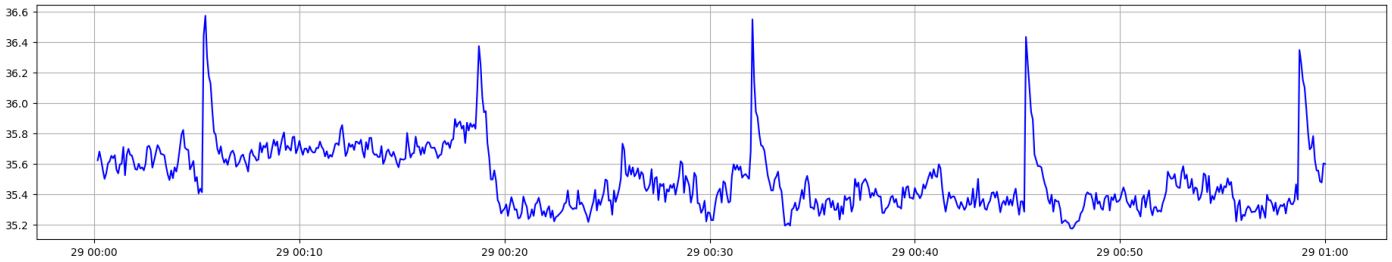
# 이상데이터 추가과정

## 시험 준비를 위한 이상데이터 추가

- TTA 등 다양한 시험의 정확도를 확인하기 위해 정상적인 데이터에 이상데이터를 추가해 다양한 패턴에 대한 모델의 성능을 확인하기 위함
- 시계열데이터의 다양한 특성을 반영하여 이상데이터를 추가하였음

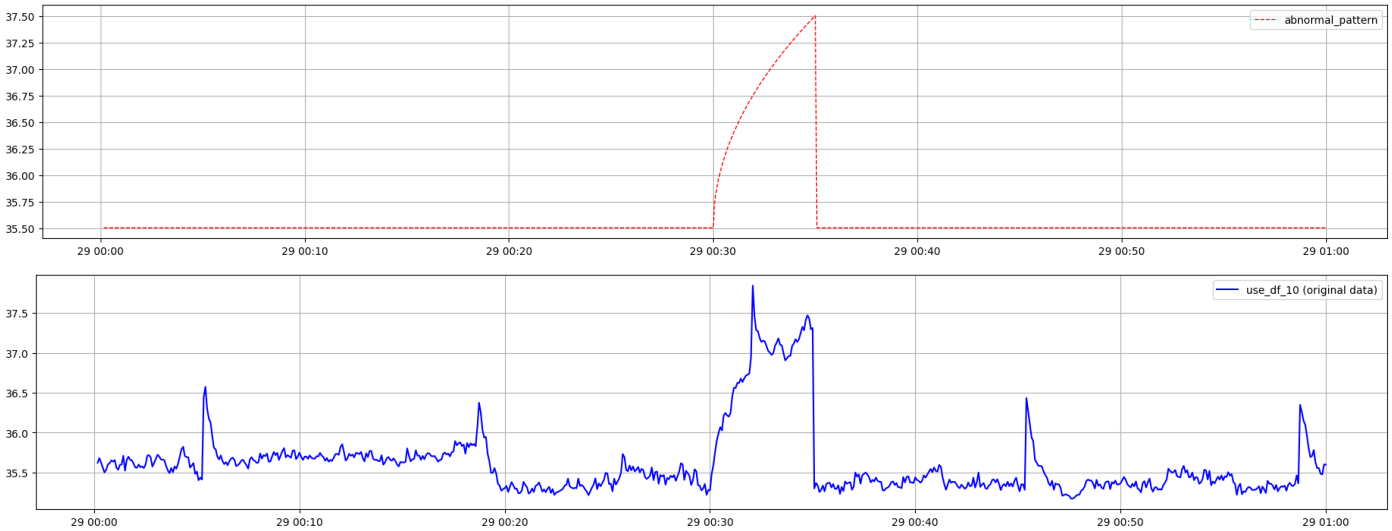
## 기존 데이터 확인



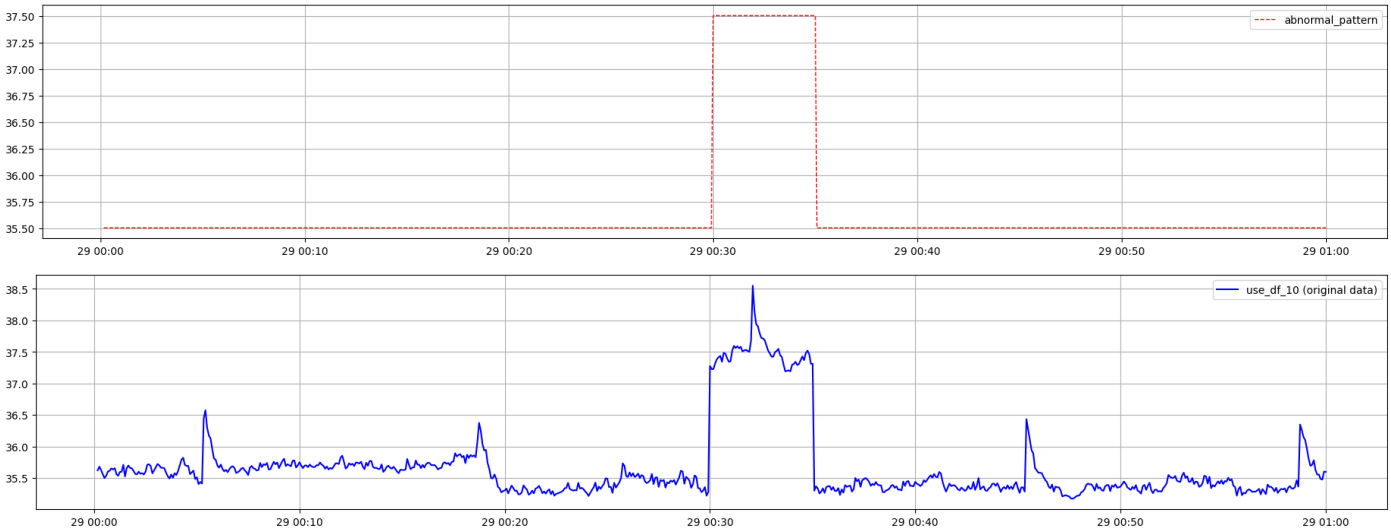

- 데이터는 열화상 온도데이터를 활용하였으며 1 시간 간격의 데이터를 사용

## 이상 데이터 종류

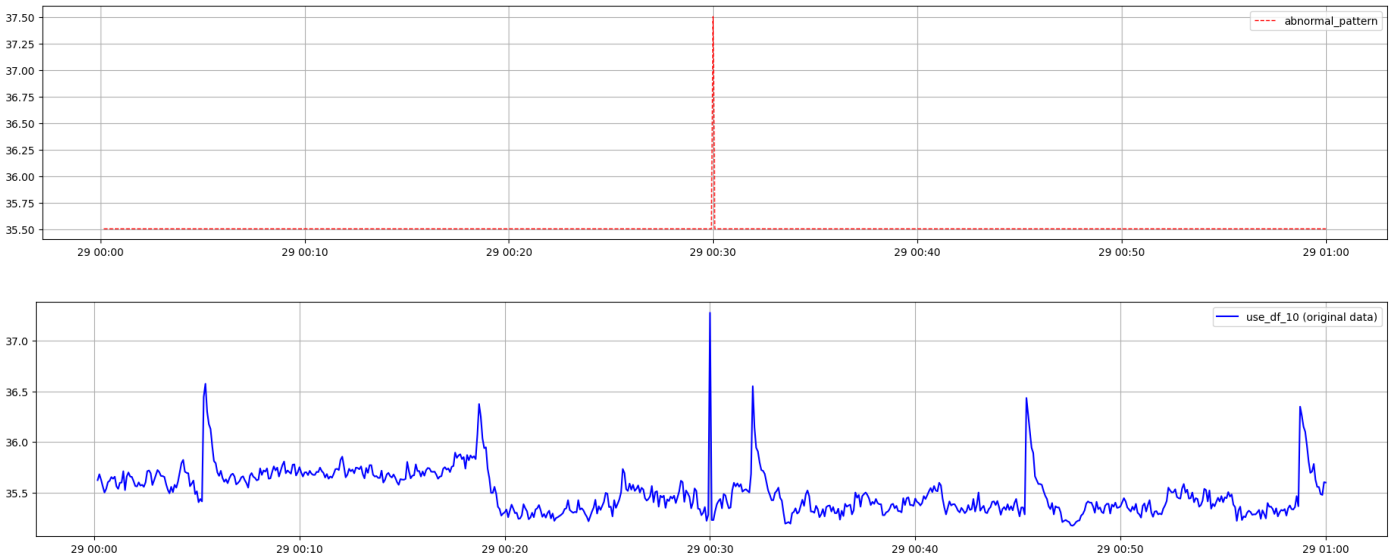
1. 값이 점차 증가했다가 뚝 떨어지는 패턴(기계 오류 발생시 가동중지 시키는 패턴)



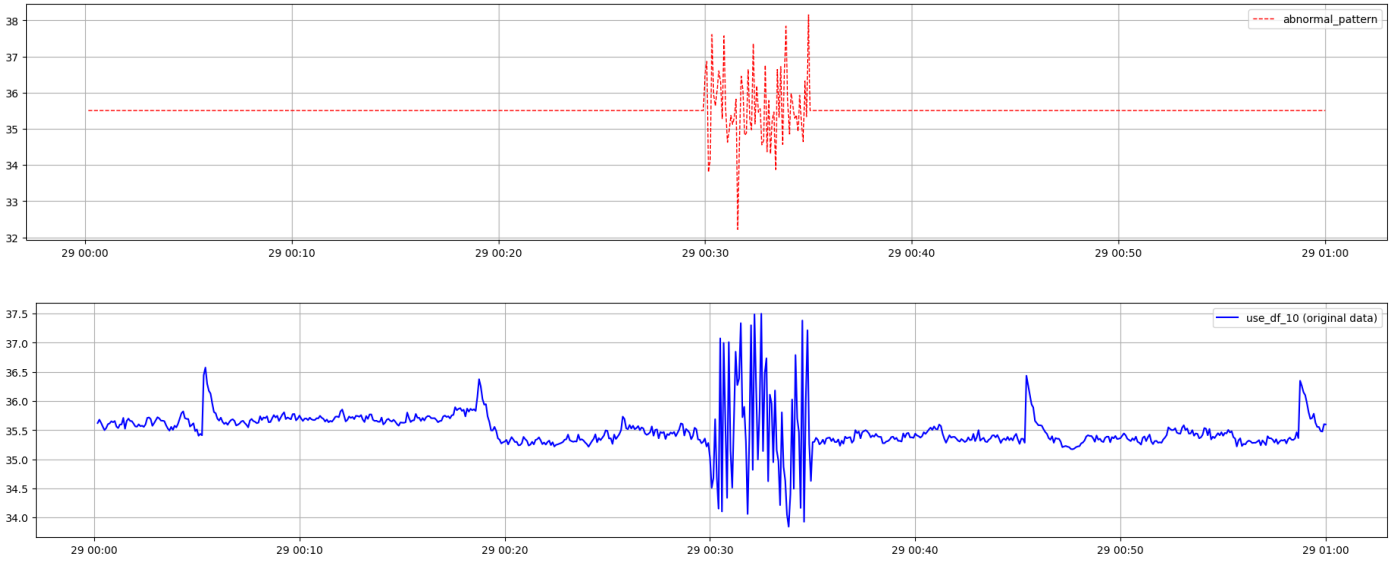
## 2. 특정 구간에서 값이 상승하는 패턴(센서데이터 오류, 외부환경 영향)



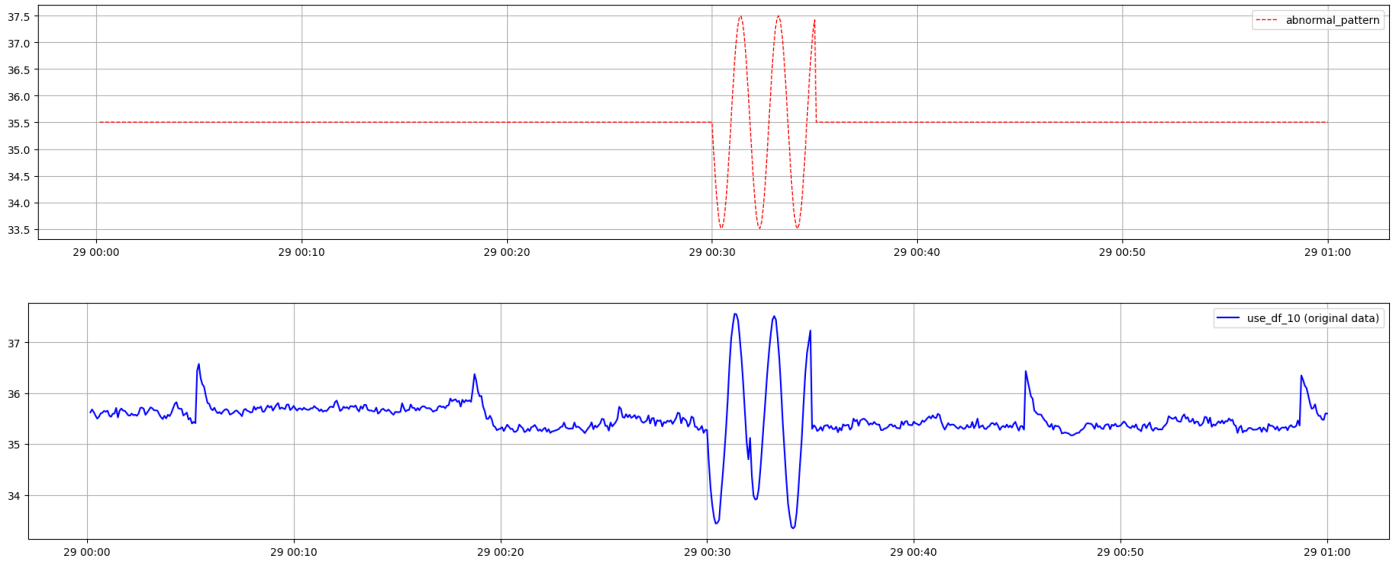

## 3. 한 시점에 값이 상승하는 패턴(센서데이터 오류, 외부환경 영향)




## 4. 노이즈 발생(압력신호에서 발생)



5. 값이 주기적으로 변함(노이즈 신호와 동일)




## 이상 데이터 추가 코드

```python
# 데이터가 서서히 증가함
def increase_data_gradually(df, start_time, end_time, increase_factor):
    use_df = df.copy()
    selected_data = use_df[start_time:end_time]
    increased_values = np.sqrt(np.linspace(0, 1, len(selected_data))) * increase_factor
    use_df.loc[start_time:end_time, '_value'] += increased_values
    return use_df

# 데이터의 특정구간 전체 값이 상승
def add_fixed_value(df, start_time, end_time, fixed_value):
    use_df = df.copy()
    use_df['_value'].loc[start_time:end_time] += fixed_value
    return use_df

# 데이터의 특정 한 기간만 값이 상승
def add_data_spike(df, spike_time, spike_value):
    use_df = df.copy()
    use_df.loc[spike_time, '_value'] += spike_value
    return use_df

# 데이터에 노이즈 추가하기
def add_noise_to_data(df, start_time, end_time, mean=0, std_dev=1):
    use_df = df.copy()
    selected_data = use_df[start_time:end_time]
    noise = np.random.normal(mean, std_dev, len(selected_data))
    use_df.loc[start_time:end_time, '_value'] += noise
    return use_df

# 데이터에 노이즈 추가하기 2번째 방법
def add_random_values(df, start_time, end_time, threshold):
    use_df = df.copy()
    random_values = np.random.normal(loc=0, scale=threshold, size=use_df['_value'].loc[start_time:end_time].shape[0])
    use_df['_value'].loc[start_time:end_time] += random_values
    return use_df

# 데이터에 주기성있는 노이즈 추가하기
def add_periodic_values(df, start_time, end_time, threshold, frequency):
    use_df = df.copy()
    time_indices = np.arange(use_df['_value'].loc[start_time:end_time].shape[0])
    periodic_values = np.sin(frequency * time_indices) * threshold * 2
    use_df['_value'].loc[start_time:end_time] += periodic_values
    return use_df
```