환경요인에 따른 지역별 범죄율 분석
2025

작성자: 202144013 기경현

인하공전 컴퓨터 정보공학과
빅데이터 수업 A반
민정혜 교수님

목차

1. 주제

1.1 주제

본 프로젝트는 범죄수와 인구수 데이터를 활용하여 지역별·연도별 범죄율을 산출하고, CCTV 설치 수, 비상벨 설치 수, 유흥지점 수, 월평균 소득과 같은 치안 및 사회·경제적 요인이 범죄율에 미치는 영향을 분석하는 것을 주제로 한다. 단순 범죄 발생 건수 비교에서 벗어나 인구 규모를 반영한 범죄율을 기준으로 분석함으로써, 지역 간 범죄 수준을 보다 객관적으로 비교하고 범죄 발생의 특성을 파악하고자 한다. 또한 다양한 공공데이터를 결합한 빅데이터 분석과 시각화를 통해 범죄와 지역 특성 간의 관계를 직관적으로 이해하는 것을 목표로 한다.

1.2 목적

본 연구의 목적은 인구수 대비 범죄율을 산출하여 지역 및 연도에 따른 범죄 발생 양상을 분석하고, 치안 인프라(CCTV, 비상벨)와 환경적 요인(유흥지점 수), 사회·경제적 요인(월평균 소득)이 범죄율과 어떠한 관계를 가지는지를 데이터 기반으로 확인하는 데 있다. 이를 통해 범죄 예방을 목적으로 설치되는 치안 시설이 실제로 범죄 억제에 기여하는지 살펴보고, 동시에 범죄 발생에 영향을 줄 수 있는 환경적·경제적 요인의 중요성을 함께 분석하고자 한다. 나아가 빅데이터 분석 및 시각화 과정을 통해 데이터 분석 역량을 강화하고, 실제 사회 문제를 데이터로 해석하는 경험을 쌓는 것을 부가적인 목적으로 한다.

1.3 예상 효과

본 프로젝트를 통해 지역별·연도별 범죄율의 차이를 명확히 파악할 수 있으며, 각 요인이 범죄율과 어떤 방향의 관계를 가지는지에 대한 기초적인 이해를 얻을 수 있을 것으로 기대된다. 또한 범죄 관련 데이터를 시각화함으로써 복잡한 수치 데이터를 보다 쉽게 해석할 수 있고, 분석 결과를 효과적으로 전달할 수 있다. 이러한 분석 결과는 범죄 예방 정책 수립이나 치안 인프라 배치와 같은 실무적 의사결정에 참고 자료로 활용될 수 있으며, 데이터 기반 의사결정의 중요성을 인식하는 데 기여할 것으로 예상된다.

1.4 예상 결과

CCTV와 비상벨과 같은 치안 인프라는 범죄율과 음의 상관관계를 보일 가능성이 높으며, 이는 치안 시설이 범죄 예방에 일정 부분 기여하고 있음을 시사할 것으로 예상된다.
반면, 유흥지점 수는 범죄율과 양의 상관관계를 보일 가능성이 있으며, 유흥시설이 밀집된 지역일수록 범죄 발생 위험이 높아질 수 있음을 확인할 수 있을 것으로 예상된다. 또한 월평균 소득과 범죄율 간의 관계를 통해 경제적 요인이 범죄 발생에 미치는 영향에 대한 기초적인 분석 결과를 도출할 수 있을 것으로 기대된다.

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

2.6 서울시 년도별 지역별 월평균 소득

https://www.data.go.kr/
국민 연금 공단에서 공개한 지역 가입자와 사업장 가입자 모두의 평균
지역가입자: 직장인이 아닌 개인사업자, 프리랜서, 농어업 종사자 등
사업장가입자: 1인 이상 근로자를 고용하는 사업장(회사)의 사용자(사업주)와 근로자

3. 데이터 전처리

3.1 범죄 발생 지역별 통계

서울시 범죄 데이터 형식

자치구별(1	자치구별(2	2014	2015	2016	2017
자치구별(1	자치구별(2	합계	합계	합계	합계
자치구별(1	자치구별(2	발생	발생	발생	발생
합계	종로구	5021	4705	4459	4057
합계	중구	5231	4954	4584	4184
합계	용산구	3799	3820	4137	4060

3.1-1. 파일을 읽어오고

```
import pandas as pd

filepath = "/content/5대+범죄+발생현황.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)}")
```

3.1-2. 컬럼명을 지정

```
# ======================================================
# 2. 컬럼명 정리
# ======================================================
df.columns = ['category', 'region', '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021', '2022', '2023', '2024']
```

3.1-3. '소계' 행 제거 및 구 데이터만 추출

```
# ======================================================
# 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())
```

3.1-4. Long format으로 변환

```
# ======================================================
# 4. Wide format → Long format 변환
# ======================================================
df_long = df_clean.melt(
    id_vars=['region'],
    value_vars=['2014', '2015', '2016', '2017','2018', '2019', '2020', '2021', '2022', '2023', '2024'],
    var_name='year',
    value_name='total'
)
```

3.1-5. 데이터 타입 변환

```
# ======================================================
# 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)
```

3.1-6. 정렬(데이터 확인)

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

print(f"₩n 최종 데이터: {len(df_long)}행")
print(f"   연도 범위: {df_long['year'].min()} ~ {df_long['year'].max()}")
print(f"   자치구 수: {df_long['region'].nunique()}")
```

3.1-7. 파일 저장

```
# ======================================================
# 7. 저장
# ======================================================
output_file = "/content/서울시_25개구_5대범죄_연도별통합.csv"
df_long.to_csv(output_file, index=False, encoding="utf-8-sig")

print(f"₩n 저장 완료: {output_file}")
```

3.1-8. 저장된 파일 미리보기

```
# ======================================================
# 8. 결과 미리보기
# ======================================================
print("₩n 데이터 샘플 (각 구별 첫 해):")
print(df_long.groupby('region').first().reset_index())

print("₩n 전체 데이터 처음 20행:")
print(df_long.head(20))

print("₩n 연도별 서울시 전체 범죄 발생 건수:")
yearly_total = df_long.groupby('year')['total'].sum().reset_index()
yearly_total.columns = ['연도', '총 발생건수']
print(yearly_total)
```

3.2 인구데이터전처리 및 범죄율 계산

서울시 인구 데이터 형식

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

3.2-1. 데이터 로드(자동 인코딩)

```
import pandas as pd
import numpy as np

# ================================
# 1. 데이터 로드
# ================================
population_file = '2025_서울시_구별_인구.csv'
crime_file = '서울시_25개구_5대범죄_연도별통합.csv'

# 인코딩을 시도하며 읽기
try:
    pop_df = pd.read_csv(population_file, encoding='cp949')
except:
    try:
        pop_df = pd.read_csv(population_file, encoding='euc-kr')
    except:
        pop_df = pd.read_csv(population_file, encoding='utf-8-sig')

try:
    crime_df = pd.read_csv(crime_file, encoding='cp949')
except:
    try:
        crime_df = pd.read_csv(crime_file, encoding='euc-kr')
    except:
        crime_df = pd.read_csv(crime_file, encoding='utf-8-sig')

print("데이터 로드 완료")
```

3.2-2. 인구 데이터 전처리 2014~2024년까지 데이터 추출해서 이상치 제거, 계 제거

```
# ============================
# 2. 인구 데이터 전처리 (2015-2024년만, 2025년과 2014년 이전 제외)
# ============================
# 첫 행이 헤더인 경우 제거
if '세대' in str(pop_df.iloc[0]['동별(2)']):
    pop_df = pop_df.iloc[1:].reset_index(drop=True)

# 범죄 데이터가 있는 연도만 사용 (2015-2024)
years = [2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024]
population_data = []

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

    # 합계, 총계, 소계 등 제외
    if pd.isna(region) or region == '' or '합계' in str(region) or '총계' in str(region) or '소계' in str(region):
        continue

    for year in years:
        # 해당 연도의 컬럼 찾기
        year_cols = [col for col in pop_df.columns if str(year) in str(col)]

        if year_cols:
            # 첫 번째 또는 두 번째 컬럼 사용 (세대가 아닌 인구수)
            col_name = year_cols[0] if len(year_cols) == 1 else year_cols[1]

            if pd.notna(row[col_name]):
                pop_value = row[col_name]
                # 콤마 제거
                if isinstance(pop_value, str):
                    pop_value = pop_value.replace(',', '').strip()
                try:
                    pop_value = float(pop_value)
                    if pop_value > 0:  # 유효한 값만 추가
                        population_data.append({
                            'region': region.strip(),
                            'year': year,
                            'population': pop_value
                        })
                except:
                    pass

pop_clean = pd.DataFrame(population_data)
print(f" 인구 데이터 전처리 완료: {len(pop_clean)}개 레코드")
print(f"   연도 범위: {pop_clean['year'].min()} - {pop_clean['year'].max()}")
print(f"   구 개수: {pop_clean['region'].nunique()}개")
```

3.2-3. 범죄 데이터 확인 및 병합

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

```
# ================================
# 3. 범죄 데이터 확인 및 병합
# ================================
print("₩n 범죄 데이터 컬럼:", crime_df.columns.tolist())


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")
```

3.2-4. 데이터 정렬 및 저장

```
# ================================
# 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")
```

3.2-5. 데이터 미리보기

```
# ================================
# 5. 미리보기
# ================================
print("₩n" + "="*60)
print(" 저장된 데이터 미리보기")
print("="*60)
print(merged_df_sorted[['region', 'year', 'crime_rate']].head(10))
print("₩n연도별 레코드 수:")
print(merged_df_sorted.groupby('year').size())

print(f"₩n {output_file} 파일이 생성되었습니다.")
```

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

3.3-1. 파일 읽기

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

3.3-2. 컬럼 공백 제거

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

3.3-3. 숫자형 데이터 변환

```
# ========================================
# 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)
```

3.3-4. 컬럼명 변환

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

3.3-5. long format 변환

```
# ====================================================
# 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)
```

3.3-6. csv저장

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

output
```

3.4 서울시 구별 비상벨 설치 정보

비상벨 설치 데이터 형식

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

3.4-1. 파일 불러오기 및 변수(컬럼명) 설정

```
# ======================================================
# 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"
```

3.4-2. 서울시 25개구 리스트 지정

```
# ======================================================
# 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"
]
```

3.4-3. 주소에서 구추출 함수

```
# ======================================================
# 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|도봉구|" #
              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
```

3.4-4. 최종 지역 추출 함수 (지역이 있을 만한 컬럼들을 확인)

```
# ======================================================
# 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)
```

3.4-5. 지역 추출 결과 확인

```
# ======================================================
# 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())
```

3.4-6. 주소/관리기관명 전체에서 등장한 구를 저장 (이상치 검출용)

```
# ======================================================
# 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))
```

3.4-7.설치연도 이상치 확인 및 제거

```
# ======================================================
# 7) 설치연도 이상치 확인 및 제거
# ======================================================
# 2025년 초과 연도 데이터 추출 (제거 전 확인)
over_2025 = df[df[col_year] > 2025]
over_2025_info = over_2025[[col_year, "최종지역", col_road, col_addr, col_manage]]

print("₩n----- 2025년 초과 설치 연도 데이터 (제거 대상) -----")
print(over_2025_info)
print(f"₩n제거할 데이터 개수: {len(over_2025)}개")

if len(over_2025) > 0:
    print("₩n제거되는 지역별 개수:")
    print(over_2025_info["최종지역"].value_counts())

# 이상치 제거: 2025년 이하 데이터만 남김
df_clean = df[(df[col_year] <= 2025) & df[col_year].notna()].copy()
df_clean["설치대수"] = 1

print(f"₩n정상 데이터 개수: {len(df_clean)}개")
```

3.4-8. 연도별 + 지역별 집계

```
# ======================================================
# 8) 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()
```

3.4-9. CSV 생성

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

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
```

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

번호	개방서비스명	서비스아이디	자치단체코드	관리번호	인허가일자	허가취소일자	영업상태구분코드	영업상태명	세부영업상태코드	세부영업상태명	폐업일자	휴업시작일자	휴업종료일자
1	유흥주점업	07_23_02_P	3060000	3060000-12025-11-1			01	영업/정상	01	영업			
2	유흥주점업	07_23_02_P	3220000	3220000-12025-10-3			01	영업/정상	01	영업			
3	유흥주점업	07_23_02_P	3180000	3180000-12025-10-2			01	영업/정상	01	영업			
4	유흥주점업	07_23_02_P	3010000	3010000-12025-10-1			01	영업/정상	01	영업			

3.5-1. 파일 읽어 오기

```
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())
```

3.5-2. 주소에서 서울시 25개구 추출하는 함수

```
# ============================================================
# 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
```

3.5-3. 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))
```

3.5-4. 인허가일자에서 연도 추출

```
# ======================================================
# 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)
```

3.5-5. 연도별 + 구별 등록된 유흥주점 수

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

3.5-6. 누적 합 계산 (총 유흥주점 수 = 저번년도 + 이번년도)

```
# ======================================================
# 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()
```

3.5-7. CSV로 저장

```
# ======================================================
# 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')

output_file
```

3.6 서울시 년도별 지역별 월평균 소득

3.6-1. 데이터 읽어오기, 확인

```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')

# ============================================================
# 1) 데이터 로딩
# ============================================================
crime_file = 'crime_rate_long.csv'
income_file = '2019년도 4월기준 14~18년 시군구별 평균소득월액.csv'

df_crime = pd.read_csv(crime_file, encoding='utf-8-sig')
df_income = pd.read_csv(income_file, encoding='cp949')

print("=" * 60)
print("✓ 데이터 로딩 완료")
print("=" * 60)
print("₩n=== 범죄 데이터 샘플 ===")
print(df_crime.head())
print(f"₩n레코드 수: {len(df_crime)}, 컬럼: {df_crime.columns.tolist()}")

print("₩n=== 평균소득 데이터 샘플 ===")
print(df_income.head())
print(f"₩n레코드 수: {len(df_income)}, 컬럼: {df_income.columns.tolist()}")
```

3.6-2. 지역명 정규화 함수

```
# ============================================================
# 2) 지역명 정규화 함수
# ============================================================
def normalize_region(text):
    """지역명을 'OO구' 형태로 통일"""
    if pd.isna(text):
        return None
    text = str(text).strip()
    text = text.replace('구청', '').replace('서울특별시', '').replace('서울시', '').replace('서울 ', '').strip()
    if not text.endswith('구'):
        text = text + '구'
    return text
```

3.6-3. 범죄 데이터 읽어오기

```
# ============================================================
# 3) 범죄 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']

print("₩n" + "=" * 60)
print("✓ 범죄 데이터 전처리 완료")
print("=" * 60)
print(df_crime_clean.head())
print(f"₩n연도 범위: {df_crime_clean['연도'].min()} ~ {df_crime_clean['연도'].max()}")
print(f"지역 수: {df_crime_clean['지역'].nunique()}개")
```

3.6-4. 평균소득 데이터 전처리

```
# ============================================================
# 4) 평균소득 데이터 전처리
# ============================================================
df_income['지역'] = df_income['시군구'].apply(normalize_region)

# 연도 컬럼을 long format으로 변환
income_years = ['2014년', '2015년', '2016년', '2017년', '2018년']
df_income_long = pd.melt(
    df_income,
    id_vars=['지역'],
    value_vars=income_years,
    var_name='연도_원본',
    value_name='평균소득'
)

# 연도를 숫자로 변환 (2014년 -> 2014)
df_income_long['연도'] = df_income_long['연도_원본'].str.replace('년', '').astype(int)
df_income_long = df_income_long[['연도', '지역', '평균소득']].copy()

print("₩n" + "=" * 60)
print("✓ 평균소득 데이터 전처리 완료")
print("=" * 60)
print(df_income_long.head())
print(f"₩n연도 범위: {df_income_long['연도'].min()} ~ {df_income_long['연도'].max()}")
print(f"지역 수: {df_income_long['지역'].nunique()}개")
```

3.6-5. 데이터 병합

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

print("₩n" + "=" * 60)
print("✓ 데이터 병합 완료")
print("=" * 60)
print(df_merged.head(10))
print(f"₩n병합된 레코드 수: {len(df_merged)}개")
print(f"분석 연도: {df_merged['연도'].min()} ~ {df_merged['연도'].max()}")
print(f"분석 지역 수: {df_merged['지역'].nunique()}개")
print(f"분석 지역: {sorted(df_merged['지역'].unique())}")
```

3.6-6. CSV 저장

```
# ============================================================
# 6) CSV 저장
# ============================================================
output_file = 'income_crime_merged.csv'
df_merged.to_csv(output_file, index=False, encoding='utf-8-sig')

print("₩n" + "=" * 60)
print(f"✓ CSV 저장 완료: {output_file}")
print("=" * 60)
print("  컬럼: 연도, 지역, 인구수, 범죄건수, 범죄율, 평균소득")
print("₩n데이터 통계:")
print(df_merged[['범죄율', '평균소득', '인구수']].describe())

print(f"₩n✓ {output_file} 파일이 생성되었습니다.")
```

4. 시각화 코드

4.1 서울시 년도별, 지역별 범죄율 시각화

4.1-1. 범죄율 데이터 읽어오기

```
import pandas as pd
import numpy as np
import plotly.graph_objects as go
import requests
import warnings
warnings.filterwarnings('ignore')

# ================================
# 1. CSV 파일 로드
# ================================
csv_file = 'crime_rate_long.csv'

try:
    merged_df = pd.read_csv(csv_file, encoding='utf-8-sig')
    print(f"✓ CSV 로드 완료: {len(merged_df)}개 레코드")
    print(f"  연도 범위: {merged_df['year'].min()} - {merged_df['year'].max()}")
    print(f"  구 개수: {merged_df['region'].nunique()}개")
except FileNotFoundError:
    print(f"오류: {csv_file} 파일을 찾을 수 없습니다.")
    print("먼저 CSV 생성 코드를 실행해주세요.")
    exit()
```

4.1-2. 호버 텍스트 생성

```
# ============================
# 2. 호버 텍스트 생성
# ============================
merged_df['text'] = merged_df.apply(
    lambda x: f"<b>{x['region']}</b><br>범죄율: {x['crime_rate']:.1f}<br>범죄 건수: {x['total']:,}건<br>인구: {x['population']:,.0f}명",
    axis=1
)
```

4.1-3. 서울시 GeoJSON 다운로드

```
# ============================
# 3. 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()
    print("✓ GeoJSON 로드 완료")

    # 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
```

4.1-4. Plotly Choropleth 지도 생성

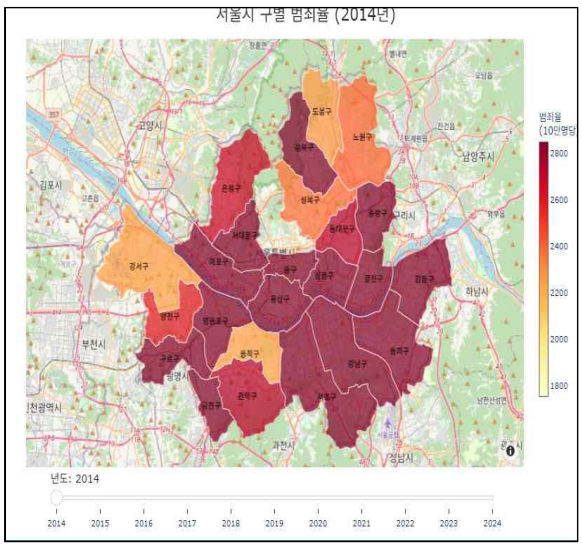
```
# ==============================
# 4. Plotly Choropleth 지도 생성
# ==============================
if seoul_geo:
    # 색상 범위를 분위수 기준으로 조정
    all_rates = merged_df['crime_rate'].values
    q25 = np.percentile(all_rates, 25)
    q75 = np.percentile(all_rates, 75)

    fig = go.Figure()

    # 각 연도별 프레임 생성
    years = sorted(merged_df['year'].unique())

    for year in years:
        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 == years[4])  # 2018년을 기본으로 표시
        ))
```

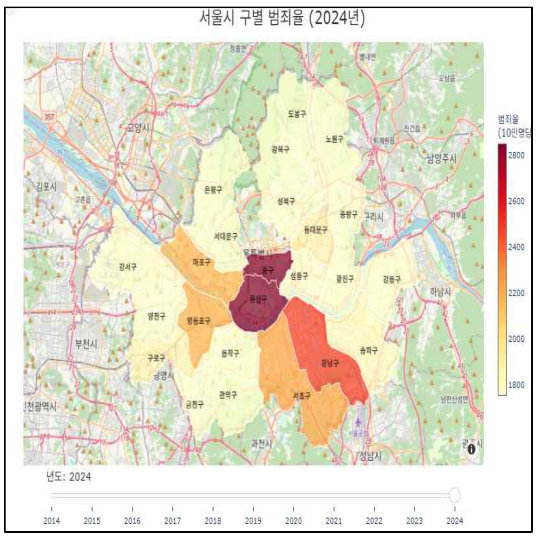
```
    # 각 구에 텍스트 라벨 추가 (지역명만 표시)
    fig.add_trace(go.Scattermapbox(
        lat=year_data['lat'],
        lon=year_data['lon'],
        mode='text',
        text=year_data['region'],
        textfont=dict(
            size=11,
            color='black',
            family='Arial Black'
        ),
        hoverinfo='skip',
        name=f'{year}_labels',
        visible=(year == years[4])  # 2018년을 기본으로 표시
    ))

# 슬라이더 생성
steps = []
for i, year in enumerate(years):
    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=4,  # 2018년 인덱스
    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=f'서울시 구별 범죄율 ({years[4]}년)',
            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,
        dragmode='zoom'
    )

    print("₩n✓ 시각화 생성 완료")
    fig.show()

else:
    print("⚠ 오류: GeoJSON을 로드할 수 없어 지도를 생성할 수 없습니다.")
```

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

4.1-5. 결과





중구와 용산구가 지속적으로 가장 범죄율이 높고 전체적으로 년도가 지날수록 범죄율이 감소한다

4.2 cctv와 범죄율 연관 시각화

```
# 먼저 이 셀을 실행하세요 (한글 폰트 설치)
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf
```

한글 폰트 설치후 런타임 재실행

4.2-1. 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings

# 한글 폰트 설정
plt.rcParams['font.family'] = 'NanumGothic'
plt.rcParams['axes.unicode_minus'] = False
warnings.filterwarnings('ignore')

# ==============================
# 1. 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)
```

4.2-2. 지역명 정규화

```
# ==============================
# 2. 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
```

4.2-3. 데이터 전처리

```
# ==============================
# 3. 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']
```

4.2-4. 데이터 병합

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

4.2-5. 상관계수 계산

```
# ================================
# 5. 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")
```

4.2-6. 시각화 (2x2 레이아웃)

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

# 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)
```

4.2-7. 연도별 평균 추세 (이중 y축)

```
# ================================
# 7. 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)
```

4.2-8. 상관계수 히트맵

```
# ================================
# 8. 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)
```

4.2-9.

```
# ================================
# 9. 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()
```

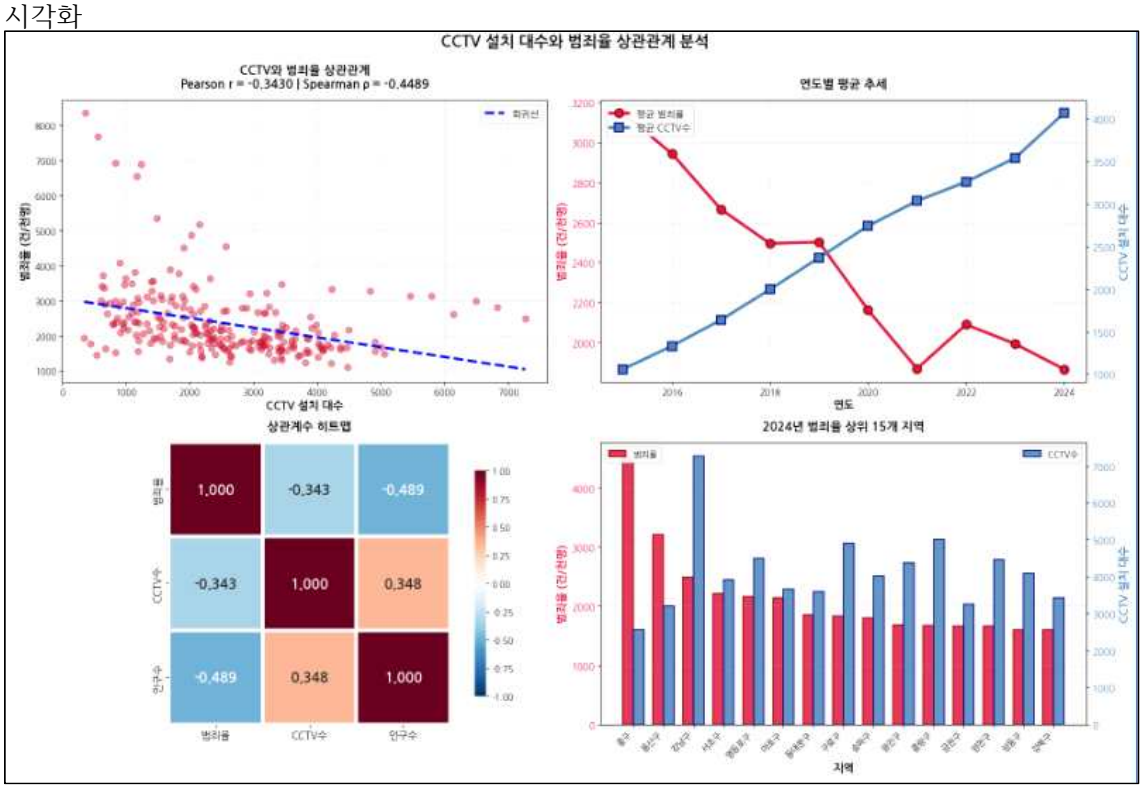
4.2-10. 결과

Pearson 상관계수: -0.3430

Spearman 상관계수: -0.4489

분석 데이터: 240개

시각화



해가 갈수록 cctv 설치 수가 꾸준히 증가하고 범죄율도 꾸준히 감소하고 있다
상위 15개 지역 그래프를 보면 cctv수가 가장 적은 두 지역의 범죄율 또한 가장 높다
지역별 cctv 설치개수가 범죄율에 유의미한 상관관계를 가진다

4.3 비상벨과 범죄율 연관 시각화

4.3-1. 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings

# 한글 폰트 설정
plt.rcParams['font.family'] = 'NanumGothic'
plt.rcParams['axes.unicode_minus'] = False
warnings.filterwarnings('ignore')

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

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

4.3-2. 지역명 정규화

```
# ================================
# 2. 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
```

4.3-3. 데이터 전처리

```
# ================================
# 3. 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']
```

4.3-4. 데이터 병합

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

4.3-5. 상관계수 계산

```
# ================================
# 5. 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")
```

4.3-6. 산점도 + 회귀선 (상관계수)

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

#  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)
```

4.3-7. 연도별 평균 추세 (이중 y축)

```
# ================================
# 7. 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)
```

4.3-8. 상관계수 히트맵

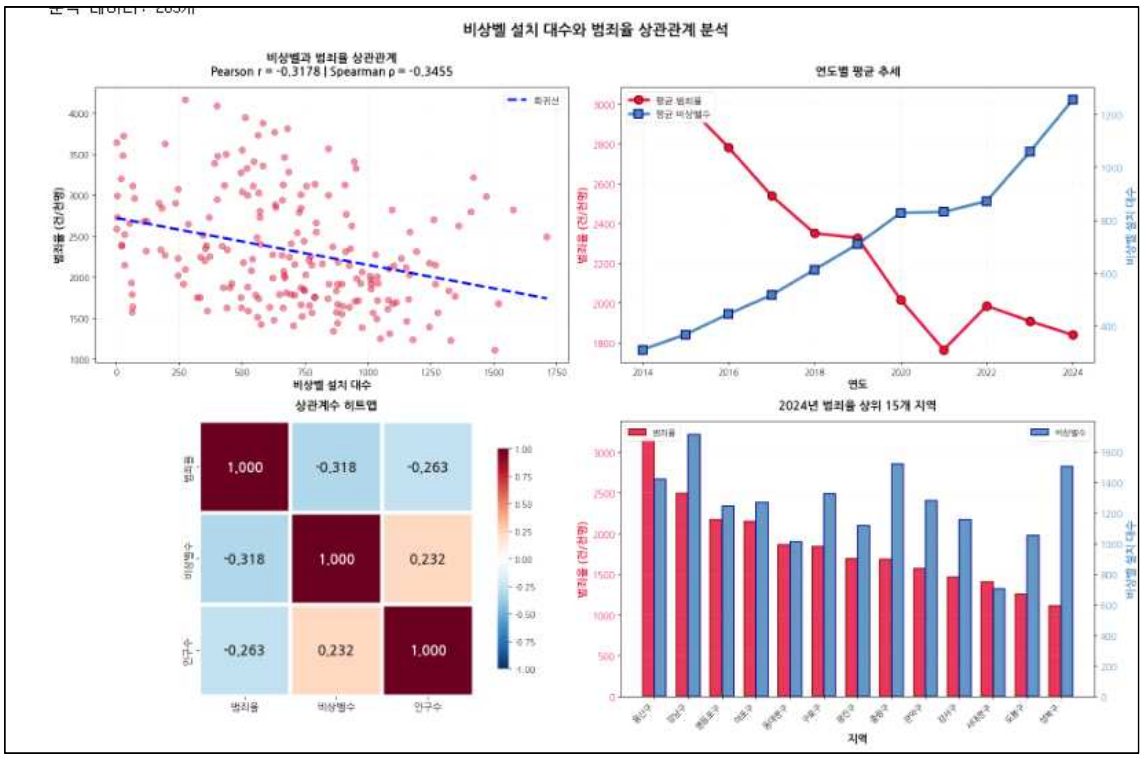
```
# ================================
# 8. 상관계수 히트맵
# ================================
ax3 = plt.subplot(2, 2, 3)
corr_matrix = df_merged[['범죄율', '비상벨수', '인구수']].corr()
sns.heatmap(corr_matrix, annot=True, fmt='.3f', cmap='RdBu_r',
            center=0, square=True, linewidths=3, cbar_kws={"shrink": 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)
```

4.3-9. 2024년 범죄율 상위 15개 지역

```
# ================================
# 9. 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()
```

Pearson 상관계수: -0.3178
Spearman 상관계수: -0.3455
분석 데이터: 205개

4.3-10. 결과



비상벨과 범죄율이 증가 함에 따라 상관계수가 cctv보다 낮지만 유의미하게 나오는 반면 2024년 상위 15개 지역을 살펴보면 비상벨 개수와 범죄율의 감소와는 유의미하게 상관관계가 있지 않다.

4.4 유흥주점 수와 범죄율 연관 시각화

4.4-1. 데이터 로딩 (파일 경로는 실제 환경에 맞게 수정)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings

# 한글 폰트 설정
plt.rcParams['font.family'] = 'NanumGothic'
plt.rcParams['axes.unicode_minus'] = False
warnings.filterwarnings('ignore')

# ==============================
# 1. 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)
```

4.4-2. 지역명 정규화

```
# ==============================
# 2. 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
```

4.4-3. 데이터 전처리

```
# ==============================
# 3. 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']
```

4.4-4. 데이터 병합

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

4.4-5. 상관계수 계산

```
# ================================
# 5. 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")
```

4.4-6. 산점도 + 회귀선 (상관계수)

```
# ================================
# 6. 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)
```

4.4-7. 연도별 평균 추세 (이중 y축)

```
# ==============================
# 7. 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)
```

4.4-8. 상관계수 히트맵

```
# ================================
# 8. 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)
```

4.4-9. 2024년 범죄율 상위 15개 지역

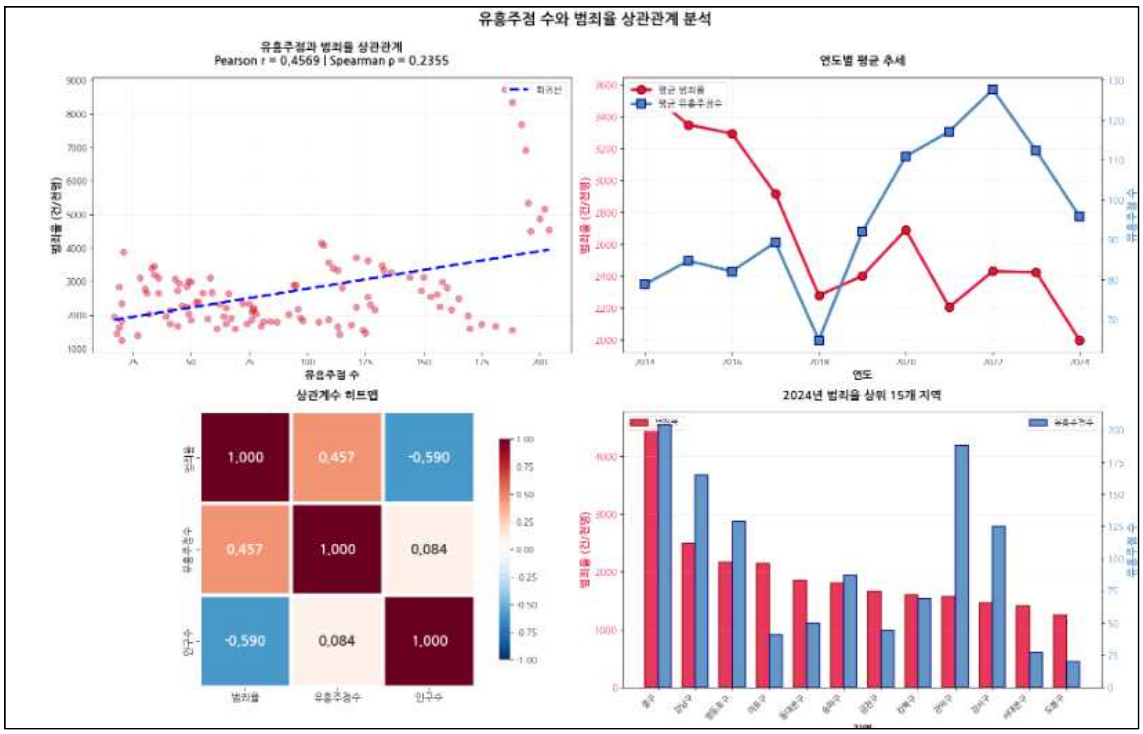
```
# ==============================
# 9. 2024년 범죄율 상위 15개 지역
# ==============================
ax4 = plt.subplot(2, 2, 4)
latest_year = df_merged['연도'].max()
df_latest = df_merged[df_merged['연도'] == latest_year].copy()
df_top = df_latest.nlargest(15, '범죄율')

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()
```

4.4-10. 결과

Pearson 상관계수: 0.4569

Spearman 상관계수: 0.2355

분석 데이터: 110개



유흥지점은 점차 증가하고 범죄율은 점차 감소하고 있다

유흥지점이 많은 지역은 범죄율 또한 높은 편이다

유흥지점과 범죄율은 높은 상관관계를 가지고 있다

4.5 유흥주점 수와 범죄율 연관 시각화

4.5-1. 데이터 읽어오기

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings

# 한글 폰트 설정
plt.rcParams['font.family'] = 'NanumGothic'
plt.rcParams['axes.unicode_minus'] = False
warnings.filterwarnings('ignore')

# ============================================================
# 1. CSV 데이터 로딩
# ============================================================
csv_file = 'income_crime_merged.csv'

try:
    df_merged = pd.read_csv(csv_file, encoding='utf-8-sig')
    print("=" * 60)
    print("✓ CSV 로딩 완료")
    print("=" * 60)
    print(f"레코드 수: {len(df_merged)}개")
    print(f"분석 연도: {df_merged['연도'].min()} ~ {df_merged['연도'].max()}")
    print(f"분석 지역 수: {df_merged['지역'].nunique()}개")
    print(f"₩n컬럼: {df_merged.columns.tolist()}")
    print("₩n데이터 샘플:")
    print(df_merged.head())
except FileNotFoundError:
    print(f"오류: {csv_file} 파일을 찾을 수 없습니다.")
    print("먼저 전처리 코드를 실행해주세요.")
    exit()
```

4.5-2. 상관계수 계산

```
# ============================================================
# 2. 상관계수 계산
# ============================================================
corr_pearson = df_merged['범죄율'].corr(df_merged['평균소득'])
corr_spearman, spearman_p = stats.spearmanr(df_merged['범죄율'], df_merged['평균소득'])

print("₩n" + "=" * 60)
print("전체 기간 상관관계 분석 결과")
print("=" * 60)
print(f"Pearson 상관계수: {corr_pearson:.4f}")
print(f"Spearman 상관계수: {corr_spearman:.4f} (p-value: {spearman_p:.4f})")
```

4.5-3. 시각화 (2x2 레이아웃) 산점도 + 회귀선 (전체 기간)

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

# 1. 산점도 + 회귀선 (전체 기간)
ax1 = plt.subplot(2, 2, 1)
ax1.scatter(df_merged['평균소득'], df_merged['범죄율'],
           alpha=0.5, s=60, c='crimson', edgecolors='darkred', linewidth=0.5)
z = np.polyfit(df_merged['평균소득'], df_merged['범죄율'], 1)
p = np.poly1d(z)
x_line = np.linspace(df_merged['평균소득'].min(), df_merged['평균소득'].max(), 100)
ax1.plot(x_line, p(x_line), "blue", linestyle='--', linewidth=3, label='회귀선')
ax1.set_xlabel('평균소득 (만원)', fontsize=13, weight='bold')
ax1.set_ylabel('범죄율 (건/10만명)', fontsize=13, weight='bold')
ax1.set_title(f'평균소득과 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)
```

4.5-4. 연도별 평균 추세 (이중 y축)

```
# ======================================================
# 4. 연도별 평균 추세 (이중 y축)
# ======================================================
ax2 = plt.subplot(2, 2, 2)
yearly = df_merged.groupby('연도')[['범죄율', '평균소득']].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['평균소득'],
                      marker='s', linewidth=3.5, markersize=10,
                      color='steelblue', label='평균소득', markeredgecolor='darkblue', markeredgewidth=1.5)

ax2.set_xlabel('연도', fontsize=13, weight='bold')
ax2.set_ylabel('범죄율 (건/10만명)', fontsize=13, weight='bold', color='crimson')
ax2_twin.set_ylabel('평균소득 (만원)', 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)
```

4.5-5. 상관계수 히트맵

```
# ======================================================
# 5. 상관계수 히트맵
# ======================================================
ax3 = plt.subplot(2, 2, 3)
corr_matrix = df_merged[['범죄율', '평균소득', '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)
```

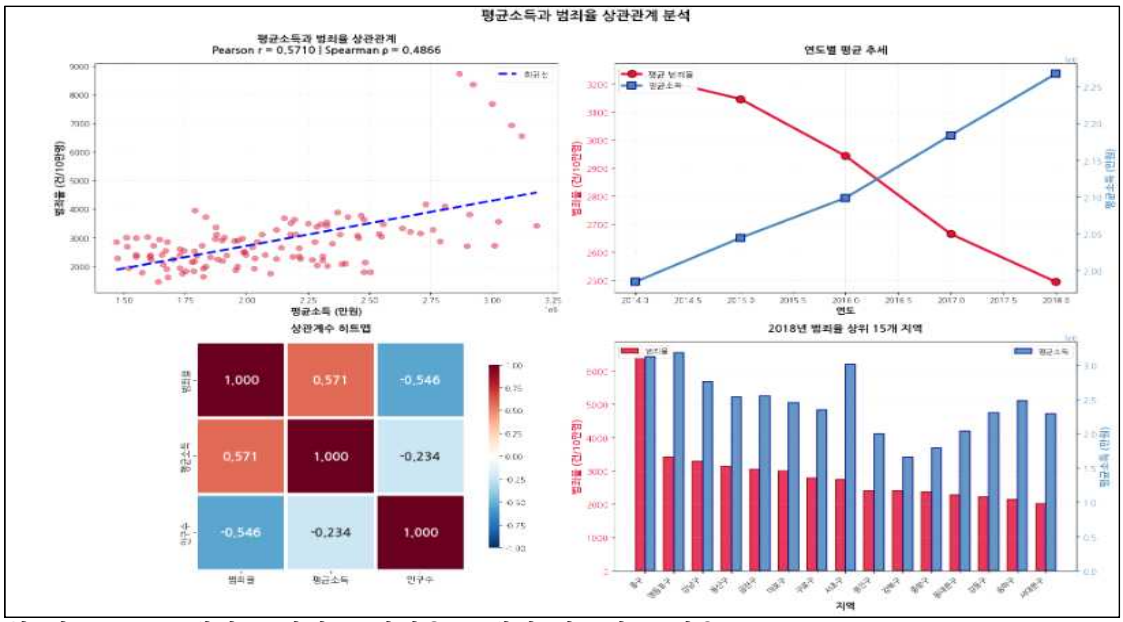
4.5-6. 최신 연도 범죄율 상위 15개 지역

```
# ======================================================
# 6. 최신 연도 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['평균소득'], width,
                     label='평균소득', color='steelblue', alpha=0.85, edgecolor='darkblue', linewidth=1.5)

ax4.set_xlabel('지역', fontsize=13, weight='bold')
ax4.set_ylabel('범죄율 (건/10만명)', fontsize=13, weight='bold', color='crimson')
ax4_twin.set_ylabel('평균소득 (만원)', 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('평균소득과 범죄율 상관관계 분석',
             fontsize=18, weight='bold', y=0.995)
plt.tight_layout()
plt.show()
```

4.5-7. 결과

Pearson 상관계수: 0.5710

Spearman 상관계수: 0.4866 (p-value: 0.0000)



월 평균소득은 매년 증가하고 범죄율은 매년 감소하고 있음

월 평균소득과 범죄율의 상관계수는 매우 높은 값을 나타내고 있다

범죄율 상위 15개 지역 그래프에서는 뚜렷한 상관관계를 가지지 않는다

5. 분석 결과

5.1 각 분석 결과
범죄율 시각화에서 매년 범죄율이 감소하고 있고

cctv와 범죄율
CCTV 설치 수가 증가할수록 범죄율이 감소하는 경향이 나타남.
특히 Spearman 계수가 더 크게 나타나, 선형 관계뿐 아니라 전반적인 감소 추세가 비교적 뚜렷함.
이는 CCTV가 범죄 예방 및 억제 효과를 일정 부분 갖고 있음을 시사함.

비상벨 설치 수와 범죄율
비상벨 설치 수가 많을수록 범죄율이 완만하게 감소하는 경향이 확인됨.
CCTV보다 상관 정도는 다소 약하지만, 일관된 음의 상관관계를 보임.

유흥주점 수와 범죄율
유흥주점 수가 증가할수록 범죄율이 증가하는 경향이 나타남.
Pearson 계수가 상대적으로 높아, 일정 수준의 선형적 증가 관계가 존재함을 시사함.

평균 월소득과 범죄율
평균 월소득과 범죄율 사이에 중간 이상 수준의 양의 상관관계가 확인됨.
p-value가 0.0000으로 통계적으로 매우 유의미한 관계임.

5.2 결론
5. 종합 분석 및 시사점
치안 시설(CCTV, 비상벨)은 범죄율과 음의 상관관계를 보여 범죄 억제 효과를 시사함.
반면, 유흥주점 수와 평균 소득은 범죄율과 양의 상관관계를 보여, 사회·환경적 요인이 범죄 발생에 중요한 영향을 미침.
따라서 범죄 예방을 위해서는 단순히 치안 시설을 늘리는 것뿐만 아니라, 지역 특성(상업·유흥 밀집도, 생활 패턴)을 고려한 종합적인 정책이 필요함.