

Lending Club Default prediction

3조 Treatment

김 | 김민정 | 박 | 박 | 박 | 양

Contents

01. 서론

:

Lending Club

데이터 EDA

목적함수 설정

선행연구

02. 본론

:

데이터 전처리

분석 요약

분석 방법론

03. 결론

:

예측 결과

최종 모델 선정

한계점

부록



서 론

Lending Club 소개

서론



렌딩클럽이란?

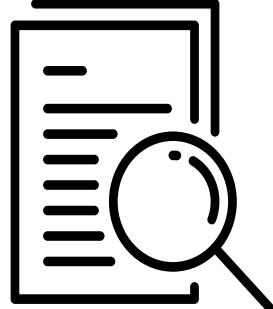
돈이 필요한 사람과 여유 자금을 운용하고 싶은 사람을 연결해주는 P2P대출 업체.

대출이 필요하면 렌딩클럽 홈페이지에 들어가 신청서를 작성한다. 렌딩클럽은 이 중 10% 정도만 추려내 대출 가능자를 정하고 이들에게 다시 A~G까지 7단계의 신용 등급을 매겨 온라인 대출 장터에 올려놓는다. 돈을 굽리고 싶은 개인 투자자들은 대출 신청자 명단을 보고 자신이 원하는 사람에게 투자한다. 이때 투자 금액은 최소 25달러를 기준으로 소액 분산투자하게 된다. 대출금리는 신용 등급에 따라 연 6.78~9.99% 수준이다. 몇몇 대출자의 채무 불이행을 감안한다고 하더라도 제로 금리에 가까운 은행 이자에 비해 높은 수익률을 기대할 수 있다. 렌딩클럽은 이 과정에서 대출금의 1~3%를 수수료로 받는다.

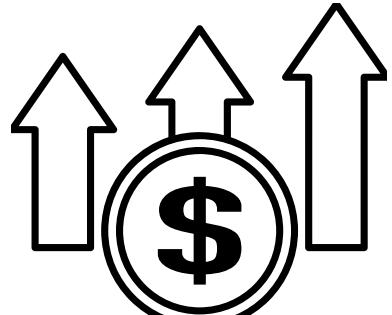
출처 : 한경닷컴

비즈니스 이해

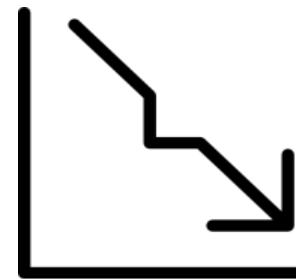
서론



Lending Club은 대출 신청을 받으면 신청자의 프로필을 보고 대출 승인 여부를 결정해야 한다. 결정에는 두 가지 유형의 위험이 관련있다.



만약 신청자가 대출금을 상환할 가능성이 있는 경우, 즉 채무 불이행 가능성이 없는 경우, 대출을 승인하는 것이 회사에 사업 이익을 발생시킨다.



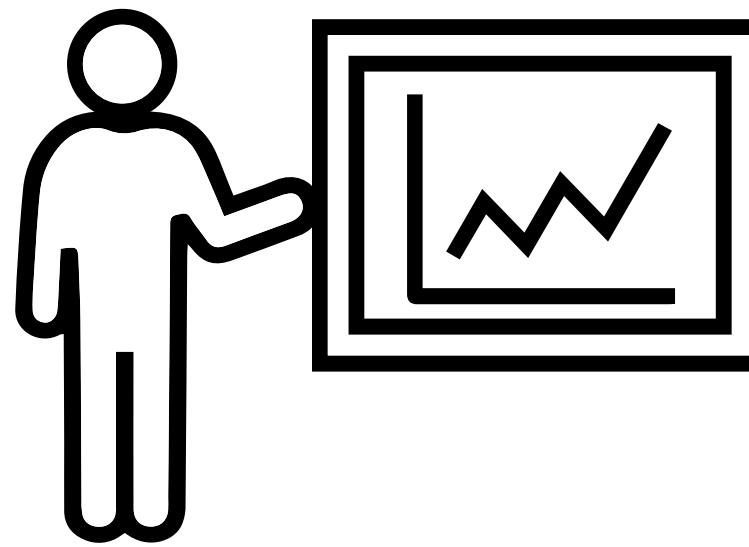
신청자가 대출금을 상환할 가능성이 없는 경우, 즉 채무 불이행 가능성이 있는 경우, 대출을 승인하면 회사에 금전적 손실을 초래할 수 있다.



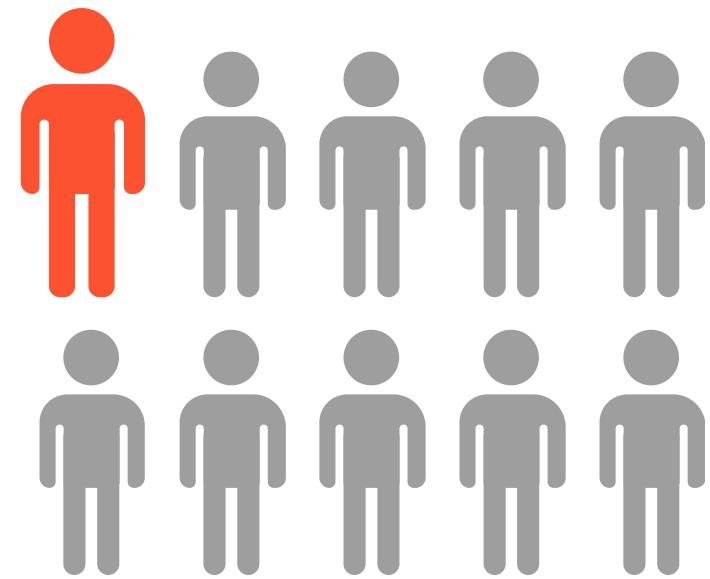
제공된 데이터에는 과거 대출 신청자에 대한 정보와 '채무불이행' 여부가 포함되어 있다. 위험한 신청자(채무 불이행 가능성이 높은자)에게 대출해주지 않는 cutoff를 지정해주고, 이에 따른 회사의 이익과 손실을 분석해주는 목표를 설정하였다.

비즈니스 목표

서론



데이터 분석



차용자 선별

이익 극대화

대출 승인된 사용자들에게 얻을 수 있는 금액

손실 최소화

잘못 예측했을 경우의 기대손실
(올바르게 예측했다면 얻을 수 있었던 이익)



목적함수

서론

이익 극대화

$$\text{Maximize } \text{순이익} = \sum_{\text{실제} = 0, \text{예측} = 0} \text{대출 금액} \times \text{이자율} - \sum_{\text{실제} = 1, \text{예측} = 0} (\text{대출금} - \text{반환한 원금})$$

손실 최소화

$$\text{Minimize } \text{손실} = \sum_{\text{실제} = 1, \text{예측} = 0} (\text{월별 상환액} \times \text{기간(월)}) - \text{부도 직전 상환액} + \sum_{\text{실제} = 0, \text{예측} = 1} \text{대출 금액} \times \text{이자율}$$

목적함수 설정 근거

서론

은행의 이자이익 구조를 선행연구로 선정하였다.

이익 극대화

$$\text{순이익} = \sum_{\text{실제} = 0, \text{예측} = 0} \text{대출 금액} \times \text{이자율} - \sum_{\text{실제} = 1, \text{예측} = 0} (\text{대출금} - \text{반환한 원금})$$

채권자

Margin : 이자금



채무자

목적함수 설정 근거

서론

기대신용손실법을 선행연구로 아이디어를 구상하였다.

손실 최소화

$$\text{손실} = \sum_{\text{실제} = 1, \text{예측} = 0} (\text{월별 상환액} \times \text{기간(월)}) - \text{부도 직전 상환액} + \sum_{\text{실제} = 0, \text{예측} = 1} \text{대출 금액} \times \text{이자율}$$

기대신용손실법(expected credit loss model)

결산일에 보유중인 매출채권 잔액에 대해서
미래 기간 동안 채무불이행으로 인해 예상되는
손실금액을 계산하여 이 금액을 현재가치로
평가한 금액을 대손충당금으로 설정하는 방법



3조 Treatment

부도 직전일에 보유중인
대출 잔액에 대해서
미래 기간 동안 채무불이행으로
간주하여 손실 예측

모델 설정

서론

김승현, 이동원, 정봉주 and 오경주. (2019). Predicting Debt Default of P2P Loan Borrowers Using Self-Organizing Map. *Quantitative Bio-Science*, 38(1), 63-71.

배재권, 이승연 and 서희진. (2018). 인공지능기법을 이용한 온라인 P2P 대출거래의 채무불이행 예측에 관한 실증연구. *한국전자거래학회지*, 23(3), 207-224.

Git hub의 fares-ds/
Predicting_Loan_Defaulters_Using_Deep_Learning
참고(Kaggle 동일)

Logistic Regression

해당 선행연구에서 로지스틱 모델을 활용하여 Lending club 핀테크 기업의 P2P 대출 부도 예측을 진행한 바가 있다.

Decision Tree

해당 선행연구에서 의사결정나무 모델을 활용하여 Lending club 핀테크 기업의 P2P 대출 불이행 예측을 진행한 바가 있다.

Random Forest

해당 선행연구에서 랜덤포레스트 모델을 활용하여 Lending club 핀테크 기업의 P2P 대출 부도 예측을 진행한 바가 있다.

Cut-off

서론

$$Cost = C_{FN} \times FN + C_{FP} \times FP$$

Provost&Fawcett의 cost 함수: 잘못 예측한 경우의 비용

예측 경우의 수

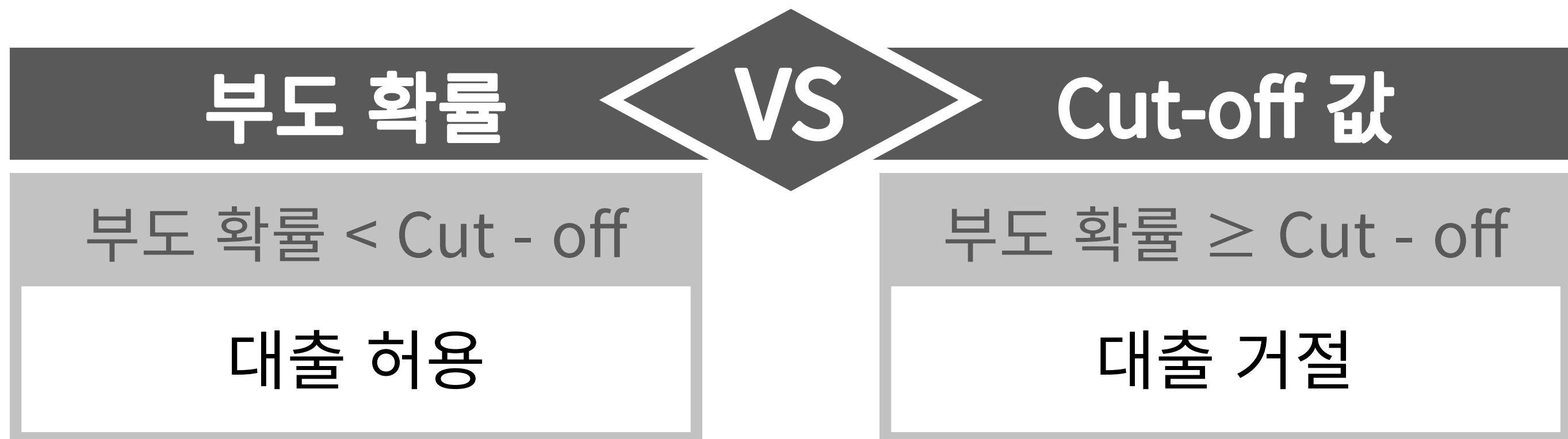
	예측	실제 결과
FN	부도 X	부도 0
FP	부도 0	부도 X



Cost 값 최소로 만드는
Cut - Off 찾기

대출 여부 결정

서론



이익 & 손실 최적화

이익 · 손실 최적화

서론



이익 · 손실 최적화

Cost 함수

이익이 최대가 되고 손실이 최소가 되는
FN & FP 가중치 계수 (C_{FN} , C_{FP})를 찾는다

모델 선정

여러 모델의 결과값을 비교하여
가장 최적화된 값을 도출하는 모델을 선정한다



본 론

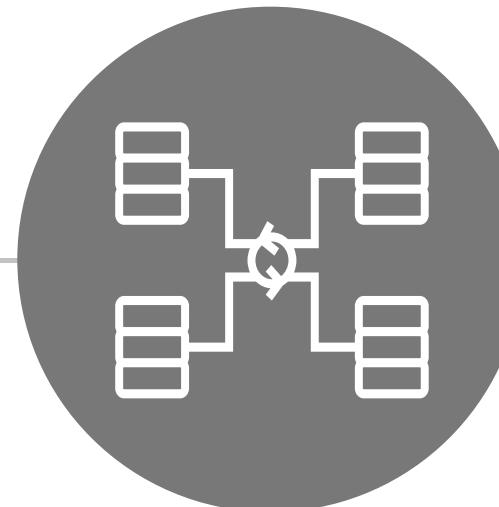
부도 확률 분포 예측 과정

서론



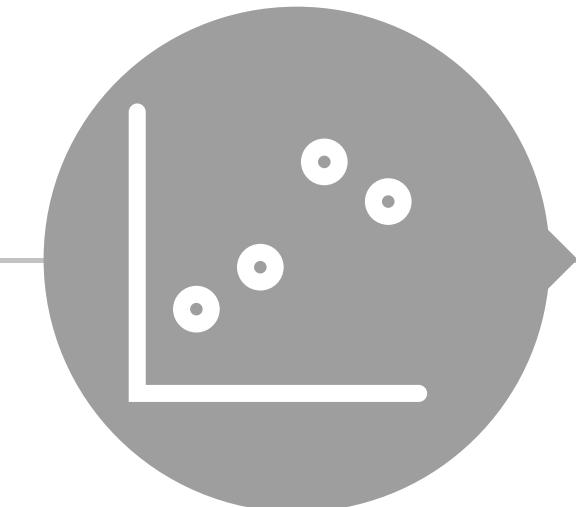
① 초기 변수 설정

선행연구
참조하여
설정



② 데이터 전처리

더미변수 처리
다중공선성 예방
이상치제거
데이터 정규화



③ 최적 변수 추출

Lasso
Elastic Net



④ 부도 확률 분포 예측

Logistic
regression
decision tree
random forest

초기 변수 설정

본론

333개의 column 중에서 최대한 유의미한 변수들을 살려놓고 최적 변수들을 추출하기 위해 3가지의 선행연구를 참고하여 초기변수를 설정하였다.

- loan_amnt
- funded_aunt
- funded_amnt_inv
- installment
- annual_inc
- dti
- delinq_2yrs
- fico_range_low
- fico_range_high

- emp_length
- home_ownership
- verification_status
- purpose
- addr_state

- collections_12_mths_ex_med
- acc_now_delinq
- tot_coll_amt
- tot_cur_bal
- chargeoff_within_12_mths
- delinq_amnt
- pub_rec_bankruptcies
- tax_liens

- inq_last_6mths
- open_acc
- pub_rec
- revol_bal
- revol_util
- total_acc

• • •

"Predicting Debt Default of P2P Loan Borrowers Using Self-Organizing Map" 논문에서 Table 2. Selected variable group 참고
 "Git hub의 fares-ds/Predicting_Loan_Defaulters_Using_Deep_Learning"에서 Data Description part 참고
 “인공지능기법을 이용한 온라인 P2P 대출거래의 채무불이행 예측에 관한 실증연구” 논문의 3.1 자료 수집 및 변수 선정 참고

데이터 전처리

본론

1

더미변수
처리

명목형 더미변수
연속형 더미변수

2

다중공선성
예방

변수 간 상관관계
의미 중복되는 변수

3

이상치
제거

Outlier 데이터
Isolation Forest

4

데이터
정규화

Scaling 일정하게
데이터 처리

더미변수 처리

본론

명목형

dummy1 열 제거

purpose_1	purpose_2	purpose_3
1	0	0
0	1	0
0	0	1
0	0	0

연속형

undummy 처리

emp_length1	emp_length2	emp_length3	emp_length4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



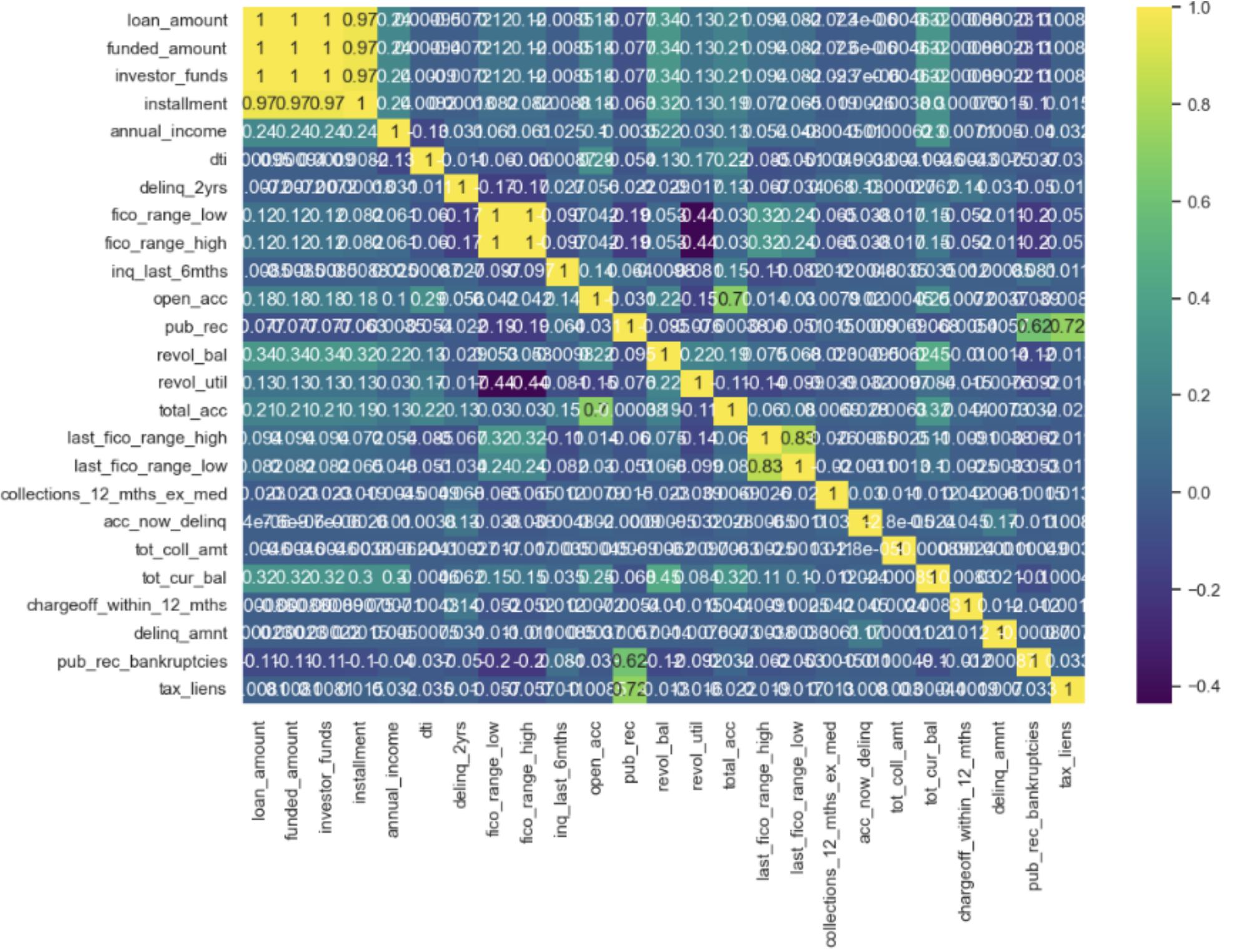
emp_length
1
2
3
4

다중공선성 예방

본로

변수간 상관관계 0.72 이상인 데이터 제거

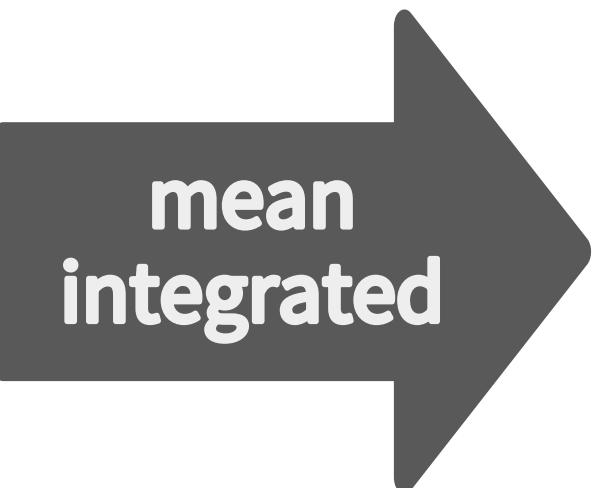
funded_amount
investor_funds
installment
pub_rec



다중공선성 예방

본론

fico_range_low	fico_range_high
690	694
680	684
670	674
660	664



fico_score
692
682
672
662

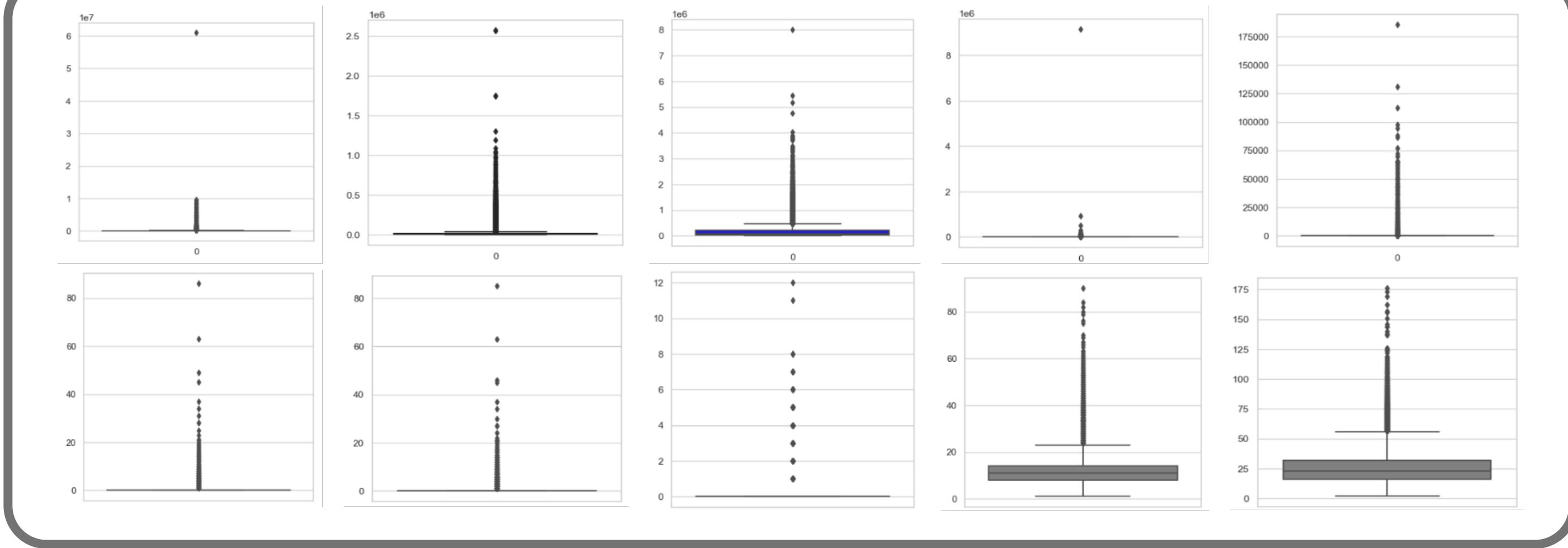
high - low 차이가 4로 일정

평균값으로 통합

이상치 제거

본론

Boxplot 그리기 (outlier 데이터 확인)



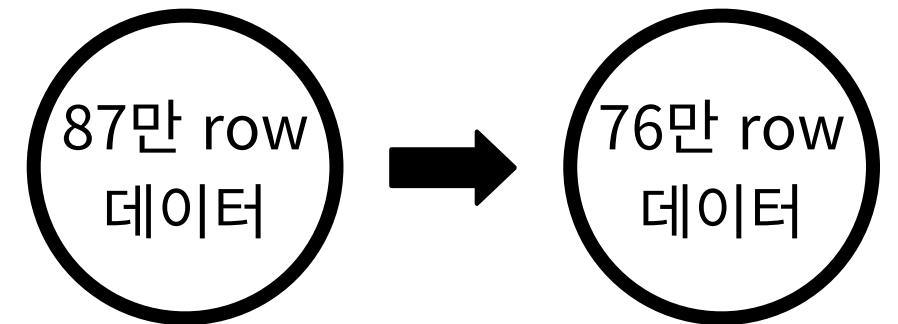
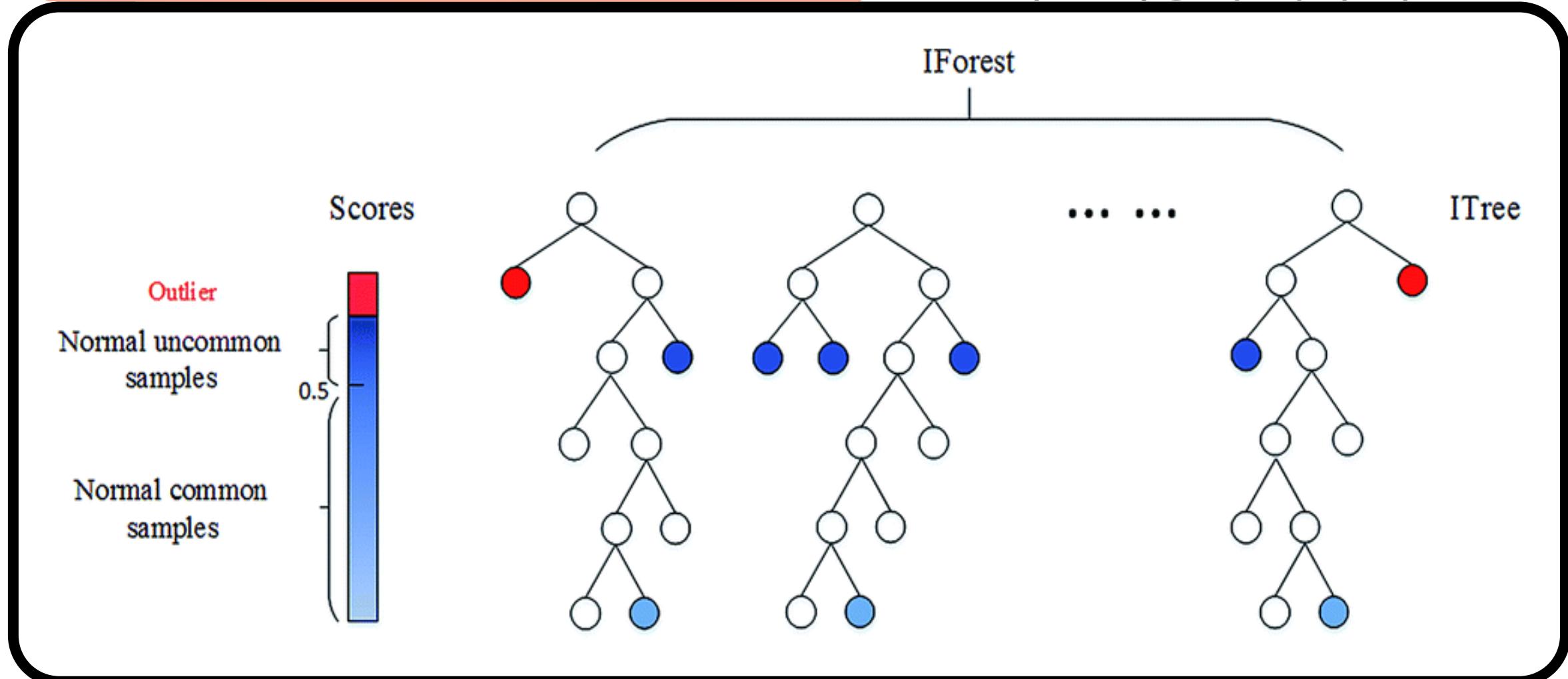
Outlier 데이터 제거

이상치 제거

본론

Isolation Forest

tree 기반 이상치 제거 기법



annual_income
revol_bal
tot_cur_bal
tot_coll_amnt
delinq_amnt
tax_liens
pub_rec_bankruptcies
open_acc
total_acc

데이터 정규화

본론

너무 큰 scale 범위

노이즈 데이터 생성
overfitting

가능성 ↑

scaling

전체 데이터 정규화

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

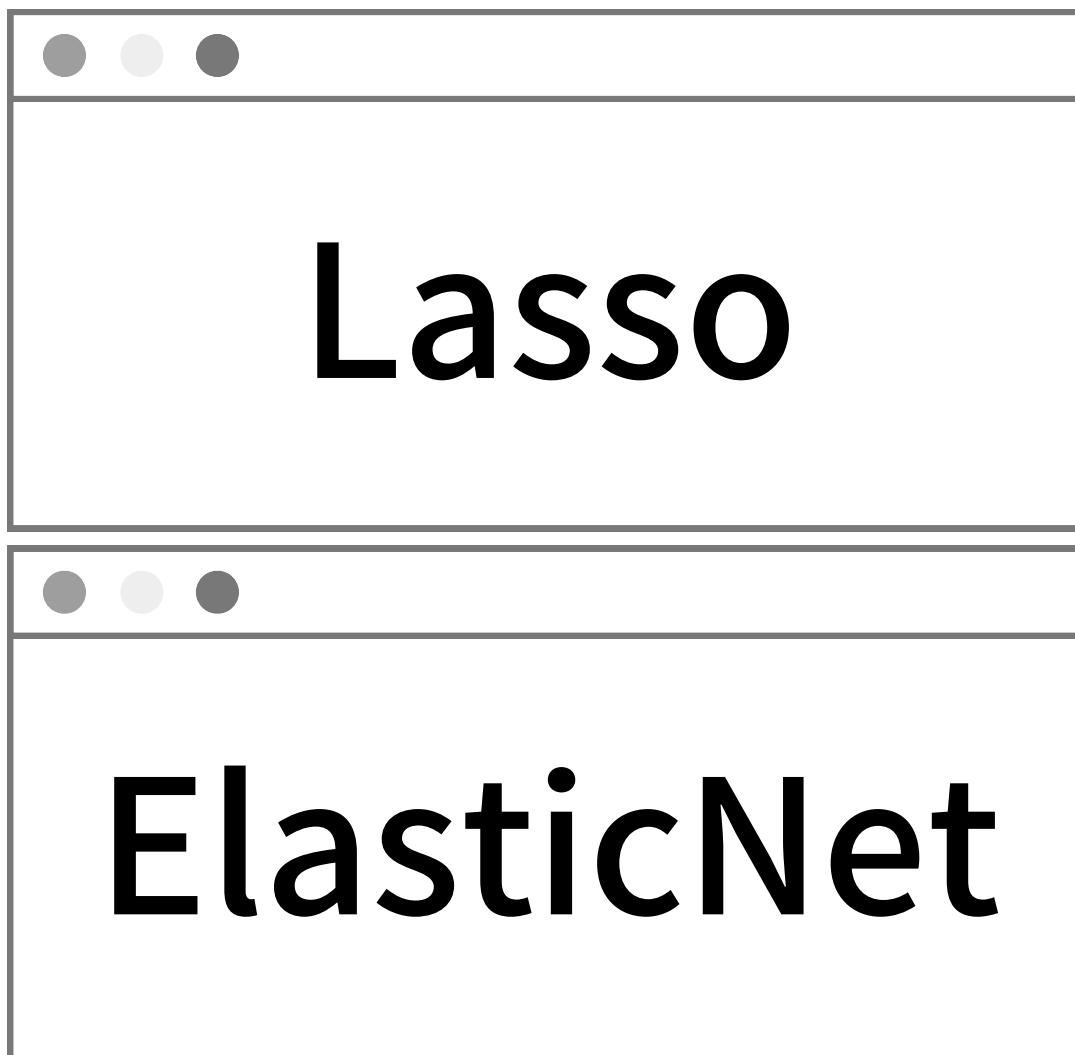
loan_amount	revol_bal	dti
12000	5673	10
24000	11980	17
36000	10285	11
28000	25930	7

scaling

loan_amount	revol_bal	dti
0.28	0.01	0.20
0.33	0.02	0.28
0.46	0.02	0.21
0.38	0.03	0.18

변수선택법 - 회귀 계수 축소

본론



영향력 없는 입력변수 계수
0으로 수렴

입력 변수의 수 감소

1. 모형의 정확도 향상
2. 모형의 연산 속도 향상

Alpha 값 결정

본론

Lasso와 ElasticNet의 파라미터 alpha는 MSE가 가장 작게 나오는 값으로 선정하였다.

ElasticNet (L1_ratio)	MSE
ElasticNet (L1_ratio = 0.8)	0.110534
ElasticNet (L1_ratio = 0.5)	0.110409
ElasticNet (L1_ratio = 0.2)	0.110125
ElasticNet (L1_ratio = 0.1)	0.109997
ElasticNet (L1_ratio = 0.01)	0.110361
ElasticNet (L1_ratio = 0.001)	0.110280

Lasso (alpha)	MSE
Lasso (alpha = 100)	0.113726
Lasso (alpha = 10)	0.113726
Lasso (alpha = 1)	0.113726
Lasso (alpha = 0.1)	0.113726
Lasso (alpha = 0.01)	0.113726
Lasso (alpha = 0.001)	0.110625
Lasso (alpha = 0.0001)	0.109861

변수선택

본론

유의미한 계수를 가진 변수들의 목록은 다음과 같다

Lasso

revol_bal	addr_state7	loan_amount
tot_cur_bal	purpose7	inq_last_6mths
fico_score	addr_state49	dti
pub_rec_bankruptcies	mths_since_last_delinq	• • •
mths_since_recent_bc	addr_state31	

53개

ElasticNet

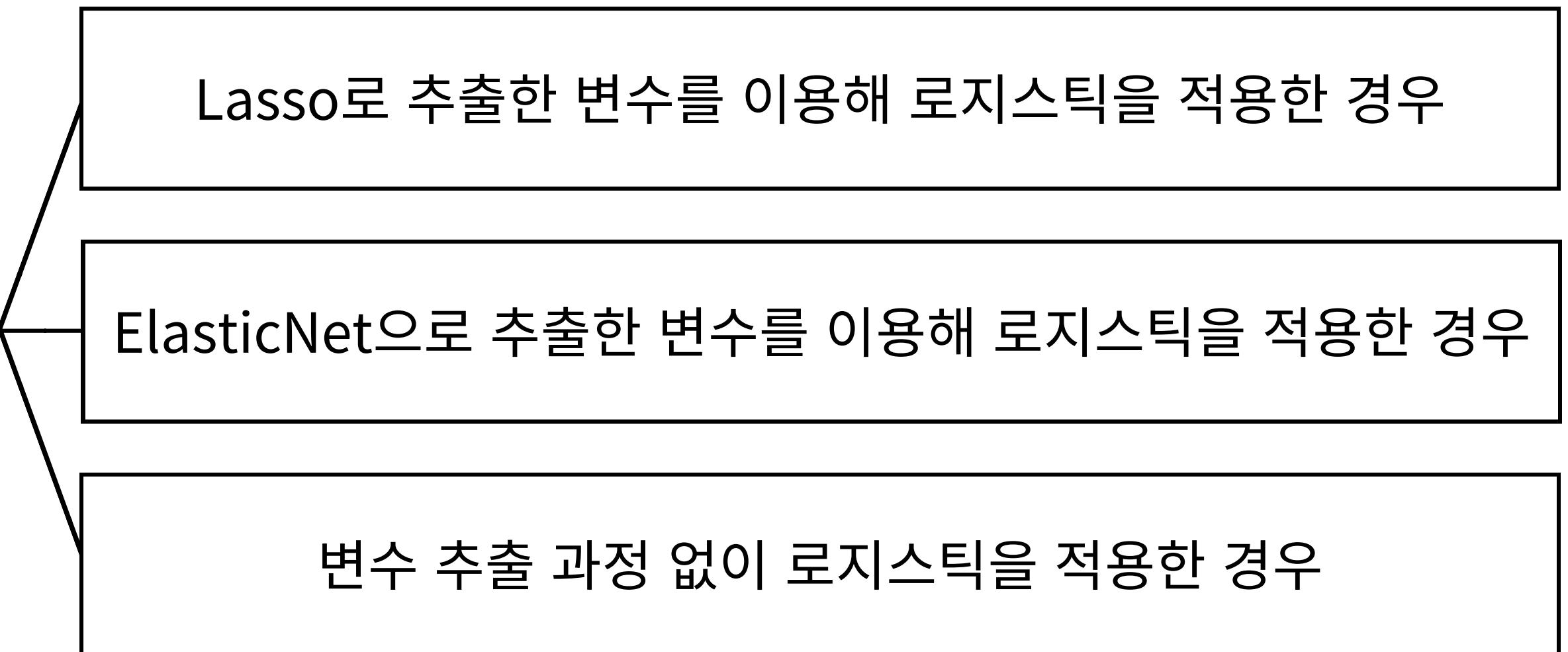
fico_score	purpose2	addr_state41
tot_cur_bal	addr_state38	mths_since_last_record
revol_bal	addr_state7	pub_rec_bankruptcies
total_acc	home_ownership5	• • •
mths_since_recent_bc	inq_last_6mths	
addr_state49	dti	

49개

부도 확률 예측: Logistic Model

본론

Logistic Model

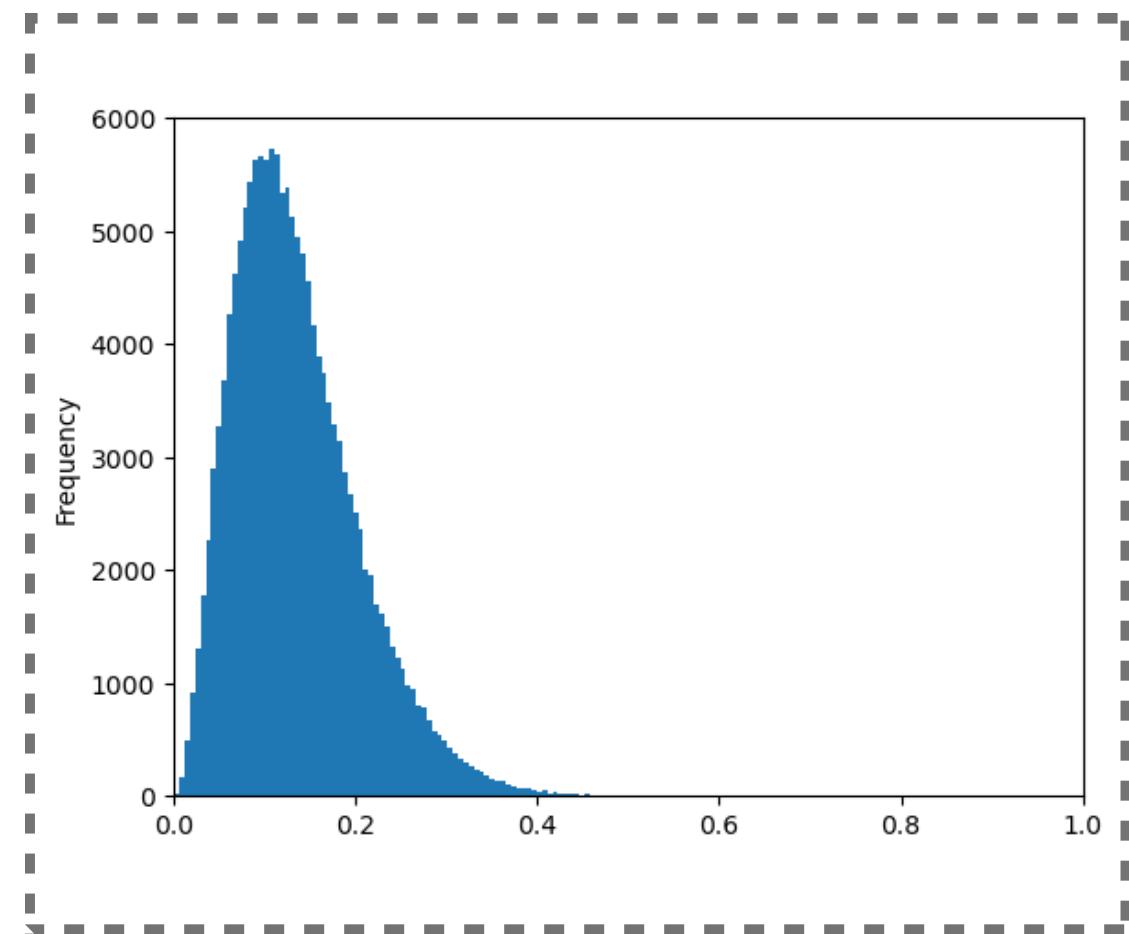


부도 확률 예측 : Logistic Model

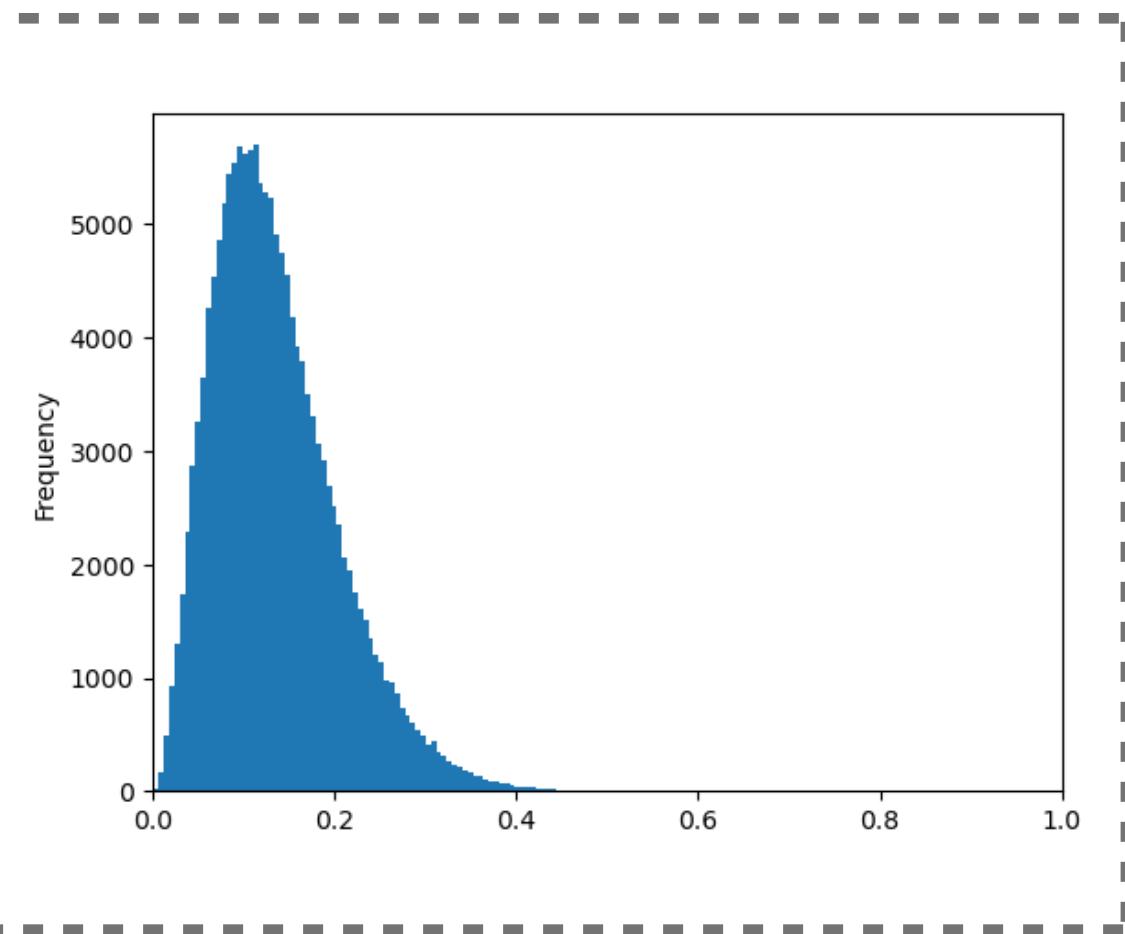
본론

Training Set로 학습된 모델의 Validation Set 부도확률 예측 결과

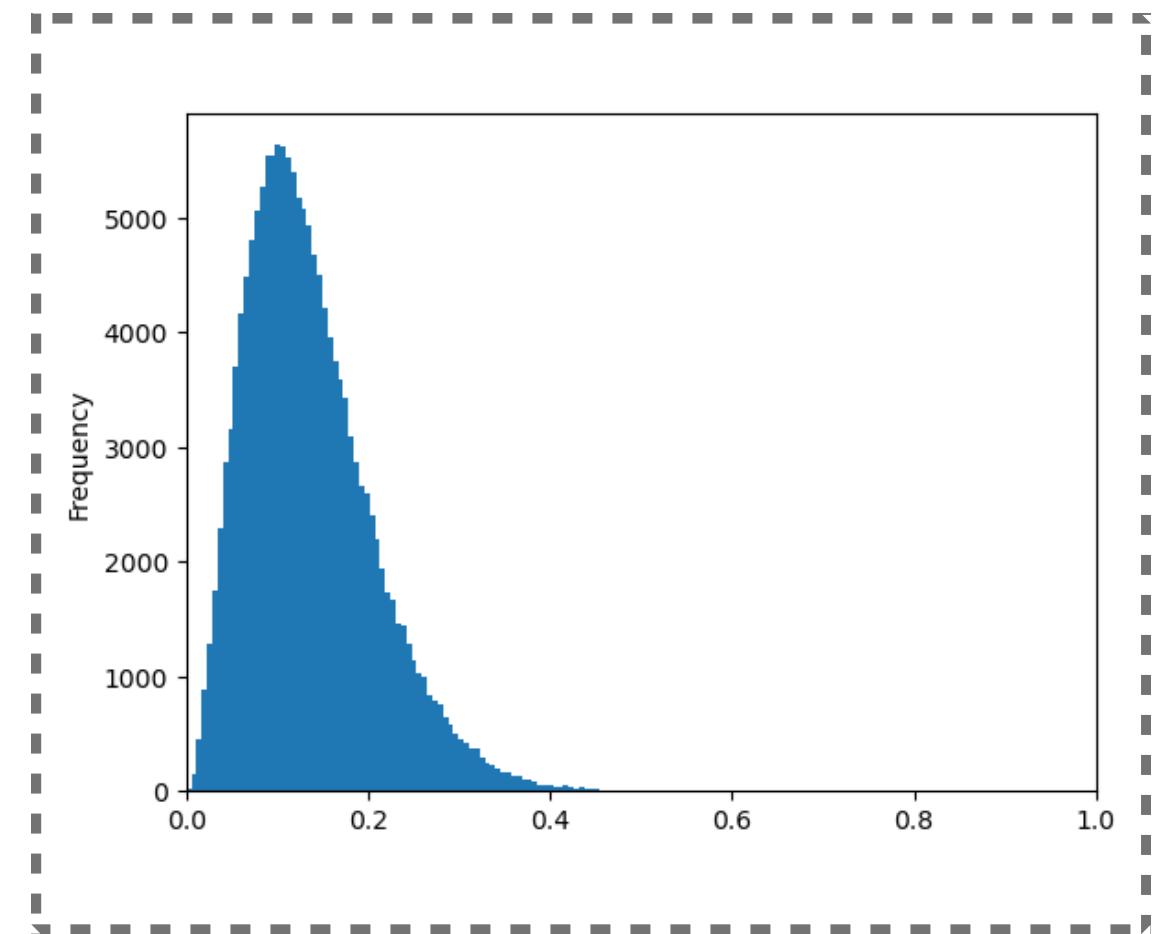
Lasso 변수 추출



ElasticNet 변수 추출



변수 추출 과정 X



부도 확률이 0에서 0.56 사이로 분포하는 것을 볼 수 있다.

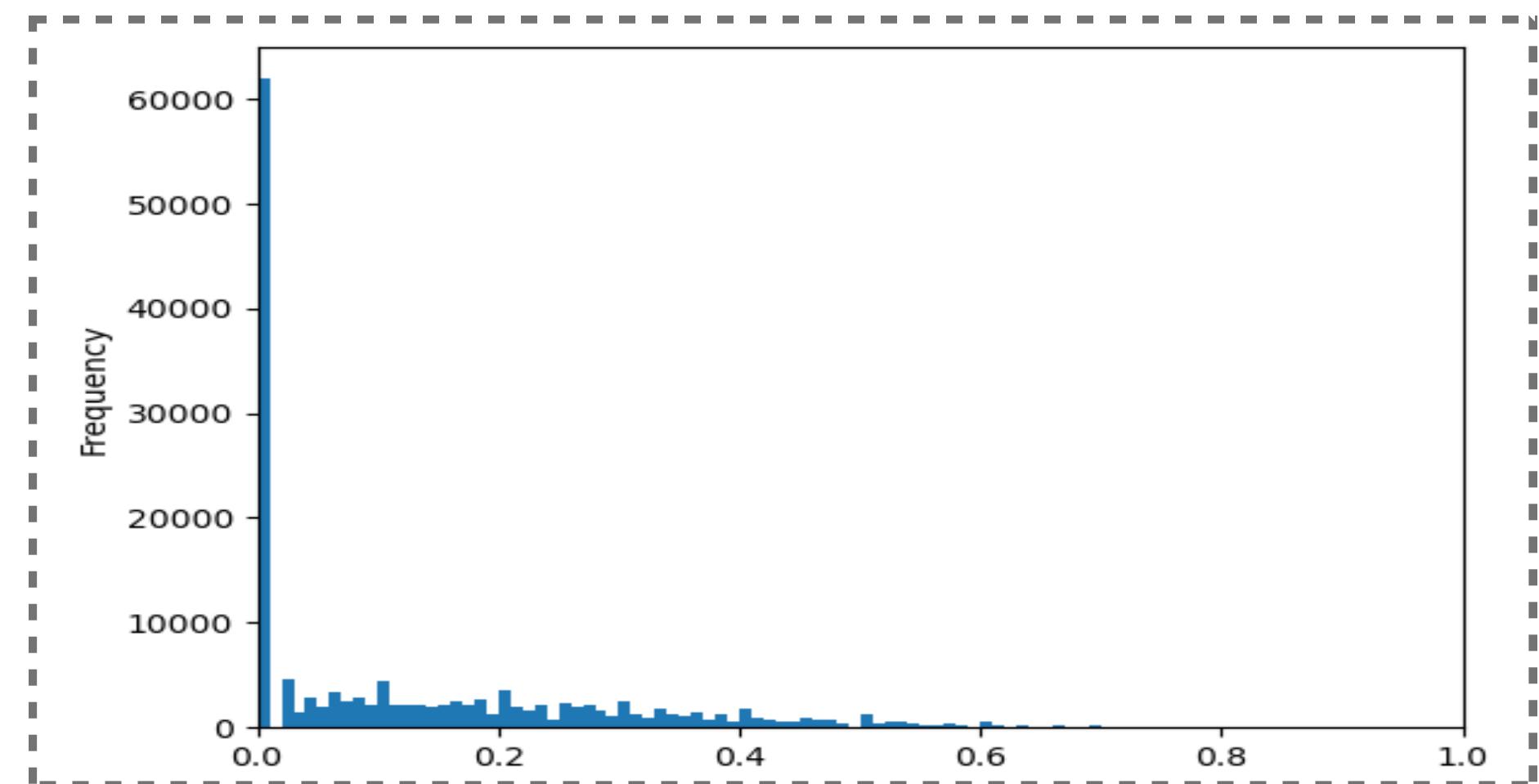
*auroc 곡선은 부록 1에 첨부

부도 확률 예측 : Decision Tree

본론

Training Set로 학습된 모델의 Validation Set 부도확률 예측 결과

GridSearch로 구한 최적 파라미터	
criterion	'gini'
max_depth	None
min_samples_leaf	10
min_samples_split	50



부도확률이 0에 많이 분포하는 것을 확인하였다.
 다른 모델과 달리 비교적 고루 분포하지 않는 것을 볼 수 있었다.

부도 확률 예측 : Random Forest

본론

Training Set로 학습된 모델의 Validation Set 부도확률 예측 결과

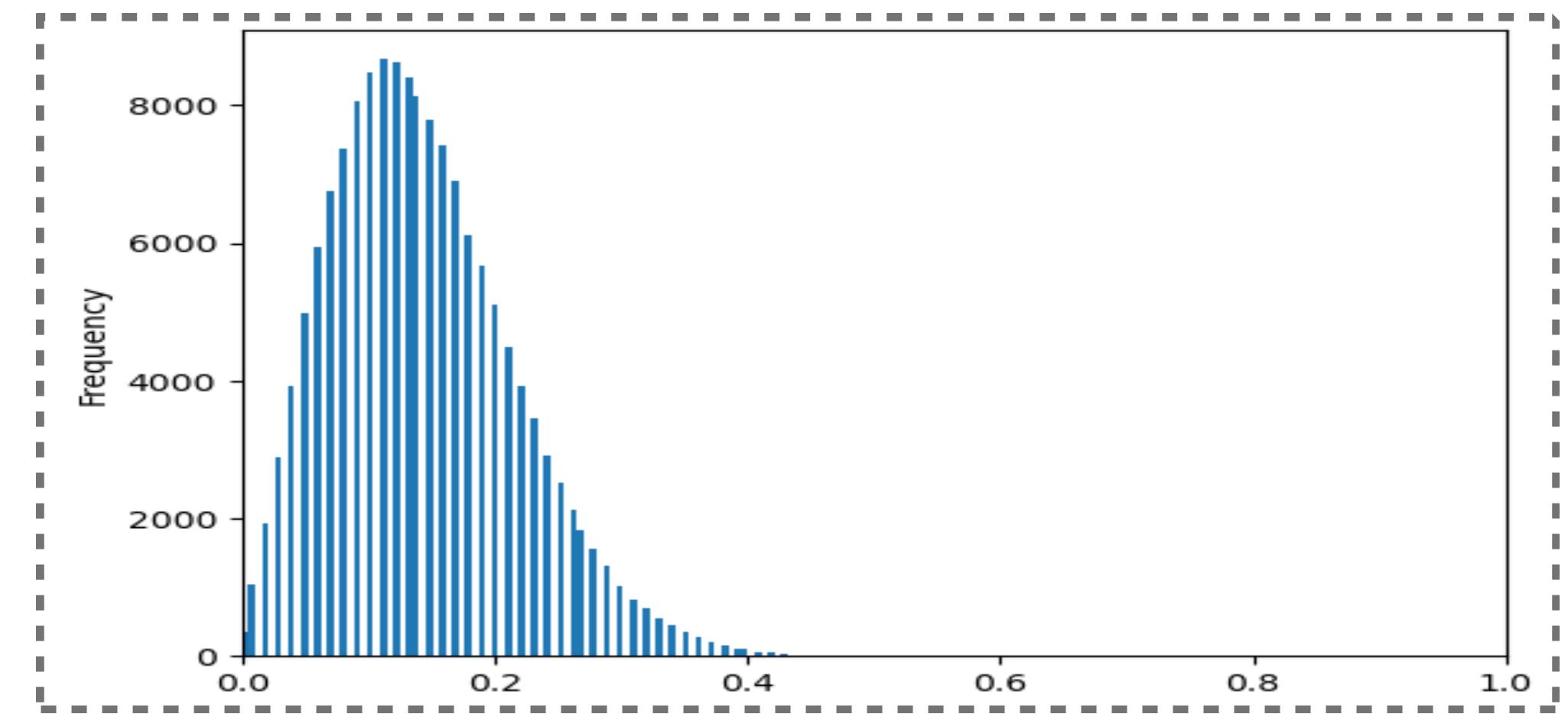
GridSearch로 구한 최적 파라미터

max_depth

None

n_estimators

100



부도 확률이 0에서 0.7 사이로 분포하는 것을 볼 수 있다.

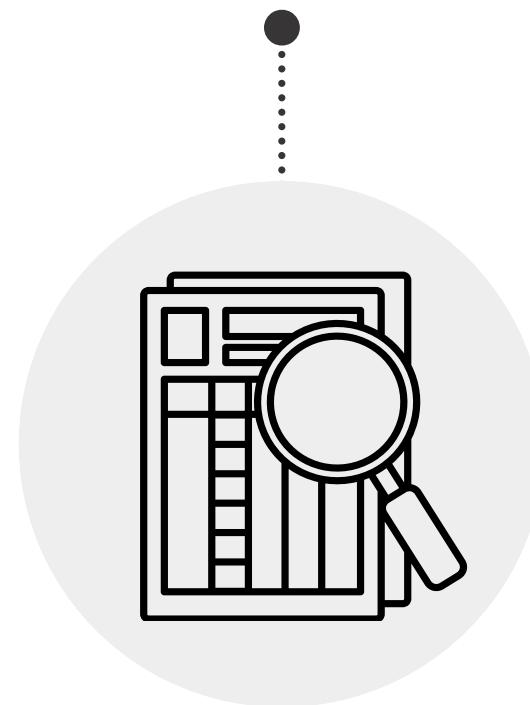
*auroc 곡선은 부록 1 첨부

분석 절차

본론

Optimal threshold 설정

다양한 가중치를 준
Cost 함수에 따라 설정

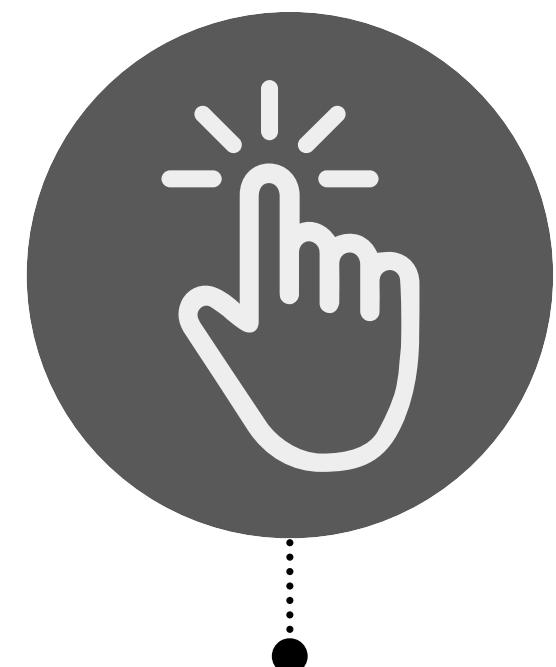


이익과 손실 비교

Optimal threshold 따라 비교

가중치 채택

이익과 손실 최적화되는
경우 채택



최적 모델 선정

Cost 함수에 대한 Optimal Threshold 비교

본론

서론 cost 함수 구현

```
def cost(c1, c2, n, cut_off):
    result = pd.DataFrame(model.predict_proba(X_val)[:,1], columns=['pred_prob'])

    #cutoff이상이면 부도 예측을 1, cutoff 미만이면 부도 예측은 0으로 둔다.
    for i in range(len(result)):
        if result.loc[i,'pred_prob']>=cut_off:
            result.loc[i,'pred_prob']=1
        else:
            result.loc[i,'pred_prob']=0
    y_pred_prob=result['pred_prob']

    #confusion matrix 명령어를 이용해 fn, fp를 구한다.
    tn, fp, fn, tp = confusion_matrix(y_val, y_pred_prob).ravel()
    return c1*fn + c2*fp
```

Cost 함수에 대한 Optimal Threshold 비교

본론

Optimal Threshold: Cost 값이 최소화 되는 cutoff

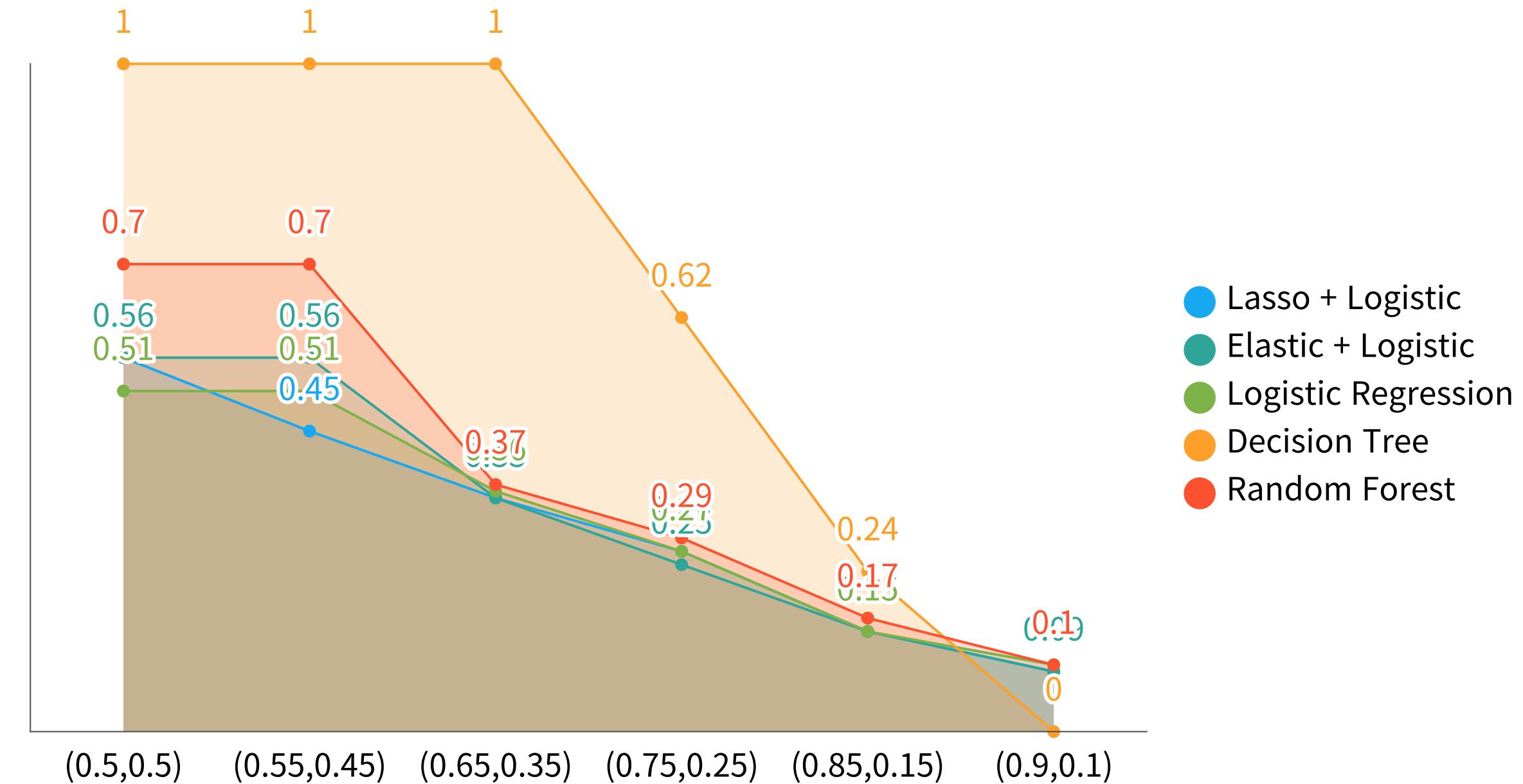
다양한 가중치에서 Optimal Threshold 비교

C_{FN}	0.5	0.55	0.65	0.75	0.85	0.9
C_{FP}	0.5	0.45	0.35	0.25	0.15	0.1

일반적으로 부도가 안날 것이라 예측했는데 난 경우(FN)의 비용이
 부도가 날 것이라 예측했는데 나지 않은 경우(FP)의 비용보다 더 크기 때문에
 C_{FN} 에 C_{FP} 보다 큰 값을 부여했다

Cost 함수에 대한 Optimal Threshold 비교

본론



FN에 대한 가중치가 커질수록 optimal threshold 값이 점점 감소하는 것을 볼 수 있다.
즉, 부도 확률이 낮은 값도 1로 예측하는 경향을 보인다.

Cost 함수에 대한 이익과 손실 비교

본론

이익 코드

```
def income(n,threshold):
    #threshold 지정했을 때 y_pred값을 저장한 dataframe에 필요한 사후변수를 추가하는 과정
    dep_pred = predict(threshold) #threshold 지정했을 때 y_pred값을 저장한 dataframe
    test = pd.concat([X_test1,y_test],axis=1)
    test = test.reset_index()
    test = pd.concat([test,dep_pred],axis=1)

    profit = 0; loss = 0
    for idx in range(n):
        if test['depvar'][idx] == 0 and test['depvar_pred'][idx] == 0:
            funded_amnt_ = test['funded_amount'][idx]
            int_rate_ = test['interest_rate'][idx]
            profit += funded_amnt_*int_rate_ #이자금 (=총금액*이자율)
        elif test['depvar'][idx] == 1 and test['depvar_pred'][idx] == 0:
            funded_amnt_ = test['funded_amount'][idx]
            total_rec_ = test['total_rec_prncp'][idx]
            loss += funded_amnt_ - total_rec_ #대출금 - 월별상환액
    return profit-loss
```

손실 코드

```
def loss(n,threshold):
    #threshold 지정했을 때 y_pred값을 저장한 dataframe에 필요한 사후변수를 추가하는 과정
    dep_pred = predict(threshold) #threshold 지정했을 때 y_pred값을 저장한 dataframe
    test = pd.concat([X_test1,y_test],axis=1)
    test = test.reset_index()
    test = pd.concat([test,dep_pred],axis=1)

    w1=0; w2=0
    for idx in range(n):
        #부도를 내지 않을 것으로 예측했지만 부도를 낸 경우 손실 계산
        if test['depvar'][idx] == 1 and test['depvar_pred'][idx] == 0:
            installment_ = test['installment'][idx]
            if test['term1'][idx] == 1:
                term = 36
            else:
                term = 60
            total_pymnt_ = test['total_pymnt'][idx]
            num = installment_*term - total_pymnt_ #월별상환액*기간 - 부도직전상환액
            w1 += num

        #부도를 낼 것으로 예측했지만 부도를 내지 않은 경우 손실 계산
        elif test['depvar'][idx] == 0 and test['depvar_pred'][idx] == 1:
            funded_amnt_ = test['funded_amount'][idx]
            int_rate_ = test['interest_rate'][idx]
            w2 += funded_amnt_*int_rate_ #이자금 (=총금액*이자율)

    loss = w1 + w2
    return loss
```

Optimal threshold 값 이상인 사용자에게는 대출 X

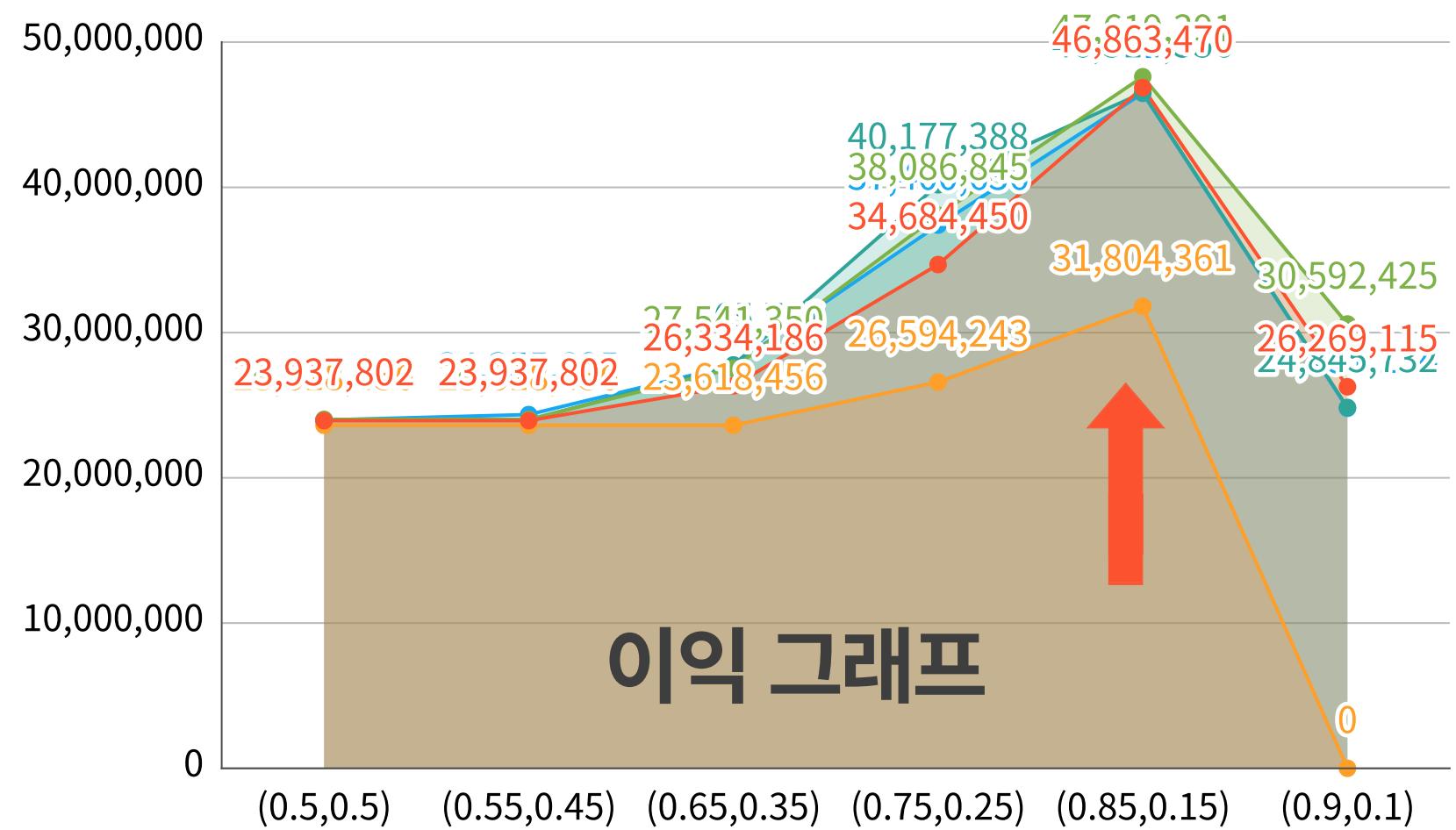
Optimal threshold 값 미만인 사용자에게는 대출 O

이 방식을 test set에 적용하고, 이익과 손실을 비교한다.

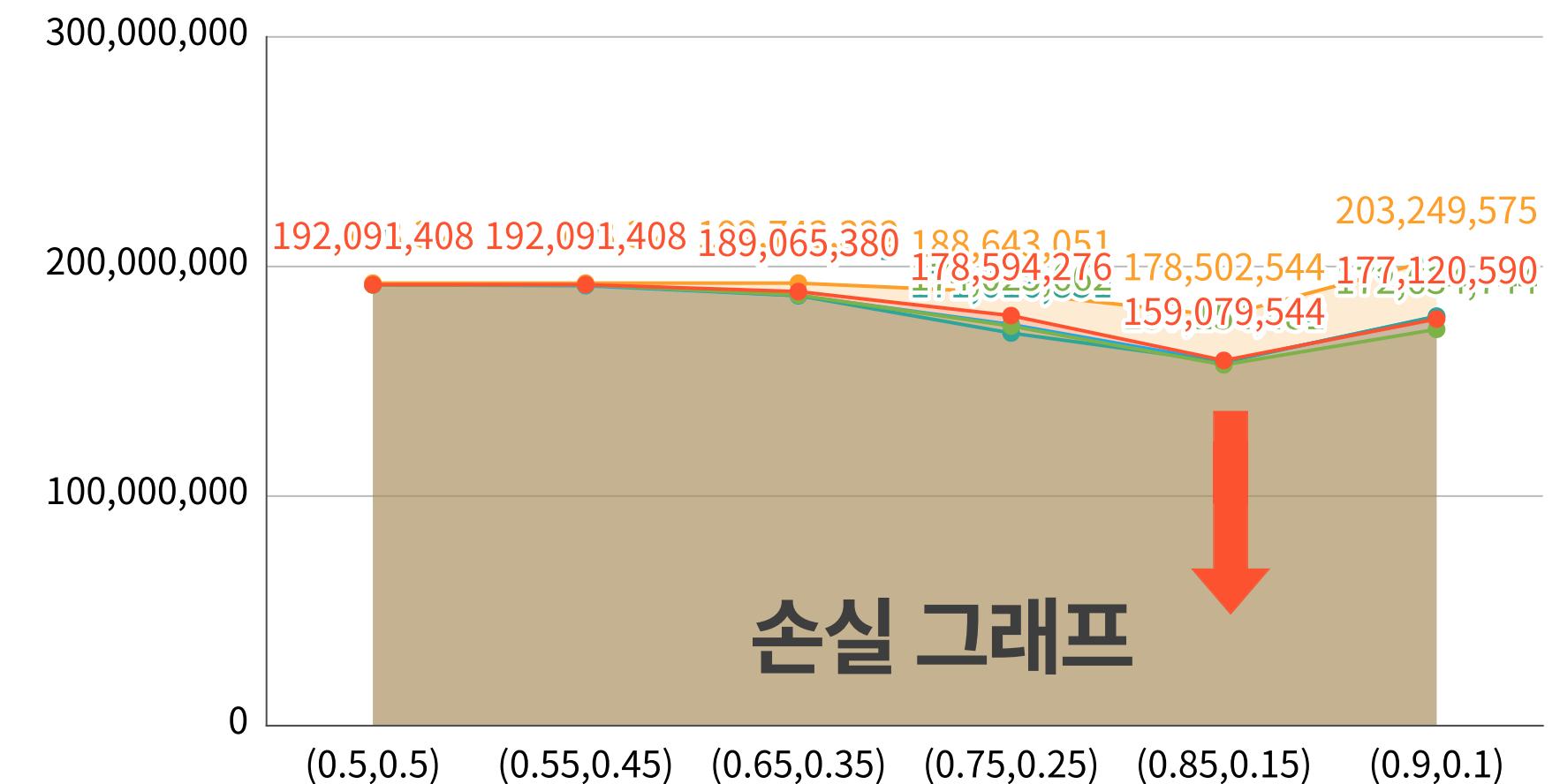
Cost 함수에 대한 이익과 손실 비교

본론

5가지 모델에 대해 가중치에 따른 이익과 손실을 비교한 결과는 다음과 같다.



- Lasso + Logistic
- Elastic + Logistic
- Logistic Regression
- Decision Tree
- Random Forest



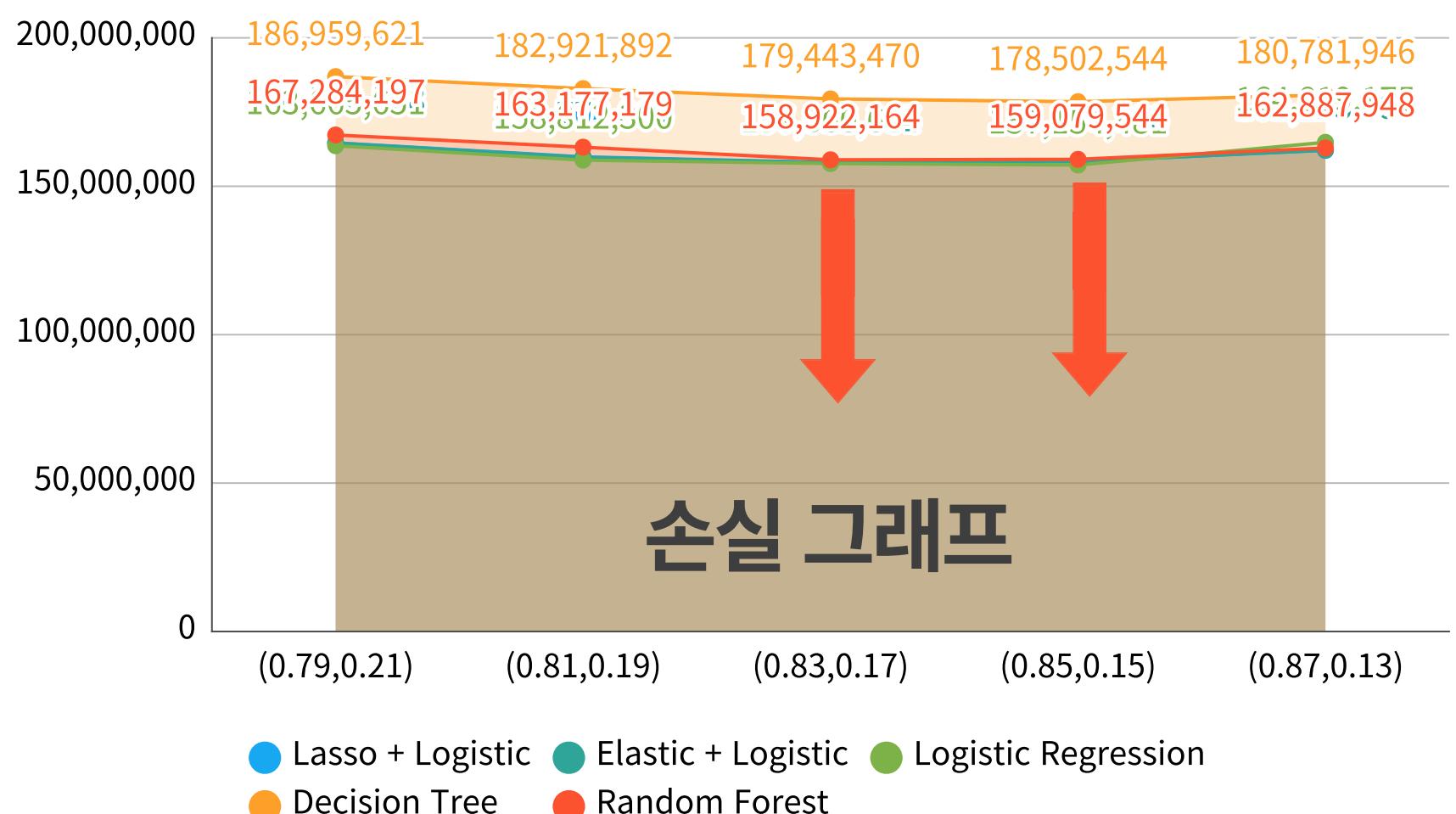
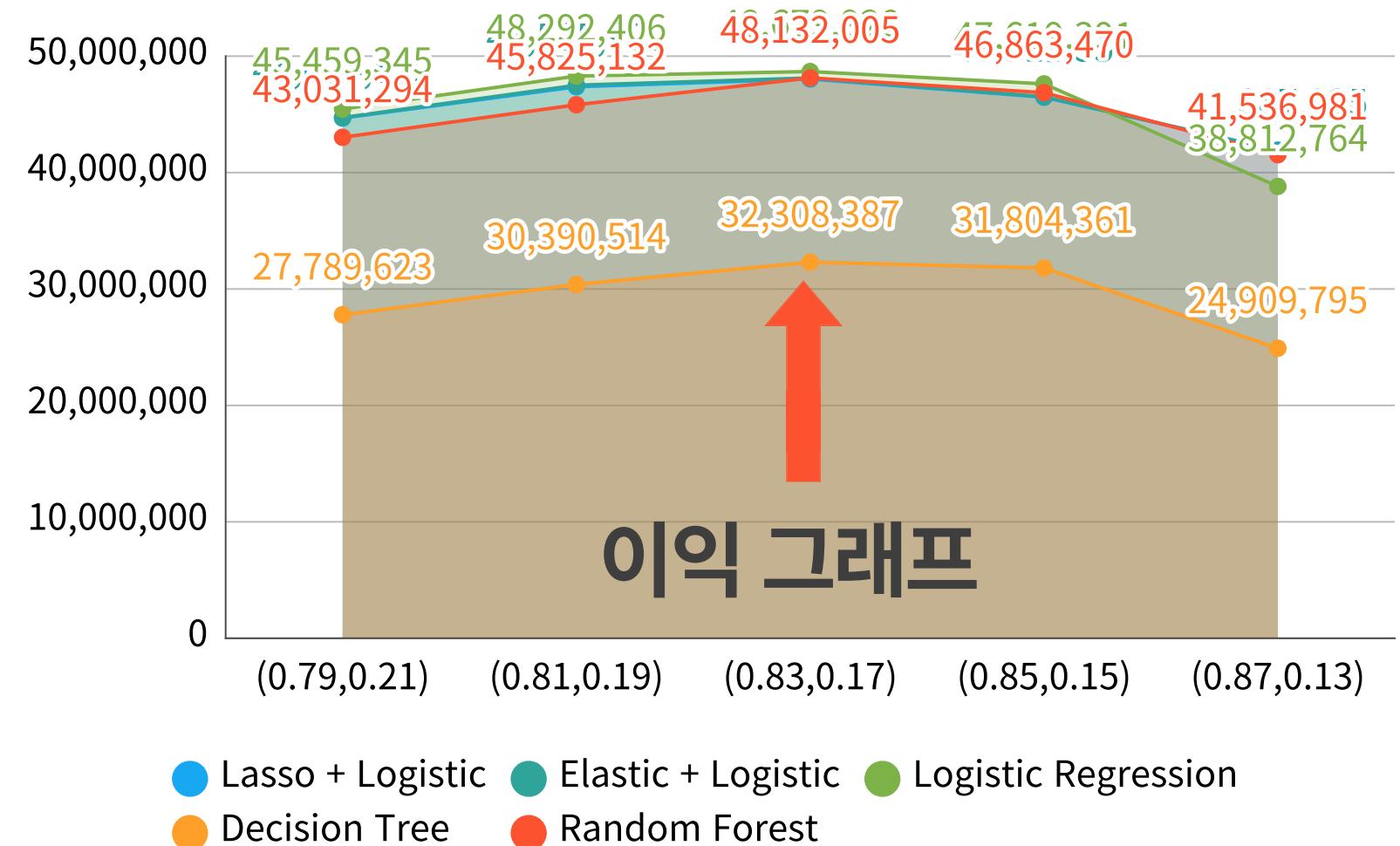
- Lasso + Logistic
- Elastic + Logistic
- Logistic Regression
- Decision Tree
- Random Forest

(C_FN, C_FP)가 (0.85, 0.15)일 때, 이익이 최대가 되고 손실이 최소가 되는 것을 알 수 있다.

Cost 함수에 대한 이익과 손실 비교

본론

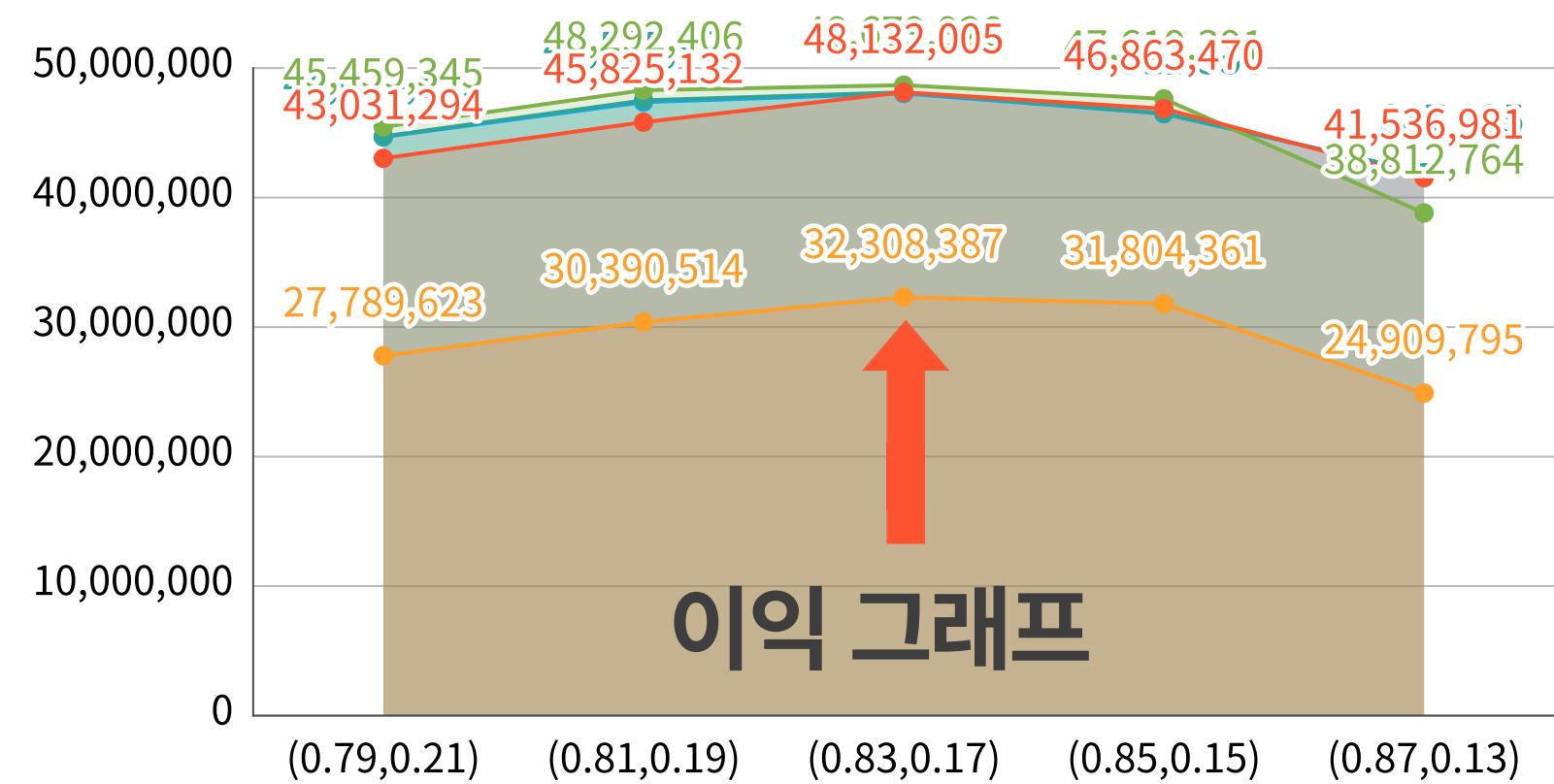
확실한 가중치 계수 (C_{FN} , C_{FP})를 얻기 위해 $(0.85, 0.15)$ 를 근방으로 다시 분석해보았다.



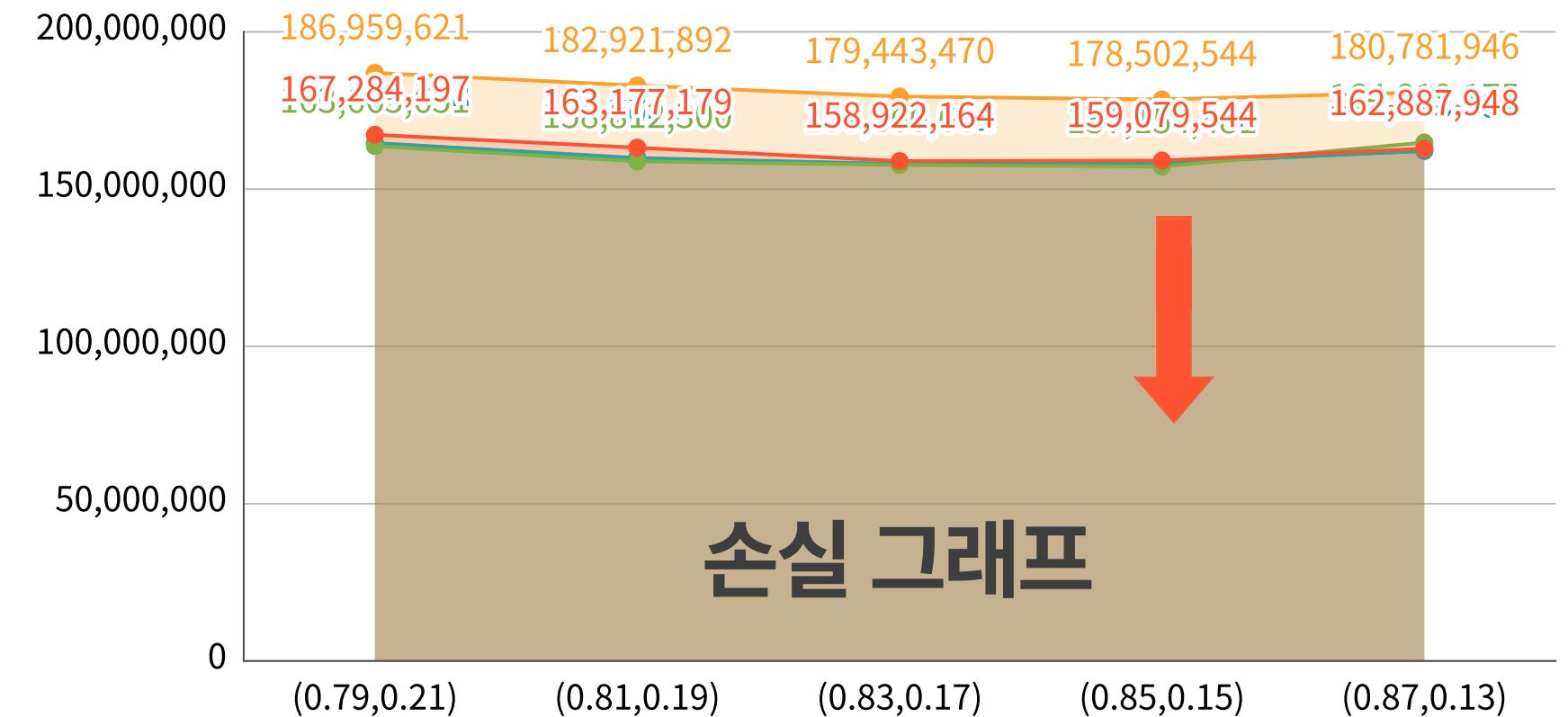
(C_{FN}, C_{FP}) 가 $(0.83, 0.17)$ 일 때, 이익이 최대가 되는 것을 알 수 있었다.
 손실이 최소가 되는 (C_{FN}, C_{FP}) 는 모델에 따라 $(0.85, 0.15)$ 혹은 $(0.83, 0.17)$ 결과가 나왔다.

Cost 함수에 대한 이익과 손실 비교

본론



- Lasso + Logistic
- Elastic + Logistic
- Logistic Regression
- Decision Tree
- Random Forest



- Lasso + Logistic
- Elastic + Logistic
- Logistic Regression
- Decision Tree
- Random Forest

이익을 최대화 하는 모델과 손실의 최소화 시켜주는 모델은 모두 변수 추출 없이 적용한 로지스틱이었다.
 이익을 최대로 될때는 가중치 (0.83, 0.17)로 구한 optimal threshold를 적용하였고,
 손실이 최소가 될 때는 (0.85, 0.15)로 구한 optimal threshold를 적용한 모델이었다.

결론 : Cost 함수

본론

$$\underline{Cost = 0.83 \times FN + 0.17 \times FP}$$

모델 종류와 관계 없이 위와 같은 Cost 함수에서 이익과 손실이 최적화 되는 것을 알 수 있다.

손실을 최소화 하기 위해서는 가중치 (0.85,0.15)를 사용해야 하지만 일반적으로 이익이 더 중시되기 때문에 다음의 cost 함수를 채택하였다.

결론 1

부도를 낸다고 예측하고 내지 않은 경우(FP)보다 부도를 내지 않는다고 낸 경우(FN)가 약 4.88배 정도의 비용을 더 치른다는 것을 알 수 있다

결론 2

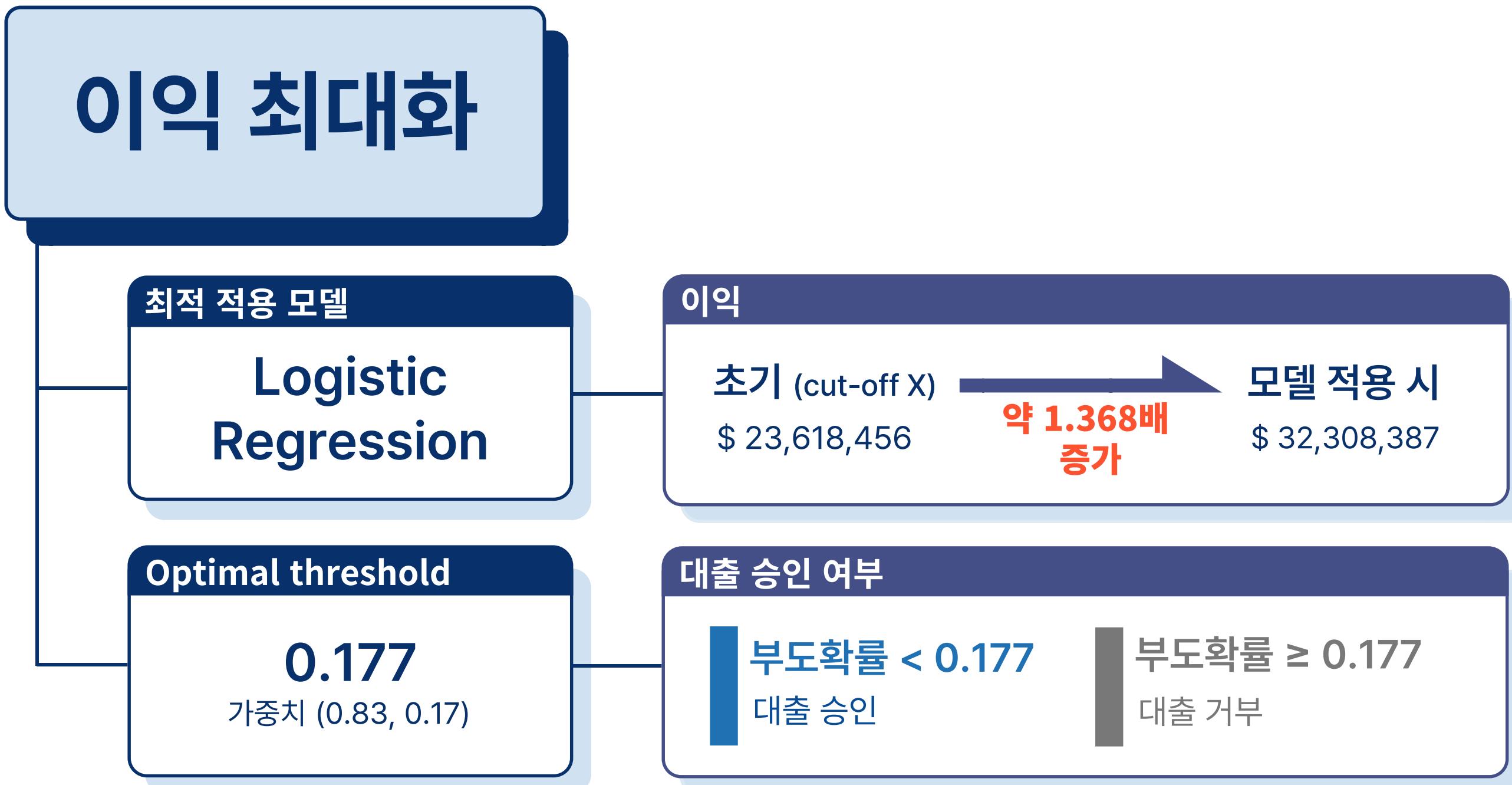
모델 종류와 관계 없이 최적 cost 함수가 일정하기 때문에 새로운 모델에서 optimal threshold를 구하기 위해 cost 함수를 새로 구할 필요가 없다

결론 3

cost 함수의 계수가 모두 상수항이기 때문에 다른 사후 변수 없이 편리하게 사용할 수 있다

결론 : 최적 모델

본론



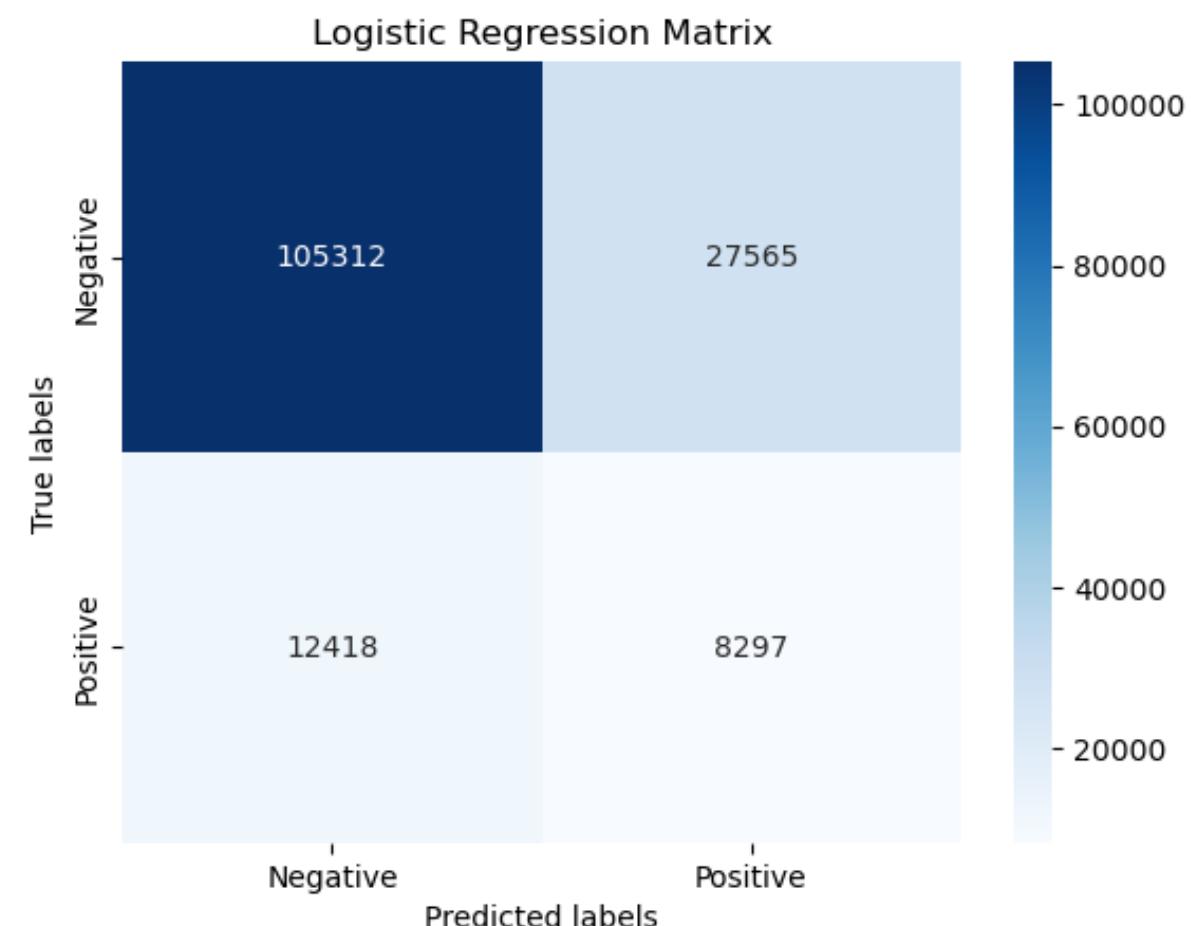
결론 : 최적 모델

본론

이익 최대화 모델

로지스틱으로 구한 확률 분포에 cost 함수의 optimal threshold 값을 기준으로 대출 승인여부를 결정하도록 제작되었다.

Confusion Matrix

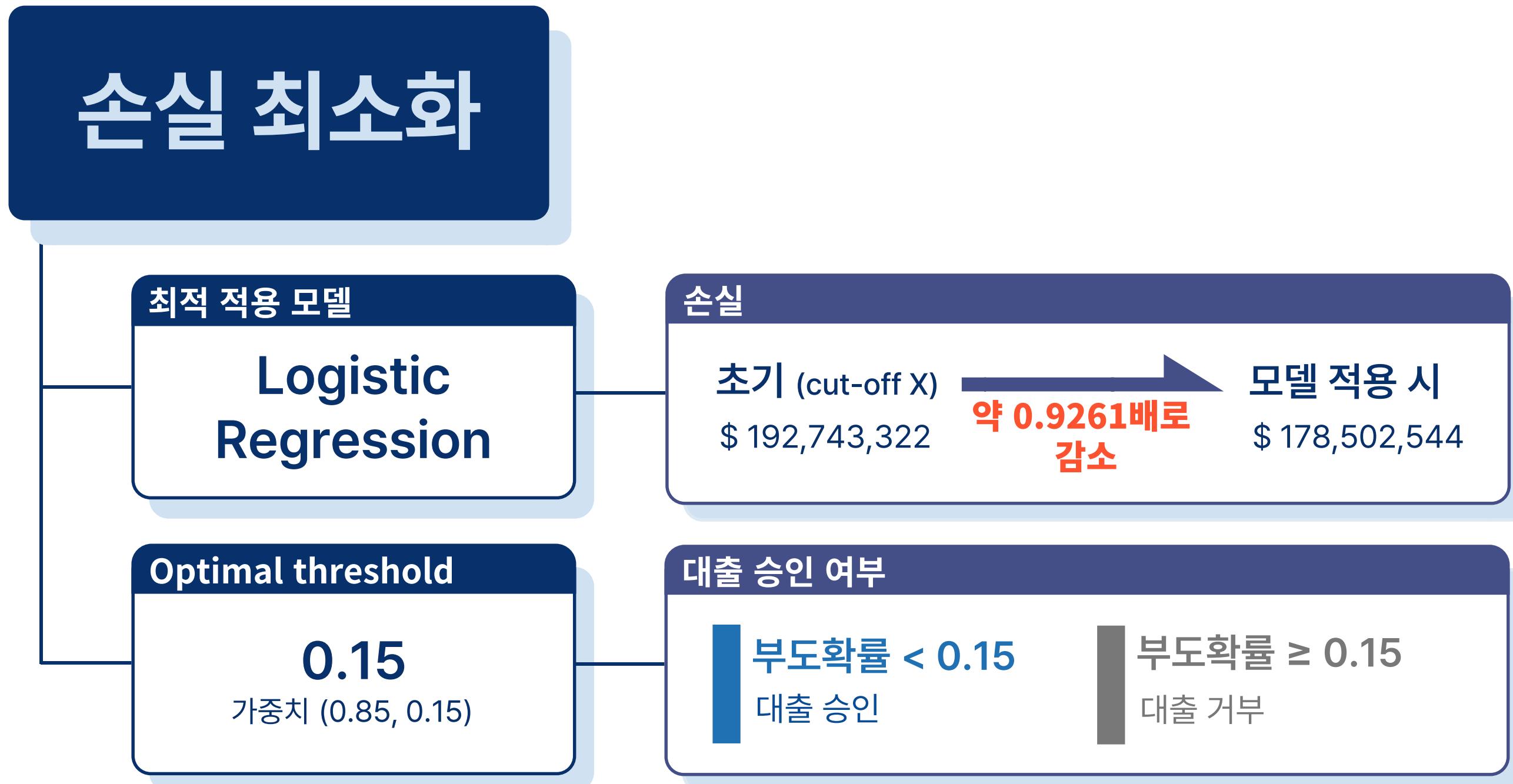


평가 지표

Precision	0.2294
Recall	0.4163
F1 score	0.2958
Accuracy	0.7327

결론 : 최적 모델

본론



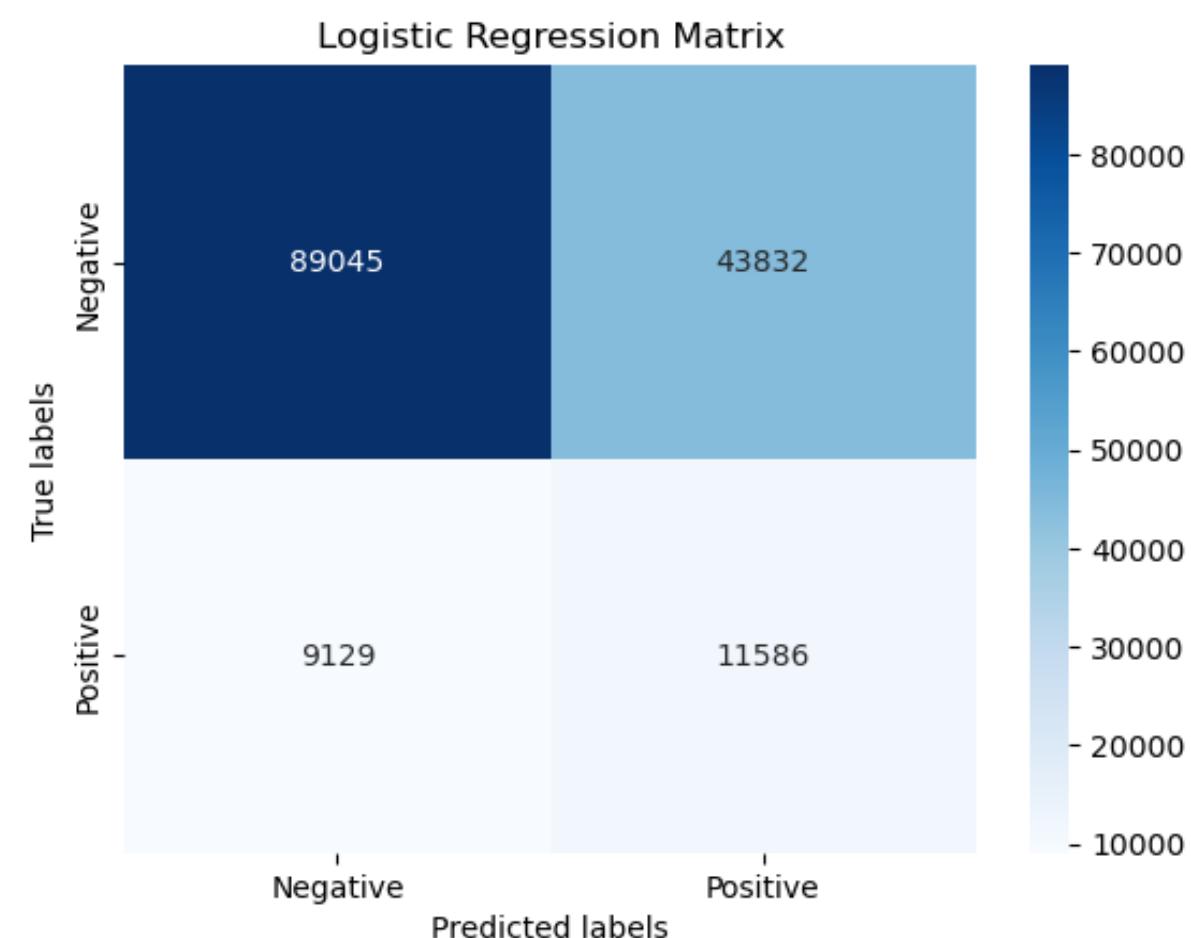
결론 : 최적 모델

본론

손실 최소화 모델

변수 추출 없이 로지스틱을 적용한 손실 최소화 모델에서의 confusion matrix와 평가 지표는 다음과 같이 나타났다.

Confusion Matrix



평가 지표

Precision	0.2163
Recall	0.5041
F1 score	0.3027
Accuracy	0.6868

*나머지 세 모델의 confusion matrix와 지표 표는 부록 2, 3에 첨부

모델을 새로운 데이터에 적용하기

본론

Out-of-sample Test 새로운 데이터 test set을 10개로 분할했을 때의 이익/손실

단위 : US dollar

Test Set	Loss	Original Loss	Income	Original Income
1	25,964,408	64,126,338	4,925,655	-129,381
2	26,340,820	63,656,234	4,766,706	702,045
3	26,357,043	63,522,983	4,303,380	-170,870
4	26,266,697	64,885,018	4,902,971	-240,517
5	26,315,804	63,053,191	4,334,434	39,838
6	26,527,880	65,082,143	4,013,135	-1,574,915
7	26,194,749	64,239,053	4,601,236	-76,325
8	27,213,329	65,501,038	3,507,606	-1,554,490
9	25,597,328	63,326,237	5,007,203	328,660
10	26,238,771	64,134,570	4,466,948	-405,129
Average	26,301,683	64,152,681	4,482,927	-308,108
Total	259,704,808	641,545,519	5,323,906	-3,089,063

모델을 새로운 데이터에 적용하기

본론

이익 최대화 모델

\$ - 3,089,063

이익 증가

\$ 5,323,906

기존 모델은 19.156%이라는 높은 부도 비율로, 수입이 적자.

손실 최소화 모델

\$ 641,545,519

손실 감소

\$ 259,704,808

추가 분석 : Important features

본론

각 모델의 중요한 변수는 다음과 같고, 최적의 모델인 Logistic Regression의 가장 영향력 있는 변수로 모든 계정의 총 현재 잔액(tot_cur_bal)이 선정되었고, 다음으로 총 신용 회전 잔액(revol_bal), 대출자의 신용 기록(pub_rec), Fico 점수(Fico_score), 세금 유치권 수(tax_liens) 순으로 중요도가 분류되었다.

5개의 중요 변수 중 pub_rec 변수는 대출자의 신용 기록을 나타내는데 신용기록이 많으면 부도가 날 확률이 높아질 것이다.
 그러나, 상관계수의 부호가 +가 아닌 - 부호로 결과가 산출되었다.

Logistic Regression	Lasso, Elastic net	Decision Tree	Randomforest
tot_cur_bal(-)	tot_cur_bal	dti	dti
revol_bal(-)	Fico_score	revol_bal	tot_cur_bal
pub_rec(-)	pub_rec_bankruptcies	tot_cur_bal	revol_bal
Fico_score(-)	dti	loan_amount	revol_util
tax_liens(-)	loan_amount	revol_util	loan_amount

한계점

서론

- 1) 단순 Lending Club 데이터 상에서의 지표만을 가지고 회사의 이익 및 손실 예측
- 2) 대출 반환 금액에 원금 균등 상환과 원리금 균등 상환의 구분이 없음
- 3) 2차 가공 데이터로 대출 시작점과 만기는 반영하지 못했음
- 4) 데이터 불균형이 있어서 모델의 정확도를 반영하지 못했음
- 5) 검증용 데이터의 상호교차검증의 단계는 거치지 못해 예측 모형의 오분류율을 검증하지 못함

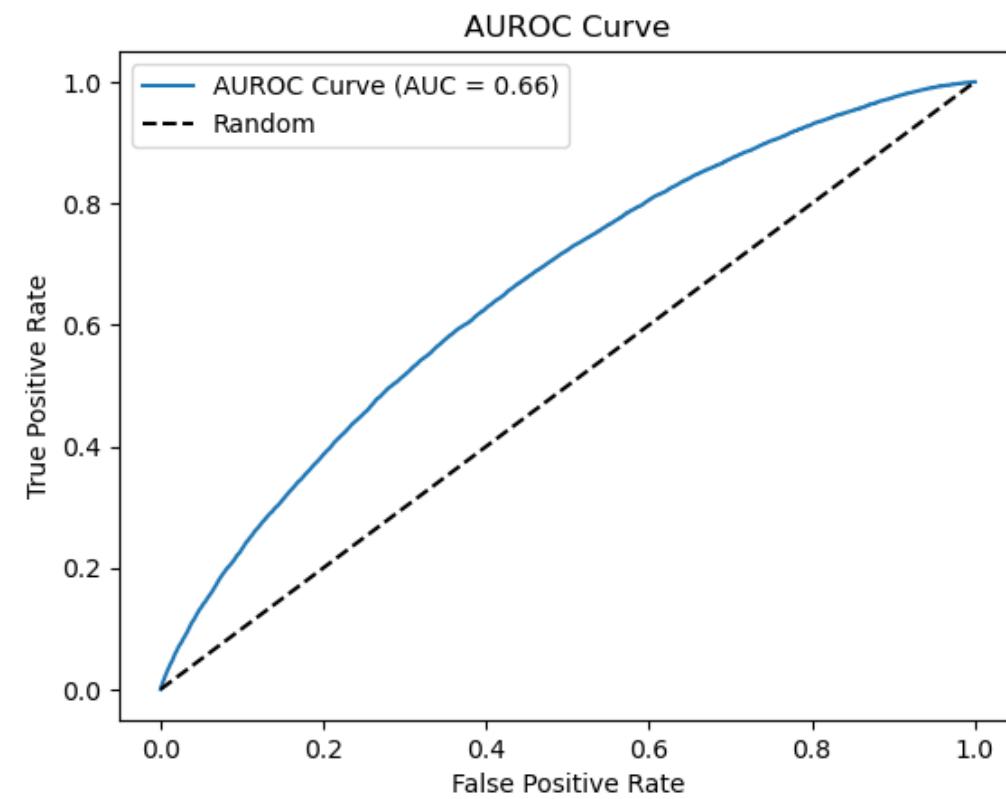


부록

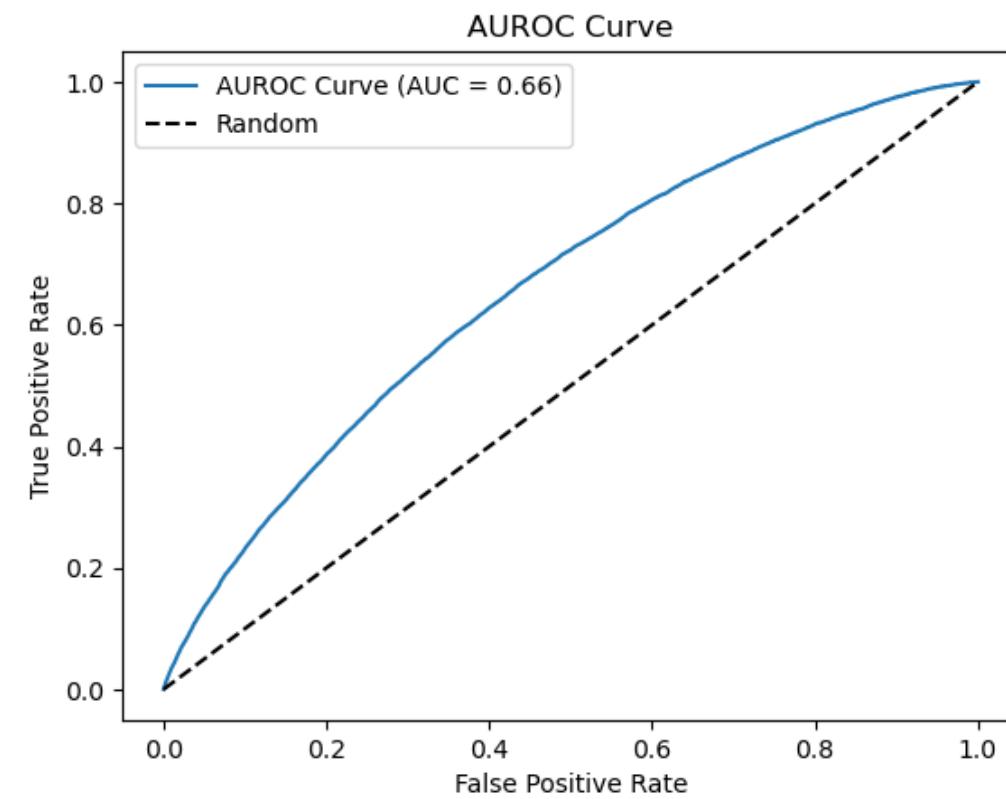
부록 1: AUROC 곡선

본론

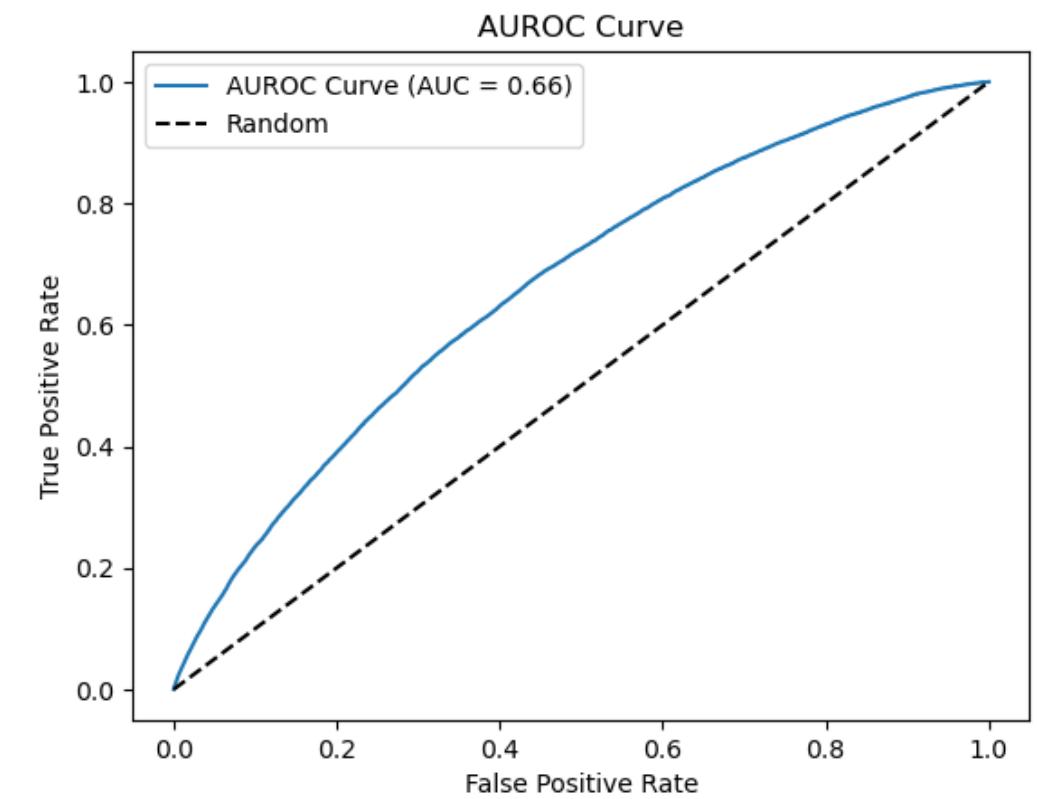
Logistic 모델의 AUROC 곡선



Lasso 변수 추출



ElasticNet 변수 추출

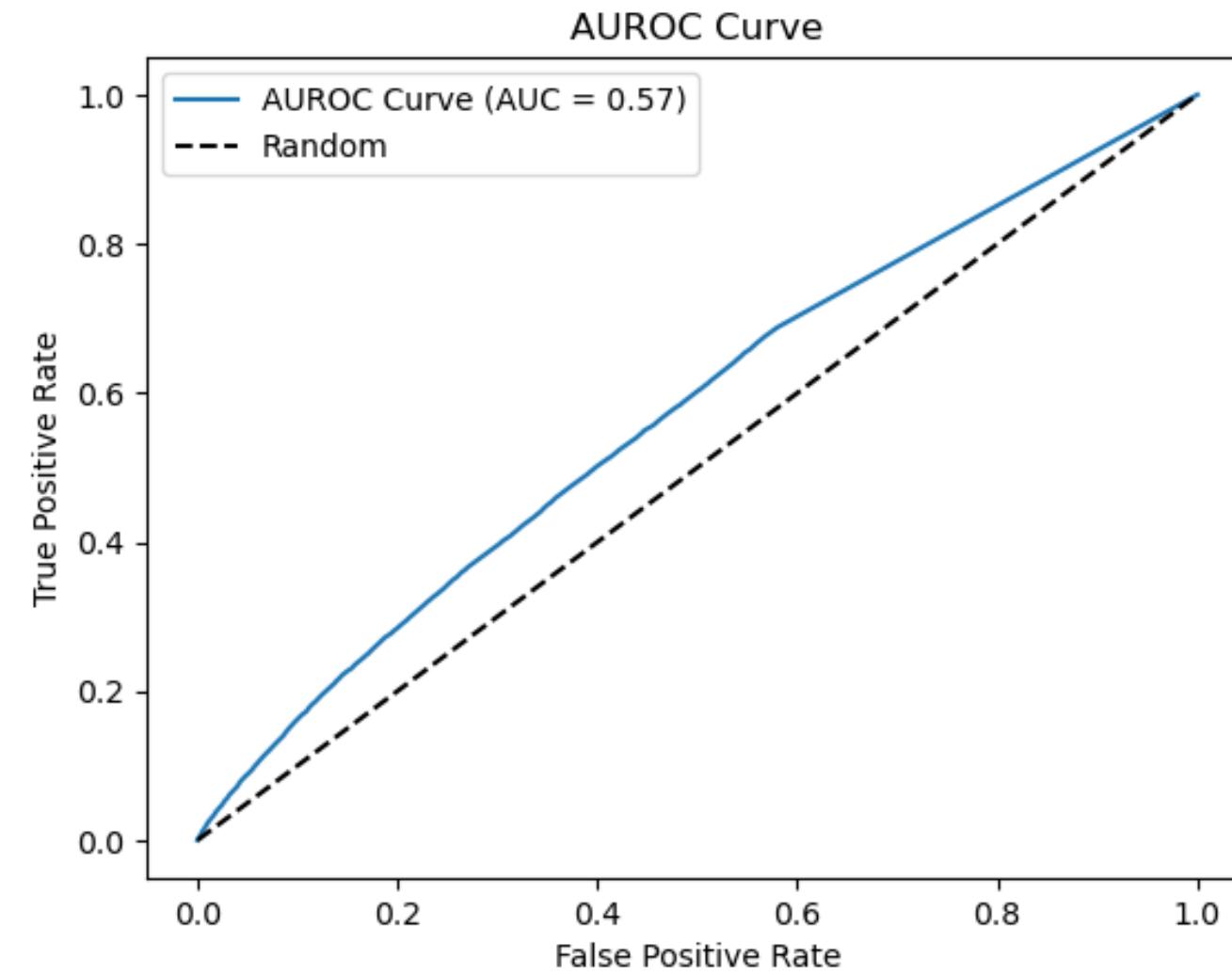


변수 추출 과정 X

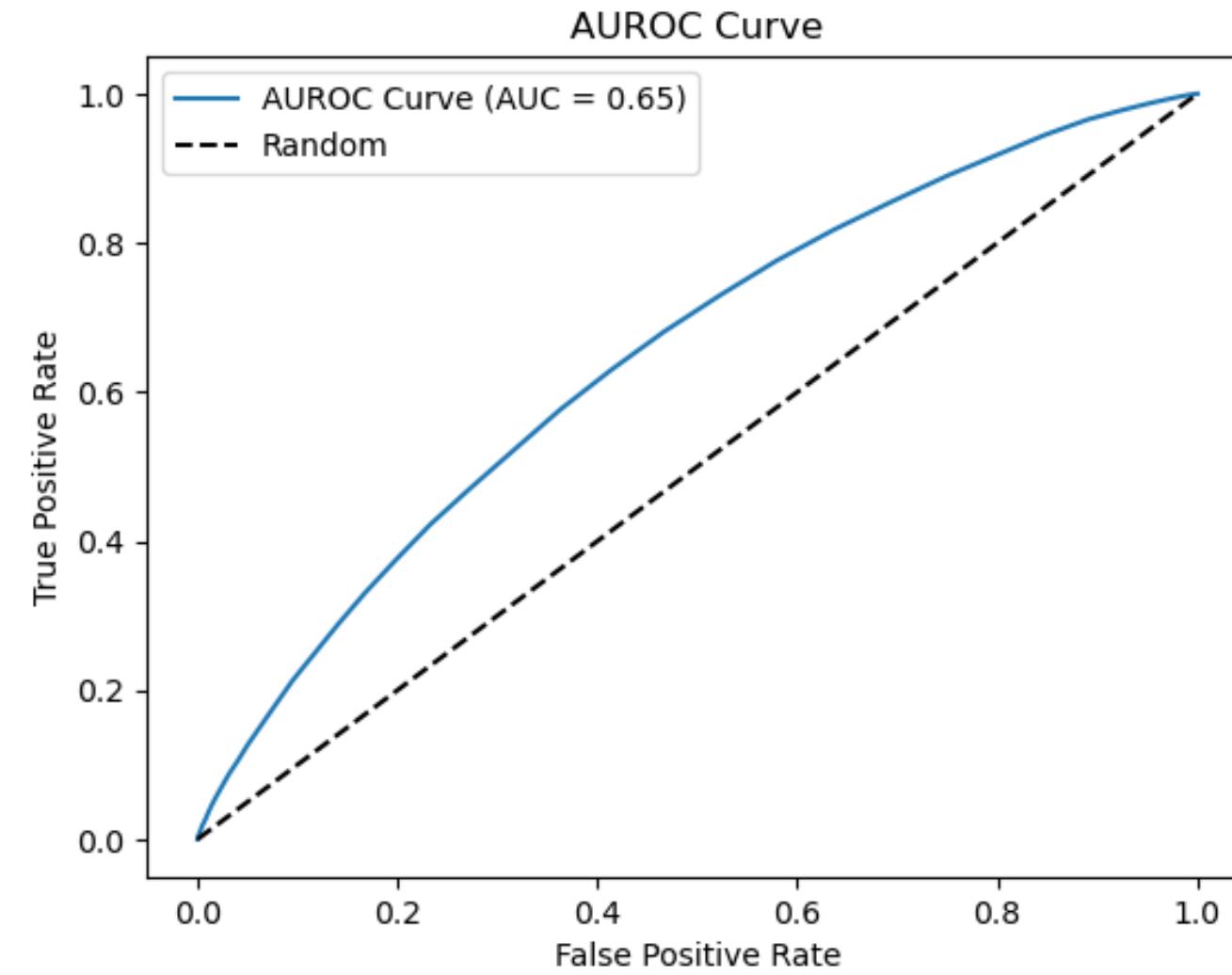
부록 1: AUROC 곡선

본문

Decision Tree



Random Forest



부록 2: Cost 함수 가중치에 따른 분석 표

계수별 cutoff (optimal threshold), 손실, 인당 손실, 수익, 인당 수익, precision, recall, f1 score, accuracy 값을 나타낸 표

계수 범위(0.5,0.5) ~ (0.9,0.1)

인당 손실: 손실 / test set 데이터 개수
 인당 수익: 수익 / test set 데이터 개수

Lasso + Logistic

	(0.5,0.5)	(0.55,0.45)	(0.65,0.35)	(0.75,0.25)	(0.85,0.15)	(0.9,0.1)
Cutoff	0.56	0.45	0.35	0.27	0.15	0.09
Loss	192039477	191616343	187293324	174726223	158396104	178301981
Loss per person	1250.33	1247.57	1219.42	1137.60	1031.28	1160.88
Net income	23979172	24355895	27764536	37400636	46471398	24791154
Net income per person	156.12	158.57	180.76	243.50	302.56	161.41
Precision	0.5	0.471	0.376	0.310	0.207	0.163
Recall	9.65	0.0015	0.0209	0.101	0.551	0.868
F1 score	0.00019	0.0031	0.0397	0.153	0.301	0.275
Accuracy	0.865	0.865	0.863	0.848	0.654	0.383

부록 2: Cost 함수 가중치에 따른 분석 표

Elastic + Logistic

	(0.5,0.5)	(0.55,0.45)	(0.65,0.35)	(0.75,0.25)	(0.85,0.15)	(0.9,0.1)
Cutoff	0.56	0.56	0.35	0.25	0.15	0.09
Loss	192039477	192039477	187303972	171016851	158363268	178269504
Loss per person	1250.33	1250.33	1219.49	1113.45	1031.07	1160.67
Net income	23979172	23979172	27765731	40177388	46514556	24845732
Net income per person	156.12	156.12	180.77	261.58	302.84	161.76
Precision	0.5	0.5	0.3757	0.2916	0.2072	0.1636
Recall	9.65	9.65	0.0209	0.143	0.551	0.868
F1 score	0.00019	0.00019	0.0396	0.192	0.301	0.275
Accuracy	0.865	0.865	0.863	0.837	0.654	0.383

부록 2: Cost 함수 가중치에 따른 분석 표

Logistic Regression

	(0.5,0.5)	(0.55,0.45)	(0.65,0.35)	(0.75,0.25)	(0.85,0.15)	(0.9,0.1)
Cutoff	0.51	0.51	0.36	0.27	0.15	0.1
Loss	191977539	191977539	187563884	174025662	157154481	172634744
Loss per person	1249.92	1249.92	1221.19	1133.04	1023.20	1123.98
Net income	24025385	24025385	27541350	38086845	47619391	30592425
Net income per person	156.42	156.42	179.31	247.97	310.04	199.18
Precision	0.4117	0.4117	0.3917	0.3121	0.2090	0.1712
Recall	0.00033	0.00033	0.018	0.107	0.559	0.824
F1 score	0.00067	0.00067	0.035	0.160	0.304	0.283
Accuracy	0.865	0.865	0.863	0.847	0.655	0.438

부록 2: Cost 함수 가중치에 따른 분석 표

Decision Tree

	(0.5,0.5)	(0.55,0.45)	(0.65,0.35)	(0.75,0.25)	(0.85,0.15)	(0.9,0.1)
Cutoff	1.01	1.01	1.01	0.62	0.24	0.0
Loss	192743322	192743322	192743322	188643051	178502544	203249575
Loss per person	1254.91	1254.91	1254.91	1228.21	1162.19	1323.31
Net income	23618456	23618456	23618456	26594243	31804361	0
Net income per person	153.77	153.77	153.77	173.14	207.07	0.0
Precision	0.0	0.0	0.0	0.265	0.178	0.134
Recall	0.0	0.0	0.0	0.0229	0.316	1.0
F1 score	0.0	0.0	0.0	0.0422	0.228	0.237
Accuracy	0.865	0.865	0.865	0.859	0.711	0.134

*cutoff 1.01은 모든 대출 신청자들의 대출을 승인한다는 의미로 사용하였다.

부록 2: Cost 함수 가중치에 따른 분석 표

Random Forest

	(0.5,0.5)	(0.55,0.45)	(0.65,0.35)	(0.75,0.25)	(0.85,0.15)	(0.9,0.1)
Cutoff	0.7	0.7	0.37	0.29	0.17	0.1
Loss	192091408	192091408	189065380	178594276	159079544	177120590
Loss per person	1250.66	1250.66	1230.96	1162.79	1035.73	1153.19
Net income	23937802	23937802	26334186	34684450	46863470	26269115
Net income per person	155.85	155.85	171.45	225.82	305.11	171.03
Precision	0.0	0.0	0.375	0.302	0.204	0.161
Recall	0.0	0.0	0.0137	0.0872	0.521	0.855
F1 score	0.0	0.0	0.0264	0.135	0.293	0.271
Accuracy	0.865	0.865	0.863	0.849	0.662	0.379

부록 2: Cost 함수 가중치에 따른 분석 표

다음은 가중치 계수를 (0.79,0.21)부터 (0.87,0.13)사이의 값으로 주었을 때의 분석표이다.

Lasso + Logistic

	(0.79,0.21)	(0.81,0.19)	(0.83,0.17)	(0.85,0.15)	(0.87,0.13)
Cutoff	0.22	0.19	0.17	0.15	0.13
Loss	164633568	159908985	158026647	158396104	162129996
Loss per person	1071.89	1041.13	1028.87	1031.28	1055.59
Net income	44679032	47367623	48037593	46471398	41923106
Net income per person	290.89	308.40	312.76	302.56	272.95
Precision	0.265	0.237	0.222	0.207	0.192
Recall	0.229	0.346	0.443	0.551	0.665
F1 score	0.246	0.282	0.296	0.301	0.298
Accuracy	0.810	0.762	0.716	0.654	0.577

부록 2: Cost 함수 가중치에 따른 분석 표

Elastic + Logistic

	(0.79,0.21)	(0.81,0.19)	(0.83,0.17)	(0.85,0.15)	(0.87,0.13)
Cutoff	0.22	0.19	0.17	0.15	0.13
Loss	164570568	159771897	157901391	158363268	162143395
Loss per person	1071.48	1040.24	1028.06	1031.07	1055.68
Net income	44724519	47505900	48127675	46514556	41887935
Net income per person	291.19	309.30	313.34	302.84	272.72
Precision	0.265	0.238	0.222	0.207	0.192
Recall	0.230	0.346	0.444	0.551	0.666
F1 score	0.246	0.282	0.296	0.301	0.298
Accuracy	0.810	0.762	0.715	0.654	0.576

부록 2: Cost 함수 가중치에 따른 분석 표

Logistic Regression

	(0.79,0.21)	(0.81,0.19)	(0.83,0.17)	(0.85,0.15)	(0.87,0.13)
Cutoff	0.22	0.188	0.177	0.15	0.12
Loss	163665651	158812300	157660664	157154481	164819175
Loss per person	1065.59	1033.99	1026.50	1023.20	1073.10
Net income	45459345	48292406	48679928	47619391	38812764
Net income per person	295.97	314.42	316.945	310.04	252.70
Precision	0.267	0.238	0.229	0.209	0.185
Recall	0.238	0.363	0.416	0.559	0.722
F1 score	0.251	0.287	0.296	0.304	0.294
Accuracy	0.809	0.757	0.733	0.655	0.533

*2,3번의 경우 소수점 2번째 자리까지의 cutoff가 같게 나와 소수점 3번째자리까지 추정하였다.

부록 2: Cost 함수 가중치에 따른 분석 표

Decision Tree

	(0.79,0.21)	(0.81,0.19)	(0.83,0.17)	(0.85,0.15)	(0.87,0.13)
Cutoff	0.54	0.43	0.33	0.24	0.01
Loss	186959621	182921892	179443470	178502544	180781946
Loss per person	1217.25	1190.96	1168.32	1162.19	1177.03
Net income	27789623	30390514	32308387	31804361	24909795
Net income per person	180.93	197.86	210.35	207.07	162.18
Precision	0.231	0.211	0.193	0.178	0.156
Recall	0.046	0.107	0.210	0.316	0.687
F1 score	0.0767	0.142	0.201	0.228	0.254
Accuracy	0.850	0.825	0.775	0.711	0.456

부록 2: Cost 함수 가중치에 따른 분석 표

Random Forest

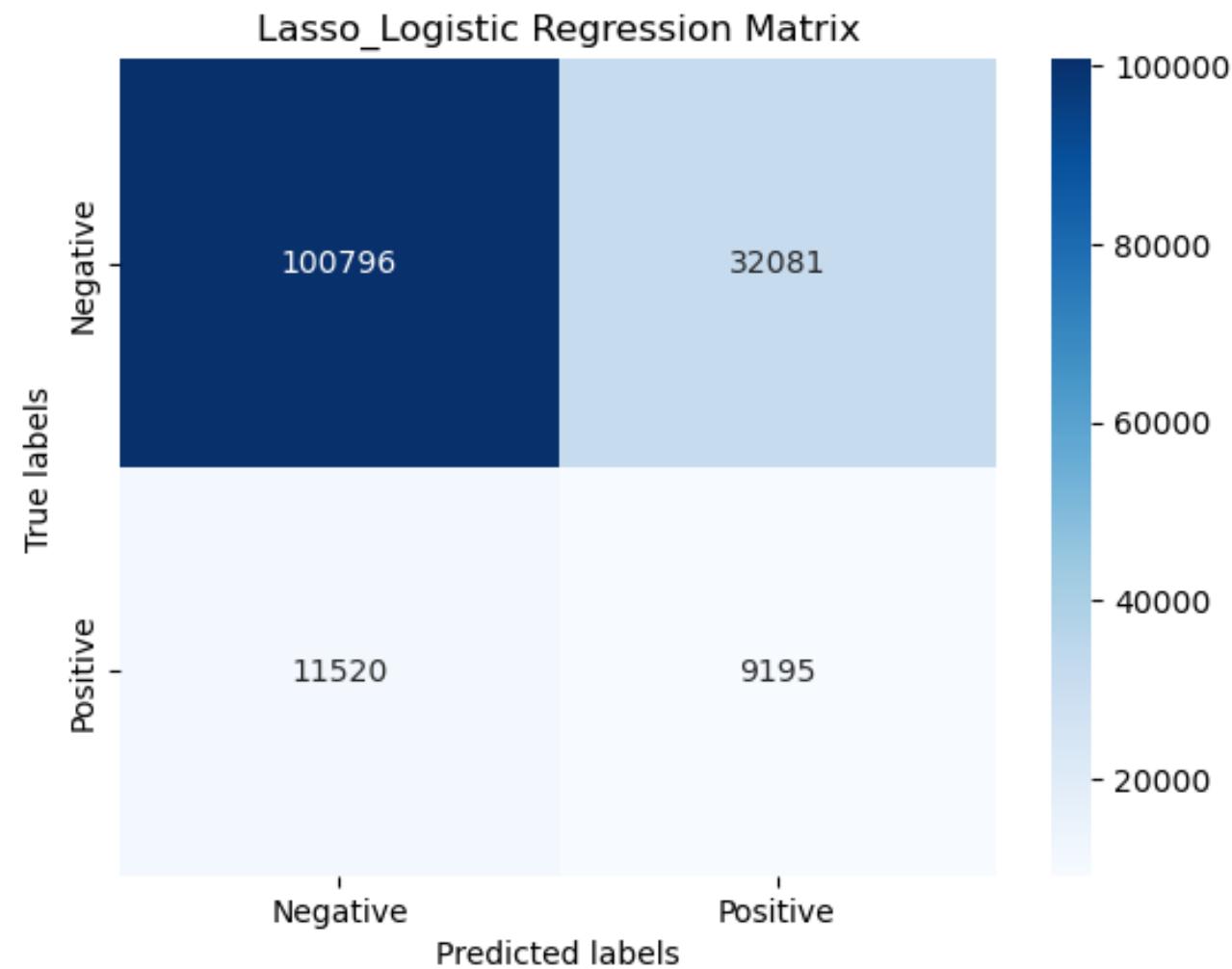
	(0.79,0.21)	(0.81,0.19)	(0.83,0.17)	(0.85,0.15)	(0.87,0.13)
Cutoff	0.24	0.22	0.19	0.17	0.14
Loss	167284197	163177179	158922164	159079544	162887948
Loss per person	1089.15	1062.41	1034.71	1035.73	1060.53
Net income	43031294	45825132	48132005	46863470	41536981
Net income per person	280.16	298.35	313.37	305.11	270.43
Precision	0.258	0.242	0.220	0.204	0.185
Recall	0.213	0.287	0.422	0.521	0.680
F1 score	0.233	0.263	0.289	0.293	0.291
Accuracy	0.811	0.782	0.720	0.662	0.553

부록 3: Confusion Matrix

모델의 confusion matrix 결과

Optimal Threshold : 가중치 계수 (C_{FN}, C_{FP}) = (0.83, 0.17)에서 구한 값

Lasso 변수 추출 후



ElasticNet 변수 추출 후

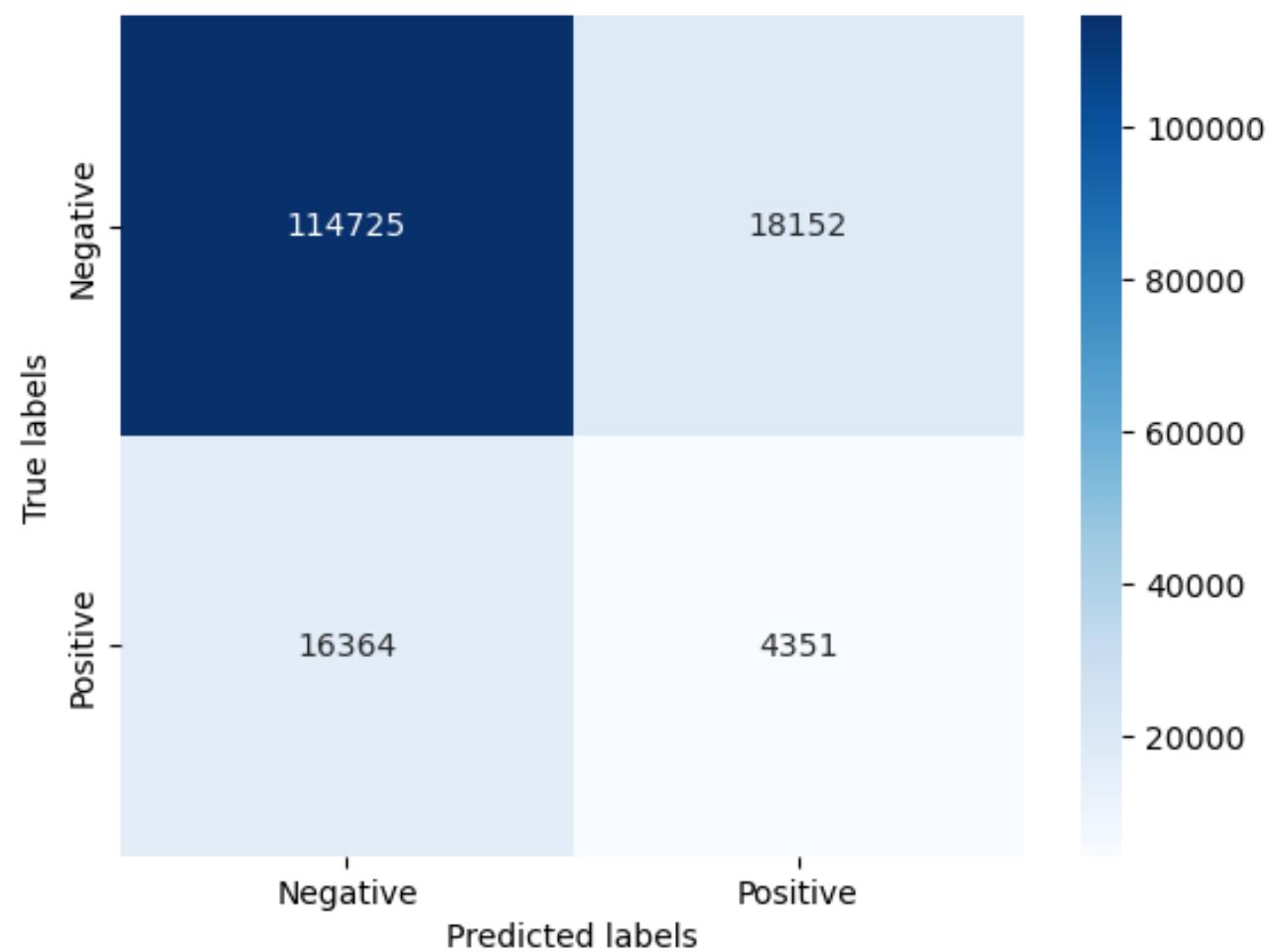


부록 3: Confusion Matrix

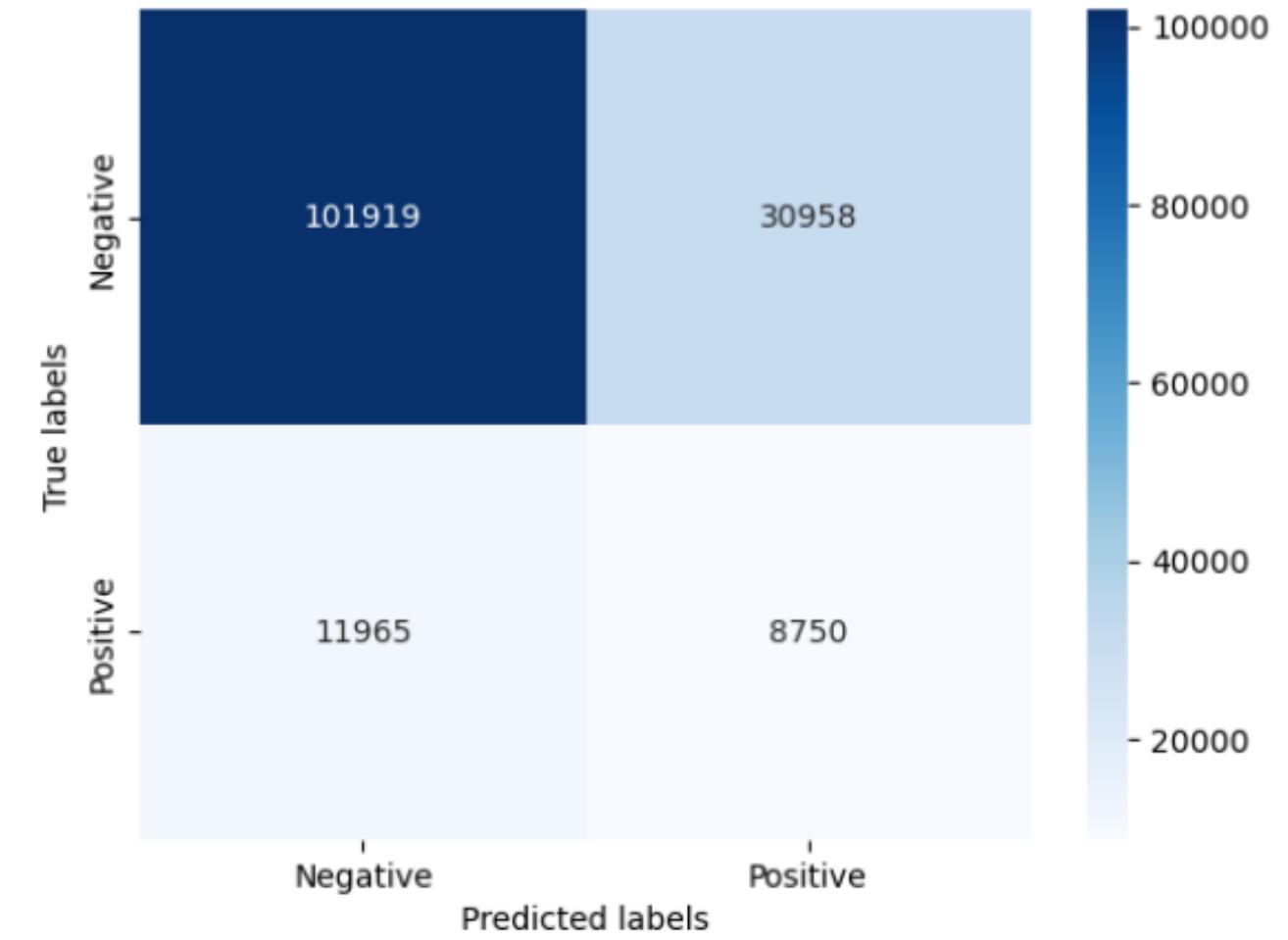
모델의 confusion matrix 결과

Optimal Threshold : 가중치 계수 (C_{FN}, C_{FP}) = (0.83, 0.17)에서 구한 값

Decision Tree



Random Forest



부록 4 : 모델을 새로운 데이터에 적용하기 (Over fitting 확인)

Out-of-sample-test

손실 최소화 모델				이익 최대화 모델			
Precision	Recall	F1_score	Accuracy	Precision	Recall	F1_score	Accuracy
0.25409	0.46722	0.32916	0.69567	0.27943	0.33639	0.30527	0.75533
0.25410	0.43088	0.31968	0.70441	0.27859	0.30911	0.29306	0.75963
0.26194	0.38859	0.31293	0.72779	0.29172	0.27473	0.28297	0.77788
0.22974	0.55713	0.32533	0.63359	0.24897	0.41979	0.31257	0.70720
0.24808	0.45977	0.32227	0.68886	0.27354	0.33864	0.30263	0.74888
0.24441	0.52484	0.33351	0.65422	0.26659	0.39023	0.31677	0.72253
0.24804	0.48935	0.32921	0.68735	0.26745	0.34666	0.30194	0.74870
0.25476	0.46898	0.33017	0.68703	0.27757	0.33602	0.30401	0.74696
0.24969	0.46433	0.32475	0.69293	0.27198	0.33458	0.30005	0.75176
0.24996	0.50014	0.33333	0.67481	0.26815	0.36279	0.30837	0.73547
0.24948	0.47512	0.32604	0.68466	0.27240	0.34489	0.30277	0.74543

THANK YOU!

3조 Treatment

김동욱 | 김민정 | 박미지 | 박민지 | 박성호 | 양수영