https://dacon.io/competitions/official/236033/codeshare/7779?page=1&dtype=recent

주제

생육 환경 생성 AI 모델 결과를 바탕으로 상추의 일별 최대 잎 중량을 도출할 수 있는 최적의 생육 환경 조성

데이터

train_input [폴더] - 총 28개 상추 케이스

DAT : 생육 일 (0~27일차)

obs_time : 측정 시간

상추 케이스 별 환경 데이터 (1시간 간격)

train_target [폴더] - 총 28개 상추 케이스

DAT : 생육 일 (1~28일차)

predicted_weight_g : 일별 잎 중량

test_input [폴더] - 총 5개 상추 케이스

DAT : 생육 일 (0~27일차)

obs_time : 측정 시간

상추 케이스 별 환경 데이터 (1시간 간격)

test_target [폴더] - 총 5개 상추 케이스

DAT : 생육 일 (1~28일차)

predicted_weight_g : 일별 예측한 잎 중량

제출을 위한 양식으로 target에 해당되는 predicted_weight_g의 값은 모두 0으로 가려져있음.

sample_submission.zip [제출양식]

[정량 평가] 예측 모델 평가를 위한 제출 양식

Test 청경채 케이스 6개에 대한 일별 추론한 결과

전처리

```
train_data1.columns
```

```
Index(['DAT', 'obs_time', '내부온도관측치', '내부습도관측치', 'co2관측치', 'ec관측치', '시간당분무량',
       '일간누적분무량', '시간당백색광량', '일간누적백색광량', '시간당적색광량', '일간누적적색광량', '시간당청색광량',
       '일간누적청색광량', '시간당총광량', '일간누적총광량'],
      dtype='object')
```

1. 누적 칼럼 삭제
2. 내부 온도, 내부 습도 이상치값을 해당 obs_time 평균 기온, 평균 습도로 변환
3. 시간당광량 마이너스 값 존재. 해당 obs_time의 1일차 데이터 값과 동일한 값 넣기
4. 시간당분무량 마이너스 값 존재. 0으로 대체.

Feature Engineering

Create new feature

- 배양액농도 : ec관측치 * 시간당분무량
- 온습도 : 내부온도관측치 * 내부습도관측치
- 온습도co2 : 내부온도관측치 * 내부습도관측치 * co2
- 적색청색광 : 시간당적색광량 * 시간당청색광량
- 광합성속도_적청광 : co2관측치/(시간당적색광+시간당청색광) * (내부온도+생육시기).
  0으로 나눠지는경우 inf 발생하므로 해당경우엔 0으로 변환.
- 케이스별 최저 ~ 최고기온, 최저 ~ 최고광량(백,적,청), 일교차(최고-최저기온) 컬럼 추가
- obs_time값을 **가장 가까운 정각(hour)**으로 반올림한 후 시간(hour) 부분만 추출. 예를 들어, `2024-12-16 11:00:00`은 `11`이라는 값으로 변환됨.

```
train_data_all['obs_time'] = pd.to_datetime(train_data_all['obs_time']).dt.round("H").dt.hour
```

```
train_data_all.columns
```

```
Index(['DAT', 'obs_time', '내부온도관측치', '내부습도관측치', 'co2관측치', 'ec관측치', '시간당분무량',
       '시간당백색광량', '시간당적색광량', '시간당청색광량', '시간당총광량', '배양액농도', '광합성_적청광'],
      dtype='object')
```

데이터 취합 및 시간별 칼럼으로 확장

```
edit_df = pd.DataFrame()

# 일별- 시간별~컬럼 펼치기
for i in range(24):
  tmp = train_data_all[train_data_all['obs_time'] == i].reset_index(drop=True)
  tmp = tmp[['내부온도관측치', '내부습도관측치', 'co2관측치', 'ec관측치', '시간당분무량',
    '시간당백색광량', '시간당적색광량', '시간당청색광량',
    '배양액농도','시간당총광량','광합성_적청광' ]]

  tmp.columns = [f'{i}내부온도관측치', f'{i}내부습도관측치', f'{i}co2관측치', f'{i}ec관측치', f'{i}시간당분무량',
          f'{i}시간당백색광량', f'{i}시간당적색광량', f'{i}시간당청색광량',
          f'{i}배양액농도',f'{i}시간당총광량',f'{i}광합성_적청광']
  edit_df = pd.concat([edit_df, tmp], axis=1)


# 일자별 신규생성 컬럼 추가
edit_df['최고기온'] = max_temp
edit_df['최저기온'] = min_temp
edit_df['최고백색광량'] = max_white
edit_df['최저백색광량'] = min_white
edit_df['최고적색광량'] = max_red
edit_df['최저적색광량'] = min_red
edit_df['최고청색광량'] = max_blue
edit_df['최저청색광량'] = min_blue


# DAT 컬럼 붙이기
tmp = train_data_all[train_data_all['obs_time'] == 23].reset_index(drop=True)
tmp = tmp[['DAT']]
edit_dat_df = pd.concat([edit_df, tmp], axis=1)
edit_dat_df['DAT'] = edit_dat_df['DAT']+1
```

1. 시간별 데이터 컬럼 펼치기 (for 반복문) : `edit_df`에는 0시부터 23시까지의 시간별 데이터를 모두 포함하는 컬럼들이 생성됨.
    - `train_data_all`에서 특정 시간(`obs_time == i`)의 데이터를 필터링.
    - 관심 있는 컬럼만 선택: (내부온도, 내부습도, CO2, EC 등).
    - 선택된 컬럼 이름에 시간을 붙임(`i`를 접두사로 추가하여 `0내부온도관측치`, `1내부온도관측치` 등 생성).
    - `edit_df`에 열 단위로(`axis=1`) 결합.
2. 일자별 지표 추가 : 일자별 추가 정보(환경 지표)가 포함된 데이터프레임 생성.
    - 사전에 계산된 일자별 데이터(`max_temp`, `min_temp`, `max_white` 등)를 `edit_df`에 각각의 새로운 컬럼으로 추가.
    - 컬럼은 최고/최저 기온, 백색광량, 적색광량, 청색광량 등으로 구성.
3. `edit_df`에 `DAT` 정보가 추가된 데이터프레임 `edit_dat_df` 완성
    - `train_data_all`에서 `obs_time`이 23인 행을 필터링.
    - `DAT` 컬럼만 추출.
    - `edit_df`와 열 단위로 결합.
    - `DAT` 값에 1을 더해 갱신.

Modeling

- LGBM 모델로 학습
- test_data_list에 포함된 여러 파일 경로를 반복적으로 읽어들여, 각각을 pandas 데이터프레임으로 생성하고, 동적으로 변수 이름을 만들어 저장하는 작업을 수행
- 왜곡도 있는 피처 로그변환하기

차별점 및 새롭게 얻은 인사이트

- 기존 칼럼에서 알 수 있는 정보로부터 얻어낼 수 있는 누적 칼럼을 과감하게 삭제한 점이 인상적임.
- 시간별 데이터를 가로로 펼쳐서 칼럼화한 후 DAT 정보를 추가하는 방법이 배울만함.