

# 방학세미나

3팀

김민  
김수빈  
안은선  
조건우

# CONTENTS

1. INTRO

2. EDA

3. PREPROCESSING

4. MODELING

5. PREDICTION

6. CONCLUSION

1

---

INTRO

## 분석 목표와 기준

### 이진 분류

규칙에 따라 입력된 값을 두 그룹으로 분류하는 작업  
주로 어떤 대상에 대한 규칙이 참(TRUE)인지 거짓(FALSE)인지를 분류

### Macro F1-score

$$\text{Macro F1 score} = \frac{1}{N} \times \sum_{i=1}^N \text{F1 score}_i$$

Macro F1은 먼저 class 각각의 F1-score를 계산한 뒤  
평균내는 방식으로 작동



## 분석 목표와 기준

### 이진 분류 왜 Macro F1-Score를 사용할까?

규칙에 따라 입력된 값을 두 그룹으로 분류하는 작업

주로 어떤 대상에 대한 규칙이 참(TRUE)인지 거짓(FALSE)인지를 분류  
F1-score와 마찬가지로 Precision과 Recall을 모두 균형 있게 반영하고

이와 더불어 모든 클래스의 값에 **동등한 중요성**을 부여

### Macro F1-score

$$\text{Macro F1 score} = \frac{1}{N} \times \sum_{i=1}^N \text{F1 score}_i$$

Macro F1은 먼저 class 각각의 F1-score를 계산한 뒤  
**클래스 데이터가 불균형할 때도 사용하기 좋음**  
평균내는 방식으로 작동

## 분석 흐름

### ✓ 모델 성능 향상을 위한 고민



#### 변수 선택

Feature Importance, KS test



#### 샘플링을 통한 불균형 해소

Random Undersampling, Random Oversampling, SMOTE



#### 모델링

Logistic Regression, Catboost, MLP, LGBM,  
Naïve Bayes, Random Forest

2

---

EDA

## 자료 구조 확인

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...
0	train_1	0	15.4140	-2.1016	10.4773	4.8941	12.6506	-3.7205	5.1426	17.7048	...
1	train_2	0	12.3576	-8.1666	11.7785	2.8869	12.3183	-6.9847	4.2671	9.6710	...
2	train_3	0	9.4142	-8.6132	7.2196	3.2496	10.6550	-3.3245	5.1010	18.5389	...
3	train_4	0	13.0647	-0.7917	13.0270	8.7865	10.2252	-2.9311	6.7299	11.8682	...
4	train_5	0	9.5222	-0.2727	8.2173	8.4071	12.7732	-10.3113	4.7486	13.7810	...
...	...	...	...	...	...	...	...	...	...	...	...
159995	train_159996	0	8.0801	-2.5523	10.6587	10.6660	8.7552	-10.8144	5.2019	21.1479	...
159996	train_159997	0	11.8371	-4.9598	12.9172	9.3759	13.0622	2.6849	5.2963	19.8298	...
159997	train_159998	0	11.3368	1.2790	14.1568	11.9850	11.4859	0.2191	5.9112	19.7031	...
159998	train_159999	0	3.4660	-0.2570	4.3530	7.2045	11.4988	-7.0838	5.5081	13.7160	...
159999	train_160000	0	8.5913	-0.8183	11.0295	5.9502	12.9875	-7.8987	5.9785	17.9756	...

160000 rows × 202 columns

구조 확인 결과, ID\_code와 target variable을 제외하고  
총 200개의 Features를 확인



## 자료 구조 확인

### 기초통계량

	target	var_0	var_1
count	160000.000000	160000.000000	160000.000000
mean	0.100487	10.686423	-1.635005
std	0.300650	3.039724	4.048365
min	0.000000	0.408400	-15.043400
25%	0.000000	8.460400	-4.747125
50%	0.000000	10.529800	-1.623700
75%	0.000000	12.764525	1.349175
max	1.000000	20.315000	10.376800

8 rows × 201 columns

### 변수 형태

변수명	class
target	int62
var_0	float64
var_1	float64
⋮	⋮
var_198	float64
var_199	float64

모두 수치형인 것을 확인

## 결측치 확인

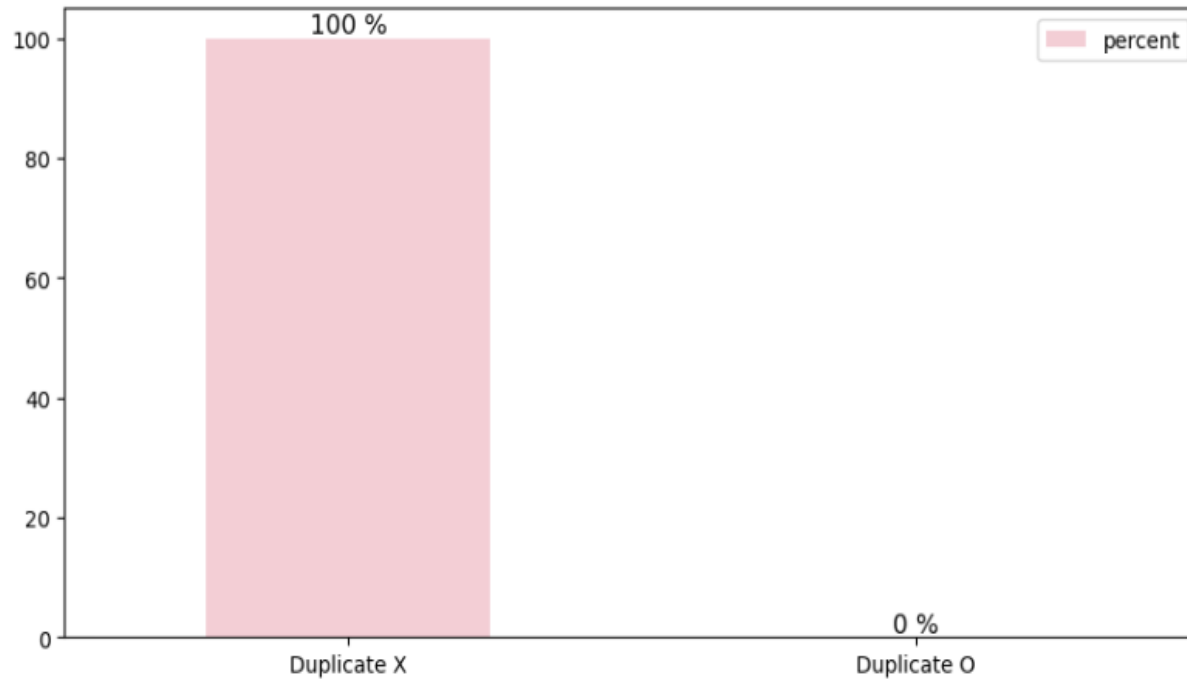
var\_199 160000  
var\_198 160000  
var\_197 160000  
var\_196 160000  
var\_195 160000  
var\_194 160000  
var\_193 160000  
var\_192 160000  
var\_191 160000  
var\_190 160000  
var\_189 160000  
var\_188 160000  
var\_187 160000  
var\_186 160000  
var\_185 160000  
var\_184 160000  
var\_183 160000  
var\_182 160000  
var\_181 160000  
var\_180 160000  
var\_179 160000  
var\_178 160000  
var\_177 160000  
var\_176 160000  
var\_175 160000  
var\_174 160000  
var\_173 160000  
var\_172 160000  
var\_171 160000  
var\_170 160000  
var\_169 160000



변수명	결측치 수
target	0
var_0	0
var_1	0
⋮	⋮
var_198	0
var_199	0

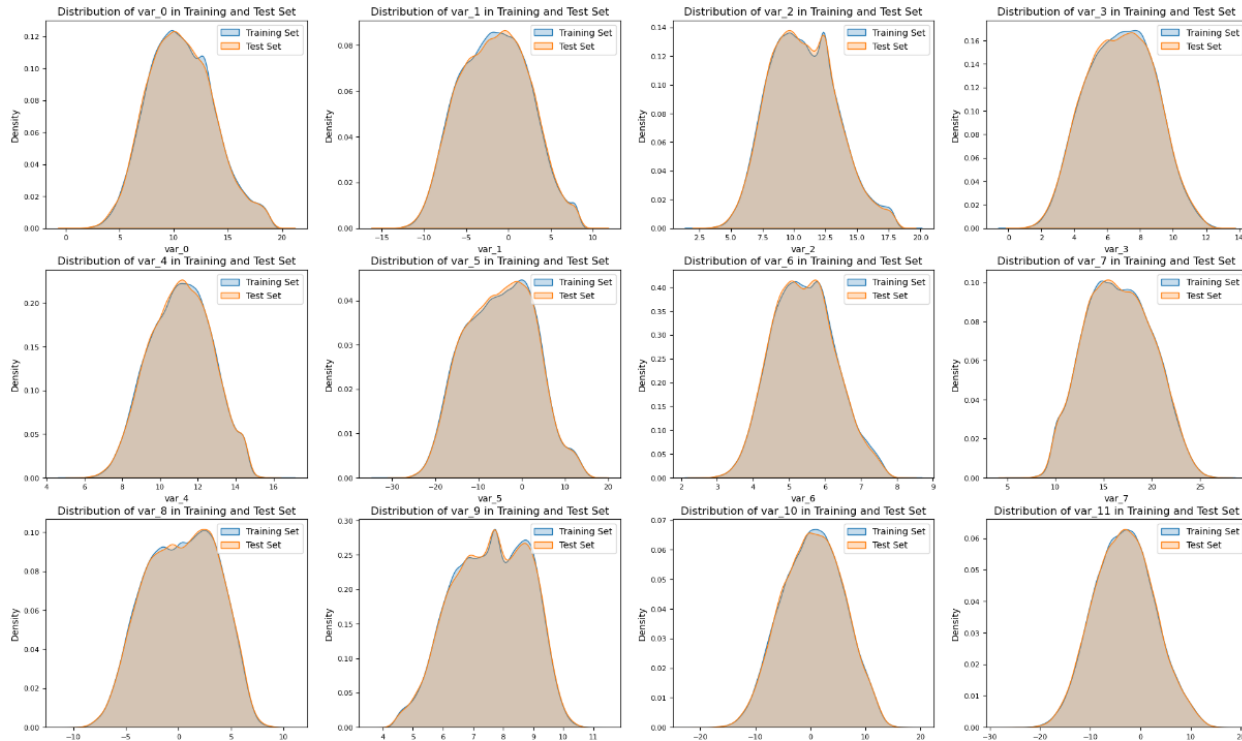
모든 변수에서 결측치가 존재하지 않음을 확인

## 중복데이터 확인



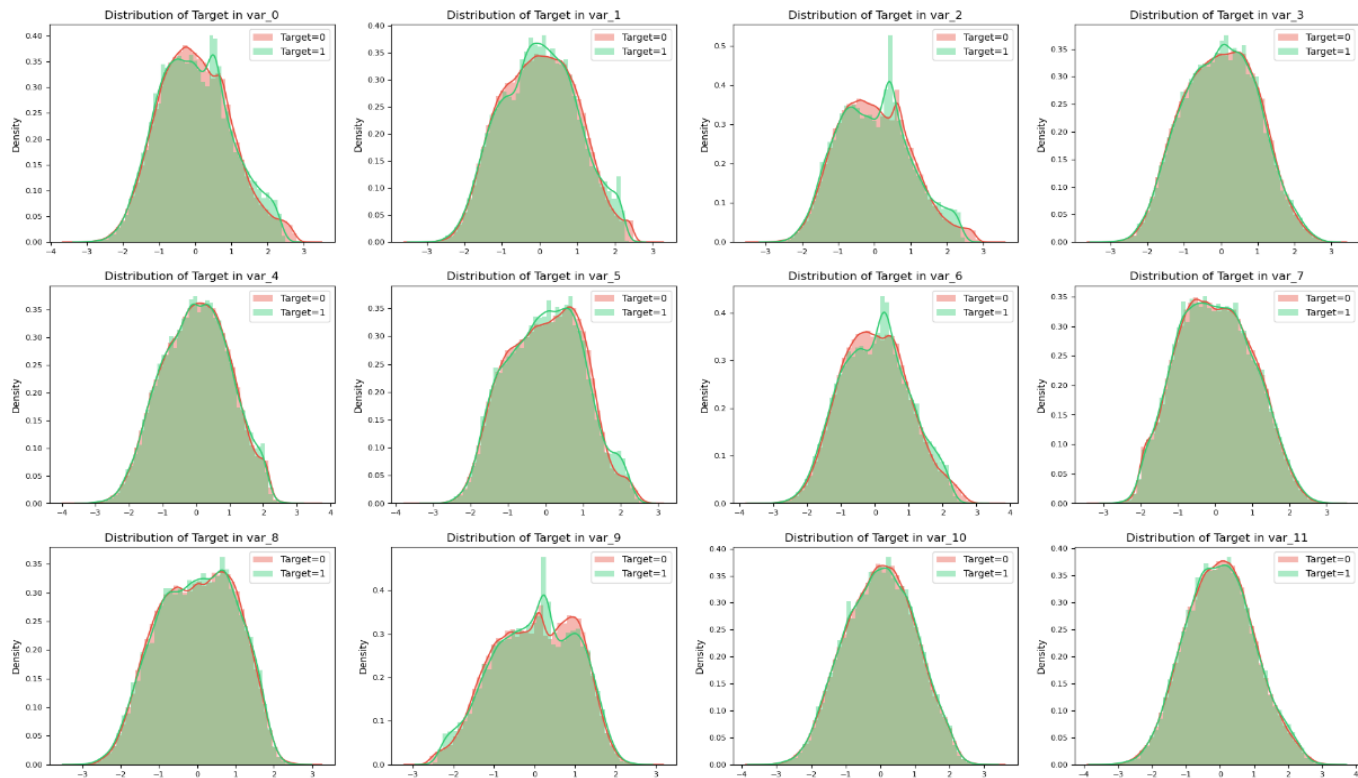
중복되는 데이터 없이 모든 데이터가 unique함을 확인

## Train, test 분포 확인



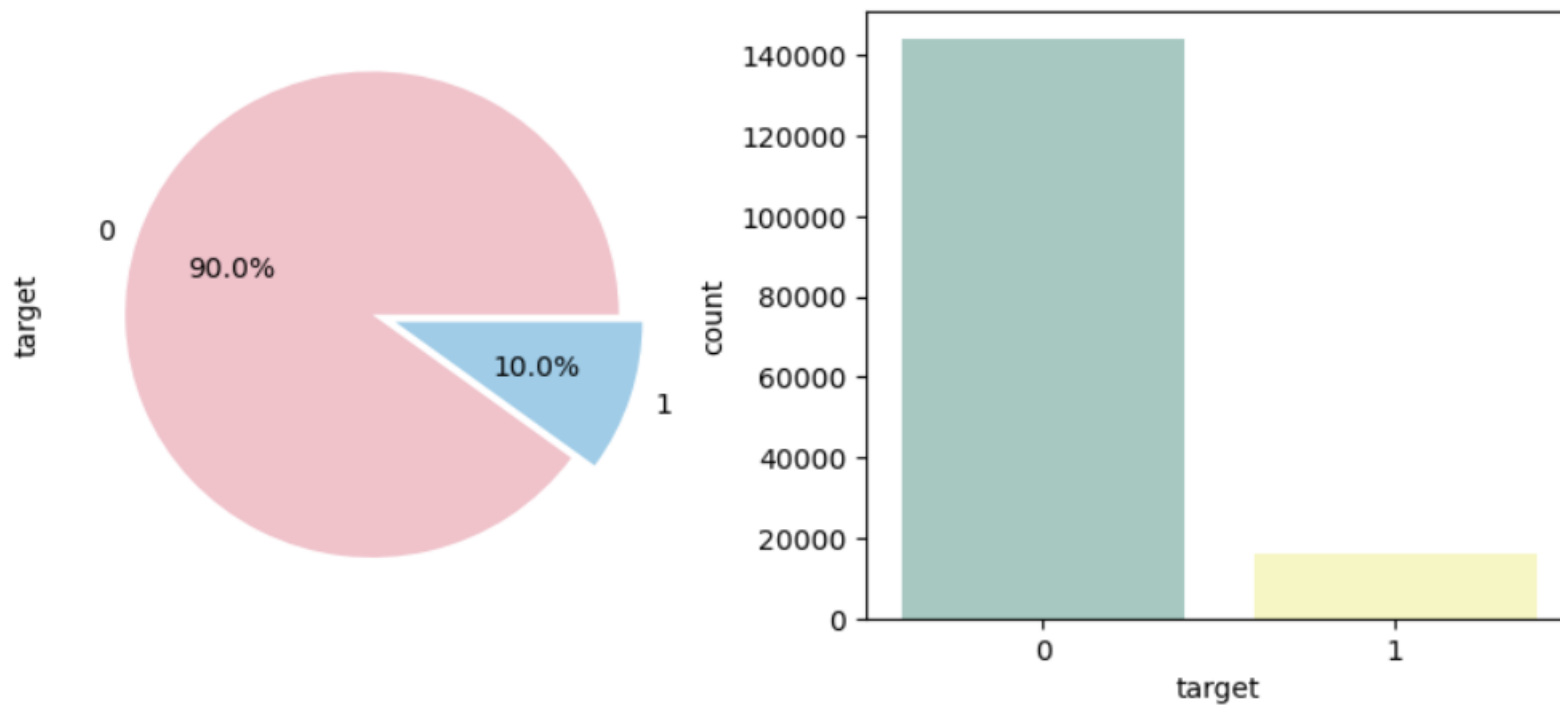
Train과 test 데이터 분포 확인 결과, train 데이터와 test 데이터가 유사한 분포를 가진다는 것을 확인

## 클래스 분포 확인



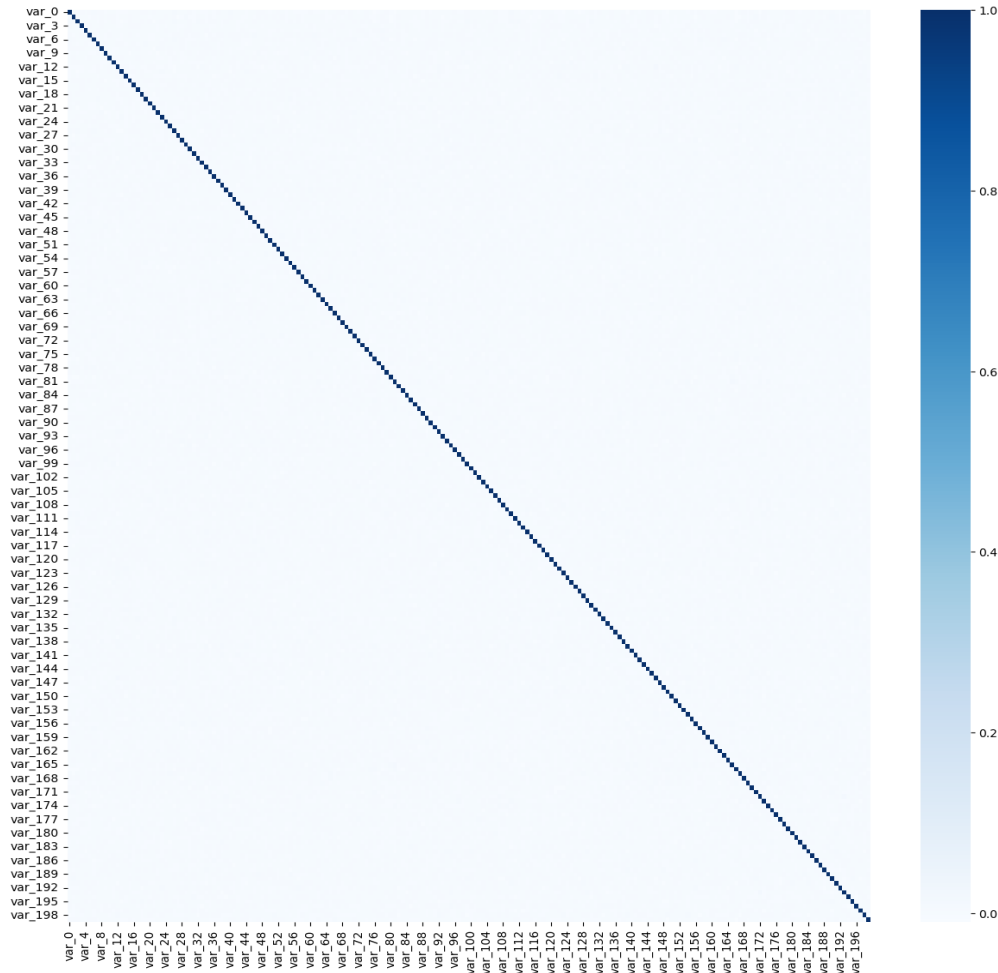
Target이 0과 1로 분류하여 데이터에 spike 값들을 확인

## 클래스 불균형



클래스 0과 클래스 1의 비율이 9:1로 불균형함을 확인

## 변수들 간 상관관계



변수들 간 상관관계가  
존재하지 않음을 확인

## 이상치 확인

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7
0	train_1	0	15.4140	-2.1016	10.4773	4.8941	12.6506	-3.7205	5.1426	17.7048
1	train_2	0	12.3576	-8.1666	11.7785	2.8869	12.3183	-6.9847	4.2671	9.6710
2	train_3	0	9.4142	-8.6132	7.2196	3.2496	10.6550	-3.3245	5.1010	18.5389
3	train_4	0	13.0647	-0.7917	13.0270	8.7865	10.2252	-2.9311	6.7299	11.8682
4	train_5	0	9.5222	-0.2727	8.2173	8.4071	12.7732	-10.3113	4.7486	13.7810
...	...	...	...	...	...	...	...	...	...	...
140017	train_159996	0	8.0801	-2.5523	10.6587	10.6660	8.7552	-10.8144	5.2019	21.1479
140018	train_159997	0	11.8371	-4.9598	12.9172	9.3759	13.0622	2.6849	5.2963	19.8298
140019	train_159998	0	11.3368	1.2790	14.1568	11.9850	11.4859	0.2191	5.9112	19.7031
140020	train_159999	0	3.4660	-0.2570	4.3530	7.2045	11.4988	-7.0838	5.5081	13.7160
140021	train_160000	0	8.5913	-0.8183	11.0295	5.9502	12.9875	-7.8987	5.9785	17.9756

IQR을 사용하여  
이상치 값들 확인



이상치 제거했을 때 row 개수가 160,000에서 140,021로 줄임



# 3

---

## PREPROCESSING

## Feature Selection



### 변수 선택 필요성

변수가 많으면 계산량이 많아지므로 비효율적인 학습을 방지

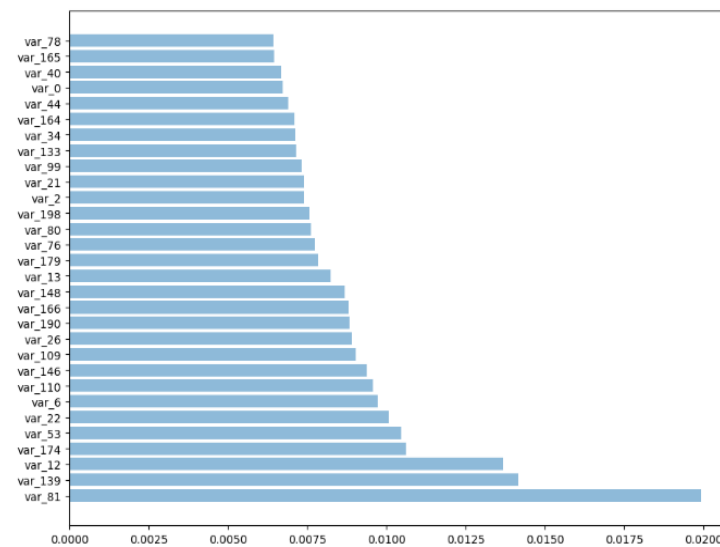
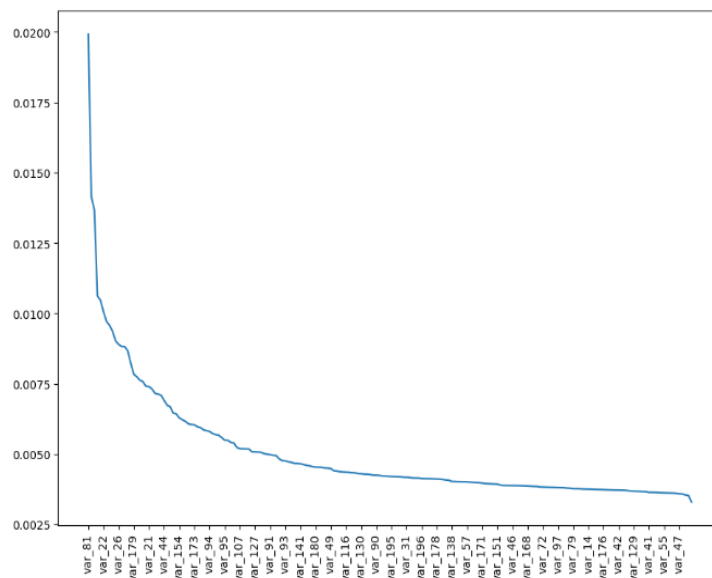
	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...
0	train_1	0	15.4140	-2.1016	10.4773	4.8941	12.6506	-3.7205	5.1426	17.7048	...
1	train_2	0	12.3576	-8.1666	11.7785	2.8869	12.3183	-6.9847	4.2671	9.6710	...
2	train_3	0	9.4142	-8.6132	7.2196	3.2496	10.6550	-3.3245	5.1010	18.5389	...
3	train_4	0	13.0647	-0.7917	13.0270	8.7865	10.2252	-2.9311	6.7299	11.8682	...
4	train_5	0	9.5222	-0.2727	8.2173	8.4071	12.7732	-10.3113	4.7486	13.7810	...
...	...	...	...	...	...	...	...	...	...	...	...
159995	train_159996	0	8.0801	-2.5523	10.6587	10.6660	8.7552	-10.8144	5.2019	21.1479	...
159996	train_159997	0	11.8371	-4.9598	12.9172	9.3759	13.0622	2.6849	5.2963	19.8298	...
159997	train_159998	0	11.3368	1.2790	14.1568	11.9850	11.4859	0.2191	5.9112	19.7031	...
159998	train_159999	0	3.4660	-0.2570	4.3530	7.2045	11.4988	-7.0838	5.5081	13.7160	...
159999	train_160000	0	8.5913	-0.8183	11.0295	5.9502	12.9875	-7.8987	5.9785	17.9756	...

160000 rows × 202 columns

## Feature Selection

### 변수 중요도 (Feature importance)

학습된 모형에 대하여 반응 변수와의 관련성 또는  
예측 관점에서 각 변수들의 영향력을 수치화



## Feature Selection

### 변수 중요도 (Feature importance)

학습된 모형에 대하여 반응 변수와의 관련성 또는  
예측 관점에서 각 변수들의 영향력을 수치화

Mean Decrease in  
Impurity (MDI),  
Gini Importance

상관관계,  
선형회귀계수

Permutation  
Importance

## Feature Selection

### 변수 중요도 (Feature importance)

학습된 모형에 대하여 반응 변수와의 관련성 또는  
예측 관점에서 각 변수들의 영향력을 수치화



Mean Decrease in  
Impurity (MDI),  
Gini Importance



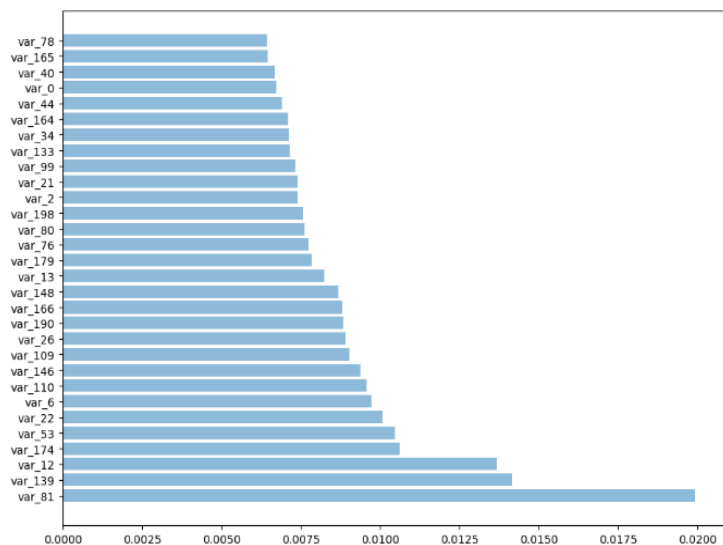
랜덤포레스트에서 개별 나무의 불순도 변화량을  
이용하여 변수 중요도를 계산하는 방법

*model.feature\_importances\_*

## Feature Selection

### 변수 중요도 (Feature importance)

학습된 모형에 대하여 반응 변수와의 관련성 또는  
예측 관점에서 각 변수들의 영향력을 수치화



	Random Forest	LGBM
중요도 기준	0.0039	0.008
변수 개수	136	152

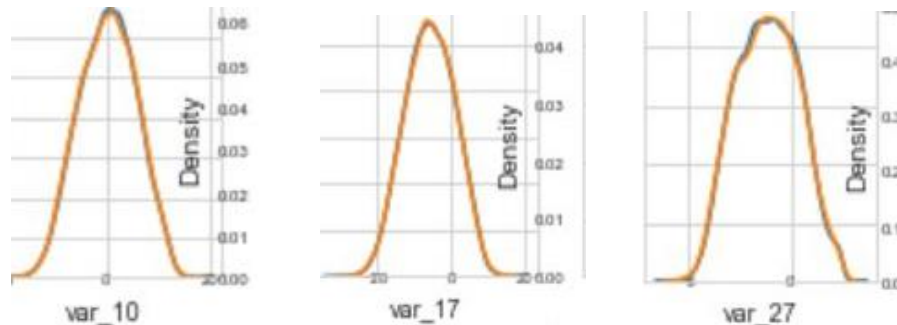
## Feature Selection

### KS TEST (Kolmogorov-Smirnov TEST)

두 데이터의 누적 확률 분포의 차이를 계산하여 데이터의 분포를 비교

H0: 두 분포가 동질적이다 vs H1: 두 분포가 동질적이지 않다

P-value > 0.05 → 두 분포가 동질적



target 0과 1의 분포가 동질적이면 해당 변수가 분류에서 영향력이 없을 것으로 기대



H0을 기각하지 못하는 변수 제외

## Feature Selection

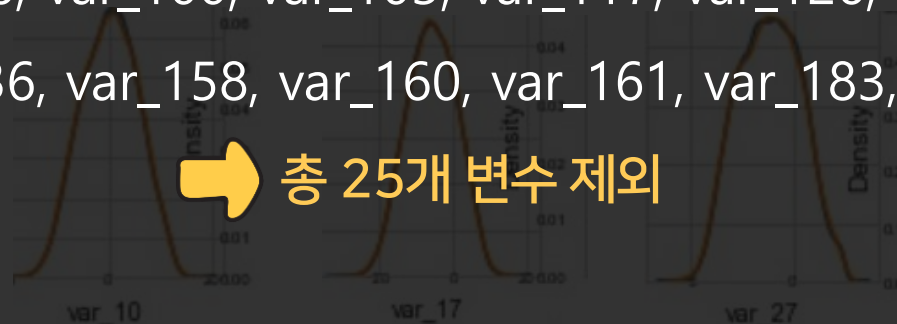
### KS TEST (Kolmogorov-Smirnov TEST)

두 데이터의 누적 확률 분포를 계산하여 데이터의 분포를 비교

$H_0$ : 두 분포가 동질적이다 vs  $H_1$ : 두 분포가 동질적이지 않다

$P\text{-value} > 0.05 \rightarrow$  두 분포가 동질적

var\_7, var\_10, var\_14, var\_17, var\_27, var\_29,  
var\_30, var\_38, var\_41, var\_46, var\_61, var\_79,  
var\_96, var\_100, var\_103, var\_117, var\_126, var\_129,  
var\_136, var\_158, var\_160, var\_161, var\_183, var\_185



➡ 총 25개 변수 제외

target 0과 1의 분포가 동질적이면 해당 변수가 분류에서 영향력이 없을 것으로 기대

➡  $H_0$ 을 기각하지 못하는 변수 제외



## Sampling

### 클래스 불균형

학습 데이터의 클래스 변수가 균일하게 분포하지 않고  
하나의 값에 치우쳐서 편향된 모델을 학습하는 문제



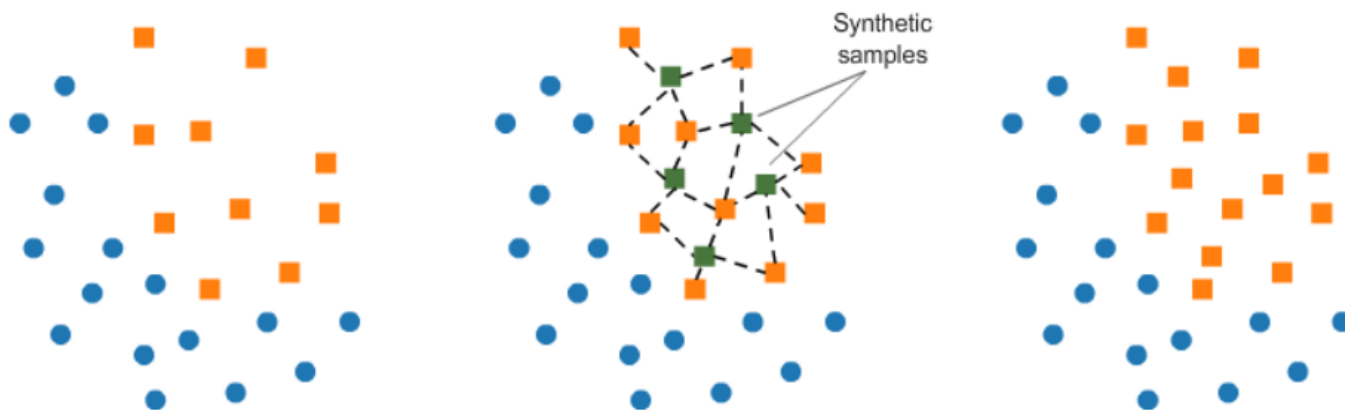
예측하고자 하는 데이터를 다수의 클래스로 분류하고  
소수 클래스에 대한 예측 성능이 매우 저하되는 문제 발생



많은 클래스로 소수의 클래스의 양을 맞추는 Oversampling  
소수의 클래스로 많은 클래스의 양을 맞추는 Undersampling

## Sampling

### SMOTE



Oversampling 기법 중 하나

낮은 비율의 클래스 데이터를 최근접 이웃 알고리즘을 활용해 새롭게 생성

➡ 알고리즘을 기반으로 데이터를 생성하기 때문에 과적합 발생 가능성이 비교적 적음

## Sampling

SMOTE

# 하지만 처참한 성능..

SMOTE는 새로운 샘플을 생성하는 동안  
데이터의 위치를 고려하지 않아  
데이터가 겹치거나 노이즈가 발생하기 때문에  
고차원에서 비효율적

Oversampling 기법 중 하나

낮은 비율의 클래스 데이터를 최근접 이웃 알고리즘을 활용해 새롭게 생성

➡ 알고리즘을 기반으로 데이터를 생성하기 때문에 과적합 발생 가능성이 높음

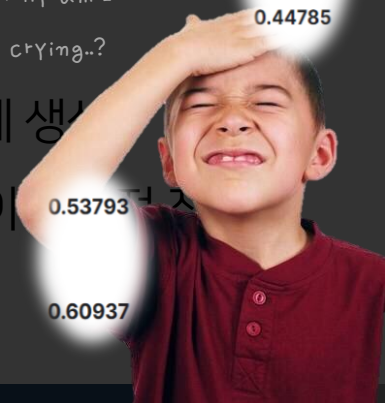
Why am I  
crying..?

0.60633

0.44785

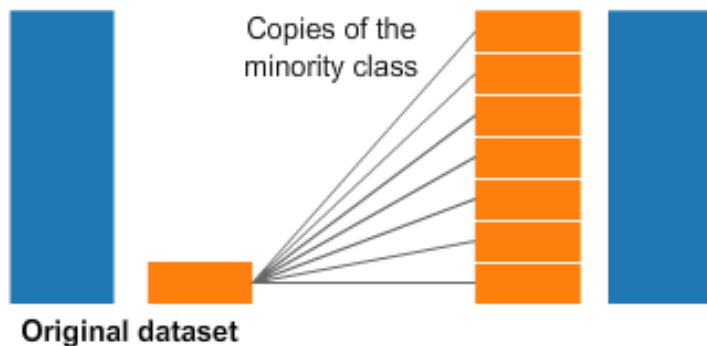
0.53793

0.60937



## Sampling

### Random Oversampling

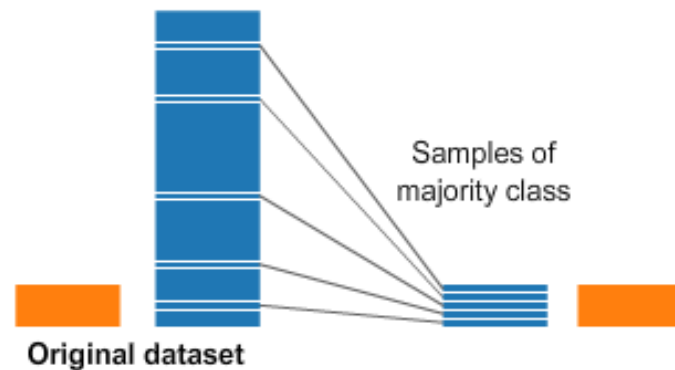


낮은 비중을 차지하는 클래스의 값들을  
임의로 복제해 클래스의 수를 늘리는 방법



노이즈 또는 이상치에 민감  
과적합 발생 가능성

### Random Undersampling



높은 비중을 차지하는 클래스의 값들을  
임의로 제거해 클래스의 수를 줄이는 방법



정보 손실 발생  
샘플의 편향 가능성

4

---

MODELING

## Logistic Regression

→ Odds !

$$\log \left( \frac{\pi_i}{1 - \pi_i} \right) = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_p X_{pi}$$

로그 오즈를 활용한 이진 분류 모델

범주형 피쳐도 설명변수로 사용 가능한 점이 특징

Odds 란?

	심장병 o	심장병 x	Total
흡연	84	2916	3000
비흡연	87	4913	5000

흡연자의 odds =  $84/2916 = 0.0288$

비흡연자의 odds =  $87/4913 = 0.0177$

## Logistic Regression

Minmax scaling

+

All features

F1 Score: 0.6093

Kaggle 예측: 0.71608  

Random  
Over Sampling

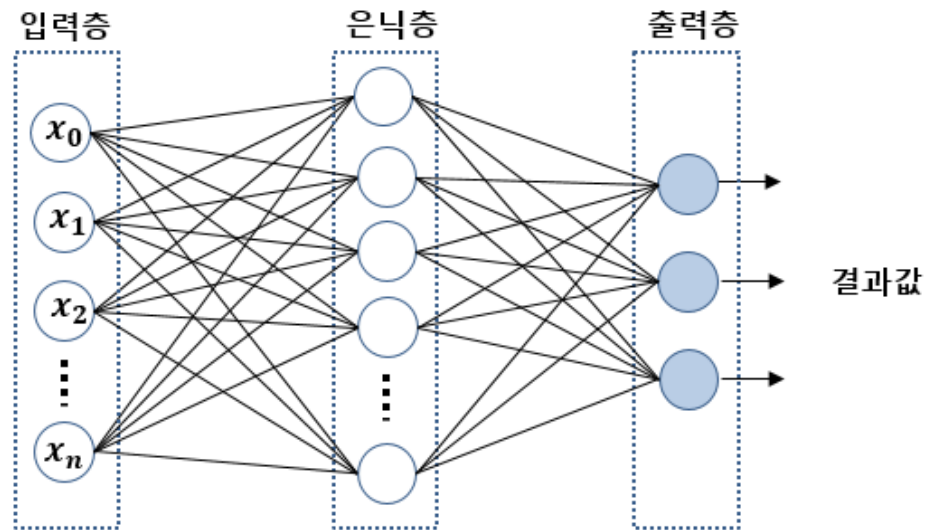
+

All features

F1 Score: 0.7612

Kaggle 예측: 0.71494

## MLP Classifier



순전파와 역전파를 통해 학습하는 인공 신경망 모델



## MLP Classifier

### 하이퍼파라미터

`hidden_layer_sizes` : 은닉층의 레이어 수

`activation` : 활성화 함수

`solver` : optimizer (sgd, adam, lbfgs)

`alpha` : L2 정규화의 강도

`max_iter` : epoch

딥러닝 프레임워크에서는 class의 수를 지정해주지만  
Sklearn의 MLPClassifier는 y 데이터를 확인하여  
자동으로 지정해주는 점이 특징 !

## MLP Classifier

Random Over +  
Random  
Under Sampling

+

All features

F1 Score: 0.7144  
Kaggle 예측: 0.71563



Random  
Under Sampling

+

All features

F1 Score: 0.5713  
Kaggle 예측: 0.69353

## Catboost classifier



# CatBoost

Overfitting을 방지하는데 집중한 트리 기반의 boosting 모델

- 깊이 우선 탐색을 하는 XGBoost와는 다르게 넓이 우선 탐색으로 트리 확장
- 학습 중 학습 데이터의 일부만 사용하여 잔차를 구하는 방식을 통해 모델을 발전시켜 나가는 점이 가장 큰 특징 !
- 범주형 데이터가 많은 경우 학습 성능이 좋다고 알려짐
- Sparse matrix에서는 성능이 좋지 않음

## Catboost classifier

### 하이퍼파라미터

max_depth	: 결정나무의 최대 깊이
l2_leaf_reg	: L2 정규화의 강도
iterations	: 반복 횟수
border count	: 각 피쳐 공간에서 수행할 분할의 수 (클수록 복잡한 모델)
class_weights	: 클래스 불균형이 있는 경우 비율 지정

이외에도 다양한 파라미터가 있어서 목적에 맞게 튜닝 가능 !

## Catboost classifier

Minmax Scaling

+

All features

F1 Score: 0.7379

Kaggle 점수: 0.75598



Random Under  
+ Random Over  
Sampling

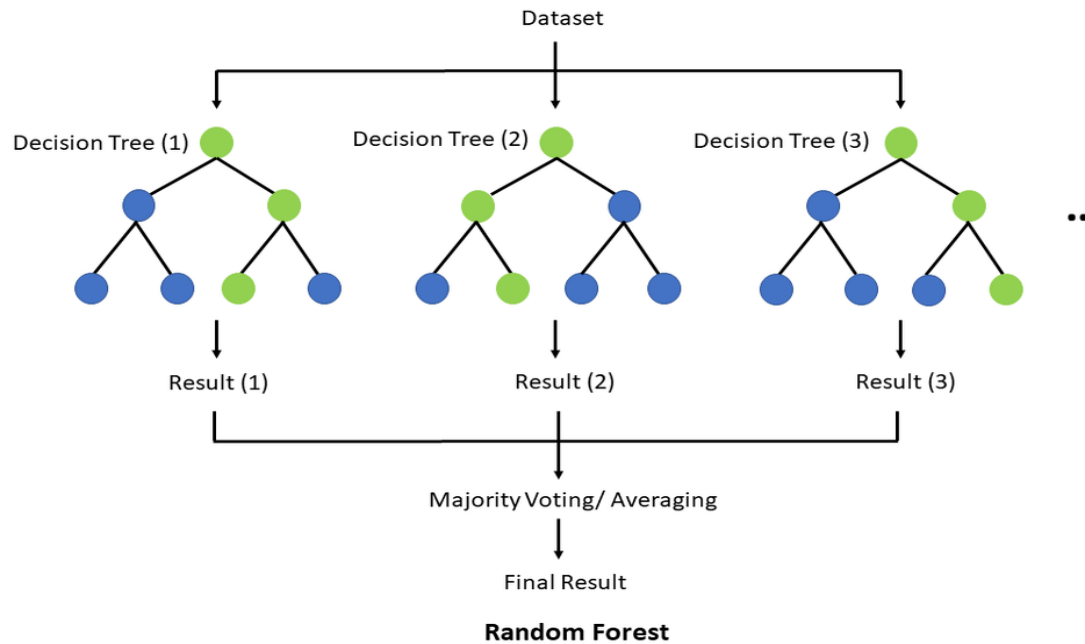
+

All features

F1 Score: 0.8971

Kaggle 점수: 0.74751

## Random Forest




편의를 줄이면서 분산을 낮추는 방향으로 변수를 랜덤으로 선택하여  
개별 트리의 상관성을 줄여서 예측력을 향상하는 앙상블 기법

## Random Forest

### 하이퍼파라미터

n_estimators	: 결정나무의 개수
max_depth	: 결정나무의 최대 깊이
max_features	: 결정나무를 분지할 때 고려하는 특징 수
min_samples_split	: 노드를 분할하기 위한 최서한의 샘플 데이터 수
min_samples_leafs	: 리프노드가 되기 위해 필요한 최소한의 샘플 데이터 수

 Randomized Search CV를 이용한 하이퍼파라미터 튜닝 진행

## Random Forest

### 하이퍼파라미터

n_estimators	: 결정나무의 개수
max_depth	: 결정나무의 최대 깊이
max_features	: 결정나무를 분지할 때 고려하는 특징 수
min_samples_split	: 노드를 분할하기 위한 최서한의 샘플 데이터 수
min_samples_leafs	: 리프노드가 되기 위해 필요한 최소한의 샘플 데이터 수

Randomized Search CV를 이용한 하이퍼파라미터 튜닝 진행

검증하려는 하이퍼파라미터 값들의 범위를 지정 후  
랜덤하게 조합하여 최종적인 최적 파라미터 값을 찾는 기법





## Random Forest

Random  
Under Sampling

+

All features

F1 Score: 0.6502   
Kaggle 예측: 0.6975 

Random  
Under Sampling

+

Feature Importance  
Feature Selection

F1 Score: 0.6713  
Kaggle 예측: 0.6968



## LGBM

## 하이퍼파라미터

n\_estimators : 결정나무의 개수  
max\_depth : 결정나무의 최대 깊이  
min\_child\_samples : 과적합을 방지하기 위한 파라미터  
num\_levels : 개별 트리가 가질 수 있는 최대 리프의 개수  
⋮



랜덤포레스트와 마찬가지로 Randomized Search

CV를 이용한 하이퍼파라미터 튜닝 진행

## LightGBM

Random  
Under Sampling

+

All Features

F1 Score: 0.78837  
Kaggle 예측: 0.75750

Random  
Over Sampling

+

All Features

F1 Score: 0.80206  
Kaggle 예측: 0.76072

Random  
Over Sampling

+



Feature Importance  
Feature Selection

F1 Score: 0.79813  
Kaggle 예측: 0.75942

Random  
Over Sampling

+

KS test  
Feature Selection

F1 Score: 0.80177  
Kaggle 예측: 0.76211  

## Gaussian Naïve Bayes

$$P(A_k|B) = \frac{P(B|A_k)P(A_k)}{P(B)}$$

Posterior probability

Likelihood

Prior probability

Predictor prior probability

변수들 사이의 독립을 가정한 후 베이즈 정리를 적용한 확률 분류 모델  
각각의 변수들이 확률에 독립적으로 기여하는 것으로 간주

➡ 변수 간 낮은 상관계수를 근거로 독립 가정을 충족한다고 판단

## Gaussian Naïve Bayes

$$p(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

연속적인 값을 지닌 데이터에서  
각 클래스의 연속적인 값들이  
정규 분포를 따른다고 가정한 후  
최대 확률을 갖는 클래스로 분류



Anderson-Darling,  
Kolmogorov-Smirnov Test를 통해  
정규성 만족 여부 확인

모델 자체가 낮은 분산을 가지고 있기 때문에  
앙상블 방법을 시도하지 않음

상관관계가 높은 경우,  
변수 선택이 성능 향상에 도움이 되지만  
현재 상황에서는 변수 간 상관관계가 낮아  
변수 선택을 진행하지 않음

## Gaussian Naïve Bayes

Random  
Over Sampling

+

All Features

F1 Score: 0.7834  
Kaggle 예측: 0.7496



Random  
Over Sampling

+

All Features  
(Outlier 제거)

F1 Score: 0.6181  
Kaggle 예측: 0.7486

Random  
Under Sampling

+

All Features

F1 Score: 0.7803  
Kaggle 예측: 0.7475

종아

계획대로 되고 있어 !!!!



## 모델 성능 요약

모델	샘플링	변수 선택	F1 Score	Kaggle 예측력
Logistic Regression	Random Over Sampling	X	0.7612	0.716
MLP Classifier	Random Over + Under Sampling	X	0.7144	0.7156
Catboost	X	X	0.7379	0.7560
Random Forest	Random Under Sampling	O	0.6713	0.6975
LGBM	Random Over Sampling	O	0.8017	0.7621
Naïve Bayes	Random Over Sampling	O	0.7834	0.7496



5

---

PREDICTION

## 최종 모델

Random  
Over Sampling

+

Feature  
Selection  
by KS test

+

LGBM



드루와!

F1 Score: 0.80177

Kaggle 예측: 0.76211

Class	예측 수
0	35838
1	4162

# 6

---

## CONCLUSION

## 의의 및 한계

### ✓ 의의

- 다양한 이진 분류 모델을 직접 구현해 볼 수 있었음
- 변수 선택과 샘플링을 통한 성능 향상을 경험할 수 있었음
- 피셋 학회원들과의 소중한 추억을 쌓을 수 있었음...^^7

대 박



### ✓ 한계

- 이상치 제거, 스케일링을 시도하였으나 유의미한 결과를 얻지 못함
- 컴퓨팅 파워의 한계로 보다 무거운 모델을 시도하지 못함



## 방학 세미나 후기



바쁘다 바빠 ~ 현대 사회! 인턴

벌려놓은 일만 많아서 인턴과 병행하느라 만나서 회의 참여도 못하고... 팀원들에게 미안함이 남습니다 흑흑... 그럼에도 불구하고 같이 파이팅해준 수빈 언니, 은선이, 건우 오빠에게 감사드립니다 ;ㅂ; 전처리부터 모델링까지 전체 과정을 가이드 없이 진행해보는 경험이 흔치 않은데, 많이 배워간 것 같습니다. 방세 3팀 최고야~~

3팀 귀요미 등장!! >ㅂ<



우와,, 5일간의 여정을 마치다니,,! 우리 모두 정말 수고 많았습니다. 학관에서 출석률이 1등인 우리팀 정말 많은 변수 처리와 모델 기법들을 머리 싸매고 같이 고민하면서 진짜 고생 많았습니다. 캐글 리더보드를 보면서 감정이복을 같이 경험했는데 ㅋㅋ 그걸로 동기부여도 되고 팀 간 끈끈함이 생겨서 너무 좋았습니다. 우리 다음학기도 파이팅~~ 방세 3팀 짱짱!! (내 11시간 어짜누)

동태눈을 뜨고 다니는 여자..



험난했던 5일.. 방세 3팀 너무 수고하셨습니다! 강남과 학관에서 함께 고군분투한 건우오빠와 수빈언니, 바쁜 와중에도 열심히 해준 민이 다들 너무 고마울 따름입니다 ♥ 덕분에 많이 배우고 힘든 와중에도 즐거웠어요! 피셋에서 남은 반년도 파이팅! (이 중 누구는 1년이겠지만 ㅋㅋ 일단 난 아님 ~)

건우의 알기 ~



5일간이지만 많은 일이 있었다. 하루는 수빈이랑 은선이랑 강남역에 갔다가 카페만 5군데를 돌다가 쓰리팝에 갔다. 수빈이는 쓰리팝에 아직도 11시간이 남아있다. 민이는 세미나를 위해 월차를 썼다. 피셋이 잘못했다. 마지막날 제출코드를 다 써서 보냈는데 은선이 성능을 올려왔다. 종교를 은선교로 바꾸기로 했다. 나름 다사다난?했지만 좋은 팀원들을 만나 너무 즐거운 시간이었다.





THANK YOU

