태양광 발전량 예측 AI 경진대회

Human_Learning

62등(상위 14%)

주최: 한국원자력연구원(KAERI) / 주관: DACON

2020.12.09 ~ 2021.01.26

1. 대회 개요

- 목적
- 모델은 7일(Day 0~ Day6) 동안의 데이터를 인풋으로 활용
- 향후 2일(Day7 ~ Day8) 동안의 30분 간격의 발전량(TARGET)을 예측

- 채점 방식
- Pinball Loss

- Public Score & Private Score
- 1차 평가(Public Score): 테스트 데이터 중 랜덤 샘플 된 50 %로 채점, 대회 기간 중 공개
- 2차 평가(Private Score): 나머지 테스트 데이터로 채점, 대회 종료 직후 공개

1. 대회 개요

- Pinball Loss
  - 심사 기준: Pinball Loss

$$L_\tau(y,z) = (y-z)\tau \quad \text{if } y \geq z$$
$$= (z-y)(1-\tau) \quad \text{if } z > y$$

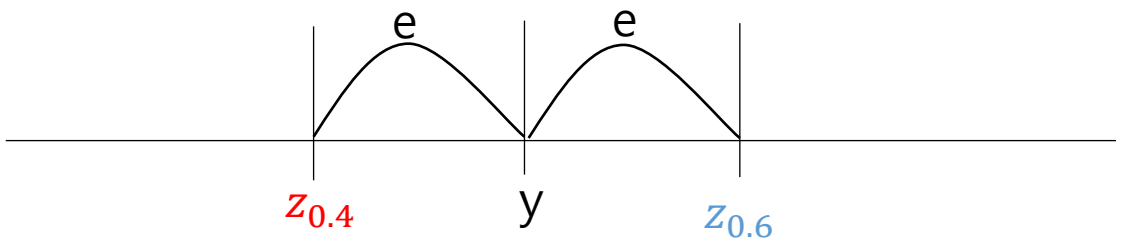
τ: 퀀타일 값 (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)

y: 실제 값

z: 퀀타일 예측값

Lτ: pinball loss 함수

- **Pinball Loss가 작아지는 상황**
  - 퀀타일 값이 작을수록 – 실제 값보다 작게 예측
  - 퀀타일 값이 클수록 – 실제 값보다 크게 예측



$L_{0.4}(y,z)$
$= (y-z)\tau$
$= e * 0.4$

$L_{0.6}(y,z)$
$= (y-z)(1-\tau)$
$= e*0.4$

Total Lower Pinball Loss = 0.8e

1. 대회 개요

- Pinball Loss
  - 심사 기준: Pinball Loss

$$L_\tau(y,z) = (y-z)\tau \quad \text{if } y \geq z$$
$$= (z-y)(1-\tau) \quad \text{if } z > y$$

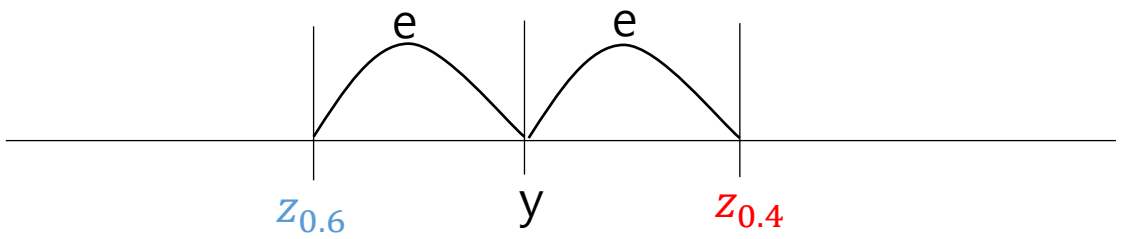
τ: 퀀타일 값 (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)

y: 실제 값

z: 퀀타일 예측값

Lτ: pinball loss 함수

- **Pinball Loss가 커지는 상황**
  - 퀀타일 값이 작을수록 – 실제 값보다 크게 예측
  - 퀀타일 값이 클수록 – 실제 값보다 작게 예측



$L_{0.6}(y,z)$
$= (y-z)(\tau)$
$= e*0.6$

$L_{0.4}(y,z)$
$= (y-z)(1-\tau)$
$= e * 0.6$

→ Total Higher Pinball Loss = 1.2e

1. 대회 개요

- Pinball Loss

• 심사 기준: Pinball Loss

$$L_\tau(y,z) = (y-z)\tau \quad \text{if } y \geq z$$
$$= (z-y)(1-\tau) \quad \text{if } z > y$$

τ: 퀀타일 값 (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)

y: 실제 값

z: 퀀타일 예측값

Lτ: pinball loss 함수

- 앞의 사례에서 보았듯이 똑같은 오차를 보이더라도 **아래의 조건을 만족시키면 더 적은 Loss가 나옴**
  - 퀀타일 값이 작을수록 – 실제 값보다 크게 예측
  - 퀀타일 값이 클수록 – 실제 값보다 작게 예측

Total Higher Pinball Loss = 1.2e
Total Lower Pinball Loss = 0.8e
Total Higher Pinball Loss – Total Lower Pinball Loss
= 같은 오차를 가졌지만, Pinball Loss의 특성으로 나타난 차이
= 1.2e – 0.8e = 0.4e
(예시를 두 퀀타일만 들어 설명했지만, 0.1부터 0.9까지 계산을 하면 위의 Pinball Loss차이는 더 커짐)

- 다시 말해, 이 대회에서 **Loss를 줄이기 위해서는 보수적으로 예측을 하는 것이 중요**.

2. 주어진 데이터

- train.csv : 훈련용 데이터 (1개 파일)
  - 3년(Day 0~ Day1094) 동안의 기상 데이터, 발전량(TARGET) 데이터

- test.csv : 정답용 데이터 (81개 파일)
  - 2년 동안의 기상 데이터, 발전량(TARGET) 데이터 제공
  - 각 파일(*.csv)은 7일(Day 0~ Day6) 동안의 기상 데이터, 발전량(TARGET) 데이터로 구성
  - 파일명 예시: 0.csv, 1.csv, 2.csv, …, 79.csv, 80.csv (순서는 랜덤이므로, 시계열 순서와 무관)
  - 각 파일의 7일(Day 0~ Day6) 동안의 데이터 전체 혹은 일부를 인풋으로 사용
  - 향후 2일(Day7 ~ Day8) 동안의 30분 간격의 발전량(TARGET)을 예측 (1일당 48개씩 총 96개 타임스텝에 대한 예측)

- sample_submission.csv : 정답제출 파일

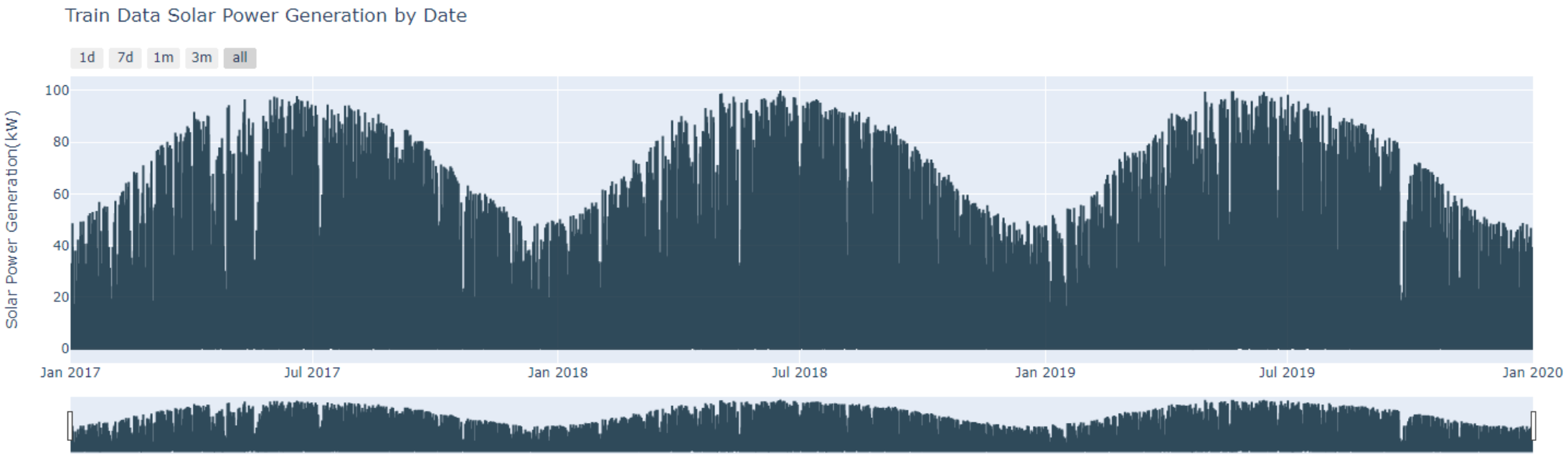
3. Columns

- Day - 날짜
- Hour+Minute - 시간
- DHI(Diffuse Horizontal Irradiance) - 수평면 산란일사량
- DNI(Direct Normal Irradiance) - 직달 일사량
  (대기에 산란, 흡수되지 않고, 지표면까지 수직으로 도달되는 일사량)
- WS(Wind Speed) - 풍속
- T(Tempertature)- 기온
- RH(Relative Humidity)- 상대습도

데이터 분석 설명

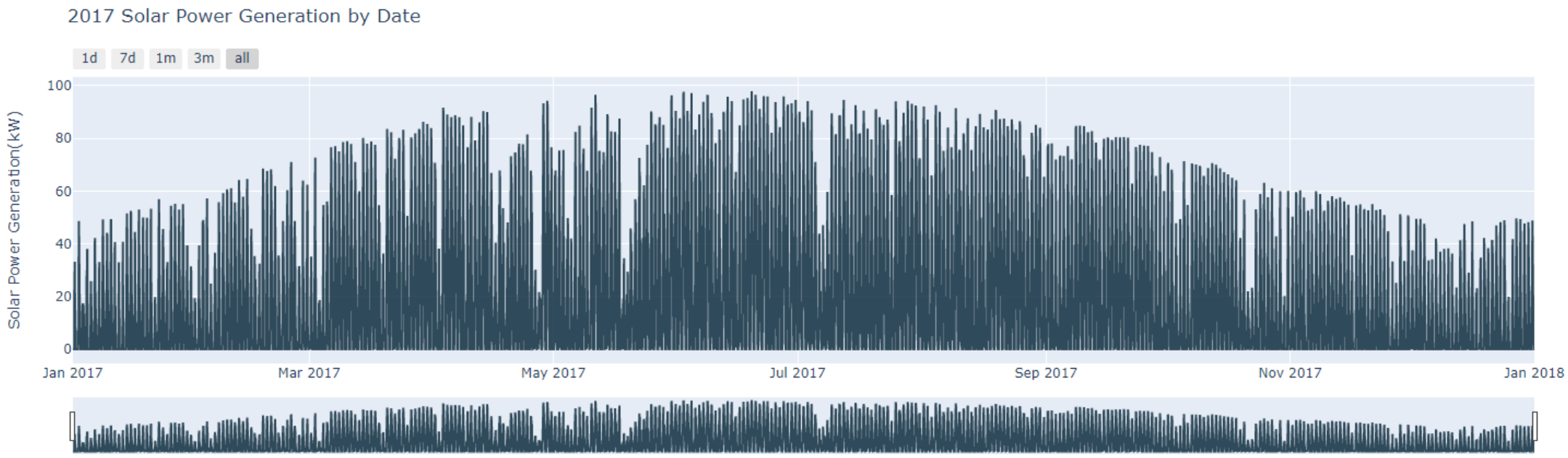
1. EDA & Visualization

1. Train의 자료 전체를 target에 대해 시각화 해보았을 때, **3년의 연속된 자료**를 받았음을 인식



1. EDA & Visualization

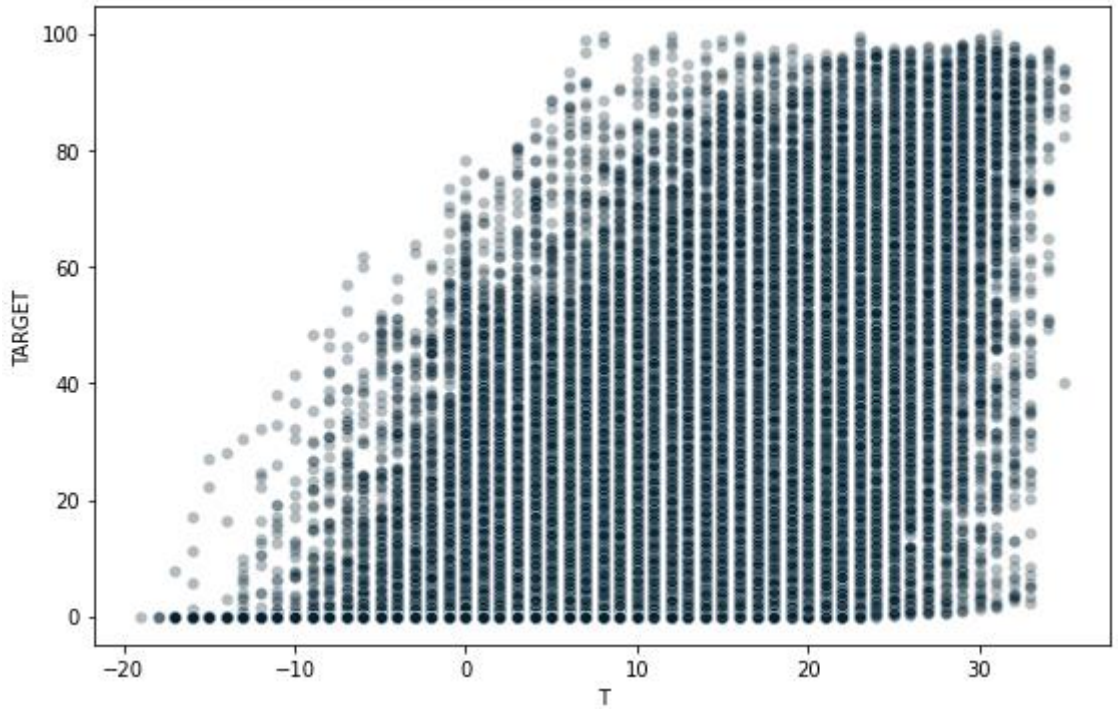
2. 임의로 365일씩 끊어서 시각화 했을 때, **1월부터 시작하는 데이터**를 받았음을 인식

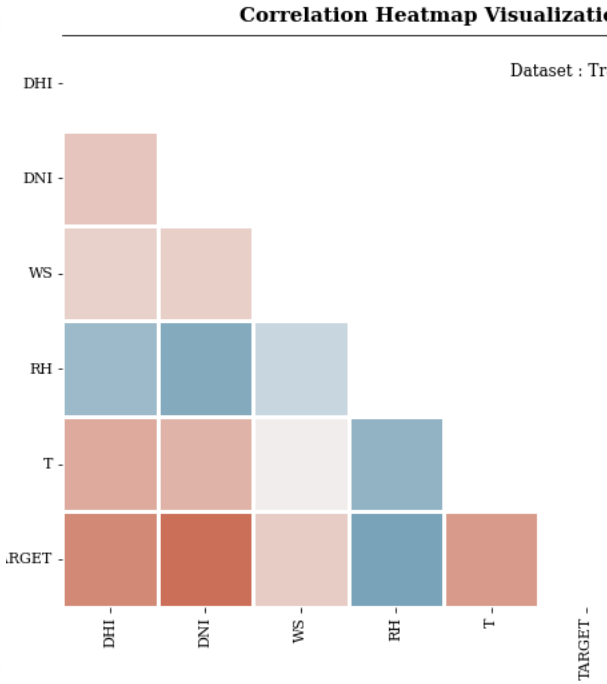


1. EDA & Visualization

3. T & Target은 어느 정도 양의 상관관계 존재

- T가 낮을 수록, T의 증가에 따라 Target의 상한선이 가파르게 올라가는 모습 관찰



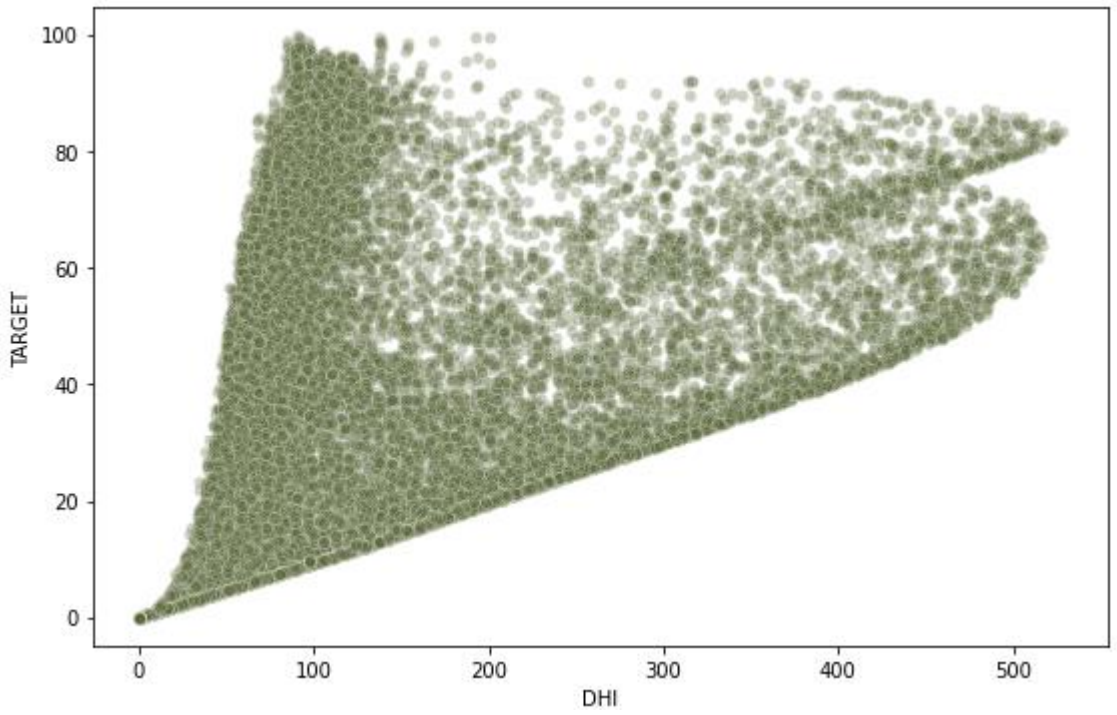


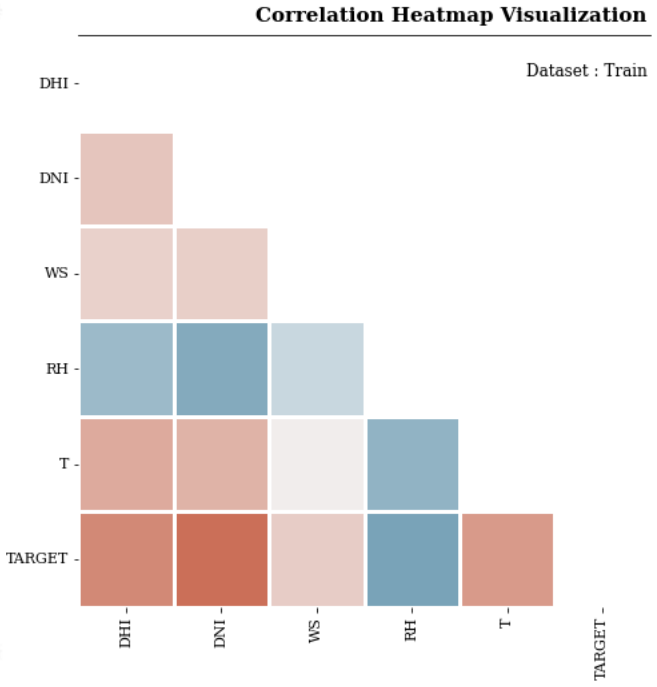
Corr Target aND Other Columns	
TARGET	1.000000
DNI	0.833547
DHI	0.666908
T	0.561990
WS	0.238521
RH	-0.677178

1. EDA & Visualization

4. DHI & Target은 어느 정도 양의 상관관계 존재

- DHI가 Target의 하한선을 이끌어 주는 듯한 모습



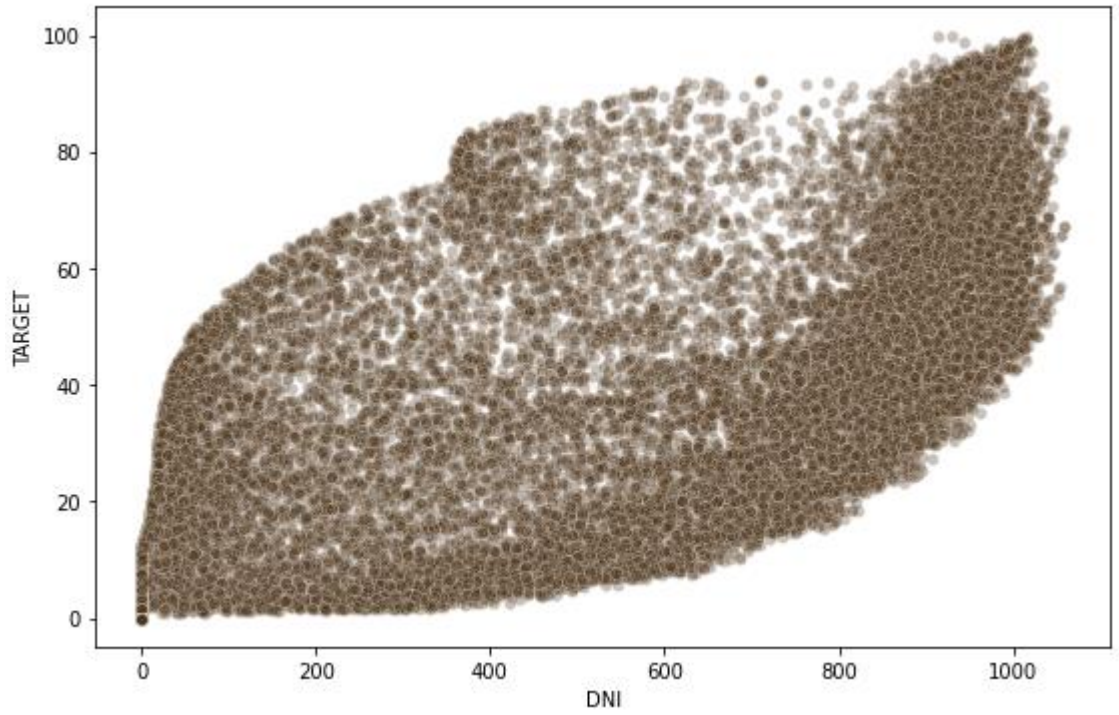


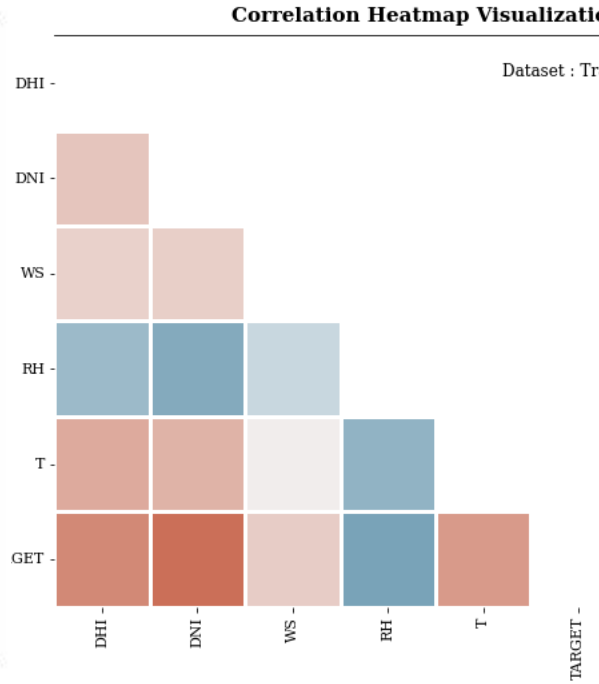
Corr Target aND Other Columns	
TARGET	1.000000
DNI	0.833547
DHI	0.666908
T	0.561990
WS	0.238521
RH	-0.677178

1. EDA & Visualization

4. DNI & Target은 강한 양의 상관관계 존재

- Target이 DHI보다 DNI의 움직임에 더 민감한 모습



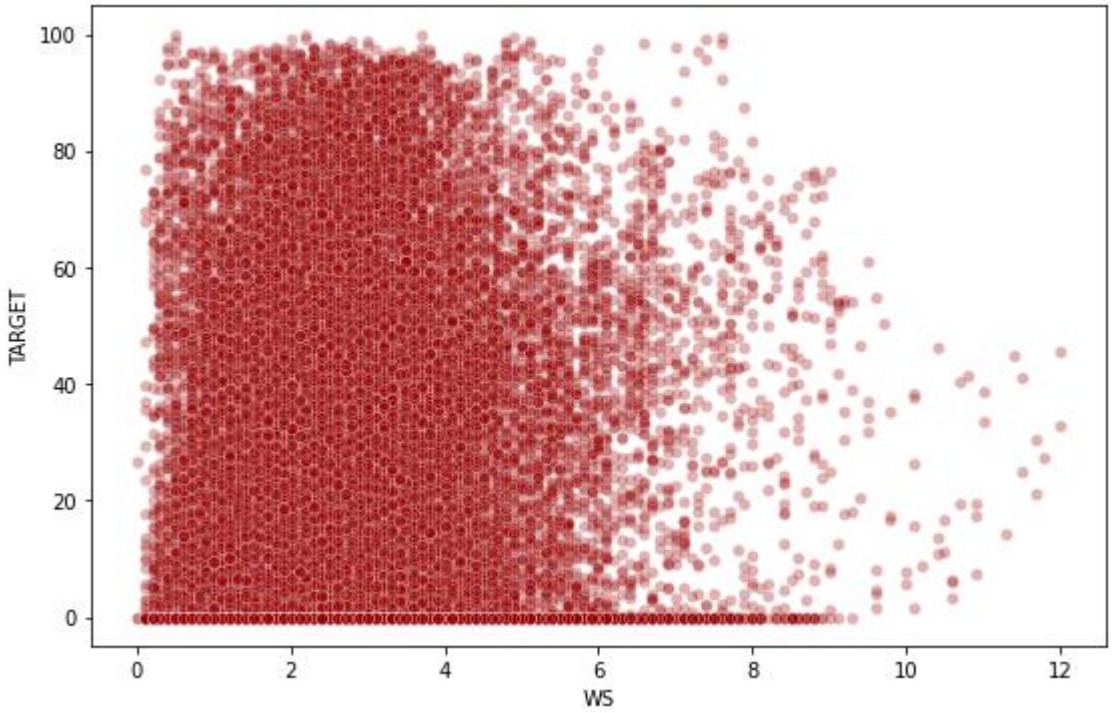


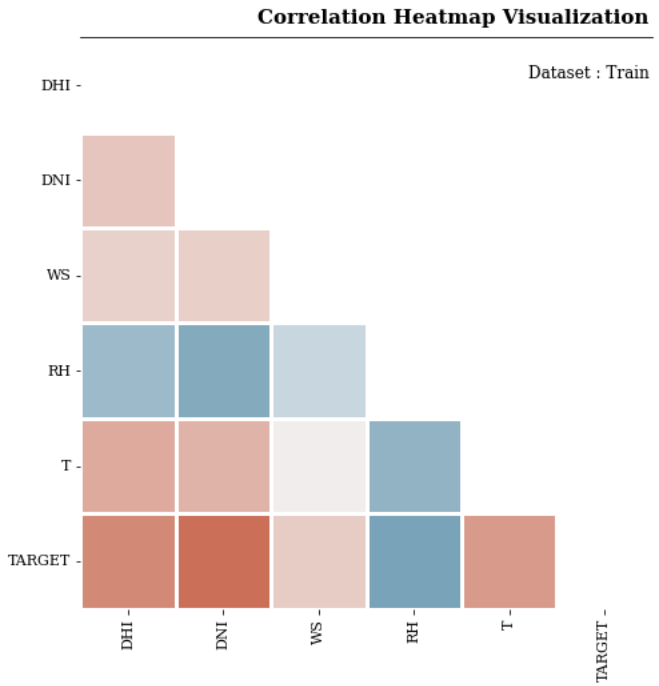

```
Corr Target aND Other Columns
TARGET     1.000000
DNI        0.833547
DHI        0.666908
T          0.561990
WS         0.238521
RH        -0.677178
```

1. EDA & Visualization

5. WS & Target은 약한 양의 상관관계 존재

- 태양광 발전량에 큰 영향은 없는 듯함.



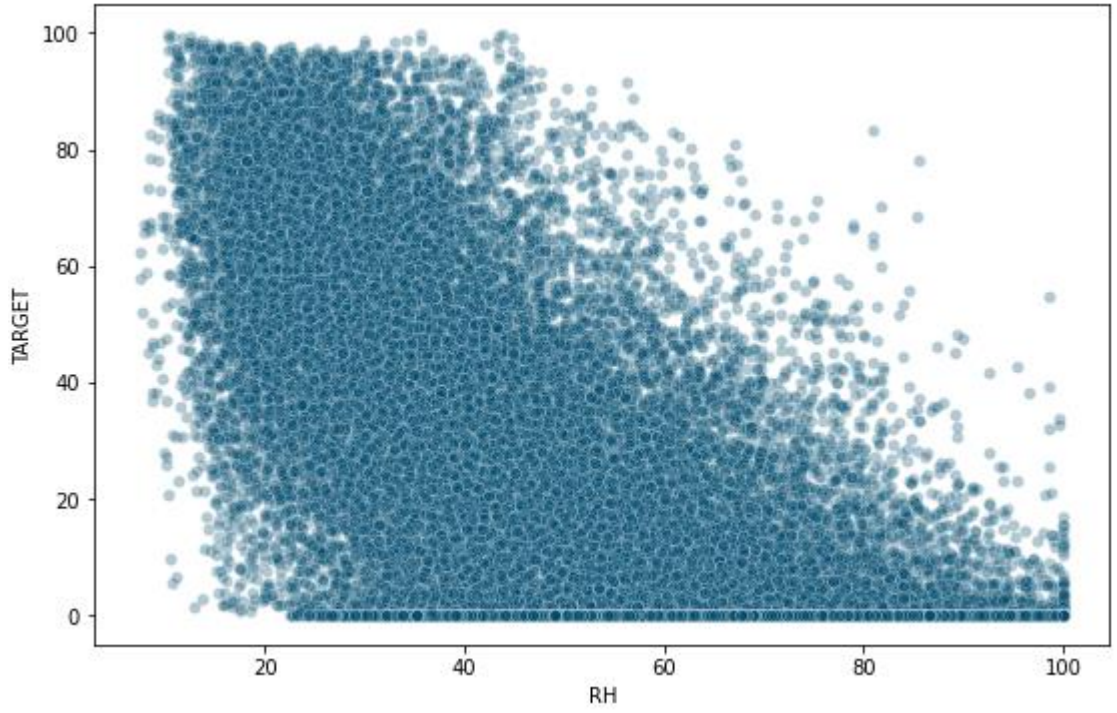


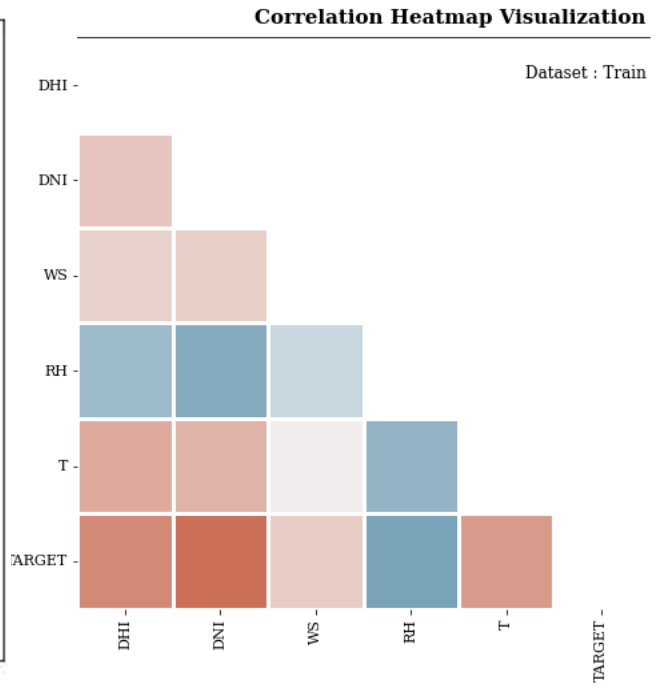
Corr Target aND Other Columns	
TARGET	1.000000
DNI	0.833547
DHI	0.666908
T	0.561990
WS	0.238521
RH	-0.677178

1. EDA & Visualization

6. RH & Target은 어느 정도 음의 상관관계 존재

- RH가 높아질수록 Target의 상한선이 떨어지는 것을 관찰
- RH가 습도이고, 이는 강수와 관련이 있어서 그렇다고 생각





Corr Target aND Other Columns	
TARGET	1.000000
DNI	0.833547
DHI	0.666908
T	0.561990
WS	0.238521
RH	-0.677178

2. Data Preprocessing & Feature Engineering

- 파생 피쳐 생성
  1. **두 일사량의 합**
     - DHI + DNI 피쳐 생성 – 두 변수가 동시에 높을 경우, Target에 큰 영향을 줄 것
  2. **태양의 유무**
     - 태양이 떠있다는 것을 알려줄 **명목형 변수**로 만듦.
     - 원래 값이 0이라면 return 0, 값이 0이 아니라면 return 1
     - Sun_TARGET ➔ Target == 0 or Target > 0
     - Sun_DHI ➔ DHI == 0 or DHI > 0
     - Sun_DNI ➔ DNI == 0 or DNI > 0
     - Sun_DHI_DNI ➔ DHI_DNI == 0 or DHI_DNI > 0

2. Data Preprocessing & Feature Engineering

3. **하루동안 해가 얼마나 떠있는가?**
   - 하루 데이터 중 태양의 유무 변수가 1인 행의 개수를 셈
   - Sun_TARGET_hour
   - Sun_DHI_hour
   - Sun_DNI_hour
   - Sun_DHI_DNI_hour

4. **DNI_DHI_plus변수 생성**
   - DHI와 DNI가 0이 아닌 행의 개수의 누적으로 해당 시점에 해가 떠있는 시간을 구하고 싶었음.
   - 하지만 낮 시간동안 두 값이 0인 행이 존재하여 중간에 처리가 애매해짐.
   - DHI와 DNI이 동시에 0인 경우는 TARGET이 0인 경우를 제외하면 없음
   - 따라서 **Sun_DHI와 Sun_DNI를 더하여 Sun_DHI_DNI_plus**를 만들고 이것의 누적합을 구하면 문제가 해결

2. Data Preprocessing & Feature Engineering

5. Sun_DHI_DNI_plus_hour

- 하루동안 해가 떠있는 시간을 DHI와 DNI로 측정해 합친 시간
- DHI와 DNI가 어느 시점에 모두 1이라면 Sun_DHI_DNI_plus_hour는 2의 값을 가질 수 있음

6. Accumulate

- **해가 뜨기 시작해서 해당 시점이 몇 번째 시점인가?**
- 태양의 고도와 관련이 있어 보였음
- Sun_TARGET_accumulate, Sun_DNI_accumulate, Sun_DHI_accumulate, Sun_DHI_DNI_accumulate, Sun_DHI_DNI_plus_accumulate

2. Data Preprocessing & Feature Engineering

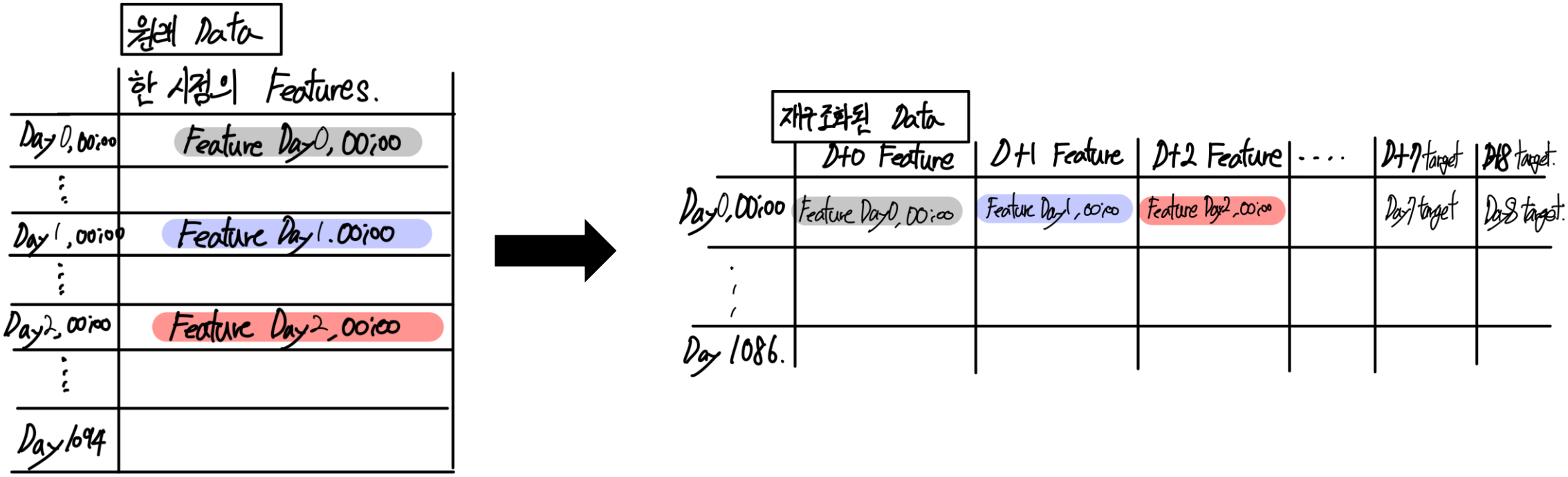
7. Acculmuate를 이용한 변수 생성

- **해당 시점이 해가 뜬 시점으로부터 몇 시간 후인가/하루 동안 해가 뜬 시간**
- 같은 오후 3시에 해가 떠있더라도, 해가 일찍 지는 겨울과 해가 긴 여름의 일사량이 다를 것이라고 생각
- Sun_TARGET_per, Sun_DNI_per, Sun_DHI_per, Sun_DHI_DNI_per, Sun_DHI_DNI_plus_per

2. Data Preprocessing & Feature Engineering

8. 데이터 재구조화

- 7일간의 데이터로 향후 2일을 예측해야함
- 데이터를 최대한 활용하기 위해, **하나의 row에 7일간의 피쳐 + Target(8,9일)을 넣음**

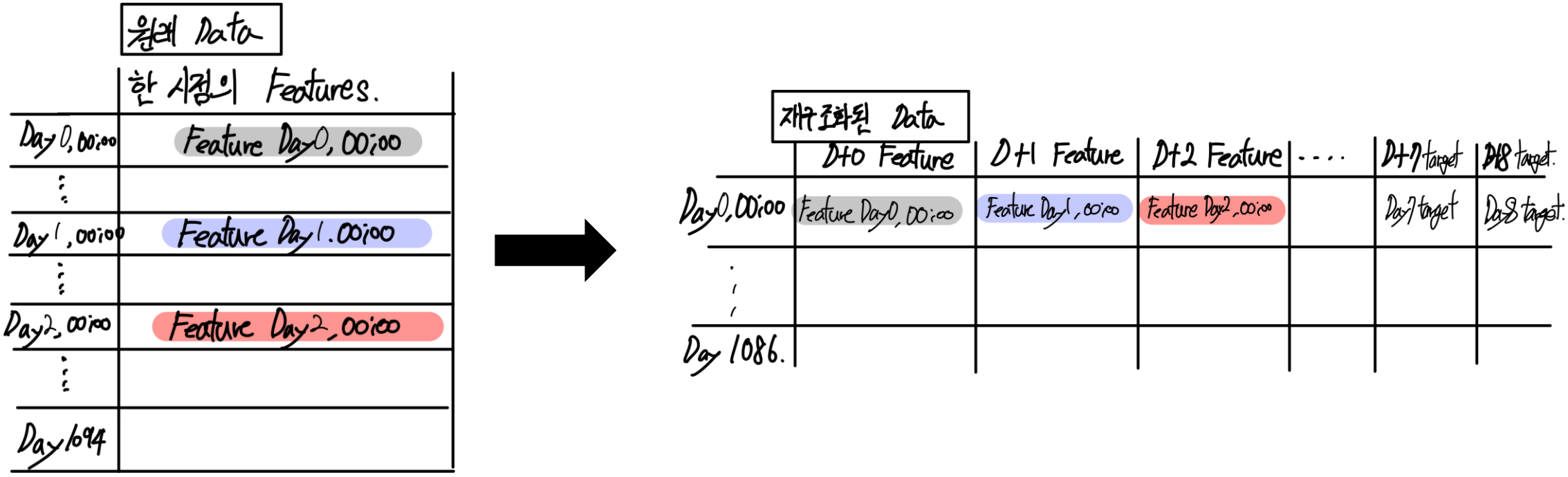


2. Data Preprocessing & Feature Engineering

8. 데이터 재구조화

- 이를 통해 **더 많은 데이터를 학습**에 사용

➔**성능 향상과 generalization의 향상**을 기대



3. Hyperparameter Optimization & Modeling

1. 데이터 불균형 해결을 위한 Trick

```
1 print("데이터에서 TARGET이 0인 비율 :",len(train[train.TARGET==0])/len(train)*100,"%")
```

```
데이터에서 TARGET이 0인 비율 : 50.7229832572298 35 %
```

- 데이터에서 **TARGET이 0을 차지하는 비율은 50%**

➔ TARGET이 0이 아니면 0이외의 양수의 실수임으로 **심한 불균형 데이터**

➔ 이러한 상황에서 train_test_split을 이용해 랜덤으로 학습세트를 나누면,
Target이 0인 경우와 0이 아닌 경우를 골고루 학습할 수 없을 가능성이 높음

3. Hyperparameter Optimization & Modeling

1. 데이터 불균형 해결을 위한 Trick

- **TARGET을 카테고리 변수로 바꾸기(Target_cate)**
  - TARGET을 0에서부터 1단위로 끊어 **카테고리 변수로 Labeling**

    ex) Target = 0 → Target_cate = 0

    Target = 1.2 → Target_cate = 1 .....

- **train_test_split의 stratified옵션 사용**
  - Stratified 옵션은 원래 분류 문제를 해결할 때, Target이 클래스별로 골고루 train과 test세트에 포함되게 만들 때 사용
  - 하지만 이 경우에 위처럼 **회귀 문제에서도 Target을 분류문제의 정답처럼 만들고 stratified옵션을 사용하면, train과 test세트에 골고루 target값들이 들어갈 수 있다고 생각함**

  ➔ 이 방법을 사용하여 **성능 향상을 이끌어냄**

3. Hyperparameter Optimization & Modeling

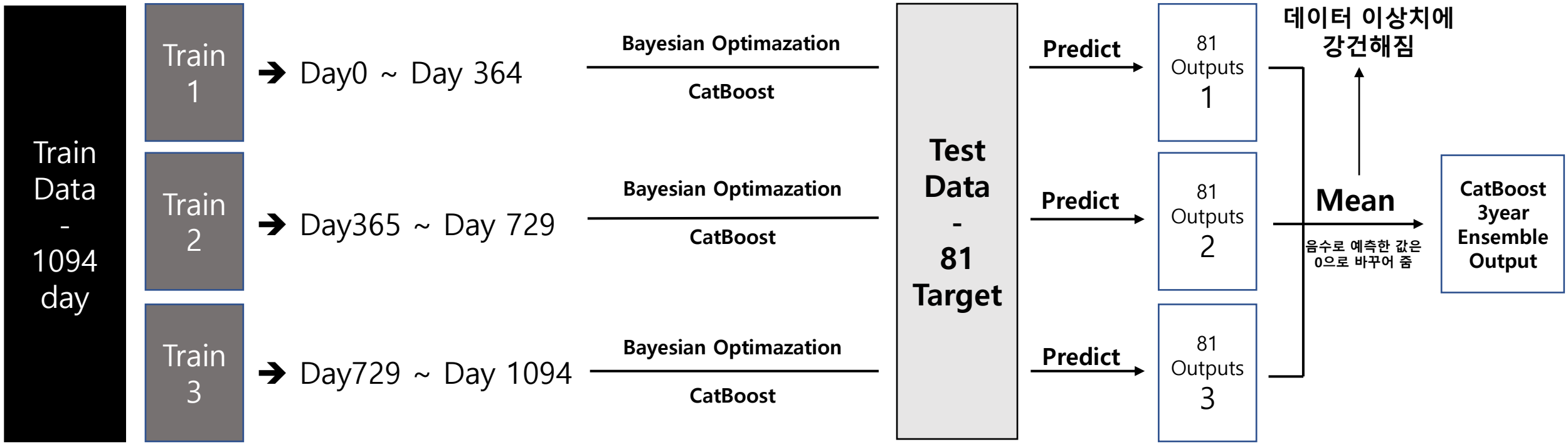
2. Train 데이터 분할, 하이퍼 파리미터 최적화, 모델링 및 앙상블

(LGBM도 아래와 동일 방법으로 LGBM 3year Ensemble Output을 얻음)

- 각기 다른 Train 세트를 학습한 모델의 예측값 앙상블, 다른 모델로 학습한 3년 앙상블 값의 앙상블

→ 더 강건하고, 이상치에 강한 모델을 만들기 위해 사용

+ LGBM의 성능과 CatBoost의 카테고리변수의 특화 이점을 누리기 위함



3. Hyperparameter Optimization & Modeling

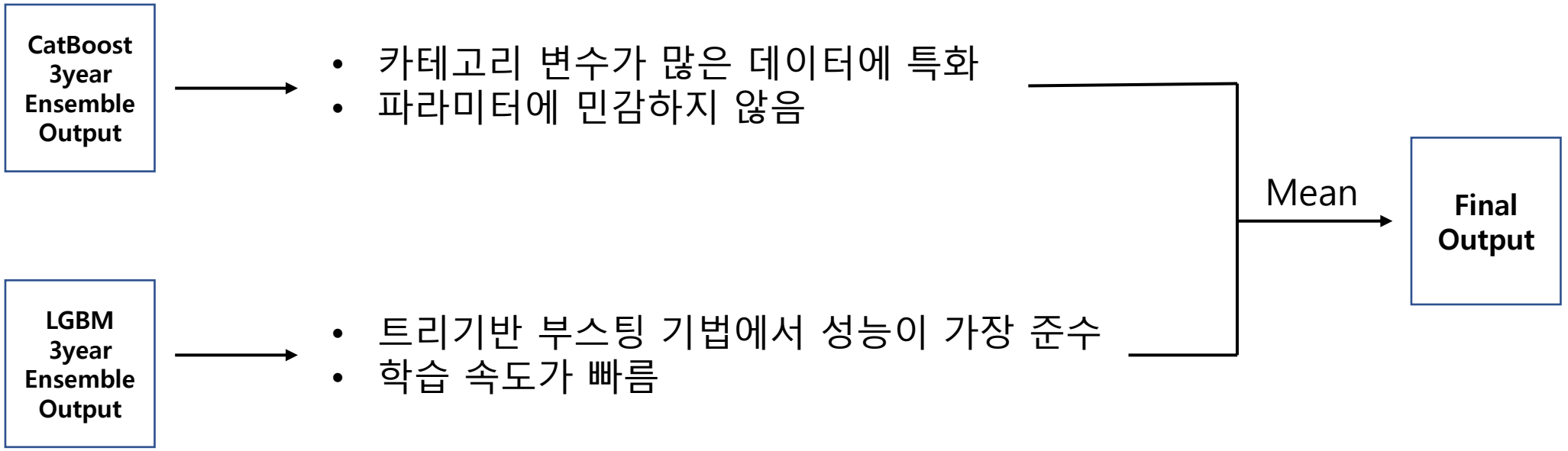
2. Train 데이터 분할, 하이퍼 파리미터 최적화, 모델링 및 앙상블

(LGBM도 아래와 동일 방법으로 LGBM 3year Ensemble Output을 얻음)

- 각기 다른 Train 세트를 학습한 모델의 예측값 앙상블, 다른 모델로 학습한 3년 앙상블 값의 앙상블

→ 더 강건하고, 이상치에 강한 모델을 만들기 위해 사용

+ LGBM의 성능과 CatBoost의 카테고리변수의 특화 이점을 누리기 위함



4. 최종 성능 및 순위

Public Score

#	팀	팀 멤버	점수	제출수	등록일
39	휴먼러닝	hk	1.82341	66	10달 전

Private Score

#	팀	팀 멤버	최종점수	제출수	등록일
62	휴먼러닝	hk	2.08297	66	10달 전