



Team
Farm in Sun

Farm in Sun Project
데이콘 Competition
참여, 그리고
식물 성장 예측 서비스

Presented by
이현경, 원종현, 윤혜림



Contents



01

프로젝트 개요

- 프로젝트 목적
- 프로젝트 기대효과
- 개발환경
- 데이터셋
- 프로젝트 추진 체계

02

Modeling / 진행 상황

- 데이터 전처리
- 이미지 데이터 처리
- 최종 모델링
- FINAL 결과
- 데이콘 대회 진행상황

03

Farm in Sun 서비스

- 웹 서비스 개요
- DB설계
- 웹페이지 시연

04

개선점 및 발전사항

- 개선점
- 발전사항



Part 1:

Project 개요

- 프로젝트 목적
- 개발환경
- 프로젝트 추진 체계
- 프로젝트 기대효과
- 데이터 셋

프로젝트 목적

Dacon 생육환경 최적화 경진대회

채소성장 분석,
성장일기 등을 구현한
서비스 제공

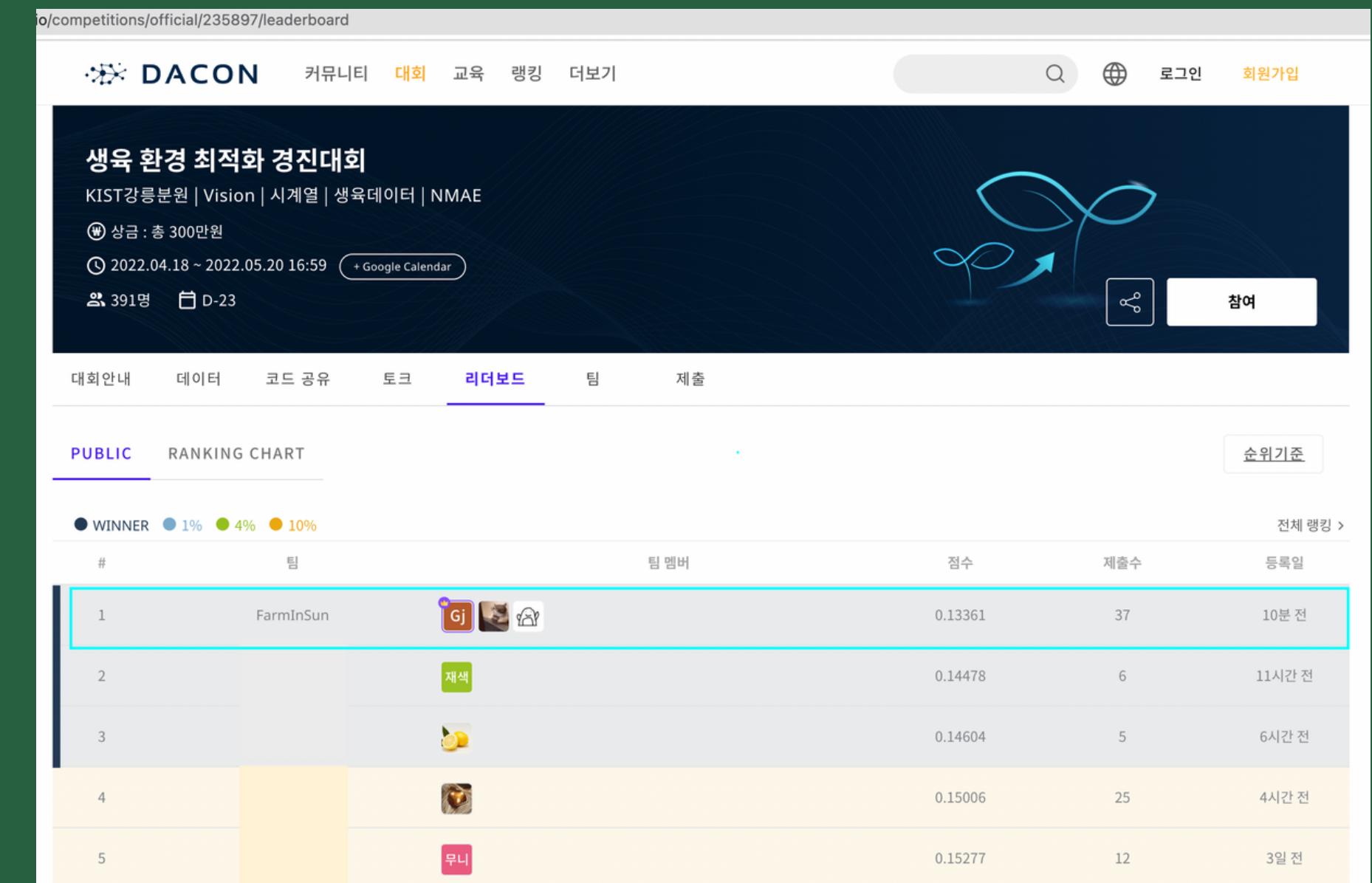
Dacon 생육환경 최적화 경진대회에 참가하여 교육받은 내용을 테스트해보고 좋은 점수를 얻는 것이 목적

Dacon의 데이터를 활용하여 잎의 크기를 예측해주는 모델을 생성하고 이를 사용자가 활용할 수 있도록 웹 서비스로 구현.

| 프로젝트 기대효과

데이콘에서 주최하는 대회에 참가하여 제공된 지표를 이용한 competition을 통해 저희 팀의 객관적 실력을 검증할 수 있을 것입니다.

좋은 성적을 얻기 위해 팀원들과 협력하고 다른이들과 경쟁하는 성공적인 팀 프로젝트를 위한 과정은 교육생들에게 앞으로의 미래에 있어 큰 경험이 될 것입니다.



< 데이콘 리더보드 >

| 프로젝트 기대효과

산업

집에서 키우는 '봄'…'반려식물' 시장 올해도 '쑥쑥'

플랜테리어·식물재배기 트렌드 확산…1인 가구 취향 저격

코로나블루(Corona Blue)에는 반려식물이 인기!①

IT > ICT

22兆 반려식물 시장, 新가전이 뛴다… 씨앗 구독부터 AI 관리까지

빠르게 성장하는 반려식물 재배기 시장
LG전자, 교원 등 관련 사업 뛰어들어

1인 가구 늘면서 '반려식물' 인기몰이

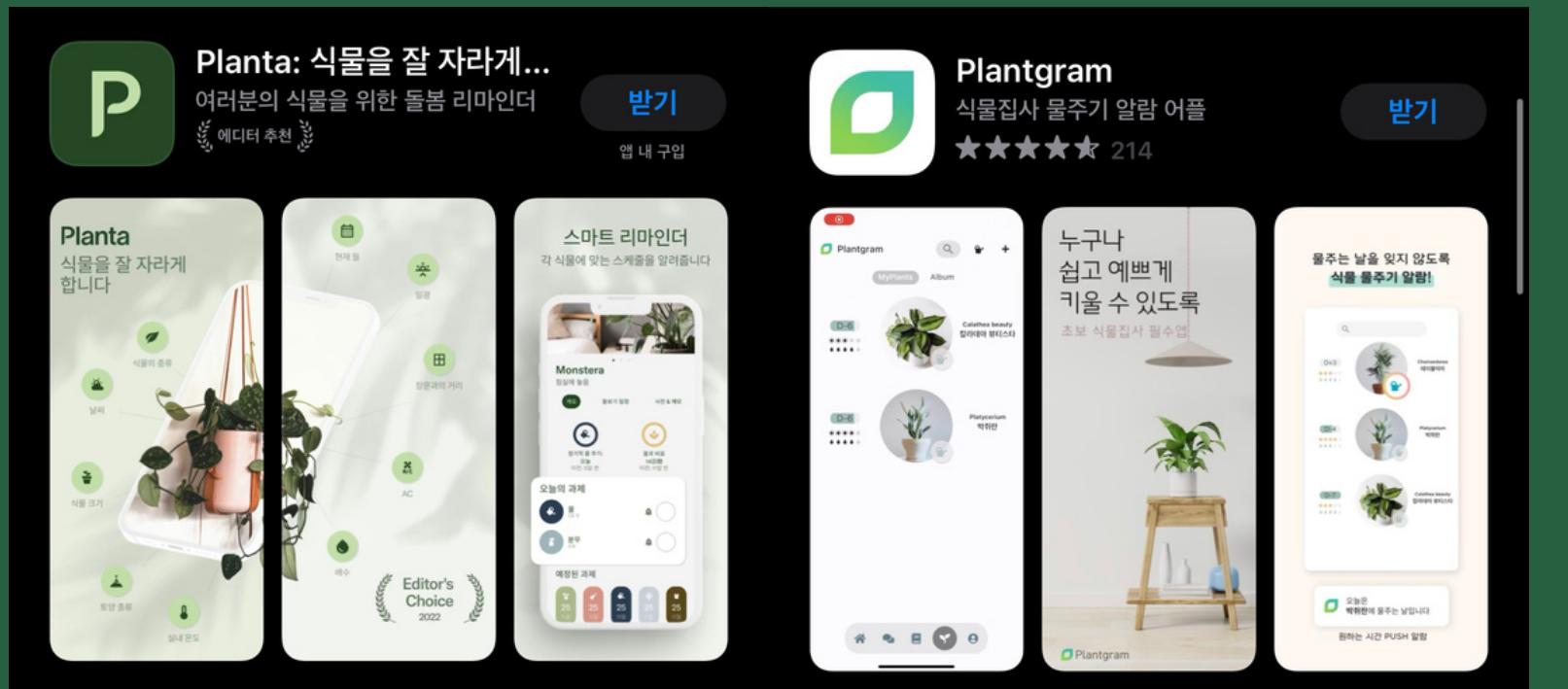
1인 가구의 증가와 코로나의 영향으로 인해 반려식물을 키우는 인구 증가

◇반려식물 인기

반려식물은 생활패턴이 불규칙한 현대인들을 포함해 누구나 쉽게 기를 수 있다는 것이 큰 장점이다. 또한 식물에 따라 다르겠으나 가격이 저렴한 종자, 화분들도 많아 초기 비용이 크게 들지 않는다는 점에서도 인기를 끌고 있다. 농업정보포털에 따르면 코로나 이후 SSG닷컴 홈가드닝 전체 매출은 97%이 상승했으며 G마켓에서도 화분(40%)과 모종(51%), 물조리개(29%)와 같은 홈가드닝 관련 제품들의 매출이 오른 것으로 조사됐다.

반려식물을 키우는 인구의 증가로 부가상품들이 출시되고 그 제품들의 매출 증가

| 프로젝트 기대효과



따라서 관상용 식물이 아닌
'텃밭'을 중심으로 서비스를
제공.

그러나 웹, 앱 서비스는 관상용 식물을
위한 서비스에 포커스가 맞춰져 있음.



개발환경



OS

windows 10 pro
macOS Monterey

IDE

Jupiter lab,
Visual Studio Code 1.66.1,
MySQLWorkBench

언어

Python 3.9.7.

머신러닝 / 딥러닝

Tensorflow 2.8.
CUDA 11.2.
scikit-learn 0.24.2

웹 서비스

Flask, JavaScript

DB

mysql 8.0.28

Data Set



- 컬럼명 (19개)

train [Folder] : 학습용 청경채 데이터셋 (1592장)

CASE : 각 CASE 별 데이터

image : 이미지 데이터

meta : 환경 데이터

↳ 촬영 시작으로부터 1일간 1분 단위로 측정된 환경데이터

label.csv

↳ img_name : 해당 이미지 파일명

↳ leaf_weight : 해당 이미지가 촬영된 시점으로 부터 1일 후의 잎면적

시간	내부온도관측치	외부온도관측치	내부습도관측치
외부습도관측치	CO2관측치	EC관측치	최근분무량
화이트 LED 동작강도	레드 LED 동작강도	블루 LED 동작강도	냉방온도
냉방부하	난방온도	난방부하	총추정광량
백색광추정광량	적색광추정광량	청색광추정광량	

test [Folder] : 테스트용 청경채 데이터셋 (460장)

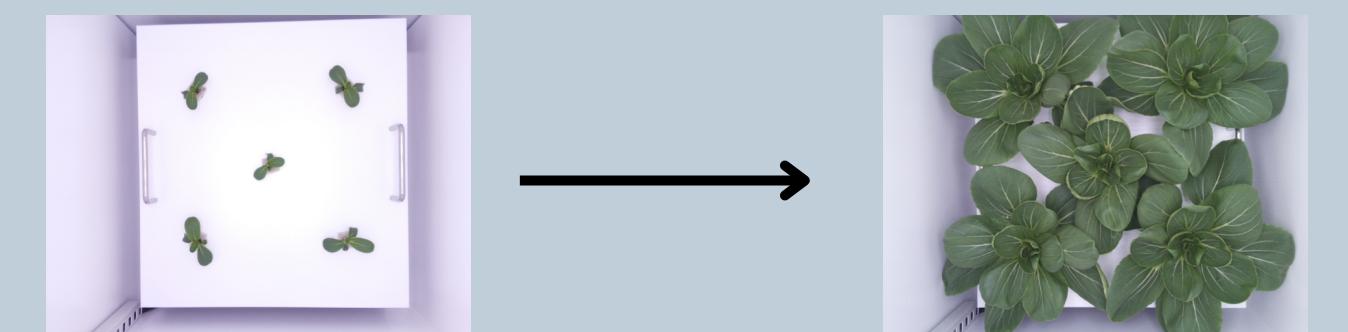
image : 이미지 데이터

meta : 환경 데이터

↳ 촬영 시작으로부터 1일간 1분 단위로 측정된 환경데이터

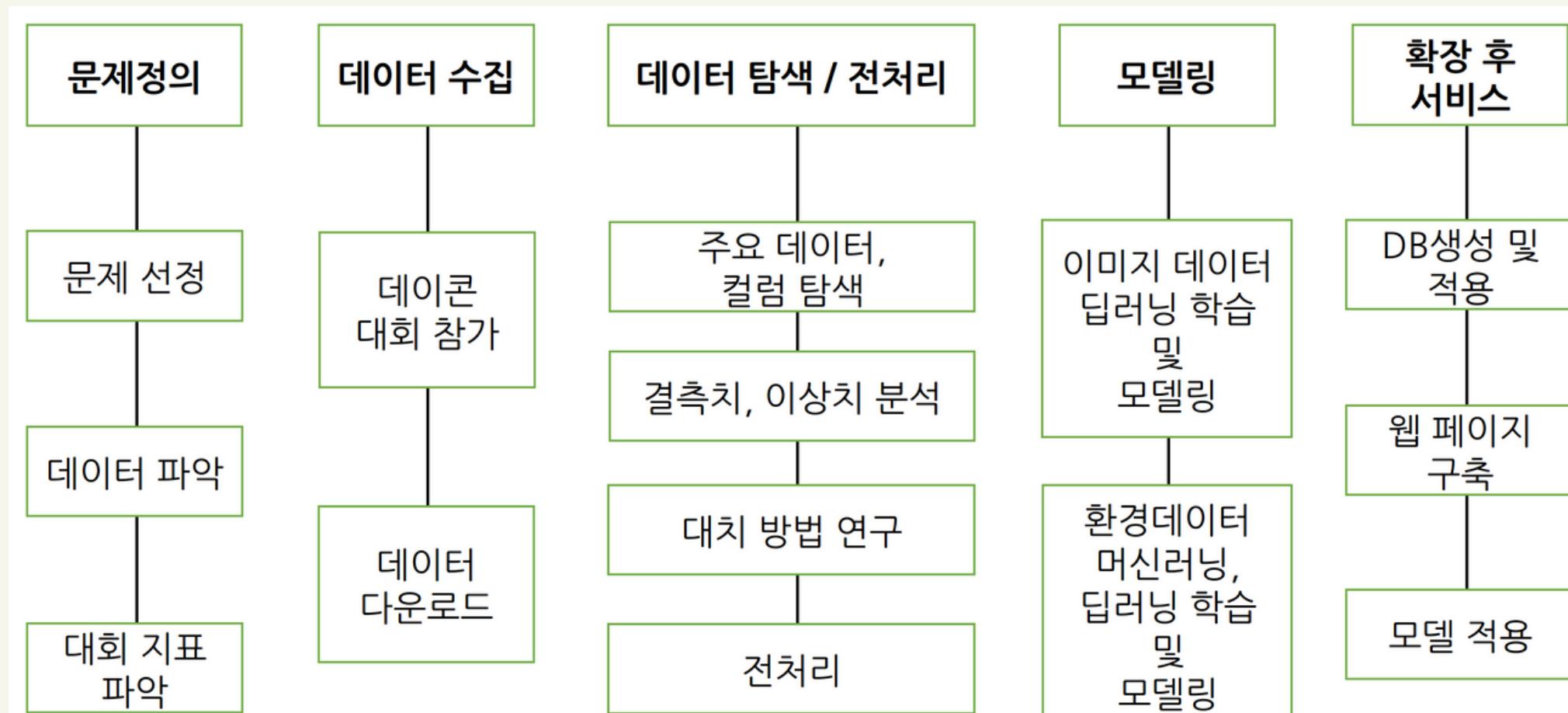
- train 이미지 데이터 예시

- CASE에 따른 시간 순 배치



| 프로젝트 추진체계

Work Breakdown Structure - 업무 분류 체계

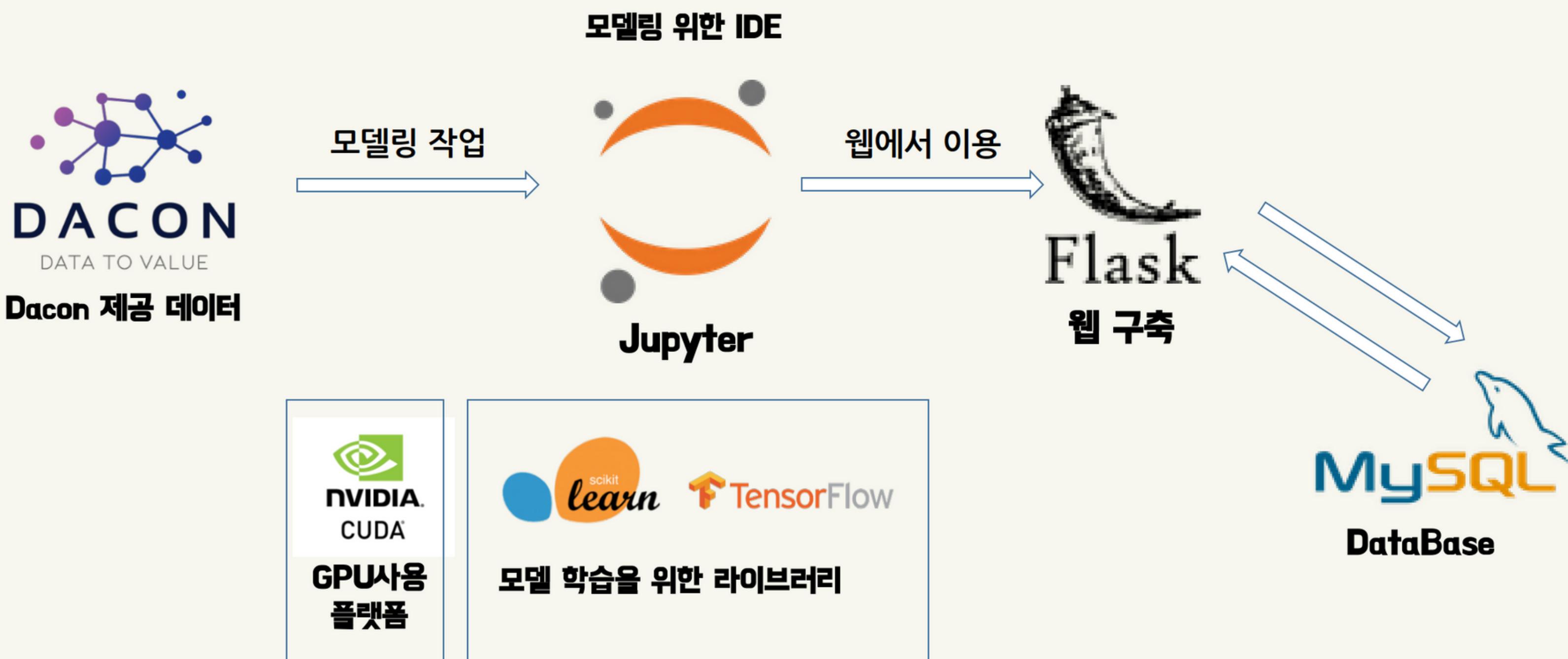


WBS

빅데이터 분석 5단계인 문제 정의, 데이터 수집, 데이터 탐색 및 전처리, 모델링, 시각화를 활용하여 작성하였습니다.

| 프로젝트 추진체계

[플랫폼 아키텍쳐]



| 프로젝트 추진체계



프로젝트 역할 및 임무

[이현경 : 팀장]

- 웹 : login페이지, 메인페이지, tip 페이지작업
- 환경 데이터 모델링

[원종현 : 프로그래머]

- 웹 : 서버, DB작업, 성장일기
- 전처리 및 딥러닝, 머신러닝 전반적인 모델링

[윤혜림 : 프로그래머]

- 웹 : community작업,
- 이미지 딥러닝 작업, 환경데이터 모델링



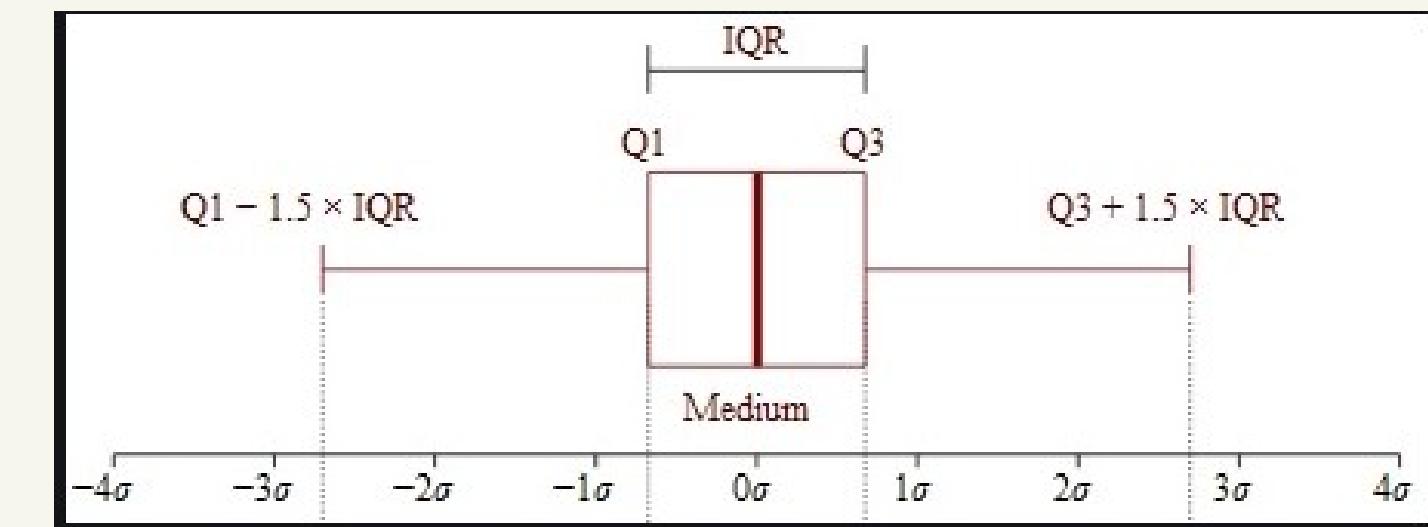
Part 2 :
Modeling

- 데이터 전처리
- 이미지 데이터처리
- 최종 모델링
- 데이콘 대회 진행상황

| Modeling - 데이터 전처리

[방향성 정의]

- 식물은 분 단위, 시간 단위의 환경 영향보다는 하루 평균의 환경 영향을 받는다고 가정하여 각 csv의 데이터의 평균을 하나의 행으로 만들어 분석
- 다음날의 크기를 예측하는 것이므로 현재 크기에 관계성이 크다.
- 이상치 처리의 경우에는 IQR을 이용하여 처리하는 것으로 방향성을 잡음.
- 환경 파라미터와 현재 크기를 이용해 다음날의 크기를 예측해내는 방향으로 설계

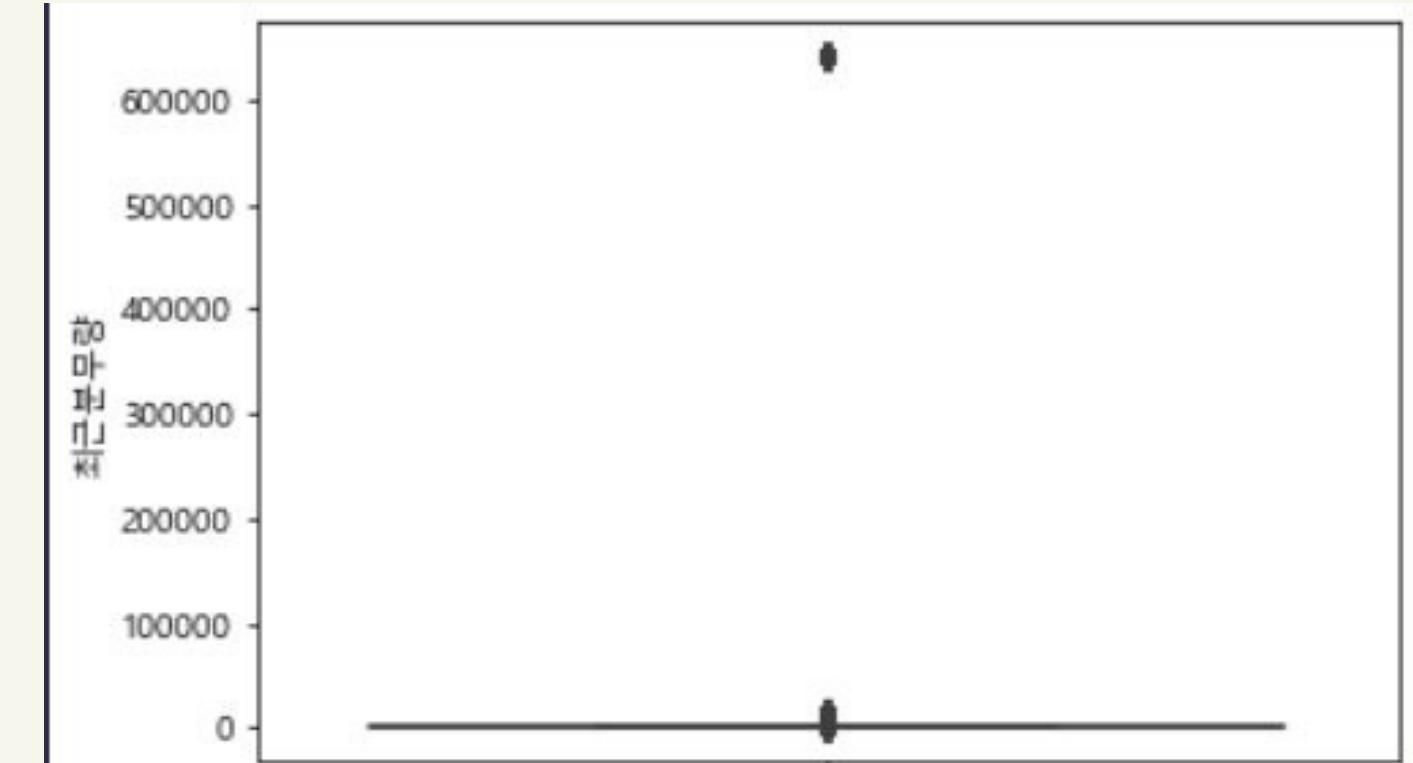


< IQR과 boxplot >

| Modeling - 데이터 전처리

[이상치 처리]

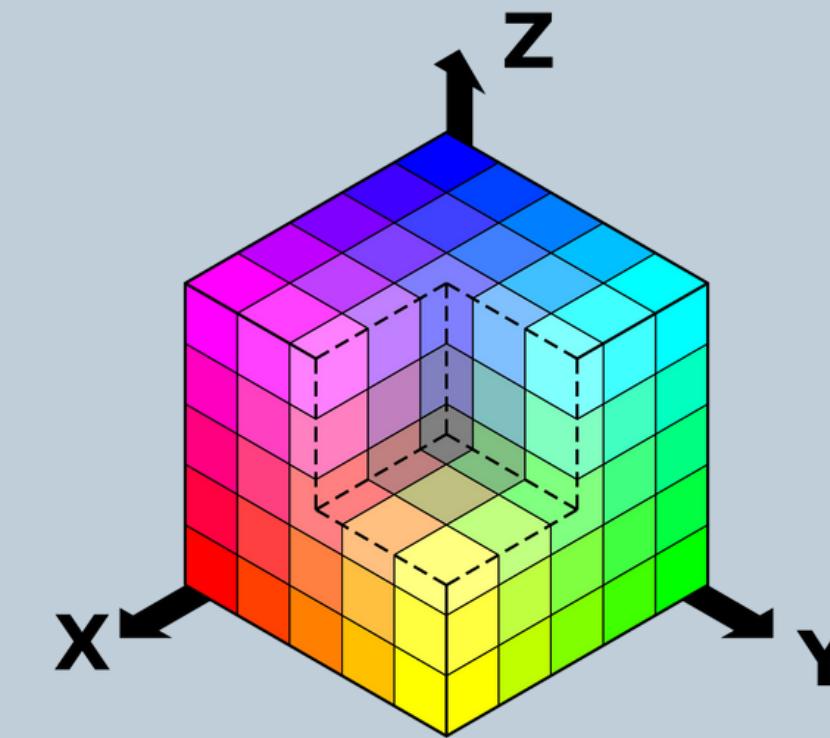
- IQR을 이용하여 이상치를 판단하여 처리하는 방식을 택함.
- 그러나 컬럼의 특성 상 IQR로 판단된 이상치를 모두 다른 수로 대체하는 것은 데이터의 왜곡을 가져올 수 있다고 판단하여 특정 컬럼의 이상치 처리를 다음과 같이하였다.
- 최근분무량은 2만이 넘는 데이터가 잘 없는 컬럼이지만 60만이 넘는 특정데이터가 발견되어 IQR을 활용한 이상치 처리를 진행.
- 내부습도의 경우 100이상이 존재 할 수 없으나 100이상인 데이터가 발견되어 조정
- ec관측치 또한 0이하가 존재할 수 없음에도 0이하의 값이 발견되어 0이하의 값을 0으로 조정



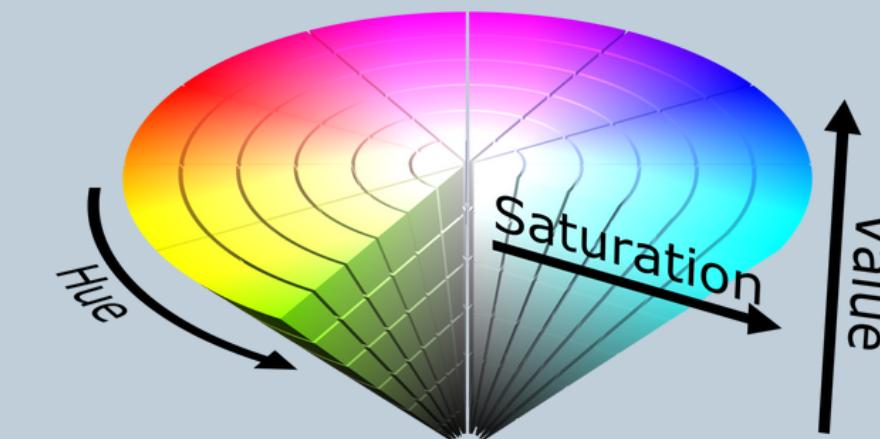
< 최근분무량의 boxplot >

| Modeling - 이미지 데이터 처리

- HSV란?
- 색상(Hue)
 - 색상이란 가시광선 스펙트럼을 원모양으로 배치한 색상환에서 가장 파장이 긴 빨간색을 0° 라고 하였을 때 상대적인 배치 각도를 의미한다.
 - 전체 원은 $0^\circ\sim360^\circ$ 의 범위를 갖고 360° 와 0° 는 결국 같은 색상인 빨강색을 의미한다.
- 채도(Saturation)
 - 채도란 특정 색상의 가장 진한 상태를 100%라 할 때, 진함의 퍼센트 정도를 나타낸다.
- 명도(Value)
 - 명도는 쉽게 말해서 특정 색의 밝은 정도이다.



<RGB 큐브 형태>



<HSV 색공간 형태>

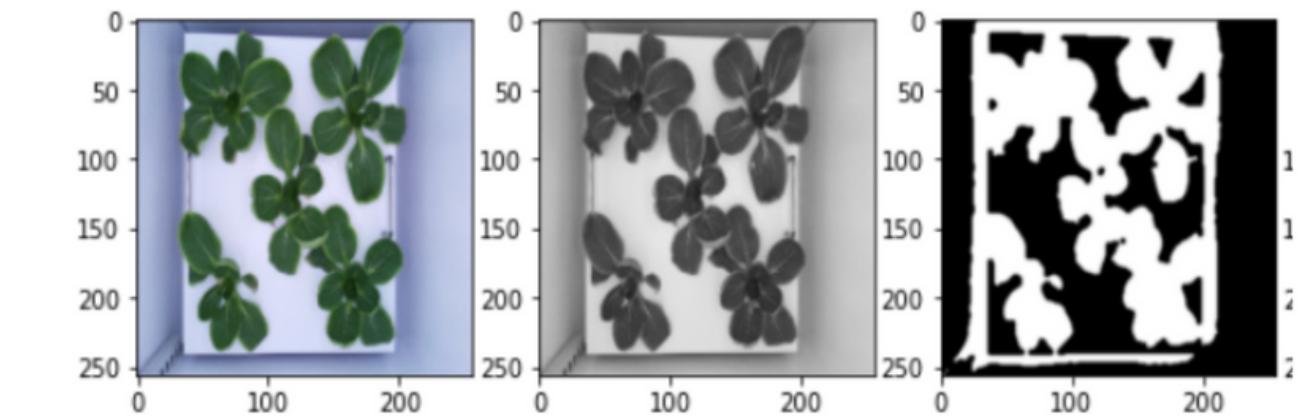
- 한계점
 - 사진상으로 관찰이 불가능한 가려진 잎의 면적은 예측할 수 없다.

| Modeling - 이미지 데이터 처리

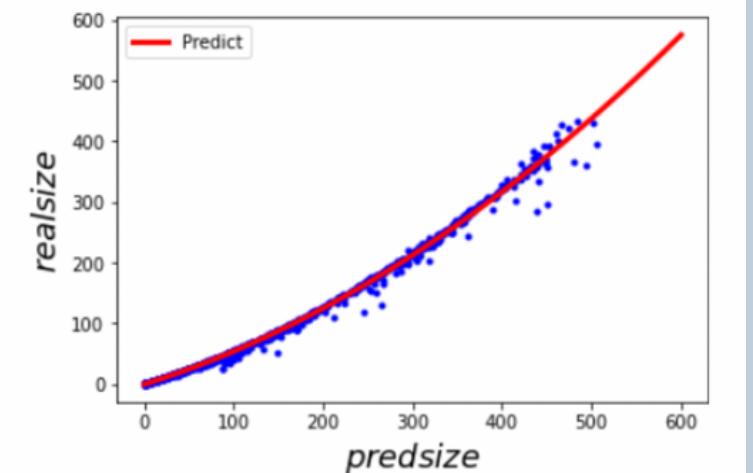
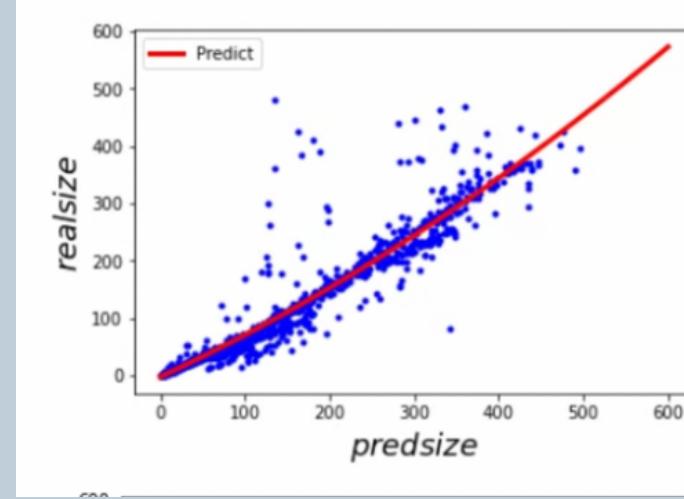
- 기획한 대로 이미지의 RGB값을 이용한 분석을 통해 이미지를 마스킹 처리를 실시.
- 그러나 RGB 값으로는 특정 색상 영역을 지정하여도 식물(청경채)과 배경을 정확하게 구분할 수 없는 한계가 존재
- 해결 방안으로 RGB이미지를 HSV 이미지로 변환한 뒤, 스펙트럼 중심 점과 검출할 범위를 파라미터로 정해 픽셀 수를 추출해 이를 이용하여 실제 잎의 면적과 비교, 다항회귀(polynomial regression)을 통해 가장 좋은 마스킹 파라미터 값을 도출.
- 다음과 같은 결과를 도출해 내었고 사진 상으로 판별 가능한 잎의 면적과 약 98% 유사한 값을 얻어낼 수 있었음

>> 이러한 마스킹 작업을 통해 실제 잎 크기와 유사한 수치를 추출 할 수 있는 최적의 회귀 모델 구현

- 예시 2) rgb로 마킹한 이미지



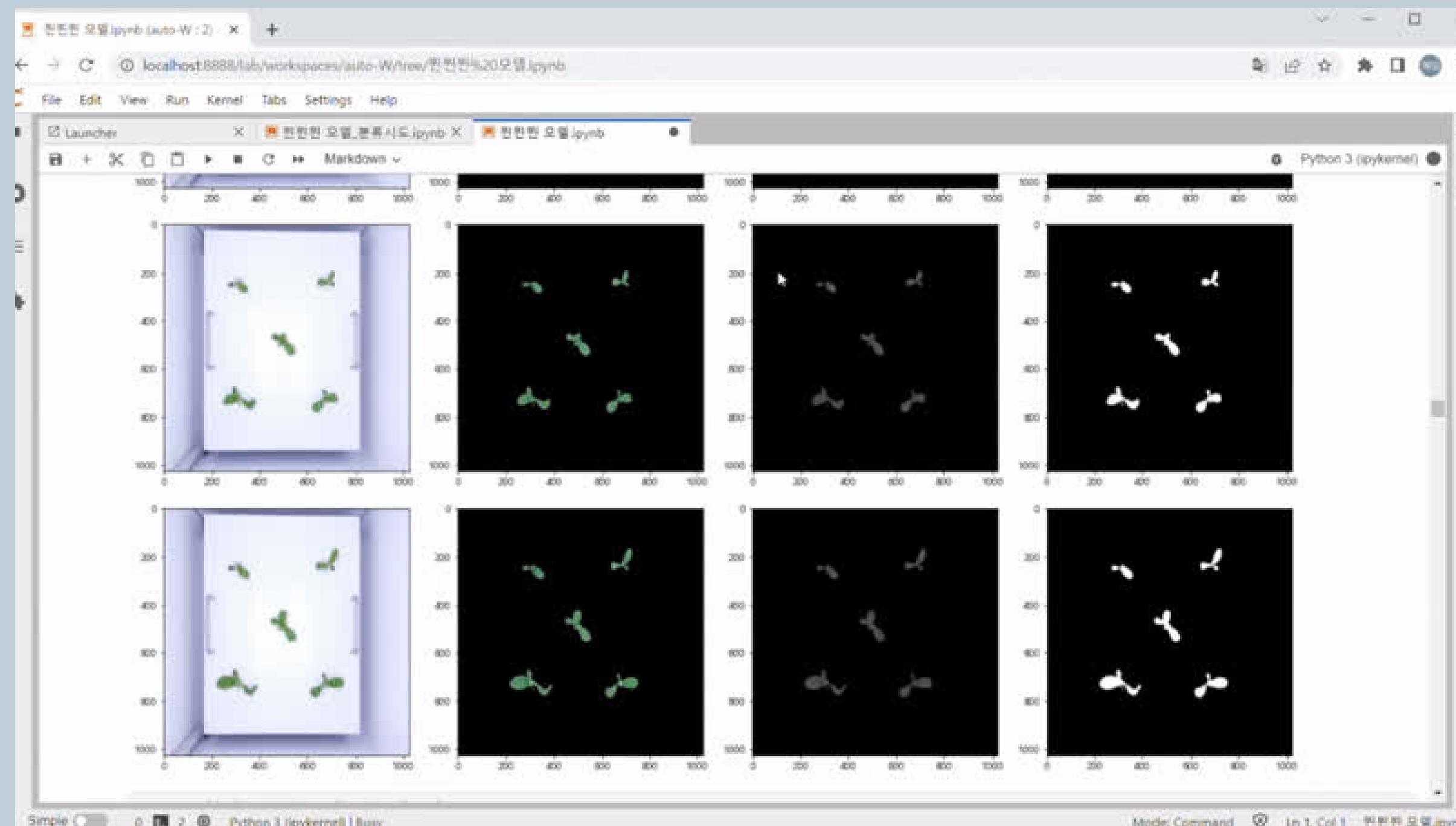
< 파라미터 최적화, 이상치 제거 전 >



< 파라미터 최적화 후 >

| Modeling - 이미지 데이터 처리

[image data 처리 완성본]

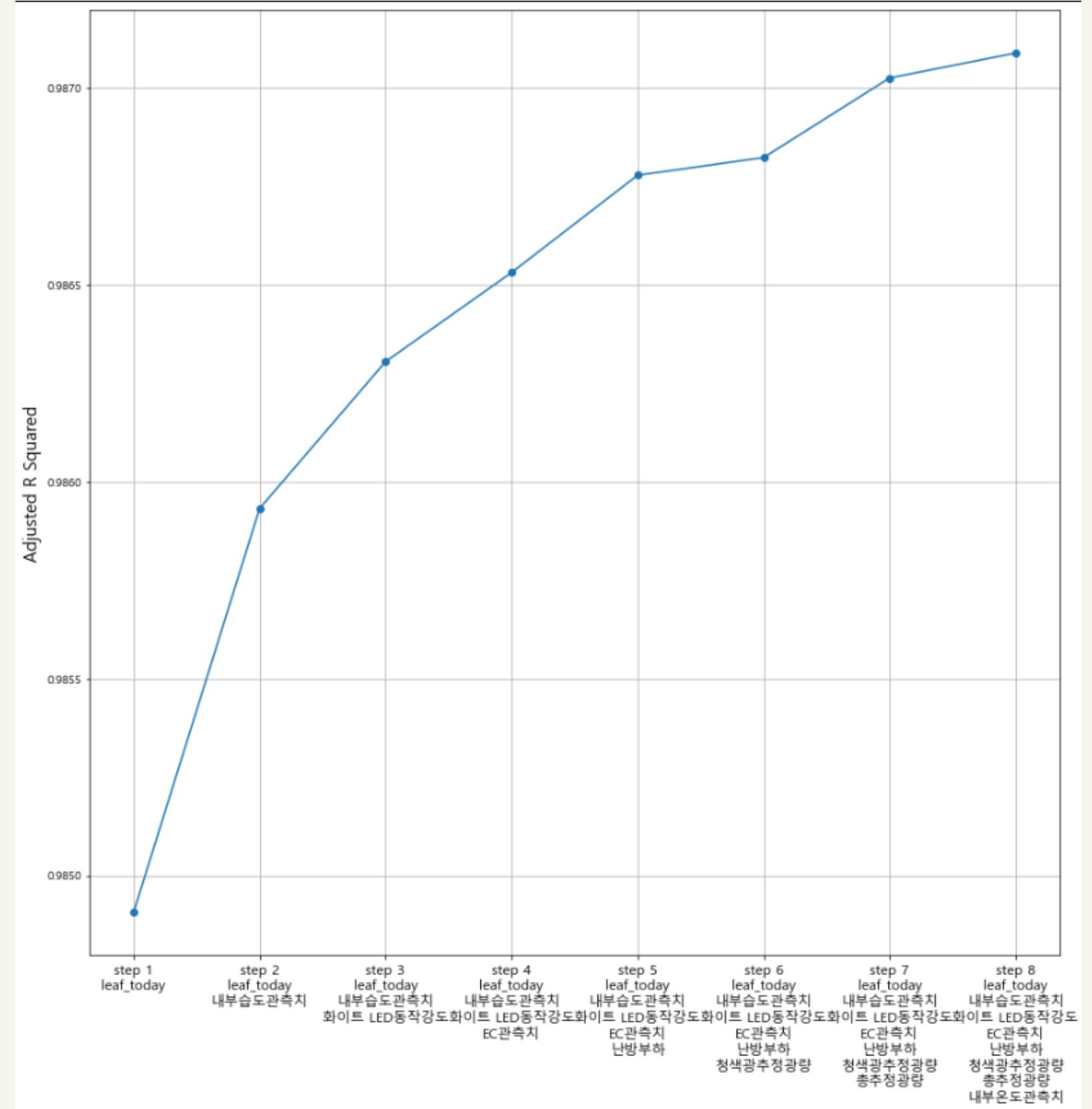


| Modeling - 최종 모델링

[Feature Engineering]

모델의 정확도 향상을 위해, Feature Engineering을 두단계로 진행

1. 직접 독립변수간의 관계를 파악해 선정
2. 단계선택법(Forward Stepwise Selection)으로 선정
 - 전진선택법
 - 변수의 개수가 많은 경우에도 사용가능
 - 변수값의 작은 변동에도 그 결과가 크게 달라져 안정성이 부족
 - 후진제거법
 - 전체 변수들의 정보를 이용하는 장점이 존재
 - 변수의 개수가 많은 경우 사용하기 어려움
 - 단계선택법
 - 두 단계의 장점을 섞은 선택법



| Modeling - 최종 모델링

[Feature Engineering]-self choice

내부온도관측치	외부온도관측치	내부습도관측치	외부습도관측치
CO2관측치	EC관측치	최근분무량	화이트 LED 동작강도
레드 LED 동작강도	블루 LED 동작강도	냉방온도	냉방부하
난방온도	난방부하	총추정광량	총추정광량
적색광추정광량	청색광추정광량		

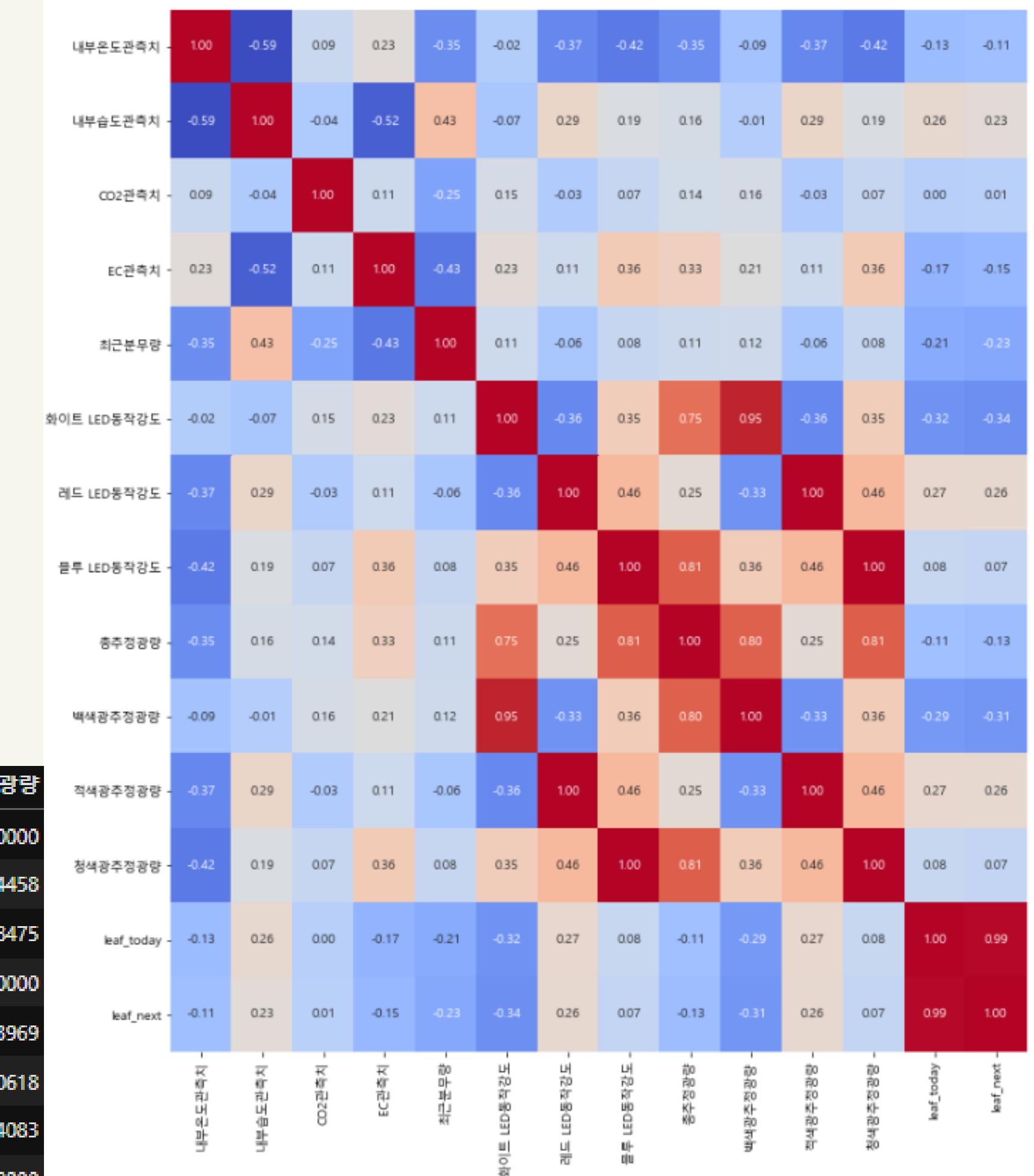
- 난방부하, 냉방부하, 난방온도, 냉방부하, 외부온도관측치
--> 내부온도관측치로 설명가능하므로, 제거
- 외부습도관측치
--> 내부습도관측치가 있으므로 제거

| Modeling - 최종 모델링

[Feature Engineering]- self choice

- LED동작강도와 광추정광량은 약 1의 상관관계를 가짐
- 총추정광량=백색+적색+청색추정광량.
- CO2관측치는 다음날 잎크기에 영향을 거의 끼치지 않음
- 현재 잎크기는 다음날 잎크기와 굉장한 연관성을 가짐(0.99계수)
- 레드LED와 적색광추정광량은 나머지feature들과 거의 유사한 상관관계
- 블루LED와 청색광추정광량은 나머지feature들과 거의 유사한 상관관계
- 블루LED는 다음날 잎크기에 영향을 거의 끼치지 않음

	광량의합	총추정광량
count	1318.000000	1318.000000
mean	160.135616	160.134458
std	63.278843	63.278475
min	0.000000	0.000000
25%	126.803969	126.803969
50%	148.080618	148.080618
75%	179.494083	179.494083
max	565.348000	565.348000



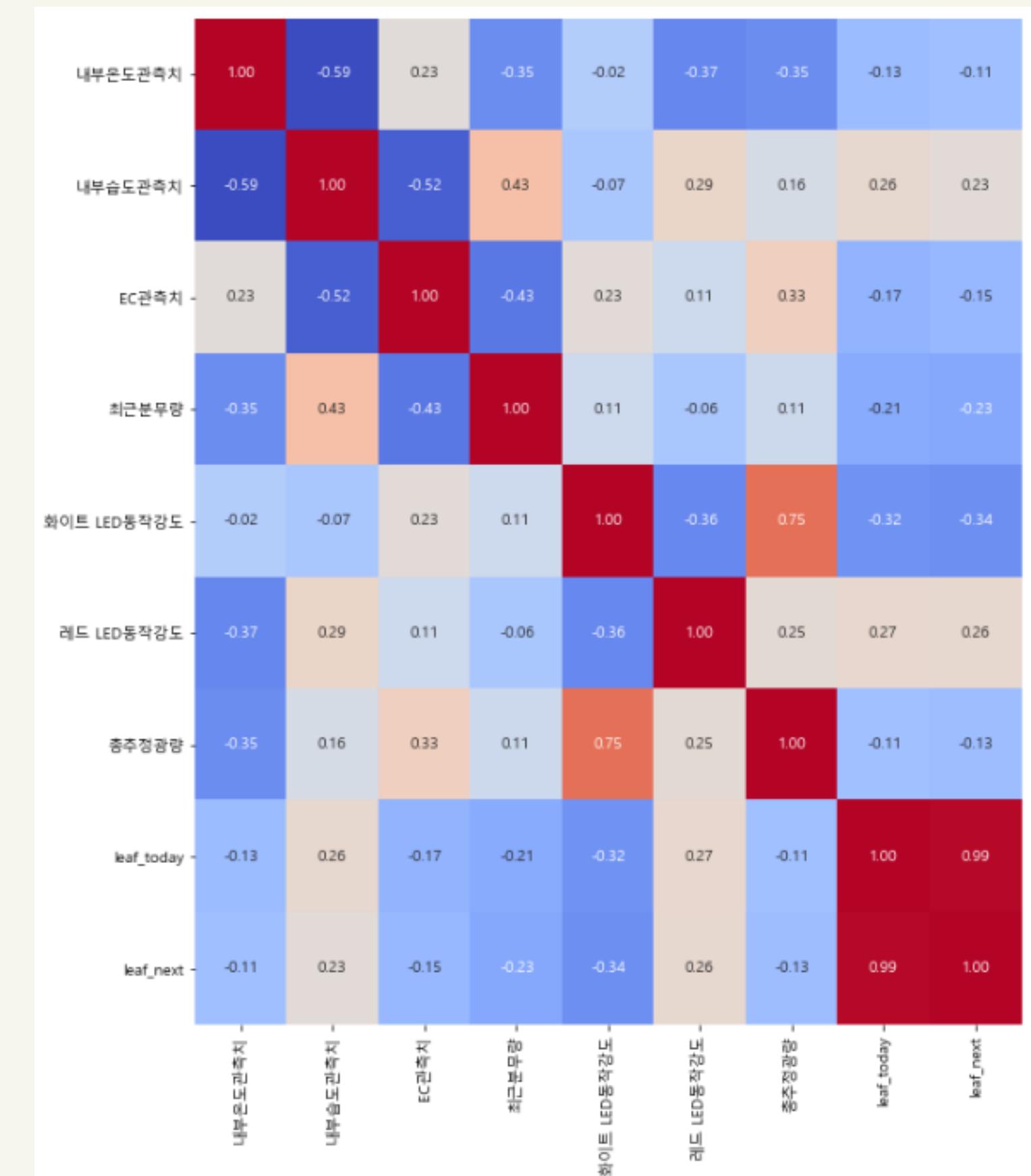
<상관분석 진행 - Heatmap>

| Modeling - 최종 모델링

[Feature Engineering]- self choice

OLS Regression Results													
Dep. Variable:	leaf_next	R-squared:	0.986										
Model:	OLS	Adj. R-squared:	0.986										
Method:	Least Squares	F-statistic:	1.209e+04										
Date:	Wed, 27 Apr 2022	Prob (F-statistic):	0.00										
Time:	21:21:15	Log-Likelihood:	-5244.1										
No. Observations:	1338	AIC:	1.051e+04										
Df Residuals:	1329	BIC:	1.055e+04										
Df Model:	8												
Covariance Type:	nonrobust												
	coef	std err	t	P> t	[0.025	0.975]							
Intercept	22.7476	6.882	3.305	0.001	9.247	36.248							
내부온도관측치	-0.0636	0.202	-0.315	0.752	-0.459	0.332							
내부습도관측치	-0.1822	0.039	-4.718	0.000	-0.258	-0.106							
EC관측치	2.5699	0.442	5.811	0.000	1.702	3.438							
최근분무량	0.0003	0.000	1.509	0.132	-9.38e-05	0.001							
화이트LED	-0.2256	0.031	-7.187	0.000	-0.287	-0.164							
레드LED	-0.1277	0.033	-3.913	0.000	-0.192	-0.064							
총추정광량	0.0056	0.007	0.783	0.434	-0.008	0.020							
leaf_today	1.0780	0.004	267.167	0.000	1.070	1.086							
Omnibus:	443.339	Durbin-Watson:	0.853										
Prob(Omnibus):	0.000	Jarque-Bera (JB):	13372.293										
Skew:	-0.905	Prob(JB):	0.00										
Kurtosis:	18.381	Cond. No.	5.16e+04										

- 내부온도관측치
- 내부습도관측치
- EC관측치
- 최근분무량
- 화이트LED
- 레드LED
- 총추정광량
- leaf_today



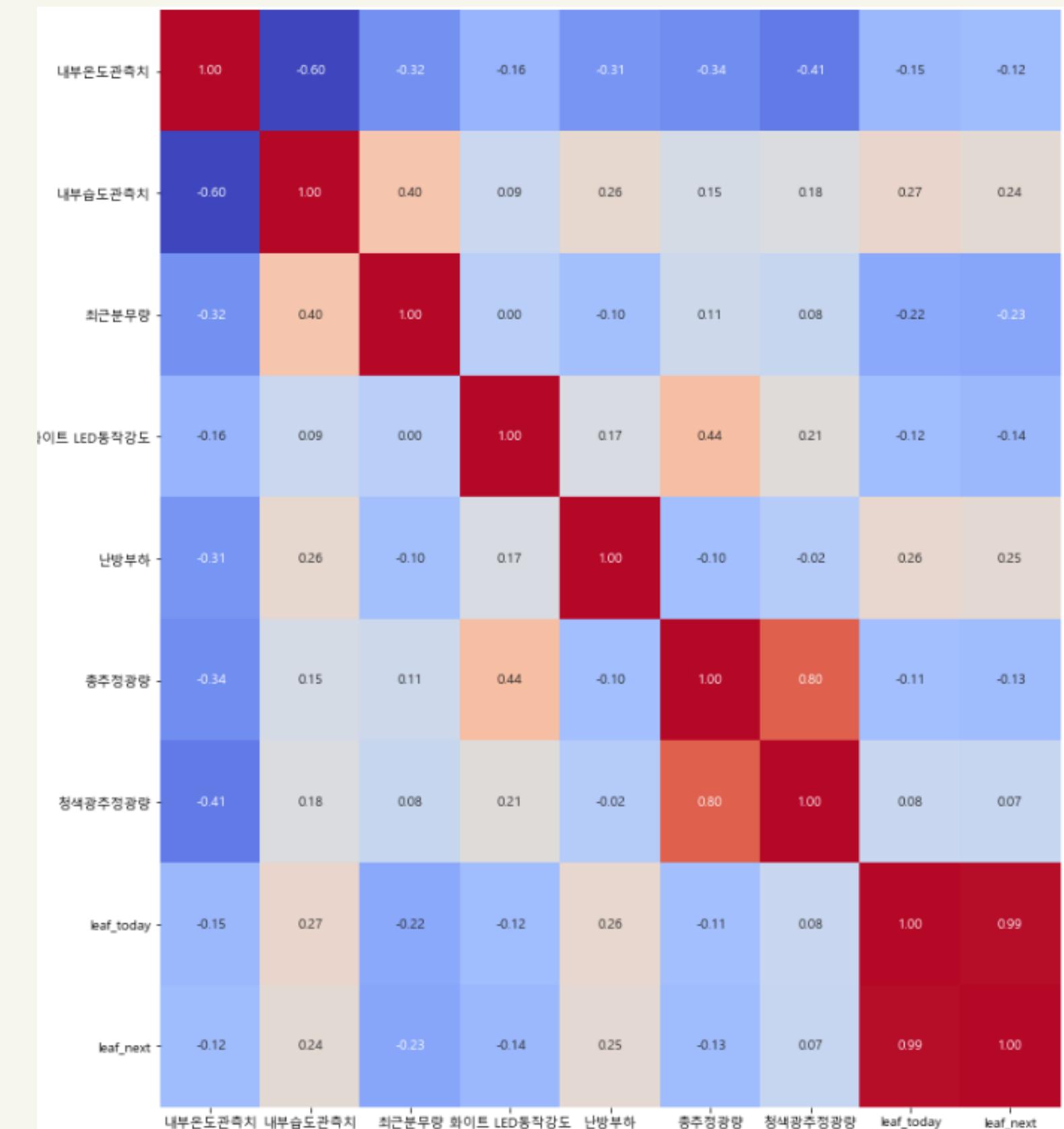
<상관분석 진행 - Heatmap>

| Modeling - 최종 모델링

[Feature Engineering]- 단계선택법

OLS Regression Results													
Dep. Variable:	leaf_next	R-squared:	0.987										
Model:	OLS	Adj. R-squared:	0.987										
Method:	Least Squares	F-statistic:	1.240e+04										
Date:	Wed, 27 Apr 2022	Prob (F-statistic):	0.00										
Time:	22:56:36	Log-Likelihood:	-5227.3										
No. Observations:	1338	AIC:	1.047e+04										
Df Residuals:	1329	BIC:	1.052e+04										
Df Model:	8												
Covariance Type:	nonrobust												
	coef	std err	t	P> t	[0.025	0.975]							
Intercept	28.2512	6.823	4.141	0.000	14.866	41.636							
내부온도관측치	-0.3674	0.200	-1.834	0.067	-0.760	0.026							
내부습도관측치	-0.1791	0.034	-5.241	0.000	-0.246	-0.112							
EC관측치	2.7778	0.346	8.036	0.000	2.100	3.456							
화이트LED	-0.3394	0.038	-8.901	0.000	-0.414	-0.265							
난방부하	-0.5915	0.094	-6.268	0.000	-0.777	-0.406							
총추정광량	0.0624	0.014	4.391	0.000	0.035	0.090							
청색광추정광량	-0.1612	0.032	-5.097	0.000	-0.223	-0.099							
leaf_today	1.0794	0.004	283.765	0.000	1.072	1.087							
Omnibus:	395.975	Durbin-Watson:	0.868										
Prob(Omnibus):	0.000	Jarque-Bera (JB):	11073.601										
Skew:	-0.757	Prob(JB):	0.00										
Kurtosis:	17.012	Cond. No.	4.25e+03										

- 내부온도관측치
- 내부습도관측치
- EC관측치
- 난방부하
- 화이트 LED
- 청색광추정광량
- 총추정광량
- leaf_today



<상관분석 진행 - Heatmap>

| Modeling - 최종 모델링

[Feature Engineering]

<최종 사용 데이터 정보>

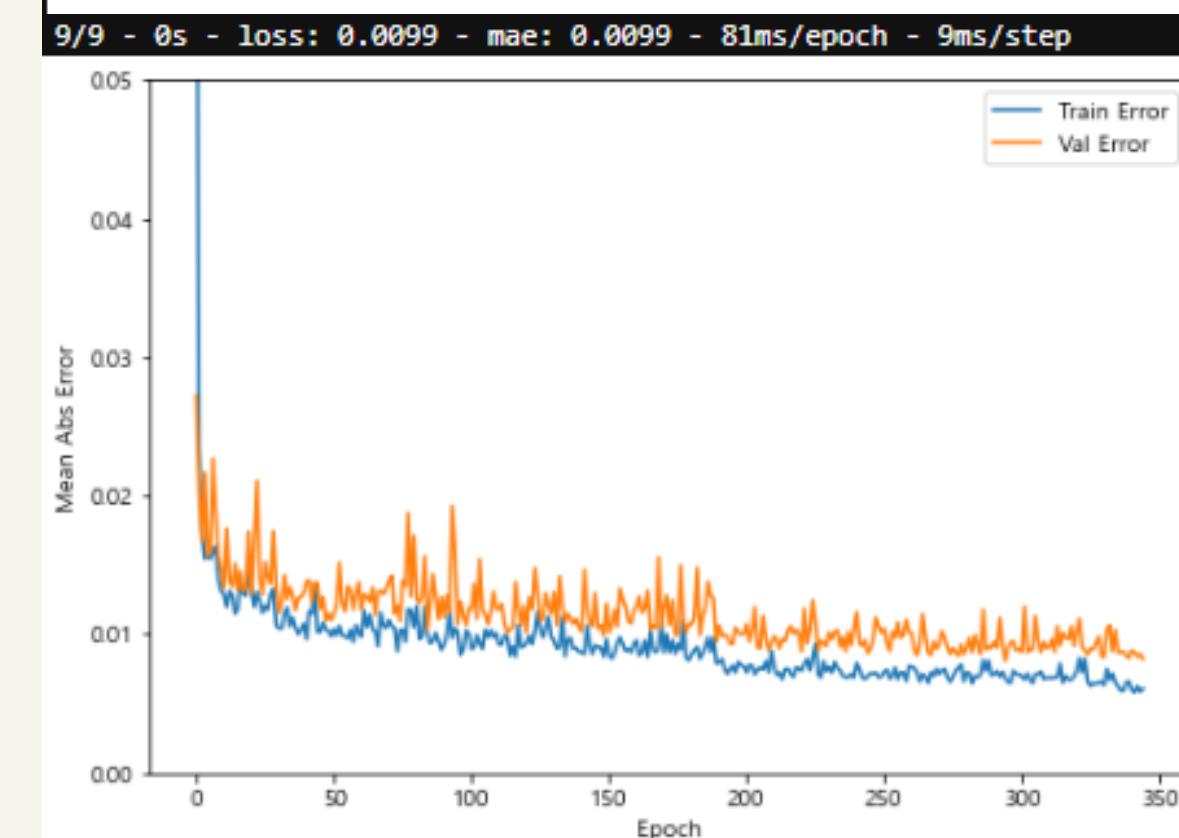
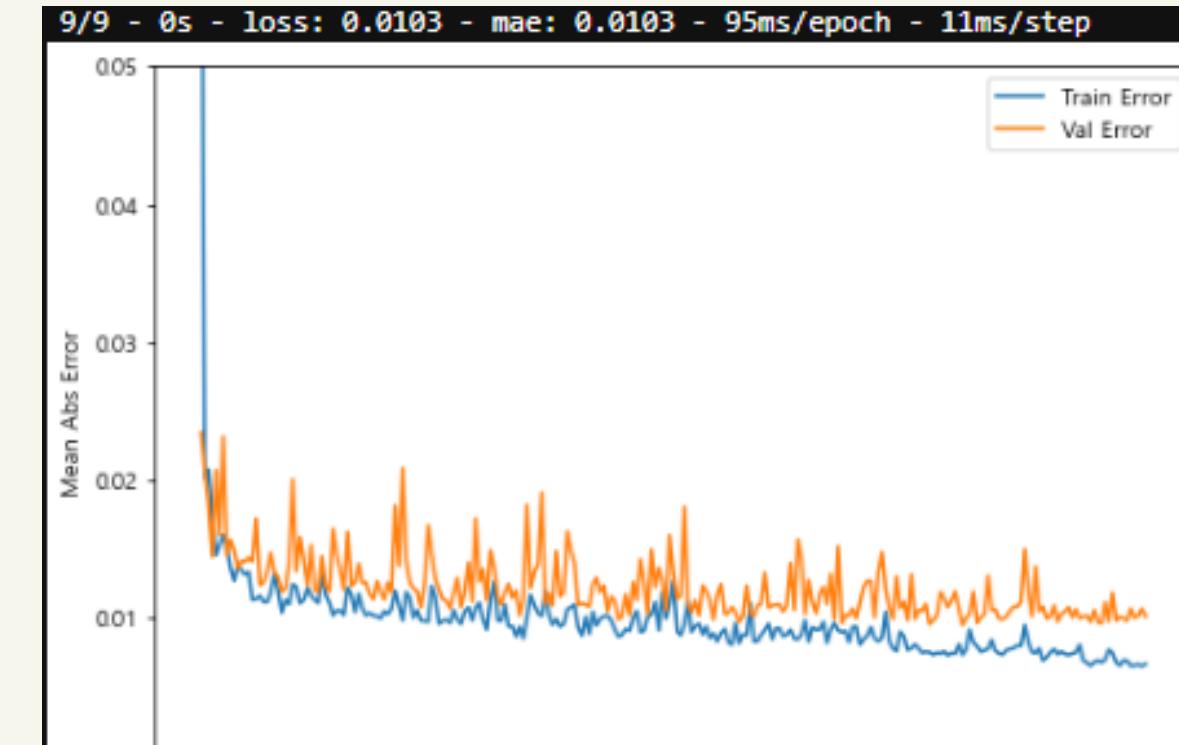
self choice한 피쳐 사용

X_train : 10707개, X_test : 2687개

- 내부온도관측치 - 내부습도관측치 - EC관측치
- 최근분무량 - 화이트LED - 레드LED
- 총추정광량 - leaf_today

y_train : 10707개, y_test: 2687개

- 다음날 잎 면적



| Modeling - 최종 모델링

[정규화 작업]

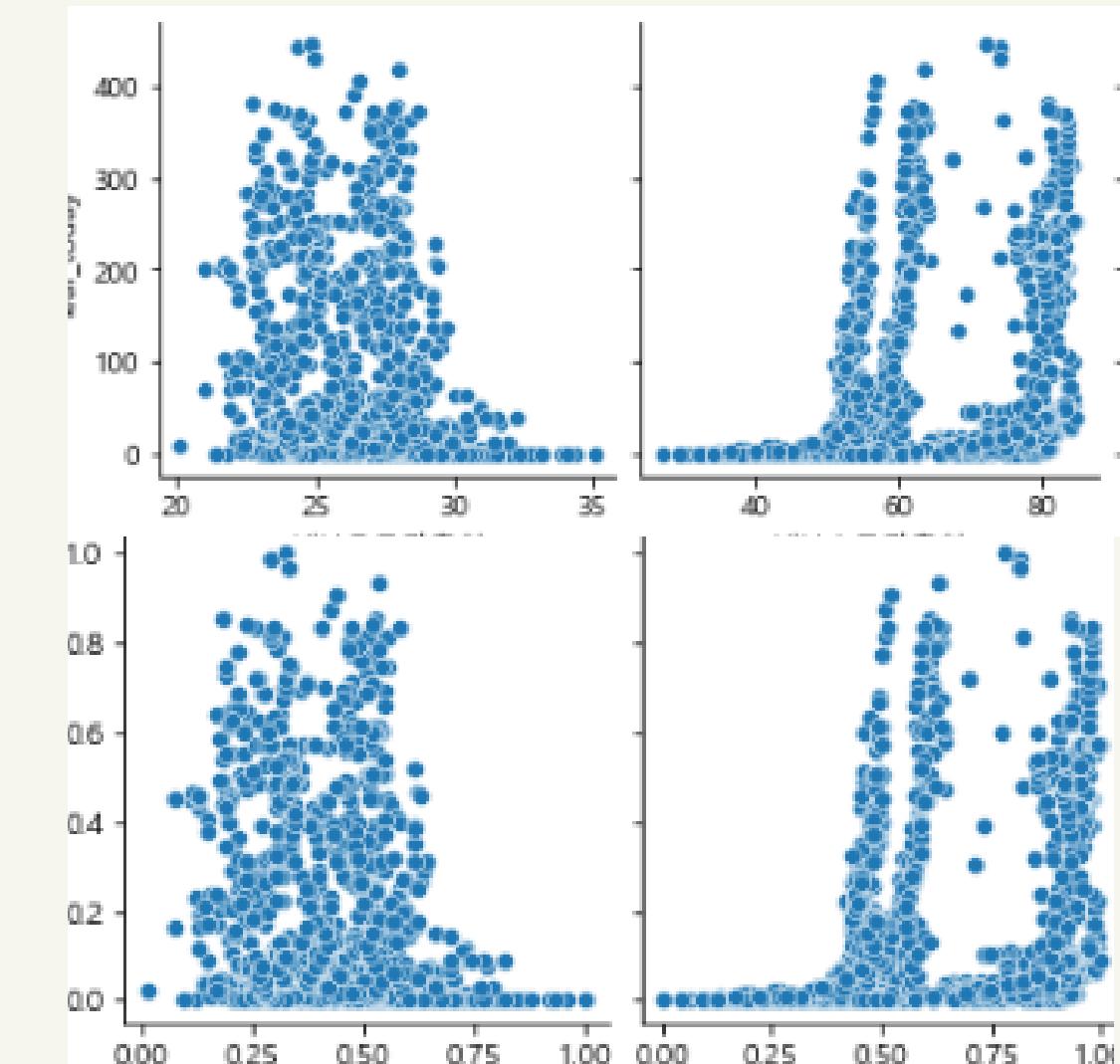
- 피처 스케일링(Feature scaling)

- 데이터 전처리를 거친 각 데이터들(학습용, 테스트용, 검증용)을 컬럼 별 단위 또는 범위를 통일시켜 주기 위해

MinMaxScaler로 정규화(normalization) 처리

	내부온도관측치	내부습도관측치	EC관측치	최근분무량	화이트 LED등작강도	레드 LED등작강도	총추정광량	leaf_today
mean	26.256048	63.509567	1.557022	1449.862598	45.279839	11.391860	159.940183	72.898913
std	2.320751	13.261884	2.569362	2041.299849	24.304300	26.242355	62.825192	97.005714
min	20.131458	26.961806	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	35.073125	84.811875	23.871458	12710.043735	201.000000	201.000000	565.348000	445.099687

	내부온도관측치	내부습도관측치	EC관측치	최근분무량	화이트 LED등작강도	레드 LED등작강도	총추정광량	leaf_today
mean	0.417560	0.631767	0.065225	0.050226	0.225273	0.056676	0.282906	0.163781
std	0.153305	0.229246	0.107633	0.070714	0.120917	0.130559	0.111127	0.217942
min	0.012980	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	0.440298	1.000000	1.000000	1.000000	1.000000



| Modeling - 최종 모델링

[Modeling - 평가지표]

대회의 engineering된 feature와 label로 ML(Machine Learning), DL(DeepLearning)학습을 별개로 진행 후,
검증데이터로 진행한 MAE(mean absolute error), NMAE(Normalized Mean Absolute Error)점수를 토대로,
최종모델 결정

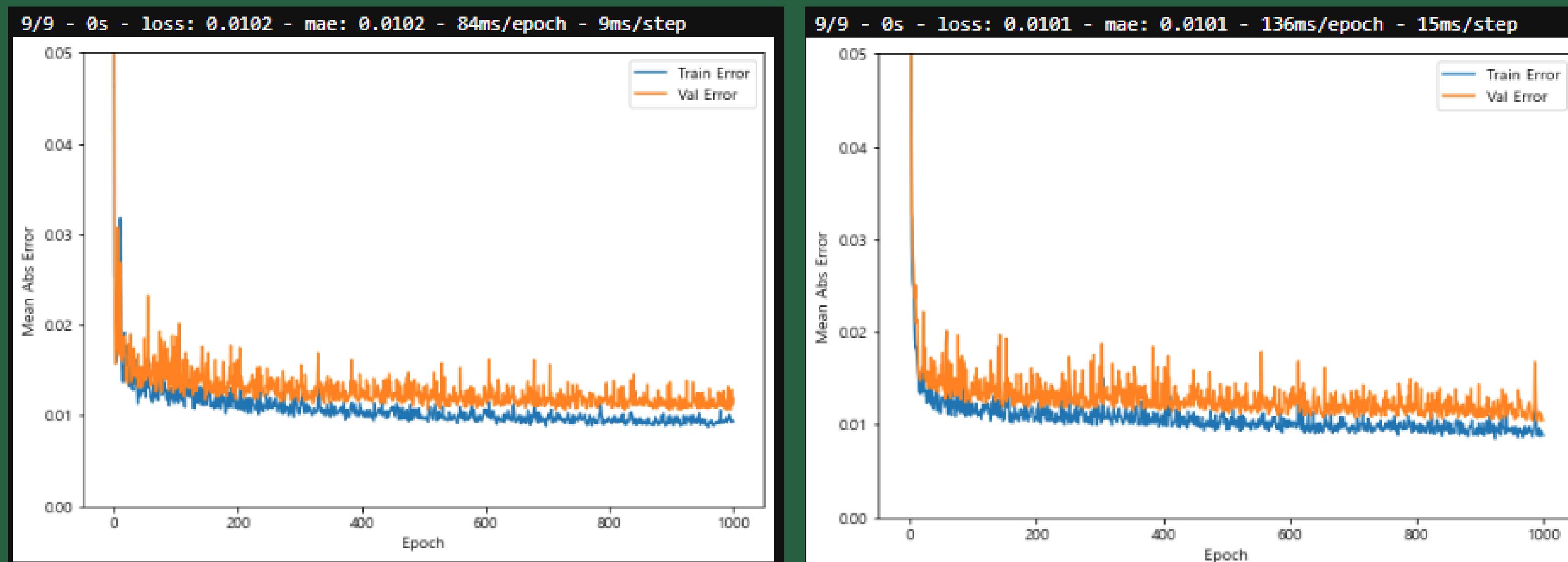
[MAE, NMAE]

$$\text{MAE} = (1/n) \sum |true - pred|$$

$$\text{NMAE} = \text{MAE} / (1/n) \sum |true|$$

| Modeling - Deep Learning

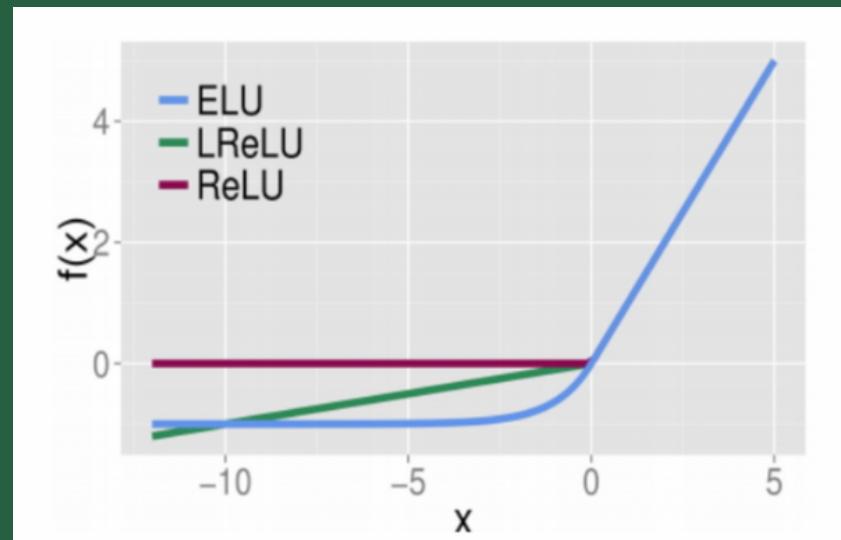
[레이어 설계]



(좌:10개의 히든레이어_낮은노드수, 우: 3개의 히든레이어_높은노드수)

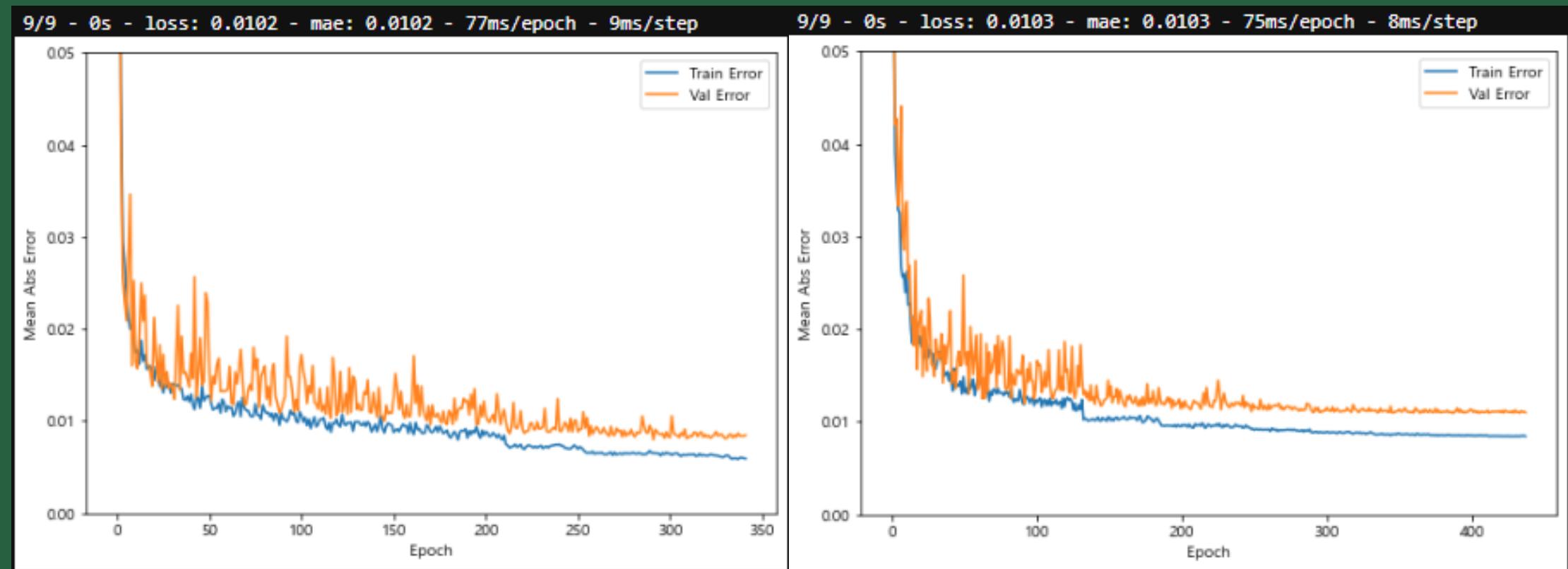
| Modeling - Deep Learning

[activation 선정]



< 일반적으로 회귀에서 많이 사용하는 activation 함수 >

- **Relu**의 경우, input값이 0보다 작을경우, **gradient가 0이 되버려** 이를 지나는 모든 노드가 죽어버리는 단점이 존재
- 이를 보완한 모델인 **ELU**과 직접 비교하였다.

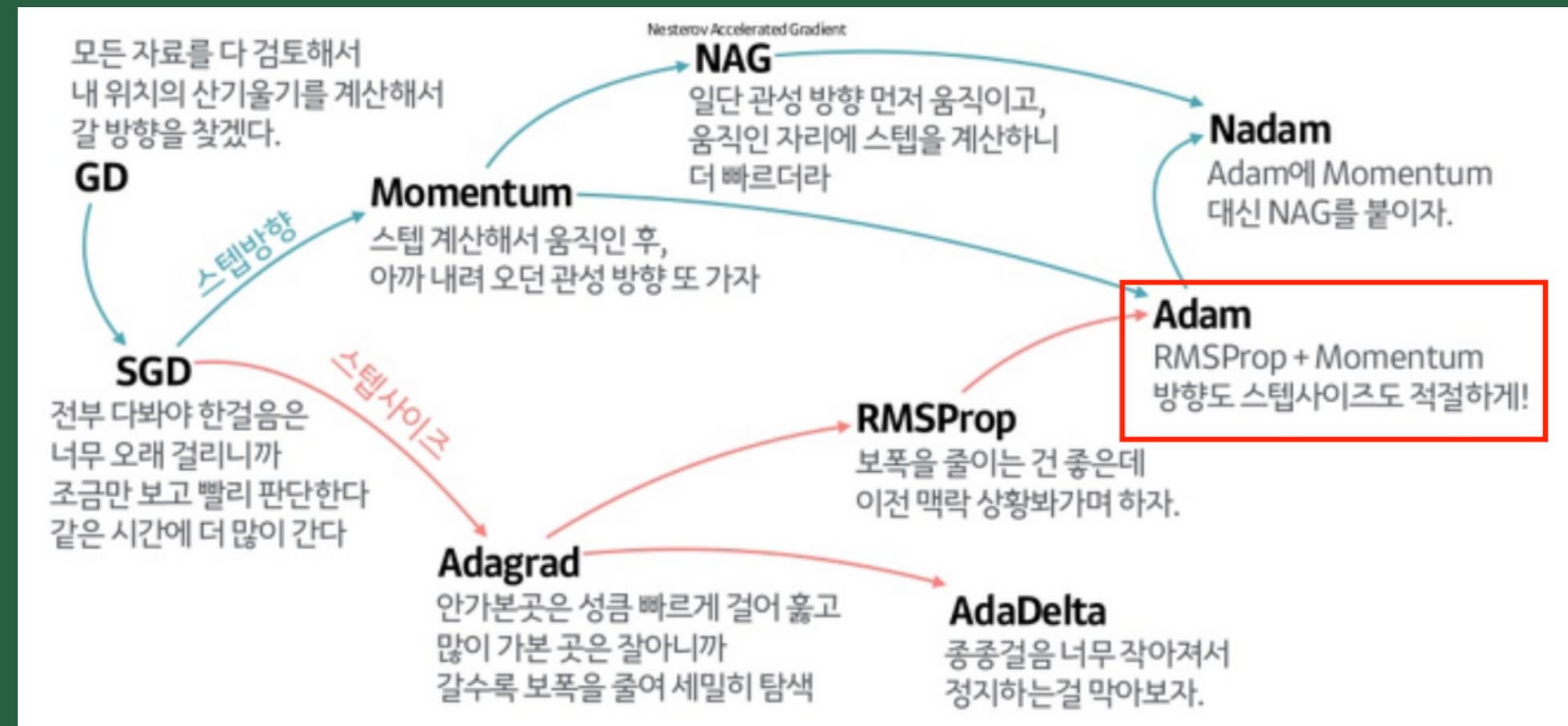


(좌: relu, 우:elu)

relu의 경우, 조금 더 적은 epoch에서 거의 동일한 mae값을 도출해냄

| Modeling - Deep Learning

[loss, optimizer 설정]



loss(손실함수)로는 대회 채점 점수에 직접적으로 관계가 있는 mae를 선택하였다.

optimizer(최적화 알고리즘)으로는 일반적으로 회귀 모델에서 많이 사용되는 Adam을 사용하였다.

| Modeling - Deep Learning

```
with tf.device('/CPU:0'):
    tf.random.set_seed(42)
    model = Sequential()
    model.add(Dense(16, input_dim=len(normed_train_df.columns), activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(16, activation='relu'))
    model.add(Dense(1))

    model.compile(loss='mean_absolute_error',
                  optimizer='adam',
                  metrics=['mae'])

    # 모델 업데이트 및 저장
    cp = ModelCheckpoint(filepath='bbbb_model.h5', monitor='val_mae', verbose=0, save_best_only=True, mode = 'min')
    # 학습 과정 중단 설정
    es = EarlyStopping(monitor='val_mae', patience=50, mode='min')
    rlrp = ReduceLROnPlateau(monitor='val_mae', factor=0.6, patience=40, mode='min')

    history=model.fit(normed_train_df, normed_train_label_df, validation_split=0.2, epochs=1000, batch_size=32, verbose=0, callbacks=[es, cp, rlrp])
    best_model=keras.models.load_model('bbbb_model.h5')
    normed_plot_history(history)
    mae, mse = best_model.evaluate(normed_val_df, normed_val_label_df, verbose=2)
```

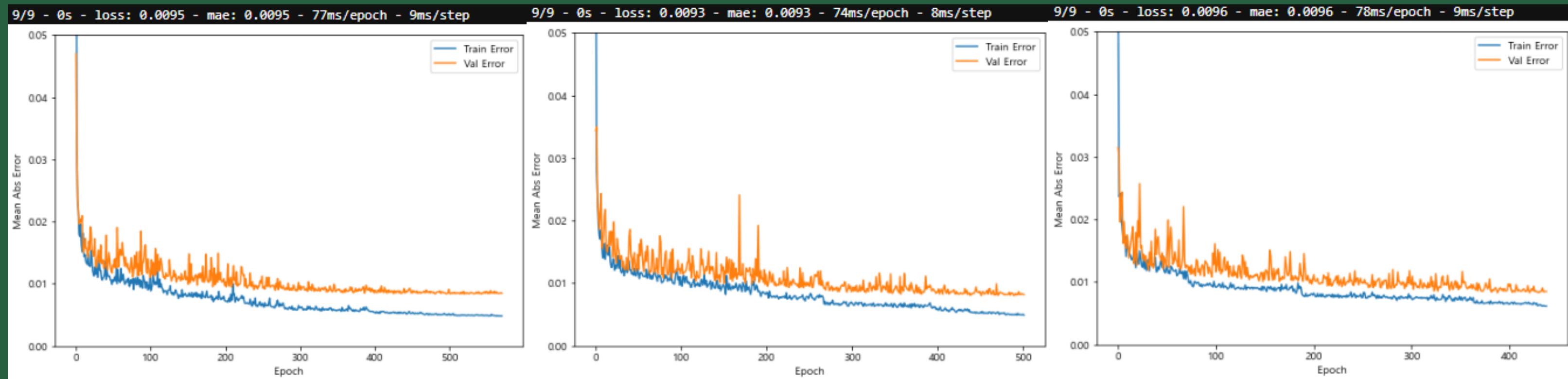
최종적으로 사용된 모델(세부 파라미터 수정전)

단계선택법으로 채택된 feature에 대해 검증 mae : 0.102 도출

직접고른 feature에 대해 검증 mae : 0.0093 도출

| Modeling - Deep Learning

[딥러닝 모델 MAE 평가 지표 결과] -파라미터수정



batch 사이즈 조절

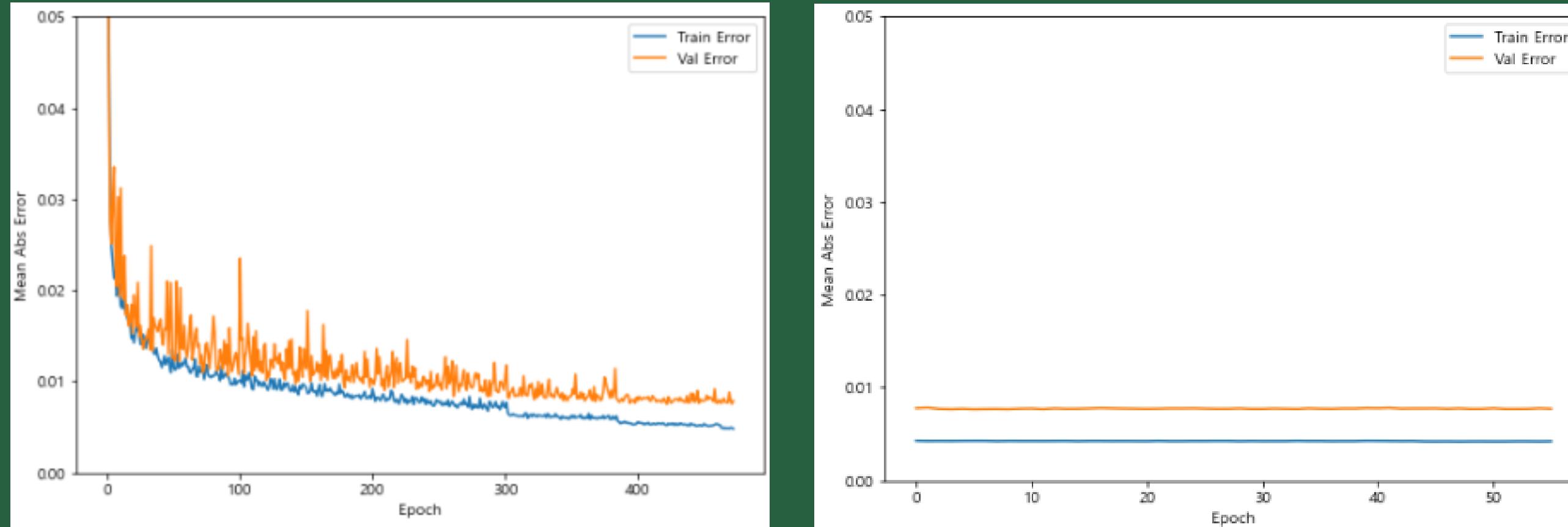
batch 16 : mae 0.0095 도출

batch 8 : mae 0.0093 도출 --> 가장 좋은수치 도출

batch 4 : mae 0.0096 도출

| Modeling - Deep Learning

[딥러닝 모델 MAE 평가 지표 결과] -Kfold



```
9/9 - 0s - loss: 0.0093 - mae: 0.0093 - 89ms/epoch - 10ms/step
9/9 - 0s - loss: 0.0096 - mae: 0.0096 - 79ms/epoch - 9ms/step
9/9 - 0s - loss: 0.0096 - mae: 0.0096 - 92ms/epoch - 10ms/step
9/9 - 0s - loss: 0.0096 - mae: 0.0096 - 84ms/epoch - 9ms/step
9/9 - 0s - loss: 0.0096 - mae: 0.0096 - 94ms/epoch - 10ms/step
```

| Modeling - Deep Learning

[딥러닝 모델 MAE 평가 지표 결과] -레이어 수정

```
model.add(Dense(16, input_dim=len(normed_train_df.columns), activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(16, activation='relu'))
model.add(Dense(1))
```

```
model.add(Dense(20, input_dim=len(normed_train_df.columns), activation='relu'))
model.add(Dense(40, activation='relu'))
model.add(Dense(80, activation='relu'))
#model.add(Dropout(0.3))
model.add(Dense(40, activation='relu'))
model.add(Dense(20, activation='relu'))
model.add(Dense(1))
```

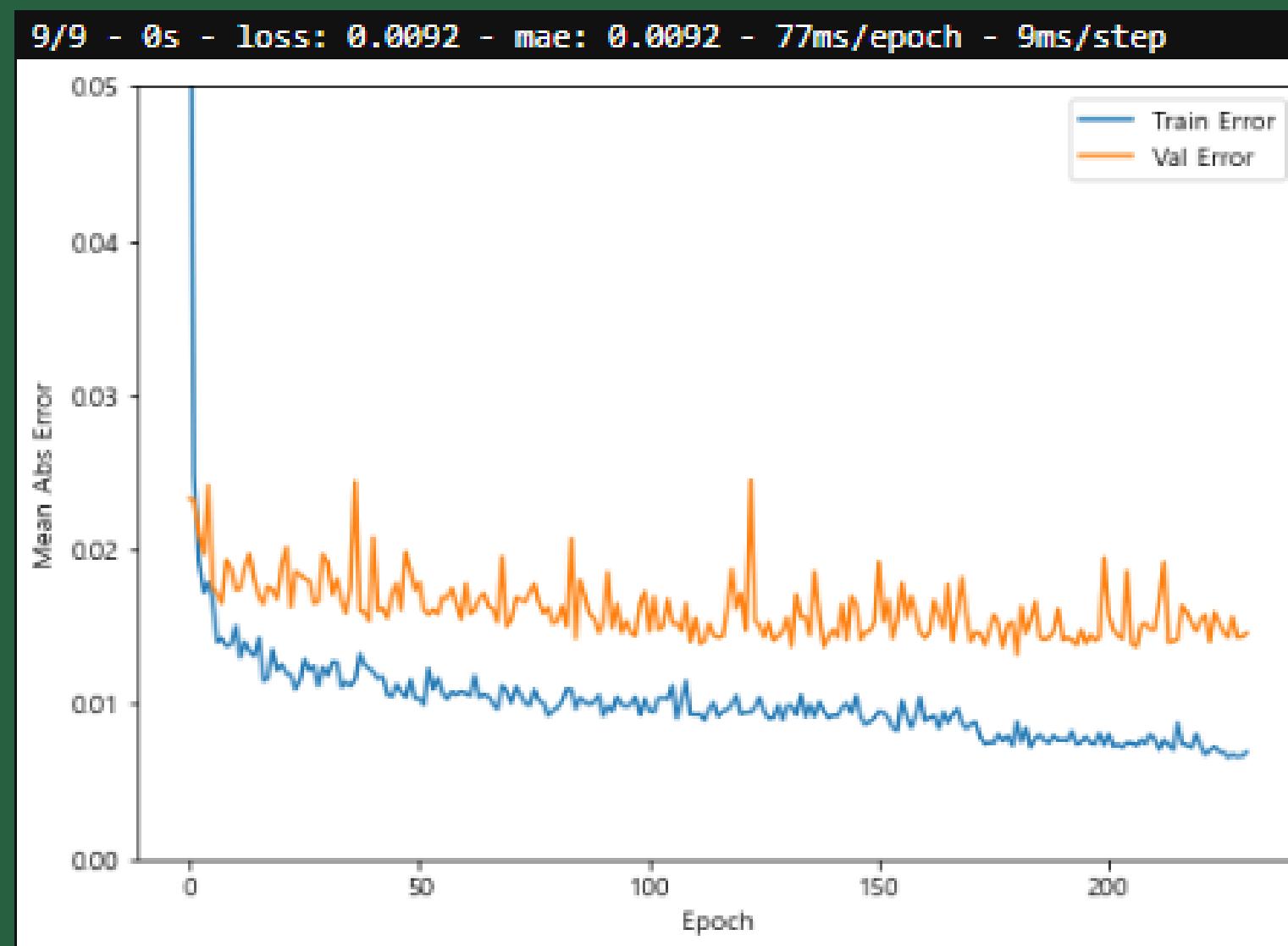
레이어의 node수 수정

dropout 추가 전 : 0.0091

dropout 추가 후 : 0.0125 + 검증점수가 후반부의 epoch로 갈수록 안좋아진다.

| Modeling - Deep Learning

[딥러닝 모델 MAE 평가 지표 결과] -최종점수



제출결과

- batch 16 : nmae 스코어 0.230844
- batch 8 : nmae 스코어 0.209930

I Modeling - Machine Learning

[예측에 사용될 모델 선정]

선형 회귀

- LinearRegression
- Ridge
- Lasso

회귀 트리

- RandomForestRegressor
- XGBoost
- LightGBM

I Modeling - Machine Learning

[선형 회귀] - LinearRegression, Ridge, Lasso

- 선형적인(=직선적인) 관계를 모델링 한 것 즉, 선형 회귀란 주어진 데이터를 이용해 일차방정식을 수정해 나가는 것

LinearRegression

LinearRegression이 회귀 모델 중 가장 기본 모델, 하지만 과적합이 일어나기 쉽기 때문에 복잡도를 제어할 수 있는 Ridge와 Lasso를 많이 사용 한다

Ridge

Ridge에 사용하는 규제방식은 L2. 규제란 과대적합이 되지 않도록 모델을 강제로 제한한다는 의미. Ridge는 모델을 단순하게 (계수를 0에 가깝게) 해주고 훈련 세트에 대한 성능 사이를 절충할 수 있는 방법 제공. alpha 매개변수로 훈련 세트의 성능 대비 모델을 얼마나 단순화할지 지정 할 수 있다.

Lasso

규제를 적용하는데 Ridge 대안으로 Lasso, Ridge와 비슷하지만 L1 규제를 사용해 일부 계수를 0으로 만든다. 일부 계수를 0으로 만들면 모델을 이해하기 쉬워지고 모델의 가장 중요한 특성이 무엇인지 드러내 준다

I Modeling - Machine Learning

[선형 회귀 모델 MAE 평가 지표 결과]

< 파라마터를 적용하지 않고 모델들 평가 >

```
LinearRegression mae : 0.016  
Ridge mae : 0.017  
Lasso mae : 0.18
```

MAE 평가 지표를 비교해보면 LinearRegression과 Ridge는
많은 차이가 안 보이지만

Lasso는 그 둘에 비해 성능이 많이 떨어집니다.
하지만 셋다 좋은 결과는 아닙니다

< alpha값을 지정하고 3개의 모델을 평가 >

alpha 값은 GridSearchCV를 사용한 최적의 파라미터
[Ridge alpha=0.1, Lasso alpha=0.0001]

```
LinearRegression mae : 0.016  
Ridge mae : 0.016  
Lasso mae : 0.016
```

Ridge와 Lasso 향상

alpha 값을 지정해 줘도 성능이 많이 좋아지는 것을 볼 수 없습니다.
이 3개의 모델은 저희 메타 데이터로 활용하기에는 좋지 않는 모델들이라고 판단
되며, Lasso의 알파값이 0.0001이라는 것은 LinearRegression과 별반 차이가
없을 것으로 판단 됩니다.

I Modeling - Machine Learning

[회귀 트리] - randomForestRegressor, XGBRegressor, LightGMB

- 회귀 트리란 일반적인 결정 트리와 같이 트리를 기반으로 하는 회귀 방식 모델

RandomForest

데이터 세트의 다양한 하위 샘플에 대한 여러 분류 의사 결정 트리에 맞추고 평균을 사용하여 예측 정확도를 높이고 과적 합을 제어하는 메타 추정기

XGBoost

여러개의 Decision Tree를 조합해서 사용하는 앙상블 알고리즘, 성능과 자원 효율이 좋아서 인기 있게 사용되는 알고리즘

LightGBM

LightGBM은 XGBoost와 함께 부스팅 계열 알고리즘에 각광받는 모델, LightGBM은 XGBoost와 비교해 큰 예측 성능 차이를 보이지 않으면서 학습 시간을 상당히 단축시킨 모델

I Modeling - Machine Learning

[회귀 트리 모델 MAE 평가 지표 결과] - randomForestRegressor

- 파라마터를 적용하지 않고 모델 평가 -

```
RandomForest without param MAE  
0.008850380723207167
```

앞서 수행한 선형 회귀 모델에 비해

회귀 트리인 randomForestRegressor가

상당히 좋은 결과를 보입니다

- 최적의 파라미터 값을 지정하고 모델 평가 -

하이퍼 파라미터 값은 GridSearchCV를 사용한 최적의 파라미터
['max_features': 6, 'n_estimators': 1000]

```
RandomForest with param MAE  
0.008723929062734281
```

눈에 뛸 정도로 개선이 되진 않았지만

하이퍼 파라미터 적용 전 보다 좋은 결과를 보입니다

I Modeling - Machine Learning

[회귀 트리 모델 훈련, 검증 세트 정확도] - randomForestRegressor

- 훈련세트와 검증 세트의 정확도 -

RandomForestRegressor 훈련 세트 점수: 1.00

RandomForestRegressor 테스트 세트 점수: 0.99

* 훈련 세트의 정확도가 100%이고 검증 세트 점수가 99%로
상당히 좋은 결과가 도출 됐습니다.

| Modeling - Machine Learning

[회귀 트리 모델 MAE 평가 지표 결과] - XGBRegressor, LightGMB 혼합

```
xgb_reg = XGBRegressor(n_estimators=2000, eta=0.005, max_depth=8,
                        colsample_bytree=0.9, subsample=0.8, random_state=42)
lgbm_reg = LGBMRegressor(n_estimators=1000, learning_rate=0.2, max_depth=6, min_child_samples=1,
                        num_leaves=10, subsample=0.1, colsample_bytree=0.5, reg_lambda=10, n_jobs=-1, random_state=42)
xgb_reg.fit(normed_train_df, normed_train_label_df)
lgbm_reg.fit(normed_train_df, normed_train_label_df)

xgb_pred = xgb_reg.predict(normed_val_df)
lgbm_pred = lgbm_reg.predict(normed_val_df)
pred = 0.9 * xgb_pred + 0.1 * lgbm_pred
preds={'최종혼합': pred,
       'XGBM' : xgb_pred,
       'LGBM' : lgbm_pred}
get_mae(preds)
```

최종혼합 모델의 MAE : 0.009813625690968098

XGBM 모델의 MAE : 0.010096918927045378

LGBM 모델의 MAE : 0.010128462254497868

앙상블 모델(XGBRegressor, LGBMRegressor) 혼합 사용한 결과가 현재까지 가장 좋은 점수가 도출 (mae: 0.00981 nmae: 0.135647)
파라미터는 GridSearchCV를 통해 최적화를 진행했고 추후, 더 정교한 파라미터 설정 및 앙상블 비율 수정, 앙상블 모델 추가를 통해 더 좋은 점수를
갱신할 수 있을 거라 생각합니다

I Modeling - Machine Learning

[스태킹 양상들]

- 스태킹 양상들은 두종류의 모델이 필요 첫번째는 개별적인 기반 모델, 두번째는 이 개별 기반 모델의 예측 데이터를 학습 데이터로 만들어서 학습하는 최종 메타 모델
- 쉽게 말해 학습용 피처 데이터 세트와 테스트용 피처 데이터 세트를 만드는 것, 데이터 셋이 부족하다 느껴 스태킹을 진행하게 되었고 스태킹 양상을 중 CV기반의 스태킹 선택

```

from sklearn.model_selection import KFold
from sklearn.metrics import mean_absolute_error

# 개별 기반 모델에서 최종 메타 모델이 사용할 학습 및 테스트용 데이터를 생성하기 위한 함수
def get_stacking_base_datasets(model, X_train_n, y_train_n, X_test_n, n_folds):
    # 지정된 n_folds 값으로 KFold 생성
    kf = KFold(n_splits=n_folds, shuffle=False)

    # 추후 메타 모델이 사용할 학습 데이터 반환을 위한 넘파이 배열 초기화
    train_fold_pred = np.zeros((X_train_n.shape[0], 1))
    test_pred = np.zeros((X_test_n.shape[0], n_folds))
    print(model.__class__.__name__, ' model 시작')

    for folder_counter, (train_index, valid_index) in enumerate(kf.split(X_train_n)):
        # 입력된 학습 데이터에서 기반 모델이 학습/예측할 폴드 데이터 세트 추출
        print('\t 폴드 세트: ', folder_counter+1, ' 시작')
        X_tr = X_train_n[train_index]
        y_tr = y_train_n[train_index]
        X_te = X_train_n[valid_index]

        # 폴드 세트 내부에서 다시 만들어진 학습 데이터로 기반 모델의 학습 수행
        model.fit(X_tr, y_tr)
        # 폴드 세트 내부에서 다시 만들어진 검증 데이터로 기반 모델 예측 후 데이터 저장
        train_fold_pred[valid_index, :] = model.predict(X_te).reshape(-1, 1)
        # 입력된 원본 테스트 데이터를 폴드 세트내 학습된 기반 모델에서 예측 후 데이터 저장
        test_pred[:, folder_counter] = model.predict(X_test_n)

    # 폴드 세트 내에서 원본 테스트 데이터를 예측한 데이터를 평균하여 테스트 데이터로 생성
    test_pred_mean = np.mean(test_pred, axis=1).reshape(-1, 1)

    # train_fold_pred는 최종 메타 모델이 사용하는 학습 데이터, test_pred_mean은 테스트 데이터
    return train_fold_pred, test_pred_mean

```

CV기반의 스태킹 순서- 기본 학습, 테스트용 피처 데이터 입력, 개별 모델이 k-폴드 세트 내부에서 원본의 학습 데이터를 다시 추출해 학습과 예측을 수행한 뒤 그 결과를 저장, 저장된 예측 데이터는 추후 메타 모델의 학습 피처 데이터 서트로 이용 또한 함수 내에서 폴드 세트 내부 학습 데이터로 학습된 개별 모델이 인자로 입력된 원본 테스트 데이터를 예측한 뒤 예측 결과를 평균해 테스트 데이터로 생성

I Modeling - Machine Learning

[스태킹 양상을 MAE 평가 지표 결과 case1]

- Lasso, XGBoost, LightGBM

선형 회귀와 회귀 트리 콜라보 스태킹 양상으로 진행

스태킹 회귀 모델의 최종(Lasso) mae 값은: 0.009696634984401712

스태킹 회귀 모델의 최종(XGBoost) mae 값은: 0.010750013157759423

스태킹 회귀 모델의 최종(LightGBM) mae 값은: 0.010750013157759423

- 앞서 시도한 다른 결과들 중에 가장 좋은 성능 평가를 보입니다
- 최종 모델은 3모델 다시도했고 그 중 Lasso를 최종 모델로 선택한 방법이 가장 좋은 성능 평가를 보였습니다.

I Modeling - Machine Learning

[스태킹 양상을 MAE 평가 지표 결과 case2]

- Lasso, RandomForest, XGBoost, LightGBM

case1번에서 사용하지 않은 RandomForest를 추가해 다시 진행해 봤습니다.

스태킹 회귀 모델의 최종(Lasso) mae 값은: 0.009371350274154965

스태킹 회귀 모델의 최종(XGBoost) mae 값은: 0.009160460686834033

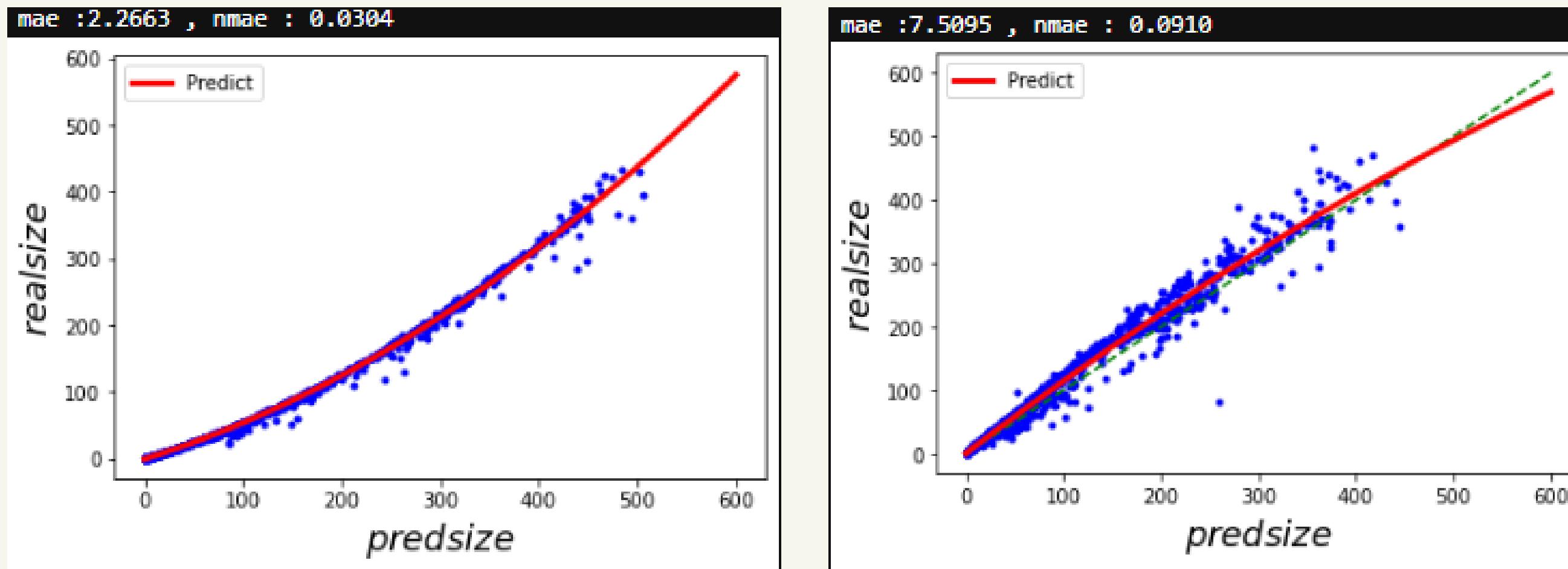
스태킹 회귀 모델의 최종(RandomForest) mae 값은: 0.009178662542169248

스태킹 회귀 모델의 최종(LightGBM) mae 값은: 0.00922170232601104

- case1번 보다 좋은 성능 평가가 보입니다.
- 미미한 차이지만 최종 모델 중 XGBoost를 최종 메타 모델로 선정 했을 때 성능이 가장 뛰어났습니다.

Modeling 결과

| Modeling - 한계점



좌 :수치화사이즈와 실제사이즈를 통한 현재사이즈를 예측해주는 다항회귀모델 mae : 0.003475

우 :위 모델을 통해 예측한 사이즈와 실제 다음날사이즈를 통해 다음날 크기를 예측해주는 모델 mae: 0.0156

훈련데이터의 실제 사이즈와 다음날 사이즈를 통해, mae값을 계산시 0.003148점이 도출됨(정규화된사이즈 기준)(이상적인 mae값)

예측사이즈와 실제 다음날사이즈로 구현한 모델의 mae값은 0.0156점임(우측 회귀모델)

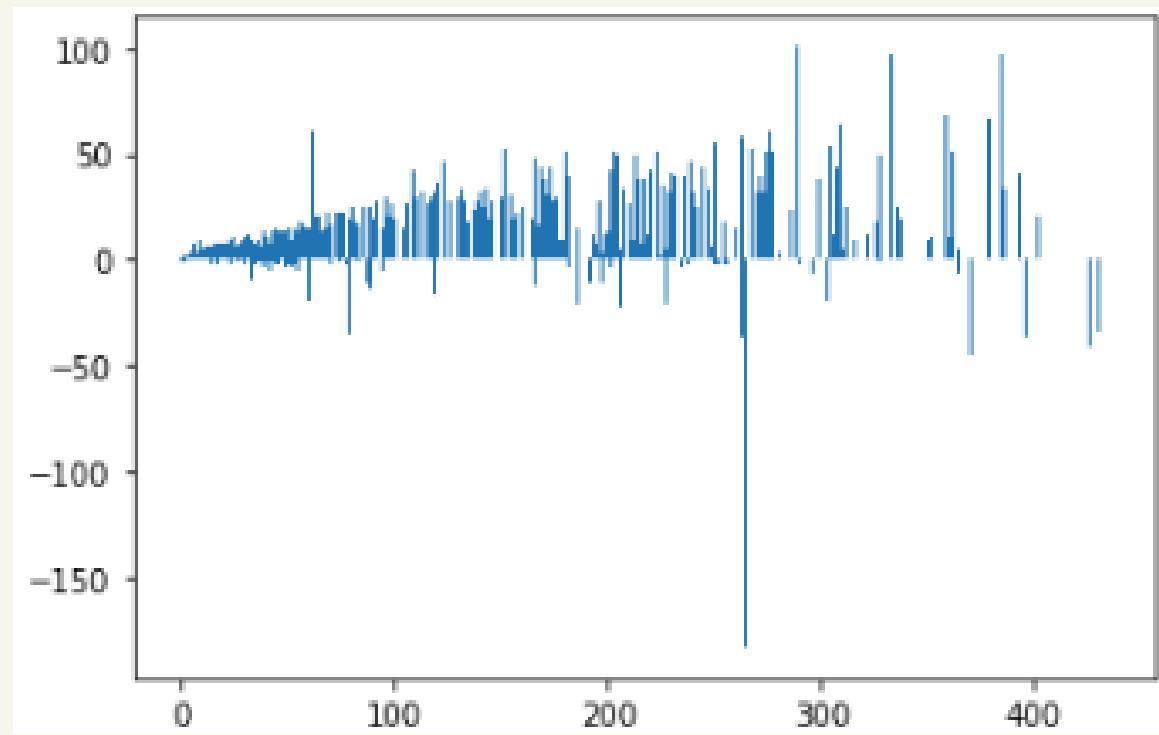
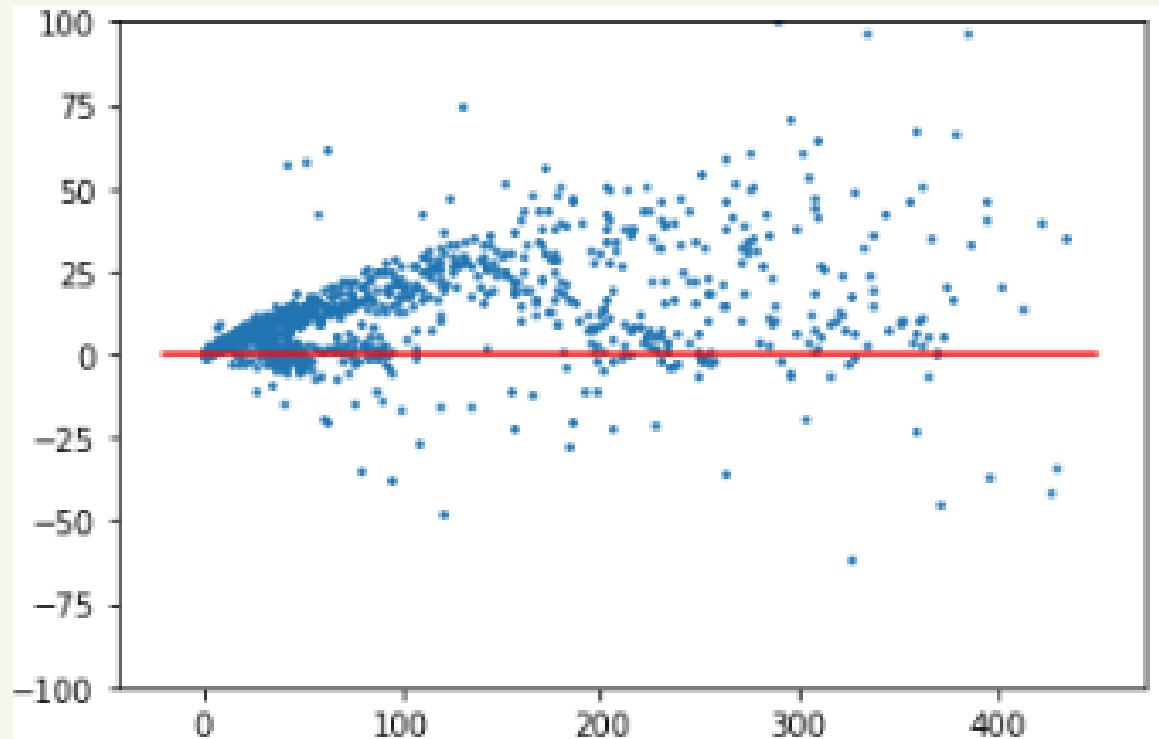
이미지를 통해 구한 현재 잎 사이즈와 실제 현재 잎 사이즈로 계산한 mae값은 0.0034725점임(좌측 회귀모델)

현재잎면적을 예측해 다음날 변화가 없다고 가정했을때 가장 좋은결과도출(대회점수 0.13361)

| Modeling - 한계점

결론

- 모델링 방향성에 문제가 있다고 판단됨
- 데이터의 수가 적어 예측 결과의 일반화가 되지 않음
- 미래 잎 면적 예측이 아닌, 성장 수치를 예측하는 모델로 전환 필요
 - 성장 수치(다음날 잎면적-현재 잎면적)를 현재 크기에 따라 가중치를 곱해주는 방식
 - 구한 성장 수치를, 이미지를 통해 얻어낸 현재 예측 사이즈에 더해준다.
 - 이때, 일정크기 이상의 실제잎면적을 가진 이미지는 이미지상으로 정확한 사이즈예측이 힘드므로 가중치 선정 시 고려
 - 또한, 성장수치에 맞는 새로운 feature 선정이 필요함
 - 성장수치에도 이상치가 존재하므로, 이를 제거할 필요가 있음
- 추후, 이와 같은 방향성으로 진행 예정 : 현재예측사이즈 + 성장정도
- 현재 도출된 모델결과값으로 비슷하게 재현을 해봄
 - 현재 예측 사이즈*(1-n)+(모델링을 통한 다음날 예측 사이즈)*(n)
 - 이를 통해 0.13564점을 기록
 - 현재예측사이즈 + 성장정도 방식으로 진행을 한다면, 현재최고점인 0.13361점에서 더 좋은 결과가 도출될 것으로 예상



현재 실제크기와 성장정도를 시각화한 그래프



데이콘 진행 상황



생육 환경 최적화 경진대회

KIST강릉분원 | Vision | 시계열 | 생육데이터 | NMAE
₩ 상금 : 총 300만원
⌚ 2022.04.18 ~ 2022.05.20 16:59 [+ Google Calendar](#)
👤 391명 🗓 D-23

참여

대회안내 데이터 코드 공유 토크 **리더보드** 팀 제출

PUBLIC RANKING CHART

#	팀	팀 멤버	점수	제출수	등록일
1	FarmInSun	Gj	0.13361	37	10분 전
2	재색		0.14478	6	11시간 전
3			0.14604	5	6시간 전
4			0.15006	25	4시간 전
5	무니		0.15277		
6			0.15898		

The chart shows the progression of scores for various teams over a week. The Y-axis represents the score, and the X-axis represents the dates from April 19 to April 26. The top six teams are tracked, with FarmInSun maintaining the lead throughout the period.

22.04.28 기준 0.13361점으로 1위

- 대회 상황 (22.04.27. 기준)

- 현재 데이콘 생육환경 최적화 경진대회 10일 차
- 이현경, 원종현, 윤혜림 팀으로 참가 중
- 현재 392명 참가
- 총 제출 수 37회



Part 3:
Farm in Sun

- 웹 서비스 개요
- DB설계
- 웹페이지 시연

웹 서비스 개요

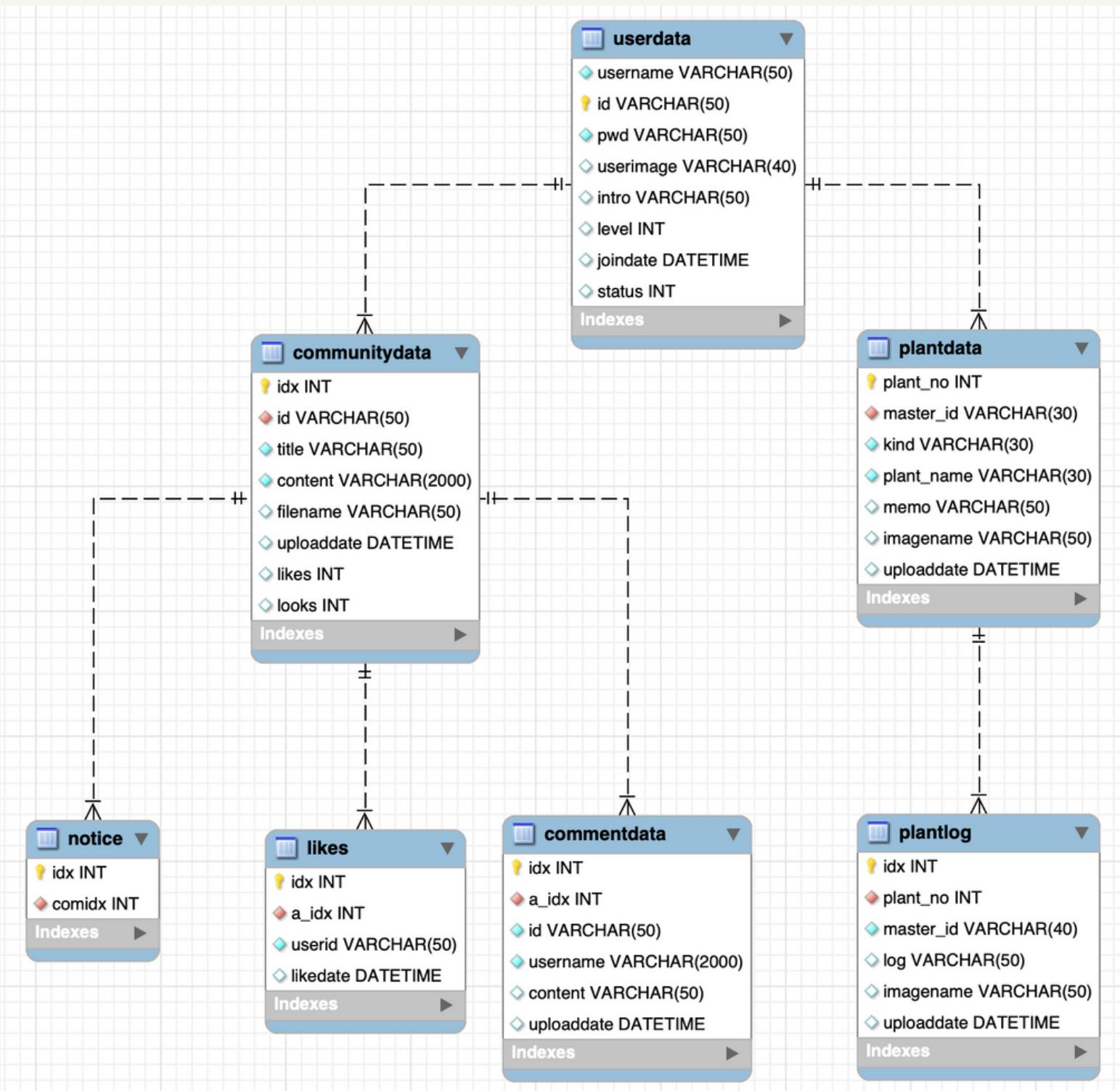
채소 성장 분석,
성장일기 등을 구현한 서비스 제공

앞서 Dacon의 데이터를 활용하여 잎의 크기를 예측해주는 모델을 생성했다.

이를 사용자가 활용할 수 있도록 서비스로 구현하고 일기, 커뮤니티, 팁 등 다양한 서비스를 제공하는 웹 서비스를 구현.

| DB 설계

[DB 관계도]



[DB 테이블]

[테이블 이름 : userdata]

userdata - login, signup 시 사용되는 table.

userdata의 id는 다른 데이터테이블의 기본 키.

[테이블 이름 : plantdata]

plantdata - 등록한 식물에 대한 table

[테이블 이름 : plantlog]

plantlog - 등록한 식물에 대한 log 데이터 테이블

[테이블 이름 : communitydata]

communitydata - 커뮤니티의 글쓰기, 사진 업로드 등을 관리하는 데이터 테이블

communitydata의 id는 commentdata / likes / notice
의 기본키

[테이블 이름 : commentdata]

commentdata - 커뮤니티의 댓글 관리

[테이블 이름 : notice]

notice - 운영진 공지 관리 테이블

[테이블 이름 : likes]

likes - 커뮤니티 좋아요 관리 테이블

| 웹 서비스 시연



Farm in Sun



Part 4:

개선점 및 발전사항

| 개선점 및 발전 사항

모델 부분 학습 데이터의 보충

이미지 데이터의 부족으로 인하여 학습량이 충분치 않아 모델 결과의 일반화에 어려움이 있음. 따라서 데이터 이미지에 맞는 유사 이미지와 현재 가진 이미지의 조정을 통해 학습 데이터를 보충하는 것이 필요.

| 개선점 및 발전 사항

모델의 한계점 개선

- 데이터의 수가 적어 예측 결과의 일반화가 되지 않으므로 모델의 방향성을 바꿀 필요가 있음.
- 미래 잎 면적 예측이 아닌, 성장 수치를 예측하는 모델로 전환 필요
- 성장 수치(다음날 잎 면적-현재 잎 면적)를 현재 크기에 따라 가중치를 곱해주고, 구해진 성장 수치를 이미지를 통해 얻어낸 현재 예측 사이즈에 더해준다.
- 결과적으로 예측 사이즈 + 성장 정도를 예측하는 모델로 전환하여 진행.

| 개선점 및 발전 사항

이미지 마스킹의 한계점

이미지 마스킹으로는 사진상으로 판별이 불가능한 겹쳐진 잎의 면적의 수치는 구해낼 수 없음. 특히, 많이 성장한 식물(청경채)의 경우 대부분의 잎이 가려져 있어 큰 오차가 발생함.
추후, CNN 모델과 결합해, 잎의 shape, 그림자 등을 통해 가려진 잎의 면적도 어느정도 유추가 가능한 방법을 찾을 수 있을 것으로 예상.

| 개선점 및 발전 사항

제한적인 예측

이미지 마스킹 모델과 딥러닝 / 머신러닝 모델이 청경채의 환경과 사진으로만 최적화가 되어있어 다른 식물들은 예측하기 힘든 부분이 있음.

확장가능한 서비스 제공 시, 예측하고자하는 식물의 데이터를 통해 좀 더 범용적인 모델구현이 가능할 것으로 예상

| 소감

이현경

경쟁을 통한 대회를 하며 프로젝트를
진행했습니다.

데이터와 모델들에 대한 부족한 점을
느꼈고 이를 보충하기 위해 밤을 지새
고 모르는 점들을 구글링하며 한발자
국 씩 나아간 노력들이 너무나 인상
깊었던 프로젝트였습니다!

그리고 너무나도 열심히 해준 우리
Farm in Sun 팀원분들께 정말 감사
하다는 말씀드리고 싶습니다!

원종현

머신러닝, 딥러닝을 배우기에 짧은시간
인 2주동안의 수업을 듣고, 프로젝트를
시작했을때는 정말 막막했습니다. 하지
만, 참여에 의미를 두었던 데이콘 경진대
회에서 프로젝트 기간동안 1등을 유지할
수 있는 정말 좋은 경험을 할 수 있었고
점수를 조금이라도 더 올리기 위해 정말
많은 정보들을 찾아보면서 열정적으로
공부할 수 있는 경험이 되었습니다. 남은
대회기간동안 틈틈히 개선을 진행해 최
종적으로 1등을 할 수 있으면 정말 좋을
것 같습니다.

윤혜림

짧은 기간에 머신러닝, 딥러닝을 배워서
잘 할 수 있을까 걱정이 많았지만 프로젝
트를 하면서 기술적인 면에서 많은 걸 배
웠고 첫 우리만의 머신러닝 을 만들면서
머신러닝에 관심이 많아 졌습니다. 머신러
닝, 딥러닝을 만들어 예측 수행하고 나오
는 결과값을 보고, 잘 만들 수록 결과값이
좋아진다는 것을 알아 열정적으로 좋은 모
델을 만드려 노력해 했습니다. 아쉬운 부
분도 있지만 3주동안 다양한 머신러닝을
구현해 볼 수 있던 기간이라 정말 보람찬
프로젝트 였습니다.



감사합니다!

QnA 시간 가지도록 하겠습니다!

