DACON

AI프렌즈 시즌1
공공데이터를 활용한 온도추정 경진대회

팀장 문성민
팀원 이유준,성민석,정재엽

https://dacon.io/competitions/official/235584/overview/

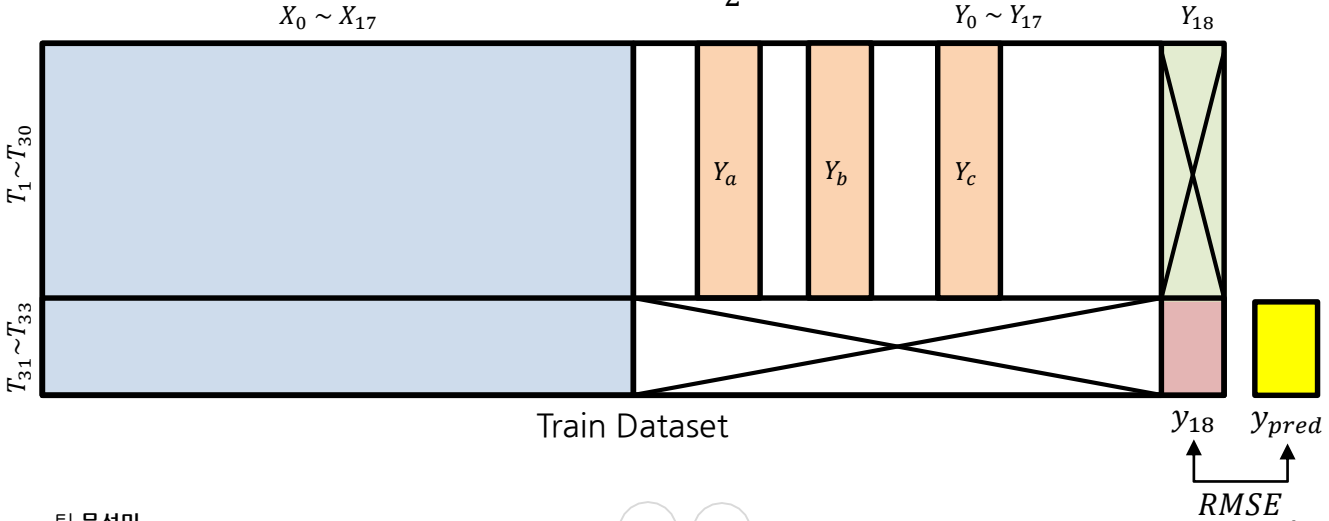
목차

데이터 전처리

Y값 설정

- Y_{18}과 상관관계가 낮았던 Y_3과 Y_4를 제외
- Y_0부터 Y_{17} 를 조합하여 Y_{18}이 없는 30일의 데이터로
- Y_{18}이 존재했던 마지막 3일을 기준으로 Y값 추정
    - 3개의 Y값 조합(Y_a, Y_b, Y_c)으로 학습하여 Y_{18} 값과 $RMSE$ 값 비교

$$Y_{18} = \frac{\left(\frac{1}{3}(Y_6 + Y_{10} + Y_{12})\right) + \left(\frac{1}{3}(Y_0 + Y_{10} + Y_{12})\right)}{2}$$



Y값 설정 (실제 코드)

DACON
DATA TO VALUE

```python
df_month = train[train.loc[:,'Y00':'Y17'].notnull().all(1)].drop(columns = 'Y18')
df_days = train[train['Y18'].notnull()].drop(train.loc[:,'Y00':'Y17'],axis=1)

trainX1, trainX2 = df_month, df_days

trainX1['Y18'] = ((trainX1['Y06'] + trainX1['Y10'] + trainX1['Y12'])/3 + (trainX1['Y00'] + trainX1['Y10'] + trainX1['Y12'])/3)/ 2

trainX1 = trainX1.drop(columns = ['Y00', 'Y01', 'Y02', 'Y03', 'Y04', 'Y05', 'Y06', 'Y07', 'Y08',
       'Y09', 'Y10', 'Y11', 'Y12', 'Y13', 'Y14', 'Y15', 'Y16', 'Y17'])

train = pd.concat([trainX1,trainX2])

train['Y18'] = np.log1p(train['Y18'])

y_train = train['Y18']

data = pd.concat([train.drop(columns=['Y18']),test])

data.index = [i for i in range(len(data))]

기온=['X00', 'X07', 'X28', 'X31', 'X32']
현지기압=['X01', 'X06', 'X22', 'X27', 'X29']
풍속=['X02', 'X03', 'X18', 'X24', 'X26']
강수량=['X04', 'X10', 'X21', 'X36', 'X39']
해면기압=['X05', 'X08', 'X09', 'X23', 'X33']
일사량=['X11', 'X14', 'X16', 'X19', 'X34']
습도=['X12', 'X20', 'X30', 'X37', 'X38']
풍향=['X13', 'X15', 'X17', 'X25', 'X35']

lst = [기온,현지기압,풍속,강수량,해면기압,일사량,습도,풍향]
name_lst = ['기온','현지기압','풍속','강수량','해면기압','일사량','습도','풍향']

col_names = {}
for l,n in zip(lst, name_lst):
    for i in range(5):
        col_names[f'{l[i]}'] = f'{n}_{i+1}'

data.rename(columns = col_names,inplace=True)
train.rename(columns = col_names,inplace=True)
test.rename(columns = col_names,inplace=True)
```

Y_{18} 설정

용이한 학습을 위한 스케일링

쉬운 식별화를 위한 데이터 전처리 작업

시간 관련 특징 추가 - 시간 범주

- 시간을 범주화 후 해당 시간의 Y_{18} 통계치를 넣음

```
minute_data = (data.id%144).astype(int)

data['min'] = minute_data

# data['hour'] = pd.Series((data.index%144/6).astype(int))

train['min'] = (train.id%144).astype(int)
```

범주화를 위한 함수 정의

```
def f(x) :
    if x< 18 :
        return('t1')
    elif x< 39 :
        return('t2')
    elif x< 48 :
        return('t3')
    elif x< 72 :
        return('t4')
    elif x< 86 :
        return('t5')
    elif x< 102 :
        return('t6')
    elif x< 114 :
        return('t7')
    elif x< 126 :
        return('t8')
    else :
        return('t9')

data['time_cat'] = data['min'].apply(f)

train['time_cat'] = train['min'].apply(f)
```

시간 범주에 해당하는 통계치

```
features = []
f = train.groupby(['time_cat'])['Y18'].agg([('시간별합','sum')]).reset_index()
features.append(f)
f = train.groupby(['time_cat'])['Y18'].agg([('시간별평균','mean')]).reset_index()
features.append(f)
f = train.groupby(['time_cat'])['Y18'].agg([('시간별편차','std')]).reset_index()
features.append(f)
```

```
for f in features :
    data = pd.merge(data, f, how='left',on=['time_cat'])
display(data.shape)
```

시간 관련 특징 추가 - 강수량

- 시간과 강수 유무를 동시에 고려한 특징 추가
- 합, 평균 그리고 편차와 같은 기초 통계량 사용

```
def f2(x) :
    if x <= 0.5 :
        return(0)
    else : return(1)
```

```
data['rain'] = data['강수량_2'].apply(f2)
train['rain'] = train['강수량_2'].apply(f2)
```

```
features = []
f = train.groupby(['time_cat','rain'])['Y18'].agg([('시간강수별합','sum')]).reset_index()
features.append(f)
f = train.groupby(['time_cat','rain'])['Y18'].agg([('시간강수별평균','mean')]).reset_index()
features.append(f)
f = train.groupby(['time_cat','rain'])['Y18'].agg([('시간강수별편차','std')]).reset_index()
features.append(f)
```

```
for f in features :
    data = pd.merge(data, f, how='left',on=['time_cat','rain'])
display(data.shape)
```

```
(16272, 50)
```

```
data = pd.get_dummies(data=data, columns=['time_cat'])
```

시간 관련 특징 추가 - 시간에 따른 기온 차

- 기온과 관련된 5개의 변수에 대하여 각각 **6시간, 12시간, 24시간 그리고 36시간 전** 기온의 차이에 대한 변수 생성

```
temper_lst = ['기온_1', '기온_2', '기온_3', '기온_4', '기온_5']
```

24시간

```
for temper in temper_lst:
    interval = 144
    diff_temper = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[temper].iloc[i] - data[temper].iloc[i-interval]
            diff_temper.append(diff)
        else:
            diff = data[temper].iloc[i] - data[temper].iloc[0]
            diff_temper.append(diff)
    data['diff_1d{}'.format(temper)] = diff_temper
```

12시간

```
for temper in temper_lst:
    interval = 72
    diff_temper = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[temper].iloc[i] - data[temper].iloc[i-interval]
            diff_temper.append(diff)
        else:
            diff = data[temper].iloc[i] - data[temper].iloc[0]
            diff_temper.append(diff)
    data['diff2_{}'.format(temper)] = diff_temper
```

6시간

```
for temper in temper_lst:
    interval = 36
    diff_temper = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[temper].iloc[i] - data[temper].iloc[i-interval]
            diff_temper.append(diff)
        else:
            diff = data[temper].iloc[i] - data[temper].iloc[0]
            diff_temper.append(diff)
    data['diff3_{}'.format(temper)] = diff_temper
```

36시간

```
for temper in temper_lst:
    interval = 216
    diff_temper = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[temper].iloc[i] - data[temper].iloc[i-interval]
            diff_temper.append(diff)
        else:
            diff = data[temper].iloc[i] - data[temper].iloc[0]
            diff_temper.append(diff)
    data['diff2_5min_{}'.format(temper)] = diff_temper
```

주의:
해당 순서로 하지 않을 시 추후 LightGBM의 성능이 달라질 수 있음

시간 관련 특징 추가 - 시간에 따른 습도 차

- 습도과 관련된 5개의 변수에 대하여 각각 **6시간, 12시간, 24시간 그리고 36시간 전** 기온의 차이에 대한 변수 생성

24시간

```
wet_lst = ['습도_1', '습도_2','습도_3']

for wet in wet_lst:
    interval = 144
    diff_wet = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[wet].iloc[i] - data[wet].iloc[i-interval]
            diff_wet.append(diff)
        else:
            diff = data[wet].iloc[i] - data[wet].iloc[0]
            diff_wet.append(diff)
    data['diff습도144_{}'.format(wet)] = diff_wet
```

12시간

```
for wet in wet_lst:
    interval = 72
    diff_wet = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[wet].iloc[i] - data[wet].iloc[i-interval]
            diff_wet.append(diff)
        else:
            diff = data[wet].iloc[i] - data[wet].iloc[0]
            diff_wet.append(diff)
    data['diff습도72_{}'.format(wet)] = diff_wet
```

36시간

```
for wet in wet_lst:
    interval = 216
    diff_wet = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[wet].iloc[i] - data[wet].iloc[i-interval]
            diff_wet.append(diff)
        else:
            diff = data[wet].iloc[i] - data[wet].iloc[0]
            diff_wet.append(diff)
    data['diff습도216_{}'.format(wet)] = diff_wet
```

6시간

```
for wet in wet_lst:
    interval = 36
    diff_wet = []
    for i in range(data.shape[0]):
        if i > interval:
            diff = data[wet].iloc[i] - data[wet].iloc[i-interval]
            diff_wet.append(diff)
        else:
            diff = data[wet].iloc[i] - data[wet].iloc[0]
            diff_wet.append(diff)
    data['diff습도36_{}'.format(wet)] = diff_wet
```

주의:
해당 순서로 하지 않을 시 추후 LightGBM의 성능이 달라질 수 있음

시간 관련 특징 추가 - 시간에 따른 습도와 일사량 차

- 습도와 관련된 5개 변수에 대해 10분과 1시간 **이전**과의 차이에 대한 변수 생성
- 또한, 누적 일사량과 관련된 5개의 변수 중 실제로 측정된 변수 2개(X11, X34)를 통해, 10분과 1시간 단위의 일사량이라는 변수 생성

```python
#interval단위로 차이를 구하는 함수를 생성.
def to_per_time(interval, df, colname):
    day = data['min'].nunique() #8*24
    diff = []
    for num_days in range(int(data.shape[0]/data['min'].nunique())):
        for i in range(day):
            if i < interval: #interval단위 이전의 기록들은 차이를 비교할 수 없어, 0으로 대체.
                diff.append(0)
            else:
                diff.append(df[colname][i] - df[colname][i-interval])
    return diff
```

```python
intervals = [1,6]
for interval in intervals:
    for i in range(1,6):
        data[f'diff_습도_{interval}_{i}'] = to_per_time(6,data,f'습도_{i}')
```

```python
intervals = [1,6]
for interval in intervals:
    for i in range(1,6):
        try : data[f'diff_일사량_{interval}_{i}'] = to_per_time(interval,data,f'일사량_{i}')
        except : pass
```

시간 관련 특징 추가 - 시간에 따른 습도와 일사량 차

DACON
DATA TO VALUE

- 습도의 10분 차에 대한 변수 5개의 평균을 도출하고 변수 생성
- 10분 단위 일사량에 대한 변수 2개의 평균을 도출하고 변수 생성
- 습도와 일사량에 대한 평균변수 2개를 곱한 변수 생성
    - Why? 습도와 일사량의 관계를 모델에 적용하기 위해

```
data['습도_mean'] = np.mean(data[data.columns[data.columns.str.contains('diff_습도_1')]],axis=1)
data['일사량_mean'] = np.mean(data[['diff_일사량_1_1','diff_일사량_1_5']],axis=1)
```

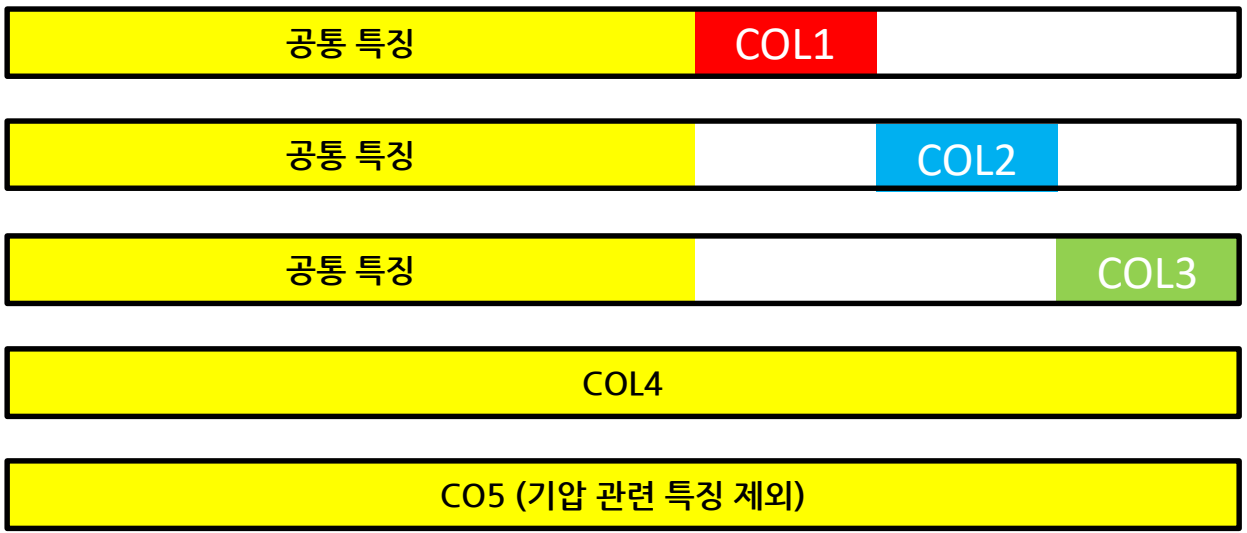
```
data['일사량_습도'] = data['습도_mean']*data['일사량_mean']
```

diff_습도_1_n : 습도의 10분 차에 대한 변수
diff_일사량_1_n : 일사량의 10분 차에 대한 변수

모델링

특징 분할

- 특징을 5개로 분할하여 앙상블 준비



Feature Set

특징 분할 (실제코드)

DACON
DATA TO VALUE

```
COL1 = ['기온_1', '현지기압_1', '풍속_1', '풍속_2', '강수량_1', '해면기압_1', '현지기압_2', '기온_2',
        '해면기압_2', '해면기압_3', '강수량_2', '일사량_1', '습도_1', '풍향_1', '일사량_2', '풍향_2',
        '일사량_3', '풍향_3', '풍속_3', '일사량_4', '습도_2', '강수량_3', '현지기압_3', '해면기압_4',
        '풍속_4', '풍향_4', '풍속_5', '현지기압_4', '기온_3', '현지기압_5', '습도_3', '기온_4',
        '기온_5', '해면기압_5', '일사량_5', '풍향_5', '강수량_4', '습도_4', '습도_5', '강수량_5',
        'min', 'diff3_기온_4', 'diff3_기온_2', 'diff_1d기온_5', 'diff3_기온_5', 'diff_1d기온_4',
        'diff_1d기온_2', 'diff3_기온_1', 'diff3_기온_3', 'diff_습도_1_2', 'rain', 'diff습도216_습도_2',
        'diff_습도_1_5', 'time_cat_t6', 'diff습도36_습도_3', 'diff습도72_습도_2', 'time_cat_t3', '일사량_습도',
        'diff_일사량_1_5', 'diff_습도_6_4', 'diff_일사량_1_4', 'diff습도144_습도_1', 'diff_습도_6_5',
        '시간강수별합', 'diff2_5min_기온_5', 'time_cat_t5', 'diff_일사량_1_3', 'diff_습도_6_3',
        'diff_일사량_1_2', 'diff2_5min_기온_3']
```

```
COL2 = ['기온_1', '현지기압_1', '풍속_1', '풍속_2', '강수량_1', '해면기압_1', '현지기압_2', '기온_2',
        '해면기압_2', '해면기압_3', '강수량_2', '일사량_1', '습도_1', '풍향_1', '일사량_2', '풍향_2',
        '일사량_3', '풍향_3', '풍속_3', '일사량_4', '습도_2', '강수량_3', '현지기압_3', '해면기압_4',
        '풍속_4', '풍향_4', '풍속_5', '현지기압_4', '기온_3', '현지기압_5', '습도_3', '기온_4',
        '기온_5', '해면기압_5', '일사량_5', '풍향_5', '강수량_4', '습도_4', '습도_5', '강수량_5',
        'min', 'diff3_기온_4', 'diff3_기온_2', 'diff_1d기온_5', 'diff3_기온_5', 'diff_1d기온_4',
        'diff_1d기온_2', 'diff3_기온_1', 'diff3_기온_3', 'diff_일사량_6_3', 'diff_습도_1_1', 'diff_일사량_6_2',
        'diff2_기온_4', 'diff2_5min_기온_2', 'diff_일사량_6_1', 'diff습도144_습도_2', 'time_cat_t9', '습도_mean',
        'diff2_5min_기온_1', 'diff_일사량_1_1', 'diff_습도_1_3', 'diff2_기온_1', 'diff습도36_습도_1', 'diff_습도_1_4',
        'diff_1d기온_3', '시간별편차', 'diff2_5min_기온_4', '시간강수별평균', '시간강수별편차', '일사량_mean']
```

```
COL3 = ['기온_1', '현지기압_1', '풍속_1', '풍속_2', '강수량_1', '해면기압_1', '현지기압_2', '기온_2',
        '해면기압_2', '해면기압_3', '강수량_2', '일사량_1', '습도_1', '풍향_1', '일사량_2', '풍향_2',
        '일사량_3', '풍향_3', '풍속_3', '일사량_4', '습도_2', '강수량_3', '현지기압_3', '해면기압_4',
        '풍속_4', '풍향_4', '풍속_5', '현지기압_4', '기온_3', '현지기압_5', '습도_3', '기온_4',
        '기온_5', '해면기압_5', '일사량_5', '풍향_5', '강수량_4', '습도_4', '습도_5', '강수량_5',
        'min', 'diff3_기온_4', 'diff3_기온_2', 'diff_1d기온_5', 'diff3_기온_5', 'diff_1d기온_4',
        'diff_1d기온_2', 'diff3_기온_1', 'diff3_기온_3', 'diff2_기온_5', 'diff_일사량_6_5',
        'diff_습도_6_2', 'diff습도36_습도_2', 'diff습도216_습도_3', 'time_cat_t8', 'diff2_기온_3', 'time_cat_t4',
        'diff습도144_습도_3', 'diff습도72_습도_1', 'time_cat_t2', '시간별평균', 'diff_1d기온_1', 'diff습도72_습도_3',
        '시간별합', 'time_cat_t1', 'diff_습도_6_1', 'diff_일사량_6_4', 'diff2_기온_2', 'time_cat_t7',
        'diff습도216_습도_1']
```

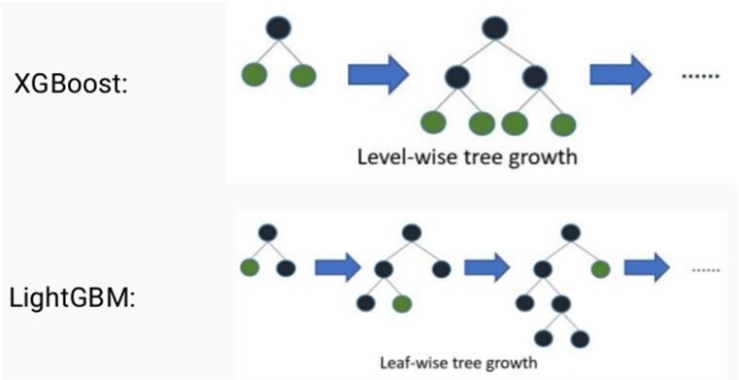
특징 분할 (실제코드)

```
COL4= ['기온_1', '현지기압_1', '풍속_1', '풍속_2', '강수량_1', '해면기압_1', '현지기압_2', '기온_2',
       '해면기압_2', '해면기압_3', '강수량_2', '일사량_1', '습도_1', '풍향_1', '일사량_2', '풍향_2',
       '일사량_3', '풍향_3', '풍속_3', '일사량_4', '습도_2', '강수량_3', '현지기압_3', '해면기압_4',
       '풍속_4', '풍향_4', '풍속_5', '현지기압_4', '기온_3', '현지기압_5', '습도_3', '기온_4',
       '기온_5', '해면기압_5', '일사량_5', '풍향_5', '강수량_4', '습도_4', '습도_5', '강수량_5',
       'min', '시간별합', '시간별평균', '시간별편차', 'rain', '시간강수별합', '시간강수별평균', '시간강수별편차',
       'time_cat_t1', 'time_cat_t2', 'time_cat_t3', 'time_cat_t4',
       'time_cat_t5', 'time_cat_t6', 'time_cat_t7', 'time_cat_t8',
       'time_cat_t9', 'diff_1d기온_1', 'diff_1d기온_2', 'diff_1d기온_3',
       'diff_1d기온_4', 'diff_1d기온_5', 'diff2_기온_1', 'diff2_기온_2', 'diff2_기온_3',
       'diff2_기온_4', 'diff2_기온_5', 'diff3_기온_1', 'diff3_기온_2', 'diff3_기온_3',
       'diff3_기온_4', 'diff3_기온_5', 'diff2_5min_기온_1', 'diff2_5min_기온_2',
       'diff2_5min_기온_3', 'diff2_5min_기온_4', 'diff2_5min_기온_5',
       'diff_습도_1_1', 'diff_습도_1_2', 'diff_습도_1_3', 'diff_습도_1_4',
       'diff_습도_1_5', 'diff_습도_6_1', 'diff_습도_6_2', 'diff_습도_6_3',
       'diff_습도_6_4', 'diff_습도_6_5', 'diff_일사량_1_1',
       'diff_일사량_1_2', 'diff_일사량_1_3', 'diff_일사량_1_4', 'diff_일사량_1_5',
       'diff_일사량_6_1', 'diff_일사량_6_2', 'diff_일사량_6_3', 'diff_일사량_6_4',
       'diff_일사량_6_5', '습도_mean', '일사량_mean', '일사량_습도']
```

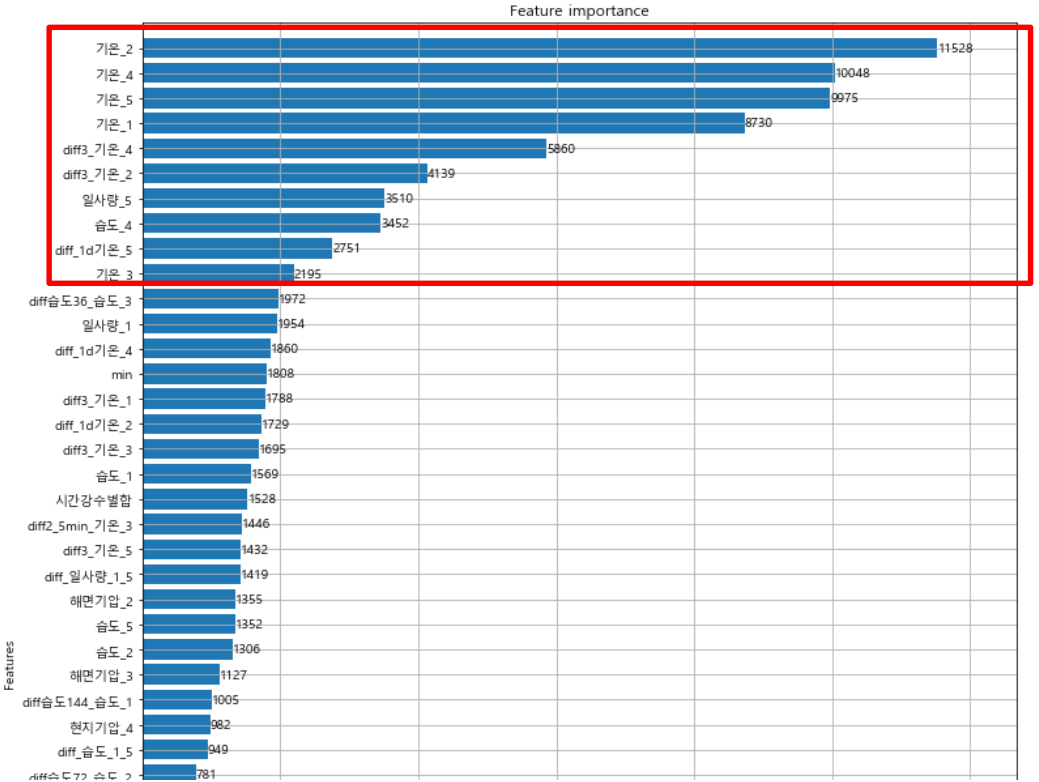
```
COL5 =['기온_1', '풍속_1', '풍속_2', '강수량_1', '기온_2', '강수량_2', '일사량_1', '습도_1',
       '풍향_1', '일사량_2', '풍향_2', '일사량_3', '풍향_3', '풍속_3', '일사량_4', '습도_2',
       '강수량_3', '풍속_4', '풍향_4', '풍속_5', '기온_3', '습도_3', '기온_4', '기온_5',
       '일사량_5', '풍향_5', '강수량_4', '습도_4', '습도_5', '강수량_5', 'min', '시간별합',
       '시간별평균', '시간별편차', 'rain', '시간강수별합', '시간강수별평균', '시간강수별편차', 'time_cat_t1',
       'time_cat_t2', 'time_cat_t3', 'time_cat_t4', 'time_cat_t5',
       'time_cat_t6', 'time_cat_t7', 'time_cat_t8', 'time_cat_t9',
       'diff_36_기온_1', 'diff_72_기온_1', 'diff_144_기온_1', 'diff_216_기온_1',
       'diff_36_기온_2', 'diff_72_기온_2', 'diff_144_기온_2', 'diff_216_기온_2',
       'diff_36_기온_3', 'diff_72_기온_3', 'diff_144_기온_3', 'diff_216_기온_3',
       'diff_36_기온_4', 'diff_72_기온_4', 'diff_144_기온_4', 'diff_216_기온_4',
       'diff_36_기온_5', 'diff_72_기온_5', 'diff_144_기온_5', 'diff_216_기온_5',
       'diff_습도_1_1', 'diff_습도_1_2', 'diff_습도_1_3', 'diff_습도_1_4',
       'diff_습도_1_5', 'diff_습도_6_1', 'diff_습도_6_2', 'diff_습도_6_3',
       'diff_습도_6_4', 'diff_습도_6_5', 'diff_일사량_1_1', 'diff_일사량_1_2',
       'diff_일사량_1_3', 'diff_일사량_1_4', 'diff_일사량_1_5', 'diff_일사량_6_1',
       'diff_일사량_6_2', 'diff_일사량_6_3', 'diff_일사량_6_4', 'diff_일사량_6_5',
       '습도_mean', '일사량_mean', '일사량_습도']
```

LightGBM

- LightGBM 2.2.3 버전 사용
- 하이퍼파라미터
    - boosting: dart
    - learning_rate: 0.05
    - max_depth: 10
    - bagging_fraction: 0.8
    - num_leaves: 128
    - feature_fraction: 0.8



LightGBM - Feature Importance



앙상블(Ensemble)

- COL1 ~ COL3까지의 멱평균(Power mean)을 진행
- 이 때, p값은 88.4

$$M_p(x_1, \ldots, x_n) = \left(\sum_{i=1}^{n} x_i^p \right)^{\frac{1}{p}}$$

```
nf = 0
for Q in os.listdir(folder):
    ext = os.path.splitext(Q)[-1]
    if ext == '.csv':
        s = pd.read_csv(folder+"/"+Q)
    else:
        continue
    if len(s.columns) !=2:
        continue
    if nf == 0:
        slist = s
    else:
        slist = pd.merge(slist, s, on="id")
    nf += 1

p = 88.4  # 이 파라미터는 멱평균 앙상블에 있어 중요한 수치임. 최적의 수치를 찾기 바랍니다.
if nf >= 2:
    pred_m = 0
    for J in range(nf): pred_m = pred_m + slist.iloc[:,J+1]**p
    pred_m = pred_m / nf
    pred_m = pred_m**(1/p)

    submit = pd.DataFrame({'id': slist.id, 'Y18': pred_m})
    fname = "data/submission_PM.csv"
    submit.fillna(0).to_csv(fname, index=False)
```

제출 파일 생성

- 최종 결과로는 3가지 결과값으로 아래 식과 같은 가중평균 생성
  - a : COL1 ~ COL3로 만든 멱평균
  - b : COL4
  - c : COL5

$$Submisssion = 0.5 \times a + 0.1 \times b + 0.4 \times c$$

```
a = pd.read_csv('data/submission_PM.csv')['Y18']
b = pd.read_csv('data/0409_test.csv')['Y18']
c = pd.read_csv('data/0413_test.csv')['Y18']
```

최종 제출 파일

```
w1 = 0.5
w2 = 0.1
w3 = 0.4

pd.DataFrame({'id': slist.id, 'Y18': (w1*a+w2*b+w3*c)}).to_csv('data/sanbest4_test.csv',index=False)
```

결과 및 결언

결과 및 결언

- **Y값 설정**이 모델의 성능을 크게 좌우함.
- Y18과 상관관계가 컸던 **Y15**와 **Y16**으로 회귀식을 만들어 Y18값을 만들었으나 Public Score가 좋지 못함.
- **시간의 흐름을 반영**하여 생성한 Feature인 온도/습도 변화량이 Public Score를 향상시키는데 크게 기여함. 그 중 특히 **기온**을 활용한 Feature(X00,X07…. 등)의 기여도가 높았음.
- *XGBoost*이나 *LSTM* 모델보단 ***LightGBM* 모델의 성능이 더 좋았으며**, GBDT보다 **DART**에서 좋은 성능을 보였음.
- **기압 관련 특징 제외**한 모델의 성능이 그렇지 않은 모델의 성능보다 우수했음.
- 모델의 **예측 값에 편향을 추가** 시, Public Score에서 더욱 좋은 성능을 보임

감사합니다