

# 신용카드 사용자 신용도 분류 모델의 해석

최종 보고서



이 름	박민규(201711906), 강윤지(201811304), 정현지(201812490), 피재희(201812639)
학 과	시스템경영안전공학부
전 공	산업경영공학전공, 기술데이터전공
지도교수	이지환 교수님
과 목 명	캡스톤 디자인
제 출 일	2022.06.27

# 목차

1. 서론	3
1.1 연구배경 및 목적	3
1.2 연구대상 및 방법	3
2. XAI 이론적 배경	3
2.1 Explainable AI(XAI)	3
2.2 XAI의 필요성	4
2.3 XAI의 기술 동향	4
3. 연구 및 분석	4
3.1 데이터 수집	4
3.2 EDA	5
3.3 데이터 전처리	5
1) 결측치	6
2) 중복 값 및 이상치	6
3) Feature Engineering	6
4) Scaling, Encoding	7
5) Over Sampling	7
3.4 모델생성 및 성능비교	7
3.5 모델성능향상 및 최종모델 선정	8
1) ID	8
2) Clustering	8
3.6 SHAP 분석 및 분석결과	9
1) SHAP Feature Importance	10
2) SHAP Summary Plot	11
3) SHAP Waterfall Plot	12
4. 결론	13
4.1 분석 의의	13
4.2 기대 효과	13
참고 문헌	14
부록	15

# 1. 서론

## 1.1 연구 배경 및 목적

맥킨지(McKinsey)의 조사에 따르면 금융 분야는 타 산업과 비교했을 때 미래 AI 서비스에 대한 수요 및 활용 정도가 가장 높은 분야인 것으로 나타난다. 글로벌 금융기업들은 고객 대응, 업무관리, 금융사기탐지, 실시간 모니터링 및 리스크 관리 등 다양한 분야에서 인공지능을 활용하고 있으며, 국내에서도 그동안 일부 업무에만 제한적으로 도입되었던 인공지능 서비스의 영역이 점차 확장되고 있다. 금융 분야에서 인공지능 솔루션이 가장 유용하고 효과적인 영역은 챗봇과 24 시간 사용자 지원, 거래 분석 및 대출 등의 고객 서비스로 나타났다. 그중 신용 평가는 통계적 방법에 기초하고 많은 양의 정보를 고려한 수학적 모델이다. AI 는 이 작업에 빠르고 효율적으로 대처하는데 도움이 된다. 따라서 신용도를 판단하기 위한 AI 의 사용이 크게 증가할 가능성이 높다고 전망된다.

그러나 AI 의 산업적 활용을 위해서는 AI 를 통해 특정 서비스를 구현하는 것은 물론이고, AI 가 행한 의사결정에 대한 서비스 제공자의 해석 및 설명이 병행되어야 할 것이다. 이제까지의 대부분의 AI 모델은 작동원리를 드러내지 않는 블랙박스 모델이다. 이러한 AI 모델이 실제 금융 서비스에 사용된다면 결론에 대해 고객이 의문을 제시하였을 때, 그저 ‘인공지능이 내린 결과’ 라는 답 밖에 내놓지 못할 것이다. 이러한 문제를 해결하기 위해 모델이 출력한 결과를 얼마나 신뢰할 수 있을지, 근거는 무엇인지 찾아낼 수 있는 설명 가능한 인공지능(XAI)의 활용이 대두되고 있다.

본 연구의 목적은 금융 서비스 분야에 도입할 수 있는 하나의 사례로 신용카드 사용자의 신용도를 분류하는 AI 모델을 만들고, XAI 를 통해 모델의 결과를 해석하는데 있다. 이를 통해 고객의 신용도 평가를 보다 합리적으로 수행하고, 그에 대한 설명력을 확보하여 금융 서비스 혁신에 기여하는 것이 최종 목표이다.

## 1.2 연구 대상 및 방법

신용카드 사용자 데이터를 활용하여 사용자의 신용카드 대금 연체 정도를 예측하여 신용도를 분류하는 모델을 만든다. XAI 분석을 통해 모델의 결과를 해석하고 신뢰성을 확보하고자 한다.

연구는 데이터 수집 및 EDA, 분류 예측 모델 생성, 모델 성능 평가, XAI 를 통한 모델 해석 순으로 진행한다. 분류 예측 모델의 생성을 위해 Logistic, Naïve Bayes, KNN, Decision Tree, Support vector machine, XGBoost, LightGBM, Catboost 총 8 개의 알고리즘을 사용한다. 이중 가장 성능이 좋은 알고리즘을 선정하여 XAI 분석을 실시하며 분석 방법으로는 SHAP 을 사용한다. SHAP 란 하나의 feature 에 대한 중요도를 알기 위해서, 여러 feature 들의 조합을 구성하고, 해당 feature 의 유무에 따른 평균적인 변화를 통해 값을 계산하는 방법이다.

# 2. XAI 이론적 배경

## 2.1 Explainable AI(XAI)

Explainable AI, 즉 XAI 란 AI 가 도출한 결과와 출력을 인간인 사용자가 이해하고 이를 신뢰할 수 있도록 해주는 기술을 의미한다. XAI 의 핵심은 해석 가능성이다. 해석 가능성이란 왜 해당 모델을 신뢰해야 하는지, 모델이 왜 그런 결정을 했는지, 어떤 결과가 예상되는지 판단하는 과정이다. XAI 는 대리 분석, 부분 의존성 플롯, 유사도 분석, 피쳐 중요도 등의 기법으로 데이터와 모델을 설명한다. XAI 에는 다양한 분석 기법이 있는데 우리는 그중 SHAP 을 이용해 알고리즘을 설명해보고자 한다. SHAP 이란 Shapley Value 를 근간으로 하는 XAI 이며 자세한 설명은 SHAP 분석 부분에서 더 자세히 설명하도록 한다.

## 2.2 XAI의 필요성

최근 인공지능에 대한 관심이 높아지고 그에 따라 발전도 가속화되면서 학술적 연구단계에만 그치지 않고 실생활이나 산업현장에서 적용이 가능해질 정도의 수준에 다다랐다. 따라서 인공지능의 작동원리, 판단 근거를 분석하는 연구에 대한 중요성이 대두되고 있다. 지금까지 인공지능은 빅데이터를 기반으로 한 기계학습 기술의 발달로 인지 및 예측의 정확도는 발전하였으나, 블랙박스 모델이라는 점에서 인공지능 모델이 도출한 최종 결과의 근거, 도출 과정의 타당성을 논리적으로 설명할 수 없다는 한계점이 있다. 이로 인해 신뢰를 기반으로 하는 시스템에 적용하기에 많은 어려움이 따른다. 유럽연합은 2018년부터 General Data Protection Regulation에 의거하여 인공지능에 의해 자동으로 결정된 사안에 대해서는 회사에서 설명을 제공하도록 강제하고 있고, 미국에서는 회사가 내린 신용카드 발급, 주택담보대출 등의 주요 금융 결정에 대해서 이유를 제시하도록 법적으로 강제하고 있다. 이러한 필요성에 따라 사용자가 인공지능 시스템의 전반적인 작동원리를 이해하도록 도와주고, 시스템이 내린 의사결정에 대한 설명을 제공하는 설명 가능한 인공지능(eXplainable AI, XAI) 기술이 주요 연구 분야로 주목받고 있다.

XAI를 통해 해석한 결과를 바탕으로 시스템의 성능을 저하하는 요인을 발견해 시스템의 성능 향상을 기대할 수 있고, 빅데이터만으로는 알 수 없었던 새로운 법칙이나 전략에 대한 발견으로 통찰을 얻을 수 있다. 또한 AI 시스템의 사용 결과로 인한 분쟁 발생 시 원인 파악이 가능해진다는 점에서 법적 판단의 근거로써 활용될 수 있다.

## 2.3 XAI의 기술 동향

인공지능의 활용 범위가 넓어지고 그에 따라 XAI에 대한 연구도 세계적으로 활발히 진행되고 있다. XAI의 개발 현황을 한국특허전략개발원의 국가별 특허 동향으로 알아보았다. 현재 Microsoft, Google, IBM, Facebook 등 미국의 IT 기업들에서 확연히 높은 특허출원 활동을 보이고 있고, 중국에서도 바이두나 베이징대학을 중심으로 많은 개발이 이루어지는 것을 알 수 있었다. 그러나 우리나라의 경우에는 ETRI(한국전자통신연구원)와 삼성을 제외하면 XAI와 관련된 기술의 개발 투자가 많이 이루어지고 있지 않은 실정이다.

# 3. 연구 및 분석

## 3.1 데이터 수집

신용카드 사용자 신용도 분류 모델의 해석 프로젝트에서 사용할 데이터는 데이터콘에서 개최한 ‘신용카드 사용자 연체 예측 AI 경진대회’의 신용카드 사용자들의 개인 신상정보 데이터이다. 대회 주최측에 활용 신청을 허가 받은 후 해당 파일을 연구 및 분석에 활용하였다. 데이터는 총 26457개의 row와 20개의 column이며, column은 3개의 float형, 8개의 int형, 9개의 object형 column으로 구성되어 있다. 각각 column에 대한 내용은 아래의 표로 설명한다.

변수종류	변수명	변수설명	변수형태	변수종류	변수명	변수설명	변수형태
종속변수	credit	신용도	범주형		house_type	생활방식	범주형
	index	-	-		DAYS_BIRTH	출생일	수치형
	gender	성별	범주형		DAYS_EMPLOYED	업무 시작일	수치형
	car	차량 소유 여부	범주형		FLAG_MOBIL	핸드폰 소유 여부	범주형
	reality	부동산 소유 여부	범주형		work_phone	업무용 전화 소유 여부	범주형
	child_num	자녀 수	수치형		phone	전화 소유 여부	범주형
	income_total	연간 소득	수치형		email	이메일 소유 여부	범주형
	income_type	소득 분류	범주형		occyp_type	직업 유형	범주형
	edu_type	교육수준	범주형		family_size	가족 규모	수치형
	family_type	결혼 여부	범주형		begin_month	신용카드 발급 월	수치형

Figure 1. 데이터 변수 설명

변수명	구분	변수명	세부설명
income_type	'Commercial associate', 'Working', 'State servant', 'Pensioner', 'Student'	DAYS_BIRTH	데이터 수집 당시 '0'부터 역으로 세어 -1은 데이터 수집일 하루 전에 태어났음을 의미한다.
edu_type	'Higher education', 'Secondary / secondary special', 'Incomplete higher', 'Lower secondary', 'Academic degree'	DAYS_EMPLOYED	데이터 수집 당시 '0'부터 역으로 세어 -1은 데이터 수집일 하루 전부터 일을 시작함을 의미한다. 따라서 양수 값은 고용되지 않은 상태를 의미한다.
family_type	'Married', 'Civil marriage', 'Separated', 'Single / not married', 'Widow'	begin_month	데이터 수집 당시 '0'부터 역으로 세어 -1은 데이터 수집일 한 달 전에 신용카드를 발급함을 의미한다.
house_type	'Municipal apartment', 'House / apartment', 'With parents', 'Co-op apartment', 'Rented apartment', 'Office apartment'	credit	사용자의 신용카드 대금 연체를 기준으로 한 신용도. 낮을수록 높은 신용의 사용자를 의미한다.

Figure 2. 데이터 세부 설명

## 3.2 EDA

데이터의 결측치를 파악해본 결과 'occyp\_type(직업유형)' column 에서 26457 개의 데이터 중 8171 개의 데이터가 결측 돼 있는 것을 확인할 수 있다. 또한 수치형 데이터의 통계적 특성을 파악해본 결과 자녀가 최대 19 명이 있는 집이 있으며, 'FLAG\_MOBIL' 의 평균이 1 인걸로 보아, 모든 고객이 휴대폰을 소지하고 있다는 것을 알 수 있다. 또한 'DAYS\_EMPLOYED' 의 데이터는 역으로 세기 때문에 음수가 되어야 하지만, 최댓값이 양수인 365243.00 인 걸로 보아 전처리라 필요했다

	index	child_num	income_total	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	family_size	begin_month
count	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00	26457.00
mean	13228.00	0.43	187306.52	-15958.05	59068.75	1.00	0.22	0.29	0.09	2.20	-26.12
std	7637.62	0.75	101878.37	4201.59	137475.43	0.00	0.42	0.46	0.29	0.92	16.56
min	0.00	0.00	27000.00	-25152.00	-15713.00	1.00	0.00	0.00	0.00	1.00	-60.00
25%	6614.00	0.00	121500.00	-19431.00	-3153.00	1.00	0.00	0.00	0.00	2.00	-39.00
50%	13228.00	0.00	157500.00	-15547.00	-1539.00	1.00	0.00	0.00	0.00	2.00	-24.00
75%	19842.00	1.00	225000.00	-12446.00	-407.00	1.00	0.00	1.00	0.00	3.00	-12.00
max	26456.00	19.00	1575000.00	-7705.00	365243.00	1.00	1.00	1.00	1.00	20.00	0.00

Figure 3. 수치형 데이터의 통계적 특성

범주형 column( 'income\_type', 'edu\_type', 'family\_type', 'house\_type' )별로 고유 값 개수를 파악해 데이터의 분포를 확인해보았는데, 4 개 column 모두 데이터 불균형이 발견되었다. 데이터 불균형이란 특정 값에 대한 데이터가 매우 높은 빈도로 등장하는 것을 말한다. 데이터 불균형이 있으면 특정 클래스에 overfitting 되므로, 데이터 불균형을 처리하기 위해 oversampling 과정이 필요함을 EDA 를 통해 알 수 있었다.

데이터를 Class 0,1,2 로 분리해 신용등급에 따른 범주형 변수의 EDA 를 진행했다. 신용 등급 비율을 따져봤을 때 Class 2 가 제일 높았으나 성별 차이, 차량 소유 차이, 부동산 소유 차이 등 신용 등급에 따른 분포 차이는 거의 존재하지 않았고 양상은 동일했다. 소득 분류에서 신용 등급에 따른 차이가 있었는데, Class 0 에서는 'Student' 가 존재하지 않았으나, Class 1 과 Class 2 에서는 'Student' 데이터가 조금씩 존재했다. 학생의 신분 특성 상 높은 신용을 가지고 있기는 어렵기 때문에 다음과 같은 결과가 나왔다고 생각된다. 그러나 학생 데이터 수가 7 명밖에 안되기 때문에 유의미한 차이가 있다고는 보기 어려웠다.

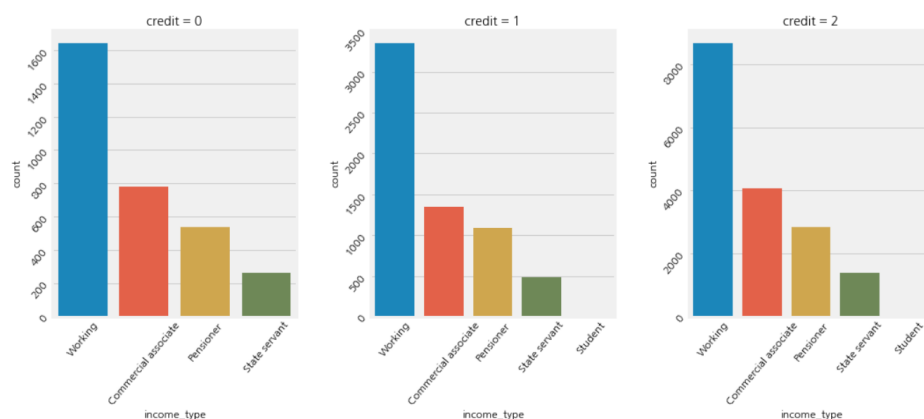


Figure 4. 소득분류에 따른 신용등급

신용등급에 따른 수치형 변수의 EDA 를 진행했다. 신용 등급에 따른 Class 별 분포 차이 또한 거의 비슷했다. 그러나 전체적으로 20~30 대 연령대 비율이 낮았고 가족 수의 차이로 봤을 때 Class 2 에서 Class 0 과 1 에 비해 적은 가족 수 분포를 가지고 있음을 알 수 있었다. 또한 카드 발급 기간에 따라 Class 2 와 다른 Class 에서 확연히 차이점이 있었는데, Class 2 에서 카드를 발급한 기간이 긴 고객들이 훨씬 많았다.

### 3.3 데이터 전처리

데이터를 모델에 적합시키기에 앞서 train, test 모두 다음과 같은 방법으로 전 처리를 수행했다.

#### 1) 결측치

우선, EDA 과정에서 ‘occyp\_type’ column 에서 총 26457 개의 데이터 중 8171 개의 결측치에 대해 분석한 결과, 결측치임에도 소득이 0 인 경우가 없음을 확인했고 해당 결측치들 역시 의미 있는 데이터라고 판단하여, 결측치를 제거하지 않고 unknown 으로 대체하였다.

Occpy\_type = NaN

Pensioner	4440
Working	2312
Commercial Associate	1026
State servant	392
Student	1

Name : income\_type, dtype : int64

Figure 5. 결측치

#### 2) 중복 값 및 이상치

데이터를 ‘Index’ 만 제외, ‘Index’ 와 ‘begin\_month’ 제외, ‘Index’ , ‘begin\_month’ , ‘Credit’ 제외 등 다양한 방법으로 쿼리를 진행해본 결과 중복 값이 다수 존재한 것으로 파악되어 ‘Index’ 를 제외하고 모든 값이 중복인 3155 개의 데이터를 제거하였다. 또한 ‘family\_size’ 가 7 보다 큰 값들은 이상치로 간주하여 제거했으며, ‘FLAG\_MOBIL’ column 은 모든 값이 1 로 동일하여 제거하였다. ‘DAYS\_EMPLOYED’ 값 중 0 보다 큰 값들은 현재 무직자로 판단하여 0 으로 처리하였다.

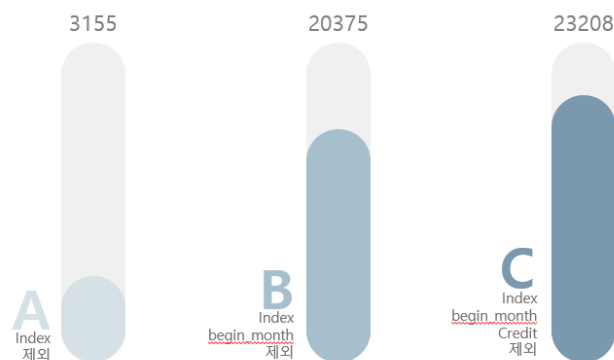


Figure 6. 중복 값

#### 3) Feature Engineering

EDA 과정을 통해 신용등급 별로 데이터의 분포에 큰 차이가 존재하지 않음을 파악했다. 따라서 최대한 다양한 특징을 보일 수 있도록 수치형 변수를 활용하여 총 12 개의 파생 변수를 생성한 후 파생 변수와 높은 상관성을 보이는 ‘child\_num’ , ‘DAYS\_BIRTH’ , ‘DAYS\_EMPLOYED’ 는 제거하였다.(부록 Figure 16 참조)

#### 4) Scaling, Encoding

총 15 개의 수치형 변수 중 ‘income\_total’ 은 단위가 다른 변수에 비해 월등히 크므로 Log Scaling 을, 나머지 14 개의 수치형 변수에 대해서는 Standard Scaling 을 수행했다. 또한 총 11 개의 범주형 변수들에 대해서 ‘One-hot Encoding’ 을 적용했다.

#### 5) Over Sampling

앞서 EDA 를 통해 ‘Credit’ 이 Class 별로 지극히 imbalanced 하기에 모델이 편향된 예측을 하지 않도록 오버 샘플링을 수행할 필요가 있었다. 다양한 오버 샘플링 기법 중 SMOTE-NC 를 활용하였다. SMOTE-NC 는 수치형 변수와 범주형 변수가 함께 존재하는 경우에 활용하는 기법이며, 유클리디안 거리를 계산하여 소수 클래스에서 각각의 샘플들의 KNN 을 찾고 그 이웃들 사이에 선을 그어 무작위 데이터를 생성하여 데이터의 불균형을 해소하는 기법이다. 이를 통해 기존의 {Class 2 : 11028, Class 1 : 4171, Class 0 : 2171} 개수에서 {Class 2 : 11028, Class 1 : 11028, Class 0 : 11028} 로 데이터를 늘려 불균형을 해소하였다.

### 3.4 모델 생성 및 성능 비교

전처리를 수행한 데이터를 토대로 가장 성능이 좋은 신용등급 분류 모델을 선정하기 위해 다양한 분류 모델을 생성하였다. 생성한 모델은 ‘Logistic Regression’, ‘Naïve Bayes’, ‘KNN’, ‘Decision Tree’, ‘Support Vector Machine’, ‘XGBoost’, ‘LightGBM’, ‘Catboost’ 으로 총 8 가지이다.

모든 모델에 대하여 n\_est = 2000, seed = 42 로 파라미터를 적용하였으며, 교차 검증은 StratifiedKFold 를 활용하였다. Fold 수는 5 부터 17 까지 모두 수행하여 본 결과 5 가 최적이라고 판단하여 선택하였다. Stratified Fold 는 Target 이 고르게 분포될 수 있도록 fold 를 나누어 데이터 셋이 불균형 할 때 유용한 기법이다.

평가는 분류 모델의 평가지표로 활용하는 Accuracy Score, F1-Score, Log Loss, ROC\_AUC Score, Confusion Matrix, Classification report 를 활용하여 모델의 성능을 측정하여 비교하였으며, 평가지표 별 8 개 모델의 성능은 다음과 같다.

Algorithm	Accuracy score	Algorithm	F1 score	Algorithm	Logloss	Algorithm	Roc auc score
<b>LightGBM</b>	<b>0.809062</b>	<b>LightGBM</b>	<b>0.808793</b>	<b>LightGBM</b>	<b>0.510125</b>	<b>LightGBM</b>	<b>0.921998</b>
Catboost	0.801808	Catboost	0.800959	XGBoost	0.542047	CatBoost	0.916272
XGBoost	0.793586	SVM	0.797224	Catboost	0.542547	XGBoost	0.912976
KNN	0.714031	XGBoost	0.694721	SVM	0.820924	KNN	0.885250
Decision Tree	0.657690	KNN	0.694721	Logistic	1.051366	SVM	0.876024
Logistic	0.455205	Decision Tree	0.657597	KNN	2.593647	Decision Tree	0.745790
SVM	0.422138	Logistic	0.450663	Naive Bayes	9.199845	Logistic	0.640787
Naive Bayes	0.384567	Naive Bayes	0.289933	Decision Tree	11.653466	Naive Bayes	0.595335

Figure 7. 평가지표별 모델 성능

LightGBM, XGBoost, Catboost 세가지 모델이 각 평가지표에서 근소한 차이로 상위를 차지함을 볼 수 있다. 따라서 해당 세가지 모델의 성능 보완 이후 재평가를 실시한다.



### 3.5 모델 성능 향상 및 최종 모델 선정

#### 1) ID

기존에 생성한 파생변수들은 수치형 변수만을 활용한 파생 변수였으나 모델이 특성을 더 잘 구분 지을 수 있도록 범주형 변수인 'gender', 'DAYS\_BIRTH', 'income\_total', 'edu\_type', 'occyp\_type'를 더해서 고유한 변수의 역할을 할 수 있는 'ID' 변수를 생성하였다. EDA 과정에서 한 사람이 여러 개의 카드를 만들 가능성이 있음을 파악했기에 이를 고려해 'begin\_month'는 활용하지 않았다. Train 데이터 셋에 대한 ID 변수의 고윳값은 7558 개, Test 데이터 셋에 대한 ID 변수의 고윳값은 4749 개이다.(부록 Figure 17,18 참조)

#### 2) Clustering

추가로 모델이 보다 Target 을 결정 짓기 용이하게끔 클러스터링을 적용하였다. 클러스터링 기법으로는 K-prototype 을 이용했다. K-Prototype 알고리즘은 연속형 자료는 유클리디안 거리를 구하고 범주형 자료는 비유사도를 구한 후, 비유사도에 가중치를 부여하여 둘을 합한 것을 거리라고 정의하는 방식이다. 따라서 여러 형태의 변수가 포함된 우리의 데이터의 경우 이를 활용하는 것이 적합하다고 판단하였다. 적절한 클러스터 개수를 구하기 위해서 Elbow method 를 사용했고, 그 결과는 다음과 같다.

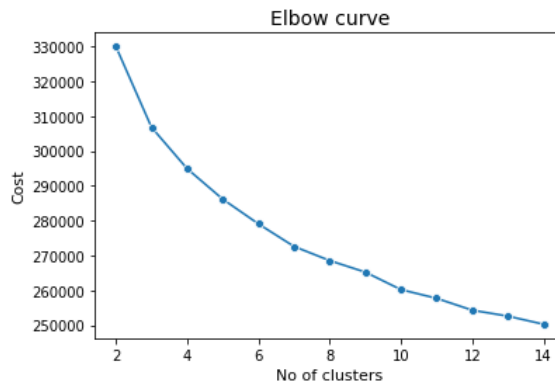


Figure 8. Elbow curve

그래프 상 클러스터의 개수가 3에서 기울기가 가장 급격하게 변화하는 것으로 보이나 실제로 클러스터 개수를 2~5까지 모두 적용하여 성능을 비교해본 결과 5를 적용했을 때의 logloss 값이 가장 낮은 것을 확인하여 최적 클러스터 개수는 5로 선정하였다.

ID 변수의 생성과 클러스터링을 적용한 후 LightGBM, XGBoost, Catboost에 각각 적합을 시켜본 결과 Catboost를 최종 모델로 선정하였다. 그 이유는 다음과 같다.

우선 XGBoost의 경우 범주형 변수를 알고리즘이 자체적으로 처리할 수 없기 때문에 범주형 변수에 대해 전처리과정에서 인코딩을 수행해야 한다. 또한 항상 균형을 이루며 2진 분할을 하는 Level-wise(균형 트리 분할) 방식을 채택하여 시간과 메모리가 많이 소요된다. 추가로 생성한 변수인 ID의 고윳값은 총 7558개로 이를 모두 One-hot Encoding을 수행할 경우 변수의 수가 급격하게 증가하는 문제가 있다.

LightGBM 은 알고리즘이 자체적으로 범주형 변수를 처리하는 것이 가능하며, Leaf-wise(리프 중심 트리 분할) 방식을 채택하여 학습속도가 빠르다는 장점이 있다. 따라서 ID 변수에 대해 인코딩을 수행하지 않아도 된다. 그러나 문제는 ID의 고윳값 중 train 데이터에만 존재하고, test 데이터에는 존재하지 않는 값들이 있는데, 해당 알고리즘의 경우 그 값을 자동으로 처리하지 못해서 직접 처리해주는 데에 어려움이 있었다.

Catboost 는 범주형 변수의 인덱스 혹은 변수 명만 지정해주면 자체적으로 처리가 가능하며, 또한 order boosting 방식을 통해 train 데이터와 test 데이터의 분포 차이로 인해 발생하는 문제를 제거할 수 있다. 그리고 트리를 분할할 때에 oblivious decision tree(망각 결정 트리) 방법론이 사용되어 과적합을 막을 수 있다.



XGBoost 와 LightGBM 의 경우 데이터가 1 만개 보다 적으면 과적합이 일어날 가능성이 높다. Catboost 는 범주형 변수를 처리할 때 feature combination 방법을 사용하여 범주형 변수 조합으로 새로운 변수 조합을 만든다. 따라서 ID 변수의 활용이 가능했으며 학습 속도 역시 빨라 최종 모델로 선정하였으며, ID 와 cluster 변수의 생성 후 모델의 성능은 생성 전과 대비해 크게 증가하였다.

Valid Accuracy Score: 0.849444  
Valid F1 Score: 0.849446  
Valid Log Loss: **0.408876**

Valid ROC\_AUC\_Score: 0.950088

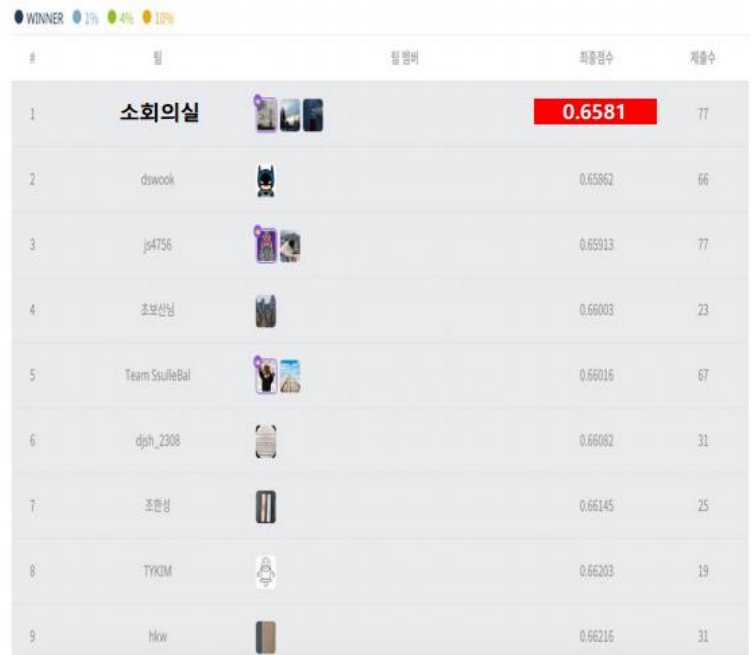
```
classification_report - valid data
              precision    recall  f1-score   support

    0.0         0.88        0.89        0.89        11028
    1.0         0.86        0.81        0.83        11028
    2.0         0.80        0.85        0.83        11028

 accuracy         0.85
 macro avg        0.85
weighted avg        0.85
```

```
Confusion_matrix - valid data
[[9806  476  746]
 [ 605 8886 1537]
 [ 671  946 9411]]
```

```
Confusion_matrix_Normailze - valid data
[[0.88919115 0.04316286 0.06764599]
 [0.05486036 0.80576714 0.13937251]
 [0.06084512 0.08578165 0.85337323]]
```



#	팀	팀 멤버	최종점수	제출수
1	소회의실		<b>0.6581</b>	77
2	dswook		0.65862	66
3	js4756		0.65913	77
4	조보산님		0.66003	23
5	Team SauleBel		0.66016	67
6	djsh_2308		0.66082	31
7	조한성		0.66145	25
8	TYKDM		0.66203	19
9	hkw		0.66216	31

Figure 9. 해당 대회 우승팀과의 모델 성능 비교

최종 모델의 성능은 Valid 데이터 셋에 대하여 Accuracy Score - 0.849, F1-Score - 0.849, Log Loss - 0.408, ROC\_AUC Score - 0.95 이며 정밀도와 재현율 모두 Class 별로 80% 이상의 높은 성능을 보이며, 해당 대회 우승팀의 LogLoss 에 비해 25%나 성능이 높음을 확인할 수 있다.

### 3.6 SHAP 분석 및 분석 결과

XAI 모델 중 하나인 SHAP(Shapley Additive exPlanations)는 Shapley value 를 기반으로 예측 값에 대하여 각 feature 가 미치는 기여도를 측정하여 예측에 대한 해석을 제공한다.

Shapley value 란, 게임이론을 바탕으로 하나의 특성 대한 중요도를 알기 위해 여러 특성들의 조합을 구성하고 해당 특성의 유무에 따른 평균적인 변화를 통해 얻어낸 값이 바로 Shapley value 이다. 즉, 특정 값이 있을 때와 없을 때의 값 차이를 가능한 모든 조합으로 구한 다음 평균값을 낸 것이 Shapley value 이다.

SHAP 는 Shapley value 를 기반으로 예측 값에 대하여 모델에 입력으로 들어가는 각 feature 가 미치는 기여도를 Shapley Value 로 계산하여 예측에 대한 해석을 제공한다. SHAP 방법에는 Kernal SHAP(LIME+SHAP), Deep SHAP(딥러닝 모델), TreeSHAP(tree 기반 모델)등이 있는데 우리는 Tree 기반 모델 중 하나인 Catboost 모델을 사용하기 때문에 TreeSHAP 를 사용하여 모델을 해석하였다.

## 1) SHAP Feature Importance

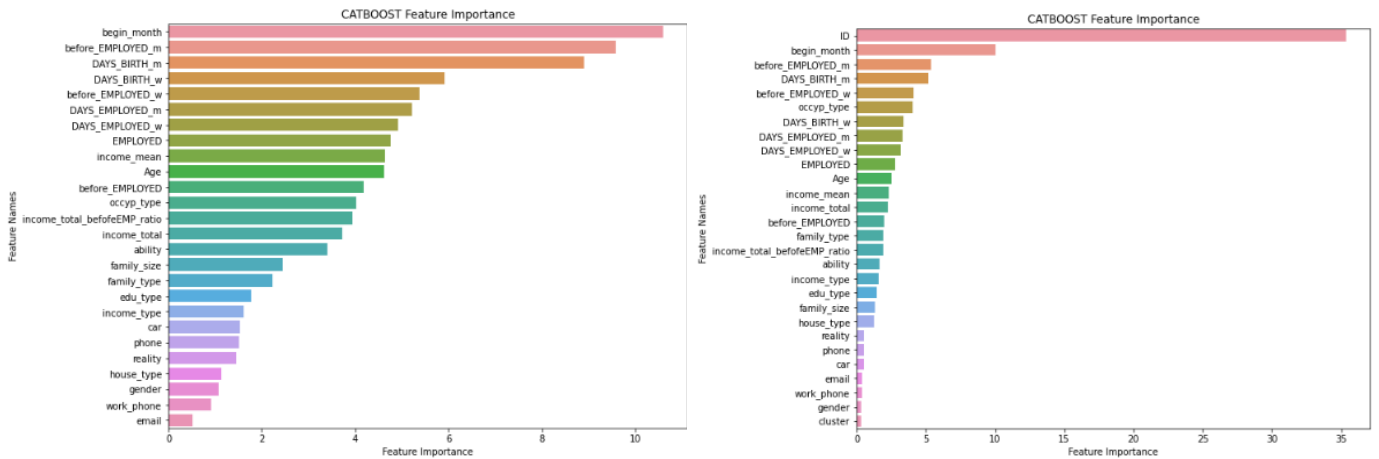


Figure 10. CatBoost Feature Importance

Figure 10은 Catboost의 자체적인 Feature Importance이다. 왼쪽 그래프는 ID, cluster 생성 전이며, 오른쪽 그래프는 생성 후이다. 이 그래프는 단편적으로 모델의 예측에 Feature Importance를 보여준다.

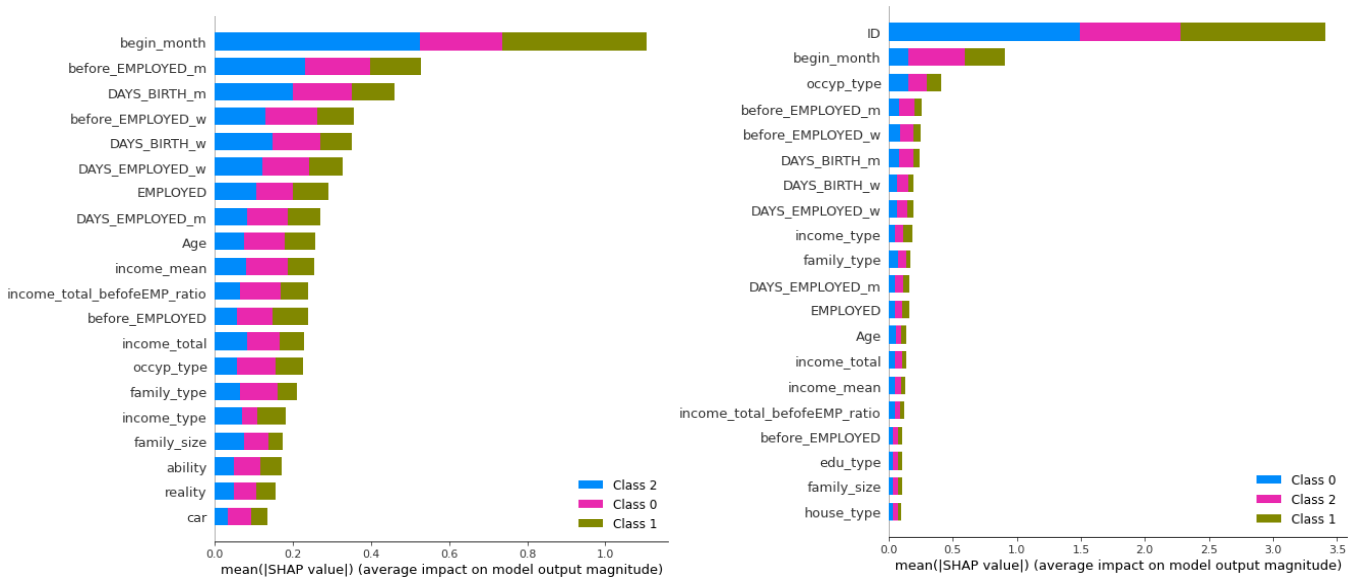


Figure 11. SHAP Feature Importance

Figure 11과 같이 SHAP 라이브러리에 내장된 Feature Importance 그래프는 feature가 Class에 미치는 impact가 쌓여서 만들어진 그래프이며, 각 Class별로 feature가 모델에 미치는 절대 영향도를 눈으로 파악할 수 있고 각 feature의 impact가 Class에 따라 전체적으로 어떻게 달라지는지 보여준다. 왼쪽 그래프는 ID, cluster 생성 전이며, 오른쪽 그래프는 생성 후이다.

Figure 11의 왼쪽 그래프 통해 알 수 있듯이 ID와 cluster 생성 전에는 Class 0,1,2 모두 'begin\_month'가 모델 예측에 제일 영향을 많이 끼치는 것을 알 수 있다. 영향을 가장 많이 주는 상위의 3개 feature인 'begin\_month', 'before\_EMPLOYED\_m', 'DAYS\_BIRTH\_m'을 제외하고는 각 클래스별로 영향을 주는 feature의 순위가 다른 것을 파악할 수 있다.

우리는 앞서 중복 데이터에 관한 이슈를 해결하기 위해 index를 제외하고 완전히 일치하는 중복 데이터는 삭제를 해주었다. 그럼에도 남아있던 중복 데이터를 구분 짓기 위해 주민등록번호와 같이 고유 식별력을 가진 데이터가 필요했고, 이를 위해 ID라는 컬럼을 생성해주었다. Figure 11의 오른쪽 그래프에서 확인할 수 있듯이 'ID'와 'cluster' 생성 후에 X축에 Shapely Value 값이 크게 증가하였고, Class 0,1,2에서 모두 파생변수 'ID'가 모델에 미치는 절대적인 영향도가 가장 큰 것을 확인할 수 있다.

## 1) SHAP Summary plot

Summary plot은 전체 특성들이 Shapley value 분포에 어떤 영향을 미치는지 시각화 한 것이다. X축은 SHAP 값의 수치, Y축은 독립변수, 색은 독립변수의 상대적인 크기를 나타낸다. 겹치는 점이 Y축 방향으로 나타남에 따라 독립변수에 관한 SHAP값의 분포를 확인할 수 있다. SHAP 값의 양수와 음수는 예측 값에 긍정적인 기여와 부정적인 기여를 각각 의미한다. 그래프의 색상이 빨간색이면 특성 값이 크음을 의미하고, 파란색이면 특성 값이 작음을 의미한다. 회색은 categorical value 값이므로, 특성 값이 작고 크음을 나타낼 수 없기 때문에 회색으로 나타난다. 신용카드 데이터에서는 'ID', 'occyp\_type', 'family\_type', 'income\_type', 'reality', 'edu\_type', 'house\_type' 은 categorical value이므로 회색으로 나타난다.

Summary Plot에서 각 feature의 순서는 예측에 미치는 영향력에 따라서 내림차순 정렬된다. Class 0,1,2의 Summary plot을 살펴보면 ID, cluster 생성 전에는 'begin\_month'가 정답을 판별하는데 가장 큰 기여를 했음을 확인할 수 있다.

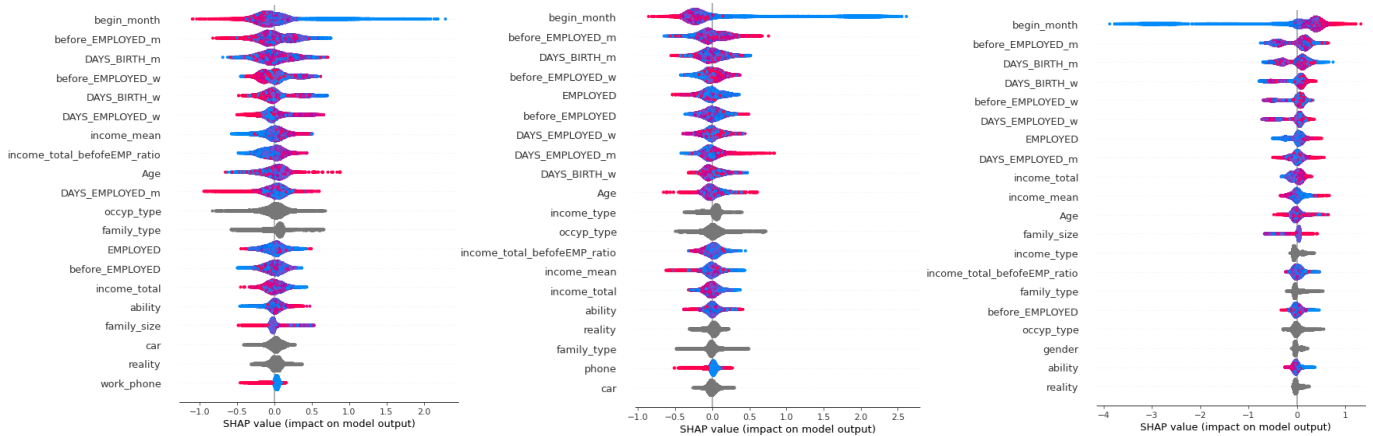


Figure 12. SHAP Summary Plot(ID, cluster 생성 전, 왼쪽부터 Class 0,1,2)

수치형 변수의 신용등급에 따른 카드 발급 기간 차이를 파악하는 EDA를 통해서 Class 2에는 대부분 'begin\_month' 값이 높은 사람들의 데이터가 분포했음을 확인할 수 있었다. 그에 비해서 Class 2보다 신용등급이 높은 Class 0과 Class 1은 'begin\_month' 값이 비교적 낮음을 알 수 있었다.

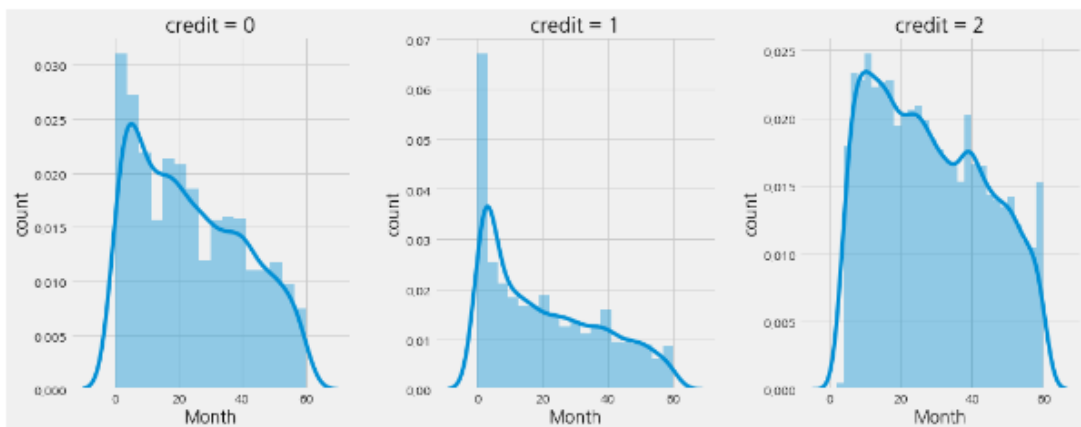


Figure 13. 신용등급에 따른 카드 발급 기간 차이 파악

Summary plot에서 Class 0과 1은 'begin\_month'의 특성 값이 작을수록 예측에 긍정적인 영향을 미치며, Class 2는 'begin\_month'의 특성 값이 클수록 모델의 예측에 긍정적인 영향을 미친다. 일반적으로는 카드를 발급하고 난 기간이 길수록 신용도가 높을 것이라고 생각이 되지만 예상과는 달리 이렇게 정반대의 결과가 나타나는 이유는 위의 그래프를 통해 확인했듯이 실제 데이터에서 'begin\_month'의 값이 높은 데이터가 Class 2에 많았고, 그래서 모델은 'begin\_month'가 비교적 높은 값을 Class 2로 분류를 했다는 것을 위의 Figure 12의 Summary plot을 통해 확인할 수 있다.

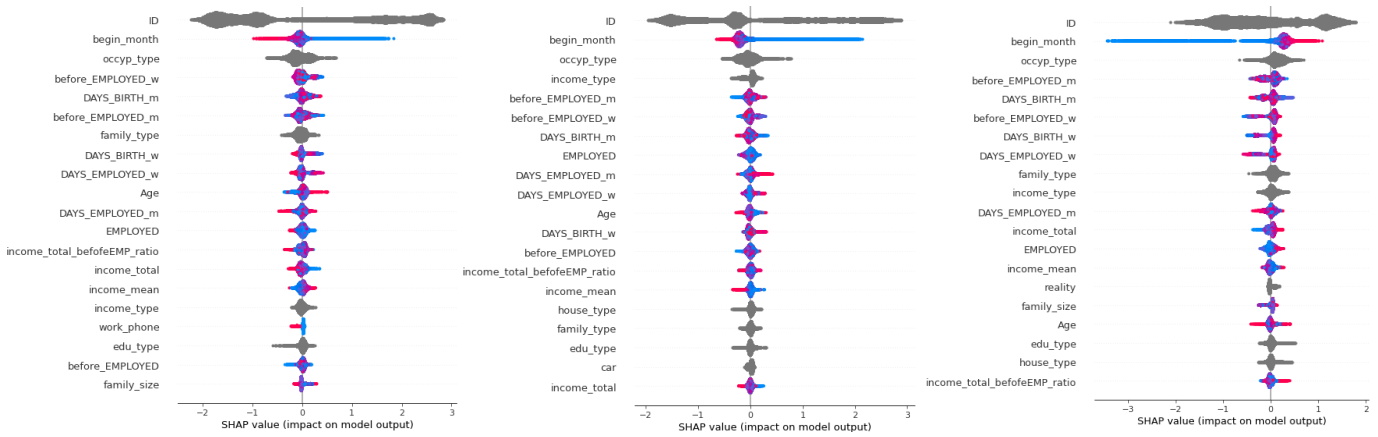


Figure 14. SHAP Summary Plot(ID, cluster 생성 후, 왼쪽부터 Class 0,1,2)

‘ID’ 는 ‘gender’ , ‘DAYS\_BIRTH’ , ‘income\_total’ , ‘income\_type’ , ‘edu\_type’ , ‘occyp\_type’ 값들을 더해서 만든 파생변수이므로 categorical 한 column 이다. ID, cluster 생성 후에는 ID 의 feature importance 가 제일 높은 것을 보았을 때 ‘ID’ 가 모델의 정확도 향상에 유의미한 효과가 있었음을 확인할 수 있다.

## 2) SHAP Waterfall Plots

Waterfall Plots는 특정 데이터가 예측된 근거를 절댓값 영향도만큼 내림차순 정렬해 아래로 펼친 그래프이다. 직관적으로 영향의 크기를 볼 수 있는 그래프이다. 각 feature들의 공헌도와 Shapley Value를 평면에 표현한다. Waterfall Plots를 활용해서 어떤 요인이 가장 정답을 예측하는데 기여를 했는지 데이터 별로 알 수 있다. 각각 정답에 해당하는 그래프는 예측 값에 긍정적 요인인 빨간색 Bar가 많다. 요인이 정답을 예측하는데 큰 영향을 준 만큼 Bar의 길이 또한 길어진다. Waterfall Plots를 통해 정답에 해당하는 Class와 그렇지 않은 Class의 Shapley Value를 비교하여 정답 별로 feature가 어떻게 영향을 미치는지 row별로 파악했다. ID, cluster 생성 전, 후 모두 정답에 해당하는 그래프는 긍정적 요인이 훨씬 많은 것을 확인할 수 있었다. (부록 Figure 19~36 참조)

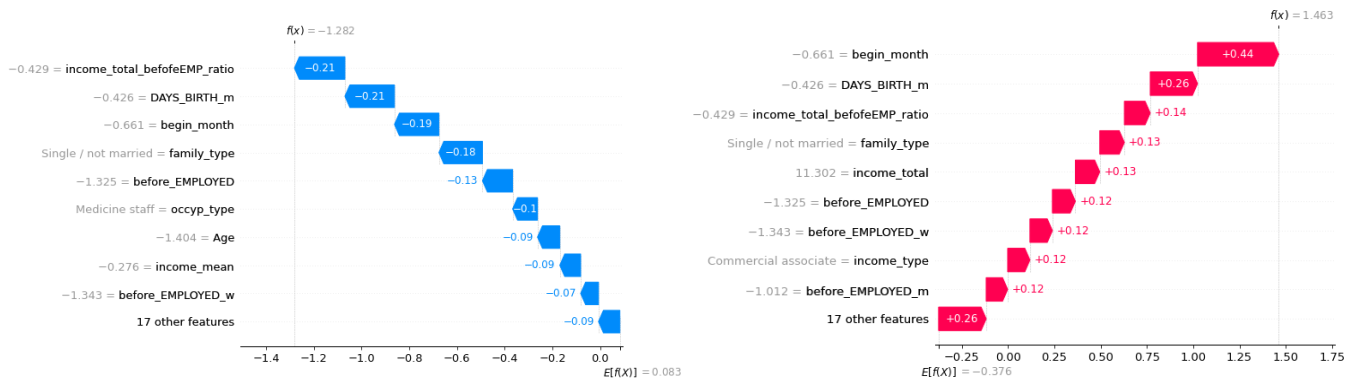


Figure 15. SHAP Waterfall Plot (왼쪽 : 오답 Class, 오른쪽 : 정답 Class)

## 4. 결론

### 4.1 분석 의의

지금까지 신용카드 이용 고객의 신용도를 예측하는 모델을 만들고 그 모델이 어느 정도의 설명력을 가지는지에 대한 분석 프로젝트를 모두 진행해보았다. 이번 분석을 통해 신용도를 예측하기 위해서는 단일 정보가 아닌 다양한 정보를 파악해야 함을 알 수 있었다. 신용도 측정을 위해서는 고객의 성별, 나이, 소득, 직업, 교육 수준으로 다양한 특성을 반영하여 분석할 필요가 있다. 이번 연구에서 구현한 모델은 대회에 참여한 상위권 그룹의 결과를 바탕으로 성능을 비교해 봤을 때 더 우수한 성능을 가지는 것으로 확인되었으며, 또한 XAI 분석을 활용하여 우리 팀이 구축한 모델의 설명력과 신뢰성을 확인할 수 있었다.

이번 분석에 활용한 데이터가 본래 대회용 데이터이다 보니 노이즈가 존재하며, 실제 은행 데이터의 용량과 큰 차이가 있어 실제 은행 고객의 데이터 특성을 완벽히 반영하지 못한 점에서 한계가 있었다. 더하여, 최근 인공지능의 발전은 자연어처리와 감성 기술 등과 같은 비정형 데이터를 통해 고객의 신용도를 평가할 수 있는 수준에 이르고 있다. 고객의 개인 정보 데이터를 활용한 인공지능 모델에서 더 나아가 인공지능으로 다양한 비정형 데이터를 분석하는 과정을 통해 또 다른 유의미한 인사이트를 얻을 수 있을 것이다.

### 4.2 기대 효과

우리가 도출해낸 일련의 과정들이 실제 금융업에 사용된다면, 고객에게는 자신의 신용도가 어떤 요인을 통해 도출되었는지 이해할 수 있는 객관적 근거를 마련하고 신용도에 문제가 있다면 문제를 파악할 수 있게 함으로써 그에 따른 해결방안을 생각할 수 있도록 한다. 은행에서는 자사 고객의 신용도를 분류하고 그 분류의 이유를 분석한 신뢰할 수 있는 결과를 토대로 기업 마케팅, 컨설팅, 금융상품 개발 등 다양한 측면에서 신뢰할 만한 인사이트를 제공할 것으로 기대된다. 뿐만 아니라, 의사결정 시간 및 자료 분석 비용의 단축의 효과를 얻을 수 있으며, 인력에 의해 발생할 수 있는 오류를 완화하고 고객 관리를 효율적으로 할 수 있을 것이다.

## 참고자료

1. 인공지능이 가져올 금융 서비스 혁신 전망, 과학기술정책연구원, 2020.12  
<https://www.dbpia.co.kr/pdf/pdfView.do?nodeId=NODE10502141>
2. 디지털 금융 혁명의 거대한 물결... AI가 은행의 높은 문턱 낮춘다, AI타임스, 2022.02  
<http://www.aitimes.com/news/articleView.html?idxno=142962>
3. 인공지능(Artificial Intelligence) 이슈와 국제 표준화 동향, 2021.04  
[https://spri.kr/posts/view/23194?code=industry\\_trend](https://spri.kr/posts/view/23194?code=industry_trend)
4. 설명가능한 인공지능  
<https://www.imgr.co.kr/post/xai>
5. 설명가능한 인공지능이란 무엇인가?  
<https://blogs.nvidia.co.kr/2021/07/27/what-is-explainable-ai/>
6. 설명가능한 인공지능에 집중하는 이유  
<http://www.aitimes.kr/news/articleView.html?idxno=14859>
7. 설명가능한 인공지능(eXplained AI, XAI)의 필요성과 연구동향, 2022  
<https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiOrteView.kci?sereArticleSearchBean.artiId=ART002834739>
8. 한국특허전략개발원  
<https://www.kista.re.kr>
9. SHAP(Shapely Additive exPlanations) 개념정리  
<https://velog.io/@sjinu/%EA%B0%9C%EB%85%90%EC%A0%95%EB%A6%ACSHAPShapley-Additive-exPlanations>
10. SHAP Documentation  
<https://shap.readthedocs.io/en/latest/index.html>
11. XGBoost, LightGBM, CatBoost 정리 및 비교  
<https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db>  
<https://statinknu.tistory.com/33>
12. XGBoost Documentation  
<https://xgboost.readthedocs.io/en/stable/>
13. LightGBM Documentation  
<https://lightgbm.readthedocs.io/en/latest/>
14. CatBoost Documentation  
[https://catboost.ai/en/docs/concepts/python-reference\\_catboostclassifier](https://catboost.ai/en/docs/concepts/python-reference_catboostclassifier)
15. CatBoost 특징  
<https://dailyheumsi.tistory.com/136>
16. LogLoss  
<https://velog.io/@skypodium/logloss-%EC%95%8C%EC%95%84%EB%B3%B4%EA%B8%B0>
17. Feature Engineering - Onehot Encoder  
<https://velog.io/@dlskawns/Machine-learning-%EB%B2%94%EC%A3%BC%ED%98%95-feature%EC%97%90-%EB%8C%80%ED%95%9C-%EC%B2%98%EB%A6%ACEncodersOneHotEncoder-OrdinalEncoder>
18. Cross validation(k-fold, stratifiedkfold)  
<https://continuous-development.tistory.com/166>
19. OverSampling(SMOTE, SMOTE-NC)  
[https://imbalanced-learn.org/stable/references/generated/imblearn.over\\_sampling.SMOTE.html](https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html)  
[https://imbalanced-learn.org/stable/references/generated/imblearn.over\\_sampling.SMOTENC.html](https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTENC.html)

## 부록

```

for df in [train, test]:
    # before_EMPLOYED: 고용되기 전까지의 일수
    df['before_EMPLOYED'] = df['DAYS_BIRTH'] - df['DAYS_EMPLOYED']
    df['income_total_beforeEMP_ratio'] = df['income_total'] / df['before_EMPLOYED']
    df['before_EMPLOYED_m'] = np.floor(df['before_EMPLOYED'] / 30) - ((np.floor(df['before_EMPLOYED'] / 30) / 12).astype(int) * 12)
    df['before_EMPLOYED_w'] = np.floor(df['before_EMPLOYED'] / 7) - ((np.floor(df['before_EMPLOYED'] / 7) / 4).astype(int) * 4)

    #DAYS_BIRTH 파생변수- Age(나이), 태어난 월, 태어난 주(출생연도의 n주차)
    df['Age'] = df['DAYS_BIRTH'] // 365
    df['DAYS_BIRTH_m'] = np.floor(df['DAYS_BIRTH'] / 30) - ((np.floor(df['DAYS_BIRTH'] / 30) / 12).astype(int) * 12)
    df['DAYS_BIRTH_w'] = np.floor(df['DAYS_BIRTH'] / 7) - ((np.floor(df['DAYS_BIRTH'] / 7) / 4).astype(int) * 4)

    #DAYS_EMPLOYED_m 파생변수- EMPLOYED(근속연수), DAYS_EMPLOYED_m(고용된 달), DAYS_EMPLOYED_w(고용된 주(고용연도의 n주차))
    df['EMPLOYED'] = df['DAYS_EMPLOYED'] // 365
    df['DAYS_EMPLOYED_m'] = np.floor(df['DAYS_EMPLOYED'] / 30) - ((np.floor(df['DAYS_EMPLOYED'] / 30) / 12).astype(int) * 12)
    df['DAYS_EMPLOYED_w'] = np.floor(df['DAYS_EMPLOYED'] / 7) - ((np.floor(df['DAYS_EMPLOYED'] / 7) / 4).astype(int) * 4)

    #ability: 소득/(살아온 일수+ 근무일수)
    df['ability'] = df['income_total'] / (df['DAYS_BIRTH'] + df['DAYS_EMPLOYED'])

    #income_mean: 소득/ 가족 수
    df['income_mean'] = df['income_total'] / df['family_size']

    #ID 생성: gender, DAYS_BIRTH, income_total, income_type, edu_type, occyp_type 값들을 더해서 고유한 사람을 파악(*한 사람이 여러 개 카드)
    df['ID'] = df['gender'].astype(str) + '_' + df['DAYS_BIRTH'].astype(str) + '_' + df['income_total'].astype(str) + '_' + df['income_ty

```

executed in 129ms, finished 16:17:39 2022-06-15

Figure 16. 수치형 파생 변수 생성 코드

train['ID'].value_counts()	
executed in 18ms, finished 16:17:39 2022-06-15	
F_12676_157500.0_State servant_Secondary / secondary special_Waiters/barmen staff	18
M_13790_562500.0_Working_Incomplete higher_unknown	16
F_15519_297000.0_Commercial associate_Secondary / secondary special_Laborers	15
M_16768_225000.0_Working_Higher education_Laborers	15
F_11126_67500.0_Working_Secondary / secondary special_Accountants	14
..	
F_22064_81000.0_Working_Secondary / secondary special_Cleaning staff	1
F_10935_67500.0_Working_Higher education_Sales staff	1
F_18920_112500.0_Commercial associate_Secondary / secondary special_High skill tech staff	1
F_12924_112500.0_Working_Higher education_Sales staff	1
F_24651_112500.0_Pensioner_Secondary / secondary special_unknown	1
Name: ID, Length: 7558, dtype: int64	

Figure 17. Train 데이터셋의 ID 고유값-7558개



test['ID'].value_counts()	
executed in 27ms, finished 20:13:06 2022-06-18	
F_15519_297000.0_Commercial associate_Secondary / secondary special_Laborers	15
F_19441_315000.0_Working_Secondary / secondary special_Managers	8
F_16391_202500.0_Working_Higher education_unknown	7
F_10512_216000.0_Working_Higher education_Core staff	7
F_18767_180000.0_Working_Secondary / secondary special_Cleaning staff	7
..	
F_21772_225000.0_Pensioner_Higher education_unknown	1
M_12167_112500.0_Working_Secondary / secondary special_Drivers	1
F_11675_148500.0_Working_Secondary / secondary special_Sales staff	1
M_20014_180000.0_Working_Secondary / secondary special_Security staff	1
F_23602_225000.0_Pensioner_Higher education_unknown	1
Name: ID, Length: 4749, dtype: int64	

Figure 18. Test 데이터셋의 ID 고유향-4749개

- ID, cluster 생성 전 -

Row 0 (정답 : Class 2)

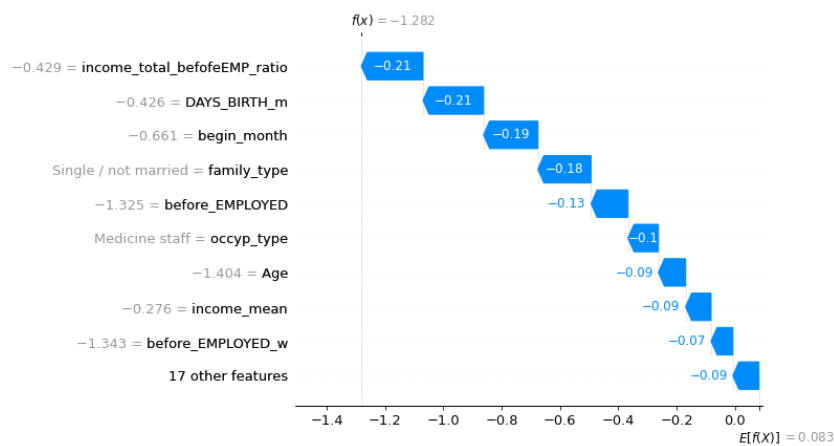


Figure 19. Class 0 Waterfall Plot

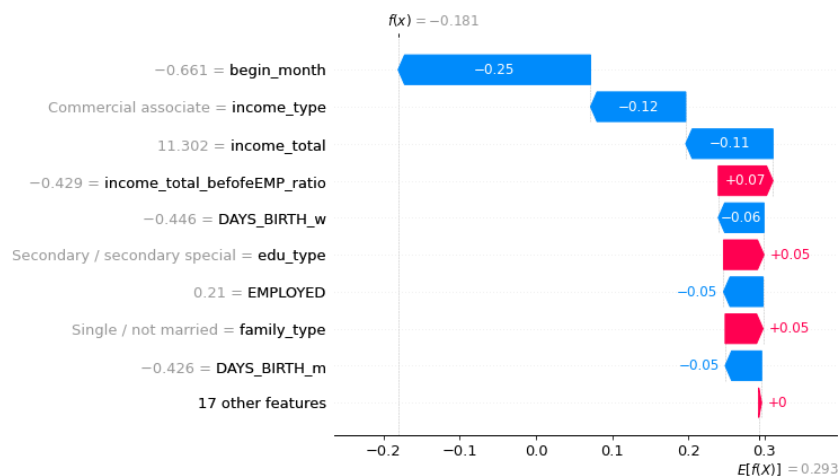


Figure 20. Class 1 Waterfall Plot

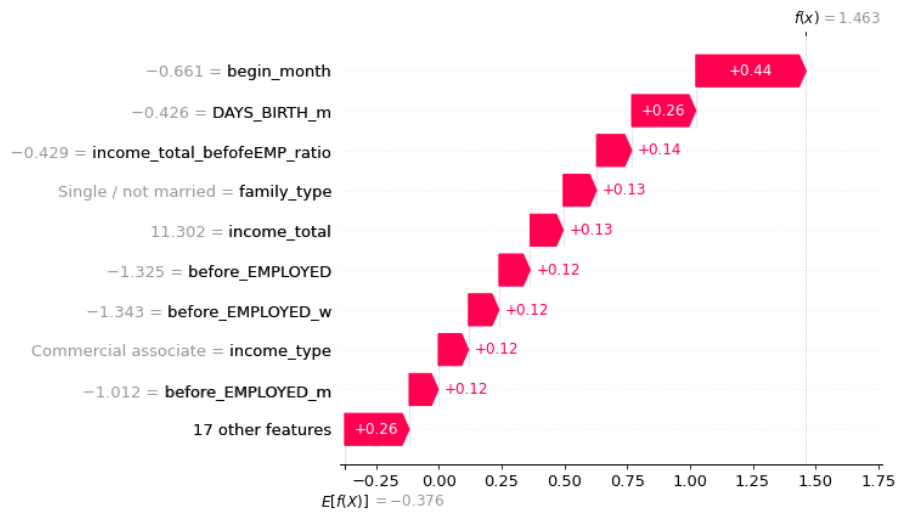


Figure 21. Class 2 Waterfall Plot

Row 16540(정답 : Class 0)

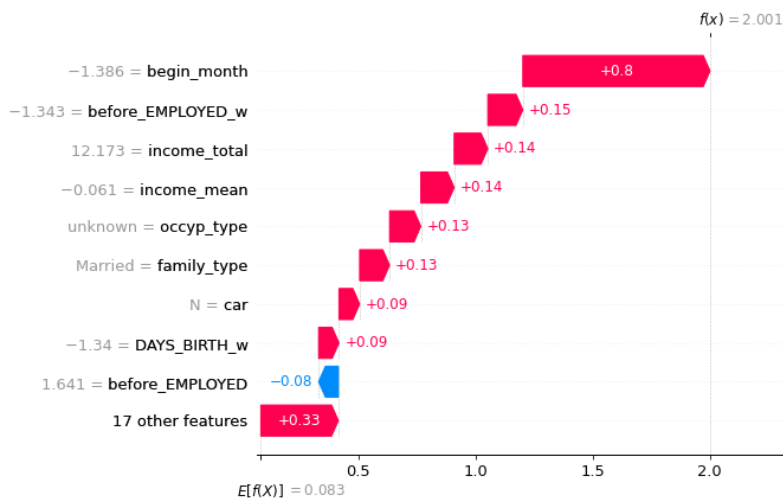


Figure 22. Class 0 Waterfall Plot

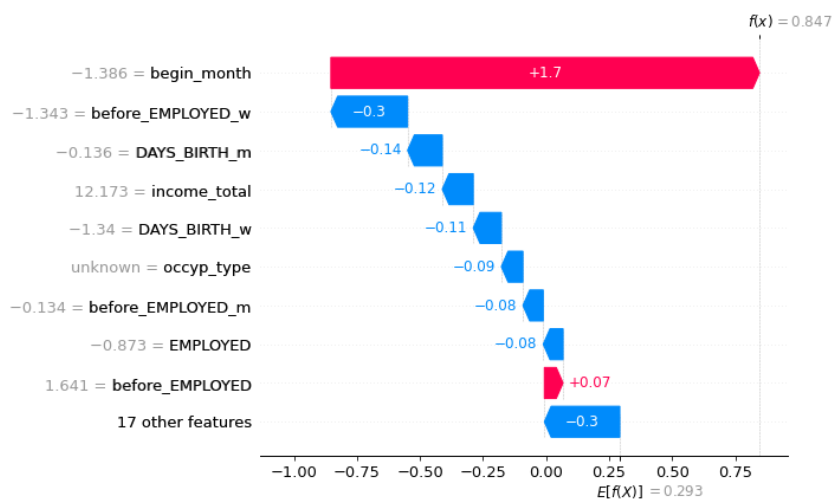


Figure 23. Class 1 Waterfall Plot

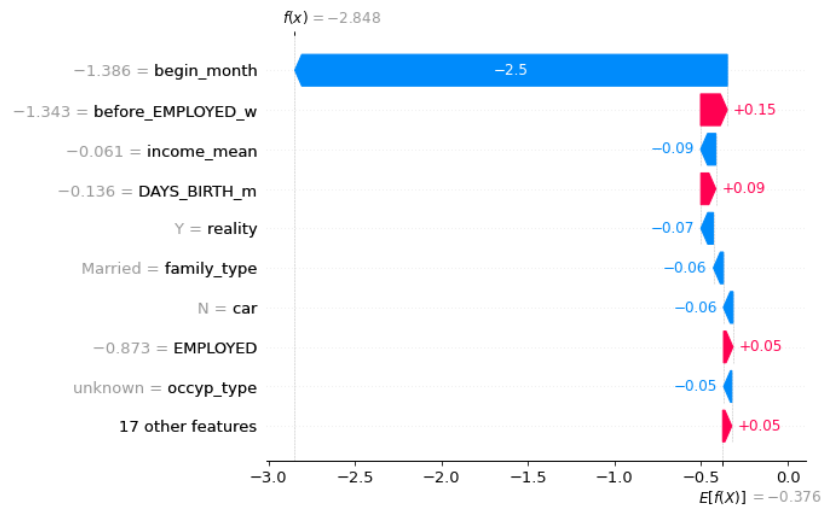


Figure 24. Class 2 Waterfall Plot

Row 33083(정답 : Class 1)

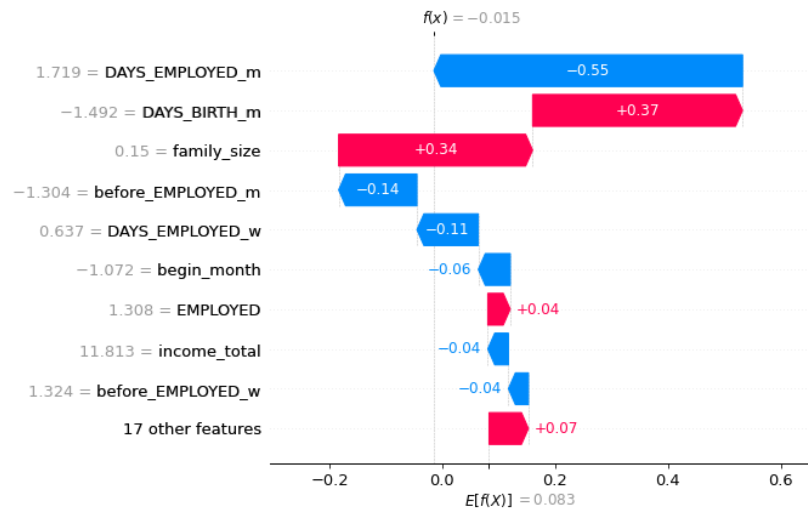


Figure 25. Class 0 Waterfall Plot

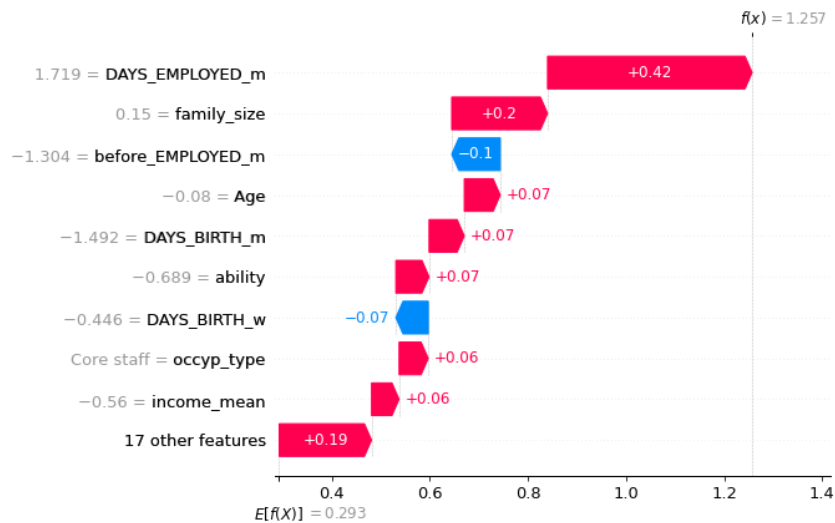


Figure 26. Class 1 Waterfall Plot

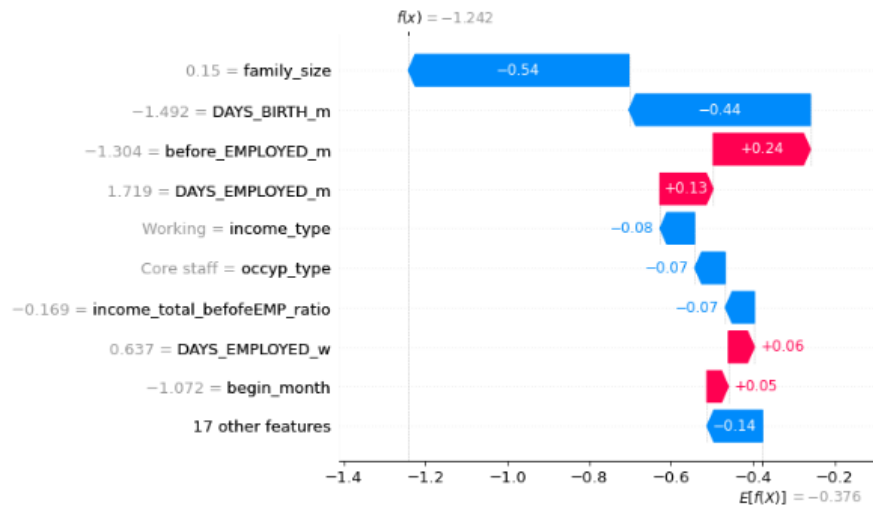


Figure 27. Class 2 Waterfall Plot

- ID, cluster 생성 후 -

Row 1 (정답 : Class 2)

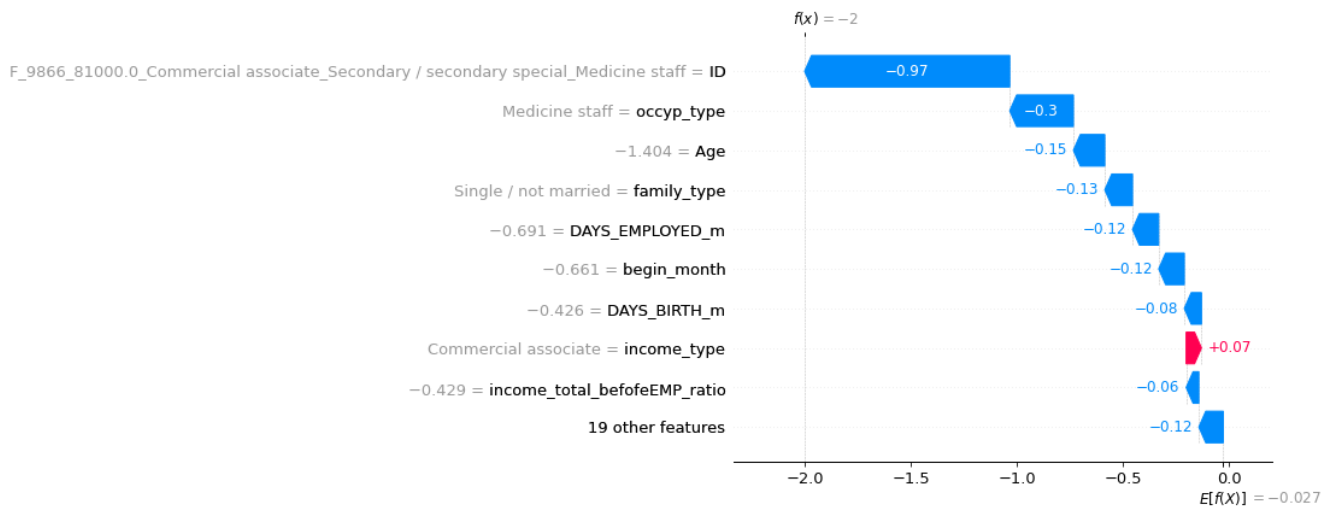


Figure 28. Class 0 Waterfall Plot

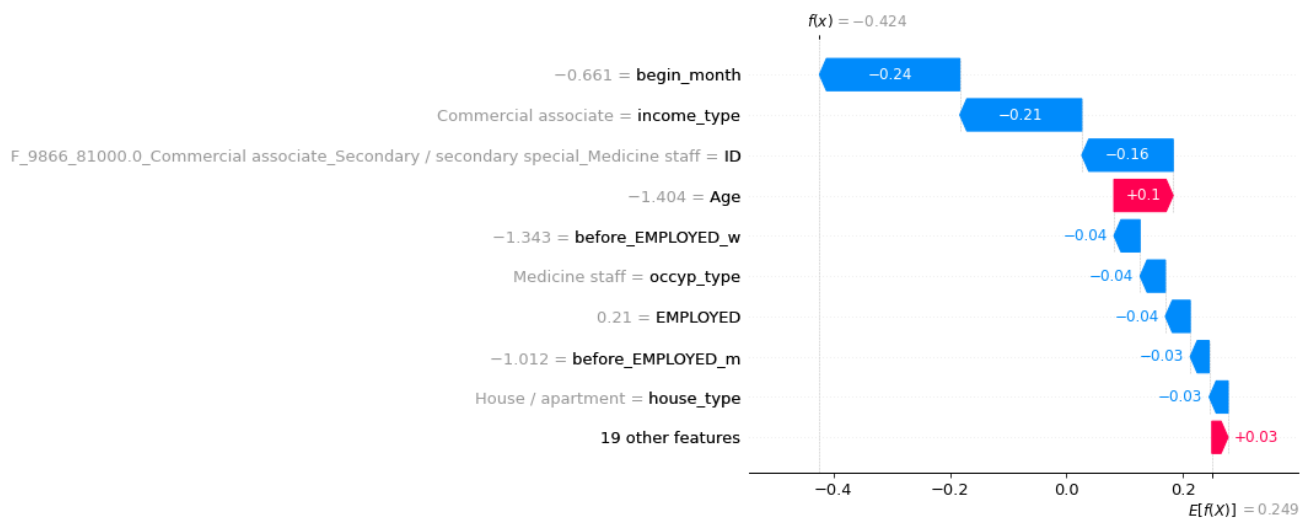


Figure 29. Class 1 Waterfall Plot

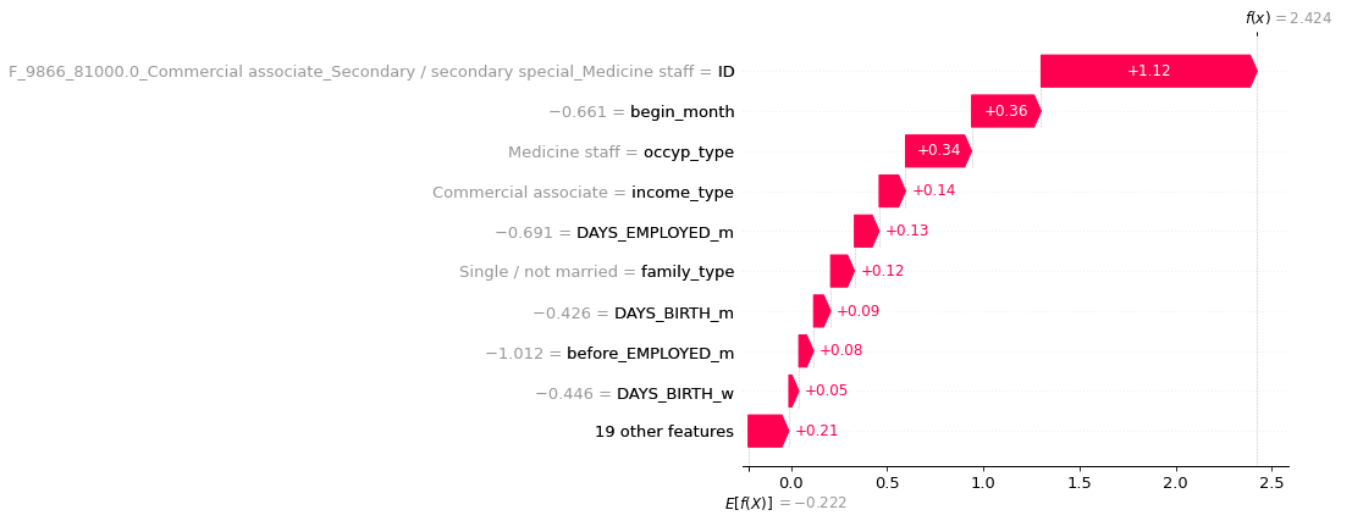


Figure 30. Class 2 Waterfall Plot

Row 16540(정답 : Class 0)

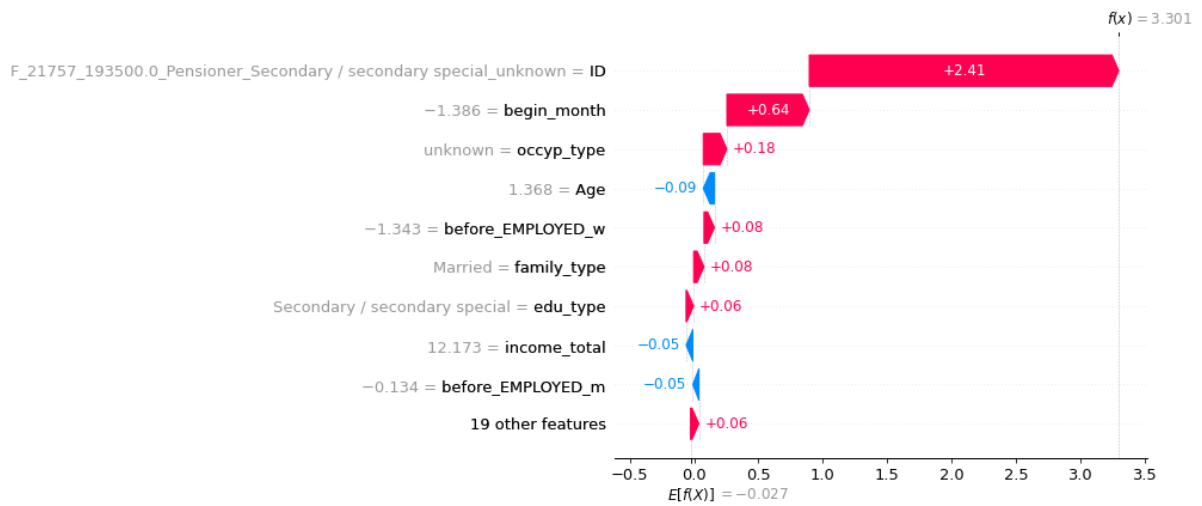


Figure 31. Class 0 Waterfall Plot

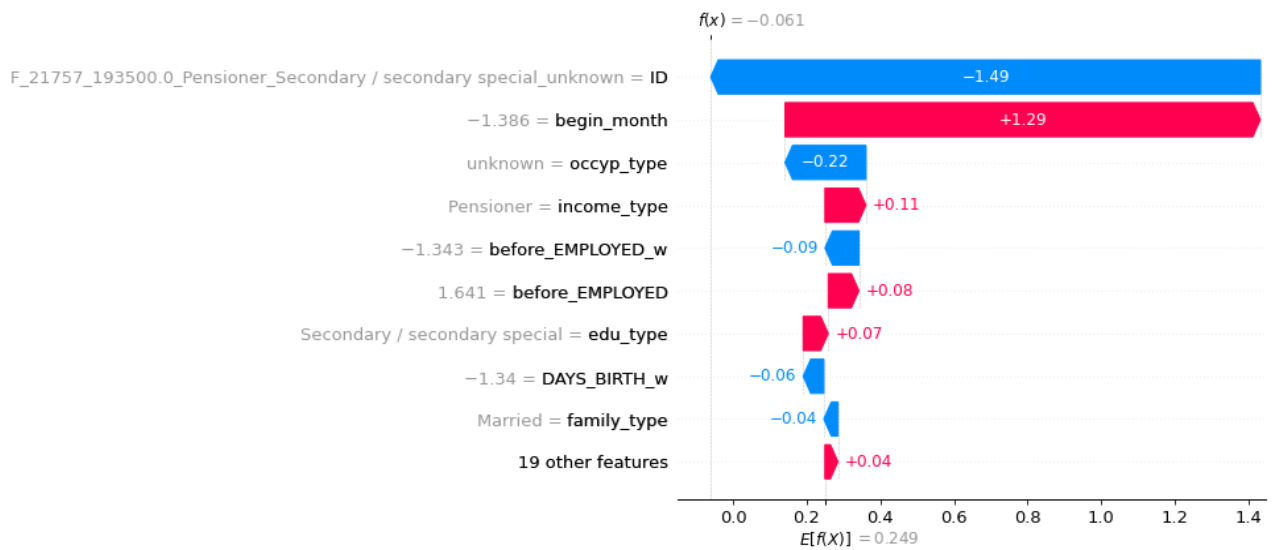


Figure 32 Class 1 Waterfall Plot

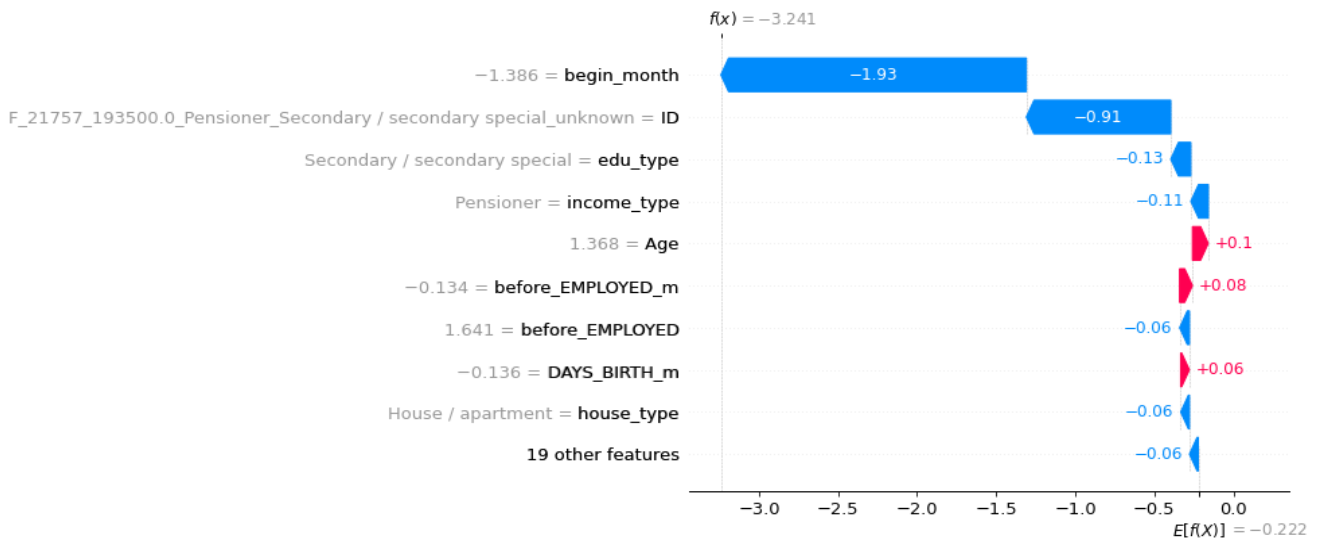


Figure 33. Class 2 Waterfall Plot

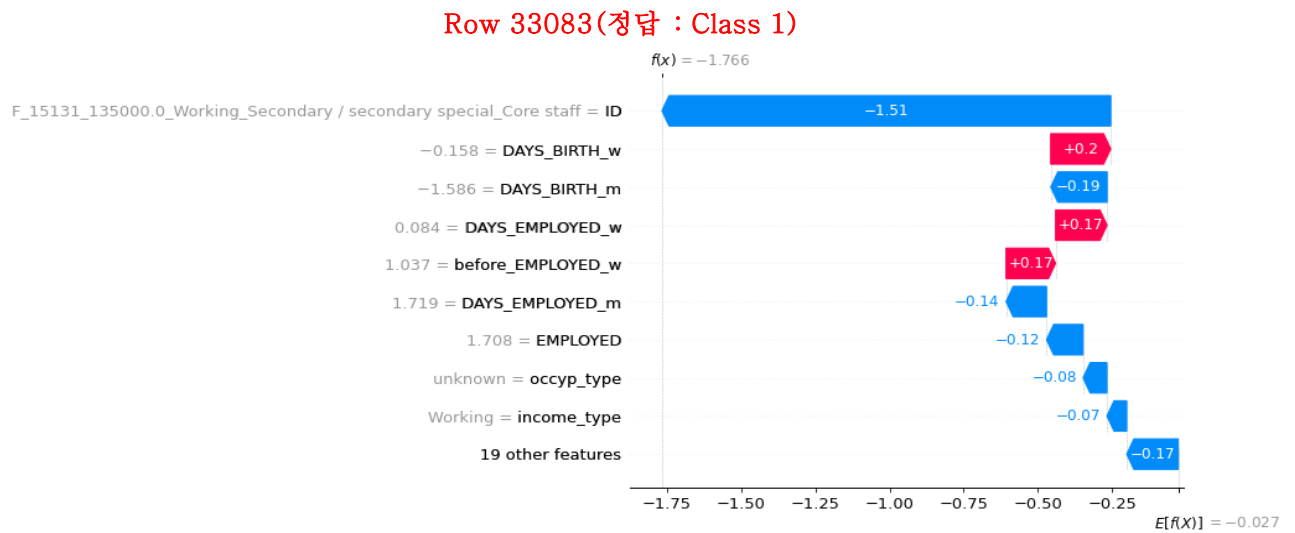


Figure 34 Class 0 Waterfall Plot

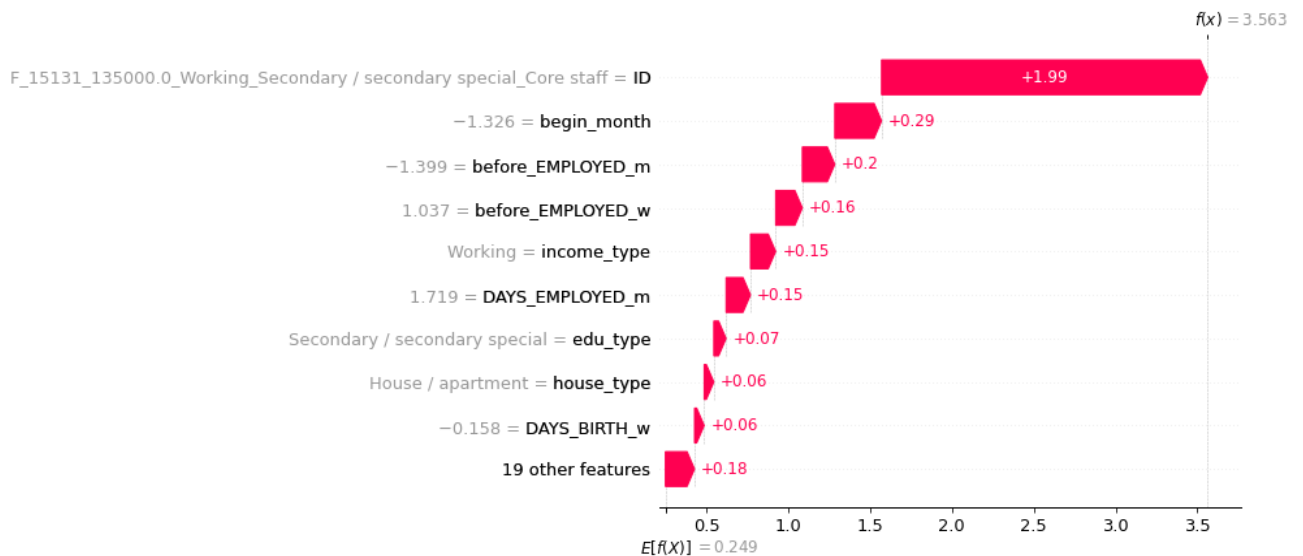


Figure 35 Class 1 Waterfall Plot

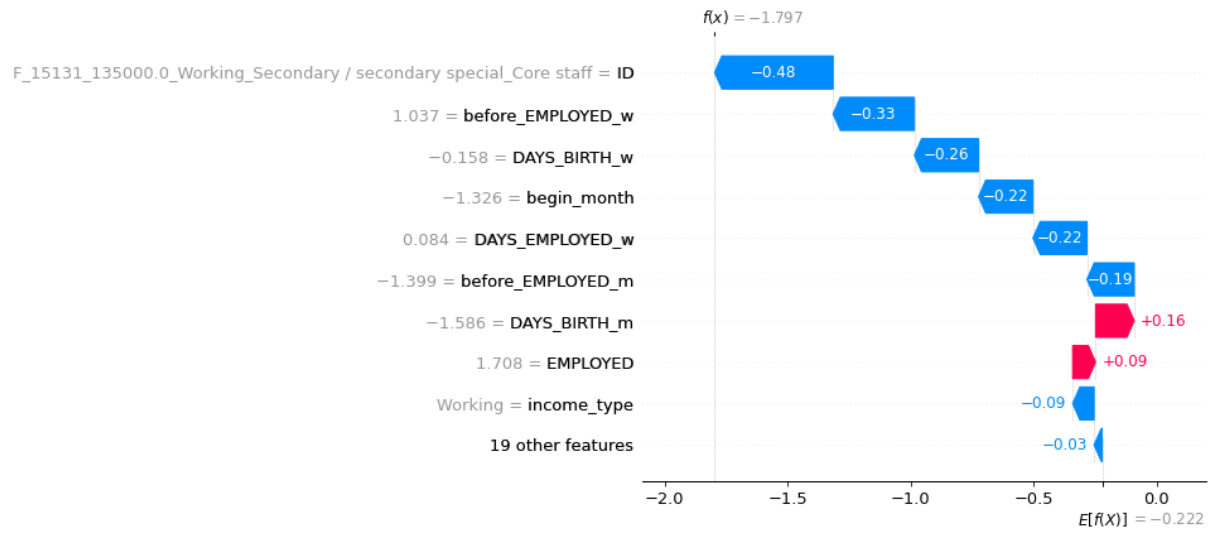


Figure 36 Class 2 Waterfall Plot