동서발전 태양광 발전량 예측 AI 경진대회

서강대학교 대학원 컴퓨터공학부 인공지능전공
120200119 김종현
120200369 하동균
120210198 윤동성

목차

대회개요

대회 설명

프로젝트 개요

[배경]

태양광 발전은 매일 기상 상황과 계절에 따른 일사량의 영향을 받습니다.
이에 대한 예측이 가능하다면 보다 원활한 전력 수급 계획이 가능합니다.
인공지능 기반 태양광 발전량 예측 모델을 만드는 프로젝트 입니다.

[주제]

시간대별 태양광 발전량 예측

[주최/주관]

주최 : 한국동서발전(주)

대회 주요 일정

리더보드 스코어 26.87% (61/227)

● WINNER ● 1% ● 4% ● 10%

#	팀	팀 멤버	점수	제출수	등록일
61	커브		8.16232	54	2시간 전
1	양현준		5.94219	53	9시간 전
2	IAI Lab		5.99754	34	하루 전
3	물린다		6.2694	26	한 달 전
4	AlanTuring		6.28166	40	6일 전
5	고라파덕		6.59466	32	한 시간 전
6	YEJang		6.60375	48	한 시간 전
7	Seven_to11		6.70758	208	36분 전
8	이름뭘로짓냐		6.78724	24	5시간 전
9	(주)로보볼트		6.91218	68	2시간 전
10	AutoEncoder Master		6.91696	6	2달 전

평가방법

2. 평가

- 심사 기준 : NMAE-10(Normalized Mean Absolute Error)
- Public 평가 :
  - 학습용 제공 데이터를 이용하여 미래 한 달간 발전량 예측 후 평가
- Private 평가 :
  - 대회 종료 시점부터 30일간 실제 발전량을 하루씩 평가
  - 1일 1회 채점 후 누적 결과 리더보드 반영
  - Private 평가 기간 제출물 업데이트 가능

대회일정

대회 기간 : 2021년 4월 7일 10:00 ~ 2021년 7월 9일 18:00

프라이빗 평가 : 2021년 6월 10일 ~ 2021년 7월 9일

코드 제출 : 2021년 7월 10일 ~ 2021년 7월 14일

최종 평가 : 2021년 7월 15일 ~ 2021년 7월 22일

최종 결과 발표 : 2021년 7월 23일

외부 데이터 및 사전학습 모델

4. 외부 데이터 및 사전학습 모델

- 예측일 전날 자정까지 확인이 가능한 데이터만 학습 및 추론 과정에서 사용 가능

ex) 2021년 6월 11일 예측 -> 2021년 6월 10일 24:00까지 획득 가능한 데이터만 사용

(6월 10일 기상 관측 정보, 6월10일에 예보한 6월 11일 예보 등...)

- 예측 이전 시점의 데이터만 사용 가능
- 공공데이터와 같이 누구나 얻을 수 있고 법적 제약이 없는 외부 데이터 허용
- 사전학습 모델의 경우 사전학습에 사용된 데이터를 명시해야함
- 대회 진행 중 data leakage 및 규칙 위반 사항이 의심되는 경우 코드 제출 요청을 할 수 있으며 요청 2일 이내 코드 미제출 혹은 외부 데이터 사용이 확인되었을 경우 리더보드 기록 삭제
- 최종 평가시 외부데이터 및 출처 제출

제출 양식

sample_submission.csv - 예측한 발전량 제출 양식

- publci LB : 2021년 2월 예측
- private LB : 2021년 6월 9일 ~ 2021년 7월 8일 30일간 예측, 평가기간 제출 가능, 예측 전날 선택된 제출물 평가

※주의1 : 2021년 2월 예측시점 전날에 확인 가능한 정보만을 feature로 사용 가능, fcst 데이터 사용시 주의

※주의2 : public 평가기간 data leakage가 의심되는 경우 코드 제출을 요청할 수 있으며 2일 내 미제출 또는 data leakage가 확인 되었을 시 LB 기록이 삭제

※주의3 : 대회 기간 동안 지속적인 data leakage가 확인된 팀은 참가 자격 박탈

	time	floating	warehouse	dangjin	ulsan
0	2021-02-01 1:00	-23.8168	-25.5496	-57.2173	-9.75394
1	2021-02-01 2:00	-21.8905	-18.8085	-47.8016	-14.0591
2	2021-02-01 3:00	-36.2473	-26.6238	-74.715	-19.4221
3	2021-02-01 4:00	-39.7011	-25.5391	-73.4612	-21.5128
		...			
1385	2021-07-08 18:00	0	0	0	0
1386	2021-07-08 19:00	0	0	0	0
1387	2021-07-08 20:00	0	0	0	0
1388	2021-07-08 21:00	0	0	0	0
1389	2021-07-08 22:00	0	0	0	0
1390	2021-07-08 23:00	0	0	0	0

평가 산식

평가 산식(NMAE-10)

- 4개의 발전소 반전량을하나로 합하여 평가
- 4개 발전소 총 발전용량으로 정규화
- 발전용량의 10% 이상 발전된 데이터만으로 평가

```python
def sola_nmae(answer_df, submission_df):
    submission = submission_df[submission_df['time'].isin(answer_df['time'])]
    submission.index = range(submission.shape[0])

    # 시간대별 총 발전량
    sum_submission = submission.iloc[:,1:].sum(axis=1)
    sum_answer = answer_df.iloc[:,1:].sum(axis=1)

    # 발전소 발전용량
    capacity = {
        'dangjin_floating':1000, # 당진수상태양광 발전용량
        'dangjin_warehouse':700, # 당진자재창고태양광 발전용량
        'dangjin':1000, # 당진태양광 발전용량
        'ulsan':500 # 울산태양광 발전용량
    }

    # 총 발전용량
    total_capacity = np.sum(list(capacity.values()))

    # 총 발전용량 절대오차
    absolute_error = (sum_answer - sum_submission).abs()

    # 발전용량으로 정규화
    absolute_error /= total_capacity

    # 총 발전용량의 10% 이상 발전한 데이터 인덱스 추출
    target_idx = sum_answer[sum_answer>=total_capacity*0.1].index

    # NMAE(%)
    nmae = 100 * absolute_error[target_idx].mean()

    return nmae
```

대회개요

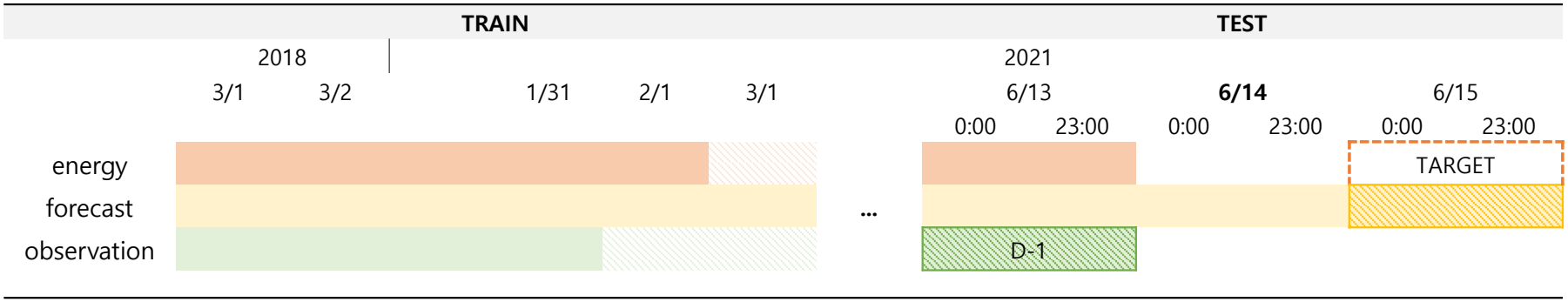
기상데이터 설명

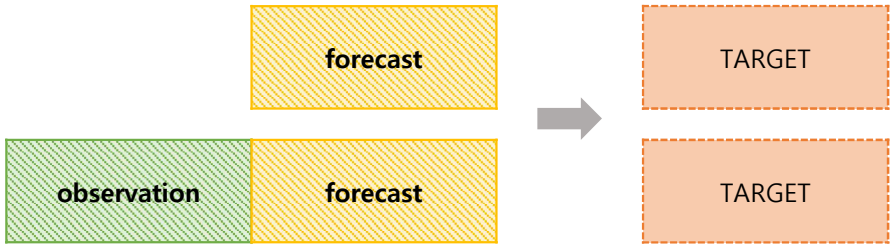
대회 제공 데이터

파일명	shape	설명		
site_info.csv	4, 7	발전소 설치 정보		
energy.csv	25632, 5	발전소별 시간당 발전량		
dangjin_fcst_data.csv	162208, 7	기상정보	당진	예보
dangjin_obs_data.csv	25626, 8			관측
ulsan_fcst_data.csv	162208, 7		울산	예보
ulsan_obs_data.csv	25632, 8			관측
sample_submission.csv	1392, 4	2월(public)과 6, 7월(private) 발전량 예측값		

2018-03-01 00:00:00 : 2021-01-31 23:00:00 = **25632**hrs

학습데이터셋 구성


TRAIN
TEST
2018
2021
3/1
3/2
1/31
2/1
3/1
6/13
6/14
6/15
0:00
23:00
0:00
23:00
0:00
23:00
energy
forecast
observation
...
TARGET
D-1


forecast
observation
forecast
TARGET
TARGET

기상관측 데이터프레임 (25632, 8)

	지점	지점명	일시	기온(°C)	풍속(m/s)	풍향(16방위)	습도(%)	전운량(10분위)
0	152	울산	2018-03-01 00:00	8.2	3.9	340.0	98.0	10.0
1	152	울산	2018-03-01 01:00	7.0	4.1	320.0	97.0	10.0
2	152	울산	2018-03-01 02:00	6.5	5.9	290.0	80.0	NaN
3	152	울산	2018-03-01 03:00	6.2	4.6	320.0	79.0	3.0
4	152	울산	2018-03-01 04:00	6.7	4.5	320.0	73.0	1.0
...	...	...	...	...	...	...	...	...
25627	152	울산	2021-01-31 19:00	8.8	2.5	200.0	50.0	5.0
25628	152	울산	2021-01-31 20:00	8.7	3.9	200.0	49.0	1.0
25629	152	울산	2021-01-31 21:00	8.4	2.4	230.0	51.0	7.0
25630	152	울산	2021-01-31 22:00	9.4	3.3	230.0	51.0	8.0
25631	152	울산	2021-01-31 23:00	9.3	3.1	200.0	56.0	9.0

...

기상예보 데이터프레임 (162208, 7)

	Forecast time	Forecast (X시간 후)	기온	습도	풍속	풍향	전운량
0		4.0	8.0	20.0	14.0	298.0	2.0
1		7.0	4.0	20.0	4.3	298.0	2.0
2	2018-03-01 11:00:00	10.0	3.0	30.0	1.9	309.0	2.0
3		13.0	0.0	40.0	1.5	318.0	2.0
4		16.0	-1.0	45.0	1.8	308.0	2.0
			...				
15		49.0	13.0	50.0	2.9	172.0	2.0
16		52.0	14.0	60.0	3.2	167.0	3.0
17	2018-03-01 11:00:00	55.0	12.0	70.0	2.6	178.0	3.0
18		58.0	11.0	75.0	2.0	200.0	3.0
19		61.0	10.0	80.0	1.2	275.0	3.0
			...				

대회개요

발전량 데이터 설명

태양광 발전소 정보

당진 수상 태양광

당진 자재창고 태양광

당진 태양광

울산 태양광

태양광 발전소 정보

	Id	Capacity(MW)	Address	Installation Angle	Incident Angle	Latitude	Longitude
1	당진수상태양광	1000					
2	당진자재창고태양광	700	충남 당진시 석문면 교로길 30	30.0	30.0	37.050753	126.510299
3	당진태양광	1000					
4	울산태양광	500	울산광역시 남구 용잠로 623	20.0	20.0	35.477651	129.380778

- InstallationAngle : 설치각(º)
- IncidentAngle : 입사각(º)

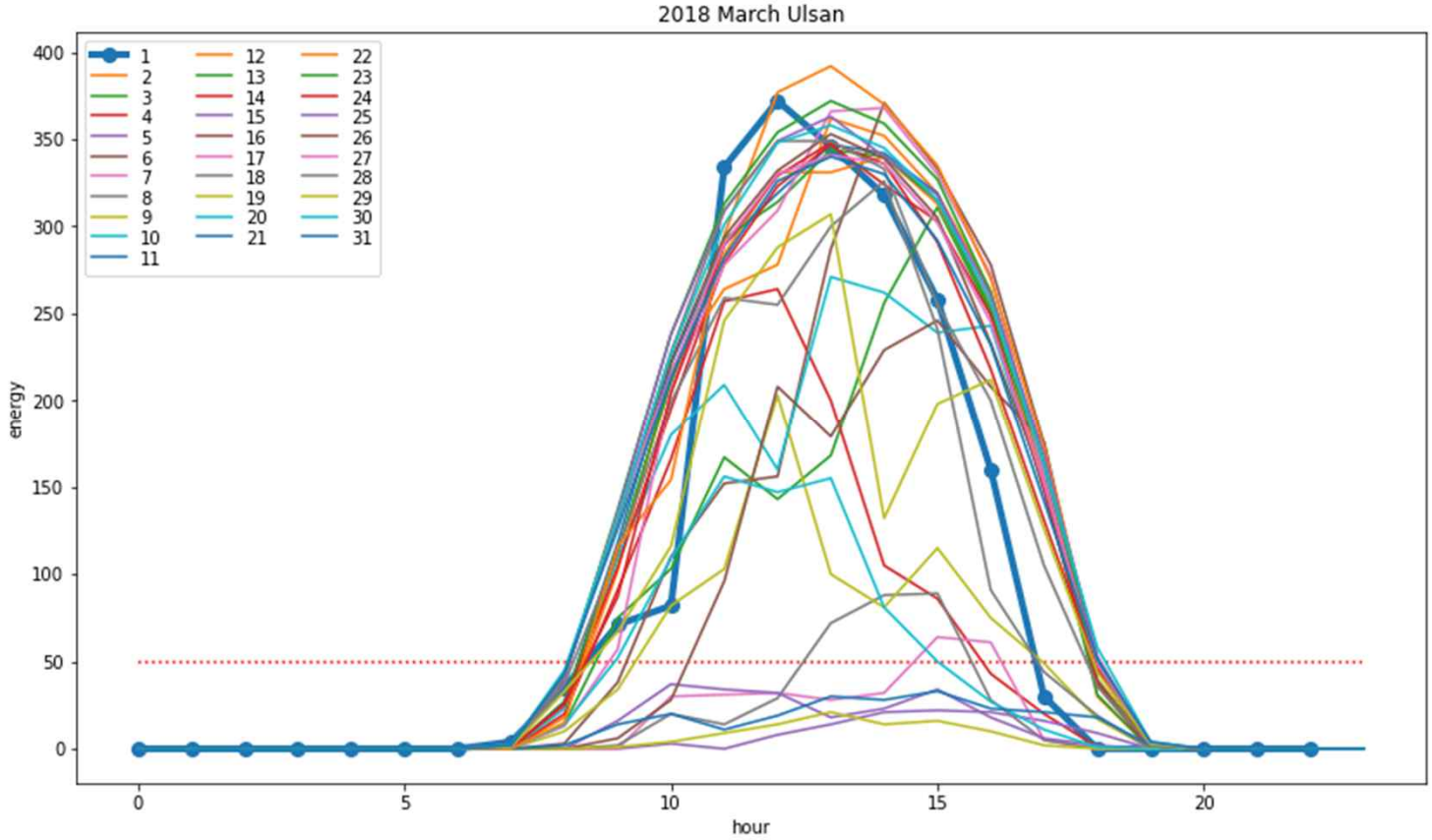
발전량 데이터프레임 (25632, 5)

(단위 kW)

	time	floating	dangjin warehouse	dangjin	ulsan
			...		
6	2018-03-01 7:00:00	0.0	0.0	0	0
7	2018-03-01 8:00:00	0.0	0.0	0	4
8	2018-03-01 9:00:00	36.0	33.0	37	35
9	2018-03-01 10:00:00	313.0	209.0	318	71
10	2018-03-01 11:00:00	532.0	296.0	490	82
11	2018-03-01 12:00:00	607.0	315.0	550	334
12	2018-03-01 13:00:00	614.0	474.0	727	372
13	2018-03-01 14:00:00	608.0	544.0	733	346
			...		

- time : 1시간 단위 계량된 시간 (ex-2018-03-01 1:00:00 => 2018-03-01 00:00:00 ~ 2018-03-01 1:00:00 1시간동안 발전량 계량)

울산의 3월 일-시간 발전량 변화



ulsan
...
0
4
35
71
82
334
372
346
...

데이터 전처리

기상예보 데이터 변환

기상예보 데이터 – 원본

	Forecast time	forecast	Weather features	*time*
22	2018-03-01 14:00	10	...	*2018-03-02 0:00*
23		13		*2018-03-02 3:00*
24		16		*2018-03-02 6:00*
25		19		*2018-03-02 9:00*
26		22		*2018-03-02 12:00*
27		25		*2018-03-02 15:00*
28		28		*2018-03-02 18:00*
29		31		*2018-03-02 21:00*
170	2018-03-02 14:00	10	...	*2018-03-03 0:00*
171		13		*2018-03-03 3:00*
172		16		*2018-03-03 6:00*
173		19		*2018-03-03 9:00*
174		22		*2018-03-03 12:00*
175		25		*2018-03-03 15:00*
176		28		*2018-03-03 18:00*
177		31		*2018-03-03 21:00*
318	2018-03-03 14:00	10	...	*2018-03-04 0:00*
319		13		*2018-03-04 3:00*

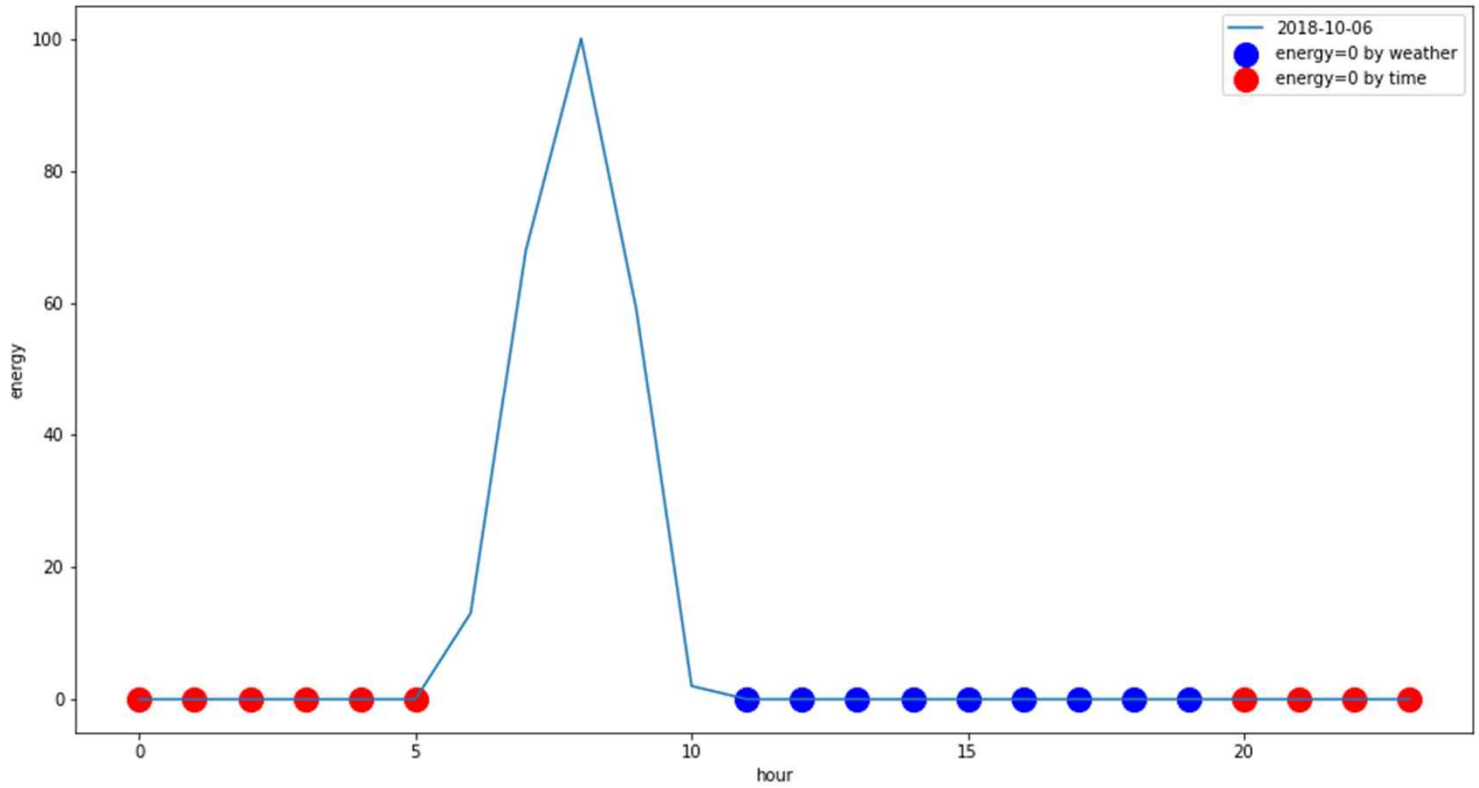
기상예보 데이터 – 변환 후

	time	Temperature_x	Humidity_x	WindSpeed_x	WindDirection_x	Cloud_x
0	**2018-03-02 0:00**	**-2**	**55**	**6.7**	**336**	**1**
1	2018-03-02 1:00	-2.33333	55	6.5	336.5	1
2	2018-03-02 2:00	-2.66667	55	6.3	337	1
3	**2018-03-02 3:00**	**-3**	**55**	**6.1**	**337.5**	**1**
4	2018-03-02 4:00	-3.33333	55	5.9	338	1
5	2018-03-02 5:00	-3.66667	55	5.7	338.5	1
6	**2018-03-02 6:00**	**-4**	**55**	**5.5**	**339**	**1**
7	2018-03-02 7:00	-4.16667	55.83333	5.3	339.8333	1
8	2018-03-02 8:00	-4.33333	56.66667	5.1	340.6667	1
9	**2018-03-02 9:00**	**-4.5**	**57.5**	**4.9**	**341.5**	**1**
10	2018-03-02 10:00	-4.66667	58.33333	4.7	342.3333	1
11	2018-03-02 11:00	-4.83333	59.16667	4.5	343.1667	1
12	**2018-03-02 12:00**	**-5**	**60**	**4.3**	**344**	**1**
13	2018-03-02 13:00	-4.5	59.16667	3.916667	344.1667	1
14	2018-03-02 14:00	-4	58.33333	3.533333	344.3333	1
15	**2018-03-02 15:00**	**-3.5**	**57.5**	**3.15**	**344.5**	**1**
16	2018-03-02 16:00	-3	56.66667	2.766667	344.6667	1
17	2018-03-02 17:00	-2.5	55.83333	2.383333	344.8333	1
18	**2018-03-02 18:00**	**-2**	**55**	**2**	**345**	**1**
19	2018-03-02 19:00	-1.5	53.33333	1.8	330.1667	1
20	2018-03-02 20:00	-1	51.66667	1.6	315.3333	1
21	**2018-03-02 21:00**	**-0.5**	**50**	**1.4**	**300.5**	**1**
22	2018-03-02 22:00	0	48.33333	1.2	285.6667	1
23	2018-03-02 23:00	0.5	46.66667	1	270.8333	1

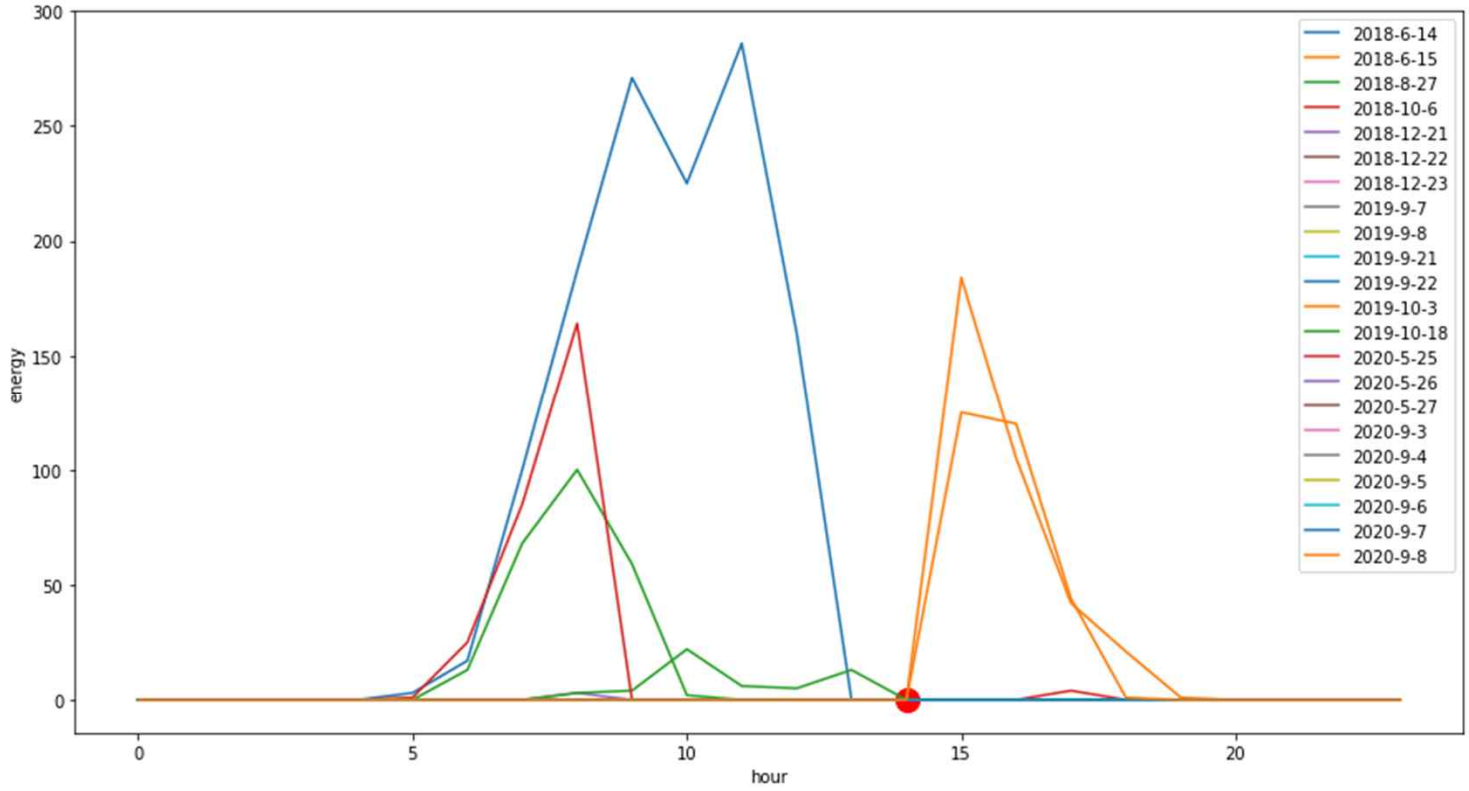
데이터 전처리

발전량==0

시간 또는 기상에 의한 발전량 0



발전량이 0인 날의 발전량 변화 그래프



성능비교

	Data length	floating	warehouse	dangjin	ulsan	Average Score
CASE0: all_time	25,632	**8.10**	**9.43**	9.97	**6.29**	8.45
CASE1: 05-20	17,629	8.19	9.70	10.01	6.33	8.56
CASE2: no_zero	12,113	8.20	9.65	**9.84**	6.33	8.50

CASE0: all_time		CASE2: 05-20		CASE1: No_zero
8.45	<	8.50	<	8.56

데이터 전처리

결측치

기상데이터 결측치 처리방법

- Hari etal.(2006, pp.46-73)에 가이드라인 제시
- 보통 결측율이 50% 이상되면 쓰지 않음

결측치 비율	처리방법
10% 미만	삭제 또는 대치
10-20%	Hot dect, Regression, model based imputation
20% 이상	Regression, model based imputation

결측치 처리방법

```
energy.isnull().sum(), energy.shape
```

```
(time                 0
 dangjin_floating    24
 dangjin_warehouse   48
 dangjin              0
 ulsan                0
 dtype: int64, (25632, 5))
```

```
dangjin_obs.isnull().sum(), dangjin_obs.shape
```

```
(지점              0
 지점명             0
 일시              0
 기온(° C)         37
 풍속(m/s)        36
 풍향(16방위)       36
 습도(%)          35
 전운량(10분위)    3970
 dtype: int64, (25626, 8))
```

```
ulsan_obs.isnull().sum(), ulsan_obs.shape
```

```
(지점              0
 지점명             0
 일시              0
 기온(° C)         4
 풍속(m/s)        1
 풍향(16방위)       1
 습도(%)          1
 전운량(10분위)    825
 dtype: int64, (25632, 8))
```

```
dangjin_fcst.isnull().sum(), dangjin_fcst.shape
```

```
(Forecast_time    0
 Temperature      0
 Humidity         0
 WindSpeed        0
 WindDirection    0
 Cloud            0
 dtype: int64, (26304, 6))
```

```
ulsan_fcst.isnull().sum(), ulsan_fcst.shape
```

```
(Forecast_time    0
 Temperature      0
 Humidity         0
 WindSpeed        0
 WindDirection    0
 Cloud            0
 dtype: int64, (26304, 6))
```

데이터의 결측치 비율이 10% 미만이므로 삭제

모델선정

모델선택

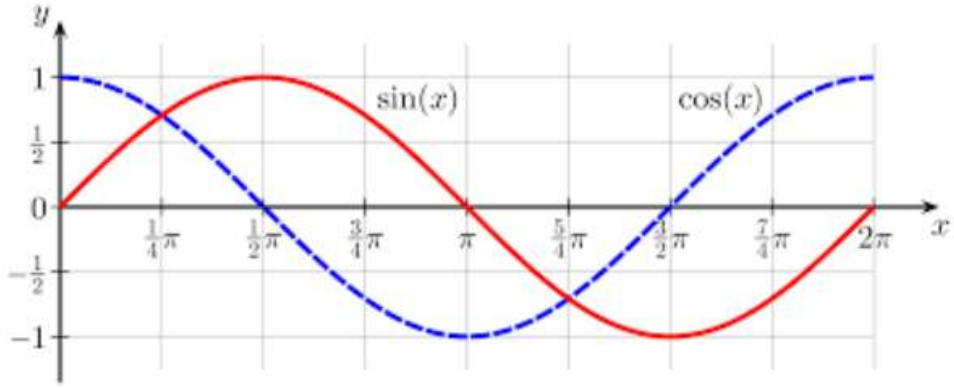
	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
rf	Random Forest Regressor	35.92	6177.28	78.55	0.87	0.81	1.27	0.76
et	Extra Trees Regressor	33.89	6184.65	78.58	0.87	0.72	1.11	0.72
lightgbm	**Light Gradient Boosting Machine**	**42.12**	**6296.44**	**79.32**	**0.86**	**1.66**	**1.56**	**0.07**
gbr	Gradient Boosting Regressor	53.23	8072.39	89.82	0.82	2.04	1.97	0.29
lr	Linear Regression	71.59	10473.88	102.33	0.77	2.74	2.91	0.25
ridge	Ridge Regression	71.60	10473.89	102.33	0.77	2.74	2.91	0.01
br	Bayesian Ridge	71.60	10473.89	102.33	0.77	2.74	2.91	0.09
lasso	Lasso Regression	75.52	11140.02	105.54	0.76	2.78	2.82	0.20
dt	Decision Tree Regressor	44.91	11700.05	108.13	0.74	0.93	1.29	0.02
huber	Huber Regressor	93.52	17609.92	132.60	0.62	2.94	3.37	0.93
omp	Orthogonal Matching Pursuit	107.07	21404.41	146.28	0.53	3.13	3.65	0.01
en	Elastic Net	122.64	26633.22	163.17	0.42	3.29	4.16	0.01
knn	K Neighbors Regressor	104.12	27191.13	164.85	0.41	2.69	3.65	0.07
ada	AdaBoost Regressor	162.50	30851.23	175.58	0.33	3.96	5.92	0.34

Feature Engineering

시간

Modeling with the Sine, Cosine Function

- Day of year
- Wind Direction




```
concat_df['WindDirection'] =np.cos(np.pi*concat_df['WindDirection']/180.0)
concat_df['day_of_year'] =np.cos(np.pi*(concat_df['day_of_year']/365.0)-0.5)
```

성능비교 – 시간

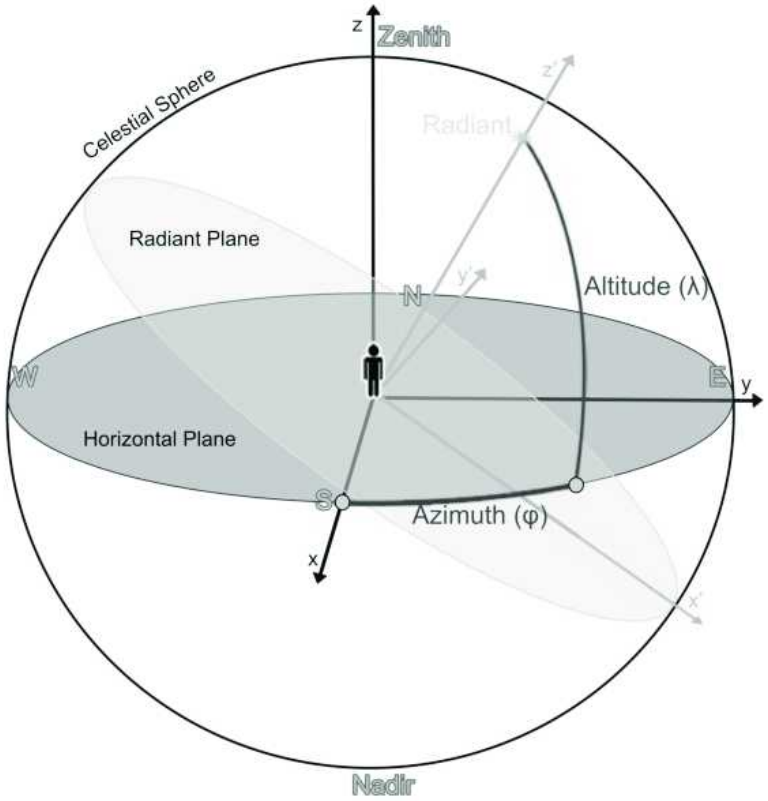
	Data length	floating	warehouse	dangjin	ulsan	Average Score
No-time	25,632	**8.20**	**9.65**	**9.84**	6.33	8.51
Non-cycle-time	25,632	8.45	10.36	10.75	5.82	8.85
Cycle-time	25,632	8.22	10.17	10.67	**5.72**	8.70

Cycle-time (Ulsan)		Non-Cycle-time (Ulsan)		No-time
5.72	<	5.82	<	6.33

Feature Engineering

태양고도

태양고도각(Solar Altitude Angle)



the cosine of the solar zenith angle (θ_0) is given by:

$$\cos\theta_0 = \sin\delta\sin\lambda + \cos\delta\cos\lambda\cos\tau \quad (1)$$

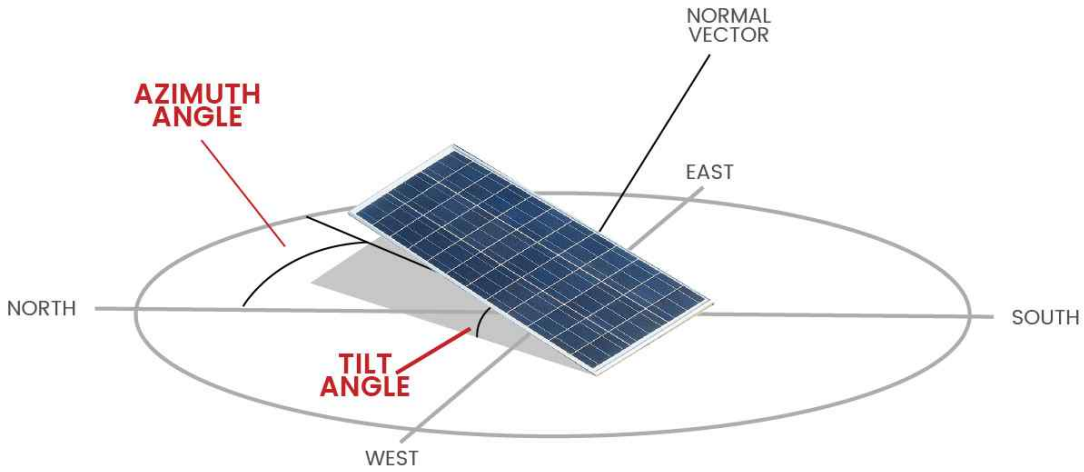
which depends on time and space via: (All units in **Radian**)

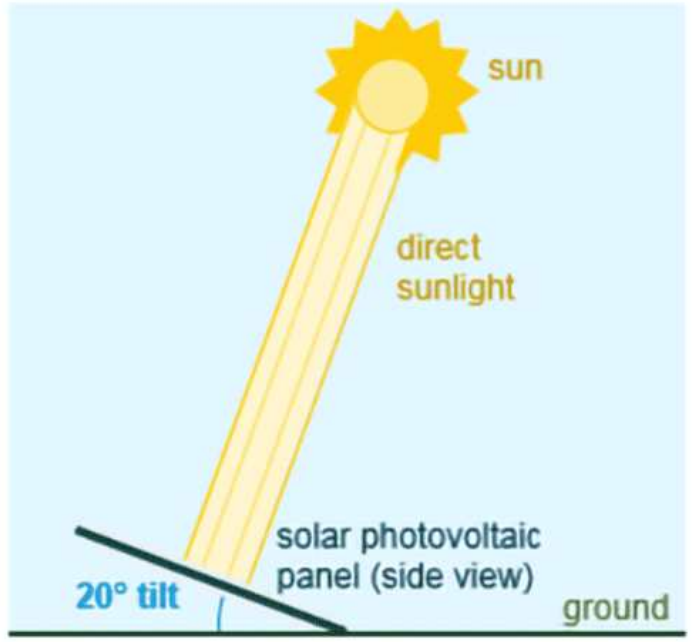
$$\delta = \text{declination angle} = \frac{23.45\pi}{180}\cos\left[\frac{2\pi}{365}(172 - DOY)\right] \quad (2)$$

$$\lambda = \text{latitude} \quad (3)$$

$$\tau = \text{hour angle} = 2\pi\frac{T_h - 12}{24} \quad (4)$$

태양고도각(Solar Altitude Angle)






```
def sun(N,la,h):
    d = -23.44* np.cos(np.radians(360/(365*(N+10))))
    s = (np.sin(np.radians(la))*np.sin(np.radians(d)) +
         np.cos(np.radians(la))*np.cos(np.radians(d))*np.cos(np.radians(h*15-180)))
    return 90- np.degrees(np.arcsin(s))
```

```
dangjin_fcst['day_of_year'] = pd.to_datetime(dangjin_fcst['Forecast_time'].str[:10], format='%Y-%m-%d', errors='raise')
dangjin_fcst['day_of_year'] = dangjin_fcst['day_of_year'].dt.dayofyear
dangjin_fcst['time'] =dangjin_fcst['Forecast_time'].str[11:13].astype(int)
dangjin_fcst['sun'] = sun(dangjin_fcst['day_of_year'],37.050753,dangjin_fcst['time']) -30
```

성능비교 – 태양고도

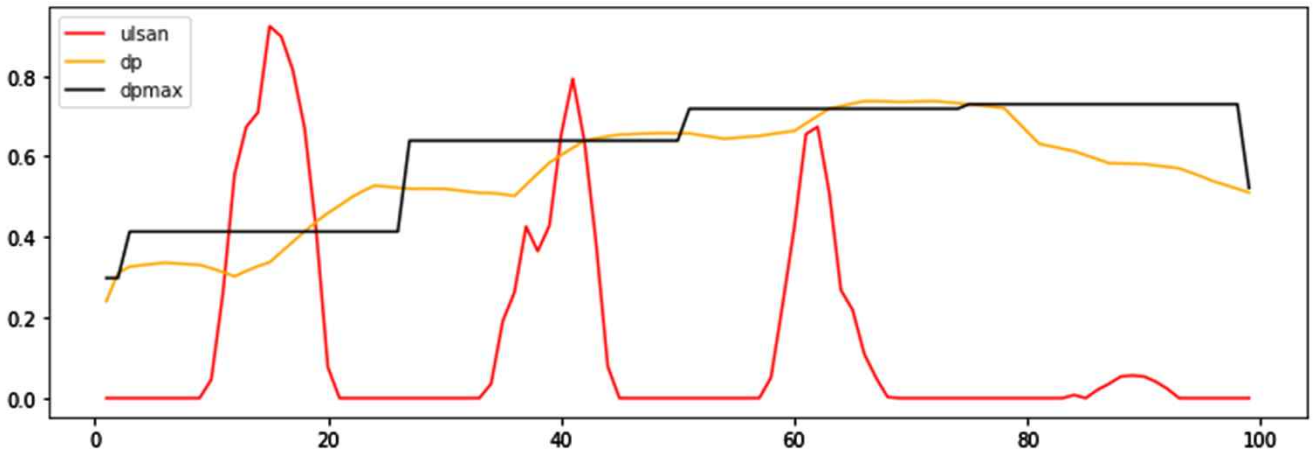
	Data length	floating	warehouse	dangjin	ulsan	Average Score
baseline	25,632	8.20	**9.65**	9.84	**6.33**	8.51
Add sun	25,632	**7.93**	9.66	**9.47**	6.37	8.36

Add sun		baseline
8.36	<	8.51

Feature Engineering

이슬점

이슬점



공기중의 수증기양이 줄어든 만큼 응결되어 안개, 전운량과 관련있을 것이다 추론

성능비교 - 이슬점

	floating	warehouse	dangjin	ulsan	Average Score
baseline	8.20	9.64	**9.84**	6.33	8.50
DPMAX-DP	**8.18**	**9.49**	9.90	**6.29**	**8.47**

DPMAX-DP		baseline
8.47	<	8.50

Feature Engineering

Delete Feature

Delete Features

- ~~Year~~, month, ~~day~~, **hour**
- Temperature, **Humidity**
- Wind Speed , Wind Direction
- ~~Cloud~~

성능비교

	Data length	floating	warehouse	dangjin	ulsan	Average Score
baseline	25,632	8.20	9.65	**9.84**	6.33	8.51
DPMAX-DP	25,632	8.18	**9.48**	9.89	6.29	8.46
Add sun,sin	25,632	7.93	9.66	9.47	6.37	8.36
ADD sun, sin DEL features	25,632	**7.91**	10.30	10.60	**5.90**	8.68

Add sun,sin		DPMAX-DP		baseline		ADD sun, sin DEL features
8.36	<	8.46	<	8.51	<	8.68

외부데이터

미세먼지

미세먼지

미세먼지 농도 등급 4단계				
좋음	0	-	30	㎍/㎥.
보통	31	-	80	㎍/㎥.
나쁨	81	-	150	㎍/㎥.
매우 나쁨	150		이상	㎍/㎥

지점	지점명	일시	미세먼지*
152	울산	2018-01-01 0:00	31
152	울산	2018-01-01 1:00	39
152	울산	2018-01-01 2:00	29
152	울산	2018-01-01 3:00	30
152	울산	2018-01-01 4:00	28
152	울산	2018-01-01 5:00	21
152	울산	2018-01-01 6:00	31

** 1시간평균 미세먼지농도(㎍/㎥)*

성능비교

	Data length	Floating	warehouse	dangjin	ulsan	Average Score
baseline	25,632	8.20	**9.65**	**9.84**	6.33	8.51
Dust-17h	25,632	7.62	10.67	11.31	5.41	8.75
Dust-23h	25,632	**7.61**	10.68	11.31	**5.38**	8.74

baseline		Dust-23h		Dust-17h
8.45	<	8.74	≤	8.75

외부데이터

과거 발전량

과거 발전량(공공데이터포털)

데이터 상세정보

분류체계	산업·통상·중소기업 - 에너지및 자원개발	제공기관	한국동서발전(주)
관리부서명	ICT총괄팀	관리부서 전화번호	070-5000-1486
업데이트 주기	연간	차기 등록 예정일	2020-12-19
매체유형	텍스트	전체 행 수	9788
확장자	CSV	다운로드(바로가기) 횟수	188
등록일	2019-12-19	수정일	2019-12-19
데이터 다운로드	XML CSV		
비용부과유무	무료	비용부과기준 및 단위	
이용허락범위	이용허락범위 제한 없음		

닫기

태양광명	용량(MW)	시간	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
당진태양광	1	2015-01-01	0	0	0	0	0	0	0	0	8	227	263	356	333	225	199	218	44	0	0	0	0	0	0	0
당진태양광	1	2015-01-02	0	0	0	0	0	0	0	0	2	59	208	378	513	560	513	309	58	0	0	0	0	0	0	0
당진태양광	1	2015-01-03	0	0	0	0	0	0	0	0	12	99	305	460	497	320	244	220	32	0	0	0	0	0	0	0
당진태양광	1	2015-01-04	0	0	0	0	0	0	0	0	5	81	293	296	280	421	354	193	41	0	0	0	0	0	0	0
당진태양광	1	2015-01-05	0	0	0	0	0	0	0	0	20	145	264	258	229	257	188	133	8	0	0	0	0	0	0	0
당진태양광	1	2015-01-06	0	0	0	0	0	0	0	0	60	253	404	523	590	609	564	365	64	1	0	0	0	0	0	0

과거 발전량(공공데이터포털)

```python
def date_gen(month_data, y, name):
    output=pd.DataFrame(columns=['Forecast time','forecast','value'])
    for i, df in enumerate(month_data):
        month = i+1
        year = y
        if month >= 13:
            month = month - 12
            year += 1
        if month <= 11 and year==2016 and name=='Cloud': # 2016년 1월 하늘형태 데이터 누락
            month = month + 1
        date = f'{year}-{month}-' + df[' format: day'].str.split(' ').str[-1] + ' ' + (df['hour'].astype(int)//100).astype(str) +

        # 시간단위 UTC => GMT
        date = pd.to_datetime(date) + pd.DateOffset(hours=9)
        data_fcst = pd.DataFrame(columns=['Forecast time','forecast','value'])
        data_fcst['Forecast time'] = date
        data_fcst['forecast'] = df['forecast']
        data_fcst['value'] = df['value']
        output = pd.concat([output, data_fcst])
    return output

def del_month(df ,y, name=' '):    # 월 구분 행 추출
    month_rows = [-1]
    month_rows.extend(df[df['hour'].isna()].index)
    month_rows.append(df.shape[0]+1)
    month_data = []
    for i in range(len(month_rows)-1):
        month_data.append(df.loc[month_rows[i]+1:month_rows[i+1]-1])
    return date_gen(month_data, y, name)
```

```python
def preprocessing(path_list, y):
    # 데이터 불러오기
    data_year_temperature = pd.read_csv(path_list[0],names=[' format: day', 'hour', 'forecast','value'],header=0)
    data_year_humidity = pd.read_csv(path_list[1],names=[' format: day', 'hour', 'forecast','value'],header=0)
    data_year_windspeed = pd.read_csv(path_list[2],names=[' format: day', 'hour', 'forecast','value'],header=0)
    data_year_winddirection = pd.read_csv(path_list[3],names=[' format: day', 'hour', 'forecast','value'],header=0)
    data_year_cloud = pd.read_csv(path_list[4],names=[' format: day', 'hour', 'forecast','value'],header=0)

    if y==2014: #2015년 5월 21일 발표시간 23시 ~ 2015년 5월 31일 발표시간 23시 데이터 추가
        tmp1=data_year_temperature.iloc[0:25109,:]
        tmp2=data_year_temperature.iloc[25109::,:]
        tmp3=data_year_temperature.iloc[16411:17911,:]
        tmp4=pd.concat([tmp1,tmp3])
        data_year_temperature=pd.concat([tmp4,tmp2]).reset_index(drop=True)

    data_year_temperature=del_month(data_year_temperature, y)
    data_year_humidity=del_month(data_year_humidity, y)
    data_year_windspeed=del_month(data_year_windspeed, y)
    data_year_winddirection=del_month(data_year_winddirection, y)
    data_year_cloud=del_month(data_year_cloud, y, name='Cloud')

    # 데이터 결합
    data_year = pd.DataFrame(columns=['Forecast time'])
    data_year['Forecast time']=data_year_temperature['Forecast time']
    data_year['forecast']=data_year_temperature['forecast']
    df_data = pd.merge(data_year, data_year_temperature, on=['Forecast time', 'forecast'], how='outer')
    df_data = pd.merge(df_data, data_year_humidity, on=['Forecast time', 'forecast'], how='outer')
    df_data = pd.merge(df_data, data_year_windspeed, on=['Forecast time', 'forecast'], how='outer')
    df_data = pd.merge(df_data, data_year_winddirection, on=['Forecast time', 'forecast'], how='outer')
    df_data = pd.merge(df_data, data_year_cloud, on=['Forecast time', 'forecast'], how='outer')
    df_data.columns=['Forecast time', 'forecast', 'Temperature', 'Humidity', 'WindSpeed', 'WindDirection', 'Cloud']
    df_data=df_data[['Forecast time', 'forecast', 'Temperature','Humidity', 'WindSpeed', 'WindDirection', 'Cloud']]
    return df_data

dangjin_data_path = 'data/dangjin'
ulsan_data_path = 'data/ulsan'
csv_list = sorted(glob(dangjin_data_path+'/*/*.csv'))
csv_list2= sorted(glob(ulsan_data_path+'/*/*.csv'))
dangjin_fcst = pd.DataFrame(columns=['Forecast time', 'forecast', 'Temperature','Humidity', 'WindSpeed', 'WindDirection', 'Cloud']
ulsan_fcst = pd.DataFrame(columns=['Forecast time', 'forecast', 'Temperature', 'Humidity', 'WindSpeed', 'WindDirection', 'Cloud'])

for i in range(4):
    dangjin_fcst = pd.concat([dangjin_fcst, preprocessing(csv_list[i*5:i*5+5], 2015+i)])
    ulsan_fcst = pd.concat([ulsan_fcst, preprocessing(csv_list2[i*5:i*5+5], 2015+i)])
print(dangjin_fcst.info())
print(ulsan_fcst.info())
ulsan_fcst
```

성능비교 - 과거발전량

	floating	warehouse	dangjin	ulsan	Average Score
baseline	8.20	**9.64**	**9.84**	**6.33**	**8.50**
2015~2020	**8.05**	10.70	11.64	6.78	9.29

baseline		2015~2020
8.50	<	9.29

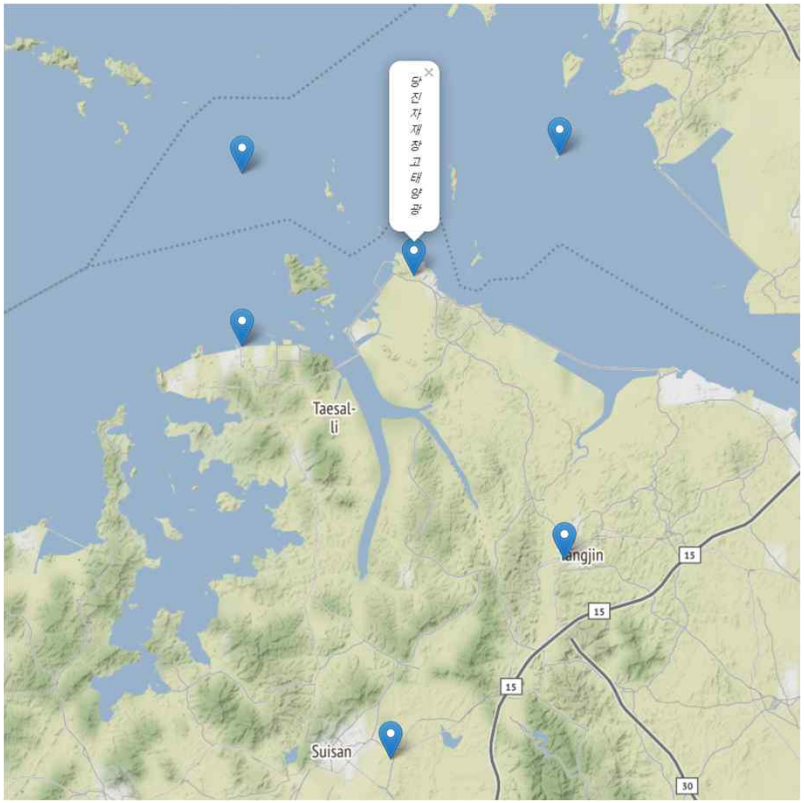
외부데이터

타지역 기상관측 데이터
데이터 유출 발생

기상관측

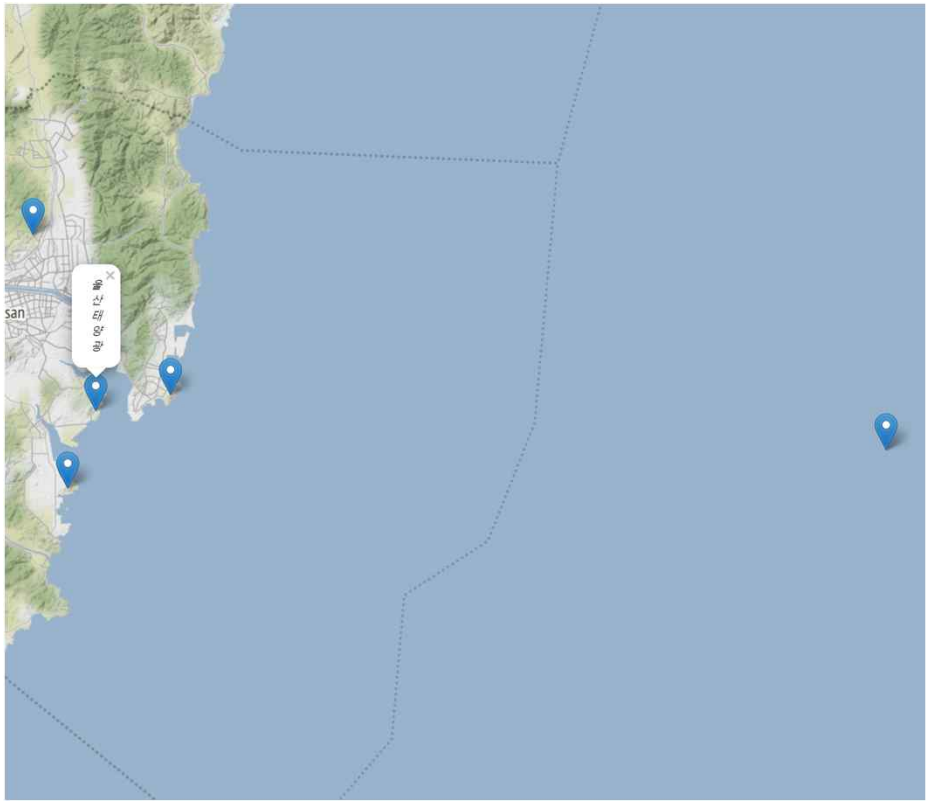
종관기상관측(ASOS)	방재기상관측(AWS)	해양기상부이
전국 102곳에서 종관(종합적으로 동시에 관측) 기상 정보를 제공	기상현상에 따른 자연재해를 막기 위해 502곳에서 실시	해수면에서 해양기상현상을 다양한 기상장비로 측정

당진태양광



	Id	Latitude	Longitude
1	서산_ASOS	36.776557	126.493908
2	당진_AWS	36.889360	126.617390
3	대산_AWS	37.010610	126.388080
4	풍도_AWS	37.109027	126.388128
5	도리도_AWS	37.119058	126.614066

울산태양광

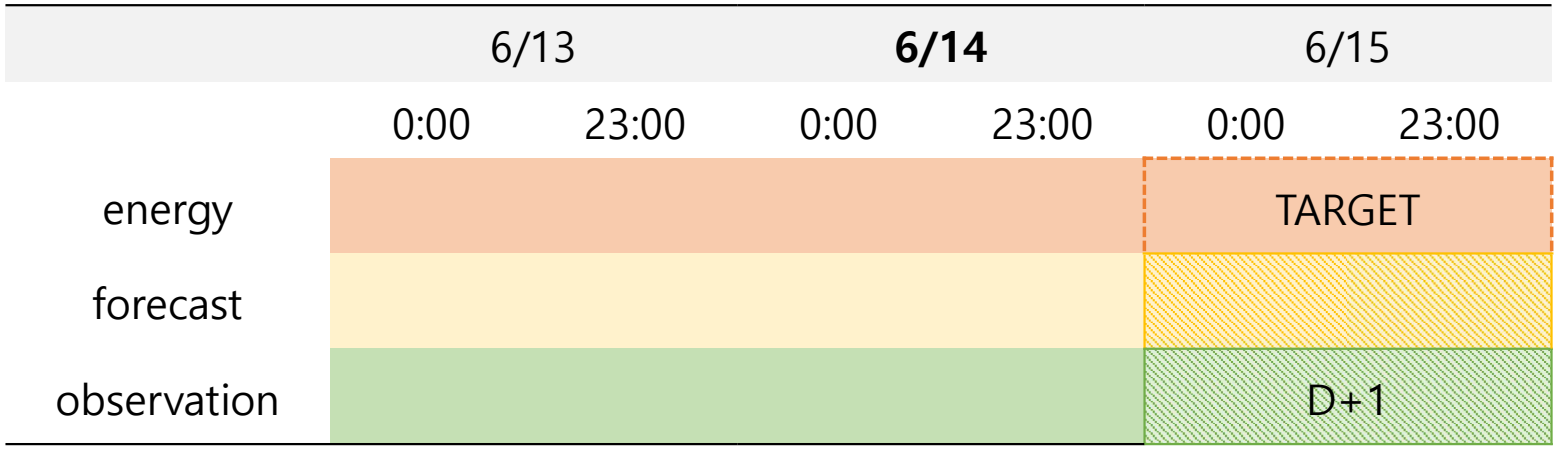


	Id	Latitude	Longitude
1	울산_ASOS	35.582370	129.334690
2	울기_AWS	35.486780	129.435310
3	장생포_AWS	35.504725	129.385066
4	온산_AWS	35.431030	129.360110
5	울산_해양부이	35.453739	129.961693

성능비교

당진	도리도	대산	이덕서	울산	온산	풍도	울기	울산 해양	floating	warehouse	dangjin	ulsan	Average score
									8.20	**9.65**	**9.84**	**6.33**	**8.503**
O									6.60	8.99	8.90		7.703
O	O								6.51	8.76	8.76		7.589
O	O	O							6.69	9.04	8.61		7.666
O	O		O						**6.42**	**8.78**	8.66		7.545
O	O	O	O						6.62	9.02	**8.56**		7.633
				O								4.33	7.023
				O	O							4.23	6.998
				O	O	O						4.23	6.997
				O	O		O					4.29	7.014
				O	O			O				4.26	7.004
				O	O	O	O	O				**4.14**	**6.975**

Data Leckage



예측하려는 발전일을 기준으로 하루 전 데이터만 모델학습에 사용해야한다.

결론

결론

- 기상예보 데이터의 정확도가 올라갈수록 발전량을 예측할 가능성이 높아진다.
- 기상데이터가 가지고 있는 복잡성 때문에 새로운 feature를 추가하는데 어려움이 있다.
- 시계열 데이터 분석에 많이 활용되는 LSTM이나 많은 분야에서 좋은 성적을 내는 Transformer 모델을 활용한다.