빅데이터 자료분석[202202-STS4044-001] 최종발표 3조

「핀다」앱 사용성 데이터를 통한 대출신청 예측모델 개발

12180627 산업경영공학과 손준영 / 12183133 경영학과 김예린 / 12161859 통계학과 권하준 / 12171924 통계학과 김지훈

INDEX

2022 빅콘테스트 데이터분석리그 퓨쳐스 부문
데이터셋을 통한 문제해결

「핀다」 앱
사용성 데이터를 통해
대출신청 여부 예측

finda

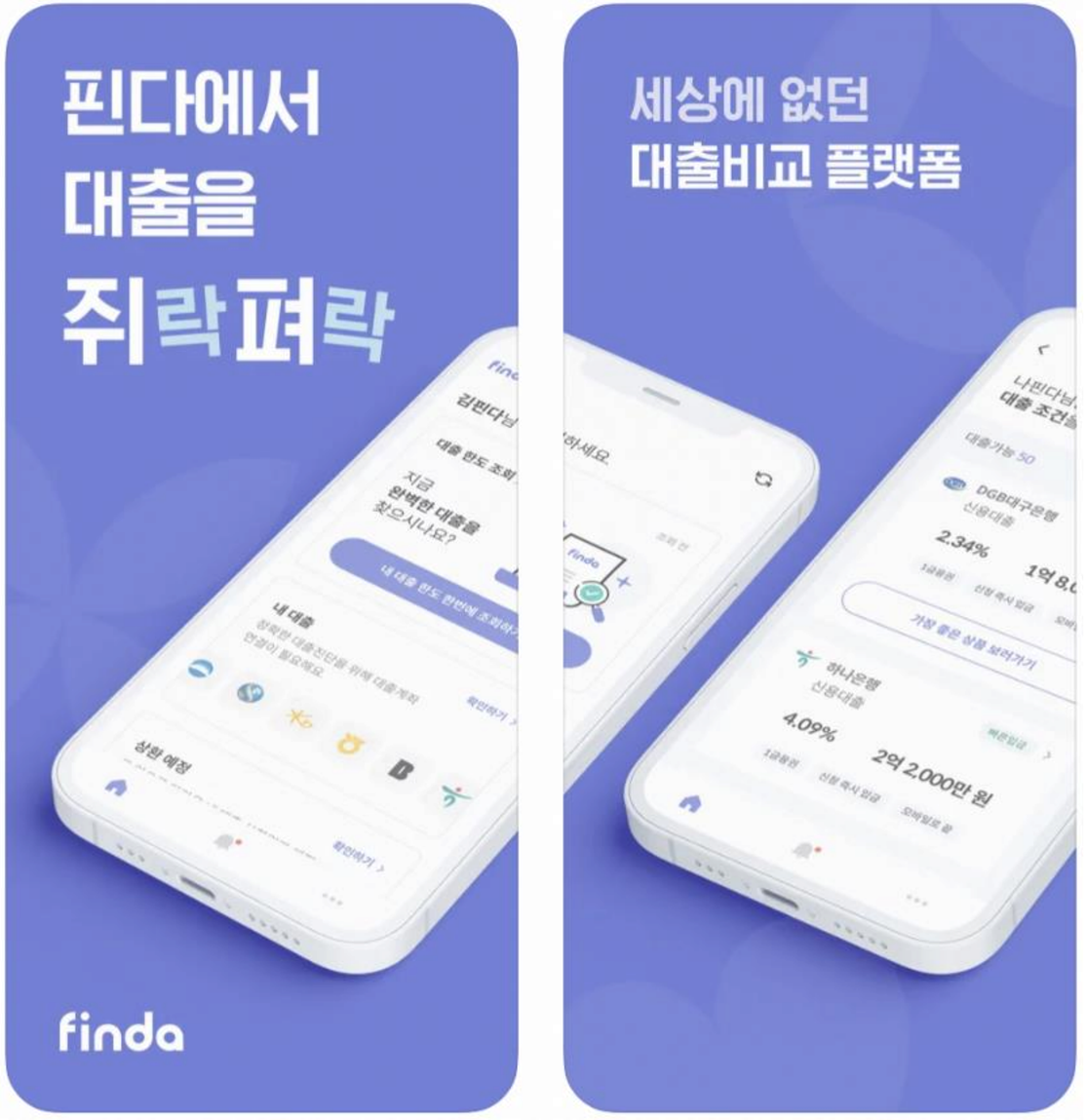
- **예측모델 기반** 탐색적 데이터 분석
- **대출신청 고객 분류**를 통한 고객특성 분석
- 앱 사용성 증진을 위한 **활용방안** 제시

1. 대출비교 / 신청대행 플랫폼

- 신용정보 등의 개인정보를 바탕으로
금융사별 대출한도와 금리를 제공하는 위탁계약업체
- 핀다 전용 금리할인
- 여러 금융사에 대출조건을 조회해도 신용조회는 1건

2. 기대출 관리 서비스 제공

- 마이데이터 기반 기대출 관리
- 대출상환 일시 안내
- 선순위 상환 대출 제안



핀다 앱 시뮬레이션을 통해 **분석 대상 피쳐** 선정

1. 나이(age)
2. 신용 점수(credit_score)
3. 대출 희망금액(desired_amount)
4. 기대출 수(existing_loan_cnt)
5. 기대출 금액(existing_loan_amt)
6. 대출 한도(loan_limit)
7. 대출 금리(loan_rate)

8. 대출 신청 여부(is_applied)

- **Shape**: (8014178, 8)

Missing Value

유추 가능한 경우

user_id	birth_year	gender
49072	NaN	NaN
49072	1985.0	0.0

user_id를 통해 유추 가능한 고정적인 정보

→ *적절하게 대체 or 삭제*

신용점수(credit_score)

insert_time	credit_score
2022-03-29 10:14:05	NaN
2022-05-20 16:49:25	560.0

신용점수는 시간에 따라 변화할 수 있는 가변적인 데이터

→ *가장 가까운 날짜의 신용점수로 대체*

기타 결측치 처리

대출한도나 대출금리의 경우,
은행에서 제공해주지 않은 데이터이므로
무시해도 좋다고 명시 → *결측치 삭제*

Missing Value

기대출 수

```
dat1['existing_loan_cnt'].isnull().sum()
```

146290

```
q="""SELECT * FROM dat1 WHERE user_id IN (SELECT user_id FROM dat1 WHERE existing_loan_cnt IS NULL)"""
exloannone=sqldf(q,locals())
# 기대출수가 결측치인 값을 보유한 user_id의 데이터를 전부 불러옴.
```

```
len(exloannone)
```

146290

기대출 수 피처가 결측치인 데이터 수
= 기대출 수 피처에 결측치 값이 있는 사용자의 수

→ *처음부터 기대출이 없었던 것으로 판단*

```
1  dat1['existing_loan_cnt'].describe()
```

```
min       1.000000e+00
```

기대출 수의 결측치를 제외한 분포는 최소값이 1
기대출 횟수가 아예 없는 0회에 데이터는 존재 x

→ *기대출 수 결측치 0으로 대체*

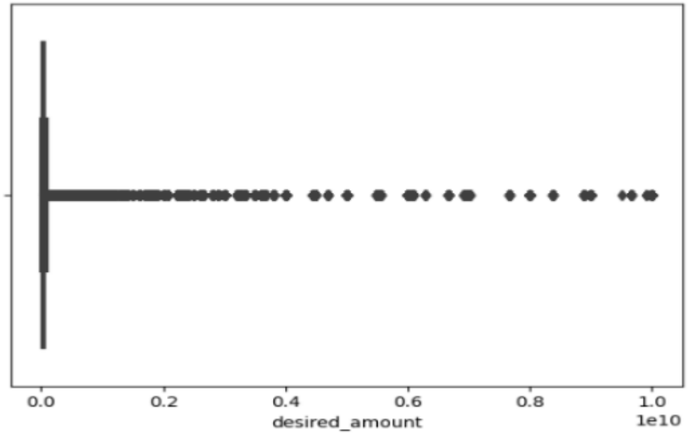
기대출 금액

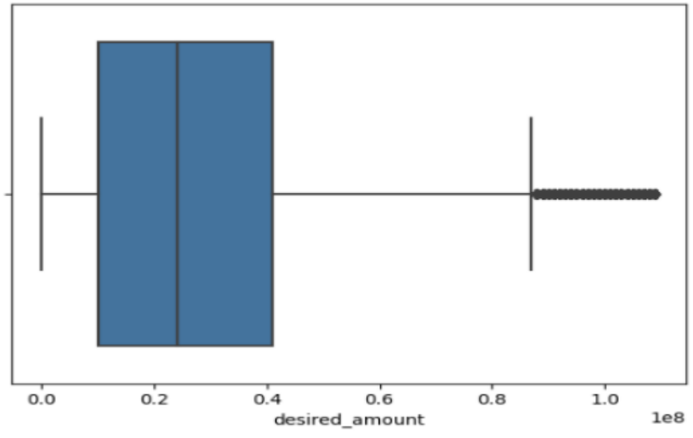
existing_loan_cnt	existing_loan_amt
1.0	None
1.0	None
1.0	None
1.0	None
1.0	None

기대출 수가 1 이상인 경우
해당 정보에 대한 예측이 어려움

→ *결측치 삭제*

Outlier





1. Boxplot으로 이상치 확인

연속형 변수에 대하여 *Boxplot*을 이용하여 데이터를 파악

2. 로그 변환

데이터의 분포를 파악하기 어려운 변수

*로그 변환*을 통해 데이터 분포를 손쉽게 파악

3. IQR 이용해 이상치 제거

```
iqr = q3 - q1
df = df[(df[column] < q3 + 1.5 * iqr) & (df[column] > q1 - 1.5*iqr)]
```

Feature Engineering

existing_loan_cnt	existing_loan_amt
3.0	76000000.0
3.0	76000000.0
2.0	64000000.0
2.0	28000000.0
2.0	28000000.0
...	...
0.0	0.0
0.0	0.0
0.0	0.0
0.0	0.0
0.0	0.0

mean_exloan
25333333.0
25333333.0
32000000.0
14000000.0
14000000.0
...
0.0
0.0
0.0
0.0
0.0

기대출 수가 1번, 2번, 3번인 사람을 구별하거나
총 기대출 금액을 계산하는 것은 의미 x

→ *평균 기대출 금액(mean_exloan)* 피처 생성

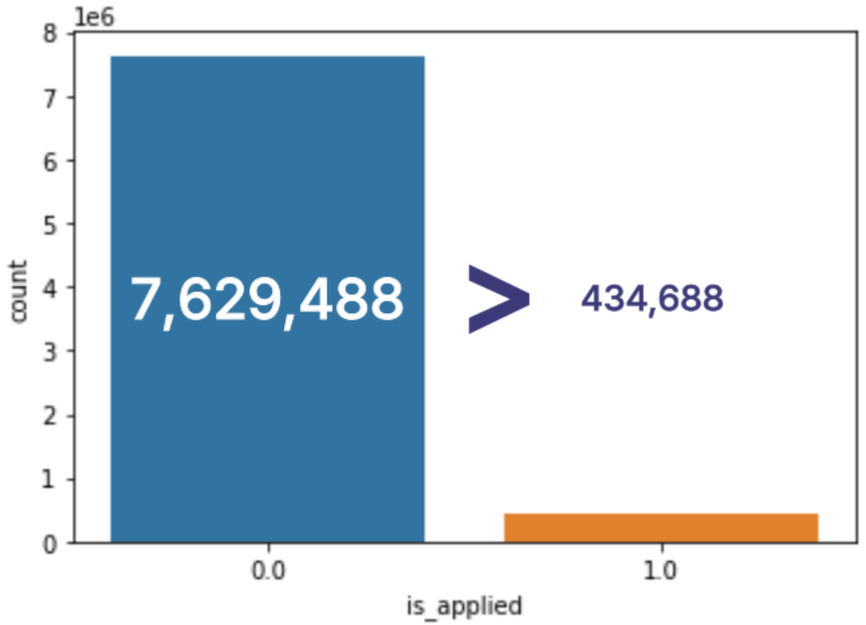
Min-Max Scaling

	credit_score	desired_amount	existing_loan_amt	loan_limit	loan_rate	is_applied	age	mean_exloan
0	670.0	7.397940	76000000.0	3000000.0	14.5	0.0	43.0	25333333.0
1	670.0	7.397940	76000000.0	1000000.0	19.9	0.0	43.0	25333333.0
2	540.0	7.176091	64000000.0	30000000.0	17.9	1.0	46.0	32000000.0
3	710.0	7.113943	28000000.0	9000000.0	9.4	0.0	22.0	14000000.0
4	710.0	7.113943	28000000.0	10000000.0	13.8	0.0	22.0	14000000.0

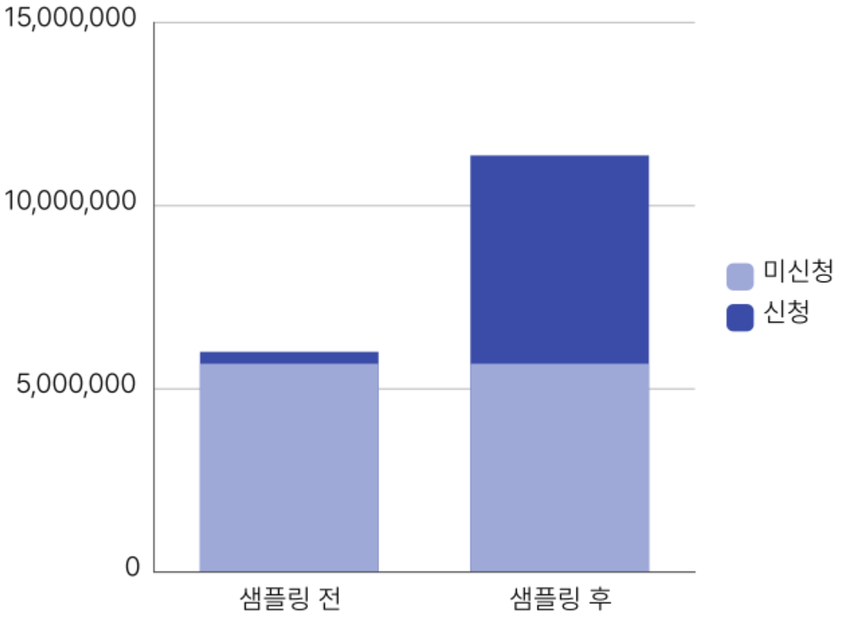
	credit_score	desired_amount	existing_loan_amt	loan_limit	loan_rate	is_applied	age	mean_exloan
0	0.633333	0.509254	0.010117	0.0003	0.702703	0.0	0.342857	0.013018
1	0.633333	0.509254	0.010117	0.0001	0.994595	0.0	0.342857	0.013018
2	0.488889	0.428437	0.008520	0.0030	0.886486	1.0	0.385714	0.016444
3	0.677778	0.405797	0.003727	0.0009	0.427027	0.0	0.042857	0.007194
4	0.677778	0.405797	0.003727	0.0010	0.664865	0.0	0.042857	0.007194

모든 변수들의 스케일을 동일하게 맞추기 위해
Min-max scaling 진행

Oversampling



17:1의 불균형 데이터셋



1:1의 균형 데이터셋

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

불균형이 심한 데이터에서는
비중이 높은 클래스에 대한 예측만 하더라도
높은 정확도가 나올 수 있음→ 부적합한 평가지표

F1 - score

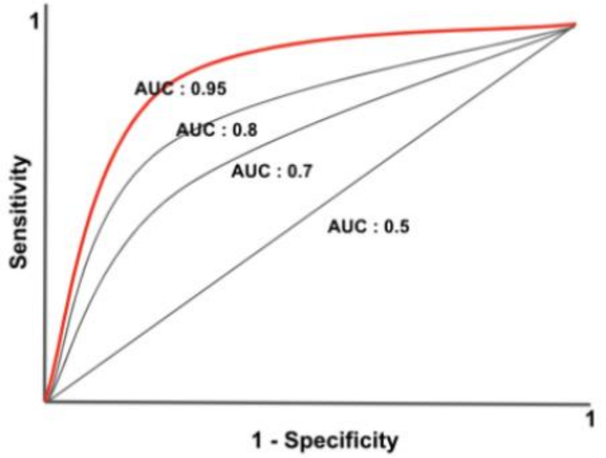
$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

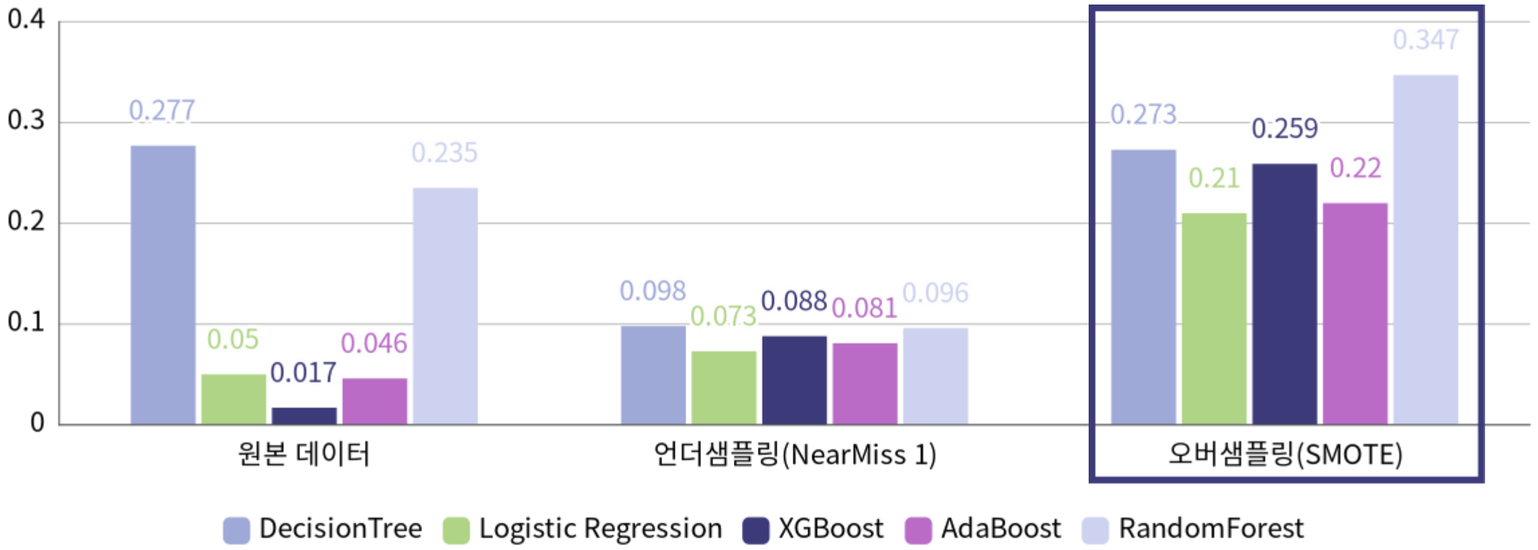
정밀도와 재현율의 수치가
적절하게 조합되어 사용
→ 1에 가까울수록 성능이 좋음

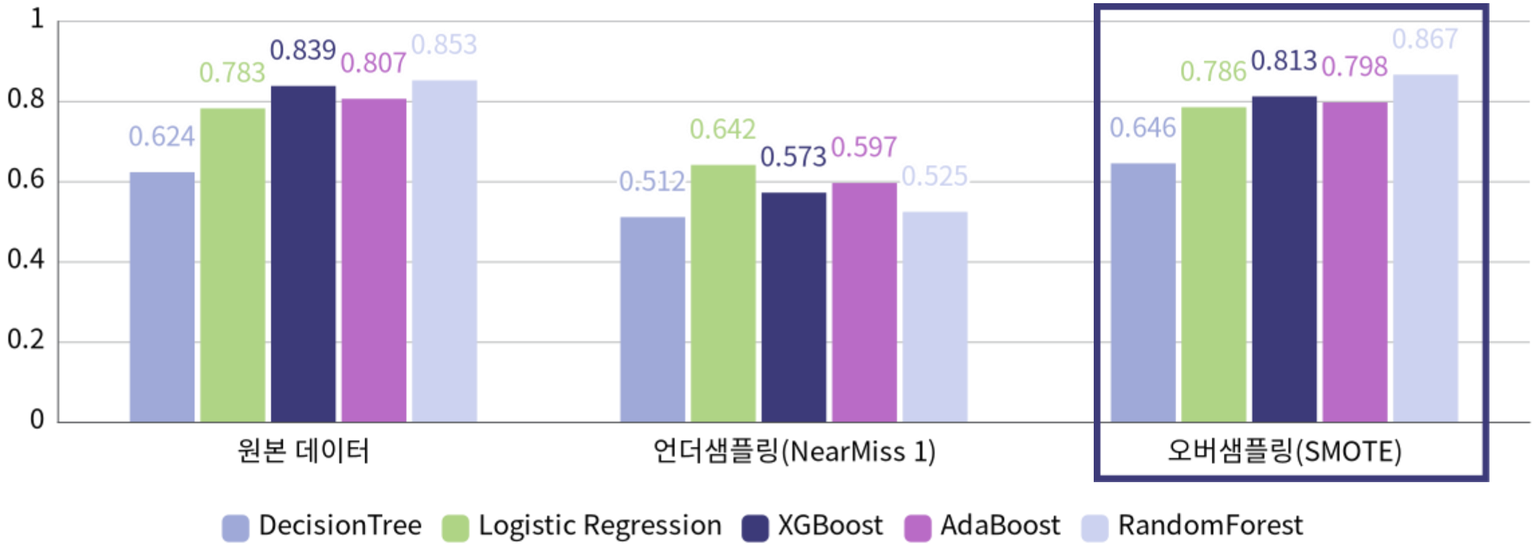
ROC-AUC score



이진 분류의 예측 성능 측정에서
중요하게 사용되는 지표
→ 1에 가까울수록 성능이 좋음

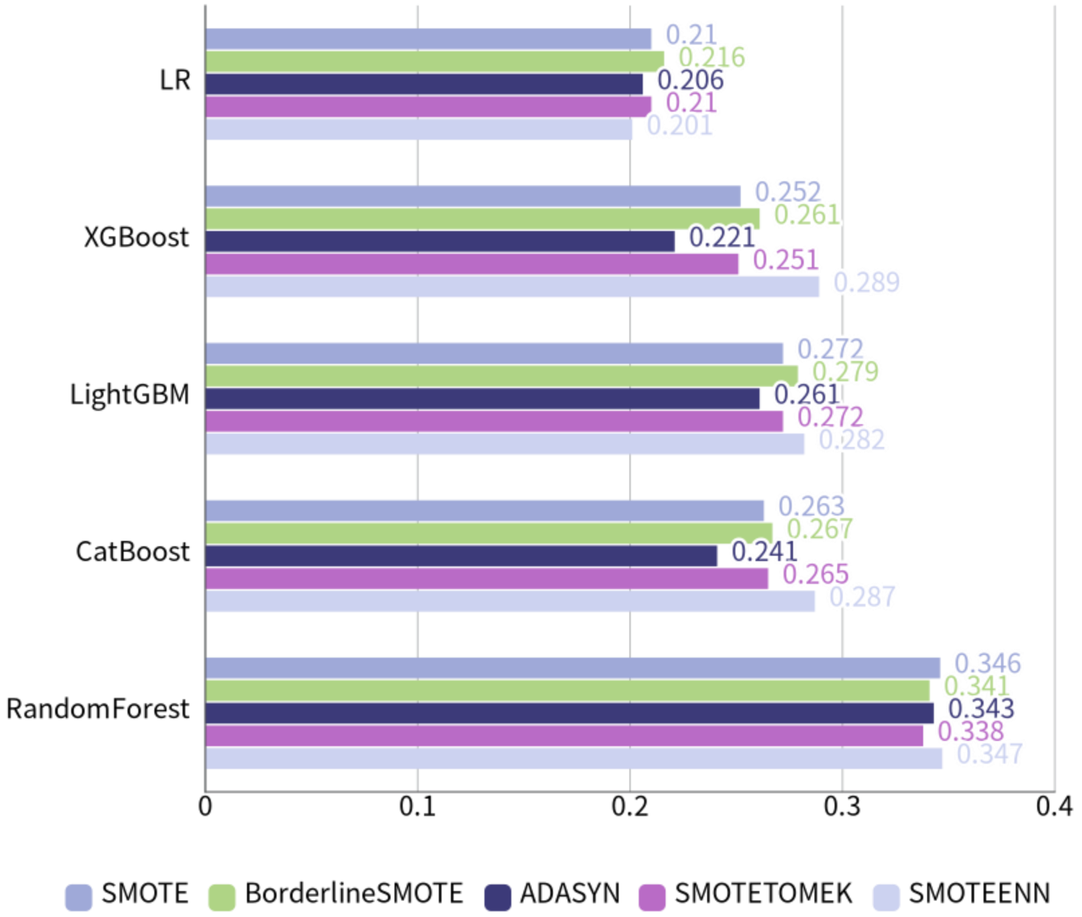
✔ F1-score 기준
샘플링 성능





✔ ROC - AUC 기준
샘플링 성능

✔ F1-score 기준 샘플링 성능



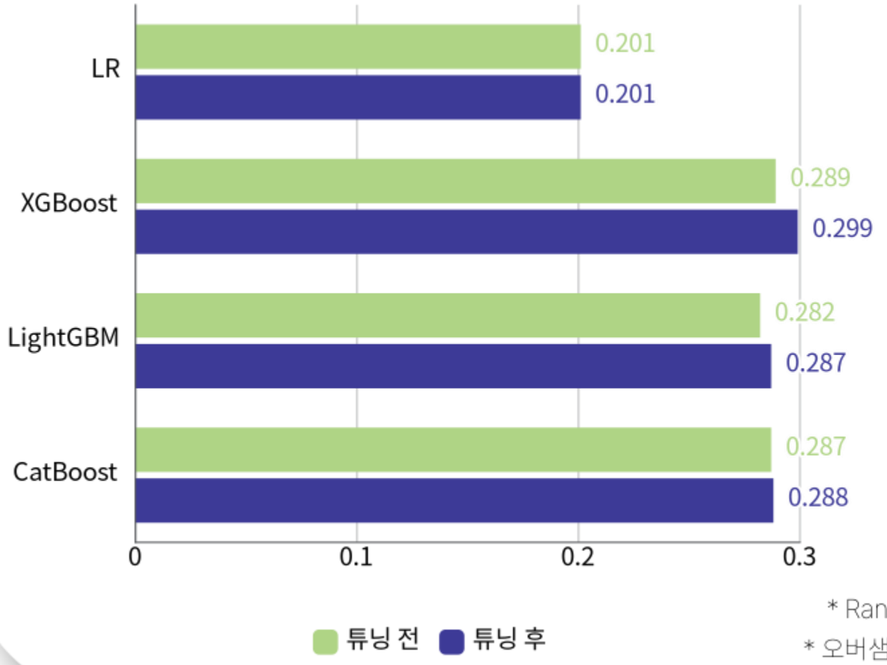
✔ ROC-AUC score 기준 샘플링 성능

	Hyper Parameter
Logistic Regression	'C' : 1, 'penalty' : l2
XGBoost	'learning_rate' : 1, 'max_depth' : 10, 'min_child_weight' : 5
LightGBM	'learning_rate' : 0.2, 'max_depth' : -1, 'min_child_samples' : 15, 'num_leavs' : 80
CatBoost	'depth' : 6, 'iterations' : 1000, 'l2_leaf_reg' : 1e-19, 'leaf_estimation_iterations' : 10, 'leaf_estimation_iterations' : 10

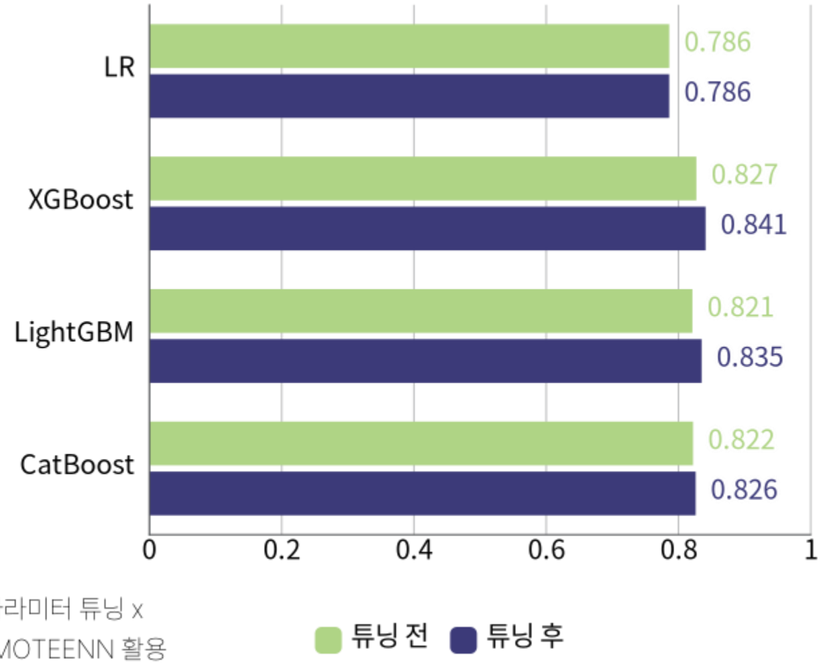
약 1,200만개의 데이터에 대한 튜닝을 위해 **GPU 가속**이 가능한 **부스팅 모델** 위주로 GridsearchCV 진행

최적 파라미터 기반
예측모델 성능비교

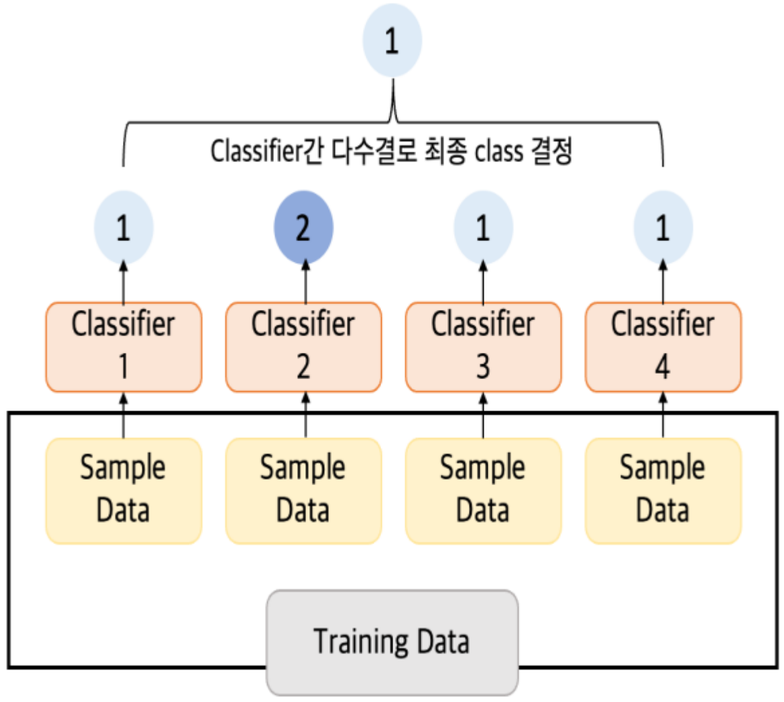
F1 score 기준



ROC-AUC score 기준

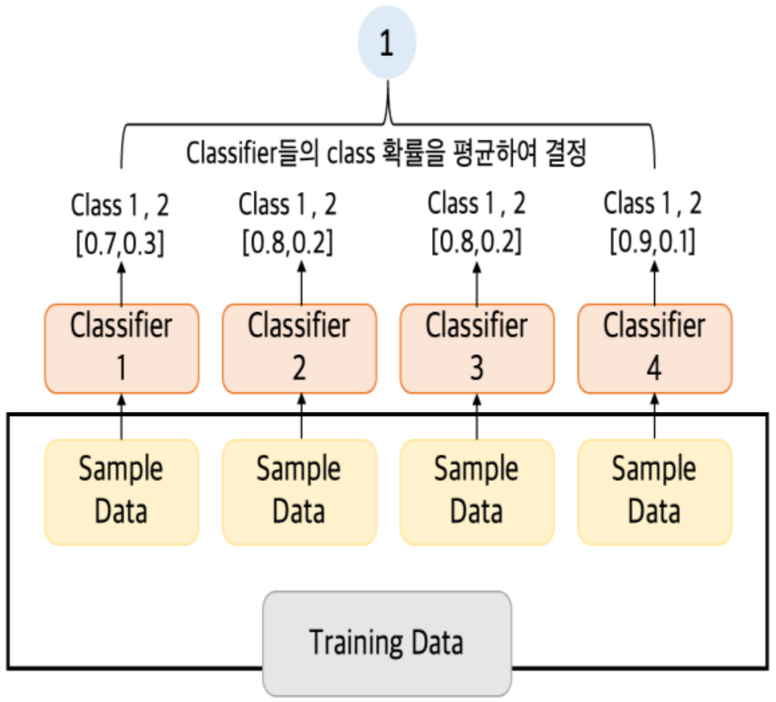


* Random Forest 파라미터 튜닝 x
* 오버샘플링 기법은 SMOTEENN 활용



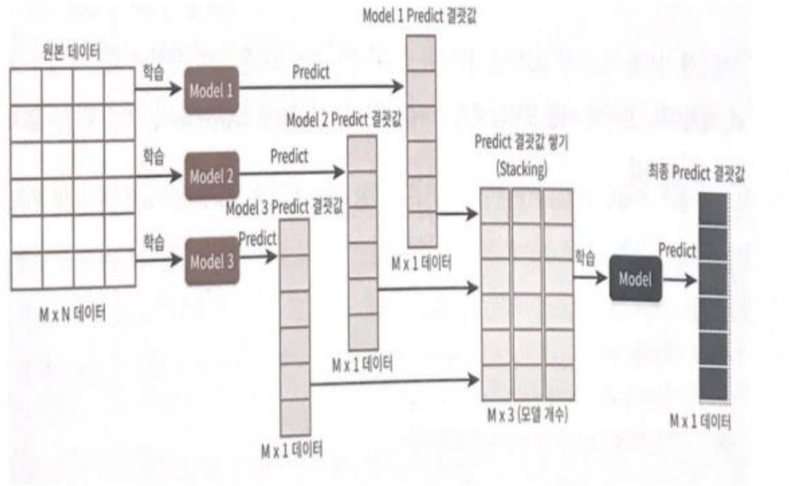
하드 보팅(Hard Voting)

분류기들의 레이블 값 결정 확률을 모두 더하고
이를 평균해서 이들 중 확률이 가장 높은 레이블 값을
최종 보팅 결괏값으로 선정



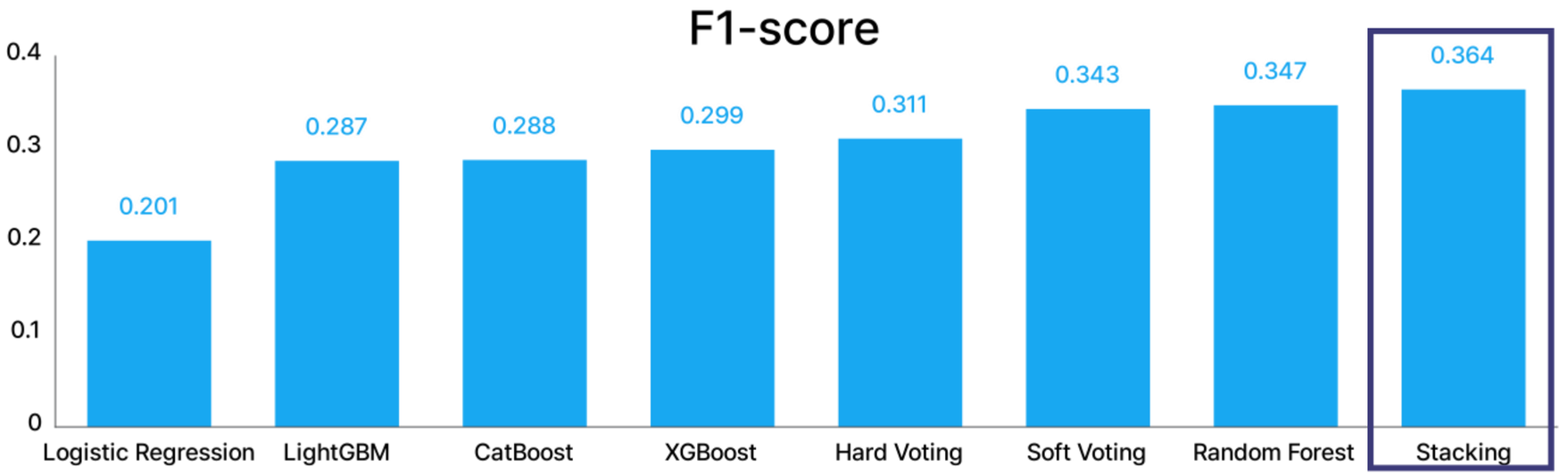
소프트 보팅(Soft Voting)

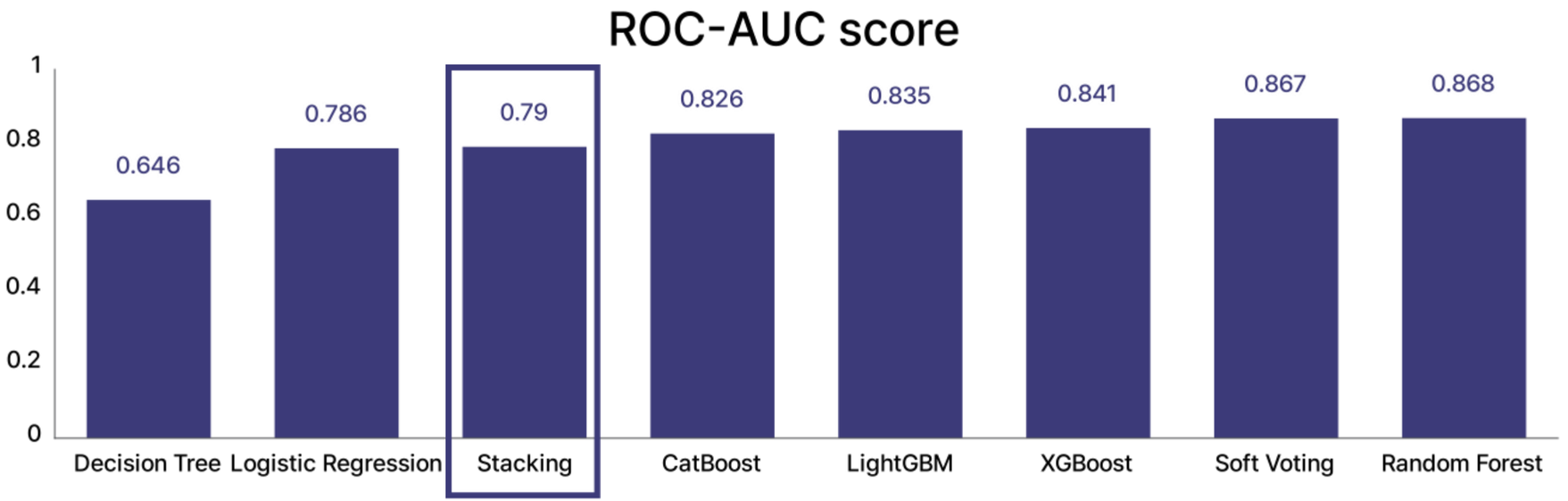
예측 결괏값들중 다수의 분류기가 결정한 예측값을
최종 보팅 결괏값으로 선정



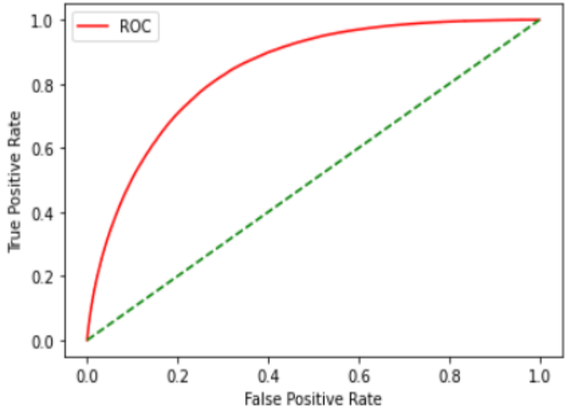
스태킹(Stacking)

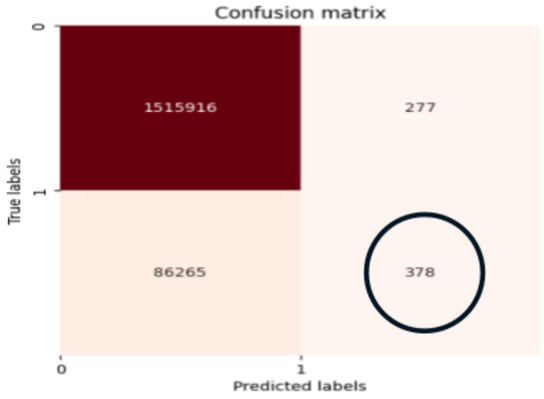
여러 가지 다른 모델의 예측 결과값을
다시 학습 데이터로 만들어 다른 모델(메타 모델)로
재학습시켜 결과를 나타내는 방법


F1-score
0.4
0.3
0.2
0.1
0
0.201
0.287
0.288
0.299
0.311
0.343
0.347
0.364
Logistic Regression
LightGBM
CatBoost
XGBoost
Hard Voting
Soft Voting
Random Forest
Stacking


ROC-AUC score
1
0.8
0.6
0.4
0.2
0
0.646
0.786
0.79
0.826
0.835
0.841
0.867
0.868
Decision Tree
Logistic Regression
Stacking
CatBoost
LightGBM
XGBoost
Soft Voting
Random Forest

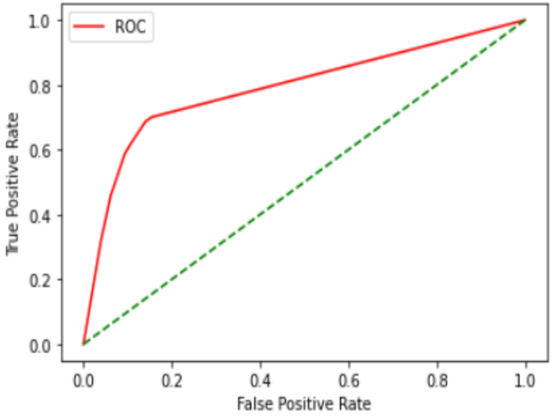
원본 데이터
LightGBM

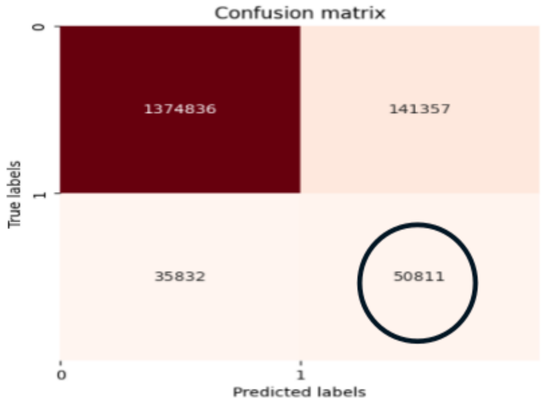




F1 Score : 0.009
ROC-AUC score : 0.841
ACC : 0.946

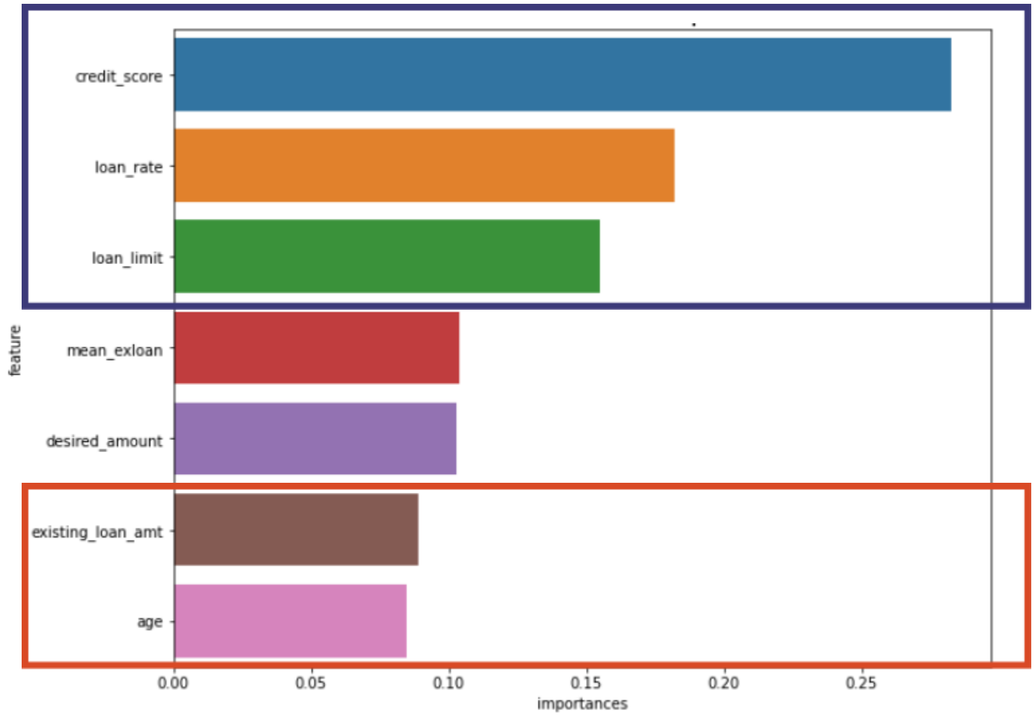
최종 예측 모델
Stacking(Meta Model : LightGBM)

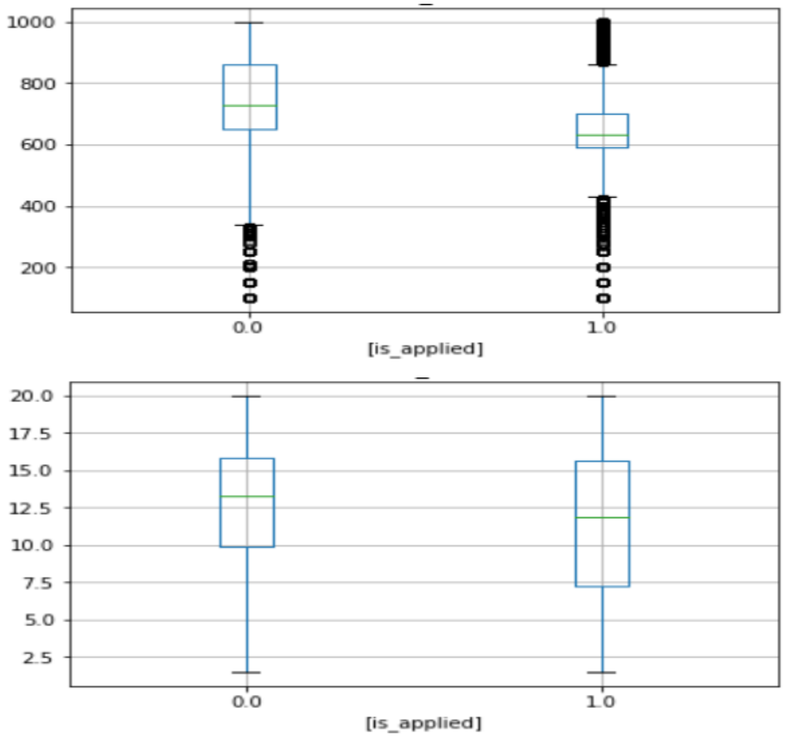




F1 Score : 0.364
ROC-AUC score : 0.790
ACC : 0.889

정확도와 ROC-AUC score는 소폭 하락했지만 F1 score가 대폭 상승하였고,
실제 86,600개의 1 class에 대하여 378개 맞추던 것이 50,811개까지 맞추면서 성능이 대폭 향상





- 대출 신청을 하는 사람들의 경우 신용 점수가 낮은 경우가 많음
- 대출 금리가 낮을수록, 대출 한도가 높을수록 대출 신청까지 이어질 가능성이 높음
- 총 기대출 금액보단 기대출 당 평균 기대출 금액이 더 중요하게 작용
- 대출 신청을 하는데 생각보다 나이는 중요하지 않게 작용

피처 선택 과정

기존 피처들에서 유의미한 인사이트를 찾지 못함
앱을 직접 사용해보면서 feature selection 진행
→ 분석 대상에서 제외된 피처가 꽤 많음

군집 분석

예측 분류 모델 개발에 더해 계획 했던
고객 특성별 군집 분석을 수행하지 못함

Random Forest 모델의 활용

분석 환경이 부족해 추가적인 활용 불가
추가적인 복합 모델 활용 및
파라미터 튜닝을 하지 못함

Q&A

경청해 주셔서 감사합니다.

빅데이터 자료분석 최종발표 3조

「핀다」앱 사용성 데이터를 통한
대출신청 예측모델 개발

손준영 김예린 권하준 김지훈