T-AVE 데이터 분석 심화 프로젝트 토담팀

데이콘 대구 교통사고 피해 예측 AI 경진대회

김소현 박준영 이다희 엄성원

목차

• 프로젝트 개요

대구 교통사고 피해 예측 AI 경진대회

- 대회 주제 : 시공간 정보로부터 사고위험도(ECLO) 예측 AI 모델 개발
  * ECLO(인명피해 심각도) = 사망자수 * 10 + 중상자수 * 5 + 경상자수 * 3 + 부상자수 * 1

- 대회 설명 : 사고 발생시간, 공간 등의 정보를 활용하여 사고 위험도를 예측하는 AI 알고리즘 개발

- 심사기준 : RMSLE(Root Mean Squared Logarithmic Error) of ECLO의 최솟값

2. 데이터 소개

내부 데이터

- train.csv
  - ID : 대구에서 발생한 교통사고의 고유 ID
  - 2019년부터 2021년까지의 교통사고 데이터로 구성
  - 해당 사고가 발생한 당시의 시공간 정보(사고일시, 요일, 기상상태..)와 사고 관련 정보(사고유형, 법규위반, 가해운전자 연령...) 포함
  - ECLO : 인명피해 심각도

- test.csv
  - ID : 대구에서 발생한 교통사고의 고유 ID
  - 2022년도의 교통사고 데이터로 구성
  - 추론 시점에서 획득할 수 있는 정보로 구성(사고일시, 요일, 기상상태, 사고유형..)

- train에는 있지만 test에 없는 컬럼이 13개

- sample_submission.csv [제출 양식]
  - ID : 추론 샘플의 고유 ID
  - ECLO : 예측한 인명피해 심각도

- 그 외 데이터
  - countrywikde_accident.csv : 대구를 제외한 전국에서 발생한 교통사고 데이터
  - 대구 보안등 정보.csv
  - 대구 어린이 보호 구역 정보.csv
  - 대구 주차장 정보.csv
  - 대구 CCTV 정보.csv

추가로 대구 빅데이터활용센터, 한국자동차연구원 자동차데이터 포털, 공공데이터포털의 외부데이터 사용가능

2. 데이터소개 - 전처리

1. 가해운전자 연령
   - 법적으로 면허취득나이인 18세를 기준으로 데이터 제거하였더니 낮은 성능을 보여 10세 기준으로 제거

2. 컬럼 세분화
   - 정규표현식을 통해 시군구 컬럼을 세 개의 컬럼(시/군/구)으로 분할
   - 도로형태를 도로형태1 과 도로형태2로 세분화 (ex. 도로형태 : 단일로-기타 > 도로형태1 : 단일로 / 도로형태2 : 기타)
   - 사고일시 컬럼을 datetime 전환 및 네 개의 컬럼(year/month/day/hour)으로 분할

3. 범주형 변수 내 unique 개수 비교
   - unique 함수를 통해 범주형 컬럼 내 test에 존재하지 않는 object 데이터는 결측값 처리 후 최빈값으로 대체

4. 파생변수 추가
   - 휴일여부 : 사고일시를 통해 휴일여부 추가
   - 시간대 : 사고일시를 통해 시간대를 네개의 범주(출근시간/낮시간/퇴근시간/새벽시간)로 변환
   - 계절 : month 컬럼을 통해 네개의 범주(봄/여름/가을/겨울)

5. 대구 주차장 정보 외부데이터 결합
   - 동별 groupby 후 급지구분(1,2,3) 컬럼 에 대해 one hot encoding 하여 동별 급지구분 개수 나타낸 후, 결측값은 0 대체

6. test 컬럼과 일치

최종컬럼

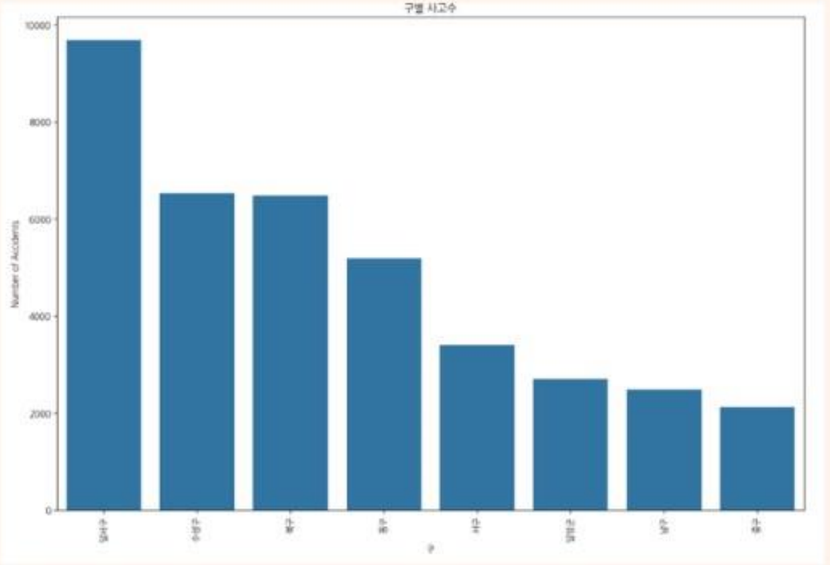
요일	기상상태	노면상태	사고유형	ECLO	시	군	구	도로형태1	도로형태2	month	day	hour	시간대	휴일여부	계절	급지구분1	급지구분2	급지구분3

2. 데이터소개 - EDA

내부데이터 시각화

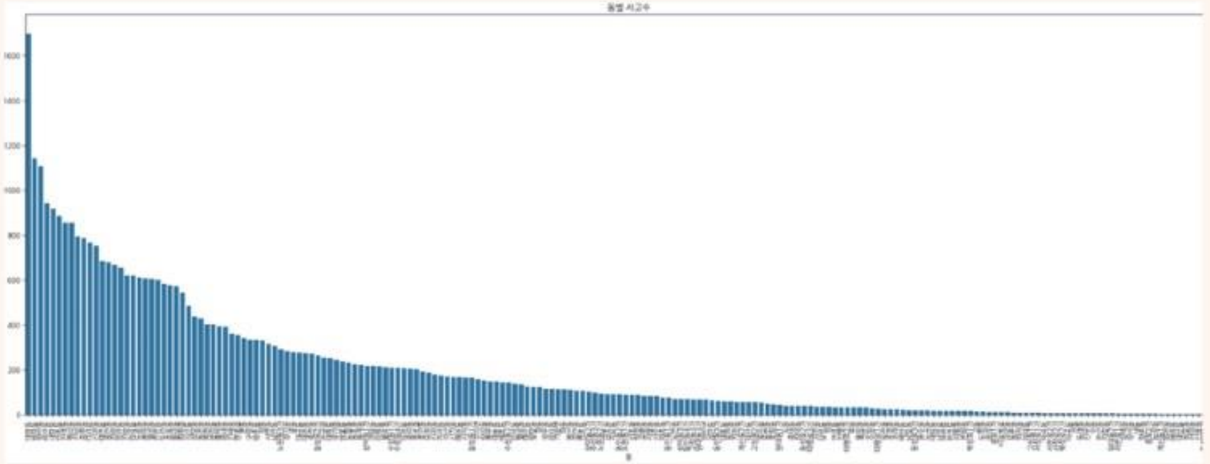
(1) '구'별 사고수 막대그래프

- 달서구에서 발생이 가장 높았음



(2) '동'별 사고수 막대그래프

- 다른 동에 비해 대명동에서 사고수 발생이 특히 높은 것을 확인

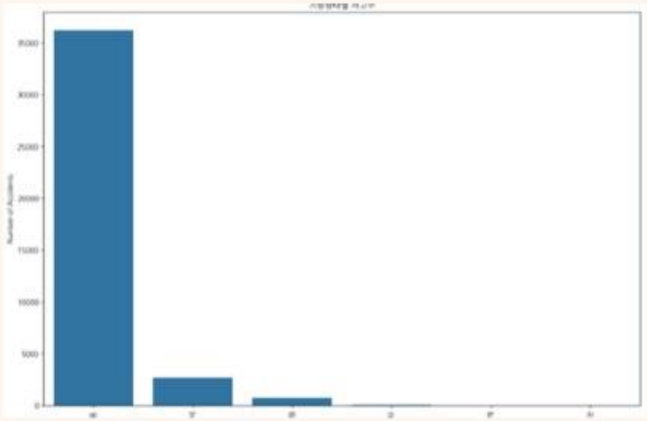


2. 데이터소개 - EDA

내부데이터 시각화

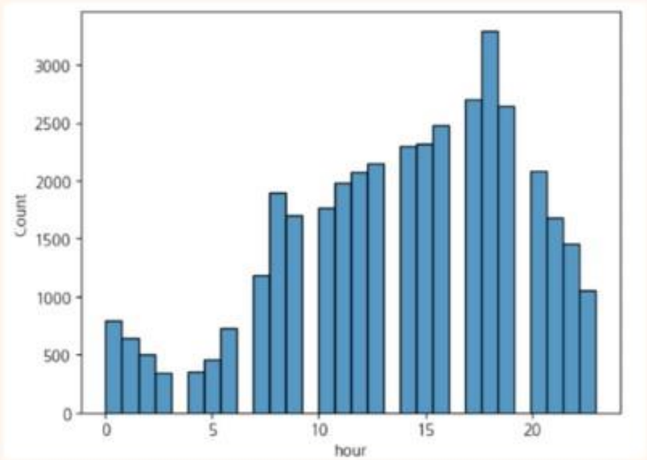
(3) 기상상태별 사고수 막대그래프

-날씨가 맑음일 때 압도적으로 사고수가 많았음



(4) 사고일시 분포 시각화

- 사고일시를 년/월/일/시로 나누어 분포 확인
- 출퇴근 시간대/새벽 시간대에 사고 수가 증가하는 것을 확인한 뒤 시간대를 4개로 나누어 라벨링



3. 중간 컨퍼런스

중간 모델링

- 불필요한 열 제거, year는 다중공선성 분석을 통해 제거함 (ID, 도시, 사고일시, year)
- 범주형 변수 -> LabelEncoding을 통해 변환
- train/test set 8 : 2
- autoML을 통해 모델 학습 및 예측 수행

```
from supervised.automl import AutoML
automl = AutoML(mode="Compete",
                algorithms = [ 'Random Forest', 'LightGBM', 'Xgboost'],
                n_jobs = -1,total_time_limit=2400, eval_metric="rmse", ml_task = "regression")
```

private score 기준 상위 10% 랭크

#	팀	팀 멤버	최종점수	제출수
90	해파리냉채	해파 공덕 so 성원	0.42741	16

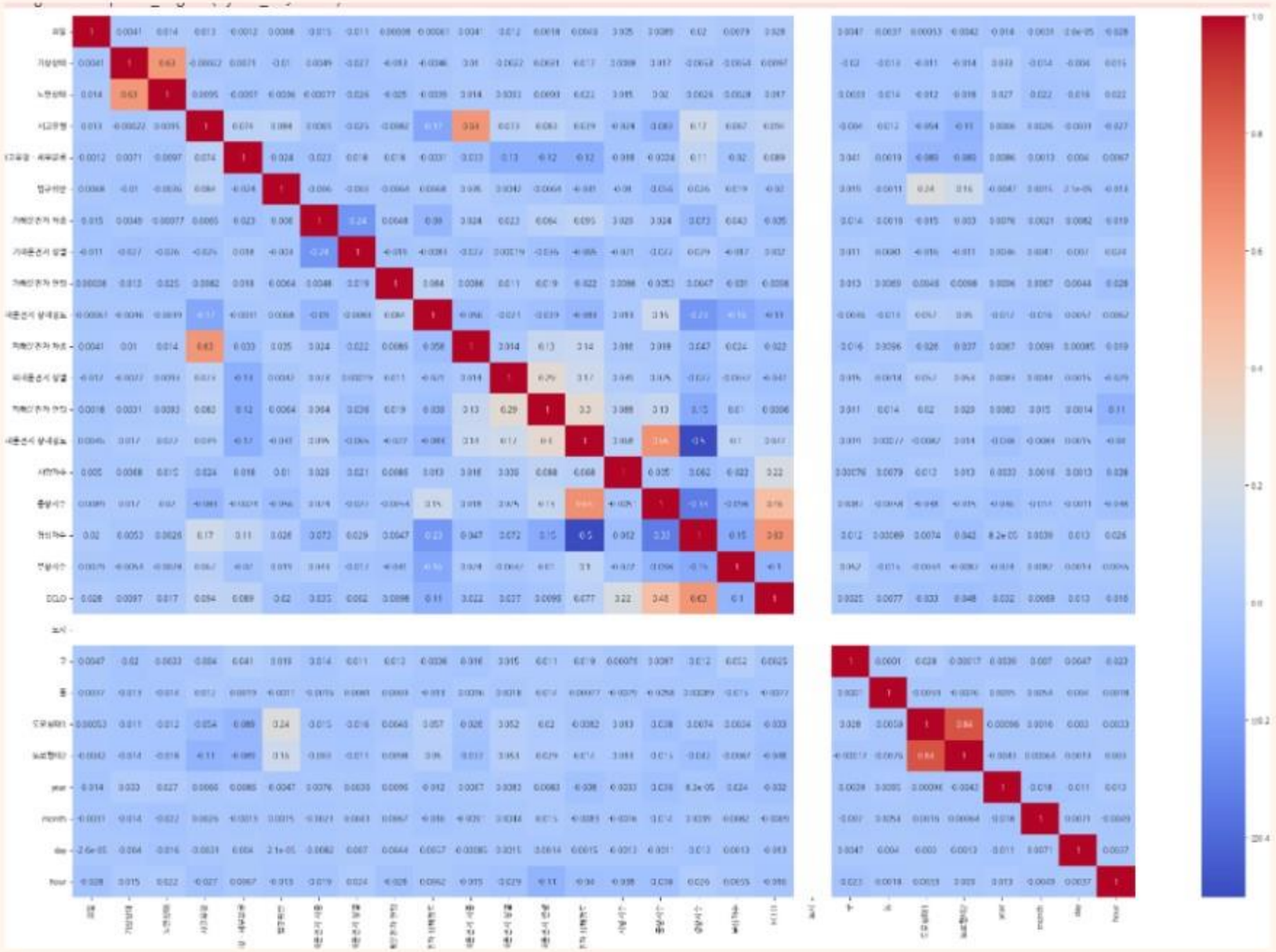
3. 중간 컨퍼런스

결과평가 및 피드백

- 준수한 결과가 나왔지만 AutoML을 활용했기 때문에 다시 모델링 진행

- 성능 향상을 위해 외부 데이터 추가 필요성
  - 시공간 정보로 사고위험도를 예측하기 때문에 시공간 정보가 충분히 확보되어야 성능 향상을 기대할 수 있을 것이라 판단
  - 어린이 보호 구역, cctv 등의 데이터와 ECLO간의 관계를 고려하여 추가

4. 최종 컨퍼런스



상관관계 점수가 너무 낮아 종속변수에
영향을 주는
독립변수를 알기 어려움
-> 더 많은 외부데이터 추가

4. 최종 컨퍼런스

외부 데이터 1. 보안등

보안등 위치, 개수 확인

*보안등은 어두운 곳에서 범죄나 사고가 일어나지 않기 위함, 가로등은 거리 조명, 안전, 미관을 위함,
기사에서는 가로등 = 보안등의 의미로 여김

"가로등 밝혀 교통사고·범죄 예방해요" 대구시 노후 가로등 LED로 교체

김근우 기자 lakehouse51@msnet.co.kr

매일신문 입력 2018-08-06 11:30:59 수정 2018-08-06 11:30:51

두 배 이상 밝고 에너지 소비량은 60% 줄어

[{IMG01}]

늦은 시간 밤거리에서 자주 일어나는 교통사고를 예방하고자 대구시가 팔을 걷어붙였다.

대구시설공단은 '교통사고 30% 줄이기 특별대책'의 일환으로 지난달 27일까지 지역 내 간선도로 5곳의 가로등을 모두 고효율 LE
교체했다고 밝혔다. 대상은 국도 5호선과 호국로, 태평로, 성서산단 과학단지 일대, 대곡택지와 동대구로 등 가로등이 낡아 어두운 탓에 사
잦다는 민원이 있었던 6곳이다.

공사를 통해 대상 구간에 지금까지 설치돼 있던 나트륨 전구가 모두 철거됐고, 대신 150W 고효율 LED 광원 1천562개가 새로 설치됐다.
나트륨 전구는 밝기가 15 lx(럭스)에 불과했지만, LED 광원은 2배 이상 밝은 30 lx의 빛을 낼 수 있다. 반대로 에너지 소비량은 60% 가
줄어들어 약 1억2천만원 가까운 비용 절감 효과를 볼 것으로 대구시는 예상했다.

김호경 대구시설공단 이사장은 "비효율적인 가로등 전구를 고효율 LED로 교체하는 작업을 지속해 밝고 안전한 야간 도로환경을 조성하는 ㅎ
교통사고 예방에도 일조해 공기업으로서 사회적 책임을 다할 것"이라고 말했다.

```
#보안등
light_df.head()
```

	보안등위치명	설치개수	소재지도로명주소	소재지지번주소	위도	경도	설치연도	설치형태
0	대명1동1	1	대구광역시 남구 현충로 155	대구광역시 남구 대명동 1722-4	35.846703	128.579803	2016.0	한전주
1	대명1동2	1	대구광역시 남구 현충로31길 10-1	대구광역시 남구 대명동 1723-11	35.846863	128.579406	2016.0	한전주
2	대명1동3	1	대구광역시 남구 현충로31길 9-8	대구광역시 남구 대명동 1722-56	35.846341	128.579342	2017.0	건축물
3	대명1동4	2	대구광역시 남구 현충로31길 19-11	대구광역시 남구 대명동 1722-41	35.846368	128.578922	2016.0	한전주
4	대명1동5	1	대구광역시 남구 현충로29길 22-2	대구광역시 남구 대명동 1721-22	35.845995	128.578858	2016.0	한전주

```
print(light_df.isnull().sum())
```

```
보안등위치명          0
설치개수            0
소재지도로명주소    37267
소재지지번주소          0
위도            28311
경도            28311
설치연도           42516
설치형태           36540
dtype: int64
```

4. 최종 컨퍼런스

외부 데이터 1. 보안등

```
light_df['위도'] = light_df['위도'].fillna(light_df['위도'].mean())
light_df['경도'] = light_df['경도'].fillna(light_df['경도'].mean())
light_df['설치연도'] = light_df['설치연도'].fillna(light_df['설치연도'].mode()[0])
light_df['설치형태'] = light_df['설치형태'].fillna(light_df['설치형태'].mode()[0])

location_pattern = r'(\S+) (\S+) (\S+) (\S+)'

light_df[['도시', '구', '동', '번지']] = light_df['소재지지번주소'].str.extract(location_pattern)

location_pattern = r'(\S+) (\S+) (\S+)'
light_df.loc[light_df['도시'].isna(), '도시'] = light_df.loc[light_df['도시'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,0]
light_df.loc[light_df['구'].isna(), '구'] = light_df.loc[light_df['구'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,1]
light_df.loc[light_df['동'].isna(), '동'] = light_df.loc[light_df['동'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,2]
light_df.isnull().sum()
```

```
보안등위치명          0
설치개수            0
소재지도로명주소     37267
소재지지번주소          0
위도              0
경도              0
설치연도             0
설치형태             0
도시              46
구               46
동               46
번지             938
dtype: int64
```

```
light_df['동'].mode()[0]
```

```
'대명동'
```

위도,경도에 대해 평균으로 결측값 대체
설치연도, 설치형태는 최빈값 대체
도시,구,동으로 split 후 동은 최빈값 대체

```
import re
location_pattern = r'(\S+) (\S+)'
light_df.loc[light_df['도시'].isna(), '도시'] = light_df.loc[light_df['도시'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,0]
light_df.loc[light_df['구'].isna(), '구'] = light_df.loc[light_df['구'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,1]
light_df['동'] = light_df['동'].fillna("대명동") #mode변환
light_df.loc[light_df['동'].str.contains("-"), '동'] = light_df.loc[light_df['동'].str.contains("-"), '동'].apply(lambda x : re.sub(r'\d+-\d+', '', x
light_df.loc[light_df['동'].str.contains("동"), '동'] = light_df.loc[light_df['동'].str.contains("동"), '동'].apply(lambda x : re.sub(r'\d\d+', '', x)
light_df.loc[light_df['동'] == '', '동'] = "대명동"
```

```
light_df['동'].value_counts()
```

```
대명동         5300
신암동         2001
남산동         1474
봉덕동         1396
송현동         1372
고산2동        1322
두류동         1292
논공읍         1153
상중이동        1149
가창면         1123
```

4. 최종 컨퍼런스

외부 데이터 1. 보안등

```
light_df = light_df.drop(columns=['소재지지번주소', '번지'])

light_df['설치형태'] = light_df['설치형태'].fillna(light_df['설치형태'].mode()[0])

#보안등 동이름 그룹핑
light_df.loc[light_df['동']=="내당1동", "동"] = "내당동"
light_df.loc[light_df['동']=="내당2·3동", "동"] = "내당동"
light_df.loc[light_df['동']=="내당4동", "동"] = "내당동"
```

groupby 이용하여 새로운 데이터프레임 생성
동이름 통일, 동별 설치갯수합, 설치개수 평균, 보안등 설치 평균 위도, 최저 위도, 최고 위도, 평균 경도, 최저 경도, 최고 경도, 평균 설치 연도

```
light_df_1 = light_df[['도시', '구', '동', '설치개수']].groupby(['도시', '구', '동']).sum().reset_index() #동별 설치갯수합
light_df_1.reset_index(inplace=True, drop=True)
light_df_1.rename(columns = {'설치개수':'보안등_설치총개수'}, inplace = True)

light_df_2 = light_df[['도시', '구', '동', '설치개수']].groupby(['도시', '구', '동']).mean().reset_index() #동별 설치갯수 평균
light_df_2.reset_index(inplace=True, drop=True)
light_df_2.rename(columns = {'설치개수':'보안등_평균설치개수'}, inplace = True)
```

	도시	구	동	보안등_설치총개수	보안등_평균설치개수	보안등_평균위도	보안등_최저위도	보안등_최고위도	보안등_평균경도	보안등_최저경도	보안등_최고경도	보안등_평균설치연도	보안등_최소설치연도	보안등_최대설치연도	보안등_설치형태종류수
0	대구광역시	남구	대명동	5377	1.023606	35.843907	35.829145	35.859098	128.576537	128.555738	128.595553	2016.426613	2000.0	2018.0	3
1	대구광역시	남구	봉덕동	1424	1.020057	35.841527	35.829227	35.849837	128.598577	128.575675	128.606482	2016.406160	2016.0	2018.0	3
2	대구광역시	남구	이천동	556	1.009074	35.852193	35.846024	35.856282	128.600837	128.593931	128.606152	2016.308530	2016.0	2018.0	3
3	대구광역시	달서구	갈산동	351	1.000000	35.842976	35.835969	35.849583	128.503158	128.492748	128.513086	2000.000000	2000.0	2000.0	1
4	대구광역시	달서구	감삼동	941	1.000000	35.849914	35.843671	35.856349	128.542153	128.530883	128.548359	1999.890542	1982.0	2016.0	1

4. 최종 컨퍼런스

외부 데이터 2. 어린이 보호 구역

cctv 설치 대수 결측값 -> 아예 없는것으로 판단하여 0으로 대체,

위도 경도 -> 평균값 대체/ 보호구역 ,도로폭, 구, 동 -> 최빈값 대체

대구 어린이보호구역 교통사고 전국 최저…5년간 사망사고 없어

(대구=뉴스1) 이재춘 기자 | 2020-10-15 14:44 송고

지난 7월9일 대구 중구 한 초등학교 앞 도로에 가로 130mm, 세로 200mm 크기의 형광 고휘도 반사지 시트로 만들어진 스쿨존(어린이보호구역) 주정차금지 알림판이 설치돼 있다. 대구경찰청은 스쿨존 내 불법주차로 인한 어린이 교통사고를 예방하기 위해 불법주차 취약 스쿨존에 주차금지 알림판 500개를 설치하는 '스쿨존 포인트존'을 운영한다. 2020.7.1/뉴스1 © News1 공정식 기자

대구의 어린이 보호구역 내 교통사고 발생률이 전국 7대 특·광역시 중 가장 낮은 것으로 나타났다.

```
#어린이 보호구역
kids_zone.head()
```

	시설종류	대상시설명	소재지도로명주소	소재지지번주소	위도	경도	관리기관명	관할경찰서명	CCTV설치여부	CCTV설치대수	보호구역도로폭	데이터기준일자
0	초등학교	남도초등학교	대구광역시 남구 현충동길 74(대명동)	대구광역시 남구 대명동 1709	35.845027	128.581402	대구광역시	남부경찰서	Y	7.0	6~8	2020-03-23
1	초등학교	영선초등학교	대구광역시 남구 영선길96(이천동)	대구광역시 남구 이천동 477	35.852901	128.596014	대구광역시	남부경찰서	Y	8.0	6~10	2020-03-23
2	초등학교	성명초등학교	대구광역시 남구 성당로 30길 55(대명동)	대구광역시 남구 대명동 3050	35.845152	128.570825	대구광역시	남부경찰서	Y	14.0	8~12	2020-03-23
3	초등학교	남덕초등학교	대구광역시 남구 앞산순환로 93길 33	대구광역시 남구 대명동 531-1	35.833042	128.573949	대구광역시	남부경찰서	Y	6.0	6~8	2020-03-23
4	초등학교	대명초등학교	대구광역시 남구 대명로 110	대구광역시 남구 대명동 960	35.838869	128.568889	대구광역시	남부경찰서	Y	5.0	6~10	2020-03-23

+ 코드 + 텍스트

```
kids_zone.isnull().sum()
```

```
시설종류          0
대상시설명         0
소재지도로명주소      0
소재지지번주소      85
위도             0
경도             0
관리기관명         0
관할경찰서명        0
CCTV설치여부       0
CCTV설치대수      175
보호구역도로폭     320
데이터기준일자       0
dtype: int64
```

```
kids_zone.dropna(subset = ['소재지지번주소'], inplace = True)
kids_zone['cnt'] = 1
kids_zone['CCTV설치대수'] = kids_zone['CCTV설치대수'].fillna(0)
kids_zone['위도'] = kids_zone['위도'].fillna(kids_zone['위도'].mean())
kids_zone['경도'] = kids_zone['경도'].fillna(kids_zone['경도'].mean())
kids_zone['보호구역도로폭'] = kids_zone['보호구역도로폭'].fillna(kids_zone['보호구역도로폭'].mode()[0])

location_pattern = r'(\S+) (\S+) (\S+) (\S+)'

kids_zone[['도시', '구', '동', '번지']] = kids_zone['소재지지번주소'].str.extract(location_pattern)
```

```
kids_zone['동'].mode()
```

```
0    다사읍
dtype: object
```

4. 최종 컨퍼런스

외부 데이터 2. 어린이 보호 구역

불필요한 칼럼 (‘소재지지번주소', ‘번지') 삭제,
도로폭_min 과 도로폭_max 정보 이용하여 도로폭의 평균값을 새로운 열로 추가
어린이 보호 구역 동이름 재지정
groupby 로 동별 평균, 최소, 최대 컬럼 나타냄

```
kids_zone.loc[kids_zone['동']=="옥포면", "동"] = "옥포읍"
kids_zone.loc[kids_zone['동']=="현풍면", "동"] = "현풍읍"

kids_zone_1 = kids_zone[['도시', '구', '동', 'CCTV설치대수']].groupby(['도시', '구', '동']).sum().reset_index()
kids_zone_1.reset_index(inplace=True, drop=True)
kids_zone_1.rename(columns = {'CCTV설치대수':'어린이보호구역_CCTV총설치대수'}, inplace = True)
```

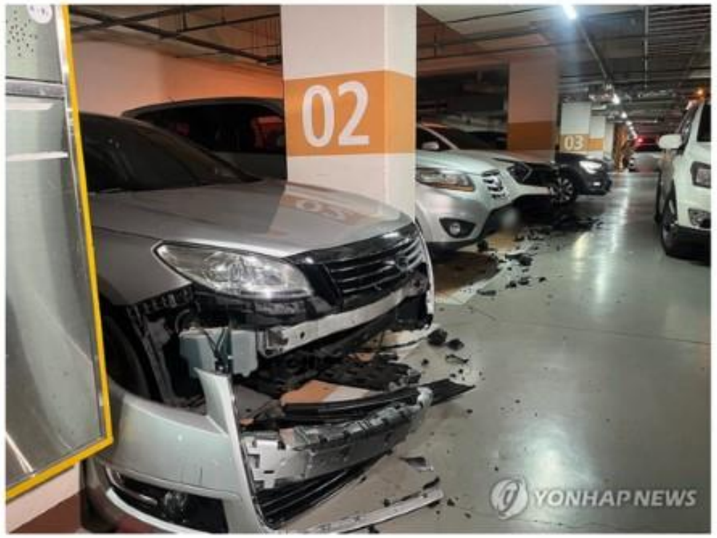
	도시	구	동	어린이보호구역_CCTV총설치대수	어린이보호구역_CCTV평균설치대수	어린이보호구역_CCTV최소설치대수	어린이보호구역_CCTV최대설치대수	어린이보호구역_최저위도	어린이보호구역_평균위도	어린이보호구역_최고위도	어린이보호구역_최저경도	어린이보호구역_평균경도	어린이보호구역_최고경도	어린이보호구역_평균최소도로폭	어린이보호구역_평균도로폭	어린이보호구역_평균최대도로폭	어린이보호구역_총개수
0	대구광역시	남구	대명동	106.0	4.076923	1.0	14.0	35.830432	35.844117	35.856459	128.560373	128.576188	128.591266	7.615385	9.057692	10.500000	26
1	대구광역시	남구	봉덕동	34.0	4.250000	1.0	8.0	35.834648	35.839129	35.844351	128.591000	128.597814	128.601102	16.500000	18.250000	20.000000	8
2	대구광역시	남구	이천동	22.0	3.666667	1.0	8.0	35.848849	35.851811	35.853598	128.596014	128.601901	128.605307	7.666667	9.166667	10.666667	6
3	대구광역시	달성군	가창면	0.0	0.000000	0.0	0.0	35.771063	35.787813	35.804564	128.621793	128.635020	128.648248	10.000000	10.000000	10.000000	8

4. 최종 컨퍼런스

외부 데이터 3. 주차장

아파트 지하주차장서 차량 15대 들이받고 잠적… 경찰 수사

송고시간 | 2023-11-29 10:50



부서진 차량

(대구=연합뉴스) 황수빈 기자 = 29일 대구 북구 한 아파트 지하 주차장에 차들이 파손된 채 주차돼있다.
대구 북부경찰서에 따르면 이날 0시 10분께 한 차주가 차량 15대를 들이받은 후 차를 놔두고 현장을 떠났다. 2023.11.29
hsb@yna.co.kr

(대구=연합뉴스) 황수빈 기자 = 대구의 한 아파트 지하 주차장에서 주차된 차량 15대를 들이받은 뒤 운전자가 자동차를 놔두고 도망치는 사건이 발생해 경찰이 수사에 나섰다.

대구 북부경찰서에 따르면 29일 0시 10분께 북구 칠성동 한 아파트 관리사무소로부터 "누군가 차를 들이받고 도망갔다"는 신고가 접수됐다.

위도, 경도 -> 평균값 대체, 동이름 통일
groupby 이용하여 컬럼별 평균, 최대, 최소 수치 나타냄

```
#주차장 데이터

parking_df.dropna(subset = ['소재지지번주소'], inplace = True)
parking_df = pd.get_dummies(parking_df, columns=['급지구분'])
parking_df['cnt'] = 1

parking_df['위도'] = parking_df['위도'].fillna(parking_df['위도'].mean())
parking_df['경도'] = parking_df['경도'].fillna(parking_df['경도'].mean())

location_pattern = r'(\S+) (\S+) (\S+) (\S+)'

parking_df[['도시', '구', '동', '번지']] = parking_df['소재지지번주소'].str.extract(location_pattern)

location_pattern = r'(\S+) (\S+) (\S+)'

parking_df.loc[parking_df['도시'].isna(), '도시'] = parking_df.loc[parking_df['도시'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,0]
parking_df.loc[parking_df['구'].isna(), '구'] = parking_df.loc[parking_df['구'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,1]
parking_df.loc[parking_df['동'].isna(), '동'] = parking_df.loc[parking_df['동'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,2]

parking_df.loc[parking_df['동'].str.contains("-"), '동'] = parking_df.loc[parking_df['동'].str.contains("-"), '동'].apply(lambda x : re.sub(r'\d+-\d+', '', 
```

```
#주차장 동이름 재지정
parking_df.loc[parking_df['동']=="옥포면", "동"] = "옥포읍"
parking_df.loc[parking_df['동']=="현풍면", "동"] = "현풍읍"
parking_df.loc[parking_df['동']=="내당4동", "동"] = "내당동"
parking_df.loc[parking_df['동']=="유곡리", "동"] = "유가읍"

parking_df_1 = parking_df[['도시', '구', '동', '주차구획수']].groupby(['도시', '구', '동']).mean().reset_index()
parking_df_1.reset_index(inplace=True, drop=True)
parking_df_1.rename(columns = {'주차구획수':'주차장_평균주차구획수'}, inplace = True)

parking_df_2 = parking_df[['도시', '구', '동', '주차구획수']].groupby(['도시', '구', '동']).min().reset_index()
parking_df_2.reset_index(inplace=True, drop=True)
parking_df_2.rename(columns = {'주차구획수':'주차장_최소주차구획수'}, inplace = True)
```

4. 최종 컨퍼런스

외부 데이터 4. CCTV

설치년도 최빈값 대체,동이름 통일
groupby 진행

대구시, 무인단속 CCTV 설치 후 교통사고 30% 감소

최태욱 / 기사승인 : 2018-10-10 16:45:34

대구시가 무인단속용 카메라를 설치하면서 교통사고가 줄어든 것으로 나타났다.

```
ctv_df.dropna(subset = ['소재지지번주소'], inplace = True)
ctv_df['설치연도'] = cctv_df['설치연도'].fillna(cctv_df['설치연도'].mode()[0])
ctv_df = pd.get_dummies(cctv_df, columns=['단속구분'])
ctv_df['cnt'] = 1

ocation_pattern = r'(\S+) (\S+) (\S+) (\S+)'

ctv_df[['도시', '구', '동', '번지']] = cctv_df['소재지지번주소'].str.extract(location_pattern)

ocation_pattern = r'(\S+) (\S+) (\S+)'

ctv_df.loc[cctv_df['도시'].isna(), '도시'] = cctv_df.loc[cctv_df['도시'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,0]
ctv_df.loc[cctv_df['구'].isna(), '구'] = cctv_df.loc[cctv_df['구'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,1]
ctv_df.loc[cctv_df['동'].isna(), '동'] = cctv_df.loc[cctv_df['동'].isna(), '소재지지번주소'].str.extract(location_pattern).iloc[:,2]

ctv_df.loc[cctv_df['동'].str.contains("-"), '동'] = cctv_df.loc[cctv_df['동'].str.contains("-"), '동'].apply(lambda x : re.sub(r'\d+-\d+', '', x))

ctv_df['도시'] = cctv_df['도시'].map({'대구':'대구광역시','대구광역시':'대구광역시'})

ctv_df.loc[cctv_df['구'] == "가창면", "구"] = "달성군"
ctv_df.loc[cctv_df['동'] == "삼산리", "동"] = "가창
ctv_df.loc[cctv_df['구'] == "다사읍", "구"] = "달성
ctv_df.loc[cctv_df['동'] == "세천리", "동"] = "다사

ctv_df = cctv_df.drop(columns=['소재지지번주소', '번
```

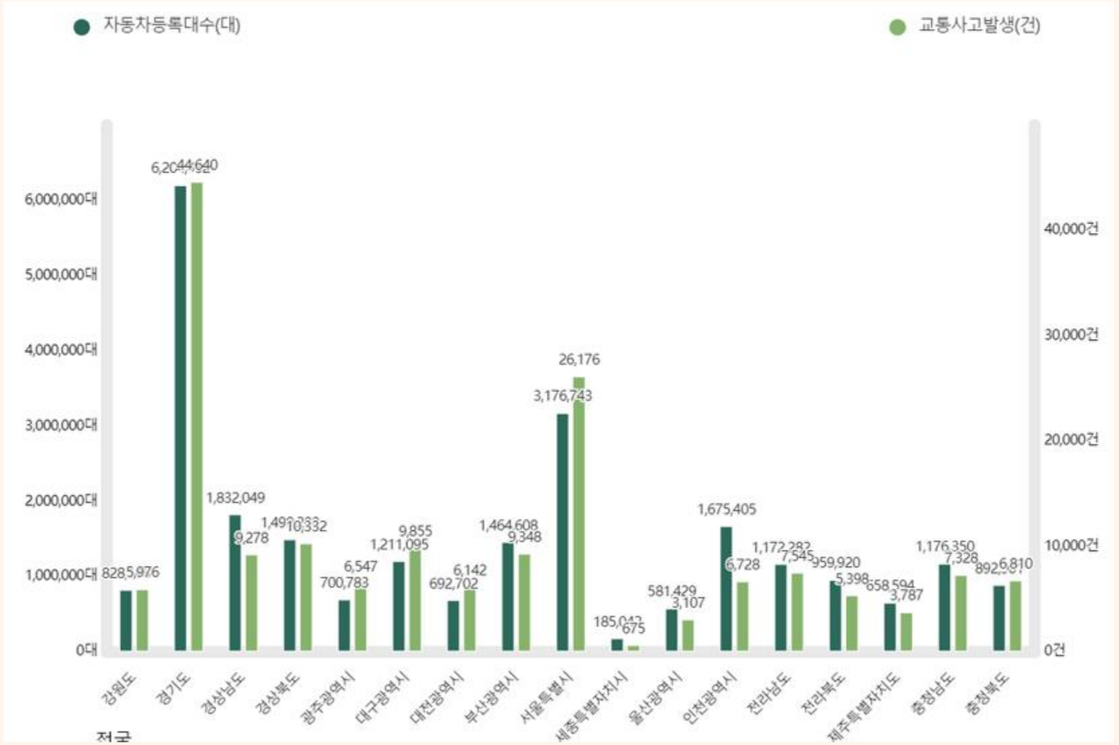
```
cctv_df.loc[cctv_df['동']=="평리2동", "동"] = "평리동"
cctv_df.loc[cctv_df['동']=="하리", "동"] = "논공읍"
cctv_df.loc[cctv_df['동']=="현풍면", "동"] = "현풍읍"

cctv_df_1 = cctv_df[['도시', '구', '동', '단속구분_1']].groupby(['도시', '구', '동']).sum().reset_index()
cctv_df_1.reset_index(inplace=True, drop=True)
cctv_df_1.rename(columns = {'단속구분_1':'cctv_총단속구분_1'}, inplace = True)
```

도시	구	동	cctv_총단속구분_1	cctv_총단속구분_2	cctv_총단속구분_4	cctv_총단속구분_99	cctv_최소제한속도	cctv_평균제한속도	cctv_최대제한속도	cctv_중간제한속도	cctv_최저위도	cctv_평균위도	cctv_최고위도	cctv_최저경도	cctv_평균경도	cctv_최고경도	cctv_평균설치연도	cctv_최소설치연도	cctv_최대설치연도	cctv_총개수
대구광역시	군위군	군위읍	1	2	0	0	30	63.333333	80	80	36.232137	36.234239	36.238444	128.564071	128.565307	128.567779	2019.333333	2017.0	2021.0	3
대구광역시	군위군	부계면	2	2	0	0	30	57.500000	70	65	36.048772	36.074368	36.101302	128.645694	128.655812	128.662532	2018.500000	2018.0	2020.0	4
대구광역시	군위군	삼국유사면	0	1	0	0	60	60.000000	60	60	36.130294	36.130294	36.130294	128.751743	128.751743	128.751743	2021.000000	2021.0	2021.0	1

4. 최종 컨퍼런스

외부 데이터 5. 자동차 등록 대수

자동차 등록대수가 많을 수록 교통사고가 어느정도 비례할 것이라고 판단,
대구광역시 자동차 등록현황 데이터 추가




```
car_register = car_register.rename(columns = {'구군':'구','읍면동':'동'})
car_register.head()
```

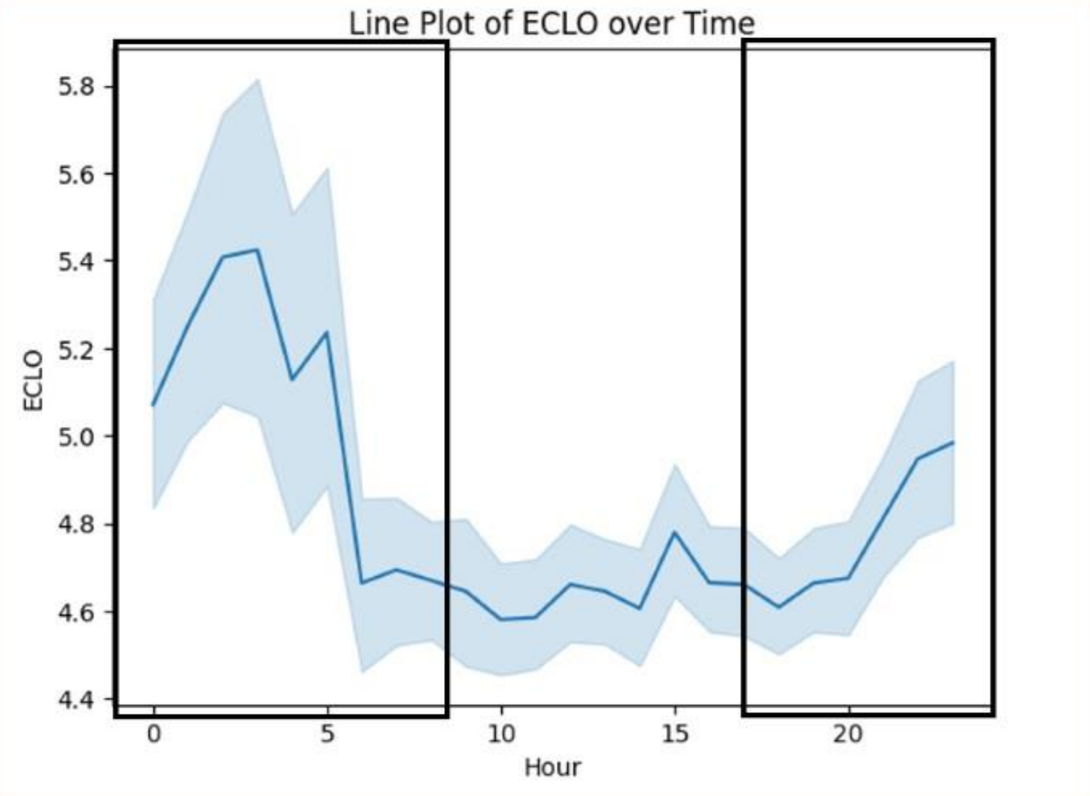
	구	동	승용	승합	화물	특수	소계
0	남구	대명동	29753	1244	4953	118	36068
1	남구	봉덕동	14627	418	1792	48	16885
2	남구	이천동	4207	116	504	14	4841
3	달서구	갈산동	1201	138	807	4	2150
4	달서구	감삼동	12204	289	1892	40	14425

```
train_d= train_org.copy()
test_d= test_org.copy()
```

4. 최종 컨퍼런스

외부 데이터 6. 일출, 일몰 시간

hour와 종속변수(ECLO) 와의 관계에서 밤, 새벽시간 일수록 ECLO가 높은 것을 보임 ->
햇빛 양과 교통사고가 상관있을거라고 판단,해가 떠있는 시간에 해당하면 1, 아니면 0으로 라벨링




```
sun['month'] = sun['일자'].apply(lambda x:sun_transform(x, 0))
sun['month'] = sun['month'].fillna(method='ffill')
sun['day'] = sun['일자'].apply(lambda x:day_transform(x, 1))
sun['일출 시'] = sun['일출'].apply(lambda x:sun_transform(x, 0))
sun['일출 분'] = sun['일출'].apply(lambda x:sun_transform(x, 1))
sun['일몰 시'] = sun['일몰'].apply(lambda x:sun_transform(x, 0))
sun['일몰 분'] = sun['일몰'].apply(lambda x:sun_transform(x, 1))
sun['일출'] = sun['일출 시'] + sun['일출 분']/60
sun['일몰'] = sun['일몰 시'] + sun['일몰 분']/60
sun = sun[['month','day','일출','일몰']]
sun.head()
```

	month	day	일출	일몰
1	1.0	1	7.783333	17.400000
2	1.0	2	7.783333	17.416667
3	1.0	3	7.783333	17.416667
4	1.0	4	7.783333	17.433333
5	1.0	5	7.783333	17.450000

4. 최종 컨퍼런스

전처리

- ‘ID’,’사고일시‘,’도시‘컬럼 drop
- 가해운전자 연령 -> 10세 미만 제거
- '보안등' , '어린이보호구역' ,'위도', '경도' , '도로폭' , '제한속도' ,'승용' ,'승합' , '화물' , '특수' , '소계' ' 연도' 를 포함하는 컬럼 -> null값에 대해 평균값 대체 (주로 연속형 변수)
- ‘구획수’,’급지구분‘,’총개수‘,’'단속구분' -> 결측값 0 대체
- 출퇴근 시간 라벨링 (5~11 : 출근시간, 11~17: 낮시간, 17~23: 퇴근시간, 나머지: 새벽시간)
- 국가지정 공휴일(대체공휴일 포함) -> 휴일/평일 라벨링
- 계절 파생변수 추가
- 2020년, 2021년 -> 코로나 변수 추가

4. 최종 컨퍼런스

최종 데이터 프레임

38585 X 84

	요일	기상상태	노면상태	사고유형	ECLO	구	동	도로형태1	도로형태2	year	month	day	hour	보안등_설치총개수	보안등_평균설치개수	보안등_평균위도	보안등_최저위도	보안등_최고위도	보안등_평균경도	보안등_최저경도	보안등_최고경도	보안등_평균설치연도	보안등_최소설치연도
0	6	2	0	0	5	7	39	2	5	2019	1	1	0	391.0	1.000000	35.867981	35.862999	35.876907	128.579156	128.573980	128.592846	2004.759591	2000.0
1	6	4	0	0	3	1	4	2	5	2019	1	1	0	941.0	1.000000	35.849914	35.843671	35.856349	128.542153	128.530883	128.548359	1999.890542	1982.0
2	6	2	0	0	3	6	64	2															
3	6	2	0	1	5	4	76	2															
4	6	2	0	1	3	3	124	2															

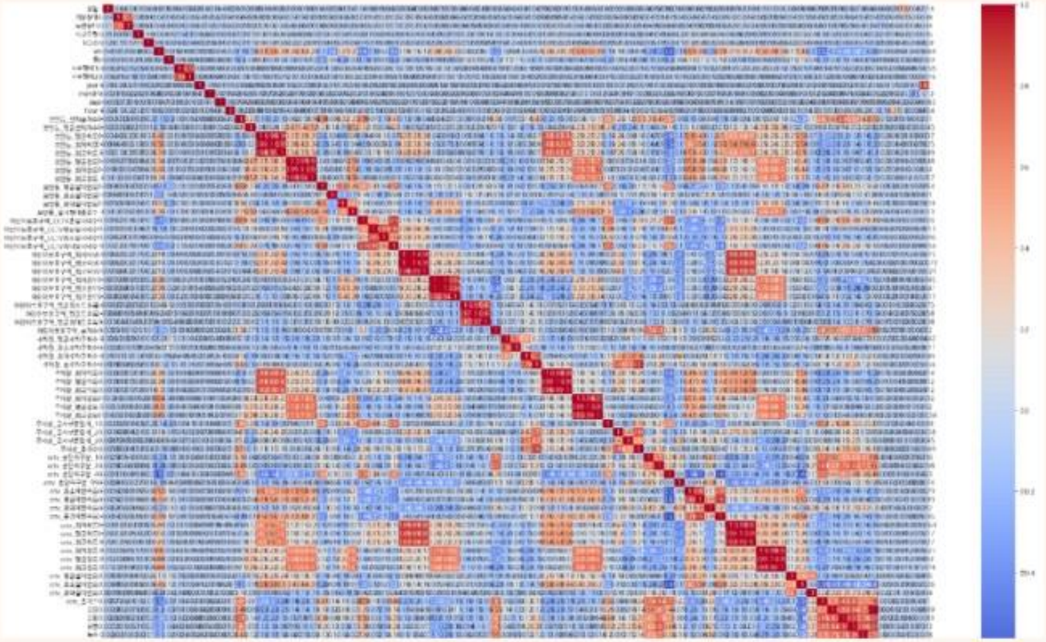
	ctv_중간제한속도	cctv_최저위도	cctv_평균위도	cctv_최고위도	cctv_최저경도	cctv_평균경도	cctv_최고경도	cctv_평균설치연도	cctv_최소설치연도	cctv_최대설치연도	cctv_총개수	승용	승합	화물	특수	소계	sun	rush	휴일여부	season	covid
	0.0	35.866283	35.868237	35.870355	128.578042	128.580886	128.582226	2013.000000	2006.0	2021.0	5.0	3004.0	86.0	626.0	7.0	3723.0	0	1	1	1	0
	0.0	35.844138	35.849099	35.854009	128.535164	128.540606	128.548048	2015.333333	2008.0	2021.0	12.0	12204.0	289.0	1892.0	40.0	14425.0	0	1	1	1	0
	55.0	35.829915	35.834183	35.837913	128.617301	128.621395	128.622898	2018.250000	2014.0	2021.0	4.0	6222.0	128.0	733.0	12.0	7095.0	0	1	1	1	0
	35.0	35.894814	35.899975	35.905147	128.613443	128.619733	128.624528	2018.300000	2014.0	2021.0	10.0	13894.0	293.0	1743.0	42.0	15972.0	0	1	1	1	0
	45.0	35.877328	35.883230	35.891982	128.612853	128.6															

	보안등_최대설치연도	보안등_설치형태종류수	어린이보호구역_CCTV총설치대수	어린이보호구역_CCTV평균설치대수	어린이보호구역_CCTV최소설치대수	어린이보호구역_CCTV최대설치대수	어린이보호구역_최저위도	어린이보호구역_평균위도	어린이보호구역_최고위도	어린이보호구역_최저경도	어린이보호구역_평균경도	어린이보호구역_최고경도	어린이보호구역_평균최소도로폭	어린이보호구역_평균도로폭	어린이보호구역_평균최대도로폭	어린이보호구역_총개수	주차장_평균주차구획수
	2017.0	3.0	13.000000	6.500000	4.000000	9.000000	35.866076	35.868541	35.871007	128.580485	128.581033	128.581581	5.500000	13.000000	20.500000	2.000000	45.454545
	2016.0	1.0	18.882136	1.763459	0.815359	3.518242	35.852758	35.859606	35.866710	128.573358	128.581492	128.592459	10.748096	11.092687	11.437277	13.814036	28.500000
	2019.0	3.0	0.000000	0.000000	0.000000	0.000000	35.824978	35.833939	35.838868	128.618451	128.621053	128.626288	10.000000	10.000000	10.000000	5.000000	0.000000
	2000.0	1.0	32.000000	2.909091	0.000000	5.000000	35.892028	35.897687	35.905993	128.617092	128.622803	128.630278	9.636364	9.636364	9.636364	11.000000	26.714286
	2000.0	3.0	18.882136	1.763459	0.815359	3.518242	35.852758	35.859606	35.866710	128.573358	128.581492	128.592459	10.748096	11.092687	11.437277	13.814036	63.000000

4. 최종 컨퍼런스

correlation 확인 및 다중공선성 분석

다중공선성 분석 후 가장 큰 3개 컬럼 drop 하여 예측 및 scoring 진행했
으나 성능이 더 안좋아져 컬럼 drop 추가로 진행하지 않음



```
from statsmodels.stats.outliers_influence import variance_inflation_factor
import pandas as pd
import numpy as np

df= train_df.copy()
numeric_cols = df.select_dtypes(include=[np.number]).columns
features = df[numeric_cols]
features = features.fillna(0)
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(features.values, i) for i in range(features.shape[1])]
vif["features"] = features.columns

print(vif)
```

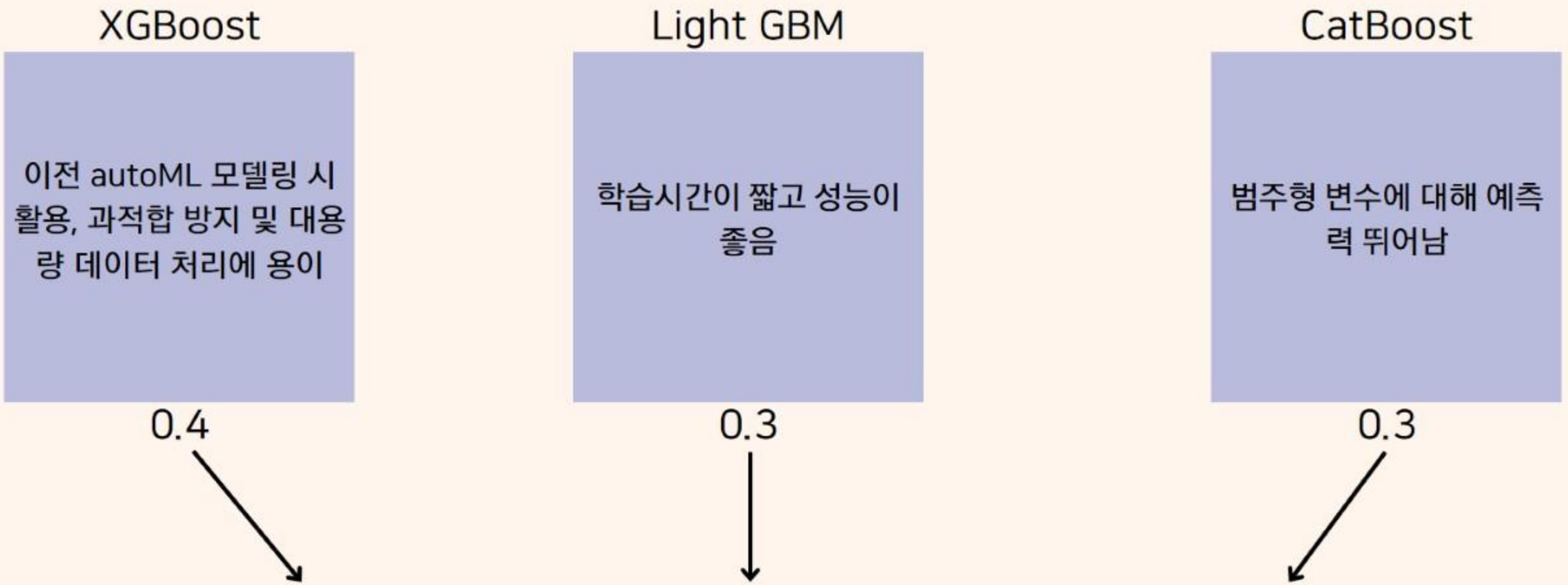
```
C:\Users\admin\anaconda3\envs\py385\lib\site-packages\statsmodels\stats\outliers_influence.py:198: RuntimeWarning: divide by zero encountered in doub
e_scalars
  vif = 1. / (1. - r_squared_i)
```

	VIF Factor	features
0	3.237974e+00	ECLO
1	7.677110e+05	year
2	4.860924e+00	month
3	4.243578e+00	day
4	7.406706e+00	hour
5	4.025967e+01	보안등_설치총개수
6	3.413765e+04	보안등_평균설치개수
7	3.700276e+06	보안등_평균위도
8	2.362935e+07	보안등_최저위도
9	1.365133e+08	보안등_최고위도
10	8.888057e+07	보안등_평균경도
11	8.595001e+07	보안등_최저경도
12	8.433033e+06	보안등_최고경도

4. 최종 컨퍼런스

modeling summary

Gradient Boosting 기반 모델 선정 및 앙상블



4. 최종 컨퍼런스

modeling code XGBoost

```
def xgb_modeling(X_train, y_train, X_valid, y_valid):
  def objective(trial):
    params = {
        'learning_rate': trial.suggest_float('learning_rate', 0.0001, 0.1),
        'min_child_weight': trial.suggest_int('min_child_weight', 1, 20),
        'gamma': trial.suggest_float('gamma', 0.01, 1.0),
        'reg_alpha': trial.suggest_float('reg_alpha', 0.01, 1.0),
        'reg_lambda': trial.suggest_float('reg_lambda', 0.01, 1.0),
        'seed':42,
        'max_depth': trial.suggest_int('max_depth', 3, 15), # Extremely prone to overfitting!
        'n_estimators': trial.suggest_int('n_estimators', 300, 3000, 200), # Extremely prone to overfitting!
        'eta': trial.suggest_float('eta', 0.007, 0.013), # Most important parameter.
        'subsample': trial.suggest_discrete_uniform('subsample', 0.5, 1, 0.1),
        'colsample_bytree': trial.suggest_discrete_uniform('colsample_bytree', 0.4, 0.9, 0.1),
        'colsample_bylevel': trial.suggest_discrete_uniform('colsample_bylevel', 0.4, 0.9, 0.1),
    }

    model = XGBRegressor(**params, random_state=42, n_jobs=-1, objective='reg:squaredlogerror')
    bst_xgb = model.fit(X_train,y_train, eval_set = [(X_valid,y_valid)], eval_metric='rmsle', early_stopping_rounds=100,verbose=False)

    preds = bst_xgb.predict(X_valid)
    if (preds<0).sum()>0:
      print('negative')
      preds = np.where(preds>0,preds,0)
    loss = msle(y_valid,preds)

    return np.sqrt(loss)

  study_xgb = optuna.create_study(direction='minimize',sampler=optuna.samplers.TPESampler(seed=100))
  study_xgb.optimize(objective,n_trials=30,show_progress_bar=True)

  xgb_reg = XGBRegressor(**study_xgb.best_params, random_state=42, n_jobs=-1, objective='reg:squaredlogerror')
  xgb_reg.fit(X_train,y_train,eval_set = [(X_valid,y_valid)], eval_metric='rmsle', early_stopping_rounds=100,verbose=False)

  return xgb_reg,study_xgb
```

eta -> learning rate가 큰 범위를 가지므로 세부적인 학습률 미세 조정

early stopping 추가

optuna로 파라미터 튜닝, 목표 함수 최소화 진행

튜닝된 파라미터로 다시 학습 진행

4. 최종 컨퍼런스

modeling code Catboost

```
def cat_modeling(X_train, y_train, X_valid, y_valid):
  def objective(trial):
    param = {
        'iterations':trial.suggest_int("iterations", 1000, 20000),
        'od_wait':trial.suggest_int('od_wait', 500, 2300),
        'learning_rate' : trial.suggest_uniform('learning_rate',0.01, 1),
        'reg_lambda': trial.suggest_uniform('reg_lambda',1e-5,100),
        'subsample': trial.suggest_uniform('subsample',0,1),
        'random_strength': trial.suggest_uniform('random_strength',10,50),
        'depth': trial.suggest_int('depth',1, 15),
        'min_data_in_leaf': trial.suggest_int('min_data_in_leaf',1,30),
        'leaf_estimation_iterations': trial.suggest_int('leaf_estimation_iterations',1,15),
        'bagging_temperature' :trial.suggest_loguniform('bagging_temperature', 0.01, 100.00),
        'colsample_bylevel':trial.suggest_float('colsample_bylevel', 0.4, 1.0),
    }


    model = CatBoostRegressor(**param, random_state=42)
    #task_type="GPU",devices='0:1'
    bst_cat = model.fit(X_train,y_train, eval_set = [(X_valid,y_valid)], early_stopping_rounds=100,verbose=False)

    preds = bst_cat.predict(X_valid)
    if (preds<0).sum()>0:
      print('negative')
      preds = np.where(preds>0,preds,0)
    loss = msle(y_valid,preds)

    return np.sqrt(loss)

  study_cat = optuna.create_study(direction='minimize',sampler=optuna.samplers.TPESampler(seed=100))
  study_cat.optimize(objective,n_trials=30,show_progress_bar=True)

  cat_reg = CatBoostRegressor(**study_cat.best_params, random_state=42)
  cat_reg.fit(X_train,y_train,eval_set = [(X_valid,y_valid)], early_stopping_rounds=100,verbose=False)

  return cat_reg,study_cat
```

od_wait -> 과적합 탐지

Bagging_temperature -> boosting
단계에서 샘플링 빈도 조절

4. 최종 컨퍼런스

modeling code Light GBM

```
def lgbm_modeling(X_train, y_train, X_valid, y_valid):
  def objective(trial):
    param = {
        'objective': 'regression',
        'verbose': -1,
        'metric': 'rmse',
        'num_leaves': trial.suggest_int('num_leaves', 2, 1024, step=1, log=True),
        'colsample_bytree': trial.suggest_uniform('colsample_bytree', 0.7, 1.0),
        'reg_alpha': trial.suggest_uniform('reg_alpha', 0.0, 1.0),
        'reg_lambda': trial.suggest_uniform('reg_lambda', 0.0, 10.0),
        'max_depth': trial.suggest_int('max_depth',3, 15),
        'learning_rate': trial.suggest_loguniform("learning_rate", 1e-8, 1e-2),
        'n_estimators': trial.suggest_int('n_estimators', 100, 3000),
        'min_child_samples': trial.suggest_int('min_child_samples', 5, 100),
        'subsample': trial.suggest_loguniform('subsample', 0.4, 1),
    }

    model = LGBMRegressor(**param, random_state=42, n_jobs=-1)
    bst_lgbm = model.fit(X_train,y_train, eval_set = [(X_valid,y_valid)], eval_metric='rmse',callbacks=[early_stopping(stopping_rounds=100)])

    preds = bst_lgbm.predict(X_valid)
    if (preds<0).sum()>0:
      print('negative')
      preds = np.where(preds>0,preds,0)
    loss = msle(y_valid,preds)

    return np.sqrt(loss)

  study_lgbm = optuna.create_study(direction='minimize',sampler=optuna.samplers.TPESampler(seed=100))
  study_lgbm.optimize(objective,n_trials=30,show_progress_bar=True)

  lgbm_reg = LGBMRegressor(**study_lgbm.best_params, random_state=42, n_jobs=-1)
  lgbm_reg.fit(X_train,y_train,eval_set = [(X_valid,y_valid)], eval_metric='rmse', callbacks=[early_stopping(stopping_rounds=100)])

  return lgbm_reg,study_lgbm
```

파라미터 추가 및 예측 optuna로 조정 및 재예측

4. 최종 컨퍼런스

Scoring

전체 100%의 비율에서 가중치를 할당하여 앙상블 진행

```
#prediction = xgb_prediction * 0.1 + cat_prediction * 0.4 + lgbm_prediction * 0.5  0116
prediction = xgb_prediction * 0.4 + cat_prediction * 0.3 + lgbm_prediction * 0.3 #0116_2 <- 채택
#prediction = xgb_prediction * 0.6 + cat_prediction * 0.1 + lgbm_prediction * 0.3 #0117
#prediction = xgb_prediction * 0.5 + cat_prediction * 0.1 + lgbm_prediction * 0.4 #0117_2
submit = sample_submission.copy()
submit['ECLO'] = prediction
```

가장 좋은 스코어가 나온 가중치 채택

이전 private 0.42741 보다 성능이 개선됨

submission_0117_2.csv edit	2024-01-17 14:53:36	0.4266788283 0.4266586039	○
submission_0117.csv edit	2024-01-17 14:51:48	0.4268438398 0.4267407154	○
submission_0116_2.csv edit	2024-01-17 14:10:29	0.4265489817 0.4265312958	○
submission_0116.csv edit	2024-01-16 14:48:57	0.4264000957 0.4265736434	○

4. 최종 컨퍼런스

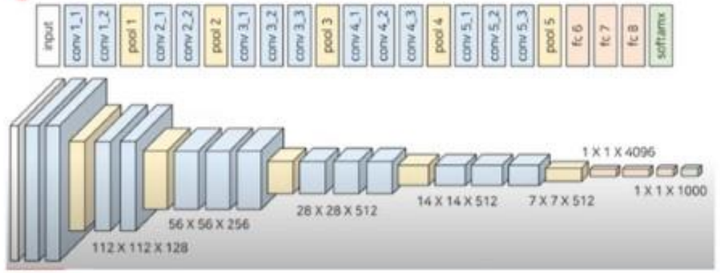
대회 종료 및 모델 추가작업 이후 2주 정도의 시간이 남아 multi- modal 관련 추가 스터디 진행

Resnet 50

깊은 네트워크를 학습 시키기 위한 방법으로 **잔여 학습** 제안
일반적인 CNN은 layer가 깊을 수록(채널의 수(=feature 갯수) increase, 너비와 높이는 decrease) error 확률 올라감 -> 잔여학습으로 이러한 한계 개선

resnet 50 나오기 전 VGG
실질적 weight layer 16개
주기적으로 pooling layer를 거쳐 downsampling 할수 있도록함, 단점이 있다면 파라미터가 많음, layer가 깊어도 잘 추출할 수 있게 하는 장점. but 무조건 layer 늘린다고 성능이 좋아지진 않음

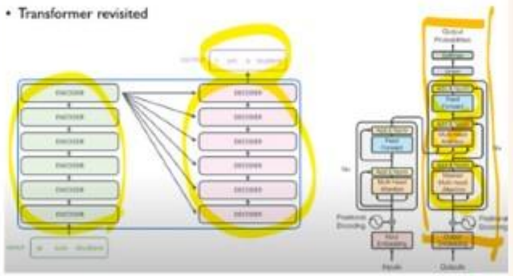
- VGG 네트워크는 작은 크기의 3x3 컨볼루션 필터(filter)를 이용해 레이어의 깊이를 늘려 우수한 성능을 보입니다.



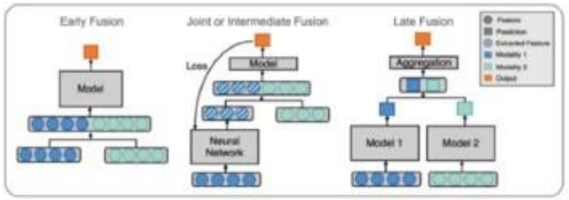
GPT2

generative pre-training 비지도 학습 기반 , 지도 학습 형태
데이터의 분포 그대로 가져와서 모델링 , fine-tuning 과정없음 (gpt1과 auto-regressive -> 토큰이 생성되면, 생성된 토큰은 다음 학습의 input

- Transformer revisited



멀티모달의 종류

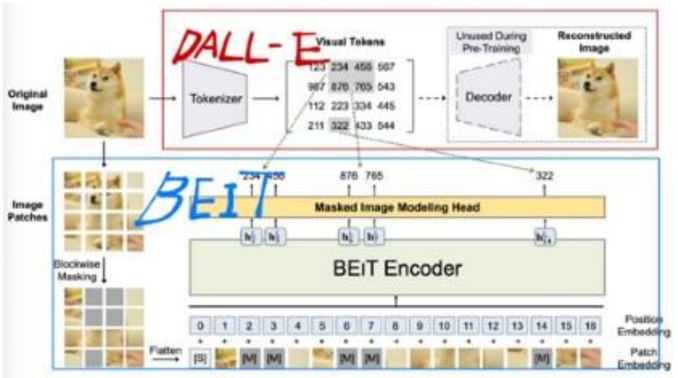


- Early Fusion
Early Fusion은 종류가 다른 두가지 데이터를 하나의 데이터로 먼저 합친 이후 모델 학습을 시키는 경우다. 이 때 형식이 다른 두 데이터를 합치기 위해서는 다양한 데이터 변환이 이루어진다. 원시데이터를 그대로 융합해도 괜찮고, 전처리를 한 이후에 융합해도 상관없다.

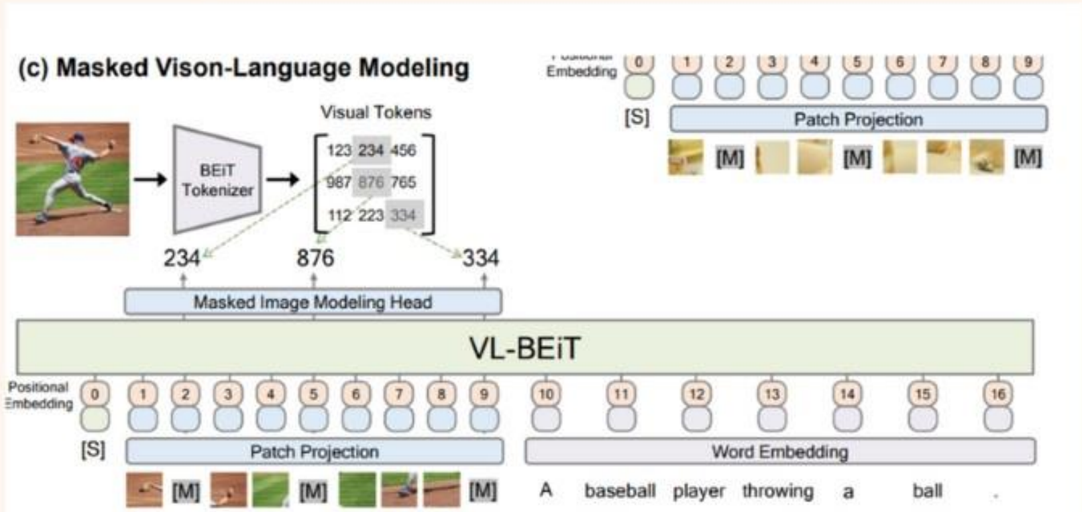
- Late Fusion
Late Fusion은 종류가 다른 두가지 데이터를 각각 다른 모델에 학습시킨 이후 나온 결과를 융합하는 방법으로, 기존의 앙상블모델이 작동하는 방식과 비슷하다.

- Joint or Intermediate Fusion
Joint Fusion은 두개의 모달리티 데이터를 동시에 학습시키지 않고 내가 원하는 모델의 깊이에서 모달리티를 병합할 수 있는 유연성을 가지고 있다. 하나의 모달리티로 모델학습을 진행하다가 모델학습의 마지막 레이어 전에 다른 모달리티와 융합하는 방법으로, 이 과정을 end-to-end learning이라고도 한다.

1. 학습된 어휘(DALL-E의 토크나이저)에 따라 개별 시각적 토큰으로 토큰화



2. 이미지패치의 일부를 랜덤하게 마스킹하고 손상된 입력을 Transfromer에 공급, 모델은 마스킹된 패치의 픽셀 대신 원본 이미지의 visual token을 복구하는 방법을 학습



참고 문헌

보안등
https://www.imaeil.com/page/view/2018080611112023698
어린이 보호구역
https://news.mt.co.kr/mtview.php?no=2020101514448211097
주차장
https://www.yna.co.kr/view/AKR20231129069000053
cctv
https://www.kukinews.com/newsView/kuk201810100352
자동차 등록대수
https://insfiler.com/detail/rt_cars-0009
외부데이터 - 자동차 등록대수
https://www.data.go.kr/data/15073712/fileData.do#layer_data_infomation
외부데이터- 일출 일몰시간
https://www.data.go.kr/data/15053554/fileData.do
보안등, cctv, 어린이 보호구역, 주차장 데이터는 주최 측 제공

감사합니다!

질문 있으시면 말씀해주세요