Solar power generation prediction based on weather and previous generation forecasts

Team deeplearning
Lee jung-hwan
Kim dong-hwan
Kwon ga-min
Park jun-young
Park chae-won

INDEX

Competition Description

Competition Description

Background of Topic selection

~~Actually, we just wanted to participate in the competition~~

With the increasing scarcity of fossil fuels, the importance of solar energy is growing. Recent studies indicate that solar energy will rapidly emerge as a dominant energy source

resourse — NewsQuest, 에너지 패턴의 대세는 태양광...금세기 중반 지배적인 에너지로 성장

Competition Description

Background of Topic selection

가치창출대학
POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

HENERGY

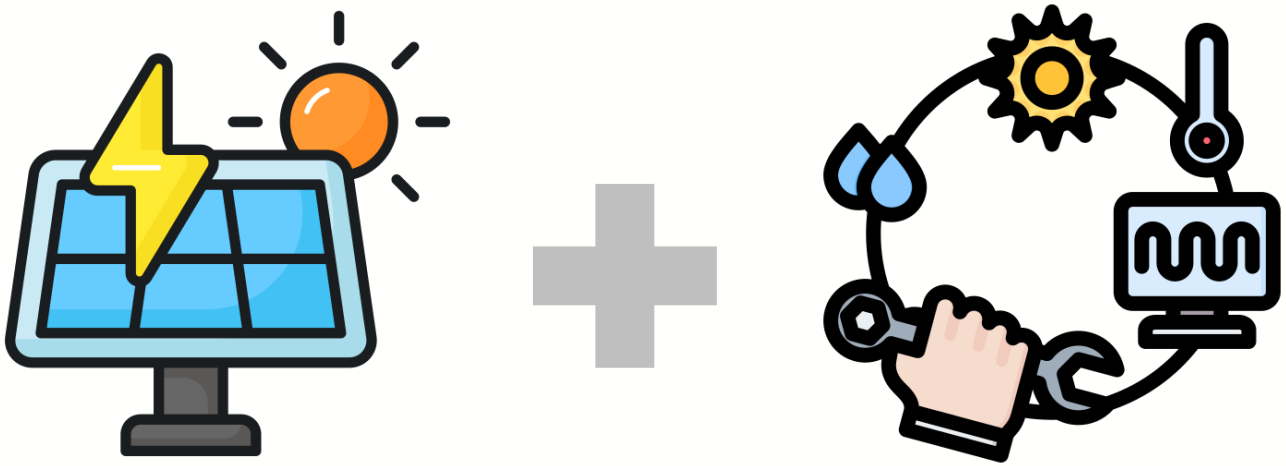
Solar Power Generation Forecast Competition hosted

by POSTECH and H energy

Competition Description

Background of Topic selection

Solar Energy Prediction



Plan on how to use the power generated through predictions

Minimize environmental impact by optimizing energy use

and reducing energy consumption

Competition Description

History of competition

1st Competition

2020

Prediction of future power generation using data from the solar power plant located in Jincheon-gun, Chungbuk

2nd Competition

2020

Optimal bidding in the electricity market by predicting the next day's power generation using data from small-scale solar power plants

3rd competition

2021

Bidding the hourly predicted power generation for the next 24 hours of a virtual power plant, with incentives based on prediction performance

4th competition

2022

Predicting the next day's power generation range for newly installed power plants using data from other regional power plants

Competition Description

History of competition

2023

5th competition

Understanding and utilizing the characteristics of various prediction models based on five developed solar power generation prediction models

→ Developing ensemble techniques to improve prediction performance

Competition Description

Review the provided data - gens.csv

	time	amount
0	2022-06-19 01:00:00+09:00	0.0
1	2022-06-19 02:00:00+09:00	0.0
2	2022-06-19 03:00:00+09:00	0.0
3	2022-06-19 04:00:00+09:00	0.0
4	2022-06-19 05:00:00+09:00	0.0
...	...	...
11611	2023-10-15 20:00:00+09:00	0.0
11612	2023-10-15 21:00:00+09:00	0.0
11613	2023-10-15 22:00:00+09:00	0.0
11614	2023-10-15 23:00:00+09:00	0.0
11615	2023-10-16 00:00:00+09:00	0.0

11616 rows × 2 columns

Actual power generation
of the target power plant

time: generation time

amount : generation amount

2023-10-15 20:00:00 means

from 2023-10-15 19:00:00 to

2023-10-15 19:59:59

Competition Description

Review the provided data – pred.csv

	round	time	model_id	amount
0	1	2022-06-19 01:00:00+09:00	0	0.0
1	1	2022-06-19 01:00:00+09:00	1	0.0
2	1	2022-06-19 01:00:00+09:00	2	0.0
3	1	2022-06-19 01:00:00+09:00	3	0.0
4	1	2022-06-19 01:00:00+09:00	4	0.0
...	...	...	...	...
116035	2	2023-10-16 00:00:00+09:00	0	0.0
116036	2	2023-10-16 00:00:00+09:00	1	0.0
116037	2	2023-10-16 00:00:00+09:00	2	0.0
116038	2	2023-10-16 00:00:00+09:00	3	0.0
116039	2	2023-10-16 00:00:00+09:00	4	0.0

116040 rows × 4 columns

Predicted power generation Data

model_id: Model ID of the predicted power generation

amount : predicted power generation by time slot

Unlike the gens.csv,

it is divided into round 1 and round 2

the models are also

classified into round 1 and round 2

predicted values at 10 a.m. and 5 p.m. are needed for power generation bidding!

Competition Description

Review the provided data – weather_actual.csv

	time	cloud	temp	humidity	ground_press	wind_speed	wind_dir	rain	snow	dew_point	vis	uv_idx	azimuth	elevation
0	2022-06-19 01:00:00+09:00	5.871524	23.030000	91.128476	1009.000000	2.394132	152.173538	0.0	0.0	20.193333	19.193333	0.0	6.704280	-31.529640
1	2022-06-19 02:00:00+09:00	5.000000	20.046829	92.000000	1009.000000	2.490000	133.000000	0.0	0.0	20.010169	16.100000	0.0	22.196370	-28.440428
2	2022-06-19 03:00:00+09:00	31.668514	20.275571	92.000000	1008.012749	2.340765	139.974501	0.0	0.0	20.304918	16.257377	0.0	35.919394	-22.437437
3	2022-06-19 04:00:00+09:00	100.000000	20.380388	93.000000	1008.000000	2.770000	142.000000	0.0	0.0	20.403077	19.004615	0.0	47.557714	-14.221450
4	2022-06-19 05:00:00+09:00	100.000000	22.030000	93.000000	1008.000000	2.557647	133.882353	0.0	0.0	20.495385	10.143077	0.0	57.378183	-4.444699
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
11611	2023-10-15 20:00:00+09:00	0.000000	18.807459	70.000000	1014.000000	6.320000	307.000000	0.0	0.0	13.204762	16.100000	0.0	277.464745	-25.379191
11612	2023-10-15 21:00:00+09:00	0.000000	17.918518	67.888518	1015.000000	5.553144	306.554073	0.0	0.0	13.300000	16.100000	0.0	287.678638	-37.409688
11613	2023-10-15 22:00:00+09:00	0.000000	17.030000	67.000000	1015.000000	5.100000	303.000000	0.0	0.0	13.055738	20.349180	0.0	301.007172	-48.655175
11614	2023-10-15 23:00:00+09:00	0.000000	18.730542	67.000000	1015.000000	5.190000	297.000000	0.0	0.0	12.183333	9.590000	0.0	320.433966	-58.056463
11615	2023-10-16 00:00:00+09:00	0.000000	14.030000	66.000000	1015.000000	5.360000	293.000000	0.0	0.0	11.700000	6.400000	0.0	349.065111	-63.421759

11616 rows × 14 columns

Providing actual measured values

for 13 weather variables in float format

Competition Description

Round1 : data predicted at 10 a.m.

Round2 : data predicted at 5 p.m.

Review the provided data – weather_forecast.csv

	round	time	cloud	temp	humidity	ground_press	wind_speed	wind_dir	rain	snow	dew_point	vis	uv_idx	azimuth	elevation
0	1	2022-06-19 01:00:00+09:00	6.0	20.03	93.0	1009.0	3.01	162.0	0.0	0.0	18.3333	16.0934	0.0	6.70428	-31.5296
1	1	2022-06-19 02:00:00+09:00	7.0	19.88	95.0	1009.0	3.16	159.0	0.0	0.0	18.3333	16.0934	0.0	22.19640	-28.4404
2	1	2022-06-19 03:00:00+09:00	17.0	19.99	96.0	1008.0	2.92	161.0	0.0	0.0	18.3333	16.0934	0.0	35.91940	-22.4374
3	1	2022-06-19 04:00:00+09:00	100.0	20.19	96.0	1008.0	2.79	157.0	0.0	0.0	17.7778	16.0934	0.0	47.55770	-14.2214
4	1	2022-06-19 05:00:00+09:00	100.0	20.34	95.0	1008.0	2.74	156.0	0.0	0.0	18.3333	16.0934	0.0	57.37820	-4.4447
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
23187	2	2023-10-15 20:00:00+09:00	0.0	18.51	69.0	1015.0	5.56	328.0	0.0	0.0	12.7778	16.0934	0.0	277.46500	-25.3792
23188	2	2023-10-15 21:00:00+09:00	0.0	18.59	70.0	1015.0	5.25	317.0	0.0	0.0	12.7778	16.0934	0.0	287.67900	-37.4097
23189	2	2023-10-15 22:00:00+09:00	0.0	18.68	69.0	1015.0	5.58	310.0	0.0	0.0	12.7778	16.0934	0.0	301.00700	-48.6552
23190	2	2023-10-15 23:00:00+09:00	0.0	18.77	66.0	1015.0	5.75	306.0	0.0	0.0	12.2222	16.0934	0.0	320.43400	-58.0565
23191	2	2023-10-16 00:00:00+09:00	0.0	18.71	66.0	1015.0	5.85	306.0	0.0	0.0	12.2222	16.0934	0.0	349.06500	-63.4218

23192 rows × 15 columns

Providing actual measured values

for 13 weather variables in float format

Competition Description

Review the provided data– incentive.csv

	time	model_id	incentive	is_utilizable
0	2022-06-19 01:00:00+09:00	0	0	0
1	2022-06-19 01:00:00+09:00	1	0	0
2	2022-06-19 01:00:00+09:00	2	0	0
3	2022-06-19 01:00:00+09:00	3	0	0
4	2022-06-19 01:00:00+09:00	4	0	0
...	...	...	...	...
57955	2023-10-16 00:00:00+09:00	0	0	0
57956	2023-10-16 00:00:00+09:00	1	0	0
57957	2023-10-16 00:00:00+09:00	2	0	0
57958	2023-10-16 00:00:00+09:00	3	0	0
57959	2023-10-16 00:00:00+09:00	4	0	0

57960 rows × 4 columns

Incentives for predicted power generation data

model_id: Model ID of the predicted generation

incentive: Settlement amount
earned at the bidding time

is_utilizable:
Checking whether the power generation of the power plant at the respective time slot meets the equipment utilization rate

We will review this with the incentive evaluation formula afterwards

Competition Description

API

HTTP request method

A way to inform the server in web communication

what operation is being requested

GET	Retrieve server data
PUT	Modify server data
DELETE	Delete server data
POST	Add server data

Competition Description

API

HTTP request method

A way to inform the server in web communication

what operation is being requested

GET — Retrieve server data

Weather forecast/actual data

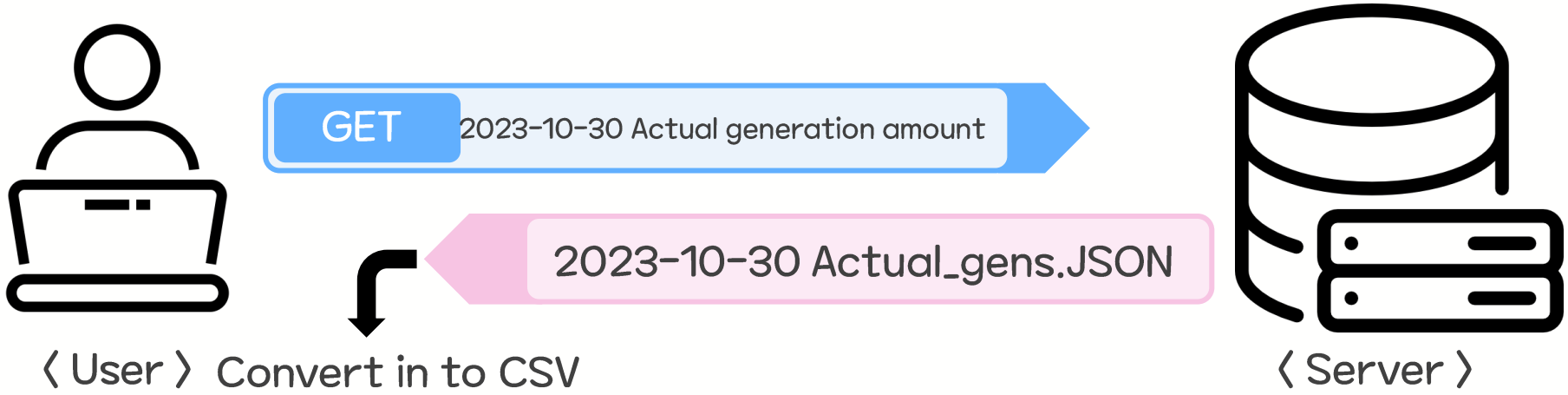
Power generation forecast/actual data

DELETE — Proceed with bidding for power generation forecast values

POST — Add server data

Competition Description

API - Data request



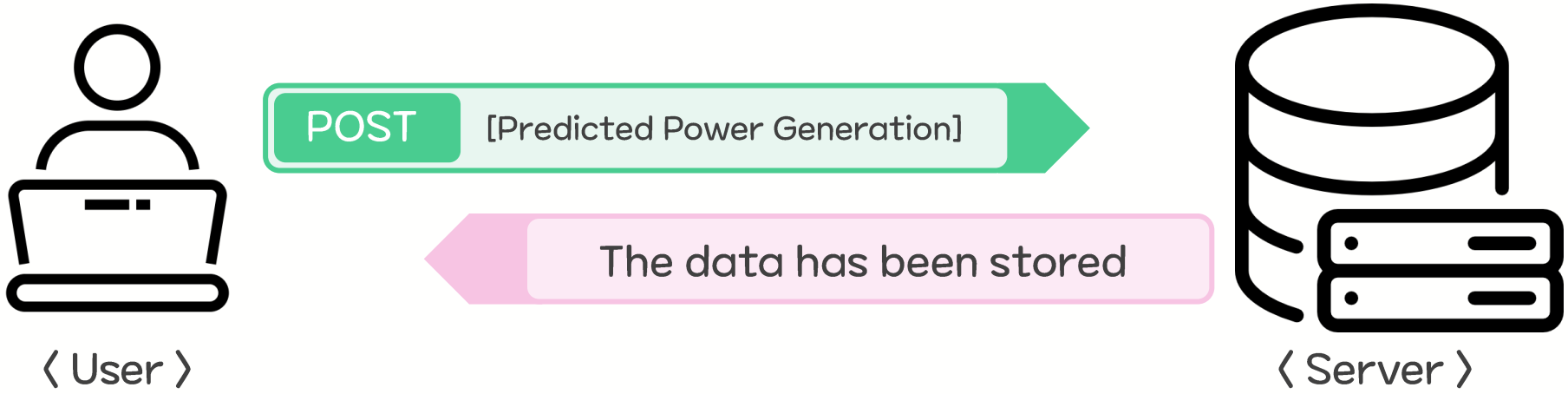
Request data for a specific date from the server

Receive JSON file and convert it to CSV and storage

Configured in key-value pairs
similar to a Python Dictionary

Competition Description

API – Prediction bidding



Bidding the predicted power generation in list format

Automatically considered as a bid for the day following the bid submission date

Competition Description

API – Prediction bidding

During the competition period, forecasted generation/weather data, actual weather data, and model-specific incentive settlement results (etc.) will be provided through the API

Competition Description

Competition Rules Explanation - Forecast Error Rate

$$\frac{\left|\frac{\widehat{G_h^{10}} + \widehat{G_h^{17}}}{2} - G_h\right|}{C} \times 100$$

10시와 17시 예측된 발전량의 평균과

실제 발전량을 비교하여 예측 오차율 정의

Competition Description

Competition Rules Explanation - Forecast Error Rate

$$\frac{\left|\frac{\widehat{G_h^{10}} + \widehat{G_h^{17}}}{2} - G_h\right|}{C} \times 100$$

C : Solar power plant capacity (kW)

G_h : Actual generation amount of the solar power plant at hour h

$\widehat{G_h^n}$: Forecasted generation amount for hour h inputted at hour n the previous day

Competition Description

Competition Rules Explanation - Forecast Error Rate

$$\frac{\left|\frac{\widehat{G_h^{10}} + \widehat{G_h^{17}}}{2} - G_h\right|}{C} \times 100$$

C is fixed at 99kW!

C : Solar power plant capacity (kW)

G_h : Actual generation amount of the solar power plant at hour h

$\widehat{G_h^n}$: Forecasted generation amount for hour h inputted at hour n the previous day

Competition Description

Competition Rules Explanation - Forecast Error Rate

Since the panels used in the competition are small,

the power plant capacity is expected to be fixed at 99 kW

Competition Description

Competition Rules Explanation - Forecast Error Rate

If the panel is small, even a small cloud can
significantly change the generation amount, and this is
assumed to be the cause of the spikes point shown in the graph

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

Forecast incentives are calculated according to the forecast error rate for each time period

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

G_h: Actual generation amount of the solar power plant at hour h

$$\sum_{h=1}^{24} G_h \times I_h$$

Forecast error rate per time period 6% or less : I_h = 4KRW/kWh

Forecast error rate per time period exceeding 6% and up to 8%: I_h = 3KRW/kWh

Forecast error rate per time period exceeding 8% : I_h = 0KRW/kWh

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

So, is it enough to just have a low forecast error rate?

G_h: Actual generation amount of the solar power plant at hour h

$$\sum_{h=1}^{24} G_h \times I_h$$

Forecast error rate per time period 6% or less : I_h = 4KRW/kWh

Forecast error rate per time period exceeding 6% and up to 8%: I_h = 3KRW/kWh

Forecast error rate per time period exceeding 8% : I_h = 0KRW/kWh

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

So, is it enough to just have a low forecast error rate?

G_h: Ratio of actual generation amount to solar power plant capacity

$$\sum_{h=1}^{24} G_h \times I_h$$

The condition of the power plant utilization rate for each time period must be met!!

Forecast error rate per time period 6% or less : I_h = 4KRW/kWh

Forecast error rate per time period exceeding 6% and up to 8%: I_h = 3KRW/kWh

Forecast error rate per time period exceeding 8% : I_h = 0KRW/kWh

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

So, is it enough to just have a low forecast error rate?

G_h: Actual generation amount of the solar power plant at hour h

$$\sum_{h=1}^{24} G_h \times I_h$$

The condition of the power plant utilization rate for each time period must be met!!

If the utilization rate is less than 10%, it is excluded from the error rate calculation, and no incentives are given!!

Forecast error rate per time period 6% or less : I_h = 4KRW/kWh

Forecast error rate per time period exceeding 6% and up to 8% : I_h = 3KRW/kWh

Forecast error rate per time period exceeding 8% : I_h = 0KRW/kWh

➔ Check the is_utilizable variable in the incentive.csv data!

Competition Description

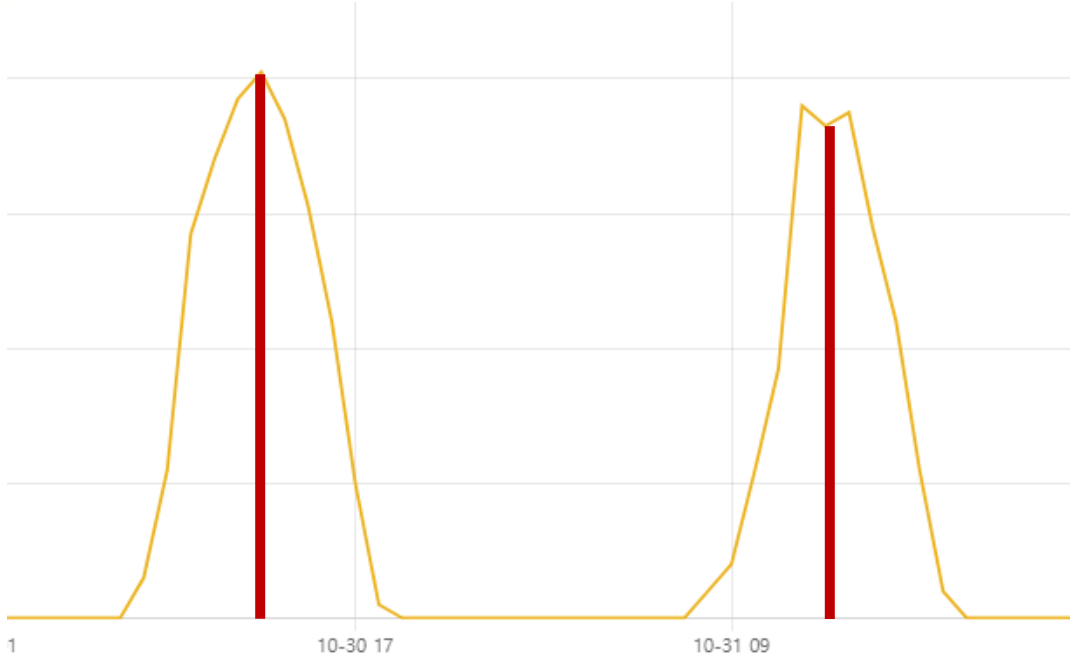
Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

During the competition period, the higher the total forecast incentive (settlement amount) sum, the better the evaluation

Competition Description

Competition Rules Explanation - Forecast Incentive (Settlement) Calculation

$$\sum_{h=1}^{24} G_h \times I_h$$



it is important to reduce the error rate during

peak hours when the generation amount is high

EDA

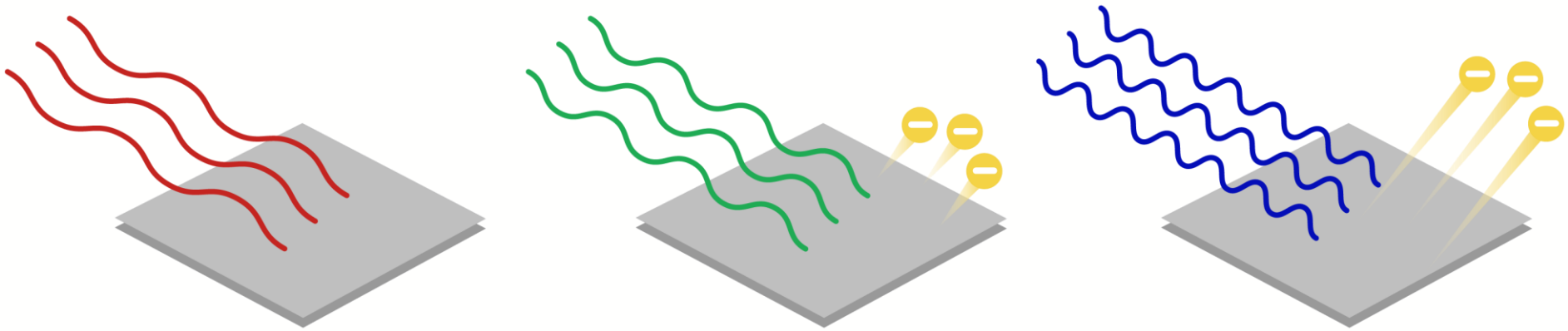
EDA

Principle of Solar Power Generation

Photoelectric Effect

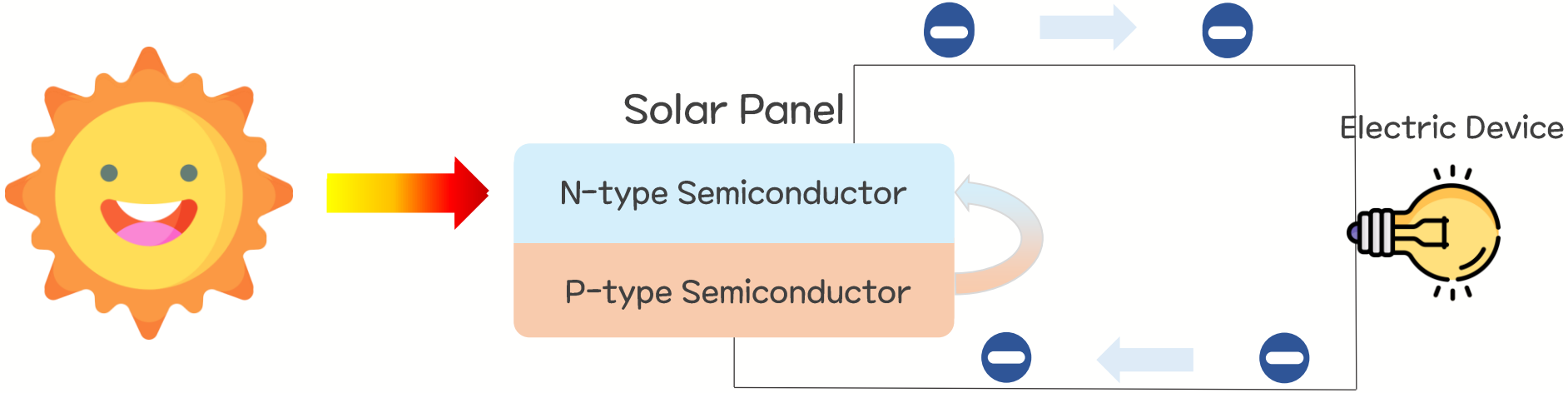
When light is applied to certain metals,

electrons are emitted from the metal



EDA

Principle of Solar Power Generation

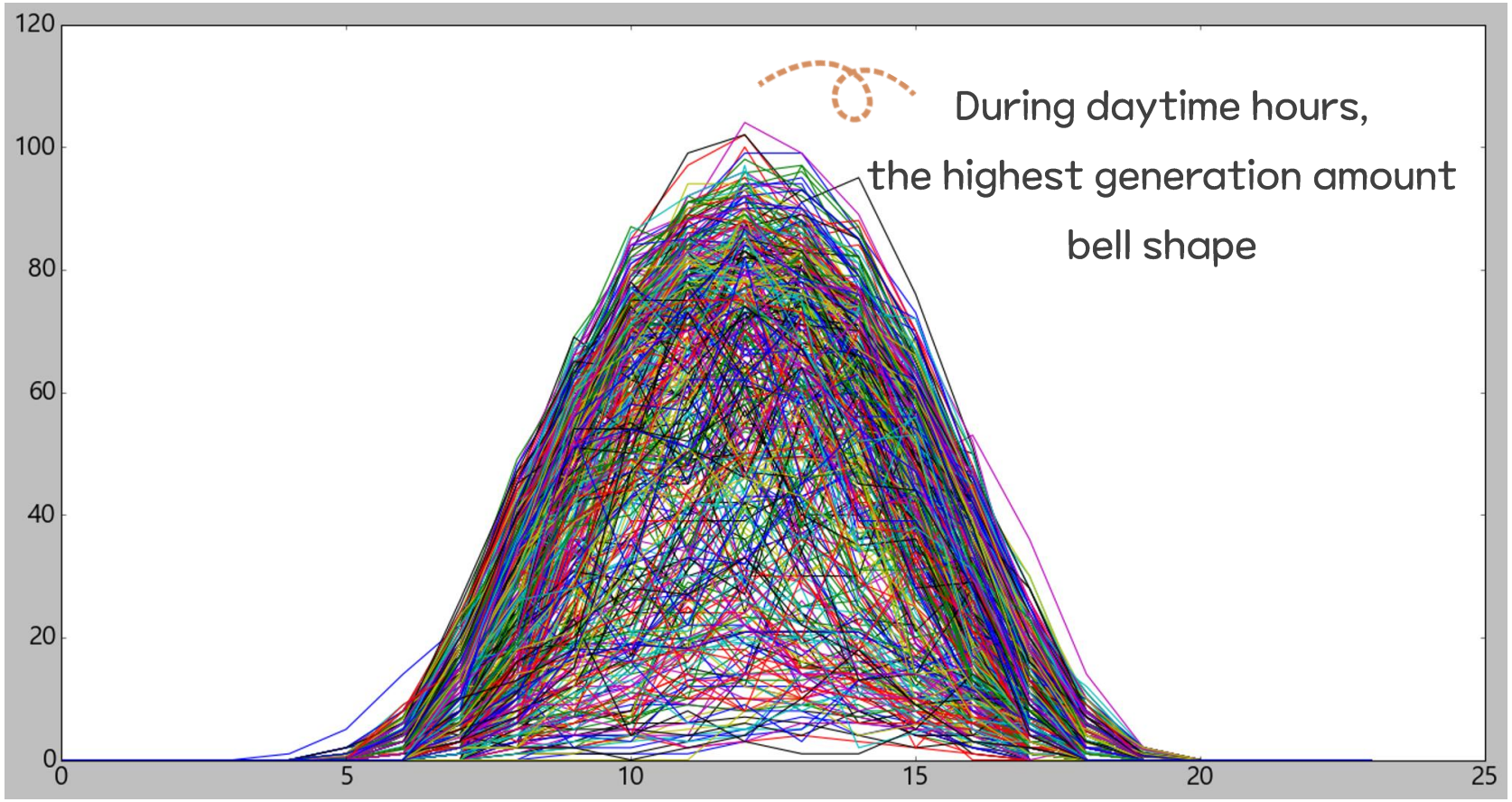


When electrons are emitted from semiconductors of different types,
electron-hole pairs are generated, causing electricity to flow

The empty space left after the electrons move

EDA

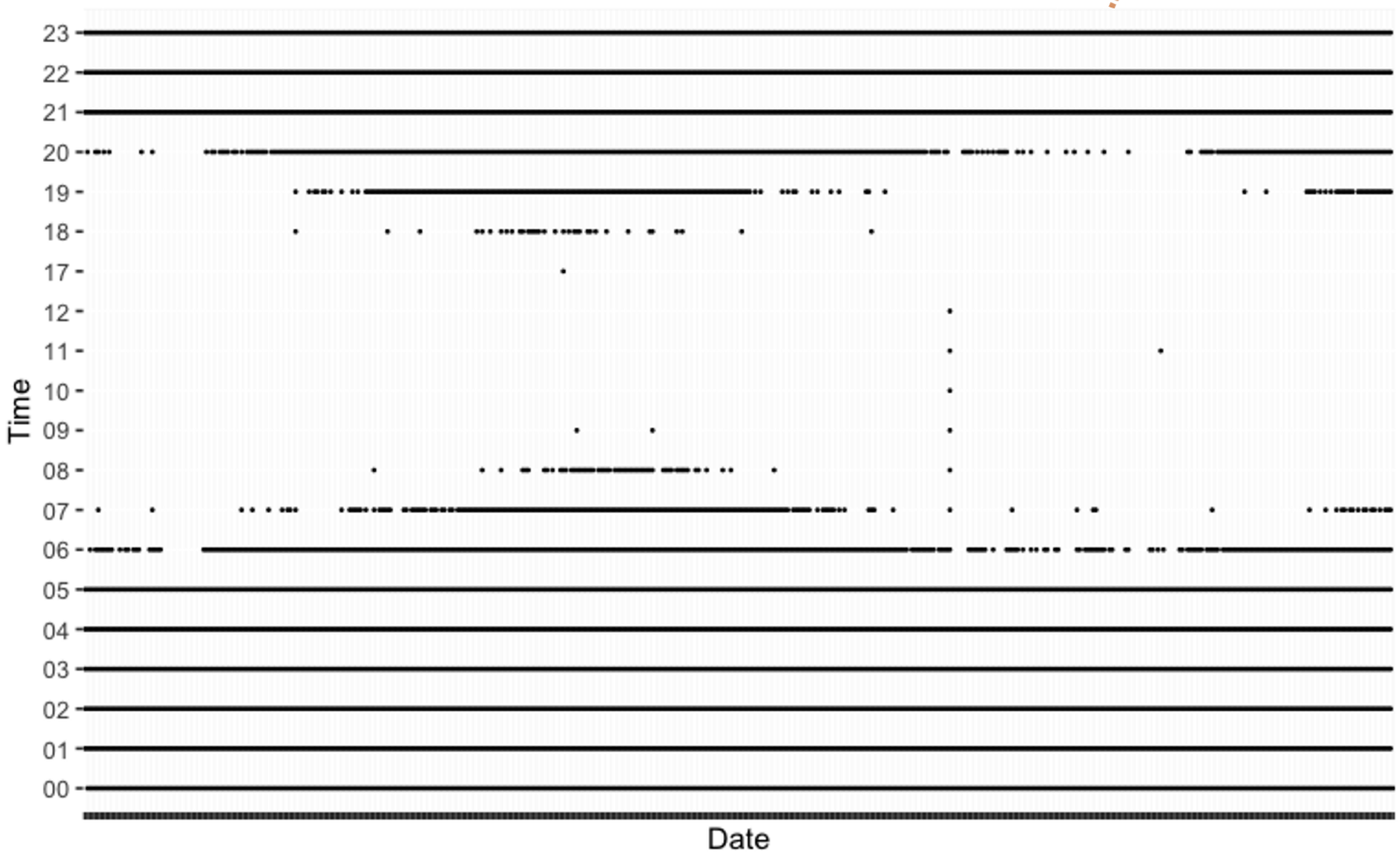
Actual generation amount



EDA

Actual generation amount

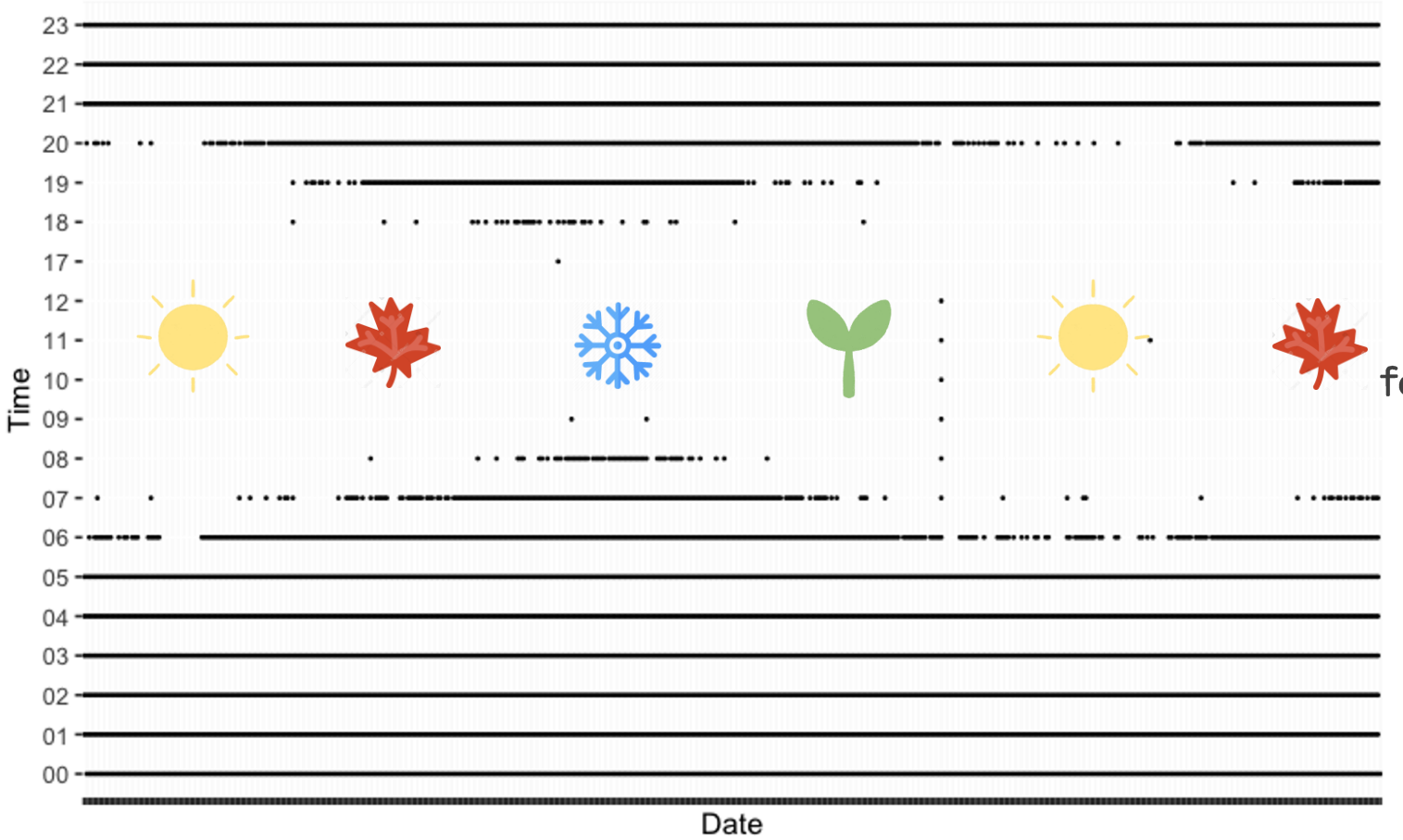
Plot for generation amount = 0



EDA

Actual generation amount

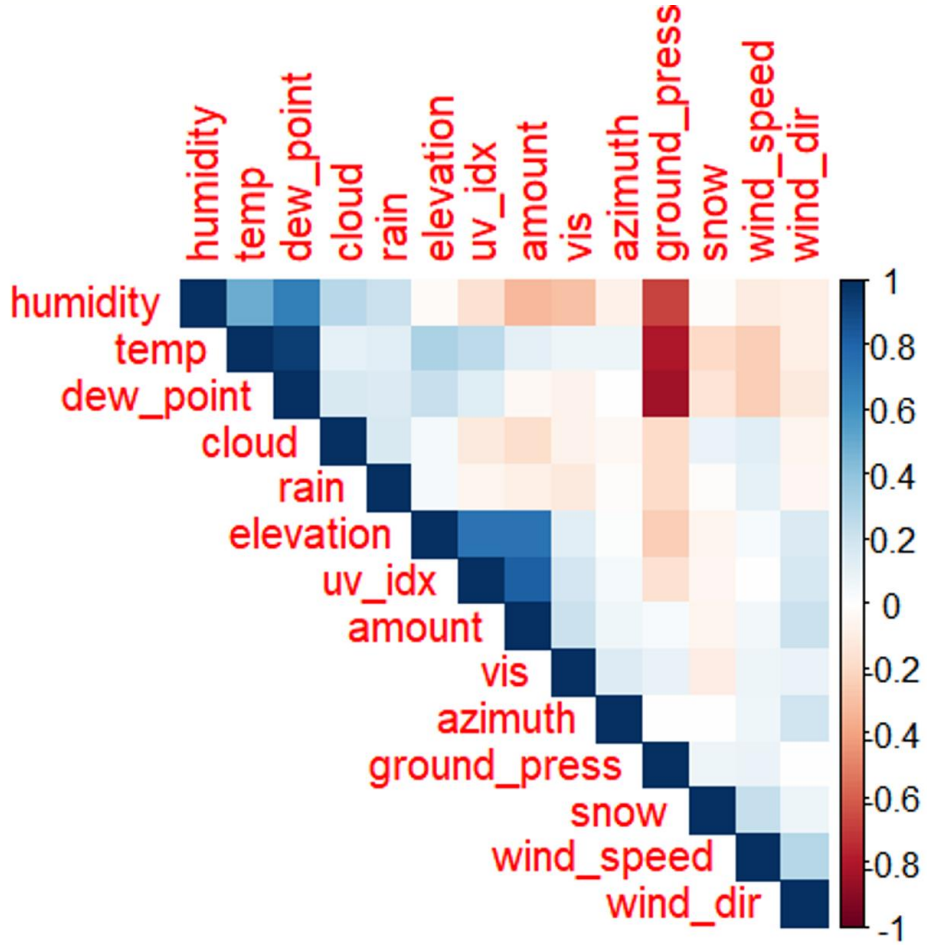
Plot for generation amount = 0



Depending on the sunlight hours
for each season, there is a difference
in the time periods when
the power generation is zero

EDA

Correlation : weather vs generation amount



Positive correlation :

elevation), uv_idx

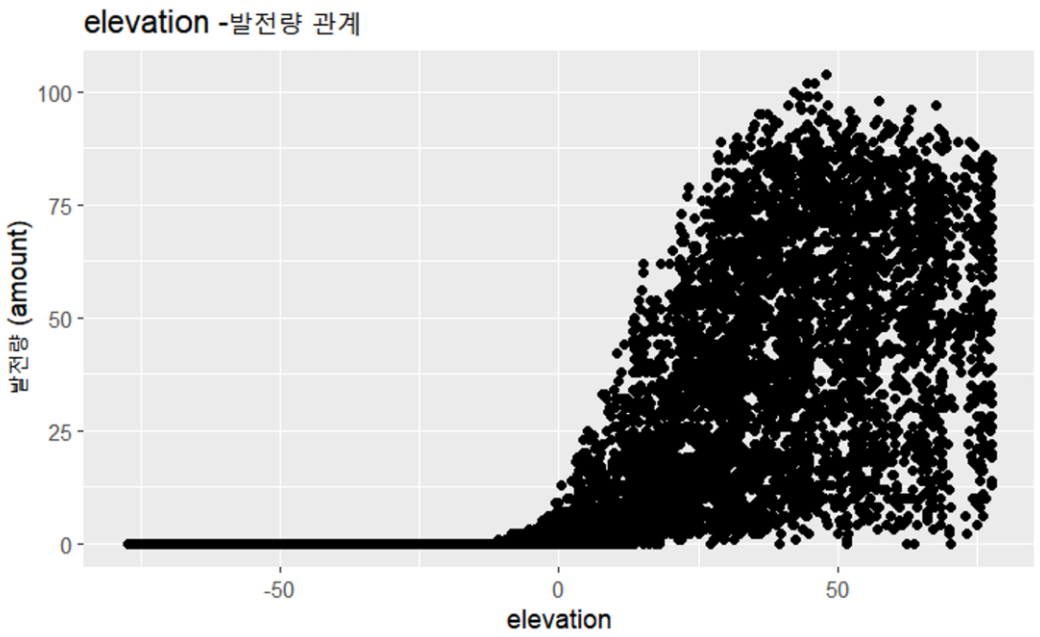
Negative correlation :

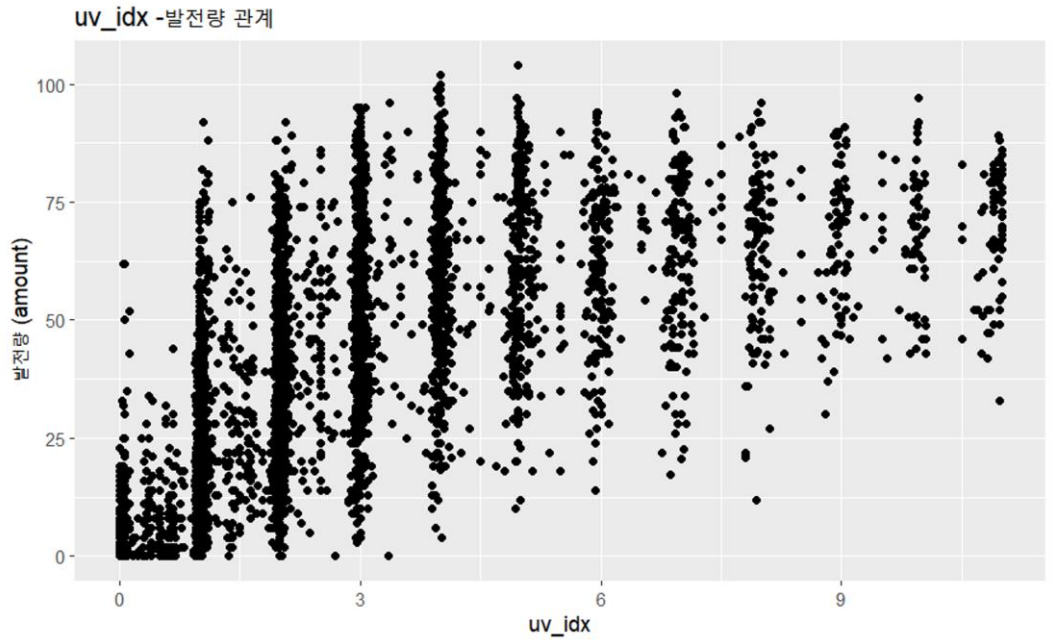
cloud, rain,

snow, humidity

EDA

Correlation – positive



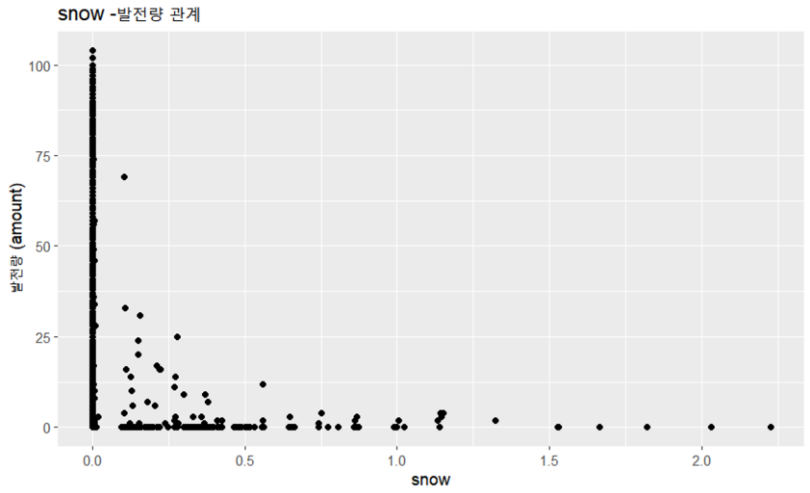


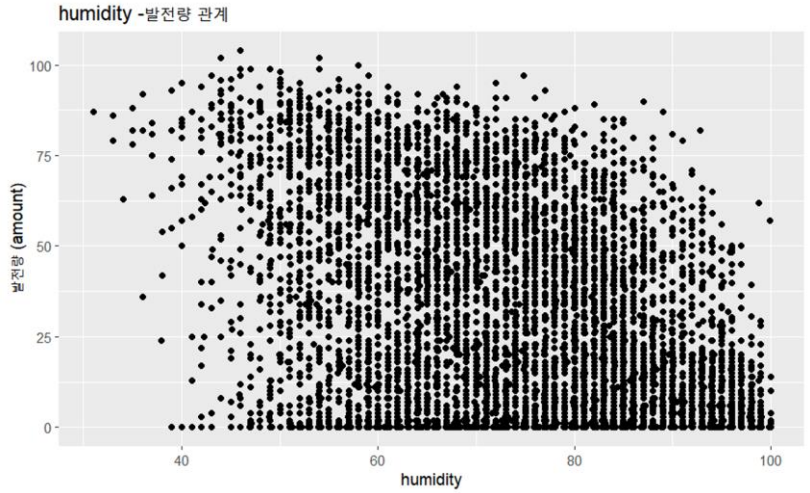
Elevation and UV_idx has high

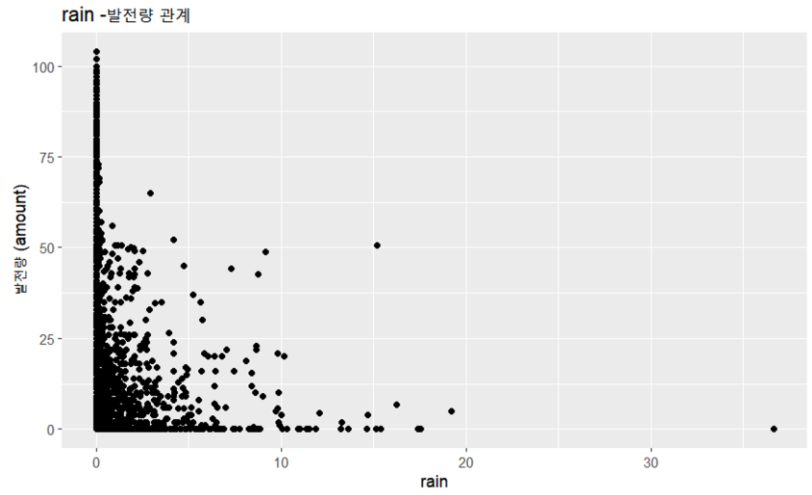
positive correlation with generation amount

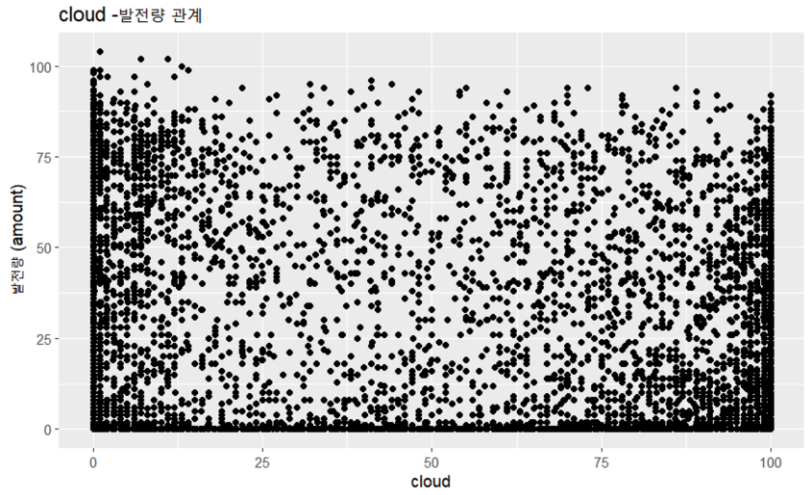
EDA

Correlation – negative



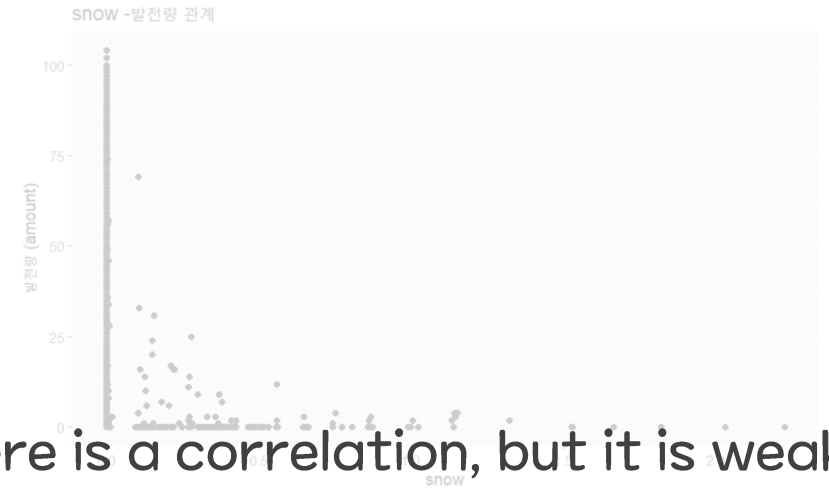






EDA

Correlation – negative

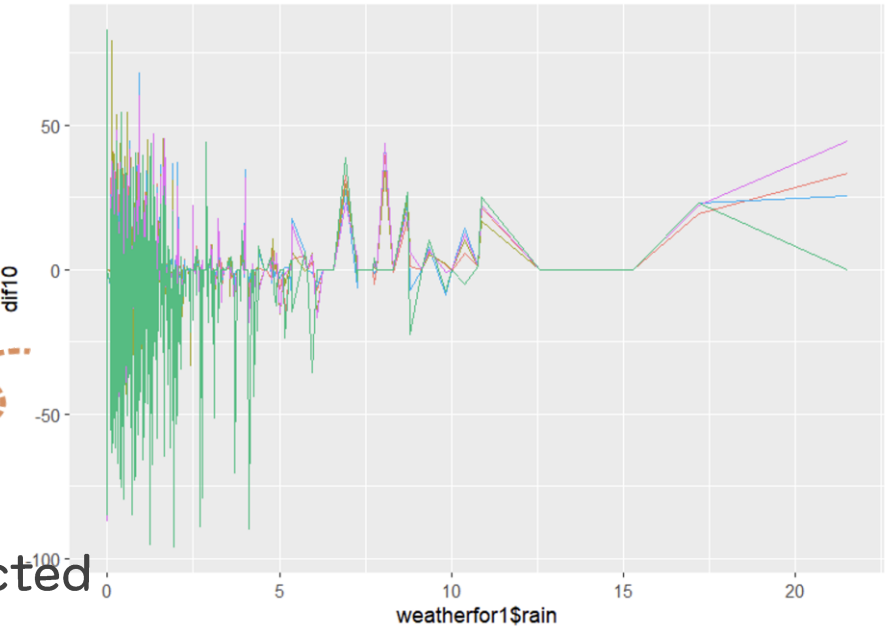


There is a correlation, but it is weak and does not appear to be significant

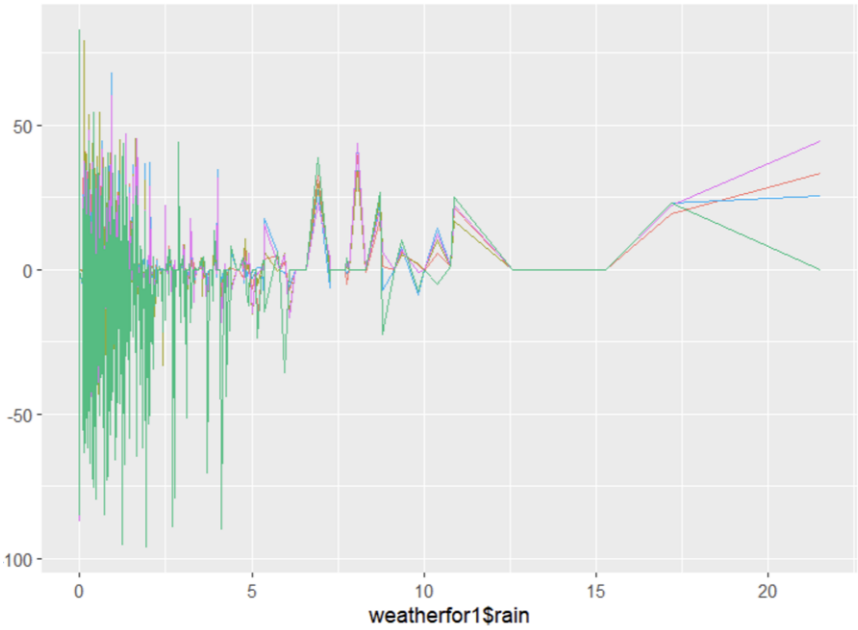
EDA

Relation between forecast weather vs predicted generation

Actual weather



Weather forecast



Y :

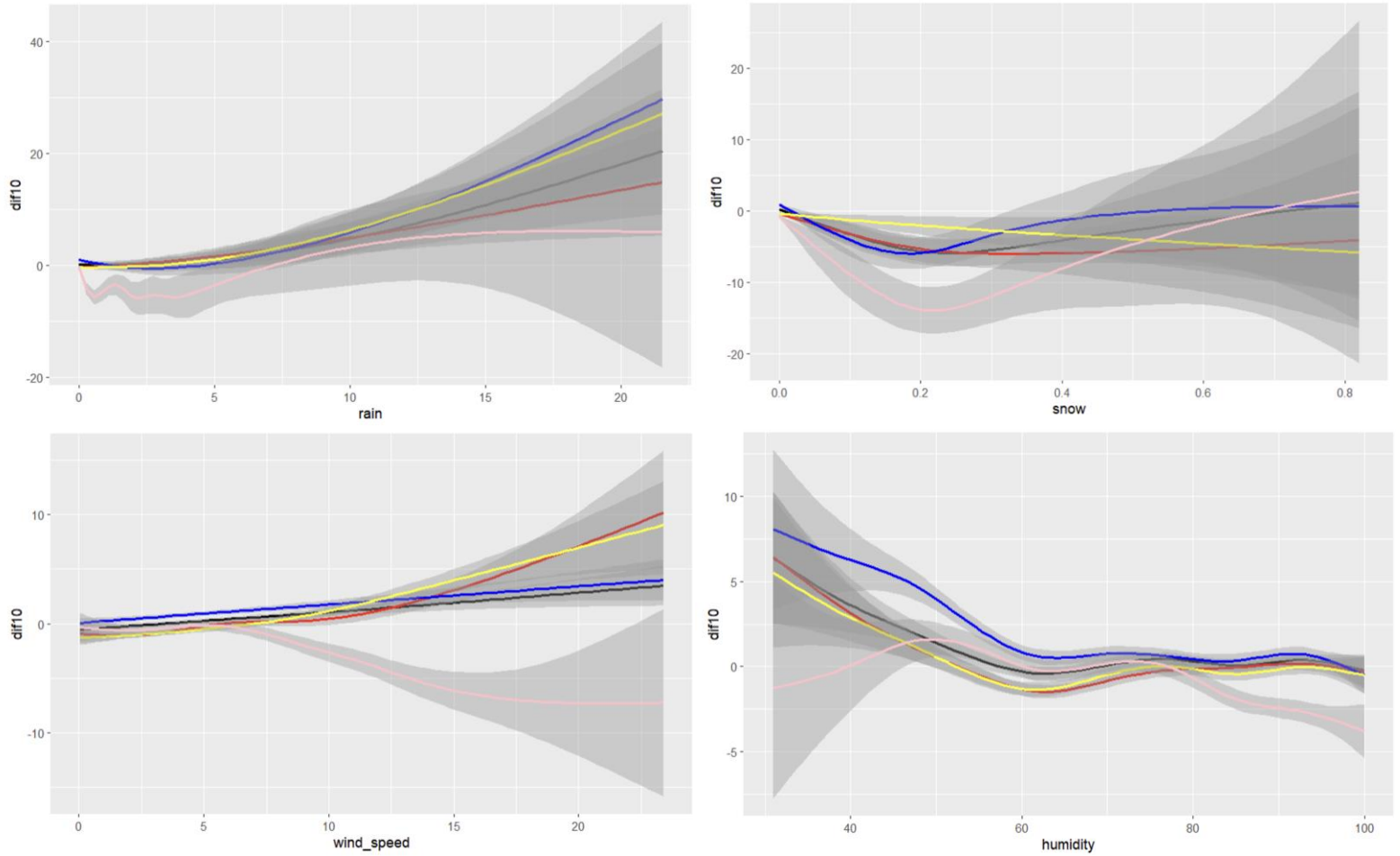
actual – predicted

In the case of Model 4, there is a tendency

for the forecasted generation amount to be higher on rainy days

EDA

Relation between forecast weather vs predicted generation



Model 4

(Pink line)

Different pattern compared to other models

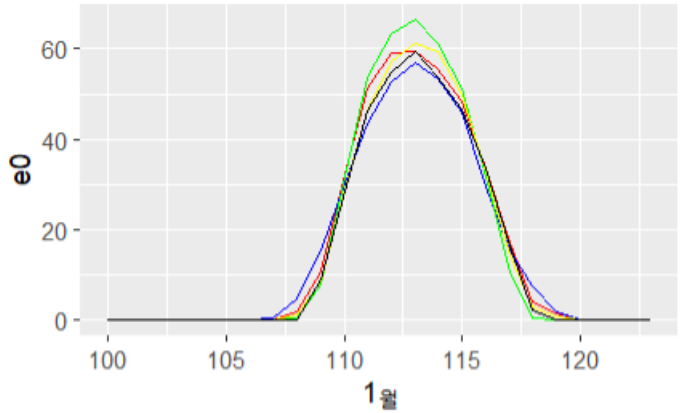
↓

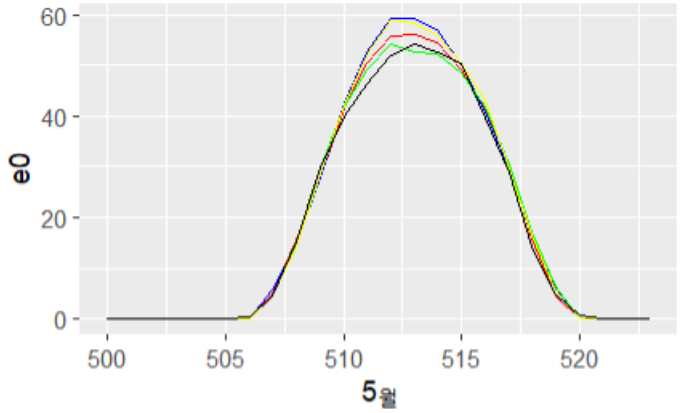
The standard error is significantly larger compared to other models

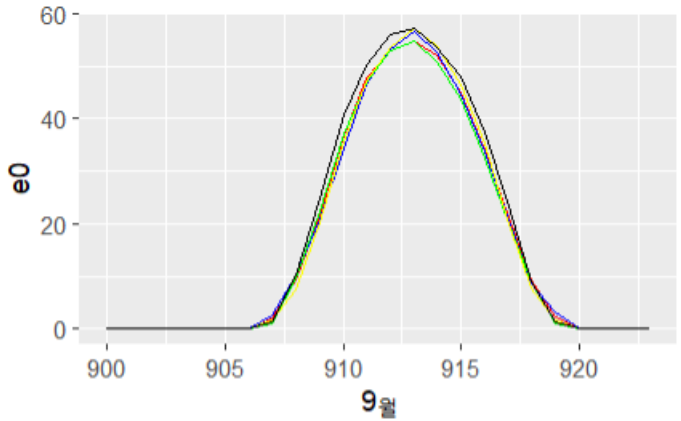
Function : geom_smooth (trend line)

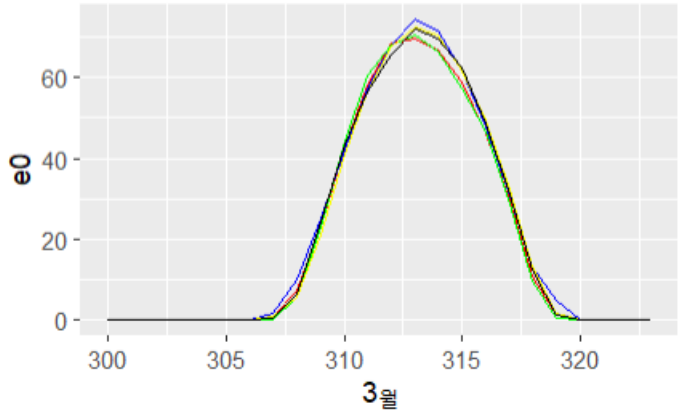
EDA

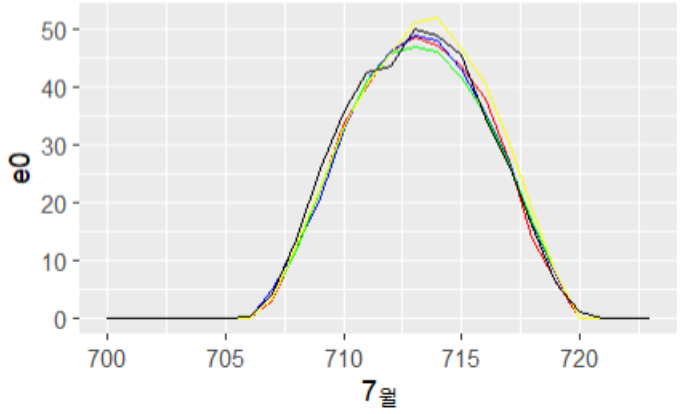
Relation between seasons vs predicted generation

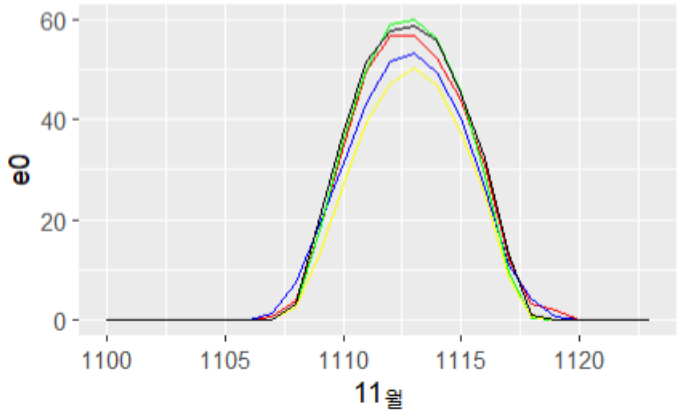










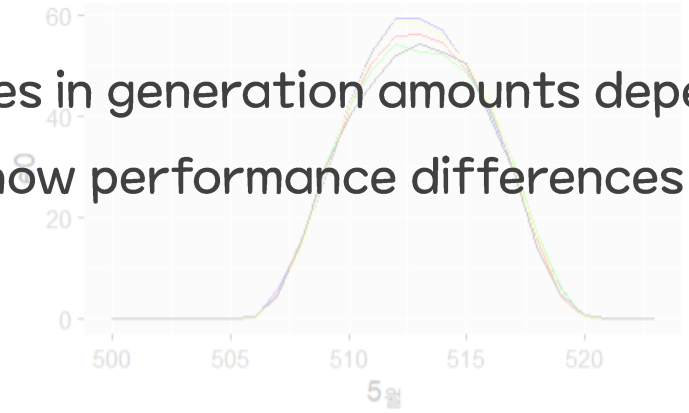


EDA

Relation between seasons vs predicted generation

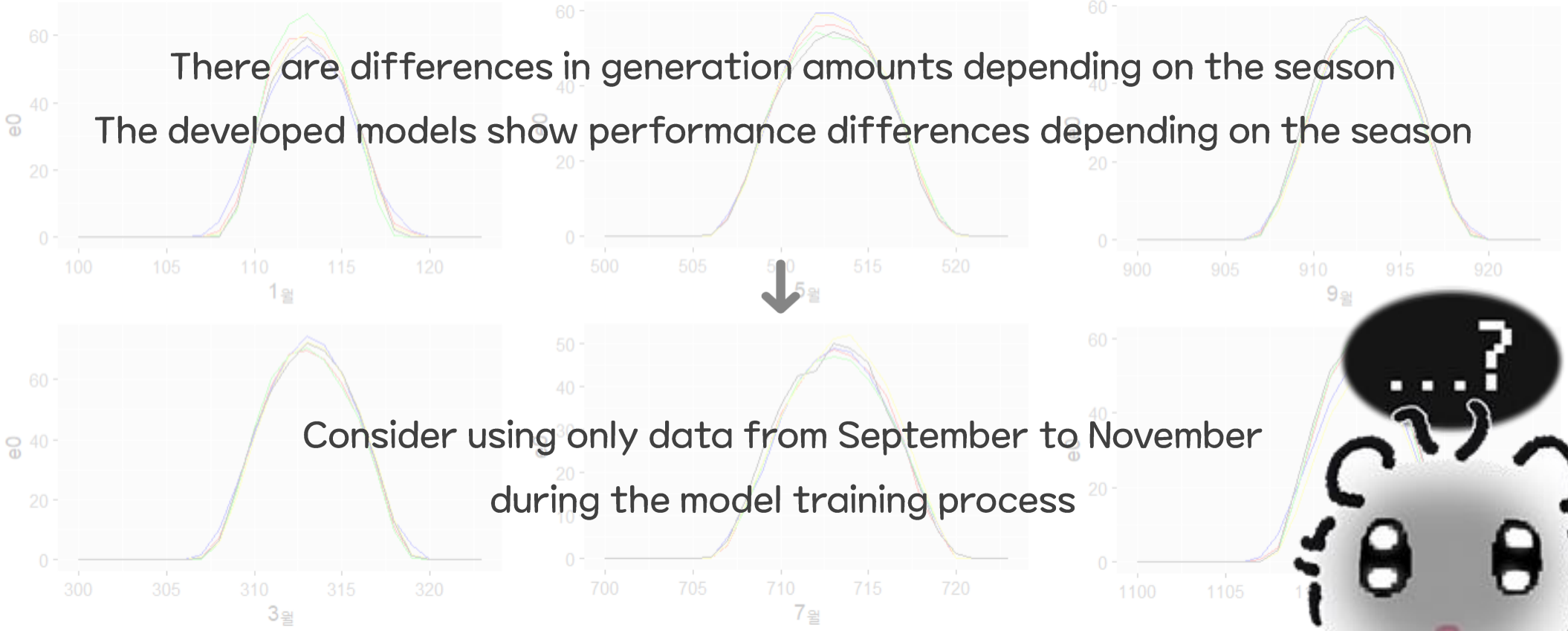
There are differences in generation amounts depending on the season

The developed models show performance differences depending on the season



EDA

Relation between seasons vs predicted generation



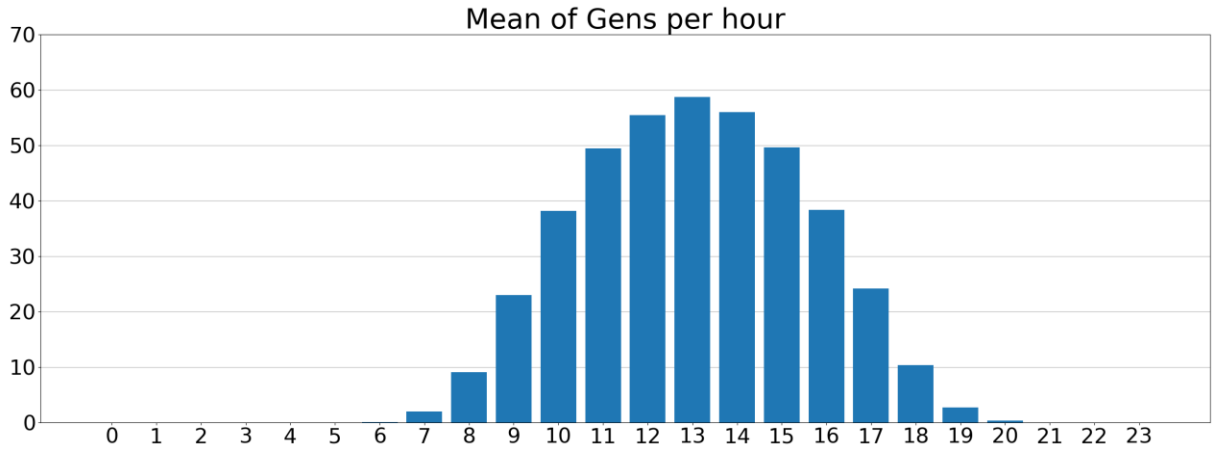
There are differences in generation amounts depending on the season

The developed models show performance differences depending on the season

Consider using only data from September to November during the model training process

EDA

Definition of Derived Variables – Time Transformation



Visualize the average of the actual generation amount for each time period

Observe the symmetrical shape with 1 PM as the reference

EDA

Definition of Derived Variables – Time Transformation

Transform time variable

through sin function

Maximum value at 1 PM

24-hour × 2 cycles

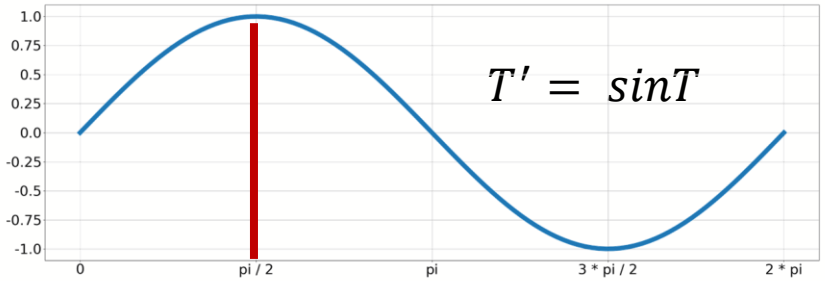
Visualize the average of the actual generation amount for each time period

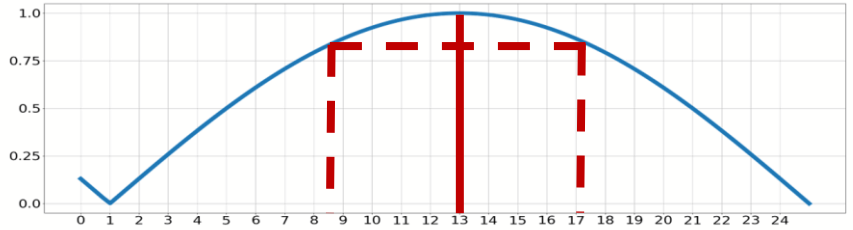
Observe the symmetrical sha ith 1 PM as the reference

EDA

Definition of Derived Variables – Time Transformation

- Multiply by $\pi/24$ to change the period
- Change the maximum value coordinates through translation
- Change to positive values using the absolute value





$$T' = \left|\sin\frac{\pi}{24}(T-1)\right|$$

The goal is to change the symmetrical points to the same value

EDA

Definition of Derived Variables – Time Transformation

Month Variable Value	Season Classification
3,4,5	Spring
6,7,8,9	Summer
10,11	Fall
12,1,2	Winter

Identify the season using the month variable

Distinguish the four seasons using One-Hot encoding method

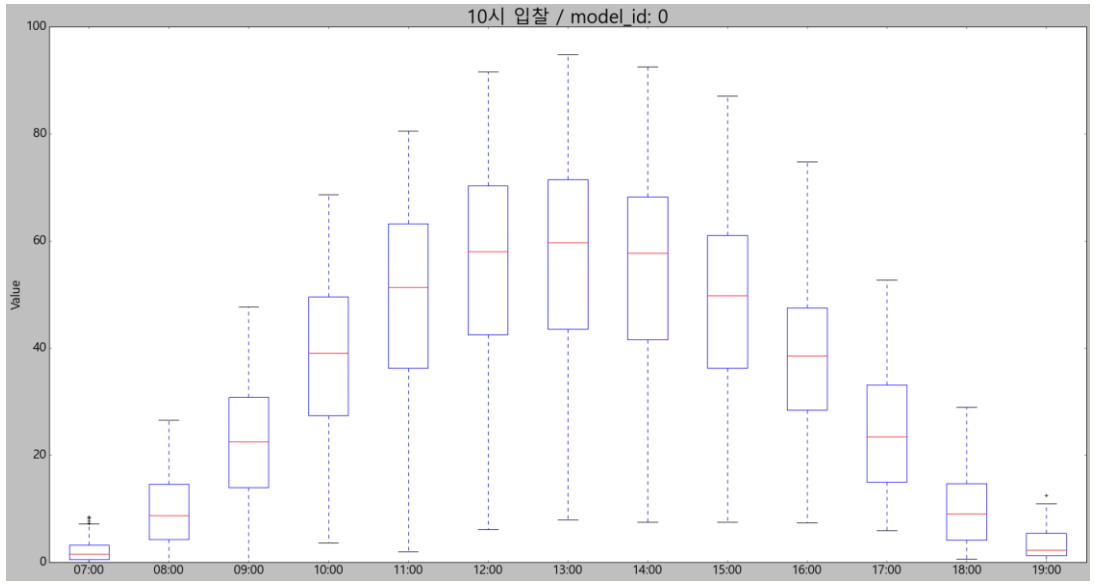
```python
def spring(m):
    if (m >= 3)  & (m <= 5):
        return 1
    else:
        return 0

def summer(m):
    if (m >= 6)  & (m <= 9):
        return 1
    else:
        return 0

def fall(m):
    if (m >= 10)  & (m <= 11):
        return 1
    else:
        return 0

def winter(m):
    if (m == 12)  & (m <= 2):
        return 1
    else:
        return 0
```

```python
train['spring'] = train['month'].apply(lambda m: spring(m))
train['summer'] = train['month'].apply(lambda m: summer(m))
train['fall'] = train['month'].apply(lambda m: fall(m))
train['winter'] = train['month'].apply(lambda m: winter(m))
```

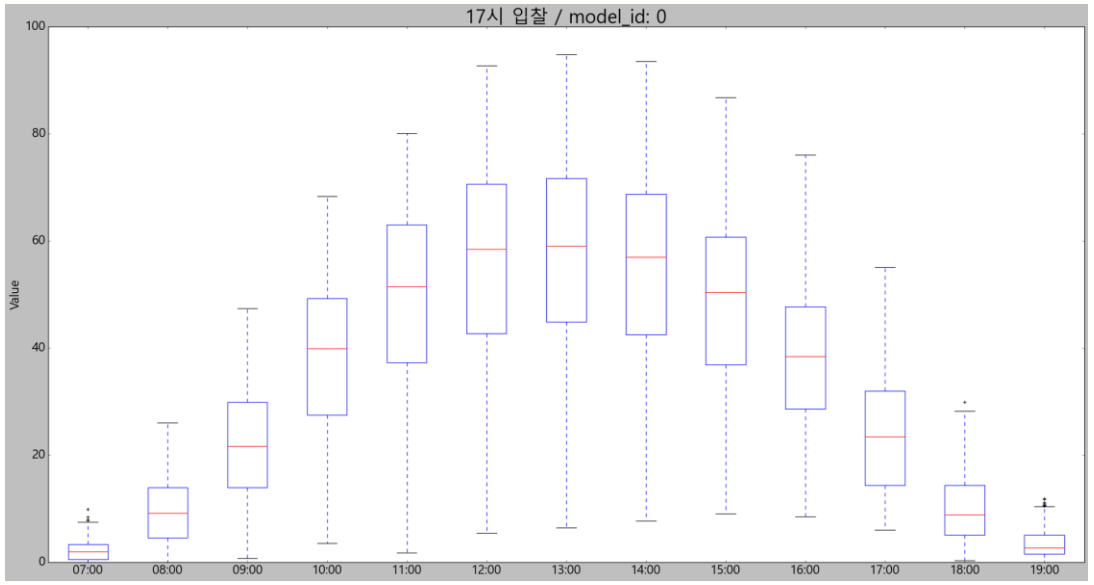
EDA

Predicted generation amount – box plot

Model 0 : Bid at 10 a.m. (round1)



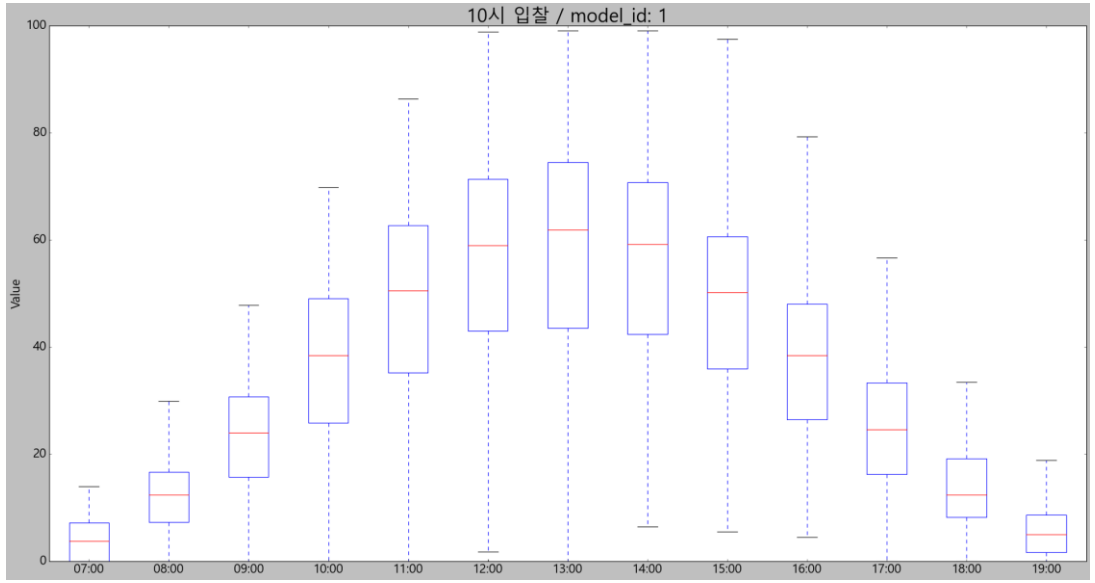
Model 0 :Bid at 17 p.m. (round2)



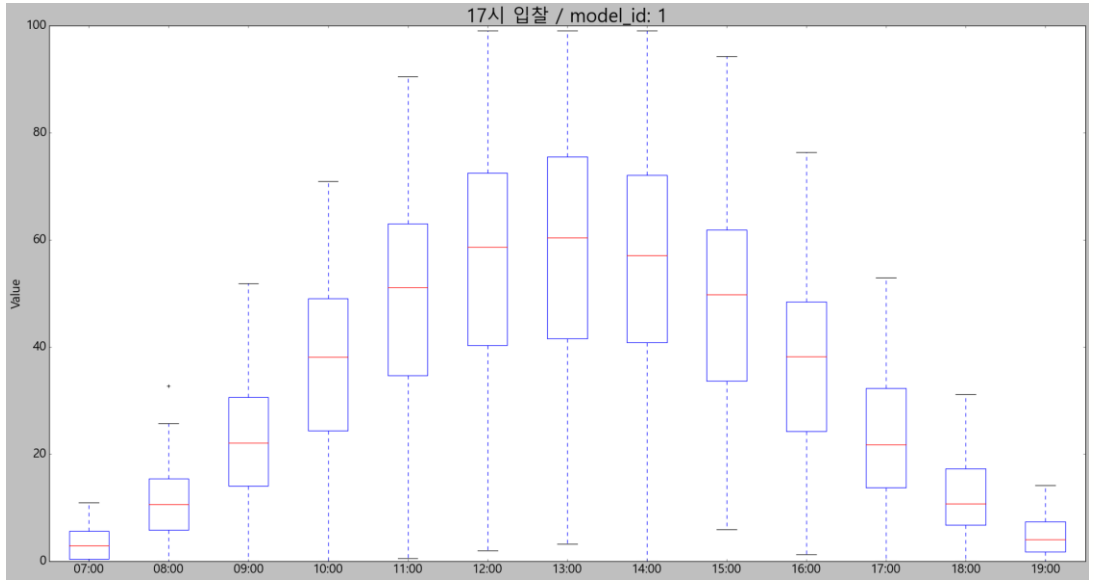
EDA

Predicted generation amount – box plot

Model 1 : Bid at 10 a.m. (round1)



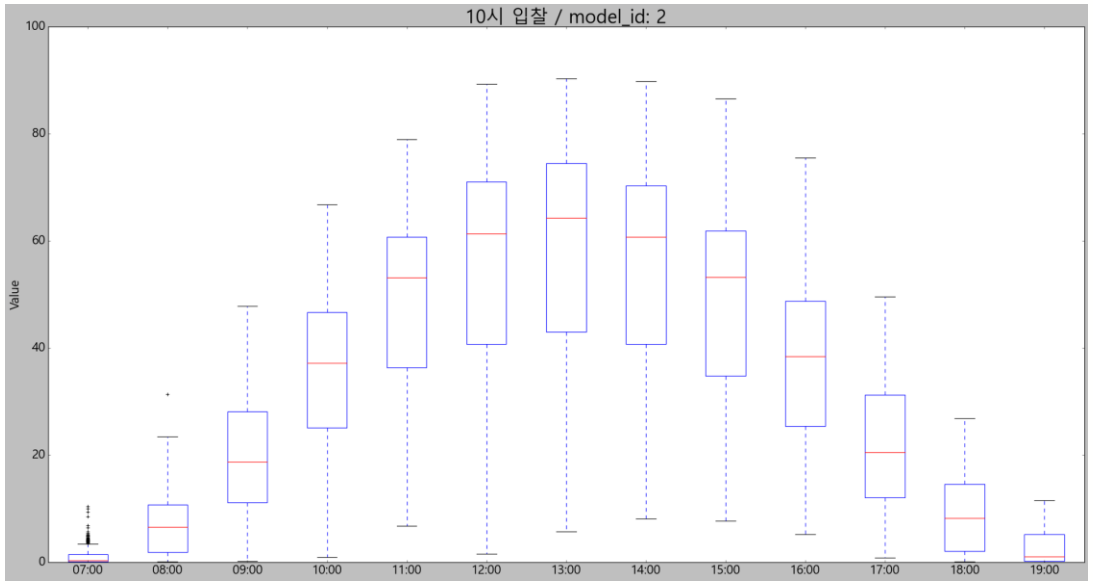
Model 1 :Bid at 17 p.m. (round2)



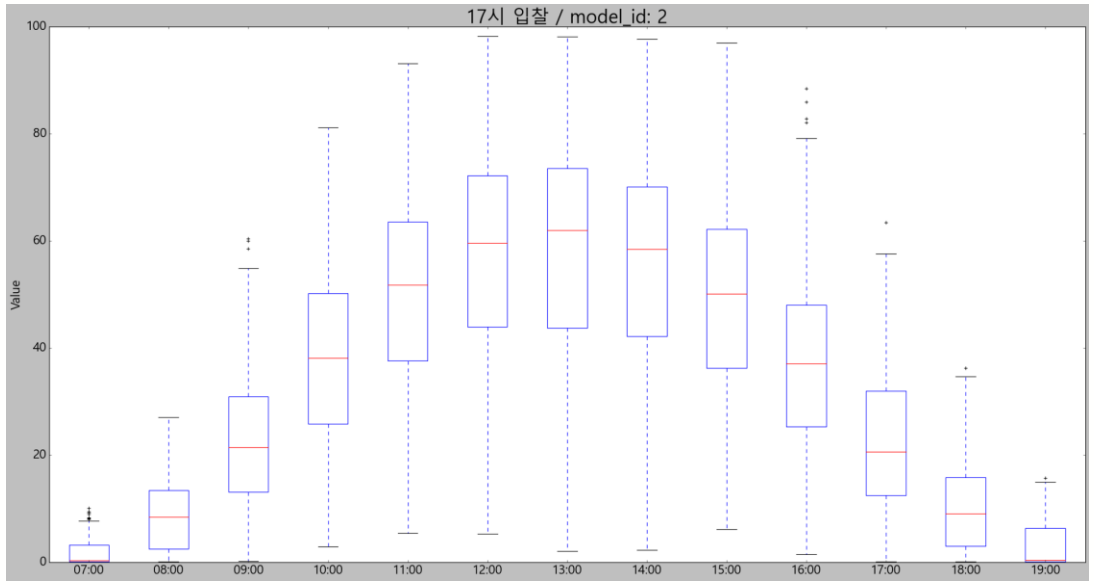
EDA

Predicted generation amount – box plot

Model 2 : Bid at 10 a.m. (round1)



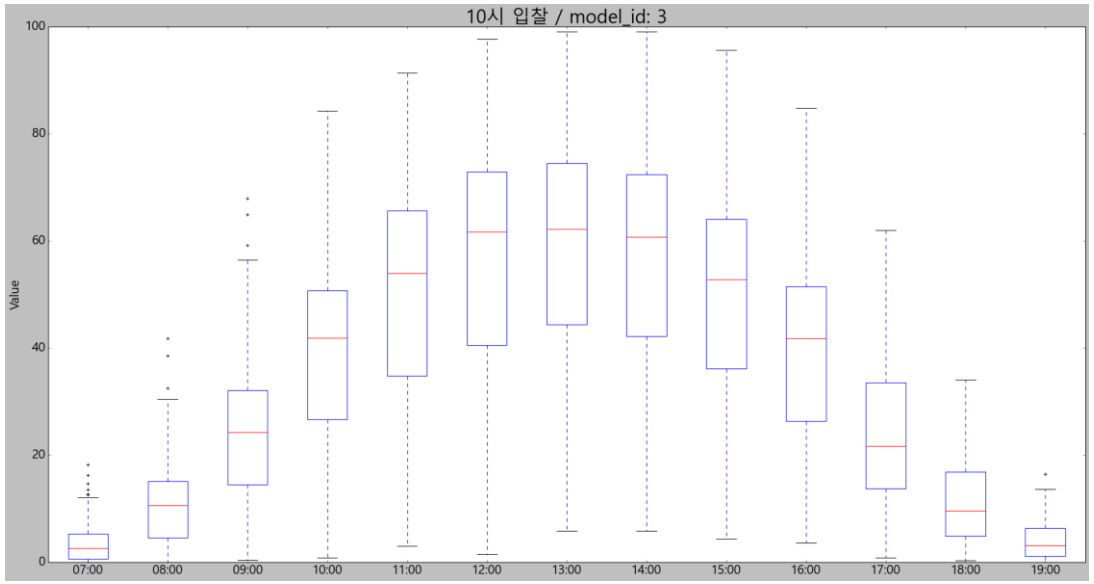
Model 2 :Bid at 17 p.m. (round2)



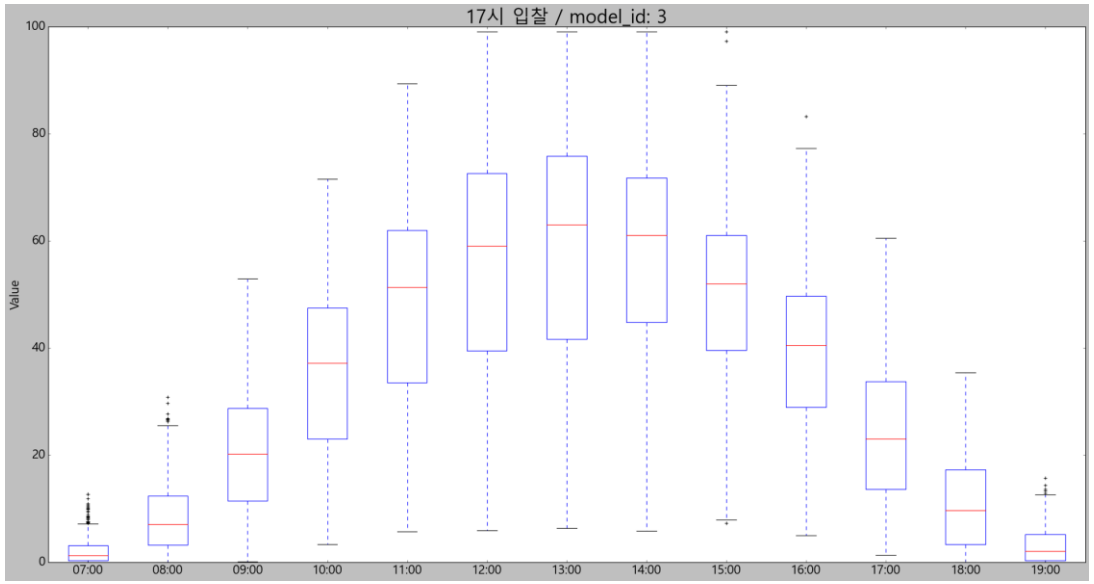
EDA

Predicted generation amount – box plot

Model 3 : Bid at 10 a.m. (round1)



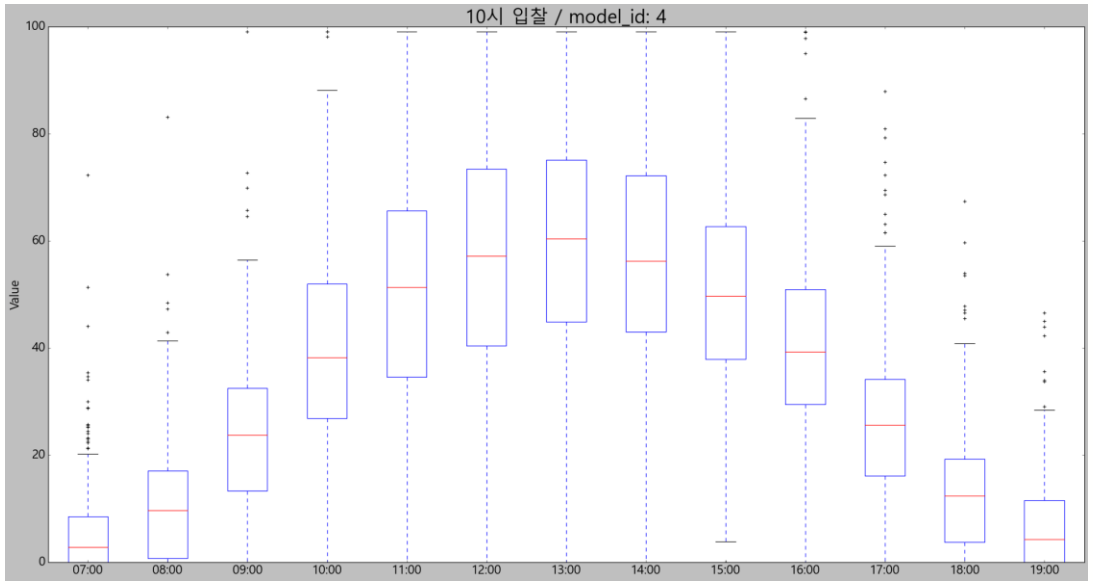
Model 3 :Bid at 17 p.m. (round2)



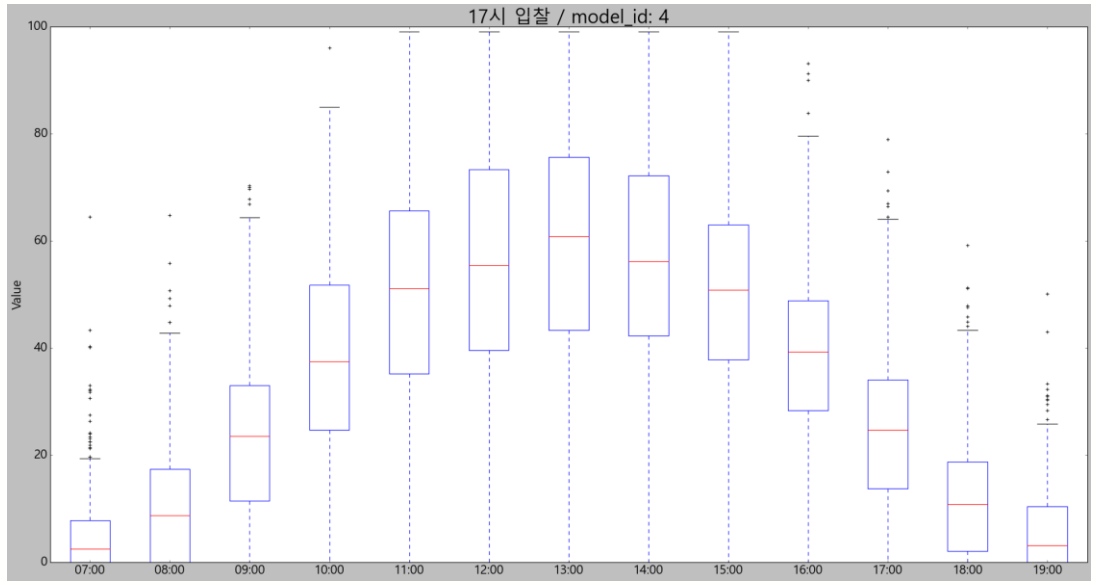
EDA

Predicted generation amount – box plot

Model 4 : Bid at 10 a.m. (round1)

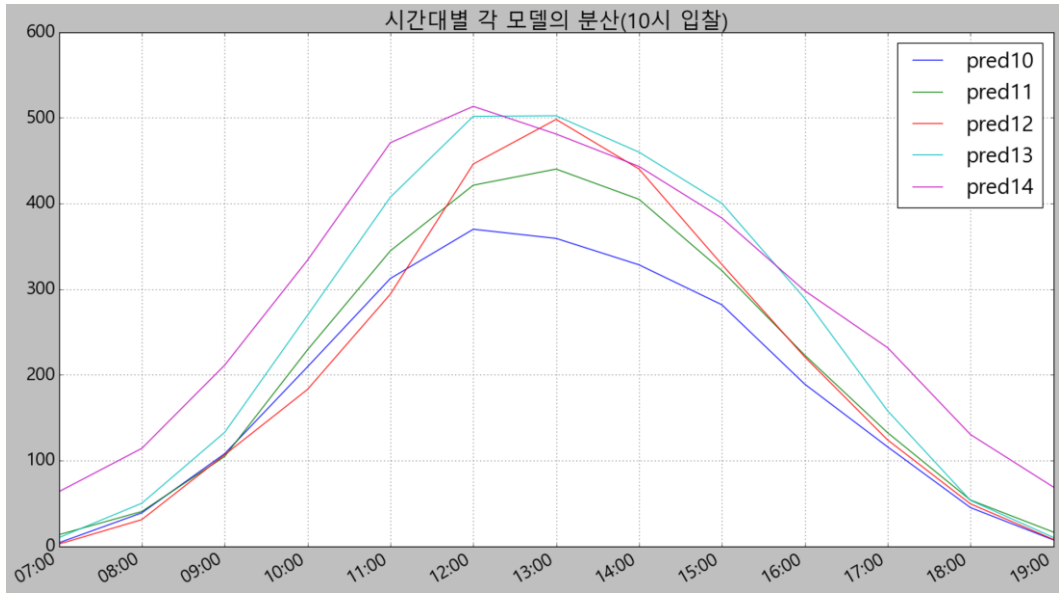


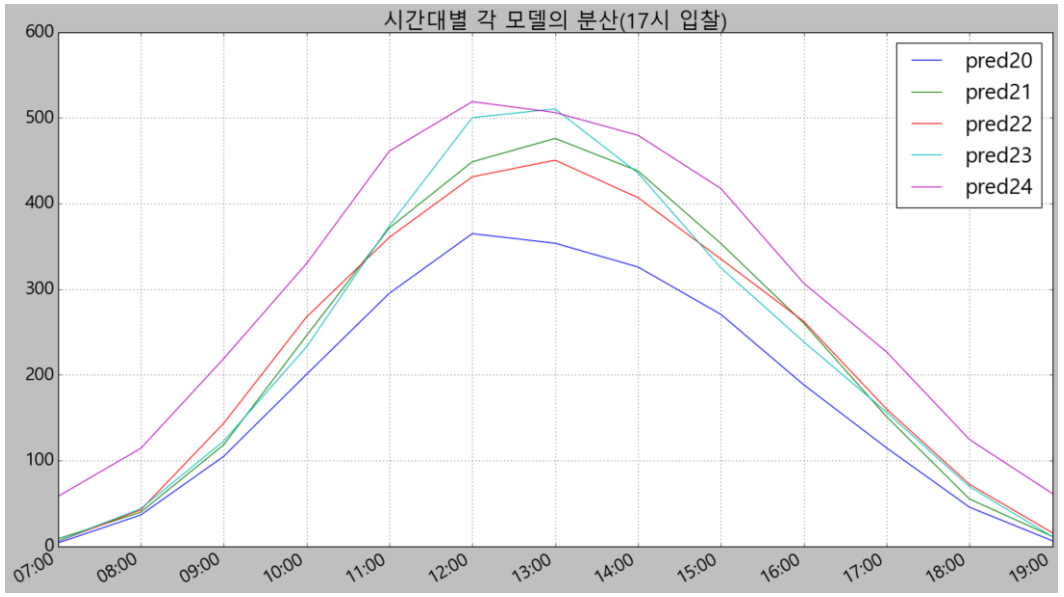
Model 4 :Bid at 17 p.m. (round2)



EDA

Predicted generation amount – variation



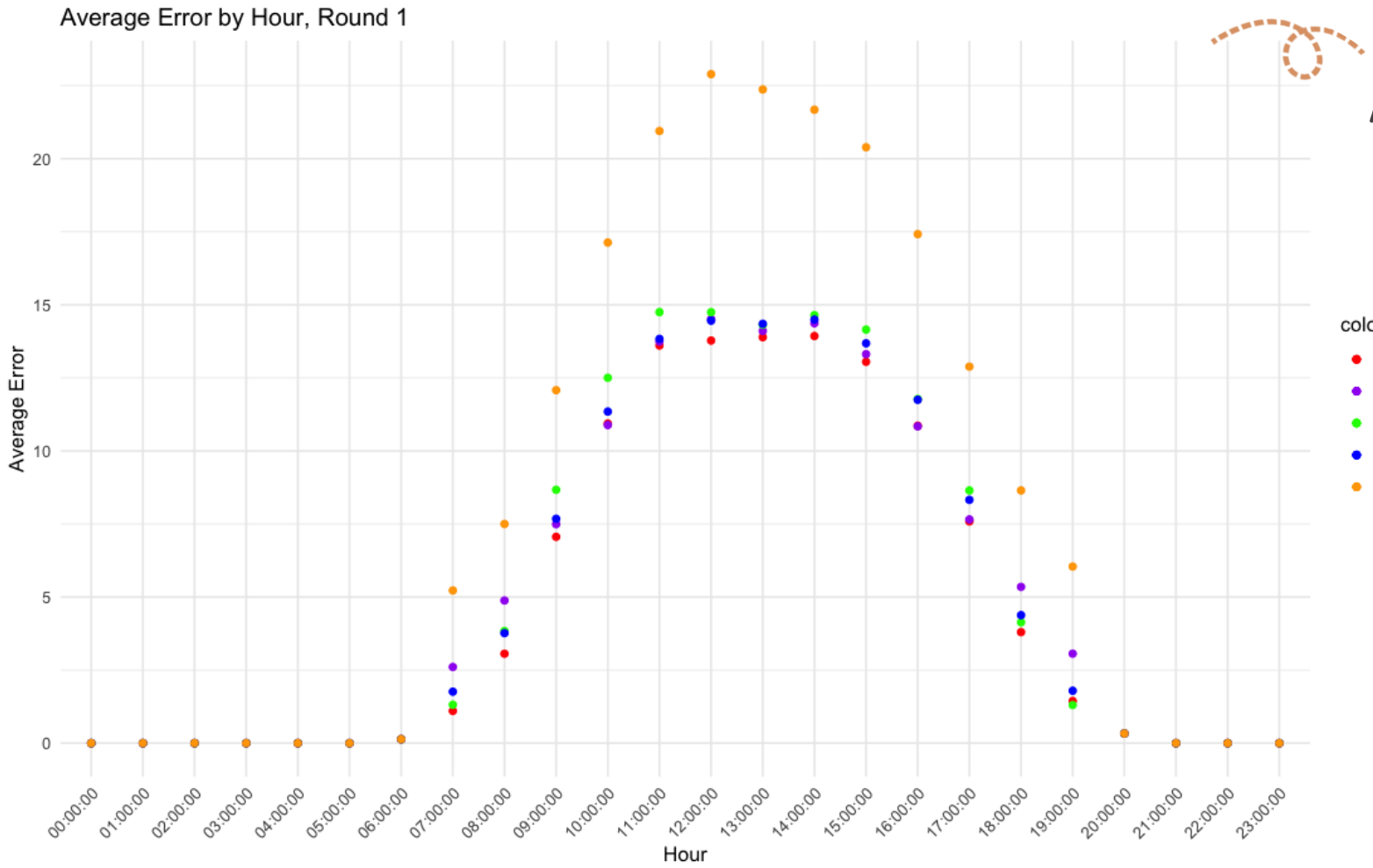


Although there are differences by time period,

the variance of the models is largest in the order of 4, 3, 2, 1, 0

EDA

Error Rate - actual generation vs forecasted generation

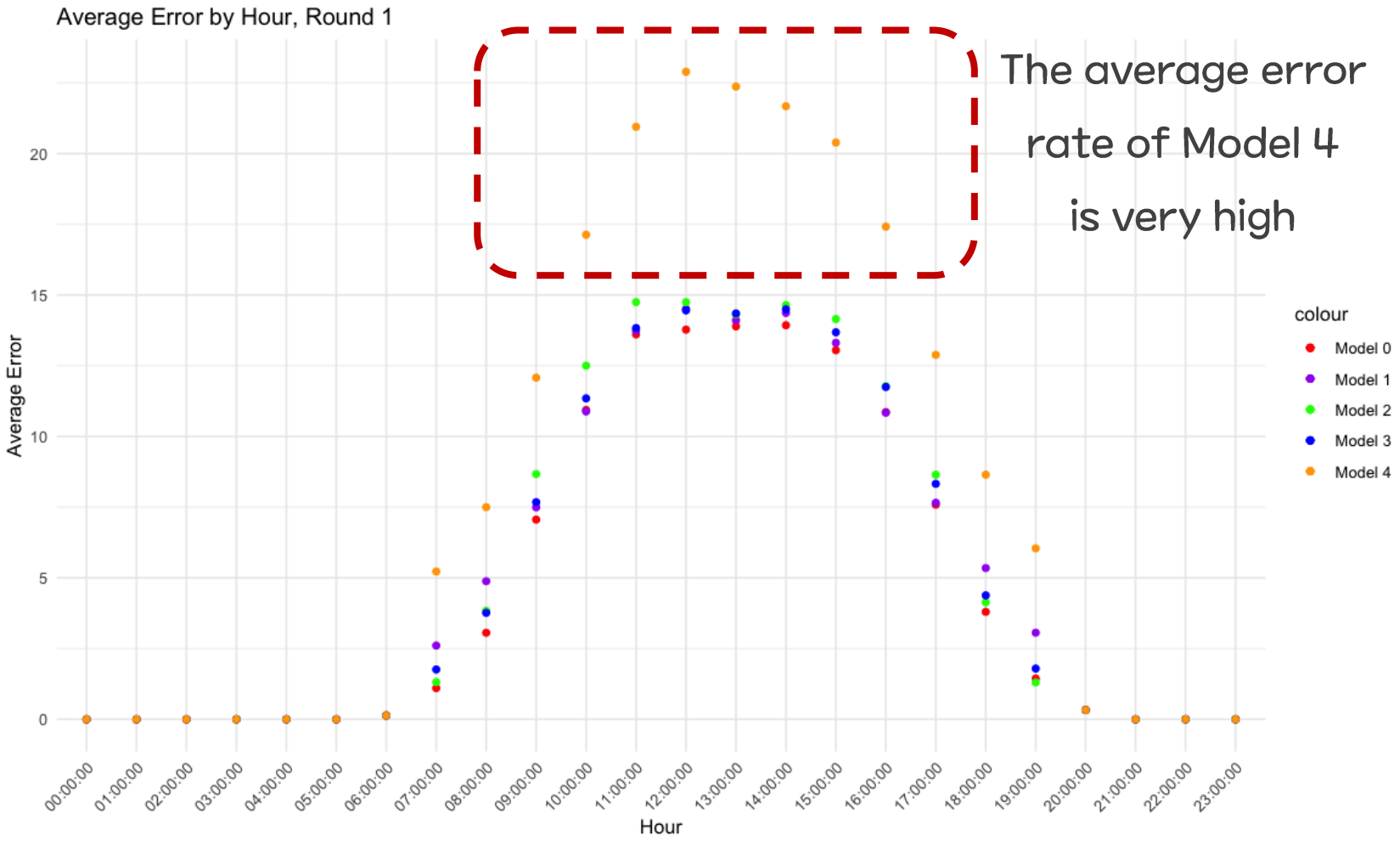


Round 1

Average error rate by model for each time period

EDA

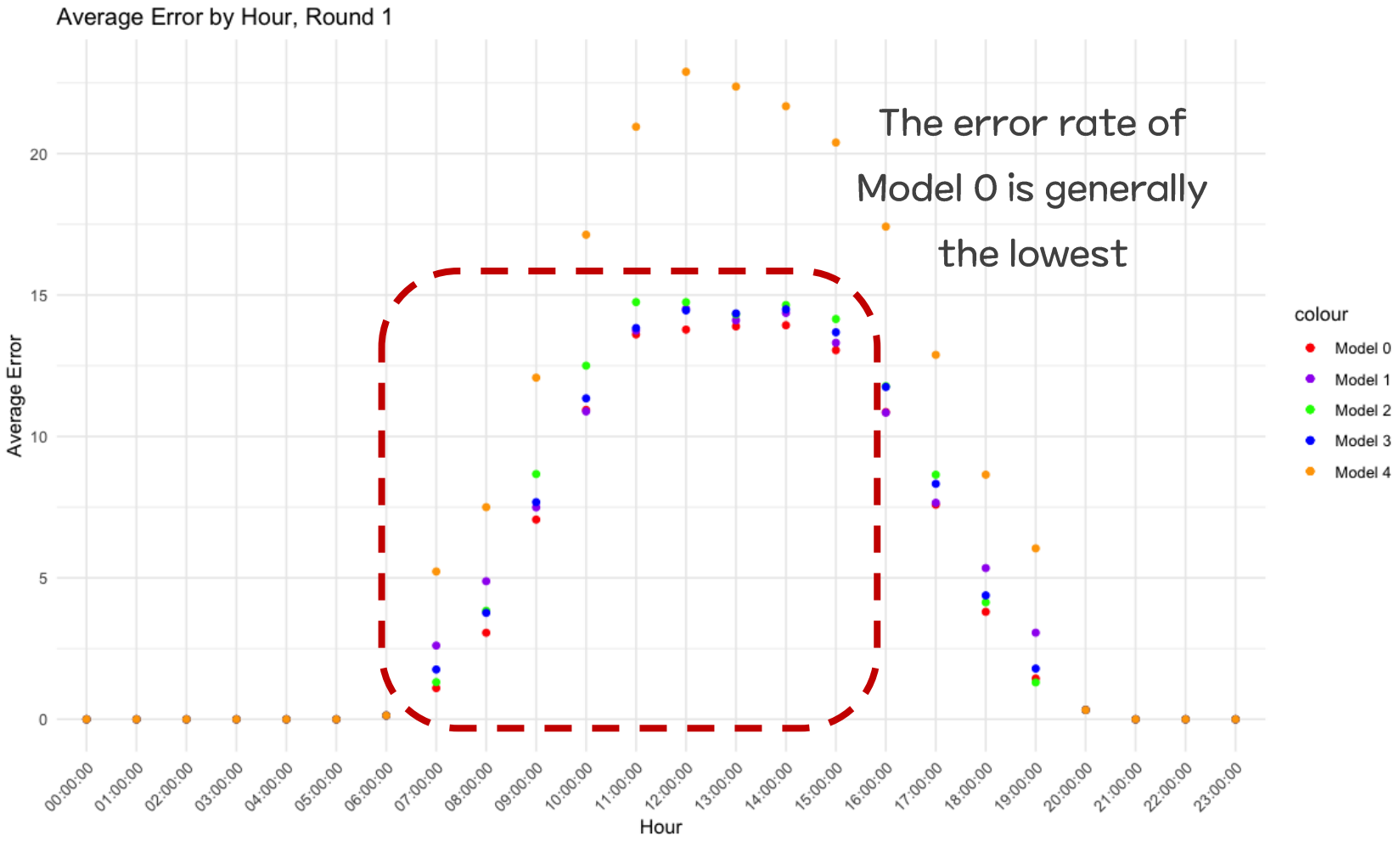
Error Rate - actual generation vs forecasted generation



The average error rate of Model 4 is very high

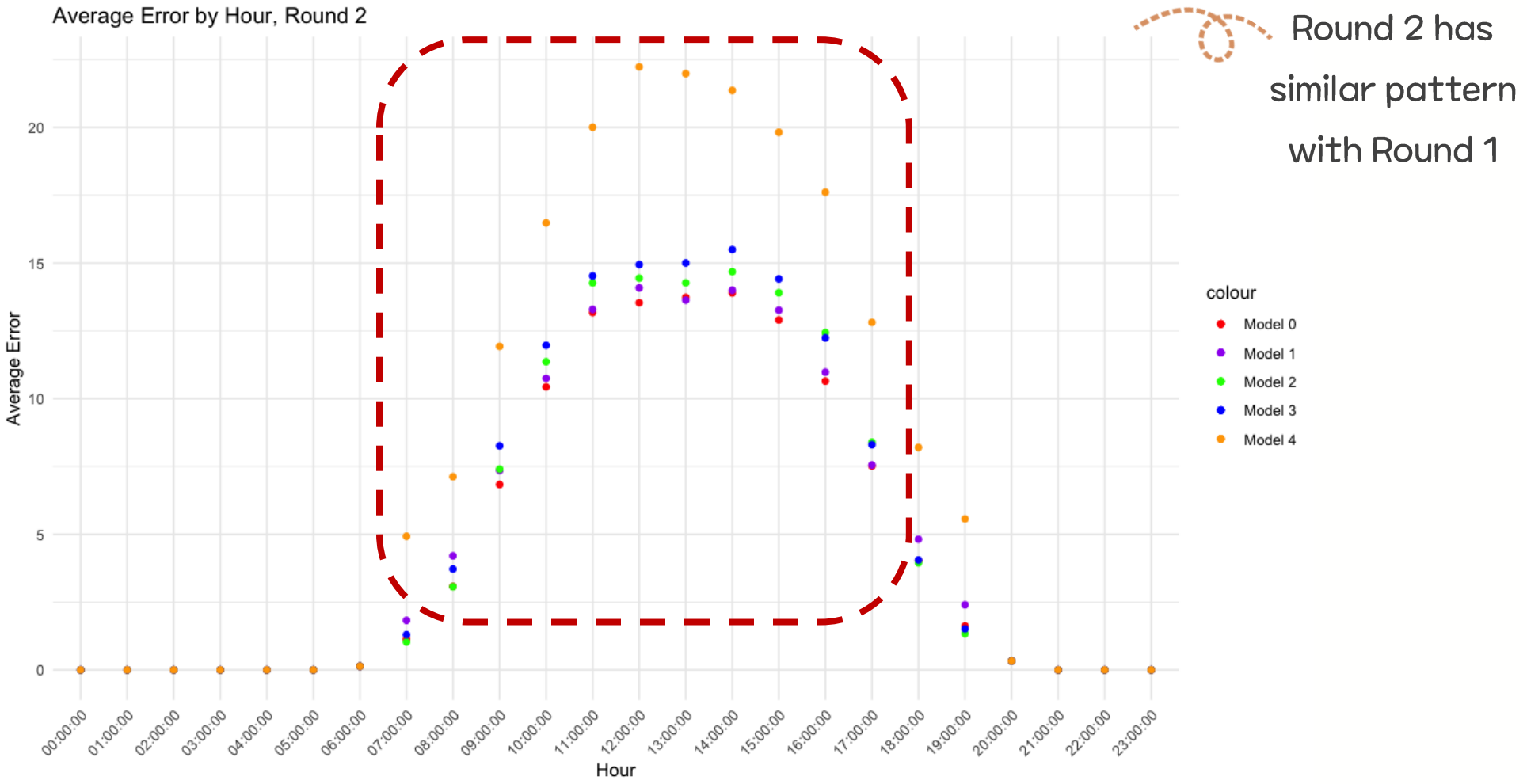
EDA

Error Rate - actual generation vs forecasted generation



EDA

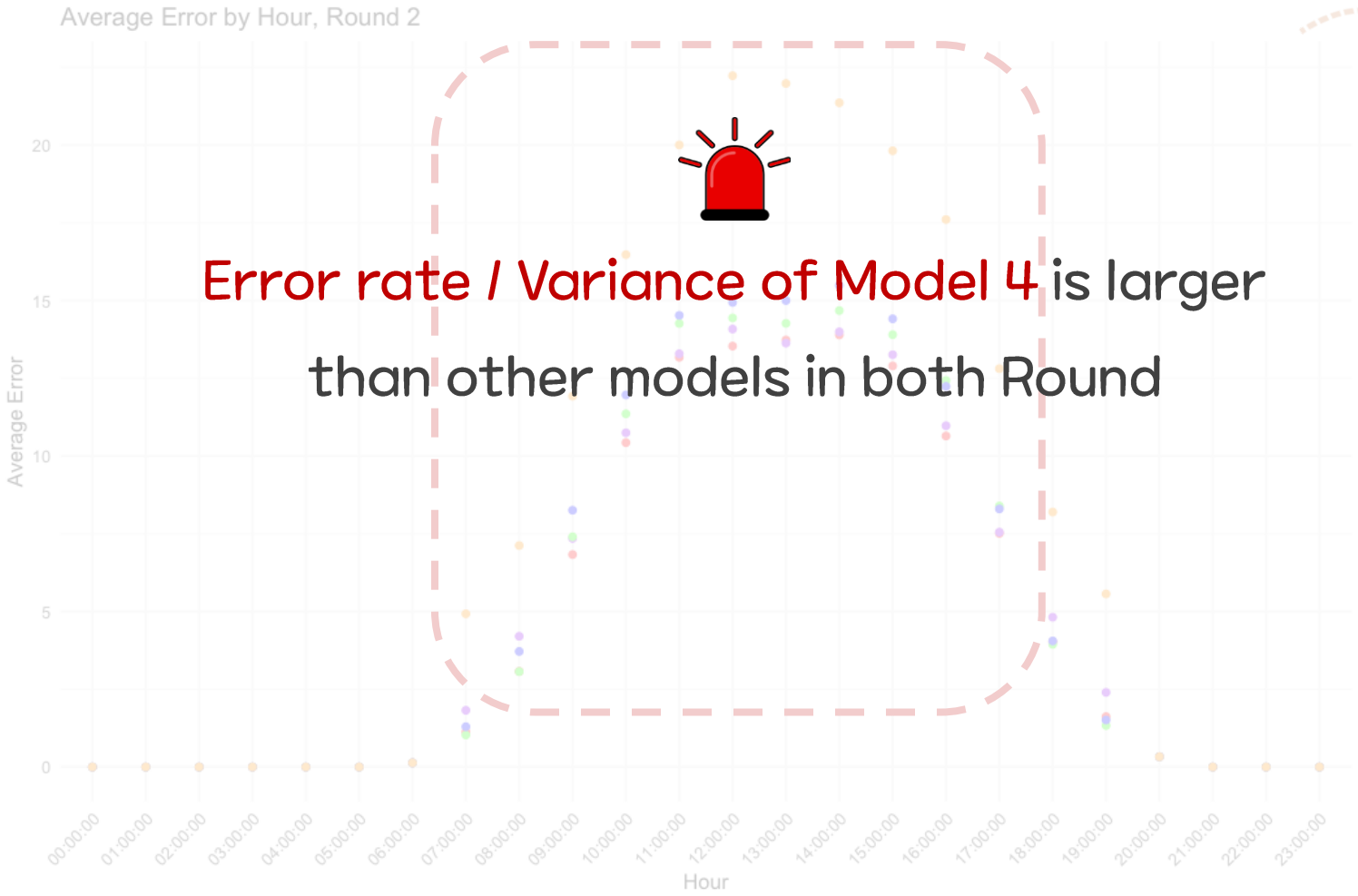
Error Rate - actual generation vs forecasted generation



Round 2 has similar pattern with Round 1

EDA

Error Rate - actual generation vs forecasted generation

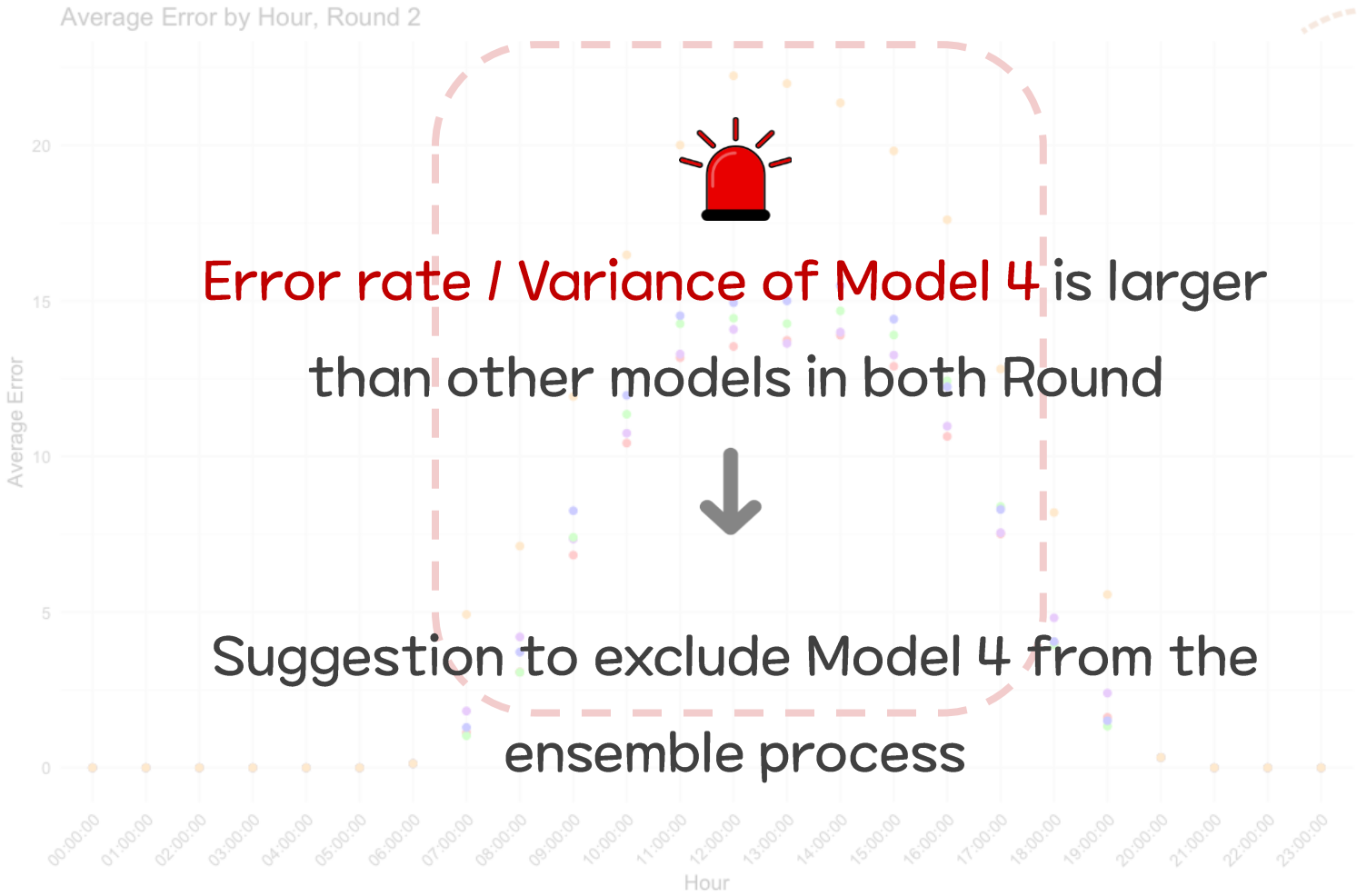


Round 2 has similar pattern with Round 1

Error rate / Variance of Model 4 is larger than other models in both Round

EDA

Error Rate - actual generation vs forecasted generation



Round 2 has similar pattern with Round 1

Error rate / Variance of Model 4 is larger than other models in both Round

↓

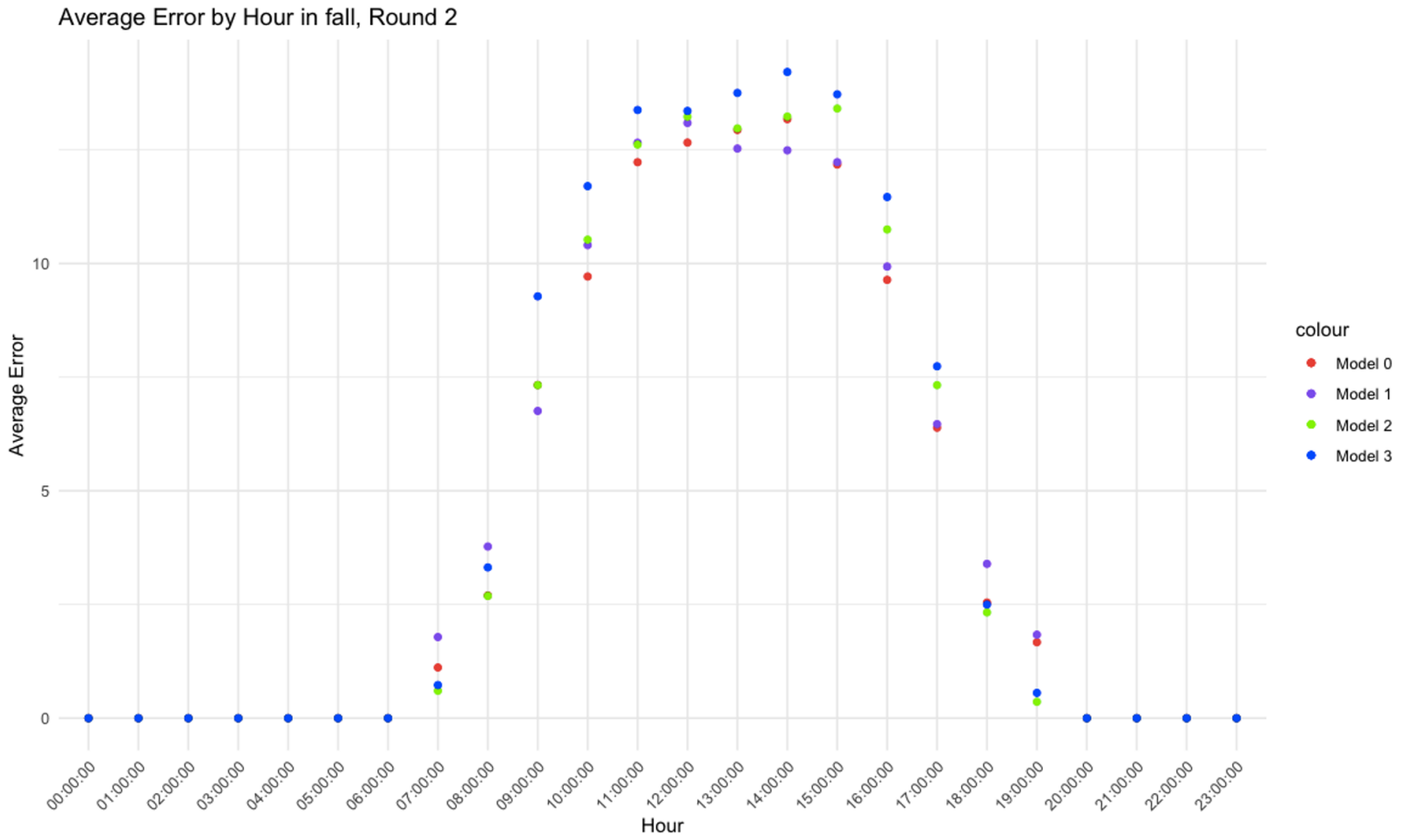
Suggestion to exclude Model 4 from the ensemble process

EDA

Error Rate - actual generation vs forecasted generation

Exclude Model 4

Consider only Fall data

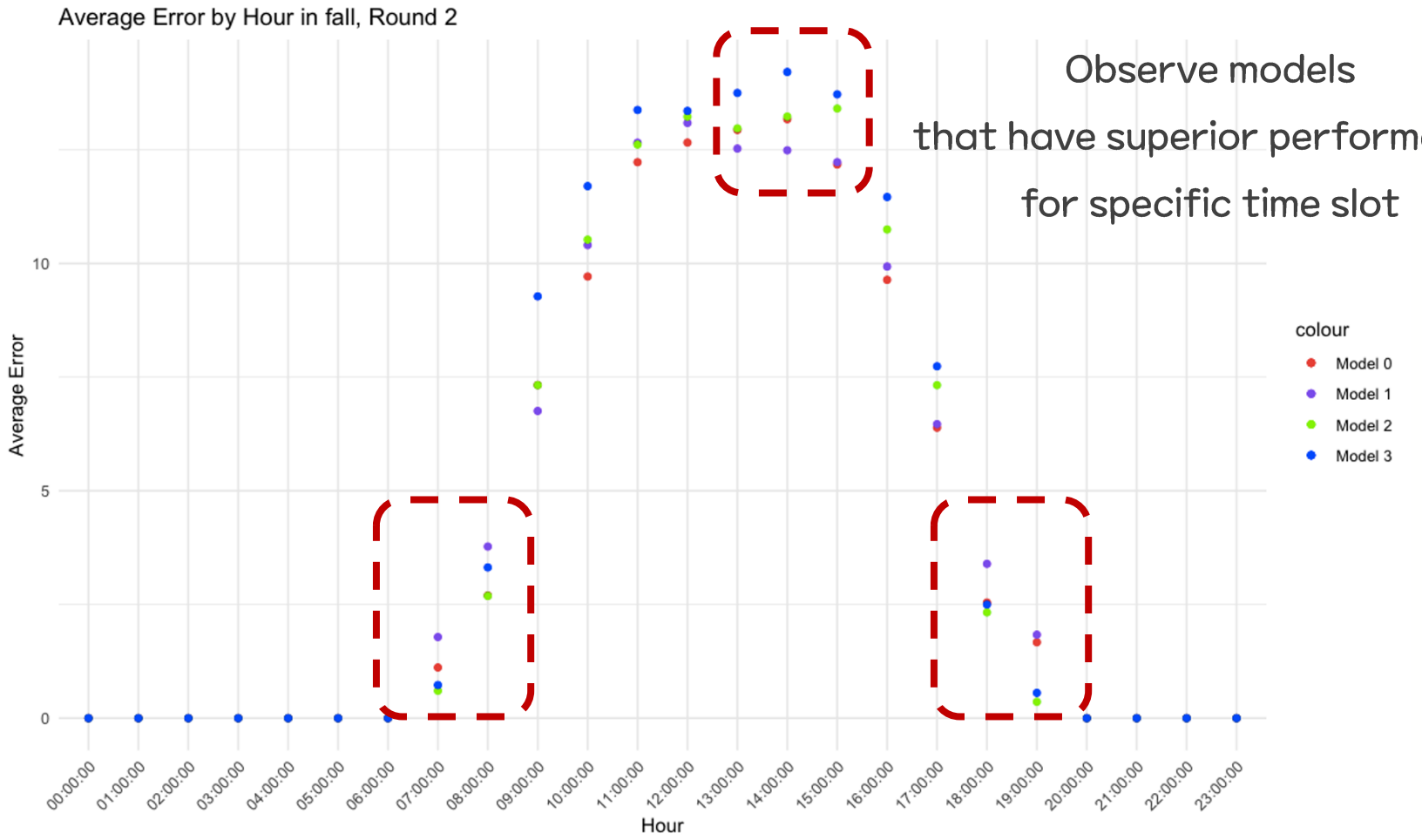


EDA

Error Rate - actual generation vs forecasted generation

Exclude Model 4

Consider only Fall data



Observe models that have superior performance for specific time slot

EDA

Error Rate - actual generation vs forecasted generation

Table of Average Error Rates for Each Model by Time Slot

hour	model0	model1	model2	model3	model4	hour	model0	model1	model2	model3	model4
7:00:00	0.8955	2.7081	0.6543	1.4359	4.5149	7:00:00	1.1146	1.7845	0.6019	0.7280	3.9420
8:00:00	2.6194	4.9056	4.4764	3.2249	7.1268	8:00:00	2.6986	3.7748	2.6881	3.3164	6.0359
9:00:00	7.7035	7.3473	11.0274	8.0444	12.7201	9:00:00	7.3279	6.7570	7.3247	9.2782	12.0800
10:00:00	10.2554	10.6475	13.5739	10.0637	16.9716	10:00:00	9.7143	10.4046	10.5251	11.7016	14.9649
11:00:00	12.8063	13.2360	14.2907	12.5647	19.7866	11:00:00	12.2291	12.6575	12.6150	13.3770	16.5532
12:00:00	13.0861	13.8327	14.3313	13.7187	23.5350	12:00:00	12.6601	13.0913	13.2291	13.3566	19.5062
13:00:00	12.5749	12.7261	13.3266	13.1153	21.6607	13:00:00	12.9375	12.5288	12.9723	13.7526	19.1405
14:00:00	13.0941	13.3052	13.8213	14.0544	20.8093	14:00:00	13.1746	12.4898	13.2326	14.2134	17.8192
15:00:00	12.5095	12.7522	13.3478	13.5479	19.0190	15:00:00	12.1772	12.2295	13.4093	13.7214	16.7193
16:00:00	9.6196	9.9271	10.4812	11.1331	15.8285	16:00:00	9.6404	9.9327	10.7492	11.4625	15.2134
17:00:00	6.3971	6.4595	7.8366	7.6823	10.8050	17:00:00	6.3885	6.4670	7.3249	7.7396	10.6010
18:00:00	2.4153	3.7629	2.9694	2.7476	6.6126	18:00:00	2.5431	3.3942	2.3264	2.4990	5.4799
19:00:00	1.1412	1.9345	0.4829	1.1759	4.7196	19:00:00	1.6700	1.8359	0.3601	0.5561	3.7383
20:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	20:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
21:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	21:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
22:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	22:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
23:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	23:00:00	0.0000	0.0000	0.0000	0.0000	0.0000

EDA

Error Rate - actual generation vs forecasted generation

Models which have lower average error rate than Model 0

Table of Average Error Rates for Each Model by Time Slot

hour	model0	model1	model2	model3	model4	hour	model0	model1	model2	model3	model4
7:00:00	0.8955	2.7081	0.6543	1.4359	4.5149	7:00:00	1.1146	1.7845	0.6019	0.7280	3.9420
8:00:00	2.6194	4.9056	4.4764	3.2249	7.1268	8:00:00	2.6986	3.7748	2.6881	3.3164	6.0359
9:00:00	7.7035	7.3473	11.0274	8.0444	12.7201	9:00:00	7.3279	6.7570	7.3247	9.2782	12.0800
10:00:00	10.2554	10.6475	13.5739	10.0637	16.9716	10:00:00	9.7143	10.4046	10.5251	11.7016	14.9649
11:00:00	12.8063	13.2360	14.2907	12.5647	19.7866	11:00:00	12.2291	12.6575	12.6150	13.3770	16.5532
12:00:00	13.0861	13.8327	14.3313	13.7187	23.5350	12:00:00	12.6601	13.0913	13.2291	13.3566	19.5062
13:00:00	12.5749	12.7261	13.3266	13.1153	21.6607	13:00:00	12.9375	12.5288	12.9723	13.7526	19.1405
14:00:00	13.0941	13.3052	13.8213	14.0544	20.8093	14:00:00	13.1746	12.4898	13.2326	14.2134	17.8192
15:00:00	12.5095	12.7522	13.3478	13.5479	19.0190	15:00:00	12.1772	12.2295	13.4093	13.7214	16.7193
16:00:00	9.6196	9.9271	10.4812	11.1331	15.8285	16:00:00	9.6404	9.9327	10.7492	11.4625	15.2134
17:00:00	6.3971	6.4595	7.8366	7.6823	10.8050	17:00:00	6.3885	6.4670	7.3249	7.7396	10.6010
18:00:00	2.4153	3.7629	2.9694	2.7476	6.6126	18:00:00	2.5431	3.3942	2.3264	2.4990	5.4799
19:00:00	1.1412	1.9345	0.4829	1.1759	4.7196	19:00:00	1.6700	1.8359	0.3601	0.5561	3.7383
20:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	20:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
21:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	21:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
22:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	22:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
23:00:00	0.0000	0.0000	0.0000	0.0000	0.0000	23:00:00	0.0000	0.0000	0.0000	0.0000	0.0000

EDA

Error Rate - actual generation vs forecasted generation

Overlaying Round1 , Round2

hour	model0	model1	model2	model3	model4
7:00:00	0.8955	2.7081	0.6543	1.4359	4.5149
8:00:00	2.6194	4.9056	4.4764	3.2249	7.1268
9:00:00	7.7035	7.3473	11.0274	8.0444	12.7201
10:00:00	10.2554	10.6475	13.5739	10.0637	16.9716
11:00:00	12.8063	13.2360	14.2907	12.5647	19.7866
12:00:00	13.0861	13.8327	14.3313	13.7187	23.5350
13:00:00	12.5749	12.7261	13.3266	13.1153	21.6607
14:00:00	13.0941	13.3052	13.8213	14.0544	20.8093
15:00:00	12.5095	12.7522	13.3478	13.5479	19.0190
16:00:00	9.6196	9.9271	10.4812	11.1831	15.8285
17:00:00	6.3971	6.4595	7.8366	7.6828	10.8050
18:00:00	2.4153	3.7629	2.9694	2.7476	6.6126
19:00:00	1.1412	1.9345	0.4829	0.1759	4.7196
20:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
21:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
22:00:00	0.0000	0.0000	0.0000	0.0000	0.0000
23:00:00	0.0000	0.0000	0.0000	0.0000	0.0000

Morning : 07 ~ 10

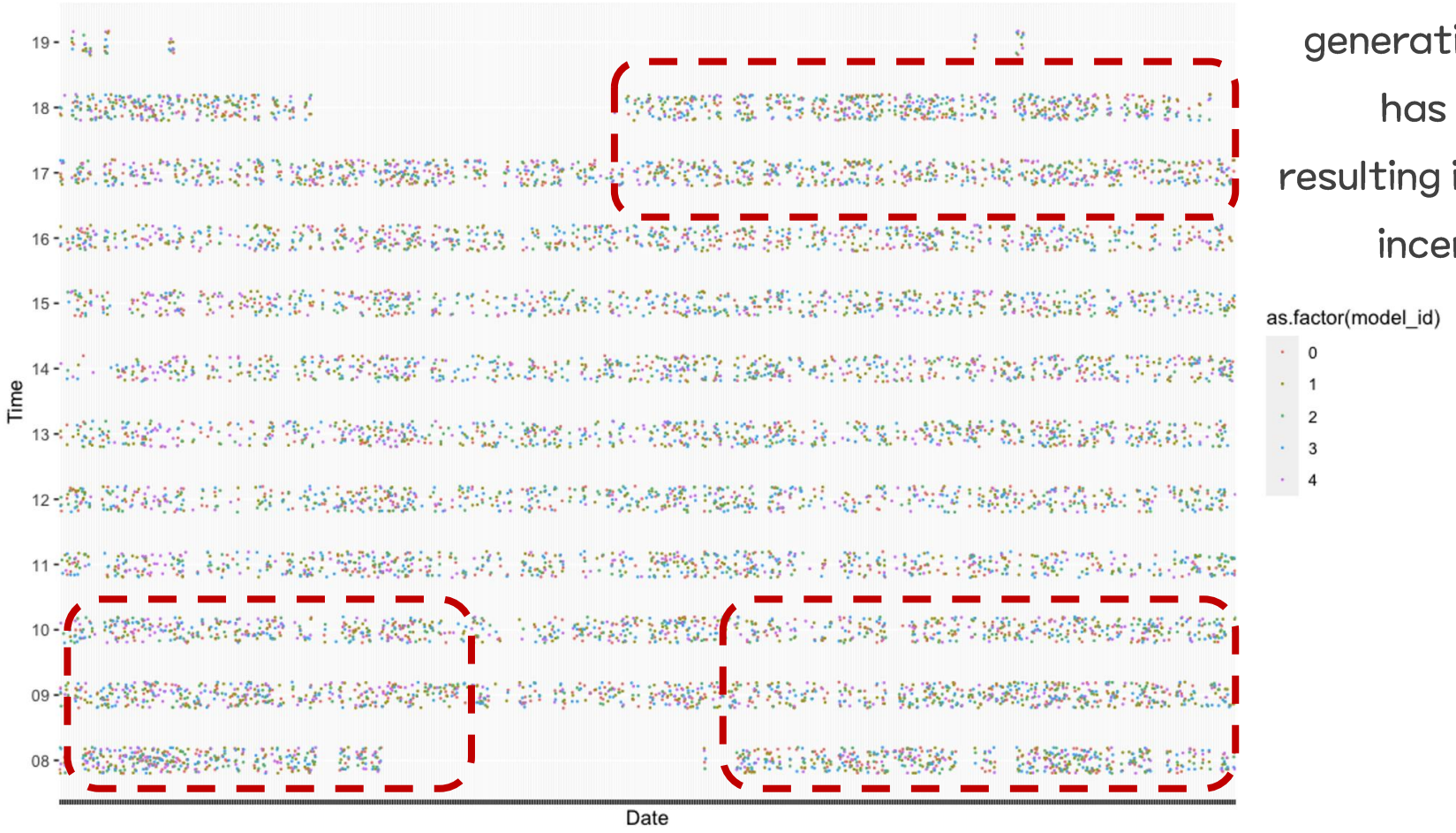
Afternoon : 11 ~ 14

Evening : 15 ~ 19

Since the model performance is differ by time slot, Divide the time slots And train model separately

EDA

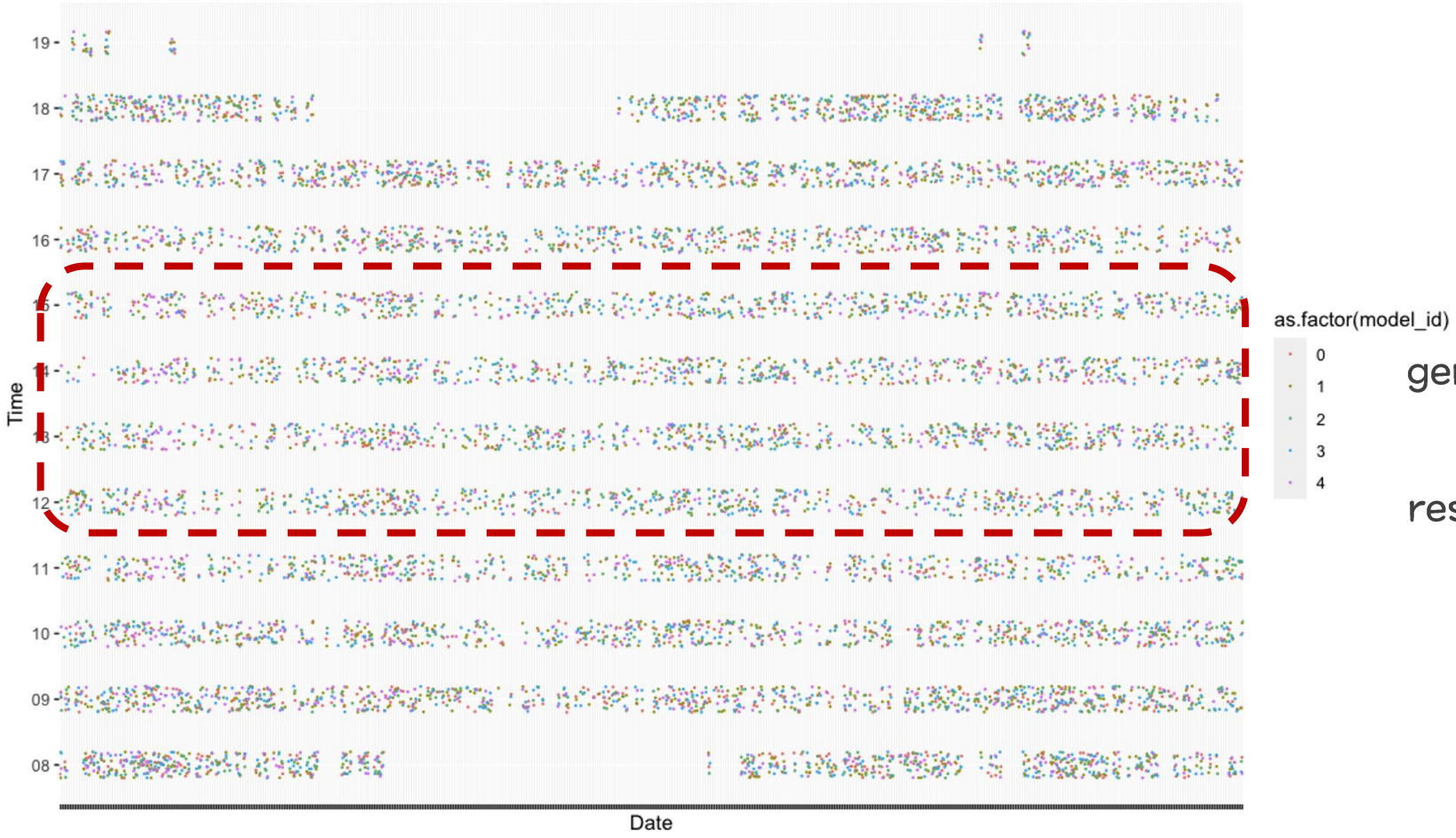
Incentive



Morning, evening slot where the absolute value of the generation amount is small, has low error rate, resulting in high proportion of incentives received

EDA

Incentive



Afternoon slot where absolute value of the generation amount is big, has the large error rate, resulting in low proportion of incentive received.

EDA

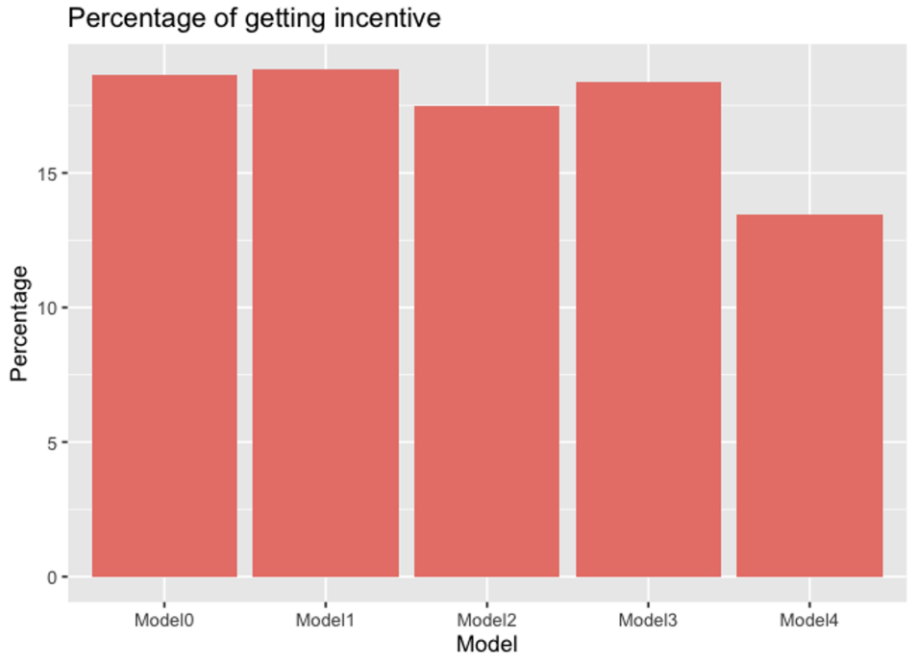
Incentive

For every model, when the incentive is 0, it can be inferred that this is because the facility utilization rate is zero

EDA

Incentive

Model <chr>	Incentive <dbl>
Model0	18.64217
Model1	18.84921
Model2	17.50345
Model3	18.38337
Model4	13.43168



In the same vein as error rate

Model 4 has a relatively low rate of incentives received

Decided to exclude Model 4 from the ensemble process

Modeling

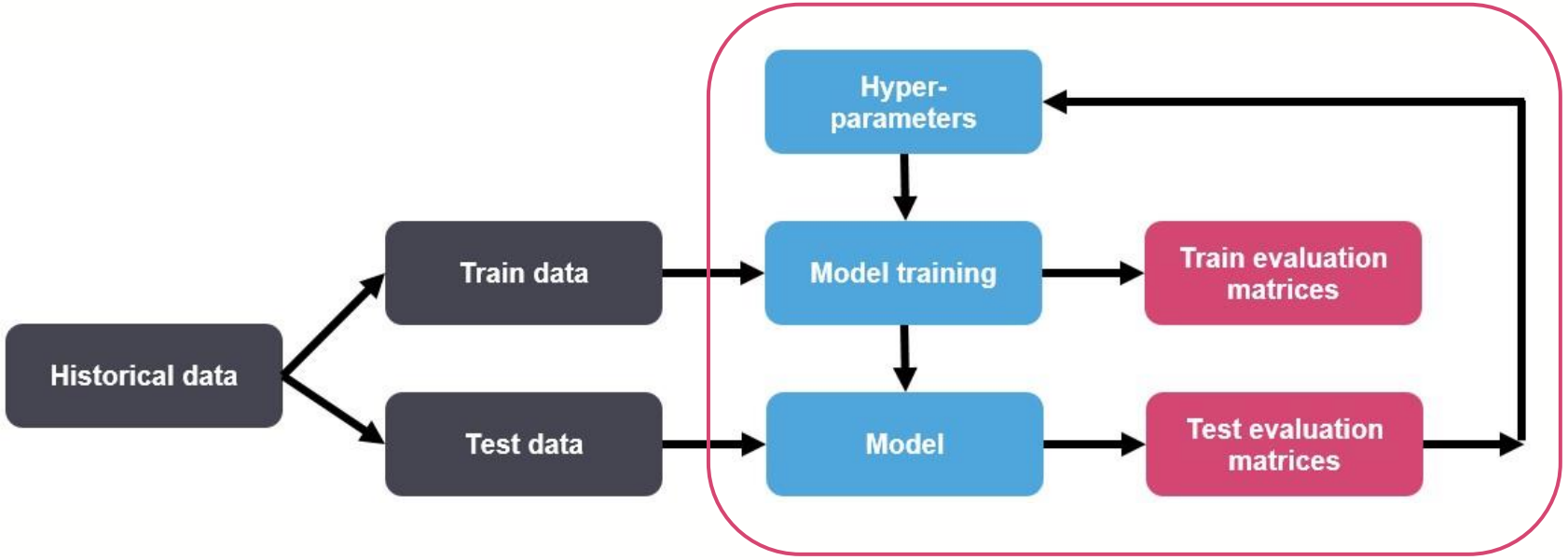
Modeling

Optimization of Hyper parameter

Hyper Parameter Optimization

The act of adjusting values in advance that will be reflected in the model training process, also known as hyperparameter tuning

Crucial for enhancing model performance



Modeling

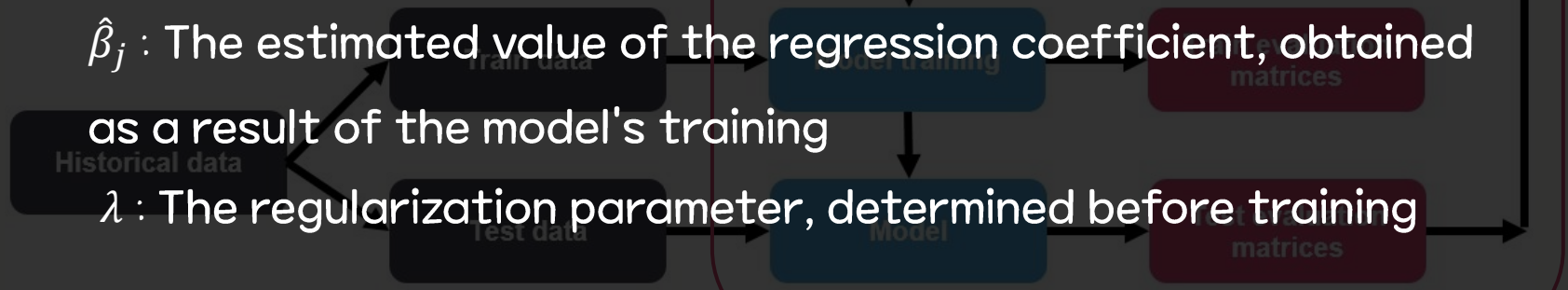
Hyperparameter vs Parameter

e.g., LASSO regression

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_1)^2 + \lambda \sum_{j=1}^{p}|\hat{\beta}_j|$$

$\hat{\beta}_j$: The estimated value of the regression coefficient, obtained as a result of the model's training

λ : The regularization parameter, determined before training



Modeling

Hyperparameter tuning methods

There are many types of hyperparameter optimization methods, but let's focus on the two techniques in this topic analysis

GRID SEARCH

- Define possible values for Hyper Parameter at regular intervals (GRID)
- Generate all possible combinations
- Train and validate the model for all combinations
  → Select the optimal hyperparameters.

RANDOM SEARCH

- Define the range that include each HP
- Select random combinations of values within the defined range
- Train and validate the model within the given time and resources→ Select the optimal hyperparameters

Modeling

Hyperparameter tuning methods

There are many types of hyperparameter optimization methods, but let's focus on the two techniques in this topic analysis

GRID SEARCH

- Define possible values for Hyper Parameter at regular intervals
- Generate all possible combinations
- Train and validate the model for all combinations

→ Select the optimal hyperparameters.

GRID

RANDOM SEARCH

For Deep Learning Models

→ Even if it takes a long time, set appropriate intervals to search for the optimal values

Modeling

Hyperparameter tuning methods

There are many types of hyperparameter optimization methods, but let's focus on the two techniques in this topic analysis

GRID SEARCH

- Define possible values for Hyper Parameter at regular intervals
- Iterate all possible combinations
- Train and validate the model for all combinations
- → Select the optimal hyperparameters.

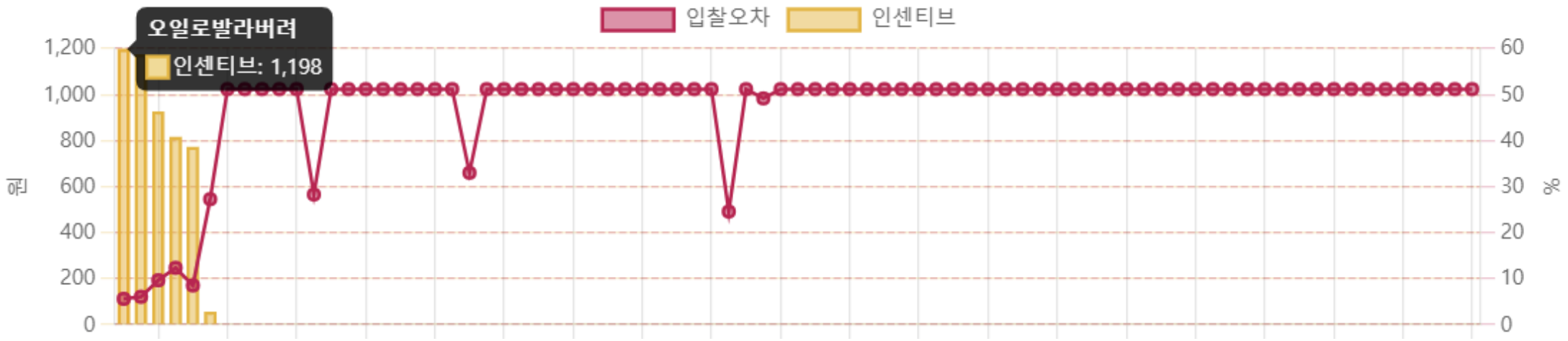
OPTUNA

RANDOM SEARCH

- Define the range that include each HP
- Select random combinations of values within the defined range
- Train and validate the model within the given time and resources→ Select the optimal hyperparameters

Modeling

Baseline score setting

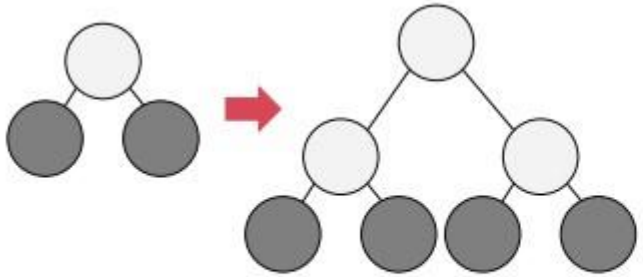


1st	-	–	오일로발라버려	6,552원	15.66	김태양대학교

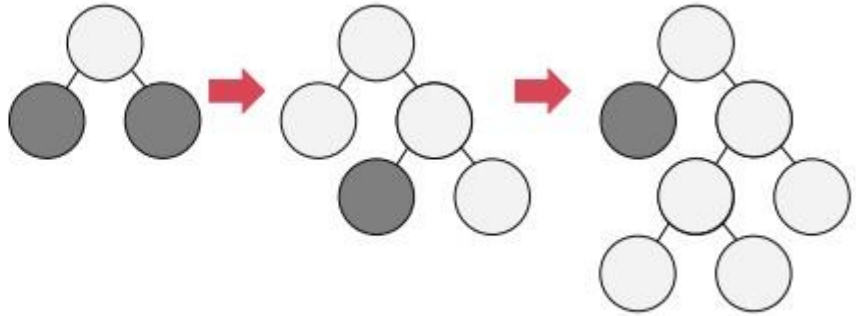
Set the cumulative incentives score of the host (오일로발라버려) as the baseline

Modeling

LGBM



Level-wise growth



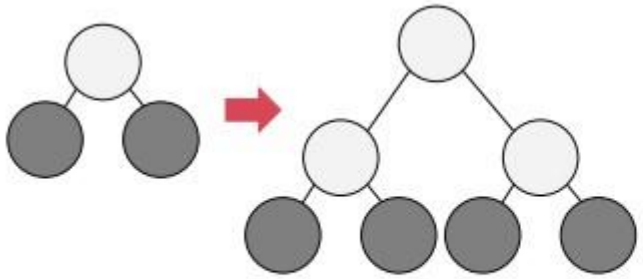
Leaf-wise growth

LGBM(LightGBM)

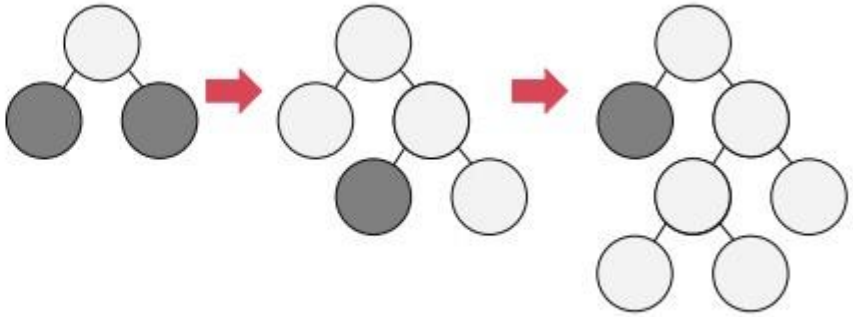
As a type of boosting, training progresses in a direction that continuously improves the initially trained model

Modeling

LGBM



Level-wise growth



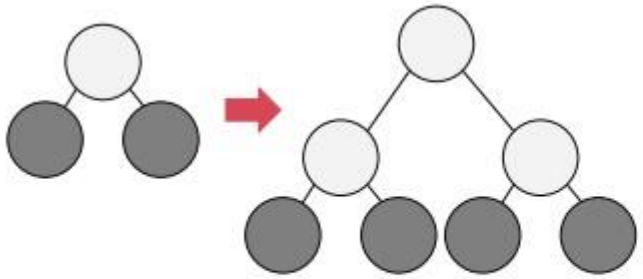
Leaf-wise growth

Unlike original Tree-base algorithms,

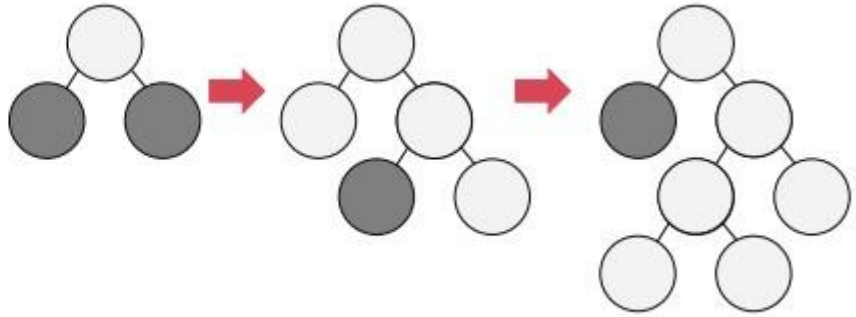
It use leaf-wise split methods.

Modeling

LGBM



Level-wise growth



Leaf-wise growth

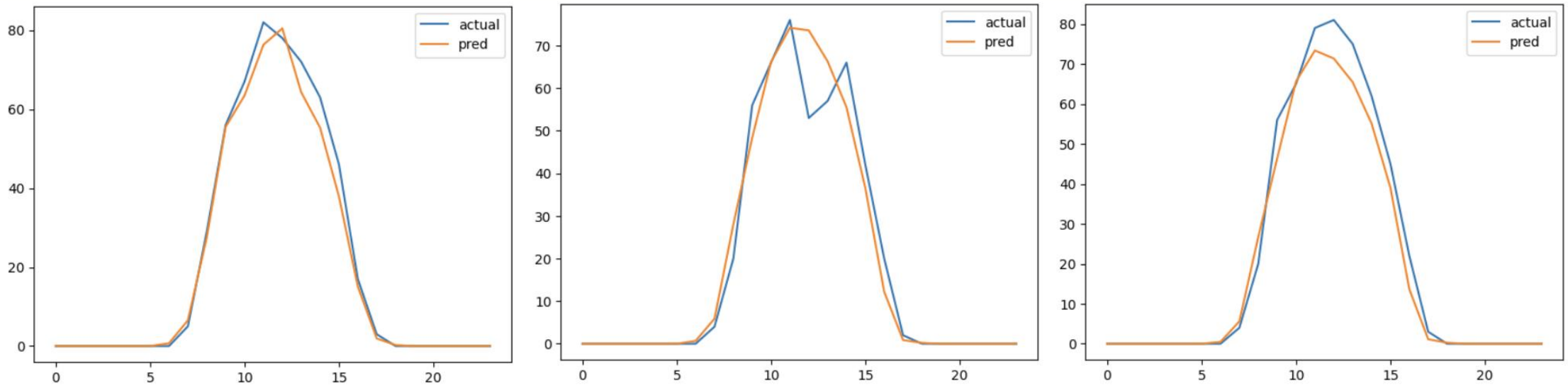
A method that focus on width whether than depth, to reduce the loss

It is important to train the model with a large amount of data,

Since it is Sensitive to overfitting

Modeling

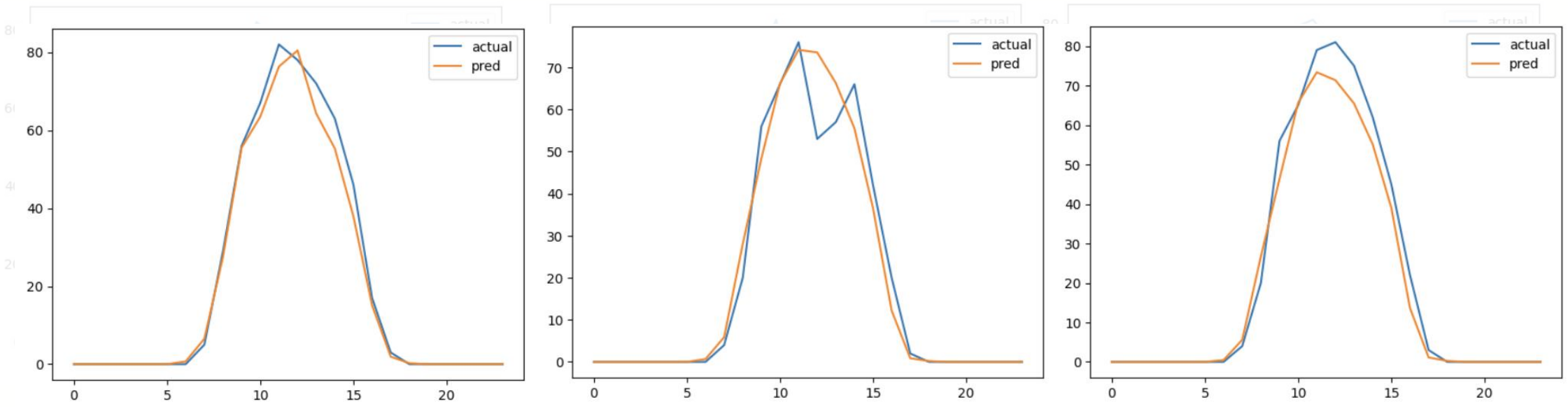
LGBM



model	data	Total incentive (10.25~10.31)	notes
LGBM	predicted generation amount from m0~m4	7088	Divided by time slot

Modeling

LGBM



Bad predictions for Spike points

Generally predict well, but sometimes

fails to predict the midday part

Modeling

LGBM

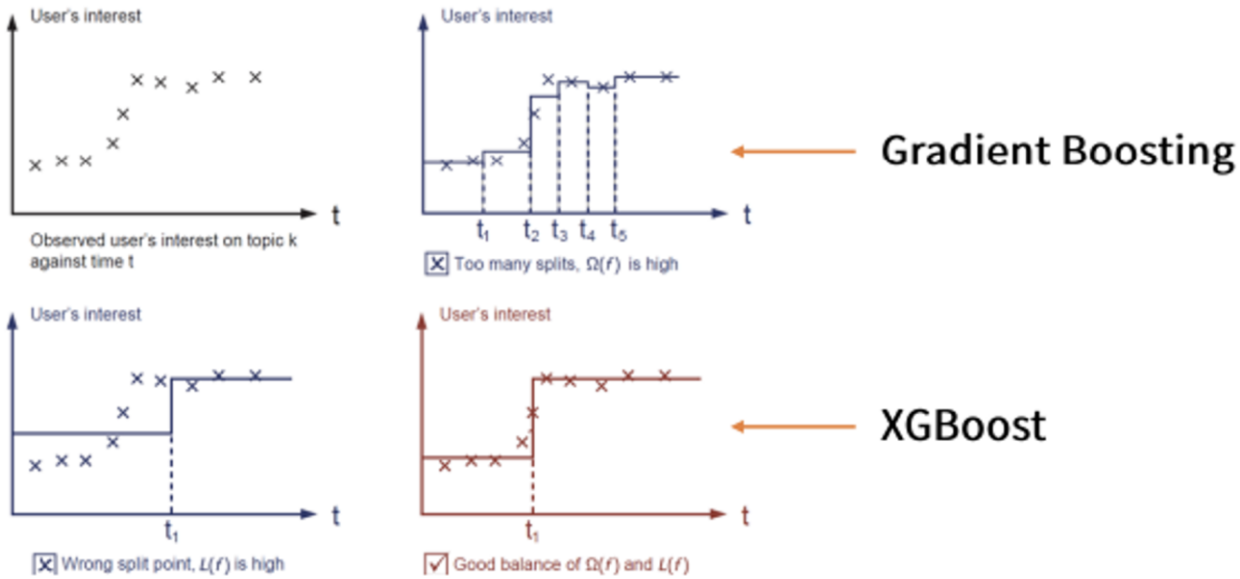
model	data	Total incentive (10.25~10.31)	notes
LGBM	predicted generation amount from m0~m4	7256	Divided by time slot Season and time derived variables are used

Try the same model structure with additional derived variables

No significant difference in performance

Modeling

XGBoost



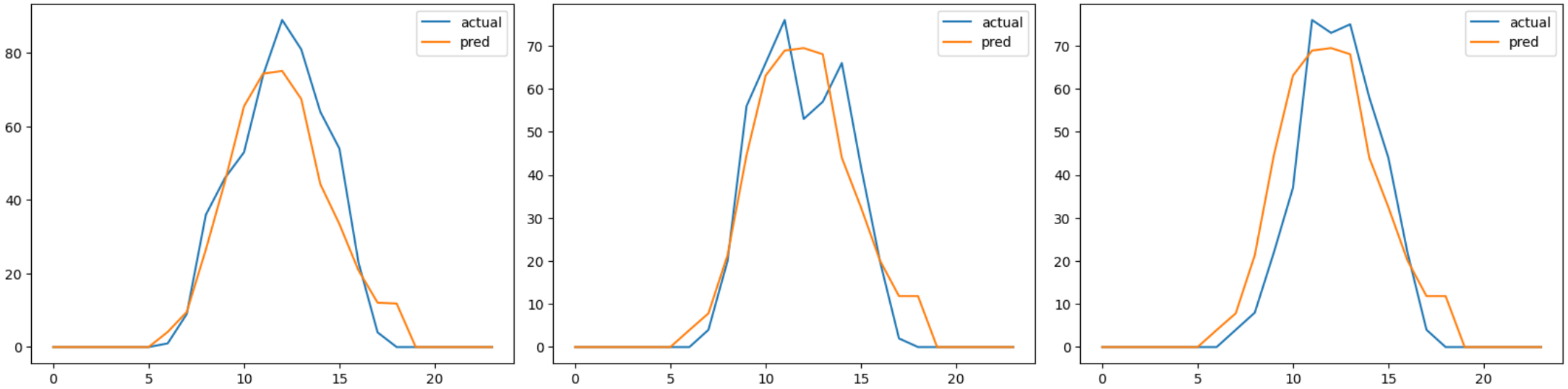
XGB(XGBoost)

imposes higher penalties on the loss as tree complexity increases

Algorithm that add regularization term to handle the overfitting issue arising from the method of reducing residuals in GBM

Modeling

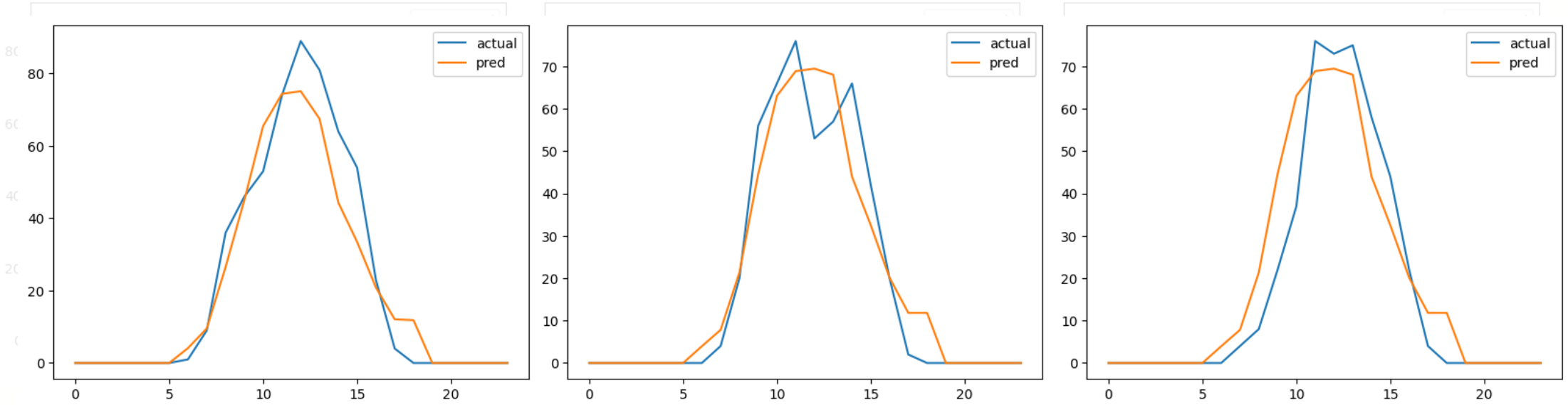
XGBoost



model	data	Total incentive (10.25~10.31)	notes
XGB	predicted generation amount from m0~m4 Weather forecast	4479	Divided by time slot

Modeling

XGBoost



model	data	Total incentive (10.25~10.31)	notes
XGB	predicted generation amount from m0~m4 weather forecast	4479	Divided by time slot

Bad predictions for Spike points

Overall, the performance is not good

Modeling

Linear Regression

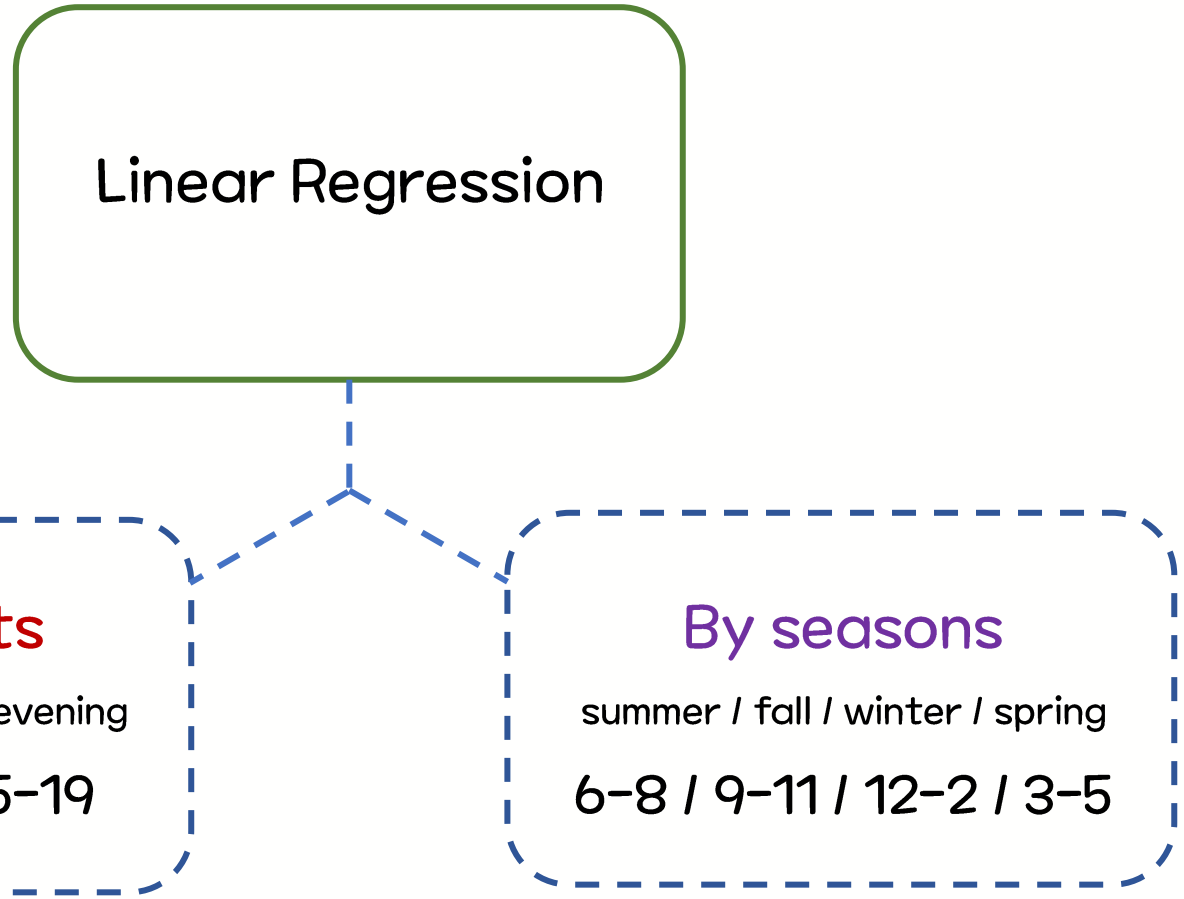
Linear Regression

Analytical technique that models the linear relationship between dependent variable Y and one or more independent variables X

Since there is a strong correlation between the predicted generation amounts provided by the host and the actual generation amounts, using the model is considered

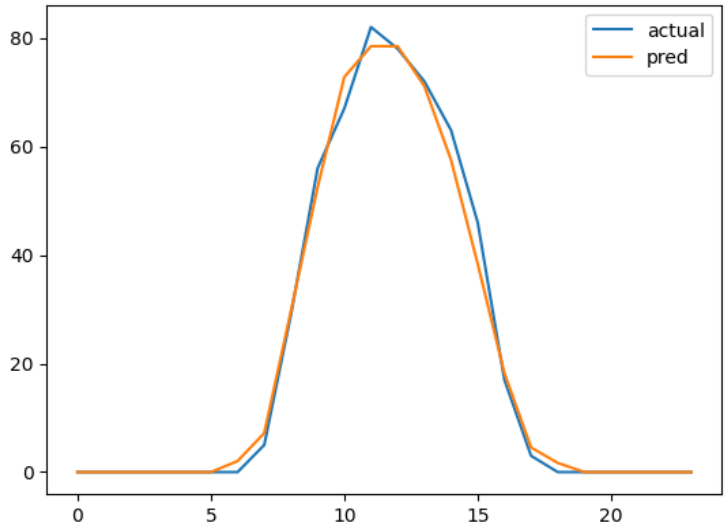
Modeling

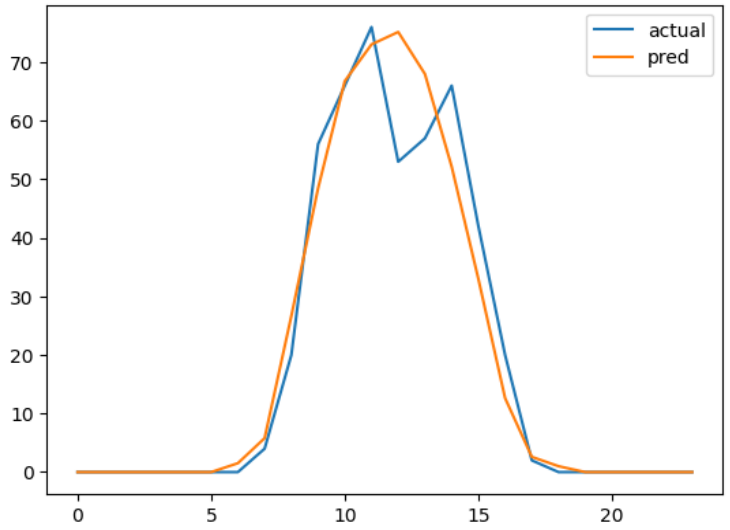
Linear Regression

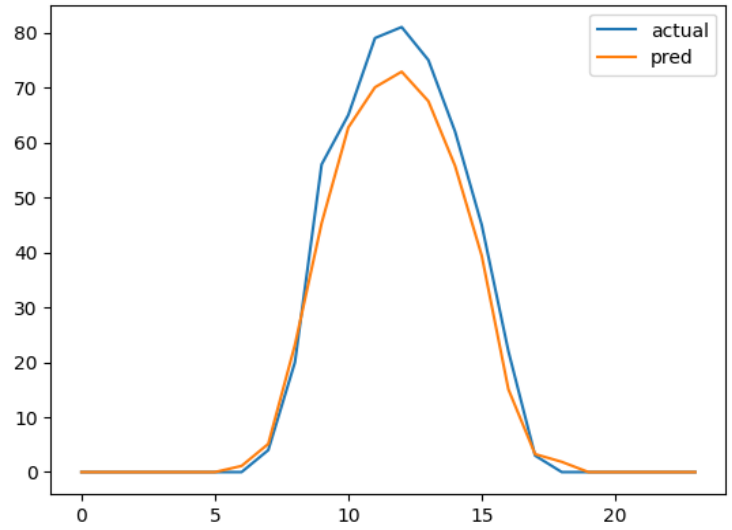


Modeling

Linear Regression – by timeslots



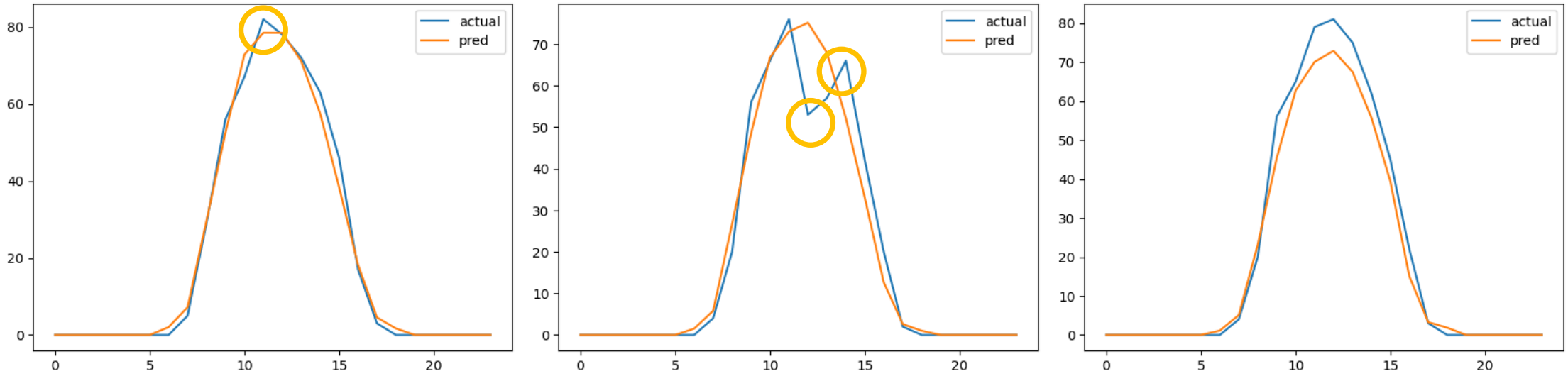




model	data	Total incentive (10.25~10.31)	notes
Linear Regression	M0~M4 (predicted value)	6956	Devided by timeslots

Modeling

Linear Regression – by timeslots



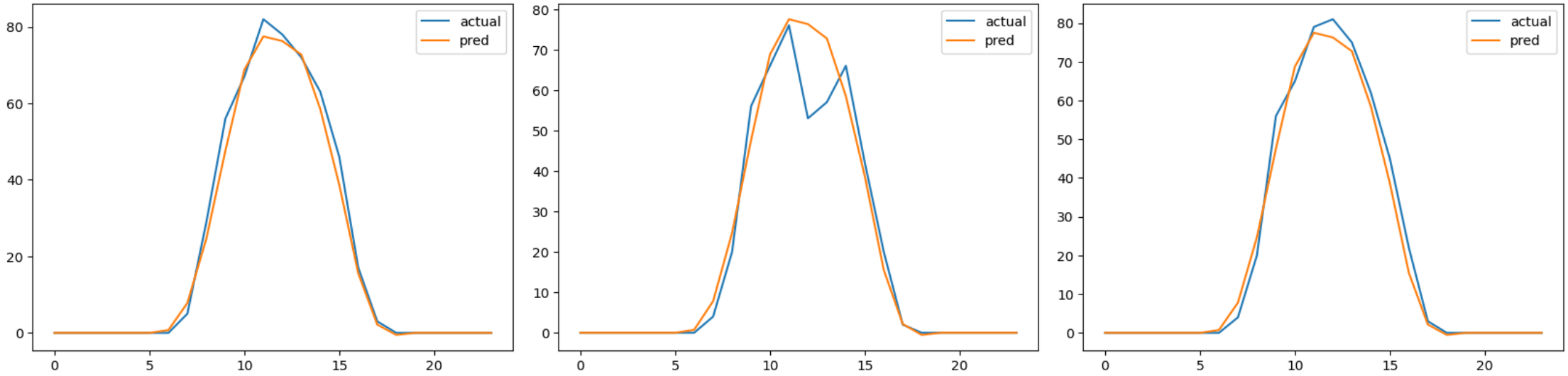
model			notes
Linear Regression	M0~M4 (predicted value)	6956	Devided by timeslots

Bad predictions for Spike points

Generally predict well, but sometimes fails to predict the midday part

Modeling

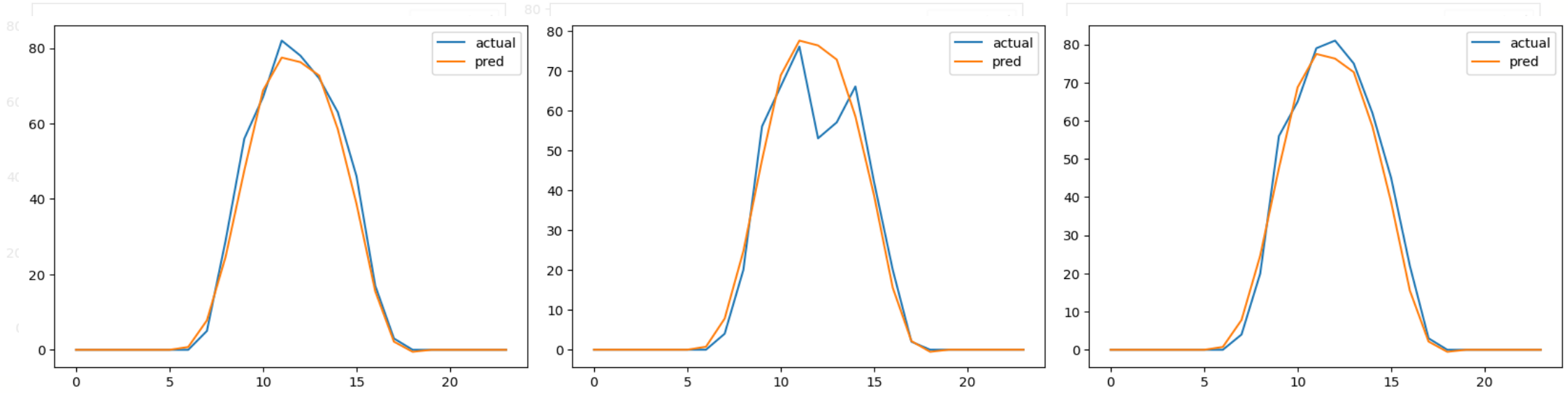
Linear Regression – by seasons



model	data	Total incentive (10.25~10.31)	notes
Linear Regression	M0~M4 (predicted value)	9020	Devided by seasons

Modeling

Linear Regression – by seasons

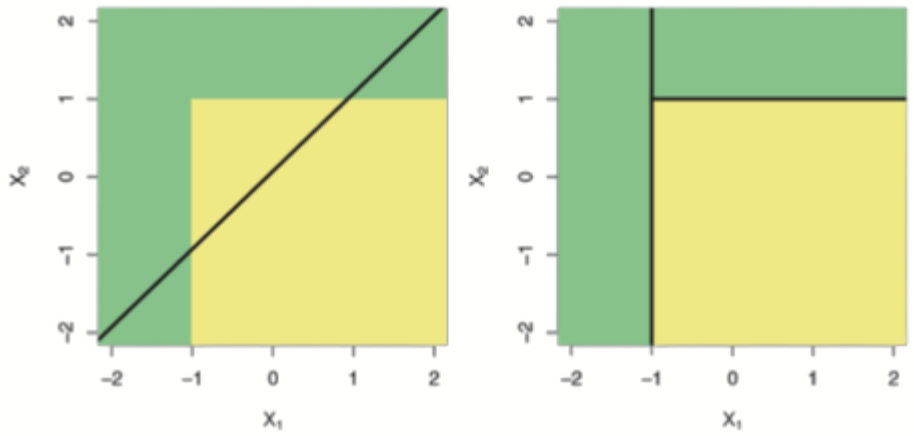


Bad predictions for Spike points

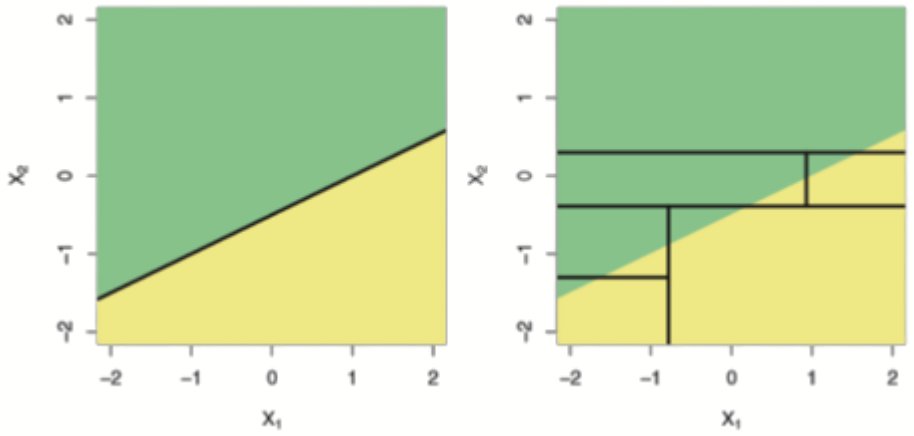
Have a tendency to predict afternoon time better than LR-by time slot

Modeling

Weak point of Tree-base model



In case of decision boundary is non-linear



In case of decision boundary in linear

Tree-base models are likely to underperform if the dependent variable and independent variables exhibit linear relationship

Modeling

Weak point of Tree-base model

Second, random forests may fail in extrapolation problems where predictions are required at points out of the domain of the training dataset.

...

As a consequence, the predictions given by random forests are always within the range of response values in the training dataset, which is problematic if the response values in the target dataset tend to fall outside this range.

Haozhe, Z., Dan , N., Zhengyuan, Z. (2019). *Regression-Enhanced Random Forests,* 1-2. doi:10.48550/arXiv.1904.10416

Also, Tree-based models tend to perform poorly on datasets that fall outside the range of the input data

Extrapolation problem

Modeling

Weak point of Tree-base model

Regression-Enhanced Random Forest Algorithm

Step 1 : Extend the p-dimensional predictor X to a (p+q)-dimensional predictor X* by adding higher-order, interaction or other known parametric functions of X

Step 2 : Run Lasso of Y on X* with a pre-specified penalty parmeter λ. Let $\hat{\beta}_\lambda$ be the estimated coefficient, and ε^λ = Y − X* $\hat{\beta}_\lambda$ be the residual from Lasso. Create a new training dataset C^λ = {C^λ = ($X_i, \varepsilon_i^\lambda$):i=1,⋯,N}.

Step 3 : Build a random forest...

⋮

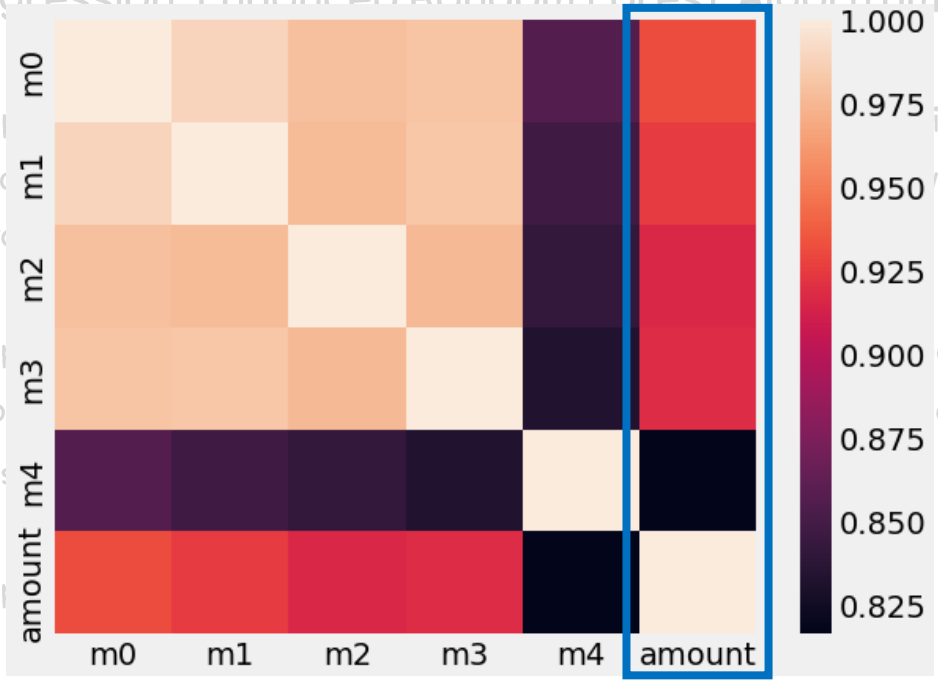
Haozhe, Z., Dan , N., Zhengyuan, Z. (2019). *Regression-Enhanced Random Forests*, 4. doi:10.48550/arXiv.1904.10416

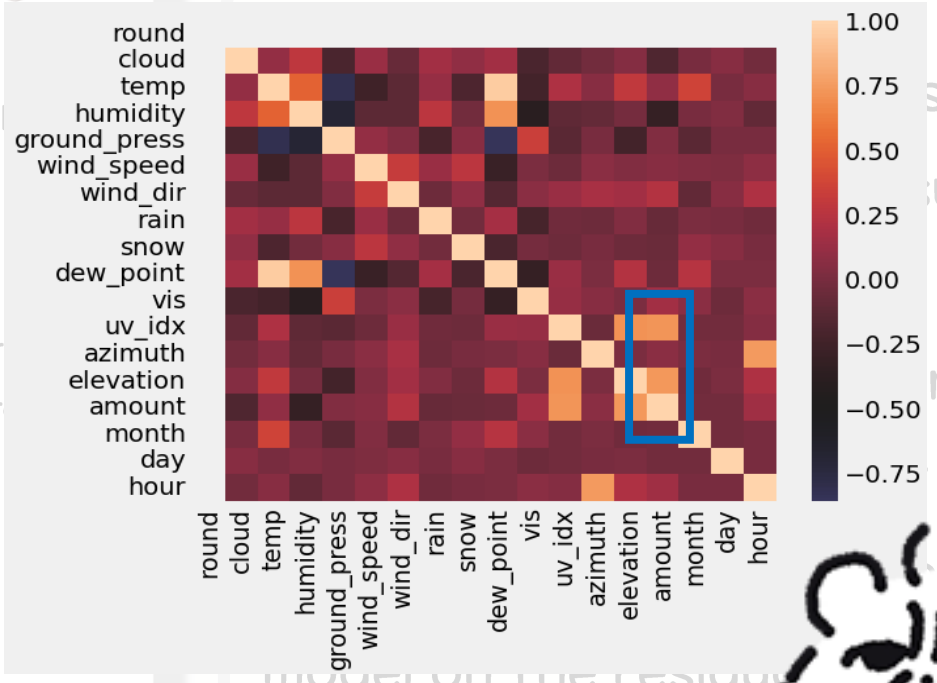
The paper suggests
a solution for this issue

1. Fit a linear regression model using X variables for Y
2. Then, fit a random forest model on the residual error $Y - \hat{Y}$

Modeling

variables that have a very high correlation with Y

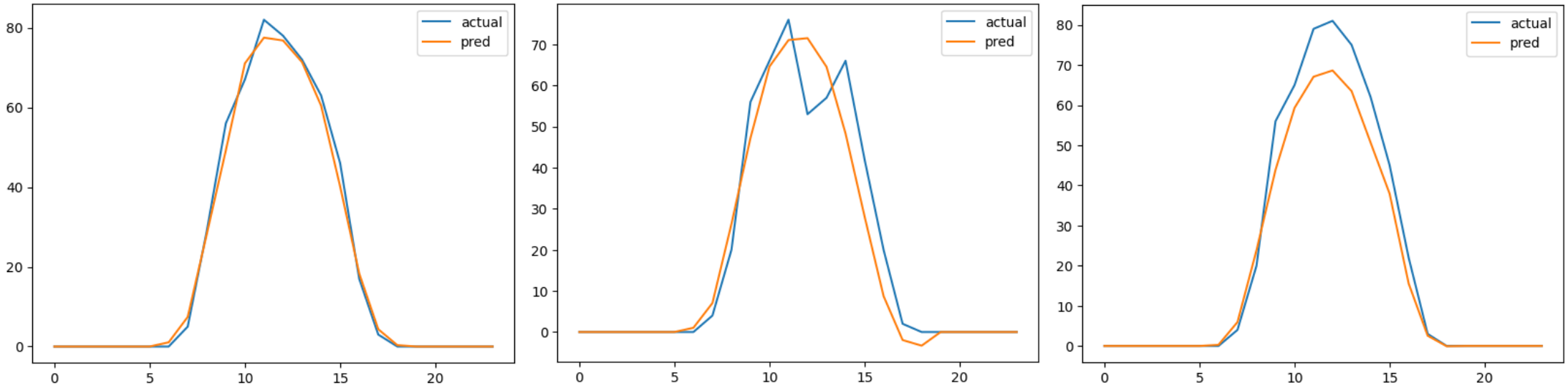




Let's adopt the idea from the paper!

Modeling

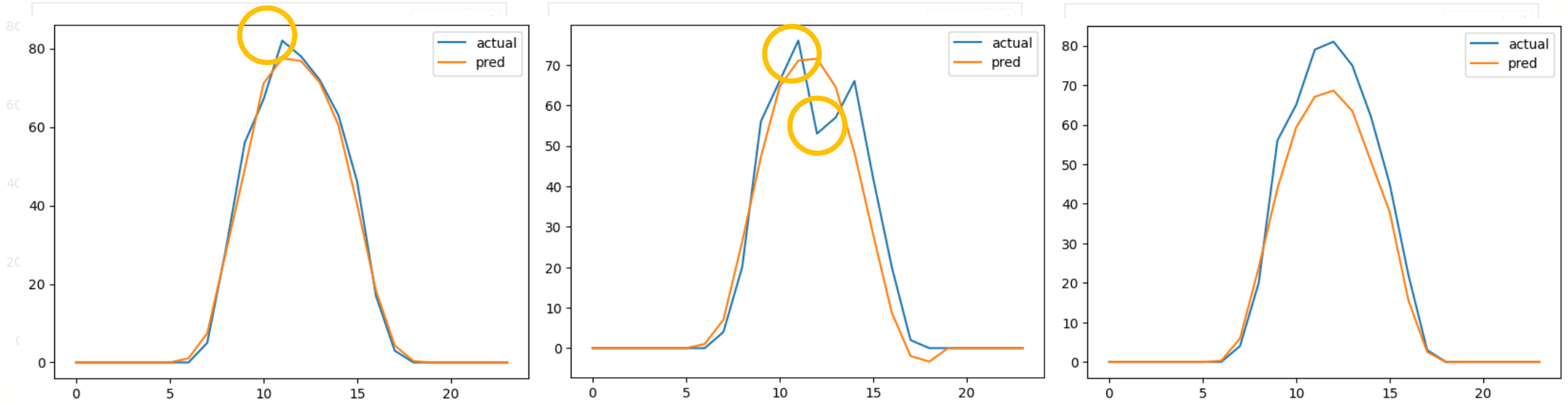
Linear Regression + LGBM



model	data	Total incentive (10.25~10.31)	notes
Linear Regression	M0~M3 , UX_idx, elevation	7184	Divided by timeslots
LGBM	Variables which is not used for the previous model		

Modeling

Linear Regression + LGBM



Bad predictions for Spike points

Generally predict well, but sometimes

fails to predict the midday part

Modeling

Linear Regression + LGBM

model	data	Total incentive (10.25~10.31)	notes
Linear Regression	M0~M3 , UX_idx, elevation	7221	Divided by timeslot Only Oct-Nov were used
LGBM	Variables which is not used for the previous model (LR)		

model	data	Total incentive (10.25~10.31)	notes
Linear Regression	M0~M3 , UX_idx, elevation	7092	Divided by timeslot Only Sep – Nov were used
LGBM	Variables which is not used for the previous model (LR)		

In an attempt to reflect the seasonal effects, the training set was reduced, but the performance did not show significant differences

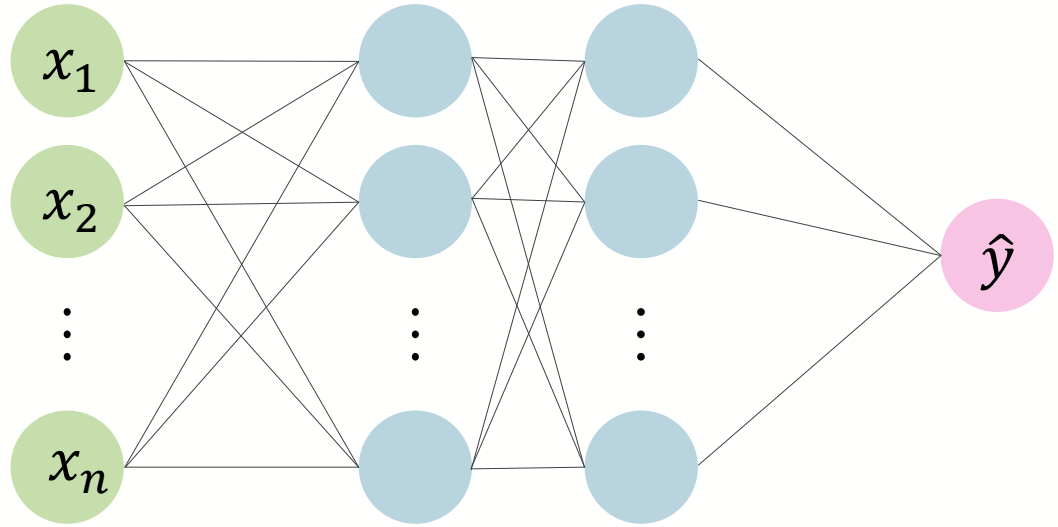
Modeling

MLP + LGBM

Multi Layer Perceptron

The most basic Deep Learning Model

Inspired by neural networks of biological brains



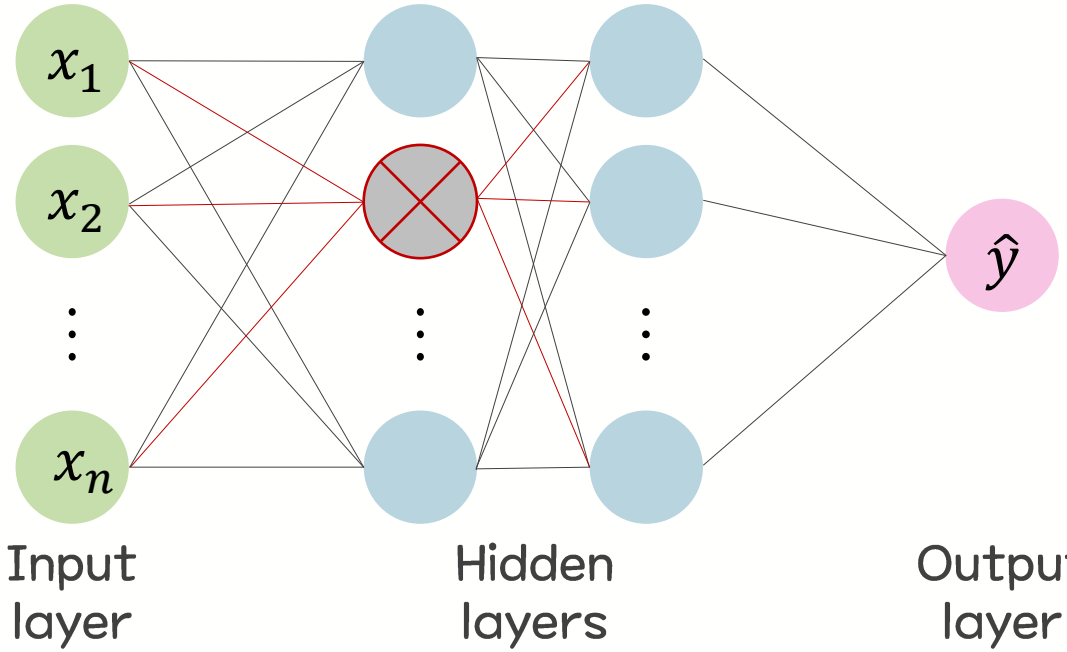
Modeling

MLP + LGBM

Dropout

During training, randomly drop nodes

In actual prediction, use all weights



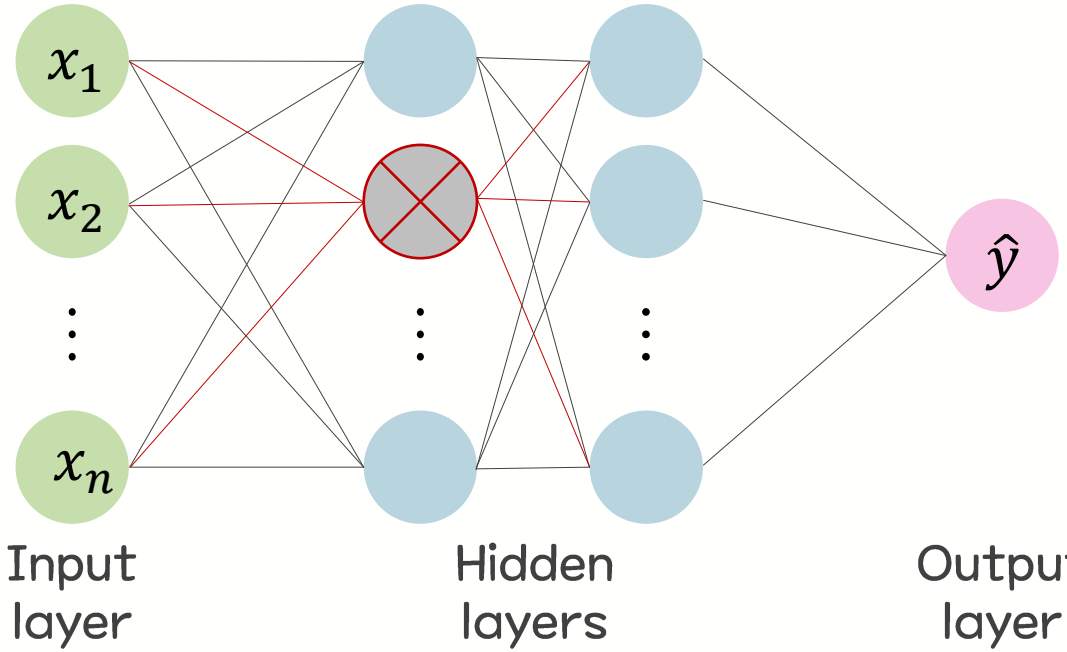
Modeling

MLP + LGBM

Dropout

During training, randomly drop nodes

In actual prediction, use all weights



Modeling

MLP + LGBM

Dropout

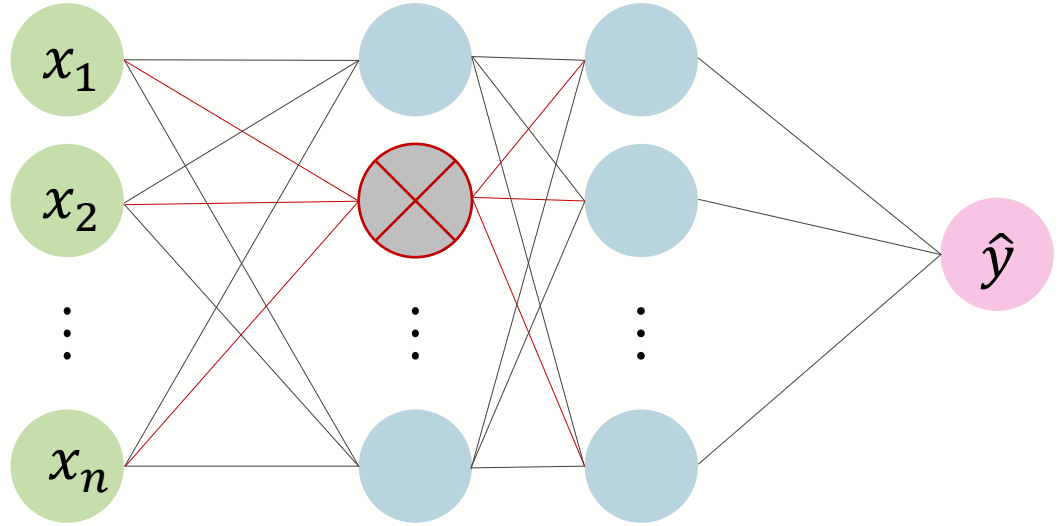
Can mimic the effect of ensembling multiple models

→ prevents overfitting

During training, randomly drop nodes

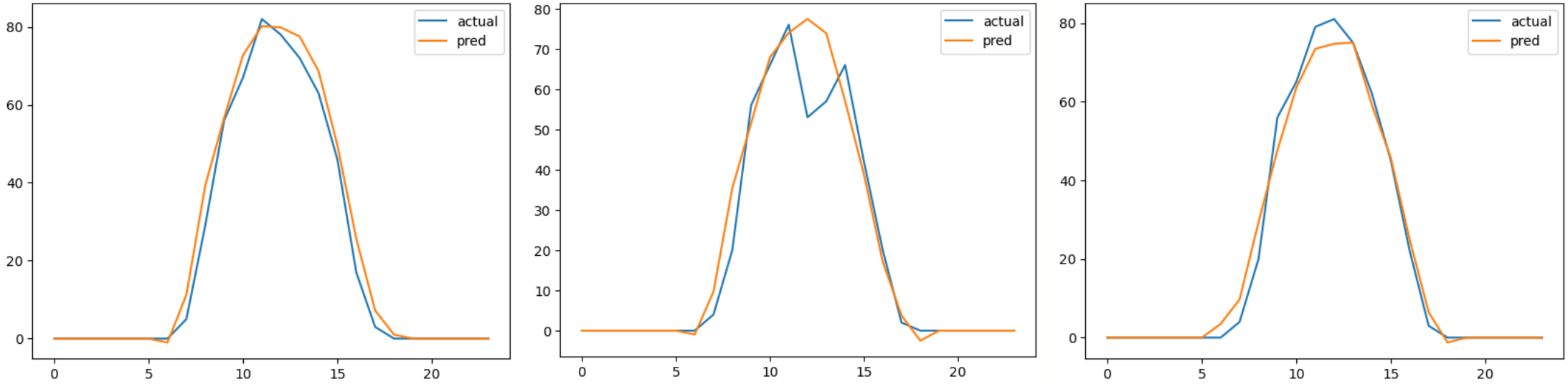
In actual prediction, use all weights

When applying dropout, training speed slows down by 2-3 times



Modeling

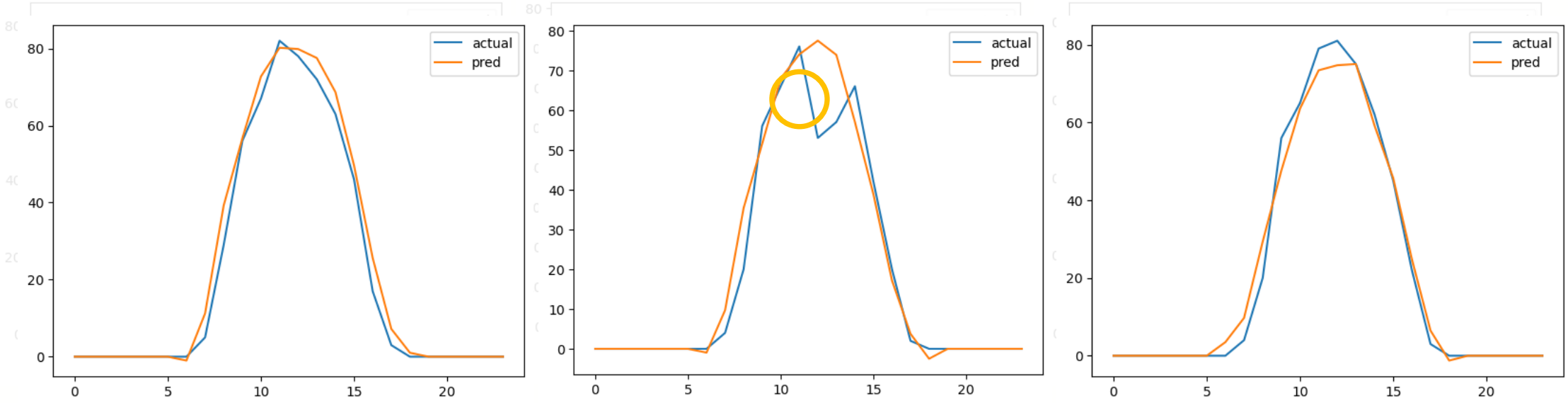
MLP + LGBM



model	data	Total incentive (10.25~10.31)	notes
Multi Layer Perceptron	M0~M3 , UV_idx , elevation	9156	Divided by timeslots
LGBM	Variables which is not used for the previous model (MLP)		

Modeling

MLP + LGBM



Still bad prediction for spike points

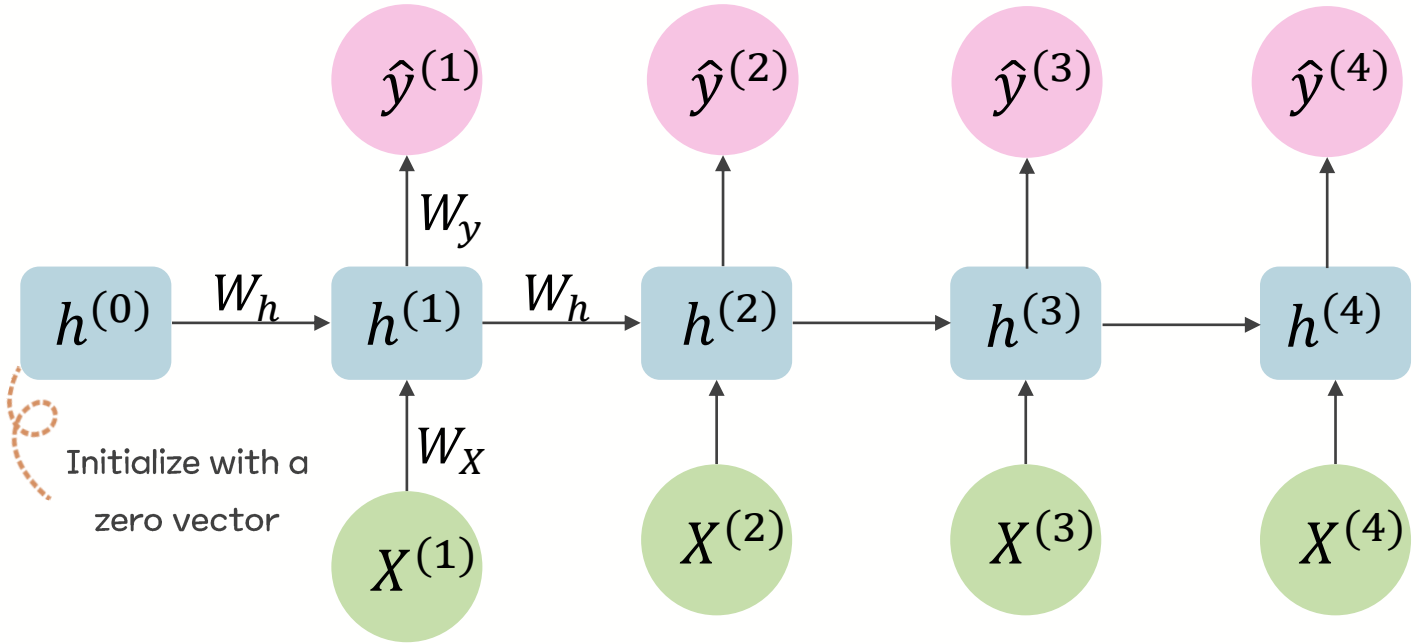
Overall, perform better than LR + LGBM

Modeling

RNN + LGBM

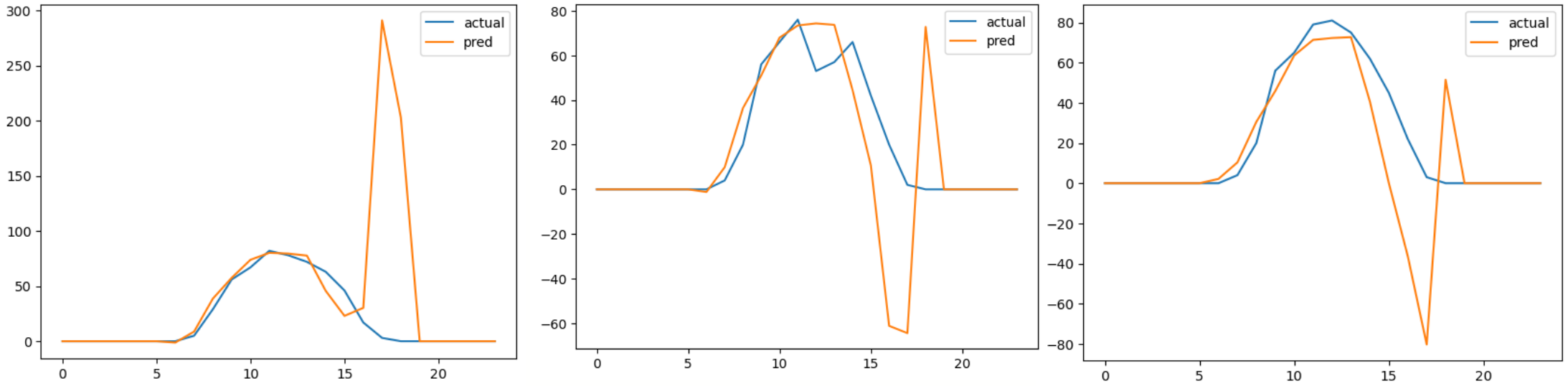
Recurrent Neural Network

A deep learning model suitable for time-series data that remembers past time point information using a Hidden State



Modeling

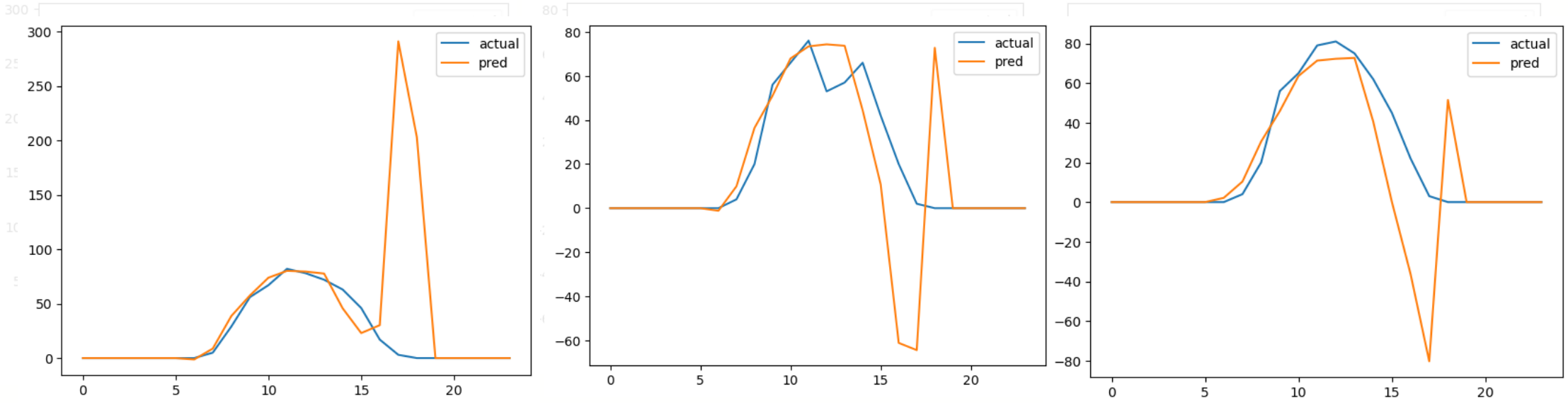
RNN + LGBM



model	data	Total incentive (10.25~10.31)	notes
Vanilla RNN	M0~M3 , UV_idx , elevation	6657	Divided by timeslots
LGBM	Variables which is not used for the previous model (RNN)		

Modeling

RNN + LGBM



The model seems to have not learned well for the afternoon time slot

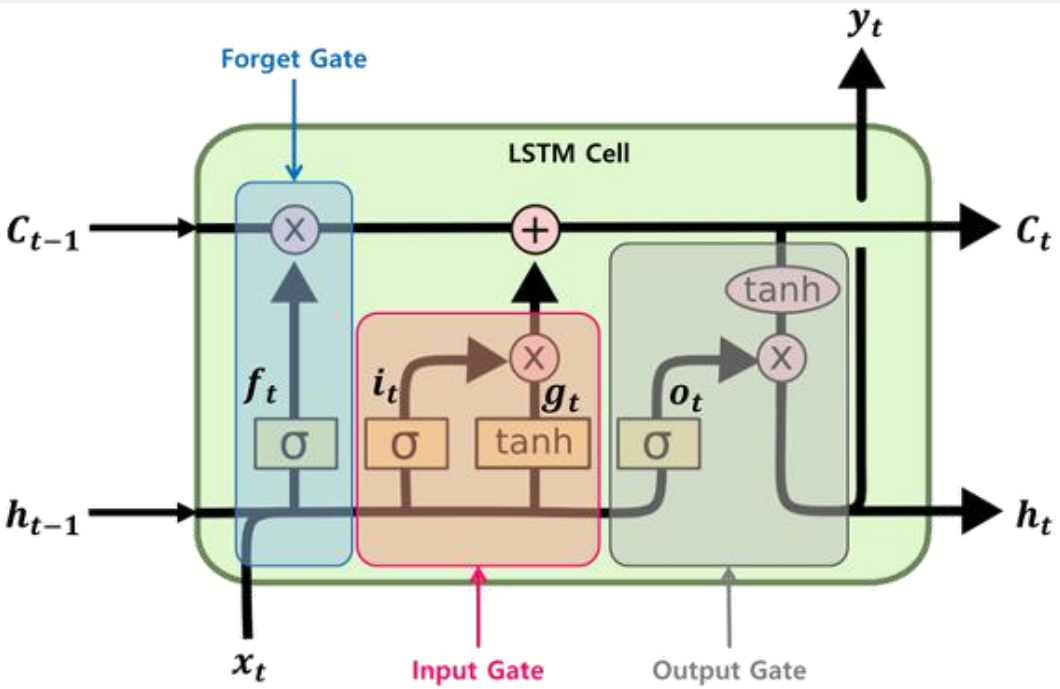
Modeling

LSTM

Long Short-Term Memory

RNN-based Time series model

Relieve the long-term memory issue by adding Cell State



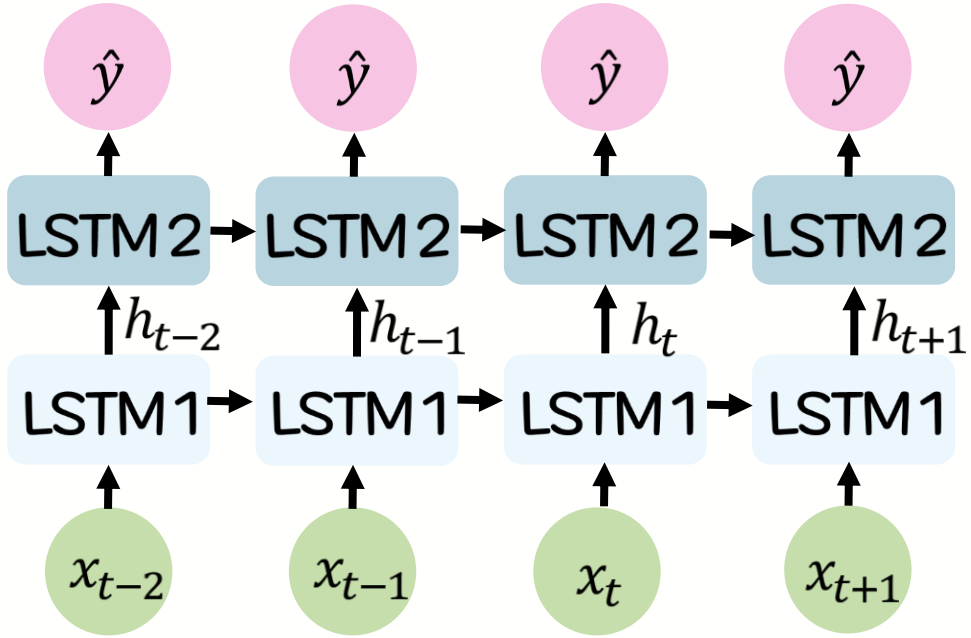
Modeling

LSTM

Stacked LSTM

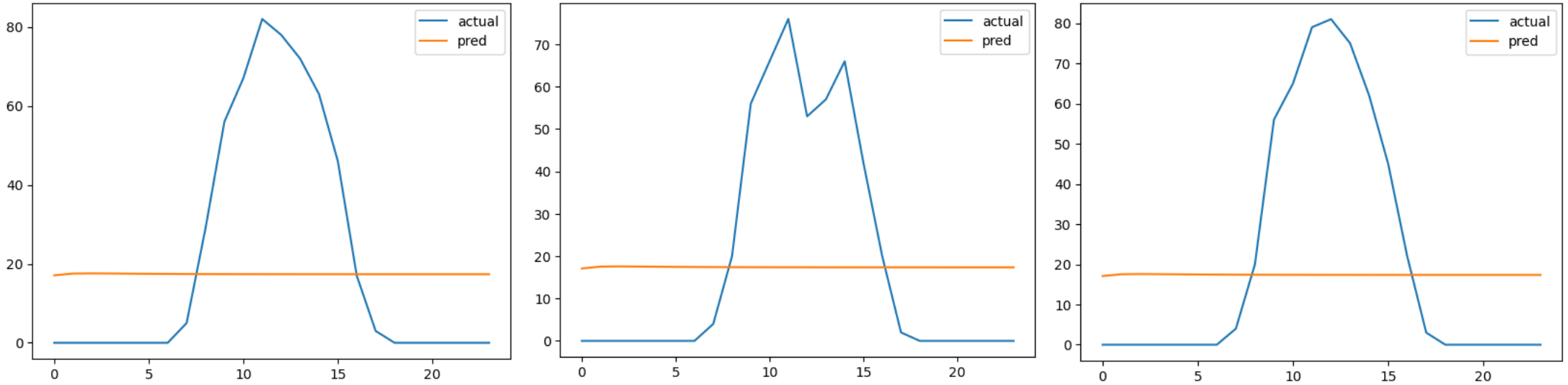
Architecture with more than 2 stacked LSTM layers

Uses the hidden state of the previous layer as input



Modeling

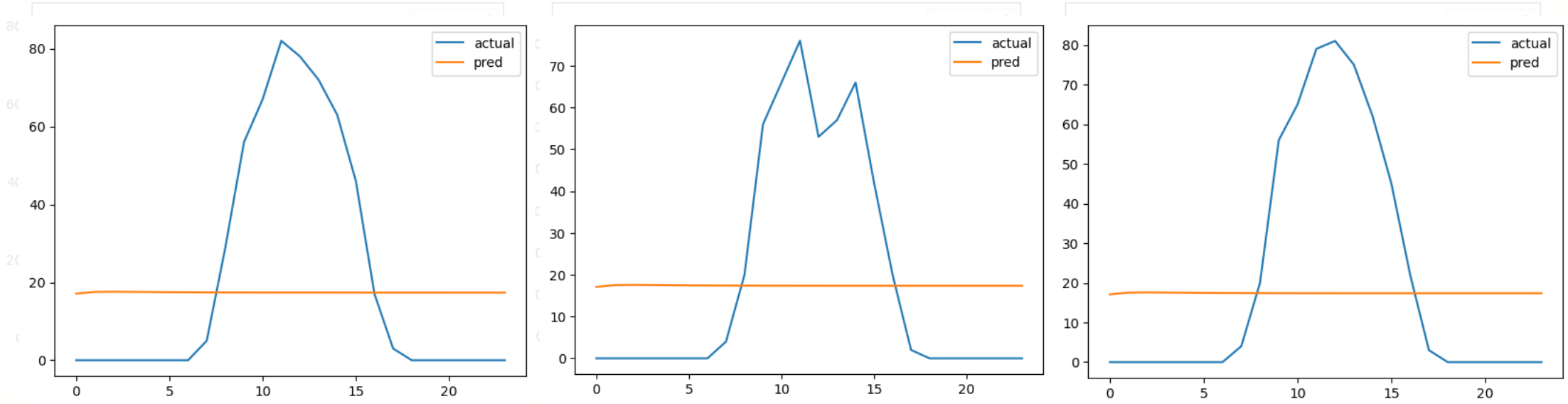
LSTM



model	data	Total incentive (10.25~10.31)	notes
Stacked LSTM	Predicted generation/weather from m0~m3	879	Divided by timeslots Derived variables are used(season/time)

Modeling

LSTM



model	data	Total incentive (10.25~10.31)	notes
Stacked LSTM	Predict each store from m0~m3	8/9	Divided by timeslots Derived variables are used(season/time)

Predict the same value for all time…

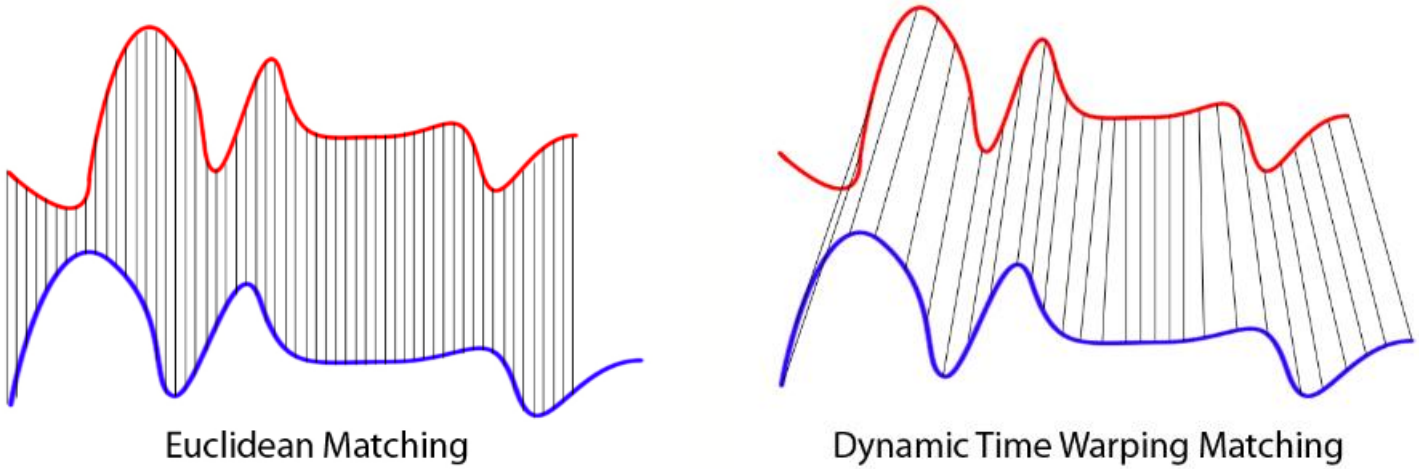
Preview for the nextweek

Preview

Time-series clustering

Dynamic Time Wrapping

Algorithm that compare the similarity between the time series data

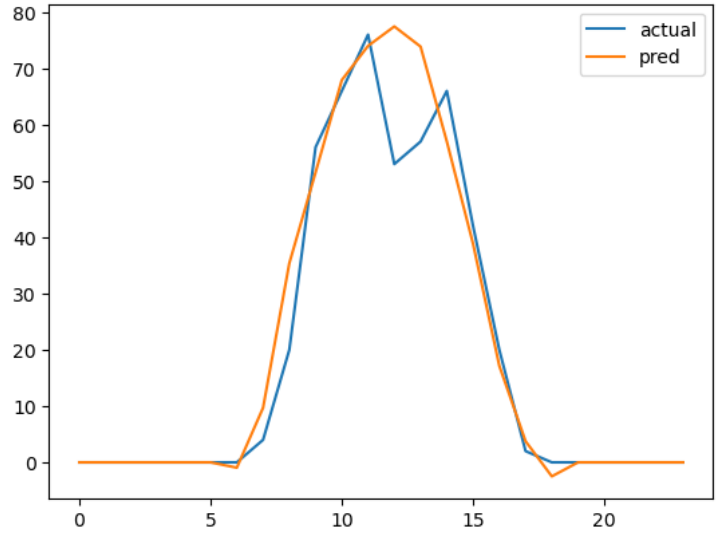


Euclidean Matching

Dynamic Time Warping Matching

Preview

Dynamic Time Wrapping

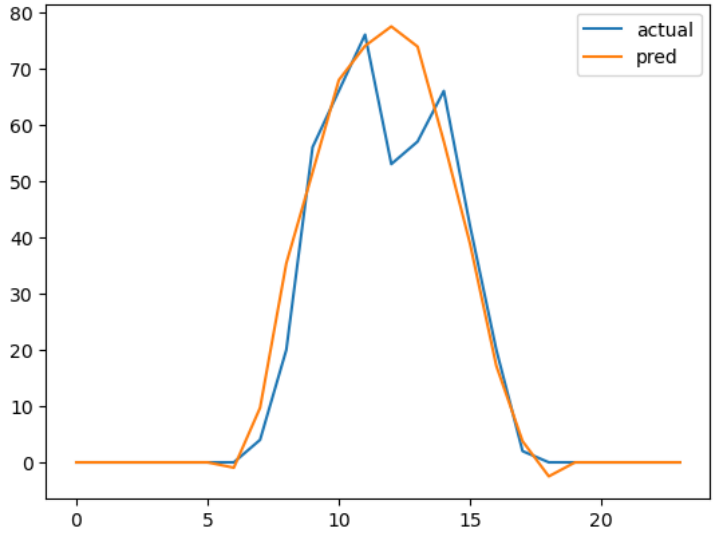


Is it possible to separate dates

with such patterns using time series clustering?

Preview

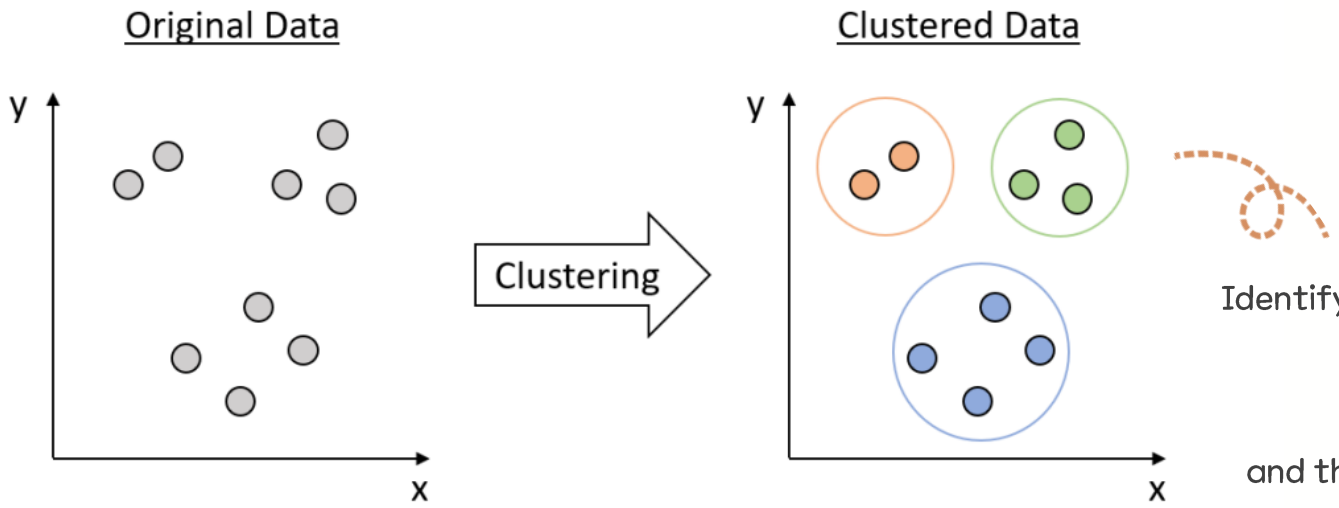
Dynamic Time Wrapping



Cluster the generation patterns, create models for each cluster, and use the corresponding cluster's model for future predictions

Preview

Error clustering



Identify the characteristics of clusters based on
the errors of each model,
including model performance,
and their relationship with weather conditions

Cluster errors of the predicted generation amount,
create models for each cluster, and use
the corresponding cluster's model for future predictions

Preview

AutoML

AutoML

전처리, 모델 학습 및 평가 등의 작업을 최대한
자동화하여 생산성과 효율을 높이기 위하여 등장

PYCARET

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.9618	0.9895	0.9618	0.9636	0.9623	0.9387	0.9391	
lightgbm	Light Gradient Boosting Machine	0.9585	0.9899	0.9585	0.9607	0.9590	0.9333	0.9338	
rf	Random Forest Classifier	0.9540	0.9889	0.9540	0.9560	0.9545	0.9258	0.9262	
xgboost	Extreme Gradient Boosting	0.9540	0.9891	0.9540	0.9559	0.9545	0.9259	0.9263	
et	Extra Trees Classifier	0.9528	0.9852	0.9528	0.9545	0.9532	0.9236	0.9241	
dt	Decision Tree Classifier	0.9506	0.9692	0.9506	0.9519	0.9509	0.9202	0.9206	
lda	Linear Discriminant Analysis	0.9315	0.9733	0.9315	0.9329	0.9320	0.8892	0.8894	
knn	K Neighbors Classifier	0.9270	0.9764	0.9270	0.9295	0.9275	0.8814	0.8821	
lr	Logistic Regression	0.9237	0.9780	0.9237	0.9241	0.9235	0.8756	0.8760	
svm	SVM - Linear Kernel	0.8911	0.0000	0.8911	0.8976	0.8849	0.8190	0.8273	
ridge	Ridge Classifier	0.8485	0.0000	0.8485	0.8440	0.8327	0.7452	0.7551	
nb	Naive Bayes	0.7362	0.9456	0.7362	0.8030	0.7252	0.6357	0.6562	
ada	Ada Boost Classifier	0.6026	0.7220	0.6026	0.7153	0.5245	0.2720	0.3554	
dummy	Dummy Classifier	0.5017	0.5000	0.5017	0.2517	0.3352	0.0000	0.0000	
qda	Quadratic Discriminant Analysis	0.3109	0.6742	0.3109	0.3547	0.1952	0.0493	0.0857	

Preview

Correlation

Professor KIM

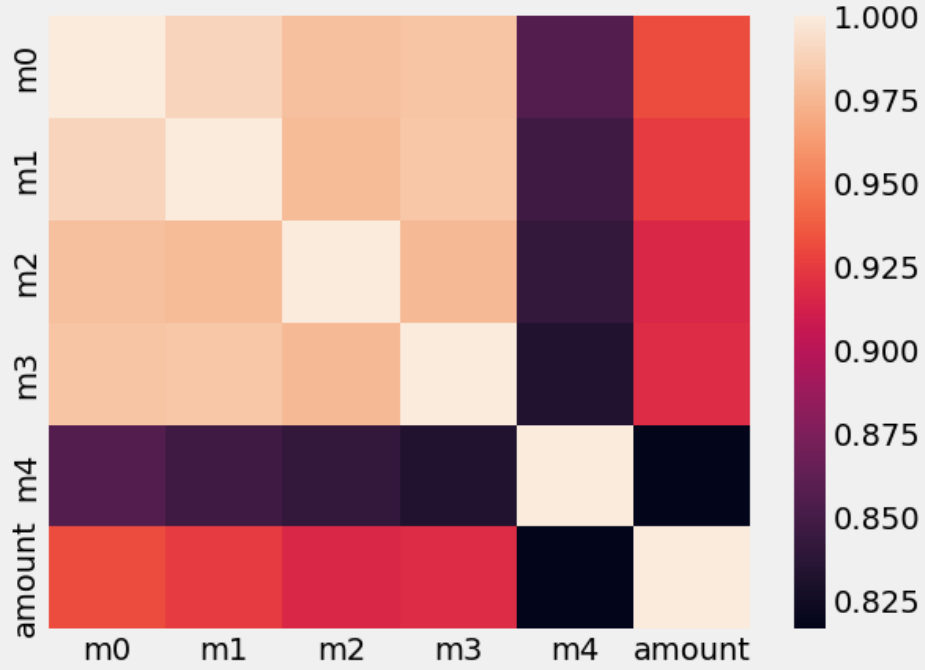
In bootstrapping, there is no guarantee that the correlation will be maintained for each sample. You must calculate the correlation for each sample

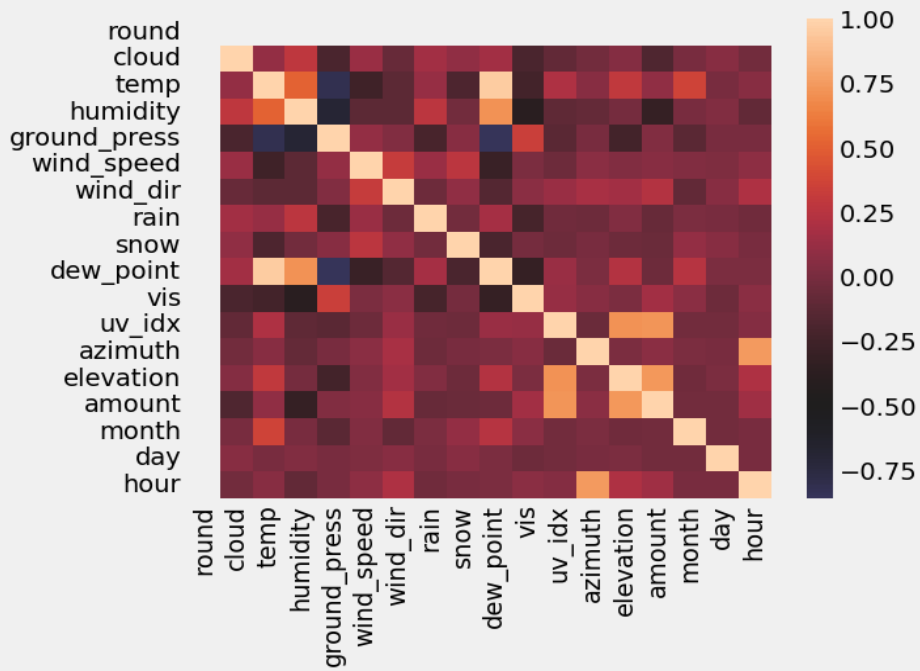
In the second semester

Statistical Data mining sample reuse method lecture

Preview

Correlation





When building the model based on seasons,

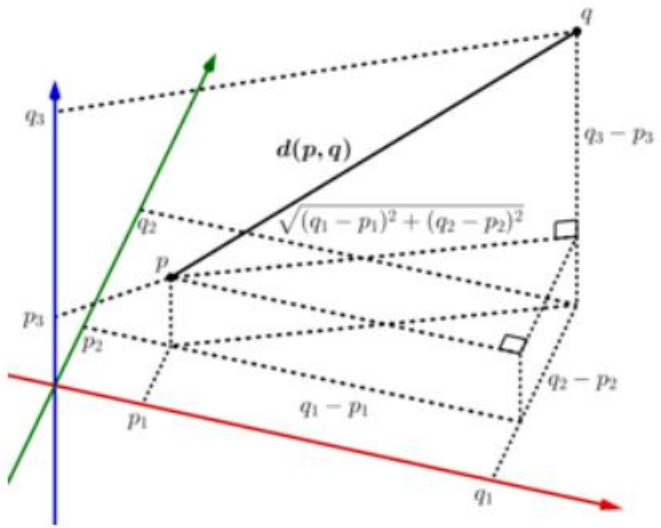
correlation must be calculated for each divided datasets

Preview

Vector Similarity

Vector Similarity

Method to find similar vectors or data points in the high-dimensional space



Preview

Vector Similarity

	round	time	cloud	temp	humidity	ground_press	wind_speed	wind_dir	rain	snow	dew_point	vis	uv_idx	azimuth	elevation
0	1	2022-06-19 01:00:00+09:00	6.0	20.03	93.0	1009.0	3.01	162.0	0.0	0.0	18.3333	16.0934	0.0	6.70428	-31.5296
1	1	2022-06-19 02:00:00+09:00	7.0	19.88	95.0	1009.0	3.16	159.0	0.0	0.0	18.3333	16.0934	0.0	22.19640	-28.4404

	time	cloud	temp	humidity	ground_press	wind_speed	wind_dir	rain	snow	dew_point	vis	uv_idx	azimuth	elevation
0	2022-06-19 01:00:00+09:00	5.871524	23.030000	91.128476	1009.000000	2.394132	152.173538	0.0	0.0	20.193333	19.193333	0.0	6.704280	-31.529640
1	2022-06-19 02:00:00+09:00	5.000000	20.046829	92.000000	1009.000000	2.490000	133.000000	0.0	0.0	20.010169	16.100000	0.0	22.196370	-28.440428
2	2022-06-19 03:00:00+09:00	31.668514	20.275571	92.000000	1008.012749	2.340765	139.974501	0.0	0.0	20.304918	16.257377	0.0	35.919394	-22.437437
3	2022-06-19 04:00:00+09:00	100.000000	20.380388	93.000000	1008.000000	2.770000	142.000000	0.0	0.0	20.403077	19.004615	0.0	47.557714	-14.221450
4	2022-06-19 05:00:00+09:00	100.000000	22.030000	93.000000	1008.000000	2.557647	133.882353	0.0	0.0	20.495385	10.143077	0.0	57.378183	-4.444699
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
11611	2023-10-15 20:00:00+09:00	0.000000	18.807459	70.000000	1014.000000	6.320000	307.000000	0.0	0.0	13.204762	16.100000	0.0	277.464745	-25.379191
11612	2023-10-15 21:00:00+09:00	0.000000	17.918518	67.888518	1015.000000	5.553144	306.554073	0.0	0.0	13.300000	16.100000	0.0	287.678638	-37.409688
11613	2023-10-15 22:00:00+09:00	0.000000	17.030000	67.000000	1015.000000	5.100000	303.000000	0.0	0.0	13.055738	20.349180	0.0	301.007172	-48.655175
11614	2023-10-15 23:00:00+09:00	0.000000	18.730542	67.000000	1015.000000	5.190000	297.000000	0.0	0.0	12.183333	9.590000	0.0	320.433966	-58.056463
11615	2023-10-16 00:00:00+09:00	0.000000	14.030000	66.000000	1015.000000	5.360000	293.000000	0.0	0.0	11.700000	6.400000	0.0	349.065111	-63.421759

11616 rows × 14 columns

Process each row as a vector to measure similarity with actual data,
and process prediction using the vector with highest similarity

Preview

MLP + LGBM retrain

	Every seasonal data	Only fall data
LR	6956	9020
LR + LGBM	7139	7221
MLP + LGBM	9156	To be revealed next week

There are some case where the model performance improved with fall data

Let's also apply this to MLP+LGBM, which currently shows the best performance

Preview

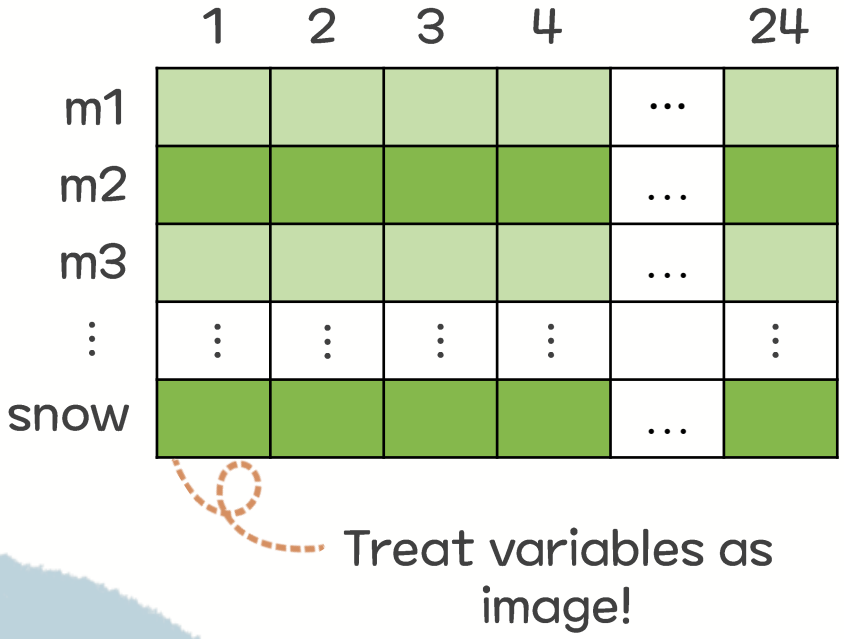
1D– CNN

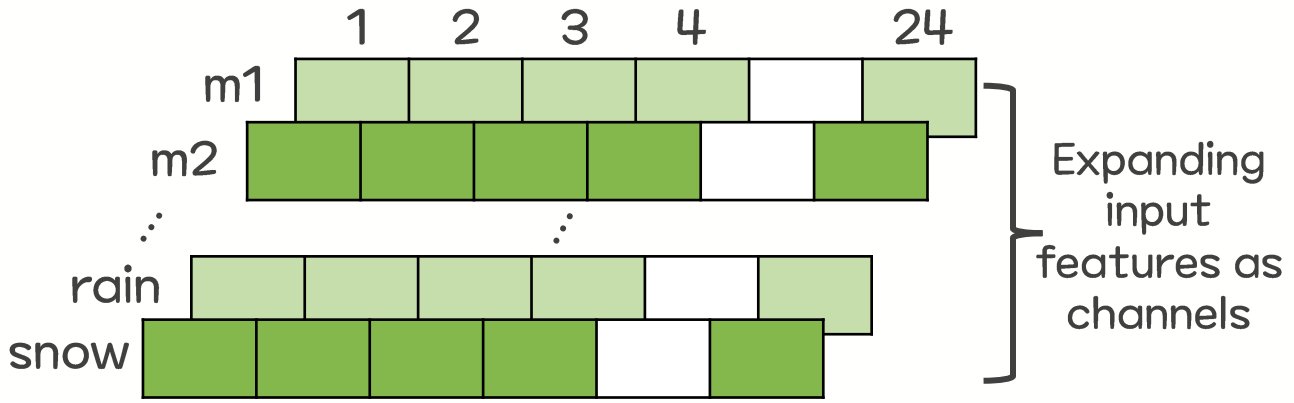
1D - CNN

A model that can be applied to time series data

Stack each variable according to time points

and treat it as an image for CNN application





To be continued...