구 수, 1인가구, 주택보급률, 역 개수, 병원 개수)들의 합을 나타내는 'SCORE' 컬럼의 값이 클수록 우선 입지 선정 지역임.

	자치구	행정동명	ST_WEEK	ST_HOLI	ST_YOUT	ST_SOLO	ST_HOUS	ST_SUBW	ST_HOSP	SCORE
278	영등포구	여의동	1.000000	1.000000	0.869725	0.947874	0.377539	0.4	0.038981	4.634119
244	관악구	신사동	0.254255	0.256578	0.489570	0.541812	0.596177	0.6	1.000000	3.738393
277	영등포구	양평2동	0.915996	0.913411	0.438394	0.198526	0.695341	0.4	0.038981	3.600648
192	서초구	서초2동	0.607524	0.608200	0.539562	0.360786	0.602151	0.8	0.038981	3.557203
174	노원구	상계5동	0.344383	0.347878	0.469115	0.261970	0.923536	1.0	0.038981	3.385863
...	...	...	...	...	...	...	...	...	...	...
283	서대문구	연희동	0.058001	0.058234	0.220092	0.131730	0.439665	0.0	0.019490	0.927213
90	강동구	둔촌1동	0.005572	0.005454	0.000000	0.000000	0.133811	0.4	0.038981	0.583818
422	동대문구	휘경1동	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
423	동대문구	휘경2동	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
424	동작구	흑석동	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

425 rows × 10 columns

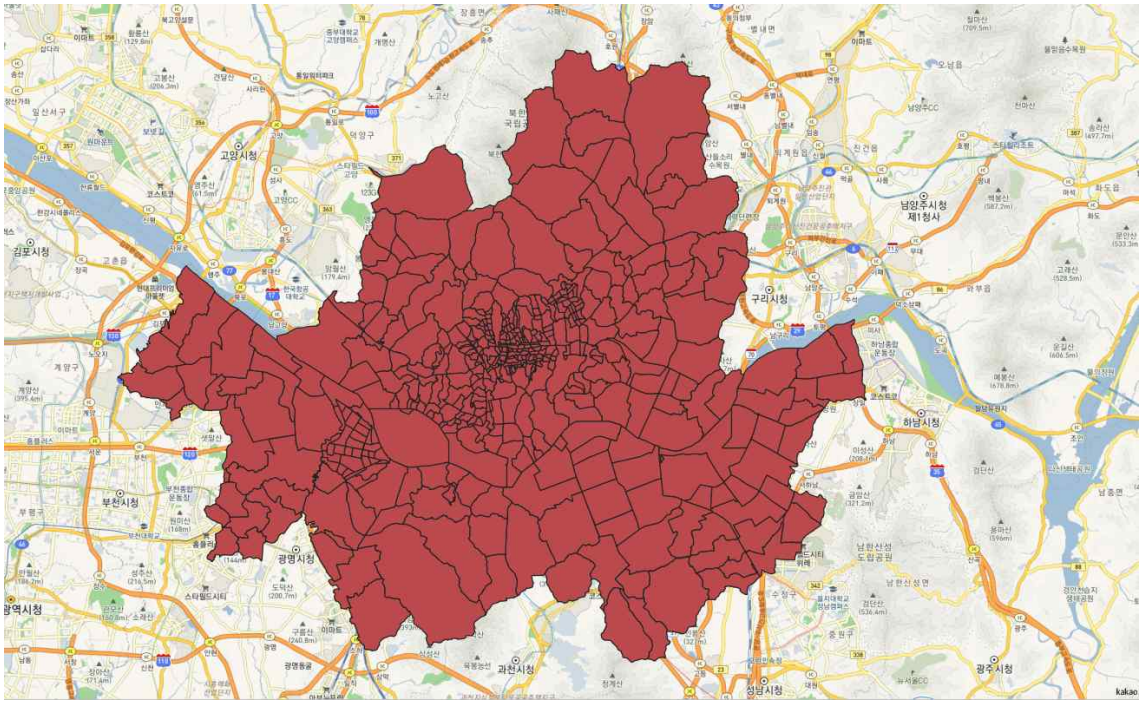
-결과

1. 영등포구 **여의동(여의도동) => 여의도동**
2. 관악구 **신사동(신림4동) => 신림동**
3. 영등포구 **양평2동 => 양평동3가**
4. 서초구 **서초2동 => 서초동**
5. 노원구 **상계5동 => 상계동**

4.1. 시각화

4.1.1. QGIS 활용 시각화

4.1.1.1. LSMD_ADM_SECT_UMD_11(서울시 행정동 구분 지도 데이터) 벡터 레이어 생성



4.1.1.2. 2019~2022년도 역세권 청년주택 공급현황 및 계획 데이터를 전처리한 후 excel 에 복사.

	A	B	C	D	E	F	G	H	I
1	No	자치구	동	번지	인근지하철	합계	공공	민간	도로명주소
2	1	광진구	구의동	587-64	강변역	84	18	66	구의강변로 53
3	2	서대문구	충정로 3기	480	충정로역	499	49	450	경기대로 26-26
4	3	성동구	용답동	233-1	장한평역	170	22	148	천호대로 416
5	4	마포구	서교동	395-43	합정역	1121	199	922	양화로 72
6	5	종로구	숭인동	207-32	동묘역	238	31	207	난계로29길 55
7	6	강서구	등촌동	648-5	등촌역	285	19	266	공항대로59가길 14
8	7	강서구	염창동	274-17	등촌역	520	49	471	공항대로 543
9	8	동작구	노량진동	37-1외7	노량진역	273	37	236	노량진로8길 55
10	9	동대문구	휘경동	192-1	회기역	99	8	91	망우로21나길 6
11	10	용산구	한강로2가	2-350일대	삼각지역	1226	421	805	백범로99길 40
12	11	서초구	서초동	1502-12외	서초역	280	68	212	반포대로27길 13
13	12	강서구	화곡동	401-1	화곡역	57	9	48	강서로 145
14	13	마포구	창전동	19-8 외1	광흥창역	681	120	561	서강로 75-16
15	14	마포구	상수동	355-2	상수역	95	27	68	토정로 131
16	15	영등포구	도림동	250-20	도림사거리	99	18	81	도신로 59
17	16	강서구	화곡동	1013-3외4	우장산역	572	87	485	강서로 231
18	17	강동구	천호동	458-3외7	천호역	225	50	175	천호대로 989
19	18	강서구	등촌동	671-1	발산역	252	53	199	강서로56길 44
20	19	중랑구	상봉동	109-34	상봉역	83	6	77	봉우재로 111
21	20	강서구	화곡동	1073-11	화곡역	83	6	77	화곡로 146

- 구별 공급실수 순위 -

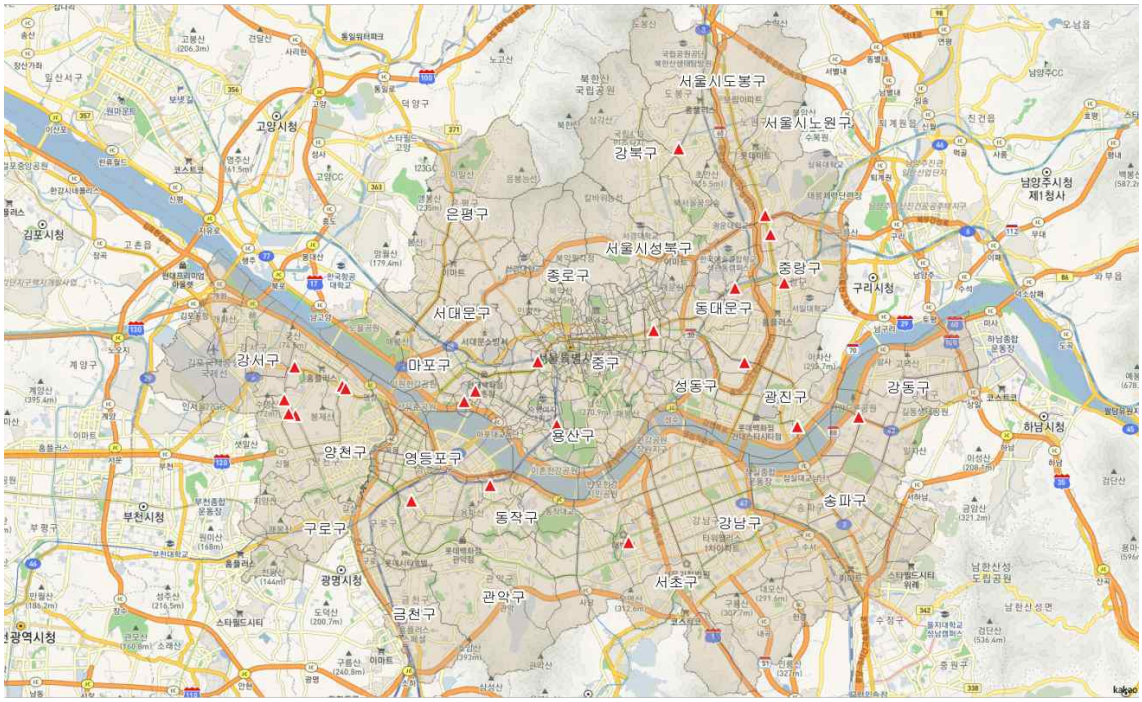
- 역세권청년주택 연도별 누적공급실수 -

4.1.1.3. Geocoder-Xr 프로그램을 사용하여 도로명주소를 이용하여 위·경도 컬럼을 추가

No	No	자치구	동	번지	인근지...	합계	공공	민간	도로명...	경도	위도	상태
1	1	광진구	구의동	587-64	강변역	84	18	66	구의강...	127.09...	37.535...	정좌표
2	2	서대문구	충정로...	480	충정로역	499	49	450	경기대...	126.96...	37.561...	정좌표
3	3	성동구	용답동	233-1	장한평역	170	22	148	천호대...	127.06...	37.560...	정좌표
4	4	마포구	서교동	395-43	합정역	1121	199	922	양화로...	127.55...	37.229...	정좌표
5	5	종로구	숭인동	207-32	동묘역	238	31	207	난계로...	127.02...	37.573...	정좌표
6	6	강서구	등촌동	648-5	등촌역	285	19	266	공항대...	126.86...	37.551...	정좌표
7	7	강서구	염창동	274-17	등촌역	520	49	471	공항대...	126.86...	37.550...	정좌표
8	8	동작구	노량진동	37-1외7	노량진역	273	37	236	노량진...	126.93...	37.512...	정좌표
9	9	동대문구	휘경동	192-1	회기역	99	8	91	망우로...	127.06...	37.590...	정좌표
10	10	용산구	한강로...	2-350...	삼각지역	1226	421	805	백범로...	126.97...	37.536...	정좌표
11	11	서초구	서초동	1502-...	서초역	280	68	212	반포대...	127.00...	37.489...	정좌표
12	12	강서구	화곡동	401-1	화곡역	57	9	48	강서로...	126.84...	37.539...	정좌표
13	13	마포구	창전동	19-8 ...	광흥창역	681	120	561	서강로...	126.93...	37.549...	정좌표
14	14	마포구	상수동	355-2	상수역	95	27	68	토정로...	126.92...	37.545...	정좌표
15	15	영등포구	도림동	250-20	도림사...	99	18	81	도신로...	126.89...	37.506...	정좌표
16	16	강서구	화곡동	1013-...	우장산역	572	87	485	강서로...	126.83...	37.546...	정좌표
17	17	강동구	천호동	458-3...	천호역	225	50	175	천호대...	127.12...	37.539...	정좌표
18	18	강서구	등촌동	671-1	발산역	252	53	199	강서로...	126.84...	37.559...	정좌표
19	19	중랑구	상봉동	109-34	상봉역	83	6	77	봉우재...	127.08...	37.592...	정좌표
20	20	강서구	화곡동	1073-11	화곡역	83	6	77	화곡로...	126.83...	37.540...	정좌표
21	21	중랑구	묵동	176-3...	먹골역	235	24	211	공릉로...	127.07...	37.611...	정좌표
22	22	광진구	구의동	593-11	강변역	98	28	70	구의강...	127.09...	37.535...	정좌표
23	23	노원구	공릉동	617-3...	태릉입...	270	75	195	동일로...	127.07...	37.619...	정좌표
24	24	도봉구	쌍문동	103-6	쌍문역	288	70	218	도봉로...	127.03...	37.645...	정좌표

4.1.1.4. QGIS을 이용해 kakao Map에 역세권청년주택 위치 현황을 나타냄.

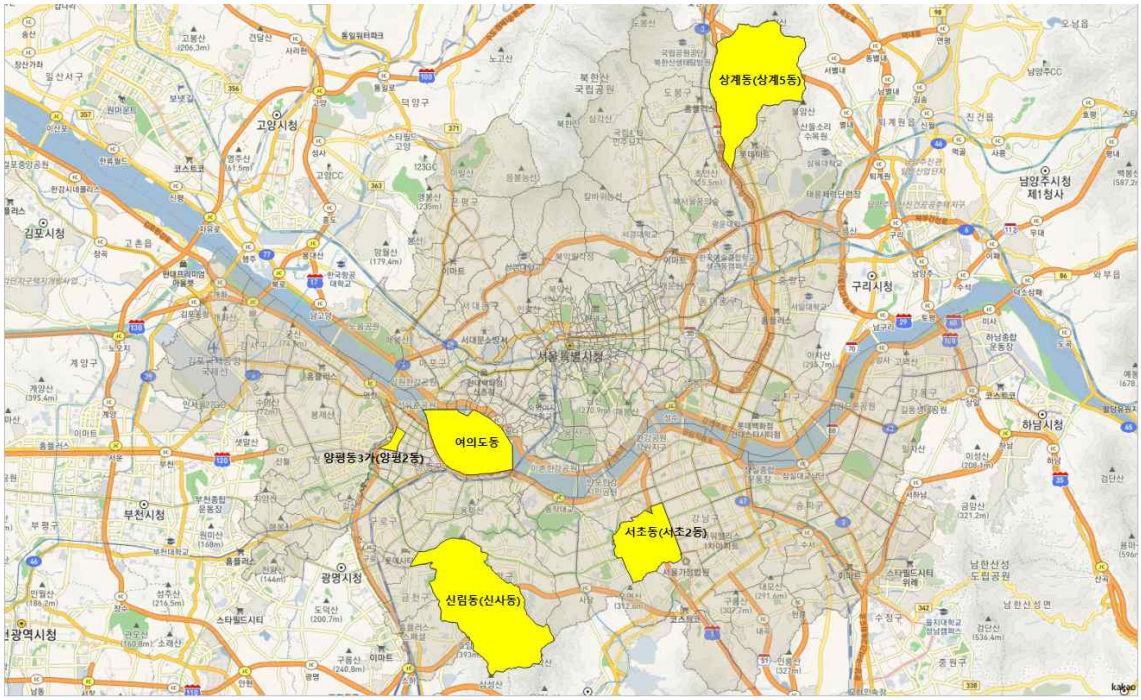
- 이 심볼은 역세권청년주택 위치 현황을 의미.



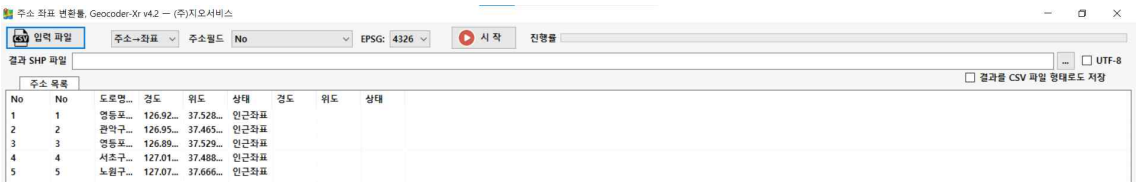
4.1.1.5. 분석 결과를 시각화하기 위해 QGIS에서 표현식을 이용, 지도 시각화

```
"EMD_NM" = '여의도동' or
"EMD_NM" = '양평동3가' or
"EMD_NM" = '상계동' or
"EMD_NM" = '서초동' or
"EMD_NM" = '신림동'
```

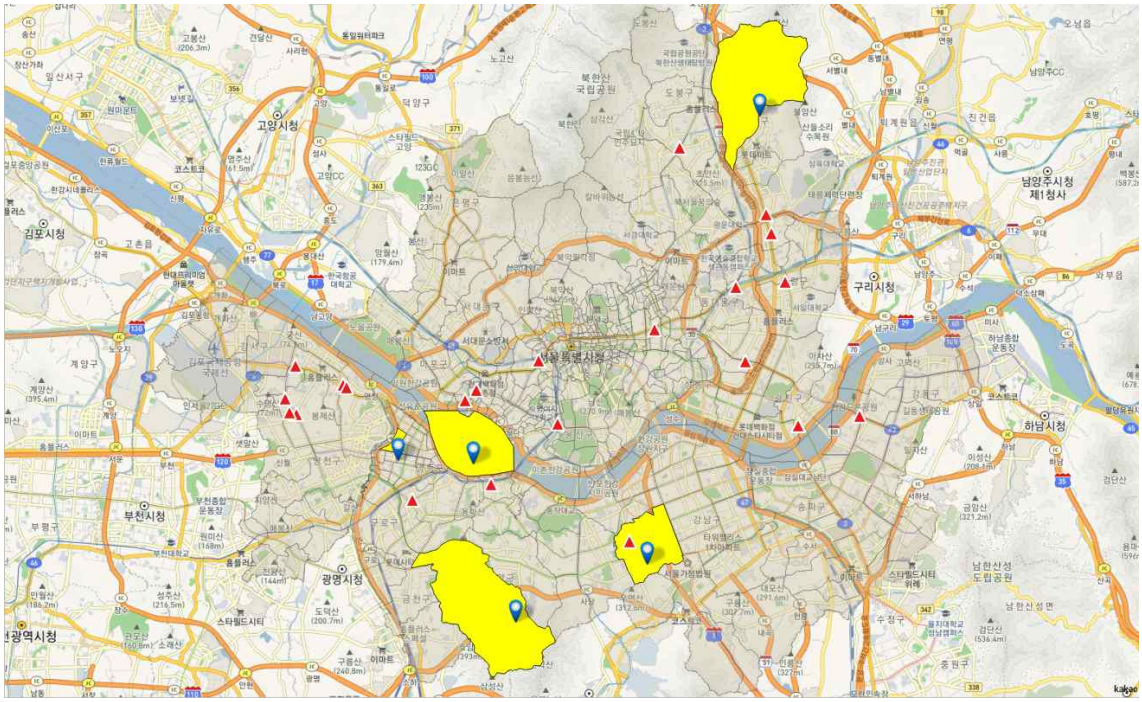
기존에 있던 서초동을 제외한
5th[상계5동(상계동)], 3rd[양평2동(양평동3가)], 1st[여의도동]은 새로운 입지 선정 대상임.
4th[신사동(신림4동)]은 2022년 입주 예정이므로
새로 입지선정한 곳은 여의도동, 상계5동, 양평동3가임.



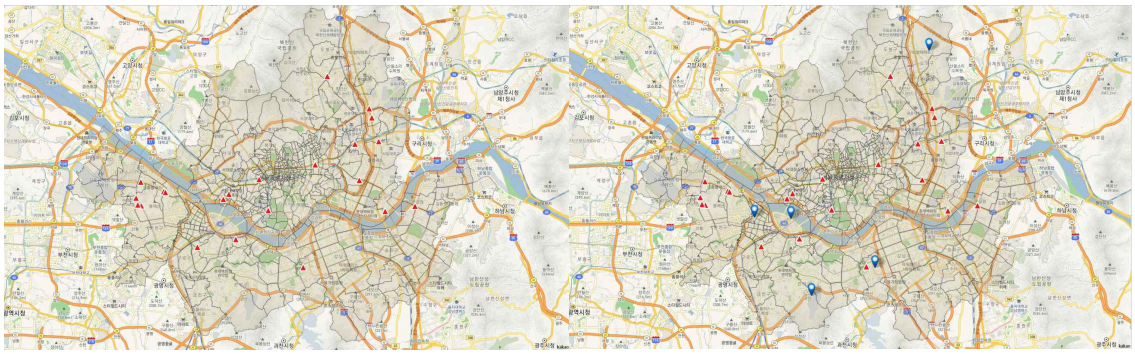
4.1.1.6 분석 결과 데이터의 위치 데이터는 Geocode-Xr을 활용하여 위·경도 컬럼을 생성하여 시각화함.



- 이 심볼은 분석을 통한 역세권청년주택 최적 입지 선정(1~5순위) 위치를 의미.



4.1.1.7. 분석 전 역세권 청년주택 위치와 분석 후 위치 비교



4.1.1.8. 역세권 청년주택 최적 입지 선정 5순위 다중 거리 버퍼 설정

DBPia에서 '연령에 따른 보행속도 및 보폭에 대한 고찰'에 대한 논문 자료를 참고하여 20~30대 남성의 평균속도(M)와 여성의 평균속도(F)의 평균(MF)으로 10분 걸을 수 있는 거리를 버퍼 고정 거리로 지정함. 실험거리는 28m 로 진행됨.

<표 2> 연령별 보행속도의 평균

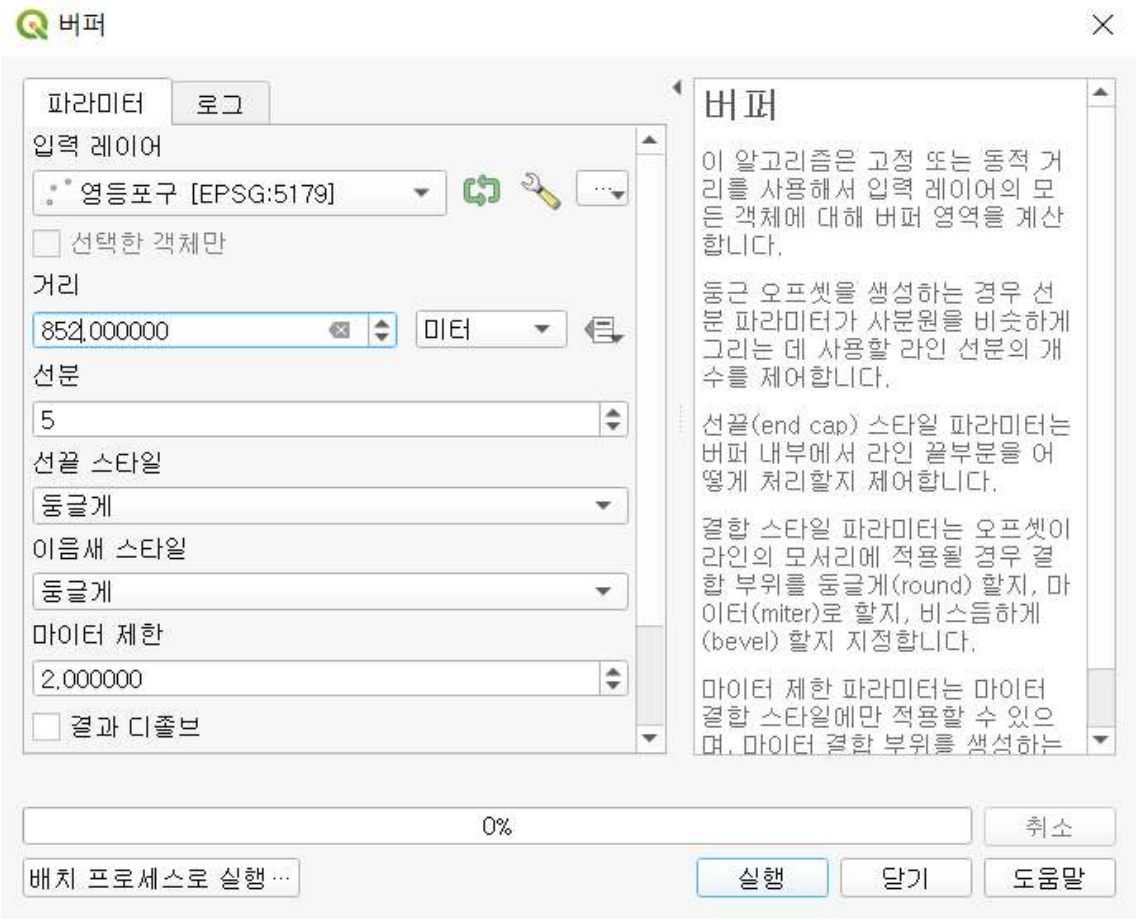
연령	10 대		2~30 대	
성별	남	여	남	여
속도(m/s)	1.3	1.4	1.49	1.35

평균속도(m/s) * 600초 = 1.42(m/s) * 600 = 852(m)

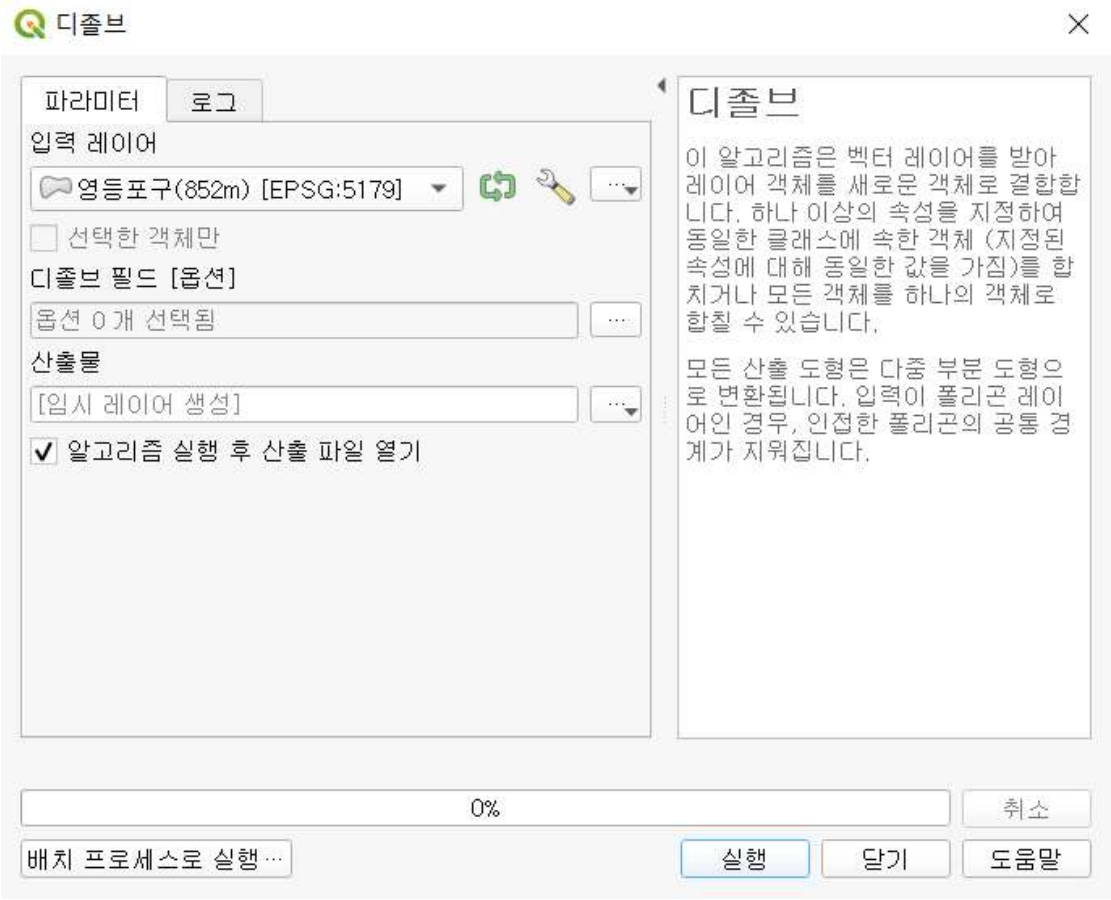
* 서울교통공사_역주소 및 전화번호_20220314.csv
도로명주소를 Geocoder-Xr 프로그램을 통해 X, Y 좌표를 생성.

	A	B	C	D	E	F	G	H
1	No	호선	역명	도로명주소	지번주소	X	Y	상태
2	1	1	서울	서울특별시	서울특별시	126.9726	37.55716	정좌표
3	2	1	시청	서울특별시	서울특별시	126.977	37.56544	정좌표
4	3	1	종각	서울특별시	서울특별시	126.9832	37.57021	정좌표
5	4	1	종로3가	서울특별시	서울특별시	126.992	37.57043	정좌표
6	5	1	종로5가	서울특별시	서울특별시	127.0019	37.57091	정좌표
7	6	1	동대문	서울특별시	서울특별시	127.0112	37.57178	정좌표
8	7	1	신설동	서울특별시	서울특별시	127.0247	37.57532	정좌표
9	8	1	제기동	서울특별시	서울특별시	127.0347	37.5782	정좌표
10	9	1	청량리(서	서울특별시	서울특별시	127.0451	37.58023	정좌표
11	10	1	동묘앞	서울특별시	서울특별시	127.0168	37.57337	정좌표
12	11	2	시청	서울특별시	서울특별시	126.9754	37.56358	정좌표
13	12	2	을지로입구	서울특별시	서울특별시	126.9824	37.56605	정좌표
14	13	2	을지로3가	서울특별시	서울특별시	126.9903	37.56627	정좌표
15	14	2	을지로4가	서울특별시	서울특별시	126.9979	37.56664	정좌표
16	15	2	동대문역사	서울특별시	서울특별시	127.0091	37.56559	정좌표
17	16	2	신당	서울특별시	서울특별시	127.0195	37.56565	정좌표
18	17	2	상왕십리	서울특별시	서울특별시	127.029	37.56447	정좌표
19	18	2	왕십리(성	서울특별시	서울특별시	127.0367	37.56122	정좌표
20	19	2	한양대	서울특별시	서울특별시	127.0436	37.55558	정좌표

서울시 내 각 지하철역의 좌표 중 최적 입지 선정된 5순위의 자치구 안에 있는 지하철역의 좌표를 선정해서 포인트 레이어로 지정 후, 다중 고리 버퍼를 형성함. 버퍼 고정 거리는 852m 로 설정.



공간 처리 툴박스에서 ‘dissolve’ 기능을 사용하여 버퍼 중첩을 방지함.

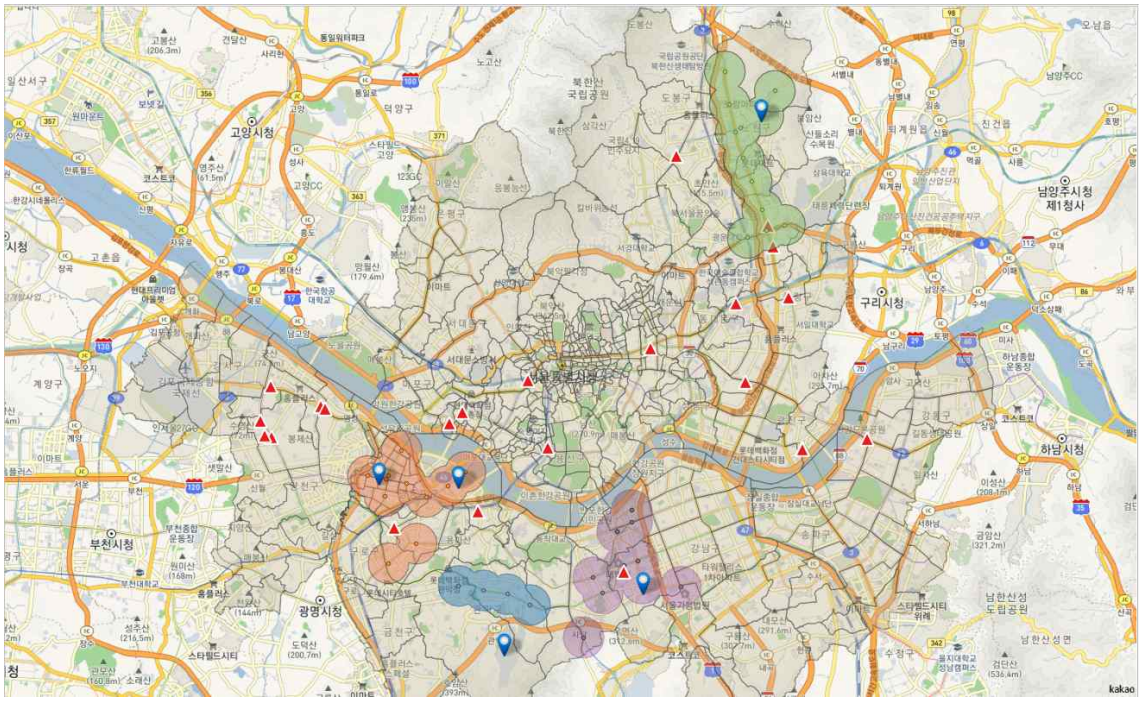


5. 결과

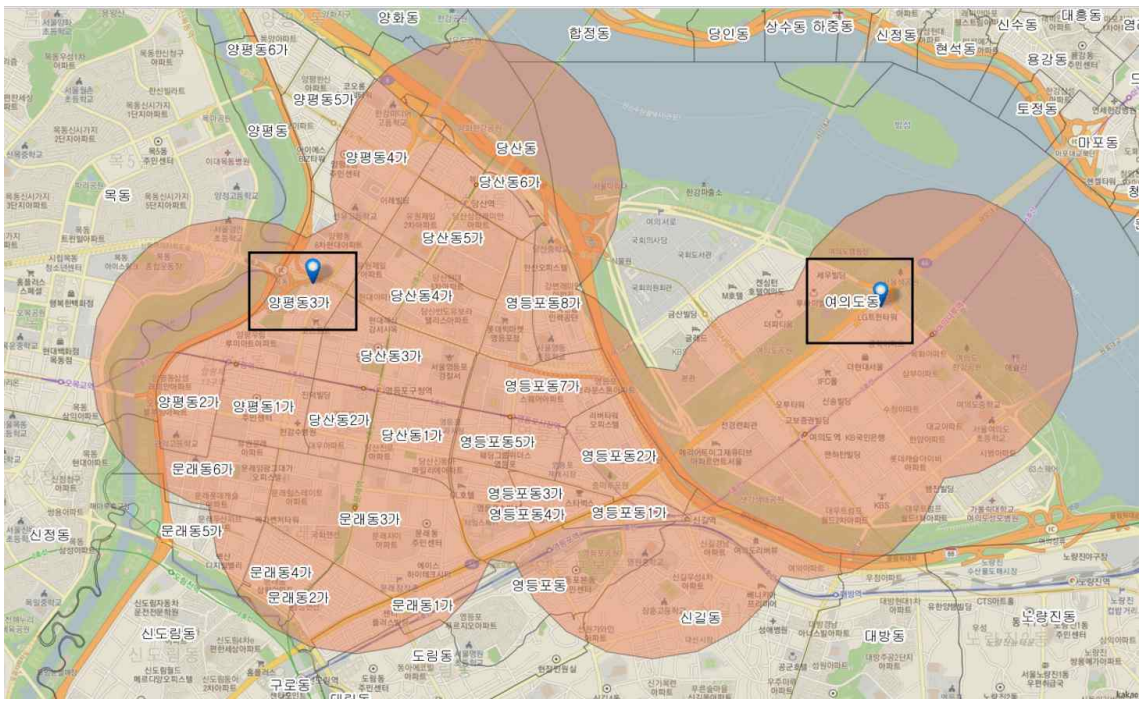
5.1. 분석 결과

5.1.1. 버퍼 생성 결과

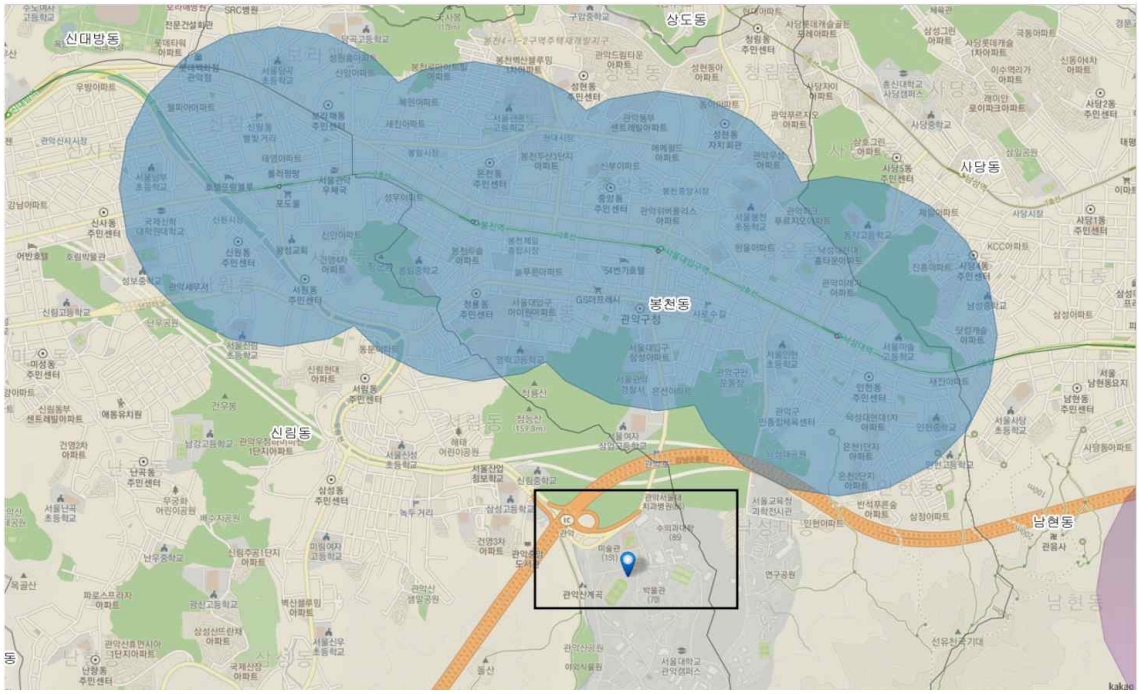
- 최종 역세권청년주택 최적 입지 선정



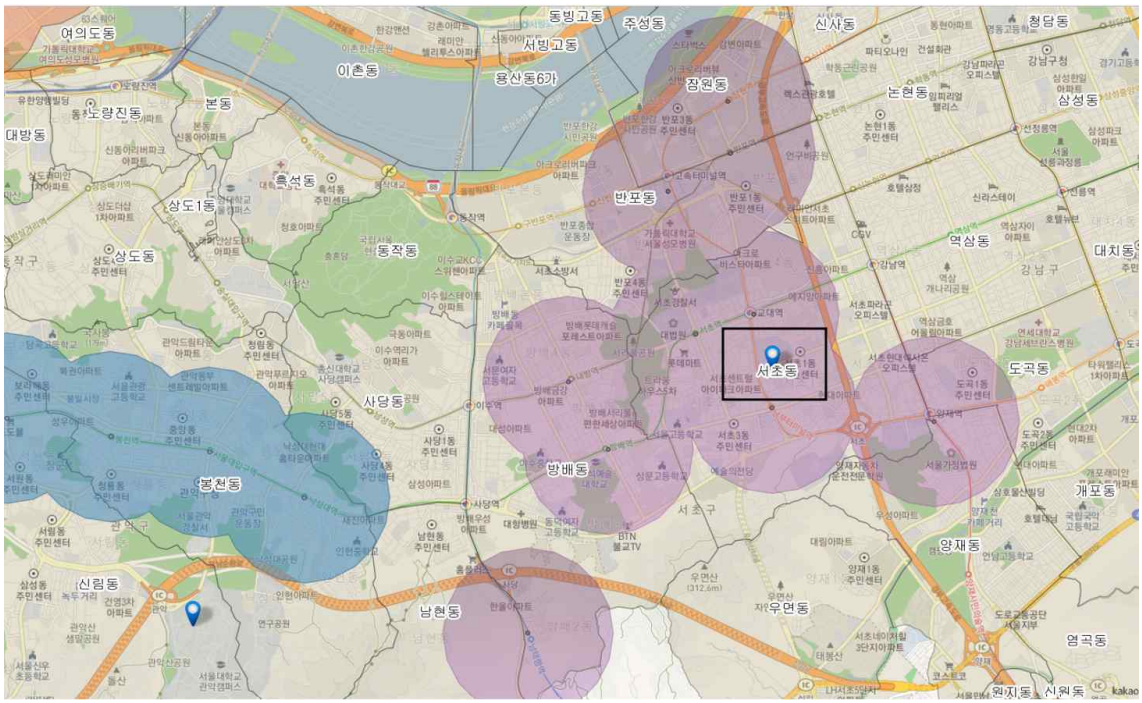
- 영등포구 여의도동(1순위), 영등포구 양평2동(양평동3가, 3순위)



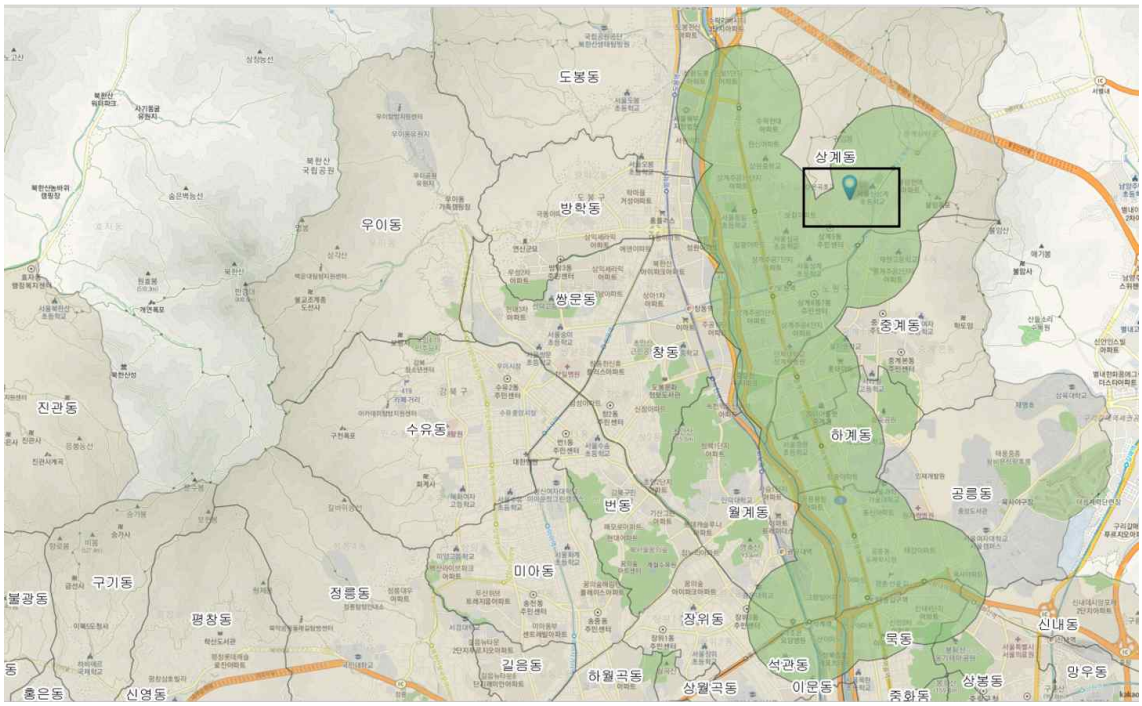
- 관악구 신림4동(2순위)



- 서초구 서초2동(4순위)



- 노원구 상계5동(5순위)



6. 결론 및 제언

6.1. 결론

역세권청년주택 최적 입지 선정 행정동은 1순위 영등포구 여의도동, 2순위 관악구 신사동(신림4동), 3순위 영등포구 양평2동(양평동3가), 4순위 서초구 서초2동, 5위 노원구 상계5동이라는 결론 도출.

6.1.1. 기대효과

1. 업무지구 중 하나이면서 중소·소기업이 많은 여의도에 역세권청년주택을 입지하면 청년의 경제활동의 접근성의 불편을 보완할 수 있음.
2. 1인가구 수의 증가가 가속화되고 있는 만큼 일시적일지라도 청년의 주거문제를 해결해줄 수 있음. 즉, 청약 경쟁률을 완화하고, 청년의 자금마련에도 실질적인 도움을 줄 수 있음.
3. 선정된 행정구역 내 생활인구 증가로 상권 활성화가 이루어질 수 있음.

6.2. 한계점

1. 데이터 수집 과정에서 역지오코딩 과정에서 어려움을 느껴 많은 변수(버스 정류장, 지가지수, 임대료)를 분석에 다 이용하지 못함.
2. 역세권에 조건 중 역의 승강장 또는 출구 경계선을 기준으로 350m의 공간을 시각화하지 못함.
3. 선정된 행정동의 위치는 정좌표가 아닌 인근좌표로, 지형적 특성이나 건립기준을 고려하지 못함.

출처

- 논문: 연령에 따른 보행속도 및 보폭에 대한 고찰
https://www.dbpia.co.kr/pdf/pdfView.do?nodeId=NODE01064047&mark=0&useDate=&ipRange=N&accessgl=Y&language=ko_KR&hasTopBanner=false

- 논문: 서울특별시 역세권 청년주택 건립 및 운영기준
https://news.seoul.go.kr

- 서울 열린데이터 광장

https://data.seoul.go.kr/

- 공공데이터포털
https://www.data.go.kr/

- 국가공간정보포털
http://www.nsdi.go.kr/lxportal/?menuno=2679

- 역세권청년주택
http://youth2030.co.kr