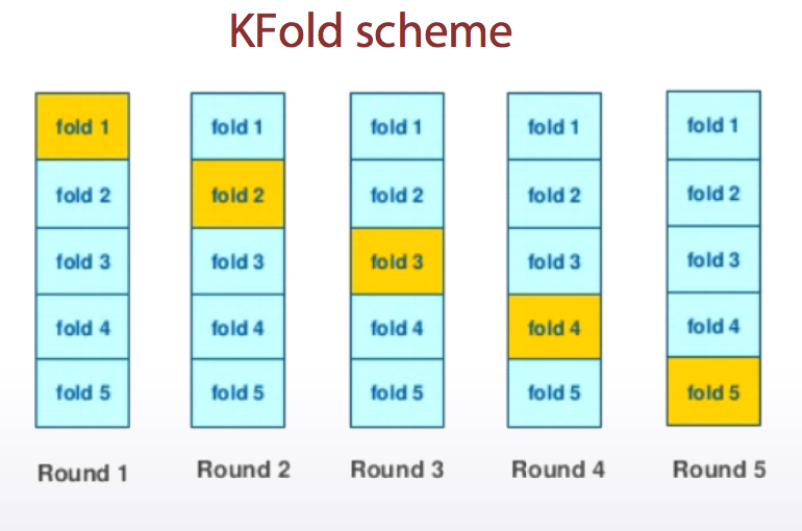
Mean_Enconding의 Overfitting 방지하기 위한 방법은 무엇이 있을까?

또한 강의안에서 제시한 카테고리 변수 enconding들 이외에 다른 변환방법들이 무엇이 있을까 조사하고 정리해보기

Mean Encoding 에 대해서 좀 더 자세한 정보를 획득하였다.
Overfitting이 발생하는 이유는 encoding된 값에 target에 대한 정보가 포함되어있다는 것 이외에도 다르면서 비슷한 이유가 있는데, 바로 label의 대표값을 trainset의 하나의 mean으로만 사용한다는 점이다.
예를 들어서 trainset에서는 국가라는 label에 USA가 50개, Europe이 5개 존재하는데에 반해서, testset에서는 각각 20개, 20개씩 존재한다고 한다면, trainset에서 Europe의 경우 평균을 /5를 통해서 구하는데 이것이 testset에서의 20개를 반영할 것인가에 대한 의문점이 생기는 것이다.

이에 따른 해결책이 각각 존재한다.

1. CV (Cross Validation) Loop
    - Trainset 안에서 Cross validation을 통해서 encoding의 값의 다양화를 얻어내는 것이다. 예를 들어서 다음과 같은 그림이 있다고 하자.



    - fold를 생성해주면 각각의 Round마다 5개의 fold가 생긴다. Round 1을 집중적으로 살펴보자. Fold 1을 validation data라고 설정을 한 뒤에 나머지 training data인 2,3,4,5 들의 각각에 대해서 mean encoding을 해준다.
    - 그 뒤에, 해당 encoding값들을 fold 1에 기록한다.
    - 그러한 방식으로 모든 round를 진행해주게 된다면 encoding의 값들을 이전에 비해서 다양하게 만들어줄 수 있다.
    - 이에 따라 트리가 만들어질 때, 더 세분화되는 효과를 거둘 수 있다.
    - 또한, Cross Validation이 이루어지기 때문에 data leakage를 줄일 수 있다는 장점이 있다.
2. Expanding mean
    - 위에 CV Loop에선 fold의 숫자가 5개에서 10개정도로 권장이 되는데 이것보다 encoding 값을 더 많이 만드는 방법이 없을까 해서 나오게 된 방법이다.
    - 누적합 함수 cumsum()과 누적개수를 세주는 함수인 cumcount()를 이용하여 위에서부터 차례대로 target의 평균을 내는 방식이다.

- 예를 들어 dataset이 다음과 같다고 가정해보자.

	feature	feature_label	feature_mean	target
0	Moscow	1	0.4	0
1	Moscow	1	0.4	1
2	Moscow	1	0.4	1
3	Moscow	1	0.4	0
4	Moscow	1	0.4	0
5	Tver	2	0.8	1
6	Tver	2	0.8	1
7	Tver	2	0.8	1
8	Tver	2	0.8	0

- 기존에 배운 mean encoding이라면 feature_mean에 쓰여진 것 처럼 되어있을 것이다.
- 우리가 Expanding mean을 사용하게 된다면 해당 row를 기준으로 **이전**에 같은 feature를 가진 row들의 target의 합을 row의 개수로 나눈 값이 된다.

$$Expanding\,Mean = \frac{\sum target_f}{count_f} \quad (f == feature)$$

- 위의 식을 이용하여 계산하게 되면 위의 dataset은 다음과 같아질 것이다.

index	feature	feature_label	feature_mean	target	expanding_mean
0	Moscow	1	0.4	0	0
1	Moscow	1	0.4	1	0
2	Moscow	1	0.4	1	0.5
3	Moscow	1	0.4	0	0.33
4	Moscow	1	0.4	0	0.5
5	Tver	2	0.8	1	0
6	Tver	2	0.8	1	1
7	Tver	2	0.8	1	1
8	Tver	2	0.8	0	1

- 이러한 방식으로 encoded 된 값의 특성 자체는 보존하면서 값 자체는 많아지게 만들 수 있다.
- 그러나 이 표를 보아도 알겠지만 이렇게 만들어낸 값이 유용한 값인지는 알 수 없다.

Reference

https://dailyheumsi.tistory.com/120

https://www.kaggle.com/vprokopev/mean-likelihood-encodings-a-comprehensive-study

https://zzsza.github.io/data/2018/09/08/feature-engineering/