

GSC Big data boot camp

김현수 조현민 이화정 송재원 이동민

2nd semi project

# Vehicle Loan repayment prediction





# CONTENTS

## 01. Project Intro

- 구성원 및 역할
- 프로젝트 소개
- 데이터 구성

## 02. EDA & Processing

- 시각화
- 전처리

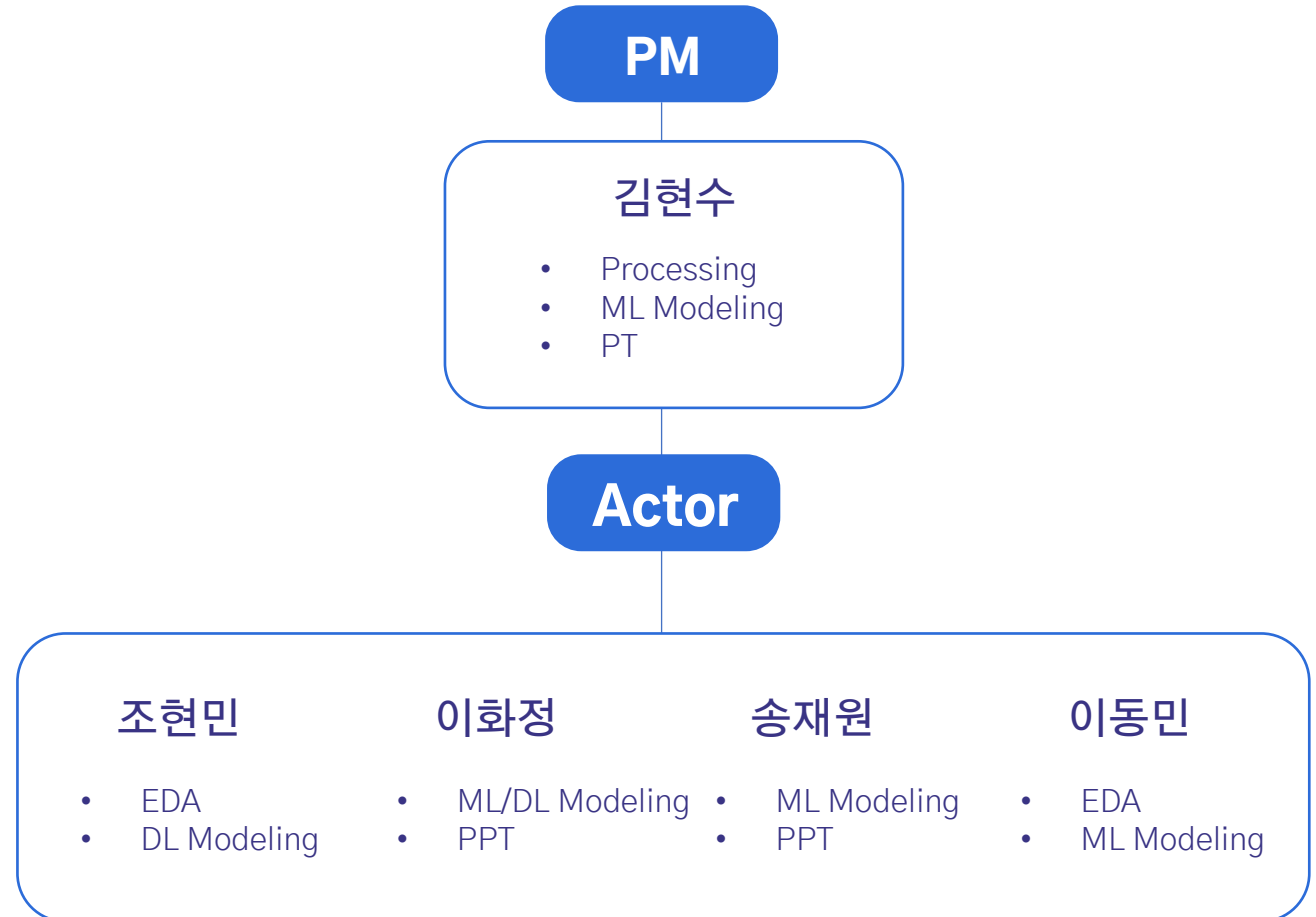
## 03. Modeling

- 적용 기법 소개
- 모델링 및 성능평가

## 04. Result & Evaluation

- 모델링 결과
- 향후 과제

# Team members



## 프로젝트 소개



# Vehicle Loan repayment prediction

**Problem:** 비은행 금융회사(NBFC)는 투자, 리스크 풀링, 계약 저축, 시장 중개 등 은행과 유사한 금융 서비스를 제공하는 데, 한 NBFC는 현재 차량 대출 부문의 채무 불이행 증가로 인해 수익성 문제에 직면

**Goal:** 고객의 대출 상환 능력에 기여하는 요소들을 살펴봄으로써 고객이 차량 대출 상환을 불이행할 가능성이 있는 지 여부를 예측

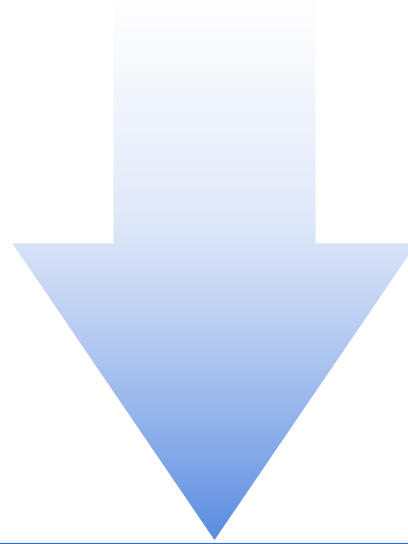
# 데이터 구성 - X data

Type	Column	Description
Identifier	ID	고객 ID
	Child_Count	고객 자녀 수
Numerical	Own_House_Age	고객 소유 주택의 나이(년)
	Credit_Amount	대출 금액(\$)
	Loan_Annuity	대출 연금(\$)
	Population_Region_Relative	고객이 거주하고 있는 지역의 상대적 인구수
	Age_Days	신청서 제출 시점의 고객 나이
	Employed_Days	대출 신청일 전 고객이 고용되어 일한 일수
	Registration_Days	대출 신청일 전 고객이 등록을 변경한 일수
	ID_Days	대출 신청일 전 고객이 대출을 신청한 신분증 변경 일수
	Client_Family_Members	고객 가족 구성원 수
	Score_Source_1	다른 출처에서 얻은 정규화된 점수
	Score_Source_2	다른 출처에서 얻은 정규화된 점수
	Score_Source_3	다른 출처에서 얻은 정규화된 점수
	Phone_Change	대출 신청 며칠 전에 고객이 휴대폰을 변경했는지
	Credit_Bureau	작년 총 문의 건수
	Client_Income	고객 소득(\$)
	Social_Circle_Default	지난 60일 동안 대출 상환을 불이행한 고객의 친구/가족 수
	Application_Process_Hour	고객이 대출을 신청한 날의 시간

Type	Column	Description
Catagorical (0 or 1 & Yes or No)	Car_Owned	다른 차량에 대한 대출을 신청하기 전에 고객이 소유한 모든 차량
	Bike_Owned	고객이 소유한 모든 자전거
	Active_Loan	대출 신청 당시 진행 중인 다른 대출이 있는지 여부
	House_Own	고객이 소유한 주택 수
	Homephone_Tag	고객이 제공한 집전화 번호
	Workphone_Working	직장 전화 번호로 연락 가능했는지
	Client_Permanent_Match_Tag	고객 연락처 주소가 영구 주소와 일치 여부
	Client_Contact_Work_Tag	고객 직장 주소가 연락처 주소와 일치 여부
	Client_Education	고객이 달성한 최고 수준의 교육 수준
	Client_Marital_Status	고객의 결혼 상태
Catagorical (the rest)	Client_Gender	고객 성별
	Cleint_City_Rating	고객 도시 등급
	Loan_Contract_Type	대출 유형
	Client_Housing_Type	고객 집 상태
	Client_Occupation	고객 직업 유형
	Type_Organization	고객이 근무하는 조직 유형
	Application_Process_Day	고객이 대출을 신청한 요일
	Accompany_Client	고객이 대출을 신청할 때 고객과 동행한 사람
	Client_Income_Type	고객 소득 유형
	Mobile_Tag	고객이 제공한 휴대폰 번호(1의 값만 가짐)
etc		

Data Source: Kaggle

## 데이터 구성 - Y data



Type	Column	Description
Categorical	Default	대출 상환 불이행 여부 (이행: 0, 불이행: 1)

식별자(ID)를 제외한 38개의 컬럼을 활용하여 대출 상환 불이행 여부 예측

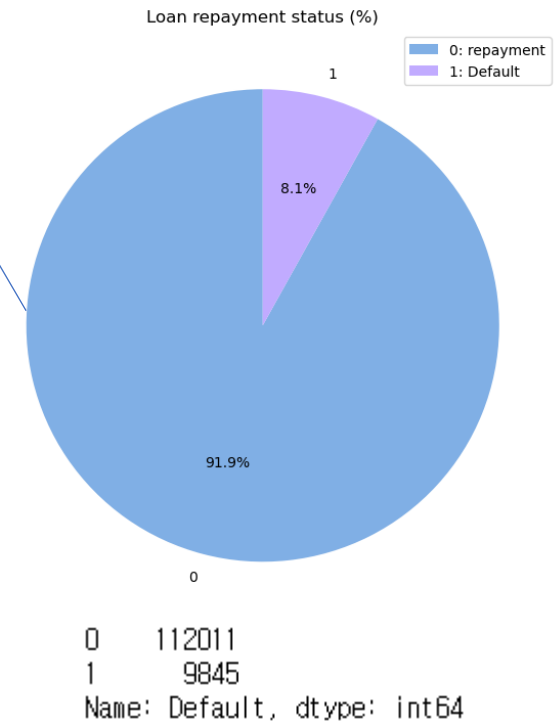
# EDA

“ 채무 불이행한 고객의 비율은 몇 % 일까? ”

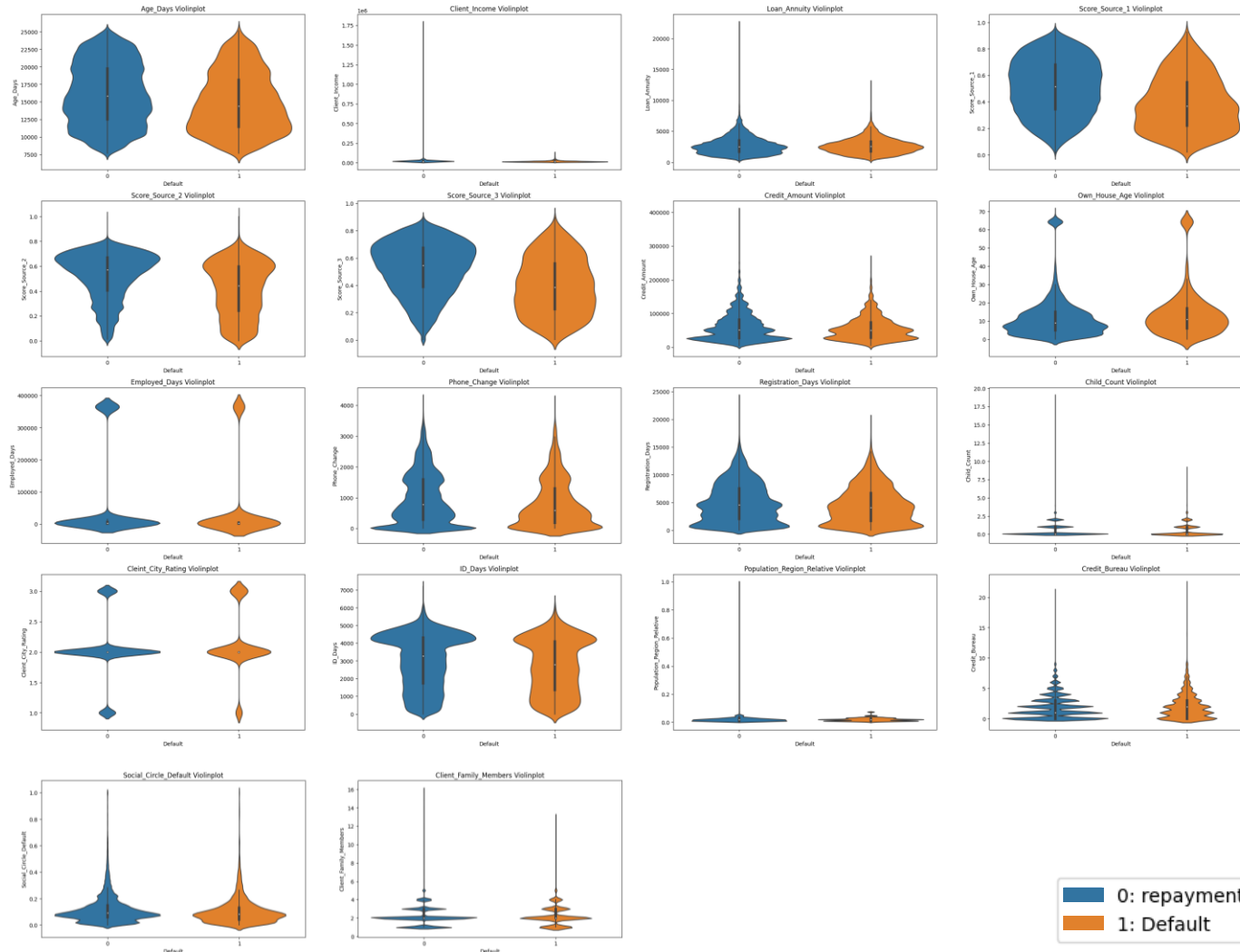
약 12만 건의 데이터 중  
채무를 상환한 고객이 91.9%,  
채무 상환을 불이행한 고객이 8.1%로

예측 라벨 값의 분포가 불균등하게 나타남을 확인 할 수 있다.

그렇다면 상환 여부에 따라  
고객 간에 어떠한 차이가 존재할까 ?



# EDA – Numerical columns

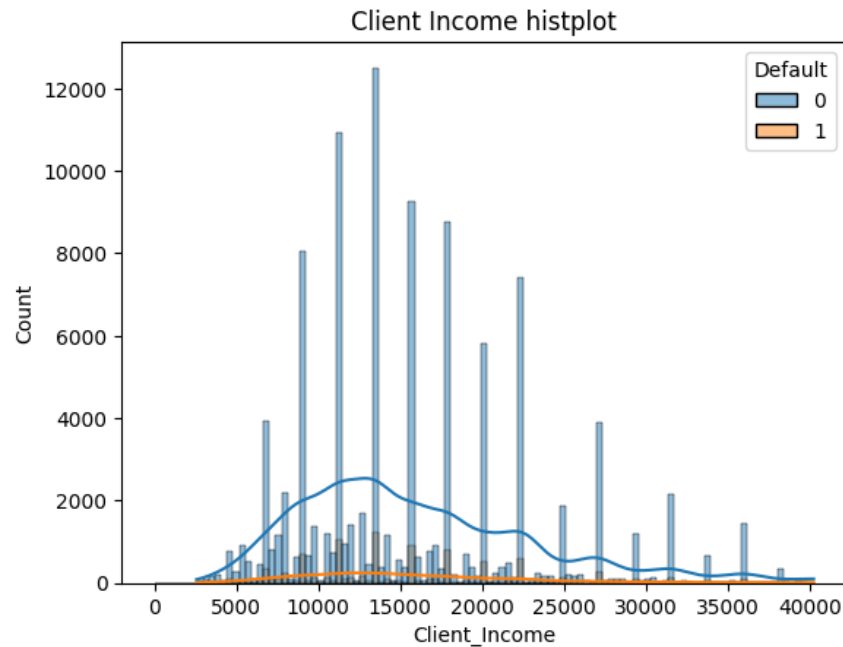


Violin Plot 결과  
채무 상환 이행 여부에 따라 분포 차이가  
크게 나타나지 않음을 확인

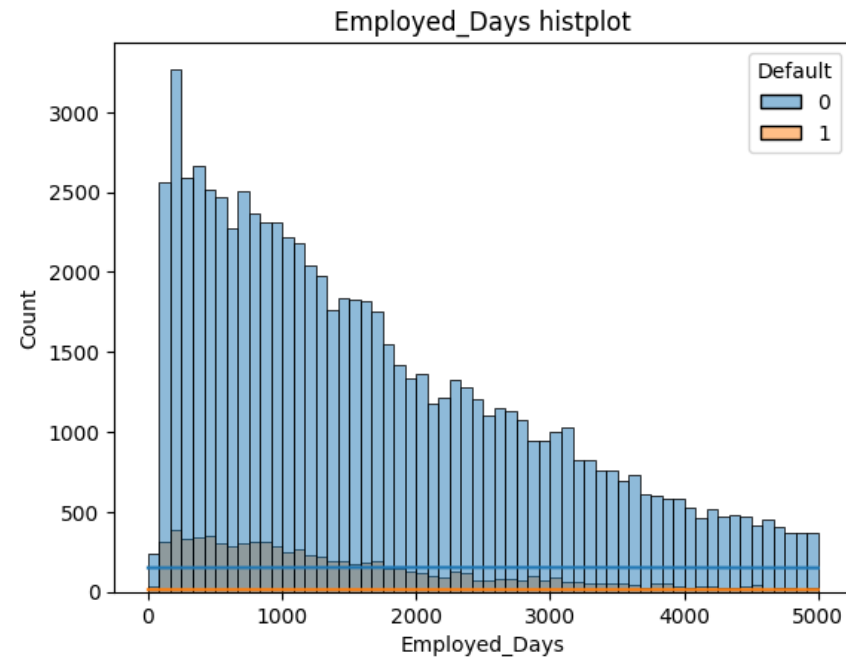


# EDA – Numerical columns

Violin Plot 결과, 분포가 극단적으로 나타나는 column만 추출해서 세부적으로 시각화 했을 때  
채무 상환 불이행 여부에 따라 분포 차이가 나지 않으며 상대적으로 소득이 낮고 일한 일수 또한 작은 방향으로 빈도수가 높게 나타남

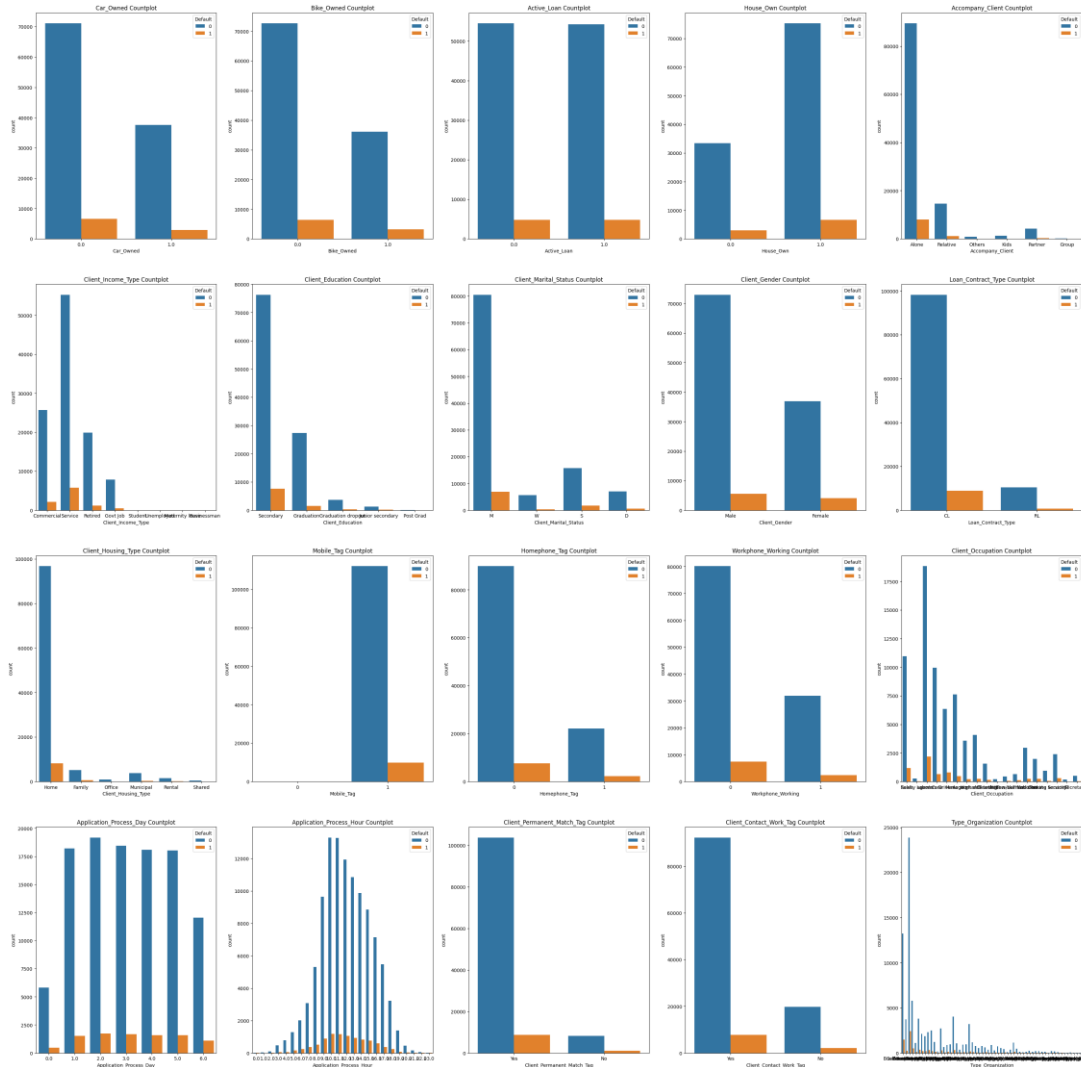


1. Client Income



2. Employed Days

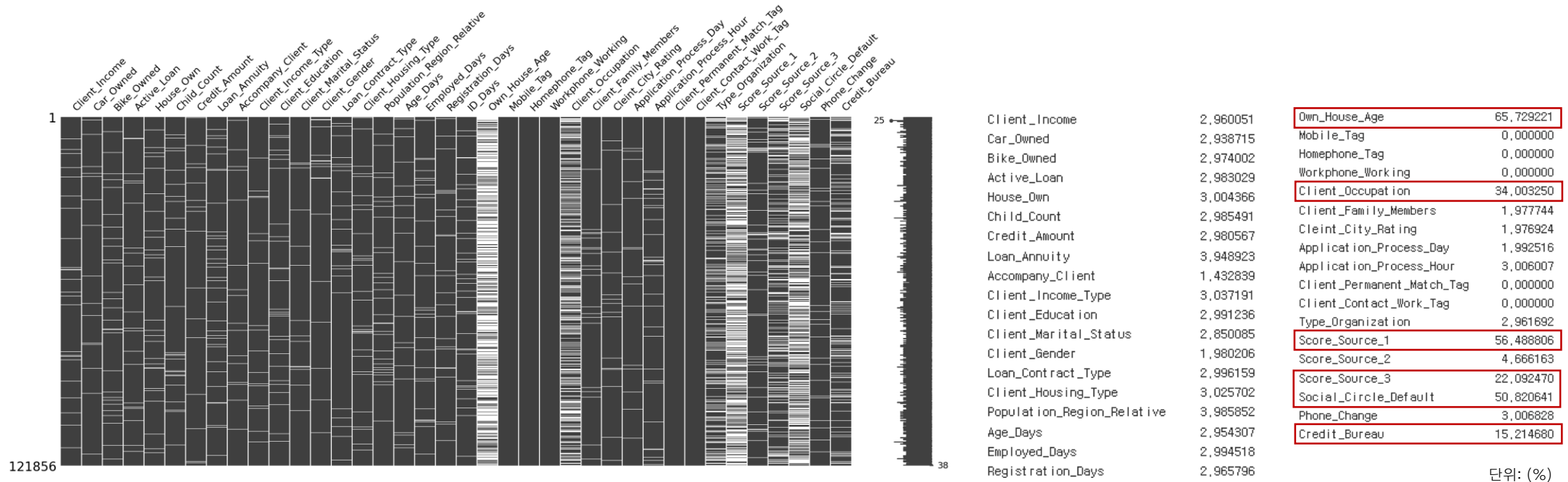
# EDA – Categorical columns



Categorical columns는 Histogram Plot 결과 Numerical columns와 마찬가지로 채무 상환 이행 여부에 따라 분포 차이가 뚜렷하게 나타나지 않음

Y data의 라벨 값 불균형으로 인해 빈도 수 차이는 존재하나 전체적인 분포는 비슷함을 파악 가능

# EDA – Missing Values



Missing Values가 10% 이상인 Columns: Credit\_Bureau(15.2%) < Score\_Source\_3(22.1%) < Client\_Occupation(34.0%) < Social\_Circle\_Default(50.8%) < Score\_Source\_1(56.5%)

# EDA – Outliers

## 1. Type Organization

XNA: 21085

Police, Trade, Hotel 등  
조직 유형이 들어가 있어야 하므로  
이상치

## 2. Client Gender

XNA: 3

성별이므로 여성 혹은 남성만  
들어갈 수 있으므로 이상치

## 3. Accompany Client

##: 3

Alone, Partner, Kids 등  
대출 당시 고객과 동행한 이들의  
유형이 들어가 있어야 하므로 이상치

Categorical columns들을 Value\_Counts()를 통해 고유값 별 개수를 센 결과 위와 같은 이상치 파악

# Processing

- **Drop Columns**

- ID : 식별자
- Own\_House\_Age, Score\_Source\_1 & Score\_Source\_3, Social\_Circle\_Default: 결측치 비중이 높으므로 제거
- Type\_Organization : Client\_Occupation과 중복되는 부분이라 판단하여 제거
- Mobile\_Tag : 데이터가 전부 1인 관계로 제거
- Application\_Process\_Hour & Accompany\_Client & Client\_Income\_Type : EDA결과를 통해 필요없는 컬럼이라 판단하여 제거

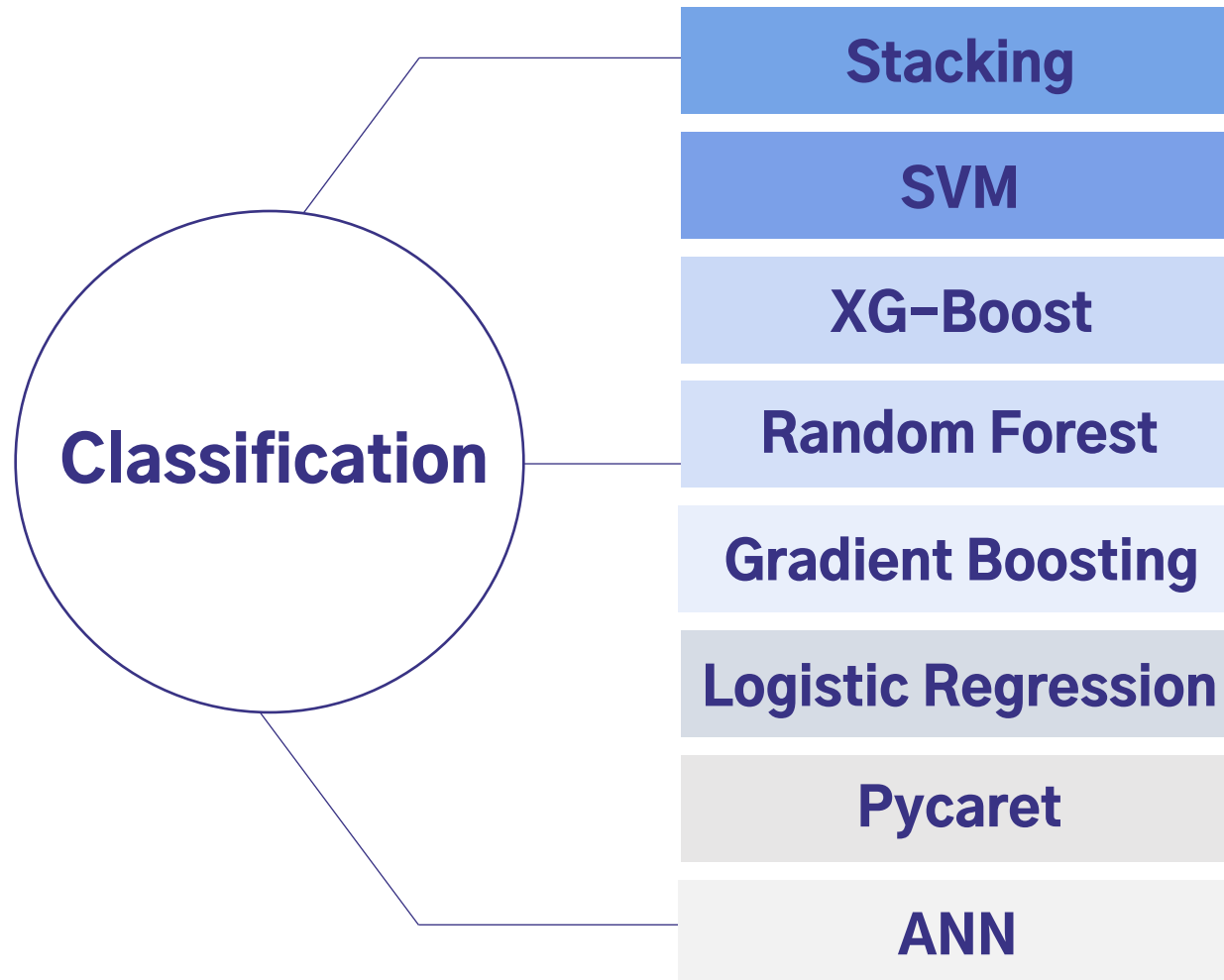
- **Missing values & Outliers**

- 이상치 값을 na\_values = ['\$', '#VALUE!', '##', 'XNA', '@', '#', 'x', '&']로 묶어서 결측치로 처리 후 아래와 같이 처리
- Client\_Occupation : 결측치가 다수였으나, 고객 직업 유형에 따라 소득 및 상환 이행 여부에 영향을 미칠 것이라 판단하여 Nojob으로 대체
- Categorical Columns:  $0 < \text{결측치} < 10000$  대상으로 각 컬럼 별 고유값 중 랜덤하게 대체
- 나머지 Categorical Columns: One-hot encoding 적용
- Numerical Columns: 평균값으로 대체

- **Oversampling(SMOTE)**

- Y data의 라벨 값 분포가 불균등하게 나타나는 관계로 Oversampling을 통해서 데이터 불균형 해소

# Model Types



\* Random Forest 외 ML Model에 Grid search 기법 적용하여 HPO 도출 후 모델링

# Model(ML)

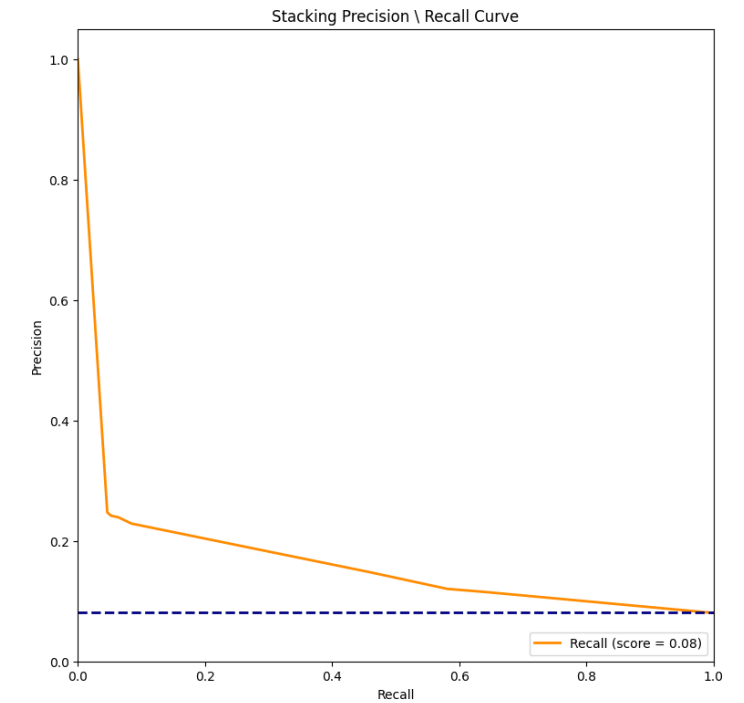
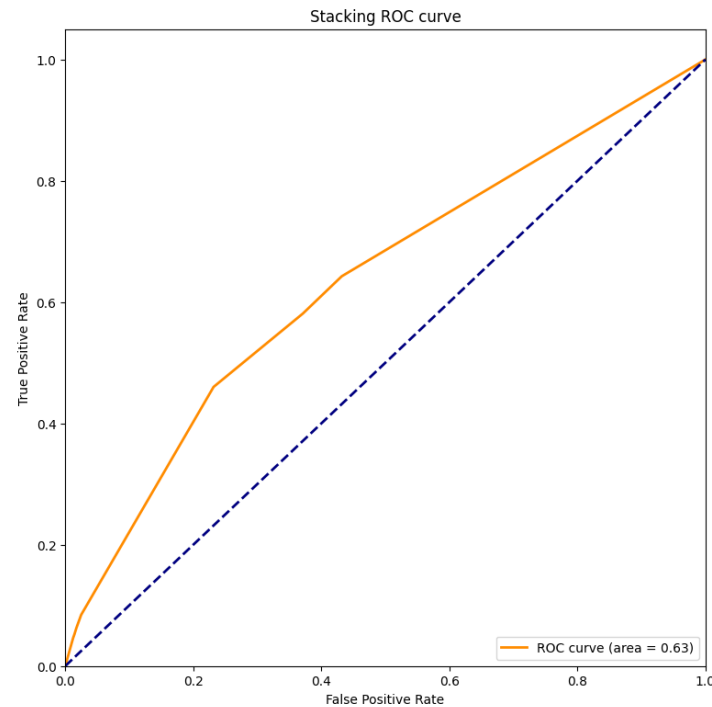
## 1. Stacking

(ExtraTrees, RandomForest, DecisionTree)

	precision	recall	f1-score	support
0	0.92	0.98	0.95	33619
1	0.23	0.08	0.12	2938
accuracy			0.90	36557
macro avg	0.58	0.53	0.54	36557
weighted avg	0.87	0.90	0.88	36557

Accuracy on Training set: 0.883

Accuracy on Test set: 0.904



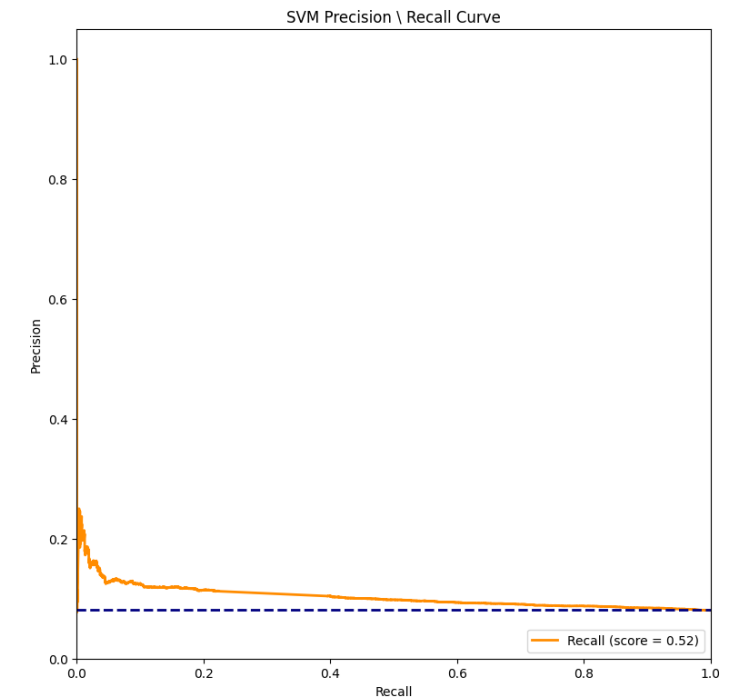
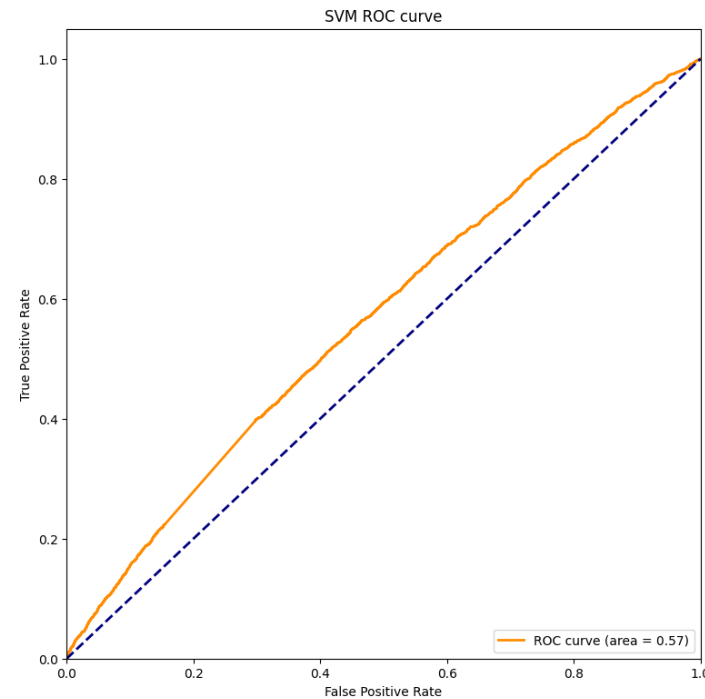
# Model(ML)

## 2. SVM

	precision	recall	f1-score	support
0	0,93	0,58	0,71	33619
1	0,10	0,52	0,16	2938
accuracy			0,57	36557
macro avg	0,51	0,55	0,44	36557
weighted avg	0,87	0,57	0,67	36557

Accuracy on Training set: 0,608

Accuracy on Test set: 0,574





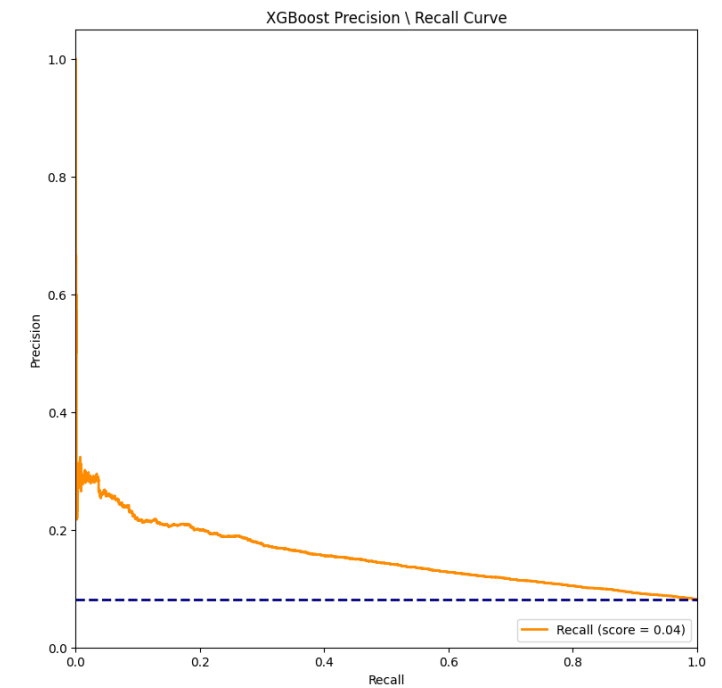
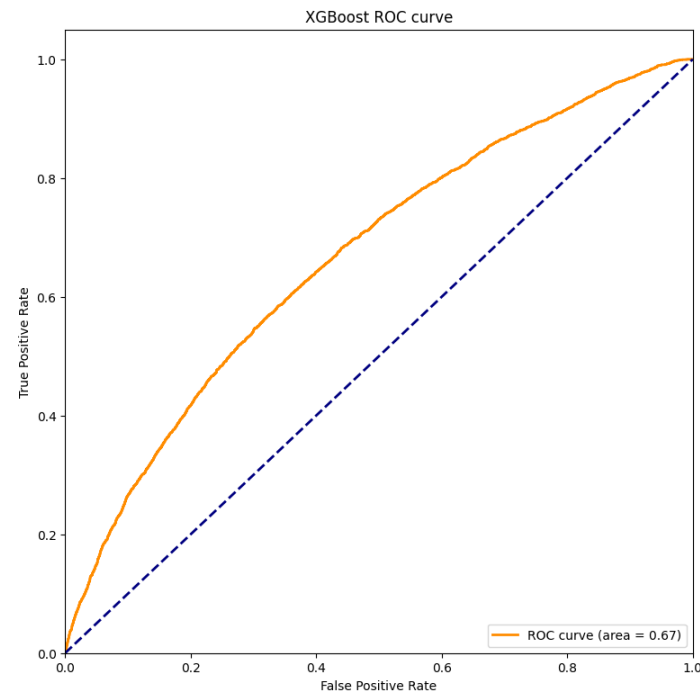
# Model(ML)

## 3. XG-Boost

	precision	recall	f1-score	support
0	0,92	0,99	0,95	33619
1	0,26	0,04	0,07	2938
accuracy			0,91	36557
macro avg	0,59	0,52	0,51	36557
weighted avg	0,87	0,91	0,88	36557

Accuracy on Training set: 0,950

Accuracy on Test set: 0,913



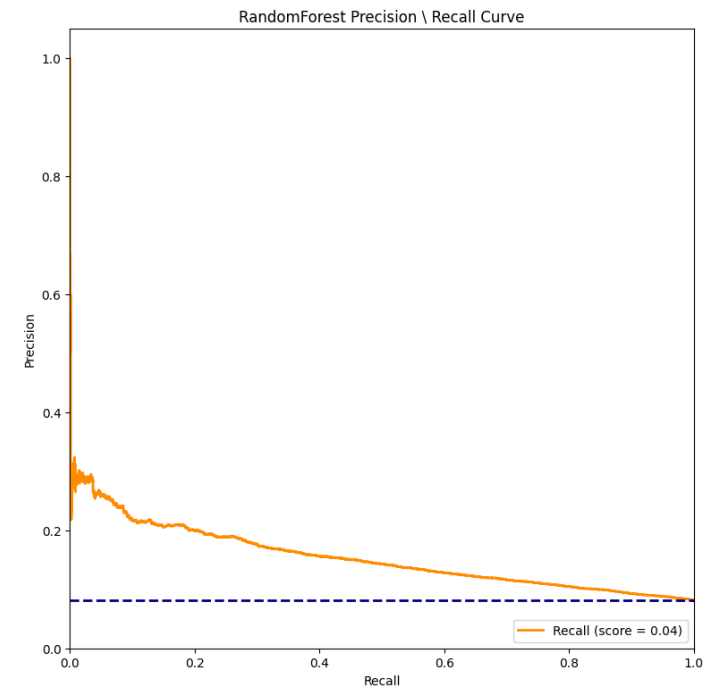
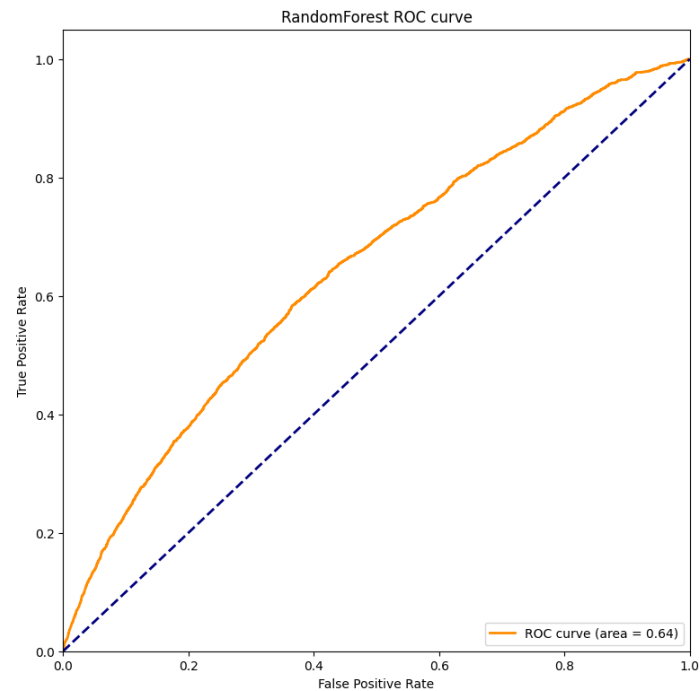
# Model(ML)

## 4. Random Forest

	precision	recall	f1-score	support
0	0,94	0,74	0,83	33619
1	0,13	0,46	0,21	2938
accuracy			0,72	36557
macro avg	0,54	0,60	0,52	36557
weighted avg	0,87	0,72	0,78	36557

Accuracy on Training set: 0,775

Accuracy on Test set: 0,717



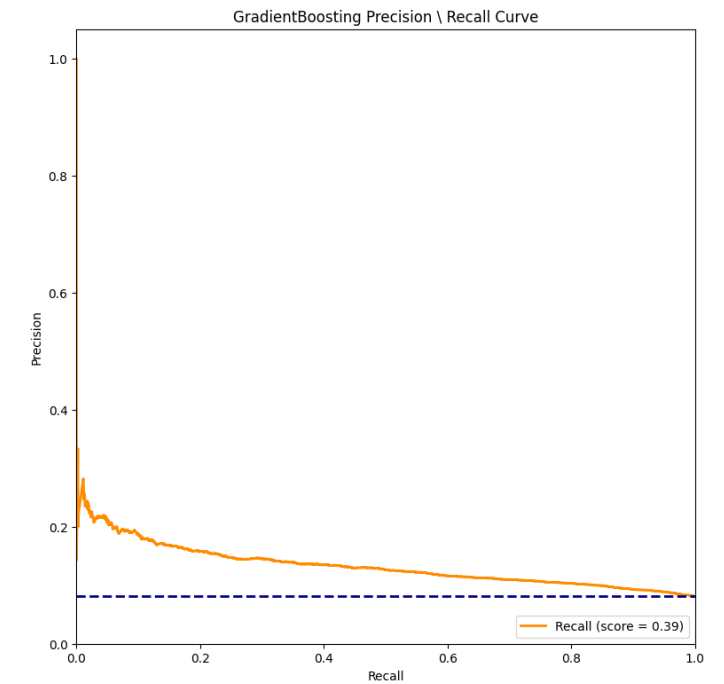
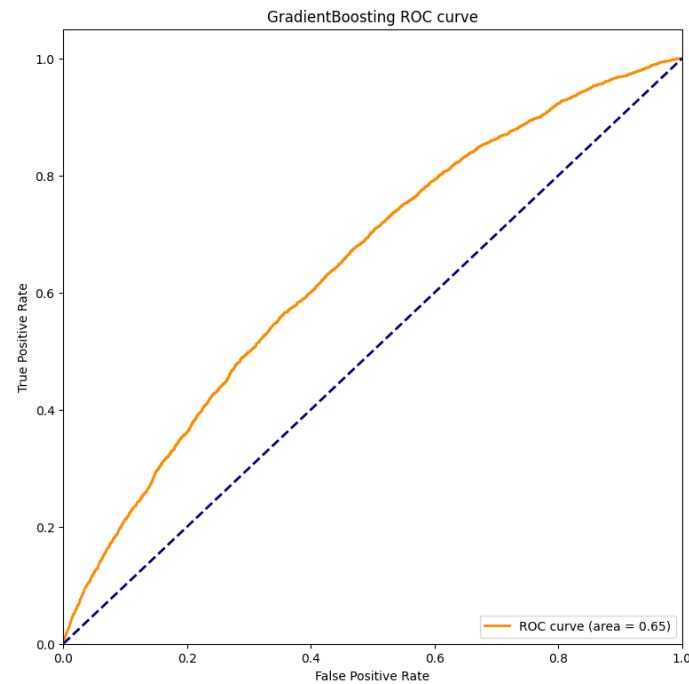
# Model(ML)

## 5. Gradient Boosting

	precision	recall	f1-score	support
0	0,94	0,79	0,85	33619
1	0,14	0,39	0,20	2938
accuracy			0,75	36557
macro avg	0,54	0,59	0,53	36557
weighted avg	0,87	0,75	0,80	36557

Accuracy on Training set: 0,789

Accuracy on Test set: 0,753



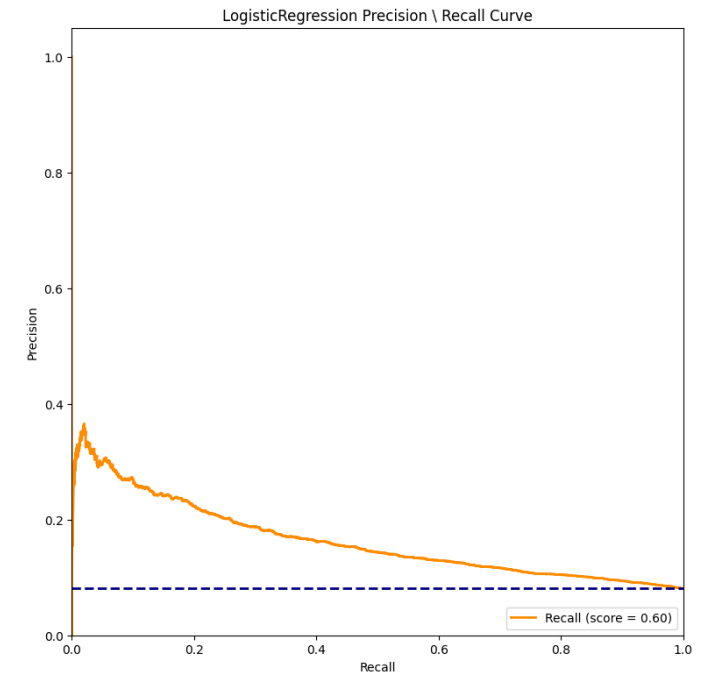
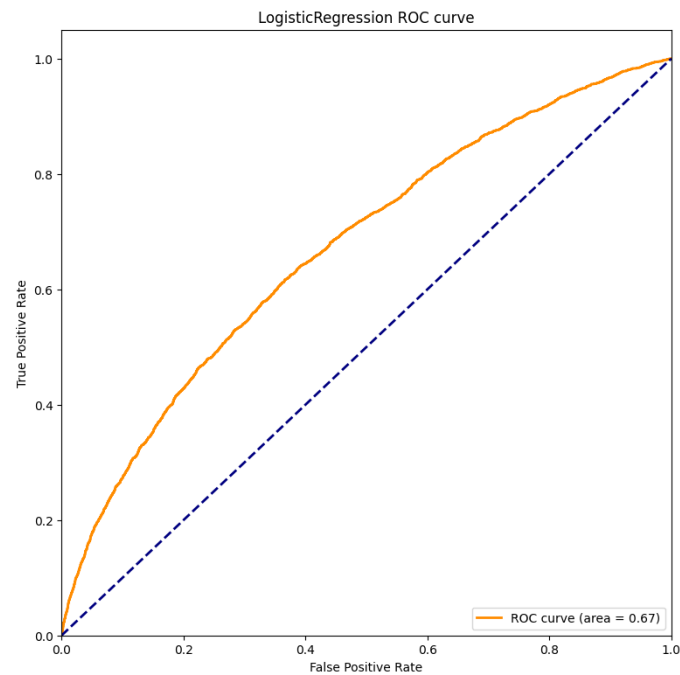
# Model(ML)

## 6. Logistic Regression

	precision	recall	f1-score	support
0	0,95	0,65	0,77	33619
1	0,13	0,60	0,21	2938
accuracy			0,65	36557
macro avg	0,54	0,62	0,49	36557
weighted avg	0,88	0,65	0,73	36557

Accuracy on Training set: 0,660

Accuracy on Test set: 0,647



# Model(ML)

## 7. Pycaret (Auto ML library)

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
<b>qda</b>	Quadratic Discriminant Analysis	0.0803	0.5000	1.0000	0.0803	0.1487	0.0000	0.0000	9.951
<b>nb</b>	Naive Bayes	0.4817	0.5746	0.6157	0.0921	0.1602	0.0238	0.0467	8.197
<b>ridge</b>	Ridge Classifier	0.6500	0.0000	0.6017	0.1319	0.2164	0.0975	0.1444	8.169
<b>lda</b>	Linear Discriminant Analysis	0.6500	0.6763	0.6017	0.1319	0.2164	0.0975	0.1444	12.161
<b>lr</b>	Logistic Regression	0.6522	0.6766	0.5996	0.1324	0.2169	0.0982	0.1449	14.620
<b>svm</b>	SVM - Linear Kernel	0.6483	0.0000	0.5872	0.1307	0.2127	0.0939	0.1375	10.769
<b>dt</b>	Decision Tree Classifier	0.8555	0.5718	0.2338	0.1845	0.2062	0.1280	0.1291	13.377
<b>ada</b>	Ada Boost Classifier	0.8931	0.6499	0.1099	0.1990	0.1404	0.0890	0.0938	32.502
<b>gbc</b>	Gradient Boosting Classifier	0.9178	0.6647	0.0168	0.3010	0.0317	0.0234	0.0541	127.509
<b>lightgbm</b>	Light Gradient Boosting Machine	0.9194	0.6952	0.0072	0.4052	0.0140	0.0112	0.0446	16.242
<b>dummy</b>	Dummy Classifier	0.9197	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	7.972

# Model(DL)

## 8. ANN

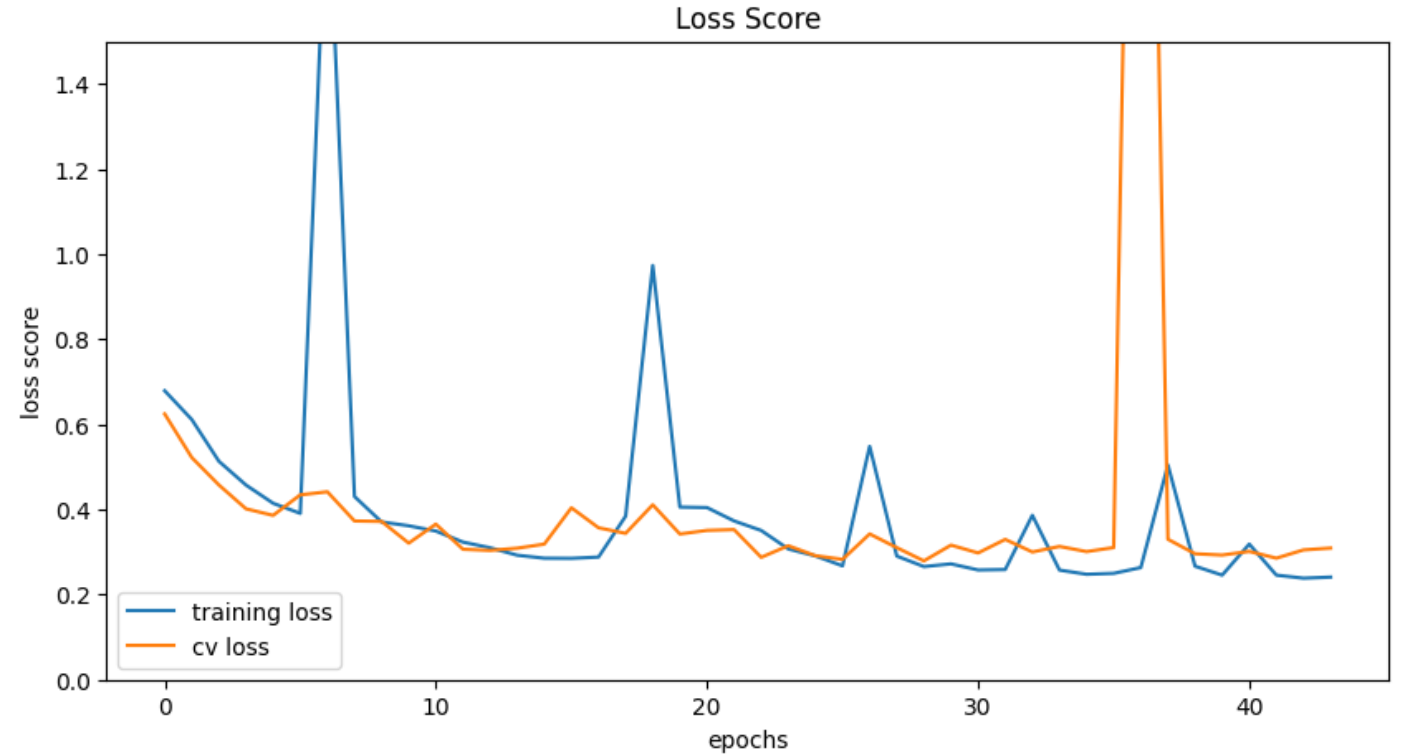
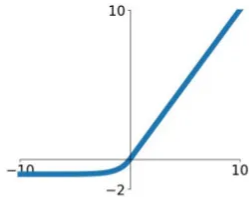
Model: "predict"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 140)	11760
dense_1 (Dense)	(None, 140)	19740
dense_2 (Dense)	(None, 40)	5640
dropout (Dropout)	(None, 40)	0
dense_3 (Dense)	(None, 1)	41

=====  
 Total params: 37,181  
 Trainable params: 37,181  
 Non-trainable params: 0

### ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Loss Score (0.2792655825614929)로 가장 낮은 29 번째 가중치를 사용

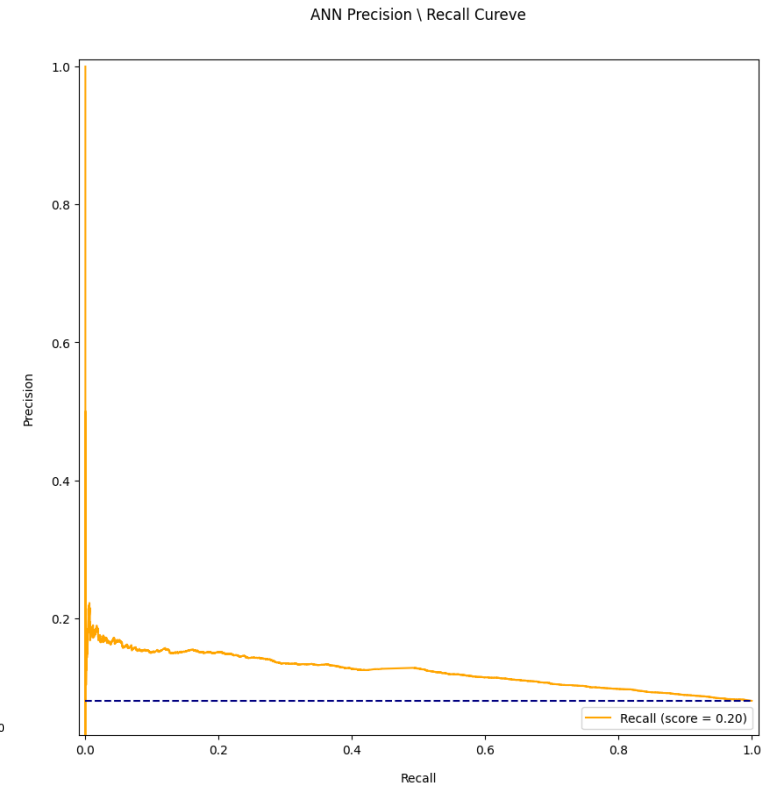
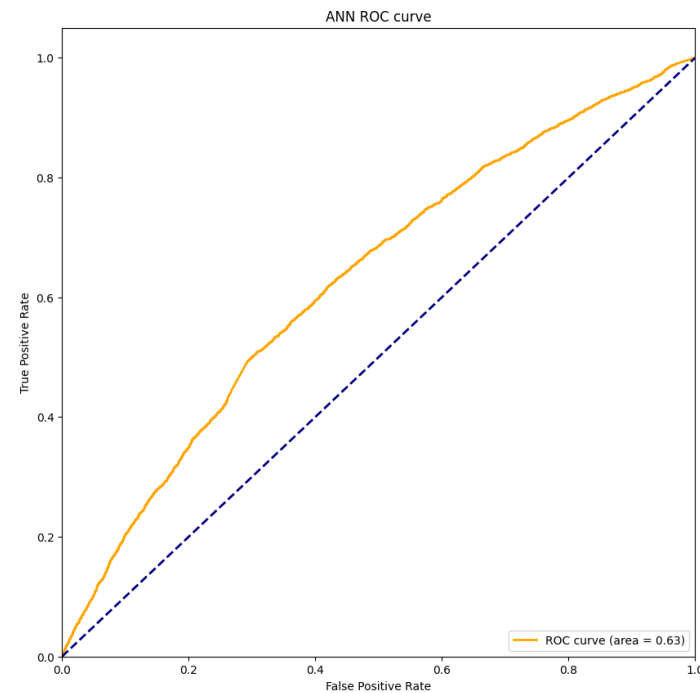
# Model(DL)

## 7. ANN

	precision	recall	f1-score	support
0	0.93	0.90	0.92	33619
1	0.15	0.20	0.17	2938
accuracy			0.85	36557
macro avg	0.54	0.55	0.54	36557
weighted avg	0.87	0.85	0.86	36557

Accuracy on Training set :0.923

Accuracy on Test set :0.846



# Result

- **Accuracy & Recall**

- Recall(재현율) 지표는 모델이 양성인 것 중 양성으로 잘 맞춘 것에 대한 비율이므로 해당 지표가 가장 중요 (Recall: Test data기준 Class 1 중에 Class 1으로 잘 맞춘 비율 - 높을 수록 좋은 모델)
- Accuracy의 경우 Random Forest가 0.927로 가장 높게 측정되었으나 Recall값은 Logistic Regression이 0.6으로 가장 높게 측정됨
- 그러나 Logistic Regression 모델의 경우 accuracy가 0.64로 신뢰성이 높은 모델이라고 판단하기 어려움

- **Evaluation**

- EDA를 통해 살펴 봤듯이 대출 상환 이행 여부에 따라 각 속성의 특징이 뚜렷한 것이 아니라 전체적인 양상이 비슷하게 나타나 분류를 잘 해내지 못 하는 것으로 해석 가능
- 따라서 정확한 예측 모델을 만들기 위해서는 현재 변수 외 다른 변수들이 추가되어야 할 것으로 보이며, 그 외 다양한 방식으로 변수 선택·파생 변수 생성 및 데이터 전처리를 통해 성능을 향상 시켜야 할 것으로 보임



# Future Challenges

---

변수 간 상관계수 시각화

Bayesian Search 기법 적용을 통해 HPO 구하기

Random Forest에도 Grid Search 적용 후 비교

컬럼들 drop하지 않고 모델링 하는 등 다양한 시도

Tree 모델의 Feature importance 확인

Under Sampling 후 결과 비교



**THANK YOU**

