

[빅데이터전문가]
빅데이터분석 머신러닝 활용

ZenSla

A팀

강신우
이○○
윤○○
장○○
정○○

2022년 7월

목 차

1. 팀 소개
2. 프로젝트 개요
3. 데이터 처리 기획
4. 데이터 분석 과정
5. 시각화 (화면 구현)

Part 1, 팀 소개



팀 소개



강신우

팀장

- 프로젝트 관리 및 회의록 작성
- 충전소 데이터 수집
- 최종 입지 시각화용 웹 개발
- 발표 자료 작성



이○○

부 팀장

- 카카오 Open API 데이터 수집 및 저장
- 입지분석 데이터 처리 및 분석
- 입지분석 결과 시각화
- 결과 보고서 작성



윤○○

팀원

- 차량, 인구수 데이터 수집 및 저장
- 뉴스기사 크롤링
- 발표 자료 ppt 작성
- 2차 군집 알고리즘 탐색



장○○

팀원

- 형태소 분석 및 워드클라우드 시각화
- 전기차 데이터 전처리 및 분석
- 지역별 전기차 등록수 시각화



정○○

팀원

- DB설계
- 최종 입지 시각화용 웹 개발
- 전주시 인구수 데이터 전처리
- 전국 연료별 차량 데이터 수집 및 저장
- 웹 이벤트 작성 및 처리

Part 2, 프로젝트 개요

1. 개발 일정 관리

2. 개발 환경

3. 프로젝트 배경



프로젝트 일정

SUN	MON	TUE	WED	THU	FRI	SAT
6.5	6 현충일	7 프로젝트 시작 컨텐츠 회의	8 요구사항 명세서 작성 및 자료 수집 시작 보고서 작성 시작	9 기능적 요구사항 정의 및 유스케이스 작성 자료 수집	10 요구 사항 명세서 및 자료수집 완료 카카오 API 활용법 공부	11 개인 별 부족한 점 보안
12 2주차 목표 설정 수집한 데이터 전처리, 전기차 예측 분석, HTML 작업 완료	13 수집 데이터 전처리	14 HTML 레이아웃 작성 및 데이터 예측 분석 모형 작업	15 HTML 작업 및 발표 자료 작성	16 HTML 작업 완료 및 발표 자료 마무리 1차 군집화 시작 Mclp,lscp 알고리즘 공부	17 학원 임시 휴일	18 개인 별 부족한 점 보안
19 3주차 목표 설정 1차 군집 분석 완성 2차 군집 분석을 위한 준 비	20 중간 발표 HTML 작업, 2차 군집 공부	21 다중 선형 회귀 분석 완 료	22 웹 페이지 작업 및 2차 군집	23 2차 군집을 위한 상관 분 석 JSP 지도, 막대 그래프 이 벤트 연동	24 1차 군집 완료 웹 페이지 구현 지도,막대그래프 이벤트 연동	25 개인 별 부족한 점 보안
26 4주차 목표 설정 2차 군집 완성, 웹 페이지 구현 완성 발표 자료 준비 완성	27 웹 페이지 지도와 막대 그래프 연동 완성 2차 군집 시작	28 웹 페이지 및 맵 카테고 리 구현 2차 군집	29 웹 페이지 구현 2차 군집 완료	30 웹 페이지 구현 최종 발표 자료 작성	7.1 웹 페이지 구현 마무리 최종 발표 자료 및 보고 서 마무리	2 웹 페이지 테스트 및 발표 준비
3 웹 페이지 테스트 및 발표 준비	4 최종 발표					

개발환경

Technic

Python 3.8.8
HTML 5
CSS 3
JAVA 1.8.0_212
Kakaomap API
Java script
jQuery 1.8.1
Tomcat 8.5
D3.js 3.1.7
BootStrap 5.1.3
Chart.js 2.6.0
Highchart.js 10.1.0

Tool

Eclipse 2019-12
Spyder 4.2.5
Jupyter Notebook 6.3.0
QGIS 3.24.3

Data Base

MySQL 8.0.19

프로젝트 관리

SVN(형상관리)
Notion(일정관리)
한글(회의록 등)
Excel(데이터 관리)
PowerPoint(발표자료)

ESG



Environment (환경)

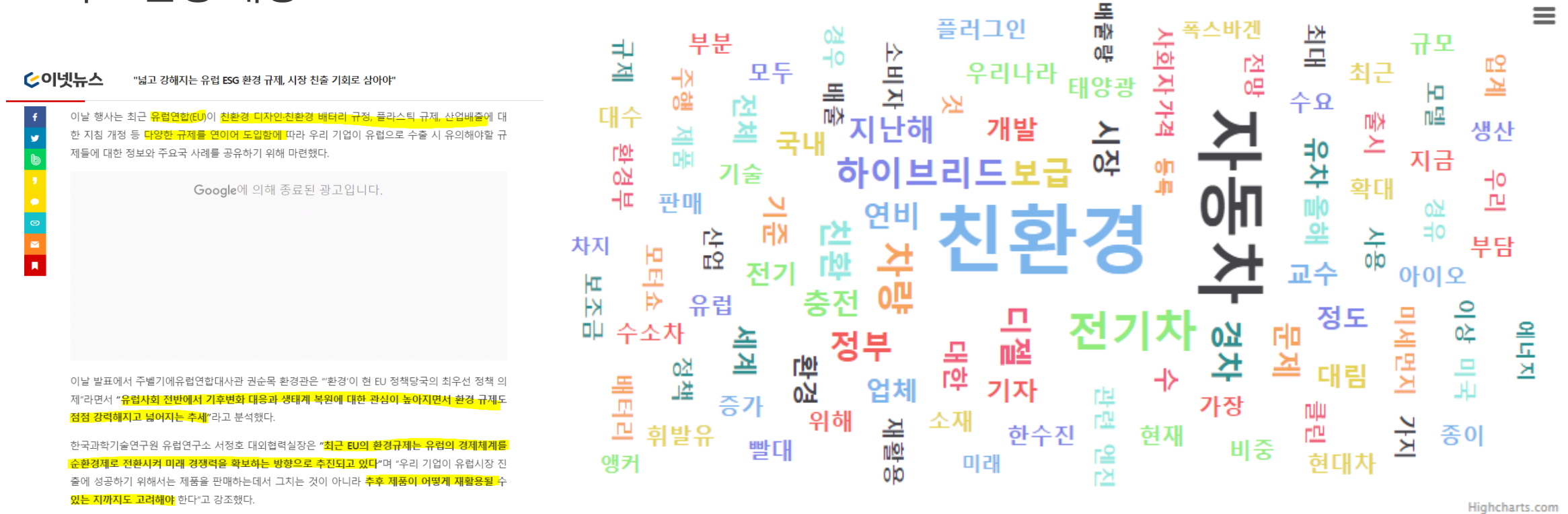
Social (사회적)

Governance (지배구조)

과거 기업은 1차원 목적(이윤추구 등)을 가지고 기업을 운영

현재 기업은 비재무적 요소 친환경, 지배구조까지 고려해 기준
을 만들

프로젝트 선정 배경



- 세계적으로 친환경 규제 강화
- 이러한 추세에 맞춰 우리나라 또한 친환경 정책을 수립하는 중

프로젝트 선정 배경

구글 전기차 관심도 변화 (2017.7 ~ 2022.6)

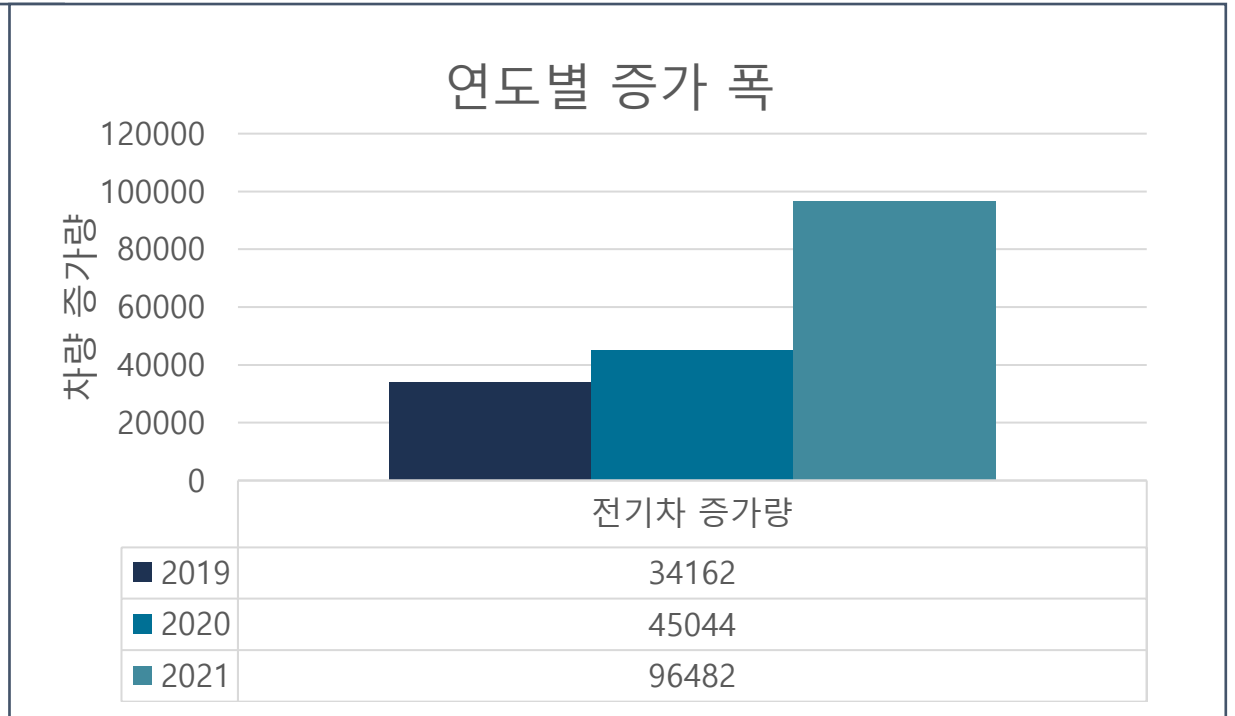
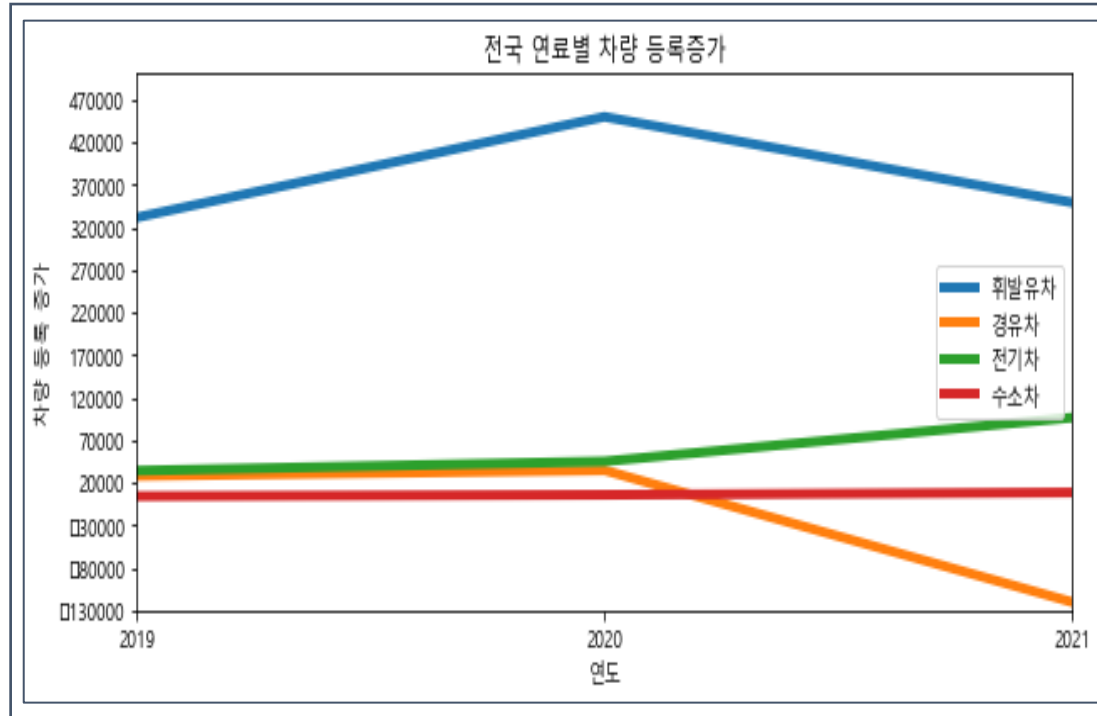


1	전기차 충전	100	<div></div>
2	전기차 보조금	91	<div></div>
3	전기차 가격	71	<div></div>
4	전기차 배터리	65	<div></div>
5	전기차 충전소	51	<div></div>

자료 : 구글 트렌드 분석

- 실제로 해당 정책으로 인해 전기차에 대한 관심이 많아졌다.

프로젝트 선정 배경



2019년 기준 휘발유차 약 1.05배 증가

2019년 기준 전기차 약 3배 증가

자료 : 국토 교통부

프로젝트 선정 배경

정부 전기차 충전 정책 만족도

단위: %



자료: 소프트베리

The JoongAng

정부 전기차 충전 정책 만족도, 그래픽 김은교 기자

시급한 개선이 필요한
전기차 충전 정책

단위: %

전기차 충전시설 보급 확대 40

미흡한 전기차 충전시설 관리 32

급속충전기 보급 확대 21

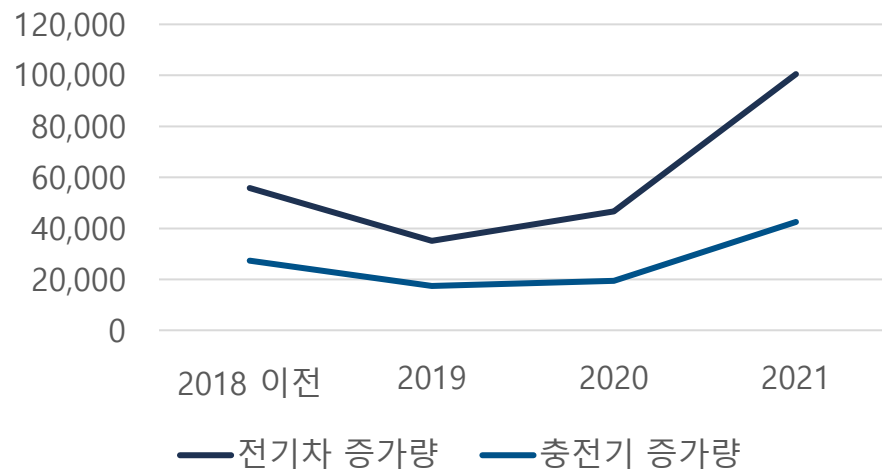
전기차 충전시설 정보의
실시간 확인 7

자료: 소프트베리

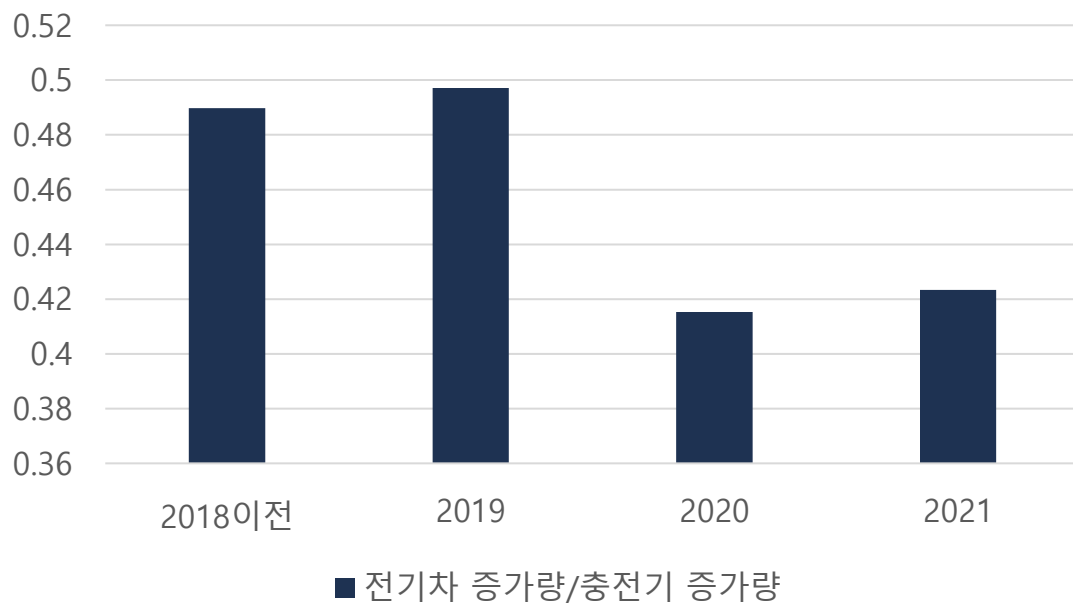
The JoongAng

● 정책으로 전기차 수요는 크게 늘었지만, 충전 시설의 확충 정책이 부족하여 문제가 생긴 걸 확인할 수 있음.

프로젝트 선정 배경

연도별 전기차, 충전기 증가
현황

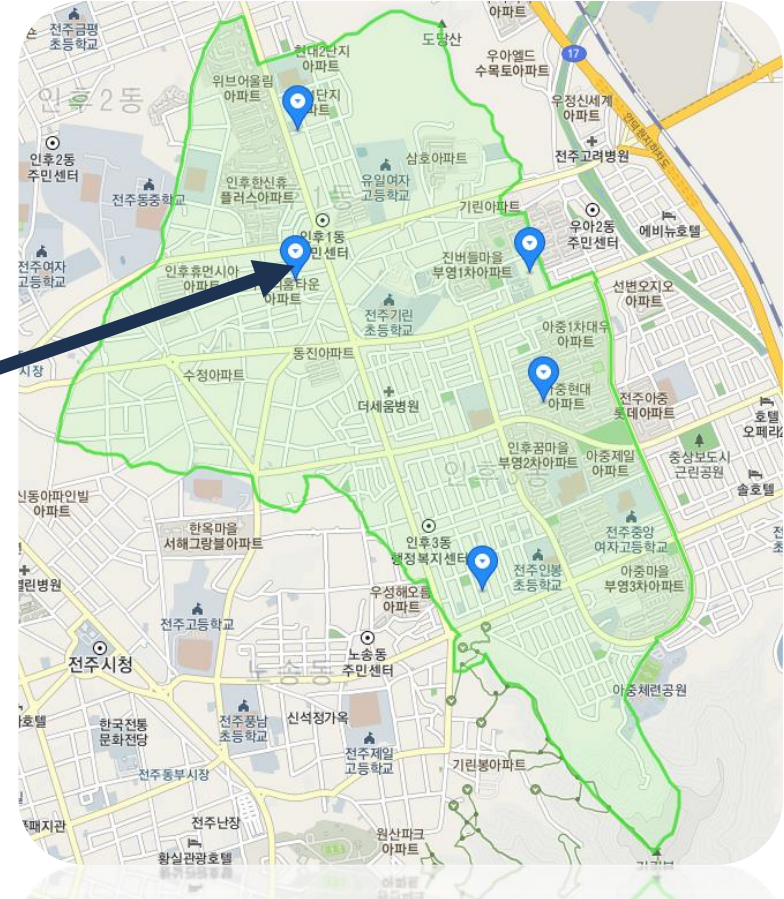
전기차 증가량 대비 충전기 증가량



Part 2, 프로젝트 개요

- 프로젝트 배경

프로젝트 목적



- 전주시 전기차 충전소 보급확대를 위한 입지 분석
- 입지 선정 알고리즘 구축, 표준화 모델 제시

Part 3,

데이터 처리 기획

1. 데이터 수집 계획

2. 데이터 처리 과정

3. 데이터 베이스 설계



데이터 수집

수집대상

국토 교통부, 공공 데이터 포털, 국가공간정보포털, 행정표준코드관리시스템,
카카오 맵 (법정동 별 건물 데이터) 등

수집 기간

2022.06.08 ~ 2022.06.15

수집 방법

크롤링, Open API, 공공 데이터 활용 및 기관 문의

수집 내용

뉴스기사 키워드 추출, 전기차 및 충전소 현황 등

데이터 처리 과정



DB설계도

1

장소				
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
장소 번호	pno	N/A	INTEGER	N-N
장소 이름	pname	N/A	VARCHAR(60)	NULL
장소 주소	padd	N/A	VARCHAR(60)	NULL
도로명 주소	pnewadd	N/A	VARCHAR(60)	NULL
동 이름	pdong	N/A	VARCHAR(30)	NULL
img url 주소	pimg	N/A	VARCHAR(256)	NULL
카테고리 대분류	pfcate	N/A	VARCHAR(30)	NULL
카테고리 소분류	plcate	N/A	VARCHAR(30)	NULL
카테고리1	pcated1	N/A	VARCHAR(30)	NULL
카테고리2	pcated2	N/A	VARCHAR(30)	NULL
카테고리3	pcated3	N/A	VARCHAR(30)	NULL
카테고리4	pcated4	N/A	VARCHAR(30)	NULL
카테고리5	pcated5	N/A	VARCHAR(30)	NULL
전화번호	ptel	N/A	VARCHAR(30)	NULL
위도	plat	N/A	VARCHAR(30)	NULL
경도	plon	N/A	VARCHAR(30)	NULL
utm_x	utmk_x	N/A	VARCHAR(30)	NULL
utm_y	utmk_y	N/A	VARCHAR(30)	NULL

2

크롤링				
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
기사번호	cr_no	N/A	INTEGER	N-N
기사제목	cr_word	N/A	VARCHAR(256)	NULL
기사주소	wordcount	N/A	INTEGER	NULL

3

차트				
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
지역번호	ano	N/A	BIGINT	N-N
시군구명	goo	N/A	VARCHAR(256)	NULL
읍면동명	dong	N/A	VARCHAR(256)	NULL
인구수	people	N/A	INTEGER	NULL
충전소 개수	charger	N/A	INTEGER	NULL
전기차 대수	car	N/A	INTEGER	NULL
주소	addr	N/A	VARCHAR(256)	NULL
전기차 예측값	pcar	N/A	INTEGER	NULL
급속 충전기	qcharger	N/A	INTEGER	NULL

4

최종결과				
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
장소 번호	bno	N/A	INTEGER	N-N
장소 이름	bname	N/A	VARCHAR(60)	NULL
장소 주소	badd	N/A	VARCHAR(60)	NULL
동 이름	bdong	N/A	VARCHAR(30)	NULL
카테고리 대분류	bfcate	N/A	VARCHAR(30)	NULL
위도	blat	N/A	VARCHAR(30)	NULL
경도	blon	N/A	VARCHAR(30)	NULL

1. Kakao Map API DB

2. 워드 클라우드를 위한 DB

3. 지역분석을 위한 DB

4. 입지 추천 장소에 관한 DB

Part 4,

데이터 분석 과정

1. 데이터 수집 및 저장

2. 데이터 처리

3. 데이터 분석



데이터 수집 및 저장

DATA
공공데이터포털
.GO .KR
데이터찾기
국가데이터맵
데이터요청
데이터방송
정보공유
이용안내

데이터 상세

한국전력공사_지역별 전기자 현황정보

지역별 운영통계는 전기자 현황정보를 제공합니다.
각 정보별 지역별 통계화되어있는 전기자 통계 데이터를 일괄 정보를 제공합니다.

0
 0
[관심]

파일데이터	오픈API	주선데이터
<p>공공데이터활용지원센터는 공공데이터포털에 개방되는 3단계 이상의 오픈 포맷 파일데이터를 오픈 API(RESTAPI 기반의 JSON/XML)로 자동변환하여 제공한다. 기존에는 공공데이터포털 조회 가능 도 활용신청이 필요하여 활용 관련 문서는 공공데이터활용지원센터로 연락주시기 바라며, 데이터 자체에 대한 문의는 아래 제공기관의 관리부서 전화번호로 연락주시기 바랍니다.</p> <p>파일데이터는 로그인 없이 다운로드를 통해 이용하실 수 있습니다.</p>		

CSV 한국전력공사_지역별 전기자 현황정보

데이터 상세

행정안전부_지역별(법정동) 성별 연령별 주민등록 인구수

법정동(읍면동) 성별 연령별 주민등록 인구에 대한 데이터입니다.
법정동은 시 또는 구의 하위 행정구역으로 법률로 지정한 구역에 말합니다.

0
 0
[관심]

파일데이터	오픈API	주선데이터
<p>공공데이터활용지원센터는 공공데이터포털에 개방되는 3단계 이상의 오픈 포맷 파일데이터를 오픈 API(RESTAPI 기반의 JSON/XML)로 자동변환하여 제공한다. 기존에는 공공데이터포털 조회 가능 도 활용신청이 필요하여 활용 관련 문서는 공공데이터활용지원센터로 연락주시기 바라며, 데이터 자체에 대한 문의는 아래 제공기관의 관리부서 전화번호로 연락주시기 바랍니다.</p> <p>파일데이터는 로그인 없이 다운로드를 통해 이용하실 수 있습니다.</p>		

CSV 한국전력공사_지역별 전기자 현황정보

데이터 상세

한국전력공사_지역별 전기자 현황정보

한국전력공사에서 운영하는 지역별 전기자 운영진소 현황 정보 입니다.
지역 및 연도(2016~2021)별 현황정보를 제공합니다.

0
 0
[관심]

파일데이터	오픈API	주선데이터
<p>공공데이터활용지원센터는 공공데이터포털에 개방되는 3단계 이상의 오픈 포맷 파일데이터를 오픈 API(RESTAPI 기반의 JSON/XML)로 자동변환하여 제공한다. 기존에는 공공데이터포털 조회 가능 도 활용신청이 필요하여 활용 관련 문서는 공공데이터활용지원센터로 연락주시기 바라며, 데이터 자체에 대한 문의는 아래 제공기관의 관리부서 전화번호로 연락주시기 바랍니다.</p> <p>파일데이터는 로그인 없이 다운로드를 통해 이용하실 수 있습니다.</p>		

CSV 한국전력공사_지역별 전기자 운영진소 현황정보

데이터 상세

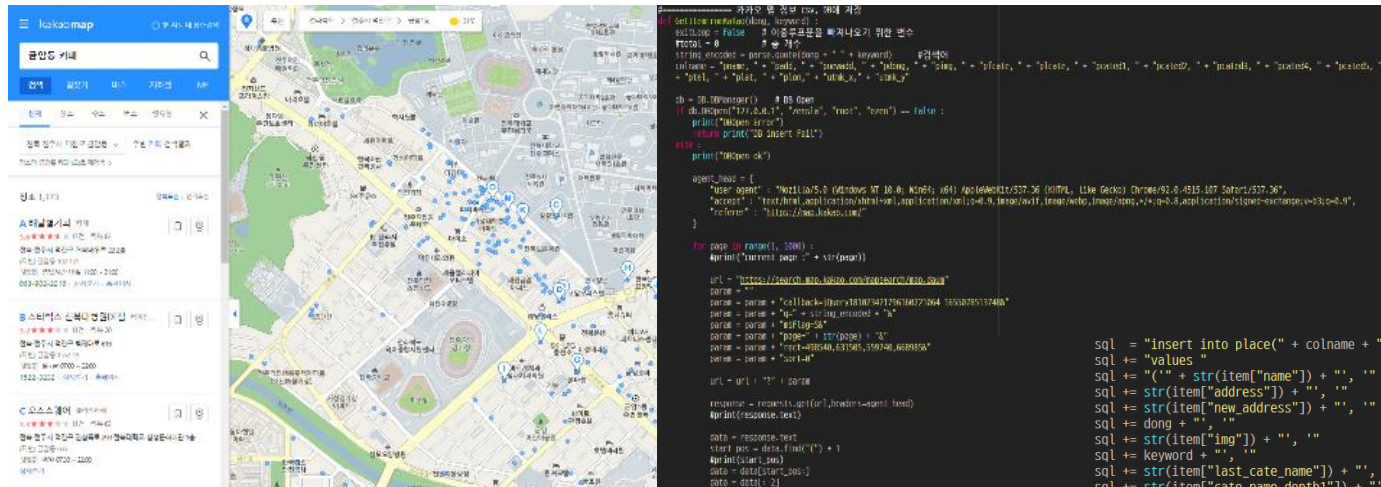
한국전력공사_지역별 전기자 운영진소 현황정보

한국전력공사에서 운영하는 지역별 전기자 운영진소 현황 정보 입니다.
지역 및 연도(2016~2021)별 현황정보를 제공합니다.

0
 0
[관심]

데이터 수집 및 저장

카카오 맵 정보 CSV를 DB에 저장



지도 좌표계 변환

```
##### 좌표계 변환기
proj_UTMK = Proj(init='epsg:5179') # UTMK
proj_WGS84 = Proj(init='epsg:4326') # WGS84

def ts_w84_to_utm(x,y):
    x1,y1 = transform(proj_WGS84, proj_UTMK, x, y)
    return x1, y1
```

```
try:
    json_data = json.loads(data)
except:
    print("error load page...")
    break
list = json_data["place"]
if list == None: break
count = 0
for item in list:
    #print(item["address"])
    temp_list = item["address"].split(" ")
    if dong not in temp_list:
        exitLoop = True
        continue
    if keyword=="주차장":
        if item["last_cate_name"] == "전기자동차 충전소":
            continue
    if keyword=="식당":
        if item["cate_name_depth2"] == "카페":
            continue
```

```
sql = "insert into place(" + colname + ") "
sql += "values "
sql += "(" + str(item["name"]) + ", "
sql += str(item["address"]) + ", "
sql += str(item["new_address"]) + ", "
sql += dong + ", "
sql += str(item["img"]) + ", "
sql += keyword + ", "
sql += str(item["last_cate_name"]) + ", "
sql += str(item["cate_name_depth1"]) + ", "
sql += str(item["cate_name_depth2"]) + ", "
sql += str(item["cate_name_depth3"]) + ", "
sql += str(item["cate_name_depth4"]) + ", "
sql += str(item["cate_name_depth5"]) + ", "
sql += str(item["tel"]) + ", "
sql += str(item["lat"]) + ", "
sql += str(item["lon"]) + ", "
# 좌표변환 실행
utm_x,utm_y = ts_w84_to_utm(item['lon'], item['lat'])
sql += str(utm_x) + ", "
sql += str(utm_y) + ", "
#print(sql)






# csv 파일에 행 단위 데이터 삽입
f = open("D:/LJH/project/data/" + fileName, "a", newline="", encoding='utf-8')
wr = csv.writer(f)
wr.writerow([str(item["name"]), str(item["address"]), str(item["new_address"]), dong, str(item["img"]), keyword, str(item["last_cate_name"]),
            str(item["cate_name_depth1"]), str(item["cate_name_depth2"]), str(item["cate_name_depth3"]), str(item["cate_name_depth4"]),
            str(item["cate_name_depth5"]), str(item["tel"]), str(item["lat"]), str(item["lon"]), str(utm_x), str(utm_y)])

f.close()











try:
    # insert 구문 실행
    db.RunSQL(sql)
except:
    print("DB insert Error")
    db.DBClose()
    break;
if (count == 0 or exitLoop == True): break
db.closeQuery()
db.DBClose()
return print("DB Close Success")
```


데이터 수집 및 저장 결과

가공

 202206291120_place_group.csv	2022-06-29 오전 11:20	한컴오피스 2018 ...	4KB
 202206291128_mM_pca_2.csv	2022-06-29 오전 11:28	한컴오피스 2018 ...	5KB
 202206291137_final_data.csv	2022-06-29 오전 11:37	한컴오피스 2018 ...	3KB
 군집_donglist.txt	2022-06-29 오후 12:41	텍스트 문서	1KB
 군집_donglist_dic.txt	2022-06-29 오후 12:43	텍스트 문서	1KB

미가공

 LSMD_ADM_SECT_UMD-전북	2022-07-04 오전 10:09	파일 폴더	
 202206271626_place.csv	2022-06-27 오후 4:35	한컴오피스 2018 ...	3,997KB
 전라북도-전기차 현황_20190620.csv	2022-06-29 오전 10:22	한컴오피스 2018 ...	8KB
 전주시 덕진구 법정동코드 조회자료.csv	2022-07-04 오전 10:53	한컴오피스 2018 ...	2KB
 전주시 덕진구 법정동코드 조회자료.xls	2022-07-04 오전 10:53	Microsoft Excel 9...	31KB
 전주시 완산구 법정동코드 조회자료.csv	2022-07-04 오전 10:54	한컴오피스 2018 ...	3KB
 전주시 완산구 법정동코드 조회자료.xls	2022-07-04 오전 10:53	Microsoft Excel 9...	8KB
 한국전력공사_지역별 전기차 현황정보_2...	2022-07-01 오후 3:28	한컴오피스 2018 ...	5KB
 한국전력공사_지역별 전기차 현황정보1....	2022-06-10 오후 3:15	한컴오피스 2018 ...	5KB
 행정안전부_지역별(법정동) 성별 연령 별 ...	2022-06-29 오전 9:32	한컴오피스 2018 ...	9,890KB

ano	pname	pdong	pfcate	plcate	plat	plon
2101	한강별 카페		카페		35.83850566	127.14094164
2102	스타벅스 전북대병원점		카페		35.84163383	127.13702234
2103	온스퀘어		카페		35.84318637	127.13002608
2104	어느날의 오후		카페	카페	35.84600052	127.13012353
2105	카페 카페 카페		카페	카페	35.84659167	127.13029193
2106	안다르		카페	카페	35.83779548	127.13448761
2107	한강별 카페		카페	카페	35.83926842	127.13038933
2108	한강별 카페		카페	카페	35.84117532	127.13280881
2109	포도출		카페	카페	35.83779937	127.12799874
2110	아미비즈		카페	카페	35.84651155	127.13351141
2111	스윙웨이		카페	카페	35.83920701	127.12842189
2112	한강별 카페		카페	카페	35.84340019	127.12813694
2113	한강별 카페		카페	카페	35.84081022	127.13455117
2114	한강별 카페		카페	카페	35.8437406	127.12753864
2115	한강별 카페		카페	카페	35.83939937	127.13486562
2116	한강별 카페		카페	카페	35.83939937	127.12761326
2117	한강별 카페		카페	카페	35.83486401	127.12938298
2118	한강별 카페		카페	카페	35.832252	127.13679828
2119	한강별 카페		카페	카페	35.83418913	127.13100304
2120	한강별 카페		카페	카페	35.83917455	127.13324269
2121	한강별 카페		카페	카페	35.83906932	127.13024555
2122	한강별 카페		카페	카페	35.83456882	127.1320184
2123	한강별 카페		카페	카페	35.83851715	127.13394896
2124	한강별 카페		카페	카페	35.8400023	127.13633483
2125	한강별 카페		카페	카페	35.83939937	127.13210767

ano	goo	dong	people	charger	car	addr	pcar	qcharger
4511110100	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	142	71	0	전주시 덕진구 법정동코드	0	0
4511110200	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	88	44	0	전주시 덕진구 법정동코드	0	0
4511110300	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	90	45	0	전주시 덕진구 법정동코드	0	0
4511110400	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	273	136	0	전주시 덕진구 법정동코드	0	0
4511110500	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	185	92	1	전주시 덕진구 법정동코드	4	0
4511110600	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	133	66	0	전주시 덕진구 법정동코드	0	0
4511110700	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	367	183	0	전주시 덕진구 법정동코드	0	0
4511110800	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	164	82	0	전주시 덕진구 법정동코드	4	0
4511110900	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	100	50	0	전주시 덕진구 법정동코드	0	1
4511111000	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	295	147	0	전주시 덕진구 법정동코드	0	0
4511111100	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	90	45	0	전주시 덕진구 법정동코드	0	0
4511111200	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	243	121	0	전주시 덕진구 법정동코드	0	0
4511111300	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	230	115	0	전주시 덕진구 법정동코드	0	0
4511111400	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	207	103	0	전주시 덕진구 법정동코드	0	0
4511111500	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	177	88	0	전주시 덕진구 법정동코드	0	0
4511111600	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	212	106	0	전주시 덕진구 법정동코드	0	0
4511111700	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	467	233	1	전주시 덕진구 법정동코드	4	0
4511111800	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	912	456	2	전주시 덕진구 법정동코드	7	1
4511111900	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	305	152	0	전주시 덕진구 법정동코드	39	1
4511112000	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	5297	2648	0	전주시 덕진구 법정동코드	0	0
4511112100	전주시 덕진구 법정동코드	전주시 덕진구 법정동코드	1616	808	4	전주시 덕진구 법정동코드	15	4

hno	hname	haddr	bdong	bfcate	blat	blon
1	노외공원주차장	전북 전주시 완산구 효자동2가 1231-8	효자동2가	주차장	35.81646895	127.10469154
2	전주대평생교육원 주차장	전북 전주시 완산구 효자동2가 1311-1	효자동2가	주차장	35.81030436	127.08910888
3	무계개미 주차장	전북 전주시 완산구 효자동2가 1158-20	효자동2가	주차장	35.81261372	127.10633292
4	한우빌딩 주차장	전북 전주시 완산구 효자동2가 1352	효자동2가	주차장	35.80800956	127.10553901
5	전주비전대학교 주차장6	전북 전주시 완산구 효자동2가 1070	효자동2가	주차장	35.80754151	127.08919527
6	평내무선 공공주차장	전북 전주시 덕진구 인후동2가 781-7	인후동2가	주차장	35.8355398	127.15562506
7	전주아중현대아파트 주차장	전북 전주시 덕진구 인후동2가 858-2	인후동2가	주차장	35.8317946	127.16515336
8	이중지구 산림청영 공공주차장	전북 전주시 덕진구 인후동2가 907-2	인후동2가	주차장	35.82588195	127.16278941
9	인후3동민배들 주차장	전북 전주시 덕진구 인후동2가 807-6	인후동2가	주차장	35.83582577	127.16401989
10	신림조합중앙회 전북지역본부 주차장	전북 전주시 덕진구 인후동2가 791	인후동2가	주차장	35.84026457	127.155736
11	근영여고 앞 공공주차장	전북 전주시 완산구 중화산동2가 651-6	중화산동2가	주차장	35.81677172	127.12372884
12	예교리음 주차장	전북 전주시 완산구 중화산동2가 570-1	중화산동2가	주차장	35.82231466	127.1215729
13	전주형원 주차장	전북 전주시 완산구 중화산동2가 166	중화산동2가	주차장	35.8151585	127.12549387
14	공공주차장	전북 전주시 완산구 중화산동2가 644-3	중화산동2가	주차장	35.81320903	127.12107216
15	24시몽천민물장어 주차장	전북 전주시 완산구 중화산동2가 591-5	중화산동2가	주차장	35.81922814	127.11877963

법정동 추출

```
path = "D://LJH//project//data"          # 경로는 사용자에게 맞게 설정
os.chdir(path)                           # 경로를 path로 설정
df1 = pd.read_csv("전주시 완산구 법정동코드 조회자료.csv", encoding="euc-kr")
df2 = pd.read_csv("전주시 덕진구 법정동코드 조회자료.csv", encoding="euc-kr")

def appendDong(df_list, df) :
    for i in range(0, len(df)):
        df_list.append(df['법정동명'][i].split(" ")[-1])

dong_list = []
appendDong(dong_list, df1)
appendDong(dong_list, df2)
print(dong_list)
```

전주시 완산구, 덕진구 법정동을 불러와서
데이터를 활용 할 수 있도록 변경

결과 :

```
In [6]: print(dong_list)
['완산구', '중앙동1가', '중앙동2가', '중앙동3가', '중앙동4가', '경원동1가', '경원동2가', '경원동3가', '풍남동1가', '풍남동2가', '풍남동3가', '전동', '전동3가', '다가동1가', '다가동2가', '다가동3가', '다가동4가', '고사동', '교동', '태평동', '중노송동', '남노송동', '동완산동', '서완산동1가', '서완산동2가', '동서학동', '서서학동', '중화산동1가', '중화산동2가', '서신동', '석구동', '원당동', '평화동1가', '평화동2가', '평화동3가', '중인동', '용복동', '삼천동1가', '삼천동2가', '삼천동3가', '효자동1가', '효자동2가', '효자동3가', '대성동', '색장동', '상림동', '서노송동', '덕진구', '진북동', '인후동1가', '인후동2가', '덕진동1가', '덕진동2가', '금암동', '팔복동1가', '팔복동2가', '팔복동3가', '산정동', '금상동', '우아동1가', '우아동2가', '우아동3가', '호성동1가', '호성동2가', '호성동3가', '전미동1가', '전미동2가', '송천동1가', '송천동2가', '반월동', '화전동', '용정동', '성덕동', '원동', '고랑동', '여의동', '만성동', '장동', '팔복동4가', '도도동', '강흥동', '도덕동', '남정동', '중동', '여의동2가']
```

주성분 분석 준비

DF 만든 후 DB에 저

[illegible]

추출한 데이터 합침

```
# 인구수와 전기와 데이터를 left조인으로 합치고 nan은 0으로 설정
df_temp = pd.merge(df, df_jeonju_people, left_index=True, right_index=True, how='left')
df_temp = pd.merge(df_temp, df_jeonju_car, left_index=True, right_index=True, how='left')
df_temp = df_temp.fillna(0)

# 전기와 충전소와 마지막 데이터 위치 전환
col_list = df_temp.columns.tolist()
col_list[col_list.index('전기차 충전소')], col_list[-1] = col_list[-1], col_list[col_list.index('전기차 충전소')]

df2_temp = df_temp[col_list]

# 최종 DataFrame = df2_temp

# 겹침치가 있는지 확인
df2_temp.isna().sum()

# 최종 DataFrame 출력 확인
df2_temp.head()
```

필요 데이터 전처리

```
##### 전주시 인구수
filename = "행정안전부_지역별(법정동) 성별 연령별 주민등록 인구수_20220531.csv"
df_people = pd.read_csv(filename, encoding="euc-kr")

# 법정동코드를 이용하여 전주시 법정동만 추출
df_jeonju_people_all = df_people[(df_people['법정동코드'] >= 4511110100) & (df_people['법정동코드'] <= 4511313800)]
df_col_list = df_jeonju_people_all.columns # columns 함수를 사용하여 col_list 생성
a = np.where(np.array(df_col_list) == '만19세남자')[0][0] # 만 19세 남자 인덱스 위치
b = np.where(np.array(df_col_list) == '만69세남자')[0][0] # 만 69세 남자 인덱스 위치
c = np.where(np.array(df_col_list) == '만19세여자')[0][0] # 만 19세 여자 인덱스 위치
d = np.where(np.array(df_col_list) == '만69세여자')[0][0] # 만 69세 여자 인덱스 위치
df_dong = df_jeonju_people_all[['읍면동명']] # 읍면동명만 추출
df_male = df_jeonju_people_all.iloc[:, a:b+1] # 만19~69세 남자만 추출
df_female = df_jeonju_people_all.iloc[:, c:d+1] # 만19~69세 여자만 추출

# 추출한 데이터들을 열 기준으로 merge //
df_jeonju_people = pd.merge(df_dong, df_male, how="left", left_index=True, right_index=True)
df_jeonju_people = pd.merge(df_jeonju_people, df_female, how="left", left_index=True, right_index=True)
# '읍면동명'을 인덱스로 설정
df_jeonju_people = df_jeonju_people.set_index('읍면동명')
# 행별 합계를 '계'로 설정
df_jeonju_people['계'] = df_jeonju_people.sum(axis=1)
# '계'만 추출하고 '계' -> 'people'로 변경
df_jeonju_people = df_jeonju_people['계'].to_frame()
df_jeonju_people.rename(columns = {'계': 'people'}, inplace = True)

##### 전주시 전기차
filename = "전라북도 전기차 현황_20190620..csv"
df_car = pd.read_csv(filename, encoding="euc-kr")

# '시.군'에서 '전주시'만 추출
df_jeonju_car = df_car[df_car['시.군'] == '전주시']
# '이하 주소'를 인덱스로 설정
df_jeonju_car = df_jeonju_car.set_index('이하 주소')
# '대수'만 추출하고 '대수' -> 'car'로 변경
df_jeonju_car = df_jeonju_car['대수'].to_frame()
df_jeonju_car.rename(columns = {'대수': 'car'}, inplace = True)
```

Part 4, 데이터 분석 과정

- 데이터 처리

주성분 분석 준비

결과 값

```
....: df2_temp.head()
Out[12]:
```

	카페	주차장	식당	편의점	대형마트	car	공공기관	영화관	people	전기차 충전소
강흥동	0	0	0	0	0.0	2	0	61.0	0	
경원동1가	20	8	34	1	0 1.0	10	0	185.0	1	
경원동2가	9	2	20	2	0 0.0	1	0	133.0	0	
경원동3가	19	5	59	4	0 0.0	26	0	367.0	1	
고량동	1	1	9	4	0 1.0	1	0	1277.0	2	



csv 생성 후 저장

```
#===== 저장할 csv 생성
current_date = datetime.today().strftime("%Y%m%d%H%M")

fileName = current_date + '_place_group.csv' # 파일 이름

df2_temp.to_csv("D:/LJH/project/data/"+fileName, sep=',', encoding="euc-kr")

# 원본 DataFrame df, 복사본 df_t
df_t = df2_temp.copy()
```



	A	B	C	D	E	F	G	H	
1	법정동	people	식당	카페	편의점	공공기관	대형마트	영화관	
2	강흥동	61	0	0	0	2	0	0	
3	경원동1가	185	34	20	1	10	0	0	
4	경원동2가	133	20	9	2	1	0	0	
5	경원동3가	367	58	19	4	27	0	0	
6	고량동	1277	9	1	4	1	0	0	
7	교사동	467	202	88	8	13	0	6	
8	교동	912	94	50	3	3	0	0	
9	금상동	582	3	2	0	2	0	0	
10	금암동	13386	10	14	45	13	0	0	
11	남노송동	1616	21	8	1	24	0	0	
12	남정동	173	0	1	0	1	0	0	
13	다가동1가	230	11	9	1	3	0	0	
14	다가동2가	207	2	2	0	1	0	0	
15	다가동3가	177	82	23	2	3	0	0	
16	다가동4가	212	64	31	1	8	0	0	
17	대성동	1039	15	6	2	7	0	0	
18	덕진동1가	5086	293	89	22	44	0	1	
19	덕진동2가	10335	186	34	16	32	0	0	
20	동서학동	2943	4	11	7	9	0	0	
21	동완산동	1177	17	1	0	4	0	0	

주성분 분석(데이터 준비 & 처리)

Min – Max Scaler

```
#===== 데이터 스케일링
from sklearn.preprocessing import MinMaxScaler # MinMaxScaler 사용 (정규화)
from sklearn.preprocessing import StandardScaler # StandardScaler 사용 (표준화)

# MinMaxScaler
scaler_mM = MinMaxScaler()
df_scaled_mM = scaler_mM.fit_transform(df_t.to_numpy())
df_scaled_mM = pd.DataFrame(df_scaled_mM, columns=df_t.columns, index=df_t.index)
# 데이터 분류 확인을 위해 임의로 충전소 10개 미만 : t1 / 10이상 20개 미만 : t2 / 20개 이상 : t3로 설정
df_scaled_mM.loc[df_t['전기차 충전소'] < 10, 'label'] = 't1'
df_scaled_mM.loc[(df_t['전기차 충전소'] >= 10) & (df_t['전기차 충전소'] < 20), 'label'] = 't2'
df_scaled_mM.loc[df_t['전기차 충전소'] >= 20, 'label'] = 't3'
# 결과 확인
print(df_scaled_mM.head())

In [15]: print(df_scaled_mM.head())
```

	카페	주차장	식당	편의점	...	영화관	people	전기차 충전소	label
강흥동	0.000000	0.000000	0.000000	0.000000	...	0.0	0.001664	0.000000	t1
경원동1가	0.121212	0.058824	0.078704	0.020833	...	0.0	0.005047	0.015873	t1
경원동2가	0.054545	0.014706	0.046296	0.041667	...	0.0	0.003629	0.000000	t1
경원동3가	0.115152	0.036765	0.136574	0.083333	...	0.0	0.010013	0.015873	t1
고랑동	0.006061	0.007353	0.020833	0.083333	...	0.0	0.034841	0.031746	t1

[5 rows x 11 columns]

Standard Scaler

```
# StandardScaler
scaler_st = StandardScaler()
df_scaled_st = scaler_st.fit_transform(df_t.to_numpy())
df_scaled_st = pd.DataFrame(df_scaled_st, columns=df_t.columns, index=df_t.index)
# 데이터 분류 확인을 위해 임의로 충전소 10개 미만 : t1 / 10이상 20개 미만 : t2 / 20개 이상 : t3로 설정
df_scaled_st.loc[df_t['전기차 충전소'] < 10, 'label'] = 't1'
df_scaled_st.loc[(df_t['전기차 충전소'] >= 10) & (df_t['전기차 충전소'] < 20), 'label'] = 't2'
df_scaled_st.loc[df_t['전기차 충전소'] >= 20, 'label'] = 't3'
# 결과 확인
print(df_scaled_st.head())

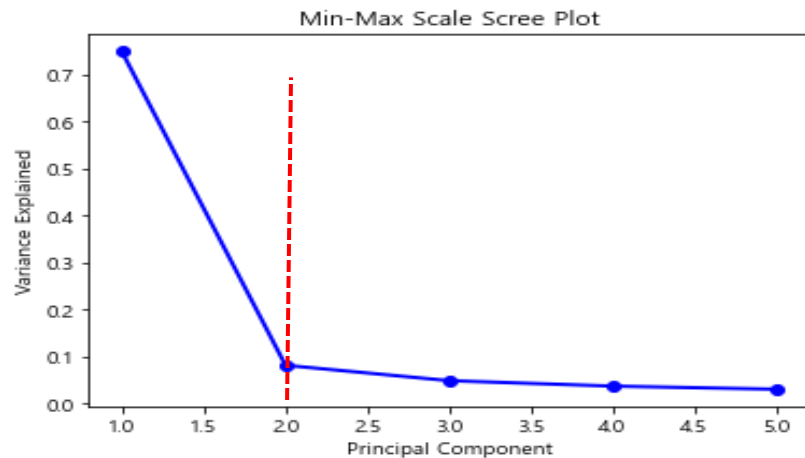
In [16]: print(df_scaled_st.head())
```

	카페	주차장	식당	...	people	전기차 충전소	label
강흥동	-0.774328	-0.525189	-0.752178	...	-0.599942	-0.613785	t1
경원동1가	-0.211559	-0.079741	-0.497975	...	-0.586916	-0.525163	t1
경원동2가	-0.521082	-0.413827	-0.602647	...	-0.592379	-0.613785	t1
경원동3가	-0.239698	-0.246784	-0.311061	...	-0.567798	-0.525163	t1
고랑동	-0.746190	-0.469508	-0.684889	...	-0.472209	-0.436542	t1

[5 rows x 11 columns]

주성분 분석

Scree Plot을 통해 분석에 포함시킬 주성분 개수 확인



```

...: x = df_scaled_mM.iloc[:, :-2].values
...: n_comp = 2 # 주성분을 2개로 설정
...: pca = PCA(n_components=n_comp)
...: pca_fit = pca.fit_transform(x)
...: print(sum(pca.explained_variance_ratio_))
0.8326530017279976

```

```

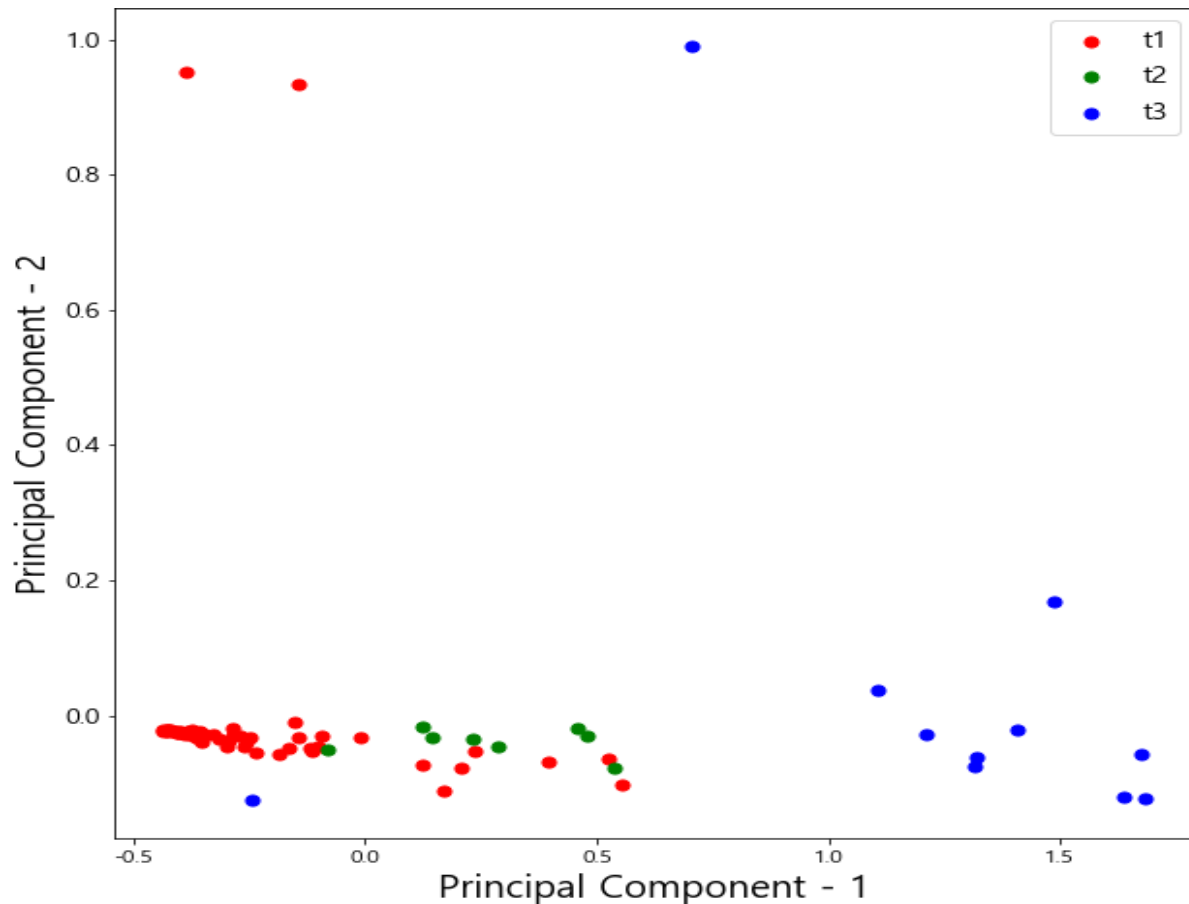
=====
PC1 구성
카페      0.339392
주차장    0.092730
식당      0.512461
편의점    0.391139
대형마트  0.026280
car       0.311160
공공기관  0.424589
영화관   0.058634
people    0.423537
dtype: float64
=====
PC2 구성
카페      -0.067713
주차장    -0.105108
식당      0.027989
편의점    -0.010707
대형마트  0.974847
car       -0.052527
공공기관  -0.104788
영화관    0.051925
people     0.129257
dtype: float64

```

제 2 주성분 이후부터 분산의 기울기가 줄어
들
주성분 2개가 적당하다고 판단

주성분 분석 (데이터 표현력 확인)

산점도를 통해 결과로 도출된 주성분이 데이터를 얼마나 잘 표현하는지 확인



T1 : 충전소 개수가 10개 미만

T2 : 충전소 개수가 10개 이상 20개 미만

T3 : 충전소 개수가 20개 이상

주성분 분석 (최종 데이터)

```
#===== 최종 데이터 셋 : df_final
# PCA분석 결과 확인하고 영향력이 높은 변수 채택
df_final = df_t.loc[:,('people','식당','카페','편의점','공공기관','대형마트','영화관')]
# 결과 확인
print(df_final.head())
```

```
In [19]: print(df_final.head())
```

	people	식당	카페	편의점	공공기관	대형마트	영화관
강흥동	61.0	0	0	0	2	0	0
경원동1가	185.0	34	20	1	10	0	0
경원동2가	133.0	20	9	2	1	0	0
경원동3가	367.0	59	19	4	26	0	0
고랑동	1277.0	9	1	4	1	0	0

```
#===== 최종 데이터 저장할 csv 생성
current_date = datetime.today().strftime("%Y%m%d%H%M")

fileName = current_date + '_final_data.csv' # 파일 이름

df_final.to_csv("D:/LJH/project/data/"+fileName, sep=',', encoding="euc-kr")
```

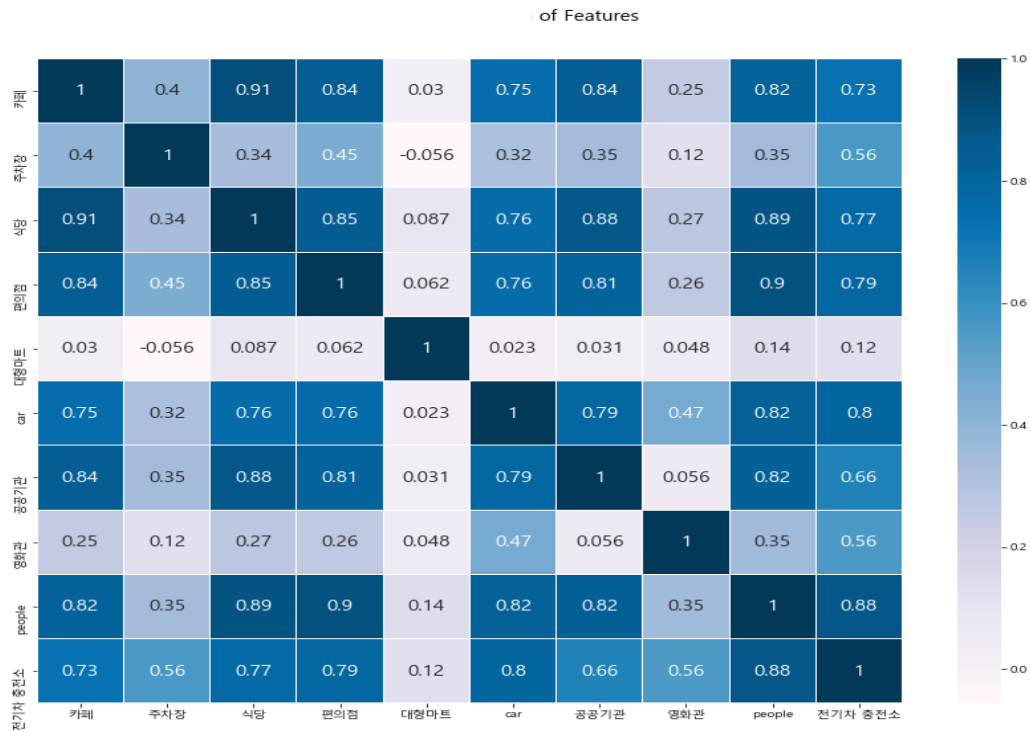
1차 군집에 활용 할 주성분 결과 및 데이터 저장

Part 4, 데이터 분석 과정

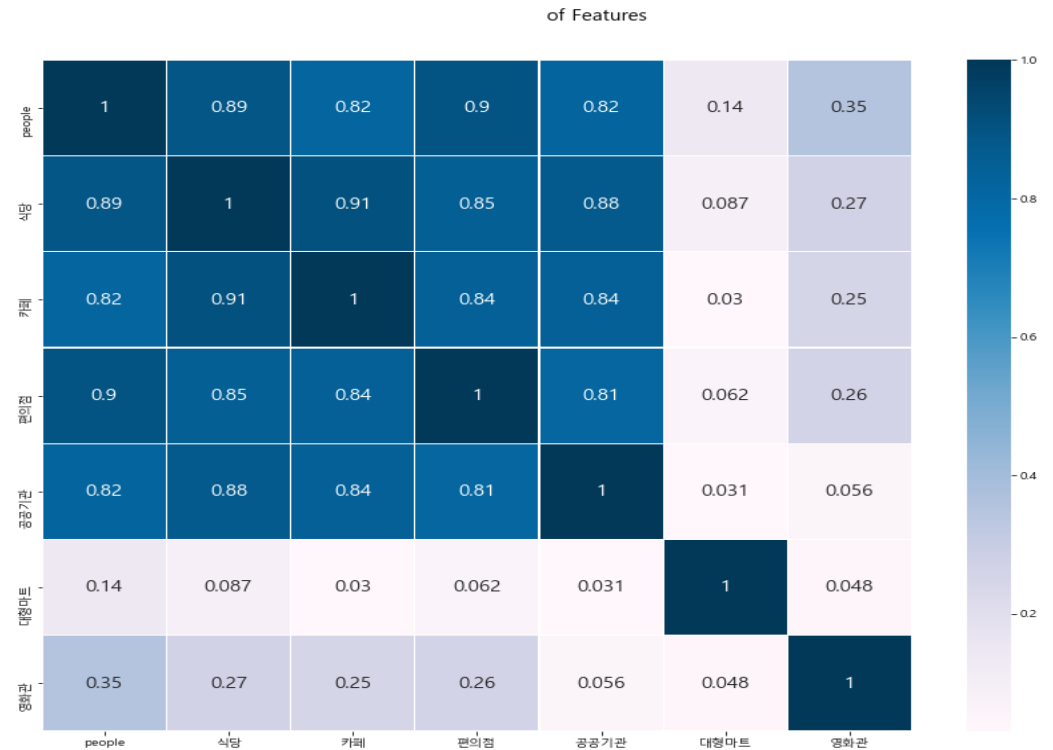
- 데이터 처리

상관 분석

모든 변수와의 상관계수



주성분 분석 후 무의미한 변수 제거



- 다중 공선성을 보이는 각 변수들과의 관계를 주성분 분석을 통해 해결 하였음.

입지분석 처리

위치

```
# 파일이 있는 곳으로 path 변경
path = "D:\LJH\seoul_data\data"
os.chdir(path)
```

```
#읍면동 경계 가져오기
emd = geopandas.read_file('LSMD_ADM_SECT_UMD_45.shp', encoding = 'ANSI')
# 전주 데이터만 사용
emd = emd[emd['COL_ADM_SE'] == '45110']
emd.head()
```

	EMD_CD	EMD_NM	SGG_OID	COL_ADM_SE	GID	geometry
293	45111115	다가동3가	10241	45110	929	POLYGON ((967577.773 1757942.305, 967598.481 1...
294	45111135	중인동	10239	45110	930	POLYGON ((964716.960 1754453.411, 964735.574 1...
295	45111106	경원동2가	10238	45110	931	POLYGON ((968422.323 1757914.224, 968421.869 1...
326	45111145	상림동	15988	45110	875	POLYGON ((960920.693 1756733.714, 960927.885 1...
327	45111139	삼전동3가	15987	45110	876	POLYGON ((962437.386 1757056.410, 962446.083 1...

Client

```
jj_restaurant = pd.read_csv('place.csv', encoding='utf-8')
jj_restaurant = jj_restaurant[(jj_restaurant['pfcate'] == '식당') |
                               (jj_restaurant['pfcate'] == '편의점') | (jj_restaurant['pfcate'] == '카페')]
```

```
jj_restaurant = jj_restaurant[['pname', 'pdong', 'pfcate', 'plcate', 'utm_k_x', 'utm_k_y']]
jj_restaurant.columns = ['pname', 'pdong', 'pfcate', 'plcate', 'utm_k_x', 'utm_k_y']
```

```
jj_restaurant = geopandas.GeoDataFrame(jj_restaurant, geometry=geopandas.points_from_xy(jj_restaurant['utm_k_x'], jj_restaurant['utm_k_y']))
jj_restaurant.set_crs(epsg = 5179, inplace = True)
jj_restaurant.head(2)
```

	pname	pdong	pfcate	plcate	utm_k_x	utm_k_y	geometry
0	필드오	진북동	카페	디저트카페	966678.137038	1.759592e+06	POINT (966678.137 1759591.504)
1	페더럴	진북동	카페	카페	967309.888649	1.759511e+06	POINT (967309.889 1759511.105)

Facility

```
jj_facility = pd.read_csv('place.csv', encoding='utf-8')
jj_facility = jj_facility[(jj_facility['pfcate'] == '주차장')]
```

```
jj_facility = jj_facility[['pname', 'padd', 'pdong', 'pfcate', 'plat', 'plon', 'utm_k_x', 'utm_k_y']]
jj_facility.columns = ['pname', 'padd', 'pdong', 'pfcate', 'plat', 'plon', 'utm_k_x', 'utm_k_y']
```

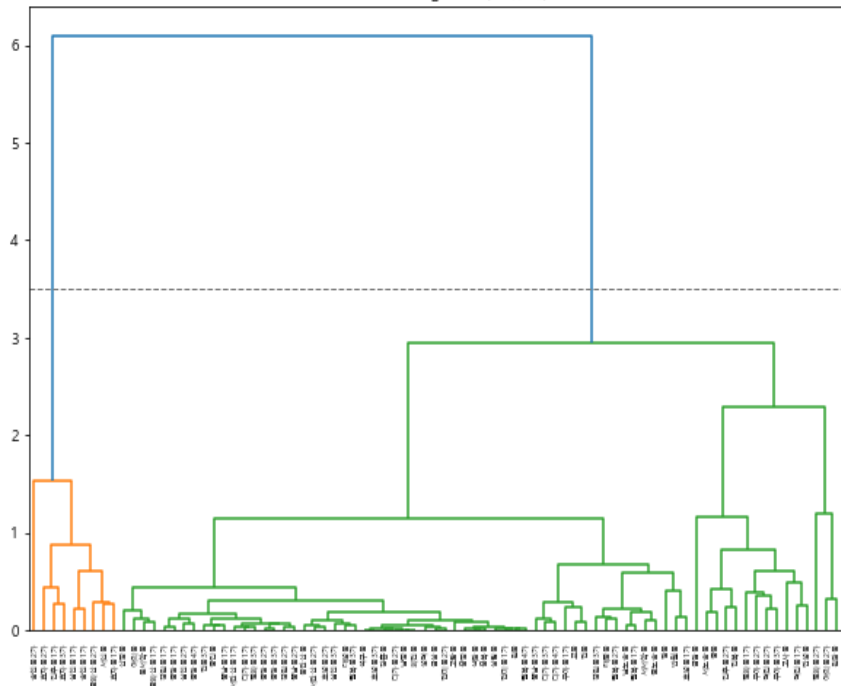
```
jj_facility = geopandas.GeoDataFrame(jj_facility, geometry=geopandas.points_from_xy(jj_facility['utm_k_x'], jj_facility['utm_k_y']))
jj_facility.set_crs(epsg = 5179, inplace = True)
jj_facility.head(2)
```

	pname	padd	pdong	pfcate	plat	plon	utm_k_x	utm_k_y	geometry
48	이정형외과의원주차장	전북 전주시 덕진구 진북동 321-7	진북동	주차장	35.829476	127.143258	967778.525143	1.759279e+06	POINT (967778.525 1759278.797)
49	덕진구청 주차장1	전북 전주시 덕진구 진북동 416-12	진북동	주차장	35.829118	127.134262	966965.843738	1.759242e+06	POINT (966965.844 1759242.164)

Hierarchical Clustering

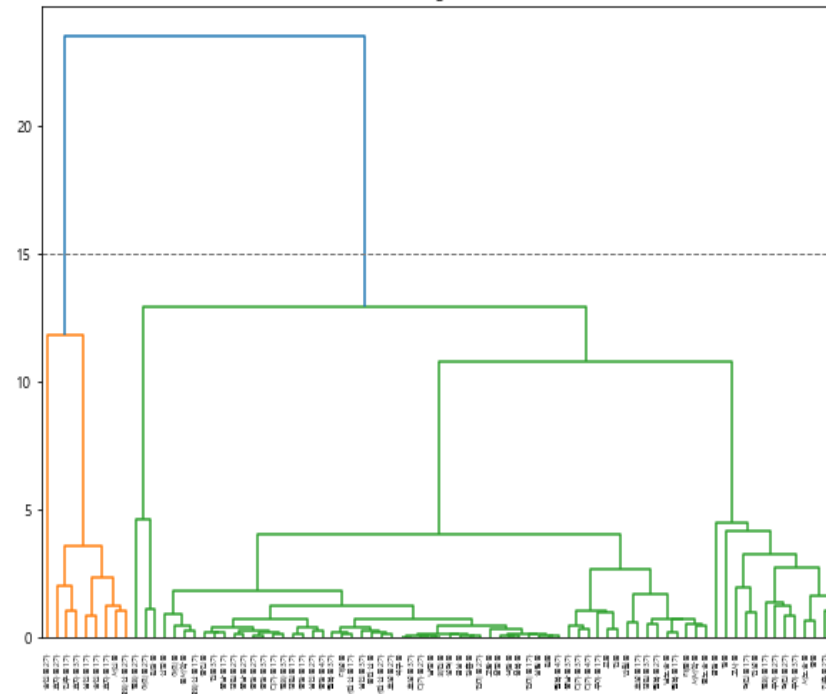
Min – Max Scaler

Ward Dendrograms (정규화)



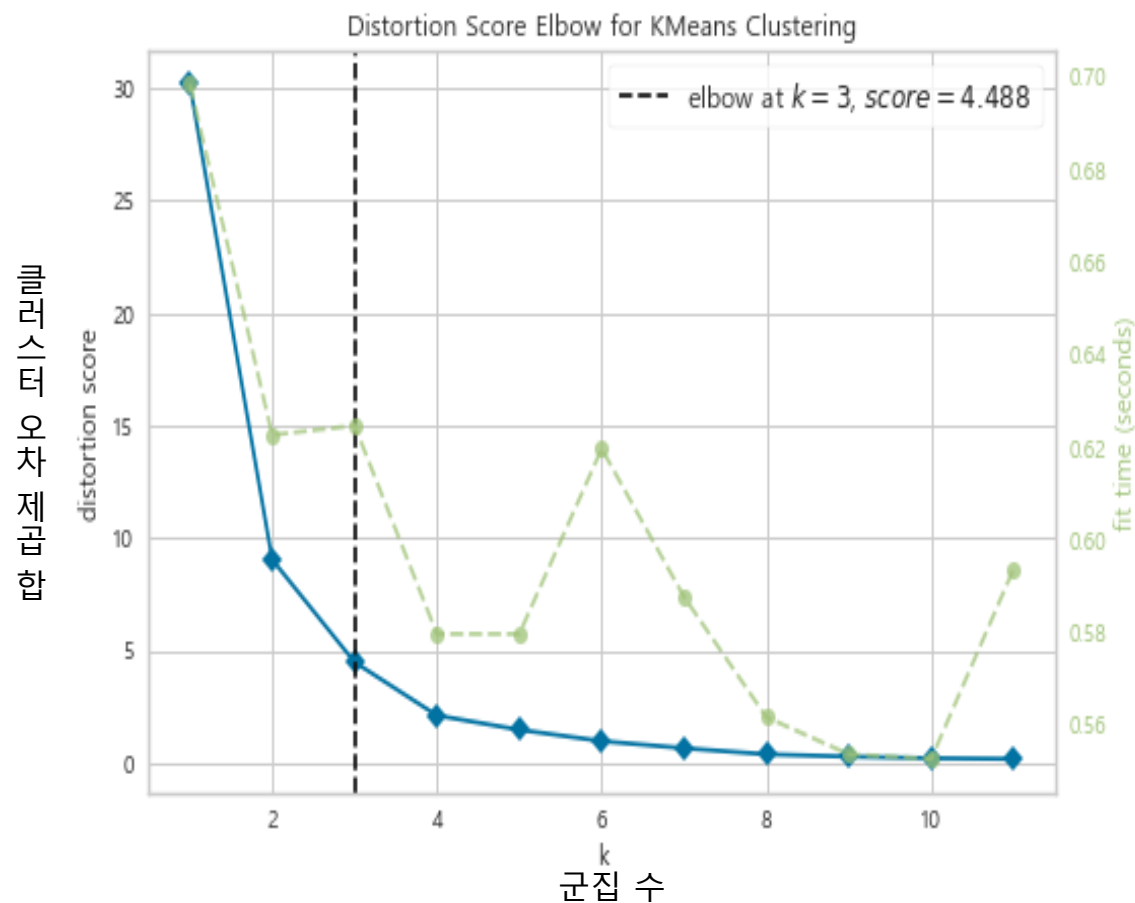
Standard Scaler

Ward Dendrograms (표준화)



- 군집간의 거리가 먼 Ward 연결법을 선택
- 군집 개수 2개

Elbow, Silhouette method



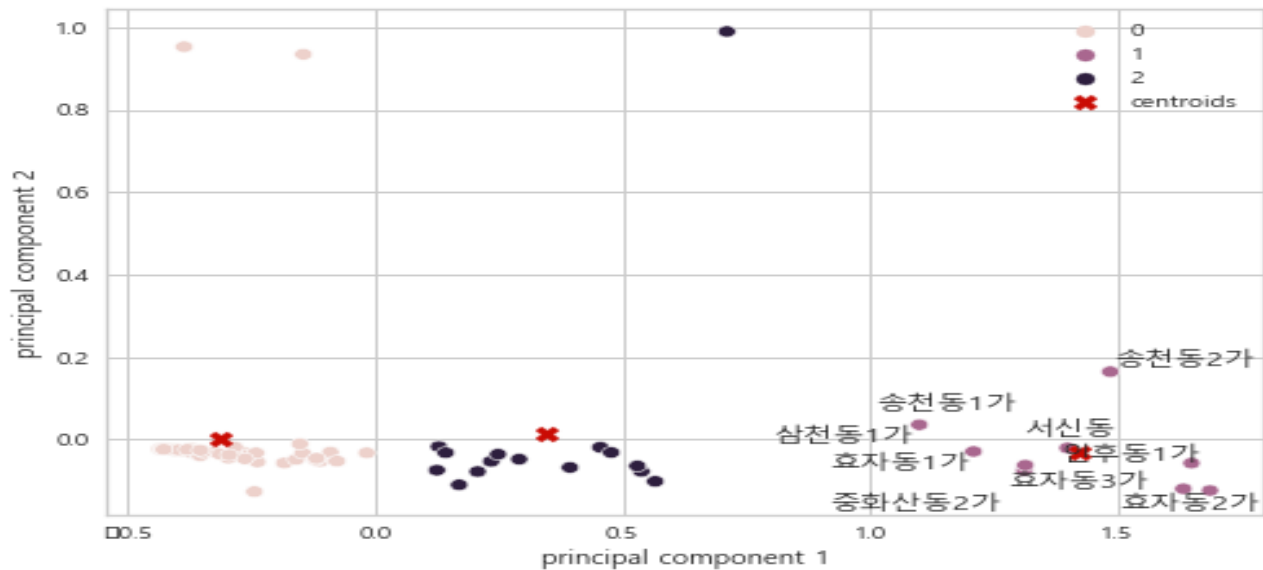
Elbow, Silhouette method 결과 값에 근거하여

K - means, K- medoids, GMM 분석을 하기 위해

최적의 군집 개수를 분석

결과 : 군집 수는 3개가 적당하다고 판단

K-Means

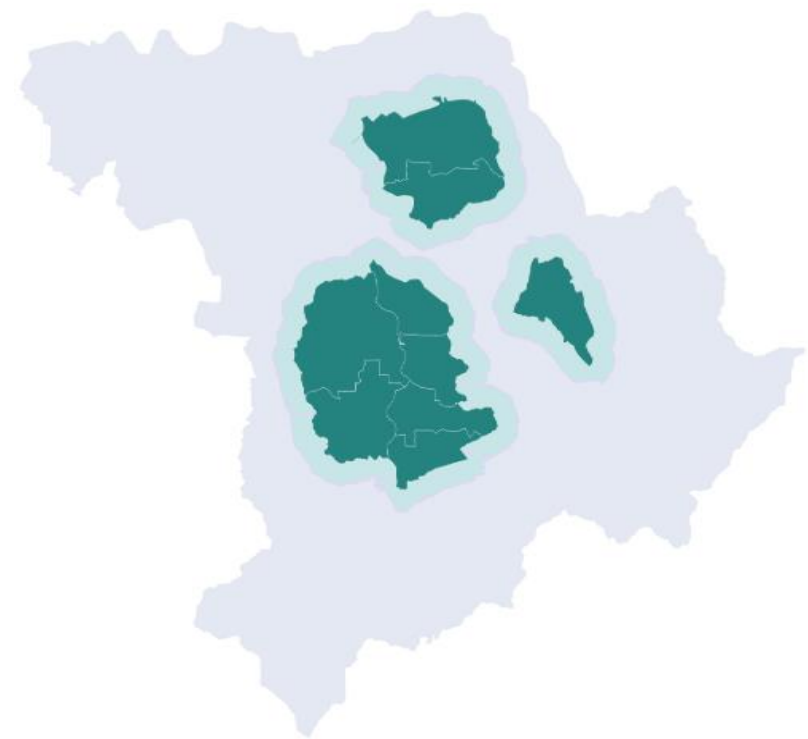


```

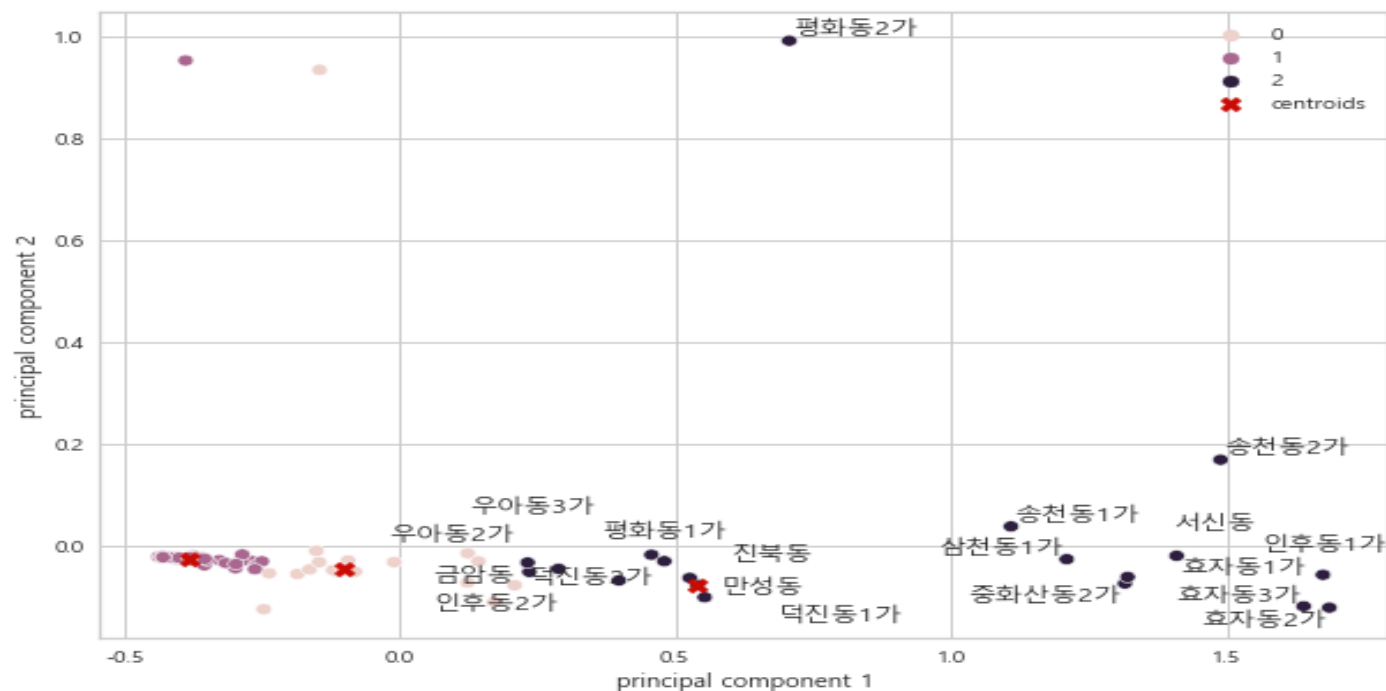
...: df_simple = df.loc[kmeans.labels_ == 1]
...: labels_s = df_simple.index
...: print(labels_s)
Index(['삼천동1가', '서신동', '송천동1가', '송천동2가', '인후동1가', '중화산동2가', '효자동1가', '효자동2가',
      '효자동3가'],
      dtype='object')

```

결과



K-Medoids

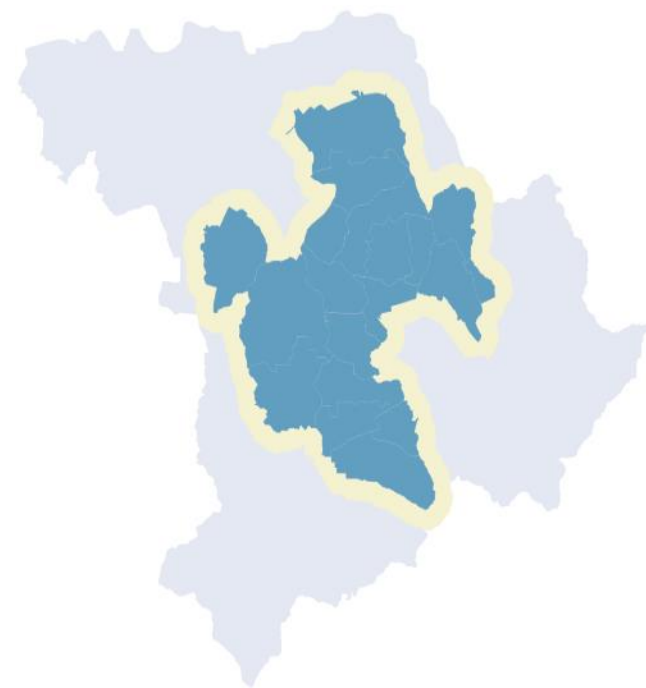


```

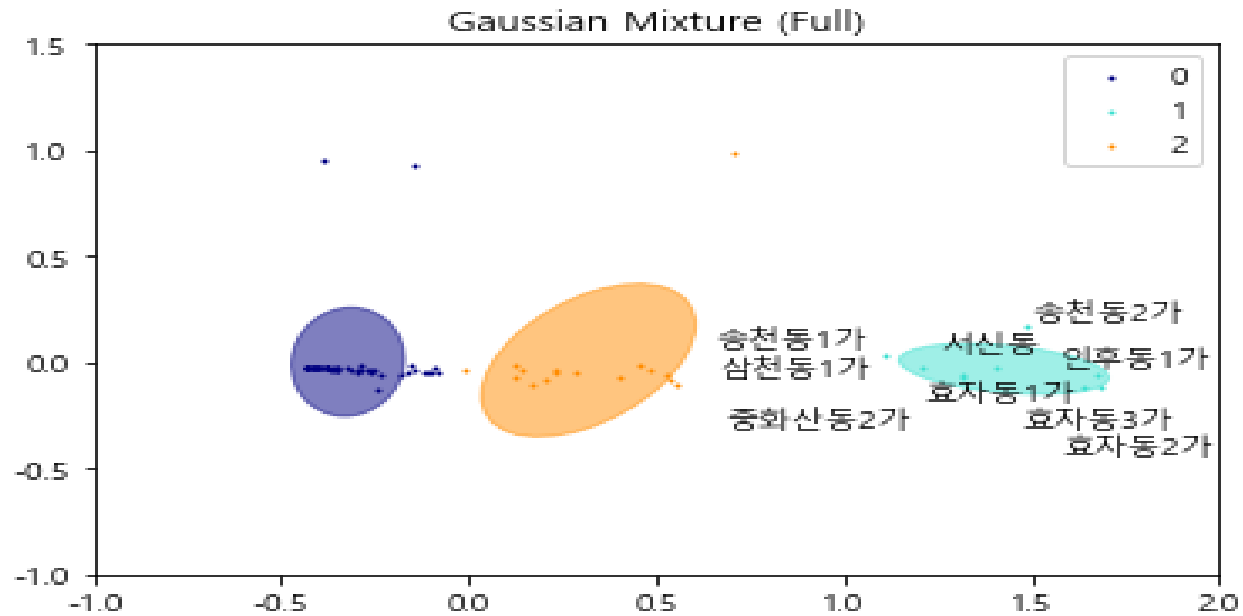
...: df_simple = df.loc[kmedoid.labels_ == 2]
...: labels_s = df_simple.index
...: print(labels_s)
Index(['금암동', '덕진동1가', '덕진동2가', '만성동', '삼천동1가', '서신동', '송천동1가', '송천동2가',
      '우아동2가', '우아동3가', '인후동1가', '인후동2가', '중화산동2가', '진북동', '평화동1가', '평화동2가',
      '효자동1가', '효자동2가', '효자동3가'],
      dtype='object')

```

결과

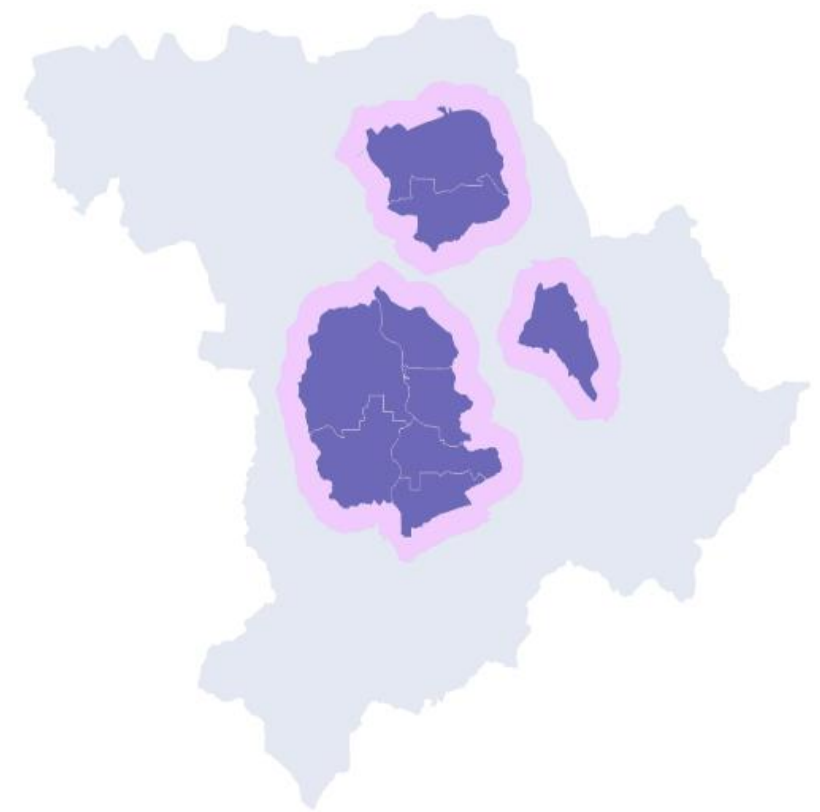


Gaussian Mixture Model: GMM

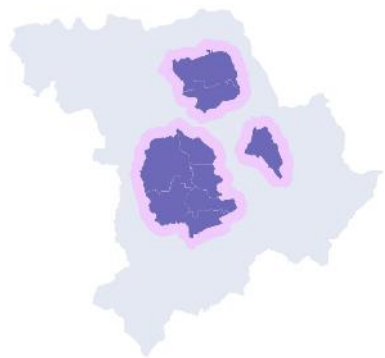


```
...: df_simple = df.loc[gmm_full.predict(data) == 1]
...: labels_s = df_simple.index
...: print(labels_s)
Index(['삼천동1가', '서신동', '송천동1가', '송천동2가', '인후동1가', '중화산동2가', '효자동1가', '효자동2가',
      '효자동3가'],
      dtype='object')
```

결과



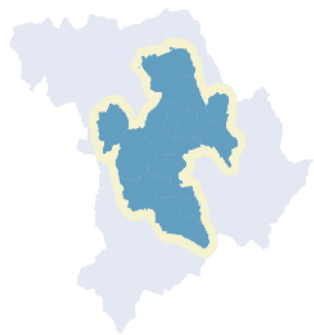
법정동 군집 분석 결과



Gaussian Mixture Model



Hierarchical Clustering



K-means Clustering



K-medoids Clustering

법정동 군집 분석 결과

군집 분석 결과

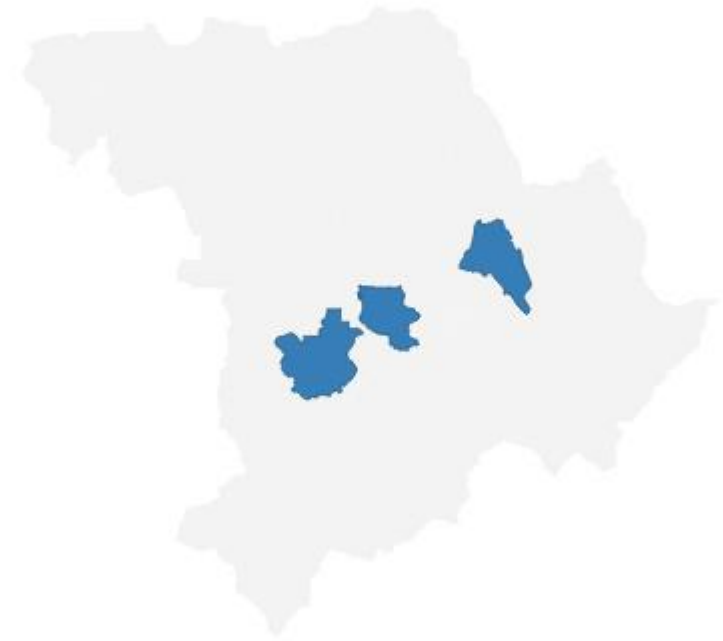
9개의 지역(삼천동1가, 서신동, 송천동1,2가 인후동 1가,
중화산동 2가 효자동1,2,3가)

PCA 가중치가 높았던
식당, 카페, 공공기관 3개를 기준

인후동 1가, 중화산동 2가, 효자동 2가

우선 입지 대상 법정동 선정

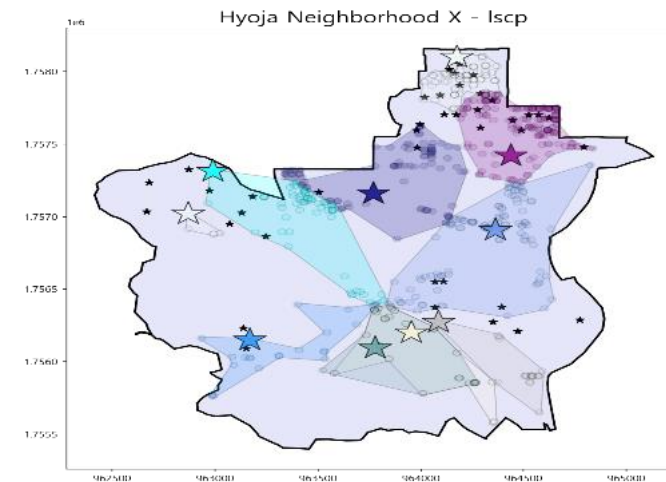
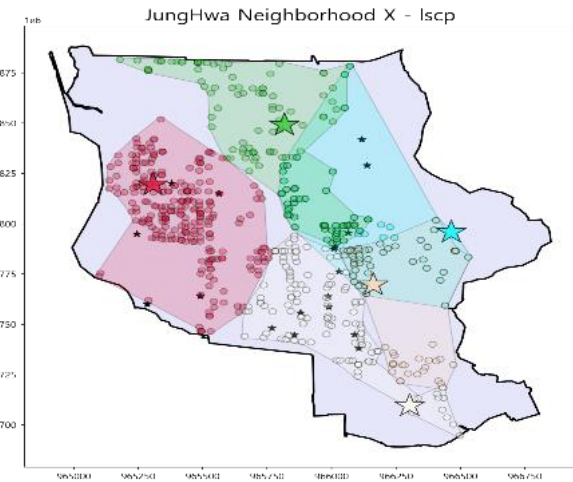
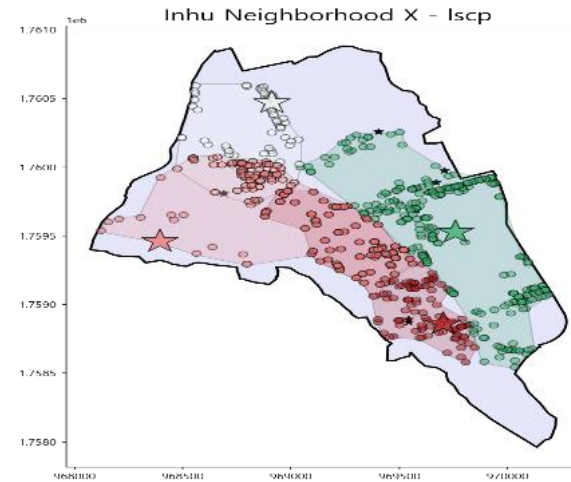
군집 결과가 많이 나온 법정동 대상



LSCP (Location Set Covering Problem)

$$\begin{aligned}
 & \text{Minimize } \sum_{j \in J} x_j \\
 & \text{s.t. } \sum_{j \in N_i} x_j \geq 1 \quad \text{for all } i \in I \\
 & \quad x_j \in 0, 1 \quad \text{for all } j \in J
 \end{aligned}$$

문자	의미
i	수요 포인트 index
j	설비지역 포인트 index
I	수요 포인트 집합
J	설비지역 포인트 집합
x	설비 후보 지역 중 위치 j 에 설치되면 1, 아니면 0
y	적어도 하나의 설비로 그 포인트가 커버되면 1, 아니면 0



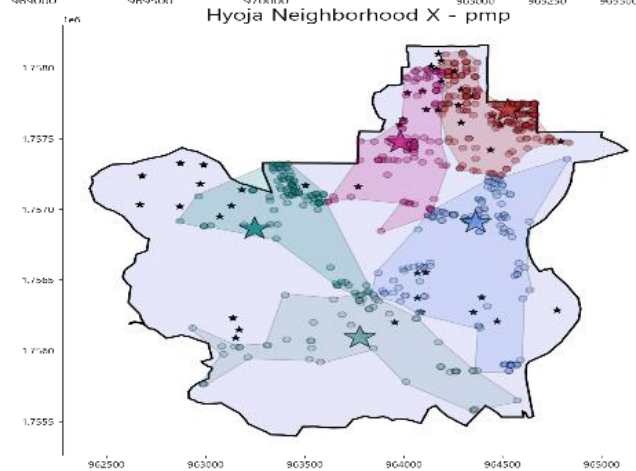
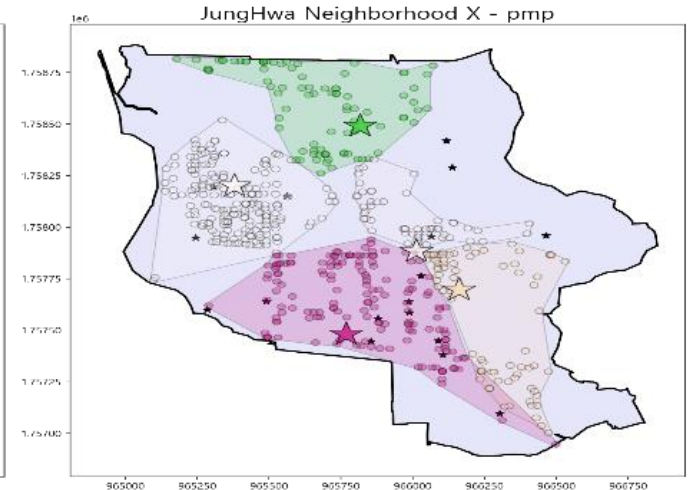
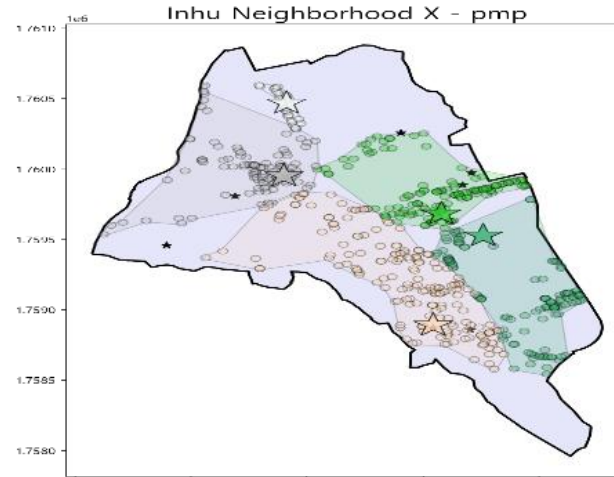
- 지역 수요를 최대한 만족시킬 수 있는 최적 입지 선정
- 법정동 내 식당, 편의점, 카페 등을 수요 포인트로 주차장을 설비 후보 지역으로 선정하여 분석

PMP (P-Median Problem)

Inputs:

 h_i = 수요지 i 의 수요량 d_{ij} = 수요지 i 와 시설물의 입지점 j 의 거리 p = 시설물의 수

Decision variables:

 $x_j =$ 1, 만약 노드 j 에 시설물이 설치되면,
0, 그렇지 않으면. $y_{ij} =$ 1, 만약 노드 j 에 시설물이 노드 i 의
총수요를 충족시키면,
0, 그렇지 않으면.Subject to $\min \sum_i \sum_j h_i d_{ij} y_{ij}$ (1-1) $\sum_j y_{ij} = 1$ (for all i) (1-2) $\sum_j x_j = p$ (1-3) $y_{ij} \leq x_j$ (for all i, j) (1-4) $y_{ij} \in 0,1$ (for all i, j) (1-5) $x_j \in 0,1$ (for all j) (1-6)

- 수요지로부터 5개의 설비 후보 지역간의 총 거리가 가장 짧게 있는 입지 선정
- 법정동 내 식당, 편의점, 카페 등을 수요 포인트로 주차장을 설비 후보 지역으로 선정하여 분석

PCP (P-Center Problem)

Inputs:

 h_i = 수요지 i 의 수요량 d_{ij} = 수요지 i 와 시설물의 입지점 j 의 거리 p = 시설물의 수

Decision variables:

$x_j =$ 1, 만약 노드 j 에 시설물이 설치되면,
0, 그렇지 않으면.

 $\min W$

s.t.

$$\sum_{j \in J} y_{ij} = 1, \quad i \in I$$

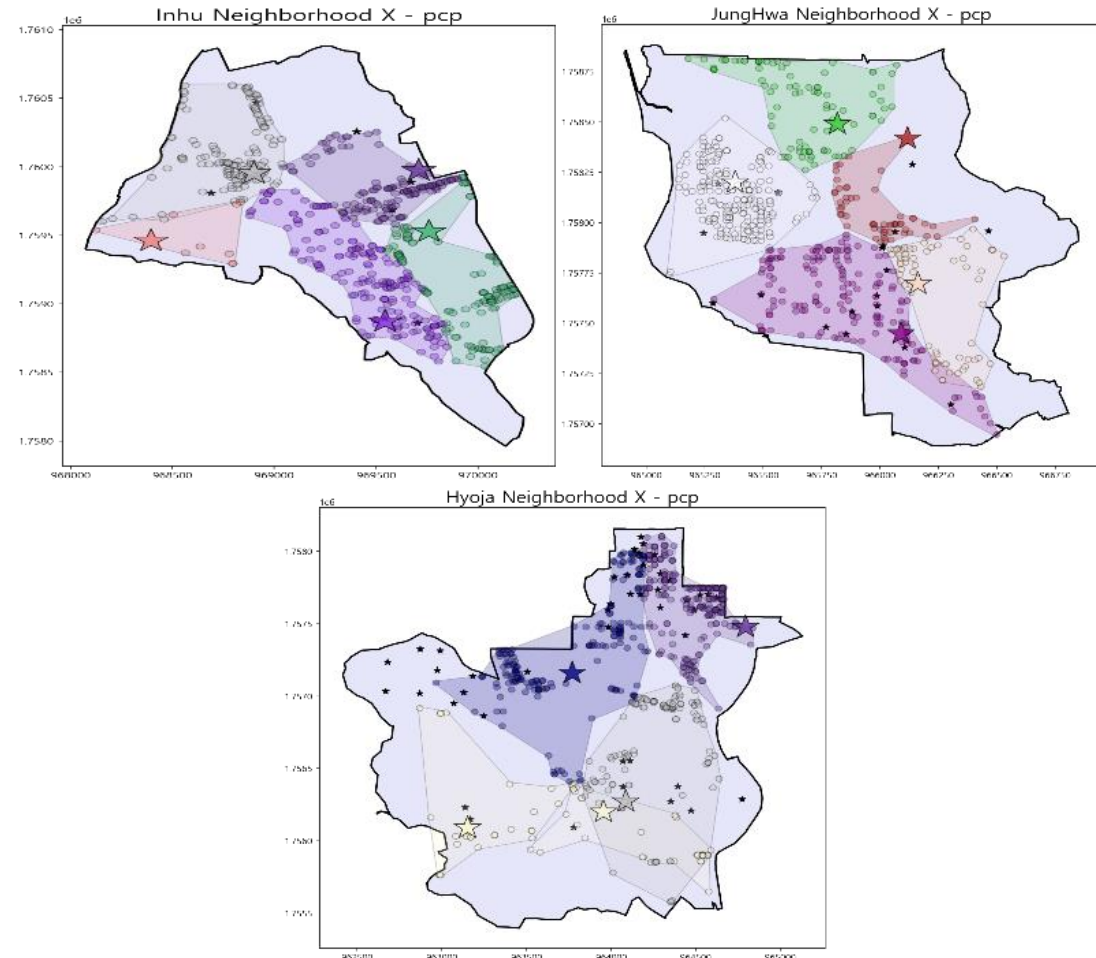
$$\sum_{j \in J} x_j = p,$$

$$y_{ij} \leq x_j, \quad i \in I, j \in J$$

$$\sum_{j \in J} d_{ij} y_{ij} \leq W, \quad i \in I$$

$$x_j \in \{0, 1\}, \quad j \in J$$

$$y_{ij} \in \{0, 1\}, \quad i \in I, j \in J$$

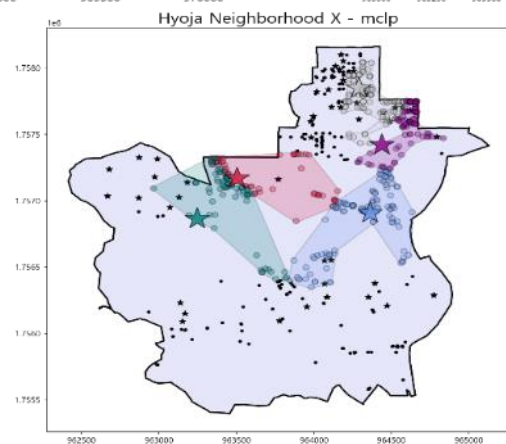
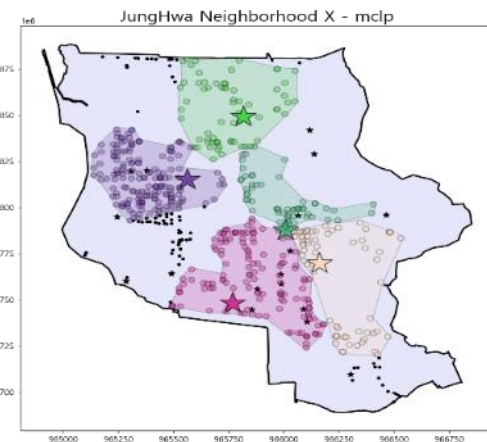
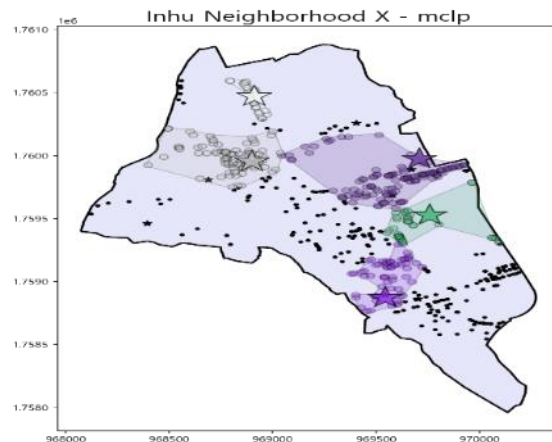


- 수요지로부터 5개의 설비 후보 지역에 가장 먼 거리의 최소값을 찾는 입지 선정
- 법정동 내 식당, 편의점, 카페 등을 수요 포인트로 주차장을 설비 후보 지역으로 선정하여 분석

MCLP (Maximal Covering Location Problem)

$$\begin{aligned}
 & \text{Maximize } \sum_{i \in I} w_i y_i \\
 & s.t. \quad y_i \leq \sum_{j \in N_i} x_j \quad \text{for all } i \in I \\
 & \quad \sum_{j \in J} x_j = K \\
 & \quad x_j, y_i \in 0, 1 \quad \text{for all } i \in I, j \in J
 \end{aligned}$$

문자	의미
i	수요 포인트 index
j	설비지역 포인트 index
I	수요 포인트 집합
J	설비지역 포인트 집합
K	설치해야하는 설비 개수
x	설비 후보 지역 중 위치 j 에 설치되면 1, 아니면 0
y	적어도 하나의 설비로 그 포인트가 커버되면 1, 아니면 0
w	입지 선정 지수=가중치



- 반경 300m 기준을 잡고 5개의 설비 후보 지역에서 최대 수요를 만족시킬 수 있는 최적 입지 선정
- 법정동 내 식당, 편의점, 카페 등을 수요 포인트로 주차장을 설비 후보 지역으로 선정하여 분석

모델 선택

수요지-입지 거리
 Min : 최소치
 Max : 최대치
 Means : 평균
 Stds : 분산

1. 인후동1가
2. 중화산동2가
3. 효자동2가

	stats	lscp	pmp	pcp	mclp
0	abs_min	0.000000	0.000000	0.000000	0.000000
1	abs_max	1246.686038	1299.523600	1219.347691	297.573683
2	mean_means	747.969143	223.690933	344.327813	69.334661
3	mean_stds	330.760099	192.637412	258.391669	56.873628
	stats	lscp	pmp	pcp	mclp
0	abs_min	0.000000	0.000000	0.000000	0.000000
1	abs_max	1197.146898	1225.255738	1151.016663	299.935282
2	mean_means	500.117026	181.158290	180.539265	115.479364
3	mean_stds	323.568070	184.301892	171.397666	71.210361
	stats	lscp	pmp	pcp	mclp
0	abs_min	0.000000	0.000000	0.000000	0.000000
1	abs_max	588.813099	1688.345740	1244.739469	295.641173
2	mean_means	200.411755	363.694810	642.582413	105.692156
3	mean_stds	126.617273	356.620215	325.503118	87.335686

- 분석 후 각 모델을 평가한 결과 MCLP모델이 다른 모델들에 비해 최적의 값이 나왔음
- 또한 제한된 조건에서 입지를 선정하는 MCLP모델이 충전소 입지 선정에 더 적합하다고 판단함

2차 군집 분석 결과

<인후동1가>

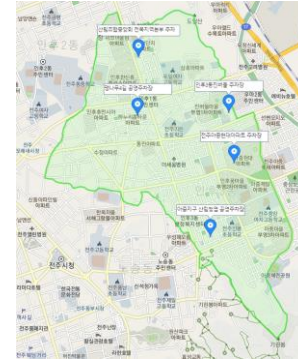
	pname	padd	pdong	pfcate	plat	plon
402	평나무4길 공영주차장	전북 전주시 덕진구 인후동1가 781-7	인후동1가	주차장	35.835600	127.155625
403	전주아중현대아파트 주차장	전북 전주시 덕진구 인후동1가 858-2	인후동1가	주차장	35.831795	127.165153
404	아중지구 산림청영 공영주차장	전북 전주시 덕진구 인후동1가 907-2	인후동1가	주차장	35.825882	127.162799
405	인후3동진버들 주차장	전북 전주시 덕진구 인후동1가 807-6	인후동1가	주차장	35.835826	127.164604
406	주차장	전북 전주시 덕진구 인후동1가 791	인후동1가	주차장	35.840265	127.155736

<중화산동2가>

	pname	padd	pdong	pfcate	plat	plon
8651	근영여고 앞 공영주차장	전북 전주시 완산구 중화산동2가 651-6	중화산동2가	주차장	35.816772	127.123729
8658	주차장	전북 전주시 완산구 중화산동2가 570-1	중화산동2가	주차장	35.822315	127.121573
8659	전주병원 주차장	전북 전주시 완산구 중화산동2가 166	중화산동2가	주차장	35.815159	127.125430
8661	공용주차장	전북 전주시 완산구 중화산동2가 644-3	중화산동2가	주차장	35.813209	127.121072
8663	주차장	전북 전주시 완산구 중화산동2가 591-5	중화산동2가	주차장	35.819228	127.118780

<효자동2가>

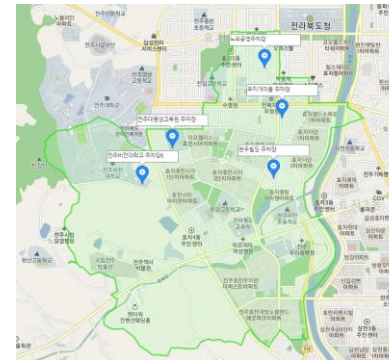
	pname	padd	pdong	pfcate	plat	plon
12663	노외공영주차장	전북 전주시 완산구 효자동2가 1237-8	효자동2가	주차장	35.816469	127.104692
12677	전주대평생교육원 주차장	전북 전주시 완산구 효자동2가 1311-1	효자동2가	주차장	35.810304	127.096009
12682	주차장	전북 전주시 완산구 효자동2가 1158-20	효자동2가	주차장	35.812614	127.106333
12688	현우빌딩 주차장	전북 전주시 완산구 효자동2가 1352	효자동2가	주차장	35.808004	127.105539
12709	전주비전대학교 주차장6	전북 전주시 완산구 효자동2가 1070	효자동2가	주차장	35.807542	127.093195



<인후동1가>



<중화산동2가>



<효자동2가>

- MCLP 분석결과 전기차 충전소 최적 입지 데이터 선정
- 선정된 데이터를 DB에 저장하여 웹을 통해 시각화

Part 5, 시각화(화면 구현)

시각화(화면 구현)

1. 프론트 페이지
2. 메인 페이지
3. 젠슬라 맵



워드 클라우드

크롤링				crawling
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
🔑 기사번호	cr_no	N/A	INTEGER	N-N
🟢 기사제목	cr_word	N/A	VARCHAR(256)	NULL
🟢 기사주소	wordcount	N/A	INTEGER	NULL

```

89 lines = text.split(/[,\.\s. ]+/g),
90 data = lines.reduce((arr, word) => {
91     let obj = Highcharts.find(arr, obj => obj.name === word);
92     if (obj)
93     {
94         obj.weight += 3000 ;
95     } else
96     {
97         obj =
98         {
99             name: word,
100            weight: 150000
101        };
102        arr.push(obj);
103    }
104    return arr;
105 }, []);
106
107 Highcharts.chart('container2', {
108     accessibility: {
109         screenReaderSection: {
110             beforeChartFormat: '<h5>{chartTitle}</h5>' +
111             '<div>{chartSubtitle}</div>' +
112             '<div>{chartLongdesc}</div>' +
113             '<div>{viewTableButton}</div>'
114         }
115     },
116     series: [{
117         type: 'wordcloud',
118         data,
119         name: 'Occurrences'
120     }],
121     title: {

```

친환경 자동차 뉴스기사 트렌드



입지 추천

최종결과 building				
번호	이름	필드 이름	도메인	데이터 타입
장소 번호	bno	N/A	INTEGER	N-N
장소 이름	bname	N/A	VARCHAR(60)	NULL
장소 주소	badd	N/A	VARCHAR(60)	NULL
동 이름	bdong	N/A	VARCHAR(30)	NULL
카테고리 대분류	bfcate	N/A	VARCHAR(30)	NULL
위도	blat	N/A	VARCHAR(30)	NULL
경도	blon	N/A	VARCHAR(30)	NULL



```

16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
// 리스트의 크기 구하기

public boolean GetBuilding(String dong)
{
    String sql = "select bno,bname,badd,bfcate,blat,blon from Building where bdong = '"+dong+"'";
    if(this.DBOpen() == false) return false;
    this.OpenQuery(sql);
    while(this.GetNext() == true)
    {
        String bno = this.GetValue("bno");
        String bname = this.GetValue("bname");
        String badd = this.GetValue("badd");
        String bfcate = this.GetValue("bfcate");
        String blat = this.GetValue("blat");
        String blon = this.GetValue("blon");

        if( this.buildinglist == null )
        {
            this.buildinglist = new ArrayList<PlaceVO>();
        }
        PlaceVO vo = new PlaceVO();
        vo.setPno(bno);
        vo.setPname(bname);
        vo.setPadd(badd);
        vo.setPfcate(bfcate);
        vo.setPplat(blat);
        vo.setPplon(blon);
        buildinglist.add(vo);
    }
    this.CloseQuery();
    this.DBClose();
    return true;
}

```



최적 입지 선정

1지역

2지역

3지역

인후동1가		중화산동2가	효자동2가
번호	이름	주소지	
1	팽나무4길 공영주차장	전북 전주시 덕진구 인후동1가 781-7	
2	전주아중현대아파트 주차장	전북 전주시 덕진구 인후동1가 858-2	
3	아중지구 산림청영 공영주차장	전북 전주시 덕진구 인후동1가 907-2	
4	인후3동진버들 주차장	전북 전주시 덕진구 인후동1가 807-6	
5	산림조합중앙회 전북지역본부 주차장	전북 전주시 덕진구 인후동1가 791	

●지도에서 해당 입지 정보 보기



ZenslaMAP

충전소입지추천










인후동1가

인구수 (19세~70세) 36652 명 | 전기차 등록수 (예측치) 89 대

시각화(화면 구현)

- 메인 페이지

히트맵, 차트

차트	chart			
논리 이름*	물리 이름*	도메인	데이터 타입	널 허용
 지역번호	ano	N/A	BIGINT	N-N
 시군구명	goo	N/A	VARCHAR(256)	NULL
 읍면동명	dong	N/A	VARCHAR(256)	NULL
 인구수	people	N/A	INTEGER	NULL
 충전소 개수	charger	N/A	INTEGER	NULL
 전기차 대수	car	N/A	INTEGER	NULL
 주소	addr	N/A	VARCHAR(256)	NULL
 전기차 예측값	pcar	N/A	INTEGER	NULL
 급속 충전기	qcharger	N/A	INTEGER	NULL



●선택된 카테고리
에 따라서 새로 정렬



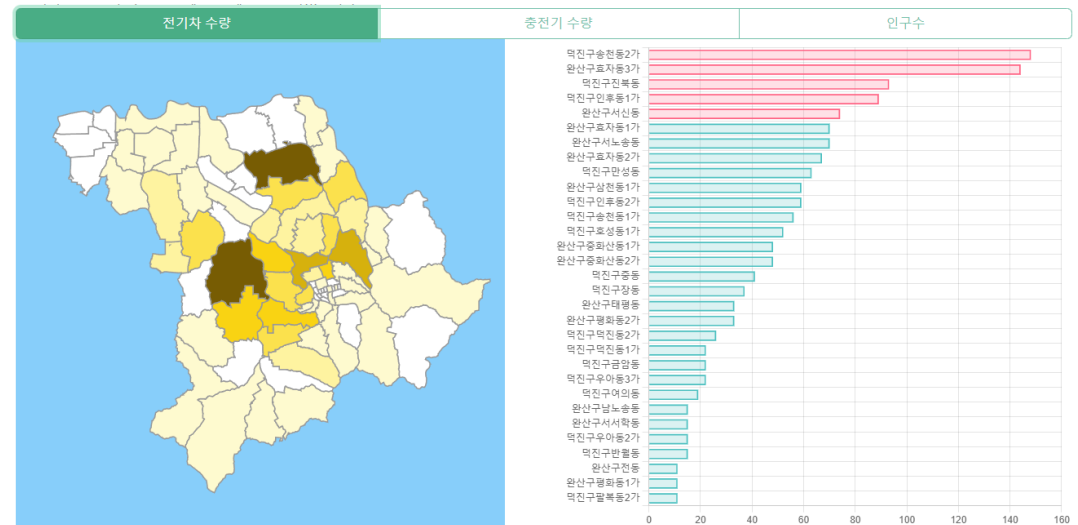
```
137 public void ThisSort(String code)
138 {
139     switch(code)
140     {
141         case "a1" :
142             list2 = this.list.stream().sorted(Comparator.comparingInt(ChartVO::getCar).reversed()).collect(Collectors.toList());
143             break;
144         case "a2" :
145             list2 = this.list.stream().sorted(Comparator.comparingInt(ChartVO::getCharger).reversed()).collect(Collectors.toList());
146             break;
147         case "a3" :
148             list2 = this.list.stream().sorted(Comparator.comparingInt(ChartVO::getPeople).reversed()).collect(Collectors.toList());
149             break;
150     }
151 }
```



```

20: public boolean getChart()
21: {
22:     String sql = "select ano,goo,dong,addr,people,charger,pcar from chart";
23:     if(this.OpenConn() == false) return false;
24:     this.OpenQuery(sql);
25:     ArrayList<Integer> carlist = new ArrayList<Integer>();
26:     ArrayList<Integer> chargerlist = new ArrayList<Integer>();
27:     ArrayList<Integer> peoplelist = new ArrayList<Integer>();
28:     while(this.GetNext() == true)
29:     {
30:         String ano = this.GetValue("ano");
31:         String goo = this.GetValue("goo");
32:         String dong = this.GetValue("dong");
33:         String addr = this.GetValue("addr");
34:         int people = this.GetInt("people");
35:         int charger = this.GetInt("charger");
36:         int car = this.GetInt("pcar");
37:
38:         if (this.list == null )
39:         {
40:             this.list = new ArrayList<ChartVO>();
41:
42:             ChartVO vo = new ChartVO();
43:             vo.setAno(ano);
44:             vo.setGoo(goo);
45:             vo.setDong(dong);
46:             vo.setAddr(addr);
47:             vo.setPeople(people);
48:             vo.setCharger(charger);
49:             vo.setCar(car);
50:             list.add(vo);
51:             carlist.add(car);
52:             chargerlist.add(charger);
53:             peoplelist.add(people);
54:         }
55:         int maxcar = carlist.get(0);
56:         int mincar = carlist.get(0);
57:         int maxcharger = chargerlist.get(0);
58:         int mincharger = chargerlist.get(0);
59:         int maxpeople = peoplelist.get(0);
60:         int minpeople = peoplelist.get(0);
61:         for(int i=0 ; i < carlist.size(); i++)

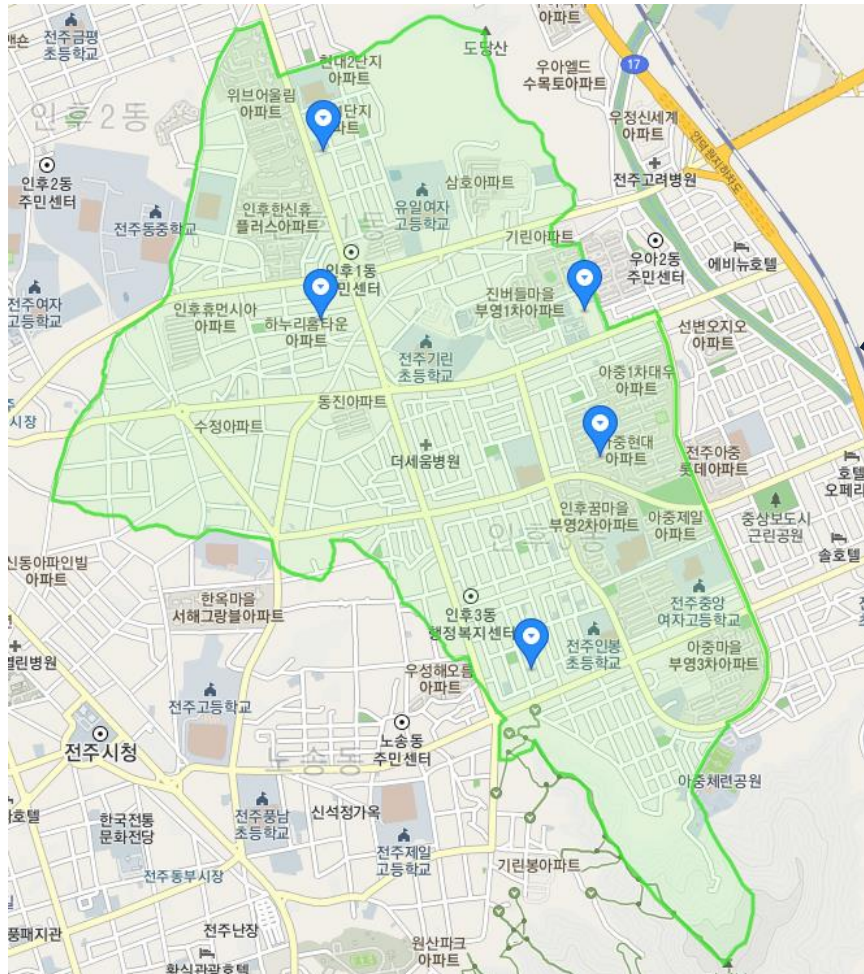
```



시각화(화면 구현)

- 젠슬라 맵

추천 지역, 입지 보기



● d3-geo json
지역정보로부터
폴리곤 생성

```
{
  "type": "FeatureCollection",
  "name": "JeonjuBound",
  "crs": { "type": "name", "properties": { "name": "urn:ogc:def:crs:OGC:1.3:CRS84" } },
  "features": [
    { "type": "Feature", "properties": { "fid": 1, "END_CD": "45111115", "END_NM": "다기동3가", "SGG_OID": 10241, "COL_ADM_SE": "45110", "GID": 929 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111115, 35.888889, 127.111115, 35.888889, 127.111115, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 2, "END_CD": "45111135", "END_NM": "종관동", "SGG_OID": 10239, "COL_ADM_SE": "45110", "GID": 930 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111135, 35.888889, 127.111135, 35.888889, 127.111135, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 3, "END_CD": "45111106", "END_NM": "종관동2가", "SGG_OID": 10238, "COL_ADM_SE": "45110", "GID": 931 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111106, 35.888889, 127.111106, 35.888889, 127.111106, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 4, "END_CD": "45111145", "END_NM": "성동", "SGG_OID": 15988, "COL_ADM_SE": "45110", "GID": 875 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111145, 35.888889, 127.111145, 35.888889, 127.111145, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 5, "END_CD": "45111139", "END_NM": "성안동", "SGG_OID": 15987, "COL_ADM_SE": "45110", "GID": 876 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111139, 35.888889, 127.111139, 35.888889, 127.111139, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 6, "END_CD": "45111120", "END_NM": "성동", "SGG_OID": 15346, "COL_ADM_SE": "45110", "GID": 877 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111120, 35.888889, 127.111120, 35.888889, 127.111120, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 7, "END_CD": "45111126", "END_NM": "성동", "SGG_OID": 12145, "COL_ADM_SE": "45110", "GID": 878 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111126, 35.888889, 127.111126, 35.888889, 127.111126, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 8, "END_CD": "45111116", "END_NM": "성동", "SGG_OID": 12466, "COL_ADM_SE": "45110", "GID": 879 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111116, 35.888889, 127.111116, 35.888889, 127.111116, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 9, "END_CD": "45111115", "END_NM": "성동", "SGG_OID": 12465, "COL_ADM_SE": "45110", "GID": 880 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111115, 35.888889, 127.111115, 35.888889, 127.111115, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 10, "END_CD": "45111120", "END_NM": "성동", "SGG_OID": 13425, "COL_ADM_SE": "45110", "GID": 881 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111120, 35.888889, 127.111120, 35.888889, 127.111120, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 11, "END_CD": "45111133", "END_NM": "성동", "SGG_OID": 13188, "COL_ADM_SE": "45110", "GID": 882 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111133, 35.888889, 127.111133, 35.888889, 127.111133, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 12, "END_CD": "45111131", "END_NM": "성동", "SGG_OID": 11513, "COL_ADM_SE": "45110", "GID": 883 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111131, 35.888889, 127.111131, 35.888889, 127.111131, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 13, "END_CD": "45111134", "END_NM": "성동", "SGG_OID": 13745, "COL_ADM_SE": "45110", "GID": 884 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111134, 35.888889, 127.111134, 35.888889, 127.111134, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 14, "END_CD": "45111123", "END_NM": "성동", "SGG_OID": 11827, "COL_ADM_SE": "45110", "GID": 885 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111123, 35.888889, 127.111123, 35.888889, 127.111123, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 15, "END_CD": "45111128", "END_NM": "성동", "SGG_OID": 15668, "COL_ADM_SE": "45110", "GID": 886 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111128, 35.888889, 127.111128, 35.888889, 127.111128, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 16, "END_CD": "45111124", "END_NM": "성동", "SGG_OID": 10295, "COL_ADM_SE": "45110", "GID": 888 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111124, 35.888889, 127.111124, 35.888889, 127.111124, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 17, "END_CD": "45111129", "END_NM": "성동", "SGG_OID": 10293, "COL_ADM_SE": "45110", "GID": 889 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111129, 35.888889, 127.111129, 35.888889, 127.111129, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 18, "END_CD": "45111107", "END_NM": "성동", "SGG_OID": 10289, "COL_ADM_SE": "45110", "GID": 890 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111107, 35.888889, 127.111107, 35.888889, 127.111107, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 19, "END_CD": "45111118", "END_NM": "성동", "SGG_OID": 10288, "COL_ADM_SE": "45110", "GID": 891 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111118, 35.888889, 127.111118, 35.888889, 127.111118, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 20, "END_CD": "45111133", "END_NM": "성동", "SGG_OID": 10287, "COL_ADM_SE": "45110", "GID": 892 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111133, 35.888889, 127.111133, 35.888889, 127.111133, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 21, "END_CD": "45111132", "END_NM": "성동", "SGG_OID": 10285, "COL_ADM_SE": "45110", "GID": 893 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111132, 35.888889, 127.111132, 35.888889, 127.111132, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 22, "END_CD": "45111107", "END_NM": "성동", "SGG_OID": 10277, "COL_ADM_SE": "45110", "GID": 896 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111107, 35.888889, 127.111107, 35.888889, 127.111107, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 23, "END_CD": "45111121", "END_NM": "성동", "SGG_OID": 10275, "COL_ADM_SE": "45110", "GID": 897 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111121, 35.888889, 127.111121, 35.888889, 127.111121, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 24, "END_CD": "45111114", "END_NM": "성동", "SGG_OID": 10272, "COL_ADM_SE": "45110", "GID": 898 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111114, 35.888889, 127.111114, 35.888889, 127.111114, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 25, "END_CD": "45111102", "END_NM": "성동", "SGG_OID": 10267, "COL_ADM_SE": "45110", "GID": 899 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111102, 35.888889, 127.111102, 35.888889, 127.111102, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 26, "END_CD": "45111136", "END_NM": "성동", "SGG_OID": 10271, "COL_ADM_SE": "45110", "GID": 900 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111136, 35.888889, 127.111136, 35.888889, 127.111136, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 27, "END_CD": "45111126", "END_NM": "성동", "SGG_OID": 10270, "COL_ADM_SE": "45110", "GID": 901 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111126, 35.888889, 127.111126, 35.888889, 127.111126, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 28, "END_CD": "45111123", "END_NM": "성동", "SGG_OID": 10265, "COL_ADM_SE": "45110", "GID": 902 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111123, 35.888889, 127.111123, 35.888889, 127.111123, 35.888889 ] ] ] ] } },
    { "type": "Feature", "properties": { "fid": 29, "END_CD": "45111125", "END_NM": "성동", "SGG_OID": 10264, "COL_ADM_SE": "45110", "GID": 903 }, "geometry": { "type": "Polygon", "coordinates": [ [ [ [ 127.111125, 35.888889, 127.111125, 35.888889, 127.111125, 35.888889 ] ] ] ] } } ] ] }
```

```
public boolean GetBuilding(String dong)
{
    String sql = "select bno,bname,badd,bfcate,blat,blon from Building where bdong = '"+dong+"'";
    if(this.DBOpen() == false) return false;
    this.OpenQuery(sql);
    while(this.GetNext() == true)
    {
        String bno = this.GetValue("bno");
        String bname = this.GetValue("bname");
        String badd = this.GetValue("badd");
        String bfcate = this.GetValue("bfcate");
        String blat = this.GetValue("blat");
        String blon = this.GetValue("blon");

        if( this.buildinglist == null )
        {
            this.buildinglist = new ArrayList<PlaceVO>();
        }
        PlaceVO vo = new PlaceVO();
        vo.setPno(bno);
        vo.setPname(bname);
        vo.setPadd(badd);
        vo.setPfcate(bfcate);
        vo.setPlat(blat);
        vo.setPlon(blon);
        buildinglist.add(vo);
    }
    this.CloseQuery();
    this.DBclose();
    return true;
}
```

최종결과		building		
장소 번호	bno	N/A	INTEGER	N-N
장소 이름	bname	N/A	VARCHAR(60)	NULL
장소 주소	badd	N/A	VARCHAR(60)	NULL
동 이름	bdong	N/A	VARCHAR(30)	NULL
카테고리 대분류	bfcate	N/A	VARCHAR(30)	NULL
위도	blat	N/A	VARCHAR(30)	NULL
경도	blon	N/A	VARCHAR(30)	NULL

시각화(화면 구현)

- 젠슬라 맵

주변시설 조회

장소	place
● 장소 이름	출력 이름
● 장소 번호	pno
● 장소 이름	pname
● 장소 주소	padd
● 도로명 주소	pnewadd
● 동 이름	pdong
● img url 주소	pimg
● 카테고리 대분류	pfcate
● 카테고리 소분류	plcate
● 카테고리1	pcated1
● 카테고리2	pcated2
● 카테고리3	pcated3
● 카테고리4	pcated4
● 카테고리5	pcated5
● 전화번호	ptel
● 위도	plat
● 경도	plon
● utmk_x	utm_k_x
● utmk_y	utm_k_y



```

63 public boolean GetCate(String cate, String area)
64 {
65     String sql = "select pno,pname,padd,pfcate,plat,plon from place where pfcate = '"+cate+"' and pdong = '"+area+"' ";
66     if(this.DBOpen() == false) return false;
67     this.OpenQuery(sql);
68     while(this.GetNext() == true)
69     {
70         String pno = this.GetValue("pno");
71         String pname = this.GetValue("pname");
72         String padd = this.GetValue("padd");
73         String pfcate = this.GetValue("pfcate");
74         String plat = this.GetValue("plat");
75         String plon = this.GetValue("plon");
76
77         if( this.cateList == null )
78         {
79             this.cateList = new ArrayList<PlaceVO>();
80         }
81         PlaceVO vo = new PlaceVO();
82         vo.setPno(pno);
83         vo.setPname(pname);
84         vo.setPadd(padd);
85         vo.setPfcate(pfcate);
86         vo.setPlat(plat);
87         vo.setPlon(plon);
88         cateList.add(vo);
89     }
90     this.CloseQuery();
91     return true;
92 }
93

```



ZenslaMAP

중전소입지추천

인후동1가
지역입지순위 1 순위 | 지역입지점수 21 점

주변 탐색

음식점 카페 편의점 공공기관

다른지역 >

- 인후동1가
- 중화산동2가
- 효자동2가

추천입지장소 5

- 1 팽나무4길 공영주차장
전북 전주시 묵진구 인후동1가 781-7
- 2 전주아중현대아파트 주차장
전북 전주시 묵진구 인후동1가 858-2
- 3 아중지구 산림청외 공영주차장
전북 전주시 묵진구 인후동1가 907-2
- 4 인후3동진버를 주차장
전북 전주시 묵진구 인후동1가 807-6
- 5 산림조합중앙회 전북지역본부 주차장
전북 전주시 묵진구 인후동1가 791

감사합니다.