서울시 공유자전거 수요 예측

ESC Final Project

박웅준, 안서연, 최익준,
서지연, 박상용

목차

데이터 전처리 & 모델링

Tree 기반 모델링

Ensemble & 성능 개선

PART 1.

데이터 처리 / 모델링

```
Data columns (total 14 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Date                      8760 non-null   object 
 1   Rented Bike Count         8760 non-null   int64  
 2   Hour                      8760 non-null   int64  
 3   Temperature(°C)           8760 non-null   float64
 4   Humidity(%)               8760 non-null   int64  
 5   Wind speed (m/s)          8760 non-null   float64
 6   Visibility (10m)          8760 non-null   int64  
 7   Dew point temperature(°C) 8760 non-null   float64
 8   Solar Radiation (MJ/m2)   8760 non-null   float64
 9   Rainfall(mm)              8760 non-null   float64
 10  Snowfall (cm)             8760 non-null   float64
 11  Seasons                   8760 non-null   object 
 12  Holiday                   8760 non-null   object 
 13  Functioning Day           8760 non-null   object 
```

Time Series Data

2017/12/01 ~

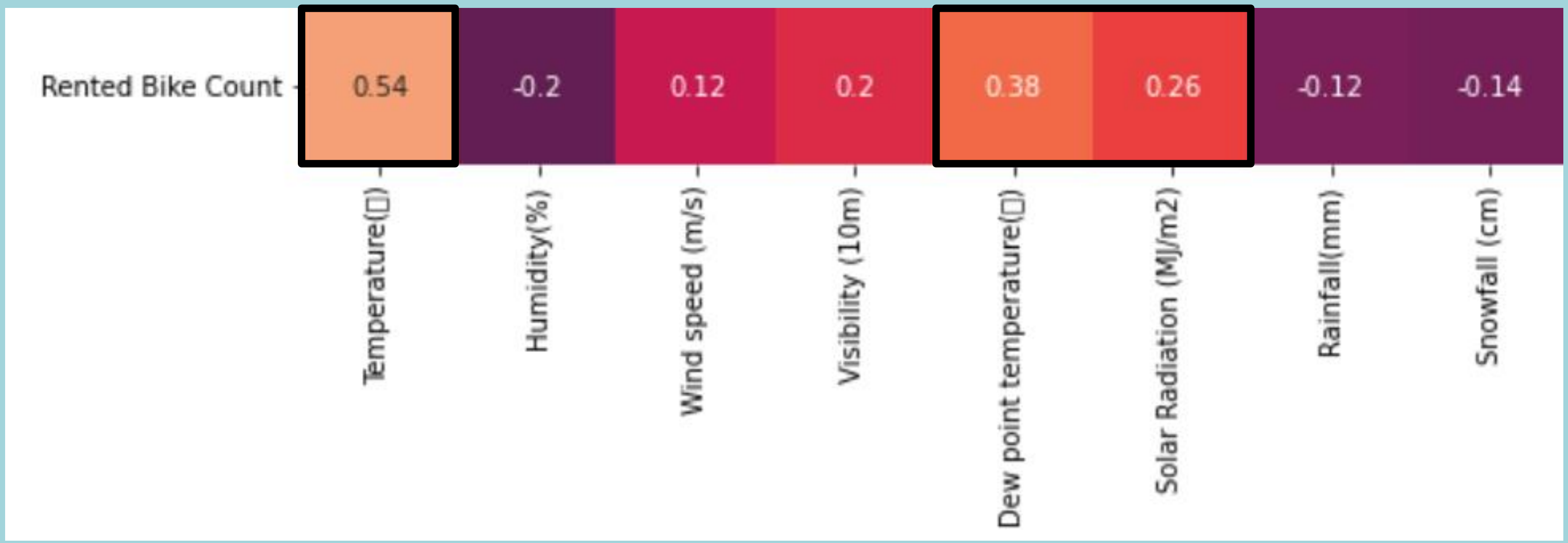
2018/11/30

14 Features

Numeric : 10개

Categorical : 4개

기상(대기) 변수	시간대 관련 변수
온도, 습도, 바람, 가시거리, 이슬점, 자외선, 강수량, 강설량	시간, 계절, 휴일 여부, 따릉이 시스템 작동 여부



주요 변수 (Y값 상관성)

- Temperature
- Dew point temperature
- Solar Radiation

→ 이후 feature importance: 비슷한 결과

변수 처리

기상 변수 PCA

Fail

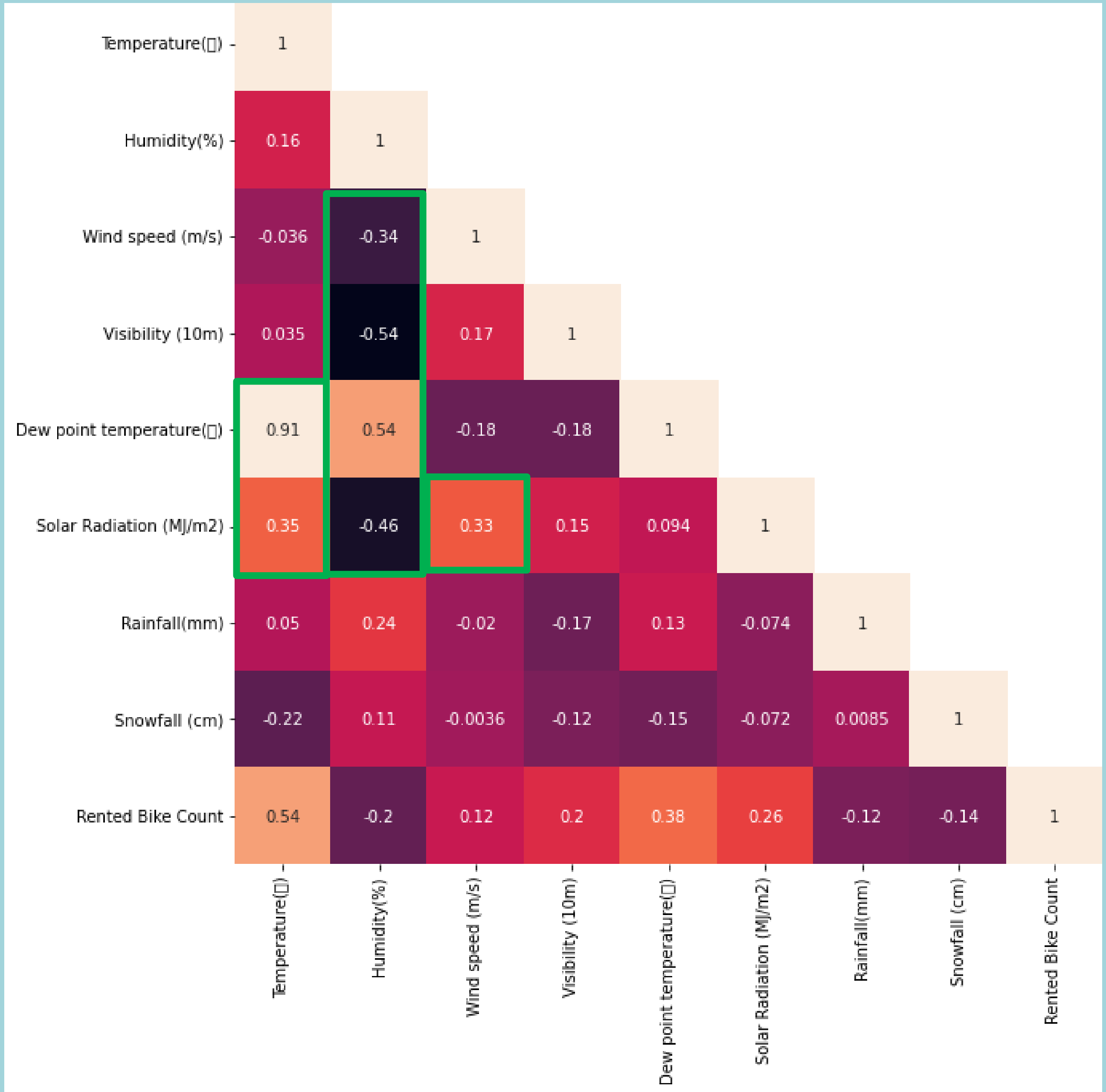
변수 Scaling

모델에 따라

PCA

- Temperature & Dew point temperature high correlation / VIF 높았던 Temperature & Dew point temperature & Humidity 끼리 PCA 시도: no improvement ☹

- 기상 변수들끼리 PCA 시도: no improvement ☹



Data Scaling

→ Categorical data 모두 dummy 화

```
data['Holiday'][data['Holiday']=='No Holiday'] = 0
data['Holiday'][data['Holiday']=='Holiday'] = 1
```

```
data['Functioning Day'][data['Functioning Day']=='Yes'] = 1
data['Functioning Day'][data['Functioning Day']=='No'] = 0

newcol=['date', 'rented bike count', 'hour', 'temperature', 'humidity',
       'wind speed', 'visibility', 'dew point temperature',
       'solar radiation', 'rainfall', 'snowfall', 'seasons',
       'holiday', 'functioning day', 'day', 'month']

data.columns=newcol
```

```
name_tonum={'Spring':1,'Summer':2,'Autumn':3,'Winter':4}

data['seasons']=data['seasons'].map(name_tonum)

cate_var=["hour",'day','month',"seasons","month","holiday"]
for var in cate_var:
    data[var] = data[var].astype("category")
```

Data Scaling

```
def log_transform(x):
    return np.log(x + 1)
```

```
from statistics import mean
from statistics import stdev as sd
```

```
def scale(x):
  return (x-mean(x))/sd(x)
```

```
cols=['temperature',
 'humidity',
 'wind speed',
 'visibility',
 'dew point temperature',
 'solar radiation',
 'rainfall',
 'snowfall']
cols1 = ['temperature', 'dew point temperature']
cols2 = ['humidity', 'wind speed', 'visibility', 'solar radiation', 'rainfall', 'snowfall']
data_scaled=data.copy()
for x in cols2:
  data_scaled[x] = log_transform(data[x])
for x in cols:
  data_scaled[x] = scale(data_scaled[x])
```

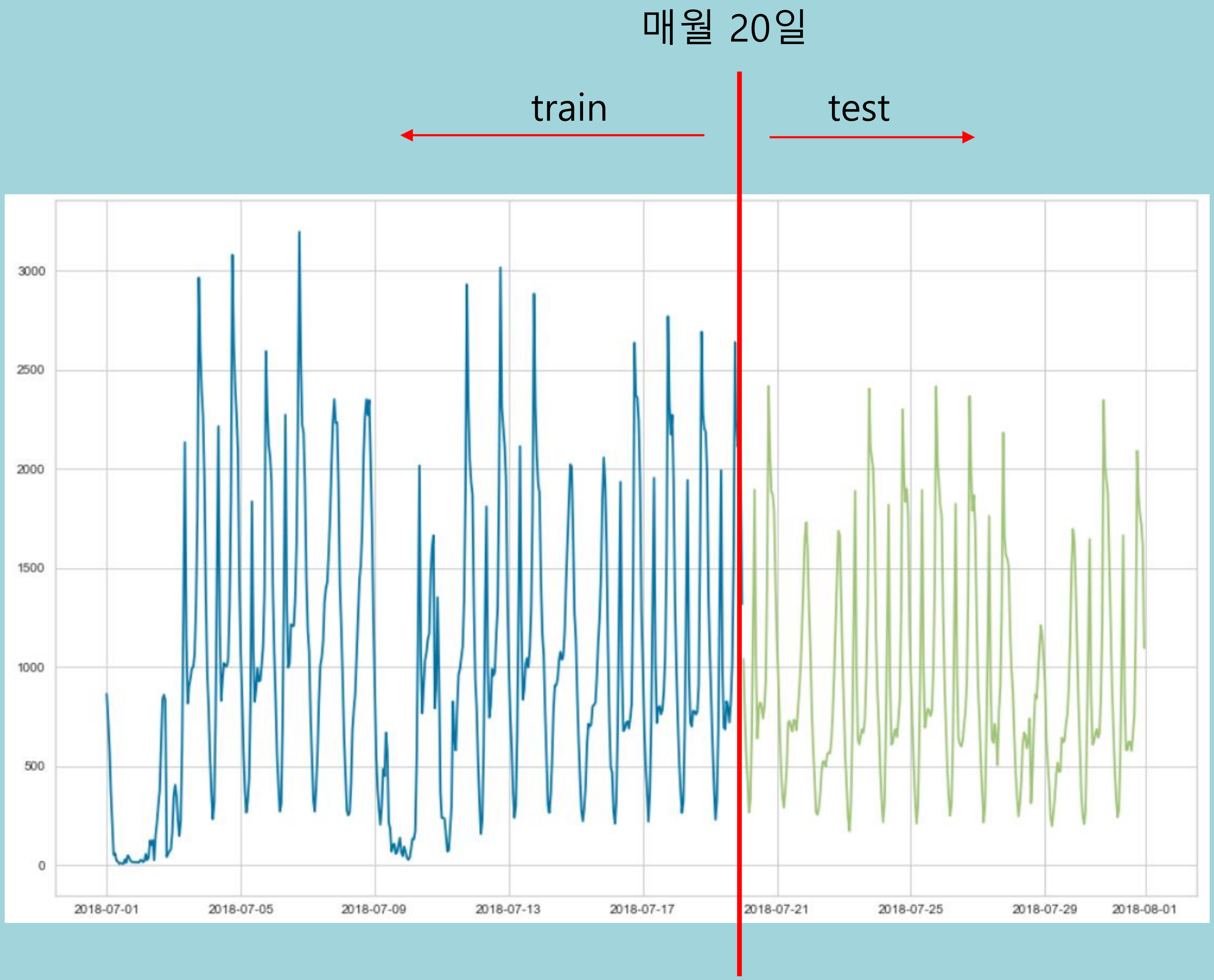
Cols 2: 지수분포와 비슷한 변수는 log_transform 후 scale 해줌

Cols 1: normal 따를 것으로 기대되는 변수는 standard_scaling

Train-Test Split

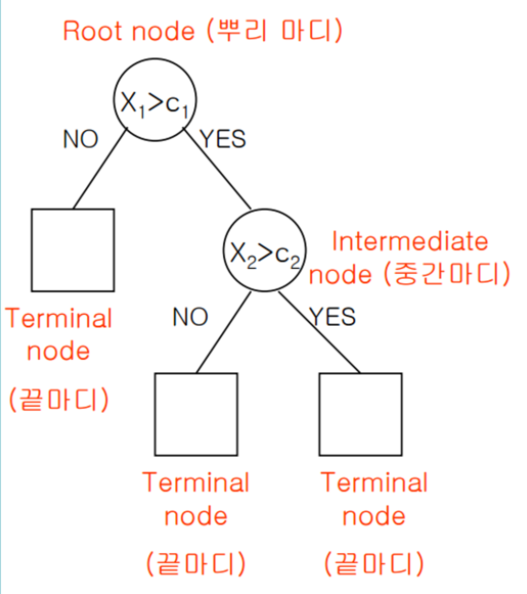
✓ 매월 20일을 기준으로
Train – Test Split

* 1일 ~ 20일 : Train
* 21일 ~ 30(31)일 : Test



PART 2.

Tree 기반 모델링



- Pycaret -

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
lightgbm	Light Gradient Boosting Machine	1.959493e+02	8.739936e+04	2.740743e+02	6.746000e-01	0.5984	7.857000e-01	0.045
rf	Random Forest Regressor	2.079232e+02	1.005079e+05	2.962512e+02	6.293000e-01	0.6099	8.525000e-01	0.505
knn	K Neighbors Regressor	2.318936e+02	1.238236e+05	3.280183e+02	5.287000e-01	0.6165	8.027000e-01	0.025
et	Extra Trees Regressor	2.065504e+02	9.895985e+04	2.875643e+02	6.616000e-01	0.6396	1.063300e+00	0.411
gbr	Gradient Boosting Regressor	2.241128e+02	1.090952e+05	3.081725e+02	6.146000e-01	0.6543	8.364000e-01	0.166
dt	Decision Tree Regressor	2.694274e+02	1.684489e+05	3.882460e+02	3.100000e-01	0.7497	9.274000e-01	0.018

→ 트리 기반의 모델 성능이

비교적 좋게 나타나는 것을 확인

→ 각각을 Modeling & Hyper-parameter tuning

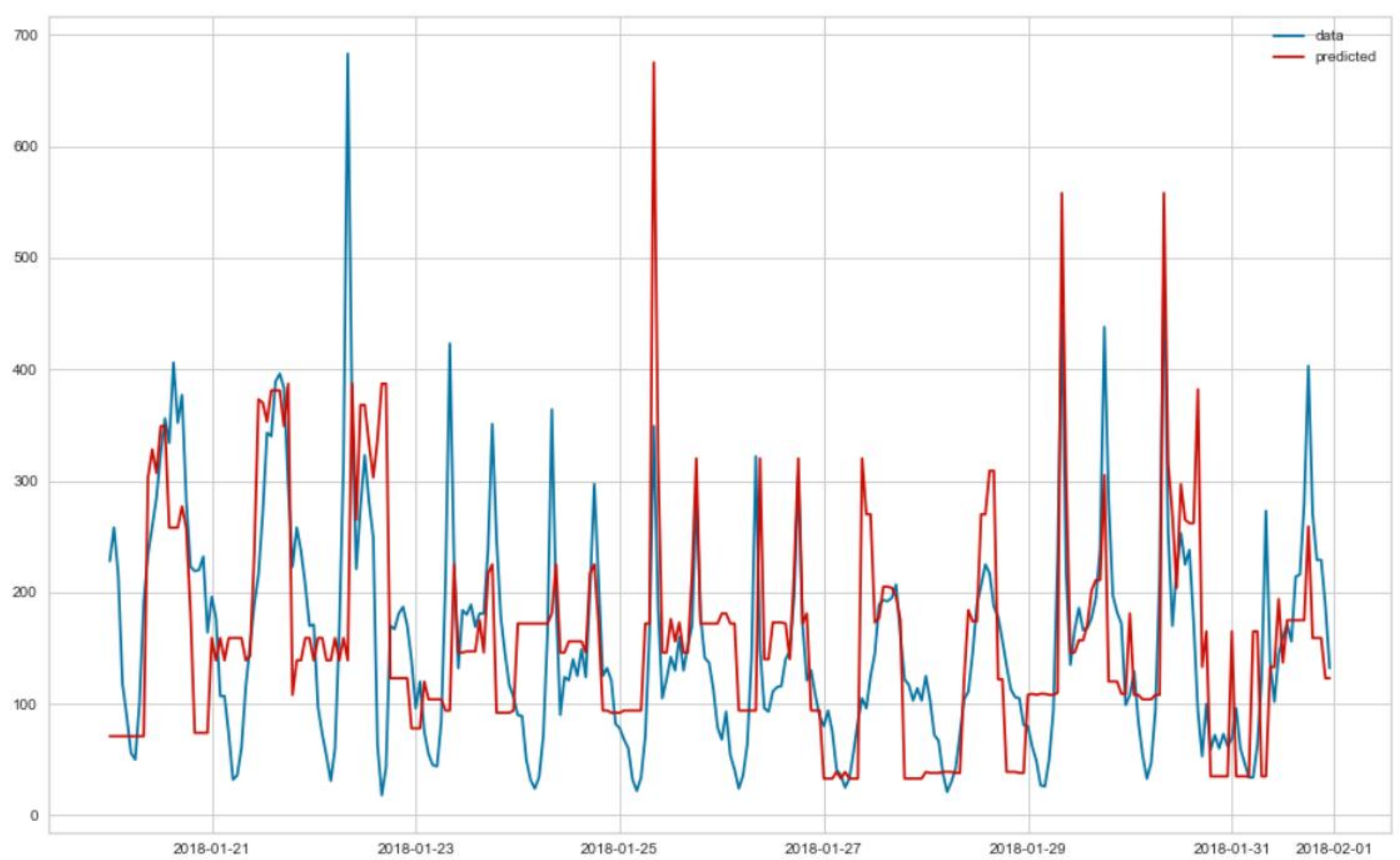
```
DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', max_depth=None,
                      max_features=None, max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, presort='deprecated',
                      random_state=123, splitter='best')
```

Decision Tree

Parameter 기본 설정

- 전반적인 경향성 찾아냄
- 오차가 큰 부분이 존재

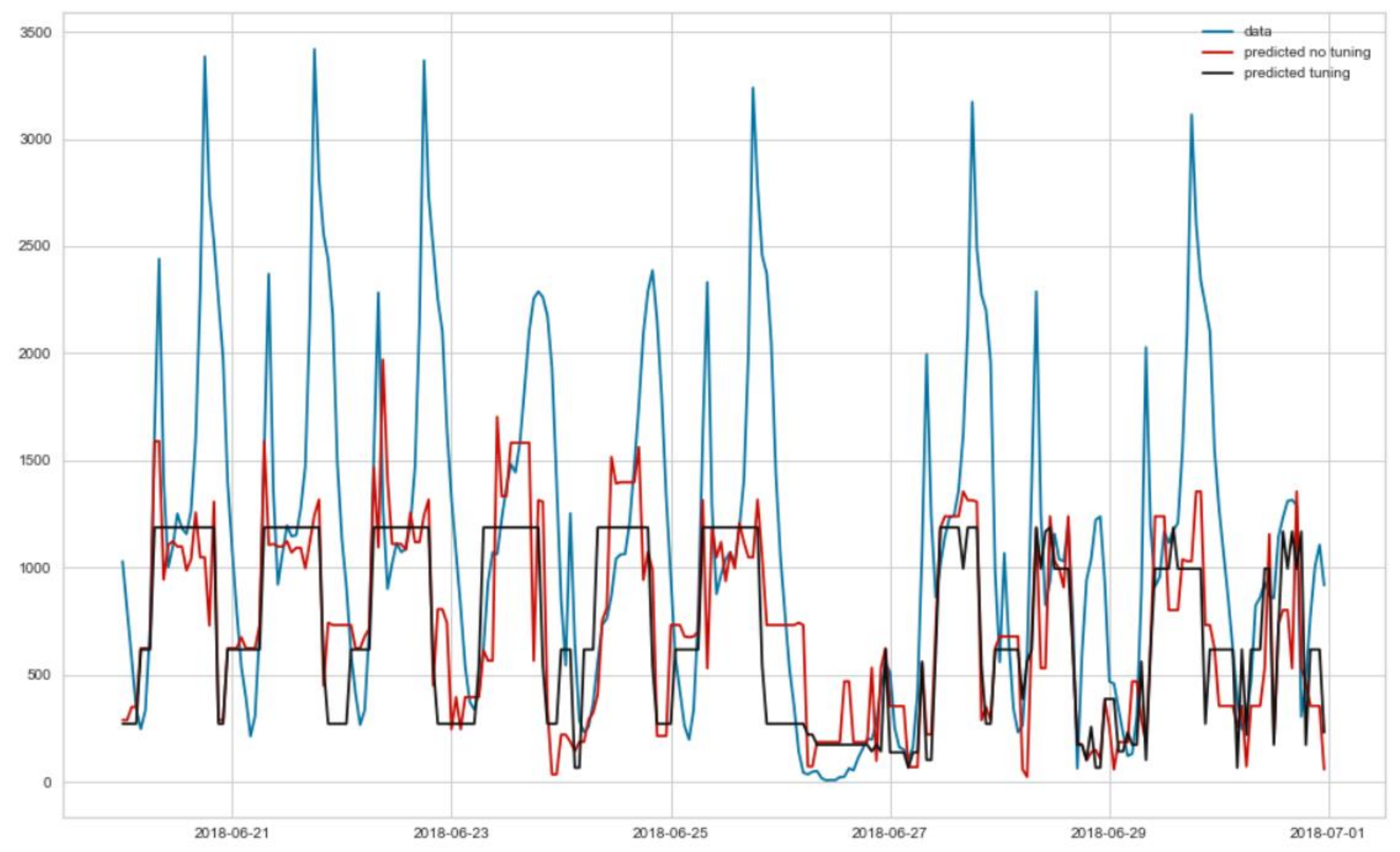
→ 다른 모델보다 예측 성능 떨어짐



Hyper-parameter Tuning 전

Model	RMSLE
Decision Tree Regressor	1.2192

```
DecisionTreeRegressor(ccp_alpha=0.0, criterion='mae', max_depth=15,
                      max_features=1.0, max_leaf_nodes=None,
                      min_impurity_decrease=0.02, min_impurity_split=None,
                      min_samples_leaf=6, min_samples_split=10,
                      min_weight_fraction_leaf=0.0, presort='deprecated',
                      random_state=123, splitter='best')
```



Hyper-parameter Tuning

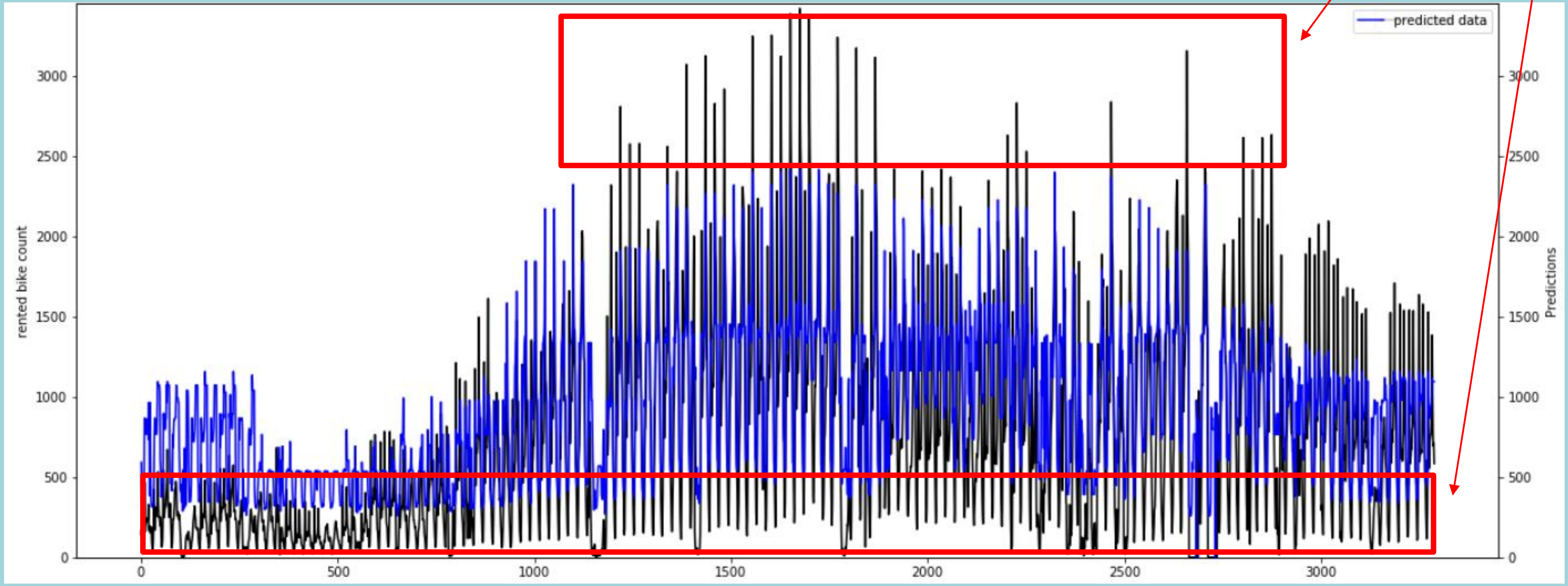
RMSLE
1.2026

Adaboost

Grid Search 활용 : Parameter tuning

RMLSE for the data: 1.2369379993004228

Peak / 낮은 값에서 큰 오차



Random Forest

모델실행

```
rf_params = {'random_state':[2016131028], 'n_estimators':[10, 20, 40, 60, 80, 100, 120, 140]}

rfModel = RandomForestRegressor()

grfModel = GridSearchCV(estimator=rfModel,param_grid=rf_params,scoring=rmsle_scorer,cv=5)


grfModel.fit(train2,np.log1p(ytrain2))
```

MSE & RMSE

```
preds = grfModel.predict(X= test2)

mean_squared_error(ytest2,np.exp(preds))
```

```
52043.657910751586
```

```
rmsle(np.log1p(ytest2),preds)
```

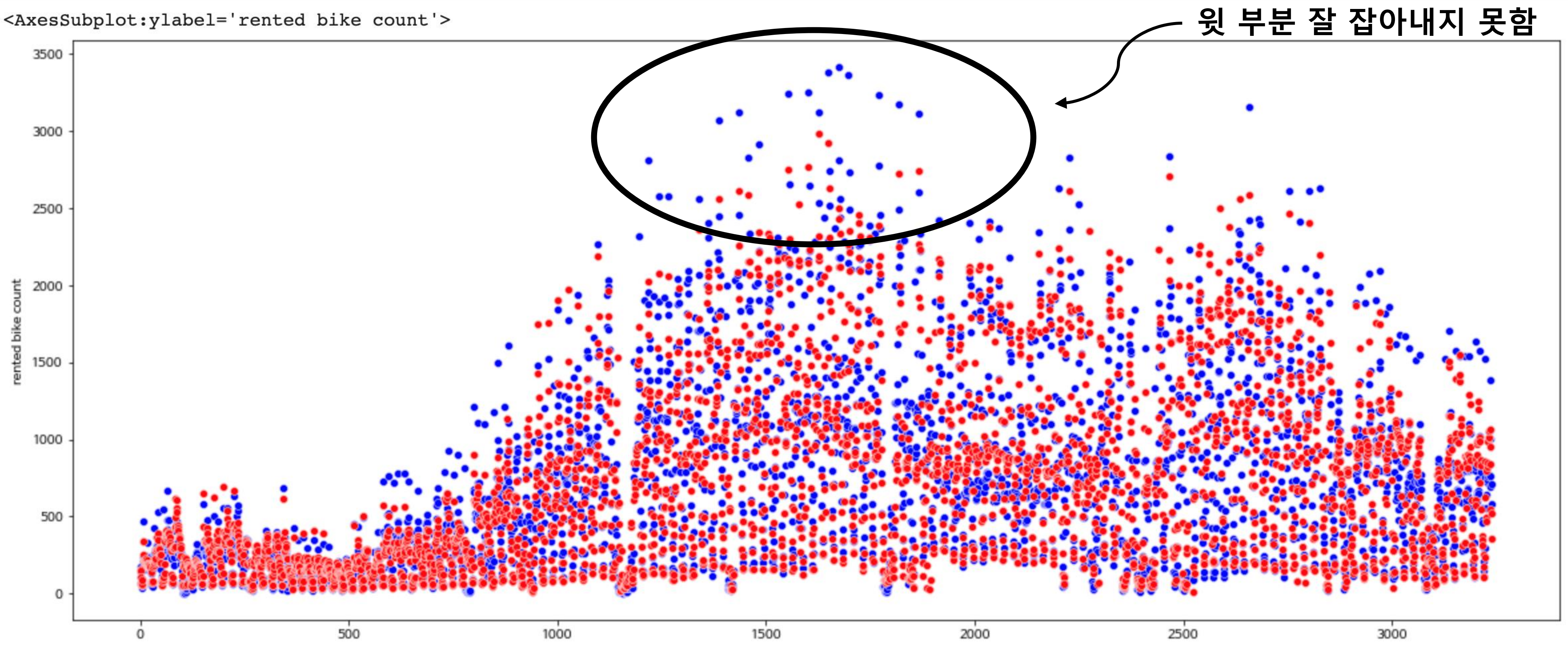
```
0.5075587696396644
```

→ **RMSLE 값이 비교적 낮게 나옴**

Random Forest - Plot

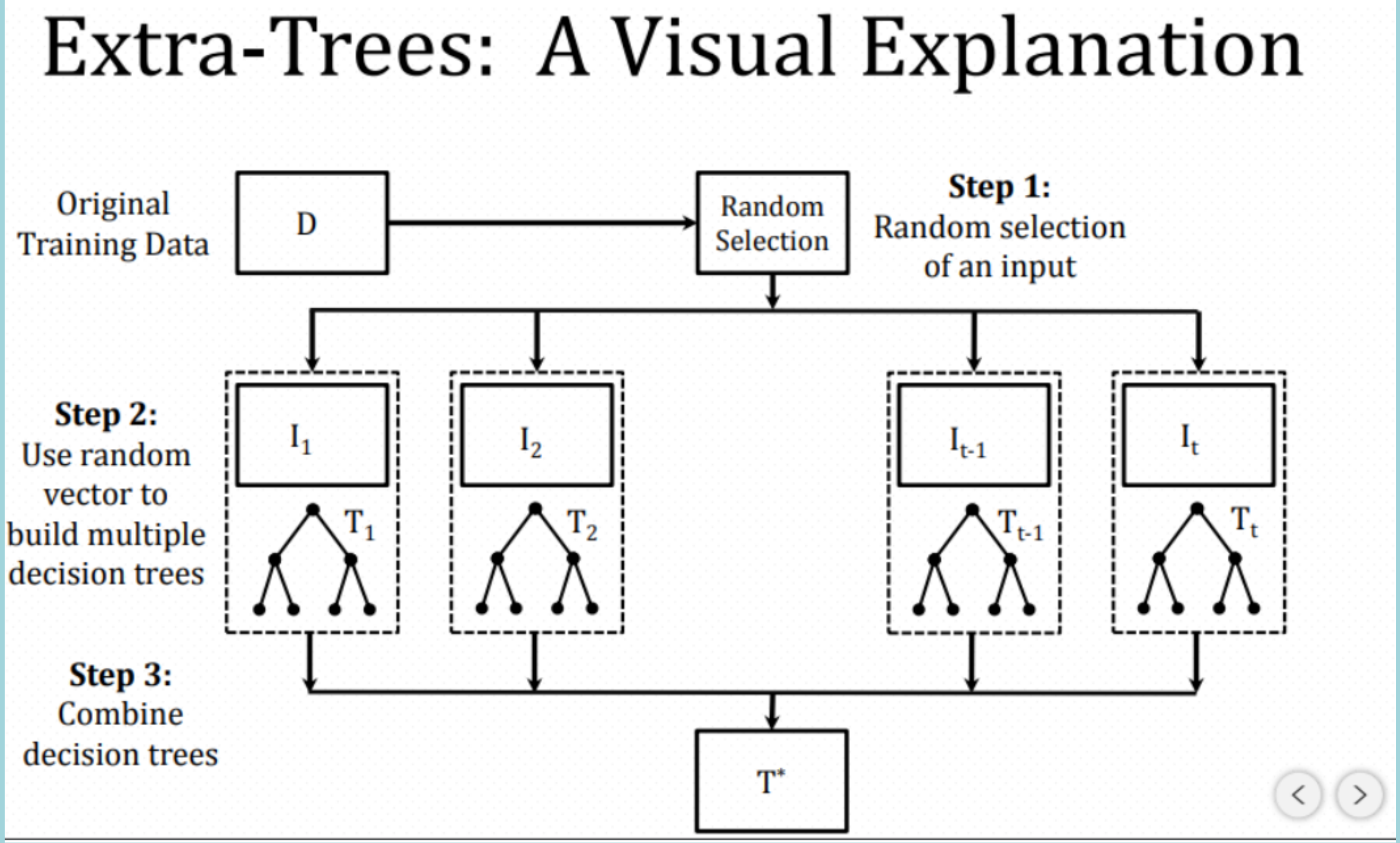
```
plt.rcParams["figure.figsize"] = (20,8)
fig1, ax1 = plt.subplots()
sns.scatterplot(ax=ax1,x=range(0,len(ytest2)),y=ytest2,color='blue')
sns.scatterplot(ax=ax1,x=range(0,len(ytest2)),y=np.exp(preds),color='red')
```

<AxesSubplot:ylabel='rented bike count'>



Extra Tree Regressor

What is Extra Tree?



Extra Tree

(랜덤포레스트와 비교)

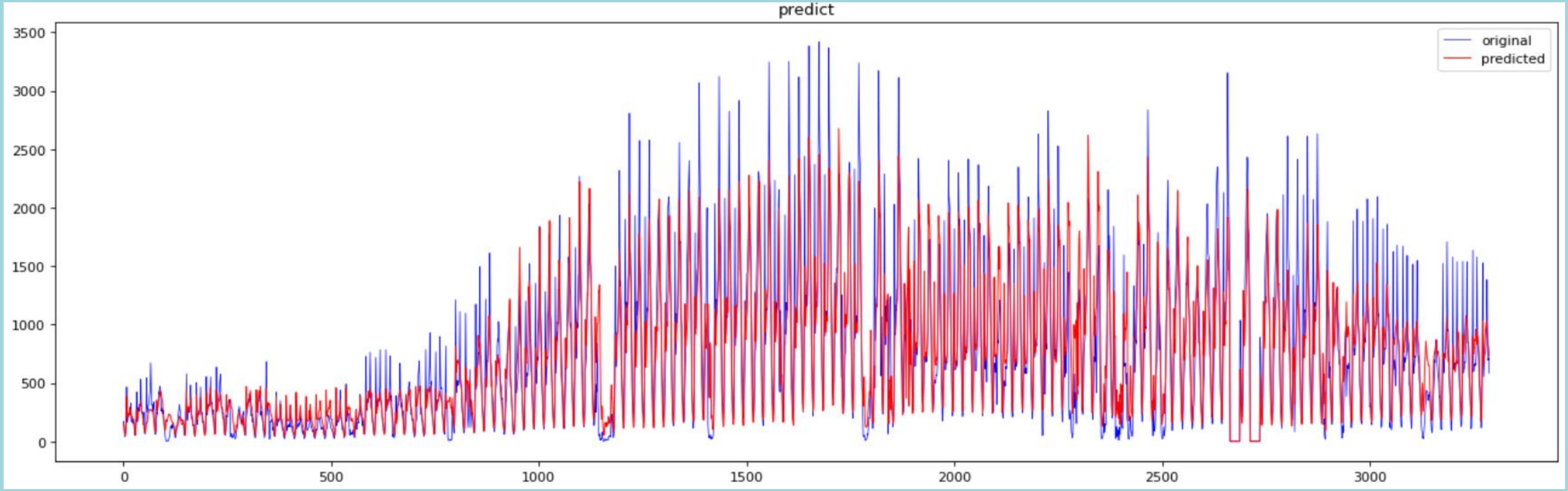
- 비복원추출(Bootstrap x)
- Random feature selection (information gain 비교 x)

Hyper-parameter tuning 전

Score: 0.9999999998228141
RMSE: 299.04

Hyper-parameter tuning By Grid-Search

RMSE: 271.84



{'min_samples_leaf': 8, 'min_samples_split': 6, 'n_estimators': 200}

Gradient Boosting Regressor

Parameter Tuning 성능

```
GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0, criterion='friedman_mse',
                          init=None, learning_rate=0.1, loss='ls', max_depth=3,
                          max_features=None, max_leaf_nodes=None,
                          min_impurity_decrease=0.0, min_impurity_split=None,
                          min_samples_leaf=1, min_samples_split=2,
                          min_weight_fraction_leaf=0.0, n_estimators=100,
                          n_iter_no_change=None, presort='deprecated',
                          random_state=123, subsample=1.0, tol=0.0001,
                          validation_fraction=0.1, verbose=0, warm_start=False)
```

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Gradient Boosting Regressor	177.4921	65626.8019	256.1773	0.8329	0.8209	0.7556

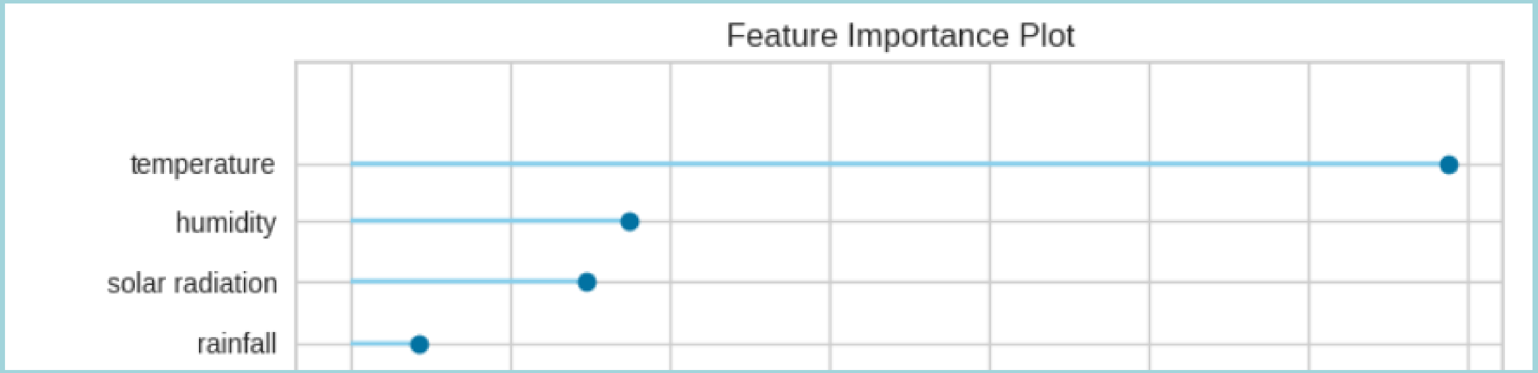
- 전반적인 경향성 찾아냄
- 오차가 큰 부분이 존재

→ 다른 모델보다 예측 성능 떨어짐

Gradient Boosting Regressor

Feature Importance

(Correlation plot과 비교)

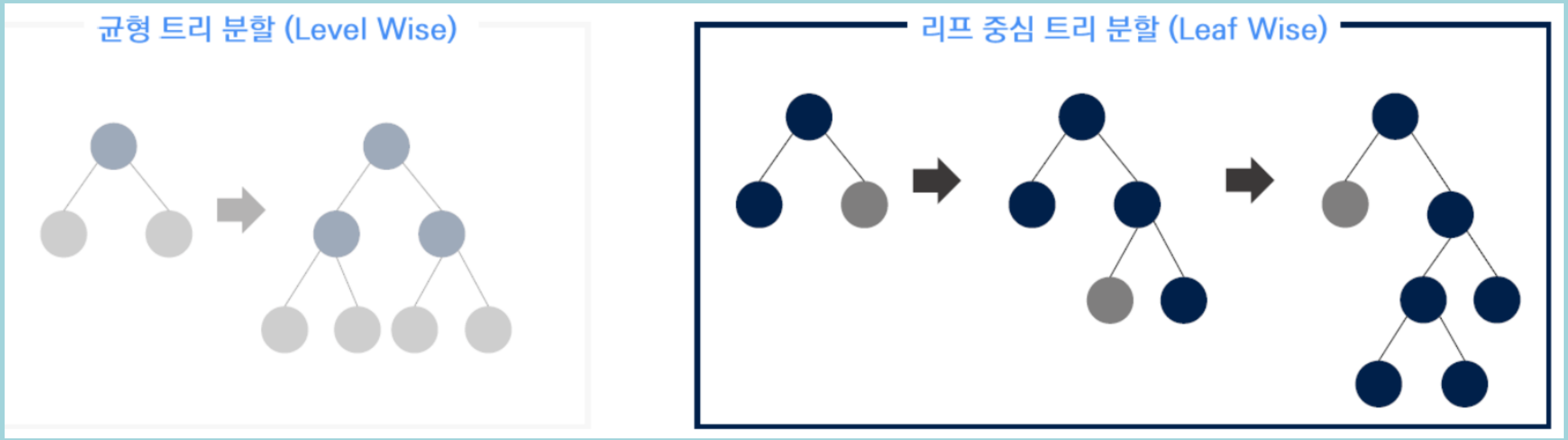


- (EDA) Correlation plot에서 살펴본 주요 변수와 동일
- 특히 Temperature의 영향이 크게 나타남

LGBM(Light Gradient Boosting Machine)

Gradient Boosting의 개선 모델

- LeafWise 트리 분할 : loss 줄이기가 빠름
- Fitting 속도가 빠름



LGBM(Light Gradient Boosting Machine)

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Gradient Boosting Regressor	177.4921	65626.8019	256.1773	0.8329	0.8209	0.7556

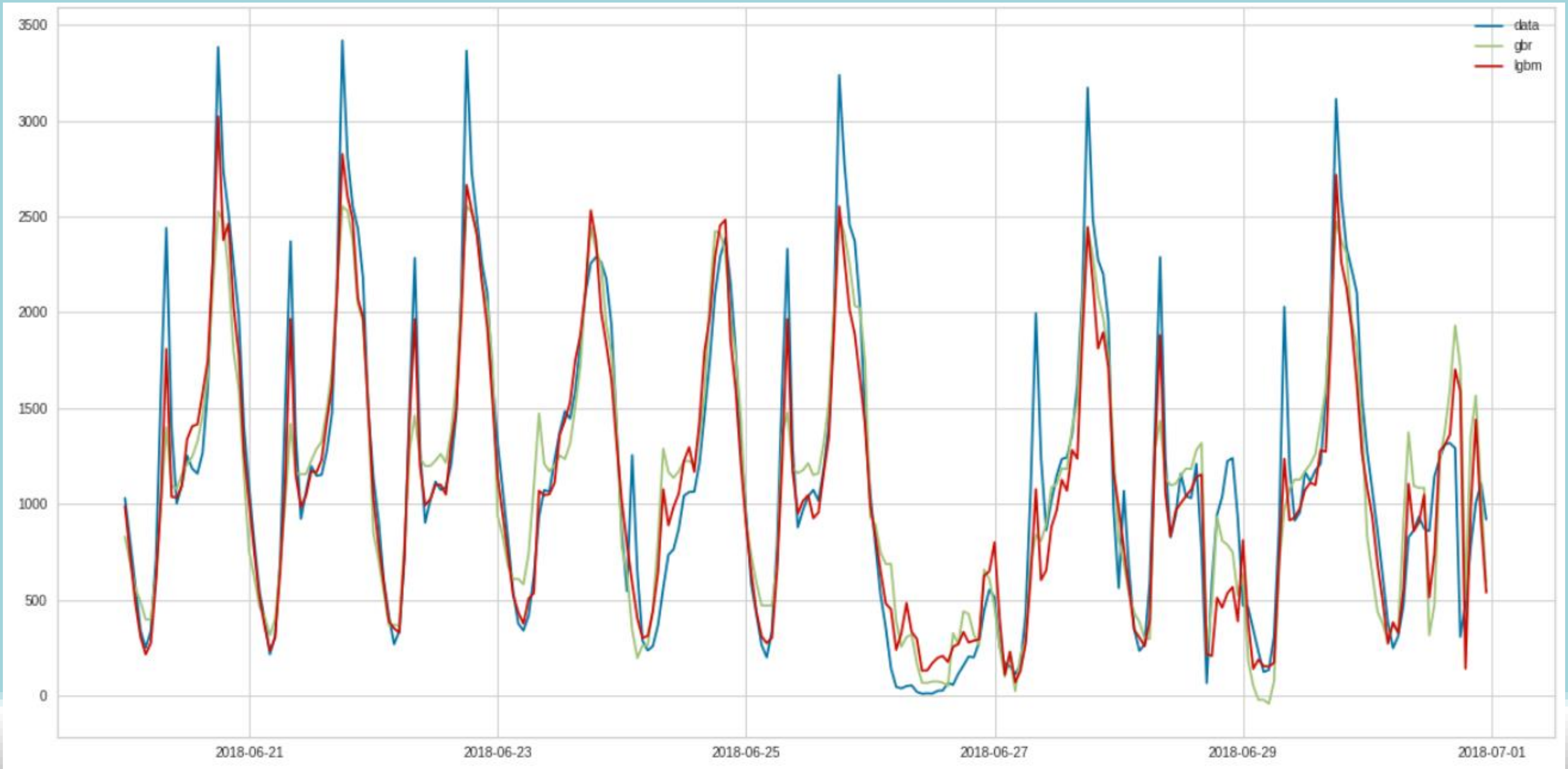
Gradient Boosting

성능 개선

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Light Gradient Boosting Machine	131.4954	41385.2511	203.4337	0.8946	0.7305	0.5657

LGBM

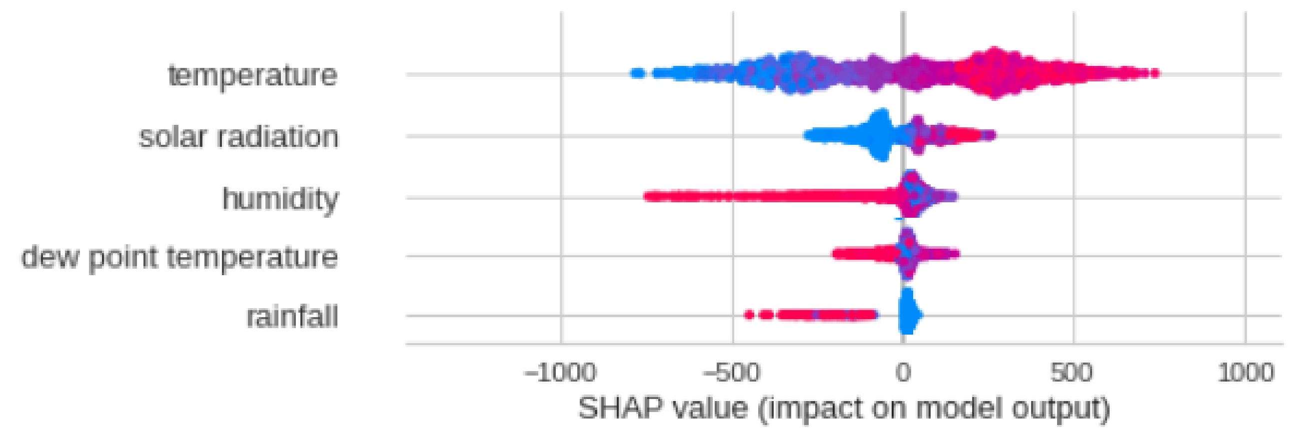
LGBM(Light Gradient Boosting Machine)



LGBM(Light Gradient Boosting Machine)

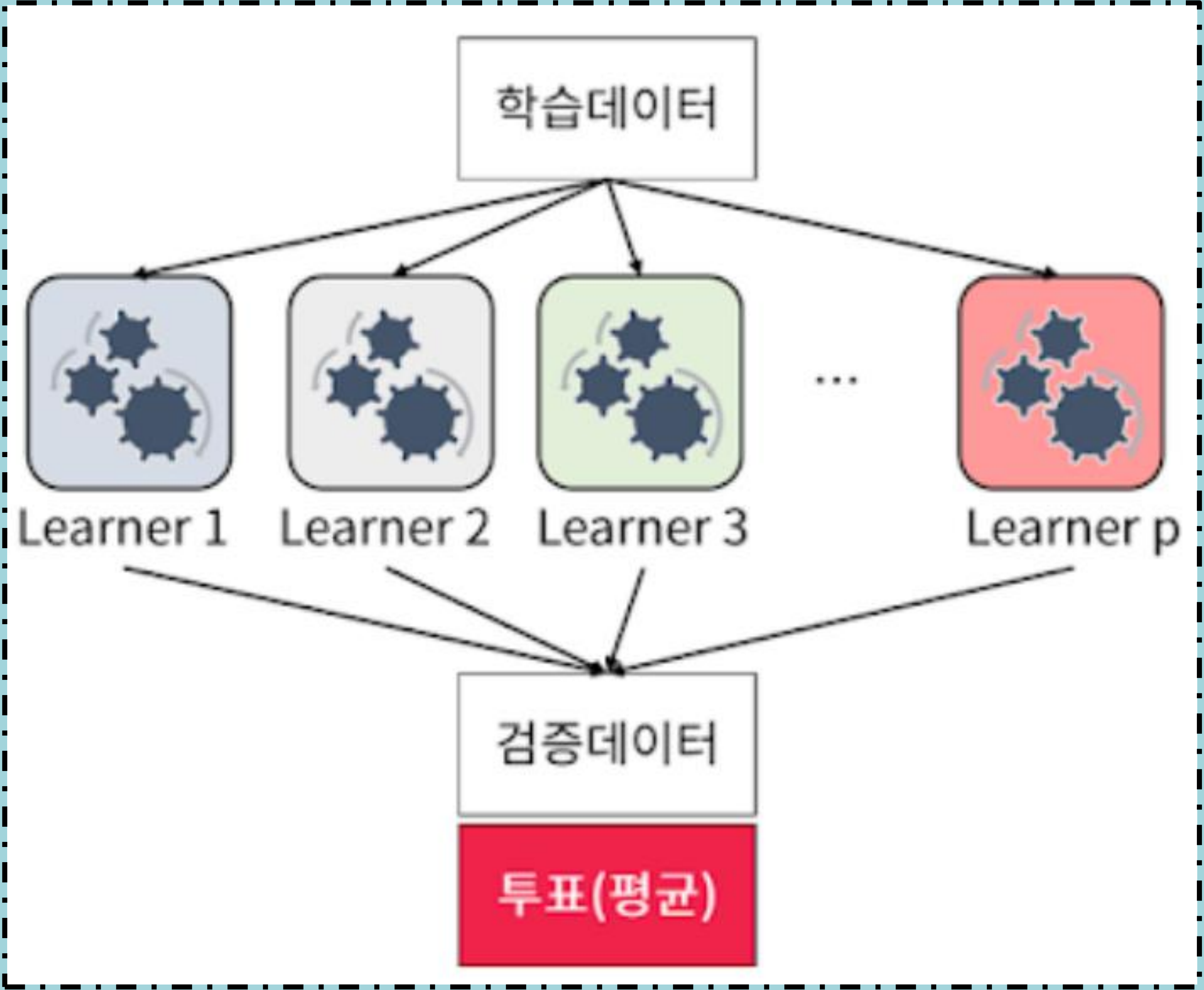
SHAP Value

예측값에 대한 직접적인 영향 정도를 표현



PART 3.

Ensemble



Objective

- 특정 모델의 경우, peak를 잘 잡지만,
  대부분 모델들이 그렇지 못함

- 성능 개선을 위해 Ensemble 사용

Ensemble Result

- **RMSLE 관점 : LGBM + Random Forest Blending**

LGBM + Random Forest Blending

```
LGBM rmsle: 0.6529353550079501
Random Forest: 0.5068182343852953
Ensemble rmsle: 0.4813819205005114
```

Base Model

Gradient Boosting

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Gradient Boosting Regressor	177.5741	65688.3357	256.2974	0.8327	0.8209	0.7549

LGBM

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Light Gradient Boosting Machine	131.4954	41385.2511	203.4337	0.8946	0.7305	0.5657

Extra Tree

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Extra Trees Regressor	148.5598	52355.8696	228.8141	0.8667	0.5961	0.8275

Bagging

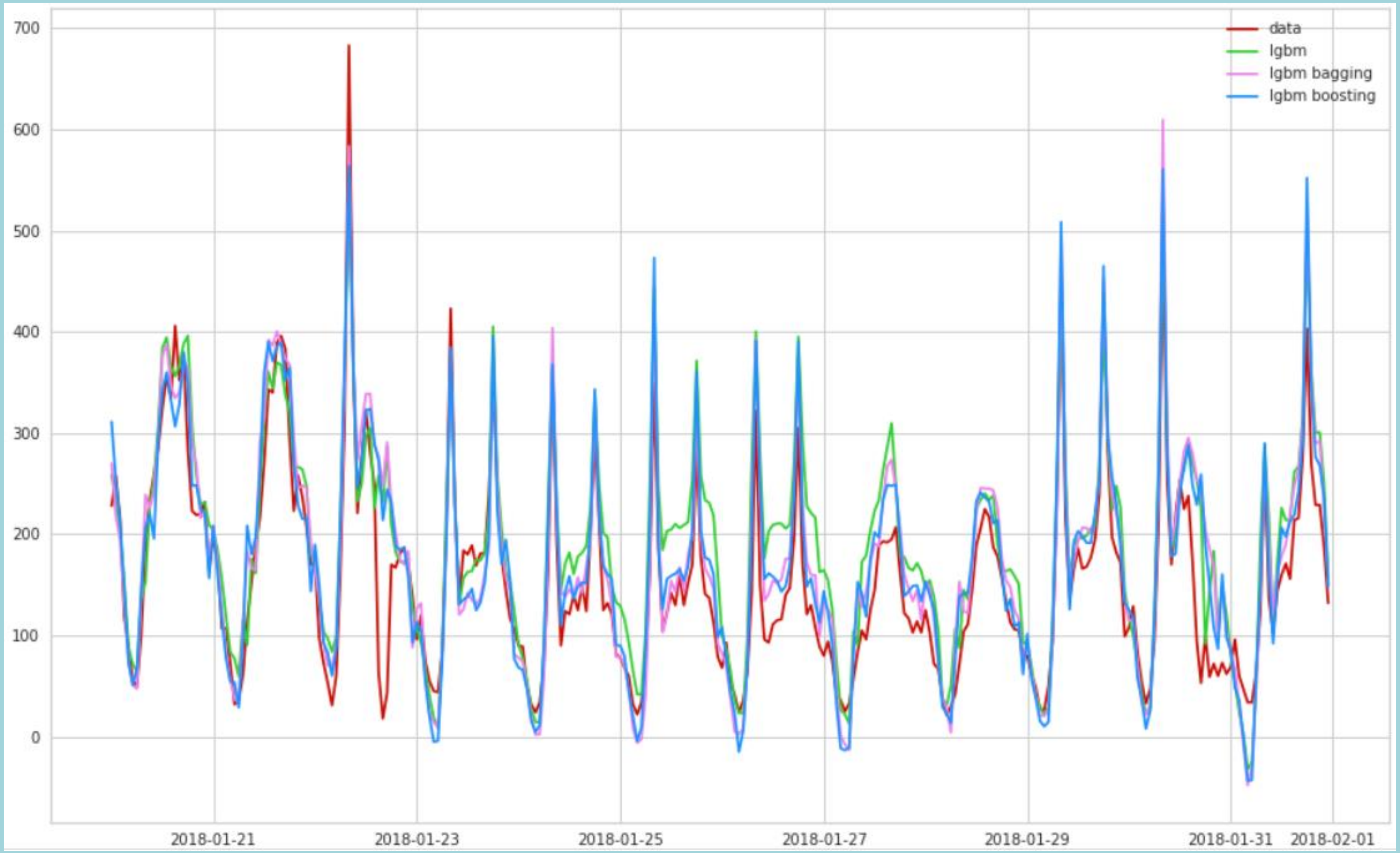
LGBM

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Light Gradient Boosting Machine	131.4954	41385.2511	203.4337	0.8946	0.7305	0.5657

RMSLE 값 작아짐

LGBM Bagging

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Light Gradient Boosting Machine	126.2197	39719.3284	199.2971	0.8989	0.7128	0.6216



data
lgbm
lgbm bagging
lgbm boosting
700
600
500
400
300
200
100
0
2018-01-21
2018-01-23
2018-01-25
2018-01-27
2018-01-29
2018-01-31
2018-02-01

Blending (Voting)

Gradient Boosting + LGBM + Extra Tree

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Voting Regressor	131.241	41427.5217	203.5375	0.8945	0.706	0.6404

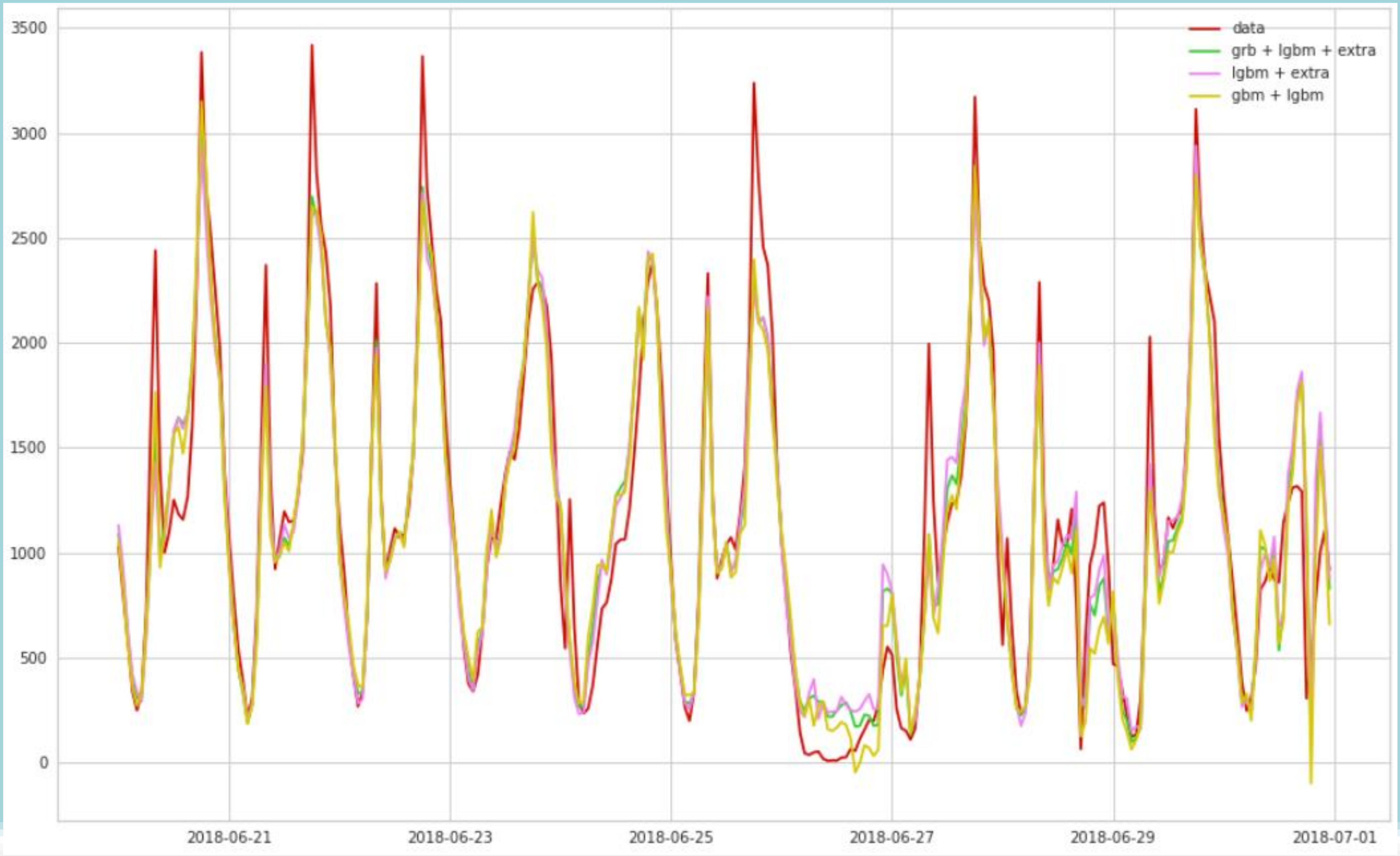
LGBM + Extra Tree

```
LGBM rmsle: 0.6529353550079501
Extra Trees rmsle: 0.5919890487905014
Ensemble rmsle: 0.5458615172683272
```

Gradient Boosting + LGBM

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Voting Regressor	131.8392	41888.6669	204.6672	0.8933	0.7389	0.5859

Plot


data
grb + lgbm + extra
lgbm + extra
gbm + lgbm
3500
3000
2500
2000
1500
1000
500
0
2018-06-21
2018-06-23
2018-06-25
2018-06-27
2018-06-29
2018-07-01

Stacking / Ensemble

Stacking (LGBM + Extra Tree + Linear Reg.)

```
Stacking rmsle: 0.9961176709985715
```

Ensemble (LGBM + Extra Tree + Random Forest)

```
LGBM rmsle: 0.6529353550079501
Extra Trees rmsle: 0.5919890487905014
Random Forest: 0.5068182343852953
Ensemble rmsle: 0.5164043923163714
```

Ensemble (LGBM + Random Forest)

```
LGBM rmsle: 0.6529353550079501
Random Forest: 0.5068182343852953
Ensemble rmsle: 0.4813819205005114
```

PART 4.

Conclusion & Limitation

☆ Data Engineering / Modeling / Ensemble & Parameter tuning : 성능 개선 & 예측

☆ 세션에서 다뤘던 여러 머신 러닝 기법 적용

🚲 시계열 데이터 : 딥러닝 RNN과의 비교 필요

🚲 데이터 자체의 날짜가 적고, feature도 한정적

Thank you <3