환경요인에 따른 지역별 범죄율 분석
2025

작성자: 202144013 기경현

인하공전 컴퓨터 정보공학과
빅데이터 수업 A반
민정혜 교수님

목차

2. 사용 데이터

2.1 서울시 범죄 발생 지역별 통계

https://data.seoul.go.kr/dataList/316/S/2/datasetView.do

사이트에서 설정을 이용해 범죄종류를 통합하고
2018년도에서 2024년도 까지의 파일 다운로드

2.2 서울시 구별 주민등록 인구

구청에 등록 되어있는 구별 주민 등록 인구 데이터

2.3 서울시 구별 cctv 설치 정보

https://data.seoul.go.kr/dataList/OA-21097/F/1/datasetView.do

범죄 예방 목적으로 설치되 CCTV데이터

2.4 서울시 비상벨 설치 정보

https://www.localdata.go.kr/lif/lifeCtacDataView.do?menuNo=40003

비상벨이란
 비상 벨은 위급 상황 발생 시 사람을 돕기 위해 설치된 장치

2.5 서울시 지역별 유흥주점 정보

2.4과 같은 사이트

유흥 주점이란
 유흥주점은 주로 주류를 판매하면서, 유흥 종사자를 두거나 유흥 시설(노래방, 춤추는 공간 등)을 설치하여 손님이 술과 함께 노래를 부르거나 춤을 추는 등 유흥 접객 행위가 허용되는 식품접객업으로, 일반음식점과 달리 손님의 흥을 돋우는 서비스가 핵심

3. 데이터 전처리

3.1 범죄 발생 지역별 통계

```
import pandas as pd

filepath = "/content/5대+범죄+발생현황_20251207115651.csv"

# ============================================================
# 1. 파일 읽기 (헤더 3줄 건너뛰기)
# ============================================================
df = pd.read_csv(filepath, skiprows=3, encoding='utf-8')


print("원본 데이터 확인")
print(df.head(10))
print(f"₩n전체 행 수: {len(df)}")
```

결과)

```
원본 데이터 확인
   합계   종로구  3690  3846  3102  2712  3138  2981  2765
0  합계    중구  4030  4327  3411  2861  3071  3348  2955
1  합계   용산구  3411  3313  2969  2381  2967  3021  3322
2  합계   성동구  2457  2512  2362  2112  2194  2023  2117
3  합계   광진구  3915  4011  3601  3087  3619  3424  2870
4  합계  동대문구  3680  3692  3401  2959  3253  2957  3216
5  합계   중랑구  4288  4268  3726  3210  3599  3324  3169
6  합계   성북구  3042  2877  2567  2411  2749  2411  2184
7  합계   강북구  3437  3838  2770  2301  2832  2497  2289
8  합계   도봉구  2249  2110  2179  1860  2141  1921  1741
9  합계   노원구  4007  4153  3743  3425  3896  3567  3101
```

해당 데이터를 원하는 형식으로 변경
컬럼명을 지정하고 불필요한 행제거 후 csv파일로 저장

```
# ============================================================
# 2. 컬럼명 정리
# ============================================================
df.columns = ['category', 'region', '2018', '2019', '2020', '2021', '2022', '2023', '2024']

# ============================================================
# 3. '소계' 행 제거 및 구 데이터만 추출
# ============================================================
df_clean = df[df['region'] != '소계'].copy()

print(f"₩n '소계' 제거 후: {len(df_clean)}행")
print("₩n 포함된 자치구:")
print(df_clean['region'].unique())

# ============================================================
# 4. Wide format → Long format 변환
# ============================================================
df_long = df_clean.melt(
    id_vars=['region'],
    value_vars=['2018', '2019', '2020', '2021', '2022', '2023', '2024'],
    var_name='year',
    value_name='total'
)

# ============================================================
# 5. 데이터 타입 변환
# ============================================================
df_long['year'] = df_long['year'].astype(int)
df_long['total'] = pd.to_numeric(df_long['total'], errors='coerce').fillna(0).astype(int)

# ============================================================
# 6. 정렬
# ============================================================
df_long = df_long.sort_values(['region', 'year']).reset_index(drop=True)
```

3.2 서울시 구별 주민등록 인구

데이터 양식

동별(2)	2025 3/4	2025 3/4	2025 3/4	2025 3/4	2018	2018	2018	2018	2019
동별(2)	세대 (세대)	계 (명)	한국인 (명)	등록외국인	세대 (세대)	계 (명)	한국인 (명)	등록외국인	세대 (세대)
종로구	72402	149165	137545	11620	73735	163026	153065	9961	73947
중구	64259	128942	118458	10484	61502	135633	125725	9908	62739
용산구	103187	215716	202636	13080	108974	245090	228999	16091	110126

데이터 불러오기

```
# 1. 데이터 로드
population_file = '2025_서울시_구별_인구.csv'
crime_file = '서울시_25개구_5대범죄_연도별통합.csv'
pop_df = pd.read_csv(population_file, encoding='cp949')
```

세대나 계 같은 요약정보 행을 제거,
2018~2024년으로 지정해 해당 연도별 인구수를 추출

```
# 2. 인구 데이터 전처리
if pop_df.iloc[0]['동별(2)'] in ['세대 (세대)', '세대', '계 (명)', '계']:
    pop_df = pop_df.iloc[1:].reset_index(drop=True)

years = [2025, 2018, 2019, 2020, 2021, 2022, 2023, 2024]
population_data = []

for idx, row in pop_df.iterrows():
    region = row['동별(2)']

    if pd.isna(region) or region == '' or '총계' in str(region) or '소계' in str(
        continue

    for year in years:
        year_cols = [col for col in pop_df.columns if str(year) in str(col)]

        if len(year_cols) >= 2:
            col_name = year_cols[1]

            if pd.notna(row[col_name]):
                pop_value = row[col_name]
                if isinstance(pop_value, str):
                    pop_value = pop_value.replace(',', '').strip()
                try:
                    pop_value = float(pop_value)
                    population_data.append({
                        'region': region,
                        'year': year,
                        'population': pop_value
                    })
                except:
                    pass

pop_clean = pd.DataFrame(population_data)
print(f"인구 데이터 전처리 완료: {len(pop_clean)}개 레코드")
```

추출한 주민등록 인구와 3-1에서 처리한 지역별 범죄 통계.csv를 사용해서
서울시 지역별 년도별 범죄율 csv 생성

범죄율 = (특정 기간의 범죄 발생 건수 ÷ 인구) × 100,000명

```
# 3. 범죄율 계산
merged_df = pd.merge(crime_df, pop_clean, on=['region', 'year'], how='inner')
merged_df['crime_rate'] = (merged_df['total'] / merged_df['population']) * 100000
merged_df['crime_rate'] = merged_df['crime_rate'].round(2)

print(f" 범죄율 계산 완료: {len(merged_df)}개 레코드")

# 4. 데이터 정렬 및 저장
merged_df_sorted = merged_df.sort_values(['year', 'region']).reset_index(drop=True)

output_file = 'crime_rate_long.csv'
merged_df_sorted[['region', 'year', 'population', 'total', 'crime_rate']].to_csv(
    output_file,
    index=False,
    encoding='utf-8-sig'
)

print(f"₩n CSV 저장 완료: {output_file}")
print("   형식: region, year, population, total, crime_rate")
```

3.3 서울시 구별 cctv 설치 정보

다운받은 xlsx 파일에 있는 불필요한 헤더를 삭제하면 해당 형식으로 수정됨

구분	2015년	2016년	2017년	2018년	2019년	2020년	2021년
종로구	935	1,066	1,225	1,322	1,327	1,510	1,573
중구	363	565	838	1,174	1,242	1,482	1,911

위와 같이 2025년까지 존재

데이터 컬럼을 정리하고 2015년으로 되어 있는 부분의 데이터 타입을 바꾸고

```
import pandas as pd

# ========================================
# 1) XLSX 파일 읽기
# ========================================
file = "서울시 자치구 (범죄예방 수사용) CCTV 설치현황.xlsx"
df = pd.read_excel(file)

# ========================================
# 2) 컬럼 정리: 공백 제거
# ========================================
df.columns = df.columns.astype(str).str.strip()


# ========================================
# 3) 숫자형 데이터 변환 (쉼표 제거)
# ========================================
year_cols = [col for col in df.columns if '년' in col]

for col in year_cols:
    df[col] = df[col].astype(str).str.replace(",", "").astype(int)
```

컬럼명을 구분을 지역으로 변경해서 csv에 저장

```
# ========================================================
# 4) 컬럼명 변환 (구분 → region)
# ========================================================
df = df.rename(columns={'구분': 'region'})

# ========================================================
# 5) wide → long 변환
#    value_name = cumulative_cctv
# ========================================================
df_long = df.melt(
    id_vars=['region'],
    value_vars=year_cols,
    var_name='year',
    value_name='cumulative_cctv'
)

# year에서 "2015년" → 2015 숫자로 변환
df_long['year'] = df_long['year'].str.replace("년", "").str.replace(" ", "")
df_long['year'] = df_long['year'].astype(int)

# ========================================================
# 7) CSV 저장
# ========================================================
output = "서울시_CCTV_설치현황_long_format.csv"
df_long.to_csv(output, index=False, encoding='utf-8-sig')
```

3.4 서울시 구별 비상벨 설치 정보

컬럼명	설명	데이터형식	예시
소재지지번주소	설치 위치 주소	String	서울특별시 서초구 서초동 1549-4 우신1549빌딩
WGS84위도	GPS 위도	float	37.4899
WGS84경도	GPS 경도	float	127.01067
연계방식	연계 유형	String	미연계
경찰연계유무	경찰 연동 여부	Y/N	N
경비업체연계유무	경비업체 연동 여부	Y/N	N
관리사무소연계유무	관리사무소 연동 여부	Y/N	N
부가기능	추가 기능	String	
안전비상벨설치연도	설치 연도	int/String	2019
최종점검일자	마지막 점검일자	Date	2019-07-02
최종점검결과구분	점검 결과 구분	String	Y
관리기관명	관리 기관 이름	String	서초구청
관리기관전화번호	관리 기관 연락처	String	02-2155-6830
데이터기준일자	데이터 기준일	Date	2020-10-30

엑셀 파일을 불러와서
필요한 컬럼 변수명을 지정
서울시 25개의 구 리스트를 작성

```
import pandas as pd
import re

# ======================================================
# 1) 파일 불러오기
# ======================================================
file = "/content/지역별_안전 비상벨 정보.xlsx"
df = pd.read_excel(file)

# 컬럼명 자동 정리(양쪽 공백 제거)
df.columns = df.columns.str.strip()

# 주소 관련 컬럼명 설정
col_road = "소재지도로명주소"
col_addr = "소재지지번주소"
col_manage = "관리기관명"
col_year = "안전비상벨설치연도"

# ======================================================
# 2) 서울 25개 구 리스트
# ======================================================
seoul_gu_list = [
    "종로구","중구","용산구","성동구","광진구","동대문구","중랑구","성북구","강북구","도봉구",
    "노원구","은평구","서대문구","마포구","양천구","강서구","구로구","금천구","영등포구",
    "동작구","관악구","서초구","강남구","송파구","강동구"
]
```

주소에서 구를 추출하는 함수와 최종적으로 분류되는 지역 함수를 작성
지역은 처음엔 소재지도로명주소에서 값이 없을 시 소재지지번주소, 관리기관명 순으로

지역을 추출 후 추출 결과 확인

```
# ====================================================
# 3) 주소에서 구 추출 함수
# ====================================================
def extract_gu(text):
    if pd.isna(text):
        return None
    pattern = r"(종로구|중구|용산구|성동구|광진구|동대문구|중랑구|성북구|강북구|도봉
              r"노원구|은평구|서대문구|마포구|양천구|강서구|구로구|금천구|영등포구|
              r"동작구|관악구|서초구|강남구|송파구|강동구)"
    m = re.search(pattern, str(text))
    return m.group(1) if m else None

# ====================================================
# 4) 최종 지역 추출 함수
# ====================================================
def get_final_region(row):
    r1 = extract_gu(row[col_road])
    if r1 in seoul_gu_list:
        return r1

    r2 = extract_gu(row[col_addr])
    if r2 in seoul_gu_list:
        return r2

    r3 = extract_gu(row[col_manage])
    if r3 in seoul_gu_list:
        return r3

    return "기타"

df["최종지역"] = df.apply(get_final_region, axis=1)

# ====================================================
# 5) 기타 목록 및 지역별 개수 출력
# ====================================================
기타값 = df[df["최종지역"]=="기타"]["최종지역"].unique()
print("----- 기타로 분류된 값 목록 -----")
print(기타값)

print("₩n----- 지역별 개수 -----")
print(df["최종지역"].value_counts())
```

추출된 지역 확인 결과)

```
----- 기타로 분류된 값 목록 -----
[]

----- 지역별 개수 -----
최종지역
강남구      1713
중랑구      1520
성북구      1519
구로구      1442
용산구      1421
영등포구     1307
관악구      1279
마포구      1268
강서구      1156
광진구      1119
도봉구      1071
강동구      1037
동대문구     1013
성동구       900
송파구       881
양천구       787
중구        744
서대문구      723
금천구       709
동작구       627
은평구       611
종로구        99
노원구        64
서초구        52
강북구        22
Name: count, dtype: int64
```

추출한 주소 이상치 검출

```python
# ======================================================
# 6) 주소/관리기관명 전체에서 등장한 구를 저장 (이상치 검출용)
# ======================================================
def find_all_gu(text):
    if pd.isna(text):
        return []
    pattern = r"(종로구|중구|용산구|성동구|광진구|동대문구|중랑구|성북구|강북구|도봉구|" \
              r"노원구|은평구|서대문구|마포구|양천구|강서구|구로구|금천구|영등포구|" \
              r"동작구|관악구|서초구|강남구|송파구|강동구)"
    return re.findall(pattern, str(text))

df["주소내_모든_구"] = df.apply(
    lambda x: find_all_gu(f"{x[col_road]} {x[col_addr]} {x[col_manage]}"),
    axis=1
)

df_problem = df[df.apply(
    lambda x: x["최종지역"] not in x["주소내_모든_구"] and x["최종지역"] != "기타",
    axis=1
)]

print("₩n 주소에서 추출한 구가 실제 문자열과 불일치한 행 수:", len(df_problem))
print(df_problem[[col_road, col_addr, col_manage, "주소내_모든_구", "최종지역"]].head(20))
```

지역별 년도별 비상벨 설치 대수 집계

설치 연도에 2025를 넘는 이상치가 있어 이상치를 확인하고
2025가 넘는 것들은 2025에 포함해서 설치 대수 집계

```
# ======================================================
# 7) 설치대수 계산 및 2025년 미후 합산
# ======================================================
df_clean = df.dropna(subset=[col_year]).copy()
df_clean["설치대수"] = 1

# 2025년 초과 연도 데이터 추출
over_2025 = df[df[col_year] > 2025]

# 설치 연도와 최종지역 확인
over_2025_info = over_2025[[col_year, "최종지역", col_road, col_addr, col_manage]]

print("----- 2025년 초과 설치 연도 데이터 -----")
print(over_2025_info)

print("₩n 2025년 초과 연도 존재하는 지역 목록:")
print(over_2025_info["최종지역"].value_counts())


# 2025년 초과 연도는 2025로 합치기
df_clean[col_year] = df_clean[col_year].apply(lambda x: 2025 if x > 2025 else x)

# 연도별 + 지역별 집계
year_region = df_clean.groupby([col_year, "최종지역"])["설치대수"].sum().reset_index()
year_region = year_region.sort_values(by=[col_year, "최종지역"])

# 누적 설치대수 계산
year_region["누적설치대수"] = year_region.groupby("최종지역")["설치대수"].cumsum()

# ======================================================
# 9) CSV 생성
# ======================================================
year_region.to_csv("비상벨_년도별_지역별_누적설치대수_2025합산.csv", index=False)
print("₩n CSV 생성 완료: 비상벨_년도별_지역별_누적설치대수_2025합산.csv")
```

년도 이상치 결과)

```
----- 2025년 초과 설치 연도 데이터 -----
      안전비상벨설치연도  최종지역              소재지도로명주소              소재지지번주소  #
16492       2031  영등포구     서울특별시 영등포구 대림동 742     서울특별시 영등포구 대림동 742
16493       2032  영등포구     서울특별시 영등포구 대림동 690     서울특별시 영등포구 대림동 690
16494       2033  영등포구     서울특별시 영등포구 대림동 715     서울특별시 영등포구 대림동 715
16495       2034  영등포구    서울특별시 영등포구 도림동 23-1    서울특별시 영등포구 도림동 23-1
```

```
⚠ 2025년 초과 연도 존재하는 지역 목록:
최종지역
영등포구    28
Name: count, dtype: int64
```

3.5 서울시 구별 유흥주점 등록 정보

컬럼은 소재지 전체수, 도로명주소, 인허가일자 등 총 47개 컬럼
데이터를 일어와서 서울시 25개구 리스트 자성과
주소를 추출하는 함수 작성

```
import pandas as pd
import re

# ====================================================
# 1) 파일 업로드 및 불러오기
# ====================================================
file = "/content/지역별_유흥주점 정보.xlsx"

df = pd.read_excel(file)

# 공백 제거
df.columns = df.columns.str.strip()

print("불러온 컬럼:", df.columns.tolist())


# ====================================================
# 2) 주소에서 서울시 25개구 추출하는 함수
# ====================================================
seoul_gu_list = [
    "종로구", "중구", "용산구", "성동구", "광진구", "동대문구", "중랑구",
    "성북구", "강북구", "도봉구", "노원구", "은평구", "서대문구", "마포구",
    "양천구", "강서구", "구로구", "금천구", "영등포구", "동작구", "관악구",
    "서초구", "강남구", "송파구", "강동구"
]

gu_pattern = "(" + "|".join(seoul_gu_list) + ")"

def extract_gu(addr):
    if pd.isna(addr):
        return None
    match = re.search(gu_pattern, addr)
    return match.group(1) if match else None
```

소재지전체주소를 우선으로 없으면 도로명전체주소로 서울시 구 추출

```
# ====================================================
# 3) 주소 컬럼 처리 ('소재지전체주소' → 없으면 '도로명전체주소')
# ====================================================
addr_col = None
if "소재지전체주소" in df.columns:
    addr_col = "소재지전체주소"
elif "도로명전체주소" in df.columns:
    addr_col = "도로명전체주소"
else:
    raise ValueError("주소 컬럼이 없습니다. '소재지전체주소' 또는 '도로명전체주소' 필요")

df['구'] = df[addr_col].apply(extract_gu)

# 서울 구만 필터링
df = df[df['구'].notna()]
print("서울 지역 데이터 개수:", len(df))
```

인허가일자에서 연도추출하고

연도별 구별 등록된 유흥주점수를 집계후
이전 년도 유흥 주점수와 이번년도 유흥주점수를 합산 후 csv로 저장

```
# ====================================================
# 4) 인허가일자에서 연도 추출
# ====================================================
if '인허가일자' not in df.columns:
    raise ValueError("'인허가일자' 컬럼이 없습니다.")

# 문자열 날짜 또는 숫자 날짜 모두 처리
df['인허가일자'] = pd.to_datetime(df['인허가일자'], errors='coerce')

df['연도'] = df['인허가일자'].dt.year
df = df[df['연도'].notna()]  # 연도 없는 값 제거
df['연도'] = df['연도'].astype(int)


# ====================================================
# 5) 연도별 + 구별 등록된 유흥주점 수
# ====================================================
year_gu_count = df.groupby(['연도', '구']).size().reset_index(name="당해연도_등록수")


# ====================================================
# 6) 누적 합 계산 (총 유흥주점 수 = 저번년도 + 이번년도)
# ====================================================
year_gu_count = year_gu_count.sort_values(['구', '연도'])
year_gu_count['누적_유흥주점수'] = year_gu_count.groupby('구')['당해연도_등록수'].cumsum()


# ====================================================
# 7) CSV로 저장
# ====================================================
output_file = "/content/서울시_연도별_구별_유흥주점_누적집계.csv"
year_gu_count.to_csv(output_file, index=False, encoding='utf-8-sig')
```

4. 시각화 코드

4.1 서울시 년도별, 지역별 범죄율 시각화

3.2에서 코드에서 이어짐 서울시 범죄율 데이터를 사용

서울시 GeoJSON을 불러오고 각 구의 중심점을 구해 지도위에 구이름을 텍스트로 표기 설정

```
# 4. 서울시 GeoJSON 다운로드
geojson_url = 'https://raw.githubusercontent.com/southkorea/seoul-maps/master/kostat/2013/json/seoul_municipalities_geo_simple.json

try:
    response = requests.get(geojson_url)
    seoul_geo = response.json()

    # GeoJSON에서 구별 중심 좌표 추출
    centroids = {}
    for feature in seoul_geo['features']:
        name = feature['properties']['name']
        coords = feature['geometry']['coordinates'][0]

        if feature['geometry']['type'] == 'MultiPolygon':
            coords = feature['geometry']['coordinates'][0][0]

        lats = [coord[1] for coord in coords]
        lons = [coord[0] for coord in coords]
        centroids[name] = {
            'lat': sum(lats) / len(lats),
            'lon': sum(lons) / len(lons)
        }

    merged_df['lat'] = merged_df['region'].map(lambda x: centroids.get(x, {}).get('lat'))
    merged_df['lon'] = merged_df['region'].map(lambda x: centroids.get(x, {}).get('lon'))

except Exception as e:
    print(f" 오류: GeoJSON 로드 실패 - {e}")
    seoul_geo = None
```

Plotly Choropleth 지도 생성

범죄율 분포의 25%~75% 분위수(q25~q75) 를 색상 범위로 설정

슬라이더를 이용해 2018~2024년 범죄율 변화를 애니메이션처럼 확인하도록 구성

각 구의 이름(label)과 범죄율 지도(Choropleth)를 함께 표시

지도는 확대/축소(zoom), 드래그 이동이 가능하도록 설정

```
# 5. Plotly Choropleth 지도 생성
if seoul_geo:
    # 색상 범위를 분위수 기준으로 조정
    all_rates = merged_df['crime_rate'].values
    q25 = np.percentile(all_rates, 25)
    q75 = np.percentile(all_rates, 75)

    fig = go.Figure()

    # 각 연도별 프레임 생성
    for year in sorted(merged_df['year'].unique()):
        year_data = merged_df[merged_df['year'] == year].copy()

        # Choropleth 레이어
        fig.add_trace(go.Choroplethmapbox(
            geojson=seoul_geo,
            locations=year_data['region'],
            z=year_data['crime_rate'],
            featureidkey="properties.name",
            colorscale='YlOrRd',
            zmin=q25,
            zmax=q75,
            marker_opacity=0.7,
            marker_line_width=1,
            marker_line_color='white',
            text=year_data['text'],
            hovertemplate='%{text}<extra></extra>',
            colorbar=dict(
                title="범죄율<br>(10만명당)",
                thickness=15,
                len=0.7,
                x=1.02
            ),
            name=str(year),
            visible=(year == 2018)
        ))
```

```
        # 각 구에 텍스트 라벨 추가 (지역명만 표시)
        fig.add_trace(go.Scattermapbox(
            lat=year_data['lat'],
            lon=year_data['lon'],
            mode='text',
            text=year_data['region'],
            textfont=dict(
                size=11,
                color='black',
                family='Arial Black'
            ),
            hoverinfo='skip',
            name=f'{year}_labels',
            visible=(year == 2018)
        ))

    # 슬라이더 생성
    steps = []
    for i, year in enumerate(sorted(merged_df['year'].unique())):
        step = dict(
            method="update",
            args=[
                {"visible": [False] * len(fig.data)},
                {"title": f"서울시 구별 범죄율 ({year}년)"}
            ],
            label=str(year)
        )
        step["args"][0]["visible"][i*2] = True
        step["args"][0]["visible"][i*2 + 1] = True
        steps.append(step)

    sliders = [dict(
        active=0,
        yanchor="top",
        y=0.02,
        xanchor="left",
        x=0.05,
        currentvalue=dict(
            prefix="년도: ",
            visible=True,
            xanchor="left",
            font=dict(size=16, color='#333')
        ),
        pad=dict(b=10, t=10),
        len=0.9,
        steps=steps
    )]

      # 레이아웃 설정
      fig.update_layout(
          mapbox=dict(
              style="open-street-map",
              center=dict(lat=37.5665, lon=126.9780),
              zoom=10
          ),
          sliders=sliders,
          title=dict(
              text='서울시 구별 범죄율 (2018년)',
              font=dict(size=24, color='#333'),
              x=0.5,
              xanchor='center'
          ),
          height=700,
          margin=dict(l=0, r=0, t=50, b=0),
          showlegend=False,
          # 확대/축소 및 이동 가능하게 설정
          dragmode='zoom'
      )

      # 확대/축소 버튼 설정
      fig.update_layout(
          mapbox=dict(
              style="open-street-map",
              center=dict(lat=37.5665, lon=126.9780),
              zoom=10
          ),
          # 줌 컨트롤 활성화
          updatemenus=[],
      )

      fig.show()

else:
    print(" 오류: GeoJSON을 로드할 수 없습니다.")
```

4.2 cctv와 범죄율 연관 시각화

데이터를 불러와서 지역명을 정규화 한다음

필요한 데이터 추출

```
# 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)
crime_file = '/content/crime_rate_long.csv'
cctv_file = '/content/서울시_CCTV_설치현황_long_format.csv'

df_crime = pd.read_csv(crime_file)
df_cctv = pd.read_csv(cctv_file)

# 지역명 정규화
def normalize_region(text):
    if pd.isna(text):
        return None
    text = str(text).strip()
    text = text.replace('구청', '').replace('서울특별시', '').replace('서울시', '').strip()
    if not text.endswith('구'):
        text = text + '구'
    return text

# 데이터 전처리
df_crime['지역'] = df_crime['region'].apply(normalize_region)
df_crime_clean = df_crime[['year', '지역', 'population', 'total', 'crime_rate']].copy()
df_crime_clean.columns = ['연도', '지역', '인구수', '범죄건수', '범죄율']

df_cctv['지역'] = df_cctv['region'].apply(normalize_region)
df_cctv_clean = df_cctv[['year', '지역', 'cumulative_cctv']].copy()
df_cctv_clean.columns = ['연도', '지역', 'CCTV수']
```

년도 지역 기준으로 병합

상관계수 계산

```
# 데이터 병합
df_merged = df_crime_clean.merge(df_cctv_clean, on=['연도', '지역'], how='inner')
df_merged = df_merged.dropna()

# 상관계수 계산
corr_pearson = df_merged['범죄율'].corr(df_merged['CCTV수'])
corr_spearman = stats.spearmanr(df_merged['범죄율'], df_merged['CCTV수'])[0]

print(f"Pearson 상관계수: {corr_pearson:.4f}")
print(f"Spearman 상관계수: {corr_spearman:.4f}")
print(f"분석 데이터: {len(df_merged)}개")
```

4.2.1 산점도 + 회귀선(상관계수) 시각화

```
# 시각화 (2x2 레이아웃)
fig = plt.figure(figsize=(18, 12))

# 1. 산점도 + 회귀선 (상관계수)
ax1 = plt.subplot(2, 2, 1)
ax1.scatter(df_merged['CCTV수'], df_merged['범죄율'],
           alpha=0.5, s=60, c='crimson', edgecolors='darkred', linewidth=0.5)
z = np.polyfit(df_merged['CCTV수'], df_merged['범죄율'], 1)
p = np.poly1d(z)
x_line = np.linspace(df_merged['CCTV수'].min(), df_merged['CCTV수'].max(), 100)
ax1.plot(x_line, p(x_line), "blue", linestyle='--', linewidth=3, label='회귀선')
ax1.set_xlabel('CCTV 설치 대수', fontsize=13, weight='bold')
ax1.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold')
ax1.set_title(f'CCTV와 범죄율 상관관계₩nPearson r = {corr_pearson:.4f} | Spearman ρ = {corr_spearman:.4f}',
             fontsize=14, weight='bold', pad=15)
ax1.grid(True, alpha=0.3)
ax1.legend(fontsize=11)
```

4.2.2 연도별 평균 추세

```
# 2. 연도별 평균 추세 (이중 y축)
ax2 = plt.subplot(2, 2, 2)
yearly = df_merged.groupby('연도')[['범죄율', 'CCTV수']].mean()
ax2_twin = ax2.twinx()

line1 = ax2.plot(yearly.index, yearly['범죄율'],
                marker='o', linewidth=3.5, markersize=10,
                color='crimson', label='평균 범죄율', markeredgecolor='darkred', markeredgewidth=1.5)
line2 = ax2_twin.plot(yearly.index, yearly['CCTV수'],
                      marker='s', linewidth=3.5, markersize=10,
                      color='steelblue', label='평균 CCTV수', markeredgecolor='darkblue', markeredgewidth=1.5)

ax2.set_xlabel('연도', fontsize=13, weight='bold')
ax2.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax2_twin.set_ylabel('CCTV 설치 대수', fontsize=13, weight='bold', color='steelblue')
ax2.set_title('연도별 평균 추세', fontsize=14, weight='bold', pad=15)
ax2.grid(True, alpha=0.3)
ax2.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax2_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)

lines = line1 + line2
labels = [l.get_label() for l in lines]
ax2.legend(lines, labels, loc='upper left', fontsize=11, framealpha=0.9)
```

4.2.3 상관계수 히트맵

```
# 3. 상관계수 히트맵
ax3 = plt.subplot(2, 2, 3)
corr_matrix = df_merged[['범죄율', 'CCTV수', '인구수']].corr()
sns.heatmap(corr_matrix, annot=True, fmt='.3f', cmap='RdBu_r',
            center=0, square=True, linewidths=3, cbar_kws={"shrink": 0.8},
            vmin=-1, vmax=1, annot_kws={'size': 16, 'weight': 'bold'}, ax=ax3)
ax3.set_title('상관계수 히트맵', fontsize=14, weight='bold', pad=15)
ax3.tick_params(labelsize=12)
```

4.2.4 2024년 범죄율 상위 15개 지역

```
# 4. 2024년 범죄율 상위 15개 지역
ax4 = plt.subplot(2, 2, 4)
latest_year = df_merged['연도'].max()
df_latest = df_merged[df_merged['연도'] == latest_year].copy()
df_top = df_latest.nlargest(15, '범죄율')

x = np.arange(len(df_top))
width = 0.35
ax4_twin = ax4.twinx()

bars1 = ax4.bar(x - width/2, df_top['범죄율'], width,
                label='범죄율', color='crimson', alpha=0.85, edgecolor='darkred', linewidth=1.5)
bars2 = ax4_twin.bar(x + width/2, df_top['CCTV수'], width,
                     label='CCTV수', color='steelblue', alpha=0.85, edgecolor='darkblue', linewidth=1.5)

ax4.set_xlabel('지역', fontsize=13, weight='bold')
ax4.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax4_twin.set_ylabel('CCTV 설치 대수', fontsize=13, weight='bold', color='steelblue')
ax4.set_title(f'{latest_year}년 범죄율 상위 15개 지역', fontsize=14, weight='bold', pad=15)
ax4.set_xticks(x)
ax4.set_xticklabels(df_top['지역'], rotation=45, ha='right', fontsize=10)
ax4.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax4_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)
ax4.legend(loc='upper left', fontsize=10, framealpha=0.9)
ax4_twin.legend(loc='upper right', fontsize=10, framealpha=0.9)
ax4.grid(True, alpha=0.3, axis='y')

plt.suptitle('CCTV 설치 대수와 범죄율 상관관계 분석',
             fontsize=18, weight='bold', y=0.995)
plt.tight_layout()
plt.show()
```

4.3 비상벨과 범죄율 연관 시각화

데이터 준비, 전처리 및 병합

```
# 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)
crime_file = '/content/crime_rate_long.csv'
bell_file = '/content/비상벨_년도별_지역별_누적설치대수_2025합산.csv'

df_crime = pd.read_csv(crime_file)
df_bell = pd.read_csv(bell_file)

# 지역명 정규화
def normalize_region(text):
    if pd.isna(text):
        return None
    text = str(text).strip()
    text = text.replace('구청', '').replace('서울특별시', '').replace('서울시', '').strip()
    if not text.endswith('구'):
        text = text + '구'
    return text

# 데이터 전처리
df_crime['지역'] = df_crime['region'].apply(normalize_region)
df_crime_clean = df_crime[['year', '지역', 'population', 'total', 'crime_rate']].copy()
df_crime_clean.columns = ['연도', '지역', '인구수', '범죄건수', '범죄율']

df_bell['지역'] = df_bell['최종지역'].apply(normalize_region)
df_bell_clean = df_bell[['안전비상벨설치연도', '지역', '누적설치대수']].copy()
df_bell_clean.columns = ['연도', '지역', '비상벨수']

# 데이터 병합
df_merged = df_crime_clean.merge(df_bell_clean, on=['연도', '지역'], how='inner')
df_merged = df_merged.dropna()
```

상관계수 계산후 시각화

4.3.1 산점도 + 회귀선 (상관계수)

```
# 상관계수 계산
corr_pearson = df_merged['범죄율'].corr(df_merged['비상벨수'])
corr_spearman = stats.spearmanr(df_merged['범죄율'], df_merged['비상벨수'])[0]

print(f"Pearson 상관계수: {corr_pearson:.4f}")
print(f"Spearman 상관계수: {corr_spearman:.4f}")
print(f"분석 데이터: {len(df_merged)}개")

# 시각화 (2x2 레이아웃)
fig = plt.figure(figsize=(18, 12))

# 1. 산점도 + 회귀선 (상관계수)
ax1 = plt.subplot(2, 2, 1)
ax1.scatter(df_merged['비상벨수'], df_merged['범죄율'],
           alpha=0.5, s=60, c='crimson', edgecolors='darkred', linewidth=0.5)
z = np.polyfit(df_merged['비상벨수'], df_merged['범죄율'], 1)
p = np.poly1d(z)
x_line = np.linspace(df_merged['비상벨수'].min(), df_merged['비상벨수'].max(), 100)
ax1.plot(x_line, p(x_line), "blue", linestyle='--', linewidth=3, label='회귀선')
ax1.set_xlabel('비상벨 설치 대수', fontsize=13, weight='bold')
ax1.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold')
ax1.set_title(f'비상벨과 범죄율 상관관계₩nPearson r = {corr_pearson:.4f} | Spearman ρ = {corr_spearman:.4f}',
             fontsize=14, weight='bold', pad=15)
ax1.grid(True, alpha=0.3)
ax1.legend(fontsize=11)
```

4.3.2 연도별 평균 추세 그래프

```
# 2. 연도별 평균 추세 (이중 y축)
ax2 = plt.subplot(2, 2, 2)
yearly = df_merged.groupby('연도')[['범죄율', '비상벨수']].mean()
ax2_twin = ax2.twinx()

line1 = ax2.plot(yearly.index, yearly['범죄율'],
                marker='o', linewidth=3.5, markersize=10,
                color='crimson', label='평균 범죄율', markeredgecolor='darkred', markeredgewidth=1.5)
line2 = ax2_twin.plot(yearly.index, yearly['비상벨수'],
                      marker='s', linewidth=3.5, markersize=10,
                      color='steelblue', label='평균 비상벨수', markeredgecolor='darkblue', markeredgewidth=1.5)

ax2.set_xlabel('연도', fontsize=13, weight='bold')
ax2.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax2_twin.set_ylabel('비상벨 설치 대수', fontsize=13, weight='bold', color='steelblue')
ax2.set_title('연도별 평균 추세', fontsize=14, weight='bold', pad=15)
ax2.grid(True, alpha=0.3)
ax2.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax2_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)

lines = line1 + line2
labels = [l.get_label() for l in lines]
ax2.legend(lines, labels, loc='upper left', fontsize=11, framealpha=0.9)
```

4.3.3 상관계수 히트맵

```
# 3. 상관계수 히트맵
ax3 = plt.subplot(2, 2, 3)
corr_matrix = df_merged[['범죄율', '비상벨수', '인구수']].corr()
sns.heatmap(corr_matrix, annot=True, fmt='.3f', cmap='RdBu_r',
            center=0, square=True, linewidths=3, cbar_kws={"shrink": 0.8},
            vmin=-1, vmax=1, annot_kws={'size': 16, 'weight': 'bold'}, ax=ax3)
ax3.set_title('상관계수 히트맵', fontsize=14, weight='bold', pad=15)
ax3.tick_params(labelsize=12)
```

4.3.4 2025년 범죄율 상위 15개 지역

```
# 4. 2024년 범죄율 상위 15개 지역
ax4 = plt.subplot(2, 2, 4)
latest_year = df_merged['연도'].max()
df_latest = df_merged[df_merged['연도'] == latest_year].copy()
df_top = df_latest.nlargest(15, '범죄율')

x = np.arange(len(df_top))
width = 0.35
ax4_twin = ax4.twinx()

bars1 = ax4.bar(x - width/2, df_top['범죄율'], width,
                label='범죄율', color='crimson', alpha=0.85, edgecolor='darkred', linewidth=1.5)
bars2 = ax4_twin.bar(x + width/2, df_top['비상벨수'], width,
                     label='비상벨수', color='steelblue', alpha=0.85, edgecolor='darkblue', linewidth=1.5)

ax4.set_xlabel('지역', fontsize=13, weight='bold')
ax4.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax4_twin.set_ylabel('비상벨 설치 대수', fontsize=13, weight='bold', color='steelblue')
ax4.set_title(f'{latest_year}년 범죄율 상위 15개 지역', fontsize=14, weight='bold', pad=15)
ax4.set_xticks(x)
ax4.set_xticklabels(df_top['지역'], rotation=45, ha='right', fontsize=10)
ax4.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax4_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)
ax4.legend(loc='upper left', fontsize=10, framealpha=0.9)
ax4_twin.legend(loc='upper right', fontsize=10, framealpha=0.9)
ax4.grid(True, alpha=0.3, axis='y')

plt.suptitle('비상벨 설치 대수와 범죄율 상관관계 분석',
             fontsize=18, weight='bold', y=0.995)
plt.tight_layout()
plt.show()
```

4.4 유흥주점 수와 범죄율 연관 시각화

데이터 전처리, 병합

```
# 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)
crime_file = '/content/crime_rate_long.csv'
bar_file = '/content/서울시_연도별_구별_유흥주점_누적집계.csv'

df_crime = pd.read_csv(crime_file)
df_bar = pd.read_csv(bar_file)

# 지역명 정규화
def normalize_region(text):
    if pd.isna(text):
        return None
    text = str(text).strip()
    text = text.replace('구청', '').replace('서울특별시', '').replace('서울시', '').strip()
    if not text.endswith('구'):
        text = text + '구'
    return text

# 데이터 전처리
df_crime['지역'] = df_crime['region'].apply(normalize_region)
df_crime_clean = df_crime[['year', '지역', 'population', 'total', 'crime_rate']].copy()
df_crime_clean.columns = ['연도', '지역', '인구수', '범죄건수', '범죄율']

df_bar['지역'] = df_bar['구'].apply(normalize_region)
df_bar_clean = df_bar[['연도', '지역', '누적_유흥주점수']].copy()
df_bar_clean.columns = ['연도', '지역', '유흥주점수']

# 데이터 병합
df_merged = df_crime_clean.merge(df_bar_clean, on=['연도', '지역'], how='inner')
df_merged = df_merged.dropna()
```

상관계산 후 시각화

4.4.1 산점도 + 회귀선 (상관계수)

```
# 상관계수 계산
corr_pearson = df_merged['범죄율'].corr(df_merged['유흥주점수'])
corr_spearman = stats.spearmanr(df_merged['범죄율'], df_merged['유흥주점수'])[0]

print(f"Pearson 상관계수: {corr_pearson:.4f}")
print(f"Spearman 상관계수: {corr_spearman:.4f}")
print(f"분석 데이터: {len(df_merged)}개")

# 시각화 (2x2 레이아웃)
fig = plt.figure(figsize=(18, 12))

# 1. 산점도 + 회귀선 (상관계수)
ax1 = plt.subplot(2, 2, 1)
ax1.scatter(df_merged['유흥주점수'], df_merged['범죄율'],
           alpha=0.5, s=60, c='crimson', edgecolors='darkred', linewidth=0.5)
z = np.polyfit(df_merged['유흥주점수'], df_merged['범죄율'], 1)
p = np.poly1d(z)
x_line = np.linspace(df_merged['유흥주점수'].min(), df_merged['유흥주점수'].max(), 100)
ax1.plot(x_line, p(x_line), "blue", linestyle='--', linewidth=3, label='회귀선')
ax1.set_xlabel('유흥주점 수', fontsize=13, weight='bold')
ax1.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold')
ax1.set_title(f'유흥주점과 범죄율 상관관계₩nPearson r = {corr_pearson:.4f} | Spearman ρ = {corr_spearman:.4f}',
             fontsize=14, weight='bold', pad=15)
ax1.grid(True, alpha=0.3)
ax1.legend(fontsize=11)
```

4.4.2 연도별 평균 추세 그래프

```
# 2. 연도별 평균 추세 (이중 y축)
ax2 = plt.subplot(2, 2, 2)
yearly = df_merged.groupby('연도')[['범죄율', '유흥주점수']].mean()
ax2_twin = ax2.twinx()

line1 = ax2.plot(yearly.index, yearly['범죄율'],
                 marker='o', linewidth=3.5, markersize=10,
                 color='crimson', label='평균 범죄율', markeredgecolor='darkred', markeredgewidth=1.5)
line2 = ax2_twin.plot(yearly.index, yearly['유흥주점수'],
                      marker='s', linewidth=3.5, markersize=10,
                      color='steelblue', label='평균 유흥주점수', markeredgecolor='darkblue', markeredgewidth=1.5)

ax2.set_xlabel('연도', fontsize=13, weight='bold')
ax2.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax2_twin.set_ylabel('유흥주점 수', fontsize=13, weight='bold', color='steelblue')
ax2.set_title('연도별 평균 추세', fontsize=14, weight='bold', pad=15)
ax2.grid(True, alpha=0.3)
ax2.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax2_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)

lines = line1 + line2
labels = [l.get_label() for l in lines]
ax2.legend(lines, labels, loc='upper left', fontsize=11, framealpha=0.9)
```

4.4.3 상관계수 히트맵

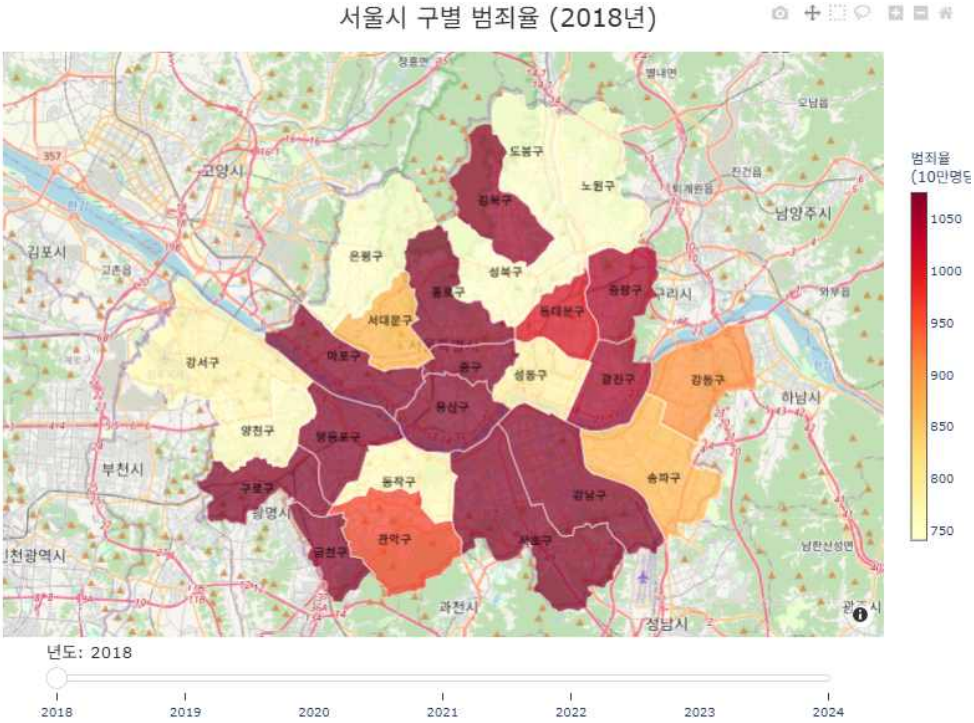
```
# 3. 상관계수 히트맵
ax3 = plt.subplot(2, 2, 3)
corr_matrix = df_merged[['범죄율', '유흥주점수', '인구수']].corr()
sns.heatmap(corr_matrix, annot=True, fmt='.3f', cmap='RdBu_r',
            center=0, square=True, linewidths=3, cbar_kws={"shrink": 0.8},
            vmin=-1, vmax=1, annot_kws={'size': 16, 'weight': 'bold'}, ax=ax3)
ax3.set_title('상관계수 히트맵', fontsize=14, weight='bold', pad=15)
ax3.tick_params(labelsize=12)
```

4.4.4 2024년 범죄율 상위 15개 지역

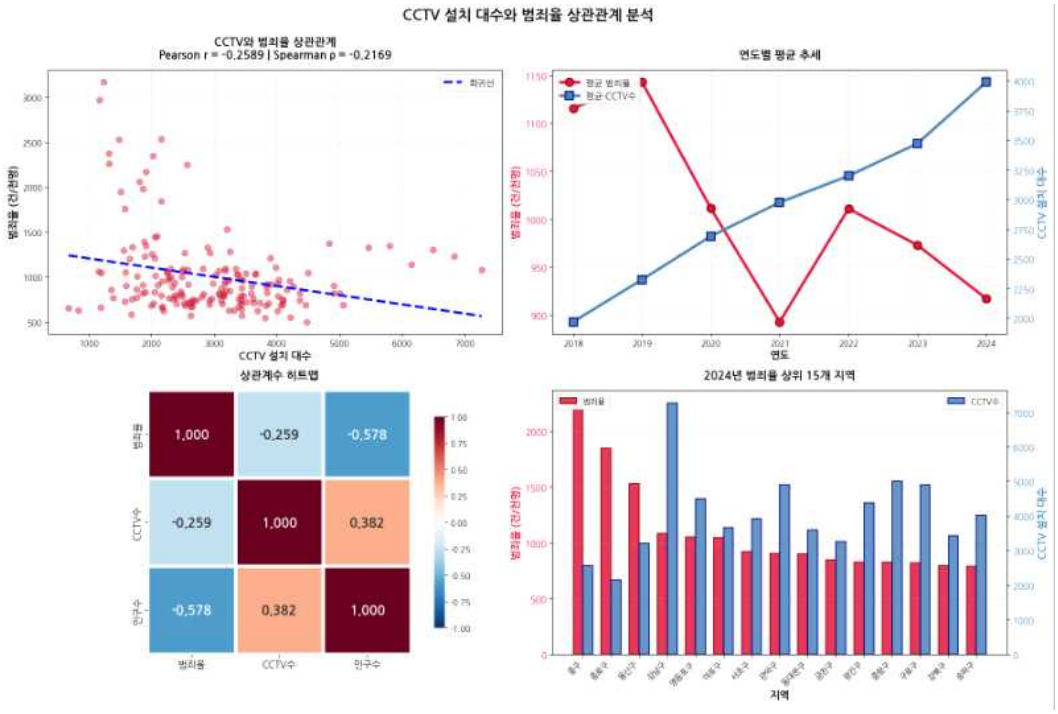
```
# 4. 2024년 범죄율 상위 15개 지역
ax4 = plt.subplot(2, 2, 4)
latest_year = df_merged['연도'].max()
df_latest = df_merged[df_merged['연도'] == latest_year].copy()
df_top = df_latest.nlargest(15, '범죄율')

x = np.arange(len(df_top))
width = 0.35
ax4_twin = ax4.twinx()

bars1 = ax4.bar(x - width/2, df_top['범죄율'], width,
                label='범죄율', color='crimson', alpha=0.85, edgecolor='darkred', linewidth=1.5)
bars2 = ax4_twin.bar(x + width/2, df_top['유흥주점수'], width,
                     label='유흥주점수', color='steelblue', alpha=0.85, edgecolor='darkblue', linewidth=1.5)

ax4.set_xlabel('지역', fontsize=13, weight='bold')
ax4.set_ylabel('범죄율 (건/천명)', fontsize=13, weight='bold', color='crimson')
ax4_twin.set_ylabel('유흥주점 수', fontsize=13, weight='bold', color='steelblue')
ax4.set_title(f'{latest_year}년 범죄율 상위 15개 지역', fontsize=14, weight='bold', pad=15)
ax4.set_xticks(x)
ax4.set_xticklabels(df_top['지역'], rotation=45, ha='right', fontsize=10)
ax4.tick_params(axis='y', labelcolor='crimson', labelsize=11)
ax4_twin.tick_params(axis='y', labelcolor='steelblue', labelsize=11)
ax4.legend(loc='upper left', fontsize=10, framealpha=0.9)
ax4_twin.legend(loc='upper right', fontsize=10, framealpha=0.9)
ax4.grid(True, alpha=0.3, axis='y')

plt.suptitle('유흥주점 수와 범죄율 상관관계 분석',
             fontsize=18, weight='bold', y=0.995)
plt.tight_layout()
plt.show()
```

5. 시각화 결과

5.1 서울시 구별 범죄율 시각화 결과



5.2 cctv 설치 대수와 범죄율 상관관계 시각화



분석

cctv 설치 수에 따라 범죄율이 감소하는 경향이 있지만

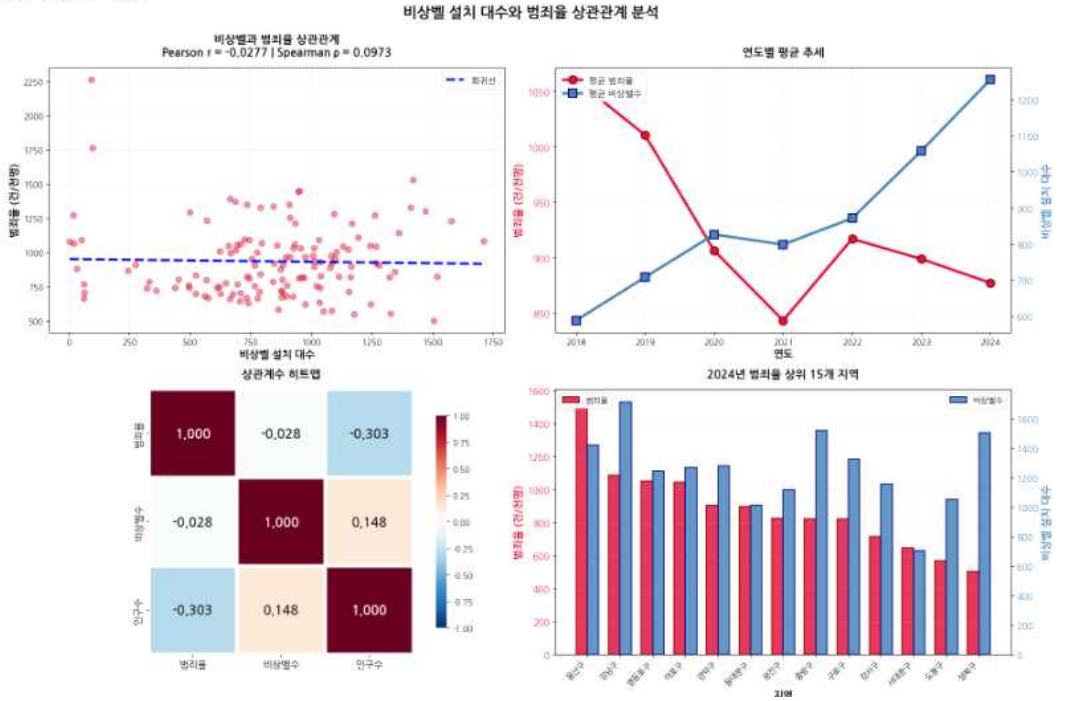
뚜렷한 직접적인 영향을 나타내기는 힘들다

cctv는 매년 증가함

2021년에 범죄율이 급격히 꺾어짐 코로나의 영향이 큰 것 예상됨

2024년 범죄율 상위 15개 지역을 보면 cctv가 많을수록 범죄율이 낮은 경향이 있으나 영등포구에서 이상치가 발생함 데이터 전처리 할 때 설치 년월이 2025이 넘는 데이터를 15개 지역에 포함시켜서 값이 이상하게 된 것으로 예상

5.3 비상벨 설치 대수와 범죄율 상관관계 시각화



분석

Pearson 상관계수: -0.0277

Spearman 상관계수: 0.0973

비상벨은 범죄율에 영향을 거의 주지 않음

5.4 유흥 주점과 범죄율 상관관계 시각화

분석
Pearson 상관계수: 0.6903
선형적 상관관계가 강하게 존재함을 의미함.
r 값이 0.69면 사회과학 분야에서는 “강한 상관관계(strong correlation)”로 분류하기도 함.
즉, 유흥주점 수가 증가할수록 범죄율도 증가하는 경향이 뚜렷함.

Spearman 상관계수: 0.5971
순위 기반 상관: 비선형 관계도 고려
이 값 역시 꽤 높은 수준의 정(+) 상관관계를 의미
즉, 단순한 직선 관계뿐 아니라 전반적인 증가 경향이 일관적임을 의미

유흥 주점 수가 증가할수록 범죄율도 증가하는 경향이 뚜렷함

결론 cctv와 비상벨은 범죄율과 상관과계가 별로 없고
유흥 주점수가 범죄율에 유의미한 영향을 준다.