신용카드 사용자 데이터를 활용한 신용카드 연체 예측

2019251126

정보통계학과

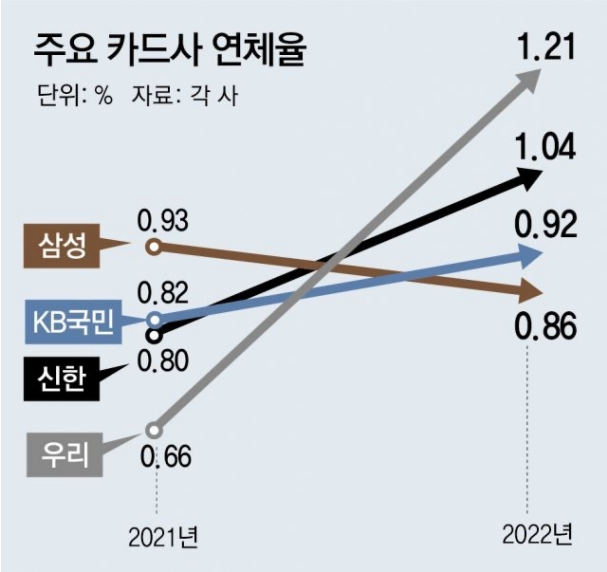
황성아

Index

1. Introduction

분석 주제 이해

"급속하게 성장하고 있는 신용카드 시장, 연체율도 함께 성장 중"



높아지는 연체율에…금융사들 '역대급' 충당금 쌓는다

카드도 증권도 '연체율 경고등'…

후불결제 1년 연체율 증가세… 네·카·토, 리스크 관리 나서

카드업계 1분기 실적 '부진'…작년보다 순익 28% 감소

신용카드 연체는 고객 뿐만 아니라 카드 회사에서도 이자 및 연체료 지불 문제, 채권 회수 문제, 운영 비용 증가, 평판 악화 등의 다양한 문제 발생

신용카드 연체에 어떠한 요인이 밀접한 연관이 있는지 알아보아 신용카드 발급 기준을 효율적으로 설정하며, 정확한 신용카드 연체 예측 모델 구축 필요

2. Data

데이터 설명

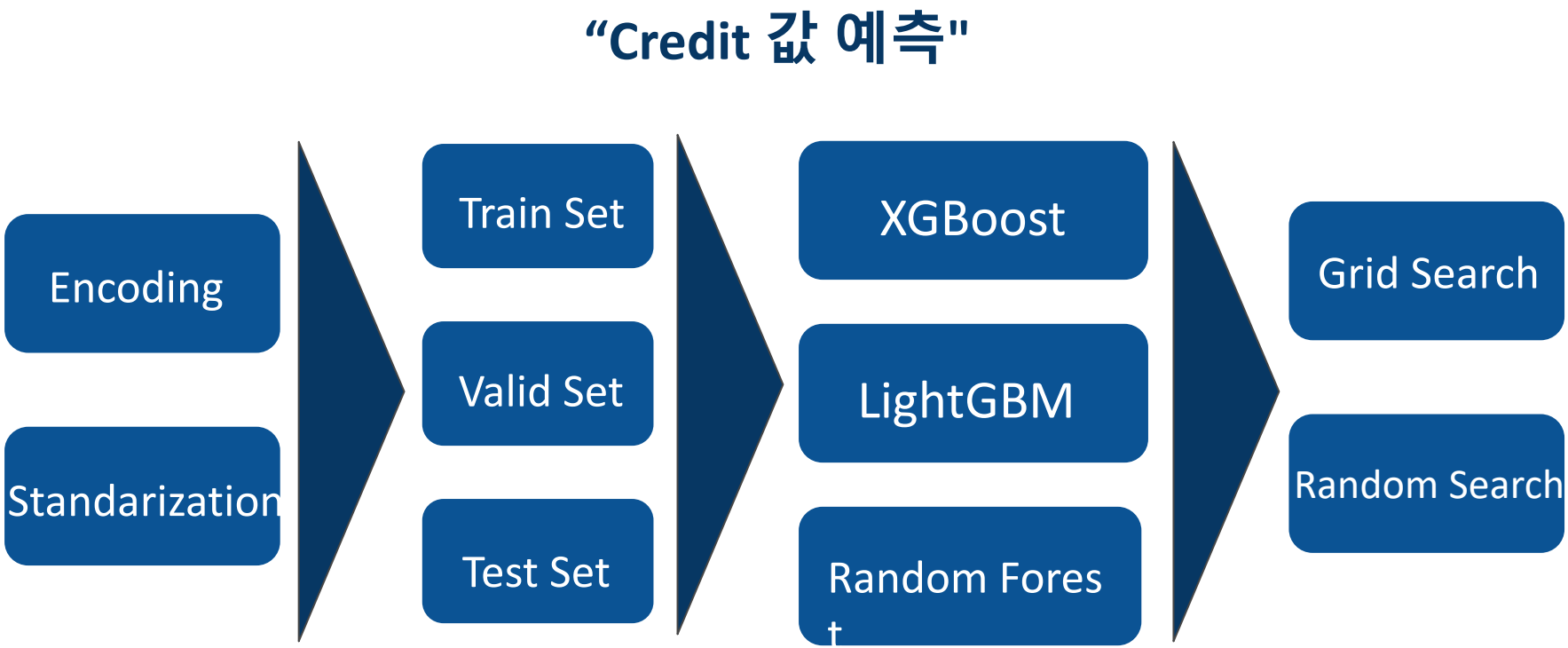
“ 신용카드 사용자들의 개인 정보 데이터” - 20개 변수, 26457개 행

변수	변수 설명	변수	변수 설명
index	인덱스	days_birth	출생일
gender	성별	days_employed	업무 시작일
car	차량 소유 여부	flag_mobil	핸드폰 소유 여부
reality	부동산 소유 여부	work_phone	업무용 전화 소유 여부
child_num	자녀 수	phone	전화 소유 여부
income_total	연간 소득	email	이메일 소유 여부
income_type	소득 분류	occyp_type	직업 유형
edu_type	교육 수준	family_size	가족 규모
family_type	결혼 여부	begin_month	신용카드 발급 월
house_type	생활 방식	**credit**	**신용도**(신용카드 대금 연체 기준)

출처: Dacon

3. Methods

모델링 절차



* 시각화와 전처리 후 진행

알고리즘 소개

XGBoost

- 머신러닝 기법들 중 뛰어난 성능
- 과적합 규제 기능
- 병렬 CPU 환경에서의 학습
- 내장된 교차검증

LightGBM

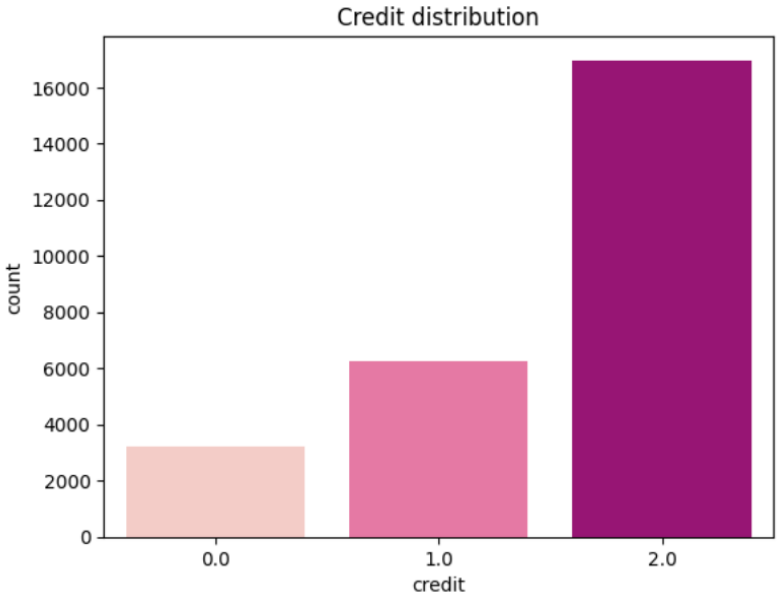
- 머신러닝 기법들 중 뛰어난 성능
- XGBoost 대비 더 빠른 수행 시간
- XGBoost 대비 더 작은 메모리 사용
- 범주형 변수의 자동 변환과 최적 분할

Random Forest

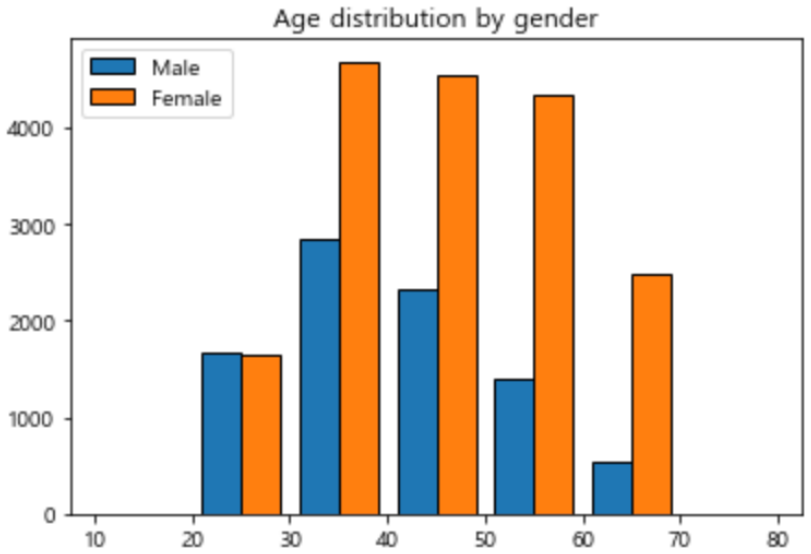
- 앙상블 알고리즘(XGBoost등) 중 비교적 빠른 수행 속도
- 기본 결정 트리보다 일반화
- 룰이 명확하고 트리 별 시각화 가능

4. Experimental Results

데이터 시각화

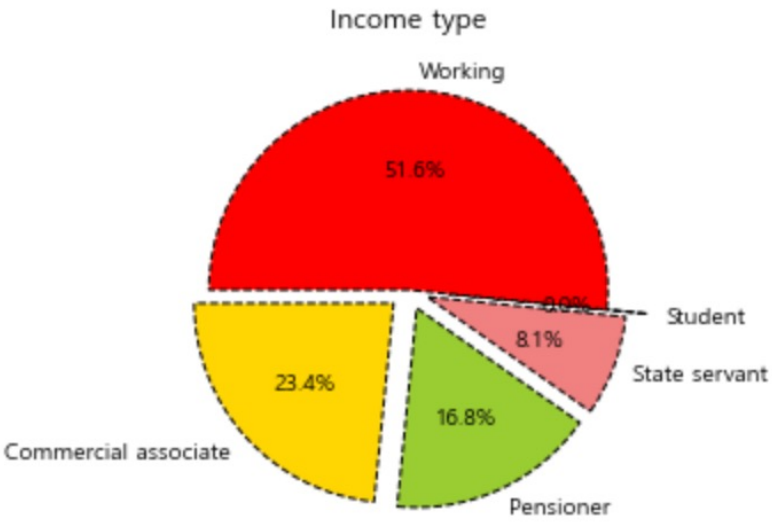


0일 수록 신용이 높은 사람을 나타내며,
고 신용자의 수가 가장 낮고 저 신용자
의 수가 가장 많은 편이다.

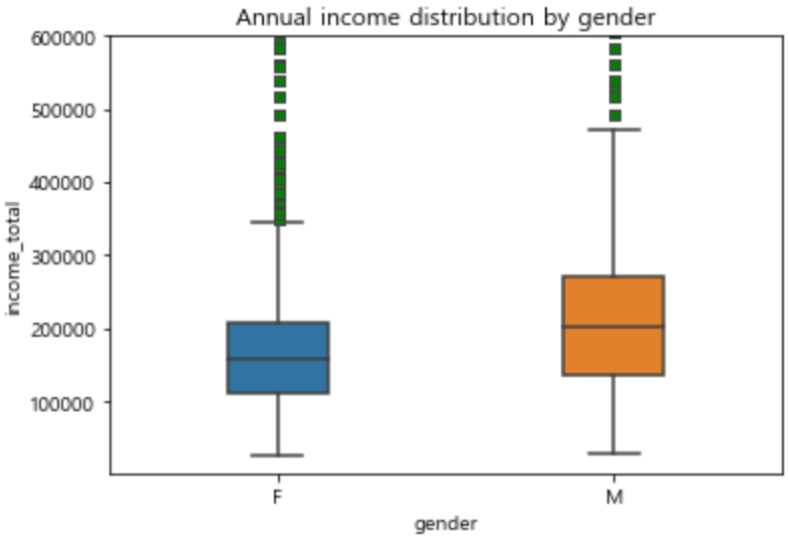


남성이 여성보다 약 2배 많고,
30~40대에 가장 많이 분포해 있다.

데이터 시각화



고객 대부분은 수입이 있는 상태이며,
학생의 수는 거의 없다.



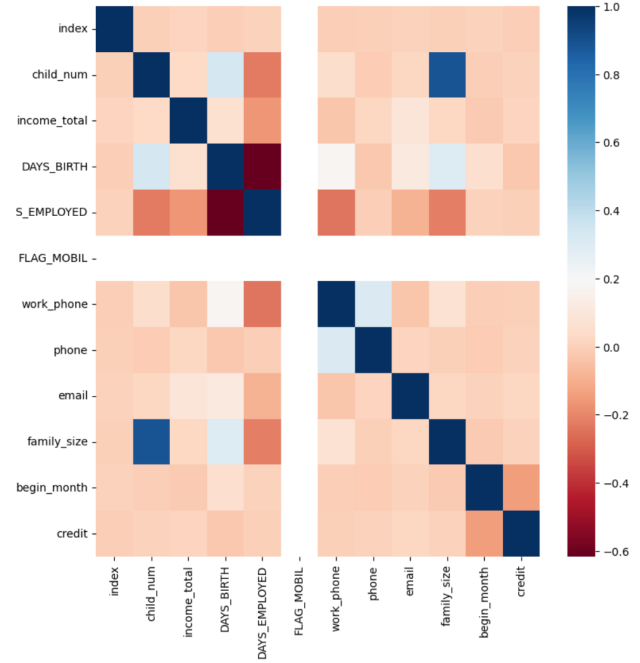
남성이 여성보다 연간 소득이 높은 편
이고, 여성은 주로 100000~200000,
남성은 주로 150000~300000 구간에
속해 있다.

데이터 전처리

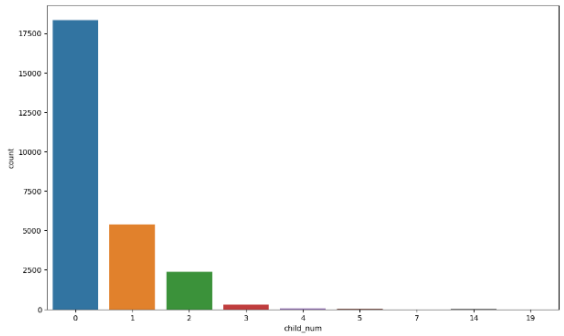
[의미 없는 변수 제거]

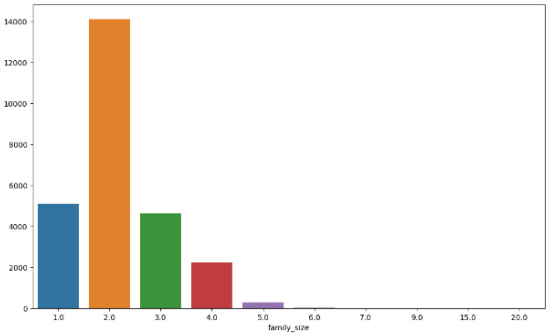
```
train.drop(['index', 'FLAG_MOBIL'], axis=1, inplace=True)
```

flag_mobil 값은 모두 1로 동일함



[이상치 처리 및 파생변수 1 생성]






```
train.loc[train['child_num'] >= 3,'child_num'] = 3
train.loc[train['family_size'] >= 5,'child_num'] = 5
```

```
train['famchild_sum'] = train[['child_num', 'family_size']].sum(axis=1)
```

child_num의 변수의 이상치는 3으로, family size의 이상치는 5로 변경 후,
두 변수의 상관관계가 높아 하나의 변수 famchild_sum으로 합쳐 파생변수 생성

데이터 전처리

[DAYS_EMPLOYED가 양수이면 무직으로 판단, 0 처리]

```
train['DAYS_EMPLOYED'] = train['DAYS_EMPLOYED'].map(lambda x: 0 if x > 0 else x)
```

[음수 값-> 양수 값 및 파생변수 2 생성]

```
feats = ['DAYS_BIRTH', 'begin_month', 'DAYS_EMPLOYED']
for feat in feats:
    train[feat]=np.abs(train[feat])
```

```
train['before_EMPLOYED'] = train['DAYS_BIRTH'] - train['DAYS_EMPLOYED']
```

데이터 수집 당시를 0이라 두고 날짜를 역으로 세어 값이 음수로 표현되어 있는 변수들을 양수 값으로 변경 후, 상관관계가 높은 DAYS_BIRTH와 DAYS_EMPLOYED의 차이 값으로 before_EMPLOYED 파생변수 생성

[결측 값 처리]

```
ocp = train[train['occyp_type'].isnull()]
ocp[ocp['DAYS_EMPLOYED'] == 0]
```

```
ocp['occyp_type']=['inoccupation' if s==0 else 'NaN' for s in ocp['DAYS_EMPLOYED']]
ocp=ocp[ocp['occyp_type']=='inoccupation']
```

occyp_type의 결측 값은 days_employed에서 무직이라고 판단한 데이터를 무직(inoccupation)이라는 카테고리를 만들고 할당하여 결측 값을 채우고 나머지 결측 값은 데이터 오류라 판단하여 삭제

XGBoost

[정확도: 0.7031]

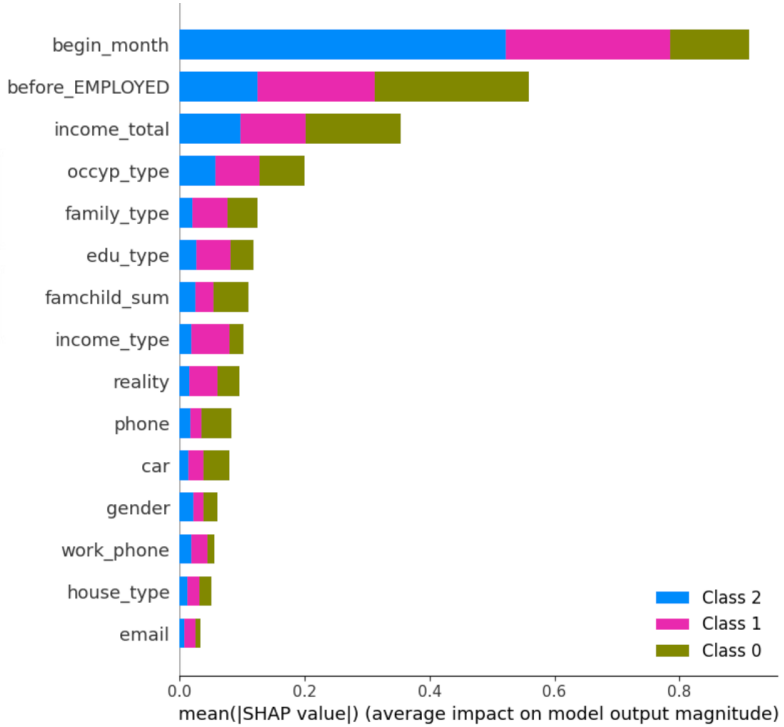
```
xgb = XGBClassifier(random_state = 42)
xgb.fit(X_train, y_train)
xgb_preds = xgb.predict(X_test)
```

```
#정확도
print('accuracy:{:0.4f}'.format(accuracy_score(y_test, xgb_preds)))
```

```
accuracy:0.7031
```

begin_month, before_EMPLOYED, income_total 변수의
변수 중요도가 높음

[SHAP Value]



LightGBM

[정확도: 0.7004]

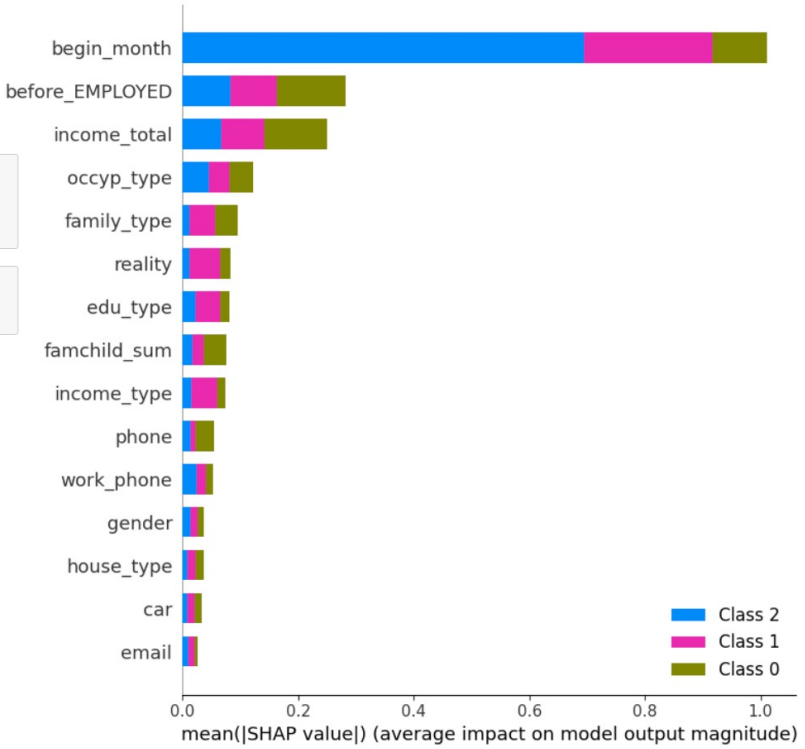
```
lgb = LGBMClassifier(random_state=42)
lgb.fit(X_train, y_train)
lgb_preds = lgb.predict(X_test)
```

```
#정확도
print('accuracy:{:0.4f}'.format(accuracy_score(y_test, lgb_preds)))
```

```
accuracy:0.7004
```

begin_month, before_EMPLOYED, income_total 변수의
변수 중요도가 높음

[SHAP Value]



Random Forest

[정확도: 0.7058]

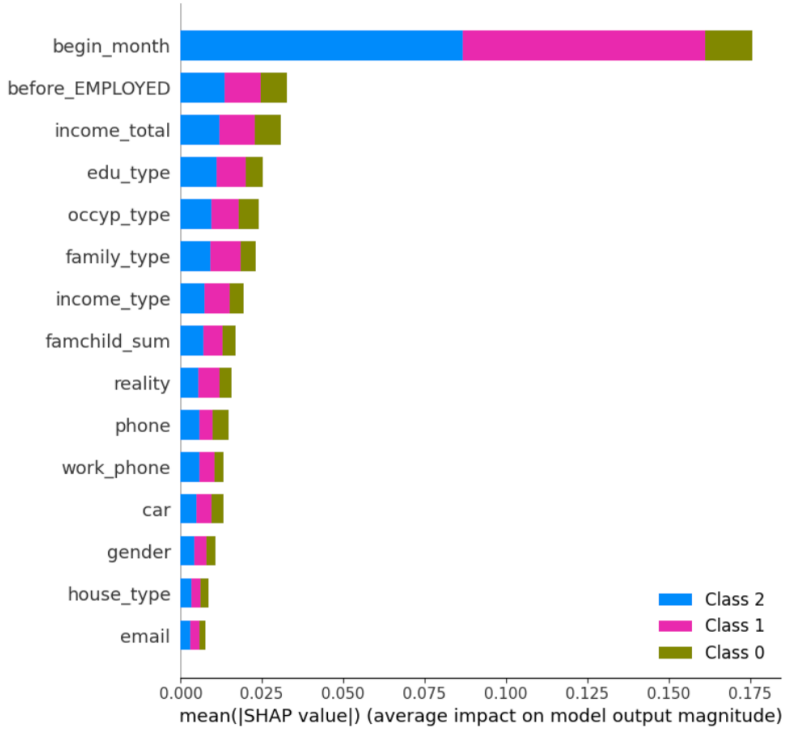
```
rf = RandomForestClassifier(n_estimators=3000,
                            max_depth= 16,
                            random_state = 42)
rf.fit(X_train, y_train)
rf_preds = rf.predict(X_test)
```

```
#정확도
print('accuracy:{:0.4f}'.format(accuracy_score(y_test, rf_preds)))
```

```
accuracy:0.7058
```

begin_month, before_EMPLOYED, income_total 변수의
변수 중요도가 높음

[SHAP Value]



모델 비교 및 선정

"정확도 비교"

	XGBoost	LightGBM	Random Forest
정확도	0.7031	0.7004	0.7058

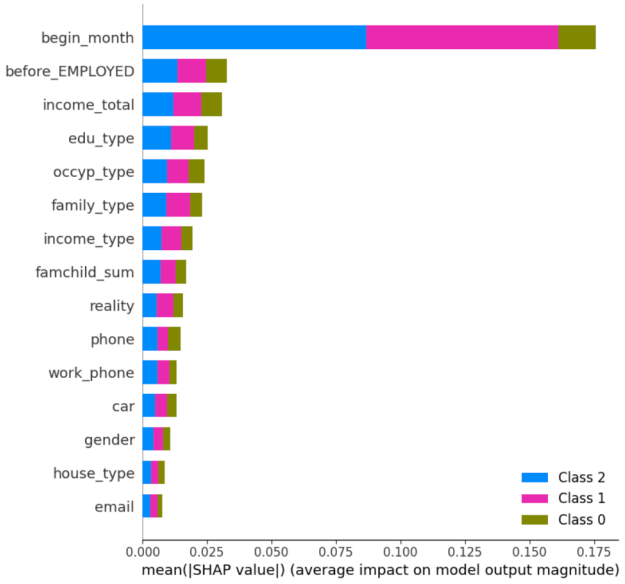
[Random Forest]

```
rf = RandomForestClassifier(n_estimators=3000,
                            max_depth= 16,
                            random_state = 42)
rf.fit(X_train, y_train)
rf_preds = rf.predict(X_test)
```

```
#정확도
print('accuracy:{:0.4f}'.format(accuracy_score(y_test, rf_preds)))
```

```
accuracy:0.7058
```

begin_month, before_EMPLOYED, income_total 변수의 변수 중요도가 높음



5. Discussion & Conclusuins

활용 방안 빛 기대 효과

“신용카드 연체 예측을 통한 예산안의 효율화"

· 위의 분석을 바탕으로 연체 가능성이 높은 고객을 식별하여 리스크 관리 및 대응.

· 연체로 인하여 발생하는 비용 절감.

· 연체 관리 서비스의 질적인 발전을 통하여 사용자의 고객 서비스 개선.

· 고객 신용 위험을 더 잘 관리하여 시장 경쟁력 강화.

신용카드 연체를 사전에 방지하여 연체율이 감소하고
신용카드 회사들이 겪고 있는 다양한 문제점들이 최소화되기를 기대함.

감사합니다.

〈References〉

- *데이터 변수 설명*. DACON. (n.d.). Retrieved May 5, 2023, from https://www.dacon.io/competitions/official/235713/talkboard/402821/
- Newsis. (2023, May 2). *높아지는 연체율에...금융사들'역대급' 충당금 쌓는다*. newsis. Retrieved from https://newsis.com/view/?id=NISX20230502_0002288266&cID=15001&pID=15000
- 이혜승. (2004). 데이터 마이닝 기법을 이용한 신용카드 연체고객 예측 및 감소 방안 제시 (국내석사학위논문). 고려대학교 대학원, 서울.
- Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.2939785
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, *30*, 3146–3154.