제주도 도로 교통량 예측

AI 경진대회

주최 : 제주 테크노파크, 제주특별자치도
주관 : 데이콘
심승현

목차

1. 문제 정의

제주도 인구는 연 평균 1.3% 정도 증가하고 있고, 외국인과 관광객을 포함하면

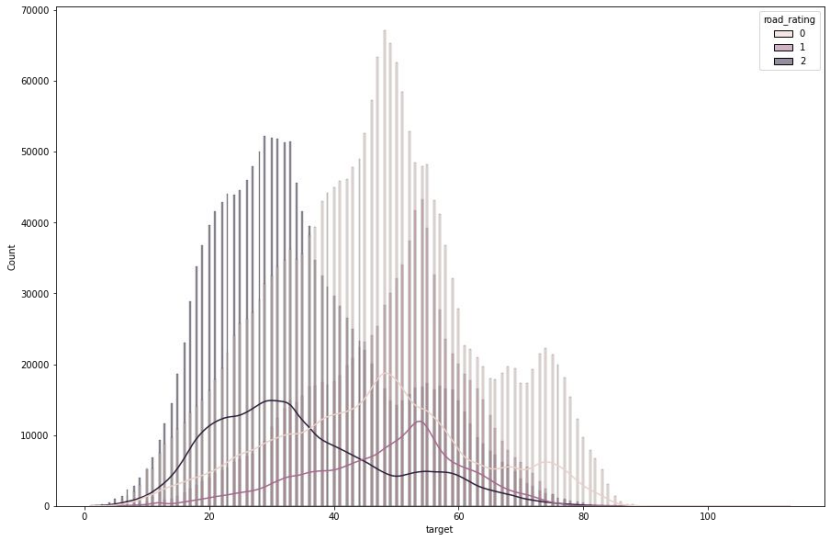
상주인구가 90만명이 넘을 것으로 추정함. 제주도 내 인구 증가로 심각한
교통체증이 문제로 떠오르고 있음.

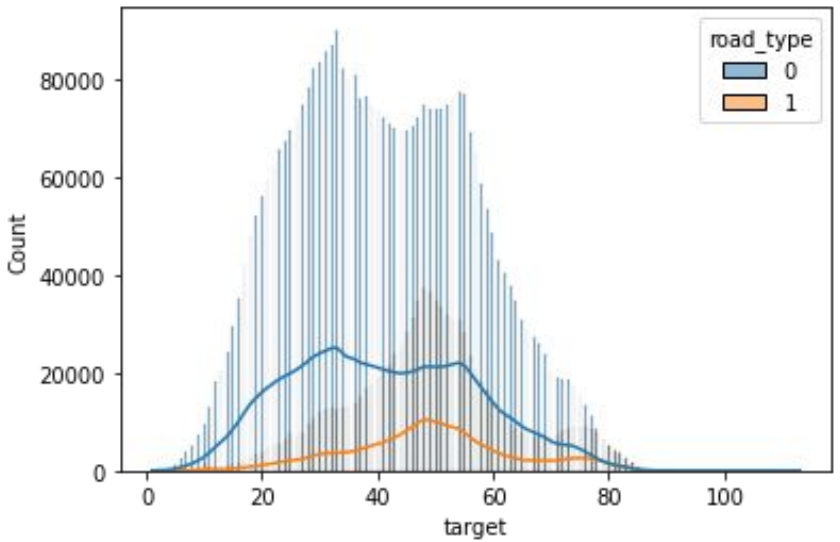
날짜, 시간, 교통 및 도로구간 정보 등, 총 21개 feature을 이용하여
8월 도로 교통량 예측

2. 데이터 분석

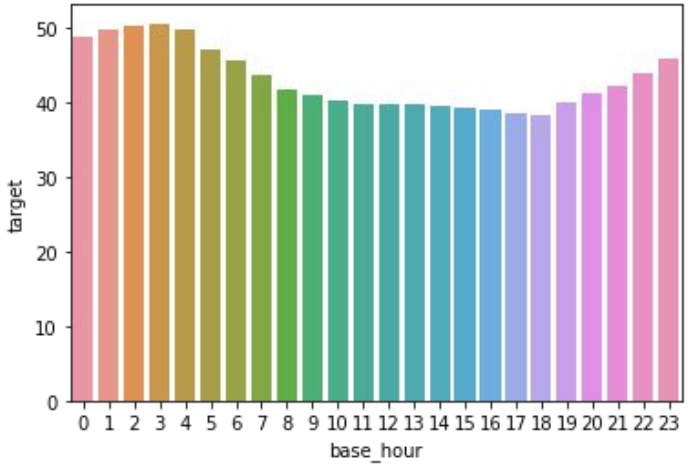
```
 #   Column                Dtype
---  ------                -----
 0   id                    object
 1   base_date             int64
 2   day_of_week           object
 3   base_hour             int64
 4   lane_count            int64
 5   road_rating           int64
 6   road_name             object
 7   multi_linked          int64
 8   connect_code          int64
 9   maximum_speed_limit   float64
 10  vehicle_restricted    float64
 11  weight_restricted     float64
 12  height_restricted     float64
 13  road_type             int64
 14  start_node_name       object
 15  start_latitude        float64
 16  start_longitude       float64
 17  start_turn_restricted object
 18  end_node_name         object
 19  end_latitude          float64
 20  end_longitude         float64
 21  end_turn_restricted   object
 22  target                float64
dtypes: float64(9), int64(7), object(7)
memory usage: 825.0+ MB
총 데이터 수 : 4701217
```

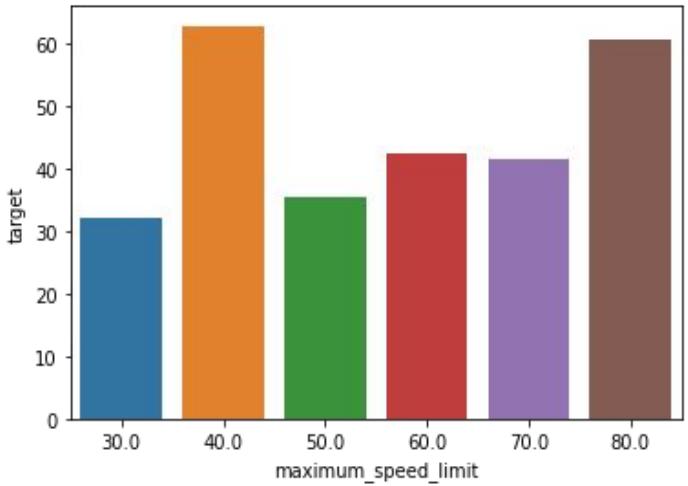
- 22개의 X feature와 1개의 Y값으로 구성
- 총 데이터 수는 4,700,000개
- categorical feature는 6개 존재





road_rating, road_type feature는 **다른 모양의 분포**를 보이기에
y에 영향을 미칠 것

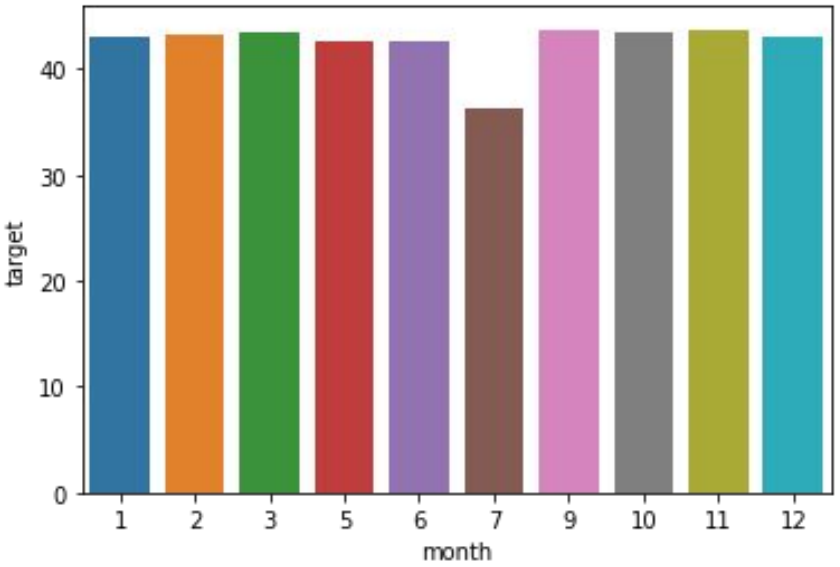




차량 이동량이 많은 오후 시간대와 저녁,밤 시간때의 차이가 뚜렷하게 보임.

두 feature는 y에 영향을 미칠거라 판단

제한 속도에 따라 차이가 보이며, 특히 제한속도 40Km 평균속도가 40Km를 넘어 과속



7월달의 평균 속도가 눈에 띄게 다름

⇩

7~8월 휴가철 때문일 것이라 가정

⇩

target은 8월 데이터이므로 파생 변수 생성
(sin + cos)

2-2. 피쳐 삭제

- id > 고유값 삭제
- base_date > year, month, day feature로 변경
- year > year는 2021.09 ~ 2022.07 데이터만 존재
- day > day별 y속도가 별 차이 없음
- vehicle_restricted > 모든 값이 0
- height_restricted > 모든 값이 0
- weight_restricted > 평균 속도에서는 차이가 나지만, 같은 도로의 해당 피쳐를 살펴보니, 그래프의 형태가 비슷함. 즉 해당 피쳐는 영향이 없다 판단
- multi_linked > 0,1 두 값이 존재하지만, 1의 비율이 0.0005로 너무 작음

3. 파생변수 생성

1. season

```
#  양력 > 3~5월이 봄, 6~8월이 여름, 9~11월이 가을, 12~2월이 겨울이다.
train["season_plus"] = train["month"].apply(lambda x :  0 if 3<= x <=5 else 1 if 6<= x <=8 else 2 if 9<= x <=10 else 3)
```

2. 시작,끝 지점 빈도 비율

```
train["startNodeRatio"] = train["start_node_name"].apply(lambda x : start_node_ratio_set.iloc[x].values[0])
train["endNodeRatio"] = train["end_node_name"].apply(lambda x : end_node_ratio_set.iloc[x].values[0])
```

3. 거리(Km)

```
for i in tqdm(range(len(train))):
  A = (train["start_latitude"].iloc[i], train["start_longitude"].iloc[i]) # (lat, lon)
  B = (train["end_latitude"].iloc[i], train["end_longitude"].iloc[i])

  km_result.append(haversine(A, B)) # km
```

4. 공휴일

```
train["holiday"] = train["base_date"].apply(lambda x: 1 if x in holiday_list else 0)
```

4. return

```
train["return"] = train["start_turn_restricted"] + train["end_turn_restricted"]
```

5. geniality(쾌적)

```
train["geniality"] = train["lane_count"] * (train["road_rating"] + 0.00001) * (train["road_type"] + 0.00001)
```

6. sin_24_1(시간 주기성)

```
train['sin_24_1'] = np.sin(2 * np.pi * train['base_hour']/23.0) * np.cos(2 * np.pi * train['base_hour']/23.0)
```

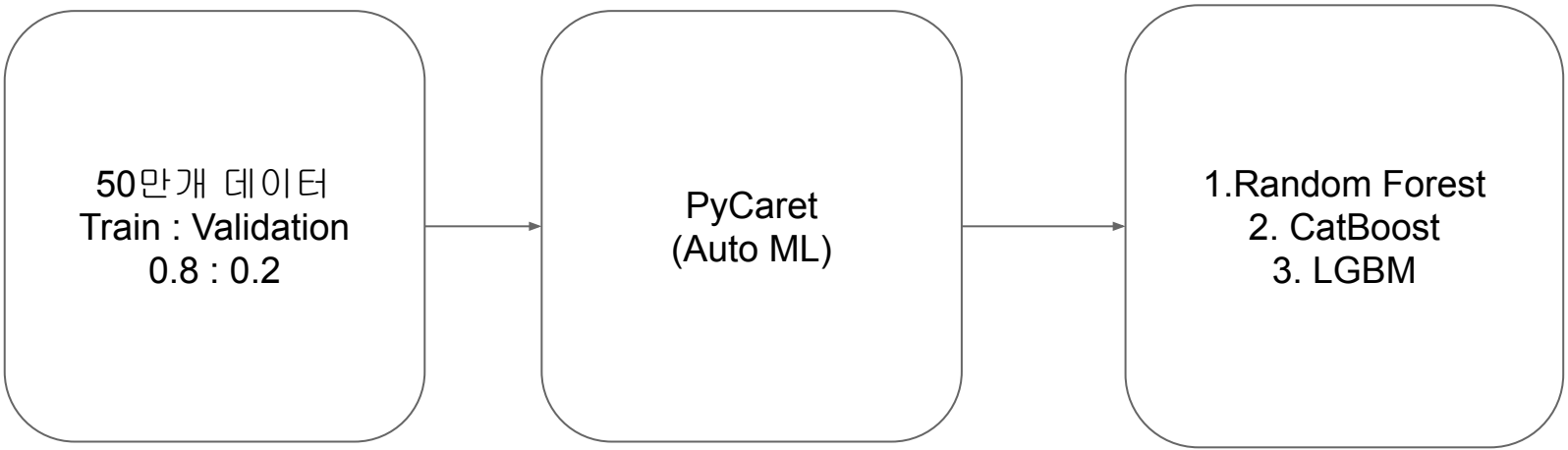
7. sin_month(달 주기성)

```
train["sin_month"] = np.sin(2 * np.pi * train['month']/12) * np.cos(2 * np.pi * train['month']/12)
```

8. sin_season(시즌 주기성)

```
train['sin_season'] = np.sin(2 * np.pi * train['season_plus']/4) * np.cos(2 * np.pi * train['season_plus']/4)
```

4. 모델링



학습 속도를 위해
50만개 데이터 사용

3가지 모델 선택
(앙상블)

가장 성능이 좋은 model을 찾아내고, 전체 데이터 학습

4-1. 최적화

Catboost

```
bagging_temperature : 1e-4 ~ 100
leaf_estimation_iterations : 5 ~ 25
min_data_in_leaf : 3 ~ 20
depth : 1 ~ 16
random_strength : 1 ~ 50
subsample : 0 ~ 1
reg_lambda : 1e-5 ~ 100
learning_rate : 1e-3 ~ 0.9
iterations : 50~2500
```

*optuna를 통하여 파라미터 최적화 실행

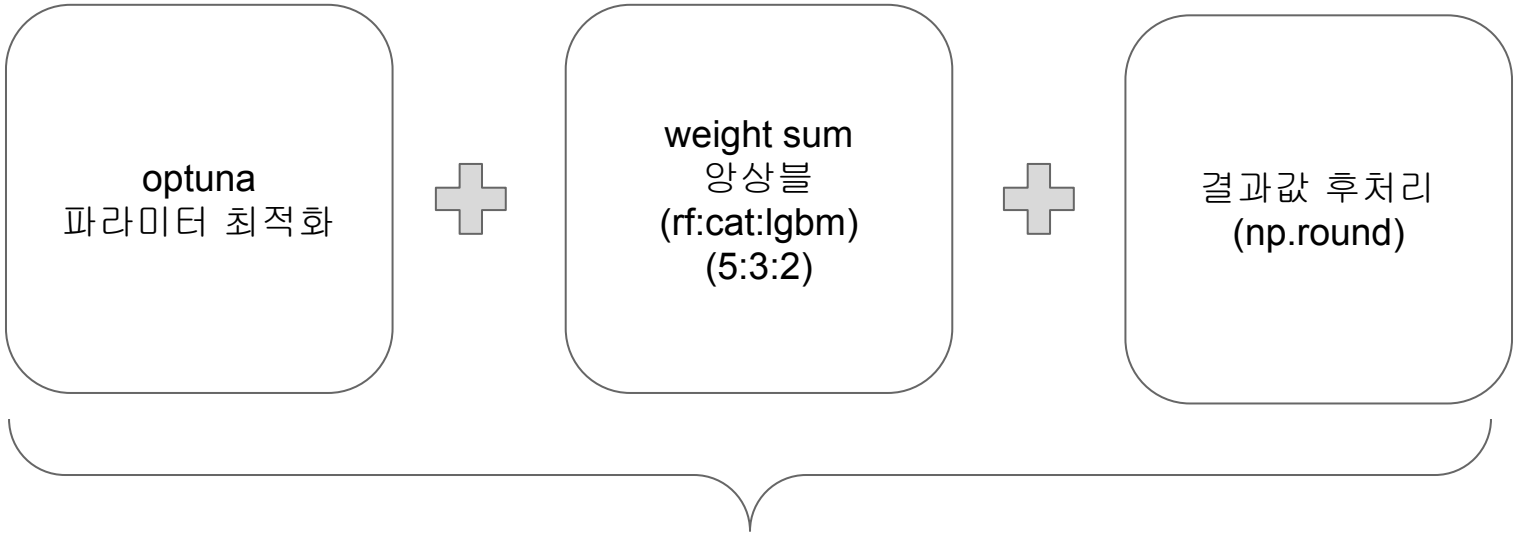
RF

```
min_samples_split : 15 ~ 70
min_samples_leaf : 2 ~ 30
n_estimators : 150 ~ 250
```

LGBM

```
num_leaves : 50 ~ 100
subsample : 1e-4 ~ 1
min_child_weight : 1e-4 ~ 0.9
min_child_samples : 25 ~ 70
n_estimators : 100 ~ 3000
```

5. 성능향상



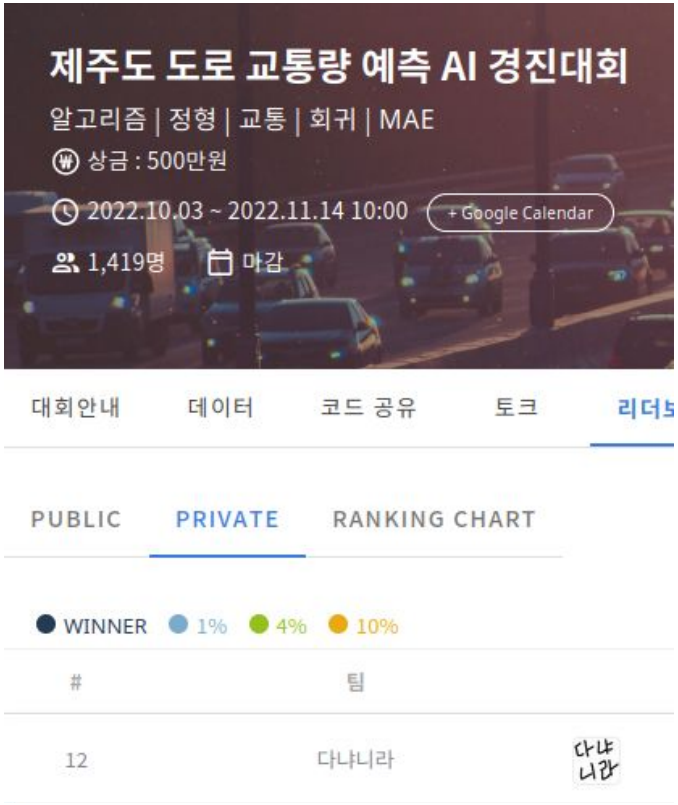
최종 제출 파일 생성

6. 결론

데이터를 살펴보고 분석하여
정통 머신러닝 기법으로 문제에 접근

catboost, lgbm의 튜닝 전 점수가
rf에 비해 성능이 좋지 않아 앙상블을 고려하지
않았음.

catboost, lgbm의 튜닝 후 성능 개선을 통해
앙상블을 진행한 것이 좋은 결과를 얻는데 중요한
부분을 차지.



- 12등 / 1,419명

참여기간 : 2022.10.21 ~ 2022.11.14

감사합니다.