범주형 자료분석 3주차 교안

2주차에서 로지스틱 회귀를 포함한 여러 GLM 모형들에 대해 알아봤습니다. 3주차에서는 분류 모델들을 평가하는 여러 지표에 대해서 알아보겠습니다. 더불어 불균형한 클래스 불균형 문제를 해결하는 여러 샘플링 방법과 범주형 자료를 수치형으로 변환하는 인코딩 방법들에 대해서도 배워볼 거예요!

3주차 클린업까지 달려오신 범주팀 여러분들 정말 고생 많으셨습니다. 마지막까지 힘내봐요~

- 목차 -

Table of Contents

1. 혼동 행렬 (Confusion Matrix)

■ 혼동 행렬이란?

혼동행렬(Confusion matrix)이란 분류 모델의 성능을 평가할 때 사용되는 지표로, 예측값($\hat{Y}$)이 실제 관측값(Y)을 얼마나 정확히 예측했는지를 보여주는 행렬이다.

		관측값(Y)	
		$Y=1$	$Y=0$
예측값($\hat{Y}$)	$\hat{Y}=1$	TP	FN
	$\hat{Y}=0$	FP	TN

앞의 T(True)와 F(False)는 실제와 예측이 같은지 혹은 다른지를 나타내고, 뒤의 P(Positive)와 N(Negative)는 예측을 긍정 혹은 부정이라 했는지에 대한 여부를 나타낸다.

예시를 하나 들어보자.

- TP(True Positive)는 병이 있을 것이라고 예측된 환자가 실제로 병을 가진 경우
- TN(True Negative)는 병이 없을 것이라고 예측된 환자가 실제로 병이 없는 경우
- FP(False Positive)는 병이 있을 것이라고 예측된 환자가 실제로 병이 없는 경우
- FN(False Negative)는 병이 없을 것이라고 예측된 환자가 실제로 병이 있는 경우

하지만 혼동행렬은 분류모델의 성능을 평가하는데 사용하기에는 한계가 존재한다. 모형은 연속적인 예측값인 확률($\hat{\pi}$)을 결과로 반환하는데, 예측은 cut-off point를 기준으로 이항변수(0 또는 1)로 범주화해야 하기 때문이다.

또한 (일반적으로 0.5로 선택되는) cut-off point인 π_0값도 임의로 지정되기 때문에 객관적이지 않다. 때문에 cut-off point가 달라지면 혼동행렬도 달라진다! 특히 클래스 불균형이 심한 경우에는 cut-off point에 따라서 혼동행렬이 크게 바뀌는 문제가 발생한다. *스포하자면… 이러한 이유로 ROC곡선이 등장한다!*

■ 분류 평가지표

우리가 다루는 범주형자료분석은 데이터마이닝 또는 머신러닝의 관점에서 분류모델이라고 할 수 있다. 평가지표 파트에서는 분류 모델의 다양한 성능 평가지표에 대해 알아볼 것이다. 특히 case에 따라 사용해야 하는 평가지표가 달라질 수 있기 때문에 상황에 맞게 적절한 평가지표를 사용하는 것이 중요하다.

1) 정확도 (Accuracy)

정확도(accuracy/ ACC /정분류율)는 전체 경우에서 실제값과 예측값이 같은 경우의 비율이다. 즉, 예측이 실제 정답과 얼마나 정확히 일치하는지를 나타내는 지표이다.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} = 1 - Error\ rate$$

정확도는 직관적이라 자주 쓰이는 지표이며, 1에 가까울수록 좋은 모형이라고 평가할 수 있다. 하지만 imbalanced data에서 모형을 평가하는 경우 문제가 발생한다.

2) 정밀도 (Precision)

정밀도(precision/ PPV/ Positive Predictive Value)는 True라고 분류한 것 중에 실제로 True인 것의 비율이다. 즉, 예측한 성공 중 실제 성공은 얼마인지를 나타내는 지표이다.

$$Precision\ = \frac{TP}{TP + FP}$$

정밀도가 중요한 지표인 경우는 스팸 메일 판단 예시를 들 수 있다. 실제로 Positive인 스팸 메일을 Negative인 일반 메일로 분류하더라도 사용자가 불편을 느끼는 정도이지만, 만약 일반 메일을 스팸 메일로 분류하게 되면 중요한 메일을 받지 못하게 되는 경우도 생기게 된다. 즉, False Positive가 더 critical한 경우에 사용된다.

3) 민감도 (Sensitivity) / 재현율 (Recall)

민감도(Sensitivity/ TPR/ True Positive Rate) 또는 **재현율(Recall)**은 실제 Positive 중 예측과 실제 값이 Positive로 일치하는 비율이다. 즉, 실제로 $Y = 1$인 값 중에서 $\hat{Y} = 1$이라고 예측된 값의 비율로, 1에 가까울수록 좋다.

$$Sensitivity\ (Recall)\ = \frac{TP}{TP + FN}$$

민감도(또는 재현율)이 중요한 지표인 경우는 실제 Positive 양성 데이터를 Negative로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우로 예로는 암 등의 질병 진단을 하는 경우가 있다. 건강한 사람을 양성이라고 예측한다면 재검사를 하는 수준의 비용이 소모되겠지만, 진짜 암 환자를 음성이라고 예측한다면 생명을 앗아갈 정도의 심각한 문제가 발생하기 때문이다. 즉, False Negative가 더 critical한 경우 사용된다. 참고로 민감도는 ROC 곡선의 Y축이다.

4) 특이도 (Specificity)

특이도(Specificity/ TNR/ True Negative Rate)는 실제 Negative 중 예측과 실제 값이 Negative로 일치하는 비율이다. 즉, 실제로 $Y = 0$인 값 중에서 $\hat{Y} = 0$이라고 예측된 값의 비율로, 1에 가까울수록 좋다.

$$Specificity = \frac{TN}{TN + FP}$$

한편 **FPR(False Positive Rate, fall-out)**는 실제 Negative 중 Positive라고 예측된 비율로, 1-특이도와 같다. FPR은 0에 가까울수록 좋다.

$$FPR = \frac{FP}{TN + FP}$$

FPR은 ROC곡선의 X축이다.

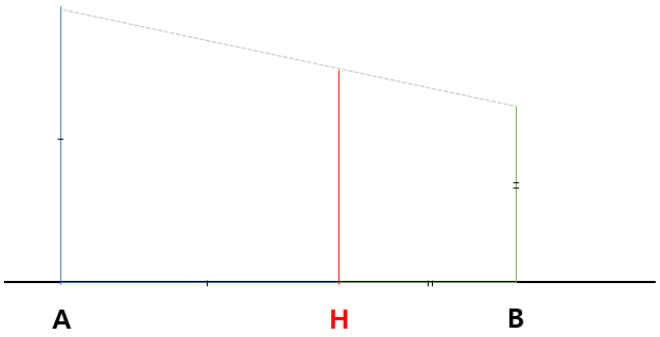
5) F1-score

F1-score는 Precision과 Recall(=Sensitivity)의 조화평균이다.

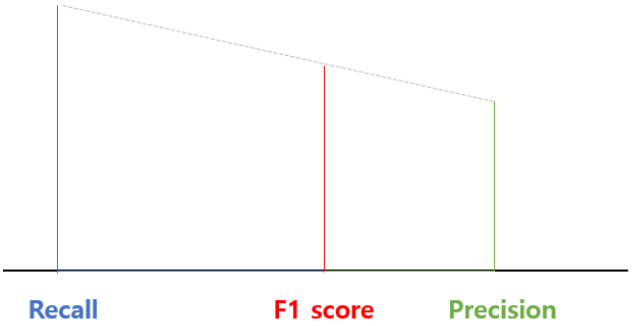
$$F1\ score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FN + FP}$$

조화평균 뭐였더라? 주어진 수들의 역수의 산술평균의 역수[$2 \times ab/(a+b)$]이다.

클래스가 불균형한 데이터일 때, Precision과 Recall의 조화평균을 구함으로써 두 지표의 밸런스를 고려해 모델의 성능을 정확히 평가할 수 있다. 왜 산술평균이 아니고 조화평균을 구하는 것일까? 기하학적으로 접근해보자.

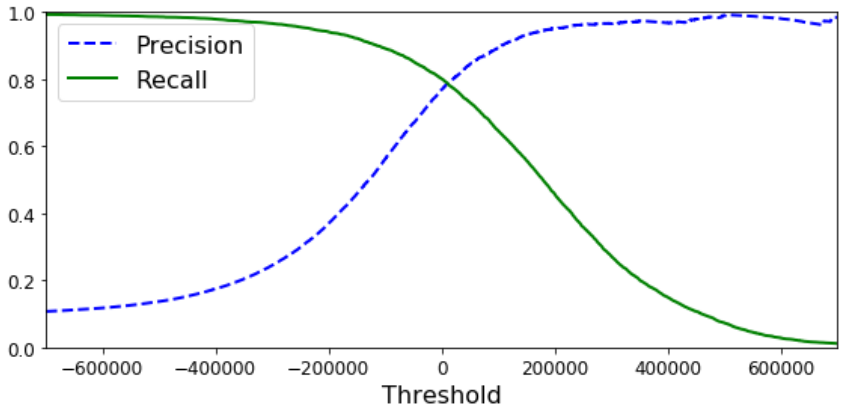


조화평균(harmonic mean)은 다음과 같이 나타낼 수 있다. 서로 다른 길이의 A, B와 이 두 길이의 합만큼 떨어진 변(AB)으로 이루어진 사다리꼴이 있을 때, 변 AB에서 각 변의 길이가 만나는 지점까지의 거리가 바로 조화평균이다. 조화평균을 구함으로써 작은 길이인 B쪽에 치우치지만, 작은 값과 큰 값 사이의 값을 가진 평균이 도출된다. 따라서 단순히 평균을 구하는 것이 아니라, 큰 값이 있다면 페널티를 주어 작은 값에 가까운 평균을 구하게 되는 것이다. 이러한 원리로 imbalanced data에서 큰 값을 가지는 클래스에 대해 페널티를 줄 수 있다. 반대로 Precision과 Recall이 어느 한쪽으로 치우치지 않는 수치를 나타낼 때에 상대적으로 높은 값을 갖는다.



그렇다면 왜 Precision과 Recall을 고려하는 것일까? 결론부터 말하자면, Precision과 Recall은 Trade-Off 관계이다. 즉, 정밀도와 재현율을 둘 다 최대값을 가질 수 없다는 것이다. 분류 업무의 특성상 정밀도 또는 재현율이 특별히 강조되어야 하는 경우는 임계값(threshold)를 정해 정밀도 또

는 재현율을 높일 수 있다. 하지만 둘은 상호 보완적인 지표이기 때문에, 어느 한쪽을 강제로 높이면 다른 쪽은 떨어지게 된다.



임계값을 낮출수록 재현율은 올라가고 정밀도는 떨어진다. 만약 임계값을 0.5에서 0.4로 낮추면 그만큼 positive 예측을 너그럽게 하기 때문에 true로 예측되는 값이 많아지게 된다. 따라서 실제 양성을 음성으로 예측하는 횟수도 줄어들 수 밖에 없다.

F1-score도 마찬가지로 1에 가까울수록 좋다. 하지만 F1-score가 높다고 항상 좋은 모형은 아니기 때문에 주의해야 한다. 따라서 정밀도와 재현율, F1-score를 모두 구한 후 비교하여 적합한 모형을 선정하는 것이 중요하다.

또한 F1-score는 FN을 고려하지 않는다는 한계가 있다. 따라서 Y=1과 Y=0이 바뀌면 성능 지표 또한 바뀌게 된다. *(이게 무슨 말인지는 뒤의 Case 예시에서 살펴보자!)*

6) MCC(Matthews Correlation Coefficient)

MCC(Matthews Correlation Coefficient/ 매튜 상관계수/ 파이계수)는 가장 균형 잡힌 척도로도 여겨진다.

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

이름에서도 알 수 있지만 상관계수의 형식이고, -1에서 1의 값을 갖는다. 1에 가까울수록 완전 예측, -1에 가까울수록 완전 역예측을 뜻하고, 0에 가까울수록 랜덤 예측이라고 볼 수 있다.

MCC가 가장 균형 잡힌 척도인 이유는 Confusion Matrix의 모든 부분을 사용하여 만들어졌기 때문이다. 따라서 imbalanced data에서도 유용하게 사용할 수 있다.

■ Case 예시

이제 여러 상황의 예시를 통해 각 지표의 장단점과 MCC가 왜 좋은 지표인지 알아보자.

1) CASE 1 : 밸런스가 무참히 깨져 있는 95:5의 imbalanced data에서 단순히 모두 $Y = 1$이라고 예측하는 모델을 적합하였다고 가정해보자.

Case1		예측값($\hat{Y}$)	
		$Y=1$	$Y=0$
관측값(Y)	$\hat{Y}=1$	95	0
	$\hat{Y}=0$	5	0

Accuracy는 $[\frac{95}{100} = 95\%]$, F1-score는 $[\frac{2\times95}{2\times95+5+0} = 97.44\%]$ 가 나온다. 모델의 성능이 높게 평가되었으니 이대로 사용해도 되는 걸까? 하지만 MCC를 구해보면 분모에 0이 들어가기 때문에 그냥 랜덤으로 예측하는 것과 다름이 없다는 결과가 나온다. 따라서 이렇게 모든 결과 값을 하나로 올인해버리는 모델의 경우 랜덤 예측이나 다를 바 없으므로… 뭔가 잘못되고 있음을 직감해야 한다!

2) CASE2 : 이번엔 조금 현실적인 경우를 생각해보자. 마찬가지로 95:5의 imbalanced data이지만, 예측 모델이 조금 달라졌다.

Case2		예측값($\hat{Y}$)	
		$Y=1$	$Y=0$
관측값(Y)	$\hat{Y}=1$	90	5
	$\hat{Y}=0$	4	1

Accuracy는 $[\frac{90+1}{100} = 91\%]$, F1-score는 $[\frac{2\times90}{2\times90+4+5} = 95.24\%]$가 나온다. 위의 경우와 마찬가지로 예측이 매우 잘 된 것 같아 보이지만, MCC를 계산해보면 $[\frac{(90\times1)-(5\times4)}{\sqrt{(90+5)(90+4)(1+5)(1+4)}} = 0.14]$로 0에 가까운 (랜덤으로 추측한 것과 다름없는) 결론에 이르게 된다. 왜 이런 결과가 나오는 것일까? F1-score는 TN을 고려하지 않는 반면 MCC는 혼동행렬의 모든 항목에 영향을 받는다. 따라서 Negative와 Positive 모두에서 잘 작동하는 경우에만 높은 결과가 나오게 된다.

3) CASE3 : 이번엔 위의 예시에서 Y=1과 Y=0이 바뀐 경우를 생각해보자.

Case2		예측값($\hat{Y}$)	
		$Y=1$	$Y=0$
관측값(Y)	$\hat{Y}=1$	1	4
	$\hat{Y}=0$	5	90

위의 CASE에서 F1-score는 $[\frac{2\times90}{2\times90+4+5} = 95.24\%]$로 모델이 좋은 성능을 보임을 나타냈지만, Y=1과 Y=0이 바뀐 경우에서는 F1-score가 $[\frac{2\times1}{2\times1+5+4} = 18.18\%]$로 모델의 성능이 떨어져 보이는 결과가 나타난다. 따라서 F1-score는 TN 부분을 활용하지 않기 때문에 Y=1과 Y=0이 바뀌게 되면 성능 지표 또한 바뀌게 된다는 것을 알 수 있다.

그렇다면 F1-score는 MCC보다 좋지 않은 지표인 것일까? 그렇지 않다! 그 이유는 Tyep1 error와 Type 2 error의 경중을 따지는 문제와 관련이 있다. 실제로 분류 예측의 많은 경우 FN으로 예측했을 때와 FP로 예측했을 때의 cost가 다르다. 하지만 일반적으로 F1-score는 비대칭 데이터를 효과적으로 평가하기 위해 적은 범주를 positive로 두고 계산한다고 한다.

지난 방학세미나 문제를 떠올려보자! 트럭의 air pressure system의 failure를 예측하는 문제에서, 고장이 아니라고 예측했는데 실제로 고장인 경우 훨씬 더 큰 페널티를 부여함으로써, cost metric을 모델 평가 지표로 삼았다. 즉 상황에 따라 적절한 평가지표를 사용하는 것은 매우 중요한 문제인 것이다!

2. ROC 곡선과 AUC

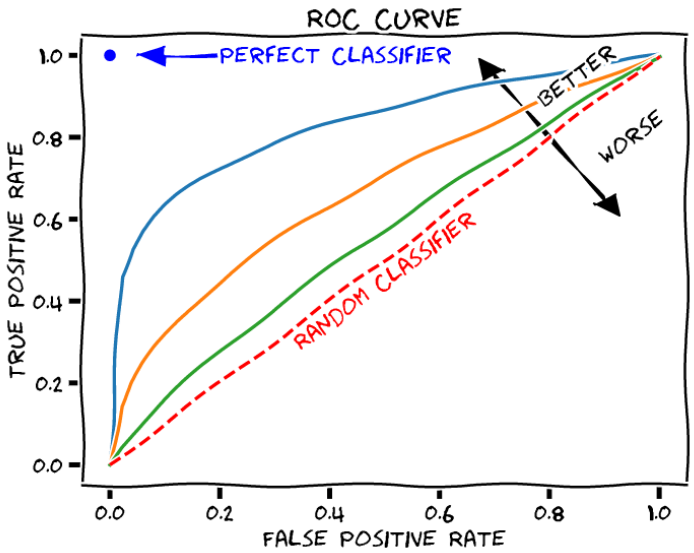
- ROC 곡선(ROC curve)

1) ROC 곡선(ROC curve)이란?

ROC 곡선 (Receiver Operating Characteristic Curve)이란 가능한 모든 cut-off point에 대해 '민감도(Sensitivity) 또는 재현율(Recall)'와 '1-특이도(Specificity)'를 나타낸 곡선(또는 직선)이다.

Confusion Matrix의 경우 분석자가 임의로 정한 cut-off point에 따라 분류되기 때문에 정보의 손실이 일어났다. 또한 cut-off point 임계값(π_0)에 따라 각각 다른 confusion matrix가 만들어지기 때문에 cutoff point에 의존적이라는 한계가 있었다. 하지만 ROC curve는 모든 cut-off point (π_0)에 대하여 예측 검정력을 구하기 때문에 더 많은 정보를 갖는다. 또한 모든 cut-off point를 고려하기 때문에 이를 활용하여 가장 적합한 cut-off point를 찾을 수 있다!

2) ROC 곡선의 형태



ROC 곡선은 X축이 FPR(1-특이도), Y축이 TPR(민감도, 재현율)이다. x와 y가 둘 다 [0,1]의 범위를 갖고, (0,0)에서 (1,1)을 잇는 우상향하는 위로 볼록한 곡선의 형태를 가진다. 왜 이런 형태를 가질까? Cut-off point가 0에 가까워질수록 (1,1)에, 1에 가까울수록 (0,0)에 가까워지는 알고리즘을 정리해보았다.

- Cut-off point가 0에 가까워지면 -> 대부분 Y=1로 예측! -> TP&FP 증가 / TN&FN 감소 ->

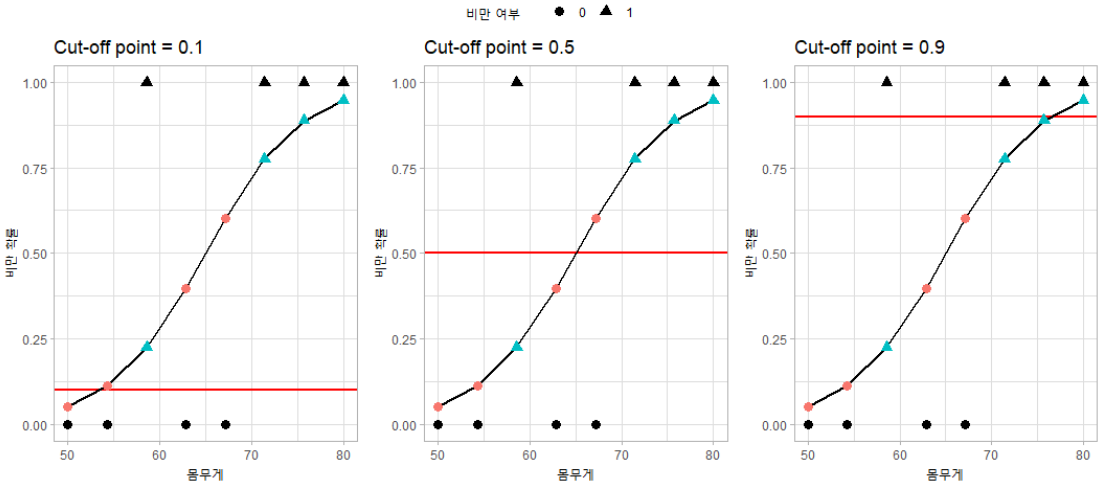
FPR과 TPR이 1에 가까워진다!

- Cut-off point가 1에 가까워지면 -> 대부분 Y=0로 예측! -> TP&FP 감소 / TN&FN 증가 -> FPR과 TPR이 0에 가까워진다!

3) ROC 곡선으로 적합한 cut off point 찾기

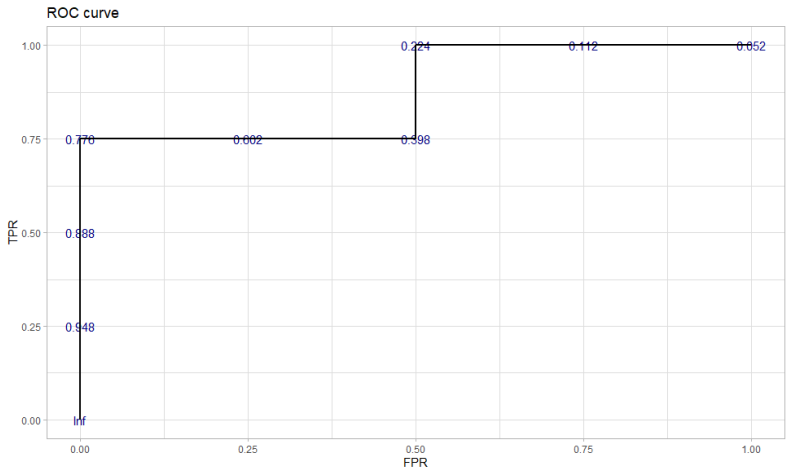
이제 예시를 통해 ROC곡선이 어떻게 그려지는지 알아보자.

수많은 cut-off point에 따라서 수많은 confusion matrix가 나올 것이며, 각각의 confusion matrix는 다른 FPR(1-특이도)과 TPR(민감도)값을 나타낼 것이다. 우리는 그 값들을 사용하여, X축에 FPR, Y축에 TPR을 둔 ROC 곡선을 그릴 수 있게 된다.



몸무게에 따른 비만 확률을 예측하는 로지스틱 회귀 모형에서 각각 cut-off point값을 0.1, 0.5, 0.9로 두고 예측한 그래프이다. cut-off point에 따라 예측이 다르게 된 것을 확인 할 수 있다.

모든 관측값에 cut-off point를 적용하여 예측한 confusion matrix를 통해 구해진 FPR과 TPR 값을 나타내는 ROC 곡선을 그려보면 다음과 같다.



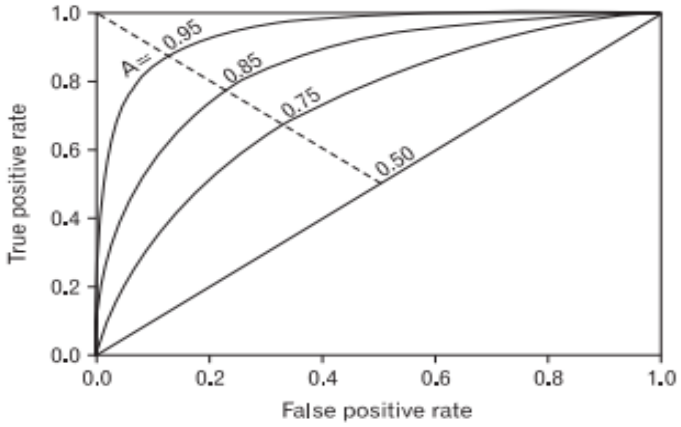
그래프의 각 점에 숫자로 나타낸 값들은 그 점에서의 cut off point이다. 실제로 cut-off point가 0에 가까울수록 (1,1)에 가깝고, 1에 가까울 때는 (0,0)이 됨을 알 수 있다.

우리는 이 ROC 곡선을 통해 가장 적합한 cut off point값을 찾을 수 있다! TPR은 예측과 실제가 같을 경우를 나타낸 지표이므로 1에 가까울수록 좋고, FPR은 예측과 실제가 다를 경우를 나타낸 지표이므로 0에 가까울수록 좋다는 것을 배웠다. 이를 ROC 곡선 내에서 다시 해석하면,

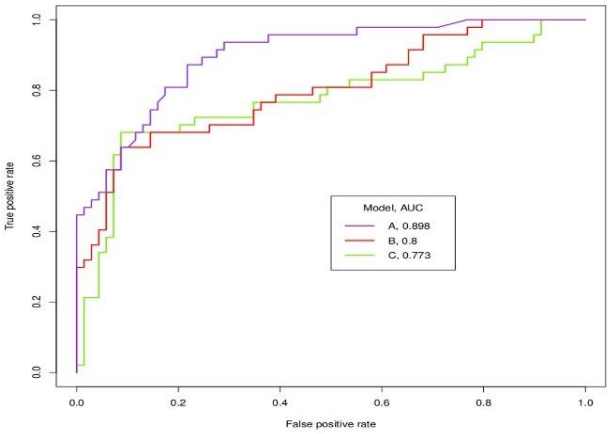
- 같은 Y값(TPR)일 때, X값(FPR)이 더 작을수록 좋은 cut-off point이고,
- 같은 X값(FPR)일 때, Y값(TPR)이 더 클수록 좋은 cut-off point인 것이다!

■ AUC (Area Under the Curve)

AUC(Area Under the Curve)란 말그대로 곡선 아래의 넓이라는 뜻으로, ROC curve 밑의 면적을 뜻한다. AUC는 0에서 1의 값을 갖는다.



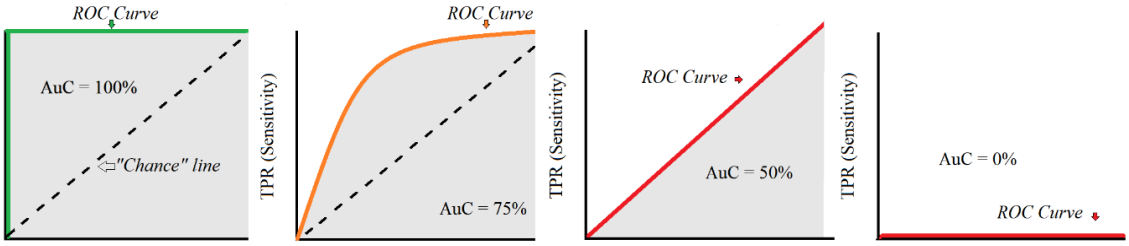
AUC 값이 높을수록, 즉 AUC가 1에 가까울수록 더 좋은 예측 성능을 가진다고 해석할 수 있다! 앞에서 X값(FPR)이 고정되었을 때 Y값(TPR)이 클수록 더 좋은 모델임을 배웠다. 이를 다시 해석하면, 특이도가 고정되었을 때, 민감도가 높을수록 좋다는 뜻이 된다. 이에 따라 ROC 곡선이 위로 볼록할수록 성능이 더 좋은 모델이고, 이는 AUC 값이 1에 가까워진다는 것과 같다.



이런 식으로 서로 다른 모델의 ROC 곡선을 나타내고 각각의 AUC값을 구하면 모델 간의 성능을 비교할 수 있다. 곡선 아래 면적이 클수록 더 좋은 성능을 갖는 모델이라고 판단할 수 있다! 위의 경우는 AUC가 0.898인 보라색 ROC 곡선을 그리는 모형이 가장 좋은 성능을 갖는다.

AUC는 모든 cut-off point르 고려하므로 특정 cut-off point를 기준으로 모델의 성능을 비교할 필요가 없다. Cut-off point와 상관없이 모델의 성능을 측정 할 수 있다.

아래에 다양한 ROC 커브에 대한 AUC를 나타낸 그림이다.



- AUC=1인 경우 : 100% 완벽하게 예측한 경우이다. 이런 경우는 모델이 과적합(overfitting)되었을 가능성을 고려해보아야 한다!
- AUC=0.5인 경우 : 50%만 맞춘 예측으로, 랜덤하게 예측한 결과와 다름없는 성능이다. 보통 0.5 이상의 값을 가져야 정상이다.
- AUC=0인 경우 : 100% 반대로 예측한 경우이다.

AUC가 0.5 미만의 값을 갖는 경우는 어떤 경우일까? 분류군을 반대로 처리한 경우라고 생각하면 된다. 말그대로 Y=1과 Y=0을 거꾸로 예측한 것이다. 따라서 정상적인 상황에서 AUC 값이 0보다 낮게 나왔다면 문제가 있다는 말이니 다시 확인해보자… *(시험문제를 완벽히 틀리는 것은 완벽히 맞는 것만큼 어려운 일이라는 뜻…AUC=100%는 AUC=0%와 같다…)*

3. Sampling

■ 클래스 불균형 (Class Imbalance)

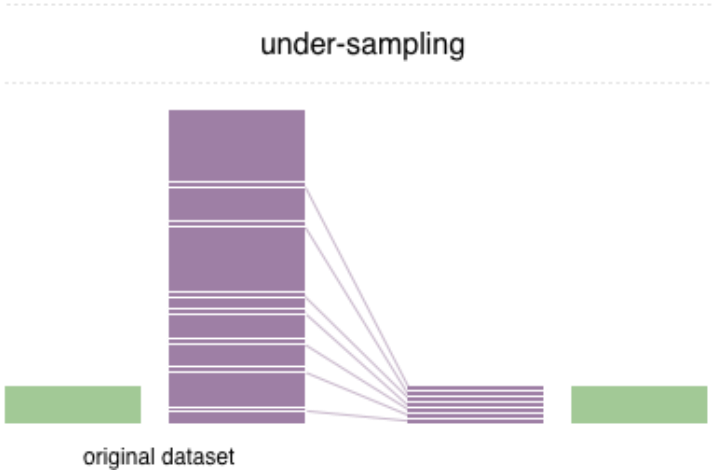
현실 데이터에서는 클래스 불균형 문제가 자주 있다. 이 때 클래스는 범주형 반응변수를 말한다. 어떤 데이터에서 각 클래스가 갖고 있는 데이터의 양에 큰 차이가 있는 경우, 우리는 **클래스 불균형(class imbalance)**이 있다고 말한다. 질병이 있는 사람의 데이터를 수집한다고 할 때, 일반적으로 질병이 있는 사람이 없는 사람에 비해 적다. 대부분의 현실 데이터에서는 이렇게 클래스 불균형이 존재한다.

그렇다면 왜 클래스 균형이 필요할까? 그리고 어떤 경우에 클래스 균형이 필요할까? 클래스 균형은 소수의 클래스에 특별히 더 관심이 있는 경우에 필요하다! 만약 Y변수의 클래스 비율이 너무 많이 차이나게 되면 단순히 우세한 클래스를 택하는 모형의 정확도가 높게 나타나기 때문에 모델의 성능을 판별하기 어렵게 된다. 정확도가 높은 것과 별개로 데이터 개수가 적은 소수 클래스의 재현율(실제 positive값 중 예측 positive값과 일치하는 비율)이 낮아지기 때문이다. 정확도가 분류 모형의 적절한 평가지표가 될 수 없다는 Accuracy Paradox의 이유이기도 하다.

GIGO(Garbage In, Garbage Out)이라고… 좋은 모델을 만들려면 좋은 train set이 필요하다… 이렇게 데이터가 비대칭인 경우, 샘플링을 통해 문제를 해결할 수 있다. 지금부터 imbalanced data를 다루는 여러 sampling 방법들을 알아보자.

■ 언더 샘플링 (Under Sampling)

언더 샘플링(Under Sampling)은 소수 클래스는 그대로 두고, 다수 클래스의 데이터를 소수 클래스에 맞추어 감소시키는 방법이다.



언더 샘플링은 데이터의 사이즈가 줄어들어 메모리 사용이나 처리 속도 측면에서 유리하다는 장점이 있다. 하지만 관측치의 손실이 일어나기 때문에 정보가 누락되는 문제가 발생한다.

일반적으로 랜덤하게 다의 샘플의 수를 줄이는 **Random Under Sampling** 방법을 사용한다. Training set의 크기가 매우 클 경우, 샘플 수를 줄임으로써 메모리 사용 문제 등을 해결할 수 있지만, 만약 샘플링해서 얻은 임의의 표본이 대표성을 띠지 못하면, 부정확한 결과를 초래할 수 있게 된다.

사실 Under Sampling은 결국 관측치를 삭제함으로써 정보를 누락시키는 방법이기 때문에 보통 Over sampling을 많이 사용한다.

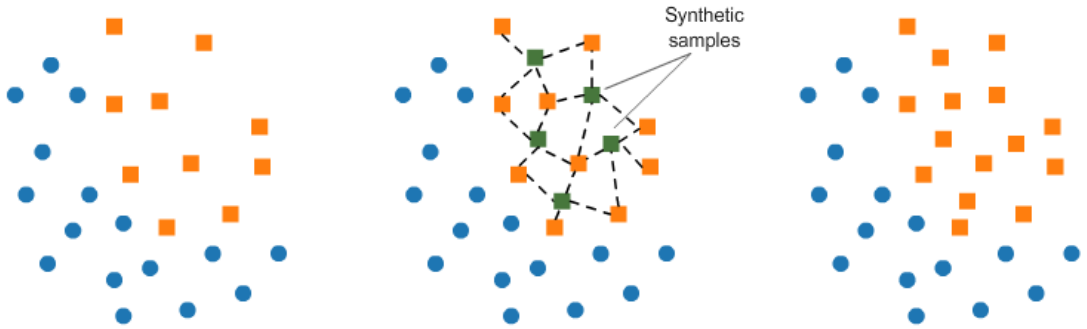
■ 오버 샘플링 (Over Sampling)

오버 샘플링(Over Sampling)은 소수 클래스의 데이터를 다수 클래스에 맞추어 증가시키는 방법이다.

오버 샘플링은 정보의 손실이 없기 때문에 언더 샘플링에 비해 성능이 좋다. 반면 관측치 수가 늘어나기 때문에 메모리 사용이나 처리속도 측면에서 상대적으로 불리하다.

가장 기본적인 오버 샘플링의 종류는 **Random Under-Sampling** 방법이다. 무작위로 소수 클래스의 데이터를 복제함으로써 소수 클래스 데이터를 늘리는 방법이다. 정보의 손실이 없지만, 임의로 데이터를 복제하기 때문에 과적합(overfitting)될 가능성이 있다.

그래서 실제로 많이 사용되는 오버 샘플링 방법이 바로 **SMOTE(Synthetic Minority Over-sampling Technique)**이다.

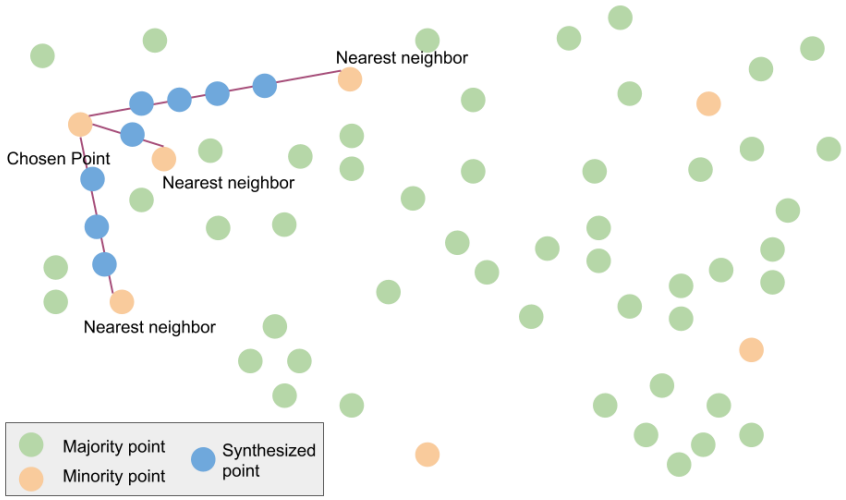


소수 클래스 데이터를 늘리는 알고리즘은 다음과 같다.

가. 소수 클래스의 데이터 중 하나를 선택한다.

나. 선택한 데이터와 가장 가까운 소수 클래스의 데이터를 탐색하고, 그 중에서 무작위로 k개의 데이터를 선택한다.

다. 처음에 선택한 데이터와 무작위로 선택한 k개의 데이터 사이의 직선을 그리고, 그 직선 상에 가상의 소수 클래스 데이터(synthetic samples)를 생성한다.



SMOTE는 이러한 과정을 통해 가상의 데이터를 생성함으로써 소수 클래스의 데이터를 늘리기 때문에, Random over-sampling보다 overfitting이 발생할 가능성이 줄어든다.

반면 새로운 소수의 가상 데이터를 생성하는 과정에서 인접한 다수 데이터의 위치는 고려하지 않기 때문에, 서로 다른 클래스의 데이터가 겹치거나 노이즈가 생성될 수 있다. 따라서 고차원 데이터에서는 효율적이지 않다고 한다.

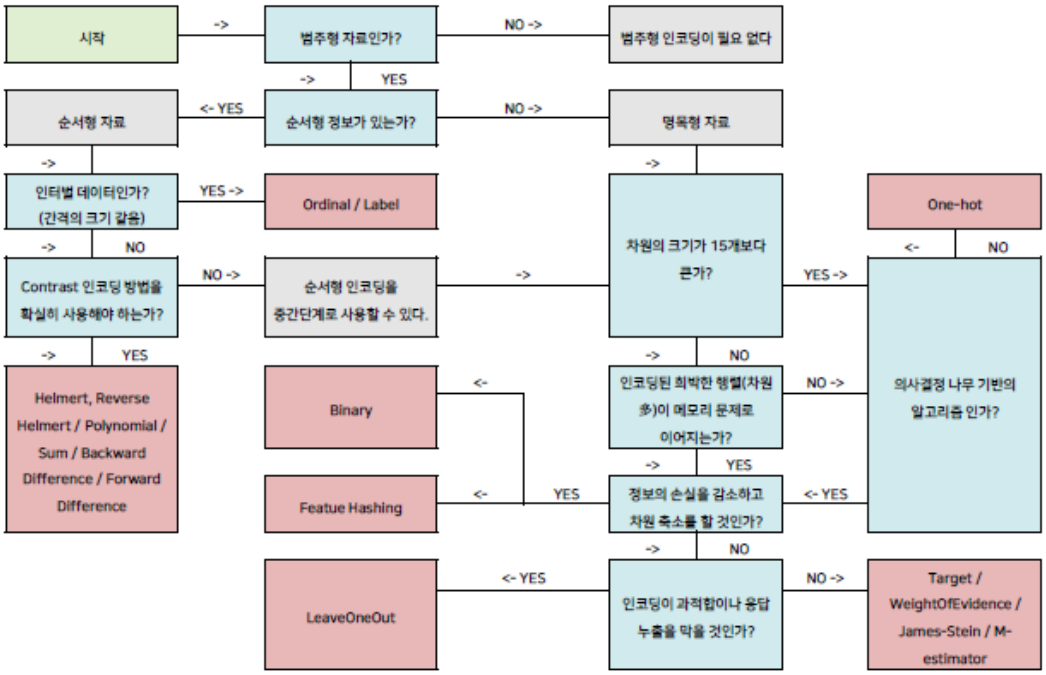
4. Encoding

■ 인코딩(encoding)이란?

컴퓨터 공학적인 관점에서 인코딩의 정의는 문자나 기호들의 집합을 컴퓨터에서 표현하는 방법을 의미한다. 즉, 사용자가 입력한 문자나 기호들을 컴퓨터가 이용할 수 있는 신호로 만드는 것을 뜻한다.

데이터 분석에서의 인코딩은 범주형 자료를 컴퓨터가 읽을 수 있도록 수치화한 것으로 범주형 인코딩 방법(Categorical Encoding Method)가 더 정확한 표현인 것 같다. 실제로 대다수의 정형 데이터를 분석하다 보면 수치형(numeric) 변수 못지않게 많은 범주형(categorical) 변수를 다루게 되는데, 범주형 자료를 수치화하는 인코딩 방식을 거치면 수치형 변수만을 설명변수로 갖는 다양한 분석기법(회귀 계열의 모형들)의 사용이 가능해진다.

Categorical Encoding Methods Cheat-Sheet 영어 자료를 번역한 것이다. *(내가 한 것은 아니다…)*



■ 인코딩의 종류

Classic	Contrast	Bayesian	기타
Ordinal	Simple	Mean (Target)	Frequency
One-Hot	Sum	Leave One Out	
Label	Helmert	Weight of Evidence	
Binary	Reverse Helmert	Probability Ratio	
BaseN	Forward Difference	James Stein	
Hashing	Backward Difference	M-estimator	
	Orthogonal Polynomial	Ordered Target	

정말 다양한 종류의 인코딩 방법이 있다… 하지만 이 많은 걸 우리가 다 알아야 할까요? 이번에도 선택과 집중의 감성으로 자주 쓰는 방법들 위주로 알아봅시다~!

1) One-Hot Encoding (Dummy Encoding)

One-Hot Encoding(Dummy Encoding)은 가변수(dummy variable)를 만들어주는 인코딩 방법이다.

원-핫 인코딩을 진행한 예시다.

혈액형	A	B	O	AB
A	1	0	0	0
B	0	1	0	0
O	0	0	1	0
AB	0	0	0	1

혈액형	A(기준 범주)	B	O	AB
A	1	0	0	0
B	0	1	0	0
O	0	0	1	0
AB	0	0	0	1

해당 범주에는 1, 그 외에는 0을 입력한 뒤 기준이 되는 범주의 열을 삭제한다. J-1개의 더미 변수로 J개의 level을 갖는 인자를 충분히 설명할 수 있는데, 모든 더미 변수가 0이라면 기준 범주일 경우를 뜻하기 때문이다.

원-핫 인코딩은 여러 장점이 있기 때문에 자주 쓰이는 인코딩 방법이다.

- 해석이 용이하다. 기준 범주의 정보가 intercept로 들어가 있기 때문에 기준 범주를 기준으로 해석하면 쉽다.
- 명목형 변수 값들을 가장 잘 반영한다.
- 다중공선성을 해결한다. 해당 범주만 1이고 나머지는 0으로 표현되기 때문에, 한 변수가 다른 변수들로 설명되는 것을 방지해주기 때문이다.
- 지도 학습의 경우 Data Leakage가 발생하지 않는다. Data Leakage란 말그대로 데이터가 누출된 것으로, 정답이 존재하는 지도학습에서 반응변수 Y에 대한 정보가 모델 학습 시 설명변수 X에 들어가는 것을 말한다. 즉 정답을 미리 주고 학습하는 것과 같은 맥락이므로 overfitting이 발생하게 된다.

하지만 범주형 변수의 Level이 너무 많거나 범주형 변수가 너무 많은 데이터의 경우 너무나도 많은 가변수가 생기기 때문에 차원이 늘어나는 문제가 발생한다. 학습속도가 느려지고 상당한 computer power가 요구된다.

원-핫 인코딩은 일반적으로 회귀 모델에서 많이 사용되는데, Tree 기반 모델을 사용할 경우는 기준 범주를 지우지 않고 N개의 가변수를 생성해야 한다. 그 이유는 Tree 기반 모델의 경우, 사용

가능한 모든 부분을 활용하여 트리를 형성하기 때문이다. 만약 원-핫 인코딩처럼 기준 범주가 되는 열을 삭제했는데 기준 범주가 트리를 생성하는데 매우 중요한 요소였다면 트리 모델이 잘못 학습될 수 있기 때문이다.

2) Label Encoding

Label Encoding은 각 범주를 나누어 주기 위해 단순히 점수를 할당하는 인코딩 방법이다. 명목형 자료에 많이 사용하는데, 말 그대로 라벨링(labeling)이기 때문에, 할당된 점수의 숫자에는 어떠한 순서나 연관성도 없다. (점수를 할당할 때 꼭 1~N으로 부여할 필요는 없다.) 원-핫 인코딩과 다르게 차원이 늘어나지 않는다.

혈액형 변수로 예시를 들면 다음과 같다.

혈액형	점수
A	1
B	2
O	3
AB	4

라벨 인코딩의 심각한 단점은 할당된 점수(Labels)들 간의 어떠한 순서나 연관성이 없음에도 불구하고, Label 간의 순서나 연관성이 존재한다고 학습이 일어날 수 있다는 것이다. 존재하지도 않던 순서나 연관성과 같이 필요 없는 정보가 들어가기 때문에 정보 왜곡의 가능성이 생긴다.

3) Ordinal Encoding

Ordinal Encoding은 순서형 정보에 대응하는 점수를 할당하는 인코딩 방법이다. 순서형 자료에 사용하는 방법으로, 점수를 할당할 때는 보통 1부터 부여한다. 위의 Label Encoding과 달리, 할당된 점수들은 순서나 연관성을 갖는다.

매움 정도	점수
맵지 않음	1
약간 매움	2
보통 매움	3
매우 매움	4

(예시는 고추의 맵기를 표현하는 농산물 표준규격을 따랐다…TMI… 재미있는 예시에 진심이니까 다들 PPT 만들 때 창의력을 발휘해주길 바란다…)

Ordinal Labeling의 장점은 차원이 늘어나지 않기 때문에 빠른 학습이 가능하다는 점이다. 또한 순서형 정보를 표현할 수 있다.

하지만 순서형 정보가 있을 때만 사용할 수 있다. 또한 범주 간의 순서를 표현할 때 level 간의 차이가 어느 정도인지를 고려하기가 쉽지 않다. 위의 예시에서 매움 정도의 점수 차이는 모두 1이지만, 그 차이가 다를 수 있다는 것이다. 보통 매움과 매우 매움의 차이는 3일수도…있다는 것이다. 이런 경우일수록 분석과제의 도메인 지식이 중요해진다.

4) Mean Encoding (Target Encoding)

위의 encoding 방법에서는 Encoding된 값 자체는 별 의미가 없었다. 그냥 랜덤하게 0,1 또는 숫자를 부여하여 '구분'하는 것 자체가 목적이었다. 하지만 **Mean encoding**은 구분을 넘어 조금 더 의미 있는 encoding을 하고자 시도한 방법이다. 내가 encoding하려는 feature 변수와 예측하려고 하는 target 변수 간의 어떤 수치적인 관계를 범주형 자료에서도 찾으려는 것이다. 즉, 범주형 변수의 각 수준에 대하여 반응변수(타겟 변수) Y의 평균으로 점수를 할당하는 인코딩 방법이다.

[Y] 토익 점수	[X] 학과	Target Encoding
780	경영	855
930	경영	855
850	통계	820
870	통계	820
810	통계	820
750	통계	820
660	경제	863.33
980	경제	863.333
950	경제	863.33

반응변수 Y가 토익 점수이고 예측변수 X가 학과일때의 예시이다. 범주형 변수인 학과에 대하여 Target Encoding을 진행하여 각 학과별 토익 점수 평균을 점수로 할당하였다.

Label Encoding과 다르게 할당된 점수가 임의의 숫자가 아니기 때문에 당위성이 존재한다는 장점이 있다. 단순한 라벨링이 아니라 타겟 변수 Y와의 연관성을 고려하여 점수를 할당했기 때문이다. 또한 차원이 늘어나지 않는다는 장점이 있다.

하지만 반응변수 Y값을 사용하므로 Data Leakage가 일어나기 때문에 overfitting 될 가능성이 크다. 또한 Training set에 없던 새로운 범주가 Test set에 등장하면 점수를 할당할 수 없게 된다는 문제가 생긴다.

5) Leave One Out Encoding (LOO Encoding)

Leave One Out Encoding (LOO Encoding)은 위의 Mean encoding 방식의 한계를 해결하기 위한 Target Encoding으로, 현재 행을 제외하고 평균을 구한 뒤 이를 점수로 할당하는 인코딩 방식이다. 그래서 이름이 Leave One Out이다! 현재 행을 제외하고 평균을 구하기 때문에 같은 범주라도 다른 점수를 할당할 수 있어서 다양한 라벨링이 가능하다.

[Y] 합격	[X] 학과	Target Encoding	LOO Encoding
1	경영	66.7%	50%
1	경영	66.7%	50%
0	경영	66.7%	100%
1	통계	50%	33.33%
1	통계	50%	33.33%

0	통계	50%	66.7%
0	통계	50%	66.7%
0	경제	33.3%	50%
0	경제	33.3%	50%
1	경제	33.3%	0%

LOO Encoding 방식은 outlier의 영향을 적게 받게 된다. *(평균은 outlier의 영향을 많이 받는다!)* 만약 현재 행이 outlier라면 현재 행을 제외하고 평균을 구하기 때문에 영향을 줄일 수 있는 것이다. 또한 Direct Response Leakage를 방지할 수 있다. 기존 Target Encoding은 현재 행에 해당하는 Y값에 대한 정보도 점수 할당 시 포함하여 평균을 계산했지만, LOO Encoding은 현재 행을 제외한 평균을 계산하기 때문에 현재 행에 해당하는 Y값 정보를 직접적으로 사용하지는 않는다. 그러므로 직접적인(direct) Data Leakage는 발생하지 않는 것이다. 그리고 다른 인코딩과 마찬가지로 차원이 늘어나지 않아 빠른 학습이 가능하다. *(원-핫 인코딩을 제외한 대부분 인코딩이 그렇다…)*

하지만 역시 반응변수 Y변수 값을 사용하기 때문에 Data Leakage가 일어나서 overfitting이 일어날 가능성이 크다. 또한 Mean Encoding과 마찬가지로 Test set에서 Training set에선 없던 새로운 범주가 등장할 경우, 점수를 매길 수 없는 문제가 발생한다.

6) Ordered Target Encoding (CatBoost Encoding)

Ordered Target Encoding은 현재 행 이전의 값들을 사용하여 평균을 구하고 이를 점수로 할당하는 Target Encoding 방식으로, 위의 Mean Encoding과 매우 유사하다. 하지만 현재 행 이전의 값들을 가지고 평균을 구하기 때문에 같은 범주라도 다른 점수를 할당할 수 있으며 LOO encoding보다도 다양한 값들로 점수 할당이 가능하다. 부스팅 모델 중 하나인 CatBoost에서 사용되는 인코딩 방식이다. 장단점은 LOO Encoding과 동일하다.

[Y] 합격	[X] 학과	Target Encoding	Ordered Target Encoding
1	경영	66.7%	100%
1	경영	66.7%	100%
0	경영	66.7%	100%
1	통계	50%	100%
1	통계	50%	100%
0	통계	50%	100%
0	통계	50%	66.67%
0	경제	33.3%	0%
0	경제	33.3%	0%
1	경제	33.3%	0%

참고로 각 수준에서 첫번째 행은 그냥 해당 행만 가지고 평균을 구하면 된다.

CatBoost 모델은 범주형 변수가 많을 때 사용하기 좋은 부스팅 모델이라고 한다. 그 이유에 대해 간단히 알아보자.

- *CatBoost의 max_ctr_complexity 파라미터를 통해 categorical feature combinations을 모델 내에서 진행한다. 무슨 말인지 아래의 예시를 살펴보자.*

country	*hair color*	*class_label*
India	*Black*	*1*
India	*Black*	*1*
Russia	*white*	*0*
Russia	*white*	*0*

위와 같은 경우 country만 알면 hair color는 알 필요가 없다는 것을 알 수 있다. 따라서 모델링 전에 지워도 되지만, catboost는 알아서 상관관계가 높은 변수끼리 묶어준다. 하지만 max_ctr_complexity를 늘릴수록 모델링 속도는 느려진다…

- *특정 범주형 변수의 level이 매우 많은 경우 one-hot encoding을 진행하면 굉장한 컴퓨팅 파워가 필요하다고 배웠다. 하지만 CatBoost는 자동으로 CatBoost Encoding을 해주기 때문에 편리하다.*

5. 맺음말 | *3주의 클린업을 달려오신 짱 멋진 범주팀에게 :)*

안녕하세요. 짱 웃기고 짱 귀여운 범주팀 여러분 ^_^ 범주팀장이 되고부터 클린업 동안 여러분들에게 부족하지 않은 지식을 전달해드리고 싶어서 방학동안 걱정도 많이 하고 준비도 많이 한다고 했는데 이렇게 열심히 공부하고 잘 따라와주어서 정말 감사할 따름이예요… '범주형자료분석'이라는 과목이 통계학과라면 꼭 배워야 하는 과목 중 하나임에도 불구하고 무엇을 배우는지 제대로 알지 못하고 지나치는 경우가 많은 것 같아요. 데이터를 다루는데 있어서 머신러닝이나 딥러닝처럼 말그대로 fancy한 방법들이 넘쳐나는 시기이지만, 통계학과라면 적어도 가장 근본적인 통계 지식들을 알고 있어야 한다고 생각해서 범주를 선택했습니다. 교안을 쓰면서 제대로 이해하지 못하고 지나쳤던 지식들을 하나하나 다시 공부할 수 있어서 저로서도 정말 배우는 게 많았던 3주였습니다.

특히 범주는 회귀분석을 재미있게 들은 사람이라면 더 쉽게 받아들일 수 밖에 없는 과목이라고 생각해요. 첫 스터디 날 직관적이면서 매력적인 과목이라고 말했던 이유는 통계학원론과 수리통계학에서 배운 확률분포, 추정과 가설검정의 원리들을 적용시킴으로써 회귀분석에서 배운 내용을 더 확장시키는 과목이라고 느껴졌거든요. 여러분들도 배우면서 그런 점들을 느끼실 수 있도록 교안을 만들고 싶었는데, 잘 전달되었을까요? 1,2주차에는 분할표와 독립성 검정, 로지스틱 회귀를 포함한 GLM 등 CDA의 기본에 대해 배웠다면 3주차에는 여러분들이 분석에 활용할 수 있는 정말 실용적인 정보들을 담았어요. 특히 분류 모델의 평가지표들에 대해 여러분들이 잘 알아 두신다면, 앞으로 분류 예측을 할 때 정말 수월할 것이라고 생각해요… 좋은 모델에는 정답이 없다고 하죠. 어떤 평가지표를 사용해서 모델을 평가할 것인지도 결국 분석자가 정해야 하는 문제인데요. 결국 분석자가 주어진 문제상황에 대해 이해하는 것(도메인 지식), 어떤 방법이 최선일지 고민하는 것(분석자의 주관)이 분석에서 가장 중요할 수 밖에 없는 것 같습니다.

여러분들이 앞으로 실제 데이터를 다루고 분석할 때 도움이 되는 내용들로 구성된 교안을 만들고 싶었어요. 방학동안 오산시 어린이 교통사고 위험지역 선정이라는 주제의 공모전을 준비하면서, 교통사고를 예측하는 모형을 만들고자 했습니다. 교통사고 수 데이터는 count data라서 GLM을 써야 하는데, 문제는 데이터가 '10년치'의 '어린이' 교통사고 수였기 때문에 대부분이 0인 데이터였던 거예요. 그래서 이름만 알았던 ZIP 회귀모형을 써보자 했는데, 제대로 모델의 원리를 이해하지 않고 코드만 구글링해서 쓰다 보니 오류 해결이 어려워서 결국 포기했던 경험이 있어요. 그래서 모델을 사용할 때 어떤 경우에 사용하고 어떤 원리로 fitting하는지를 아는 것이 정말 중요하다는 생각을 했습니다. 이번 교안을 쓸 때도 이런 것들을 최대한 담으려고 했고요. 주제분석이나 또는 그 이후에 여러분들이 하시게 될 공모전이나 다른 데이터 분석 프로젝트에서도 이번 클린업에서 배운 지식들이 도움이 된다면 참 뿌듯할 것 같아요.

3주 클린업 시간이 정말 빨리 지나간 것 같아요. 하루에 짧은 시간이라도 매일 무엇을 하면 고수가 된다고 하던데 학회를 하는 동안 통계공부도, 코딩 실력도, 분석에 도움되는 지식들도 하다못해 PPT 만드는 스킬이나 발표 같은 것들도 많이 배웠던 것 같아요. 매주 교안 공부하고 피피티 만들고 패키지 과제까지 하면서 바쁜 하루하루 보내셨을 텐데, 그만큼 고수가 되셨을겁니다… 여러분들은 저보다 웃기고 멋진 사람들이니까 더 많이 배워 가실 거라고 믿어요. 거리두기 단계로 다섯명이 다같이 밥 한 번 아직 못 먹어봤는데, 그럼에도 이렇게 어려운 상황 속에서 열심히 해 주셔서 감사합니다. 범주팀과 만날 때마다 즐거운 에너지가 생겨서 정말 행복해요. 중간고사 잘 보시고 재밌고 멋진 주제 정해서 4,5월의 주제분석도 파이팅해봅시다! :)