빅데이터기반경영/사업타당성분석

- 공유주방 입지 선정

소속: 홍익대학교 산업공학과/경영학과/영어영문학과

수업: 빅데이터기반경영/사업타당성분석

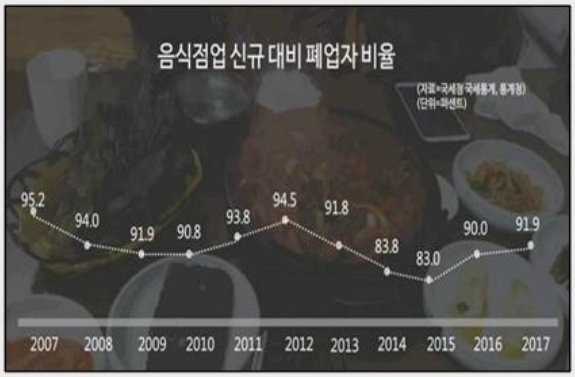
담당 교수 명: 전홍배 교수님 김승범 교수님

학번/이름: B411008 고재승
B884012 김별희
B531072 김정호
B671039 임수민

목차

1. 주제 선정 배경

<음식점업 신규 대비 폐업자 비율>



<음식점업 신규 창업자 대표 유형>

60-70세 은퇴 후 자영업자

- 베이비부머 시대 은퇴자↑
- 진입장벽 낮은 음식점업, 하지만 경험, 준비 부족 등 폐업할 확률↑
- 노인빈곤율 가속화

30대 이하 젊은 층

- 1년 미만에 폐업하는 비율은 12.5%.(4-50대에 비해 2-3배 높음)
- 다른 연령대에 비해 경험이 적고, 노하우나 준비기간이 없기 때문

[그림] 1

우리나라의 소상공인 경제 중 고질적인 문제 중 하나로, 음식점업의 높은 창업 대비 폐업률이 존재한다. 음식점업은 2007년부터 2017년까지 최소 83%에서 최대 95.2%를 기록하고 있다. 그렇다면, 어떤 유형이 음식점업에 뛰어들어 주로 실패하는 지 조사한 결과, 크게 두 유형으로, 6-70세 은퇴자들과 30세 이하 젊은 층이었다. 음식점업의 특성 상 진입장벽이 상대적으로 낮은 편이기에 그만큼 유입도 많지만, 경험과 준비 부족으로 금방 폐업하는 상황이다.

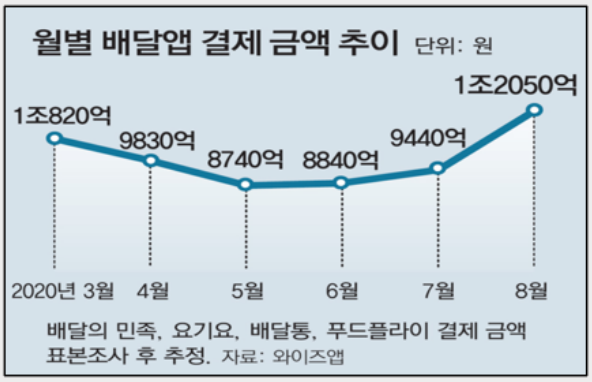
<20년도 분기별 상가 현황>

<20년도 업종별 폐업 현황>

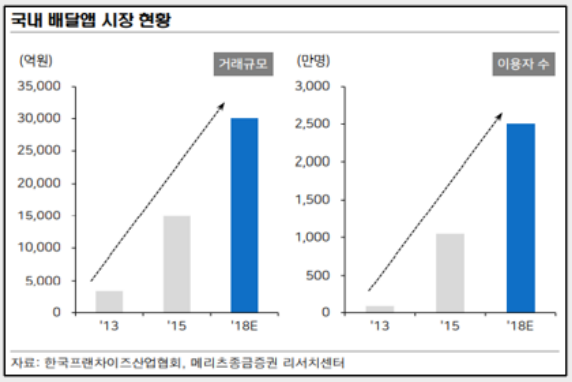
[그림] 2

이러한 상황은 올해 코로나19 사태 이후 더욱 심각해지고 있다. 20년도 분기별 상가 현황을 보면, 1분기 대비 2분기에 2만 개 이상의 상가가 사라졌으며, 특히 음식업종에서 상가 감소율이 7.5%로, 타업종과 비교해도 감소세가 두드러진다는 것을 알 수 있다. 또한 20년도 업종별 폐업 현황에 따르면, 전년 대비 평균 생존기간이 6개월 감소하였고, 폐업 상위 10개 업종 중 1위부터 5위에 통신판매업을 제외한 4개 업종이 모두 음식업종임을 확인할 수 있다

<20년도 월별 배달앱 결제 금액 추이>



<국내 배달앱 시장 현황>



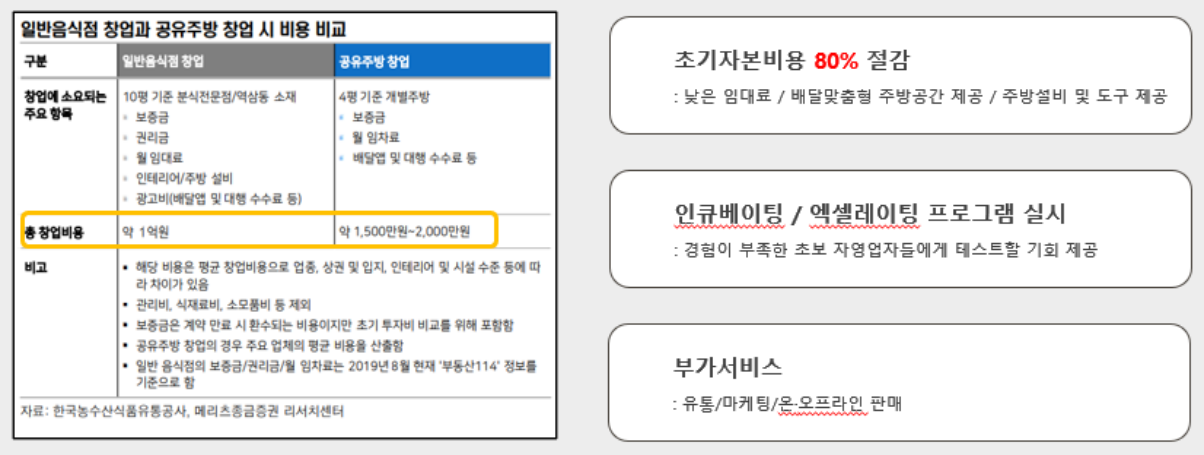
[그림] 3

이처럼 수많은 소상공인들, 그 중에서도 음식업종에 종사하는 분들이 크게 힘들어진 반면에, 오히려 음식배달시장은 급성장하고 있다. 뿐만 아니라, 음식배달시장의 성장은 코로나19의 영향 때문만이 아닌, 전부터 꾸준히 성장해오고 있던 만큼, 음식배달업은 앞으로도 더욱 성장할 것으로 보인다. 이를 바탕으로, 현재와 같은 경제난과 음식점업에 뛰어들 소상공인들에게 힘이 될 수 있도록 공유주방이라는 사업을 제안하고자 한다.

<개방형 공유주방의 모습>

[그림] 4

공유주방이란 단어에서 금방 알 수 있듯이, 개인 사업자들이 하나의 주방을 공유하는 형태를 말하며, 위 사진과 같은 모습이다.



일반음식점 창업과 공유주방 창업 시 비용 비교

구분	일반음식점 창업	공유주방 창업
창업에 소요되는 주요 항목	10평 기준 분식전문점/역삼동 소재 • 보증금 • 권리금 • 월 임대료 • 인테리어/주방 설비 • 광고비(배달앱 및 대행 수수료 등)	4평 기준 개별주방 • 보증금 • 월 임차료 • 배달앱 및 대행 수수료 등
총 창업비용	약 1억원	약 1,500만원~2,000만원
비고	• 해당 비용은 평균 창업비용으로 업종, 상권 및 입지, 인테리어 및 시설 수준 등에 따라 차이가 있음 • 관리비, 식재료비, 소모품비 등 제외 • 보증금은 계약 만료 시 환수되는 비용이지만 초기 투자비 비교를 위해 포함함 • 공유주방 창업의 경우 주요 업체의 평균 비용을 산출함 • 일반 음식점의 보증금/권리금/월 임차료는 2019년 8월 현재 '부동산114' 정보를 기준으로 함	

자료: 한국농수산식품유통공사, 메리츠종금증권 리서치센터

초기자본비용 80% 절감
: 낮은 임대료 / 배달맞춤형 주방공간 제공 / 주방설비 및 도구 제공

인큐베이팅 / 엑셀레이팅 프로그램 실시
: 경험이 부족한 초보 자영업자들에게 테스트할 기회 제공

부가서비스
: 유통/마케팅/온·오프라인 판매

[그림] 5

공유주방의 장점 및 기능을 일반음식점 창업과 공유주방 창업을 비교하며 살펴보면, 총 창업비용이 일반음식점은 1억원, 공유주방은 1500만원에서 2000만원 사이로, 초기자본비용이 80% 절감된다. 또한 단순히 공간 임대 뿐 아니라 경험이 부족한 초보 자영업자들을 위해 인큐베이팅 및 엑셀레이팅 프로그램을 실시하고 있다. 그 외에도 유통, 마케팅, 온, 오프라인 판매에 걸쳐 전반적인 운영과 영업을 지원해준다.

2. 문제정의

[그림] 6

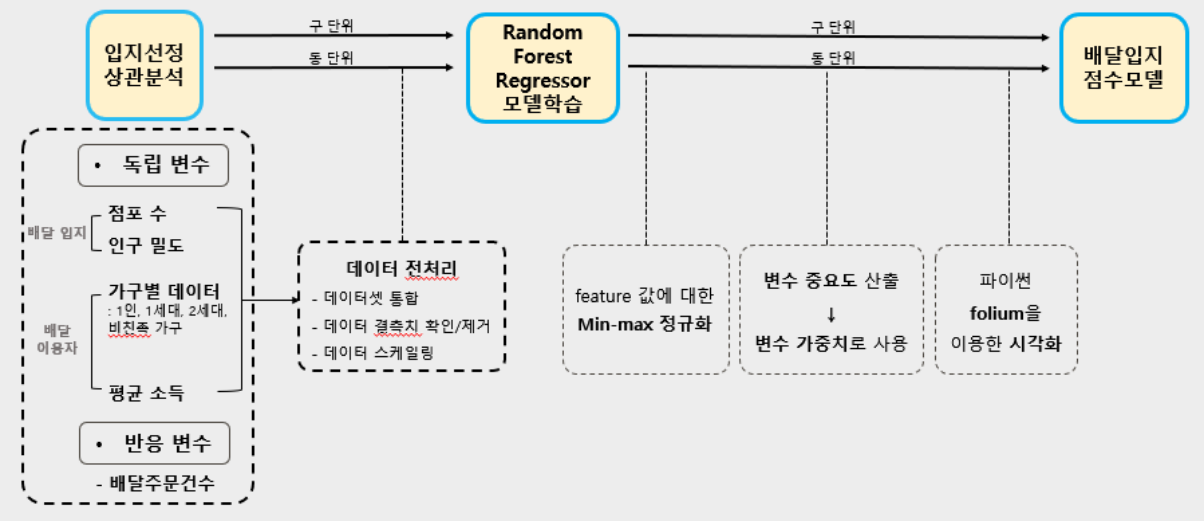
이러한 공유주방을 소상공인들에게 공공서비스로 제안하고자 하여, 정부지원형 공유주방에 대해 사전조사를 진행하였다. 실제 모습과 사업내용이 민간업체들이 운영하는 방식과 달라, 문제점이 있었다. 먼저, 하나의 주방에 하나의 입점 기업만 기수별로 3개월 사용 가능하다는 점에서 사실상 공유주방이 갖는 '효율적인 공간'을 활용하지 못하고 있다. 그리고 교육에 치중된 사업내용은 실제 사업의 장이라기 보다 창업 아카데미에 가까워 보였고, 그만큼 배달 서비스를 고려하지 않았기에 그 공간과 위치는 배달에 유리한 공유주방의 장점을 살리지 못하고 있었다. 그 외의 공유주방을 표방한 사례로 마을부엌 프로젝트, 공간이음 등이 있었지만, 이들은 마을 공동체 복지 차원의 부대시설 수준에 그친 것으로 확인됐다.

앞서 살펴본 문제점들 중 입지에 관련한 문제점을 본 프로젝트의 주제로 삼아 해결하고자 하였다.

공유주방이라는 사업을 배달에 적합한 최적의 입지에 위치시켜, 어려움을 겪는 소상공인들에게 공공서비스의 형태로 제공하고자 한다.

3. 데이터 분석

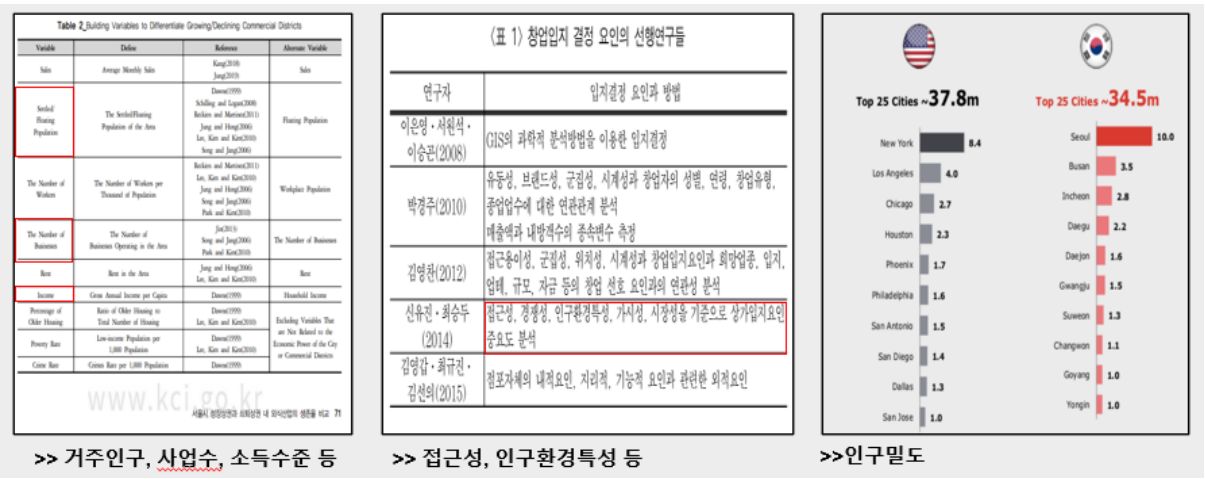
데이터 분석 개요



[그림] 7

개요는 다음과 같다. 배달주문건수 데이터의 경우 행정동 데이터의 결측치가 많은 문제로, 배달주문건수 행정구 데이터를 이용해 배달주문건수에 영향을 미치는 독립변수들을 파악하고자 하였다. '구' 단위의 데이터를 이용하여 최적의 행정구를 선정하고, 행정구내 '동' 단위의 데이터를 이용하여 최적의 행정동까지 선정하는 것이 프로젝트의 핵심이다. 독립 변수와 반응 변수를 위와 같이 설정하였다. 독립변수는 크게 두 분류로 배달입지 관련하여 점포 수, 인구 밀도를 / 배달 이용자 관련하여 가구별 데이터, 평균 소득을 변수로 설정하였다. 그리고 이에 대한 반응 변수는 배달주문건수로 설정하였다. 상관분석을 시작으로, 데이터 전처리 후, 각 변수들에 Random Forest Regressor모델학습을 적용하였다. 그 후, featur값에 대한 min-max정규화를 진행하여 변수 중요도를 산출하고, 그 중요도를 변수 가중치로 사용한 후, 파이썬 folium을 이용하여 서울시 지도에 시각화를 진행하여 최종적으로 배달상권 점수 모델을 만들었다.

행정구 변수 선정



>> 거주인구, 사업수, 소득수준 등 >> 접근성, 인구환경특성 등 >>인구밀도

[그림] 8

변수 선정 시에 다음과 같이 상권분석이나 입지 선정과 관련된 논문과 보고서 자료를 참고하여 선정하였다. 변수에 인구통계적 변수와 소득수준 등 관련 논문에서 공통적으로 포함시킨 것을 바탕으로 선정하였으며, 오른쪽 인구밀도의 경우는 독일 배달 앱 회사 '딜리버리 히어로'가 '요기요'와 '배달통'을 운영하는 회사가 한국 배달시장의 가치를 인구밀도를 통해 설명한 것을 토대로 선정하였다.

행정구 데이터 전처리

	A	B	C	D
1	업종	시	행정구	건수
2	도시락	서울특별시	구로구	12
3	도시락	서울특별시	도봉구	3
4	돈까스/일	서울특별시	구로구	11
5	돈까스/일	서울특별시	도봉구	4
6	돈까스/일	서울특별시	영등포구	5
7	돈까스/일	제주특별자	서귀포시	5
8	돈까스/일	충청북도	제천시	14
9	분식	서울특별시	구로구	244
10	분식	서울특별시	도봉구	54
11	분식	서울특별시	영등포구	40
12	아시안/양	서울특별시	구로구	3
13	아시안/양	서울특별시	영등포구	107
14	야식	서울특별시	강서구	1
15	야식	서울특별시	구로구	40
16	야식	서울특별시	도봉구	2
17	야식	서울특별시	영등포구	3
18	족발/보쌈	서울특별시	강서구	5
19	족발/보쌈	서울특별시	구로구	81
20	족발/보쌈	서울특별시	도봉구	33
21	족발/보쌈	서울특별시	서대문구	4
22	족발/보쌈	서울특별시	영등포구	33

C	D	E	F	G	H
20세미만(1	20~24세(1	25~29세(1	30~34세(1	35~39세(1	40~44세(1
507	3280	3903	2647	1872	1405
489	2251	3054	2561	1920	1462
252	2636	5349	4985	3959	2701
587	4709	6878	4907	3655	2752
576	6710	11939	8398	5649	3767
1543	10659	9771	5351	3895	3031
201	2044	5604	5126	4408	3431
1355	10608	9391	5161	3810	2938
295	2816	4321	3369	3028	2572
159	1509	2452	2299	2371	2070
724	5509	4848	3449	3391	2985
251	2539	6610	5708	4901	3679
744	8292	8485	4713	3337	2480
554	6608	12239	9331	6562	4275
142	1182	3314	3489	3174	2653
325	4108	13600	11648	8271	5633
232	2609	7184	5957	4424	3317
138	1881	6120	4790	3370	2326
251	3277	11754	9638	5907	3781
891	7837	13256	8329	5228	3396
1023	14851	33127	19915	10984	6510

A	B	C	D	E	F	G	H
코드	시	행정구	가구특성	세부가구특	주거특성	가구	가구수
11110	서울특별시	종로구	1세대 가구	부부	주택_단독	가구	3409
11110	서울특별시	종로구	1세대 가구	부부	주택_아파	가구	1873
11110	서울특별시	종로구	1세대 가구	부부	주택_연립	가구	784
11110	서울특별시	종로구	1세대 가구	부부	주택_다세	가구	1290
11110	서울특별시	종로구	1세대 가구	부부	주택_비거	가구	249
11110	서울특별시	종로구	1세대 가구	부부	주택이외의	가구	557
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택_단독	가구	34
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택_아파	가구	11
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택_연립	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택_다세	가구	14
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택_비거	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+미혼	주택이외의	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+기타	주택_단독	가구	9
11110	서울특별시	종로구	1세대 가구	부부+기타	주택_아파	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+기타	주택_연립	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+기타	주택_다세	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+기타	주택_비거	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	부부+기타	주택이외의	가구	5가구 미만
11110	서울특별시	종로구	1세대 가구	가구주+미	주택_단독	가구	466
11110	서울특별시	종로구	1세대 가구	가구주+미	주택_아파	가구	175
11110	서울특별시	종로구	1세대 가구	가구주+미	주택_연립	가구	72

[그림] 9

<서울시 행정구 단위 최종 데이터셋>

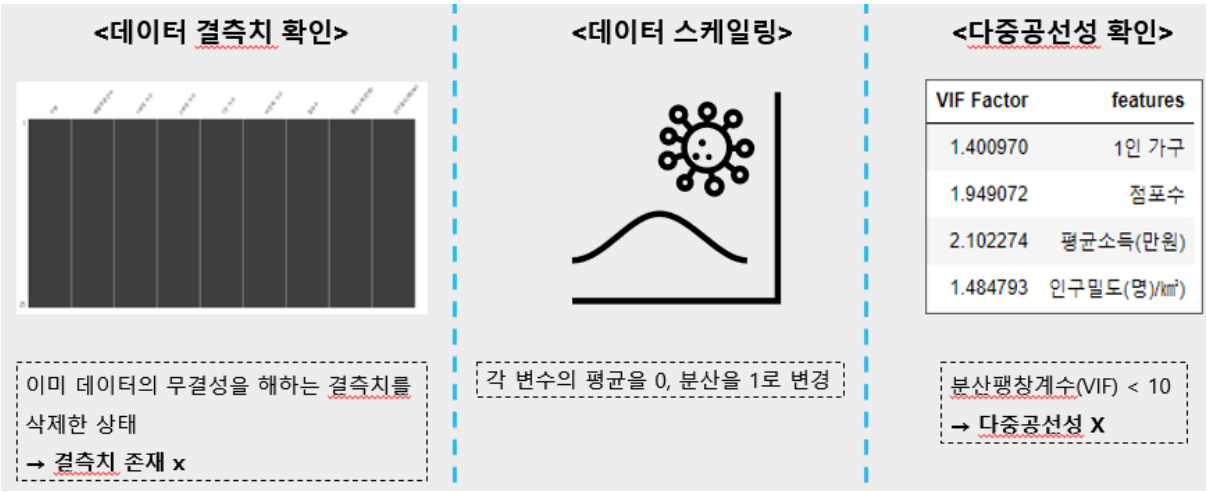
	A	B	C	D	E	F	G	H	I
1	구분	배달주문건수	1세대 가구	2세대 가구	1인 가구	비친족 가구	점포수	평균소득(만원)	인구밀도(명)/㎢)
2	종로구	43495	30072	68811	70933	4853	7061	403	6869
3	중구	68097	26031	56057	60511	4235	6683	407	13514
4	용산구	63386	46317	103180	99247	7244	4951	346	11179
5	성동구	43067	60818	154481	117220	6254	3818	332	18551
6	광진구	28868	66141	179513	164617	7244	5178	323	21819
7	동대문구	55358	65493	170807	162109	7604	5262	300	25748
8	중랑구	49208	79382	216091	142648	5318	4481	268	22318
9	성북구	56914	80282	230141	162748	8072	4900	287	18533
10	강북구	56503	61773	169562	117265	4790	4074	279	13898
11	도봉구	26885	66408	189329	88426	4379	3095	299	16752
12	노원구	62386	93322	317652	147814	7151	4713	298	15748
13	은평구	60948	92954	255887	139624	8423	4658	281	16534
14	서대문구	66939	59866	155717	130273	7223	4643	337	18440
15	마포구	56464	74703	182104	162505	10943	8878	330	16174
16	양천구	30060	76694	269636	99283	5195	4184	323	27289

변수

행정구 단위 데이터

[그림] 10

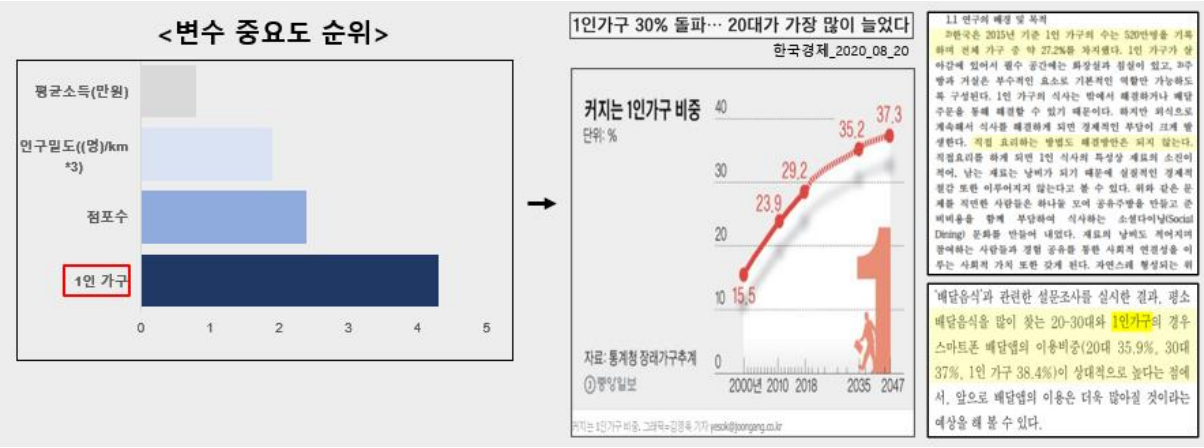
사용된 데이터는 배달주문건수, 1세대가구, 2세대 가구, 1인가구, 비친족 가구, 점포수, 평균소득(만원), 인구밀도(명/km^2)이다.



[그림] 11

데이터를 수집, 통합하는 과정에서 데이터의 무결성을 해하는 결측치를 삭제한 상태이기 때문에 최종 데이터 셋의 결측치를 확인하고자 시각화 하였을 때, 결측치가 존재하지 않는 것을 확인할 수 있었다. 데이터 마다 값의 단위, 크기가 모두 다르기 때문에 데이터의 값이 너무 크거나 작은 경우 모델 학습 알고리즘 과정에서 0으로 수렴하거나 무한대로 발산할 위험이 있기 때문에 그 다음으로 데이터 스케일링을 진행하였다. 파이썬의 sklearn을 이용하여 StandardSclaer을 통해 각 feature의 평균을 0, 분산을 1로 변경해주었다. 데이터 사용 목적은 배달주문건수를 정확히 예측하고자 하는 것이 아닌 목표변수인 배달주문건수와 독립변수들 간의 관계를 설명하고 독립변수가 목표변수에 미치는 영향력의 크기를 측정하는 것이다. 따라서 다중공선성 문제의 위험을 제거하기 위해 분산팽창계수(VIF)를 사용하였다. VIF >10 이면 다중공선성의 문제가 있을 수 있기에, 1세대가구, 2세대가구, 비친족가구의 경우 VIF를 이용해 처리하는 과정에서 모두 10을 초과하는 값이 나와 제거하였다.

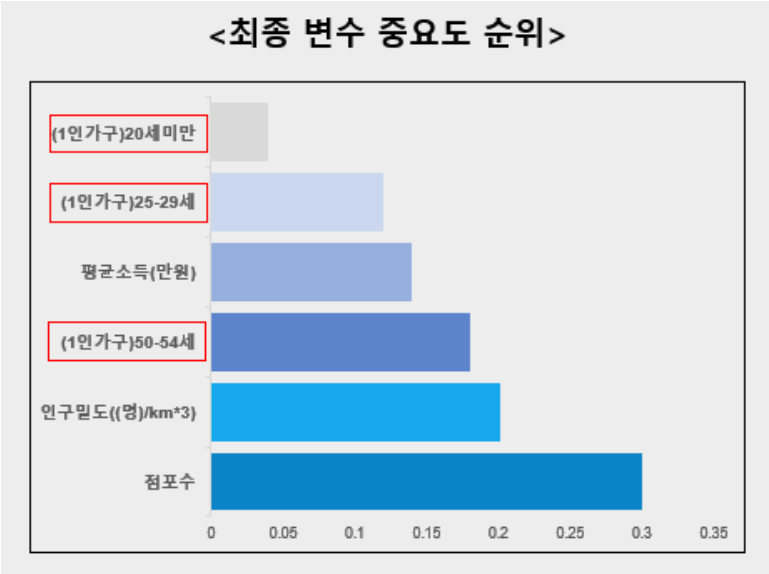
행정구 변수 중요도 측정



[그림] 12

목표변수를 배달주문건수로 설정하고 독립변수들로 1인가구, 점포수, 평균소득, 인구밀도로 설정하여 목표변수와 독립변수들 사이의 인과관계를 파악하기 위해 회귀분석을 진행하였다. 목표변수에 영향을 끼치는 독립변수들의 중요도를 이용해 구 선정에 있어 가중치를 부여하는 것이 목적이기에 RandomForestRegressor을 이용하여 학습을 진행하였다. 학습을 진행하고, 변수 중요도를 산출한 결과 1인가구, 점포수, 인구밀도, 평균소득 순으로 중요도가 나타났다. 논문을 참고한 결과 1인 가구 변수의 중요도가 높기에 1인가구 연령별 데이터를 추가하였다.

행정구 최종 변수 중요도 순위



[그림] 13

1인가구 연령별 데이터를 포함한 데이터 셋을 다시 스케일링 과정과 분산팽창계수를 이용해 다중공선성의 위험이 의심되는 feature들을 제거하였더니 최종적으로 남은 feature는 위와 같다. 앞선 과정처럼 RandomForestRegressor 모델을 이용해 학습한 후, 변수 중요도를 산출한 결과 점포수, 인구밀도, (1인가구)50~54세, 평균소득(만원), (1인가구)25~29세, (1인가구)20세 미만 순으로 나타났습니다.

행정구 변수 가중치

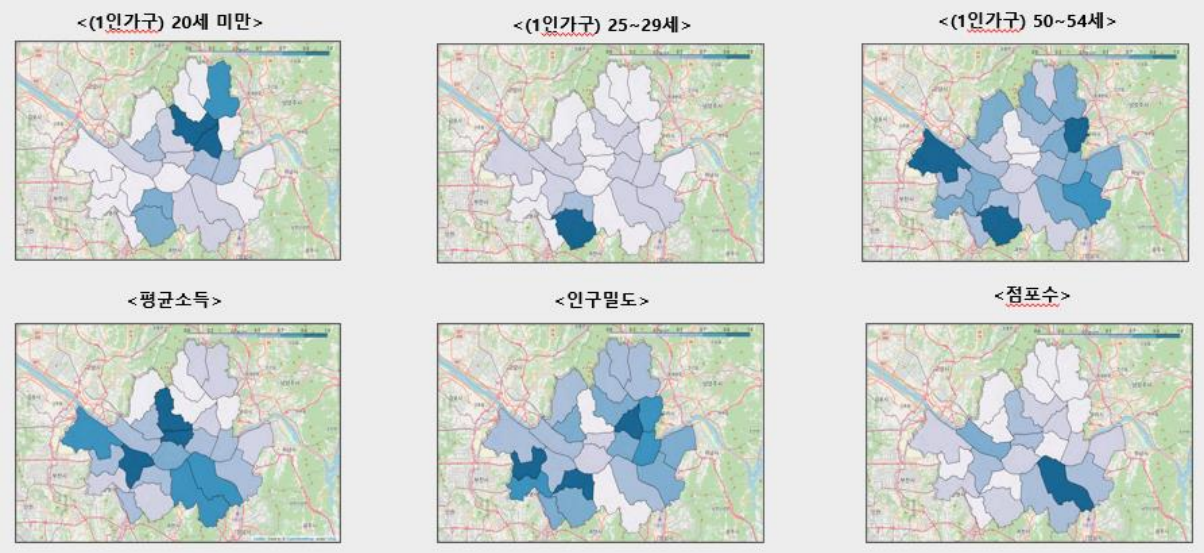
<변수 별 가중치>

변수 명	가중치
점포수	0.298423
인구밀도	0.209311
(1인가구) 50~54세	0.182578
(1인가구) 25~29세	0.131361
(1인가구) 20세미만	0.037763
평균소득(만원)	0.140564

[그림] 14

최종적으로 구한 Feature 별 가중치는 위와 같다. 또한 각 Feature별 가중치를 사용하기 위해, 모든 feature들에 대해 각각의 최소값0, 최대값1로, 0과 1 사이의 값으로 반환하는 Min-Max 정규화 과정을 진행하였다.

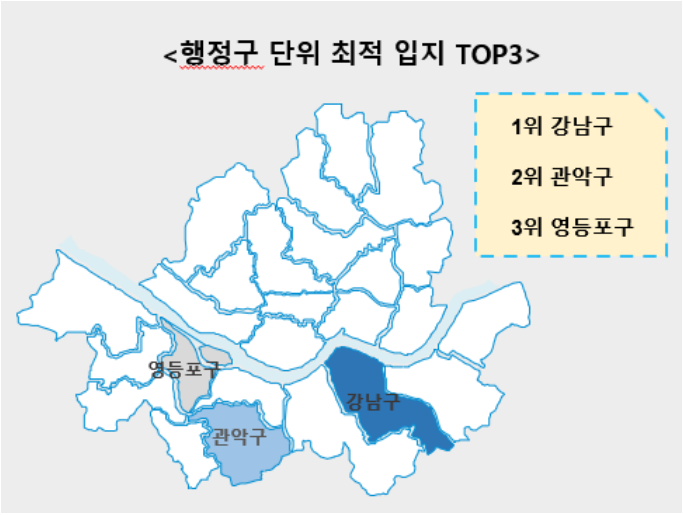
행정구 변수 별 지도 시각화



[그림] 15

파이썬의 folium을 사용하여 선정된 변수별로 구단위 지도에 시각화한 결과이다. 색이 진할 수록 점수가 높은 지역을 뜻한다.

행정구 최종 점수

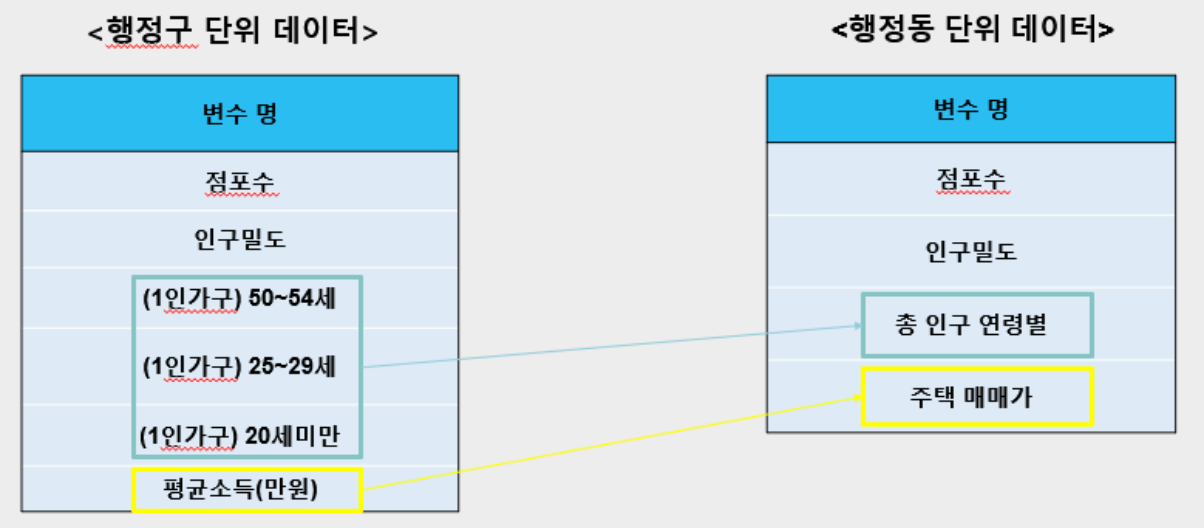


[그림] 16

행정구 배달 입지 점수=변수 별 행정구 점수x변수 별 가중치

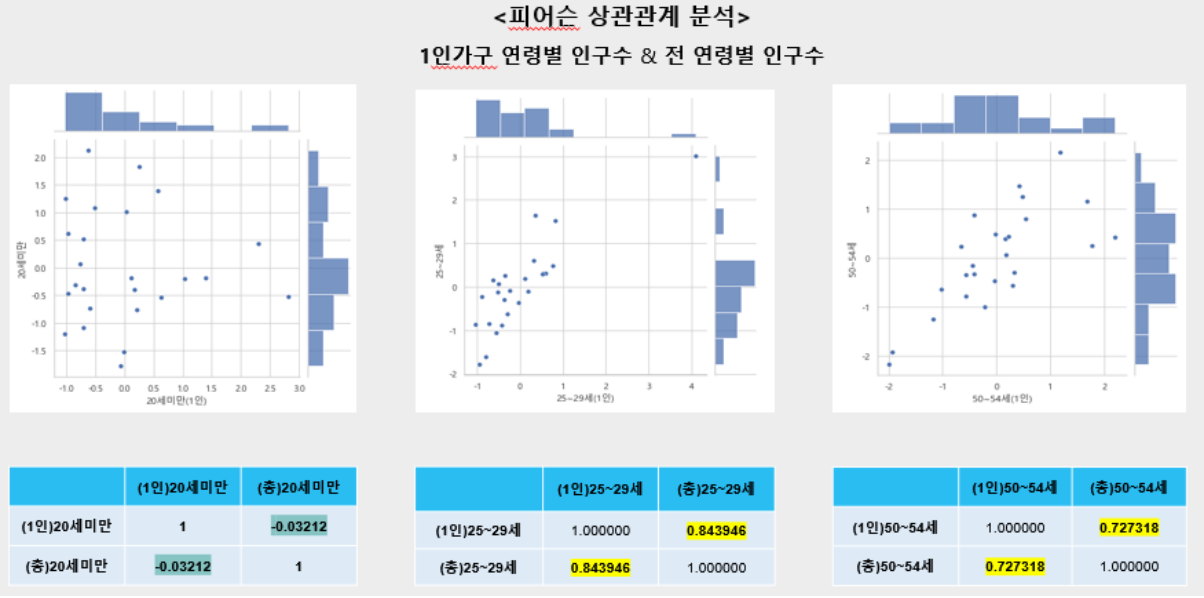
변수별 행정구 점수에 변수별 가중치를 곱해 최종적으로 최적의 행정구 입지를 선정하였다. 1위 강남구, 2위 관악구, 3위 영등포구의 결과가 나왔다.

행정동 변수 선정



[그림] 17

구 단위로 입지를 선정한 후, 행정동 단위로 들어가 보다 구체적인 입지선정을 하려 했다. 하지만 구 단위 입지선정에서 사용되었던 1인가구 연령별 데이터와 평균 소득데이터는 동 단위 데이터가 존재하지 않았다. 따라서 이 변수들을 대체할 수 있는 변수로 총인구 연령별 데이터와 주택 매매가 데이터를 활용하였다. 새로운 변수를 활용하기 이전에 과연 이 새로운 변수가 기존의 변수를 대체할 수 있는지를 입증하기 위해 피어슨 상관관계분석을 진행하였다.



	(1인)20세미만	(총)20세미만
(1인)20세미만	1	-0.03212
(총)20세미만	-0.03212	1

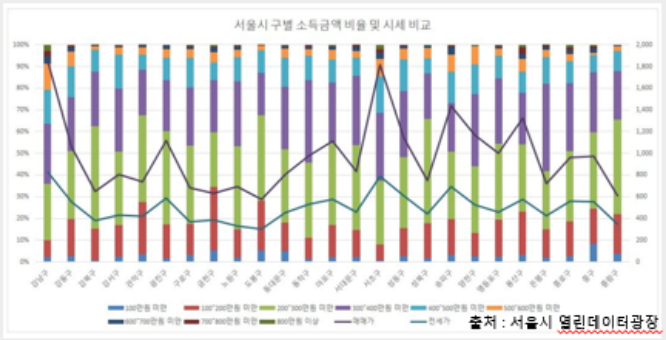
	(1인)25~29세	(총)25~29세
(1인)25~29세	1.000000	0.843946
(총)25~29세	0.843946	1.000000

	(1인)50~54세	(총)50~54세
(1인)50~54세	1.000000	0.727318
(총)50~54세	0.727318	1.000000

[그림] 18

행정구 선정에 사용한 1인가구 연령별 데이터와 행정동 선정에 사용할 전체인구 연령별 데이터를 피어슨 상관관계 분석을 한 결과 20세미만은 -0.03, 25~29세는약 0.85, 50~54세는 약 0.72가 나왔다. 25~29세와 50~54세 데이터는 1인가구와 전체인구간 상관관계가 0.7이상으로 높게 나타났지만, 20세미만 데이터는 상관관계가 -0.032로 매우 낮게 나타났다. 이는 전체인구의 경우 실제로 20세 미만 전체의 인구통계의 값이며, 1인가구의 경우는 실제로 0~4세나 5~9세, 10~14세,15~19세의 1인가구의 수가 극히 작기 때문일 것으로 보인다. 또한 20세 데이터의 가중치의 경우 약 0.038로 다른 데이터의 가중치보다 5~7배 작아 최종점수에 미치는 영향이 작을 것이라 생각하고 해당 데이터는 최종점수에 사용하지 않았다.

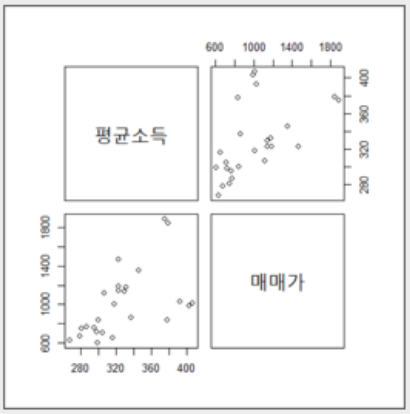
<서울시 구별 소득금액 비율 및 시세 비교>



→ 평균소득이 높은 행정구 ≒ 매매가가 높은 행정구

<피어슨 상관관계 분석>

평균 소득 & 부동산 실매매가



	평균소득	매매가
평균소득	1	0.53749
매매가	0.53749	1

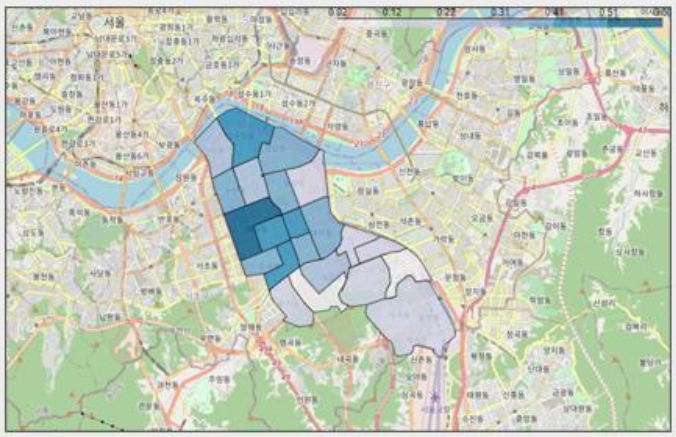
[그림] 19

서울시 열린 데이터광장의 '서울시 구별 소득금액 비율 및 시세 비교' 자료에 따르면 평균소득이 높은 지역과 부동산 매매가가 높은 지역이 거의 일치했다. 또한 피어슨 상관관계 분석을 해본 결과 평균소득과 주택 매매가 변수 간 상관관계가 약 0.54 수준으로 나와, 주택 매매가로 소득수준 변수를 대체 가능하다고 판단하였다.

행정동 변수 점수

<1위 강남구 행정동>

구분	최종점수
역삼1동	0.602892
도곡2동	0.482255
압구정동	0.461718
역삼2동	0.458767
대치4동	0.430043
삼성2동	0.394945
대치1동	0.3826
대치2동	0.362812
논현1동	0.346062
청담동	0.342297

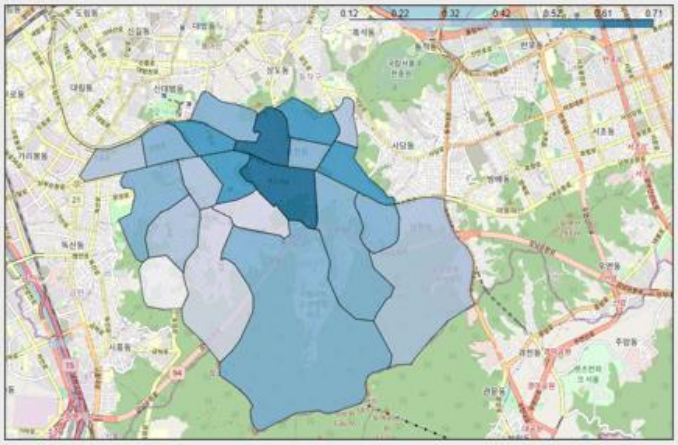


[그림] 20

최종점수 산출을 위해 데이터 정규화 스케일링 한 후, 행정동의 각 데이터들에 행정구에서 얻어낸 가중치를 이용하여 최종점수를 구하였다. 구 단위입지선정에서 1위를 했던 강남구에서 최적의 동 단위 입지를 선정한 결과 1위는 역삼1동, 2위 도곡2동, 3위 압구정동 순으로 배달상권 입지 점수가 높았다.

<2위 관악구 행정동>

구분	최종점수
청룡동	0.712916
은천동	0.681274
행운동	0.612722
신림동	0.542296
성현동	0.52915
서원동	0.520797
보라매동	0.508886
낙성대동	0.491228
인헌동	0.482526
신사동	0.466918

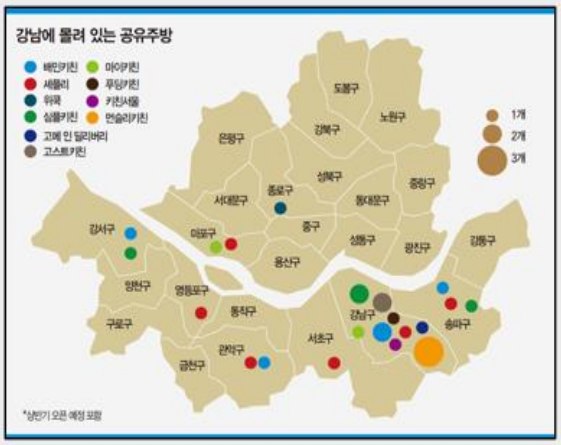


[그림] 21

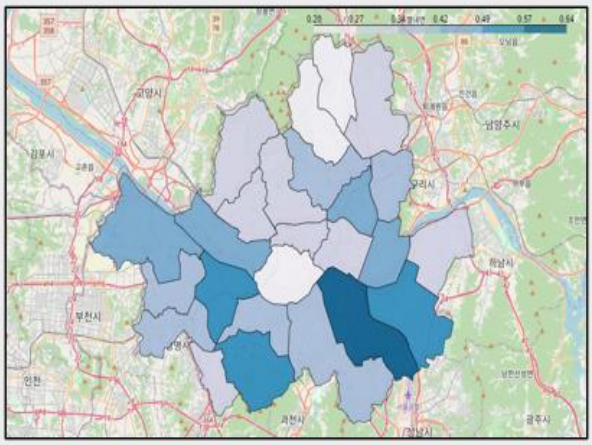
구 단위입지선정에서 2위를 했던 강남구에서 최적의 동단위 입지를 선정한 결과 1위는 청룡동, 2위 은천동, 3위 행운동 순으로 배달상권 입지 점수가 높았다.

모델 타당성

<서울시 공유주방 분포 현황>



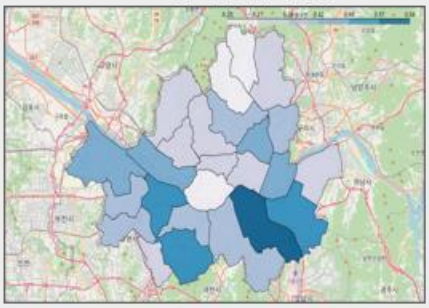
<배달입지 점수모델 결과>



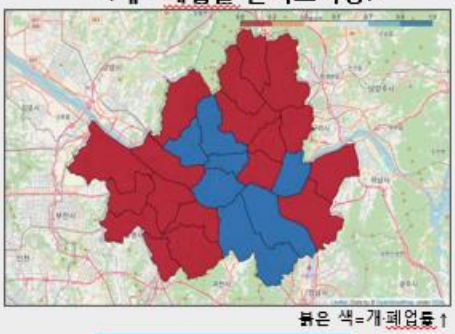
[그림] 22

현재 공유주방이 위치해 있는 모습과 비교해 본 결과, 두 지도 모두에서 현재 강남구에 공유주방의 대부분이 다수 포진해있는 것을 알 수 있었다. 이외 송파구, 강서구 등 공유주방 최적의 입지 모델과 유사한 분포를 가지고 있음을 확인할 수 있었다. 이러한 점은 본 모델의 타당성과 효과를 반증하고 있다고 볼 수 있다.

<BEST 입지 선정 결과>



<개 · 폐업률 클러스터링>



구분	최종점수
강남구	0.642548
관악구	0.557274
영등포구	0.510406
송파구	0.497348
강서구	0.478144
동대문구	0.455875
마포구	0.454974
광진구	0.438486
동작구	0.409355
중랑구	0.394023

강남구, 광진구를 제외한
나머지 **8개 행정구** 일치

개폐업률 높은 클러스터
마포구
관악구
영등포구
동대문구
송파구
동작구
중랑구
강서구

[그림] 23

본 프로젝트의 목적은 결국 공유주방을 통해 소상공인들의 창업의 어려움을 덜어주고자 함이었다. 따라서 추가적으로 소상공인 창업 개폐업률이 높은 지역을 클러스터링 기법으로

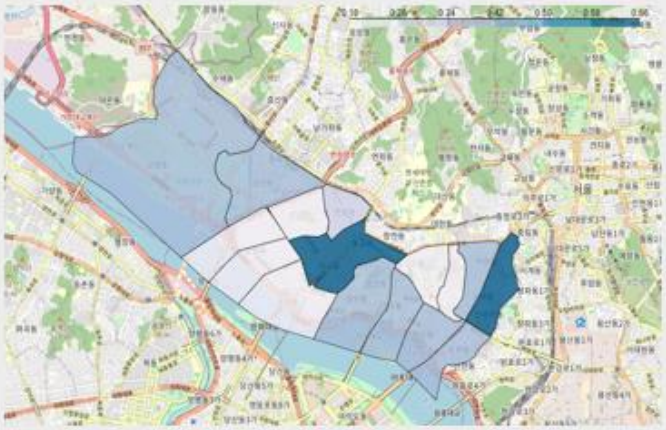
찾아보았고 다수의 최적의 입지가 개폐업률이 높은 지역과 일치함을 알 수 있었다. 예외적으로 최적의 입지 1순위인 강남구와 광진구만이 개폐업률이 낮았다. 즉 현재 강남구는 이미 공유주방이 적극적으로 활성화되고 있으며, 개폐업률도 낮다. 강남구를 제외한 후순위의 다른 구들은 개폐업률이 높은 상태이지만, 공유주방이 아직 활성화되어 있지 않은 모습이다. 따라서 해당 지역들의 양질의 입지에 공공서비스 형태의 공유주방 사업을 적극적으로 도입한다면 배달음식업 소상공인들에게 큰 도움이 될 수 있을 것으로 보인다.

모델 추가 테스트

<입지선정 모델 최종점수 결과>

구분	최종점수
서교동	0.662041
공덕동	0.615168
서강동	0.398754
성산2동	0.398035
아현동	0.388959
용강동	0.364374
신수동	0.353198
도화동	0.350428
상암동	0.342845
연남동	0.291575

<점수를 시각화한 마포구 지도>



결과 : 1위 서교동(0.66) 2위 공덕동(0.62) 3위 서강동(0.40)

[그림] 24

추가적으로, 모두에게 익숙한 마포구에서는 어느 '동'에 공유주방이 들어가면 좋을 지를 알아보았다. 결과는 위와 같이 서교동이 0.66으로 1위로 나왔고 뒤이어 공덕동이 2위, 서강동이 3위였다.

사용 데이터 목록 및 출처

데이터 명	출처
행정동 코드	KT 빅데이터 플랫폼
서울시 행정구 geojson	서울 열린데이터 광장
대한민국 행정동 geojson	Vuski의 github
행정구 서울시 연령별 1인가구 수	서울 열린데이터 광장
주문배달건수	SKT 빅데이터 허브
행정구/행정동 서울시 점포수	우리마을 상권분석 서비스
서울시 부동산 실거래가 정보	서울부동산정보광장
행정동 연령인구 수	통계청
서울시 개폐업 수	우리마을 상권분석 서비스
행정구/행정동 서울시 인구밀도	서울 열린데이터 광장
서울시 평균소득	보통사람 금융생활 보고서

부록: 중요 분석 상세설명

배달주문건수 데이터의 경우 행정동 데이터의 결측치가 많은 문제로, 배달주문건수 행정구 데이터를 이용해 배달주문건수에 영향을 미치는 독립변수들을 파악하고자 하였다.
RandomForestRegressor 모델학습을 통해 독립변수들의 중요도를 산출하고 이를 가중치로 이용하여 행정구 배달상권 점수 모델을 만든다.

행정구 배달상권 점수 모델의 결과와 실제 공유주방이 위치하고 있는 행정구를 비교하여 모델의 타당성을 검증한다.

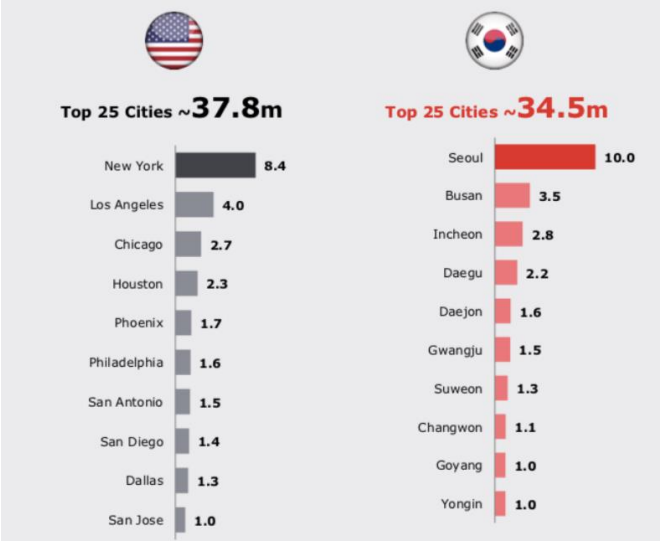
최종 목표는 행정동을 추천하는 것이기 때문에, 행정구 배달상권 점수 모델의 타당성을 바탕으로 같은 가중치를 사용하여 행정동 배달상권 입지를 선정한다.

행정동 데이터의 경우 연령별 1인가구 데이터나, 평균소득 데이터가 존재하지 않는 관계로 해당 데이터들을 대체할 데이터를 선정하였다. 대체할 데이터의 선정 과정은 상관계수 분석을 통해 이루어졌다. 연령별 1인가구 데이터는 연령별 인구 데이터로 대체하였으며 평균소득 데이터는 매매가데이터로 대체하였다.

1. 자료 수집 과정

분석에 사용된 자료는 SKtelecom Bigdatahub에서 제공하는 서울시 행정구별 배달주문건수 데이터(2017년도)를 목표 변수로 사용하고, 논문과 기사 등을 활용하여 배달 상권에 영향을 미치는 요인들을 정성적 조사를 진행하였다. 요인들은 배달 입지와 배달이용자를 기준으로 나누어 독립 변수들로 선정하여 데이터를 수집하였다. 배달 입지에 관한 변수로는 점포 수, 인구밀도를 선정하였고, 배달이용자에 관한 변수로는 가족 구성 특성인 1인가구, 1세대가구 ,2세대가구, 1인가구, 비친족가구, 평균소비수준, 평균소득을 선정하였다.

'서울시 성장상권과 쇠퇴상권 내 외식산업의 생존율 비교' 논문에서 상권분석의 변수로 거주인구, 점포 수, 소득 수준을 활용하였으며 '상권분석 방법론을 이용한 외식창업 사례연구' 논문에서는 인구환경특성을 상권분석 변수로 활용하였다. 1인가구는 다양한 배달 앱들의 경우 1인 메뉴 배너가 존재하며, 2047년까지 1인 가구 비중이 37.3%까지 증가할 것이라는 통계청의 예측을 토대로 선정하였다. 인구밀도의 경우는 독일 배달앱 회사 '딜리버리 히어로'는 '요기요'와 '배달통'을 운영하는 회사가 한국 배달시장의 가치를 인구밀도를 통해 설명한 것을 토대로 선정하였다. 배달은 지역 서비스이기 때문에 전체 국가의 인구보다 도시별 인구수가 중요하며, 서울과 뉴욕을 비교해보았을 때 서울이 뉴욕보다 인구밀도가 1.5배 높아 배달 서비스가 성공하기에 서울이 뉴욕보다 훨씬 높다고 보았다.



<그림1.1> 미국과 한국의 주요도시 인구밀도 비교

2. 데이터 전처리

앞서 설명한 자료수집 과정을 통해 서울시 행정구 별 최종 데이터 셋은 아래 <그림1.2>와 같이 배달주문건수, 1세대가구, 2세대가구, 1인가구, 비친족가구, 점포수, 평균소득, 인구밀도로 이루어져있다. 해당데이터들은 KT_data_20200717데이터의 'adstrd_master'의 서울시 행정구 코드를 사용하여 행정구 별로 데이터를 통합해주었다.

	A	B	C	D	E	F	G	H	I
1	구분	배달주문건수	1세대 가구	2세대 가구	1인 가구	비친족 가구	점포수	평균소득(만원)	인구밀도(명)/㎢)
2	종로구	43495	30072	68811	70933	4853	7061	403	6869
3	중구	68097	26031	56057	60511	4235	6683	407	13514
4	용산구	63386	46317	103180	99247	7244	4951	346	11179
5	성동구	43067	60818	154481	117220	6254	3818	332	18551
6	광진구	28868	66141	179513	164617	7244	5178	323	21819
7	동대문구	55358	65493	170807	162109	7604	5262	300	25748
8	중랑구	49208	79382	216091	142648	5318	4481	268	22318
9	성북구	56914	80282	230141	162748	8072	4900	287	18533
10	강북구	56503	61773	169562	117265	4790	4074	279	13898
11	도봉구	26885	66408	189329	88426	4379	3095	299	16752
12	노원구	62386	93322	317652	147814	7151	4713	298	15748
13	은평구	60948	92954	255887	139624	8423	4658	281	16534
14	서대문구	66939	59866	155717	130273	7223	4643	337	18440
15	마포구	56464	74703	182104	162505	10943	8878	330	16174
16	양천구	30060	76694	269636	99283	5195	4184	323	27289

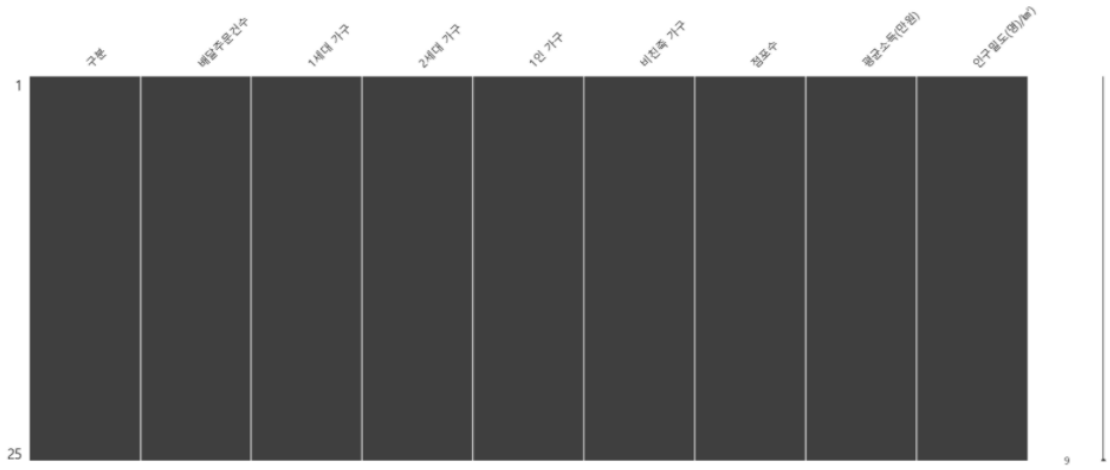
<그림1.2> 서울시 행정구 별 최종 데이터 셋

데이터를 수집, 통합하는 과정에서 데이터의 무결성을 해하는 결측치를 삭제한 상태이기 때문에 최종 데이터 셋의 결측치를 확인하고자 시각화하였을 때, 결측치가 존재하지 않는 것을 확인할 수 있다.

```
In [9]: import missingno as msno

        msno.matrix(df_1)

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x1dd5d1939d0>
```



<그림1.3> 데이터 셋 결측치 시각화

데이터 마다 값의 단위, 크기가 모두 다르기때문에 데이터의 값이 너무 크거나 작은 경우 모델 학습 알고리즘 과정에서 0으로 수렴하거나 무한대로 발산할 위험이 있기 때문에 그 다음으로 데이터 스케일링을 진행하였다. 파이썬의 sklearn을 이용하여 StandardSclaer을 통해 각 feature의 평균을 0, 분산을 1로 변경해주었다.

<표1.1> 데이터 스케일링

	배달주문건수	1세대 가구	2세대 가구	1인 가구	비친족 가구	점포수	평균소득(만원)	인구밀도(명)/㎢
0	-0.714446	-1.994282	-1.848603	-1.397777	-0.849579	0.630377	1.906767	-2.253482
1	0.339468	-2.178628	-2.024080	-1.603678	-1.007551	0.450062	2.008624	-0.857201
2	0.137656	-1.253201	-1.375736	-0.838393	-0.238399	-0.376143	0.455302	-1.347843
3	-0.732781	-0.591679	-0.669908	-0.483310	-0.491460	-0.916611	0.098801	0.201199
4	-1.341046	-0.348849	-0.325504	0.453086	-0.238399	-0.267858	-0.130377	0.887887

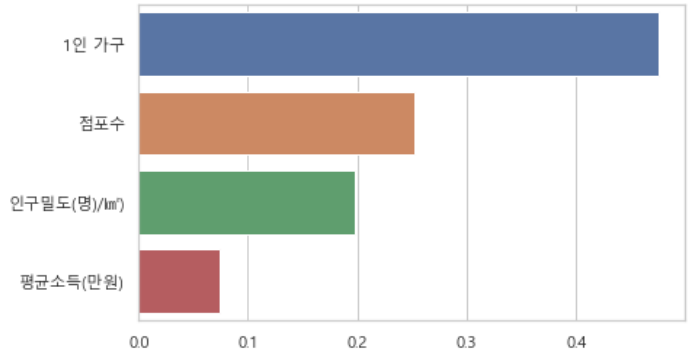
데이터 사용 목적은 배달주문건수를 정확히 예측하고자 하는 것이 아닌 배달주문건수와 배달입지와 배달이용자를 기준으로 나누어 선정한 독립변수들 간의 관계를 설명하고 독립변수가 목표변수에 미치는 영향력의 크기를 측정하는 것이다. 따라서 다중공선성 문제의 위험을 제거하기 위해 분산팽창계수(VIF , Variance Inflation Factor)를 사용하였다. VIF >10 이면 다중공선성의 문제가 있을 수 있다 판단하였다. 1세대가구, 2세대가구, 비친족가구의 경우 VIF를 이용해 처리하는 과정에서 모두 10을 초과하는 값이 나와 제거하였다. 최종 feature은 <표1.2>와 같다.

<표1.2> feature들의 VIF 표

	VIF Factor	features
0	1.400970	1인 가구
1	1.949072	점포수
2	2.102274	평균소득(만원)
3	1.484793	인구밀도(명)/㎢)

3. 랜덤포레스트 모델 학습

목표변수를 배달주문건수로 설정하도 독립변수들로 1인가구, 점포수, 평균소득, 인구밀도로 설정하여 목표변수와 독립변수들 사이의 인과관계를 파악하기 위해 회귀분석을 진행하였다. 목표변수에 영향을 끼치는 독립변수들의 중요도를 이용해 구 선정에 있어 가중치를 부여하는 것이 목적이기에 파이썬 sklearn의 RandomForestRegressor을 이용하여 학습을 진행하였다. 학습을 진행하고, 변수 중요도를 산출한 결과 <그림1.4>와 1인가구, 점포수, 인구밀도, 평균소득 순으로 중요도가 나타났다.



<그림1.4> 변수 중요도

1인가구 중요도가 다른 feature들의 중요도에 비해 상당히 높게 나타났음을 알 수 있다. 따라서 서울열린데이터 광장의 서울시 1인가구 연령별 데이터를 포함하여 변수 중요도를 다시 살펴보고자 하였다.

<표 1.3> 서울시 1인가구 연령별 데이터

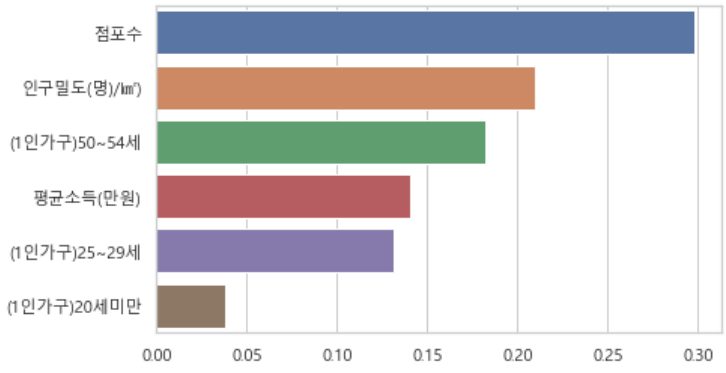
(1인가구)20세미만	(1인가구)20~24세	(1인가구)25~29세	(1인가구)30~34세	(1인가구)35~39세	(1인가구)40~44세	(1인가구)45~49세	(1인가구)50~54세	(1인가구)55~59세	(1인가구)60~64세	(1인가구)65~69세	(1인가구)70~74세
507	2992	3627	2552	1796	1518	1723	1540	1660	1471	1151	998
499	2097	2669	2169	1686	1400	1632	1367	1510	1376	1097	912
265	2658	4614	4620	3503	2638	2742	2041	2195	1984	1535	1340
708	4467	6284	4603	3560	2876	2877	2529	2728	2309	1767	1476
802	6280	10335	7976	5421	3936	3814	3276	3405	2881	2099	1499

1인가구 연령별 데이터를 포함한 데이터 셋을 다시 스케일링 과정과 분산팽창계수를 이용해 다중공선성의 위험이 의심되는 feature들을 제거하였더니 최종적으로 남은 feature는 <표 1.4>와 같다.

<표 1.4> 1인가구 연령별 데이터 포함 VIF

	VIF Factor	features
0	1.803779	(1인가구)20세미만
1	2.342715	(1인가구)25~29세
2	2.762378	(1인가구)50~54세
3	1.847142	점포수
4	2.791304	평균소득(만원)
5	1.546053	인구밀도(명)/㎢)

앞선 과정처럼 RandomForestRegressor 모델을 이용해 학습한 후, 변수 중요도를 산출한 결과 <그림1.5>와 같이 점포수, 인구밀도, (1인가구)50~54세, 평균소득(만원), (1인가구)25~29세, (1인가구)20세 미만 순으로 나타났다.



<그림 1.5> 1인가구 연령별 데이터 포함 변수 중요도

4. 행정구 선정

서울시 행정구 별 feature들의 값에 RandomForestRegressor을 통해 파악한 중요도를 가중치로 사용하여 행정구 별 배달상권 입지 점수를 매기고자 하였다. Feature 별 가중치는 아래 표와 같다. 또한 각 Feature별 가중치를 이용하기 전에 모든 feature들에 대해 각각의 최소값0, 최대값1로, 다른 값들은 0과 1 사이의 값으로 반환하는 Min-Max 정규화 과정을 진행하였다.

<표1.4> Min-Max 정규화 된 데이터

	배달주문건수	(1인가구)20세미만	(1인가구)25~29세	(1인가구)50~54세	점포수	평균소득(만원)	인구밀도(명)/㎢)	구분
0	0.180567	0.254078	0.066596	0.053182	0.398153	0.971223	0.000000	종로구
1	0.448015	0.249059	0.028823	0.000000	0.360205	1.000000	0.325416	중구
2	0.396802	0.102258	0.105512	0.207193	0.186327	0.561151	0.211068	용산구
3	0.175914	0.380176	0.171359	0.357209	0.072583	0.460432	0.572086	성동구
4	0.021557	0.439147	0.331086	0.586843	0.209116	0.395683	0.732125	광진구
5	0.309530	1.000000	0.269340	0.503535	0.217548	0.230216	0.924535	동대문구

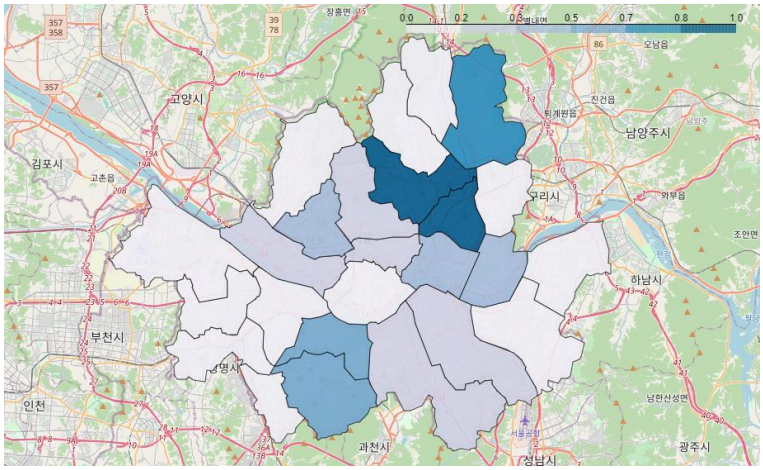
<표1.5> Feature 별 가중치

변수 명	가중치
점포수	0.298423
인구밀도	0.209311
(1인가구) 50~54세	0.182578
(1인가구) 25~29세	0.131361
(1인가구) 20세미만	0.037763
평균소득(만원)	0.140564

4.1 (1인가구) 20세 미만

(1인가구)20세 미만 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. (1인가구) 20세미만 점수가 높은 행정구는 동대문구, 성북구, 노원구, 동작구, 관악구 순으로 나타났다. 또한 파이썬의 folium을 이용해 행정구를 점수 별로 시각화 하였다. 지도 시각화의 경우 색이 진할수록 높은 점수임을 의미한다.

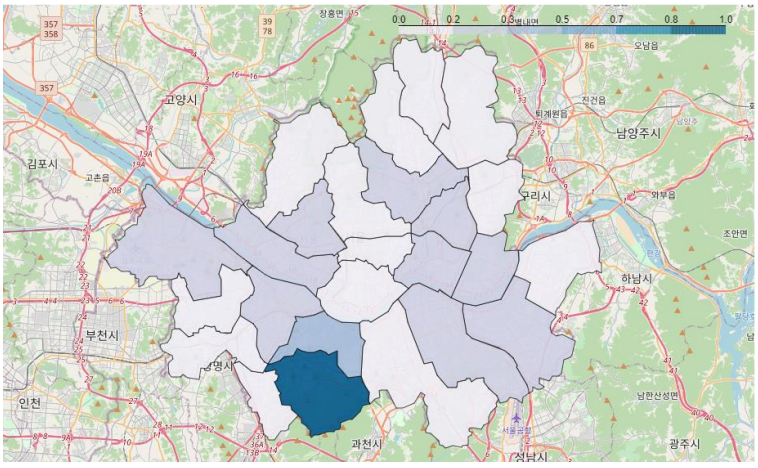
	구분	(1인가구)20세미만
5	동대문구	1.000000
7	성북구	0.885822
10	노원구	0.668758
19	동작구	0.658093
20	관악구	0.627353
4	광진구	0.439147
12	서대문구	0.436010
3	성동구	0.380176
13	마포구	0.319322
22	강남구	0.274153



4.2 (1인가구) 25~29세

(1인가구) 25~29세 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. (1인가구) 25~29세 점수가 높은 행정구는 관악구, 동작구, 마포구, 광진구, 강서구 순으로 나타났다.

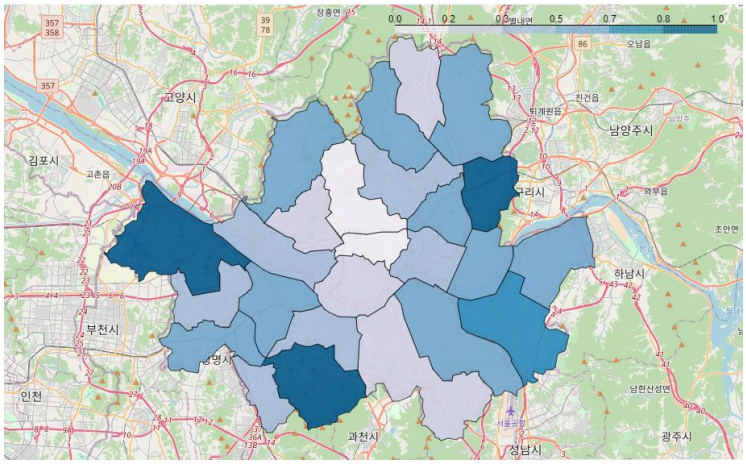
	구분	(1인가구)25~29세
20	관악구	1.000000
19	동작구	0.353560
13	마포구	0.332032
4	광진구	0.331086
15	강서구	0.326552
22	강남구	0.305733
18	영등포구	0.276003
5	동대문구	0.269340
7	성북구	0.252031
23	송파구	0.240044



4.3 (1인가구) 50~54세

(1인가구) 50~54세 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. (1인가구) 50~54세 점수가 높은 행정구는 관악구, 중랑구, 강서구, 송파구, 영등포구 순으로 나타났다.

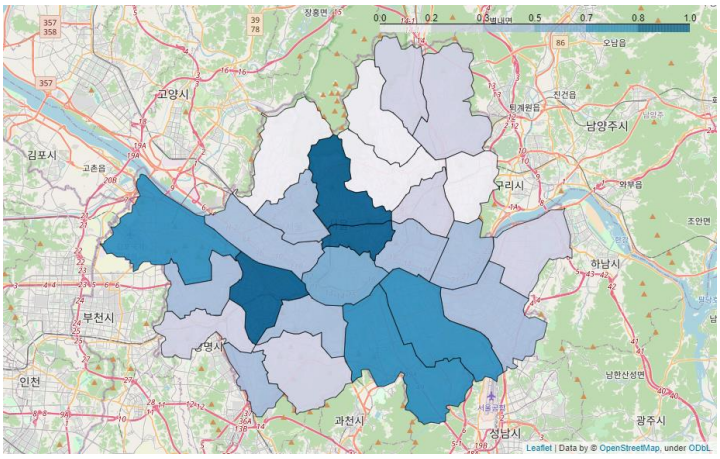
	구분	(1인가구)50~54세
20	관악구	1.000000
6	중랑구	0.983400
15	강서구	0.866585
23	송파구	0.709499
18	영등포구	0.608054
11	은평구	0.605902
22	강남구	0.603750
4	광진구	0.586843
24	강동구	0.578235
8	강북구	0.557332



4.4 평균소득

평균소득 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. 평균소득 점수가 높은 행정구는 중구, 종로구, 영등포구, 서초구, 강서구 순으로 나타났다.

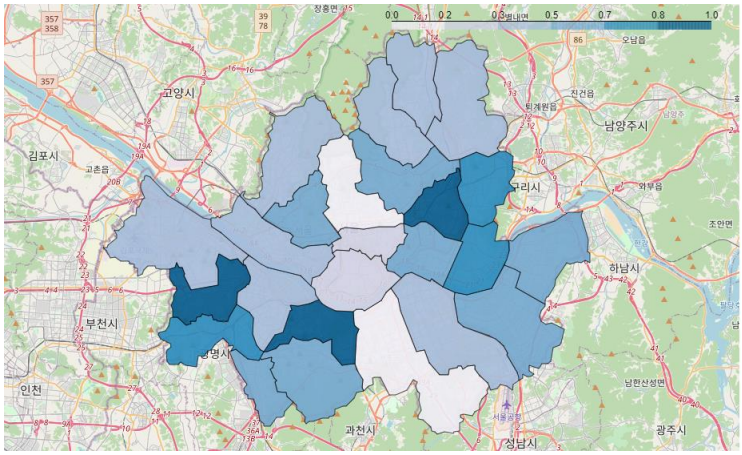
	구분	평균소득(만원)
1	중구	1.000000
0	종로구	0.971223
18	영등포구	0.899281
21	서초구	0.798561
15	강서구	0.791367
22	강남구	0.769784
2	용산구	0.561151
12	서대문구	0.496403
3	성동구	0.460432
13	마포구	0.446043



4.5 인구밀도

인구밀도 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. 인구밀도 점수가 높은 행정구는 양천구, 동대문구, 동작구, 중랑구, 구로구 순으로 나타났다.

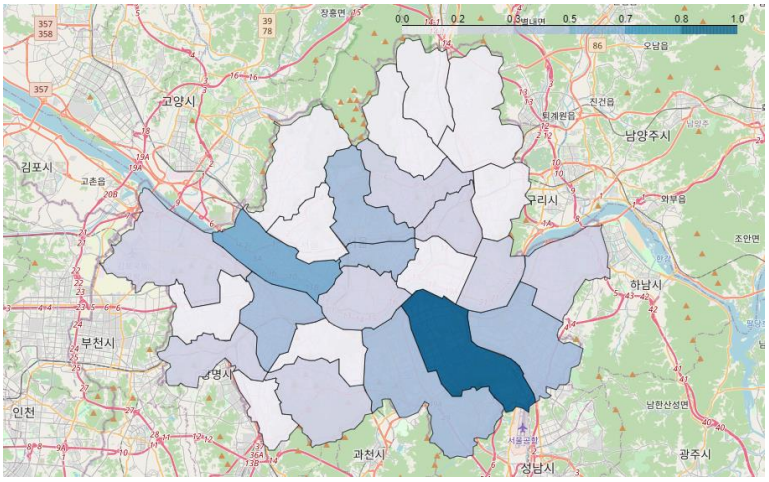
	구분	인구밀도(명)/㎢
14	양천구	1.000000
5	동대문구	0.924535
19	동작구	0.886827
6	중랑구	0.756562
16	구로구	0.738345
4	광진구	0.732125
23	송파구	0.633888
17	금천구	0.617042
3	성동구	0.572086
7	성북구	0.571205



4.6 점포수

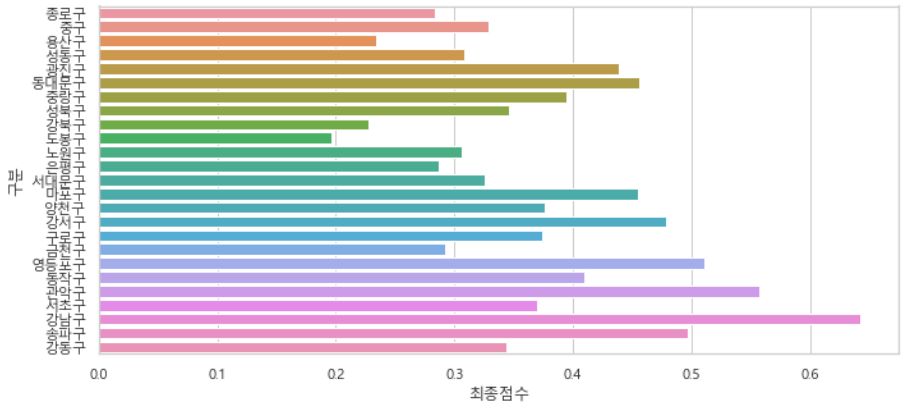
점포수 데이터를 기준으로 상위 10개의 서울시 행정구를 나열하였다. 점포수 점수가 높은 행정구는 중구, 종로구, 영등포구, 서초구, 강서구 순으로 나타났다.

	구분	점포수
22	강남구	1.000000
13	마포구	0.580564
21	서초구	0.499649
23	송파구	0.481578
18	영등포구	0.459291
0	종로구	0.398153
1	중구	0.360205
15	강서구	0.275073
20	관악구	0.271961
24	강동구	0.238229



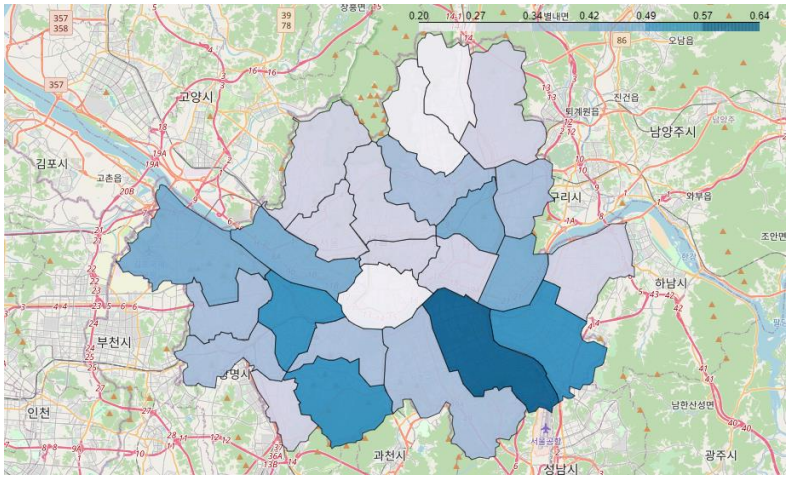
4.7 배달상권 최종 점수

각 feature 별 행정구 점수를 살펴본 후 각 점수에 가중치를 곱하여 최종 배달상권 입지 점수를 구하였다. 강남구가 1 순위, 관악구가 2 순위, 영등포구가 3 순위를 차지하였다.

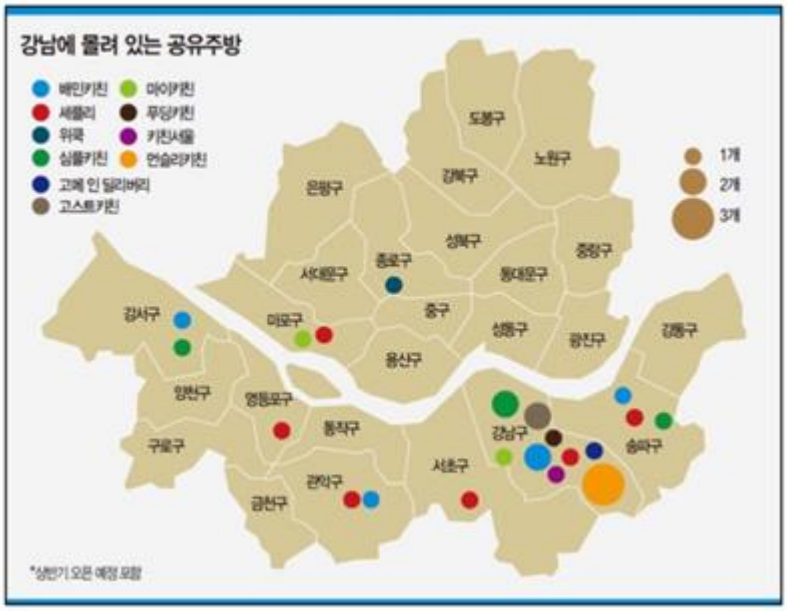


<그림 1,?> 배달상권 입지 최종점수 막대그래프

	구분	최종점수
22	강남구	0.642548
20	관악구	0.557274
18	영등포구	0.510406
23	송파구	0.497348
15	강서구	0.478144
5	동대문구	0.455875
13	마포구	0.454974
4	광진구	0.438486
19	동작구	0.409355
6	중랑구	0.394023



5. 타당성 검증



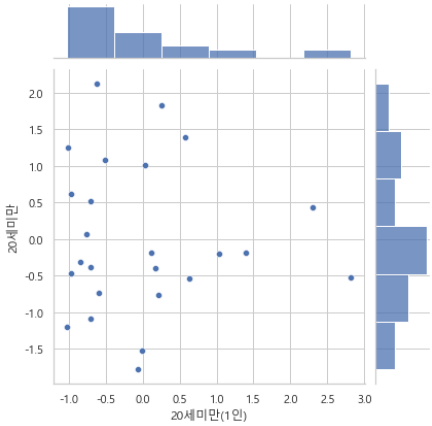
현재 공유주방의 위치와 비교해본 결과, 강남구에 공유주방이 몰려있으며, 이외에도 관악구, 영등포구, 송파구 등에 공유주방이 위치해 있음을 확인 할 수 있었다. 이는 모델을 통한 공유주방 입지선정 위치와 흡사함을 보이고있다. 공유주방업체 또한 상권과 입지를 분석하여 공유주방을 입점시키기에, 수립된 행정구 단위의 공유주방 입지선정 모델은 타당성이 있음을 확인할 수 있었다.

1. 행정동 배달상권 입지 점수

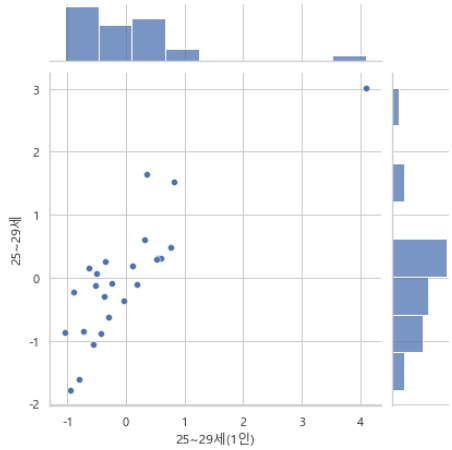
1.1 상관관계 분석

행정동 데이터의 경우 행정구 데이터와 다르게 1 인인구의 연령별 데이터와 평균소득 데이터가 존재하지 않았다. 따라서 이 데이터를 대체할 데이터를 피어슨 상관관계 분석을 통해 선정하였다.

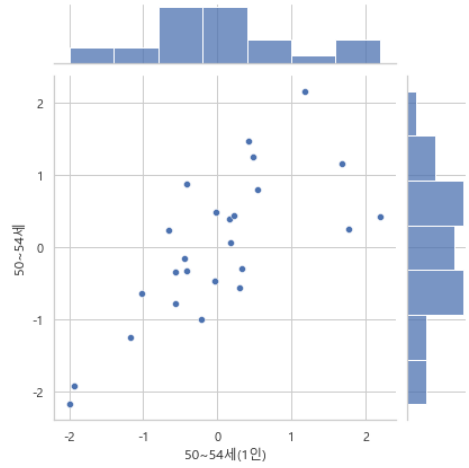
행정구에서 1 인가구 연령별 사용한 데이터는 20 세미만, 25~29 세, 50~54 세이다. 따라서 1 인가구 연령별 데이터와 전체인구 연령별 데이터를 피어슨 상관관계 분석을 한 결과 20 세미만은 -0.032312, 25~29 세는 0.843946, 50~54 세는 0.727318 이 나타났다.



	20세미만(1인)	20세미만
20세미만(1인)	1.000000	-0.032123
20세미만	-0.032123	1.000000

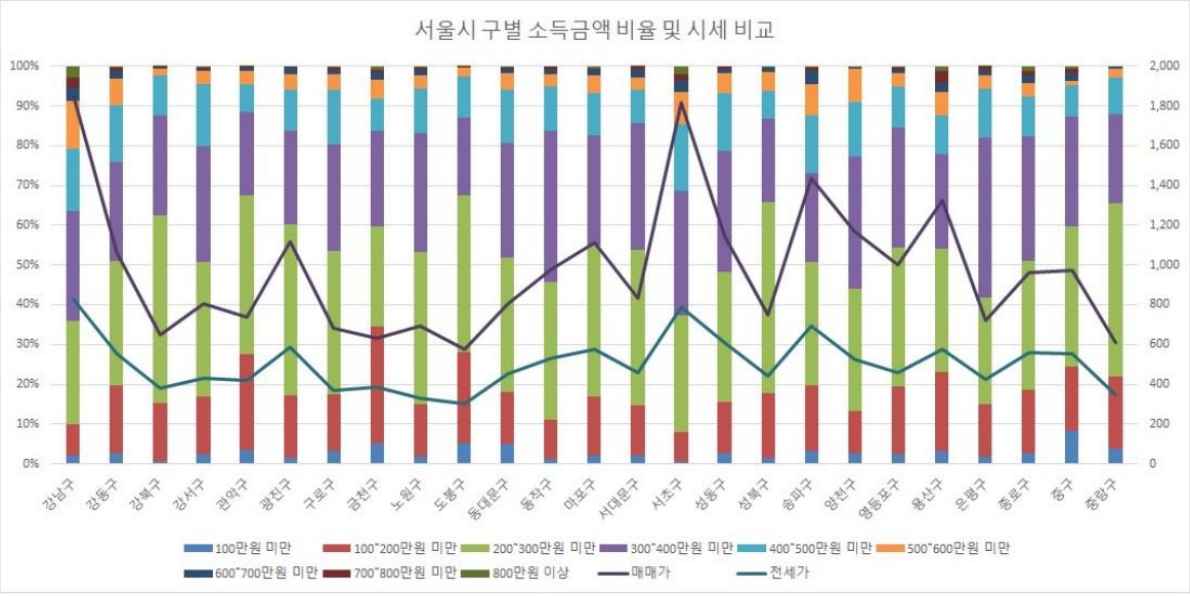


	25~29세(1인)	25~29세
25~29세(1인)	1.000000	0.843946
25~29세	0.843946	1.000000



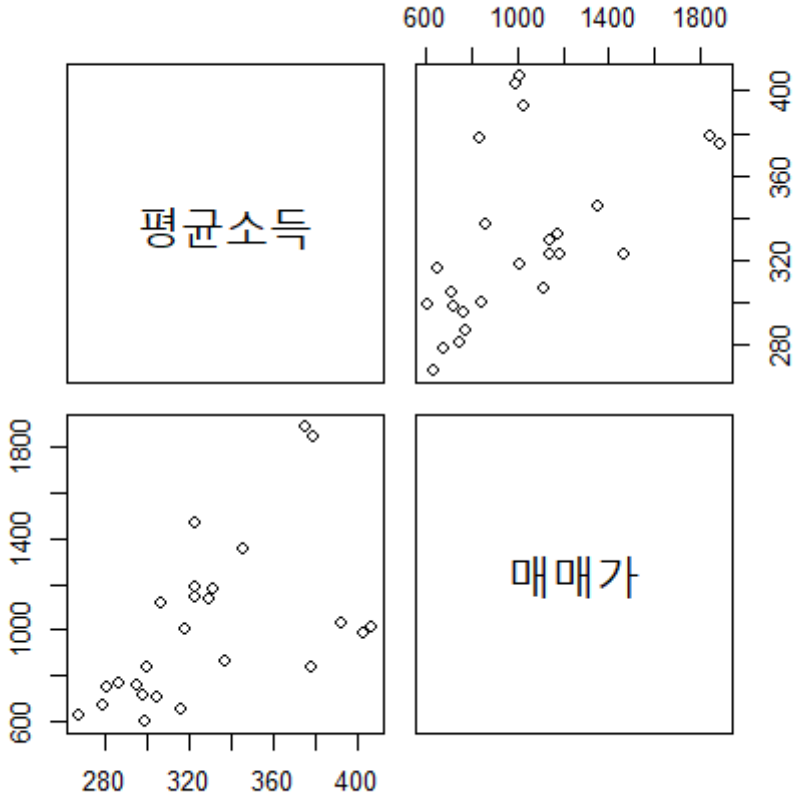
	50~54세(1인)	50~54세
50~54세(1인)	1.000000	0.727318
50~54세	0.727318	1.000000

평균소득과 매매가의 경우 서울시 열린 데이터 광장의 자료에 따르면 아래와 같이 평균소득이 높은 행정구와 매매가가 높은 곳은 행정구가 흡사함을 알 수 있다.



강남구, 서초구, 송파구, 용산구, 양천구에서 특히 평균소득과 매매가가 높음을 보여주고있다.
따라서 행정구에서 평균소득 데이터와 매매가 데이터를 피어슨 상관관계 분석을 한 결과 0.5388236 으로 나타났다.

```
> cor(income[,2:3])
           평균소득     매매가
평균소득 1.0000000 0.5388236
매매가   0.5388236 1.0000000
```



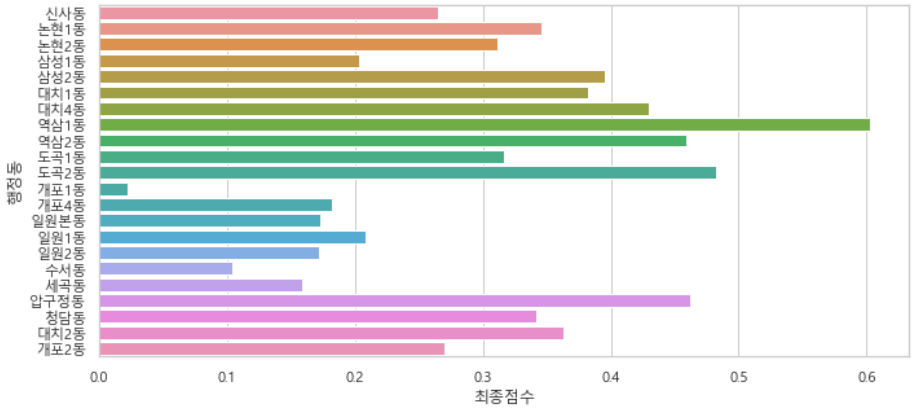
2. 1 순위 강남구의 행정동 배달상권 점수

앞선 상관관계 분석을 통해 평균소득은 강남구 행정동의 매매가 데이터로, 1 인가구 연령별 데이터는 전체인구 연령별 데이터로 대체하였다. 25~29 세와 50~54 세 데이터는 1 인가구와 전체인구간 상관관계가 0.7 이상으로 높게 나타났지만, 20 세미만 데이터는 상관관계가 -0.032 로 매우 낮게 나타났다. 이는 전체인구의 경우 실제로 20 세 미만 전체의 인구통계의 값이며, 1 인가구의 경우는 실제로 0~4 세나 5~9 세, 10~14 세,15~19 세의 1 인가구의 수가 극히 작기때문일 것으로 보인다. 또한 20 세 데이터의 가중치의 경우 0.037762 로 다른 데이터의 가중치보다 5~7 배 작아 최종점수에 미치는 영향이 작을 것이라 생각하고 해당 데이터는 최종점수에 사용하지 않았다.

<표 2.1> 강남구 행정동 데이터

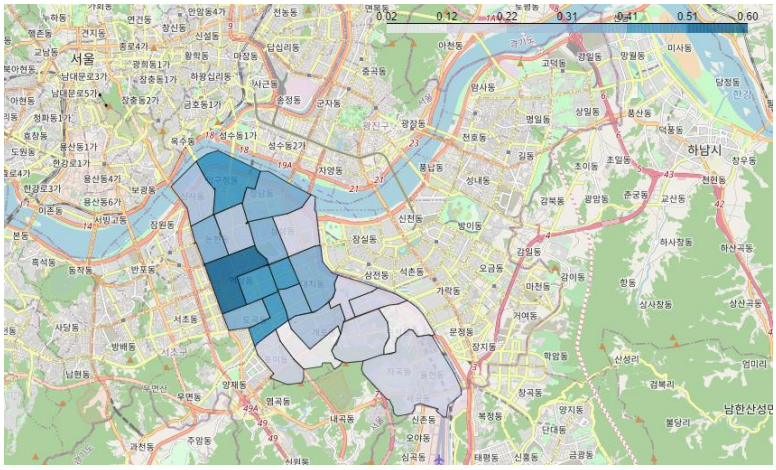
	행정구	행정동	인구밀도	점포수	매매가	25~29세	50~54세
0	강남구	신사동	9284	1074	1045341000	1367	1423
1	강남구	논현1동	19034	997	682708500	3396	1502
2	강남구	논현2동	14901	1213	682708500	2265	1541
3	강남구	삼성1동	7691	822	945919200	1145	1289
4	강남구	삼성2동	24737	567	945919200	2245	2464

최종점수 산출을 위해 데이터를 정규화 스케일링 한 후, 행정동의 각 데이터들에 행정구에서 얻어낸 가중치를 이용하여 최종점수를 구하였다. 이를 아래 그림들처럼 막대그래프와 folium 을 이용해 지도로 시각화하였다. 그 결과 역삼 1 동, 도곡 2 동, 압구정동 순으로 배달상권 입지 점수가 높았다.



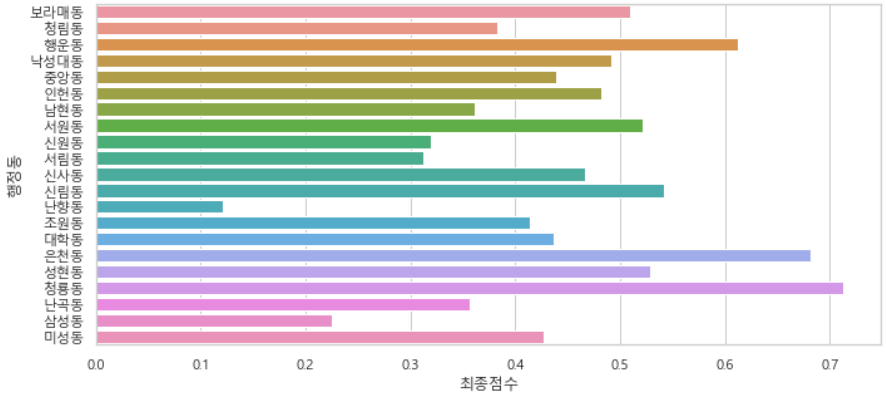
<그림 2.1> 강남구 행정동 배달입지 점수 막대그래프

	구분	최종점수
7	역삼1동	0.602892
10	도곡2동	0.482255
18	압구정동	0.461718
8	역삼2동	0.458767
6	대치4동	0.430043
4	삼성2동	0.394945
5	대치1동	0.382600
20	대치2동	0.362812
1	논현1동	0.346062
19	청담동	0.342297



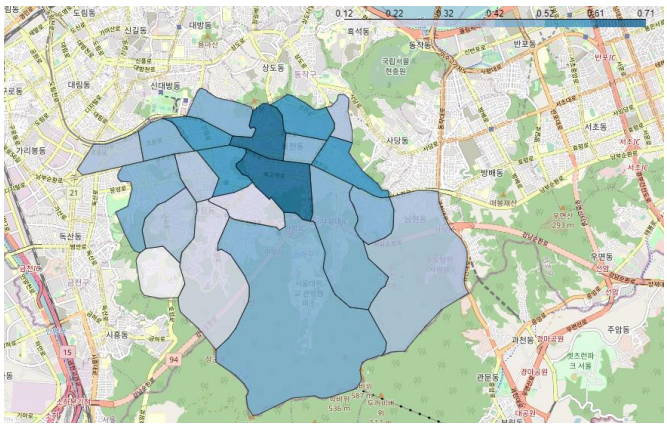
3. 2 순위 관악구 행정동 배달상권 점수

관악구 역시 강남구처럼 최종점수 산출을 위해 데이터를 정규화 스케일링 한 후, 행정동의 각 데이터들에 행정구에서 얻어낸 가중치를 이용하여 최종점수를 구하였다. 이를 아래 그림들처럼 막대그래프와 folium 을 이용해 지도로 시각화하였다. 그 결과 청룡동, 온천동, 행운동 순으로 배달상권 입지 점수가 높았다.



<그림 3.1> 관악구 행정동 배달상권 점수 막대그래프

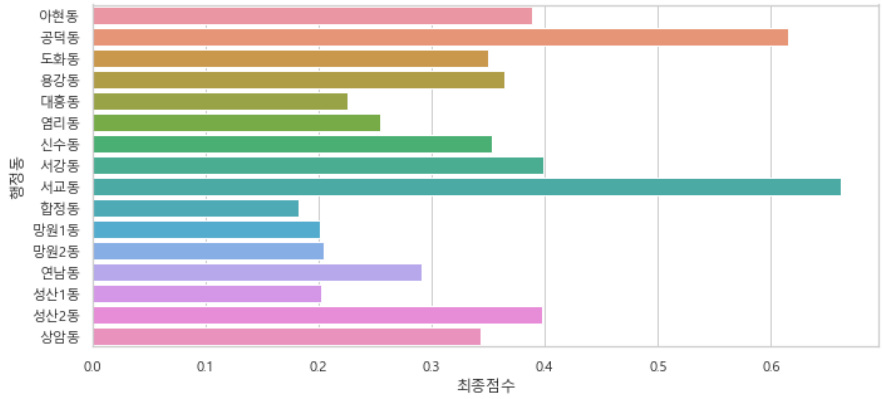
	구분	최종점수
17	청룡동	0.712916
15	은천동	0.681274
2	행운동	0.612722
11	신림동	0.542296
16	성현동	0.529150
7	서원동	0.520797
0	보라매동	0.508886
3	낙성대동	0.491228
5	인헌동	0.482526
10	신사동	0.466918



4. 마포구 행정동 배달상권 점수

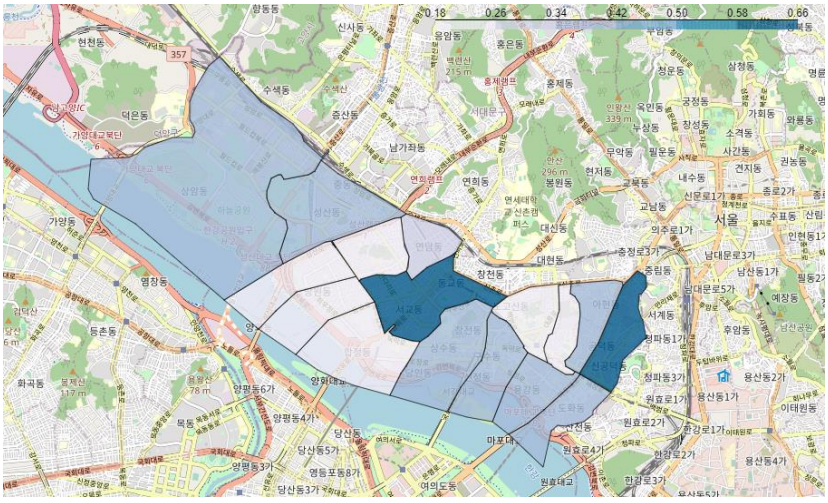
홍익대학교가 위치하는 마포구에 공유주방을 차리고 싶은 경우, 어떤 행정동에 입점하여야할까? 앞서 만든 모델을 통해 마포구의 행정동 배달상권 점수를 도출하였다.

최종점수 산출을 위해 데이터를 정규화 스케일링 한 후, 행정동의 각 데이터들에 행정구에서 얻어낸 가중치를 이용하여 최종점수를 구하였다. 이를 아래 그림들처럼 막대그래프와 folium 을 이용해 지도로 시각화하였다. 그 결과 서교동, 공덕동, 성산 2 동 순으로 배달상권 입지 점수가 높았다.



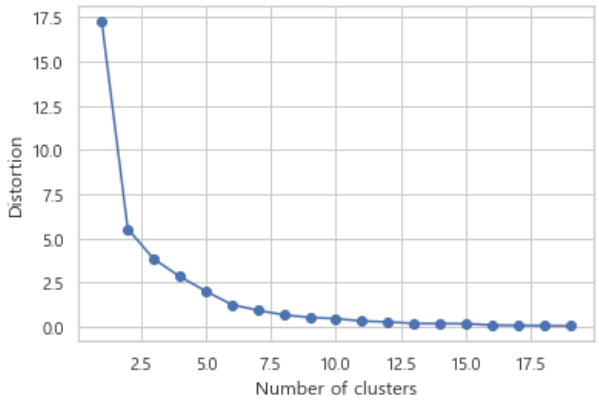
<그림 2.?> 마포구 행정동 배달입지 점수 막대그래프

	구분	최종점수
8	서교동	0.662041
1	공덕동	0.615168
7	서강동	0.398754
14	성산2동	0.398035
0	아현동	0.388959
3	용강동	0.364374
6	신수동	0.353198
2	도화동	0.350428
15	상암동	0.342845
12	연남동	0.291575



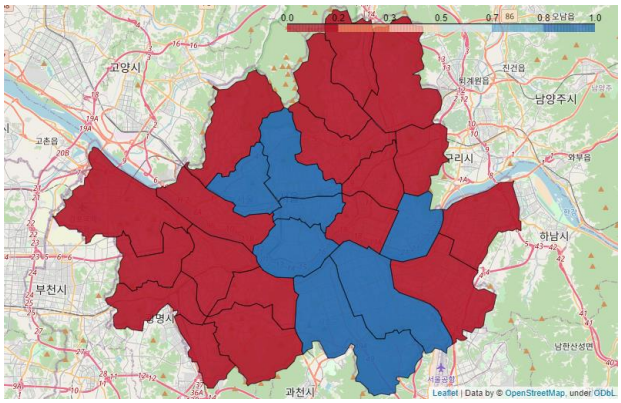
개폐업률 클러스터링

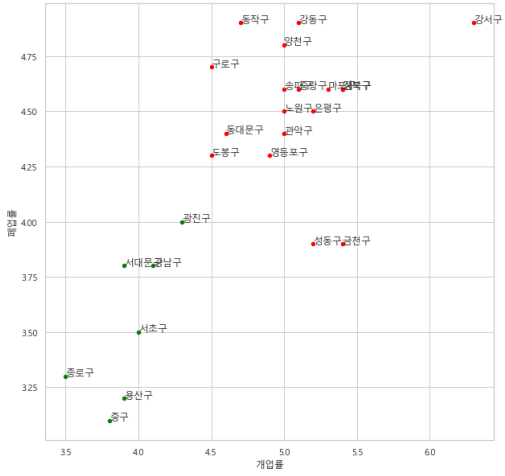
서울시 행정구의 개폐업률 데이터를 이용하여 개폐업률 정도에 따른 행정구 그룹을 K-means 클러스터링하고자 한다. 최적의 클러스터링을 결정하기 위해 elbow 기법을 사용하여 최적의 클러스터 개수를 2 로 설정하였다.



<그림> Elbow point 그래프

클러스터링 결과를 mapping 하고 이를 folium 을 이용해 지도로 시각화 하였다. 해당 결과를 통해 지도의 빨간색으로 칠해진 행정구의 경우 개폐업률이 높은 군집임을 알 수 있다.





부록: 중요 분석 소스코드

행정구 데이터 스케일링 (StandardScaler)

```
scale_df = df_1.drop("구분",axis=1)
```

```
from sklearn.preprocessing import StandardScaler
standardScaler = StandardScaler()
standardScaler.fit(scale_df)
standard_df = standardScaler.transform(scale_df)
```

```
df_1 = pd.DataFrame(standard_df)
```

```
df_1.columns = [ '배달주문건수', '1세대 가구', '2세대 가구', '1인 가구', '비친족 가구', '점포수', '평균소득(만원)',
       '인구밀도(명)/㎢)']
```

```
df_1.head()
```

	배달주문건수	1세대 가구	2세대 가구	1인 가구	비친족 가구	점포수	평균소득(만원)	인구밀도(명)/㎢)
0	-0.714446	-1.994282	-1.848603	-1.397777	-0.849579	0.630377	1.906767	-2.253482
1	0.339468	-2.178628	-2.024080	-1.603678	-1.007551	0.450062	2.008624	-0.857201
2	0.137656	-1.253201	-1.375736	-0.838393	-0.238399	-0.376143	0.455302	-1.347843
3	-0.732781	-0.591679	-0.669908	-0.483310	-0.491460	-0.916611	0.098801	0.201199
4	-1.341046	-0.348849	-0.325504	0.453086	-0.238399	-0.267858	-0.130377	0.887887

행정구 데이터 다중공선성 VIF 계수로 확인 과정

```
dfvif=df_1.drop('배달주문건수',axis=1)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif=pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dfvif.values,i) for i in range(dfvif.shape[1])]
vif['features']=dfvif.columns
vif
```

	VIF Factor	features
0	28.458586	1세대 가구
1	19.677390	2세대 가구
2	9.372515	1인 가구
3	12.149220	비친족 가구
4	5.266284	점포수
5	2.340829	평균소득(만원)
6	1.635437	인구밀도(명)/㎢)

```
dfvif=dfvif.drop(['비친족 가구'], axis=1)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif=pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dfvif.values,i) for i in range(dfvif.shape[1])]
vif['features']=dfvif.columns
vif
```

	VIF Factor	features
0	1.400970	1인 가구
1	1.949072	점포수
2	2.102274	평균소득(만원)
3	1.484793	인구밀도(명)/㎢)

랜덤포레스트 학습 및 변수중요도 도출

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

y_df = df_1["배달주문건수"]
X_df = df_1.drop("배달주문건수",axis=1)

X_train, X_test, y_train, y_test = train_test_split(X_df, y_df, test_size=0.2, random_state = 216)
```

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators=400, min_samples_split=3,random_state=1)
rf.fit(X_train, y_train)

importance_values = rf.feature_importances_
ftr_importances=pd.Series(importance_values, index=X_train.columns)
ftr_top20 = ftr_importances.sort_values(ascending=False)[:20]
sns.barplot(x=ftr_top20, y=ftr_top20.index)
plt.show()
```

1 인가구 연령별 데이터 추가 데이터셋

2세대 가구	1인 가구	비친족 가구	(1인가구)20세미만	(1인가구)20~24세	(1인가구)25~29세	(1인가구)30~34세	(1인가구)35~39세	(1인가구)40~44세	(1인가구)45~49세	(1인가구)50~54세	(1인가구)55~59세	(1인가구)60~64세	(1인가구)65~69세	(1인가구)70~74세	점포수	평균소득(만원)	인구밀도(명)/㎢)
68811	70933	4853	507	2992	3627	2552	1796	1518	1723	1540	1660	1471	1151	998	7061	403	6869
56057	60511	4235	499	2097	2669	2169	1686	1400	1632	1367	1510	1376	1097	912	6683	407	13514
103180	99247	7244	265	2658	4614	4620	3503	2638	2742	2041	2195	1984	1535	1340	4951	346	11179
154481	117220	6254	708	4467	6284	4603	3560	2876	2877	2529	2728	2309	1767	1476	3818	332	18551
179513	164617	7244	802	6280	10335	7976	5421	3936	3814	3276	3405	2881	2099	1499	5178	323	21819

추가된 데이터 셋 스케일링(StandardScaler)

```
scale_df = df_2.drop("구분",axis=1)
```

```
from sklearn.preprocessing import StandardScaler
standardScaler = StandardScaler()
standardScaler.fit(scale_df)
standard_df = standardScaler.transform(scale_df)
```

```
df_2 = pd.DataFrame(standard_df)
```

```
df_2.columns = [ '배달주문건수', '(1인가구)20세미만',
       '(1인가구)20~24세', '(1인가구)25~29세', '(1인가구)30~34세', '(1인가구)35~39세',
       '(1인가구)40~44세', '(1인가구)45~49세', '(1인가구)50~54세', '(1인가구)55~59세',
       '(1인가구)60~64세', '(1인가구)65~69세', '(1인가구)70~74세', '점포수', '평균소득(만원)',
       '인구밀도(명)/㎢)']
```

연령별 1 인가구 추가된 데이터 셋 다중공선성 VIF 계수로 확인 과정

```
dfvif=df_2.drop('배달주문건수',axis=1)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif=pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dfvif.values,i) for i in range(dfvif.shape[1])]
vif['features']=dfvif.columns
vif
```

	VIF Factor	features
0	15.084472	(1인가구)20세미만
1	53.287970	(1인가구)20~24세
2	326.040572	(1인가구)25~29세
3	823.857075	(1인가구)30~34세
4	1390.720409	(1인가구)35~39세
5	1102.135745	(1인가구)40~44세
6	254.213108	(1인가구)45~49세
7	250.157813	(1인가구)50~54세
8	1225.886916	(1인가구)55~59세
9	651.707190	(1인가구)60~64세
10	141.759037	(1인가구)65~69세
11	46.126598	(1인가구)70~74세
12	14.901969	점포수
13	10.349517	평균소득(만원)
14	3.539966	인구밀도(명/㎢)

```
dfvif=dfvif.drop(['(1인가구)40~44세'], axis=1)

from statsmodels.stats.outliers_influence import variance_inflation_factor

vif=pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(dfvif.values,i) for i in range(dfvif.shape[1])]
vif['features']=dfvif.columns
vif
```

	VIF Factor	features
0	1.803779	(1인가구)20세미만
1	2.342715	(1인가구)25~29세
2	2.762378	(1인가구)50~54세
3	1.847142	점포수
4	2.791304	평균소득(만원)
5	1.546053	인구밀도(명/㎢)

랜덤포레스트 재 학습 및 변수중요도 도출

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

y_df = df_2["배달주문건수"]
X_df = df_2.drop("배달주문건수",axis=1)

X_train, X_test, y_train, y_test = train_test_split(X_df, y_df, test_size=0.2, random_state = 216)
```

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators=400, min_samples_split=3,random_state=1)
rf.fit(X_train, y_train)

importance_values = rf.feature_importances_
ftr_importances=pd.Series(importance_values, index=X_train.columns)
ftr_top20 = ftr_importances.sort_values(ascending=False)
sns.barplot(x=ftr_top20, y=ftr_top20.index)
plt.show()
```

가중치를 이용하기 위한 데이터 셋 정규화(MinMaxScaler)

```
from sklearn import preprocessing

x=df_2.values.astype(float)
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
df_3=pd.DataFrame(x_scaled,columns=df_2.columns)
```

```
df_3['구분']=df['구분'].values
```

```
df_3
```

	배달주문건수	(1인가구)20세미만	(1인가구)25~29세	(1인가구)50~54세	점포수	평균소득(만원)	인구밀도(명)/㎢	구분
0	0.180567	0.254078	0.066596	0.053182	0.398153	0.971223	0.000000	종로구
1	0.448015	0.249059	0.028823	0.000000	0.360205	1.000000	0.325416	중구
2	0.396802	0.102258	0.105512	0.207193	0.186327	0.561151	0.211068	용산구
3	0.175914	0.380176	0.171359	0.357209	0.072583	0.460432	0.572086	성동구
4	0.021557	0.439147	0.331086	0.586843	0.209116	0.395683	0.732125	광진구
5	0.309530	1.000000	0.269340	0.503535	0.217548	0.230216	0.924535	동대문구
6	0.242673	0.063363	0.092895	0.983400	0.139143	0.000000	0.756562	중랑구
7	0.326445	0.885822	0.252031	0.472794	0.181207	0.136691	0.571205	성북구
8	0.321977	0.138645	0.062298	0.557332	0.098283	0.079137	0.344221	강북구
9	0.000000	0.074655	0.000000	0.331386	0.000000	0.223022	0.483986	도봉구

구 별 변수별 시각화

```
import folium

korea_location = [37.563,126.982]
```

```
geo_path = 'C:₩₩Users₩₩user7₩₩Desktop₩₩data₩₩시각화₩₩상가상권정보₩₩seoul_municipalities_geo_simple.json'

import json
geo_json = json.load(open(geo_path, encoding="utf-8"))
```

```
m_result1 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','(1인가구)20세미만'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result1)

m_result1
```

```
m_result2 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','(1인가구)25~29세'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result2)

m_result2
```

```
m_result3 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','(1인가구)50~54세'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result3)

m_result3
```

```
m_result4 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','평균소득(만원)'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result4)

m_result4
```

```
m_result5 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','인구밀도(명)/㎢)'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result5)

m_result5
```

```
m_result7= folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','점포수'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result7)

m_result7
```

최종점수 산출 및 데이터 시각화

```
df_3['최종점수']=(df_3['(1인가구)20세미만']*0.037763)+(df_3['(1인가구)25~29세']*0.131361)+(df_3['(1인가구)50~54세']*0.182578)+(df_3['평균소
```

```
y =df_3['최종점수'].sort_values(ascending=False)
```

```
plt.figure(figsize=(10,5))
sns.barplot(x=y, y=df['구분'])
plt.show()
```

```
m_result9 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_3,
    columns=['구분','최종점수'],
    key_on= 'feature.properties.name',
    fill_color='PuBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result9)

m_result9
```

연령별 1 인가구 – 연령별 총인구 상관관계 분석

```
df= pd.read_csv('C:₩₩Users₩₩user7₩₩Desktop₩₩1인가구연령별.csv',encoding='cp949')
```

```
df['20세미만']=df['15~19세']+df['10~14세']+df['5~9세']
```

```
df = df.drop('성별',axis=1)
```

```
scale_df = df.drop("구분",axis=1)
```

```
from sklearn.preprocessing import StandardScaler
standardScaler = StandardScaler()
standardScaler.fit(scale_df)
standard_df = standardScaler.transform(scale_df)
```

```
df = pd.DataFrame(standard_df)
```

```
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	-0.014337	-0.408376	-0.797816	-1.040673	-1.317370	-1.566400	-1.805119	-1.929582	-1.702881	-1.818846	-1.799809	-1.746271	-1.593388	-1.647709
1	-0.063585	-0.712073	-0.939958	-1.063011	-1.296343	-1.522290	-1.947508	-1.993680	-1.892387	-1.823471	-1.824892	-1.863887	-1.828608	-1.769229
2	-0.712026	-0.598445	-0.555723	-0.433402	-0.403114	-0.563468	-0.832124	-1.173465	-1.305329	-1.319326	-1.328011	-1.296950	-1.166566	-0.780715
3	0.204546	0.013376	-0.299733	-0.453662	-0.536288	-0.524001	-0.584366	-0.565723	-0.676044	-0.763378	-0.914739	-0.959960	-0.798937	-0.764357
4	0.174450	0.603946	0.547593	0.453090	0.337228	0.261474	0.100052	0.328086	0.177765	-0.135278	-0.503857	-0.637506	-0.699242	-0.815769

```
df_corr1 = df[['20세미만(1인)','20세미만']]
corr = df_corr1.corr(method='pearson')
print(corr)
```

```
           20세미만(1인)      20세미만
20세미만(1인)    1.000000 -0.032123
20세미만       -0.032123  1.000000
```

```
sns.jointplot(x=df['20세미만(1인)'],y=df['20세미만'],kind='scatter')
```

```
df_corr = df[['25~29세(1인)','25~29세']]
corr = df_corr.corr(method='pearson')
print(corr)
```

```
             25~29세(1인)     25~29세
25~29세(1인)     1.000000  0.843946
25~29세          0.843946  1.000000
```

```
sns.jointplot(x=df['25~29세(1인)'],y=df['25~29세'],kind='scatter')
```

```
df_corr = df[['50~54세(1인)','50~54세']]
corr = df_corr.corr(method='pearson')
print(corr)
```

```
            50~54세(1인)     50~54세
50~54세(1인)     1.000000  0.727318
50~54세          0.727318  1.000000
```

```
sns.jointplot(x=df['50~54세(1인)'],y=df['50~54세'],kind='scatter')
```

평균소득 – 매매가 상관관계 분석

```
> income=read.csv(file="C://Users//user//Documents//카카오톡 받은 파일//고재승//행정구 별균소득,매매가.csv",header=T)
> head(income)
      구분 평균소득 매매가
1   종로구       403    988
2     중구       407   1009
3   용산구       346   1353
4   성동구       332   1179
5   광진구       323   1141
6 동대문구       300    841
> cor(income[,2:3])
           평균소득     매매가
평균소득 1.0000000 0.5388236
매매가   0.5388236 1.0000000
```

강남구 행정동 배달상권 점수 산출 과정

```
df_gng=df_gn.drop(['행정구','행정동'],axis=1)
```

```
from sklearn import preprocessing

x=df_gng.values.astype(float)
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
df_3=pd.DataFrame(x_scaled,columns=df_gng.columns)
```

```
df_3['구분']=df_gn['행정동'].values
```

```
df_3['최종점수']=(df_3['25~29세']*0.131361)+(df_3['50~54세']*0.182578)+(df_3['매매가']*0.140564)+(df_3['인구밀도']*0.209311)+(df_3['점포수
```

```
y =df_3['최종점수'].sort_values(ascending=False)

plt.figure(figsize=(10,5))
sns.barplot(x=y, y=df_gn['행정동'])
plt.show()
```

```
import folium
korea_location = [37.563,126.982]

geo_path = 'C:₩₩Users₩₩user7₩₩Desktop₩₩gng.json'

import json
geo_json = json.load(open(geo_path, encoding="utf-8"))
```

```
geo_json
```


```
{'type': 'FeatureCollection',
 'features': [{'type': 'Feature',
   'properties': {'adm_nm': '신사동',
    'adm_cd': '1123051',
    'adm_cd2': '1168051000',
    'sgg': '11680',
    'sido': '11',
    'sidonm': '서울특별시',
    'sggnm': '강남구'},
   'geometry': {'type': 'MultiPolygon',
    'coordinates': [[[[127.0280781532314 9, 37.519584003369],
       [127.02720592329885, 37.51926729481519],
       [127.02526080146446, 37.51855642054487],
       [127.02116872995951, 37.516819200932005],
       [127.01944104956044, 37.51598724164066],
       [127.01947830573368, 37.51634195419062],
       [127.0190251971729, 37.51823713836514],
       [127.01891540067083, 37.51857494289874],
       [127.01828346580481, 37.52051905145247 4],
```


```
m_result1 = folium.Map(location=korea_location,
                zoom_start=12.45)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=data,
    columns=['행정동','최종점수'],
    key_on= 'properties.adm_nm',
    fill_color='PuBu',
    fill_opacity=0.8,
    line_opacity=0.7).add_to(m_result1)

m_result1
```

관악구 행정동 배달상권 점수 산출 과정

```
df_ga=pd.read_csv('C:₩₩Users₩₩user7₩₩Desktop₩₩관악구.csv',encoding='cp949')
```

```
df_gag=df_ga.drop(['행정구','행정동'],axis=1)
```

```
from sklearn import preprocessing

x=df_gag.values.astype(float)
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
df_3=pd.DataFrame(x_scaled,columns=df_gag.columns)
```

```
df_3['구분']=df_ga['행정동'].values
```

```
df_3['최종점수']=(df_3['25~29세']*0.131361)+(df_3['50~54세']*0.182578)+(df_3['매매가']*0.140564)+(df_3['인구밀도']*0.209311)+(df_3['점포수
```

```
y =df_3['최종점수'].sort_values(ascending=False)

plt.figure(figsize=(10,5))
sns.barplot(x=y, y=df_ga['행정동'])
plt.show()
```

```
import folium
korea_location = [37.563,126.982]

geo_path = 'C:₩₩Users₩₩user7₩₩Desktop₩₩gag.json'

import json
geo_json = json.load(open(geo_path, encoding="utf-8"))
```

```
m_result3 = folium.Map(location=korea_location,
                zoom_start=12.45)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_ga,
    columns=['행정동','최종점수'],
    key_on= 'properties.adm_nm',
    fill_color='PuBu',
    fill_opacity=0.8,
    line_opacity=0.7).add_to(m_result3)

m_result3
```

마포구 행정동 배달상권 점수 산출 과정

```
df_mp=pd.read_csv('C:₩₩Users₩₩user7₩₩Desktop₩₩마포구.csv',encoding='cp949')
```

```
df_mpg=df_mp.drop(['행정구','행정동'],axis=1)

from sklearn import preprocessing

x=df_mpg.values.astype(float)
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
df_3=pd.DataFrame(x_scaled,columns=df_mpg.columns)

df_3['구분']=df_mp['행정동'].values

df_3.head()
```

	인구밀도	점포수	매매가	25~29세	50~54세	구분
0	0.813721	0.125278	0.624663	0.202583	0.366068	아현동
1	1.000000	0.029281	0.591756	1.000000	1.000000	공덕동
2	0.898710	0.087472	0.383332	0.191144	0.313414	도화동
3	0.629956	0.084136	0.891933	0.307380	0.228165	용강동
4	0.318045	0.057821	0.657430	0.318081	0.043878	대흥동

```
df_3['최종점수']=(df_3['25~29세']*0.131361)+(df_3['50~54세']*0.182578)+(df_3['매매가']*0.140564)+(df_3['인구밀도']*0.209311)+(df_3['점포수
```

```
y =df_3['최종점수'].sort_values(ascending=False)

plt.figure(figsize=(10,5))
sns.barplot(x=y, y=df_mp['행정동'])
plt.show()
```

```
import folium
korea_location = [37.563,126.982]

geo_path = 'C:₩₩Users₩₩user7₩₩Desktop₩₩mpg.json'

import json
geo_json = json.load(open(geo_path, encoding="utf-8"))
```

```
m_result2 = folium.Map(location=korea_location,
                zoom_start=12.45)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df_mp,
    columns=['행정동','최종점수'],
    key_on= 'properties.adm_nm',
    fill_color='PuBu',
    fill_opacity=0.8,
    line_opacity=0.7).add_to(m_result2)

m_result2
```

개폐업률 클러스터링 과정

```
data_cluster3=df2[['개업률','폐업률']]
```

```
distortions = []
for i in range(1, 20):
    km = KMeans(
        n_clusters=i, init='random',
        n_init=10, max_iter=300,
        tol=1e-04, random_state=0
    )
    km.fit(data_cluster3)
    distortions.append(km.inertia_)

plt.plot(range(1, 20), distortions, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Distortion')
fig=plt.figure(figsize=(18, 16), dpi= 80, facecolor='w', edgecolor='k')
plt.show()
```

```
kmeans = KMeans(n_clusters=2, random_state=216)
kmeans.fit(data_cluster3)
labels = kmeans.labels_
centers=kmeans.cluster_centers_
```

```
df2['clustering']=labels
df2.head()
```

```
fig, ax = plt.subplots(figsize=(10,10))

df0=df2[df2['clustering']==0]
df0.plot.scatter(x='개업률',y='폐업률',ax=ax,color='RED')

df0=df2[df2['clustering']==1]
df0.plot.scatter(x='개업률',y='폐업률',ax=ax,color='Green')

df0=df2[df2['clustering']==2]
df0.plot.scatter(x='개업률',y='폐업률',ax=ax,color='Blue')


for ind in df2.index:
    ax.annotate(df2.loc[ind]['행정구'], (df2.loc[ind]['개업률'], df2.loc[ind]['폐업률']))
```

```
m_result3 = folium.Map(location=korea_location,
                zoom_start=10.5)

folium.Choropleth(
    geo_data=geo_json,
    name='choropleth',
    data=df2,
    columns=['행정구','clustering'],
    key_on= 'feature.properties.name',
    fill_color='RdBu',
    fill_opacity=0.9,
    line_opacity=0.7).add_to(m_result3)

m_result3
```