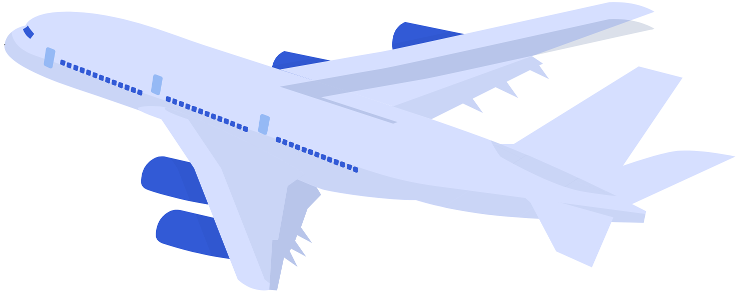
Selecting the optimal location & route for air taxis to relieve urban traffic congestion



Team4
: 18102095 - Jihwan Hwang
: 19102077 - Seokjun Kang
: 19102099 - Junseok Jeon
: 19102127 - Suho Lee

Business Analytics – Project Progress 14th Nov.

Table of contents

1

OD data EDA

OD data

수도권 주수단 OD - 국가 교통 DB (https://www.ktdb.go.kr/www/index.do)
- The number of moves from origin to destination by different transportation methods

Year	2019
Time zone (in each file)	AM Peak times(07~09) PM Peak times(18~20) Full times(0~24)
Area	Metropolitan area
Unit of region	Eup, Myeon, Dong (읍, 면, 동)
Size	1048575 rows x 15 columns

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	1111054000	586744	1	1111054000	586744	283.889	9.038	0.256	0.003	44.15	122.309	48.201	1.324	0.403
1	1111054000	586744	2	1111060000	586741	106.335	38.647	1.89	0	0.961	4.584	13.287	0.3	0.071
1	1111054000	586744	3	1111051500	586743	89.261	25.016	0.429	0	2.675	8.072	2.655	0.351	0.039
1	1111054000	586744	4	1111055000	586745	68.382	35.283	5.816	0	24.519	0.761	5.5	0.45	0.009
1	1111054000	586744	5	1111056000	586746	11.559	23.443	24.609	0	5.002	2.345	1.337	0.128	0.013

Description for OD data

Column	Description
1~3	ID(code) of Origin
4~6	ID(code) of Destination
07	The number of moves by walk or bikes
08	The number of moves by trucks
09	The number of moves by intercity, express buses
10	The number of moves by trains
11	The number of moves by cars
12	The number of moves by taxis
13	The number of moves by buses
14	The number of moves by subways
15	The number of moves by buses and subways

COLUNM 순서	설명
1	기점_존번호
2	기점_CODE
3	기점_노드
4	종점_존번호
5	종점_CODE
6	종점_노드
7	도보/자전거
8	화물/기타
9	기타버스(시외,고속,기타버스)
10	일반철도/KTX
11	승용차
12	택시
13	버스
14	지하철
15	버스+지하철

Feature Selection of OD data

1. Consider **AM/PM peak** demand data types

2. Select rows **with Seoul's 'dong**' as origin and destination

3. Select columns for **car, taxi, bus, and subway**, excluding other modes of transport.

- People who commute by foot or bicycles, or ride trucks will not take air taxis.
- People who commute by other buses or trains will leave Seoul
- Data is duplicated due to the transfer(환승) in 'bus and subway'

4. Add **a 'total' column** to calculate the sum of car, taxi, bus, and subway

5. Add columns of **'gu' and 'dong'** for origin/destination.

2016년 행정구역	시도	시군구	읍면동	TAZ	10code	Node_id	행자부6code
서울특별시 종로구 삼청동	서울특별시	종로구	삼청동	1	1111054000	586744	1101054

6. Drop the rows that has **same origin and destination**

Feature Selection of OD data

	zone_origin	code_origin	node_origin	zone_dest	code_dest	node_dest	walk_and_bike	truck	other_bus	train	car	taxi	bus	subway	bus_and_subway
0	1	1.111054e+09	586744	1	1.111054e+09	586744	12.759	0.194	0.041	0.0	0.687	0.052	0.960	0.056	0.001
1	1	1.111054e+09	586744	2	1.111060e+09	586741	29.078	5.872	0.611	0.0	0.135	0.065	4.007	0.066	0.014
2	1	1.111054e+09	586744	3	1.111052e+09	586743	24.138	8.941	0.063	0.0	0.844	0.862	0.920	0.110	0.007
3	1	1.111054e+09	586744	4	1.111055e+09	586745	29.797	8.332	0.787	0.0	10.162	0.249	2.177	0.177	0.002
4	1	1.111054e+09	586744	5	1.111056e+09	586746	0.951	2.605	0.514	0.0	0.498	0.055	0.143	0.012	0.001

1048575 rows x 15 columns

	code_origin	code_dest	gu_origin	dong_origin	gu_dest	dong_dest	car	taxi	bus	subway	total
0	1.111054e+09	1.111054e+09	종로구	삼청동	종로구	삼청동	0.687	0.052	0.960	0.056	1.755
1	1.111054e+09	1.111060e+09	종로구	삼청동	종로구	가회동	0.135	0.065	4.007	0.066	4.273
2	1.111054e+09	1.111052e+09	종로구	삼청동	종로구	청운효자동	0.844	0.862	0.920	0.110	2.736
3	1.111054e+09	1.111055e+09	종로구	삼청동	종로구	부암동	10.162	0.249	2.177	0.177	12.765
4	1.111054e+09	1.111056e+09	종로구	삼청동	종로구	평창동	0.498	0.055	0.143	0.012	0.708

179352 rows × 11 columns

Statistics of OD data(am)

```
am_peak[['car', 'taxi', 'bus', 'subway', 'total']].describe()
```

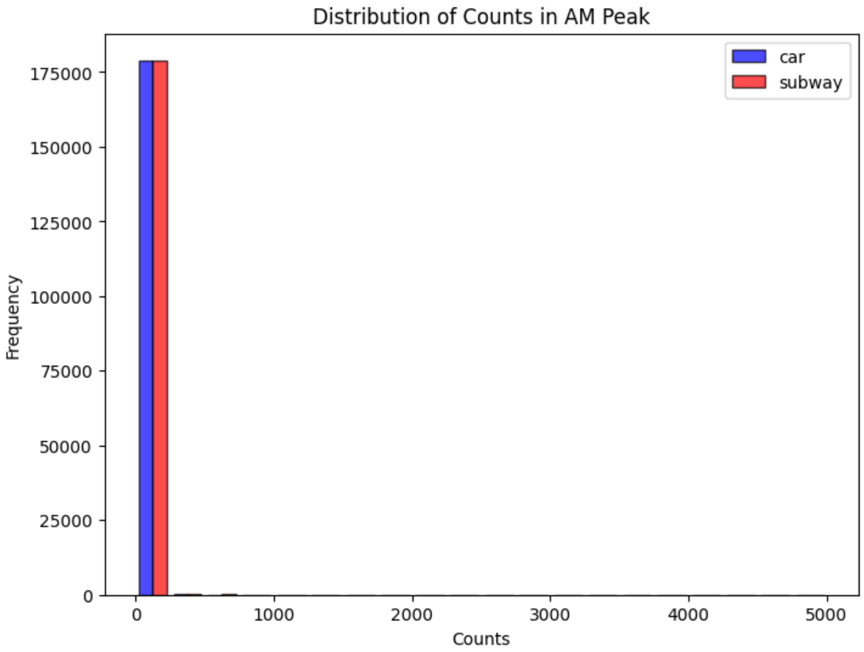
	car	taxi	bus	subway	total
count	179352.000000	179352.000000	179352.000000	179352.000000	179352.000000
mean	4.467919	0.771198	3.740953	4.902831	13.882900
std	30.683193	7.833841	32.247364	39.923781	70.974478
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.071000	0.001000	0.001000	0.094000	0.654000
75%	1.117000	0.047000	0.236000	1.201000	5.357000
max	5007.257000	528.249000	2602.173000	3282.048000	5250.784000

Statistics of OD data(pm)

```
pm_peak[['car', 'taxi', 'bus', 'subway', 'total']].describe()
```

	car	taxi	bus	subway	total
count	179352.000000	179352.000000	179352.000000	179352.000000	179352.000000
mean	4.102677	1.002165	2.760725	4.112264	11.977832
std	32.831698	12.706302	23.902696	37.408524	67.486163
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.058000	0.001000	0.001000	0.076000	0.501000
75%	0.925000	0.066000	0.181000	0.923250	4.240000
max	6050.520000	1354.078000	2407.245000	2818.066000	6305.669000

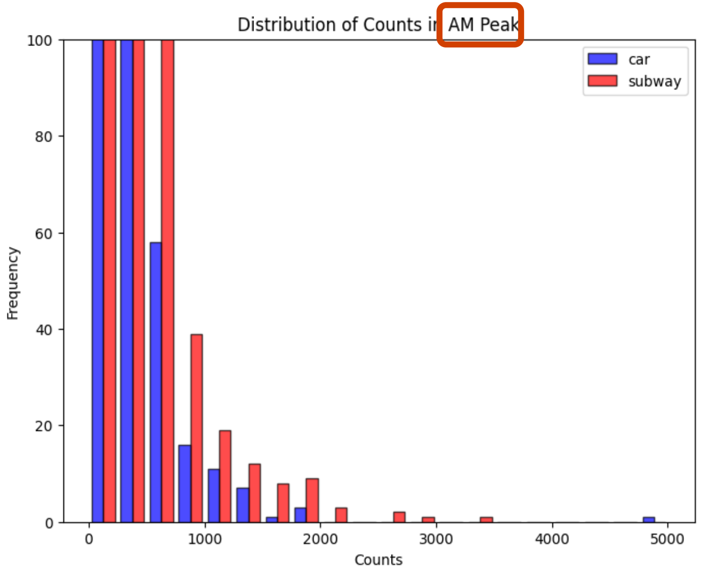
Distribution of OD data (AM)

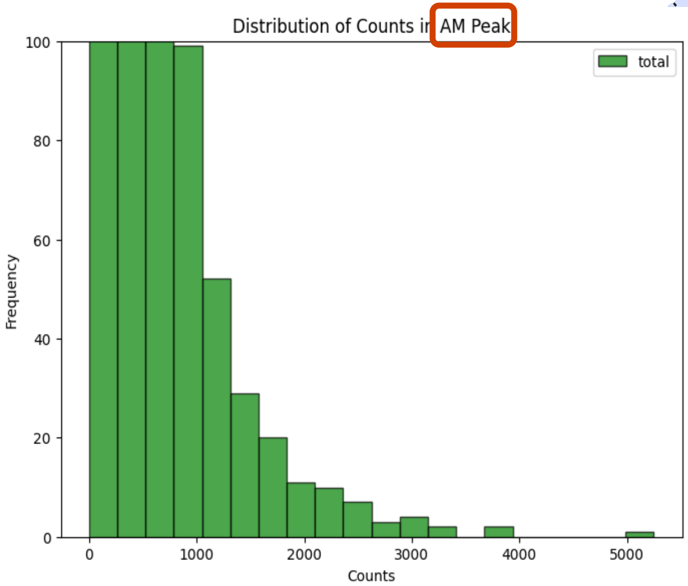


X-axis: **the amount of traffic from origin to destination.**

- Very frequent: a small amount of traffic from origins and destinations
- Therefore, let’s reduce range of the y-axis then look at the other values.

Distribution of OD(AM) data

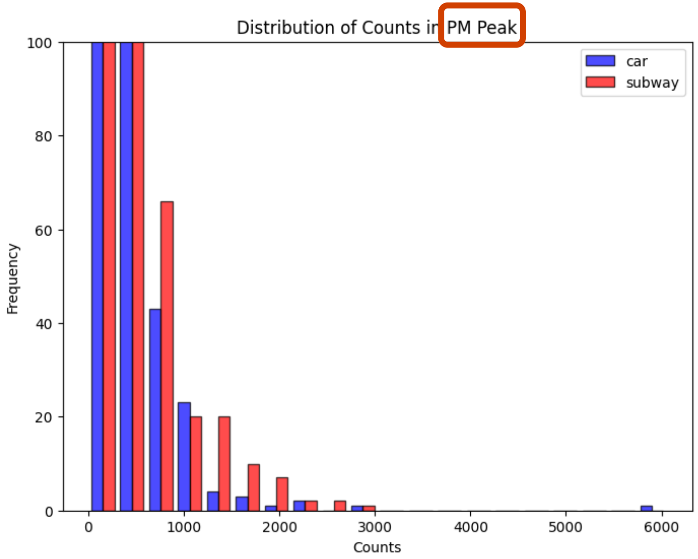


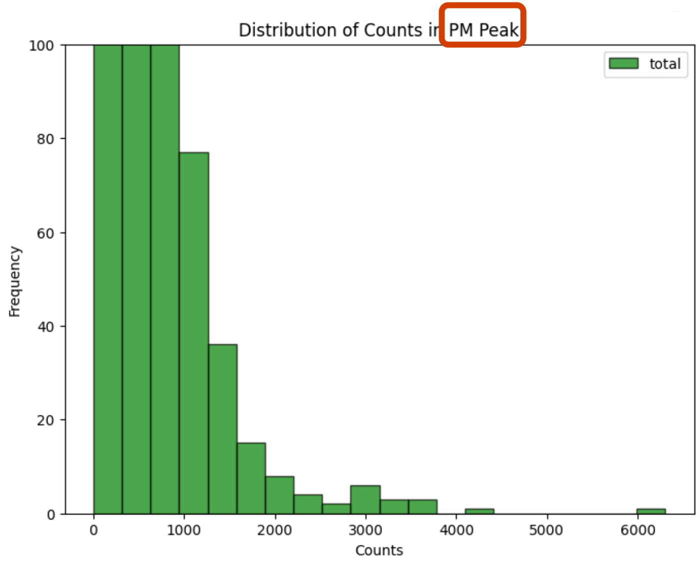


The traffic between very few areas is very high.

	code_origin	code_dest	gu_origin	dong_origin	gu_dest	dong_dest	car	taxi	bus	subway	total
166613	1.171065e+09	1.171071e+09	송파구	잠실본동	송파구	잠실6동	5007.257	22.452	180.079	40.996	5250.784
139433	1.162064e+09	1.168064e+09	관악구	서원동	강남구	역삼1동	1791.188	23.242	11.174	1987.597	3813.201
165297	1.171058e+09	1.168064e+09	송파구	송파1동	강남구	역삼1동	137.521	15.021	539.419	2995.111	3687.072
28427	1.121582e+09	1.114055e+09	광진구	자양1동	중구	명동	3.569	0.474	16.027	3282.048	3302.118

Distribution of OD(PM) data





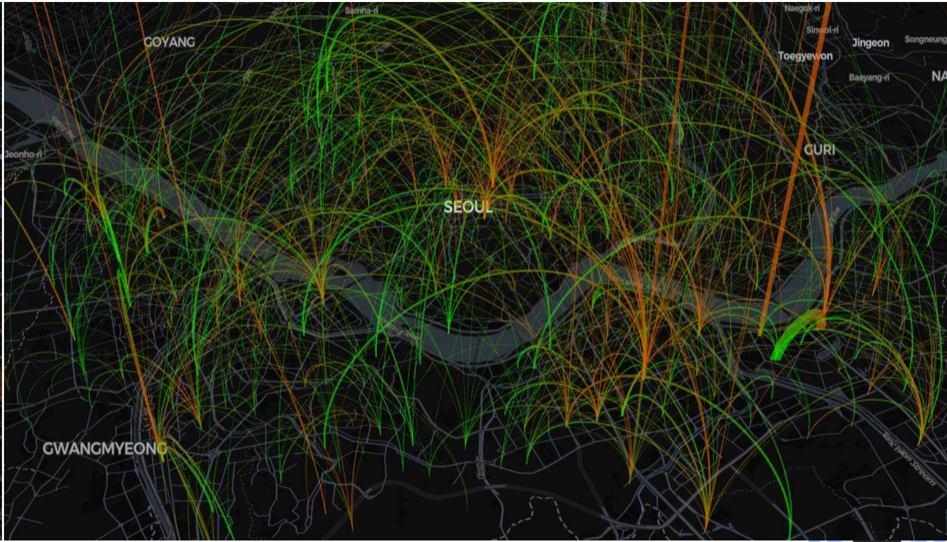
The traffic between very few areas is very high.

	code_origin	code_dest	gu_origin	dong_origin	gu_dest	dong_dest	car	taxi	bus	subway	total
172112	1.171071e+09	1.171065e+09	송파구	잠실6동	송파구	잠실본동	6050.520	33.867	174.692	46.590	6305.669
116880	1.154551e+09	1.154567e+09	금천구	가산동	금천구	시흥1동	405.682	119.938	2407.245	1323.882	4256.747
94386	1.147051e+09	1.150062e+09	양천구	목1동	강서구	우장산동	636.928	10.195	400.764	2675.003	3722.890
8086	1.114055e+09	1.114064e+09	중구	명동	중구	약수동	1424.241	29.941	42.047	1982.347	3478.576

Visualization of OD data

[top 1000 traffic by cars]

AM

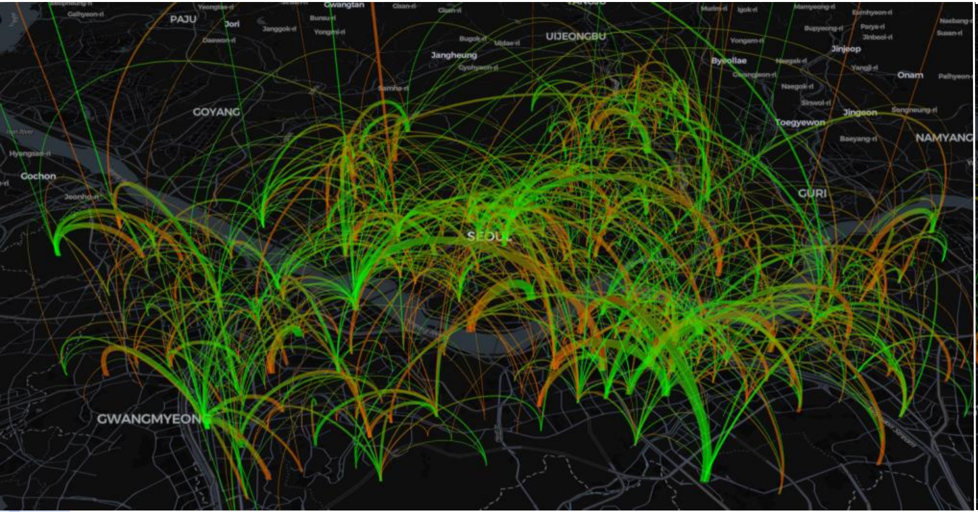


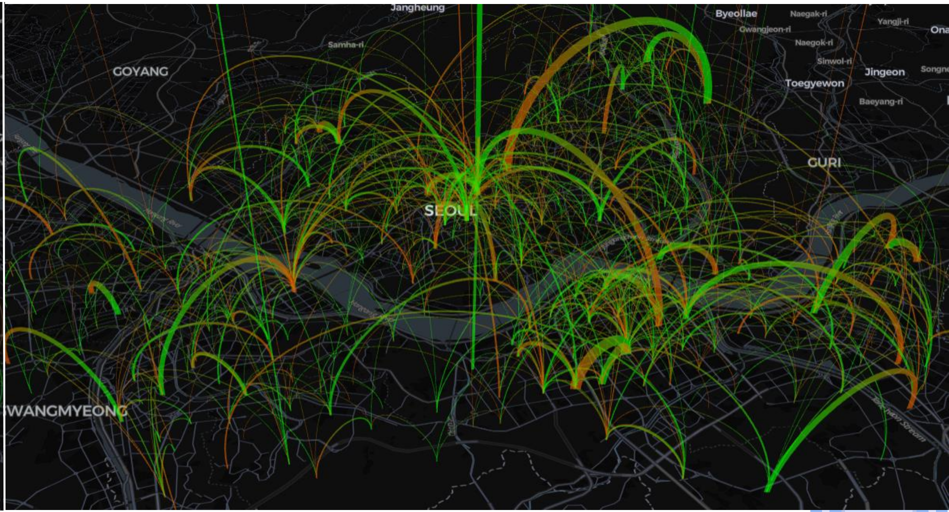
PM

origin: red / destination: green

Visualization of OD data

[top 1000 traffic by taxis]





AM

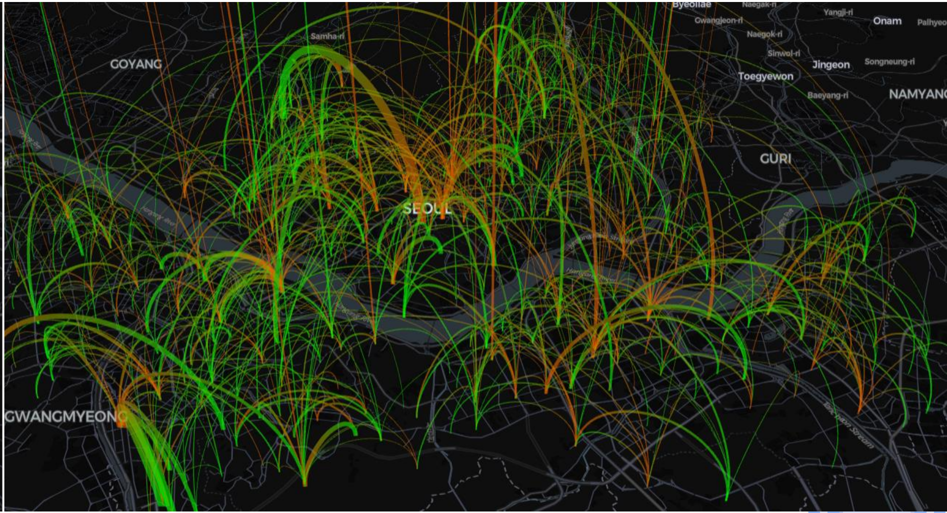
PM

origin: red / destination: green

Visualization of OD data

[top 1000 traffic by buses]

AM

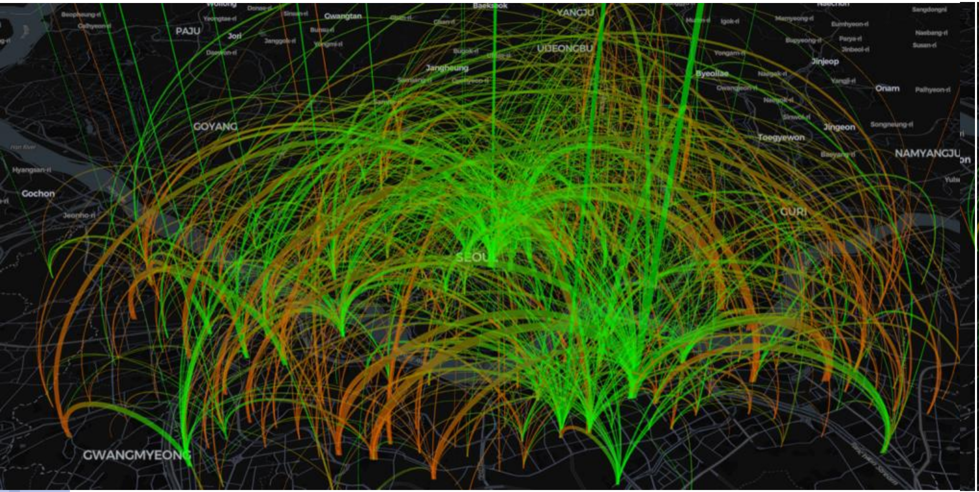


PM

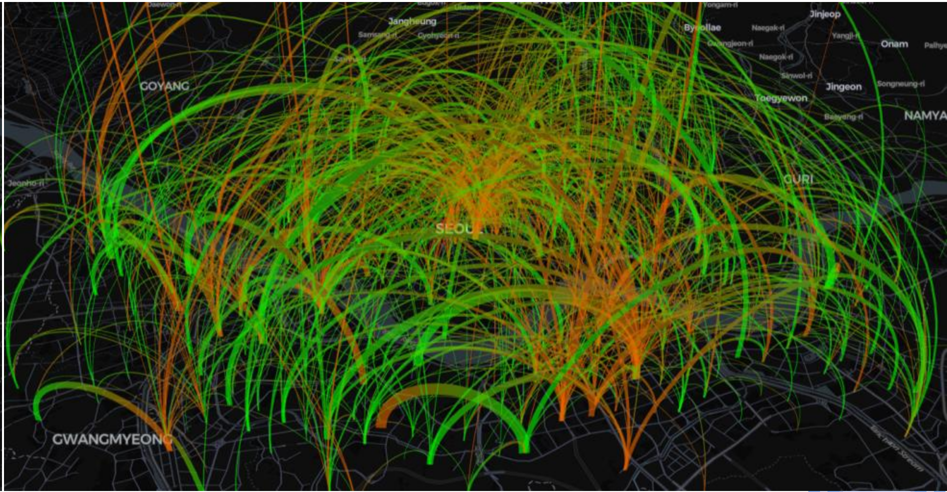
origin: red / destination: green

Visualization of OD data

[top 1000 traffic by subways]



AM

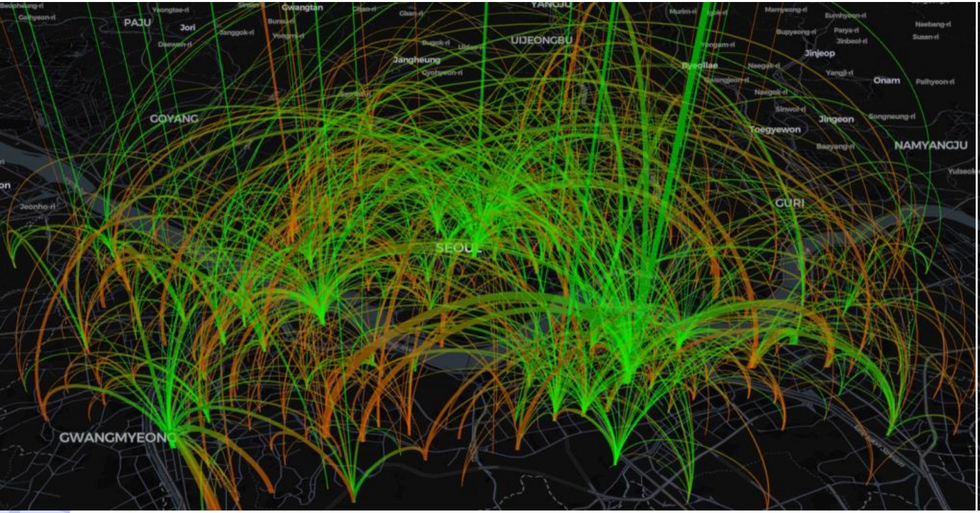


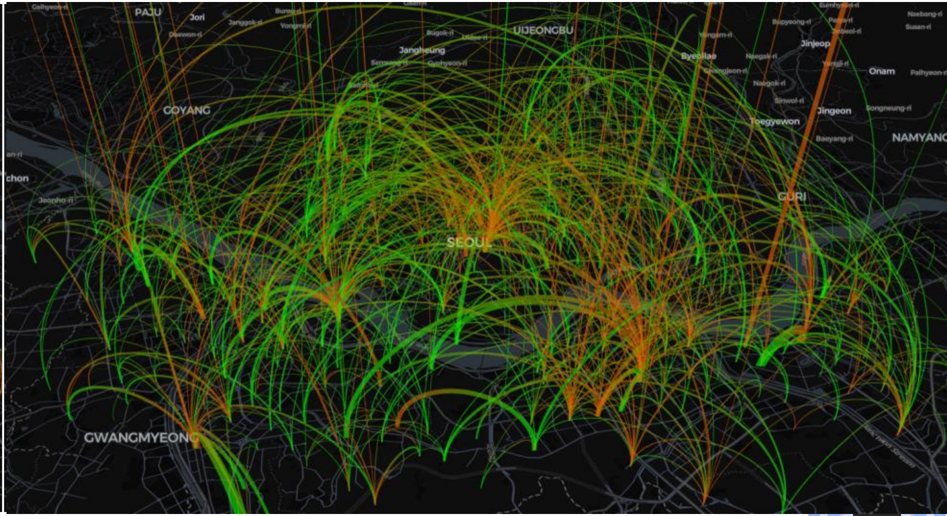
PM

origin: red / destination: green

Visualization of OD data

[top 1000 traffic in total]





AM

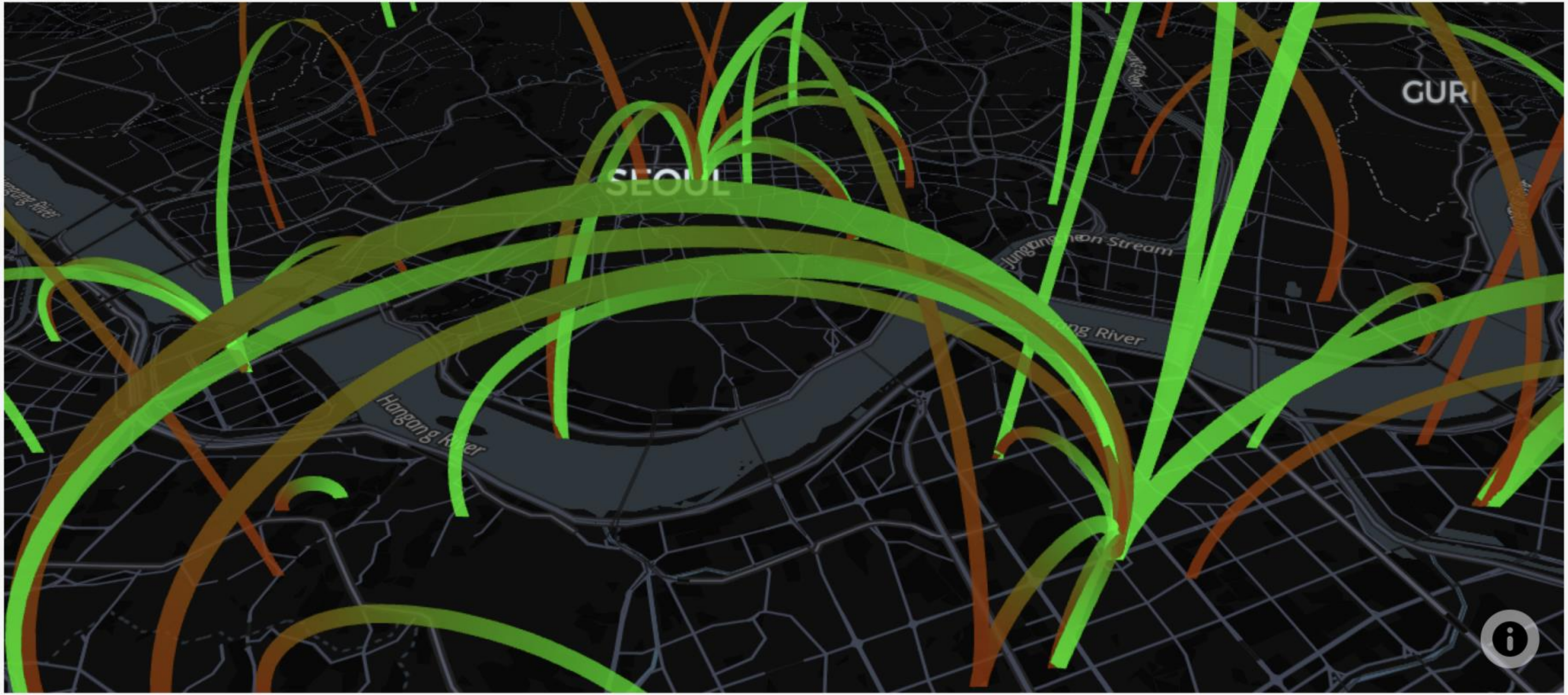
PM

origin: red / destination: green

[Top 30 Traffic by Whole Methods in Each Time Slot(AM & PM)]

(60 Arcs)

Link



Source: https://ksj27.github.io/

Web Viewer Terms | Privacy & Cookies

Visualization of OD data

By visualization with arcs between origins and destinations....

- Between high-traffic areas, **cars or subways cover longer distances** than other methods of transportation.

- In general, the origin and destination in the **morning and evening** are **opposite** to each other.

- The points where the heavy traffic lines(arcs) gather are identified.
  e.g.) Myeong-dong, Yeouido, Samsung, Seocho, Yeoksam, Jamsil, etc

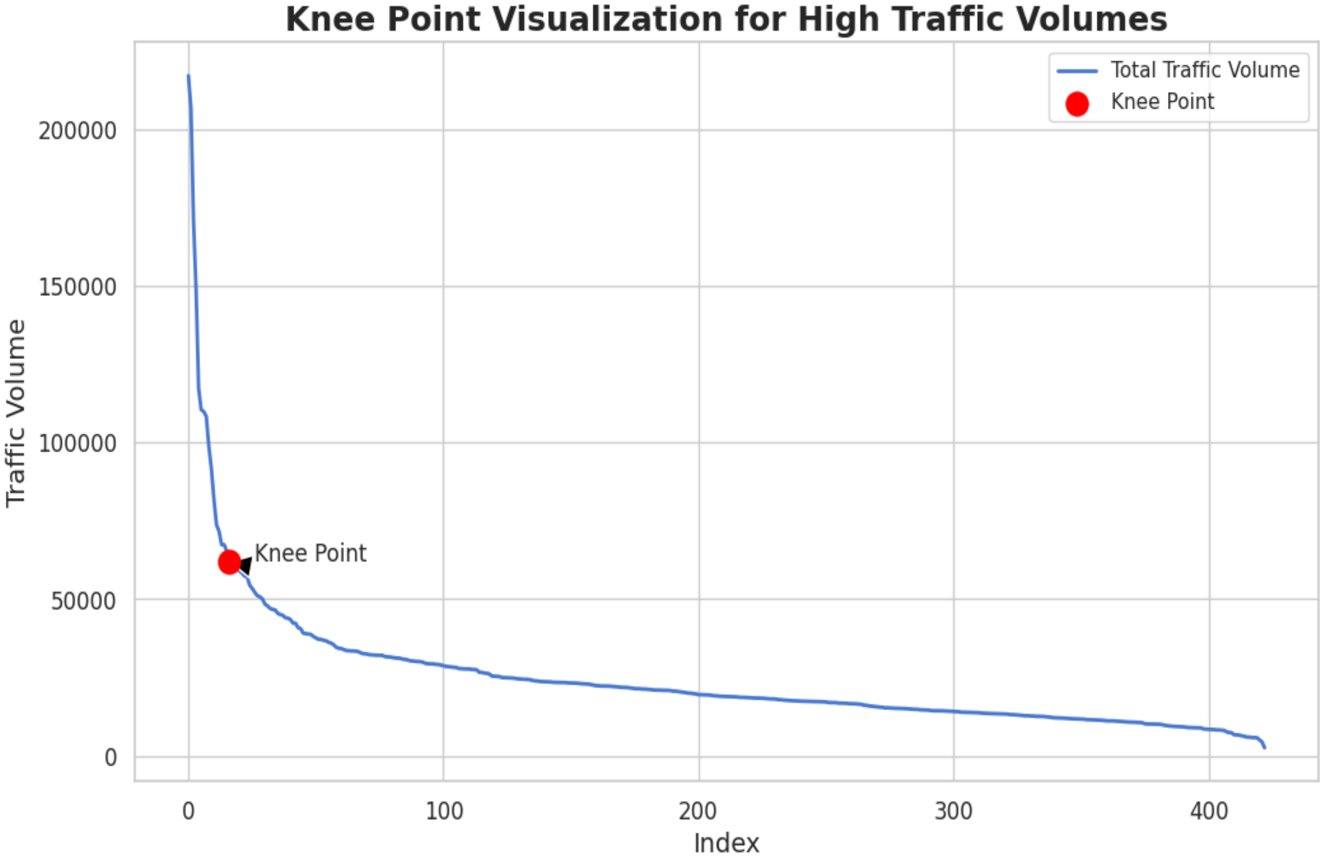
2

OD data processing

Find the highest traffic volumes

FInd the number of candidates based on traffic volumes

```
Knee Point Index: 15
```



Find the highest traffic volumes

Why do we use knee point for traffic volumes?

Knee point: Inflection point which figures out **important change points** among data points

We can find **the optimal number of most crowded 'dong'** which could be our main target location for air taxis.

Find the highest traffic volumes

transportation_total	gu_origin	dong_origin
217219.624	강남구	역삼1동
206814.538	중구	명동
170774.229	영등포구	여의동
148897.950	종로구	종로1·2·3·4가동
117291.472	서초구	서초3동
110525.866	금천구	가산동
109919.968	강남구	삼성1동
108411.681	서초구	서초1동
98826.372	마포구	서교동
91692.785	중구	소공동
73768.345	강남구	논현1동
71682.788	서초구	양재1동
67424.200	강서구	가양1동
67408.203	구로구	구로3동
65392.306	영등포구	영등포동
61943.549	송파구	잠실6동

Find the highest traffic volumes

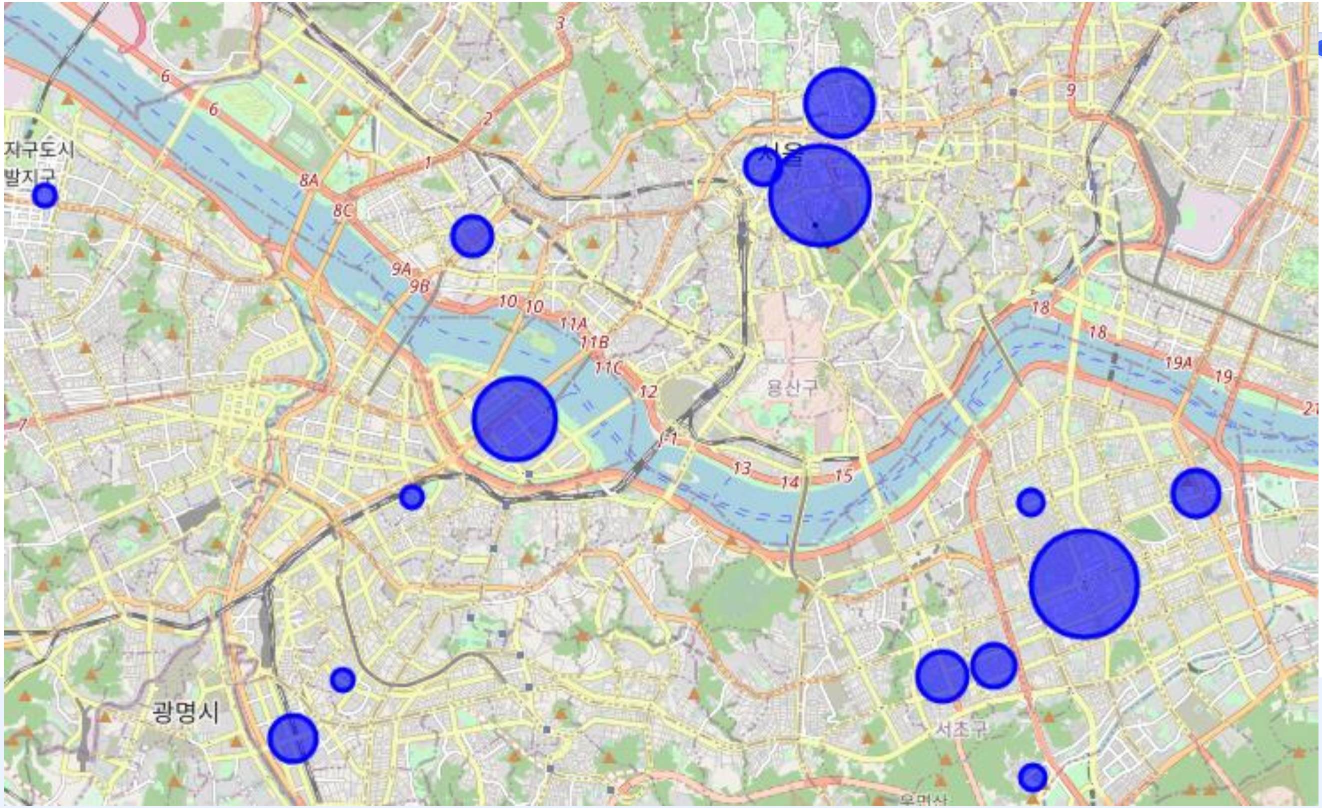
Match latitude and longitude for top 16 'dong'

gu_origin	dong_origin	x	y
강남구	역삼1동	37.500509	127.036990
중구	명동	37.559980	126.985830
영등포구	여의동	37.525880	126.926920
종로구	종로1·2·3·4가동	37.574164	126.989729
서초구	서초3동	37.486200	127.009561
금천구	가산동	37.476686	126.883777
강남구	삼성1동	37.514427	127.058650
서초구	서초1동	37.487819	127.019591
마포구	서교동	37.553781	126.918677
중구	소공동	37.564413	126.974918
강남구	논현1동	37.512941	127.026507
서초구	양재1동	37.470860	127.027005
강서구	가양1동	37.560176	126.835692
구로구	구로3동	37.485738	126.893313
영등포구	영등포동	37.513783	126.906919
송파구	잠실6동	37.516880	127.102022

Find the highest traffic volumes

Folium: python map module

Find the highest traffic volumes



Is it enough to find best location using only traffic volume data?

Nope!!

3

Additional Dataset EDA

Needed Features

Income Level

High-income areas in Seoul are anticipated **to have great purchasing power**

Population density

Seoul's **densely** populated areas may face more **congestion & high demand**

Number of Company

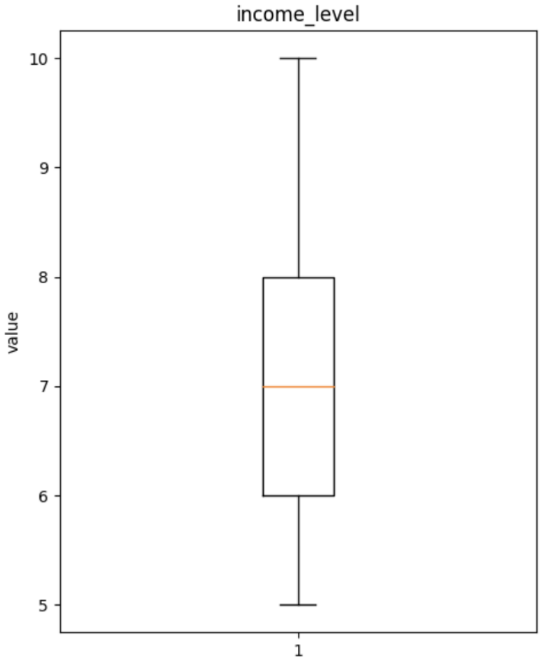
Areas with **many companies** are likely to see high usage for **business purposes**

Income Level in Seoul

Resource: Seoul Commercial District Analysis Service(https://golmok.seoul.go.kr)

분위	연평균 보험료 범위 (원)
10분위	6,945,812원 이상
9분위	4,890,362원 ~ 6,945,811원
8분위	3,741,083원 ~ 4,890,361원
7분위	2,983,559원 ~ 3,741,082원
6분위	2,440,600원 ~ 2,983,558원
5분위	2,020,852원 ~ 2,440,599원

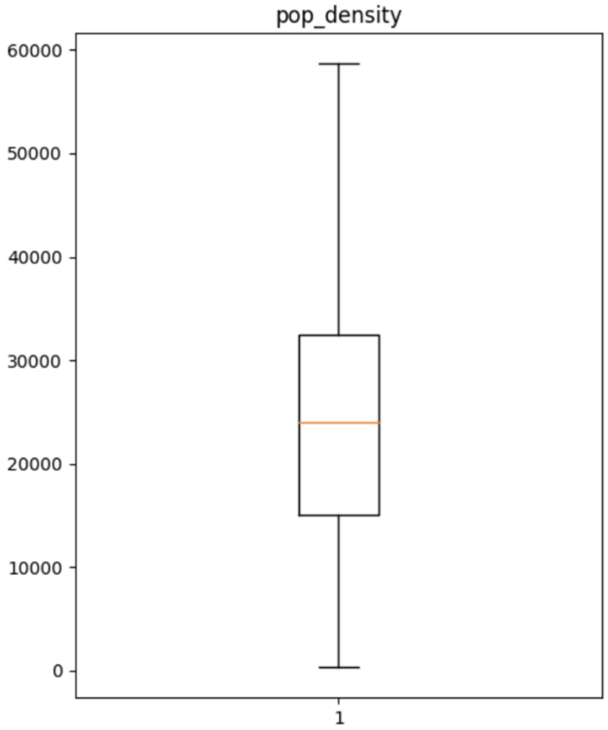
	income_level
mean	6.978774
std	1.074978
min	5.000000
25%	6.000000
50%	7.000000
75%	8.000000
max	10.000000



Population density in Seoul

Resource : Seoul Open Data Plaza
(https://data.seoul.go.kr)

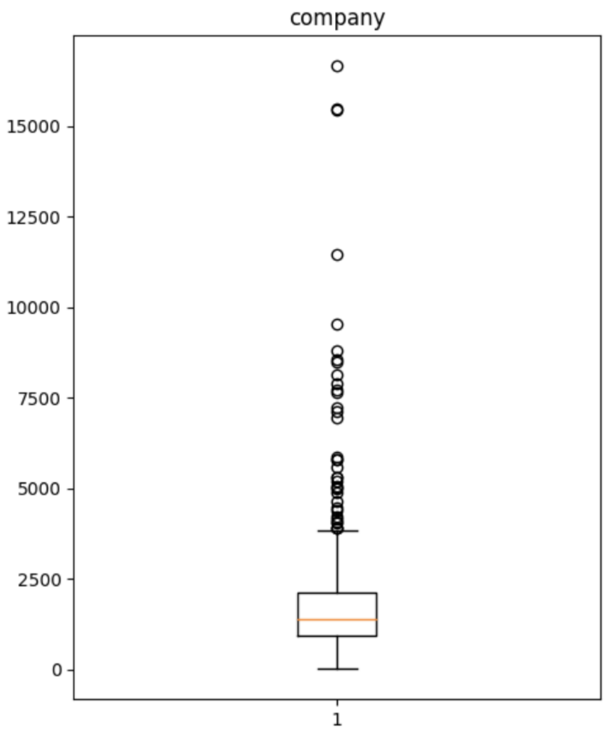
	pop_density
mean	24000.318396
std	11987.705656
min	308.000000
25%	15018.250000
50%	24001.000000
75%	32526.500000
max	58667.000000



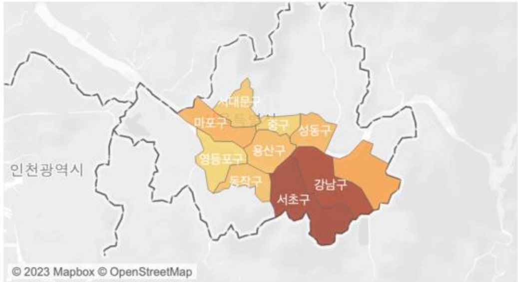
Number of Company in Seoul

Resource : Seoul Open Data Plaza
(https://data.seoul.go.kr)

	company
mean	1942.509434
std	1928.956080
min	17.000000
25%	958.000000
50%	1410.000000
75%	2123.000000
max	16661.000000



Gu Visualization

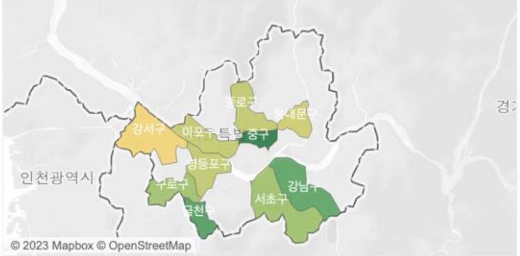


Gu_IncomeLevelRanking

RANK(Inco..	Gu	
1	서초구	8.500
2	강남구	8.455
3	송파구	7.519
4	마포구	7.375
5	성동구	7.353
6	용산구	7.313
7	동작구	7.267
8	서대문구	7.214
9	중구	7.133
10	영등포구	7.111

Gu_PDensityRanking

RANK(POP)	Gu	
1	관악구	29,830
2	양천구	29,212
3	동작구	28,543
4	구로구	28,285
5	도봉구	28,156
6	강서구	27,062
7	동대문구	26,031
8	노원구	25,805
9	성북구	25,696



Gu_CompanyRanking

RANK(Com..	Gu	
1	중구	4,008
2	금천구	3,381
3	강남구	3,229
4	서초구	2,608
5	구로구	2,584
6	영등포구	2,354
7	종로구	2,334
8	마포구	2,331
9	동대문구	2,237

Dong Visualization

Dong_IncomeRanking

RANK(Inco..	Dong	
1	대치1동	10.000
2	개포2동	9.000
	광장동	9.000
	대치2동	9.000
	도곡1동	9.000
	도곡2동	9.000
	목1동	9.000
	목5동	9.000
	무악동	9.000

Dong_DensityRanking

RANK(POP)	Dong	
1	행당2동	58,667
2	상계5동	57,916
3	구로4동	56,998
4	청림동	54,287
5	삼각산동	53,065
6	돈암2동	52,142
7	화곡8동	50,017
8	화곡1동	48,716
9	목4동	47,209
10	길음1동	46,939

Dong_CompanyRanking

RANK(Com..	Dong	
1	가산동	16,661
2	종로1·2·3·4가동	15,472
3	역삼1동	15,451
4	신당동	11,459
5	서교동	9,552
6	을지로동	8,812
7	회현동	8,566
8	광희동	8,472
9	여의동	8,132

Visualization of data

By visualization with Income Level, Population density, Number of Company....

- The areas with **a high income** level are **Gangnam and the central region** of Seoul.
- The population density is high in the **northeastern and southwestern** regions.
- The number of companies is abundant in **the Gangseo and Gangnam** areas.

4

Additional Data Analysis

For clustering

Income level

Number of company

Population density

```
result_df = pd.merge(df, company, on=['gu', 'dong'], how='left')
result_df
```

	gu	dong	code	income_level	pop_density	company
0	종로구	청운효자동	1111051500	8.0	5051	1028.0
1	종로구	사직동	1111053000	9.0	7980	3574.0
2	종로구	삼청동	1111054000	7.0	2011	732.0
3	종로구	부암동	1111055000	7.0	4648	599.0
4	종로구	평창동	1111056000	8.0	2121	761.0
...	...	...	...	...	...	...
419	강동구	성내2동	1174065000	6.0	37591	1836.0
420	강동구	성내3동	1174066000	7.0	33692	2439.0
421	강동구	길동	1174068500	7.0	21367	5037.0
422	강동구	둔촌1동	1174069000	7.0	308	17.0
423	강동구	둔촌2동	1174070000	7.0	28318	1553.0

For clustering

```python
# Assuming your DataFrame is named df
df['address'] = '서울시 ' + df['gu'] + ' ' + df['dong']
df
```

	gu	dong	code	income_level	pop_density	address
0	종로구	청운효자동	1111051500	8.0	5051	서울시 종로구 청운효자동
1	종로구	사직동	1111053000	9.0	7980	서울시 종로구 사직동
2	종로구	삼청동	1111054000	7.0	2011	서울시 종로구 삼청동
3	종로구	부암동	1111055000	7.0	4648	서울시 종로구 부암동
4	종로구	평창동	1111056000	8.0	2121	서울시 종로구 평창동
...	...	...	...	...	...	...
419	강동구	성내2동	1174065000	6.0	37591	서울시 강동구 성내2동
420	강동구	성내3동	1174066000	7.0	33692	서울시 강동구 성내3동
421	강동구	길동	1174068500	7.0	21367	서울시 강동구 길동
422	강동구	둔촌1동	1174069000	7.0	308	서울시 강동구 둔촌1동
423	강동구	둔촌2동	1174070000	7.0	28318	서울시 강동구 둔촌2동

Google API

```python
import requests

def geocode_address(address):
    api_key = 'AIzaSyBygRp967UbOGGODVYYOm6O0divESJ[redacted]
    base_url = 'https://maps.googleapis.com/maps/api/geocode/json'

    params = {
        'address': address,
        'key': api_key,
    }

    response = requests.get(base_url, params=params)
    data = response.json()

    if data['status'] == 'OK':
        location = data['results'][0]['geometry']['location']
        latitude = location['lat']
        longitude = location['lng']
        return latitude, longitude
    else:
        return None
```

X-Y coordinates

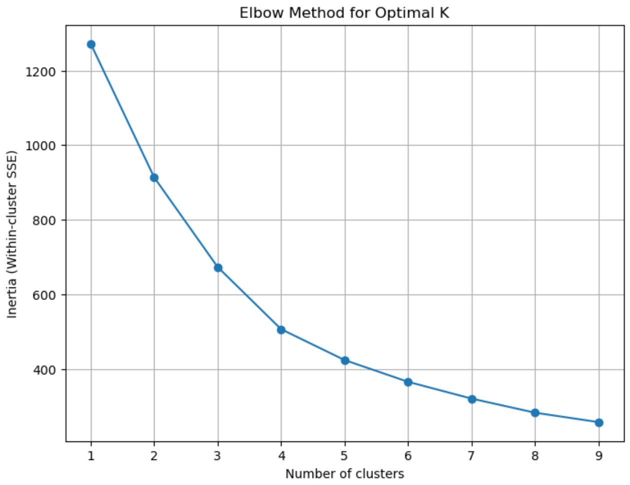
```python
df.iloc[:, 1:]
```

	gu	dong	code	income_level	pop_density	address	x	y
0	종로구	청운효자동	1111051500	8	5051	서울시 종로구 청운효자동	37.583776	126.970663
1	종로구	사직동	1111053000	9	7980	서울시 종로구 사직동	37.573878	126.970561
2	종로구	삼청동	1111054000	7	2011	서울시 종로구 삼청동	37.590765	126.981016
3	종로구	부암동	1111055000	7	4648	서울시 종로구 부암동	37.594759	126.965589
4	종로구	평창동	1111056000	8	2121	서울시 종로구 평창동	37.613029	126.974503
...	...	...	...	...	...	...	...	...
419	강동구	성내2동	1174065000	6	37591	서울시 강동구 성내2동	37.534486	127.127931
420	강동구	성내3동	1174066000	7	33692	서울시 강동구 성내3동	37.528451	127.133761
421	강동구	길동	1174068500	7	21367	서울시 강동구 길동	37.539616	127.145929
422	강동구	둔촌1동	1174069000	7	308	서울시 강동구 둔촌1동	37.522886	127.140539
423	강동구	둔촌2동	1174070000	7	28318	서울시 강동구 둔촌2동	37.531452	127.146715

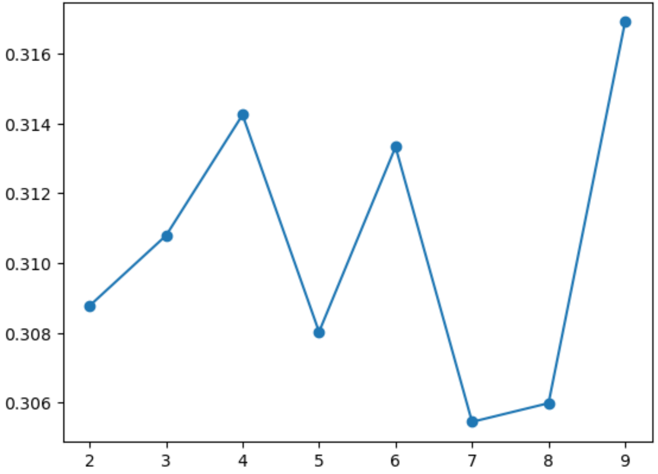
424 rows × 8 columns

K means Clustering

To find optimal K, Using Elbow method and Silhouette Analysis



Appropriate 4 or 5?



High is good 4, 6, 9

By considering two analysis, we select K as ***4***

K-means Clustering

```
kmeans.labels_
```

```
array([0, 0, 2, 2, 0, 0, 0, 2, 1, 1, 2, 2, 2, 3, 3, 3, 3, 0, 1, 1, 2, 2,
       1, 1, 1, 3, 3, 3, 3, 3, 3, 0, 2, 2, 2, 2, 0, 0, 2, 3, 0, 0, 2, 2,
       2, 0, 0, 2, 3, 3, 2, 2, 0, 3, 0, 3, 3, 0, 0, 0, 2, 2, 1, 2, 2, 2,
       3, 2, 3, 3, 2, 2, 0, 3, 2, 0, 2, 3, 2, 0, 1, 2, 3, 2, 3, 3, 3, 3,
       2, 2, 2, 2, 2, 3, 3, 2, 2, 3, 2, 2, 3, 3, 3, 2, 3, 3, 2, 2, 2, 2,
       2, 3, 2, 3, 3, 2, 3, 3, 2, 2, 2, 3, 3, 3, 3, 2, 3, 2, 3, 2, 3, 2,
       3, 3, 3, 3, 2, 2, 2, 3, 3, 2, 2, 3, 3, 2, 0, 3, 2, 3, 2, 3, 3, 3,
       3, 3, 2, 2, 2, 3, 3, 2, 0, 3, 0, 3, 2, 3, 2, 3, 2, 3, 3, 3, 3, 2,
       2, 2, 2, 2, 3, 2, 3, 2, 3, 3, 3, 3, 2, 2, 2, 0, 0, 3, 2, 2, 2, 2,
       2, 0, 2, 2, 0, 3, 3, 3, 0, 3, 3, 0, 2, 3, 0, 0, 1, 2, 2, 3, 3, 2,
       2, 0, 0, 3, 3, 3, 0, 3, 3, 2, 3, 2, 2, 2, 0, 3, 2, 3, 0, 0, 0, 3,
       2, 3, 3, 3, 3, 3, 3, 2, 3, 1, 2, 2, 2, 0, 2, 3, 2, 2, 0, 2, 1, 1,
       3, 3, 3, 2, 3, 2, 3, 3, 3, 2, 2, 1, 2, 3, 3, 3, 2, 2, 2, 2, 2, 2,
       1, 1, 3, 0, 2, 1, 2, 0, 3, 3, 3, 2, 3, 0, 3, 3, 3, 0, 2, 0, 0, 3,
       3, 0, 3, 0, 0, 3, 2, 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 2, 2, 3, 3,
       2, 3, 3, 2, 3, 2, 2, 2, 3, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 2, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2,
       0, 0, 2, 0, 0, 2, 0, 2, 3, 3, 2, 0, 3, 0, 2, 3, 3, 3, 3, 0, 0, 3,
       3, 1, 2, 2, 3, 0, 0, 0, 0, 0, 2, 2, 3, 0, 0, 2, 3, 0, 0, 3, 2, 3,
       3, 3, 3, 1, 2, 3])
```

```
len(kmeans.labels_)
```

```
424
```

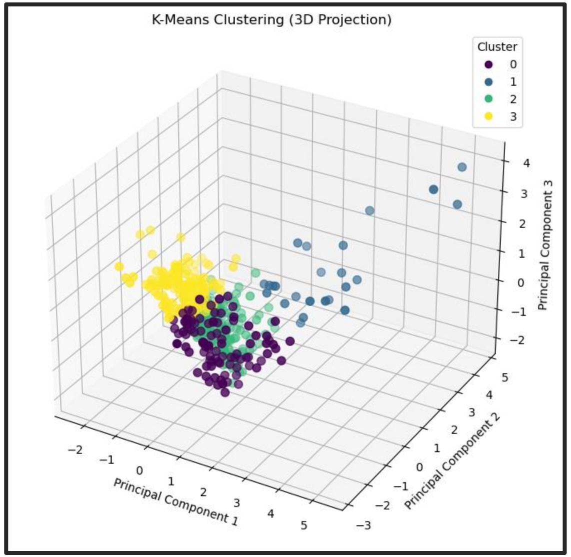
Clustering Results

	gu	dong	code	income_level	pop_density	company	label
0	종로구	청운효자동	1111051500	8.0	5051	1028.0	0
1	종로구	사직동	1111053000	9.0	7980	3574.0	0
2	종로구	삼청동	1111054000	7.0	2011	732.0	2
3	종로구	부암동	1111055000	7.0	4648	599.0	2
4	종로구	평창동	1111056000	8.0	2121	761.0	0
...	...	...	...	...	...	...	...
419	강동구	성내2동	1174065000	6.0	37591	1836.0	3
420	강동구	성내3동	1174066000	7.0	33692	2439.0	3
421	강동구	길동	1174068500	7.0	21367	5037.0	1
422	강동구	둔촌1동	1174069000	7.0	308	17.0	2
423	강동구	둔촌2동	1174070000	7.0	28318	1553.0	3

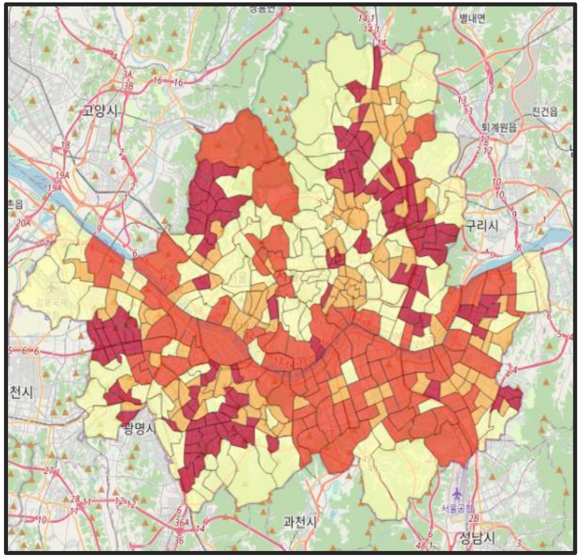
424 rows × 7 columns

Labeling Data

Visualization



3D clustering



Follium Mapping

Clustering Result Interpretation

```
df.iloc[:,4:].groupby('label').mean()
```

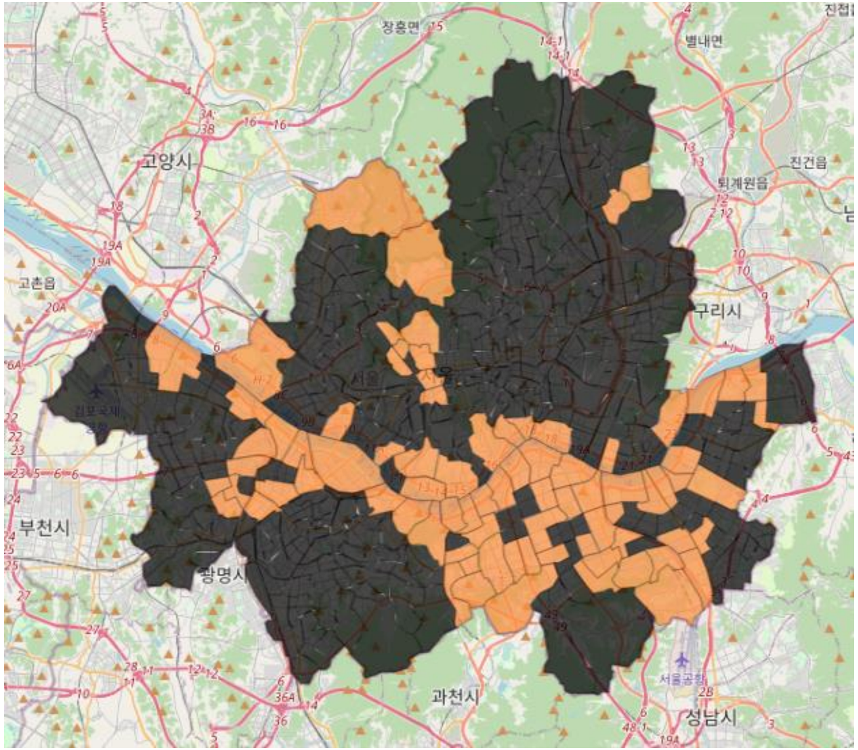
label	income_level	pop_density	company
0	6.383562	16223.506849	1477.109589
1	6.601266	36013.151899	1527.537975
2	8.416667	18850.760417	1776.375000
3	7.333333	12823.000000	8170.125000

Classify High/Mid/Low

label	income_LV	pop_density	company
cluster 0	Low	Mid	Low
cluster 1	Mid	High	Low
cluster 2	High	Mid	Mid
cluster 3	Mid	Low	High

Consider cluster 2

Selected Cluster 2



5

Future Plan

Analysis Steps

STEP 1) MCLP for specific location based on kneed point & cluster2 region

STEP 2) Flight Route Selection

2-1) Considering zone where air-taxi cannot fly (ex. Air prohibition zone)

2-2) Using Dijkstra or BFS algorithm to select the optimal shortest path

STEP 3) Calculate route efficiency

3-1) Compare time required for distance to each station with existing transportation

Reference

1) 성유진, 김새벽, 최윤수, 조성길. (2023). 도심항공교통(UAM) 버티포트(Vertiport) 최적입지 선정 및 경로선정을 통한 이동시간 효용성 분석. 대한공간정보학회
2) [쏠림 사회 한국, 강남 리포트] '교통의 요지'라는 강남의 역설...더 많이 모이고 더 막힌다. 경향신문. (2023, October 5). https://m.khan.co.kr/economy/economy-general/article/202310051500011
3) [취재수첩] 대중교통 혼잡문제, 국토부가 나서야. 한국경제. (2023, May 5). https://www.hankyung.com/article/2023050556961
4) 소방차 출동도 어려운 "테헤란로" 교통혼잡 이대로 괜찮은가? 서울시 - 내 손안에 서울. (2023, May 2). https://mediahub.seoul.go.kr/archives/2007809

Thank you