



## [돌체라떼] 멋사 - 프로젝트 2차 최종

### [한국가스공사] 가스공급량 수요예측 모델개발

- Daycon , 한국가스공사 주최 대회.
- 대회 설명 링크 : <https://dacon.io/competitions/official/235830/overview/description>

#### 프로젝트 개요

대회 목표 : 2013년도 ~ 18년도 까지의 데이터를 바탕으로 19년도 1월 ~ 3월의 가스공급량 예측

- 팀 이름 : 돌체라떼
- 팀장 : 박지용
  - 역할
    - 프로젝트 전 총괄 및 검토
    - 원 데이터 + 외부데이터 병합.
    - 외부데이터 전 처리 및 특성 구하기 - 발전량 데이터
      - 외부 데이터 예측 모델 구현
    - 단일 알고리즘 튜닝 및 성능 확인
- 팀원 : 최두호
  - 역할
    - 프로젝트 수행
    - 외부데이터 전 처리 및 특성 구하기 - 기온 데이터
    - Stacking 알고리즘 구현 및 성능 확인

#### 프로젝트 수행 절차 및 방법

##### ~ 중간 발표

- 각 팀원 당 한개의 외부 데이터를 전처리 해본다.

- Baseline에서 제공된 lgbm 모델로 외부데이터의 성능을 일차적으로 확인해본다.

## ~ 최종발표

- Feature 선택을 위해 Heatmap 시각화 작업
- PolynomialFeature과 스케일러, 파라미터 튜닝을 적용해본 단일 알고리즘의 성능 확인.
- CV Stacking을 활용한 모델 개선해보기.

## 목차

1. 외부데이터를 얼마나 잘 정제해서 가져오는가? - (~ 중간 발표)
  - a. 외부데이터 : 기온, 화력발전량
2. 상관관계수
3. 사용한 알고리즘에 따른 결과
  - a. 1개의 알고리즘만 사용할때 - LGBM , xgboost (+ Scaler)
  - b. CV Stacking ensemble
4. 추후 개선사항 및 자체 평가
5. 시행착오

## 1. 외부데이터를 얼마나 잘 정제해서 가져오는가?

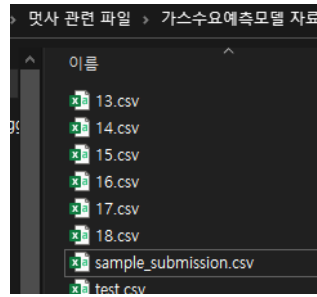
### ▼ 1. 기온 데이터

기상청 기상자료 개방포털에서 1년단위로 된 csv파일을 총 6개(2013년도~2018년도) 다운로드.

The screenshot shows the 'ASOS' (Automated Surface Observing System) data page on the Korea Meteorological Administration's Open Data Portal. The page includes a sidebar with navigation links and a main content area with a table of data details.

자료형태	분, 시간, 일, 월, 연	제공기간	1904년~(지정별, 요소별 다름)
제공지점	102개	제공요소	기온, 강수, 바람, 기압, 습도, 일사, 일조, 눈, 구름, 시정, 지면상태, 지면·초상온도, 일기현상, 증발량, 현상번호
유의사항	1회 조회 가능 최대 기간: 분 1일, 시간 1년, 일 10년, 월·연 제한 없음(장기간 자료는 '파일셋 조회' 메뉴 이용) 시간/분 자료에 대해 관측값의 정상 여부를 판단하는 품질검사 플래그(QC FLAG) 정보 제공 * 제공 요소: 기온, 습도, 기압, 지면온도, 풍향, 풍속, 일조 / 플래그 종류(의미): 이정상, 1(오류), 9(결측)		
비고	10분 또는 1시간 최대강수시각은 최대강수가 나타난 시작 시간으로, (-) 표기가 있는 경우 전날을 뜻함		

데이터프레임으로 하나씩 불러온뒤, 하나의 데이터 프레임으로 병합해주었음(이슈)



(숫자파일)

기온DF를 기본DF과 병합한뒤, 해당 데이터 프레임으로 기온예측모델생성

```
1 gas=gas.join(tmp) # gas , tmp 합치기
```

```
1 gas.fillna(0,inplace=True)
2 gas.tail()
```

	연월일	시간	구분	공급량	지점	지점명	기온(° C)	감수량(mm)	풍속(m/s)	습도(%)	일조(hr)	적설(cm)	지면온도(° C)
368083	2018-12-31	20	H	681.033	108	서울	-3.1	0.0	2.1	43.0	0.0	0.0	-2.2
368084	2018-12-31	21	H	669.961	108	서울	-3.7	0.0	0.8	39.0	0.0	0.0	-3.0
368085	2018-12-31	22	H	657.941	108	서울	-4.6	0.0	1.1	44.0	0.0	0.0	-4.1
368086	2018-12-31	23	H	610.953	108	서울	-5.4	0.0	1.3	46.0	0.0	0.0	-5.0
368087	2018-12-31	24	H	560.896	108	서울	-5.2	0.0	1.6	47.0	0.0	0.0	-5.1

기온예측모델을 이용해서 test데이터프레임에 붙여주었음

```
2
3 test.head()
```

	일자	시간	구분	Year	Month	Day	weekday	기온(° C)
0	2019-01-01	1	0	2019	1	1	1	-3.254073
1	2019-01-01	2	0	2019	1	1	1	-3.608198
2	2019-01-01	3	0	2019	1	1	1	-3.935810
3	2019-01-01	4	0	2019	1	1	1	-4.113031
4	2019-01-01	5	0	2019	1	1	1	-4.464113

이제, 기온의 특성추가를 완료하였기때문에 구분, 년, 월, 일, 시간, **기온** 데이터를 사용하여  
공급량 예측 모델 생성 후, 제출.

기본데이터만을 가지고 예측을 했을 때와 비교해서,

602784	electric_baseline.csv 기온 특성 추가 edit	2021-11-04 07:49:53	0.1307686256	●
602514	electric_baseline.csv 발전량 특성 추가 edit	2021-11-03 16:07:07	0.1544943825	○
595317	baseline.csv test_submission edit	2021-10-15 13:49:27	0.1617253368	○

0.1617253368 → 0.1307686256

0.1307686256

0.031점 정도 차이나는 유의미한 성과를 냈음.

## ▼ 2. 화력 발전량 데이터

### 한국전력거래소 데이터:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	거래일	1시	2시	3시	4시	5시	6시	7시	8시	9시	10시	11시	12시	13시	14시	15시	16시
2	2017-01-01	55122	52867.08	51112.86	50228.44	50270.22	50799.94	50977.49	50159.77	49592.17	49722.52	49954	49637.59	49429.44	49185.34	49326.41	49371.41
3	2017-01-02	53229.03	52920.74	52521.67	52558.05	53271.3	54905.06	56645.02	58570.71	64429.09	67943.26	68466.82	67668.77	63281.21	65109.58	65638.65	65560.16
4	2017-01-03	58347.55	57012.44	56541.88	56513.32	57373.7	58908.7	61596.1	64252.67	68662.06	71201.37	70521.89	69079.54	63811.02	66080.48	66752.05	66488.25
5	2017-01-04	60016.32	58366.49	58643.35	58715.11	58916.99	59710.07	61895.51	64259.83	69180.34	71630.16	71161.96	69726.92	64560.58	66751.45	67257.06	66809.36
6	2017-01-05	59751.78	57865.01	58001.78	57893.72	58456.74	59722.97	62058.1	64262.8	69080.15	71715.62	71807.74	70984.73	66580.81	69163.49	69860.24	69280.43
7	2017-01-06	59385.22	57436.47	57132.62	56848.14	57575.21	58817.56	61444.64	63599.19	67805.22	69729.7	68865.57	67563.64	62785.76	64903.78	65292.68	64896.55
8	2017-01-07	59325.36	56590.18	55137.04	55494.96	56221.11	57226.4	58335.87	58768.71	60453.49	61067.76	61312.54	60400.47	57329.6	57851.59	57833.93	57249.27
9	2017-01-08	55301.45	52764.44	51774.51	51918.51	52546.71	52829.68	52952.63	52647.52	52735.18	52362.8	52330.05	51785.4	51776.27	51785.17	51856.74	51674.16
10	2017-01-09	52537.46	51703.2	51249.84	51599.06	52469.27	54302.69	57056.2	59958.21	67177.29	70436.9	70619.36	69422.13	64533.13	66384.16	67026.32	66842.24
11	2017-01-10	60341.31	59209.59	58882.72	58723.11	59284.12	60808.54	63114.25	64579.89	69925.68	72206.17	71648.68	70362.27	65934.91	68176.07	69055.23	68919.98
12	2017-01-11	61655.7	59820.11	59892.45	60464.83	61250.09	62805.91	65118.37	67506.54	71870.57	74259.23	73753.86	72295.92	66913.32	69373.24	70370.54	70197.07
13	2017-01-12	61659.18	59759.68	60013.32	60078.48	60829.32	62265.36	64085.09	66160.81	70943.02	73633.75	73611.96	72306.93	67149.11	69682.83	70500.99	70232.24
14	2017-01-13	62544.12	60795.58	60651.93	60382.93	60895.4	62307.78	64885.09	67706.09	72171.26	74771.5	75220.32	74594.97	69622.64	71951.89	72575.03	72210.05
15	2017-01-14	63741.89	61373.33	60888.47	60757.52	61016.58	61489.27	62522.74	63009.79	63897.17	64508.78	65345.96	64927.19	62330.24	62349.32	61990.43	61647.91
16	2017-01-15	61503.2	59091.74	57548.01	57205.4	57386.1	57531.82	57466.99	57352.54	57255.47	56694.37	57043.13	56664.89	56448.21	55873.28	55068.08	55130.87
17	2017-01-16	58174.59	56755.01	56874.17	57094.89	57654.6	59337.83	62213.24	65780.78	72443.47	75745.72	75909.69	74771.27	69408.52	71406.79	72059.15	71649.77
18	2017-01-17	63924.97	62161.21	61960.82	61904.85	62265.25	63676	65901.73	68400.13	73788.86	76680.7	76593.5	75143.61	69570.16	71835.76	72505.59	72083.41
19	2017-01-18	63326.31	62067.83	61679.16	61334.56	61868.3	62995.15	64217.72	66505.19	72357.35	75139.39	74803.35	73851.55	68651.55	71335.72	72143.17	71821.48
20	2017-01-19	63193.46	61409.27	61433.98	61021.27	61299.68	62614.51	65293.89	67992.2	72674.11	75258.15	74866.14	73312.81	67832.39	69789.8	70102.16	69810.04
21	2017-01-20	62890.95	61467.58	60963.19	60687.8	61286.94	62620.85	65077.37	67412.16	71619.45	75464.08	76765.05	76357.4	71836.7	74079.57	74270.14	73509.27
22	2017-01-21	64532.85	61914.59	61072.2	61429.4	61788.91	62297.78	63341.5	63652.34	65138.97	66050.05	66965.75	66449.58	63090.76	63736.63	63696.09	63137.51
23	2017-01-22	60738.14	58186.85	56724.97	56008.96	55928.33	56265.29	56504.97	56410.24	57293.67	57417.49	57886.21	57928.82	57808.65	58069.62	57819.46	57586.3
24	2017-01-23	59349.53	58197.15	58427.39	58517.25	59377.86	61261.26	63863.42	66329.05	73699.35	77597.98	78736.66	78067.33	73158.06	75326.97	75759.96	75300.62
25	2017-01-24	66139.84	64776.3	64743.02	64328.04	64704.7	66037.77	67880.54	70376.79	75248.17	78047.4	77879.97	76674.08	71367.8	73704.46	74191.1	73619.95
26	2017-01-25	65609.3	63019.75	61991.32	61996.43	62651.24	63860.68	66380.98	69087.26	74901.85	76975.23	76751.31	75374.56	69949.04	71893.82	72094.96	71506.23
27	2017-01-26	65011.9	62453.28	62085.93	62232.88	62744.17	63817.17	65700.72	68216.98	72440.7	74106.57	73550.19	71820.67	66594.79	67301.49	66284.02	64291.11

### 한국 전력공사 연별 전력 통계:

통계표명: 에너지원별 발전량 현황		단위: GWh, %							
		계	원자력	석탄	가스	신재생	유류	양수	기타
2011	발전량	496,893	154,723	202,856	112,646	12,190	11,245	3,233	-
	비중	100.0	31.1	40.8	22.7	2.5	2.3	0.7	-
2012	발전량	509,574	150,327	202,191	125,285	12,587	15,501	3,683	-
	비중	100.0	29.5	39.7	24.6	2.5	3.0	0.7	-
2013	발전량	517,148	138,784	204,196	139,783	14,449	15,832	4,105	-
	비중	100.0	26.8	39.5	27.0	2.8	3.1	0.8	-
2014	발전량	521,971	156,407	207,214	127,472	17,447	8,364	5,068	-
	비중	100.0	30.0	39.7	24.4	3.3	1.6	1.0	-
2015	발전량	528,091	164,762	211,393	118,695	19,464	10,127	3,650	-
	비중	100.0	31.2	40.0	22.5	3.7	1.9	0.7	-
2016	발전량	540,441	161,995	213,803	121,018	25,836	14,001	3,787	-
	비중	100.0	30.0	39.6	22.4	4.8	2.6	0.7	-
2017	발전량	553,530	148,427	238,799	126,039	30,817	5,263	4,186	-
	비중	100.0	26.8	43.1	22.8	5.6	1.0	0.8	-
2018	발전량	570,647	133,505	238,967	152,924	35,598	5,740	3,911	0
	비중	100.0	23.4	41.9	26.8	6.2	1.0	0.7	0.0
2019	발전량	563,040	145,910	227,384	144,355	36,392	3,292	3,458	2,249
	비중	100.0	25.9	40.4	25.6	6.5	0.6	0.6	0.4
2020	발전량	552,162	160,184	196,333	145,911	36,527	2,255	3,271	7,681
	비중	100.0	29.0	35.6	26.4	6.6	0.4	0.6	1.4
출처: 한국전력공사 월별 전력통계속보, 연도별 한국전력통계									

- 공공데이터에서 17~18년 화력 발전소 시간데이터를 베이스로 데이터 구상을 시작했음.
- 13년부터 15년 데이터는 에너지원별 발전량 데이터를 참고하여 총 생산량 비율로 17년, 18년 기준으로 나눈 후 두개를 추합 후, 반으로 나눔.

- 16년도 윤년 데이터는 예측하지 않고 제거하였음. → 이후, 새로운 특성 데이터와 합칠때 오류 발생 우려로 해결.

## 16년 2월 29일 데이터를 예측하기

	date
0	2013-01-01 00:00:00
1	2013-01-01 01:00:00
2	2013-01-01 02:00:00
3	2013-01-01 03:00:00
4	2013-01-01 04:00:00
...	...
52580	2018-12-31 20:00:00
52581	2018-12-31 21:00:00
52582	2018-12-31 22:00:00
52583	2018-12-31 23:00:00
52584	2019-01-01 00:00:00

- pd.date\_range를 이용해 시간데이터를 임의로 생성.
- 16년 2월 29일에 해당되는 구간만 잘라내기.

```
In [43]: t_df.loc[(t_df["일자"] == '2016-02-29')].index
Out[43]: Int64Index([27696, 27697, 27698, 27699, 27700, 27701, 27702, 27703, 27704,
                    27705, 27706, 27707, 27708, 27709, 27710, 27711, 27712, 27713,
                    27714, 27715, 27716, 27717, 27718, 27719],
                    dtype='int64')
```

- 기존 발전량과 합치고, 잘라낸 구간 합치고. sort\_index()로 다시 구간 맞추기.

	date	일자	시간
27697	2016-02-29 01:00:00	2016-02-29	1
27698	2016-02-29 02:00:00	2016-02-29	2
27699	2016-02-29 03:00:00	2016-02-29	3
27700	2016-02-29 04:00:00	2016-02-29	4
27701	2016-02-29 05:00:00	2016-02-29	5
27702	2016-02-29 06:00:00	2016-02-29	6
27703	2016-02-29 07:00:00	2016-02-29	7
27704	2016-02-29 08:00:00	2016-02-29	8
27705	2016-02-29 09:00:00	2016-02-29	9
27706	2016-02-29 10:00:00	2016-02-29	10
27707	2016-02-29 11:00:00	2016-02-29	11
27708	2016-02-29 12:00:00	2016-02-29	12
27709	2016-02-29 13:00:00	2016-02-29	13
27710	2016-02-29 14:00:00	2016-02-29	14
27711	2016-02-29 15:00:00	2016-02-29	15
27712	2016-02-29 16:00:00	2016-02-29	16
27713	2016-02-29 17:00:00	2016-02-29	17
27714	2016-02-29 18:00:00	2016-02-29	18
27715	2016-02-29 19:00:00	2016-02-29	19
27716	2016-02-29 20:00:00	2016-02-29	20
27717	2016-02-29 21:00:00	2016-02-29	21
27718	2016-02-29 22:00:00	2016-02-29	22
27719	2016-02-29 23:00:00	2016-02-29	23
27720	2016-03-01 00:00:00	2016-03-01	24

- interpolate(method="time")을 사용하여, 시간별로 결측치 보간
- 유의미한 결과를 가져오는 가에, baseline과 동일하게 lgbm을 사용해서 발전량의 예측량을 예측하고, 공급량 예측 모델을 생성
- 베이스라인 점수와 비교했을때

---

사용한 CSV:


[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/e380cd3f-e1dc-4df9-8a2b-bf2c0691279a/energy\\_ratio.xls](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/e380cd3f-e1dc-4df9-8a2b-bf2c0691279a/energy_ratio.xls)

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/f1528e4e-963e-49a8-87c6-e7a577154c04/한국전력거래소\\_시간별\\_발전량\\_20201231.csv](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/f1528e4e-963e-49a8-87c6-e7a577154c04/한국전력거래소_시간별_발전량_20201231.csv)

- 제출한 내역

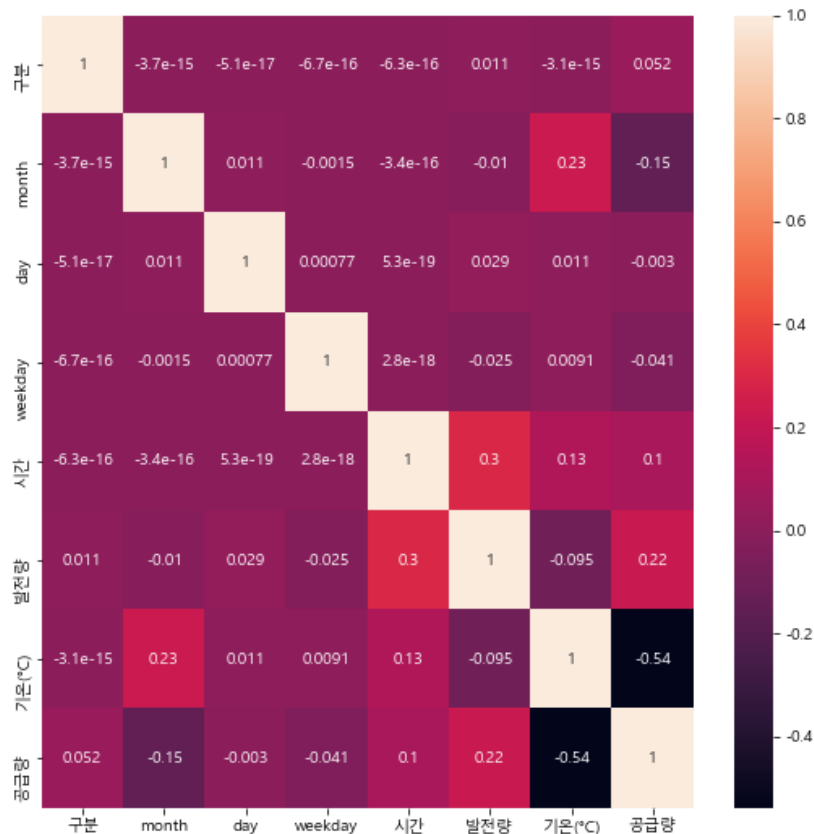
602860	<b>elect_temp_first.csv</b> 발전량 + 온도 특징 + 파라미터 튜닝 (반복량 - Early Stop 1000 - 100) edit	2021-11-04 13:29:25	0.1348557792	○
602803	<b>electric_baseline_fixed.csv</b> 발전량 오류 해결 재제출 edit	2021-11-04 10:02:50	0.1597967984	○
602784	<b>electric_baseline.csv</b> 기온 특성 추가 edit	2021-11-04 07:49:53	0.1307686256	●
602514	<b>electric_baseline.csv</b> 발전량 특성 추가 edit	2021-11-03 16:07:07	0.1544943825	○
595317	<b>baseline.csv</b> test_submission edit	2021-10-15 13:49:27	0.1617253368	○

#	팀	팀 멤버	점수	제출수
85	돌체라떼		0.13076	7

## 2. 외부데이터와 가스 공급량 상관관계수:

### ▼ Seaborn Heatmap



- 외부 데이터 **기온이 0.54** , **발전량이 0.22** 로 유의미한 특징을 찾았다.
  - 1차적으로 만든 lgbm 모델에서 발전량과 기온을 특징으로 선택하여 사용했을 때, 기온만을 사용한 것보다 점수가 낮게 측정되었다.
- Feature Selection으로 상관계수가 낮은 day와 weekday를 삭제하고 모델링을 하였을때 오히려 더 점수가 떨어진 결과를 도출했다. - (적용하지 않을 예정.)

### 3. 사용한 알고리즘에 따른 결과

#### 3-A. 1개의 알고리즘만 적용

- LGBM - Baseline params:

```
params = {
  'objective': 'regression',
  'metric': 'mae',
  'seed': 42
}
```

- LGBM - 튜닝한 파라미터:

```
params = {'learning_rate': 0.01,
  'max_depth': 16,
  'objective': 'regression',
  'metric': 'mae',
  'is_training_metric': True,
  'num_leaves': 144,
```



```
'feature_fraction': 0.9,
'bagging_fraction': 0.7,
'bagging_freq': 5,
'seed': 42
}
```

- **Polynomial Feature (Degree = default) + StandardScaler - LGBM :**

- 특성 개수 : 35개

Mean squared error: 24349.70591286092  
R2 score: 0.9761128701805538

- **Polynomial Feature (Degree = 5) + StandardScaler - LGBM :**

- 특성 개수: 791개

Mean squared error: 25284.566580018487  
R2 score: 0.9751957692430968

- **StandardScaler - LGBM** 파라미터 튜닝했을때 :

Mean squared error: 24546.727737721496  
R2 score: 0.9759195912175738

- **MinMaxScaler - LGBM** 파라미터 튜닝했을때 :

Mean squared error: 33363.985009238204  
R2 score: 0.9672698370952896

- **Scaler 없는 LGBM :**

Mean squared error: 24798.451798518785  
R2 score: 0.9756726492076595

- **Scaler 없는 XGBoost :**

Mean squared error: 24836.00095305641  
R2 score: 0.9756358133817049

## 결론

- 1개의 알고리즘과 파라미터 튜닝으로는 마지막으로 제출한 모델과 큰 격차를 만들지 못한다.
- Scaler의 사용으로 크게 격차를 만들지는 못하였으나, StandardScaler가 의의가 있다.
- Stacking을 사용하여 차이를 본다.

### 3-B. CV Stacking ensemble

- 점수를 올리고자 k-fold 기반 스택킹 실행
- 스택킹에 사용한 모델
  - GradientBoosting
  - XGB (메타모델)
  - LGBM
  - RandomForest



스택킹은 각 모델마다 파라미터 튜닝을 하는게 좋지만, 시간관계상 하지 못했다.

- (1)원본 train데이터를 위 4개의 모델이 학습한다.
- (2)각 모델마다 test로 pred를 뽑아낸다.
- (3)이 pred로 메타모델이 다시 학습데이터로 사용한다.

### 결론

1. 메타모델을 xgboost 로 사용해서, 기존 기온데이터로 돌렸더니:

**제출점수가 0.13 (80등) → 0.11대로 성능이 향상되었다. (60등)**

2. 스택킹 후 생각보다 데이콘 제출점수가 높게 나오지 않았다.
  - a. 과대적합 되었다고 생각해서, 블랜드 기법을 사용했다.
    - i. 블랜딩 중 구글 Colab에서 모델생성할때 RAM이 부족해 다운되는 현상  
→ 블랜딩에서 랜덤포레스트 삭제
    - ii. 블랜딩 후 점수가 0.15대 로, 성능이 더 떨어졌다.  
→ 과대적합 되지 않았다고 판단

→ 하이퍼 파라미터 튜닝을 하면 성능이 더 향상 될 것으로 예상

```
n_folds = 5

def rmsle_cv(model):
    kf = KFold(n_folds, shuffle=True, random_state=42).get_n_splits(X_train.values)
    rmse= np.sqrt(-cross_val_score(model, X_train.values, y_train, scoring="neg_mean_squared_error", cv = kf))
    return(rmse)

gb = GradientBoostingRegressor(n_estimators=250, random_state=0)
xgboost = xgb.XGBRegressor(n_estimators=250, random_state=0, nthread = -1)
lightgb = lgb.LGBMRegressor(n_estimators=250, random_state=0)
rf = RandomForestRegressor(n_estimators=250, random_state=0)
ridge = RidgeCV(cv = 10)
```

```

ridge = make_pipeline(RobustScaler(),ridge)

stack_model = stack.fit(np.array(X_train), y_train)

xgboost_model = xgboost.fit(X_train, y_train)
lightgb_model = lightgb.fit(X_train, y_train)
rf_model = rf.fit(X_train, y_train)
ridge_model = ridge.fit(X_train, y_train)

def blended_predictions(X):
    return ((0.13 * ridge_model.predict(X)) + \
            (0.2 * xgboost_model.predict(X)) + \
            (0.17 * lightgb_model.predict(X)) + \
            (0.5 * stack_model.predict(np.array(X))))

pred = blended_predictions(X_test)
sub['공급량'] = np.exp1(pred)
sub.to_csv('electric_baseline.csv', index=False)

```

## 4. 추후 개선사항 및 자체 평가

- 원-핫 인코딩 - 구분(공급사)에 적용 (선형 모델을 사용하기 때문에).
- Stacking - 하이퍼파라미터 조정
- 

### 자체 평가

- 팀장 박지용
  - 단일 알고리즘의 성능 한계로 어떤 방향으로 나아갈지 고민일때 두호님께서 Stacking을 시도한다고 할 때 비교되는 두개의 군을 발표할 수 있는 좋은 결과를 예상했다.
  - 특성을 제한적으로 사용했고, 파라미터를 완벽하게 적용하지 못한 Stacking이어서 점수의 개선이 예상된다.
- 부팀장 최두호
  -

## 5. 시행착오 :

- 중간 평가 전 :
  - 윤년이 끼있는 2016/2/29일 데이터 처리
  - 데이터 결측치 처리
- 최종 발표 전 :
  - 데이터 전처리에서 외부데이터 처리를 공급사와 같이 묶어 7번 반복된 데이터를 기반으로 예측치를 처리 하였다. → 논리적으로 하루의 기온과 발전량은 1대 1 대응이므로 반복이 안된 데이터로 예측치를 새로 만들고 모델링하였다.
  - 히트맵으로 보았을 때 공급사만큼 반복했을 때보다 기온에서 상관관계수가 작게 나왔다.
    - 발전량 데이터에서 일정 구간이 겹치는 구간을 찾아서 재설정하였는데 변동이 없었고, 오히려 기온의 데이터의 변동으로 7번 반복된 기존의 방식의 예측 모델로 진행하기로 결정하였다.

