멋쟁이 승우처럼

팀 구성 및 역할

▼ 문승우 팀장

- 데이터 전처리, 시각화, 추가 변수 생성, 모델 성능 개선
  - merging dataset and processing missing value
  - feature engineering (using statistics, model)
  - evaluate model (using Cross Validation, Grid Search)
  - evaluation metrics – by MSE, RMSE, finally evaluated by NMAE
  - Cross Validation and Parameter Tuning

▼ 정진우 부팀장

- 모델 성능 향상, 발표
  - compare Machine Learning Model
  - Cross Validation and Parameter Tuning

▼ 오소영 팀원

- 데이터 시각화, 발표 자료 준비,PPT 자료 정리
  - Matplotlib, seaborn 등을 활용한 데이터 탐색 및 시각화

▼ 강수정 팀원

- 데이터 탐색 및 수집

DACON - 가스 공급량 수요예측 모델개발 대회

한국 가스공사의 시간단위 공급량 내부 데이터와 기상정보 및 가스 외 발전량 등 외부 데이터를 포함한 데이터셋 구축하여 90일 한도 일간 공급량을 예측하는 인공지능 모델 개발

---

1. 주제 선정 및 목표

- 지금까지 배운 내용을 토대로 실전 분석 대회에 참여하여 팀원 전체의 실전 역량 향상

- 데이터 시각화를 통해 데이터의 특징점을 찾아 데이터를 한눈에 확인할 수 있도록 분석
- 데이터가 의미하는 바를 쉽게 확인할 수 있도록 시각화하여 데이터 활용도 향상
- 가스 공급량을 예측하는 머신러닝(또는 딥러닝) 모델을 구축 및 평가
- 다양한 모델의 비교를 통해 최적의 모델을 구축 및 적용

---

2. 데이터 수집 및 전처리

- 내부 데이터(한국가스공사) - 2013 ~ 2018년 시간별 공급량 데이터
- 외부 데이터(기상청 기상자료개방포털) - 2013 ~ 2018년 서울 기상 데이터 수집
- 공급량 데이터
  - 문자열 데이터 타입의 `구분` 컬럼 → 정수형 데이터 `구분_int` 컬럼 생성
  - 문자열 데이터 타입의 `일자|시간|구분` 컬럼으로 파생변수 생성
    - `year` `month` `day` `weekday` 정수형 데이터 타입의 컬럼 생성
  - `일시` 컬럼 데이터 타입 변경 : int → datetime
  - 기온 데이터 전처리 후 공급량 데이터에 추가
    - 2013년 ~ 2018년 서울 기온 데이터 6개의 CSV 파일 → 서울 기온 데이터 파일 병합
    - 문자열 데이터 타입의 `일시` 컬럼으로 파생변수 생성
      - `year` `month` `day` `hour` 정수형 데이터 타입의 컬럼 생성
    - 연속된 행의 결측치 처리 : 전년 같은 일시의 평균 변화량만큼 증감
    - 1행의 결측치 처리 : 직전 행의 값과 직후 행의 값의 평균
- 테스트 데이터(예측해야할 19년도 데이터)
  - `일자|시간|구분` 을 `year` `month` `day` `hour` `weekday`로 새로운 컬럼 생성
  - `일자|시간|구분` 의 데이터 타입 : 문자열 → int형 및 datetime
  - 기온 예측 모델을 통해 예측한 2019년도 기온데이터 추가

---

3. 데이터 분석 및 시각화

▼ 분석 과정

- .shape

```
code : train.shape
(368088, 12)

code : test.shape
(15120, 12)

code : submission.shape
(15120, 2)
```

- .info()

```
code : train.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 368088 entries, 0 to 368087
Data columns (total 12 columns):
 #   Column   Non-Null Count   Dtype
---  ------   --------------   -----
 0   연월일      368088 non-null  object
 1   시간       368088 non-null  int64
```

```
 2   일시       368088 non-null  datetime64[ns]
 3   year     368088 non-null  int64
 4   month    368088 non-null  int64
 5   day      368088 non-null  int64
 6   hour     368088 non-null  int64
 7   weekday  368088 non-null  int64
 8   구분       368088 non-null  object
 9   구분_int   368088 non-null  int64
 10  공급량      368088 non-null  float64
 11  기온(°C)   368088 non-null  float64
dtypes: datetime64[ns](1), float64(2), int64(7), object(2)
memory usage: 33.7+ MB
None

code : test.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15120 entries, 0 to 15119
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   일자|시간|구분  15120 non-null  object
 1   일자        15120 non-null  object
 2   시간        15120 non-null  int64
 3   구분        15120 non-null  object
 4   구분_int    15120 non-null  int64
 5   일시        15120 non-null  datetime64[ns]
 6   year      15120 non-null  int64
 7   month     15120 non-null  int64
 8   day       15120 non-null  int64
 9   hour      15120 non-null  int64
 10  weekday   15120 non-null  int64
 11  기온(°C)    15120 non-null  float64
dtypes: datetime64[ns](1), float64(1), int64(7), object(3)
memory usage: 1.4+ MB
None

code : submission.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15120 entries, 0 to 15119
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   일자|시간|구분  15120 non-null  object
 1   공급량       15120 non-null  int64
dtypes: int64(1), object(1)
memory usage: 236.4+ KB
None
```

- .isnull().sum()

```
code : train.isnull().sum()
연월일        0
시간         0
일시         0
year       0
month      0
day        0
hour       0
weekday    0
구분         0
구분_int     0
공급량        0
기온(°C)     0
dtype: int64

code : test.isnull().sum()
일자|시간|구분    0
일자          0
시간          0
구분          0
구분_int      0
일시          0
year        0
month       0
day         0
hour        0
weekday     0
기온(°C)      0
dtype: int64

code : submission.isnull().sum()
일자|시간|구분    0
```

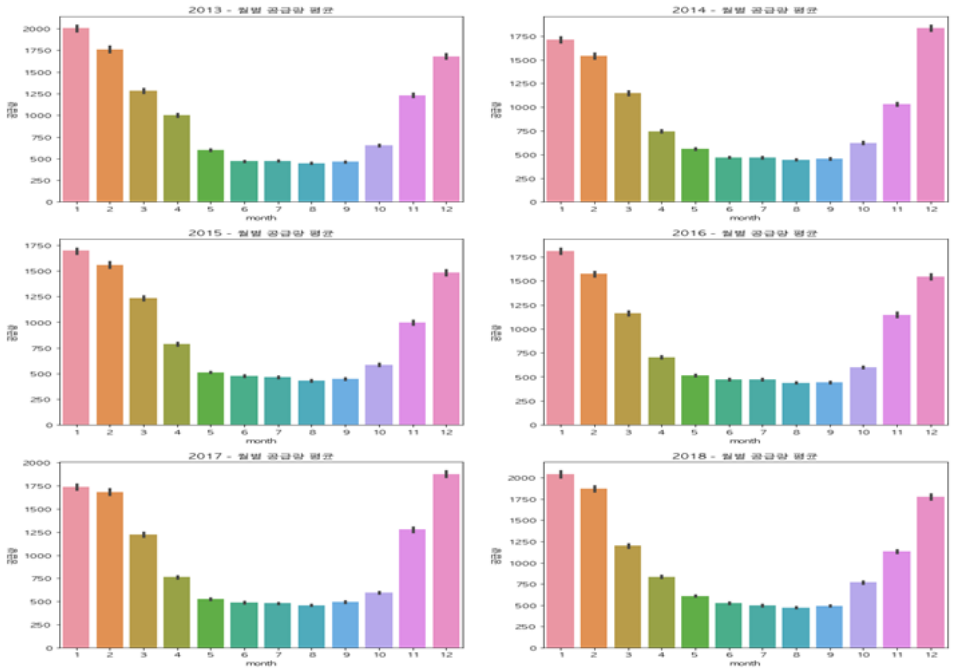
```
공급량         0
dtype: int64
```

- .describe()

```
code : train.describe()
                시간           year  ...            공급량         기온(°C)
count  368088.000000  368088.000000  ...  368088.000000  368088.000000
mean       12.500000    2015.500228  ...     948.100037      13.202670
std         6.922196       1.707471  ...     927.211578      11.267132
min         1.000000    2013.000000  ...       1.378000     -18.000000
25%         6.750000    2014.000000  ...     221.973000       3.700000
50%        12.500000    2016.000000  ...     637.014000      14.300000
75%        18.250000    2017.000000  ...    1398.919000      22.900000
max        24.000000    2018.000000  ...   11593.617000      39.400000

[8 rows x 9 columns]

code : test.describe()
                시간        구분_int  ...       weekday        기온(°C)
count  15120.000000  15120.000000  ...  15120.000000  15120.000000
mean      12.500000      3.000000  ...      3.033333      0.894907
std        6.922415      2.000066  ...      1.985848      7.503847
min        1.000000      0.000000  ...      0.000000    -17.100000
25%        6.750000      1.000000  ...      1.000000     -4.200000
50%       12.500000      3.000000  ...      3.000000      1.100000
75%       18.250000      5.000000  ...      5.000000      5.500000
max       24.000000      6.000000  ...      6.000000     21.100000

[8 rows x 8 columns]
```

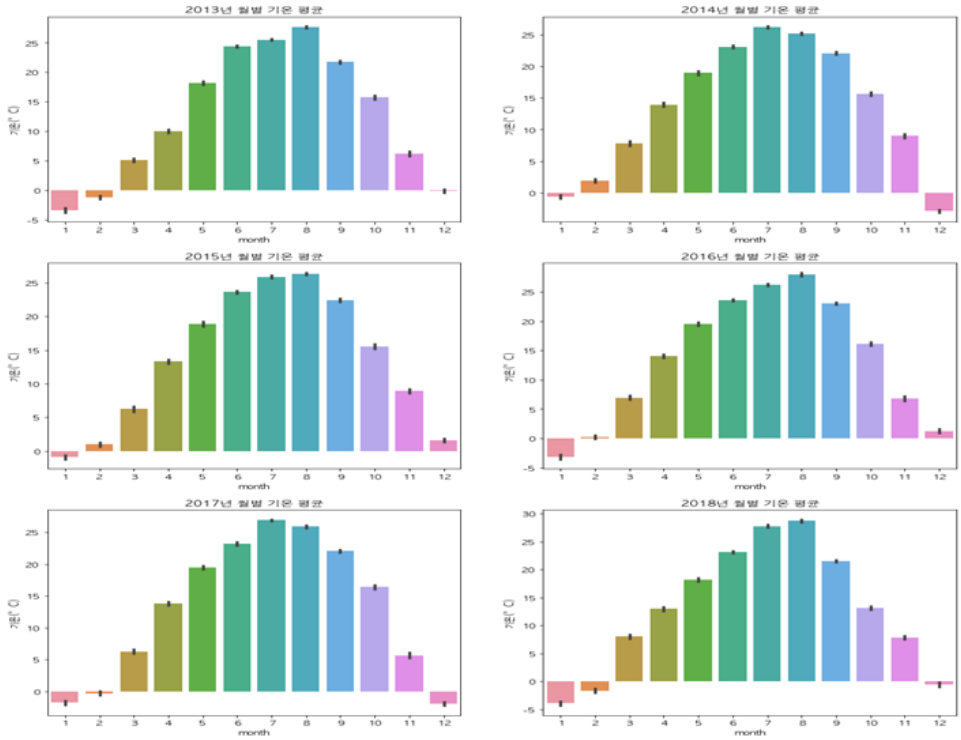
▼ 데이터 시각화

- 연도별 월평균 공급량 (데이터셋 : 훈련용 공급량 데이터)



- 2014년과 12월과 2017년 12월의 월평균 공급량이 약 1800정도로 다른 해의 12월 월평균 공급량 보다 높다.
- 2013년 1월과 2018년 1월의 월평균 공급량이 2000이상으로 다른 해의 1월 월평균 공급량에 비해 높다.

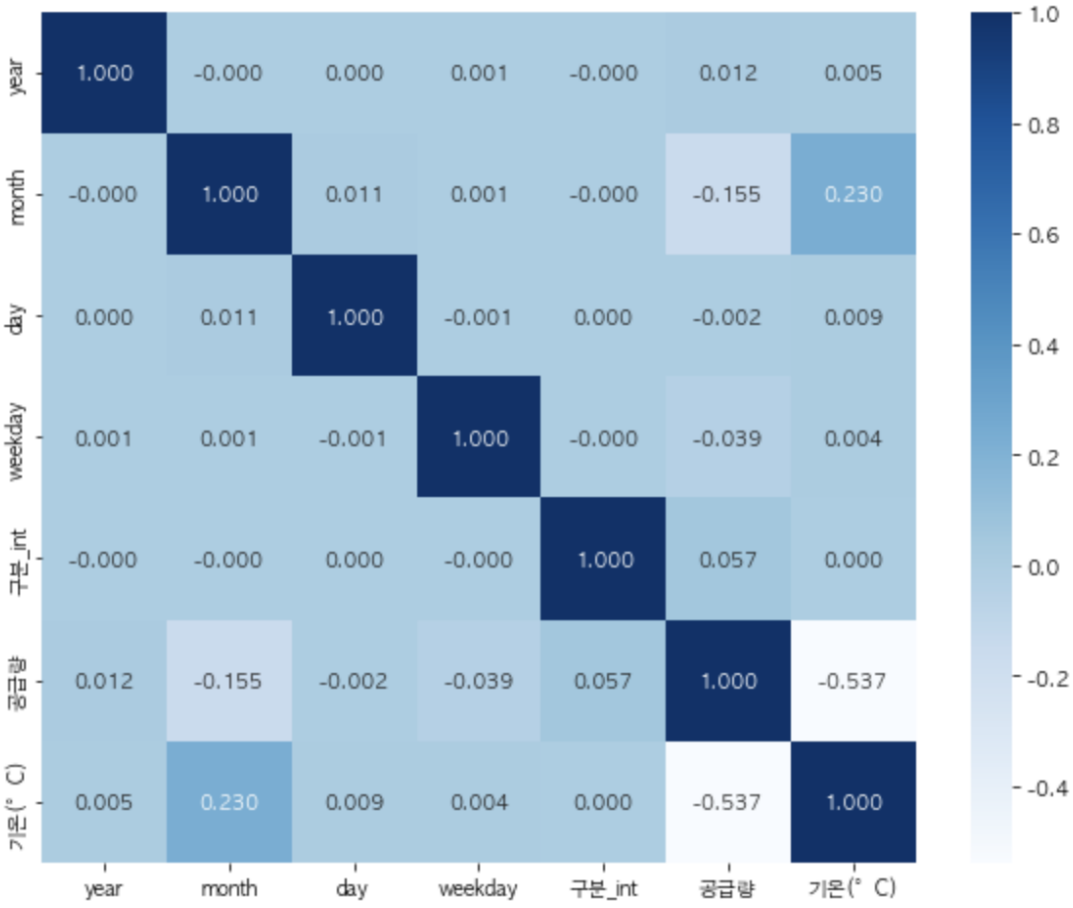
- 연도별 월평균 기온 (데이터셋 : 훈련용 기온 데이터)



- 2014년 12월과 2017년 12월의 월평균 기온이 약 -3°C정도로 다른 해의 12월 월평균 기온에 비해 낮다.
- 2013년 1월과 2018년 1월의 월평균 기온이 약 -3°C정도로 다른 해의 1월 월평균 기온에 비해 낮다.
- 해당 연도의 월 평균 기온이 해당 연도의 공급량에 영향을 미치고 있다.

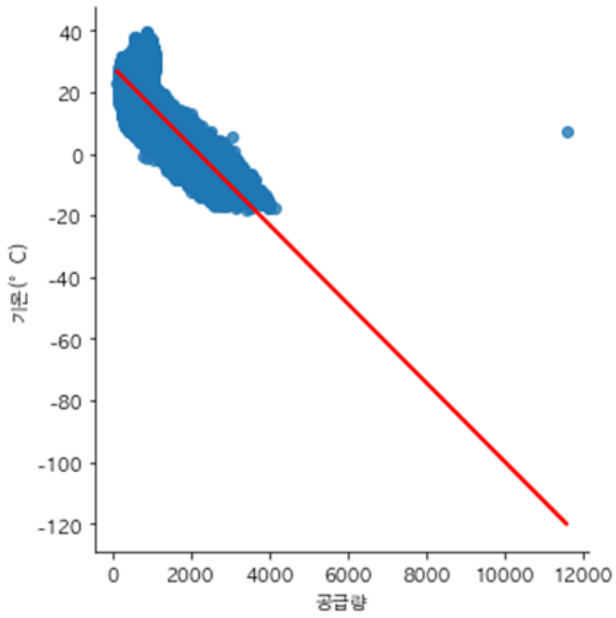
📌 해당 연도의 월 평균 기온 해당 연도 가스 공급량에 영향

- 구분A의 데이터간 상관관계 (데이터셋 : 훈련용 공급량 + 기온 데이터)



- 기온과 공급량의 음의 상관관계 존재
- 외부 기온데이터를 활용시 유의미한 결과가 예상

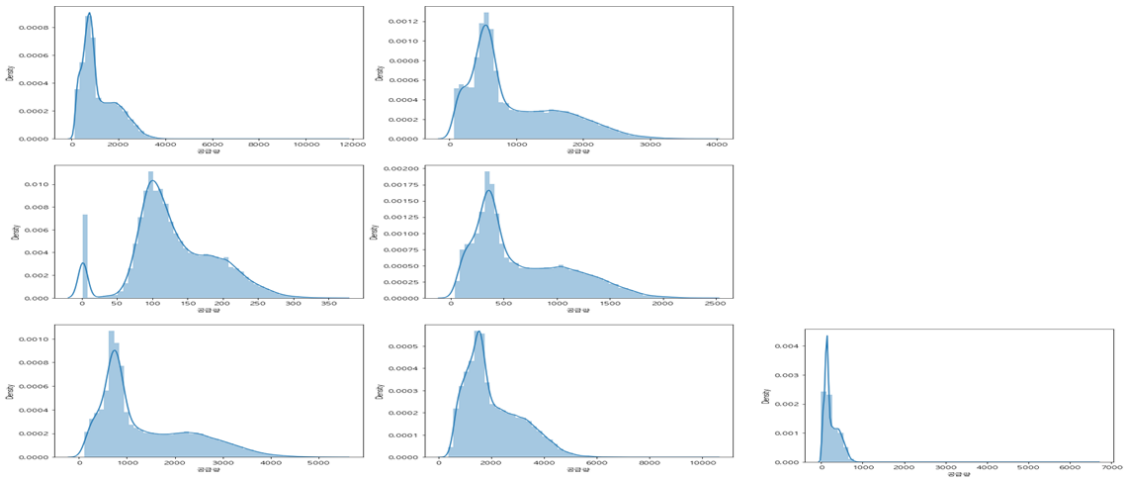
- 공급량과 기온의 상관관계 그래프 (데이터셋 : 훈련용 공급량 + 기온 데이터)



- 제공된 데이터의 공급량 값에서 이상치 확인
- 제공된 데이터로 이상치를 처리하지 않고 진행하기로 결정

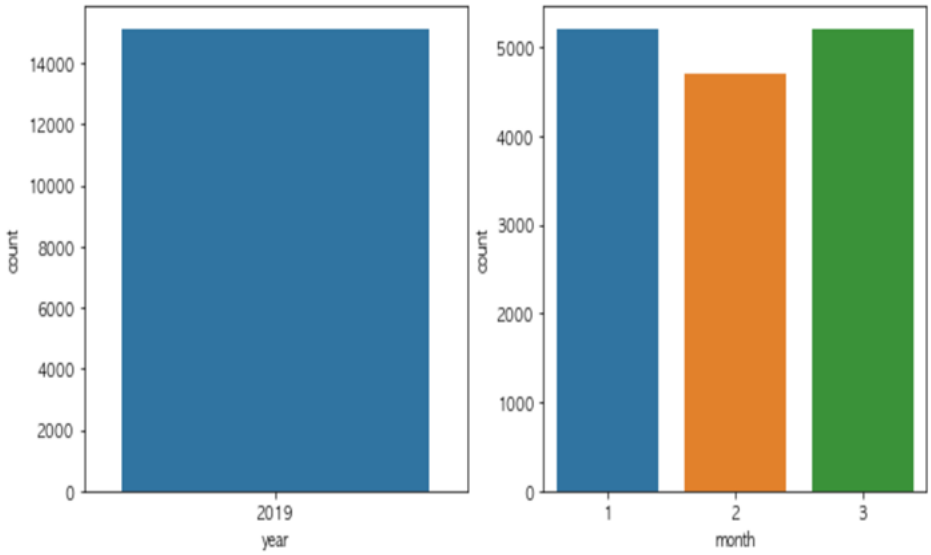
📌 가스 공급량과 기온은 음의 상관관계 존재

- 구분별 공급량 분포



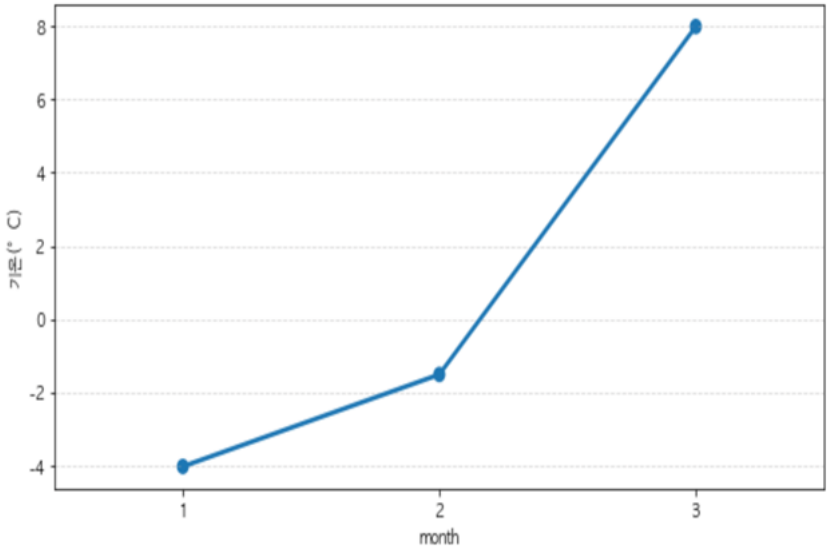
- 구분마다 공급량의 편차가 크기 때문에, 모델링 학습시 훈련 데이터를 구분별로 나눠야한다고 판단

- 예측 대상 일자(년월일시) 확인



- 테스트 데이터에 날짜 범주 확인
- 예측해야하는 날짜는 2019년 1월~3월까지로 확인

- 예측한 기온 확인 (2019년 월평균 기온)



- 예측한 2019년 월평균 기온이 1월부터 3월까지 점점 증가하는 것을 확인

---

4. 모델 구축

▼ 구축한 모델(각각 모델 비교)

▼ 공급량 예측 머신러닝 모델 구축

- 사용한 데이터 셋
  - 내부 데이터 (2013 ~ 2018년 일자별 시간단위 공급량)
- 훈련에 사용된 컬럼 : `year` `month` `day` `hour` `weekday` `구분_int`

공급량 예측 모델(기온 데이터 제외)

Aa Model	# Test Size	# Train Score	# Test Score	# MAE	# MSE	# RMSE	# NMAE
Linear Regression	0.4	3.903318	3.807003	705.069958	826141.217153	908.923108	6.687121
Linear Regression	0.8	4.055309	3.809685	705.900011	826363.177704	909.045201	6.669365
Decuision Tree Regressor	0.1	100	98.704075	47.882783	11266.179675	106.142261	0.053466
Random Forest Regressor	0.1	99.889329	99.225722	37.45872	6731.217788	82.043999	0.044543
Gradient Boosting Regressor	0.6	91.492946	91.507102	178.220452	72694.287407	269.618782	0.951729
XGB Regressor	0.2	98.398153	98.395758	74.465153	13820.674945	117.561367	0.479315
XGB Regressor	0.3	98.457612	98.352964	73.30676	14170.371153	119.039368	0.505775
LGBM Regressor	0.5	97.351338	97.219483	95.461049	23789.734282	154.239211	0.587989
LGBM Regressor	0.8	97.292267	97.111893	97.632771	24811.493274	157.516644	0.54622

▼ 2019년 기온 예측 머신러닝 모델 구축

- 사용한 데이터 셋
  - 내부 데이터 (2013 ~ 2018년 일자별 시간단위 공급량)
  - 외부 데이터 (2013 ~ 2018년 일자별 시간단위 서울 기온)
- 훈련에 사용된 컬럼 : `year` `month` `day` `hour` `weekday`

기온 예측 모델

Model	Test Size	Train Score	Test Score	MAE	MSE	RMSE	NMAE
Linear Regression	0.5	32.267667	32.162134	578.60088	580411.742242	761.847585	4.224121
Linear Regression	0.9	32.385276	32.184565	580.335669	582621.802422	763.296667	4.149284
Decuision Tree Regressor	0.9	100	96.28123	97.234126	31949.020146	178.742888	0.192316
Random Forest Regressor	0.1	99.872323	99.065209	43.288447	8126.644841	90.147905	0.065775
Gradient Boosting Regressor	0.7	95.56674	95.492504	137.911102	38648.118572	196.591247	0.945711
XGB Regressor	0.8	95.378574	95.319769	140.082871	40207.488211	200.51805	0.942795
XGB Regressor	0.2	99.177069	99.143179	53.050791	7381.581264	85.916129	0.469383
LGBM Regressor	0.2	98.66097	98.672492	67.460102	11436.590272	106.941995	0.541551
LGBM Regressor	0.3	98.707716	98.605099	67.409104	12001.11313	109.549592	0.551524

▼ 공급량 예측 머신러닝 모델 구축(기온 데이터 포함)

- 사용한 데이터 셋
  - 내부 데이터 (2013 ~ 2018년 일자별 시간단위 공급량)
  - 외부 데이터 (2013 ~ 2018년 일자별 시간단위 서울 기온)
- 훈련에 사용된 컬럼 : `year` `month` `day` `hour` `weekday` `구분int` `기온(°C)`

공급량 예측 모델(기온 데이터 포함)

Model	Test Size	Train Score	Test Score	MAE	MSE	RMSE	NMAE
Linear Regression	0.5	32.267667	32.162134	578.600880	580411.742242	761.847585	4.224121
Linear Regression	0.8	32.123475	32.226587	580.008349	582235.882776	763.043828	4.171860
Decuision Tree Regressor	0.1	100.000000	98.319467	58.284638	14609.784299	120.870941	0.074891
Random Forest Regressor	0.1	99.872323	99.065209	43.288447	8126.644841	90.147905	0.065775
Gradient Boosting Regressor	0.7	95.566740	95.492504	137.911102	38648.118572	196.591247	0.945711
XGB Regressor	0.8	95.378574	95.319769	140.082871	40207.488211	200.518050	0.942795
XGB Regressor	0.2	99.177069	99.143179	53.050791	7381.581264	85.916129	0.469383
LGBM Regressor	0.3	98.707716	98.605099	67.409104	12001.113130	109.549592	0.551524
LGBM Regressor	0.9	98.704939	98.561491	69.041222	12358.645205	111.169444	0.527669

▼ 최종 선택한 모델

- ML Model : 모두 Random Forest Regressor

최적 모델 선정

분류	Model	Test Size	Train Score	Test Score	MAE	MSE	RMSE	NMAE
2019년 공급량 예측(기온 제외)	RandomForest Regressor	0.1	99.889329	99.225722	37.45872	6731.217788	82.043999	0.044543
2019년 기온 예측	RandomForest Regressor	0.1	99.872323	99.065209	43.288447	8126.644841	90.147905	0.065775
2019년 공급량 예측 (기온 포함)	RandomForest Regressor	0.1	99.872323	99.065209	43.288447	8126.644841	90.147905	0.065775

▼ 오차값 확인, 데이콘 점수 확인

- ML Model : Random Forest Regressor
- DACON Score : 0.187347

DACON SCORE

Model	Model_name	feature	feature_count	MSE	NMAE	DACON	제출
lightgbm	lightgbm	day, hour, month, weekday, year, 구분	6			0.8256978382	@Octol
lightgbm	lightgbm	hour, month, 구분, 기온	4	32673.187915	0.533953	0.1720360982	@Nove
lightgbm	lightgbm	hour, month	2			1.6029209046	@Nove
GradientBoosting	GradientBoostingRegressor	all	7			1.7102144418	@Octol
lightgbm	lightgbm	all	7	24638.230991	0.554829	0.1843468833	@Nove
lightgbm	lightgbm	all	7			0.2108816354	@Octol
RandomForest	RandomForestRegressor	all	7	7742.093434	0.073383	0.1873469855	@Nove
RandomForest	RandomForestRegressor	day, hour, month, weekday, year, 구분	6	6601.293267	0.049287	0.1908511748	@Nove
lightgbm	LGBMRegressor	all	7	9335.072581	0.500986	0.2220811792	@Nove
lightgbm	lightgbm	day, hour, month, weekday, year, 구분	6	68453.678468	0.516178	0.1617253368	@Octol
lightgbm	lightgbm	hour, month, 구분, 기온	4	32673.187915	0.533953	0.1616776028	@Nove
lightgbm	lightgbm	hour, month, 구분, 기온	4	23935.708176	0.492063	0.1646732857	@Nove

Model	Model_name	feature	feature_count	MSE	NMAE	DACON	제출
lightgbm	lightgbm	hour, month, 구분, 기온	4	30325.16415	0.475883	0.1646659209	@Nove
DeepLearning	DeepLearning	hour, month, 구분, 기온	4	32059.9921875	0.486311	0.1754457096	@Nove
lightgbm	lightgbm	hour, month, 구분, 기온	4	46183.541544	0.489383	0.990723098	@Nove
lightgbm	lightgbm	hour, month, 구분, 기온	4	46183.541544	0.489383	0.1653082101	@Nove

5. 중간 결론

- 가장 좋았던 모델, 점수

최적 모델 선정

Model	Test Size	Train Score	Test Score	MAE	MSE	RMSE	NMAE	DACON Score
Random Forest Regressor	0.1	99.872323	99.065209	43.288447	8126.644841	90.147905	0.065775	0.187347

- 우리가 생각했던 것과 달랐던 점, 이유?
  - 내부 데이터 훈련 모델, 외부 데이터 훈련 모델 정확도 향상 미비
  - 기온 데이터에 오류가 있는 것일까?
  - 훈련이 충분하지 않았던 것일까?
- 중간발표 이후 계획
  - [x] ~~하이퍼파라미터 튜닝을 통한 모델 개선~~
  - [x] ~~공급량 데이터 비선형 변환 적용~~
  - [x] ~~Deep Learning Model 구축~~
  - [ ] 서울 지역 외 각 지방 데이터를 활용하여 구분별로 서로 다른 지역의 기온 데이터를 훈련시켜 정확도 향상

6. 모델 변경

- RandomForestRegressor : NMAE 값이 가장 작지만, 결정 계수가 **0.99** 로 과적합 문제 발생
- DecisionTreeRegressor : RandomForestRegressor와 같은 이유로 선택하지 않음
- xgboost : 결정 계수가 **0.99** 로 과적합 문제가 존재하는 것으로 판단
- **lightgbm** 모델의 하이퍼파라미튜 튜닝을 통해 모델을 개선하기로 선택

Model Compare

Name	train_score	test_score	MAE	MSE	RMSE	NMAE
LinearRegression	0.322446	0.321294	579.143569	583219.98038	763.688405	4.203546
DecisionTreeRegressor	100	0.982109	62.086361	15373.939242	123.99169	0.077439
RandomForestRegressor	0.998569	0.990789	45.611428	7915.180903	88.967302	0.075378
GradientBoostingRegressor	0.952538	0.952479	142.654715	40835.358653	202.077606	1.008954
xgboost	0.991449	0.991218	54.486956	7546.847455	86.872593	0.490392
lightgbm	0.986545	0.9867	68.074064	11428.442172	106.903892	0.536466

7. Cross Validation & GridSearchCV

- Cross Validation Score → **0.964** 에서 **0.976** 으로 향상

▼ 하이퍼파라미터 튜닝 및 교차 검증 → RandomForest 모델과 비교하여 **0.015** 향상

```
grid_parameters = {"max_depth" : [12, 14, 16],
                   "n_estimators" : [1000, 1500, 2000],
                   "learning_rate" : [0.01, 0.05, 0.1]}

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

kfold = KFold(n_splits = 10, shuffle = True, random_state = 27)

model = lgb.LGBMRegressor(**params)
grid_search = GridSearchCV(model, param_grid = grid_parameters, cv = kfold, n_jobs = -1)
grid_search.fit(train_x, train_y)

print('GridSearchCV 최적 파라미터 : ', grid_search.best_params_)
print('GridSearchCV 최고 정확도 : {0:.4f}'.format(grid_search.best_score_))

# GridSearchCV 최적 파라미터 :  {'learning_rate': 0.05, 'max_depth': 12, 'n_estimators': 1000}
# GridSearchCV 최고 정확도 : 0.9717
```

```
GridSearchCV 최적 파라미터 :
{'learning_rate': 0.1, 'max_depth': 12, 'n_estimators': 2000}
GridSearchCV 최고 정확도 : 0.9771
```

모델 비교

구분	MODEL	MSE	NMAE	DACON
변경 전 모델	RandomForestRegressor	7742.093434	0.073383	0.187347
변경 후 모델	lightgbm	24638.230991	0.554829	0.184347
GridSearchCV 적용 모델	lightgbm	32673.187915	0.533953	0.172036

8. 데이콘 점수가 낮은 이유 분석

- 19년도 기온 예측 모델 오류 → 구분 데이터를 포함하여 모델 생성
- XGBRegressor() 모델 학습 → 교차 검증 정확도 : **0.975**

9. 새롭게 예측한 기온 데이터를 활용한 모델 구축 및 제출 점수 확인

- 같은 모델에 테스트 데이터만 다르게 하여 확인
- 변경 후 **0.01** 향상

변경 전후 비교

구분	MODEL	MSE	NMAE	DACON
변경 전	lightgbm	32673.187915	0.533953	0.172036
변경 후	lightgbm	32673.187915	0.533953	0.161677

10. 가스공급량 데이터 비선형 변환 적용

- 비선형 변환 적용 전후 NMAE 비교
  - 0.533953 → 0.518289
  - 0.015 감소 → DACON 점수 확인 필요

11. Deep Learning Model 구축

- lightgbm 모델에서 높은 점수를 보여줬던 feature를 사용해 모델 구축
- 5개의 은닉층을 사용한 모델의 DACON 점수 → **0.1754457096**
- Underfitting, Overfitting 여부 확인 필요

Deep Learning Model

layers	Hidden layers	epochs	batch_size	MSE	NMAE	DACON
4, 32, 32, 1	2	100	10	32944.64453125	0.643535	
4, 16, 32, 64, 32, 16, 1	5	100	10	32059.9921875	0.486311	0.1754457096

12. 향후 계획

- [ ] 기온 데이터 구간분할 적용
- [ ] 특성자동 선택 적용
- [ ] Deep Learning Model 성능 개선
- [ ] 계속 모델을 개선하면서 점수 확인