



미니프로젝트 발표

DA팀 손소현 오수진 오연재 이의진

목차

#01 데이터 및 주제 소개

#02 EDA

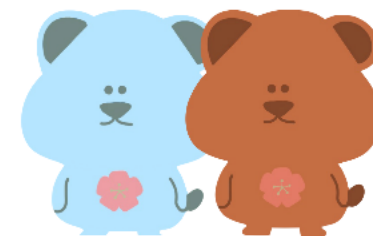
#03 분류

#04 클러스터링

#05 한계점 및 마무리



01. 데이터 및 주제 소개



1. 데이터/주제 소개

Featured Prediction Competition

Home Credit Default Risk

Can you predict how capable each applicant is of repaying a loan?

Home Credit Group

7,176 teams

4 years ago

Overview

Data

Code

Discussion

Leaderboard

Rules

Team

My Submissions

Late Submission

...

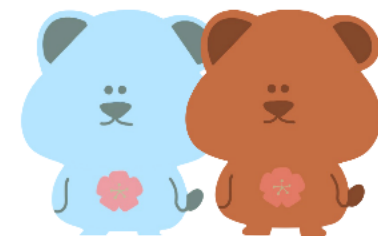
<https://www.kaggle.com/competitions/home-credit-default-risk/data>

💡 주제	사용자의 대출금 납부 여부에 대한 예측 모델 구축 및 사용자 군집화
데이터	122개의 열에 개별 고객에 대한 정보(나이, 성별, 대출금 상환 여부, 지역, 직업 등에 대한 정보 포함)
	(307511, 122)

↓ 데이터 예시

TARGET	FLAG_OWN_CAR	FLAG_OWN_REALTY	AMT_INCOME_TOTAL	AMT_CREDIT	...	AMT_GOODS_PRICE	NAME_TYPE_SUITE	CNT_CHILDREN	AMT_ANNUITY
1	Y	Y	202500	406597.5		351000	Unaccompanied	0	24700.5
0	N	N	270000	1293503		1129500	Family	2	35698.5

02. EDA

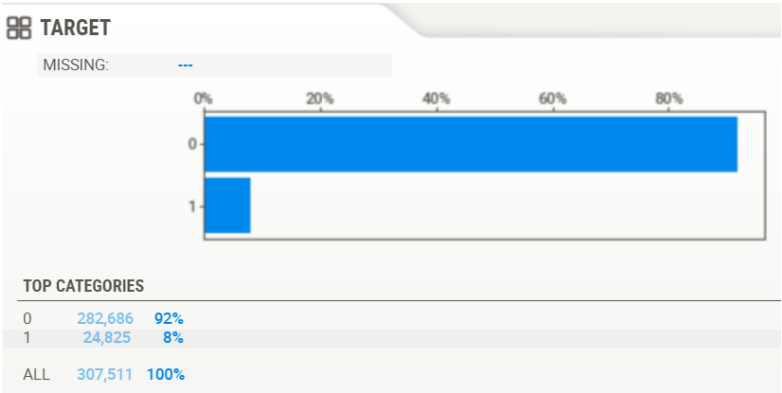


1. EDA

Auto EDA(sweetviz)를 이용하여 데이터의 전반적인 분포 확인
(고객의 나이, 직업의 분포 등 확인)

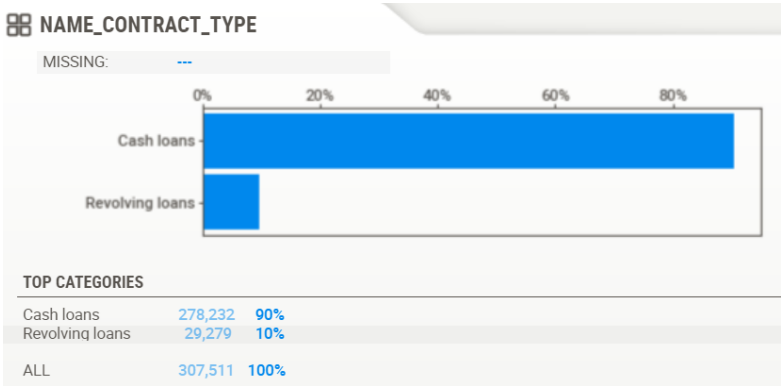
<TARGET 변수 분포>

- 심한 불균형 데이터
- 대출금을 상환한 값(0) >>> 상환하지 못한 값(1)



<TARGET 변수 외의 변수 분포 >

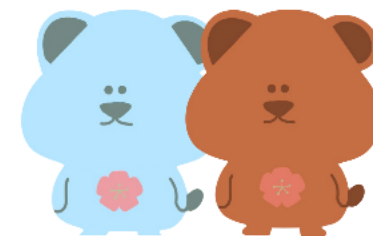
TARGET 변수 외에도 심한 불균형 분포를 가지는 변수들 다수 존재함



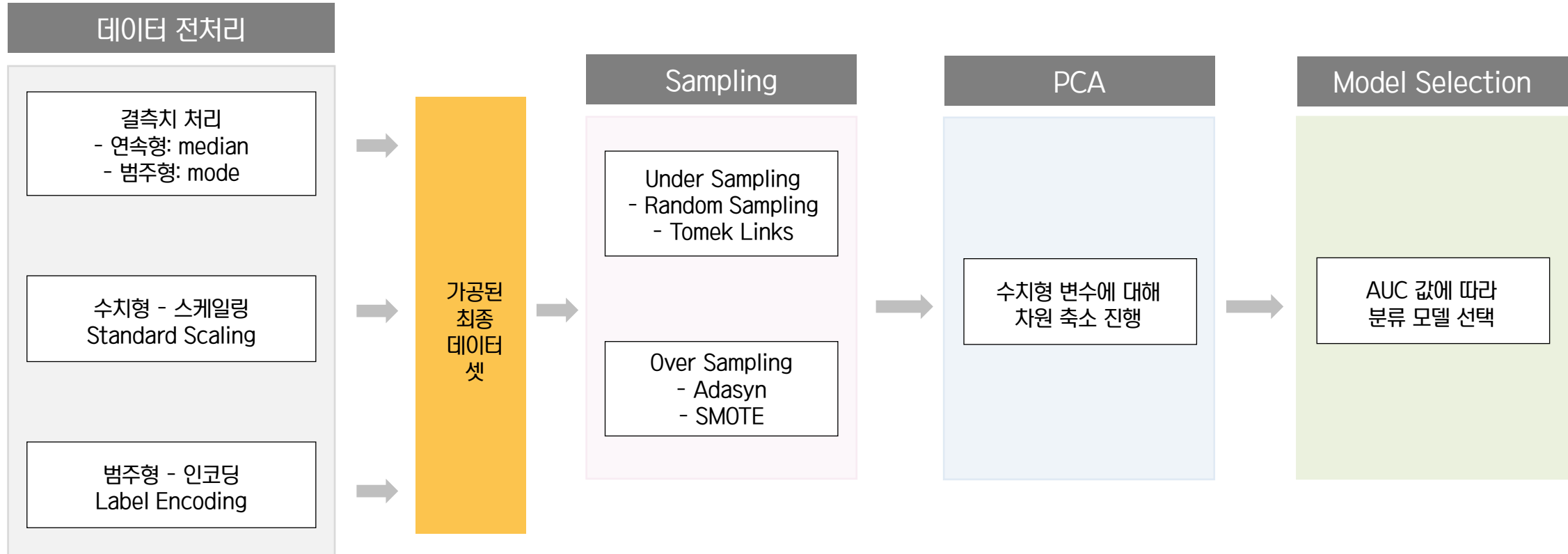
<불균형 문제 해결 위해
→ 다양한 Sampling 방식 사용하여 분류 및 분석 진행>

Sampling 방식	
Over Sampling	SMOTE
	Adasyn
Under Sampling	Random Sampling
	Tomek Links

03. 분류



1. 분류 프로세스



2. 모델 학습 및 결과

LogisticRegression, LGBM, XGBoost, RandomForest 4가지 모델에 각각 데이터를 학습시키고 예측 후 성능 평가

➡ AUC 값에 따라 최종 모델 선택

모델	
LogisticRegression	<pre>from sklearn.linear_model import LogisticRegression lr_clf = LogisticRegression() get_model_train_eval(lr_clf, ftr_train=X_train, ftr_test=X_test, tgt_train=y_train, tgt_test=y_test)</pre>
LightGBM	<pre>from lightgbm import LGBMClassifier lgbm_clf = LGBMClassifier(n_estimators=1000, num_leaves=64, n_jobs=-1, boost_from_average=False) get_model_train_eval(lgbm_clf, ftr_train=X_train, ftr_test=X_test, tgt_train=y_train, tgt_test=y_test)</pre>
XGBoost	<pre>from xgboost import XGBClassifier xgb_wrapper = XGBClassifier(n_estimators=400, learning_rate=0.1, max_depth=3) get_model_train_eval(xgb_wrapper, ftr_train=X_train, ftr_test=X_test, tgt_train=y_train, tgt_test=y_test)</pre>
RandomForest	<pre>from sklearn.ensemble import RandomForestClassifier rf_clf = RandomForestClassifier(random_state=0) get_model_train_eval(rf_clf, ftr_train=X_train, ftr_test=X_test, tgt_train=y_train, tgt_test=y_test)</pre>

3. 분류 결과

4가지 샘플링 X 4개의 모델

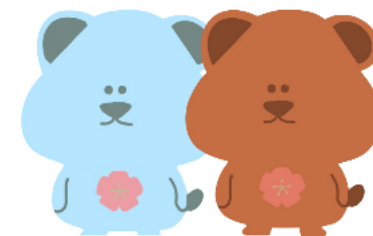
	LGBM	XGBoost	RandomForest	Logistic
TomekLinks	정확도: 0.6829, 정밀도: 0.1589, 재현율: 0.6812, F1: 0.2577, AUC:0.7479	정확도: 0.6835, 정밀도: 0.1583, 재현율: 0.6760, F1: 0.2565, AUC:0.7436	정확도: 0.9194, 정밀도: 0.6552, 재현율: 0.0038, F1: 0.0076, AUC: 0.7170	정확도: 0.9191, 정밀도: 0.4483, 재현율: 0.0105, F1: 0.0205, AUC: 0.7476
SMOTE	정확도: 0.9183, 정밀도: 0.4316, 재현율: 0.0369, F1: 0.0679, AUC: 0.7467	정확도: 0.9195, 정밀도: 0.4392, 재현율: 0.0324, F1: 0.0604, AUC:0.7453	정확도: 0.9150, 정밀도: 0.2630, 재현율: 0.0296, F1: 0.0532, AUC: 0.6927	정확도: 0.7251, 정밀도: 0.1545, 재현율: 0.5378, F1: 0.2400, AUC: 0.6963
Adasyn	정확도: 0.9187, 정밀도: 0.4657, 재현율: 0.0396, F1: 0.0731, AUC:0.7459	정확도: 0.9190, 정밀도: 0.4717, 재현율: 0.0201, F1: 0.0386, AUC:0.7345	정확도: 0.9143, 정밀도: 0.2397, 재현율: 0.0280, F1: 0.0501, AUC:0.6893	정확도: 0.7179, 정밀도: 0.1509, 재현율: 0.5385, F1: 0.2357, AUC:0.6901
Random Under Sampling	정확도: 0.6835, 정밀도: 0.1583, 재현율: 0.6760, F1: 0.2565, AUC:0.7436	정확도: 0.6967, 정밀도: 0.1668, 재현율: 0.6895, F1: 0.2687, AUC:0.7595	정확도: 0.6931, 정밀도: 0.1612, 재현율: 0.6655, F1: 0.2595, AUC:0.7410	정확도: 0.6829, 정밀도: 0.1589, 재현율: 0.6812, F1: 0.2577, AUC:0.7479

3. 분류 결과

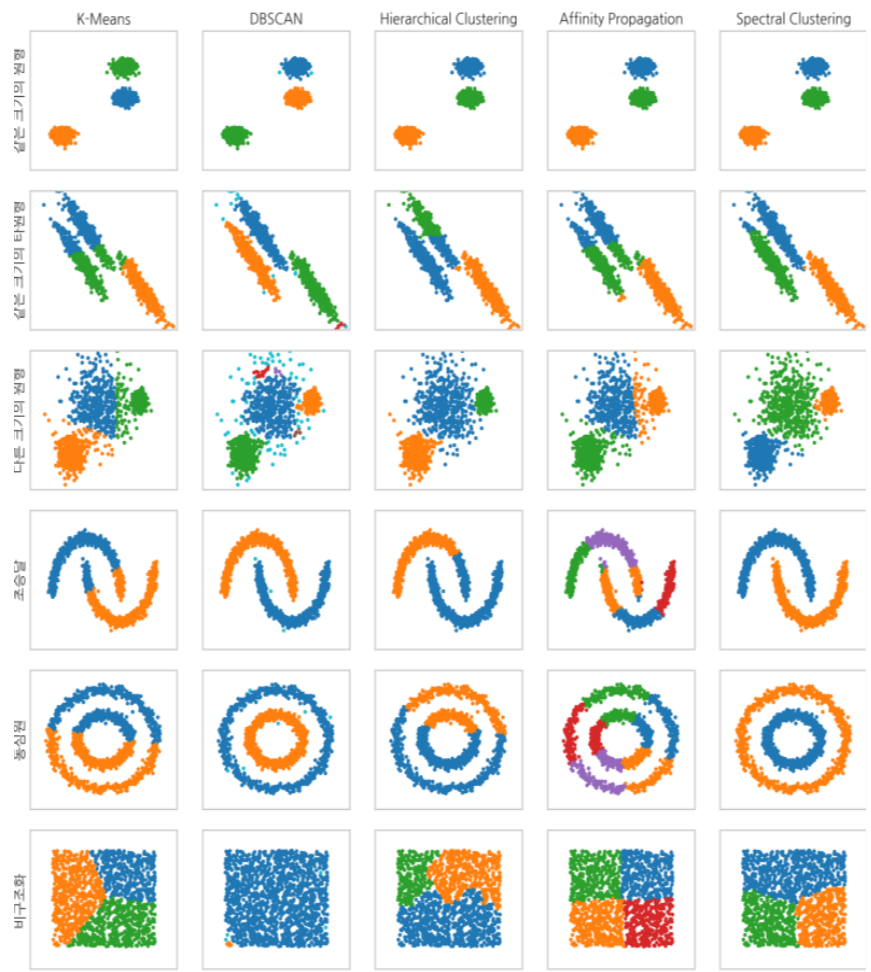
4가지 샘플링 + PCA X 4개의 모델

	LGBM	XGBoost	RandomForest	Logistic
TomekLinks	정확도: 0.9191, 정밀도: 0.4783, 재현율: 0.0066, F1: 0.0131, AUC:0.7318	정확도: 0.9194, 정밀도: 0.5789, 재현율: 0.0066, F1: 0.0131, AUC: 0.7394	정확도: 0.9191, 정밀도: 0.4667, 재현율: 0.0014, F1: 0.0028, AUC:0.6848	정확도: 0.9188, 정밀도: 0.3297, 재현율: 0.0060, F1: 0.0119, AUC: 0.7309
SMOTE	정확도: 0.8425, 정밀도: 0.1744, 재현율: 0.2546, F1: 0.2070, AUC: 0.6699	정확도: 0.7607, 정밀도: 0.1488, 재현율: 0.4163, F1: 0.2193, AUC: 0.6673	정확도: 0.8780, 정밀도: 0.1763, 재현율: 0.1298, F1: 0.1496, AUC:0.6529	정확도: 0.8532, 정밀도: 0.1822, 재현율: 0.2344, F1: 0.2050, AUC: 0.6748
Adasyn	정확도: 0.8338, 정밀도: 0.1666, 재현율: 0.2642, F1: 0.2044, AUC:0.6672	정확도: 0.7534, 정밀도: 0.1472, 재현율: 0.4279, F1: 0.2190, AUC:0.6652	정확도: 0.8471, 정밀도: 0.1741, 재현율: 0.2383, F1: 0.2012, AUC:0.6729	정확도: 0.6918, 정밀도: 0.1410, 재현율: 0.5526, F1: 0.2247, AUC:0.6752
Random Under Sampling	정확도: 0.6620, 정밀도: 0.1469, 재현율: 0.6625, F1: 0.2405, AUC:0.7212	정확도: 0.6751, 정밀도: 0.1551, 재현율: 0.6794, F1: 0.2525, AUC:0.7408	정확도: 0.6818, 정밀도: 0.1510, 재현율: 0.6357, F1: 0.2440, AUC:0.7170	정확도: 0.6678, 정밀도: 0.1510, 재현율: 0.6732, F1: 0.2467, AUC:0.7333

04. 클러스터링



1. 클러스터링



	설명	사용 기법	변수 종류
군집화 Clustering	신용 정보 데이터를 유사한 그룹끼리 묶어, 데이터 내 패턴을 파악하고자 함	KModes	범주형 변수
		KMeans	수치형 변수
		KPrototypes	범주형 + 수치형
스케일링 Scaling	데이터 불균형을 완화하여, 클러스터별로 충분한 데이터가 확보되도록, 여러 스케일링 방법을 시도하여 최적의 클러스터 개수 확인	log 변환	
		Standard Scaling	
		MinMax Scaling	

➡ 군집화 결과 및 데이터의 특성을 고려하여
StandardScaling 데이터를 이용하여 KMeans 클러스터링 진행함.

2. KModes

데이터 소개

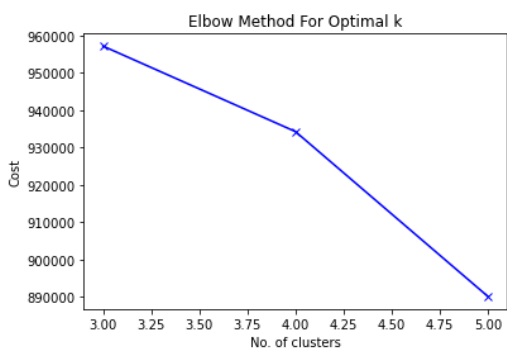
- 1. 분류 성능이 가장 우수했던 XGBoost를 기준으로 피쳐 중요도가 높은 상위 28개의 변수 추출
- 2. 중요도가 높은 28개의 변수 중, 범주형 변수만(=19개 변수) 추출한 데이터 사용

최종 사용 데이터 소개

전처리

KModes는 범주형 변수에 대한 클러스터링 방법
→ 별도의 전처리 과정 X

최적 k 설정



명확한 Elbow Point 지점 없음
→ 최적의 클러스터 개수 찾지 못함
→ KMeans로 군집화 시도

TARGET	FLAG_OWN_CAR	FLAG_OWN_REALTY	FLAG_EMP_PHONE	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL	FLAG_DOCUMENT_3	FLAG_DOCUMENT_6	FLAG_DOCUMENT_8
납부 여부	자동차 소유 여부	주택 소유 여부	직장 전화 제공 여부	집전화 제공 여부	집전화 제공 여부	Email 제공 여부	문서3 제공여부	문서6 제공여부	문서8 제공여부

REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	REG_CITY_NOT_WORK_CITY	REG_CITY_NOT_LIVE_CITY	NAME_CONTRACT_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_TYPE_SUITE	WEEKDAY_APPR_PROCESS_START
고객 거주 도시 평가	고객 거주지 평가	영구 주소와 직장 주소 일치 여부	영구 주소와 연락처 주소 일치 여부	계약 상품 종류	최종 학력	가족 형태	고객과 동행한 사람	Application 신청 요일

3. KMeans

데이터 소개

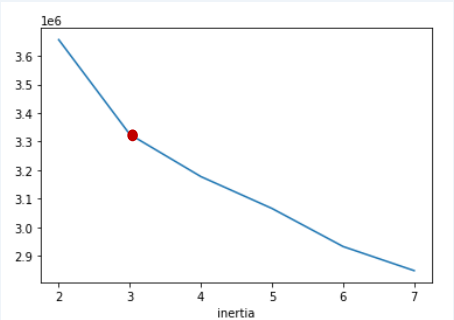
- 1. XGBoost를 기준으로
피쳐 중요도 높은
상위 28개의 변수 추출
- 2. 선택한 28개의 변수 중,
null값이 50%가 넘어가는
변수 삭제
- 3. null값이 하나라도 포함된
행 삭제

최종 사용 데이터 소개 (244291, 29)

전처리

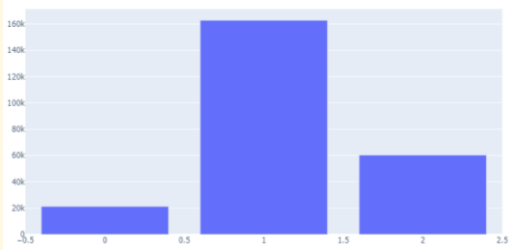
- 범주형 변수:
One-hot encoding
- 수치형 변수:
군집화 결과가 가장 좋았던
Standard Scaling을 사용
한 데이터로 군집화 진행

최적 k 설정



Elbow Point인 K=3 에서
군집 개수 설정

Kmeans 군집화 결과



총 3개의 군집 생성

→ 군집별 특성 추출

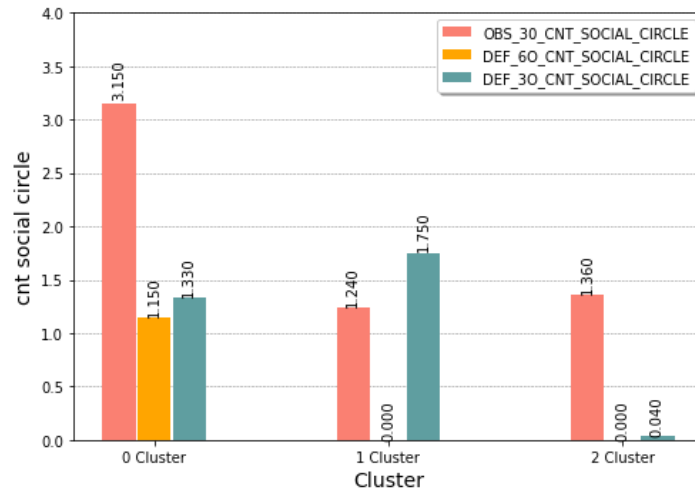
- 수치형 변수: 중간값으로 군집 특성 파악
- 범주형 변수: 최빈값으로 군집 특성 파악

TARGET	FLAG_OWN_CAR	FLAG_OWN_REALTY	FLAG_EMP_PHONE	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL	FLAG_DOCUMENT_3	FLAG_DOCUMENT_6	FLAG_DOCUMENT_8
납부 여부	자동차 소유 여부	주택 소유 여부	직장 전화 제공 여부	집전화 제공 여부	집전화 제공 여부	Email 제공 여부	문서3 제공여부	문서6 제공여부	문서8 제공여부
EXT_SOURCE_2	EXT_SOURCE_3	CNT_CHILDREN	CNT_FAM_MEMBERS	OBS_30_CNT_SOCIAL_CIRCLE	...	DEF_60_CNT_SOCIAL_CIRCLE	AMT_REQ_CREDIT_BUREAU_MON	AMT_REQ_CREDIT_BUREAU_YEAR	AMT_REQ_CREDIT_BUREAU_QRT
외부 데이터의 정규화 점수_2	외부 데이터의 정규화 점수_3	자녀 수	가족 구성원 수	고객 주변 30 DPD 수		고객 주변 60 DPD 수	신청 1개월 전 문의 건수	신청 1년 전 문의 건수	신청 3개월 전 문의 건수

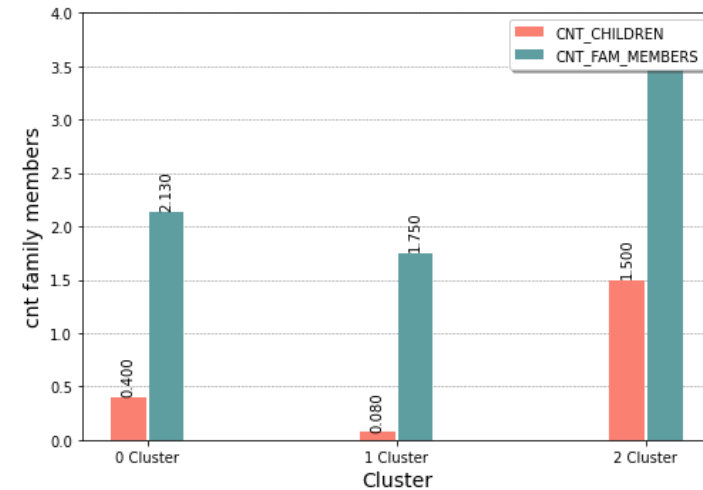
4. 클러스터링 결과

	0번 군집	1번 군집	2번 군집
가족 구성원 수	<ul style="list-style-type: none"> - 세 집단 중, 두 번째로 가족 구성원 수 많음 - 세 집단 중, 두 번째로 자녀 수 많음 	<ul style="list-style-type: none"> - 세 집단 중, 가족 구성원 수와 자녀 수 모두 가장 적음 	<ul style="list-style-type: none"> - 세 집단 중, 가족 구성원 수 가장 많음 - 자녀 수도 가장 많음
군집 주변 연체자 수	<ul style="list-style-type: none"> - 세 집단 중, 연체된 지인의 수가 가장 많음 	<ul style="list-style-type: none"> - 해당 군집의 지인 중, 연체된 지인 거의 없음 	<ul style="list-style-type: none"> - 해당 군집의 지인들 중, 연체된 지인 아주 조금 있음
기한 내 납부 여부	<ul style="list-style-type: none"> - 세 집단 중, 기한 내 납부할 확률 가장 작음 	<ul style="list-style-type: none"> - 세 집단 중, 기한 내 납부할 확률 가장 큼 	<ul style="list-style-type: none"> - 세 집단 중, 기한 내 납부할 확률 두 번째로 큼
군집화 결과	<ul style="list-style-type: none"> - 고객 주변에 연체된 지인이 많을 수록 기한내 납부하지 못할 확률 높아짐 (→ 0번 군집) - 고객 주변에 연체된 지인 많지 않더라도, 가족 구성원 수가 많으면 기한내 납부하지 못할 확률 높아질 가능성 있음 (→ 2번 군집) - 고객 주변에 연체된 지인이 거의 없고, 가족 구성원 수가 많지 않다면 기한 내 납부하지 못할 확률 0에 수렴함 (→ 1번 군집) 		

*모든 비교는 군집의 평균값을 기준으로 함

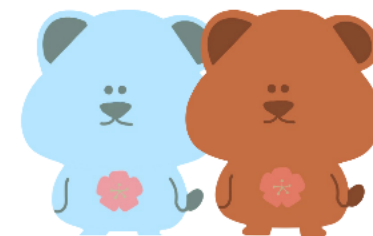


<군집과 군집 주변 연체자 수>




<군집과 가족 구성원 수>

05. 한계점 및 마무리



1. 한계점 및 마무리

변수 구분의 어려움	원본 데이터의 칼럼이 122개가 넘어가기 때문에 모든 columns 파악하지 못함
	모델 학습 시, 변수의 데이터 타입을 기준으로(float vs. object) 수치형과 범주형 변수로 구분함
	데이터가 float형인 범주형 변수가 수치형 변수로 인식되는 문제 발생
	AutoML 사용으로 위의 문제 해결
	다만, PCA와 over sampling 데이터에 대해서는 AutoML 적용하지 못함
클러스터링의 어려움	columns 수가 너무 많아, 유의미한 변수 확인 어려움 & Feature Importance 해석의 어려움
	심한 불균형 데이터
	<div><div><div>Any Success with Clustering?</div><div>Posted in home-credit-default-risk 4 years ago</div></div><div></div><div><div>SteveKane • (179th in this Competition) • 4 years ago • Options • Report • Reply</div><div>just failure and tears.</div></div></div>

THANK YOU

