북촌한옥마을 CCTV 분석

목차

문제 정의

북촌한옥마을은 **관광명소**이므로 이 지역을 오가는 유동인구는 대부분 **관광객**입니다. 또한 **서울 중심지**에 위치해 있기 때문에 **출퇴근을 하는 사람**들도 많을 것 입니다. CCTV디바이스(위치)별, 월 별, 요일 별, 시간 별 CCTV 유동 인구 추이를 통해 북촌한옥마을 지역을 오가는 사람들의 특징을 파악하여 그에 맞는 편의점 상품들을 준비할 수 있도록 분석해야 합니다.

- 가설
    1. 북촌한옥마을과 가까운 곳에 위치해 있는 CCTV에 유동인구가 많을 것 같다.
    2. 날씨가 추운 12월, 1월 보다는 10월 11월에 유동인구가 많을 것 같다.
    3. 다음날 출근해야 하는 일요일보다는 토요일에 유동인구가 더 많을 것 같고 평일 중에는 내일 쉬는 날인 금요일에 유동인구가 가장 많을 것 같다. 또한 수요일은 평일 중 중간에 껴있기 때문에 유동인구가 가장 적을 것 같다.
    4. 새벽 시간대(0시~5시)에 유동인구가 가장 적을 것이고 출근 시간대(6시~9시)와 퇴근 시간대(18시~20시)에 유동인구가 증가했다가 그 외 시간은 점차 감소하는 추세일 것이다.

다음과 같은 가설들을 입증해나가면서 유동인구 추이에 따라 적절하게 상품들을 준비할 수 있는 근거를 제시해야 합니다.

데이터 수집

```
sample = '52586e6e70706a683133304a4f55714d' # api 인증키
file_type = 'json'
service_name = 'BukChonInOutPeopleInfo'

# 전체 데이터의 수를 알기 위해 처음 1000개 데이터만 불러오기
url = os.path.join("http://openapi.seoul.go.kr:8088/", sample, file_type, service_name,"1","1000")

data = requests.get(url).text
dict_data = json.loads(data)
row_data = dict_data['BukChonInOutPeopleInfo']['row']
df = pd.DataFrame(row_data) # 초기 1000개의 데이터

# list_total_count 값을 알아냈기 때문에 전체 데이터 불러오기
for i in range(1001, dict_data['BukChonInOutPeopleInfo']['list_total_count'] + 1,1000):
    url = os.path.join("http://openapi.seoul.go.kr:8088/", sample, file_type, service_name, str(i), str(i+999))

    if (i + 999) > dict_data['BukChonInOutPeopleInfo']['list_total_count']:
        url = os.path.join("http://openapi.seoul.go.kr:8088/", sample, file_type, service_name,str(i), str(dict_data['BukChonInOutPeopleInf

    data = requests.get(url).text
    dict_data = json.loads(data)
    row_data = dict_data['BukChonInOutPeopleInfo']['row']
    df_new = pd.DataFrame(row_data)
    df = pd.concat([df,df_new], ignore_index = True)
```

```
df
```

```
# 시간순으로 정렬
df.sort_values(by='STARTTIME')
```

	DEVICEID	DEVICENAME	DESCRIPTION	STARTTIME	ENDTIME	INCOUNT	OUTCOUNT
466	2.0	계동길 69	계동교회 앞	2022-10-27 00:00:40	2022-10-27 00:10:41	31.0	40.0
467	4.0	북촌로5가길 38	삼청파출소 사잇길	2022-10-27 00:00:40	2022-10-27 00:10:41	5.0	11.0
473	4.0	북촌로5가길 38	삼청파출소 사잇길	2022-10-27 00:10:41	2022-10-27 00:20:40	7.0	13.0
472	2.0	계동길 69	계동교회 앞	2022-10-27 00:10:41	2022-10-27 00:20:40	54.0	70.0
475	4.0	북촌로5가길 38	삼청파출소 사잇길	2022-10-27 00:20:40	2022-10-27 00:30:40	7.0	30.0
...	...	...	...	...	...	...	...
29329	1.0	율곡로3길 50	덕성여고 앞	2023-01-30 20:40:41	2023-01-30 20:50:41	6111.0	4704.0
29330	2.0	계동길 69	계동교회 앞	2023-01-30 20:50:41	2023-01-30 21:00:41	2946.0	4655.0
29331	1.0	율곡로3길 50	덕성여고 앞	2023-01-30 20:50:41	2023-01-30 21:00:41	6131.0	4727.0
29332	2.0	계동길 69	계동교회 앞	2023-01-30 21:00:41	2023-01-30 21:10:41	2975.0	4669.0
29333	1.0	율곡로3길 50	덕성여고 앞	2023-01-30 21:00:41	2023-01-30 21:10:41	6173.0	4767.0

29334 rows × 7 columns

```
df.isnull().sum()
```

```
DEVICEID      0
DEVICENAME    0
DESCRIPTION   0
STARTTIME     0
ENDTIME       0
INCOUNT       0
OUTCOUNT      0
dtype: int64
```

- 데이터는 실시간으로 반영되는 데이터기 때문에 보고서를 작성하는 시간을 기준으로 수집한 데이터를 사용하였다.
- 수집 결과 2022년 10월 27일 부터 2023년 1월 30일 9시까지 데이터인 것을 확인 할 수 있습니다.
- 변수 의미

1	DEVICEID	카메라번호
2	DEVICENAME	주소
3	DESCRIPTION	상세설명
4	STARTTIME	측정 시작 시간
5	ENDTIME	측정 종료 시간
6	INCOUNT	카메라 통과 인원 (IN)
7	OUTCOUNT	카메라 통과 인원 (OUT)

- 결측치는 없는 것을 확인할 수 있습니다.

데이터 전처리

```
df_new = df.copy() # 원본 데이터 보호

# ENDTIME 변수 타입 변경
df_new['ENDTIME'] = pd.to_datetime(df_new['ENDTIME'])

# 파생변수 만들기
df_new['MONTH'] = df_new['ENDTIME'].dt.month
df_new['HOUR'] = df_new['ENDTIME'].dt.hour
df_new['MON_SUN'] = df_new['ENDTIME'].dt.weekday # 0-월 1-화 ... 6-일

# 평일/주말 구분 변수
df_new.loc[df_new['MON_SUN'] == 0, 'WEEKDAY'] = '평일'
df_new.loc[df_new['MON_SUN'] == 1, 'WEEKDAY'] = '평일'
df_new.loc[df_new['MON_SUN'] == 2, 'WEEKDAY'] = '평일'
df_new.loc[df_new['MON_SUN'] == 3, 'WEEKDAY'] = '평일'
df_new.loc[df_new['MON_SUN'] == 4, 'WEEKDAY'] = '평일'
df_new.loc[df_new['MON_SUN'] == 5, 'WEEKDAY'] = '주말'
df_new.loc[df_new['MON_SUN'] == 6, 'WEEKDAY'] = '주말'

# 요일변수를 숫자에서 문자로 변경
df_new.loc[df_new['MON_SUN'] == 0, 'MON_SUN'] = '월요일'
df_new.loc[df_new['MON_SUN'] == 1, 'MON_SUN'] = '화요일'
df_new.loc[df_new['MON_SUN'] == 2, 'MON_SUN'] = '수요일'
df_new.loc[df_new['MON_SUN'] == 3, 'MON_SUN'] = '목요일'
df_new.loc[df_new['MON_SUN'] == 4, 'MON_SUN'] = '금요일'
df_new.loc[df_new['MON_SUN'] == 5, 'MON_SUN'] = '토요일'
df_new.loc[df_new['MON_SUN'] == 6, 'MON_SUN'] = '일요일'

# 필요없는 변수 지우기
df_new.drop(['DEVICENAME', 'STARTTIME', 'ENDTIME'], axis = 1, inplace = True)

df_new
```

	DEVICEID	DESCRIPTION	INCOUNT	OUTCOUNT	MONTH	HOUR	MON_SUN	WEEKDAY
0	2.0	계동교회 앞	9386.0	13749.0	11	21	수요일	평일
1	1.0	덕성여고 앞	7903.0	5279.0	11	21	수요일	평일
2	2.0	계동교회 앞	9398.0	13776.0	11	21	수요일	평일
3	1.0	덕성여고 앞	7927.0	5312.0	11	21	수요일	평일
4	2.0	계동교회 앞	9414.0	13800.0	11	21	수요일	평일
...	...	...	...	...	...	...	...	...
29329	1.0	덕성여고 앞	6111.0	4704.0	1	20	월요일	평일
29330	2.0	계동교회 앞	2946.0	4655.0	1	21	월요일	평일
29331	1.0	덕성여고 앞	6131.0	4727.0	1	21	월요일	평일
29332	2.0	계동교회 앞	2975.0	4669.0	1	21	월요일	평일
29333	1.0	덕성여고 앞	6173.0	4767.0	1	21	월요일	평일

29334 rows × 8 columns

- DEVICENAME 변수는 DEVICEID와 DESCRIPTION 변수로도 CCTV의 특징을 구분할 수 있기 때문에 제거했습니다.
- STARTTIME, ENDTIME은 ENDTIME을 기준으로 월, 시간, 요일, 평일/주말 여부 변수를 파생 했기 때문에 제거했습니다.

데이터 시각화 및 해석

디바이스 유형(위치)별

```
# ID별 데이터 수 확인
df_new['DEVICEID'].value_counts()
```

```
# 위치별 데이터 수 확인
df_new['DESCRIPTION'].value_counts()
```

```
2.0    13712
1.0    12049
4.0     3573
Name: DEVICEID, dtype: int64
```

```
계동교회 앞        13712
덕성여고 앞        12049
삼청파출소 사잇길     3573
Name: DESCRIPTION, dtype: int64
```

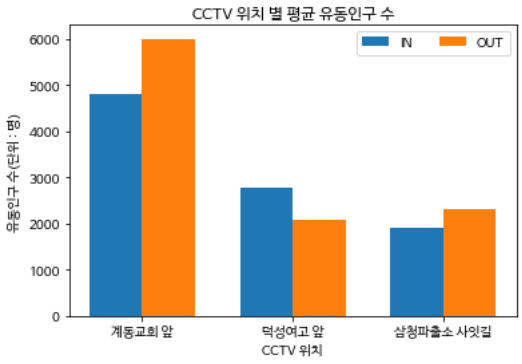
- 디바이스 1 : 덕성여고 앞
- 디바이스 2 : 계동교회 앞
- 디바이스 4 : 삼청파출소 사잇길

```
# 위치별 평균 유동인구 관련 변수 데이터 추출
df_device = df_new.groupby('DESCRIPTION')[['INCOUNT','OUTCOUNT']].mean().round(0).reset_index()
df_device
```

	DESCRIPTION	INCOUNT	OUTCOUNT
0	계동교회 앞	4817.0	6002.0
1	덕성여고 앞	2778.0	2079.0
2	삼청파출소 사잇길	1913.0	2299.0

```
# 시각화
w = 0.35
idx = np.arange(3)

plt.title('CCTV 위치 별 평균 유동인구 수')
plt.xlabel('CCTV 위치')
plt.ylabel('유동인구 수(단위 : 명)')
plt.b ar(idx - w/2, df_device['INCOUNT'], width = w, label = 'IN')
plt.bar(idx + w/2, df_device['OUTCOUNT'], width = w, label = 'OUT')
plt.xticks(idx, df_device['DESCRIPTION'])
plt.legend(ncol = 2)
plt.show()
```

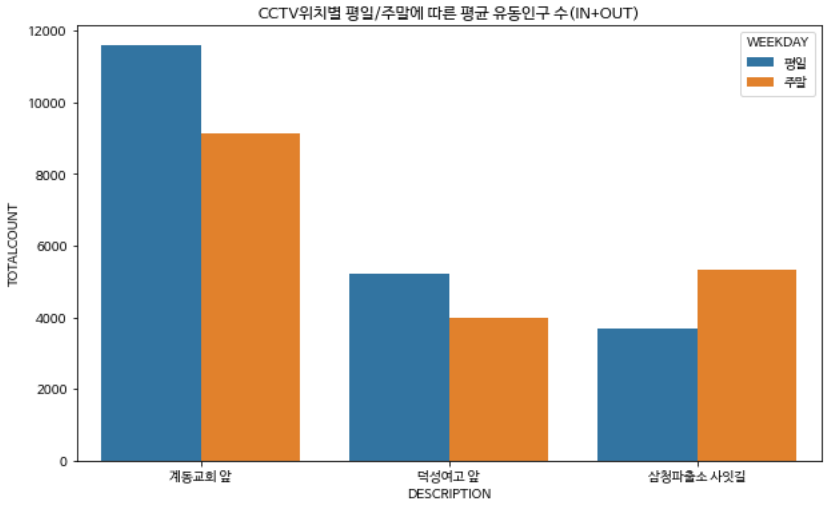


- 가설 1 검증 : CCTV위치가 3가지가 있는데 북촌한옥마을과 가장 가까운 계동교회 앞쪽이 평균 유동인구가 많은 것을 확인할 수 있었다.
- 추가 가설 : 평일에는 출퇴근하는 사람이 많기 때문에 안국역과 가까운 덕성여고 앞쪽에 유동 인구가 많을 것이고 주말에는 놀러나온 사람들이 많기 때문에 북촌한옥마을과 가까운 계동교회 앞쪽에 유동 인구가 많을 것이다.

```
# 위치별 평일/주말 여부에 따른 평균 유동인구 관련 데이터 추출
df_day = df_new.groupby(['DESCRIPTION', 'WEEKDAY'])[['INCOUNT','OUTCOUNT']].mean().round(0).sort_values(by = ['WEEKDAY'], ascending = False

# 총 유동인구 변수 생성
df_day['TOTALCOUNT'] = df_day['INCOUNT'] + df_day['OUTCOUNT']
df_day
```

```
# 시각화
plt.figure(figsize = (10,6))
sns.barplot(data = df_day, x = 'DESCRIPTION', y = 'TOTALCOUNT', hue = 'WEEKDAY')
plt.title('CCTV위치별 평일/주말에 따른 평균 유동인구 수(IN+OUT)')
plt.show()
```



- 평일, 주말 상관없이 모두 계동교회 앞쪽이 평균 유동인구 수가 가장 많았다.
- 주말에 더 유동인구가 많을 것이라고 예상했던 것과는 달리 대체로 평일에 더 유동인구가 많았고 삼청동 문화거리와 가까운 삼청파출소 사잇길쪽만 주말에 더 증가하였다.

월별

```
# 월별 데이터 수 확인
df_new['MONTH'].value_counts()
```

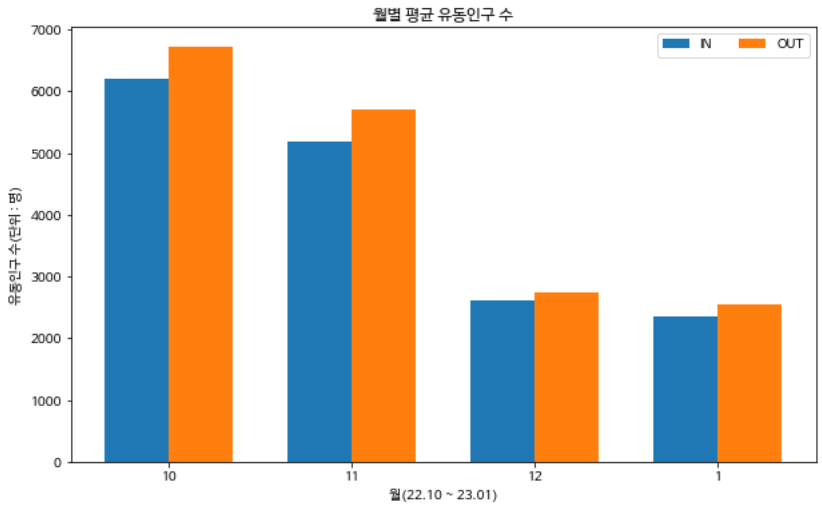
```
11    10496
12     8863
1      8545
10     1430
Name: MONTH, dtype: int64
```

```
# 월별 평균 유동인구 관련 변수 데이터 추출
df_month = df_new.groupby('MONTH')[['INCOUNT','OUTCOUNT']].mean().round(0).reset_index()
month_order = [1, 2, 3, 0]
df_month = df_month.loc[month_order] # 10월 ~ 1월 순으로 정렬
df_month
```

	MONTH	INCOUNT	OUTCOUNT
1	10	6206.0	6715.0
2	11	5182.0	5709.0
3	12	2602.0	2734.0
0	1	2343.0	2553.0

```
# 시각화
w = 0.35
idx = np.arange(4)

plt.figure(figsize = (10, 6))
plt.title('월별 평균 유동인구 수')
plt.xlabel('월(22.10 ~ 23.01)')
plt.ylabel('유동인구 수(단위 : 명)')
plt.bar(idx - w/2, df_month['INCOUNT'], width = w, label = 'IN')
plt.bar(idx + w/2, df_month['OUTCOUNT'], width = w, label = 'OUT')
plt.xticks(idx, df_month['MONTH'])
plt.legend(ncol = 2)
plt.rc("axes", unicode_minus=False)
plt.show()
```



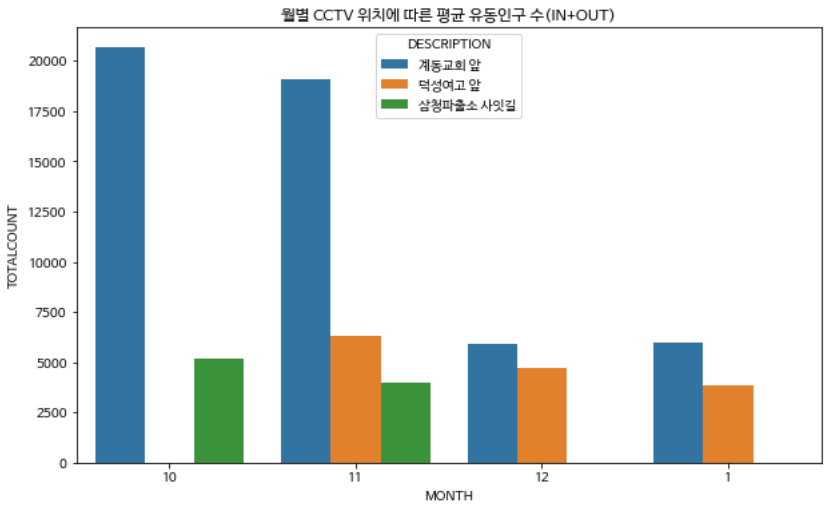
- 가설 2 검증 : 날씨가 추운 12월, 1월이 10월,11월보다 유동인구가 적은 것을 확인할 수 있었다.
- 추가 가설 : 학교 근처인 덕성여고 앞 부근은 특히 방학 시즌인 12월, 1월에 유동인구가 더 적을 것이다.

```
# 월별 위치에 따른 평균 유동인구 관련 데이터 추출
df_month2 = df_new.groupby(['MONTH','DESCRIPTION'])[['INCOUNT','OUTCOUNT']].mean().round(0).sort_values(by = ['DESCRIPTION']).reset_index()

# 총 유동인구 변수 생성
df_month2['TOTALCOUNT'] = df_month2['INCOUNT'] + df_month2['OUTCOUNT']
df_month2
```

	MONTH	DESCRIPTION	INCOUNT	OUTCOUNT	TOTALCOUNT
0	1	계동교회 앞	2496.0	3451.0	5947.0
1	10	계동교회 앞	10024.0	10652.0	20676.0
2	11	계동교회 앞	8714.0	10364.0	19078.0
3	12	계동교회 앞	2444.0	3491.0	5935.0
4	1	덕성여고 앞	2191.0	1654.0	3845.0
5	11	덕성여고 앞	3550.0	2758.0	6308.0
6	12	덕성여고 앞	2761.0	1976.0	4737.0
7	10	삼청파출소 사잇길	2388.0	2777.0	5165.0
8	11	삼청파출소 사잇길	1794.0	2179.0	3973.0

```
# 시각화
plt.figure(figsize = (10,6))
sns.barplot(data = df_month2, x = 'MONTH', y = 'TOTALCOUNT', hue = 'DESCRIPTION', order = [10, 11, 12, 1])
plt.title('월별 CCTV 위치에 따른 평균 유동인구 수(IN+OUT)')
plt.show()
```



- 덕성여고 앞의 경우 10월에는 데이터가 존재하지 않았고 11월에 비해 12월 1월이 크게 감소하는 추세는 아니었다. 오히려 계동교회 앞의 경우가 더 크게 감소하는 추세였다. → 관광객의 수가 많이 차지하고 있다는 근거
- 아쉬운 점 : 삼청 파출소 데이터가 11월까지 존재하고 덕성여고 데이터가 10월에는 존재하지 않아 형평성에 어긋나는 것 같다.

요일별

```
# 요일별 데이터 수 확인
df_new['MON_SUN'].value_counts()
```

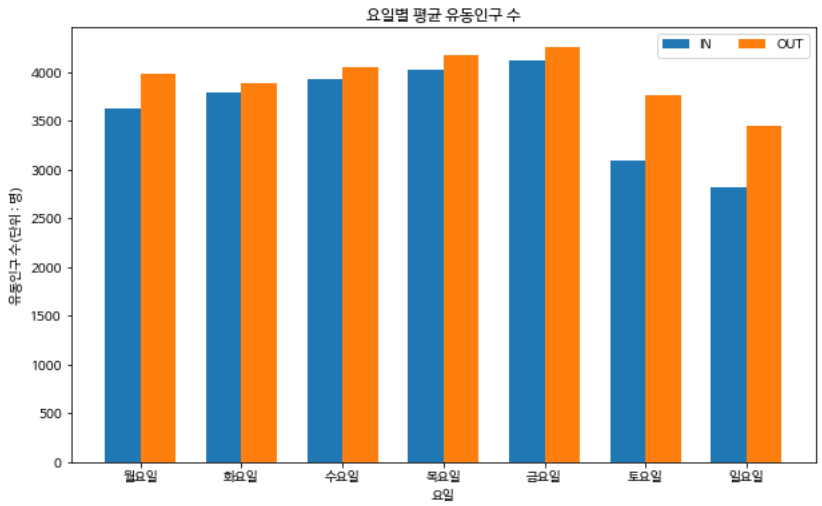
```
목요일    4290
일요일    4289
금요일    4289
토요일    4289
월요일    4173
수요일    4003
화요일    4001
Name: MON_SUN, dtype: int64
```

```
# 요일별 평균 유동인구 관련 변수 데이터 추출
df_week = df_new.groupby('MON_SUN')[['INCOUNT','OUTCOUNT']].mean().round(0).reset_index()
week_order = [3,6,2,1,0,5,4]
df_week = df_week.loc[week_order]
df_week
```

	MON_SUN	INCOUNT	OUTCOUNT
3	월요일	3623.0	3981.0
6	화요일	3794.0	3893.0
2	수요일	3925.0	4053.0
1	목요일	4033.0	4180.0
0	금요일	4123.0	4257.0
5	토요일	3094.0	3772.0
4	일요일	2819.0	3447.0

```
# 시각화
w = 0.35
idx = np.arange(7)

plt.figure(figsize = (10, 6))
plt.title('요일별 평균 유동인구 수')
plt.xlabel('요일')
plt.ylabel('유동인구 수(단위 : 명)')
plt.bar(idx - w/2, df_week['INCOUNT'], width = w, label = 'IN')
plt.bar(idx + w/2, df_week['OUTCOUNT'], width = w, label = 'OUT')
plt.xticks(idx, df_week['MON_SUN'])
plt.legend(ncol = 2)
plt.show()
```



- 가설 3 검증 : 먼저 일요일보다는 토요일에 유동인구가 많은 것을 확인할 수 있었고 평일 중에서도 금요일에 가장 많은 것을 확인할 수 있었다. 하지만 예측과 달리 평일 중 수요일에 유동인구가 가장 적지 않았고 월요일부터 금요일까지 점차 증가하는 형태임을 알 수 있었다.

시간별

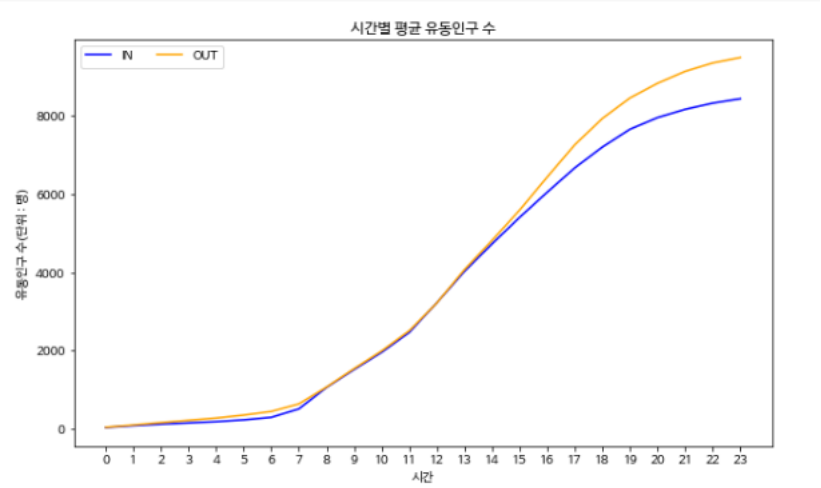
```
# 시간별 데이터 수 확인
df_new['HOUR'].value_counts()
```

```
17    1236
19    1236
16    1236
15    1236
14    1236
18    1236
13    1231
1     1230
20    1230
10    1230
11    1230
5     1230
9     1230
7     1230
12    1230
3     1230
4     1230
6     1230
8     1230
2     1230
23    1224
22    1224
21    1224
0     1015
Name: HOUR, dtype: int64
```

```
# 시간별 평균 유동인구 관련 변수 데이터 추출
df_hour = df_new.groupby('HOUR')[['INCOUNT','OUTCOUNT']].mean().round(0).reset_index()
df_hour
```

	HOUR	INCOUNT	OUTCOUNT
0	0	32.0	37.0
1	1	78.0	95.0
2	2	117.0	157.0
3	3	149.0	215.0
4	4	181.0	274.0
5	5	226.0	350.0
6	6	292.0	442.0
7	7	507.0	635.0
8	8	1054.0	1064.0
9	9	1512.0	1536.0
10	10	1953.0	1989.0
11	11	2461.0	2505.0
12	12	3219.0	3215.0
13	13	4021.0	4067.0
14	14	4724.0	4813.0
15	15	5402.0	5587.0
16	16	6045.0	6433.0
17	17	6668.0	7258.0
18	18	7200.0	7930.0
19	19	7655.0	8455.0
20	20	7953.0	8830.0
21	21	8164.0	9135.0
22	22	8323.0	9350.0
23	23	8435.0	9489.0

```
# 시각화
plt.figure(figsize = (10, 6))
plt.title('시간별 평균 누적 유동인구 수')
plt.xlabel('시간')
plt.ylabel('유동인구 수(단위 : 명)')
plt.plot(df_hour['HOUR'], df_hour['INCOUNT'], 'b', label = 'IN')
plt.plot(df_hour['HOUR'], df_hour['OUTCOUNT'], 'orange', label = 'OUT')
plt.xticks(df_hour['HOUR'])
plt.legend(ncol = 2)
plt.show()
```

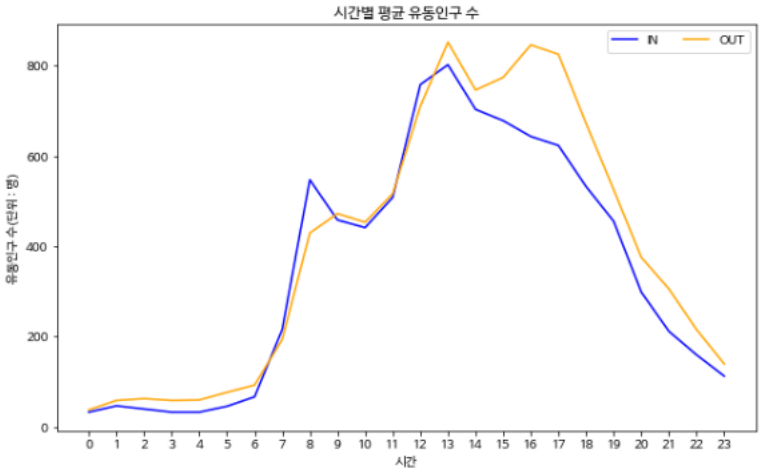



```
# 누적이 아닌 1시간별 증가하는 유동인구 수 데이터
for i in range(1, 24):
    df_hour.loc[i,"INCOUNT2"] = df_hour.loc[i, "INCOUNT"] - df_hour.loc[i-1, "INCOUNT"]
    df_hour.loc[i,"OUTCOUNT2"] = df_hour.loc[i, "OUTCOUNT"] - df_hour.loc[i-1, "OUTCOUNT"]

df_hour.loc[0,"INCOUNT2"] = df_hour.loc[0,"INCOUNT"]
df_hour.loc[0,"OUTCOUNT2"] = df_hour.loc[0,"OUTCOUNT"]
df_hour
```

	HOUR	INCOUNT	OUTCOUNT	INCOUNT2	OUTCOUNT2
0	0	32.0	37.0	32.0	37.0
1	1	78.0	95.0	46.0	58.0
2	2	117.0	157.0	39.0	62.0
3	3	149.0	215.0	32.0	58.0
4	4	181.0	274.0	32.0	59.0
5	5	226.0	350.0	45.0	76.0
6	6	292.0	442.0	66.0	92.0
7	7	507.0	635.0	215.0	193.0
8	8	1054.0	1064.0	547.0	429.0
9	9	1512.0	1536.0	458.0	472.0
10	10	1953.0	1989.0	441.0	453.0
11	11	2461.0	2505.0	508.0	516.0
12	12	3219.0	3215.0	758.0	710.0
13	13	4021.0	4067.0	802.0	852.0
14	14	4724.0	4813.0	703.0	746.0
15	15	5402.0	5587.0	678.0	774.0
16	16	6045.0	6433.0	643.0	846.0
17	17	6668.0	7258.0	623.0	825.0
18	18	7200.0	7930.0	532.0	672.0
19	19	7655.0	8455.0	455.0	525.0
20	20	7953.0	8830.0	298.0	375.0
21	21	8164.0	9135.0	211.0	305.0
22	22	8323.0	9350.0	159.0	215.0
23	23	8435.0	9489.0	112.0	139.0

```
# 시각화
plt.figure(figsize = (10, 6))
plt.title('시간별 평균 유동인구 수')
plt.xlabel('시간')
plt.ylabel('유동인구 수(단위 : 명)')
plt.plot(df_hour['HOUR'], df_hour['INCOUNT2'], 'b', label = 'IN')
plt.plot(df_hour['HOUR'], df_hour['OUTCOUNT2'], 'orange', label = 'OUT')
plt.xticks(df_hour['HOUR'])
plt.legend(ncol = 2)
plt.show()
```



- 검증하기 전에 우선 시간별 시각화를 통해 INCOUNT와 OUTCOUNT변수가 시간에 따른 누적 유동인구 수 임을 알게 되었다. 증가량을 알기 위해 현 시간 평균 누적 유동인구 수 1시간 전 평균 누적 유동인구수를 빼주었다.
- 가설 4 검증 : 새벽시간대(0시~5시)에 유동인구가 가장 적었고 이후 8시까지 급격하게 증가하다가 10시까지 잠시 감소하고 이후 점심시간대(11시~ 13시)까지 또 급격하게 증가하였고 이후 점차 감소하는 추세로 변화가 있었다. 퇴근 시간에 유동인구가 증가하는 것이 아닌 점심시간에 유동인구가 증가하는 것을 참고해야 할 것 같다.

분석을 마치며

- 북촌쪽은 출퇴근 유동인구보다는 북촌한옥마을이나 삼청동 문화거리를 목적으로 온 유동인구가 더 많은 것 같다. 그러므로 관광객을 대상으로 한 물품을 배치하는 것이 좋아보인다.(ex. 물) + 무인사물함 같은 것도 설치하는 것이 좋아보인다.
- 특히 관광객 중 외국인도 있을 수 있기 때문에 외국인들을 위한 물품도 배치하는 것이 좋아 보인다.
    - 출처 : https://www.donga.com/news/article/all/20230102/117264509/1
- 점심시간대에 유동인구가 증가한 것으로 보아 점심식사 이후 간단하게 먹을 수 있는 커피나 껌 같은 물품도 배치하는 것이 좋아보인다.
- 12월, 1월에 유동인구수가 다른 월에 비해 많지 않기 때문에 방한용품을 현재 상황보다 크게 늘릴 필요는 없을 것 같다.

향후 계획

- 요일별 / 시간별 분석에서도 추가 가설을 세워보자
- 10월 ~ 1월 외에도 다른 월 데이터를 포함 시켜 분석해보자
- 최대한 공평하게 데이터 수를 구성하자