■ 프로젝트에 사용되는 데이터

a. 제공된 데이터

KP_2020, KP2021의 데이터

Column	Description
RECV_DEPT_NM	접수 부서
RECV_CPLT_DM	접수 완료 일시
NPA_CL	경찰청 구분(코드번호)
EVT_STAT_CD	사건 상태 코드
EVT_CL_CD	사건 종별 코드
RPTER_SEX	성별 (남:1, 여:2, 불상:3)
HPPN_PNU_ADDR	사건 발생 주소
HPPN_X	주소 좌표 (경도)
HPPN_Y	주소 좌표 (위도)
SME_EVT_Y	동일 사건 여부

NPA_2020데이터

Column	Description
RECV_CPLT_DT	접수 완료 일자
RECV_CPLT_TM	접수 완료 시간
NPA_CL	경찰청 구분(코드번호)
EVT_STAT_CD	사건 상태 코드
EVT_CL_CD	사건 종별 코드
RPTER_SEX	성별 (남:1, 여:2, 불상:3)
HPPN_PNU_ADDR	사건 발생 주소
HPPN_X	주소 좌표 (경도)
HPPN_Y	주소 좌표 (위도)
SME_EVT_Y	동일 사건 여부

제공된 데이터는 위와 같이 구성되어 있음. 접수 부서, 접수 완료 일시, 성별, 동일 사건 여부와 같은 정보는 향후 프로젝트에 사용되지 않으므로 제거하여 나머지 컬럼만 사용하였음.

b. 충남 지역 CCTV 데이터

전국무인교통단속카메라표준데이의 아산, 보령, 충남, 대전, 금산, 세종, 서천, 서산, 태안, 예산, 당진의 CCTV 데이터를 활용합니다. **'도로노선명', '위도', '경도', '설치장소'** 컬럼만 사용하고 이외의 컬럼은 제거하였음.

	도로노선명	위도	경도	설치장소
0	실옥로	36.787433	126.992743	천도초등학교주변
1	시민로	36.781864	127.000437	온양관광호텔 사거리
2	온천대로	36.778944	127.018453	동신초등학교주변
3	번영로	36.786259	127.005606	아고오거리주변
4	번영로	36.786764	127.002256	시민로사거리주변

그림 3 아산 CCTV 데이터 상위 5개

■ 데이터 전처리

a. 널값 제거

```
KP_2021.isnull().sum()
✓ 0.3s
NPA_CL              0
EVT_STAT_CD         0
EVT_CL_CD           0
RPTER_SEX       23894
HPPN_X         694401
HPPN_Y         694401
SME_EVT_YN    2079123
dtype: int64
```

KP_2020, KP_2021, NPA_2020의 경우 'RPTER_SEX',' 'HPPN_X', 'HPPN_Y', 'SME_EVT_YN' 컬럼의 경우 위 사진과 같이 수많은 공백 값이 있으므로 대체값을 적용하려 하였으나, 성별과 주소 좌표는 고유의 값이 필요한 경우이므로, 이 경우 또한 컬럼을 삭제하였음.

또한 각 **지역별 CCTV 설치 현황**을 확인하기 위한 데이터도 동일하게 널값을 제거하였음.

■ 데이터 통합

a. CCTV 데이터

위에서 전처리된 지역별 CCTV 데이터를 하나의 DataFrame으로 통합하였음.
또한 비교적 큰 범위인 충남, 대전, 세종 DataFrame으로도 나눠 저장하였음.

```
#전체
CCTV = pd.concat([ASAN_CCTV,BORYEONG_CCTV,CHEONAN_CCTV,CHUNGYANG_CCTV,DAEJEON_CCTV,DANGJIN_CCTV,GEOMSAN_CCTV,SEJONG_CCTV,
                  SEOCHEON_CCTV,SEOSAN_CCTV,TAEAN_CCTV,YESAN_CCTV  ])
#충남
CH_CCTV = pd.concat([ASAN_CCTV,BORYEONG_CCTV,CHEONAN_CCTV,CHUNGYANG_CCTV,DANGJIN_CCTV,GEOMSAN_CCTV,
                  SEOCHEON_CCTV,SEOSAN_CCTV,TAEAN_CCTV,YESAN_CCTV  ])
#대전
DJ_CCTV = pd.concat([DAEJEON_CCTV])
#세종
SJ_CCTV = pd.concat([SEJONG_CCTV])
```

b. 경찰청 데이터

전처리된 KP_2020, KP_2021, NPA_2020 데이터를 통합하였음.

```
All = pd.concat([k_2020,k_2021,n_2020])
```

■ 교통사고 건 추출

```
k_2020 = k_2020[k_2020['EVT_CL_CD']==401]
k_2021 = k_2021[k_2021['EVT_CL_CD']==401]
n_2020 = n_2020[n_2020['EVT_CL_CD']==401]
```

구분	건 수
k_2020	6207
k_2021	164582
n_2020	93659
전체	264448

■ 충남 대전 세종 지역 추출

■

데이터의 위·경도 중 충남, 대전, 세종 지역만 다루기 위해 지역을 추출하는 과정이 필요함.

참고문헌에 탑재된 동,서,남,북단을 참고 및 직접 지도에서 추출하였음.

구분	동단	서단	남단	북단
충남	127.62692947603428	126.13148910811715	35.98803519542761	37.056951491204124
대전	127.55529876331815	127.25007536223731	36.186117749974656	36.49779017085958
세종	127.40746822278025	127.12869800443497	36.41194803285692	36.70492

■ 지역 추출 변환 함수

```
# 원하고자 하는 지역 외의 데이터 삭제 및 뽑기
# 변환데이터, 원본데이터, 동단경도, 서단경도, 남단위도, 북단위도

# 대전, 세종에 해당
def translocation(TransData, Data,  Eastlongtitude, Westlongtitude,  Southlatitude, Northlatitude):

     #동단 삭제
    TransData = Data.drop(Data[(Data['HPPN_X']>Eastlongtitude)].index)

    #서단 삭제
    TransData = TransData.drop(TransData[(TransData['HPPN_X']<Westlongtitude)].index)

    #남단 삭제
    TransData = TransData.drop(TransData[(TransData['HPPN_Y']<Southlatitude)].index)

    #북단 삭제
    TransData =TransData.drop(TransData[(TransData['HPPN_Y']>Northlatitude)].index)

    return TransData

# 원하고자 하는 지역 만 뽑기(충남의 경우만 해당)

def translocation_CH(TransData, Data, Eastlongtitude, Westlongtitude, Southlatitude, Northlatitude):

    TransData = Data.drop(Data[(Data.HPPN_X < Eastlongtitude) & (Data.HPPN_X > Westlongtitude) &
                               (Data.HPPN_Y > Southlatitude) & (Data.HPPN_Y < Northlatitude)
                               ].index)
    return TransData

def duplicated_location(Data):
    print('중복 처리 전 교통사고 건수: ', len(Data))
    Data[Data.duplicated(['HPPN_X','HPPN_Y'])==True]
    Data = Data.drop_duplicates(['HPPN_X','HPPN_Y'])
    print('중복 처리 후 교통사고 건수 : ', len(Data))

    return Data
```

■ 지역추출변환함수 상세내용

구분	매개변수	설명
translocation	변환데이터, 데이터, 동단, 서단, 남단, 북단	매개변수로 받은 위·경도를 사각형기준으로 그렸을 때 해당되지 않은 데이터는 제거 후 반환
translocation_CH	변환데이터, 데이터, 동단, 서단, 남단, 북단	매개변수로 받은 위·경도를 사각형기준으로 그렸을 때 사각형 부분에 포함되는 데이터 제거 후 반환
duplicated_location	데이터	데이터의 위·경도가 같은 경우 삭제 – 예) 위도 0° 경도 0° 후 변환 내용 출력

충남 코드

```
# 충남 - 대전
Chungnam = pd.DataFrame()

Chungnam = translocation_CH(Chungnam, All, 127.55529876331815, 127.25007536223731, 36.186117749974656, 36.49779017085958 )
print('충남지역에서 대전지역에 해당하는 데이터 제거 후 개수 : ', len(Chungnam))

# 충남 - 세종   127.23, 127.1, 36.24, 36.43)
Chungnam = translocation_CH(Chungnam, Chungnam, 127.40746822278025, 127.12869800443497, 36.41194803285692, 36.70492)
print('충남지역에서 세종지역에 해당하는 데이터 제거 후 개수 : ', len(Chungnam))

Chungnam = translocation(Chungnam,Chungnam, 127.62692947603428,126.13148910811715,35.98803519542761,37.056951491204124 )
```

대전 코드

```
DAEJEON = pd.DataFrame()
DAEJEON = translocation(DAEJEON, All, 127.55529876331815, 127.25007536223731, 36.186117749974656, 36.49779017085958)
print('대전지역 교통사고 건수 : ', len(DAEJEON))
```

세종 코드

```
SEJONG = pd.DataFrame()
SEJONG = translocation(SEJONG, All, 127.40746822278025, 127.12869800443497, 36.41194803285692, 36.70492)
print('세종지역 교통사고 건수 : ', len(SEJONG))
```

지역추출변환 함수를 이용한 결과

데이터	중복 처리 전	중복 처리 후
충남	115403	105032
대전	72981	70712
세종	15588	14854
전체	188695	183404

■ 연도별 데이터

- 2020년

2020년 데이터는 k_2020, NPA_2020의 합으로 구성됨.

데이터	중복처리 전	중복처리 후
k_2020 + NPA_2020	99866	72985

지역 추출 변환 함수를 이용하여 2020년 데이터를 도출한 결과

2020년	건수	중복처리 전	중복 처리 후
충남	40888		
대전	27743	74281	72280
세종	5650		
2020년 전체	74281		

- 2021년

2021년 데이터는 k_2021으로 구성됨.

데이터	중복처리 전	중복처리 후
k_2021	164582	124871

2021년	건수	중복처리 전	중복 처리 후
충남	68511		
대전	49276	128108	124375
세종	10321		
2020년 전체	128108		

■ folium을 이용한 지도시각화

a. 교통사고 발생 지역 시각화

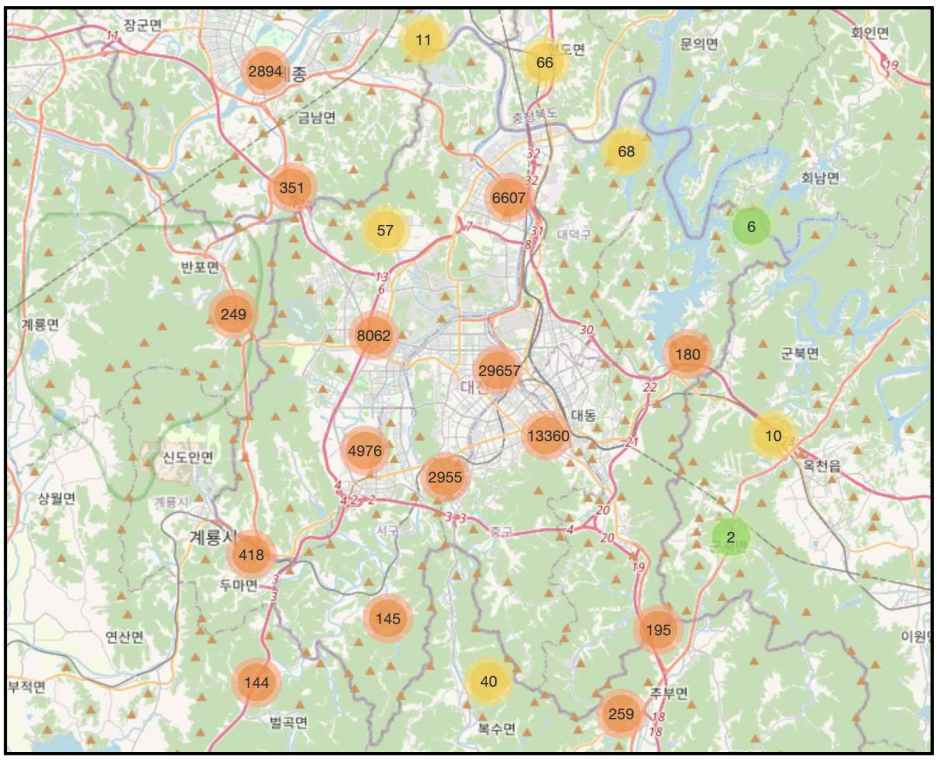
충남, 대전, 세종 지역별 교통사고 발생위치 파악을 위하여 Folium 지도 시각화를 진행하였음.

```
m = f.Map(location=[36.3504119,127.3845475], tiles='openstreetmap', zoom_start=11)

# Add  points to the map
mc = MarkerCluster()
for _, row in DAEJEON.iterrows():
    mc.add_child(
        Marker(location = [row['HPPN_Y'], row['HPPN_X']],
        popup= "{0}, {1}".format(TransName(DAEJEON),TransPolice(DAEJEON)),
        )
    )

m.add_child(mc)
m.add_child(f.LatLngPopup()) # 클릭시 위도, 경도 출력

# Display the map
m
```

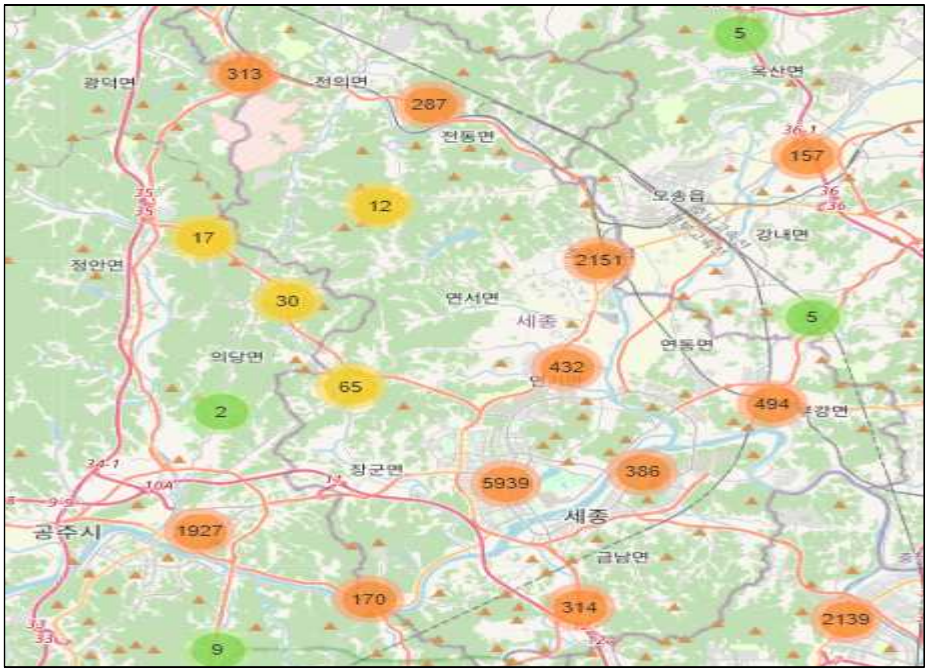
충남


20
11
1715
9549
21
46020
9485
490
5910
6154
266
438
3
2
5971
129
3376
48
7166
4477
1877

대전


11
2894
66
68
351
6607
57
6
249
8062
180
29657
13360
10
4976
2955
2
418
145
195
144
40
259

세종



b. 지역별 CCTV 설치 현황 시각화

CCTV 설치 구역을 시각화하여 마커로 표시하였음.

```
cctv_map = f.Map(location=[36.5847,126.8999],
    tiles='OpenStreetMap',
    zoom_start=10
)


for i in range(len(SJ_CCTV)):
    marker_cctv = f.Marker([SJ_CCTV.iloc[i]['위도'],SJ_CCTV.iloc[i]['경도']],
    popup = "{0}, {1}".format(SJ_CCTV.iloc[i]['도로노선명'],SJ_CCTV.iloc[i]['설치장소']),
    icon = f.Icon(color='red',icon='star'))

    marker_cctv.add_to(cctv_map)
cctv_map
```