https://dacon.io/competitions/open/235837/codeshare/3724?page=2&dtype=vote&fType=&category=codeshare

주제 및 데이터

주제

2018년 4-6월, 2019년 4-6월, 2020년 4-6월 따릉이 대여량 데이터를 바탕으로 2021년 데이터를 예측

데이터

- date_time : 일별 날짜
- wind_direction: 풍향 (degree)
- sky_condition : 하늘 상태 (하단 설명 참조)
    - 맑으면 1, 구름 많으면 3, 흐림은 4
- precipitation_form : 강수 형태 (하단 설명 참조)
    - 없으면 0, 비면 1
    - 4~6월의 데이터만 추출하여 큰 예외 상황이 없음
- wind_speed : 풍속 (m/s)
- humidity : 습도 (%)
- low_temp : 최저 기온 (`C)
- high_temp : 최고 기온 (`C)
- Precipitation_Probability : 강수 확률 (%)
- number_of_rentals : 따릉이 대여량

코드 흐름

EDA

Q1. 연도가 증가분에 영향을 미칠것인가?

연도 변수와 월일 변수를 구분하여 연도별 대여량 흐름 보기
→ 전체적으로 최근으로 올수록 사용량 증가. 반면 바닥을 칠 땐 연도에 상관없이 바닥을 침 → 실제로 증가 추세가 맞는지 연도별 대여량 중간값 비교
→ 맞음.

Q2. 변수들간 관계 분석

Heatmap, pairplot (hue=연도 로 두어 연도별로 변화가 있는지 관찰)
→ 생각보다 연도에 따른 기후적인 요소의 차이는 크지 않음. 연도별로 사용량이 증가하는 이유는 기후적 요인보단 그 자체인듯. & 강수량, 바람, 하늘상태에 따라 대여량이 크게 달라짐.

Q3. 기사에 따르면 따릉이의 경우 출퇴근 시간에 사용 비중이 높다고 하던데 평일, 주말 사용량이 많이 다를까?
평일, 주말 각각의 중간값 비교. 날씨에 따라 대여량이 너무 낮은 영향을 없애기 위해서 중간값 사용.
→ 중간값 자체는 더 높았음. 일단은 평일/주말 변수 생성.

Feature Engineering

Drop original feature

바람의 방향(안 중요해보임), 강우 확률(실제로 내린 양을 나타내지 않음. 강우량 변수 존재), 달(기온 변수가 설명.) 피처 삭제

Create new feature

- 불쾌지수 : 습도, 최대 기온, 최저 기온
  temp = (min_t + max_t)/2
  humid = humid / 100
  discomfort = 1.8 * temp - 0.558 * (1 - humid) * (1.8 * temp - 26) + 32
- 악조건 : 풍속 * 하늘 상태
  train_df['hardship'] = train_df['sky_condition'] * train_df['wind_speed']
- 추운정도 : 최저 기온 / 풍속
  train_df['cold_measure'] = train_df['low_temp'] / train_df['wind_speed']
- 일교차(체감온도) : 최고 기온 - 최저 기온
  train_df['temp_diff'] = train_df['high_temp'] - train_df['low_temp']

Modeling

평가 지표

NMAE = np.mean(abs(pred - y) / y)

train-test 분리**?**

각 일자별로 어떤 기후 특성을 가지고 있는지 알기 어려움 & 그래도 일자가 흐름에 따른 어느정도의 대여량 증가 추세가 보임 → 분리를 했을 때, 학습이 정말 유의미한지 알기가 어렵고 test에서도 동일한 기간에 대해 예측해야 함 → 따라서 train 과 test를 분리하지 않고, X, y만 분리하여여 여러번의 trial and error를 거치자.

모델 생성 계획

- 선형회귀: 단순선형회귀
- 회귀트리: XGBoost (LGBM은 데이터수가 많지 않아서 제외)

단순선형회귀

오차율 약 30%의 좋지 않은 결과
→ 이상치를 제거하여 재적용하는 방법이 있지만 날씨가 bad하여 대여량이 매우 낮은 값들을 다 쳐낼거같음.
→ **statmodels**의 **summary**로 간단히 확인
→ 유의한 피처가 적지만, 나머지는 무작정 버리기엔 유실되는 정보가 너무 많을 듯
→ 특정 기준으로 날씨가 매우 나쁘거나 하면 선제적으로 쳐내어 낮은 값을 할당해주겠다는 기대를 하며 트리기반 모델 적용

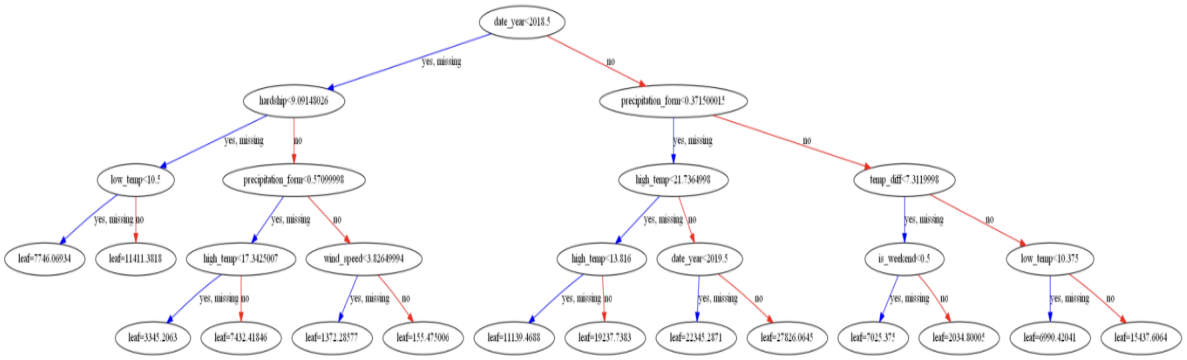
XGBoost

오차율 5%의 좋은 정확도. 확실히 분류 기준으로 특정값을 부여하는 방식이 훨씬 좋음.

→ **GridSearchCV**로 최적의 파라미터 조사**.** 'max_depth': 4, 'n_estimators': 50

→ 전체 데이터를 학습하여 아예 새로운 연도를 학습해줘야 하기에 n_estimators는 100으로 유지

→ 트리의 시각화를 통해 **XGBoost**가 어떤 결정을 내리고 있는지를 확인



→ 우측에서 말단에서 한번 date_year를 사용한 것 빼고는 year를 생각보다는 많이 쓰고 있지 않음. 연도가 가장 1차적인 분류 기준으로 쓰이는 것을 고려한다면 2021년의 데이터는 2020년과는 차등을 둬야 할 것임.

→ **plot_importance**로 변수별 중요도 확인. '연도'가 학습에 반영이 잘 안됨. 새로 만든 변수인 불쾌지수, 주말여부도 그닥.

→ 연도별 사용량 합계의 상승분 확인. 2018년에서 2019년으로 갈때, 동기간 따릉이 이용량은 약 2배, 2019년에서 2020년은 1.2배 증가.

→ 최종 예측치에 **2019**년**~20**년도의 상승분인 **1.2**배를 동일하게 적용하는 것이 최선

차별점 및 새롭게 얻은 인사이트

- 이상치가 유의미하게 중요한 데이터는 선형회귀보다 트리회귀에서 좋은 성능을 얻을 수 있겠다.
- 변수 간 관계를 관찰하거나 새로운 변수를 생성할 때 도메인 지식이 없으면 많이 힘들겠다 싶었다.
- 코딩 결과로 나온 모델을 그대로 따르지 않고 인사이트만을 근거로 최종 예측치에 가중치를 적용했음에도 좋은 결과를 얻은 점이 인상적이었다.