데이터분석을 통한

따릉이 적자문제 해결방안

따릉플러스

중앙대학교

중앙대학교 응용통계학과
CHUNG-ANG UNIVERSITY APPLIED STATISTICS

김소은 20196981
김서린 20191376
신정아 20194354
이윤지 20190765

목차

0 서론

1 . 분석 배경

2 . 분석 목표

1 광고

1. 변수 파악
2. 변수간 상관관계 분석
3. 적정한 군집 수 정하기
4. 군집분석_k-means
5. 군집별 특징 파악
6. 군집별 광고 타겟팅

2 재배치

1. 재배치 자치구 선정
2. 재배치 수행 날짜 / 시간대 선정
3. 군집분석_k-means clustering
4. TSP 알고리즘
5. 비용함수
6. 재배치 결결과

3 결론

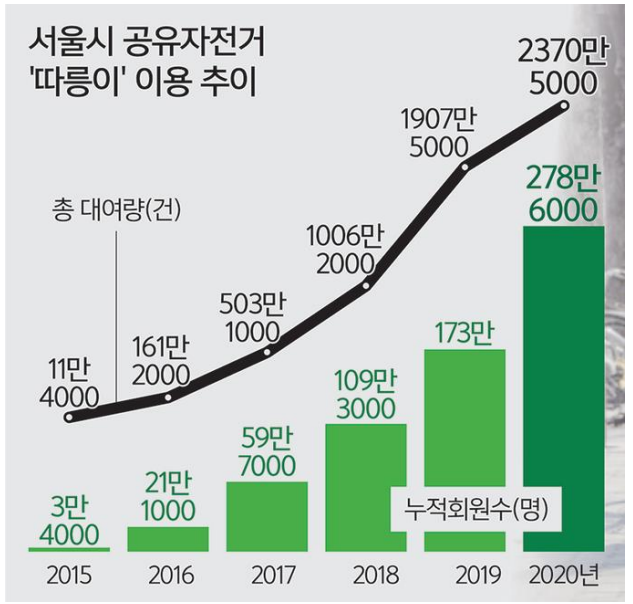
1. 군집별 광고 마켓팅 결론 및 한계
2. 자전거 재배치 효율화 결론 및 한계

0. 서론

0.1 분석 배경

따릉이 이용현황 및 문제제기

서울자전거 SEOUL BIKE 따릉이



- 서울시 대표 공공 사업
- 코로나19 이후 비대면 교통수단으로 부상

매년 따릉이 이용량 꾸준히 증가, 특히 코로나 이후로 크게 증가

But 운영비용 많이 들어서 적자가 큰 문제

지속가능한 사업이 되기 위해선 적자폭 해소가 필수적

0.1 분석 배경

따릉이 적자폭 해소 방안 아이디어

1. 요금인상 - 서울시, 요금인상 계획 없음 ✕

따릉이, 인기만큼 늘어가는 '적자'… 서울시 반응은?

심혁주 기자 | 조회수 : 21,808 | 입력 : 2018.09.29 07:18

서울시 자전거교통과 관계자는 요금인상에 대해 "전혀 생각하고 있지 않다"며 "요금인상은 아주 민감한 문제라 현재로선 검토단계도 아니다. 공공성을 최우선으로 두고 있다"고 전했다. 시는 따릉이 사업이 수익사업이 아닌 만큼 공공성을 위해 어느 정도의 적자는 감수해야 한다는 입장이다.

◆광고유치 '신중', 요금인상 '검토 안해'

적자폭 해소를 위한 대안은 크게 두가지로 꼽힌다. 광고유치와 사용료 인상. 하지만 두 방법 모두 현실적으로 도입이 쉽지 않다. 시는 올해 초 적자해소 방안으로 광고사업자를 뽑으려 했지만 기업들이 신규광고를 망설이며 한차례 무산된 바 있다.

2. 광고유치 - 서울시, 광고운영 검토 ○

서울시, '연 100억 적자' 따릉이에 광고 운영 검토

입력 : 2021.06.29 09:59:23 수정 : 2021.06.29 10:02:12 0

서울시, '100억 적자' 줄일 방안 모색중

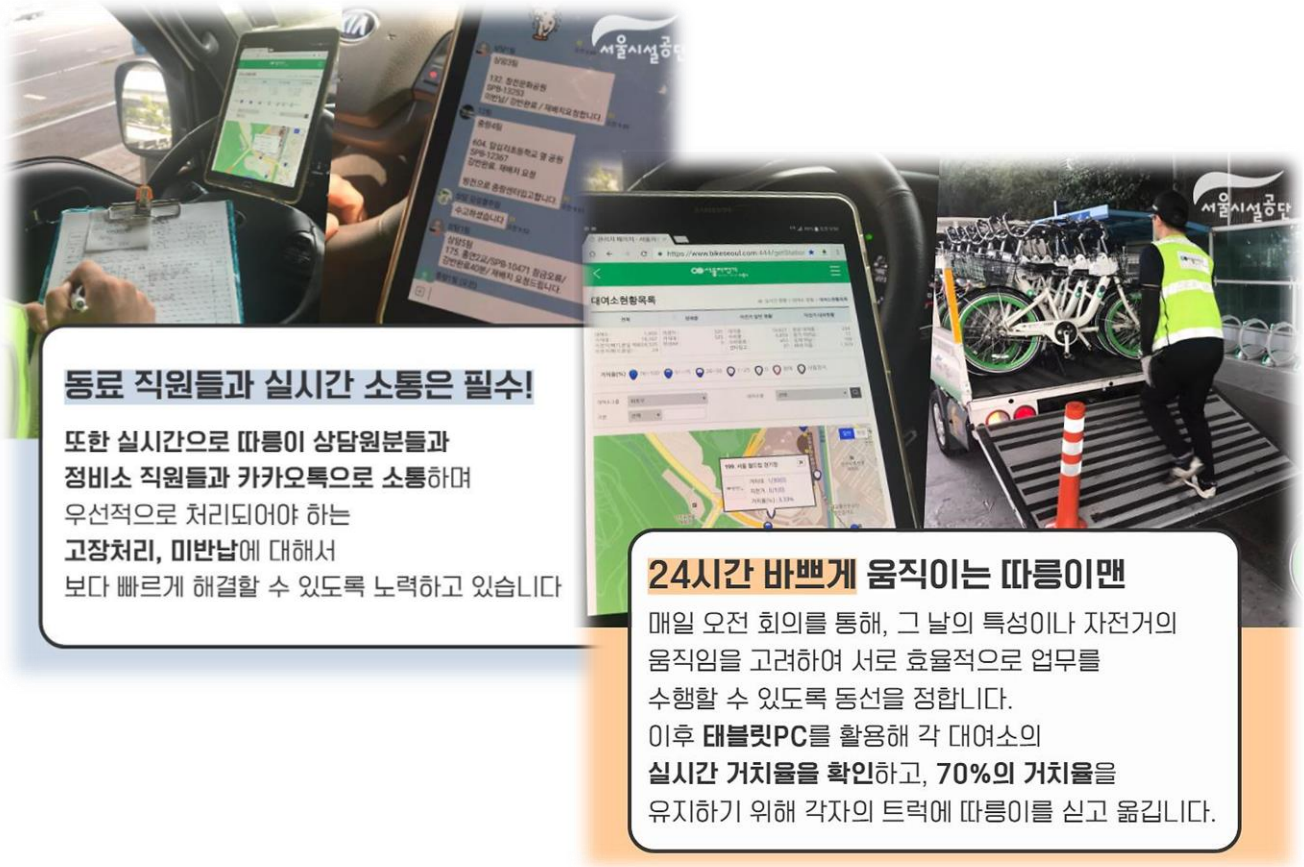
당초 따릉이 확대 중단 논의가 나온 건 매년 커지는 적자 때문이다. 따릉이는 올해 5월 이용자 300만명을 돌파했지만 2017년 42억원, 2018년 67억원, 2019년 89억원, 작년 100억원 등 적자 규모가 갈수록 커지고 있다. 이미 경기 수원시와 고양시 등 공공자전거 사업을 접은 지자체도 있다. 다만 따릉이가 공공서비스 성격이 강한 만큼 시는 손실을 줄여가며 따릉이를 계속 운영해나가겠단 방침이다.

따릉이에 광고판을 다는 방안 등도 검토되고 있다.

∴ 실현 가능한 대안 → **기업에 투자 받아 광고사업**

0.1 분석 배경

따릉이 재배치 현황



현재 정해진 경로 없이

실시간으로 거치율 확인 후 재배치

문제점

→ 따릉이맨 과도한 업무량
→ 많은 인건비 필요
→ 최적의 경로 판단 불가할 수도

정해진 재배치 시스템

기대효과

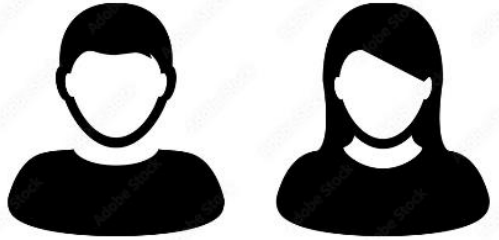
→ 인건비 절감
→ 효율적 재배치 가능

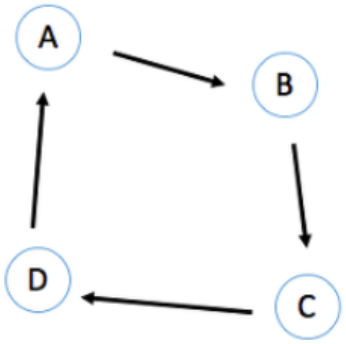
0.2 분석목표

따릉이 적자 감소를 위한 2가지 방안

1. 따릉이 대여시설 광고사업

- 서울시 전체 대여소를 대상으로 led 광고판 설치 가정
- **사용자 맞춤광고**로 광고수익 극대화
- 대여소 **군집분석**을 통한 타겟층 선정





2. 자전거 효율적 재배치 경로 설정

- TSP 알고리즘을 통한 최적 경로 모색
- **재배치 비용이 최소**가 되는 경로 선택

1. 광고 사업

1.1 데이터 전처리 후 변수 파악

대여시간 변수

22시~03시 : 1

04시~09시 : 2

10시~15시 : 3

16시~21시 : 4

휴일여부 변수

주말 : 1

평일 : 2

성별 변수

sex_0 : 남자

sex_1 : 여자

연령대 변수

age_10 : 10대 이하

age_20 : 20대

age_30 : 30대

age_40 : 40대

age_50 : 50대

age_60 : 60대

age_70 : 70대 이상

사용시간 변수

usetime_1: 0~10분

usetime_2: 11~21분

usetime_3: 22~47분

usetime_4: 48~103분

usetime_5: 103~400분

usetime_6: 401~999분

대여소번호

2114개

	대여소번호	휴일여부	대여시간	sex_0	sex_1	age_10	age_20	age_30	age_40	age_50	age_60	age_70	usetime_1	usetime_2	usetime_3	usetime_4	usetime_5	usetime_6
1	3	1	1	11	0	0	1	10	0	0	0	0	10	1	0	0	0	0
2	3	1	2	9	0	0	0	9	0	0	0	0	0	9	0	0	0	0
3	3	1	3	9	0	0	0	9	0	0	0	0	0	9	0	0	0	0
4	3	1	4	18	0	0	0	18	0	0	0	0	0	18	0	0	0	0
5	3	2	1	32	0	0	1	28	0	3	0	0	29	2	1	0	0	0
6	3	2	2	49	0	0	38	9	0	2	0	0	2	10	0	19	18	0
7	3	2	3	457	10	0	385	45	1	36	0	0	122	75	124	92	54	0
8	3	2	4	340	0	0	10	91	18	221	0	0	120	80	30	110	0	0

1.2 변수간 상관관계 분석

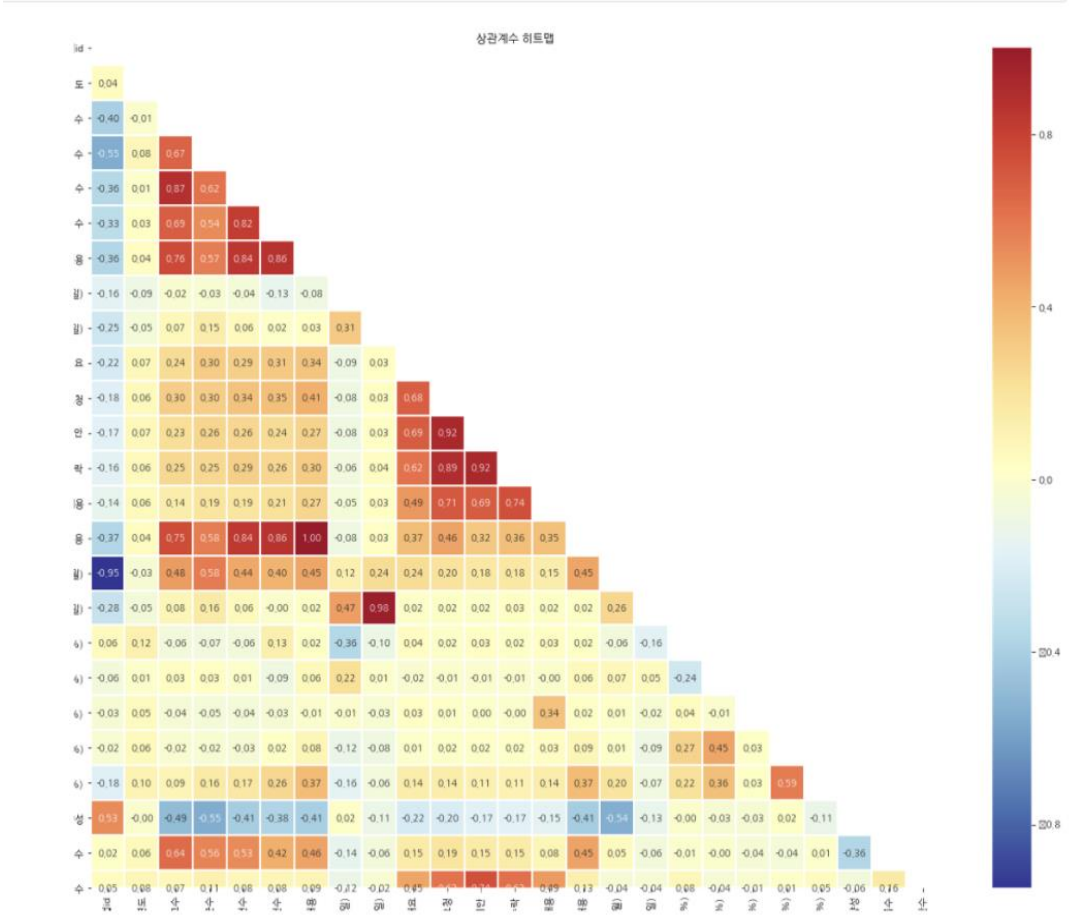
히트맵이란?

: 변수들간의 상관계수를 직관적으로 시각화 한 그래프

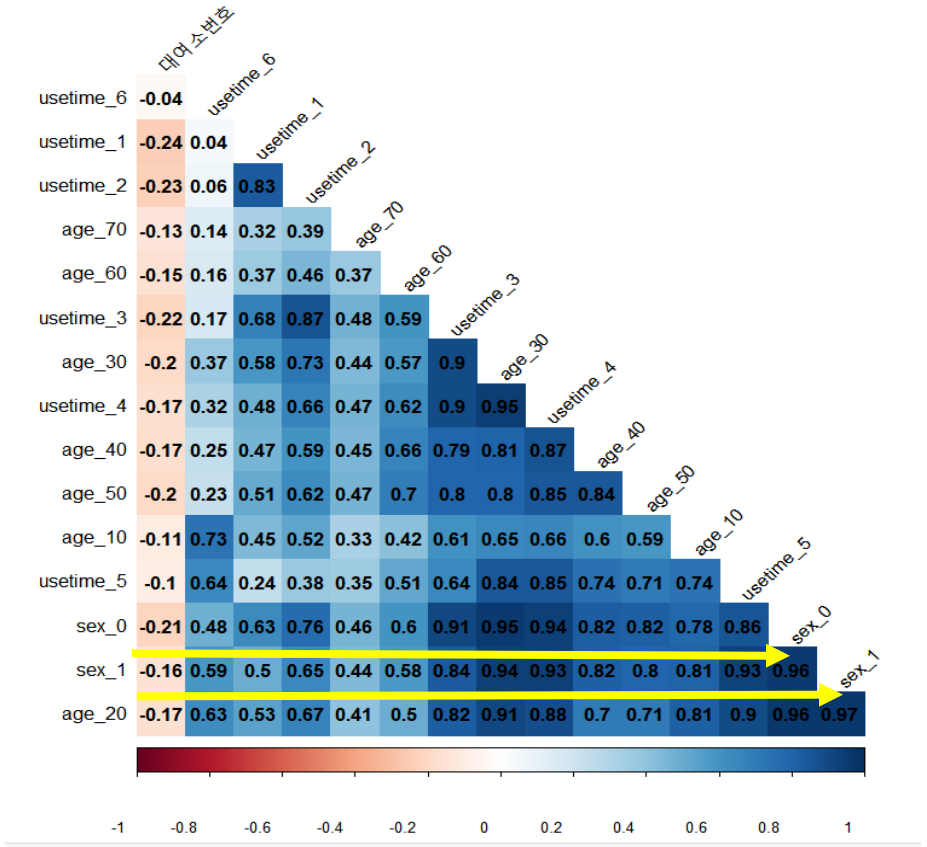
- 특히 높은 상관을 보이는 변수 묶음이 있는지 확인

- 묶음별로(성별, 나이, 사용시간) 상대적으로 중요도가 낮다고 판단되는 변수는 분석에서 제외 및 병합

=> **군집분석의 변별력을 높이기 위해 변수 줄이기**

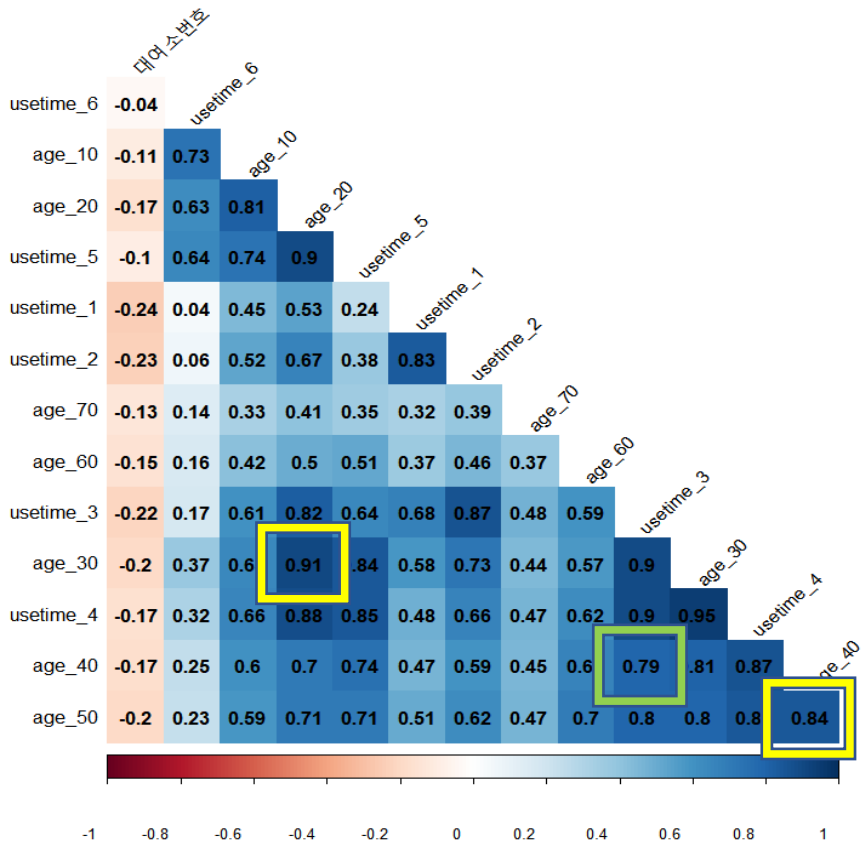


1.2 변수간 상관관계 분석



0.7 이상은 강한 상관관계를 가지고 있다고 판단

⇒ **성별 변수 삭제**



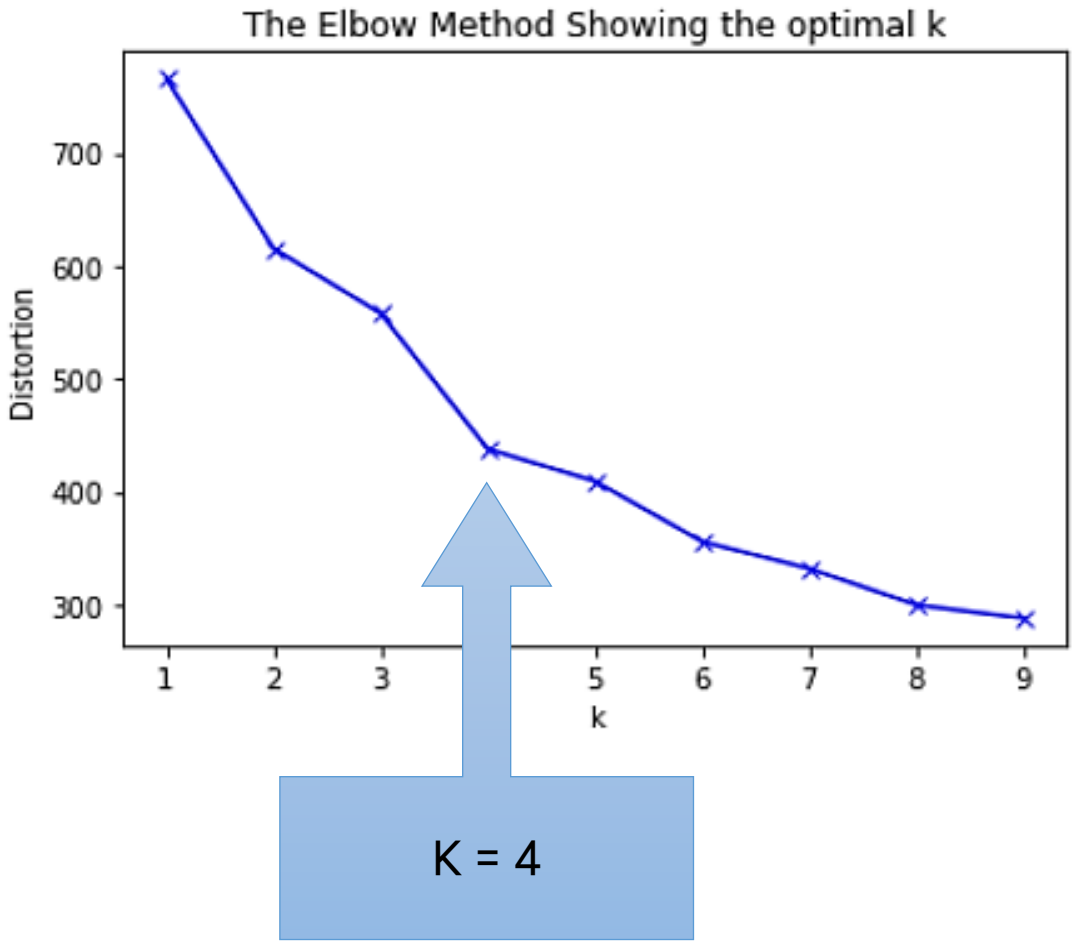
- 20대 & 30대 : 0.91
- 40대 & 50대 : 0.84
- 사용시간3 & 사용시간4 : 0.7

→ **변수병합**

1.3 적정한 군집 수 정하기

Elbow method

: Cluster 간의 거리의 합을 나타내는 inertia가 급격히 떨어지는 구간이 생기는데, 이 지점의 K 값을 군집의 개수로 사용



1.4 군집분석

군집화란?

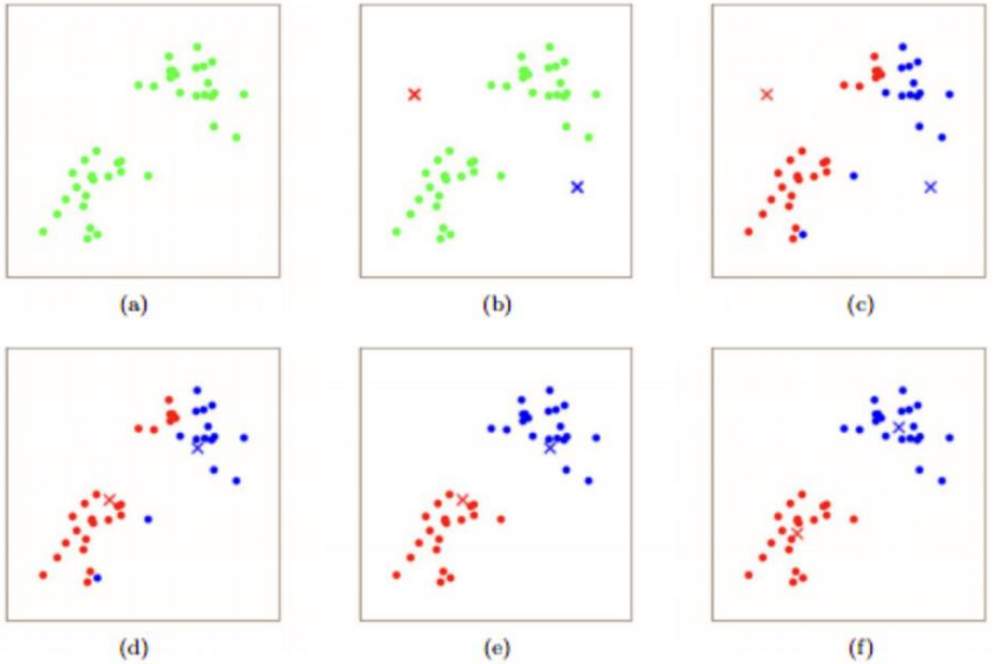
: 서로 유사한 데이터들은 같은 그룹으로,
서로 유사하지 않은 데이터는 다른 그룹으로 분리하는 것

k-means 군집화의 원리

'K'는 데이터 세트에서
찾을 것으로 예상되는 클러스터(군집) 수

'Means'는 각 데이터로부터
그 데이터가 속한 클러스터의 중심까지의 평균 거리

* 목표: 평균 거리 최소화



1.4 군집분석

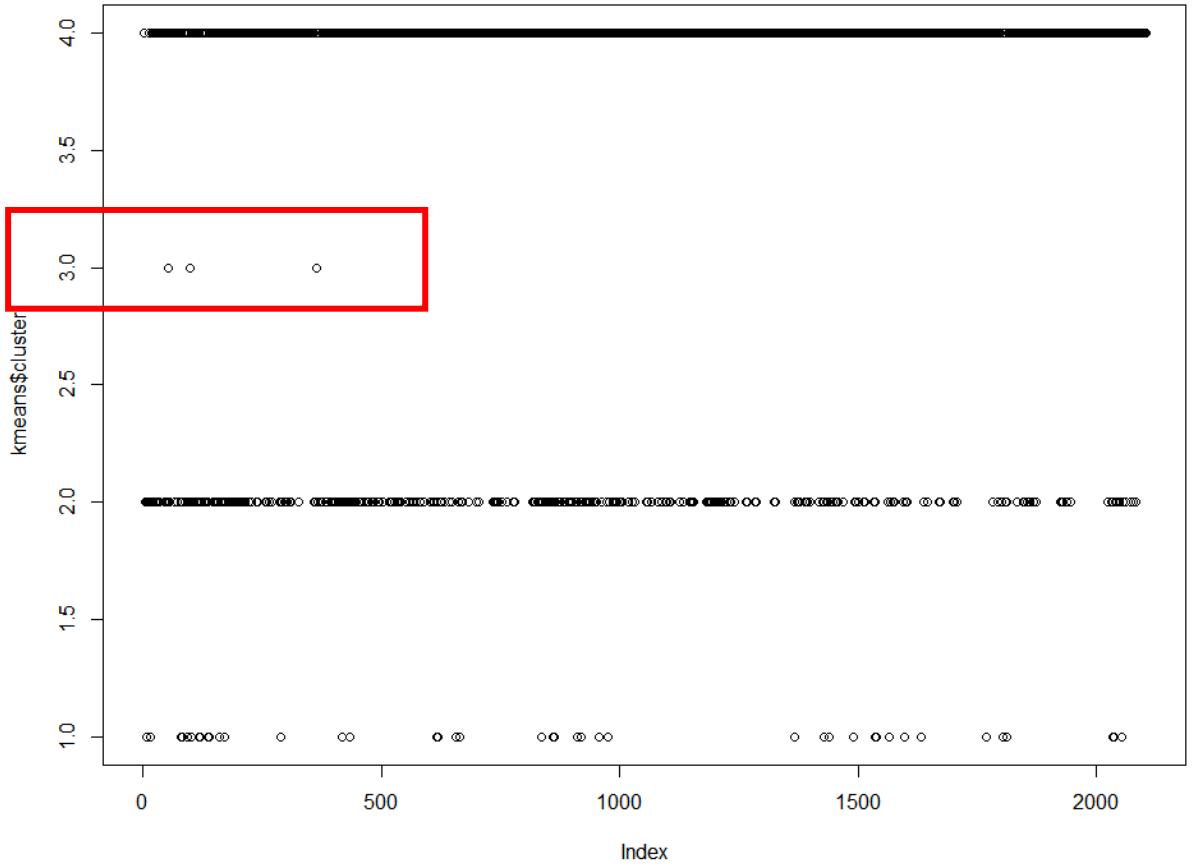
군집 확인

대여소 수가 아주 작은 군집을

이상치로 판단 후 제거

→ 3번째 군집제거

→ 3개의 군집으로 군집화

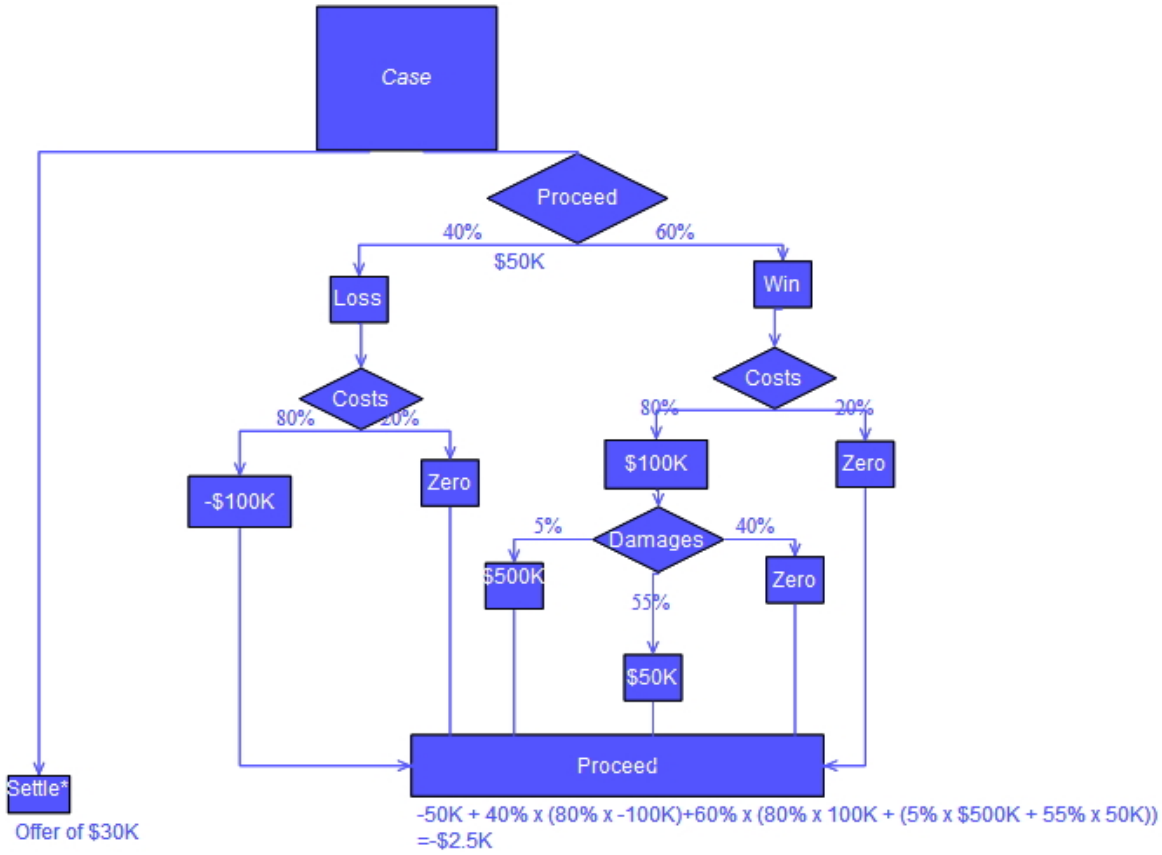


1.5 군집별 특징 파악

Decision Tree(결정트리)란?

- 분류(Classification)와 회귀(Regression) 모두 가능한 지도 학습 모델 중 하나
- 특정 기준에 따라 데이터를 구분하는 모델로 한번의 분기 때마다 변수 영역을 두 개로 구분

→ 군집분석의 결과를 Decision Tree를 사용해 **군집별 특징 파악**

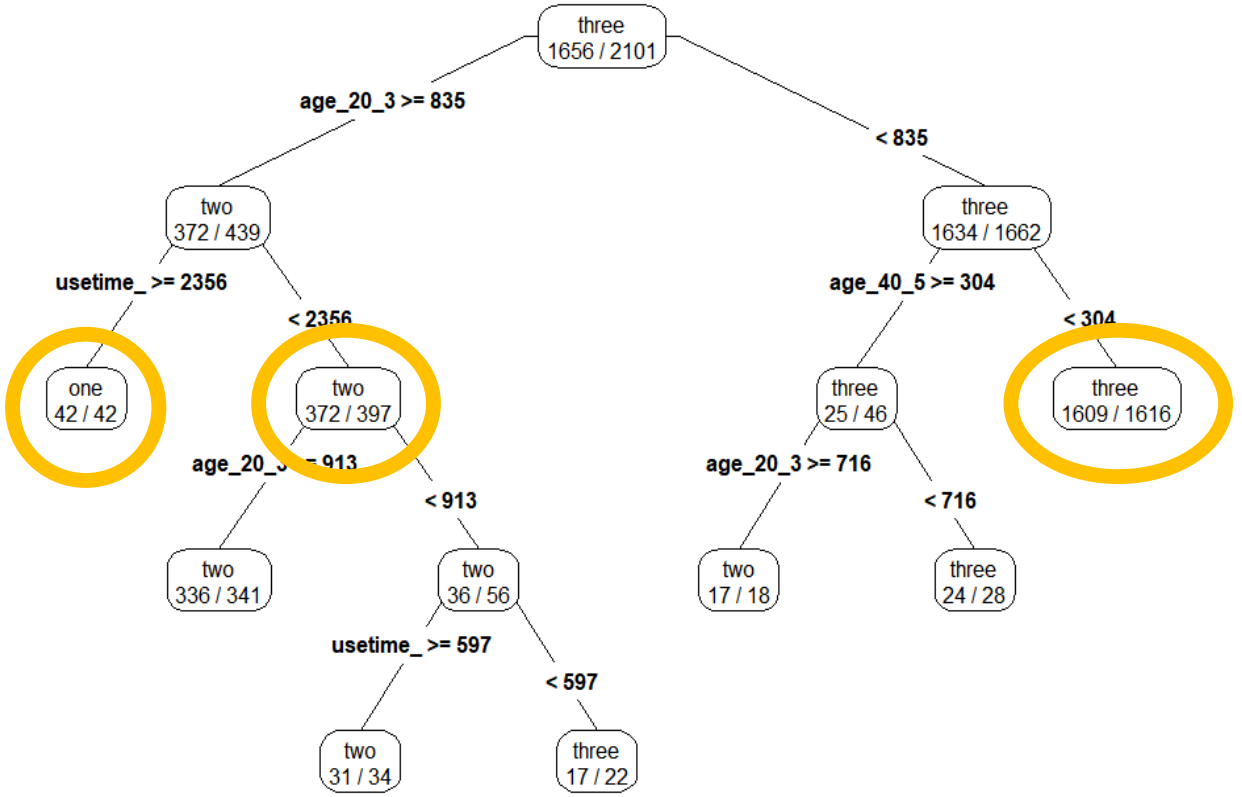


1.5 군집별 특징 파악

One
20,30대 연령층의 사용이 많은 편,
22분 이상 103분 이하
이용하는 사람이 많은 군집

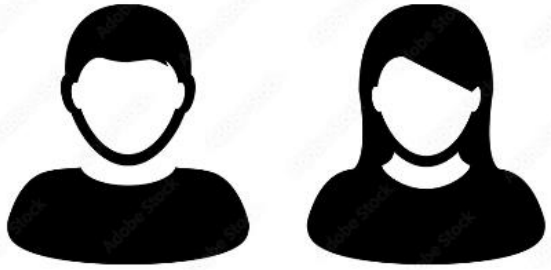
Two
20,30대 연령층의 사용이 많고,
22분 이상 103분 이하
사용하는 사람이 적은 군집

Three
20,30대 연령층의 사용이 적고,
40,50대 연령층의 사용이 많은 군집



age_20_30 : 20, 30대
age_40_50 : 40, 50대
usetime_3_4: 사용시간 22분 이상 103분 이하

1.6 군집별 광고 타겟팅



One & Two

- **남녀 상관없이 2030대를 위한 상품 광고**
  - 따릉이 이용시간이 긴 것을 고려해 운동용품 또는 간편 먹거리 등 제안

Three

- 2030대를 제외한, **4050대를 위한** 상품광고, 제약광고, 아웃도어광고 등 제안

2. 재배치

#1 재배치 자치구 선정

2.1 재배치 자치구 선정

- 서울시 자치구 중 최적 재배치를 수행할 구 선정
- 선정기준: 가장 재배치가 비효율적인 구
- ‘일별 반납/대여’ 의 분산 값이 크면

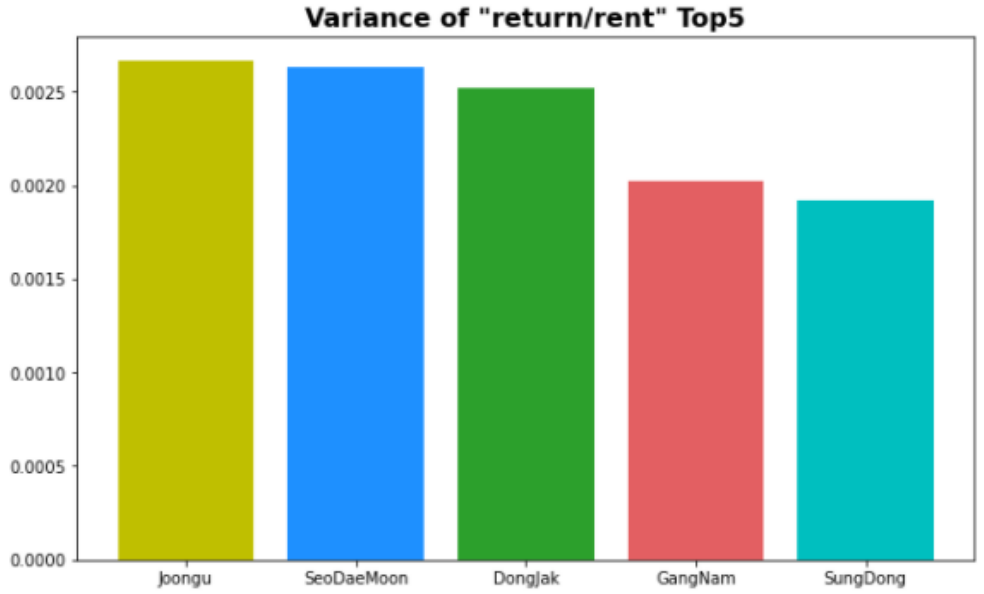
  자전거 배치가 불균형/비효율적 이라고 판단
- 2019 1월~ 2021 6월 대여반납 데이터 이용
- 자치구 일별 반납/대여 계산한 열 추가

	자치구	일시_day	대여수	반납수	반납/대여
0	강남구	2019-01-01	322	304	0.944099
1	강남구	2019-01-02	602	582	0.966777
2	강남구	2019-01-03	669	627	0.937220
3	강남구	2019-01-04	658	633	0.962006
4	강남구	2019-01-05	458	439	0.958515
...	...	...	...	...	...
22770	중랑구	2021-06-26	2857	2871	1.004900
22771	중랑구	2021-06-27	2760	2773	1.004710
22772	중랑구	2021-06-28	2679	2841	1.060470
22773	중랑구	2021-06-29	2646	2740	1.035525
22774	중랑구	2021-06-30	3006	3158	1.050566

22775 rows × 5 columns

2.1 재배치 자치구 선정

자치구	반납/대여_var	자치구
강남구	0.002023	강남구
강동구	0.000845	강동구
강북구	0.001042	강북구
강서구	0.000374	강서구
관악구	0.000981	관악구
광진구	0.000501	광진구
구로구	0.000782	구로구
금천구	0.001819	금천구
노원구	0.000388	노원구
도봉구	0.001738	도봉구
동대문구	0.001339	동대문구
동작구	0.002518	동작구
마포구	0.001324	마포구
서대문구	0.002632	서대문구
서초구	0.000951	서초구
성동구	0.001918	성동구
성북구	0.001441	성북구
송파구	0.000259	송파구
양천구	0.000908	양천구
영등포구	0.001249	영등포구
용산구	0.001575	용산구
은평구	0.001487	은평구
종로구	0.001715	종로구
중구	0.002663	중구
중랑구	0.001417	중랑구



‘일별 반납/대여’ 의 분산값이 가장 큰 자치구 TOP 5

→ *중구, 서대문구, 동작구, 강남구, 성동구*

‘중구’ 선택

#2 재배치 수행 날짜/시간대 선정

2.2 재배치 수행 날짜/시간대 선정

고려사항 1. 비 오는 날 따릉이 이용량

감소를 고려해 맑은 날 설정

→ 기상청에 강수량 0.0mm인

가장 최신 날짜

고려사항 2. 주중과 주말의

따릉이 이용 양상이 다를 것으로 예상

→ 주중과 주말 두 일자 선택

2021년 4월 23일,
2021년 4월 24일

날씨달력 서울(유)/ 2021년 4월

일요일	월요일	화요일	수요일	목요일	금요일	토요일
				1일	2일	3일
				평균기온:17.7℃ 최고기온:24.0℃ 최저기온:12.0℃ 평균운량:5.3 일강수량: -	평균기온:17.8℃ 최고기온:23.0℃ 최저기온:12.5℃ 평균운량:8.1 일강수량:0.0mm	평균기온:14.9℃ 최고기온:18.0℃ 최저기온:10.9℃ 평균운량:9.9 일강수량:56.2mm
4일	5일	6일	7일	8일	9일	10일
평균기온:11.9℃ 최고기온:15.8℃ 최저기온:9.0℃ 평균운량:6.3 일강수량:0.6mm	평균기온:11.9℃ 최고기온:19.3℃ 최저기온:5.4℃ 평균운량:2.3 일강수량: -	평균기온:13.9℃ 최고기온:19.2℃ 최저기온:9.4℃ 평균운량:2.4 일강수량: -	평균기온:14.1℃ 최고기온:20.4℃ 최저기온:8.1℃ 평균운량:0.0 일강수량: -	평균기온:13.0℃ 최고기온:18.0℃ 최저기온:7.2℃ 평균운량:0.0 일강수량: -	평균기온:13.6℃ 최고기온:19.6℃ 최저기온:7.6℃ 평균운량:0.6 일강수량: -	평균기온:12.6℃ 최고기온:18.8℃ 최저기온:7.7℃ 평균운량:0.5 일강수량: -
11일	12일	13일	14일	15일	16일	17일
평균기온:15.0℃ 최고기온:21.8℃ 최저기온:7.5℃ 평균운량:3.5 일강수량: -	평균기온:13.8℃ 최고기온:17.8℃ 최저기온:11.9℃ 평균운량:9.8 일강수량:22.0mm	평균기온:10.4℃ 최고기온:13.5℃ 최저기온:5.7℃ 평균운량:6.6 일강수량:8.0mm	평균기온:8.4℃ 최고기온:13.3℃ 최저기온:3.1℃ 평균운량:0.3 일강수량: -	평균기온:11.6℃ 최고기온:18.3℃ 최저기온:6.3℃ 평균운량:4.9 일강수량: -	평균기온:11.3℃ 최고기온:14.8℃ 최저기온:8.1℃ 평균운량:6.9 일강수량:0.7mm	평균기온:9.6℃ 최고기온:14.6℃ 최저기온:6.0℃ 평균운량:3.8 일강수량:0.0mm
18일	19일	20일	21일	22일	23일	24일
평균기온:9.9℃ 최고기온:15.3℃ 최저기온:5.6℃ 평균운량:0.9 일강수량: -	평균기온:12.2℃ 최고기온:19.4℃ 최저기온:5.8℃ 평균운량:0.0 일강수량: -	평균기온:14.8℃ 최고기온:22.0℃ 최저기온:8.1℃ 평균운량:2.9 일강수량: -	평균기온:19.1℃ 최고기온:28.2℃ 최저기온:10.3℃ 평균운량:3.9 일강수량: -	평균기온:20.9℃ 최고기온:27.5℃ 최저기온:16.1℃ 평균운량:8.9 일강수량: -	평균기온:19.0℃ 최고기온:21.7℃ 최저기온:17.0℃ 평균운량:9.6 일강수량:0.0mm	평균기온:18.3℃ 최고기온:22.2℃ 최저기온:15.3℃ 평균운량:6.8 일강수량:0.0mm
25일	26일	27일	28일	29일	30일	
평균기온:18.2℃ 최고기온:24.0℃ 최저기온:12.6℃ 평균운량:1.1 일강수량: -	평균기온:16.2℃ 최고기온:20.6℃ 최저기온:12.3℃ 평균운량:4.5 일강수량: -	평균기온:15.4℃ 최고기온:20.0℃ 최저기온:12.5℃ 평균운량:9.4 일강수량:1.7mm	평균기온:14.5℃ 최고기온:18.2℃ 최저기온:11.2℃ 평균운량:7.9 일강수량:4.7mm	평균기온:14.6℃ 최고기온:20.1℃ 최저기온:10.9℃ 평균운량:7.6 일강수량:3.9mm	평균기온:11.3℃ 최고기온:14.6℃ 최저기온:8.6℃ 평균운량:9.5 일강수량:26.3mm	

2.2 재배치 수행 날짜/시간대 선정

고려사항 3. 중구의 대여수와 반납수의

차이가 큰 시간대를 재배치 필요성이

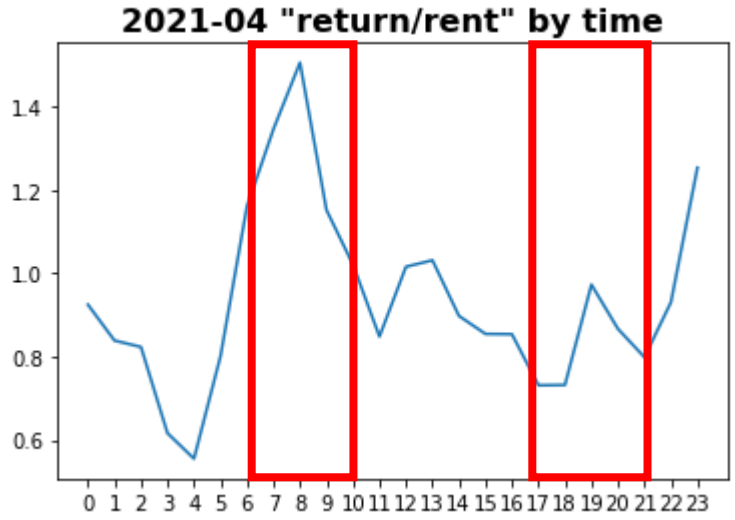
높은 시간대라고 판단

→ 2021년 4월 '평균 반납/대여의 절댓값'이

큰 시간대 선별

반납이 많은 **6시~10시**
대여가 많은 **17시~21시**

	시간대	반납수	대여수	반납/대여
0	0	753	814	0.925061
1	1	531	633	0.838863
2	2	443	538	0.823420
3	3	334	542	0.616236
4	4	293	528	0.554924
5	5	362	452	0.800885
6	6	784	675	1.161481
7	7	2526	1875	1.347200
8	8	5551	3684	1.506786
9	9	3140	2724	1.152717
10	10	2297	2249	1.021343
11	11	2969	3499	0.848528
12	12	3566	3509	1.016244
13	13	3571	3461	1.031783
14	14	3400	3786	0.898045
15	15	3714	4345	0.854776
16	16	4163	4874	0.854124
17	17	4859	6644	0.731337
18	18	6260	8551	0.732078
19	19	3938	4046	0.973307
20	20	2634	3038	0.867018
21	21	2283	2858	0.798810
22	22	2141	2300	0.930870
23	23	1356	1081	1.254394



2.2 재배치 수행 날짜/시간대 선정

재배치 수행 일시

(1) 2021-04-23 / 6시~10시
(2) 2021-04-23 / 17시~21시
(3) 2021-04-24 / 6시~10시
(4) 2021-04-24 / 17~21시

4개의 재배치 수행 일시에
해당되는 행을 추출하여
각 일시마다 분석

	반납일시	대여소번호	자치구	반납일시_hour	반납일시_date
659892	2021-04-23 08:16:21	475	중구	8	2021-04-23
659893	2021-04-23 08:43:07	475	중구	8	2021-04-23
659894	2021-04-23 08:46:06	475	중구	8	2021-04-23
659895	2021-04-23 08:51:53	475	중구	8	2021-04-23
659896	2021-04-23 08:56:32	475	중구	8	2021-04-23
...	...	...	...	...	...
2974337	2021-04-23 07:54:27	4760	중구	7	2021-04-23
2974338	2021-04-23 08:49:51	4760	중구	8	2021-04-23
2981851	2021-04-23 09:22:07	4766	중구	9	2021-04-23
2981852	2021-04-23 10:47:43	4766	중구	10	2021-04-23
2987576	2021-04-23 08:54:07	381	중구	8	2021-04-23

605 rows × 5 columns

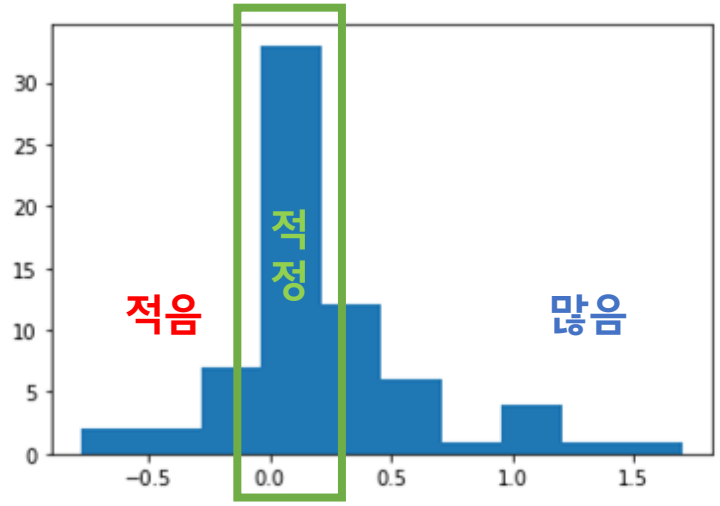
#3 K-Means Clustering

2.3 K-Means clustering

Step 1) group 분류

- 각 대여소를 반납수, 대여수, 거치대수를 이용해 group(적음, 적정, 많음)으로 분류
- 반납수와 대여수가 비슷하면 0에 가까움

```
(array([ 2.,  2.,  7., 33., 12.,  6.,  1.,  4.,  1.,  1.]),
 array([-0.77777778, -0.53      , -0.28222222, -0.03444444,  0.21333333,
         0.46111111,  0.70888889,  0.95666667,  1.20444444,  1.45222222,
         1.7       ]),
 <BarContainer object of 10 artists>)
```



기준	group
$-0.0344 < \frac{(\text{반납수}-\text{대여수})}{\text{거치대수}} < 0.2133$	적정
$\frac{(\text{반납수}-\text{대여수})}{\text{거치대수}} \leq -0.0344$	적음
$0.2133 \leq \frac{(\text{반납수}-\text{대여수})}{\text{거치대수}}$	많음

2.3 K-Means clustering

Step 1) group 분류

- 각 대여소를 반납수, 대여수, 거치대수를 이용해 group(적음, 적정, 많음)으로 분류

	대여소번호	대여수	반납수	거치대수	(반납-대여)/거치대수	위도	경도	group
0	300	17	19	9	0.222222	37.568050	126.969231	많음
1	310	5	13	8	1.000000	37.568878	126.977470	많음
2	318	6	10	12	0.333333	37.568527	126.982552	많음
3	320	8	16	17	0.470588	37.566223	126.983589	많음
4	321	8	12	17	0.235294	37.565464	126.984139	많음
...	...	...	...	...	...	...	...	...
64	4785	2	13	8	1.375000	37.566898	126.987740	많음
65	4791	9	10	10	0.100000	37.561096	126.987465	적정
66	4792	8	6	10	-0.200000	37.562527	126.998314	적음
67	4793	3	10	10	0.700000	37.564308	127.006477	많음
68	4796	6	11	10	0.500000	37.566471	126.979256	많음

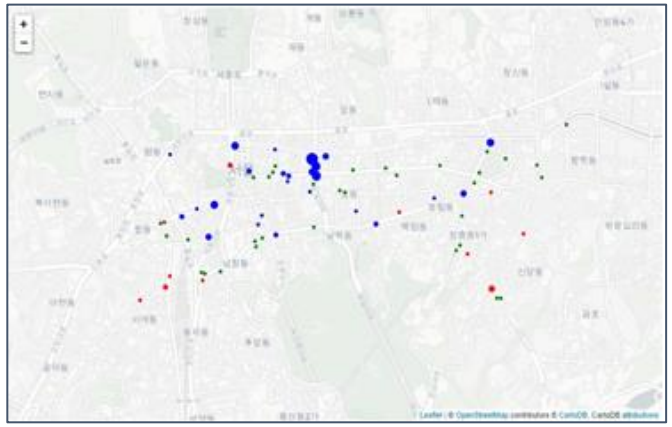
69 rows × 8 columns

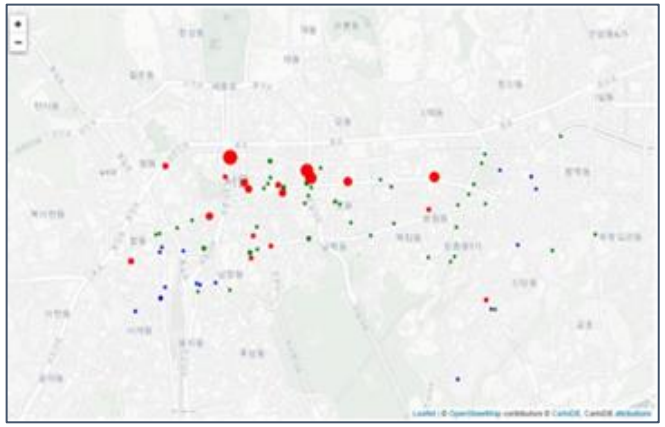
2.3 K-Means clustering

Step 1) group 분류

- group 많음 / 적정 / 적음 대여소 분포 확인

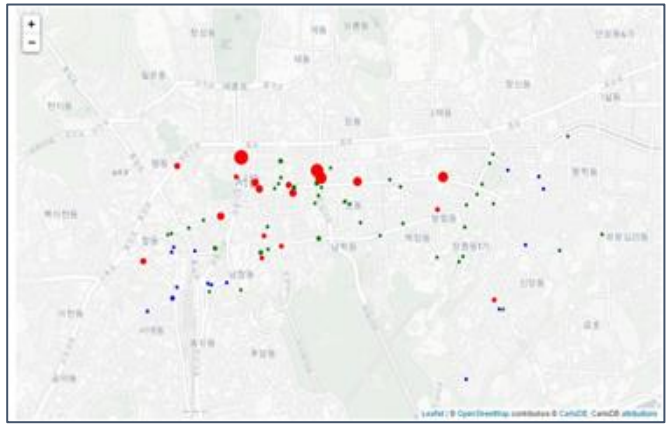
2021-04-23
6시~10시





2021-04-23
17시~21시

2021-04-24
6시~10시



2021-04-24
17시~21시

2.3 K-Means clustering

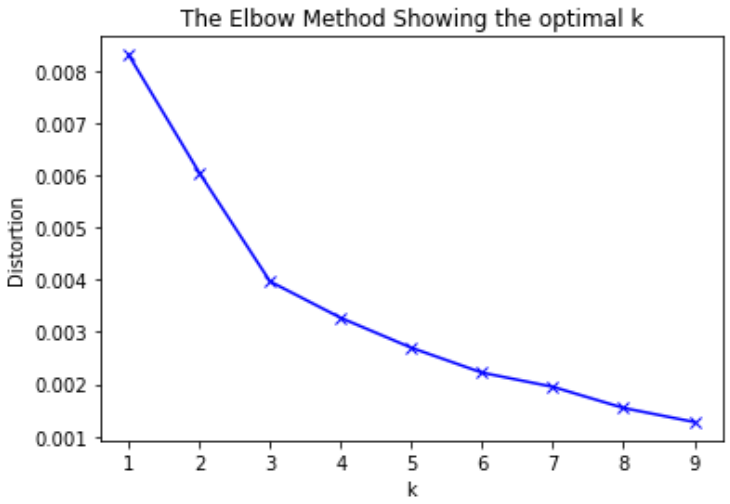
Step 2) 최적 k 선정

- ‘group 많음’, ‘group 적음’인 대여소를 **각각 K-means cluster** (위도와 경도를 이용)

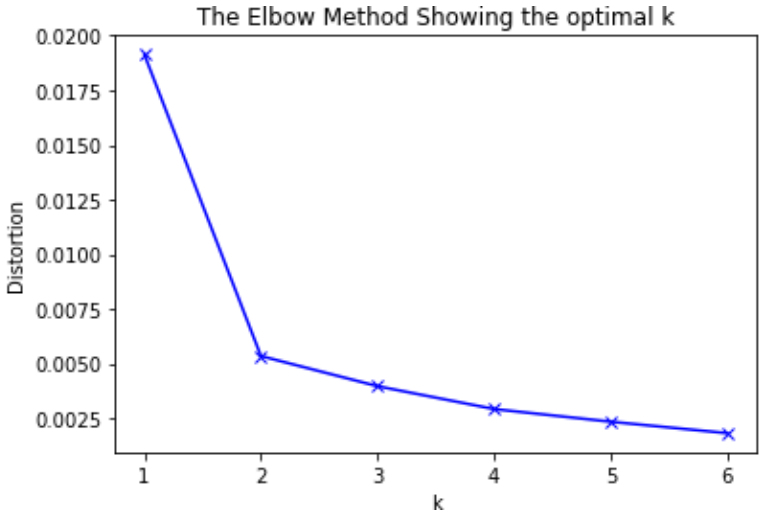
  (‘group 적정’인 곳은 재배치할 필요 없다고 파악 -> 제외)

- **Elbow Method** 활용해 k에 따른 성능 비교 (k=2 또는 k=3 일 때 최적화)

Group 많음



Group 적음

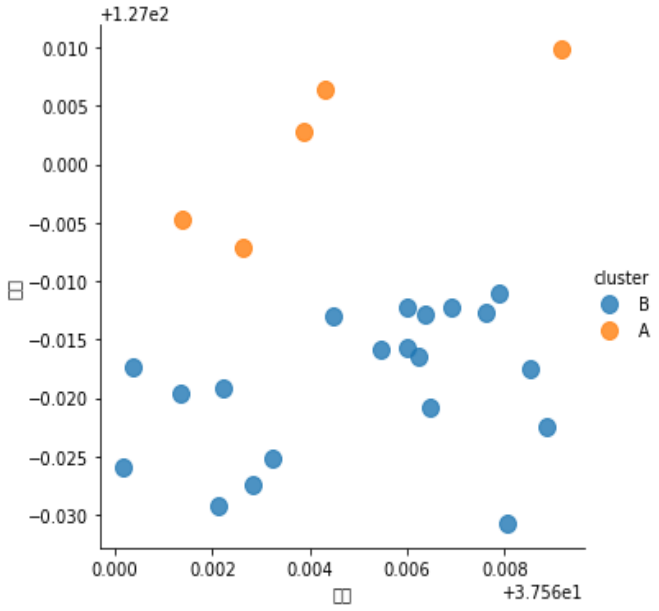


2.3 K-Means clustering

Step 3) k에 따라 K-means clustering 실행 (k=2)

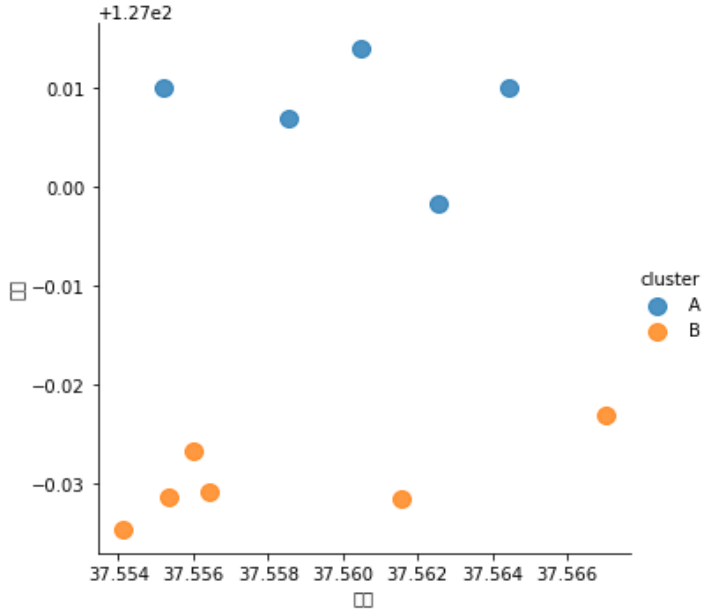
Group 많음

Clustering center : [37.57, 126.98], [37.56, 127.00]



Group 적음

Clustering center : [37.56, 127.01], [37.56, 126.97]

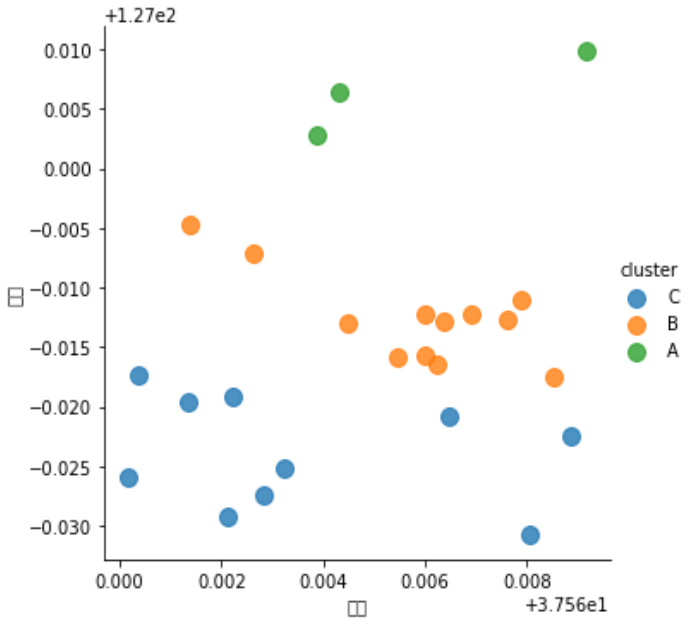


2.3 K-Means clustering

Step 3) k에 따라 K-means clustering 실행 (k=3)

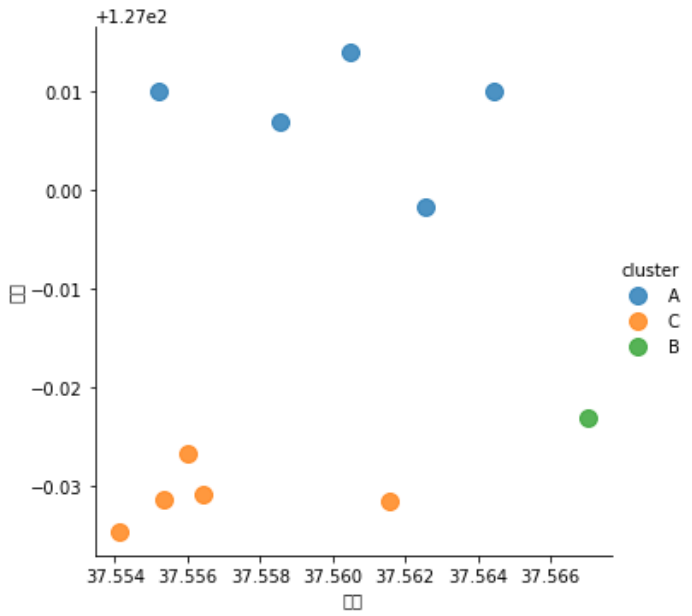
Group 많음

Clustering center : [37.56, 126.98]
[37.57, 127.01]
[37.57, 126.99]



Group 적음

Clustering center : [37.56, 126.97]
[37.56, 127.01]
[37.57 , 126.98]

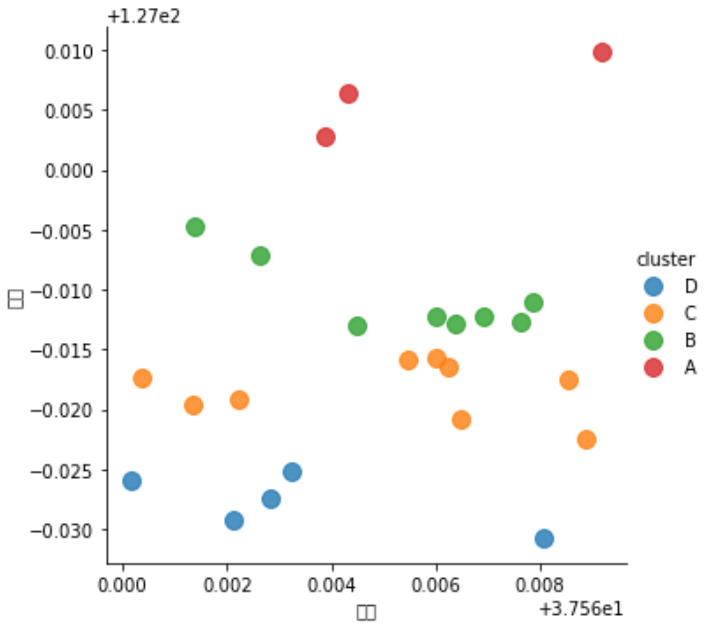


2.3 K-Means clustering

Step 3) k에 따라 K-means clustering 실행 (k=4)

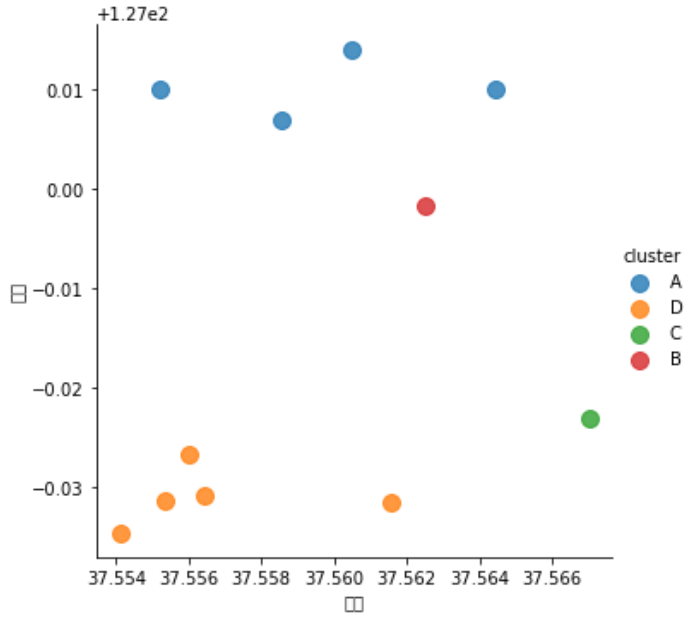
Group 많음

Clustering center : [37.57, 126.98], [37.57, 127.01],
[37.57, 126.99], [37.56, 126.97]



Group 적음

Clustering center : [37.56, 127.01], [37.56, 126.97],
[37.57 , 126.98], [37.56, 127.00]

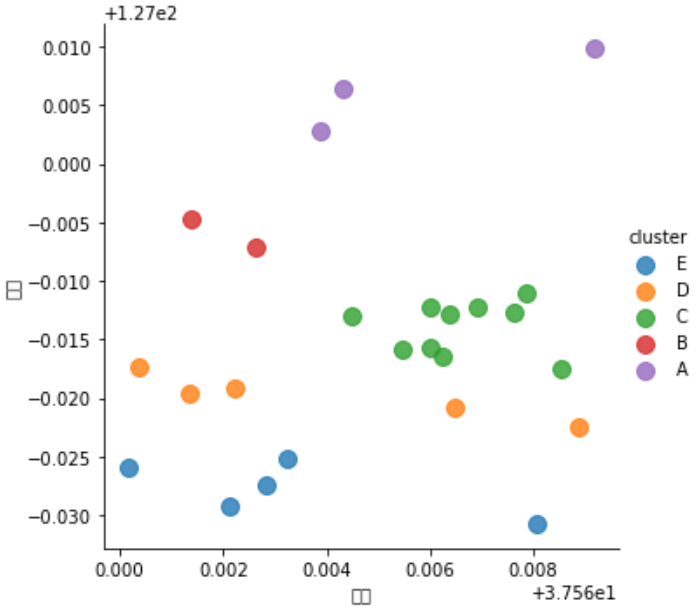


2.3 K-Means clustering

Step 3) k에 따라 K-means clustering 실행 (k=5)

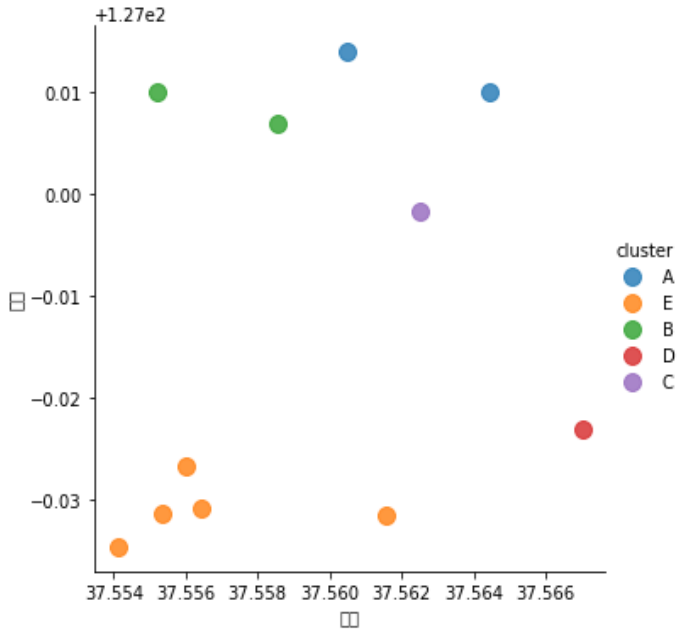
Group 많음

Clustering center : [37.56, 126.99], [37.56, 126.98], [37.57, 126.98], [37.57, 127.01], [37.56, 126.97]



Group 적음

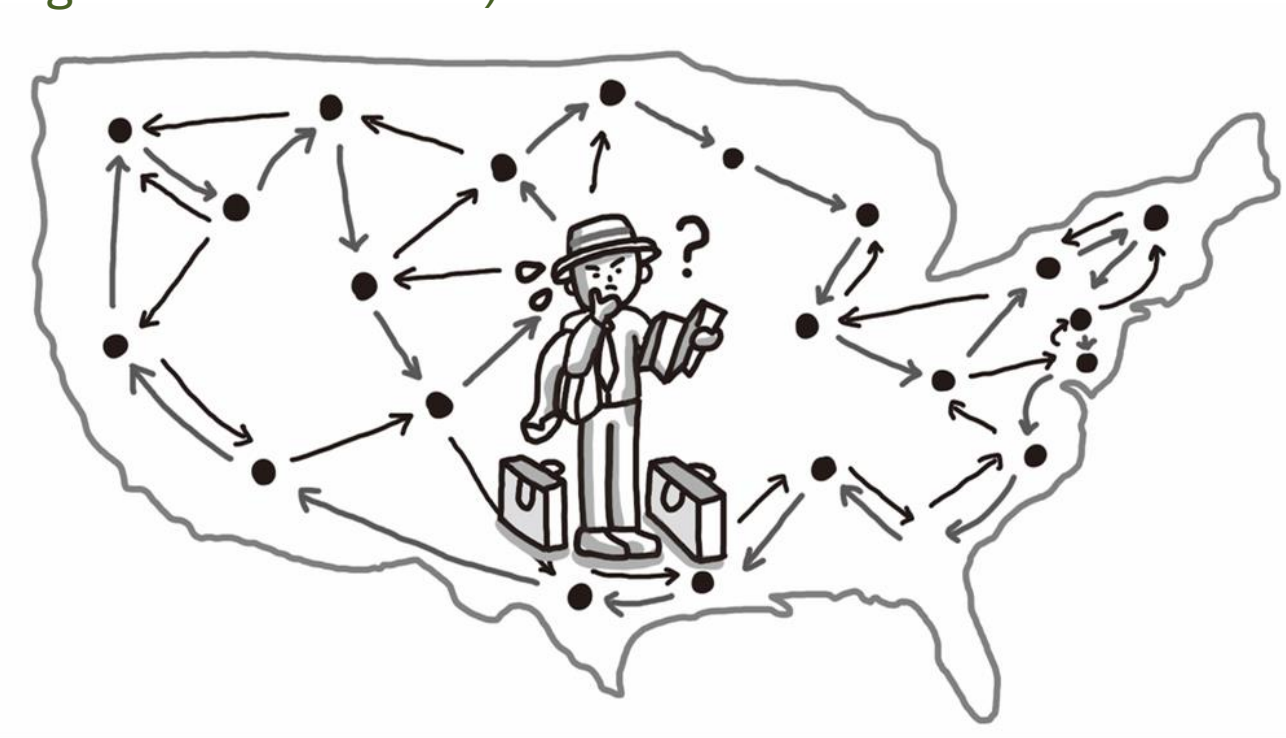
Clustering center : [37.57 , 126.98], [37.56, 127.01], [37.56, 126.97], [37.56, 127.01], [37.56, 127.00]



#4 TSP

2.4 TSP

Step 1) TSP(Traveling Salesman Problem)

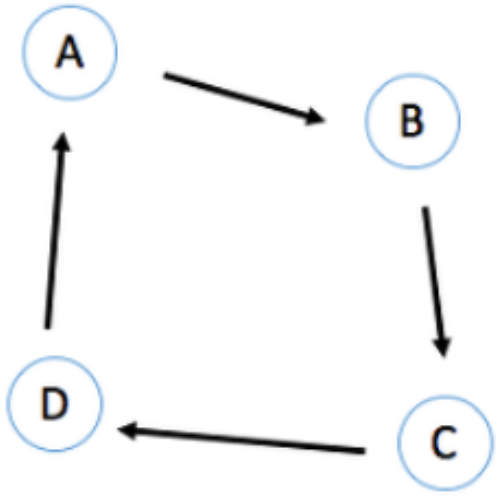


그래프에서 **모든 노드를 한 번만 방문**하고 시작점으로 돌아오는

최소비용의 이동경로를 찾는 알고리즘

2.4 TSP

Step 1) TSP – 완전 탐색 알고리즘

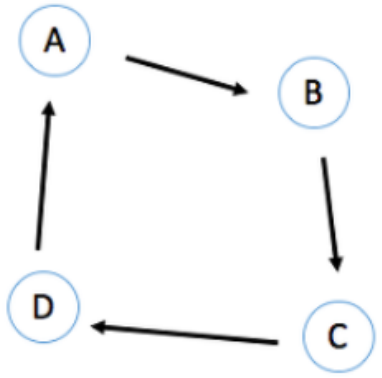


1) A -> B -> C -> D
2) A -> B -> D -> C
3) A -> C -> B -> D
4) A -> C -> D -> B
5) A -> D -> B -> C
6) A -> D -> C -> B
7) B -> A -> C -> D
8) B -> A -> D -> C
9) B -> C -> A -> D
10) B -> C -> D -> A
11) B -> D -> A -> C
12) B -> D -> C -> A
13) C -> A -> B -> D
14) C -> A -> D -> B
15) C -> B -> A -> D
16) C -> B -> D -> A
17) C -> D -> A -> B
18) C -> D -> B -> A
19) D -> A -> B -> C
20) D -> A -> C -> B
21) D -> B -> A -> C
22) D -> B -> C -> A
23) D -> C -> A -> B
24) D -> C -> B -> A

시작점 다르지만 동일한 경로 -> 중복계산 -> **비효율**

2.4 TSP

Step 1) TSP – 동적계획법



- 현재까지 방문한 노드들의 최소비용을 저장하여 중복계산 줄이는 방법
- 현재 노드 -> {현재까지 방문한 노드} ->이동할 수 있는 노드

#1 A -> {A} -> B, C, D

#2 B -> {A, B} -> C, D

#3 C -> {A, C} -> B, D

#4 D -> {A, D} -> B, C

#5 B -> {A, D, B} -> C

#6 C -> {A, D, C} -> B

#7 B -> {A, D, C, B}

(마지막으로 모두 방문했으므로 B->A return 종료)

Bit	…	A	C	AB	AC	AD	ADB	ADC	…
A		#1							
B				#2			#5		
C					#3			#6	
D						#4			

2.4 TSP

Step 2) TSP 전처리 (2021.04.23, 6-10시, k=2, A)

	0	1	2	3	4	5	6	7	8	9
0	0.000000	0.619526	1.546055	1.670230	0.863102	1.036615	1.211413	0.528565	1.259811	1.555098
1	0.619526	0.000000	1.032775	1.216975	0.332498	0.793938	0.643781	0.306942	0.746316	1.060699
2	1.546055	1.032775	0.000000	0.260583	0.707396	1.653431	1.073646	1.330777	0.289929	1.466853
3	1.670230	1.216975	0.260583	0.000000	0.884486	1.887317	1.327623	1.522416	0.482946	1.726993
4	0.863102	0.332498	0.707396	0.884486	0.000000	1.067455	0.701034	0.638573	0.418115	1.158866
5	1.036615	0.793938	1.653431	1.887317	1.067455	0.000000	0.661012	0.570395	1.407974	0.685733
6	1.211413	0.643781	1.073646	1.327623	0.701034	0.661012	0.000000	0.706030	0.884245	0.458443
7	0.528565	0.306942	1.330777	1.522416	0.638573	0.570395	0.706030	0.000000	1.047794	1.026537
8	1.259811	0.746316	0.289929	0.482946	0.418115	1.407974	0.884245	1.047794	0.000000	1.316733
9	1.555098	1.060699	1.466853	1.726993	1.158866	0.685733	0.458443	1.026537	1.316733	0.000000

위도, 경도를 이용해
대여소 간 거리 구하기 (km단위)

	index	key_0	위도_x	경도_x	cluster	대여소번호	대여수	반납수	거치대수	(반납-대여)/거치대수	위도_y	경도_y	group	node_num
0	10	10	37.569183	127.009880	A	346	7	20	12	1.083333	37.569183	127.009880	많음	1
1	23	67	37.564308	127.006477	A	4793	3	10	10	0.700000	37.564308	127.006477	많음	2
2	13	23	37.561390	126.995354	A	390	12	17	12	0.416667	37.561390	126.995354	많음	3
3	9	9	37.562618	126.992836	A	336	7	11	12	0.333333	37.562618	126.992836	많음	4
4	12	17	37.563866	127.002747	A	380	10	13	10	0.300000				5
5	25	15	37.560474	127.014076	A	374	11	10	12	-0.083333	37.560474	127.014076	적음	6
6	27	18	37.558533	127.006989	A	381	2	1	12	-0.083333	37.558533	127.006989	적음	7
7	31	44	37.564430	127.009956	A	476	3	2	10	-0.100000	37.564430	127.009956	적음	8
8	35	66	37.562527	126.998314	A	4792	8	6	10	-0.200000	37.562527	126.998314	적음	9
9	28	19	37.555199	127.010048	A	382	13	6	9	-0.777778	37.555199	127.010048	적음	10

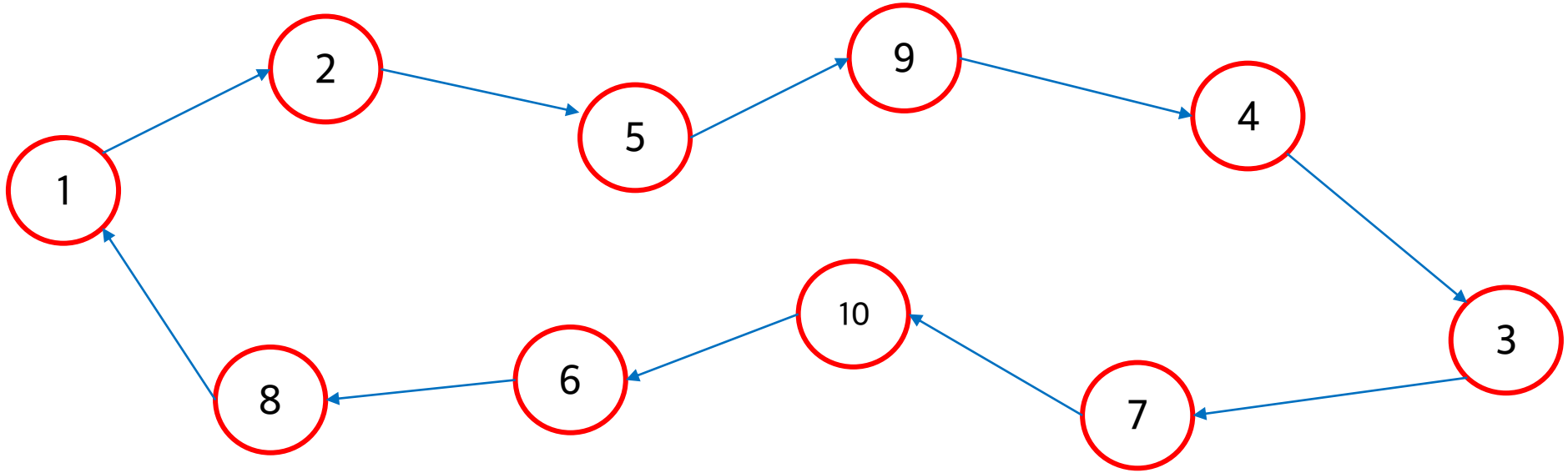
해당 cluster의 대여소 추출

‘(반납-대여)/거치대수‘ 기준으로
내림차순 정렬하여 노드 번호 붙이기

2.4 TSP

Step 3) 동적계획법을 이용해 경로 찾기 (2021.04.23, 6-10시, k=2, A)

Solution to TSP : {1, 2, 5, 9, 4, 3, 7, 10, 6, 8, 1}



2.4 TSP

Step 4) TSP 해당 경로 총거리 구하기 (2021.04.23, 6-10시, k=2, A)

```
node.append(1)

dist_total = 0
for i in range(len(node)-1):
    dist_total += df_dist[node[i]-1][node[i+1]-1]
dist_total
```

```
5.430450546303053
```

해당 경로의 총 거리 구하기

	0	1	2	3	4	5	6	7	8	9
0	0.000000	0.619526	1.546055	1.670230	0.863102	1.036615	1.211413	0.528565	1.259811	1.555098
1	0.619526	0.000000	1.032775	1.216975	0.332498	0.793938	0.643781	0.306942	0.746316	1.060699
2	1.546055	1.032775	0.000000	0.260583	0.707396	1.653431	1.073646	1.330777	0.289929	1.466853
3	1.670230	1.216975	0.260583	0.000000	0.884486	1.887317	1.327623	1.522416	0.482946	1.726993
4	0.863102	0.332498	0.707396	0.884486	0.000000	1.067455	0.701034	0.638573	0.418115	1.158866
5	1.036615	0.793938	1.653431	1.887317	1.067455	0.000000	0.661012	0.570395	1.407974	0.685733
6	1.211413	0.643781	1.073646	1.327623	0.701034	0.661012	0.000000	0.706030	0.884245	0.458443
7	0.528565	0.306942	1.330777	1.522416	0.638573	0.570395	0.706030	0.000000	1.047794	1.026537
8	1.259811	0.746316	0.289929	0.482946	0.418115	1.407974	0.884245	1.047794	0.000000	1.316733
9	1.555098	1.060699	1.466853	1.726993	1.158866	0.685733	0.458443	1.026537	1.316733	0.000000

0.619526 + 0.332498 + 0.418115 + 0.482946 + 0.260583 + 1.073646 + 0.458443 + 0.685733 + 0.570395 + 0.528565

= **5.43045** (km)

2.4 TSP

Step 4) TSP 결과 (2021.04.23, 6-10시)

K=2

cluster	number	tsp route	distance
A	10	1, 2, 5, 9, 4, 3, 7, 10, 6, 8, 1	5.43
B	26		

** Cluster B는 노드가 26개로 너무 많아서 경로탐색 불가능

K=3

cluster	number	tsp route	distance
A	8	1, 2, 3, 7, 5, 8, 4, 6, 1	4.50
B	13	1, 5, 2, 4, 3, 10, 8, 11, 12, 6, 7, 13, 9, 1	4.07
C	15	1, 8, 5, 12, 14, 15, 11, 13, 3, 7, 6, 9, 4, 2, 10, 1	6.07

K=4

cluster	number	tsp route	distance
A	7	1, 6, 4, 7, 5, 3, 2, 1	3.90
B	9	1, 2, 4, 3, 8, 7, 6, 9, 5, 1	2.69
C	10	1, 6, 4, 2, 9, 5, 7, 8, 3, 10, 1	2.69
D	10	1, 4, 5, 3, 7, 9, 10, 6, 8, 2, 1	4.23

K=5

cluster	number	tsp route	distance
A	5	1, 3, 2, 4, 5, 1	3.09
B	4	1, 2, 3, 4, 1	3.51
C	11	1, 8, 7, 6, 10, 9, 11, 3, 4, 2, 5, 1	3.58
D	6	1, 2, 5, 3, 4, 6, 1	2.21
E	10	1, 4, 5, 3, 7, 9, 10, 6, 8, 2, 1	4.23

2.4 TSP

Step 4) TSP 결과 (2021.04.23, 17-21시)

K=2

cluster	number	tsp route	distance
A	10	1, 3, 10, 8, 9, 5, 4, 7, 2, 6, 1	7.26
B	23		

** Cluster B는 노드가 23개로 너무 많아서 경로탐색 불가능

K=3

cluster	number	tsp route	distance
A	5	1, 2, 4, 5, 3, 1	4.58
B	14	1, 3, 12, 13, 14, 9, 8, 11, 10, 5, 7, 6, 2, 4, 1	8.04
C	14	1, 3, 4, 2, 6, 7, 13, 10, 14, 11, 9, 5, 12, 8, 1	5.19

K=4

	number	tsp route	distance
A	5	1, 2, 4, 5, 3, 1	4.58
B	9	1, 3, 7, 8, 9, 5, 6, 2, 4, 1	6.50
C	12	1, 2, 4, 10, 8, 12, 11, 9, 5, 6, 7, 3, 1	3.60
D	7	1, 2, 3, 5, 6, 7, 4, 1	3.37

K=5

cluster	number	tsp route	distance
A	2	1, 2, 1	1.81
B	4	1, 2, 3, 4, 1	3.56
C	8	1, 2, 8, 4, 5, 7, 6, 3, 1	6.00
D	12	1, 2, 4, 10, 8, 12, 11, 9, 5, 6, 7, 3, 1	3.60
E	7	1, 2, 3, 5, 6, 7, 4, 1	3.37

2.4 TSP

Step 4) TSP 결과 (2021.04.24, 6-10시)

K=2

cluster	number	tsp route	distance
A	16	1, 9, 11, 13, 12, 16, 8, 10, 14, 15, 6, 2, 5, 3, 4, 7, 1	11.83
B	14	1, 3, 2, 9, 11, 12, 14, 10, 13, 4, 7, 5, 8, 6, 1	9.39

K=3

cluster	number	tsp route	distance
A	12	1, 6, 3, 2, 4, 11, 8, 12, 10, 9, 7, 5, 1	5.02
B	11	1, 6, 8, 4, 2, 10, 9, 5, 3, 11, 7, 1	5.58
C	7	1, 6, 4, 2, 7, 5, 3, 1	3.78

K=4

cluster	number	tsp route	distance
A	5	1, 2, 4, 3, 5, 1	3.71
B	8	1, 2, 8, 7, 4, 6, 5, 3, 1	5.97
C	10	1, 3, 2, 5, 8, 6, 7, 9, 10, 4, 1	4.70
D	7	1, 2, 4, 5, 7, 3, 6, 1	3.67

K=5

cluster	number	tsp route	distance
A	7	1, 4, 3, 6, 2, 7, 5, 1	2.90
B	5	1, 2, 3, 4, 5, 1	4.20
C	3	1, 2, 3, 1	0.98
D	6	1, 2, 5, 6, 4, 3, 1	2.37
E	9	1, 2, 8, 5, 9, 7, 6, 4, 3, 1	4.05

2.4 TSP

Step 4) TSP 결과 (2021.04.24, 17-21시)

K=2

cluster	number	tsp route	distance
A	12	1, 10, 8, 3, 7, 5, 4, 12, 9, 2, 11, 6, 1	7.06
B	23		

** Cluster B는 노드가 23개로 너무 많아서 경로탐색 불가능

K=3

cluster	number	tsp route	distance
A	7	1, 6, 7, 3, 4, 5, 2, 1	4.51
B	21	1, 2, 6, 12, 11, 14, 18, 19, 20, 10, 15, 16, 9, 13, 2 1, 17, 8, 7, 3, 4, 5, 1	5.41
C	7	1, 4, 7, 5, 6, 3, 2, 1	4.70

K=4

cluster	number	tsp route	distance
A	5	1, 4, 5, 3, 2, 1	4.42
B	5	1, 3, 2, 5, 4, 1	2.45
C	13	1, 4, 2, 11, 10, 13, 7, 5, 9, 8, 6, 12, 3, 1	3.90
D	12	1, 5, 4, 3, 7, 8, 10, 12, 11, 9, 2, 6, 1	3.87

K=5

cluster	number	tsp route	distance
A	5	1, 4, 5, 3, 2, 1	4.42
B	5	1, 3, 2, 5, 4, 1	2.45
C	5	1, 2, 4, 3, 5, 1	2.92
D	11	1, 2, 10, 4, 6, 7, 3, 5, 11, 8, 9, 1	2.56
E	9	1, 5, 4, 3, 7, 8, 9, 6, 2, 1	2.77

#5 비용함수

2.5 비용함수

Step 1) 비용함수 식

- 가장 효율적인 k를 찾기 위해 비용(cost) 계산
- cluster 내 대여소 수 < 10 : 1t 트럭
- cluster 내 대여소 수 ≥ 10 : 2.5t 트럭

$$Cost = \frac{distance}{\text{연비}} \times (\text{1km당 기름값}) + \text{인건비} + \text{고정비}$$

- **distance** : tsp를 통해 구한 cluster내 대여소끼리의 총거리
- **연비** : 1t 트럭 : 8.5km
  2.5t 트럭 : 6.8km
- **1km 기름값** : 1332.71(원) (*2021년 4월 기준)
- **인건비** : 시간당 임금 x 시간
  = 23,013.75(원) x 4(시간) = 92,055(원)
- **고정비** : 한나절의 {판매가 + 등록비용(세금 포함)}

$$1t \text{ 트럭} : \frac{\{17{,}500{,}000(\text{원}) + 875{,}000(\text{원})\}}{\{16.8(\text{년}) \times 365(\text{일}) \times 2(\text{하루의 절반})} = 1498.21(\text{원})$$

$$2.5\text{t 트럭} : \frac{\{49{,}000{,}000(\text{원}) + 2{,}450{,}000(\text{원})\}}{16.8(\text{년}) \times 365(\text{일}) \times 2(\text{하루의 절반})\}} = 4195.21(\text{원})$$

2.5 비용함수

Step 2) 식 이용해 트럭 종류와 비용 도출 (2021.04.23, 6-10시)

K=2

	cluster	number	tsp route	distance	truck type	cost
0	A	10	1, 2, 5, 9, 4, 3, 7, 10, 6, 8, 1	5.430451	2.5t	97314.525871
1	B	26	NaN	NaN	2.5t	NaN

K=3

	cluster	number	tsp route	distance	truck type	cost
0	A	8	1, 2, 3, 7, 5, 8, 4, 6, 1	4.497520	1.0t	94258.467027
1	B	13	1, 5, 2, 4, 3, 10, 8, 11, 12, 6, 7, 13, 9, 1	4.065258	2.5t	97046.960226
2	C	15	1, 8, 5, 12, 14, 15, 11, 13, 3, 7, 6, 9, 4, 2,...	6.067822	2.5t	97439.444961

K=4

	cluster	number	tsp route	distance	truck type	cost
0	A	7	1, 6, 4, 7, 5, 3, 2, 1	3.896194	1.0t	94164.183370
1	B	9	1, 2, 4, 3, 8, 7, 6, 9, 5, 1	2.694156	1.0t	93975.712371
2	C	10	1, 6, 4, 2, 9, 5, 7, 8, 3, 10, 1	2.691666	2.5t	96777.748254
3	D	10	1, 4, 5, 3, 7, 9, 10, 6, 8, 2, 1	4.232430	2.5t	97079.724504

K=5

	cluster	number	tsp route	distance	truck type	cost
0	A	5	1, 3, 2, 4, 5, 1	3.088498	1.0t	94037.542364
1	B	4	1, 2, 3, 4, 1	3.513502	1.0t	94104.180008
2	C	11	1, 8, 7, 6, 10, 9, 11, 3, 4, 2, 5, 1	3.575121	2.5t	96950.897559
3	D	6	1, 2, 5, 3, 4, 6, 1	2.209753	1.0t	93899.761302
4	E	10	1, 4, 5, 3, 7, 9, 10, 6, 8, 2, 1	4.232430	2.5t	97079.724504

2.5 비용함수

Step 3) 마다의 총거리와 총비용 도출 (2021.04.23, 6-10시)

	k=2	k=3	k=4	k=5
total_distance(km)	NA	14.63	13.51	16.62
total_cost(원)	NA	288,745	381,997	476,072

최소 total_distance(km) : k=4

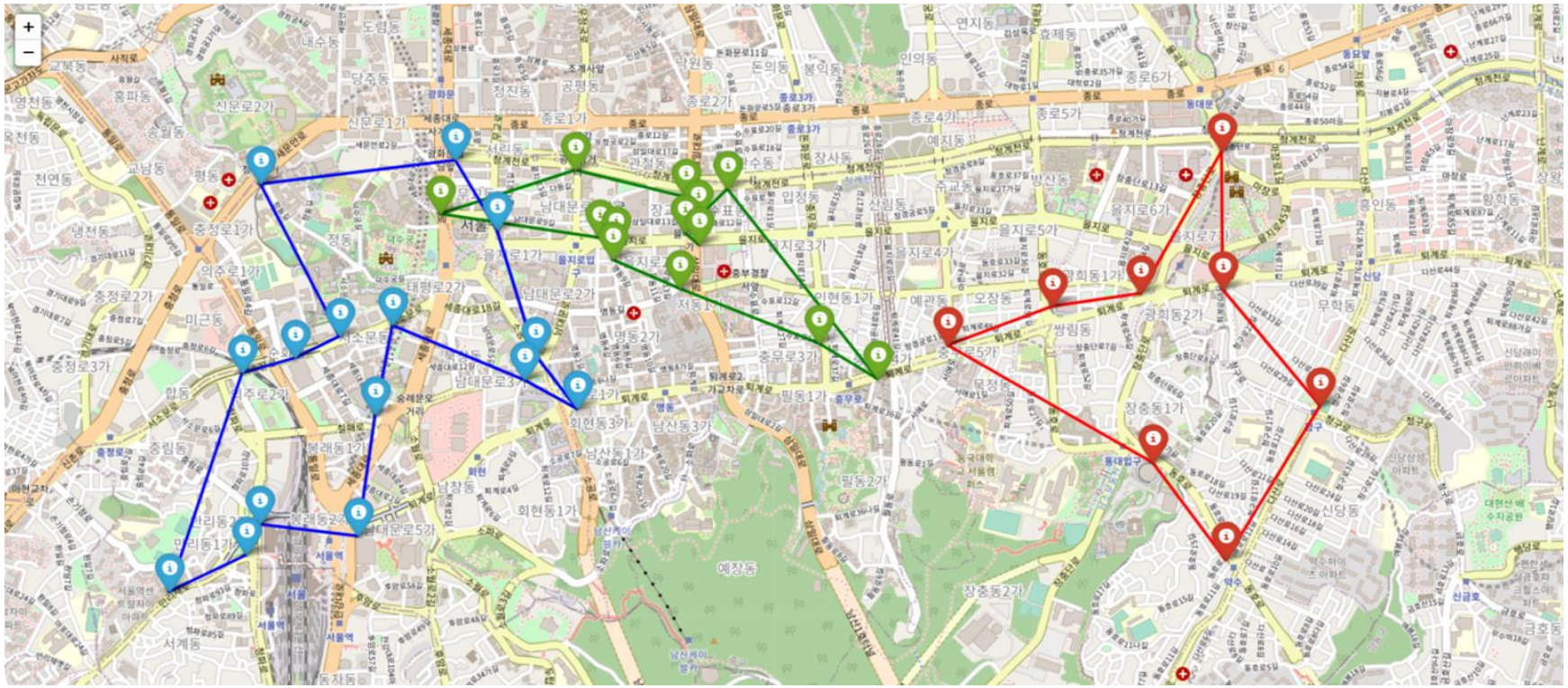
최소 total_cost(원) : k=3

→ 비용 측면에서 k=3일 때 가장 효율적

2.5 비용함수

Step 4) 재배치 결과 (2021.04.23, 6-10시)

- k=3일 때 최소 비용



A : 1t 트럭

B : 2.5t 트럭

C : 2.5t 트럭

2.5 비용함수

Step 3) 마다의 총거리와 총비용 도출 (2021.04.23, 17-21시)

	k=2	k=3	k=4	k=5
total_distance(km)	NA	17.81	18.05	18.34
total_cost(원)	NA	289,364	379,881	473,780

최소 total_distance(km) : k=3

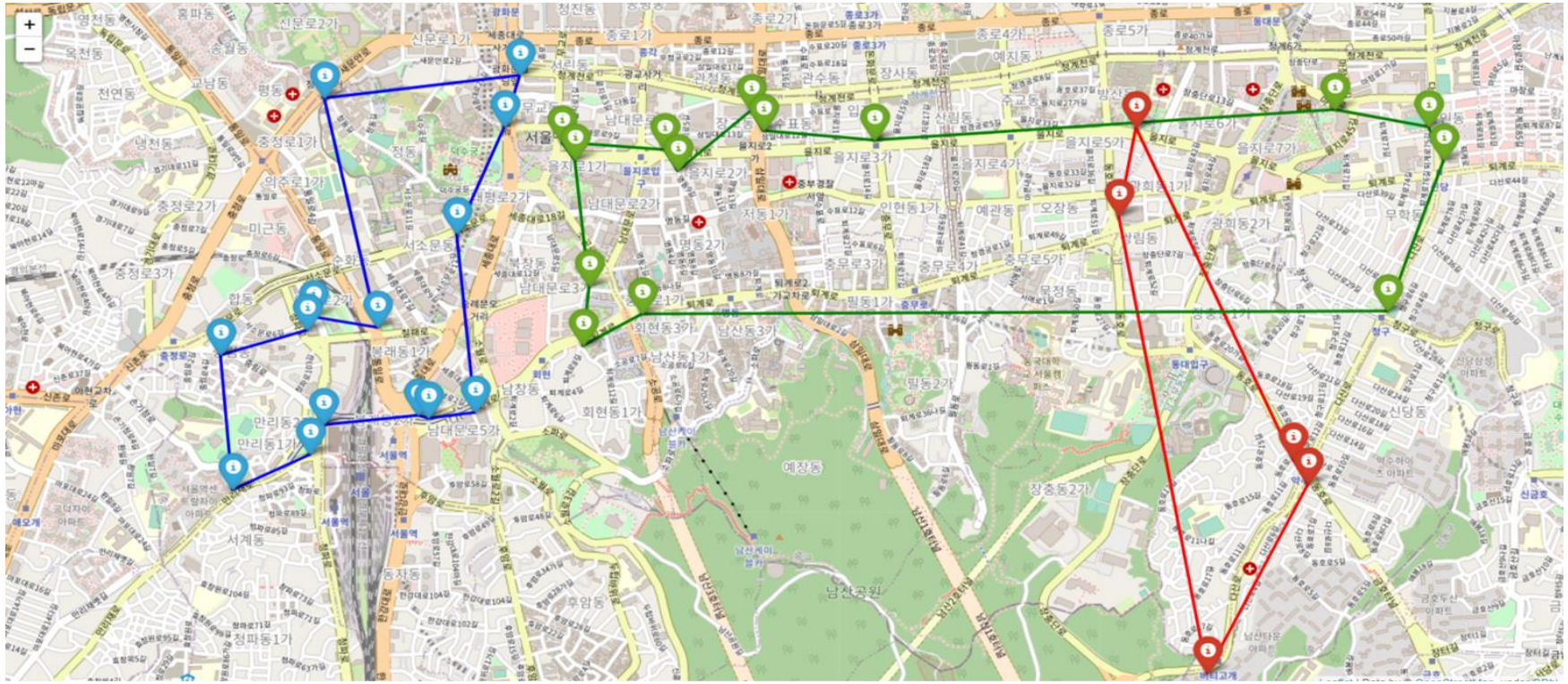
최소 total_cost(원) : k=3

→ 비용 측면에서 k=3일 때 가장 효율적

2.5 비용함수

Step 4) 재배치 결과 (2021.04.23, 17-21시)

- k=3일 때 최소 비용



2.5 비용함수

Step 3) 마다의 총거리와 총비용 도출 (2021.04.24, 6-10시)

	k=2	k=3	k=4	k=5
total_distance(km)	21.22	14.39	18.06	14.51
total_cost(원)	196,659	288,725	379,925	470,041

최소 total_distance(km) : k=3

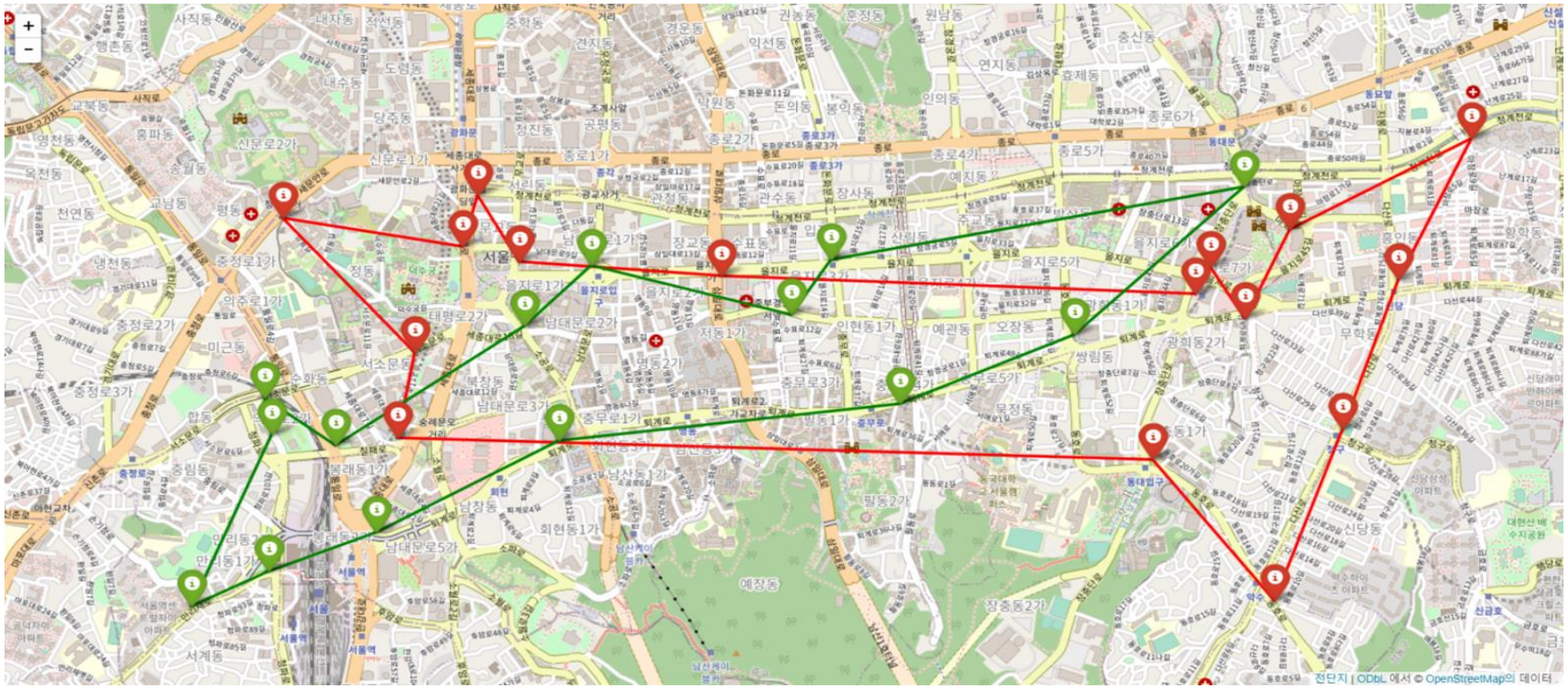
최소 total_cost(원) : k=2

→ 비용 측면에서 k=2일 때 가장 효율적

2.5 비용함수

Step 4) 재배치 결과 (2021.04.24, 6-10시)

- k=2일 때 최소 비용



2.5 비용함수

Step 3) 마다의 총거리와 총비용 도출 (2021.04.24, 17-21시)

	k=2	k=3	k=4	k=5
total_distance(km)	NA	14.62	14.63	15.11
total_cost(원)	NA	285,861	382,206	472,932

최소 total_distance(km) : k=3

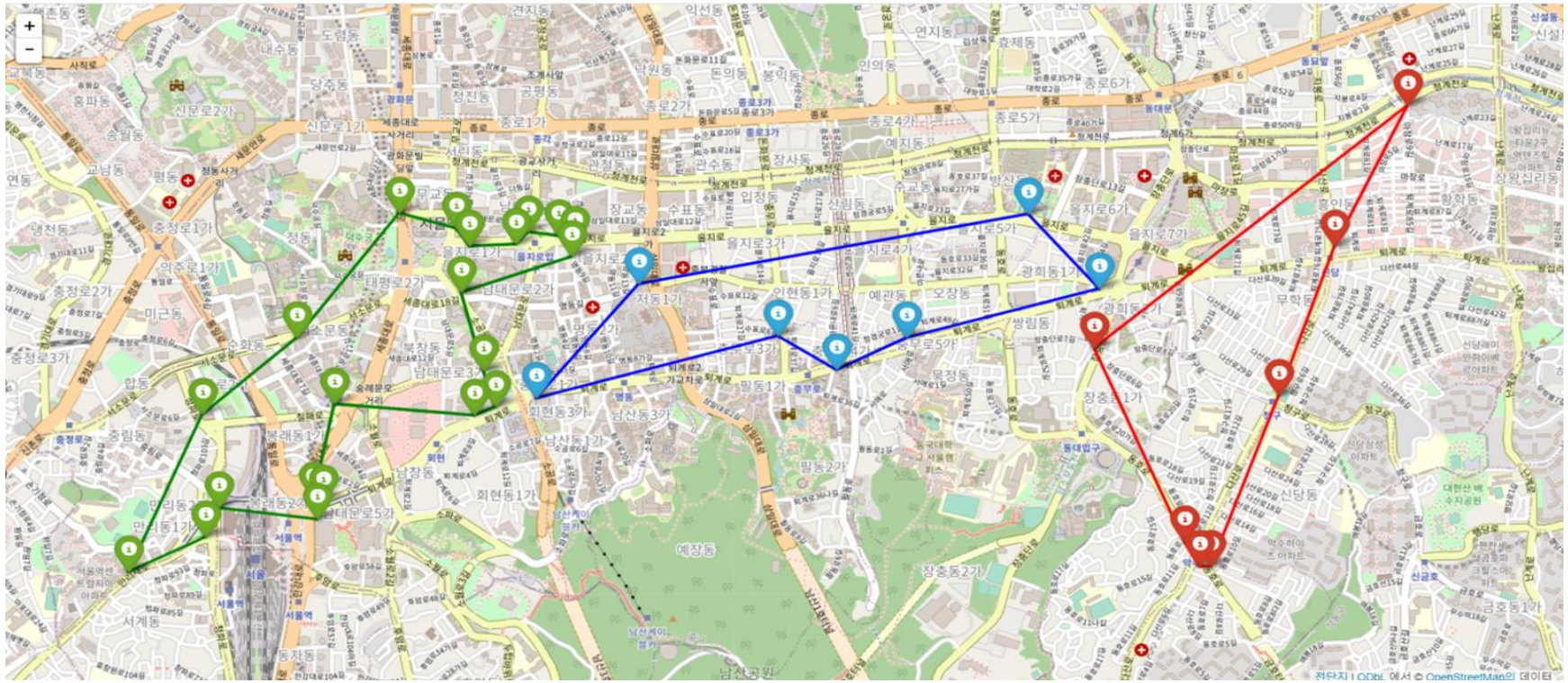
최소 total_cost(원) : k=3

→ 비용 측면에서 k=3일 때 가장 효율적

2.5 비용함수

Step 4) 재배치 결과 (2021.04.24, 17-21시)

- k=3일 때 최소 비용



A : 1t 트럭

B : 2.5t 트럭

C : 1t 트럭

3. 결론

3.1 군집별 광고 타겟팅 결론 및 한계

결론

1. 제안한 타겟층에 맞게 대여소별로 광고를 게시하면 보다 광고 효과가 클 것으로 예상

2. 주말/평일, 시간대별 차별적 특징 발견 X

3. 외부 데이터, 사용자의 개별적인 특징 데이터를 이용한 추가적인 분석 필요

한계

- 군집분석은 비지도 학습으로 목표변수가 없음

  → 비교가 불가능

  → 군집이 잘 나뉘었는지 평가시 문제가 되기도

3.2 자전거 재배치 효율화 결론 및 한계

결론

1. 출, 퇴근 시간대에 자전거의 대여수와 반납수가 불균형하게 분포

   but 주말, 평일에 두드러진 분포차이 X

2. 거리보다는 군집이 적을수록 비용이 적음

3. 인건비가 비용의 큰 비중을 차지

→ 대여와 반납의 차이가 큰 출퇴근 시간에만 재배치 제안

한계

1. 자전거 거치율 데이터가 없어, 대여소에 실제 자전거수 파악에 어려움.

→ 보다 정확한 재배치를 위해선 실시간 거치율 데이터를 수집해 회수하는 자전거의 대수까지 고려해야

2. 비용함수 식을 임의로 결정한 것이기에 실제 데이터에 적절한지 확인할 수 없음

감사합니다

따릉플러스
김소은 20196981
김서린 20191376
신정아 20194354
이윤지 20190765