

# 범주형자료분석팀

**2팀**

김찬영  
이혜인  
김서윤  
심은주  
진수정

# INDEX

---

0. 지난 주 리뷰

1. Confusion Matrix

2. ROC & AUC

3. Unbalanced Data

4. Encoding

# "Confusion Matrix" 혼동 행렬

: 예측 성능 측정을 위해, 학습을 통해 도출한 예측값과 실제 관측값을 비교한 표

- 분류 알고리즘의 성능을 시각화한 표

		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실제 ( $Y$ )	$Y=1$	TP	FN
	$Y=0$	FP	TN

- T(True)와 F(False):  
실제와 예측이 같은지 다른지 여부
- P(Positive) N(Negative):  
예측을 긍정 혹은 부정이라 했는지 여부

**"ACC"** : Accuracy, 정분류율

$$\text{ACC} = (\text{TP} + \text{TN}) / (\text{전체}) = 1 - \text{Error Rate}$$

		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실 제 ( $Y$ )	$Y=1$	TP	FN
	$Y=0$	FP	TN

- 실제와 예측이 맞은 경우의 비율
- 1에 가까울수록 좋은 모형
- Unbalanced Data 모형 평가 시 문제 발생

**"TPR"** : True Positive Rate

*Sensitivity 민감도 / Recall 재현도*

$$TPR = TP / (TP + FN) = 1 - FNR$$

		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실제 (Y)	Y=1	TP	FN
	Y=0	FP	TN

- 민감도  $P(\hat{Y}=1|Y=1)$   
: 실제 성공을 얼마나 잘 예측했는가?에 대한 답
- 1에 가까울수록 좋다
- ROC곡선의 Y축 값

“*TNR*”: True Negative Rate

*Specificity* 특이도

$$\text{TNR} = \text{TN} / (\text{FP} + \text{TN}) = 1 - \text{Error Rate}$$

		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실 제 ( $Y$ )	$Y=1$	TP	FN
	$Y=0$	FP	TN

- 특이도  $P(\hat{Y}=0|Y=0)$   
: 실제 실패를 얼마나 잘 예측했는가?에 대한 답
- 1에 가까울수록 좋다
- FPR의 정확히 반대 개념

# "F1-Score"

$$\begin{aligned} & 2TP / (2TP + FN + FP) \\ & = 2(\text{Precision} * \text{Recall}) / \text{Precision} + \text{Recall} \end{aligned}$$

		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실제 (Y)	Y=1	TP	FN
	Y=0	FP	TN

Sensitivity

Precision

이름에서 알 수 있듯 상관계수 감성이다!

**"MCC"** : Matthews Correlation Coefficient

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

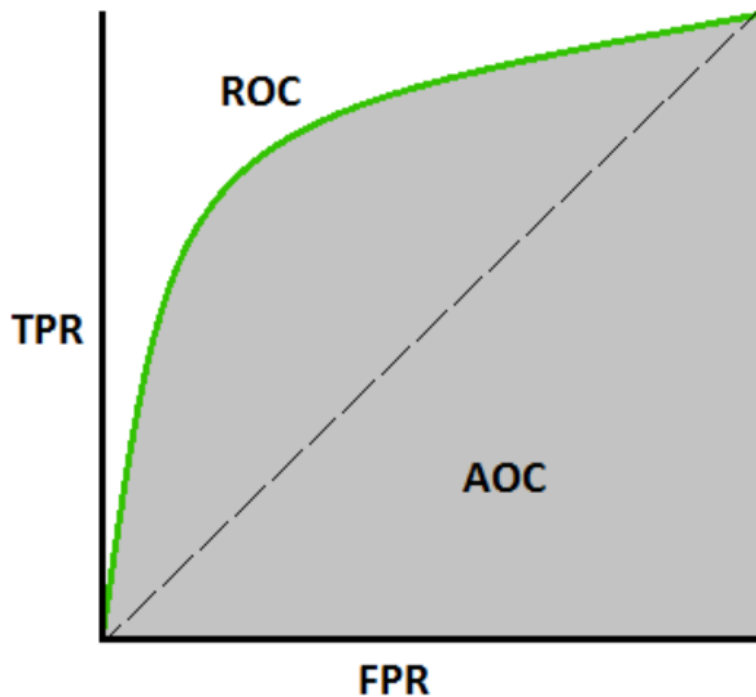
		예측( $\hat{Y}$ )	
		$\hat{Y}=1$	$\hat{Y}=0$
실제 (Y)	Y=1	TP	FN
	Y=0	FP	TN

- 모든 부분을 사용 → unbalanced data에도 유용하다!



- ROC Curve 형태

: 우상향하는 위로 볼록한 곡선 혹은 직선



- Y축:  $TPR = \frac{TP}{TP+FN}$

→ 예측과 실제가 일치

→ Y값이 클수록 좋음

- X축:  $FPR = \frac{FP}{TN+FP}$

→ 예측과 실제가 불일치

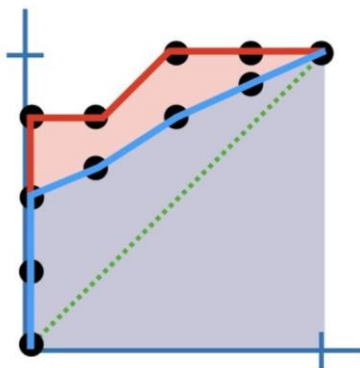
→ X값이 작을수록 좋음

## AUC 의 특징

- $0 \leq AUC \leq 1$
- 모델의 성능을 비교하는 지표

Cut-off point와 상관없이 모델의 성능 측정 가능

- AUC가 1에 가까워질수록 모델의 성능이 좋음



$$AUC_{Blue} < AUC_{Red} < 1$$

→ 빨간색 모델의 성능이 더 좋음

# "언더 샘플링(Under-Sampling) "

: 다수 클래스의 데이터를 소수 클래스에 맞추어 감소시킴

## 장점

메모리 사용, 처리속도 측면에서 유리

## 단점

데이터 손실로 인한 정보 누락 가능성



보통 정보를 누락시키지 않는 오버 샘플링 많이 사용

# "오버 샘플링(Over-Sampling) "

: 소수 클래스의 데이터를 다수 클래스에 맞추어 증가시킴

## 장점

정보의 손실이 없기 때문에, under-sampling에 비해 성능이 좋다

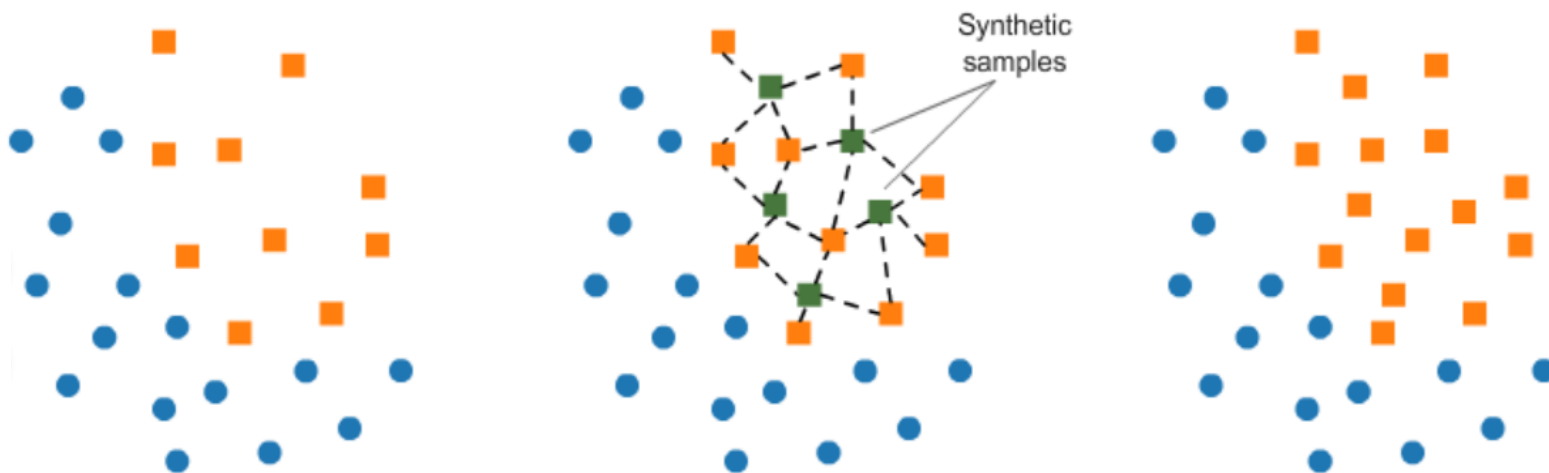
## 단점

메모리 사용, 처리속도 측면에서 불리

# "SMOTE" (*Synthetic Minority Over-sampling Technique*) "

## 알고리즘

1. 소수 클래스의 데이터 하나를 선택
2. 선택된 데이터와 가까운 소수 클래스 데이터에서 랜덤하게  $k$ 개 선택
3. 선택된 데이터와  $k$ 개의 데이터 사이의 가상의 직선 상에 소수 클래스 데이터 생성



# "MSMOTE"

(Modified Synthetic Minority Over-sampling Technique)

: 소수 클래스의 분포와 잠재적인 노이즈 고려

## 특징

소수 클래스 데이터 간의 거리를 기준으로 세 가지 그룹으로 분류

- Security/safe samples: 분류 모델의 성능을 높이는 데이터
- Latent noise samples: 분류 모델의 성능을 낮추는 데이터
- Border samples: 두 그룹에 속하지 않는 데이터



Security/safe samples 위주로 데이터 생성

Latent noise sample에 대해서는 데이터 생성 X

# Encoding 인코딩

- 필요성

1. 수치형 변수보다 범주형 변수가 더 많은 경우가 대부분

2. 수치형 변수만을 설명변수로 받는 분석기법 사용 가능

→ 이번 주 패키지 과제에 있는 XGBoost랄까..?

→ 다양한 회귀계열 모델도 사용할 수 있게 된다!

- 2-0번. XGboost 기본 세팅

참고) xgboost 패키지를 사용하세요.

- 2-0-1번. Train, Test에 있는 범주형 변수들을 one-hot-encoding 해주세요.


(뜨거웠던 패키지와의 기억들 생생하쥬..?)

참고) Xgboost는 numeric 변수만 받으므로 필수적으로 범주형 변수에 대해 encoding을 해야 합니다.

# "One-hot Encoding"

*Treatment Encoding, Dummy Encoding*

MBTI	
ESFJ	
ISFJ	
ENTP	
INTP	



ESFJ	ISFJ	ENTP	INTP
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

: 가변수(dummy variable)를 만들어 주는 것



## *“Ordinal Encoding”*

행복정도
매우 불행
불행
행복
매우 행복



행복 정도	점수
매우 불행	1
불행	2
행복	3
매우 행복	4

- **순서형 정보에 대응되는 점수를 할당** → 대체로 1부터 부여
- 순서형 자료에 사용
- Label Encoding과 달리 **할당된 점수들 간의 순서나 연관성 0**

# "Target Encoding" Mean Encoding

범주형 변수의 각 수준에 대해, **반응 변수 Y의 평균**으로 점수 할당

[Y] 통학 시간 (분)	[X] 팀	Target Encoding
100	범주	58.3
70	범주	58.3
5	범주	58.3
15	회귀	32.5
50	회귀	32.5
20	선대	35
50	선대	35



$$\frac{100+70+5}{3} = \text{범주팀 통학시간의 평균}$$

# *"Leave One Out Encoding"* L00 Encoding

- 현재 행을 제외하고 평균을 구해 이를 점수로 할당하는 방식
  - Outlier의 영향을 줄일 수 있음
- Target Encoding (Mean Encoding)과 매우 유사한 방법
  - L00 Encoding은 같은 범주더라도 다른 점수를 할당할 수 있음  
즉, 다양한 라벨링 가능!

# *"Ordered Target Encoding"*

CATBOOST Encoding

- 현재 행 이전의 값들을 사용하여 구한 평균을 점수로 할당하는 방법
- Target Encoding (Mean Encoding)과 매우 유사한 방법
  - Ordered Target Encoding은  
같은 범주더라도 다른 점수를 할당할 수 있음
- 부스팅 모델 중 하나인 CATBOOST에서 사용하는 인코딩 방법

(지난학기 데마팀 클린업에 CATBOOST에 대한 설명이 있다는 사실...!!)