각 feature 값을 pandas에 저장하고, 범위별로 양자화했으면 좋겠어.

date는 연도-월별로 나눌꺼야. (일자는 사용하지 않아)

avg (temperature),max (temperature),min (temperature) 3가지 features는 5 단위로 쪼갤게. 예를 들어 0~5그룹, ....

avg (humidity),max (humidity),min (humidity) 3가지 features는 5단위로 쪼갤게.

power는 10단위로 쪼갤게.

label은 정상, 비정상이고 정상일 경우 0으로, 비정상인 경우 1로 표시해야 해.

avg(wind speed),avg(local pressure),avg(cloud cover) 3가지 features는 0.5 단위로 쪼갤게.

sun\_rise,sun\_set 2가지 features는 시간:분 단위로 되어 있어. 이거는 분을 10분 단위로 쪼갤게.

sun\_max는 시간:분:초 단위로 되어 있어. 이거는 분을 5분 단위로 쪼갤게.

day\_duration,night\_duration 2가지 features는 'n days 시간:분:초' 단위로 되어 있어. n은 무시하고, 분을 5분 단위로 쪼갤게.

day\_avg\_temperature\_app,night\_avg\_temperature\_app 2가지 features는 5 단위로 쪼갤게.

Nitrogen Dioxide Concentration (ppm),Ozone Concentration (ppm) 2가지 features는 0.005 단위로 쪼갤게.

Carbon Monoxide Concentration (ppm)는 0.1 단위로 쪼갤게.

Sulfur Dioxide Concentration (ppm)는 0.001 단위로 쪼갤게.

Particulate Matter (㎍/㎥),Fine Particulate Matter (㎍/㎥),AQI 3가지 features는 10 단위로 쪼갤게.

이렇게 쪼개서 pandas에 각 features 값 대신 그룹명으로 표기되도록 해 줘. 그룹명은 작은 값에서 큰 값으로 1부터 시작하는 자연수로 정하는 코드를 작성해 줘.

참고로 csv 파일 일부를 제시할게:

date,avg (temperature),max (temperature),min (temperature),avg (humidity),max (humidity),min (humidity),power,label,avg(wind speed),avg(local pressure),avg(cloud cover),avg(ground temperature),avg(evaporation volume),sun\_rise,sun\_max,sun\_set,day\_duration,night\_duration,day\_avg\_temperature\_app,night\_avg\_temperature\_app,Nitrogen Dioxide Concentration (ppm),Ozone Concentration (ppm),Carbon Monoxide Concentration (ppm),Sulfur Dioxide Concentration (ppm),Particulate Matter (㎍/㎥),Fine Particulate Matter (㎍/㎥),AQI

2021-01-01,-4.45,1.4,-9.7,64.87,75,52,242.75,0,2.7,1019.8,1.3,6.3,1.6,7:47,12:35:14,17:24,0 days 09:37:00,0 days 14:23:00,-1.525,-7.074999999999999,0.023,0.013,0.5,0.003,32.0,13.0,93.0

2021-01-02,-4.79,-1.5,-8.3,38.43,50,31,229.78,0,2.5,1021.0,0.0,6.2,1.3,7:47,12:35:42,17:25,0 days 09:38:00,0 days 14:22:00,-3.145,-6.545,0.015,0.019,0.4,0.003,44.0,9.0,122.0

2021-01-03,-5.59,-2.3,-9.0,44.39,58,30,252.5,0,1.8,1019.5,0.0,6.1,1.3,7:47,12:36:10,17:25,0 days 09:38:00,0 days 14:22:00,-3.945,-7.295,0.017,0.017,0.4,0.003,46.0,11.0,127.0

2021-01-04,-3.67,0.2,-8.3,48.91,69,40,298.45,0,1.9,1019.5,1.9,6.0,1.5,7:47,12:36:38,17:26,0 days 09:39:00,0 days 14:21:00,-1.735,-5.985,0.033,0.008,0.5,0.003,47.0,19.0,129.0

2021-01-05,-4.96,-2.5,-9.5,53.43,82,31,333.4,0,1.6,1019.6,3.6,5.9,0.9,7:47,12:37:05,17:27,0 days 09:40:00,0 days 14:20:00,-3.73,-7.23,0.018,0.018,0.4,0.003,33.0,12.0,95.0

**1. 이거는 각 feature의 연관성보다 feature 그룹과 label 간의 연관성이 중요하다 생각하기에 apriori의 강점을 포기하고 더 빠르고 compact한 FP-Growth로 진행**

**2. anomal인 부분을 찾아내는 것이 중요하므로 recall 보다는 precision이 중요하다 판단**

**양자화된 구간을 추출하는 과정**

양자화된 구간을 추출하는 과정은 training.csv에 포함된 데이터의 **값 범위**를 이용하여, 우리가 양자화를 할 때 사용할 **구간(bin)을 계산**하는 것입니다.

예를 들어, training.csv에서 'avg (temperature)' 컬럼이 있고, 이 컬럼의 값이 10에서 30 사이일 수 있다고 가정해보겠습니다. 이 값들을 구간(예: 5 단위)으로 나누면, 구간이 [10, 15), [15, 20), [20, 25), [25, 30] 이렇게 나뉘게 됩니다.

get\_bins\_from\_training 함수는 **training.csv에서 값의 최소값과 최대값을 구한 후**, 그 범위에 맞게 구간을 나누는 역할을 합니다.

**양자화 구간 추출 방법**

여기서 핵심은 **값의 최소값**과 **최대값**을 이용해서 양자화 구간을 나눈다는 점입니다. 이 구간은 training.csv에서 얻은 정보에 기반하여 testing.csv에도 동일한 구간을 적용하려는 목적입니다.

**예시로 보겠습니다.**

python

코드 복사

def get\_bins\_from\_training(df\_train, feature, step):

"""Training 데이터에서 feature의 구간을 얻어옵니다."""

min\_val = df\_train[feature].min() # 최소값

max\_val = df\_train[feature].max() # 최대값

bins = np.arange(min\_val, max\_val + step, step) # step 크기로 구간을 나누기

return bins

이 함수는 training.csv에서 특정 컬럼(예: avg (temperature))의 최소값과 최대값을 구하고, 그 범위에 맞춰 구간을 나눕니다. 예를 들어, avg (temperature)가 10에서 30 사이라면, 구간을 np.arange(10, 30+5, 5)로 만들 수 있습니다. 이렇게 하면 [10, 15), [15, 20), [20, 25), [25, 30] 구간이 나옵니다.

**예시**

**training.csv의 'avg (temperature)' 컬럼의 값이 다음과 같다고 가정:**

* 최소값: 10
* 최대값: 30

python

코드 복사

temperature\_bins = get\_bins\_from\_training(df\_train, 'avg (temperature)', 5)

이렇게 하면 temperature\_bins는 [10, 15, 20, 25, 30]과 같은 값으로 반환됩니다. 이 구간을 testing.csv의 같은 컬럼에 적용하여, 값들을 해당 구간으로 분류할 수 있습니다.

**전체 흐름**

1. **training.csv에서 구간 추출**: get\_bins\_from\_training() 함수를 사용하여 각 특성(feature)별로 구간을 추출합니다.
2. **testing.csv에 같은 구간 적용**: 추출한 구간을 testing.csv에 동일하게 적용하여, 양자화된 값을 할당합니다.

따라서, training.csv에서 구간을 추출하는 방법은 **단순히 데이터의 최솟값과 최댓값을 기준으로 범위를 설정**하는 것입니다. training.csv의 데이터 범위가 testing.csv에서도 동일한 방식으로 적용되기 때문에, **양자화된 구간을 유지하면서 새로운 데이터에 동일한 기준을 적용**할 수 있습니다.

이해가 되셨나요? 추가적으로 궁금한 부분이 있으면 언제든지 질문해 주세요!

1. 이걸 우선 pandas를 통해 label=1인 값만 모읍니다.

2. FP-Growth를 적용해서 label=1인 그룹을 딕셔너리로 모읍니다. feature:value의 형태로 모읍니다. confidence=0.7로 가정합니다.