Classifying Credit Card Users: A Machine Learning Analysis

Zero-base 1팀 / 김경훈, 목해민, 안선경, 윤세종, 이선명

Contents

1. 프로젝트 소개

주제선정 배경 및 목표

- 신용카드 사용률은 매년 증가하고 있으며 개인 이용금액 또한 증가하고 있음.
- 금리 상승, 신용카드 연체율 급증으로 인해 신용카드사의 자산건전성 약화와 위험부담이 증가하고 있음.
- 국내 빅테크 기업 주도 아래 후불 결제 시장이 성장하면서 연체율 관리가 매우 중요해짐.
- 신용카드사는 신용등급으로 연체 가능성을 판단하기에 신용등급 산정은 매우 중요함.

프로젝트 목표

카드 대금 연체 집단의 정보를 통해 연체 정도를 예측할 수 있는 알고리즘을 개발하고
건전한 금융시장 유지에 도움이 되는 인사이트를 제공한다.

1. 프로젝트 소개

데이터 소개

- 출처 : Dacon 사이트, https://dacon.io/competitions/official/235713/data
- Data Shape : (26457, 20)

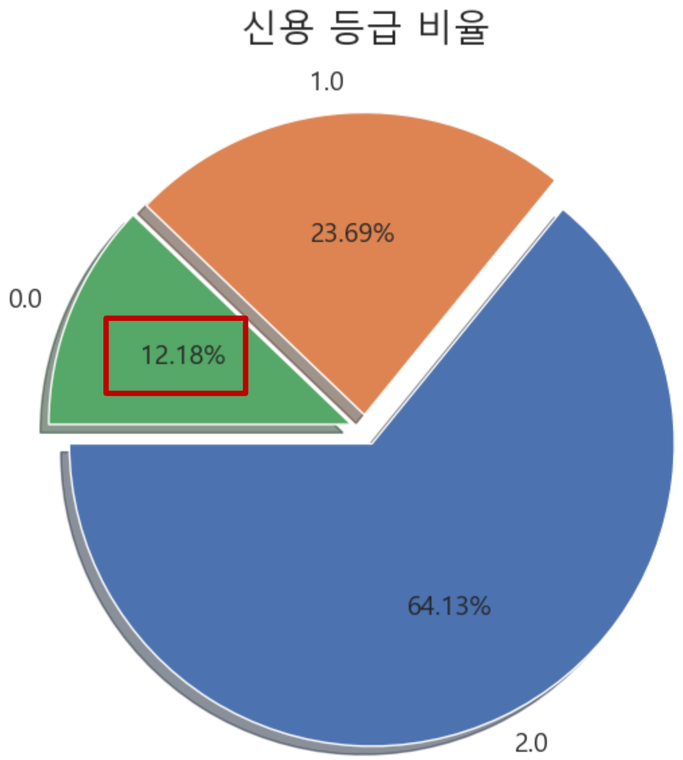
	child_num	income_total	DAYS_BIRTH	DAYS_EMPLOYED	occyp_type	begin_month	credit
0	0	202500.0	-13899	-4709	NaN	-6.0	1.0
1	1	247500.0	-11380	-1540	Laborers	-5.0	1.0
2	0	450000.0	-19087	-4434	Managers	-22.0	2.0

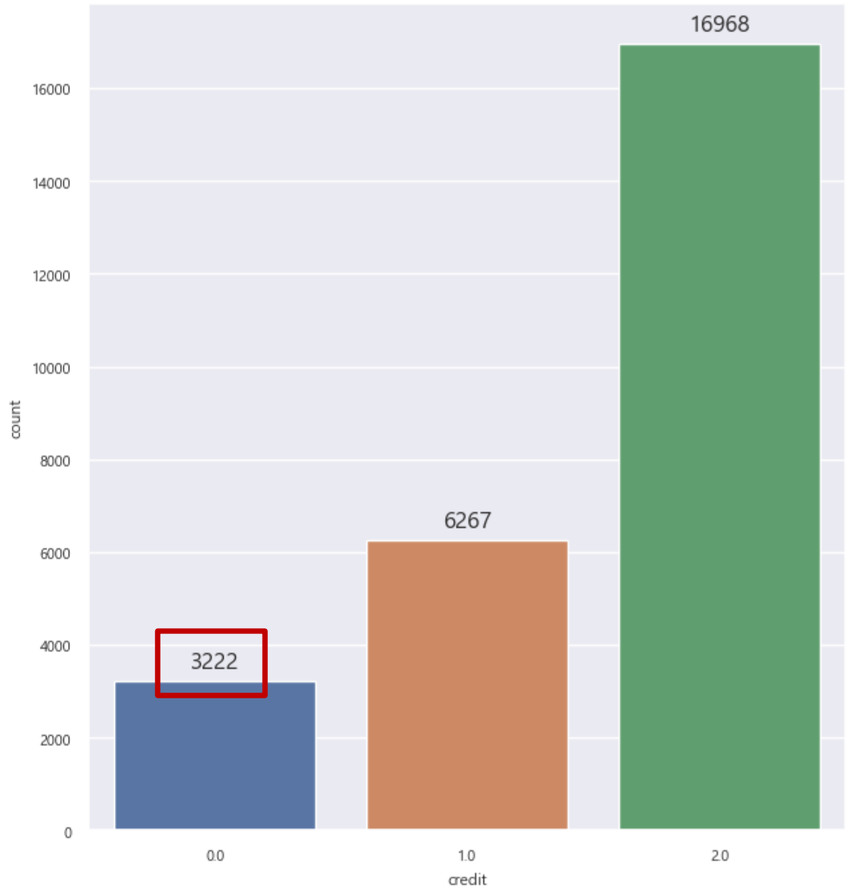
- Gender : 성별
- Car : 차량 소유 여부
- Reality : 부동산 소유 여부
- Child_num : 자녀수
- Income_total : 연간 소득
- Income_type : 소득 분류
- Edu_type : 교육 수준
- Family_type : 결혼 여부
- House_tyep : 생활 방식
- Days_birth : 태어난 일수

- Days_employed : 일한 일수
- FLAG_MOBIL : 핸드폰 소유 여부
- Work_phone : 업무용 전화 소유 여부
- Occupy_type : 직업 유형
- Family_size : 가족 규모
- Begin_month : 신용카드 발급 개월
- Credit : 신용카드 대금 연체를 기준으로 한 신용도

1. 프로젝트 소개

데이터 소개





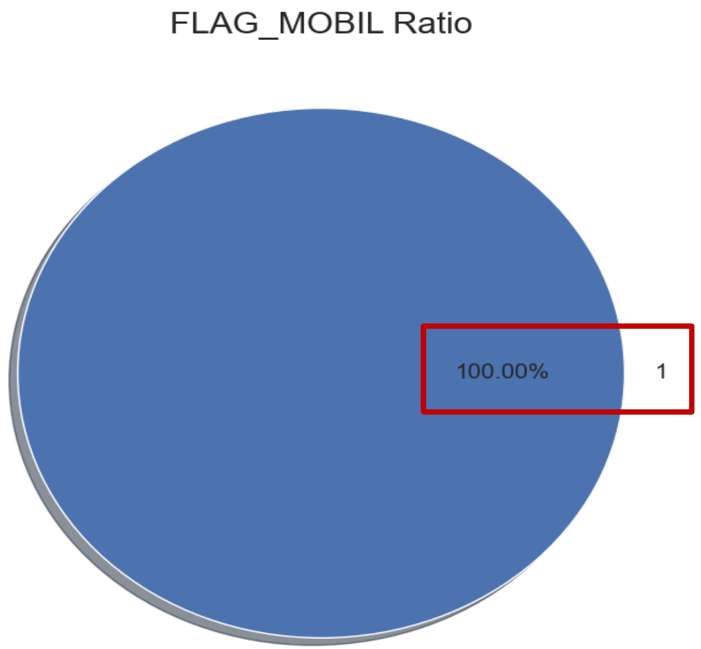
Label 설명

1. 신용도는 숫자가 숫자가 클수록 '나쁨'을 뜻함.
2. 0, 1, 2 데이터에 대한 불균형이 존재함.

2. EDA 및 전처리

Feature 제거

〈Child_Num 과 family_size의 관계〉

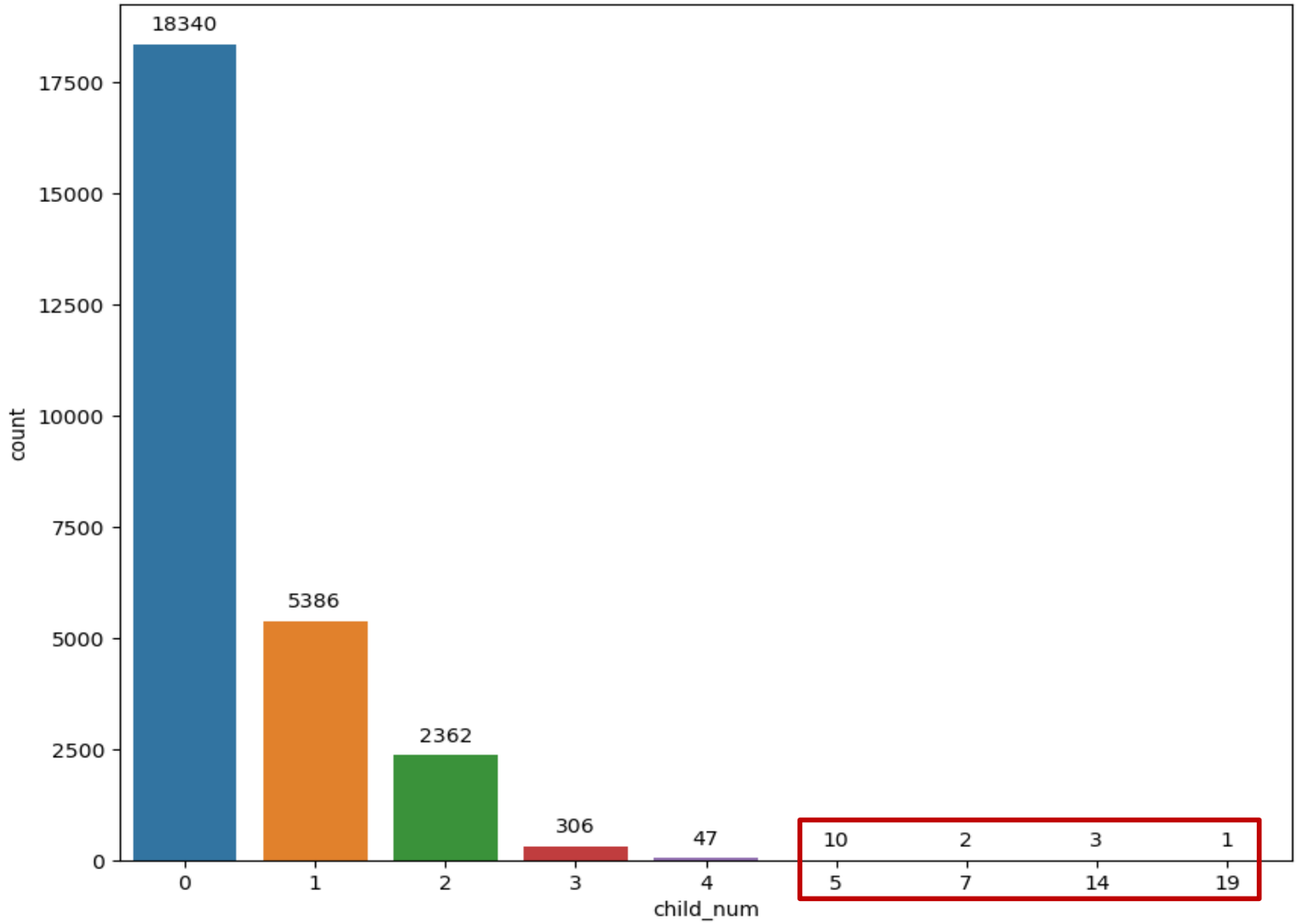


〈FLAG_MOBIL의 값 분포〉

1. **Index column 삭제**
2. **family_size column 삭제**
3. **FLAG_MOBIL column 삭제**

2. EDA 및 전처리

이상치 제거



1. **child_num column에 5명 이상인 값은 전체 데이터의 0.1%**
2. **이상치 데이터로 판단, 제거함.**

2. EDA 및 전처리

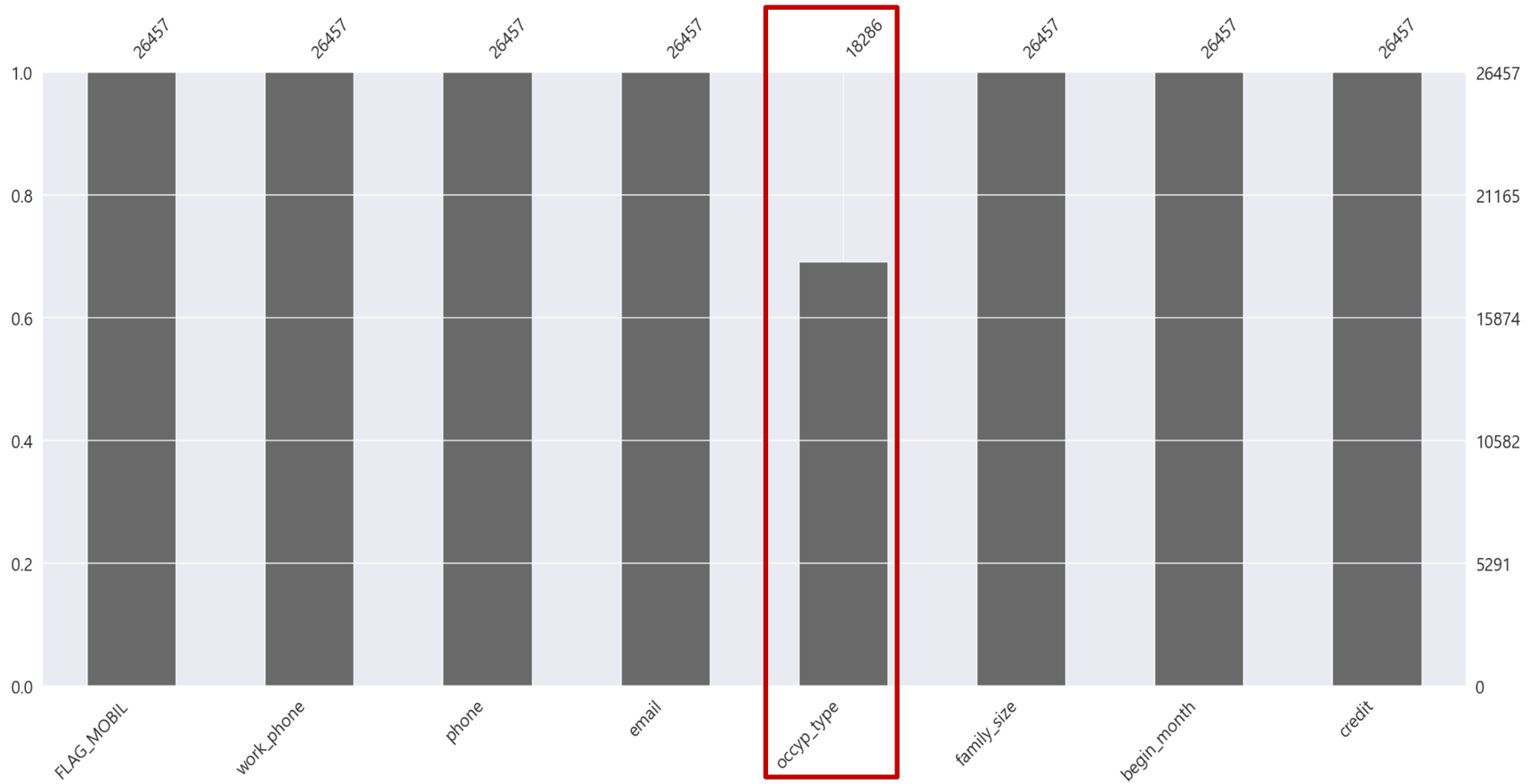
연속형 자료 변환

income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	work_phone	phone	email
Pensioner	Secondary / secondary special	Married	House / apartment	-23113	365243	0	0	0
Working	Secondary / secondary special	Married	House / apartment	-13727	-6031	0	0	0
Working	Secondary / secondary special	Married	House / apartment	-19850	-1753	0	1	0
Pensioner	Secondary / secondary special	Married	House / apartment	-21253	365243	0	1	0
Commercial associate	Secondary / secondary special	Civil marriage	House / apartment	-15198	-1357	0	0	0

1. **DYAS_EMPLOYED column의 0 이상 값은 모두 365243**
2. **이는 모두 무직자이기 때문에 0으로 변환함.**
3. **음수의 경우 정도를 나타내는 것이기 때문에 양수로 변환함.**

2. EDA 및 전처리

결측치 확인



1. **occyp_type column만 결측치가 존재함.**

2. EDA 및 전처리

Feature 범주화 및 단위 조정

	income_total	income_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	occyp_type	begin_month
0	20.25	0	2	38.079452	392.416667	18	6.0
1	24.75	0	1	31.178082	128.333333	8	5.0
2	45.00	4	1	52.293151	369.500000	10	22.0
3	20.25	0	1	41.336986	174.333333	14	37.0
4	15.75	2	1	41.197260	175.416667	10	26.0
...	...	...	...	...	...	...	...
26452	22.50	2	1	33.093151	165.333333	3	2.0
26453	18.00	4	1	41.893151	206.250000	18	47.0
26454	29.25	4	5	27.621918	167.916667	3	25.0
26455	17.10	4	1	27.794521	8.916667	8	59.0
26456	8.10	4	1	53.613699	84.416667	16	9.0

26457 rows × 8 columns

1. 범주형 Data의 경우 LabelEncoder를 통해 변환
2. 연속형 Data의 경우 각 데이터에 맞게 단순 단위 변환

3. ML Modeling

평가 지표 선정

1. Recall Result
2. Log loss Result

3. ML Modeling

- **Log loss**

$$-\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1-y_i) \cdot log(1-p(y_i))$$

- p(yi) is the probability of one

- Target의 실제 값에 대한 Predict probability를 log변환한 값의 평균
- 자연로그의 특성상 probability가 0에 가까워지는 경우 그 값이 음의 무한대로 수렴하기 때문에 예측실패한 데이터의 probability에 따라 가중치가 더해짐.
- 불균형 데이터의 경우 다수를 차지하는 데이터에 대한 예측은 좋고 반대의 경우 예측이 낮아지는 경우가 많은데, 예측의 실패정도에 가중치가 주어지기 때문에 평가지표로 적당하다 판단되어 선정.

3. ML Modeling

모델 성능 결과 확인

	XGBoost			LightGBM			CatBoost			RandomForest			DecisionTree		
Accuracy_score	0.71			0.71			0.72			0.71			0.63		
Log_Loss	0.85			0.82			0.72			0.74			12.81		
	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	0.27	0.44	0.33	0.23	0.41	0.29	0.12	0.58	0.19	0.25	0.40	0.31	0.34	0.28	0.30
Credit_1	0.48	0.61	0.54	0.46	0.62	0.53	0.36	0.73	0.48	0.49	0.60	0.54	0.50	0.48	0.49
Credit_2	0.87	0.76	0.82	0.89	0.76	0.82	0.97	0.73	0.83	0.86	0.77	0.81	0.73	0.77	0.75

3. ML Modeling

파생변수 생성

income_occupy: 소득에 따른 직업 유형

car_reality: 자산 소유 여부

income_wage: 연소득

employed_wage: 근로소득

card_begin_before_employed: 카드 발급일 기준 근로 여부

before_EMPLOYED: 미취업기간

income_total_beforeEMP_ratio: 취업 전 소득

DAYS_BIRTH_m: 태어난 월

DAYS_BIRTH_w: 태어난 주

DAYS_EMPLOYED_m: 고용된 월

DAYS_EMPLOYED_w: 고용된 주

ability: 연령/근무일 대비 소득

income_mean: 가족 수를 고려한 소득 평균

3. ML Modeling

모델 성능 결과 확인

성능향상, 성능하락

	XGBoost			LightGBM			CatBoost			RandomForest			DecisionTree		
Accuracy_score	▼ 0.70			▼ 0.70			▼ 0.71			(-) 0.71			(-) 0.63		
Log_Loss	▼ 0.88			▼ 0.84			▼ 0.75			▲ 0.73			▲12.71		
	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	▼ 0.25	0.41	0.31	▼ 0.20	0.40	0.26	▼ 0.08	0.52	0.14	▼ 0.22	0.39	0.28	▼ 0.32	0.28	0.30
Credit_1	▼ 0.45	0.59	0.51	▼ 0.44	0.60	0.51	▼ 0.33	0.73	0.46	▼ 0.47	0.61	0.53	▼ 0.48	0.46	0.47
Credit_2	(-) 0.87	0.75	0.81	(-) 0.89	0.75	0.81	(-) 0.97	0.71	0.82	▲ 0.88	0.76	0.81	▼ 0.73	0.77	0.76

문제점

1. 파생 변수를 추가했을 때 오히려 성능이 떨어지는 문제

2. 다양한 방식으로 column 선택의 변화를 주었으나 성능 개선이 이루어지지 않음

3. ML Modeling

중복 데이터 확인

1. Begin_month는 같지만, Credit이 다른 경우

2. Credit은 같지만, Begin_month가 다른 경우

income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	work_phone	phone	email	occyp_type	begin_month	credit
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-22.0	0.0
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-22.0	2.0
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-36.0	0.0
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-18.0	0.0
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-5.0	0.0
270000.0	4	4	1	1	-14488	-1630	0	1	0	8	-36.0	0.0

3. ML Modeling

고유ID column 생성

한 사람이 여러 카드를 발급받았다는 가정, begin_month와 라벨 credit를 제외한 모든 컬럼을 합쳐서 한 사람을 식별하는 고유 ID컬럼을 생성

```
# 개인의 식별번호 컬럼
raw_df["SSN"] = raw_df["gender"].astype("str") +  raw_df["car"].astype("str") + raw_df["child_num"].astype("str") +
raw_df["income_total"].astype("str") + raw_df["income_type"].astype("str") + raw_df["edu_type"].astype("str") +
raw_df["family_type"].astype("str") + raw_df["house_type"].astype("str") + raw_df["DAYS_BIRTH"].astype("str") +
raw_df["DAYS_EMPLOYED"].astype("str") + raw_df["work_phone"].astype("str") + raw_df["phone"].astype("str") +
raw_df["email"].astype("str") + raw_df["occyp_type"].astype("str") + raw_df["family_size"].astype("str")
```

3. ML Modeling

모델 성능 결과 확인

성능향상, 성능하락

	XGBoost			LightGBM			CatBoost		
Accuracy_score	(-) 0.71			(-) 0.71			▲ 0.74		
Log_Loss	▲ 0.80			▲ 0.79			▲ 0.66		
	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	▲ 0.29	0.50	0.37	▲ 0.24	0.47	0.32	▲ 0.26	0.61	0.36
Credit_1	▼ 0.47	0.61	0.53	▼ 0.42	0.62	0.50	▼ 0.44	0.70	0.54
Credit_2	▲ 0.88	0.88	0.81	▲ 0.90	0.74	0.81	▲ 0.94	0.75	0.83

- 중복데이터를 고유 ID 부여로 처리 후 큰 성능 개선이 나타남.

- 특히, CatBoost의 경우 LogLoss 지표에서 다른 모델에 비해 결과값에 유의미한 차이가 있음.

3. ML Modeling

중복 데이터 제거

Credit은 같지만, Begin_month가 다른 경우를 삭제

성능향상, 성능하락

	XGBoost			LightGBM			CatBoost		
Accuracy_score	▲ 0.77			▲ 0.77			▲ 0.77		
Log_Loss	▲ 0.73			▲ 0.73			▲ 0.62		
	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	▲ 0.35	0.55	0.43	▲ 0.33	0.59	0.42	▼ 0.14	0.77	0.24
Credit_1	▲ 0.53	0.61	0.57	▲ 0.51	0.62	0.56	▼ 0.36	0.74	0.48
Credit_2	▲ 0.89	0.82	0.86	▲ 0.91	0.81	0.86	▲ 0.98	0.77	0.86

- 고유ID를 부여하는 것보다 중복 데이터를 제거한 것이 더 좋은 결과값을 보여줌

3. ML Modeling

Clustering / PCA

성능향상, 성능하락

	XGBoost			LightGBM			CatBoost		
Accuracy_score	(-) 0.77			(-) 0.77			▲ 0.78		
Log_Loss	▼ 0.74			▼ 0.76			▲ 0.60		
	Recall	Precision	F1 score	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	▲ 0.37	0.60	0.45	▼ 0.30	0.59	0.40	▲ 0.25	0.73	0.38
Credit_1	▲ 0.54	0.64	0.58	▲ 0.51	0.63	0.56	▲ 0.42	0.71	0.52
Credit_2	▲ 0.90	0.82	0.86	(-) 0.91	0.81	0.86	▼ 0.96	0.79	0.87

3. ML Modeling

SMOTE

SMOTE						
	Before			After		
Accuracy_score	0.78			0.75		
Log_Loss	0.60			0.65		
	Recall	Precision	F1 score	Recall	Precision	F1 score
Credit_0	0.25	0.73	0.38	0.40	0.49	0.44
Credit_1	0.42	0.71	0.52	0.53	0.58	0.55
Credit_2	0.96	0.79	0.87	0.86	0.82	0.84

- SMOTE 적용 후 0과 1에 대한 Recall 값은 개선되었으나 목표로 하는 2에 대한 Recall 값은 감소함.

3. ML Modeling

최종 모델

최종모델		성능 지표	
분류 모델	CatBoost	**Recall_0**	0.25
주요 처리 사항	• occpy_type NaN 처리 • 중복데이터 제거 • Clustering / PCA 적용 • 연속형 변수 단위 변환 미적용	**Recall_1**	0.44
		Recall_2	0.95
사용 Feature	• Feature engineering Income_total, edu_type, family_type, house_type, day_birth, day_employed, begin_month, income_occpy, car_reality, car_begin_before_employed	**Accuracy**	0.77
		Log_Loss	0.60

프로젝트 목표

카드 대금 연체 집단의 정보를 통해 연체 정도를 예측할 수 있는 알고리즘을 개발하고
건전한 금융시장 유지에 도움이 되는 인사이트를 제공한다.

Q&A