📊 CSV 파일 그룹별 전처리 전략

데이터 성격에 따라 **인구, 교통, 교통안전시설** 세 그룹으로 분류하여 데이터 정제 및 분석 기반을 마련합니다.

---

1. 데이터 그룹별 전처리 및 활용 방안

데이터 그룹	파일명	분석 활용 방안	전처리 전략
1. 인구 데이터	03_성연령별_거주인구 04_성연령별_유동인구 05_시간대별_직장인구 06_시간대별_방문인구 07_주중주말_서비스인구	교통약자 인구 규모 파악 및 교통안전 취약지역 식별	1. **결측치 처리**: 결측값은 0으로 대체 2. **타입 표준화**: `gid`는 문자열, 인구 컬럼은 숫자형 통일 3. **파생변수**: '어린이', '노인', '총 교통약자' 변수 생성 4. **데이터 통합**: `gid` 기준으로 모든 인구 데이터 병합
2. 교통 데이터	09_평균속도 10_추정교통량 11_혼잡빈도강도 12_혼잡시간강도	과속, 교통량, 상습 정체 현황 분석을 통한 사고 위험도 예측	1. **이상치 처리**: 누락/이상 데이터를 주변 평균값 대체 또는 제거 2. **파생변수**: '종합 혼잡도', '과속 위험 구간' 분류 변수 생성 3. **데이터 통합**: `link_id` 기준으로 교통 데이터 병합
3. 교통안전시설	14_어린이보호구역, 15_학교현황 16_유치원, 17_어린이집, 18_횡단보도 19_버스정류장, 20_CCTV, 21_방지턱	기존 안전 인프라 현황 파악 및 안전시설 사각지대 도출	1. **공간 변환**: 좌표 데이터를 `GeoDataFrame`으로 변환 2. **데이터 표준화**: 시설물 종류 범주화 3. **공간 조인(Spatial Join)**: 격자/도로 데이터와 결합하여 시설물 개수, 거리 변수 생성 4. **데이터 통합**: 시설물 데이터를 하나의 통합 테이블로 구성

---

2. 종합 및 후속 조치

1. **최종 데이터셋 구축**
   - 세 그룹의 데이터를 `gid`(격자) 또는 `link_id`(도로)를 Key로 사용하여 최종 통합합니다.
2. **공간 데이터 결합**
   - 사고 이력 등 `geojson` 파일을 최종 데이터셋과 공간 결합하여 사고 관련 종속 변수 및 독립 변수를 확충합니다.
3. **모델링 및 분석**
   - 머신러닝 모델을 활용해 **교통사고 위험지역을 예측**합니다.
   - 분석 결과를 바탕으로 **교통안전시설물 최적 설치 위치**를 도출합니다.

CSV 파일별 상세 전처리 전략

---

1. 인구 데이터

1.1. `03._성연령별_거주인구(격자).csv`

- **데이터 목적**: 특정 격자(`gid`) 내 상주하는 성별/연령별 인구 분포를 파악하여 교통약자(어린이, 노인)의 **정주 환경 취약성**을 분석합니다.
- **주요 컬럼**: `gid` (격자 ID), `std_ymd` (기준 연월일), `pop_0_9`, `pop_10_19`, ..., `pop_80_` (연령대별 인구)
- **전처리 전략**:
  - **결측치 처리**: 인구 데이터가 없는 `gid`는 분석에서 제외하거나, 해당 지역의 인구는 0으로 간주합니다.
  - **데이터 타입 변환**: `gid`는 문자열, 연령대별 `pop_` 컬럼은 정수(int) 또는 실수(float)로 변환합니다. `std_ymd`는 datetime 객체로 변환하여 시계열 분석에 대비합니다.
  - **파생변수 생성**:
    - `어린이_거주인구` : `pop_0_9` + `pop_10_19` (또는 실제 어린이 연령대에 해당하는 컬럼 합산)
    - `노인_거주인구` : `pop_60_69` + `pop_70_79` + `pop_80_` (또는 실제 노인 연령대에 해당하는 컬럼 합산)
    - `총_교통약자_거주인구` : `어린이_거주인구` + `노인_거주인구`
  - **데이터 통합**: `gid`를 키로 하여 다른 격자 기반 데이터와 병합할 준비를 합니다.

1.2. `04._성연령별_유동인구.csv`

- **데이터 목적**: 특정 격자(`gid`) 내 성별/연령별 유동인구(이동 인구) 분포를 파악하여 교통약자의 **이동 동선 및 취약 시간대**를 분석합니다.
- **주요 컬럼**: `gid`, `base_ymd` (기준 연월일), `time` (시간대), `resd_sex_cd` (성별), `resd_age_cd` (연령대), `pop` (유동인구)
- **전처리 전략**:
  - **결측치 처리**: 유동인구 데이터가 없는 경우 0으로 간주하거나, 해당 시간대의 평균값으로 대체(imputation)를 고려합니다.
  - **데이터 타입 변환**: `gid`는 문자열, `pop`은 정수(int) 또는 실수(float)로 변환합니다. `base_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**:
    - `어린이_유동인구` : `resd_age_cd`가 어린이 연령대에 해당하는 `pop` 합산
    - `노인_유동인구` : `resd_age_cd`가 노인 연령대에 해당하는 `pop` 합산
    - `총_교통약자_유동인구` : `어린이_유동인구` + `노인_유동인구`
    - 시간대별(`time`)로 유동인구의 변화를 분석하기 위한 집계(aggregation) 작업이 필요합니다.
  - **데이터 통합**: `gid`, `base_ymd`, `time`을 키로 하여 다른 시공간 데이터와 병합할 준비를 합니다.

1.3. `05._시간대별_직장인구.csv`

- **데이터 목적**: 특정 격자(`gid`) 내 시간대별 직장인구 분포를 파악하여 출퇴근 시간 등 **특정 시간대의 교통량 예측 및 혼잡도** 분석에 기여합니다.
- **주요 컬럼**: `gid`, `std_ymd` (기준 연월일), `time` (시간대), `job_pop` (직장인구)
- **전처리 전략**:
  - **결측치 처리**: 직장인구 데이터가 없는 경우 0으로 간주합니다.
  - **데이터 타입 변환**: `gid`는 문자열, `job_pop`은 정수(int) 또는 실수(float)로 변환합니다. `std_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**: 출퇴근 시간대(예: 07-09시, 17-19시)의 직장인구 합계 등 특정 시간대별 특성을 나타내는 변수를 생성할 수 있습니다.
  - **데이터 통합**: `gid`, `std_ymd`, `time`을 키로 하여 다른 시공간 데이터와 병합할 준비를 합니다.

1.4. `06._시간대별_방문인구.csv`

- **데이터 목적**: 특정 격자(`gid`) 내 시간대별 방문인구 분포를 파악하여 상업/공공시설 주변 등 **주요 방문지의 교통량 및 보행량 특성**을 분석합니다.
- **주요 컬럼**: `gid`, `base_ymd` (기준 연월일), `time` (시간대), `visit_pop` (방문인구)
- **전처리 전략**:
  - **결측치 처리**: 방문인구 데이터가 없는 경우 0으로 간주합니다.
  - **데이터 타입 변환**: `gid`는 문자열, `visit_pop`은 정수(int) 또는 실수(float)로 변환합니다. `base_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**: 주말/평일 및 특정 시간대(예: 점심시간, 저녁시간)의 방문인구 합계 등 방문 행태를 나타내는 변수를 생성할 수 있습니다.
  - **데이터 통합**: `gid`, `base_ymd`, `time`을 키로 하여 다른 시공간 데이터와 병합할 준비를 합니다.

1.5. `07._주중주말_서비스인구.csv`

- **데이터 목적**: 특정 격자(`gid`) 내 주중/주말 서비스인구 분포를 파악하여 요일별/주말별 **지역 활성화 정도 및 교통량 변화**를 분석합니다.
- **주요 컬럼**: `gid`, `dt` (기준 일자), `day_gb` (주중/주말 구분), `pop_cnt` (서비스인구)
- **전처리 전략**:
  - **결측치 처리**: 서비스인구 데이터가 없는 경우 0으로 간주합니다.
  - **데이터 타입 변환**: `gid`는 문자열, `pop_cnt`는 정수(int) 또는 실수(float)로 변환합니다. `dt`는 datetime 객체로 변환합니다. `day_gb`는 범주형(category)으로 변환합니다.
  - **파생변수 생성**: 주중 서비스인구와 주말 서비스인구의 차이 또는 비율을 계산하여 지역 특성을 파악할 수 있습니다.
  - **데이터 통합**: `gid`, `dt`, `day_gb`를 키로 하여 다른 시공간 데이터와 병합할 준비를 합니다.

2. 교통 데이터

2.1. `09._평균속도.csv`

- **데이터 목적**: 도로 링크(`link_id`)별 평균 통행 속도를 파악하여 **과속 위험 구간 또는 정체 발생 구간**을 식별하고, 교통사고 발생 가능성과 연관성을 분석합니다.
- **주요 컬럼**: `link_id` (도로 링크 ID), `std_ymd` (기준 연월일), `time` (시간대), `avg_spd` (평균 속도)
- **전처리 전략**:
  - **결측치 처리**: 평균 속도 데이터가 없는 `link_id`는 해당 시간대의 평균 속도로 대체하거나, 분석에서 제외합니다.
  - **이상치 처리**: 비정상적으로 높거나 낮은 속도(예: 0km/h 또는 제한속도를 크게 초과하는 속도)는 이상치로 판단하여 해당 `link_id`의 시간대별 평균값 등으로 보정하거나 제거합니다.
  - **데이터 타입 변환**: `link_id`는 문자열, `avg_spd`는 실수(float)로 변환합니다. `std_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**:
    - `과속_위험_여부`: 해당 도로의 제한 속도를 초과하는 경우 '1', 아니면 '0'으로 분류하는 이진 변수. (제한 속도 정보 필요)
    - `정체_여부`: 평균 속도가 일정 기준 이하인 경우 '1', 아니면 '0'으로 분류하는 이진 변수.
  - **데이터 통합**: `link_id`, `std_ymd`, `time`을 키로 하여 다른 도로 기반 시공간 데이터와 병합할 준비를 합니다.

2.2. `10._추정교통량.csv`

- **데이터 목적**: 도로 링크(`link_id`)별 추정 교통량을 파악하여 **도로의 혼잡 수준 및 교통 부하**를 분석하고, 교통사고 발생 가능성과의 관계를 분석합니다.
- **주요 컬럼**: `link_id`, `std_ymd`, `time`, `volume` (교통량)
- **전처리 전략**:
  - **결측치 처리**: 교통량 데이터가 없는 `link_id`는 해당 시간대의 평균 교통량으로 대체하거나, 0으로 간주합니다.
  - **이상치 처리**: 비정상적으로 높은 교통량은 이상치로 판단하여 해당 `link_id`의 시간대별 평균값 등으로 보정하거나 제거합니다.
  - **데이터 타입 변환**: `link_id`는 문자열, `volume`은 정수(int) 또는 실수(float)로 변환합니다. `std_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**: 출퇴근 시간대 등 특정 시간대의 교통량 합계 또는 평균을 계산하여 `link_id`별 교통 패턴을 분석합니다.
  - **데이터 통합**: `link_id`, `std_ymd`, `time`을 키로 하여 다른 도로 기반 시공간 데이터와 병합할 준비를 합니다.

2.3. `11._혼잡빈도강도.csv`

- **데이터 목적**: 도로 링크(`link_id`)별 혼잡 빈도 강도를 파악하여 **상습적인 정체 구역**을 식별하고, 사고 위험도에 미치는 영향을 분석합니다.
- **주요 컬럼**: `link_id`, `std_ymd`, `time`, `cf_intensity` (혼잡 빈도 강도)
- **전처리 전략**:
  - **결측치 처리**: 데이터가 없는 경우 0으로 간주하거나, 해당 `link_id`의 시간대별 평균값으로 대체합니다.
  - **데이터 타입 변환**: `link_id`는 문자열, `cf_intensity`는 실수(float)로 변환합니다. `std_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **데이터 통합**: `link_id`, `std_ymd`, `time`을 키로 하여 다른 도로 기반 시공간 데이터와 병합할 준비를 합니다.

2.4. `12._혼잡시간강도.csv`

- **데이터 목적**: 도로 링크(`link_id`)별 혼잡 시간 강도를 파악하여 **정체가 지속되는 구역**을 식별하고, 사고 위험도에 미치는 영향을 분석합니다.
- **주요 컬럼**: `link_id`, `std_ymd`, `time`, `ct_intensity` (혼잡 시간 강도)
- **전처리 전략**:
  - **결측치 처리**: 데이터가 없는 경우 0으로 간주하거나, 해당 `link_id`의 시간대별 평균값으로 대체합니다.
  - **데이터 타입 변환**: `link_id`는 문자열, `ct_intensity`는 실수(float)로 변환합니다. `std_ymd`는 datetime, `time`은 시간(time) 또는 정수(int)로 변환합니다.
  - **파생변수 생성**: `11._혼잡빈도강도.csv`와 결합하여 '종합 혼잡도' 지수를 생성할 수 있습니다.
  - **데이터 통합**: `link_id`, `std_ymd`, `time`을 키로 하여 다른 도로 기반 시공간 데이터와 병합할 준비를 합니다.

3. 교통안전시설 및 기타 시설 데이터

3.1. `14._어린이보호구역.csv`

- **데이터 목적**: 어린이보호구역의 위치와 범위를 파악하여 **어린이 교통사고 예방의 핵심 구역**을 식별합니다.
- **주요 컬럼**: `gid` (격자 ID), `std_ymd`, `sc_zone_nm` (어린이보호구역명), `sc_zone_cd` (어린이보호구역 코드), `sc_zone_cnt` (어린이보호구역 개수)
- **전처리 전략**:
  - **결측치 처리**: `gid`에 해당하는 어린이보호구역 정보가 없으면 해당 격자에는 보호구역이 없는 것으로 간주합니다.
  - **데이터 타입 변환**: `gid`는 문자열, `sc_zone_cnt`는 정수(int)로 변환합니다.
  - **파생변수 생성**: 해당 `gid`에 어린이보호구역이 존재하는지 여부를 나타내는 이진 변수(`is_school_zone`)를 생성합니다.
  - **데이터 통합**: `gid`를 키로 하여 격자 기반 데이터와 병합할 준비를 합니다.

3.2. `15._학교현황.csv`, `16._유치원현황.csv`, `17._어린이집현황.csv`

- **데이터 목적**: 학교, 유치원, 어린이집의 위치를 파악하여 **어린이 밀집 지역**을 식별하고, 스쿨존 및 통학로 지정의 근거 자료로 활용합니다.
- **주요 컬럼**: `sig_cd` (시군구 코드), `fac_nm` (시설명), `fac_type` (시설 유형), `lon` (경도), `lat` (위도)
- **전처리 전략**:
  - **결측치 처리**: 위치 정보(`lon`, `lat`)가 없는 시설은 분석에서 제외합니다.
  - **데이터 타입 변환**: `lon`, `lat`은 실수(float)로 변환합니다. `fac_nm`, `fac_type` 등은 문자열로 유지합니다.
  - **좌표 데이터 변환**: `lon`, `lat`을 사용하여 GeoDataFrame으로 변환합니다. (CRS 일치 확인)
  - **공간 조인**: 변환된 GeoDataFrame을 사용하여 각 시설물이 어떤 격자(`gid`)에 속하는지 또는 어떤 도로(`link_id`)와 가까운지 공간 조인합니다.
  - **파생변수 생성**: 각 격자(`gid`) 또는 도로(`link_id`)별 `학교_수`, `유치원_수`, `어린이집_수` 등의 집계 변수를 생성합니다.

3.3. `18._횡단보도_위치정보.csv`

- **데이터 목적**: 횡단보도의 위치를 파악하여 보행자 이동이 많은 지점 및 **횡단 중 사고 위험이 높은 지점**을 분석합니다.
- **주요 컬럼**: `x`, `y` (좌표), `road_nm` (도로명), `crosswalk_type` (횡단보도 종류)
- **전처리 전략**:
  - **결측치 처리**: 위치 정보(`x`, `y`)가 없는 횡단보도는 분석에서 제외합니다.
  - **데이터 타입 변환**: `x`, `y`는 실수(float)로 변환합니다.
  - **좌표 데이터 변환**: `x`, `y`를 사용하여 GeoDataFrame으로 변환합니다. (CRS 일치 확인)
  - **공간 조인**: 변환된 GeoDataFrame을 사용하여 각 횡단보도가 어떤 격자(`gid`)에 속하는지 또는 어떤 도로(`link_id`)에 위치하는지 공간 조인합니다.
  - **파생변수 생성**: 각 격자(`gid`) 또는 도로(`link_id`)별 `횡단보도_수`를 집계 변수로 생성합니다.

3.4. `19._버스정류장_위치정보.csv`

- **데이터 목적**: 버스정류장의 위치를 파악하여 대중교통 이용자의 밀집 지역 및 **승하차 관련 보행자 사고 위험 지점**을 분석합니다.
- **주요 컬럼**: `lon`, `lat` (경도, 위도), `bus_st_nm` (정류장명), `bus_st_id` (정류장 ID)
- **전처리 전략**:
  - **결측치 처리**: 위치 정보(`lon`, `lat`)가 없는 정류장은 분석에서 제외합니다.
  - **데이터 타입 변환**: `lon`, `lat`은 실수(float)로 변환합니다.
  - **좌표 데이터 변환**: `lon`, `lat`을 사용하여 GeoDataFrame으로 변환합니다. (CRS 일치 확인)
  - **공간 조인**: 변환된 GeoDataFrame을 사용하여 각 정류장이 어떤 격자(`gid`)에 속하는지 또는 어떤 도로(`link_id`)와 가까운지 공간 조인합니다.
  - **파생변수 생성**: 각 격자(`gid`) 또는 도로(`link_id`)별 `버스정류장_수`를 집계 변수로 생성합니다.

3.5. `20._CCTV_현황.csv`

- **데이터 목적**: CCTV의 설치 위치를 파악하여 **감시 사각지대를 식별하고, 사고 발생 시 증거 확보 및 예방 효과**를 분석합니다.
- **주요 컬럼**: `lon`, `lat` (경도, 위도), `cctv_type` (CCTV 종류), `inst_purp` (설치 목적)
- **전처리 전략**:
  - **결측치 처리**: 위치 정보(`lon`, `lat`)가 없는 CCTV는 분석에서 제외합니다.
  - **데이터 타입 변환**: `lon`, `lat`은 실수(float)로 변환합니다.
  - **좌표 데이터 변환**: `lon`, `lat`을 사용하여 GeoDataFrame으로 변환합니다. (CRS 일치 확인)
  - **공간 조인**: 변환된 GeoDataFrame을 사용하여 각 CCTV가 어떤 격자(`gid`)에 속하는지 또는 어떤 도로(`link_id`)와 가까운지 공간 조인합니다.
  - **파생변수 생성**: 각 격자(`gid`) 또는 도로(`link_id`)별 `CCTV_수`를 집계 변수로 생성합니다. 설치 목적에 따른 분류 변수도 유용합니다.

3.6. `21._과속방지턱_현황.csv`

- **데이터 목적**: 과속방지턱의 설치 위치를 파악하여 **차량 속도 저감 시설의 분포**를 분석하고, 사고 발생과의 연관성을 분석합니다.
- **주요 컬럼**: `x`, `y` (좌표), `road_type` (도로 종류), `speed_hump_type` (과속방지턱 종류)
- **전처리 전략**:
  - **결측치 처리**: 위치 정보(`x`, `y`)가 없는 과속방지턱은 분석에서 제외합니다.
  - **데이터 타입 변환**: `x`, `y`는 실수(float)로 변환합니다.
  - **좌표 데이터 변환**: `x`, `y`를 사용하여 GeoDataFrame으로 변환합니다. (CRS 일치 확인)
  - **공간 조인**: 변환된 GeoDataFrame을 사용하여 각 과속방지턱이 어떤 격자(`gid`)에 속하는지 또는 어떤 도로(`link_id`)에 위치하는지 공간 조인합니다.
  - **파생변수 생성**: 각 격자(`gid`) 또는 도로(`link_id`)별 `과속방지턱_수`를 집계 변수로 생성합니다.

종합 및 후속 조치

각 파일별 전처리 전략을 기반으로 다음과 같은 단계를 수행합니다:

1. **데이터 클리닝**: 각 파일의 결측치 및 이상치를 처리하고 데이터 타입을 적절히 변환합니다.
2. **피처 엔지니어링**: 각 파일에서 분석 목적에 맞는 파생 변수를 생성합니다.
3. **공간 데이터 통합**:
   - 모든 시설물 위치 정보(`lon`, `lat`, `x`, `y` 컬럼을 포함하는 파일들)를 GeoDataFrame으로 변환하고, 좌표계(CRS)를 통일합니다.
   - 이를 `01._격자_(4개_시·구).geojson`, `02._격자_(하남교산).geojson`, `08.상세도로망_네트워크.geojson`, `13._교통사고이력.geojson` 등의 공간 데이터와 공간적으로 조인하여 각 격자 및 도로 링크에 해당하는 통계적, 공간적 피처를 통합합니다.
4. **최종 데이터셋 구축**: `gid` (격자 ID) 또는 `link_id` (도로 링크 ID)를 기준으로 모든 전처리된 데이터를 병합하여 최종 분석 데이터셋을 구축합니다. 이 데이터셋은 각 격자 또는 도로 링크별 인구, 교통량, 시설물 현황, 사고 이력 등의 정보를 모두 포함하게 됩니다.
5. **모델링**: 최종 데이터셋을 사용하여 교통사고 위험도 예측 모델을 개발하고, 스마트 안전 교통시설물의 최적 설치 위치를 제안합니다.