

신용카드 사용자 연체 예측 AI 경진대회

TEAM3 (2조) 팀프로젝트

안동현, 김태용, 안준용, 이문형, 이종섭

목차

1. 팀 소개
2. 프로젝트 개요 및 진행
3. 데이터 확인 및 문제 파악
4. 문제점 및 해결방안 모색

1) 결측치 문제 해결

2) 불균형 문제 해결

3) 중복 데이터 문제

1. 전처리 및 특성 공학

1) 도메인 지식 관련

2) 수치형 데이터 전처리

3) 범주형 데이터 전처리

1. 모델링

2. 최종 결과

3. 시사점 및 개선 방향

1. 팀 소개

팀명 : TEAM3

팀장 : 안동현

팀원 : 김태용, 안준용, 이문형, 이종섭



안동현



김태용



안준용



이문형



이종섭

2. 프로젝트 개요 및 진행

❑ 프로젝트 개요

데이콘 - 신용카드 사용자 연체 예측 AI 경진대회

❑ 관련 링크 : <https://dacon.io/competitions/official/235713/>

❑ 분석개요

- 신용 카드 **고객의 카드 발급 정보로 신용도를** 예측함
(연간 소득, 직업, 가족 구성, 부동산 보유 여부 등)
- 연체와 관련된 과거 이력 데이터가 없으므로,
에 대한 선행 지표를 도출하는 것과 관련이 있음
- **한정된 변수 및 데이터를 기반으로 예측 모델을 만들기 때문에,
실제 신용도 예측과는 다소 차이가 있을 수 있음**



2. 프로젝트 개요 및 진행

□ 분석 필요성

- 신용등급 정보 적시성 저하
- 정성적 평가에 있어 객관적 근거 제시 미흡
- 신용도와 비선형적인 관계를 가지는 요소에 대한 부정확한 판단
- 기업의 입장에서 신용도가 불량한 사람을 미리 예측
- 이익 극대화 및 손실 최소화 실현



2. 프로젝트 개요 및 진행

□ 프로젝트 진행

2021.05.24



3. 데이터 확인 및 문제 파악

□ 정량적 분석

- 훈련 데이터 셋 : 26,457 rows X (18 features + 1 target)

```
[124] train.head(3)
```

	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	occyp_type	family_size	begin_month	credit
index																			
0	F	N	N	0	202500.0	Commercial associate	Higher education	Married	Municipal apartment	-13899	-4709	1	0	0	0	NaN	2.0	6.0	1.0
1	F	N	Y	1	247500.0	Commercial associate	Secondary / secondary special	Civil marriage	House / apartment	-11380	-1540	1	0	0	1	Laborers	3.0	5.0	1.0
2	M	Y	Y	0	450000.0	Working	Higher education	Married	House / apartment	-19087	-4434	1	0	1	0	Managers	2.0	22.0	2.0

- 평가 데이터 셋 : 10,000 rows X (18 features)

```
[125] test.head(3)
```

	index	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	occyp_type	family_size	begin_month
0	26457	M	Y	N	0	112500.0	Pensioner	Secondary / secondary special	Civil marriage	House / apartment	-21990	365243	1	0	1	0	NaN	2.0	-60.0
1	26458	F	N	Y	0	135000.0	State servant	Higher education	Married	House / apartment	-18964	-8671	1	0	1	0	Core staff	2.0	-36.0
2	26459	F	N	Y	0	69372.0	Working	Secondary / secondary special	Married	House / apartment	-15887	-217	1	1	1	0	Laborers	2.0	-40.0

3. 데이터 확인 및 문제 파악

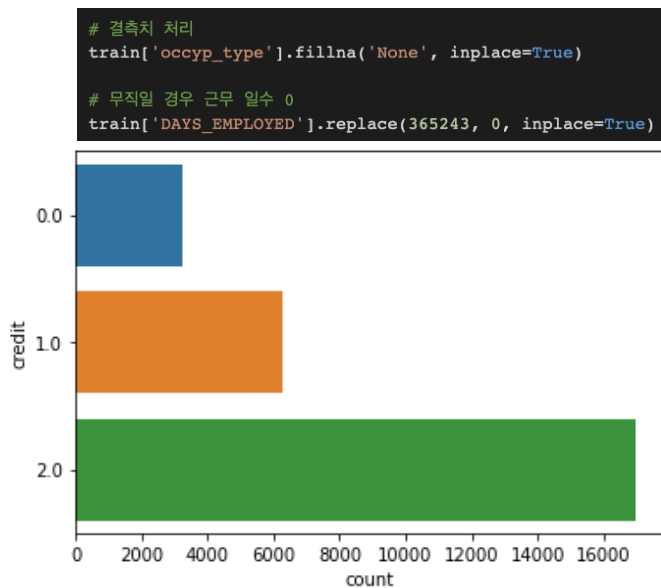
□ 데이터 특성

특성 이름	설명	데이터 유형	특성 이름	설명	데이터 유형
gender	성별	binary	DAYS_BIRTH	나이 (일수로 계산)	numeric
car	차량 소유 여부	binary	DAYS_EMPLOYED	근속일 수	numeric
reality	부동산 소유 여부	binary	FLAG_MOBIL	핸드폰 소유 여부	binary
child_num	자녀 수	numeric	work_phone	업무용 전화 소유 여부	binary
income_total	연간 소득	numeric	phone	전화 소유 여부	binary
income_type	소득 분류	category	email	이메일 소유 여부	binary
edu_type	교육 수준	category	occyp_type	직종	category
family_type	결혼 여부	category	family_size	가족 규모	numeric
house_type	거주 형태	category	begin_month	신용카드 이용 기간	numeric

3. 데이터 확인 및 문제 파악

□ 훈련 데이터 분포 시각화

- 레이블 별 데이터 불균형
- 각 특성 별 데이터 분포



3. 데이터 확인 및 문제 파악

□ 훈련 데이터 결측치 확인

- 누락 데이터 : occyp_type 특성 / 8,171개

```
raw_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 26457 entries, 0 to 26456  
Data columns (total 19 columns):  
#   Column             Non-Null Count  Dtype    
---  --  
0   gender              26457 non-null  object   
1   car                 26457 non-null  object   
2   reality             26457 non-null  object   
3   child_num           26457 non-null  int64    
4   income_total        26457 non-null  float64  
5   income_type         26457 non-null  object   
6   edu_type            26457 non-null  object   
7   family_type         26457 non-null  object   
8   house_type          26457 non-null  object   
9   DAYS_BIRTH          26457 non-null  int64    
10  DAYS_EMPLOYED        26457 non-null  int64    
11  FLAG_MOBIL           26457 non-null  int64    
12  work_phone           26457 non-null  int64    
13  phone                26457 non-null  int64    
14  email                26457 non-null  int64    
15  occyp_type           18286 non-null  object   
16  family_size          26457 non-null  float64  
17  begin_month          26457 non-null  float64  
18  credit               26457 non-null  float64  
dtypes: float64(4), int64(7), object(8)  
memory usage: 4.0+ MB
```

```
raw_train.isnull().sum()
```

```
gender      0  
car          0  
reality     0  
child_num   0  
income_total 0  
income_type 0  
edu_type     0  
family_type 0  
house_type   0  
DAYS_BIRTH   0  
DAYS_EMPLOYED 0  
FLAG_MOBIL   0  
work_phone   0  
phone        0  
email        0  
occyp_type   8171  
family_size  0  
begin_month  0  
credit       0  
dtype: int64
```

3. 데이터 확인 및 문제 파악

□ 평가 데이터 결측치 확인

- 누락 데이터 : occyp_type 특성 / 3,152개

```
[122] test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 19 columns):
#   Column              Non-Null Count  Dtype
---  ---
0   index               10000 non-null  int64
1   gender              10000 non-null  object
2   car                 10000 non-null  object
3   reality             10000 non-null  object
4   child_num           10000 non-null  int64
5   income_total        10000 non-null  float64
6   income_type         10000 non-null  object
7   edu_type            10000 non-null  object
8   family_type         10000 non-null  object
9   house_type          10000 non-null  object
10  DAYS_BIRTH          10000 non-null  int64
11  DAYS_EMPLOYED       10000 non-null  int64
12  FLAG_MOBIL          10000 non-null  int64
13  work_phone          10000 non-null  int64
14  phone               10000 non-null  int64
15  email               10000 non-null  int64
16  occyp_type           6848 non-null   object
17  family_size          10000 non-null  float64
18  begin_month          10000 non-null  float64
dtypes: float64(3), int64(8), object(8)
memory usage: 1.4+ MB
```

```
[121] test.isnull().sum()
```

```
index                0
gender                0
car                  0
reality              0
child_num            0
income_total         0
income_type          0
edu_type             0
family_type          0
house_type           0
DAYS_BIRTH           0
DAYS_EMPLOYED        0
FLAG_MOBIL           0
work_phone           0
phone                0
email                0
occyp_type           3152
family_size          0
begin_month          0
dtype: int64
```

3. 데이터 확인 및 문제 파악

❑ 중복 데이터 이슈

- Target을 포함한 모든 열의 데이터가 일치하는 경우
- 3,155개의 완전 중복 데이터
- 해석 : 사용자가 복수 개의 카드를 신청하여 같은 신용도를 얻음

```
train [ train.duplicated(train.columns, keep=False) ]
```

	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	occyp_type	family_size	begin_month	credit
index																			
19	F	N	Y	0	180000.0	Working	Secondary / secondary special	Married	House / apartment	-13727	-6031	1	0	0	0	None	2.0	-7.0	2.0
21	F	N	N	0	157500.0	Pensioner	Secondary / secondary special	Married	House / apartment	-21253	0	1	0	1	0	None	2.0	-7.0	2.0
29	F	N	Y	0	121500.0	Commercial associate	Secondary / secondary special	Married	Co-op apartment	-12017	-1711	1	0	1	0	Sales staff	2.0	-22.0	0.0
48	F	N	Y	0	99000.0	Pensioner	Secondary / secondary special	Married	House / apartment	-23585	0	1	0	0	0	None	2.0	-4.0	2.0
56	F	Y	Y	0	130500.0	Working	Secondary / secondary special	Married	House / apartment	-16137	-9391	1	0	1	0	Laborers	2.0	-29.0	2.0
...
26374	F	Y	N	0	112500.0	Working	Secondary / secondary special	Married	Municipal apartment	-17372	-978	1	0	1	0	Sales staff	2.0	-15.0	1.0
26393	M	Y	Y	0	247500.0	Working	Secondary / secondary special	Married	House / apartment	-14122	-3383	1	0	0	0	Managers	2.0	-31.0	2.0
26428	F	Y	Y	2	270000.0	Working	Secondary / secondary special	Civil marriage	House / apartment	-12745	-525	1	0	0	1	Core staff	4.0	-23.0	1.0
26446	F	N	Y	0	135000.0	Working	Secondary / secondary special	Civil marriage	House / apartment	-16300	-9698	1	0	0	1	Managers	2.0	-41.0	2.0
26451	F	N	Y	0	202500.0	Working	Higher education	Married	House / apartment	-12831	-803	1	1	1	0	Accountants	2.0	-44.0	1.0

3155 rows x 19 columns

3. 데이터 확인 및 문제 파악

□ 중복 데이터 이슈

- Target을 제외한 모든 열의 데이터가 일치하는 경우
- 4,497개의 중복 데이터
- 해석 : 어떤 사용자가 같은 달에 카드를 복수개 발급 받은 경우
3,155개 : 동일한 신용도 부여 / 1,342개 : 상이한 신용도 부여 (noise)

```
[22] train.iloc[:, 1-1]
train [ train.iloc[:, 1-1].duplicated(train.iloc[:, 1-1].columns, keep=False) ]
```

index	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	occyp_type	family_size	begin_month	credit
1	F	N	Y	1	247500.0	Commercial associate	Secondary / secondary special	Civil marriage	House / apartment	-11380	-1540	1	0	0	1	Laborers	3.0	-5.0	1.0
2	M	Y	Y	0	450000.0	Working	Higher education	Married	House / apartment	-19087	-4434	1	0	1	0	Managers	2.0	-22.0	2.0
19	F	N	Y	0	180000.0	Working	Secondary / secondary special	Married	House / apartment	-13727	-6031	1	0	0	0	None	2.0	-7.0	2.0
21	F	N	N	0	157500.0	Pensioner	Secondary / secondary special	Married	House / apartment	-21253	0	1	0	1	0	None	2.0	-7.0	2.0
24	F	N	N	0	202500.0	Pensioner	Secondary / secondary special	Single / not married	House / apartment	-22361	0	1	0	1	0	None	1.0	-5.0	2.0
...
26430	F	N	Y	0	112500.0	Working	Incomplete higher	Civil marriage	House / apartment	-9301	-1751	1	0	0	0	None	2.0	-19.0	2.0
26431	F	N	Y	0	225000.0	Pensioner	Secondary / secondary special	Widow	House / apartment	-21151	0	1	0	0	1	None	1.0	-60.0	1.0
26432	F	Y	Y	0	72000.0	Pensioner	Secondary / secondary special	Married	House / apartment	-22314	0	1	0	1	0	None	2.0	-17.0	1.0
26446	F	N	Y	0	135000.0	Working	Secondary / secondary special	Civil marriage	House / apartment	-16300	-9698	1	0	0	1	Managers	2.0	-41.0	2.0
26451	F	N	Y	0	202500.0	Working	Higher education	Married	House / apartment	-12831	-803	1	1	1	0	Accountants	2.0	-44.0	1.0

4497 rows × 19 columns

3. 데이터 확인 및 문제 파악

❑ 중복 데이터 이슈

- Target과 begin_month 특성을 제외한 데이터가 일치하는 경우
- 23,208 개의 중복 데이터
- 해석 : 카드를 두 개 이상 발급한 이용자의 모든 기록
신용카드 발급 기간에 따라 신용도가 측정된 결과가 포함

```
[23] train.iloc[:, :-2]  
train [ train.iloc[:, :-2].duplicated(train.iloc[:, :-2].columns, keep=False) ]
```

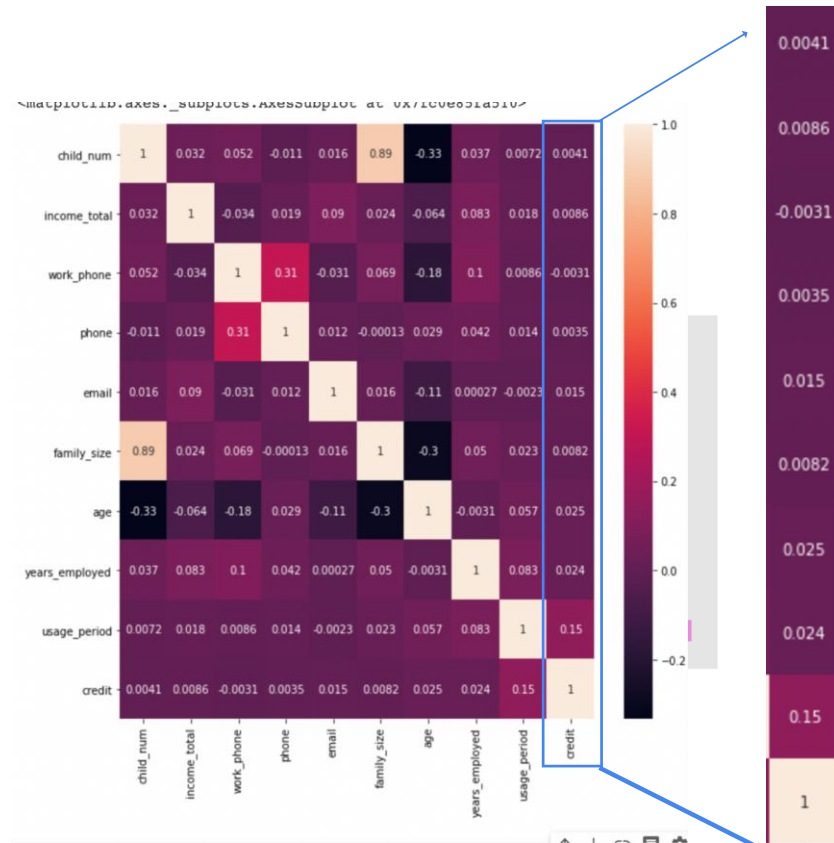
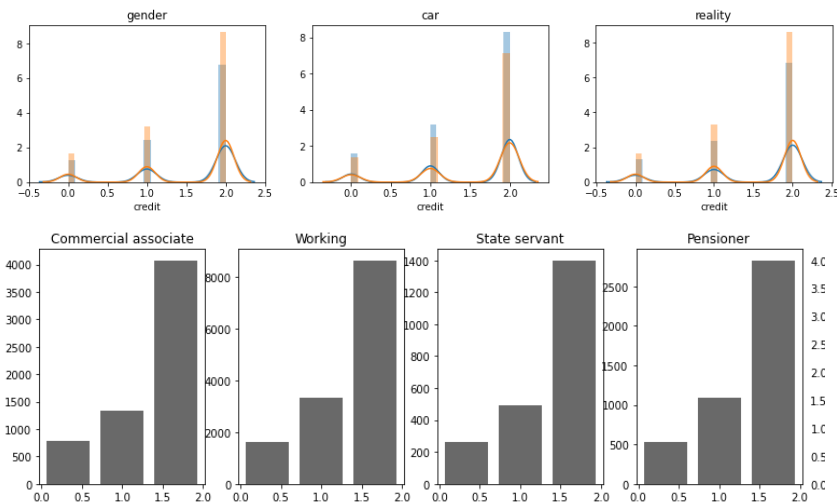
	gender	car	reality	child_num	income_total	income_type	edu_type	family_type	house_type	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	work_phone	phone	email	occyp_type	family_size	begin_month	credit
index																			
0	F	N	N	0	202500.0	Commercial associate	Higher education	Married	Municipal apartment	-13899	-4709	1	0	0	0	None	2.0	-6.0	1.0
1	F	N	Y	1	247500.0	Commercial associate	Secondary / secondary special	Civil marriage	House / apartment	-11380	-1540	1	0	0	1	Laborers	3.0	-5.0	1.0
2	M	Y	Y	0	450000.0	Working	Higher education	Married	House / apartment	-19087	-4434	1	0	1	0	Managers	2.0	-22.0	2.0
3	F	N	Y	0	202500.0	Commercial associate	Secondary / secondary special	Married	House / apartment	-15088	-2092	1	0	1	0	Sales staff	2.0	-37.0	0.0
6	F	N	N	0	315000.0	Working	Secondary / secondary special	Separated	House / apartment	-17570	-1978	1	0	0	1	Core staff	1.0	-41.0	2.0
...
26447	M	N	Y	2	99000.0	Working	Secondary / secondary special	Married	House / apartment	-14226	-1026	1	1	1	0	Laborers	4.0	-43.0	2.0
26448	M	N	Y	0	292500.0	Commercial associate	Higher education	Married	House / apartment	-16280	-887	1	0	0	0	Laborers	2.0	-23.0	0.0
26449	F	N	N	0	90000.0	Working	Secondary / secondary special	Married	House / apartment	-10498	-2418	1	1	1	0	None	2.0	-2.0	1.0
26451	F	N	Y	0	202500.0	Working	Higher education	Married	House / apartment	-12831	-803	1	1	1	0	Accountants	2.0	-44.0	1.0
26452	F	N	N	2	225000.0	State servant	Secondary / secondary special	Married	House / apartment	-12079	-1984	1	0	0	0	Core staff	4.0	-2.0	1.0

23208 rows x 19 columns

3. 데이터 확인 및 문제 파악

□ 데이터 문제 요약

- 특성과 레이블 간 낮은 상관관계



Correlation between Credit & 18 features

4. 문제점 및 해결방안 모색 1) 결측치 문제 해결

- <occyp_type> 특성의 결측치 데이터
- 문제 해결 방안
 - a. 데이터 제거
 - b. 새로운 값으로 분류
 - '미입력' 데이터로 분류
 - a. 최빈값 대치
 - <income_type> 분류에 맞는 <occyp_type> 최빈값 채택
 - a. ML 예측값 대치
 - Random Forest Classifier, knn
 - a. 범주 축소
 - b. 혼합하여 적용

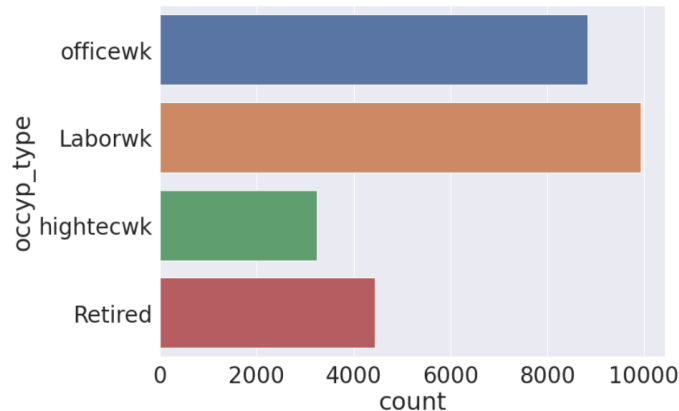
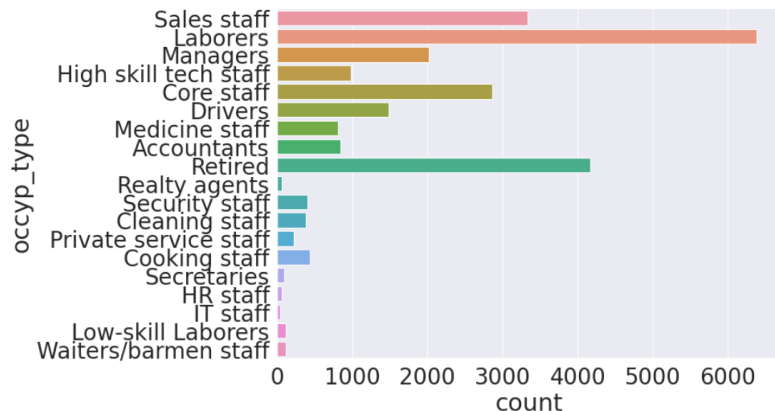
4. 문제점 및 해결방안 모색 1) 결측치 문제 해결

❑ 최빈값 대치

- <income_type> 분류에 맞는 <occyp_type> 최빈값 채택
- <DAYS_EMPLOYED> 값이 0을 갖는 경우 새로운 분류 추가 : 은퇴한 고객

❑ 범주 축소

- 값 대치 부작용 최소화
- 데이터 불균형 해소 기대



4. 문제점 및 해결방안 모색 1) 결측치 문제 해결

- <occyp_type>의 누락 데이터를 기계 학습을 통한 예측 값으로 대체
 - 훈련 데이터
기존 훈련 / 평가 데이터 중 <occyp_type> 값이 누락되지 않고,
<DAYS_EMPLOYED> 값이 0이 아닌 데이터
 - 평가 데이터
기존 훈련 / 평가 데이터 중 <occyp_type> 값이 누락되고,
<DAYS_EMPLOYED> 값이 0이 아닌 데이터
 - Random Forest Classifier 모델 사용
<occyp_type> 특성과 상관 관계가 높은 상위 10개 특성 이용
 'income_type', 'work_phone', 'house_type', 'edu_type', 'child_num',
 'gender', 'car', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'income_total'

```
train['occyp_fill'] = fill_occyp(train, test, train)
test['occyp_fill'] = fill_occyp(train, test, test)

Train Set CV Score : [0.91842419 0.91961799 0.92200557 0.92717867 0.92955224]
Train Set CV Score : [0.91842419 0.91961799 0.92200557 0.92717867 0.92955224]
```

4. 문제점 및 해결방안 모색 2) 불균형 문제

❑ 데이터 분포 이슈

- 훈련 / 평가 데이터의 데이터 분포 확인 필요
- 원본 데이터에 대한 대립 검증 (**Adversarial Validation**)을 통해 분포 유사성 확인
- 훈련 / 검증 데이터 분리 시, 기존 분포를 유지

❑ 레이블 불균형

- **Stratify** 기법 이용 : 데이터 분리 시 레이블의 비율에 맞게 분리
- SMOTE 오버샘플링 (Synthetic Minority Oversampling Technique)
 - 소수의 레이블 데이터 추가 수집
 - 기존의 데이터와 비슷한 가상의 데이터 추가

4. 문제점 및 해결방안 모색 2) 불균형 문제

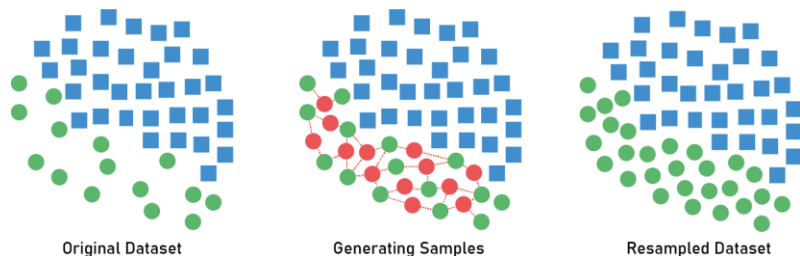
❑ 오버샘플링 (SMOTE)

- 레이블에 대한 데이터 불균형 해소를 위해 도입

❑ 분류형 데이터의 노이즈를 최소화 하기 위해 설계

- 해시테이블을 이용해 모든 분류형 데이터를 1차원으로 차원 축소
- 축소된 분류형 데이터와 수치형 데이터로 오버샘플링 (SMOTE)
- 오버샘플링 후 리버스 해시테이블로 원본 데이터 입력

Synthetic Minority Oversampling Technique



4. 문제점 및 해결방안 모색 3) 중복 데이터 이슈

- ❑ 기준에 따라 중복 데이터 분류
 - 모든 값이 동일한 경우 : 완전 중복 데이터
 - 레이블만 다른 경우 : noise 데이터
 - 레이블과 <begin_month>만 다른 경우 : 중복 발급
- ❑ noise 처리 방법 : 딥 러닝 모델에서 label smoothing
 - 최빈값 유지 후 제거
 - 최빈값 대체
 - 머신 러닝 예측값 대체
- ❑ 특성 추가
 - 파생 변수를 생성하여 중복 데이터 구분

5. 전처리 및 특성 공학 1)도메인 지식 관련

- 신용 평가모델(신용도 예측) 논문에서의 분석 결과 활용
나이, 고용 기간, 수익 등의 실제 신용도 반영 방식을 이용하여 데이터 변환 (비닝 및 점수화)

Table 3.1: Application scorecard with a credit score for applicant X

Feature	Attribute	Points	Attribute value for applicant X	Points for applicant X
Age	< 25	69	34	84
	25 - 29	77		
	30 - 34	84		
	35 - 41	93		
	42 - 50	104		
	50+	110		
Years at current job	< 1	20	4	29
	1 - 3	24		
	4 - 6	29		
	7+	36		

```
score_labels={1: 69, 2: 77, 3: 84, 4: 93, 5: 104, 6: 110}  
bins = [20,25,30,35,42,50,999]  
train_adj['age_score'] = np.digitize(train_adj['age'],bins=bins)  
train_adj['age_score'].replace(score_labels,inplace=True)
```

```
score_labels={1: 20, 2: 24, 3: 29, 4: 36}  
bins = [0,13,37,72,999]  
train_adj['employ_score'] = np.digitize(train_adj['employ_month'],bins=bins)  
train_adj['employ_score'].replace(score_labels,inplace=True)
```

5. 전처리 및 특성 공학 1)도메인 지식 관련

□ ID: 고유 고객 ID 부여 → 만들어 놓고 피쳐로 반영을 못해서 아쉬움

- <begin_month> / <credit> 외 나머지 특성을 이용해 ‘고유 고객 발급 데이터’ 파악

```
print(f"단 하나의 카드를 받은 고객의 카드 발급 정보 : \n\n{train_dup [ ~train_dup.duplicated( columns, keep=False ) ].shape[0]}\n")

print(f"두 개 이상의 카드를 발급 받은 고객의 카드 발급 정보 : \n\n{train_dup [ train_dup.duplicated( columns, keep=False ) ].shape[0]}\n")

print(f"두 개 이상의 카드를 발급 받은 고유 고객의 수 : \n\n{train_dup [ ~train_dup.duplicated(columns, keep='first') ].shape[0]}\n")

단 하나의 카드를 받은 고객의 카드 발급 정보 : 3249

두 개 이상의 카드를 발급 받은 고객의 카드 발급 정보 : 23208

두 개 이상의 카드를 발급 받은 고유 고객의 수 : 8759
```

□ cards: 발급 받은 카드의 수

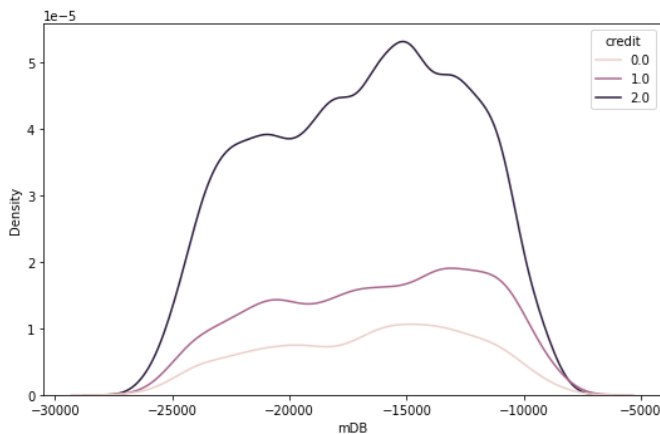
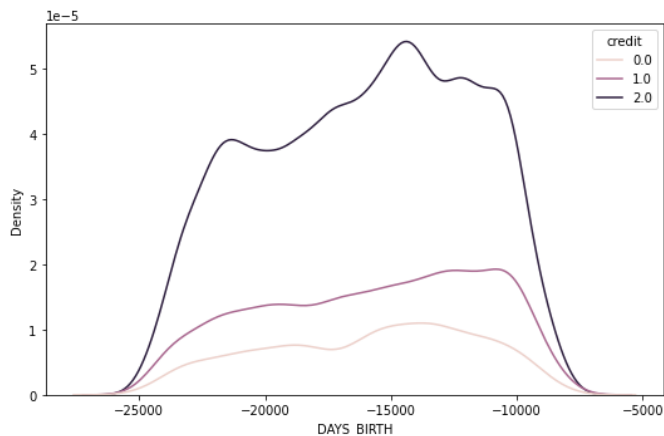
- 고유 고객 데이터 & 중복 카드 발급 데이터 > 각 고유 고객의 중복 발급 데이터
- 개별 고객의 중복 발급 데이터를 시간 순서에 따라 (<begin_month>) 데이터 차등 부여

5. 전처리 및 특성 공학 1) 도메인 지식 관련

❑ mDB : DAYS_BIRTH 데이터의 절대 수치화

- 같은 고객에 대해 일괄적인 DAYS_BIRTH 데이터를 '발급 시점' 기준으로 변경
- 최근 발급 정보를 기준 (lastest_begin_month)

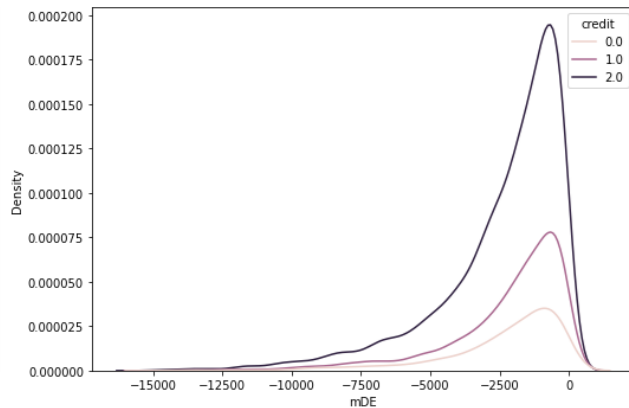
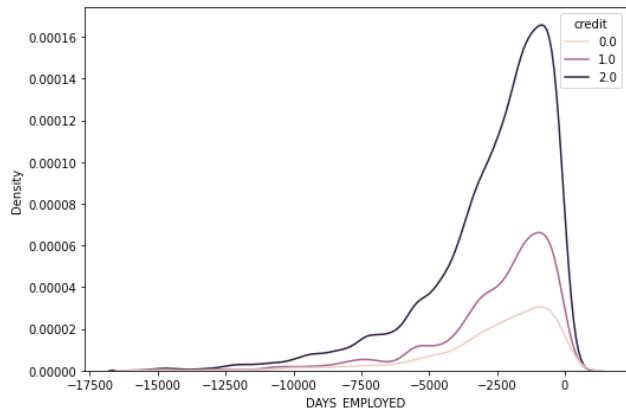
$$\text{<mDB>} = \text{<DAYS_BIRTH>} - (\text{<begin_month>} - \text{lastest_begin_month}) * 30$$



5. 전처리 및 특성 공학 1) 도메인 지식 관련

❑ mDE : DAYS_EMPLOYED 데이터의 절대 수치화

- 같은 고객에 대해 일괄적인 DAYS_EMPLOYED 데이터를 '발급 시점' 기준으로 변경
- 최근 발급 정보를 기준 (lastest_begin_month)
$$\text{<mDE> = <DAYS_EMPLOYED> - (<begin_month> - \text{lastest_begin_month}) * 30}$$
- 계산 결과가 양수인 경우, 0 값 부여



5. 전처리 및 특성 공학 1) 도메인 지식 관련

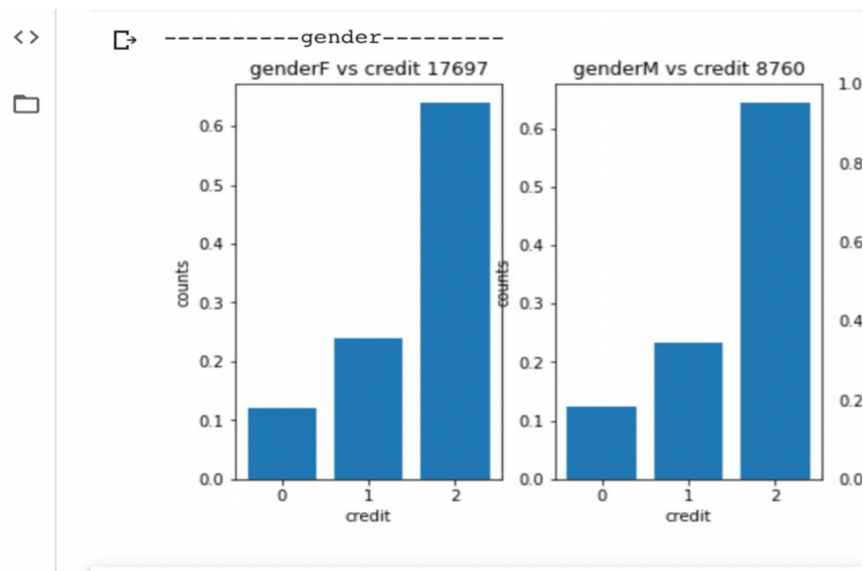
파생 변수 생성

- ❑ reissue: 재발급 여부 이진 특성
 - 중복 발급인 경우 1 / 최초 발급인 경우 0
- ❑ new_issue: 신규 발급 정보 이진 특성
 - <begin_month> 6개월 이내의 데이터 1 / 그 외 데이터 0
- ❑ total_begin_month: 최초 발급부터 재발급까지 경과일 (단위: 월)
- ❑ interval_begin_month: 재발급 경과일 (단위: 월)
 - (현재 데이터의 <begin_month>) - (가장 최근 발급 데이터의 <begin_month>)
- ❑ total_begin_month / cards: 최초 발급 경과일을 발급한 카드 개수로 나눔
- ❑ previous_credit: 이전 카드 발급 시 책정된 신용도 (0, 1, 2)
 - 최초 발급인 경우 1 부여 (최빈값)
- ❑ customer_id: 고객 id도 특성으로 사용 가능

5. 전처리 및 특성 공학 2) 수치형 데이터 전처리

18개의 데이터 특성 중 begin month를 제외한 17개의 특성들에서는 credit 2 > credit 1 > credit 0의 분포를 보임
→ 데이터의 양이 credit 2 > credit 1 > credit 0 순서인 것이 가장 영향이 큼

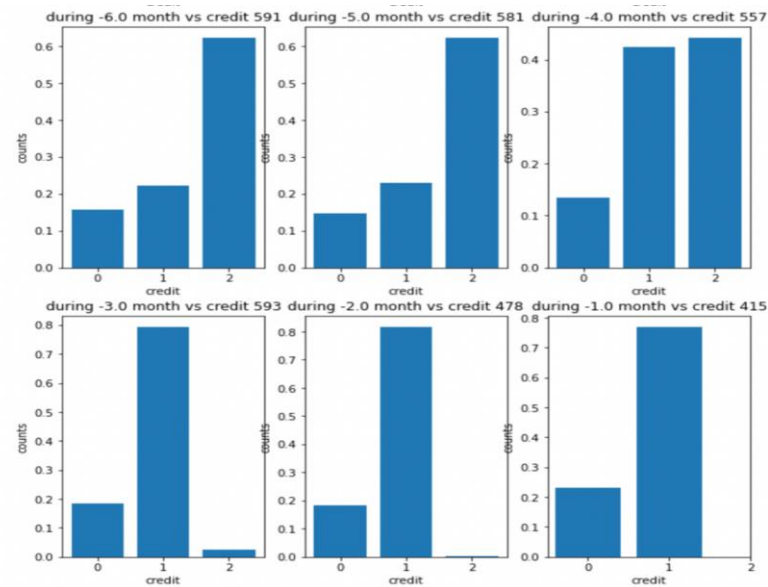
예) gender 특성의 credit 분포



그러나, begin month(카드 발급달부터 이번달까지 기간)에서는 다른 분포임

발급일로부터 4달 이내의 유저들은 credit 분포가 전형적인 분포와 다른 양상을 보임

→ 분포 형태별로 begin month 특성을 재범주화



5. 전처리 및 특성 공학 2) 수치형 데이터 전처리

- DAYS_BIRTH, DAYS_EMPLOYED : 일별 데이터를 연도 별 데이터로 변환함
- DAYS_BIRTH, DAYS_EMPLOYED, income_total : 신용도와 양의 상관 관계

→ Ordinal Encoding

- begin_month : 최초 발급 및 발급 초기에는 신용도를 좋게 평가 받는 경향이 존재함
- AGE_GROUP, EMP_GROUP, income_group, month_group : 도메인 지식을 바탕으로 수치형 변수를 범주화 처리 (Binning) → Ordinal Encoding
- 이상치 제거 : Z-Score 방식으로 이상치를 제거함

5. 전처리 및 특성 공학 3) 범주형 데이터 전처리

❑ 더미화 (N:0, Y:1)

gender, car, reality, FLAG_MOBIL, phone, work_phone, email

→ 자동차와 부동산을 보유할 수록 더 높은 값 부여 또는 **ordinal encoding**

- gender, work_phone, email, car 특성은 제거함

❑ 범주형 변수 매핑

edu_type

→ 교육 수준이 높을 수록 더 높은 값 부여 또는 **ordinal encoding**

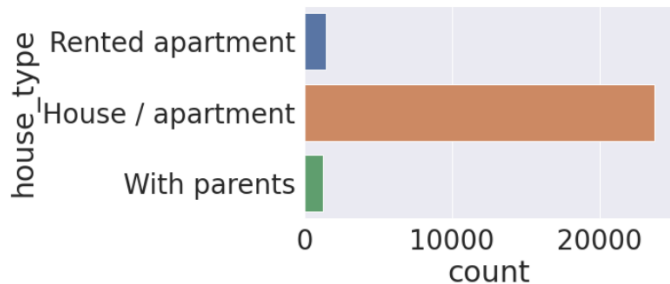
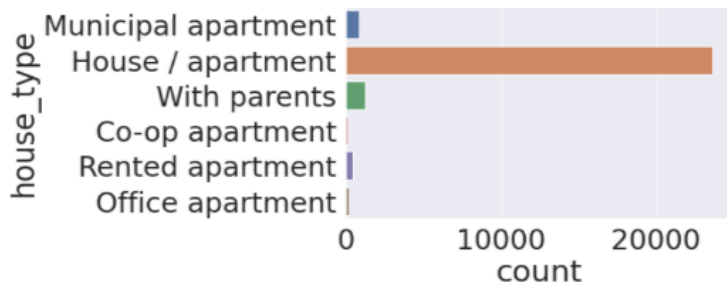
```
edu_dic = {'Lower secondary':1, 'Secondary / secondary special':2, 'Incomplete higher':3, 'Higher education':4, 'Academic degree':5}
train_adj['edu_type'].replace(edu_dic, inplace=True)
```

5. 전처리 및 특성 공학 3) 범주형 데이터 전처리

범주 축소

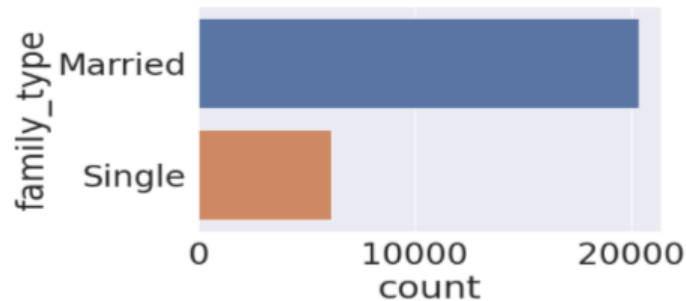
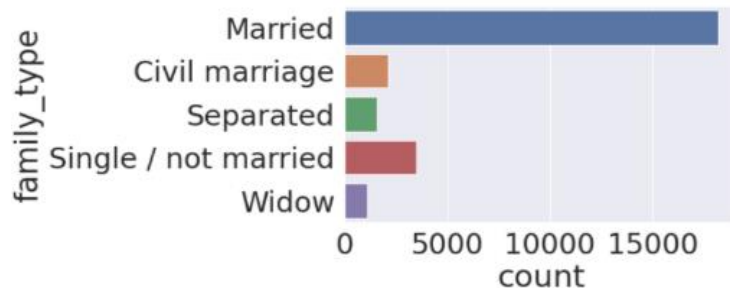
house_type ,

Rented < parents < Apartment 순으로 ordinal encoding



family_type (더미화 추가진행)

Married / Single



5. 전처리 및 특성 공학 3) 범주형 데이터 전처리

□ 범주화

child_num, family_size

자녀의 수 3명 이상 → 3으로 그룹화

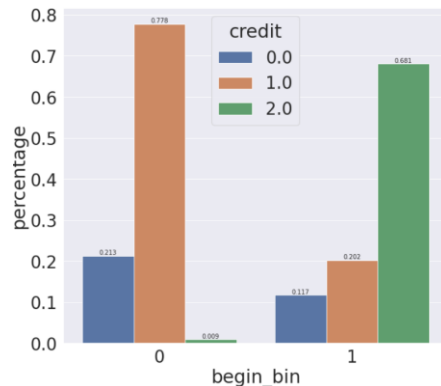
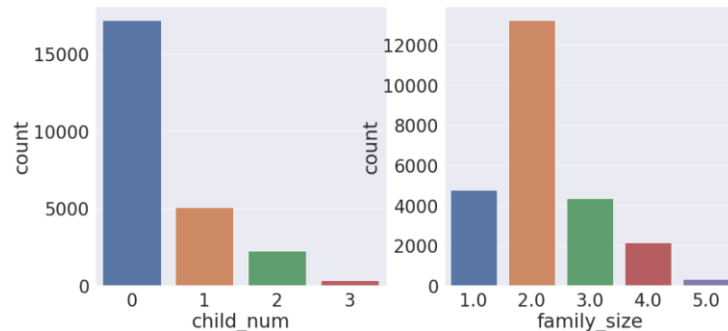
가족 규모 5명 이상 → 5로 그룹화

수치형 이지만 비선형성이 존재할 것이라고 생각했기 때문임

begin_month (더미화 추가진행)

카드를 발급 받은 초창기(0~3개월)과 그 이후로 나눔

□ combine_rare_level : 희소 범주들을 결합함 (occpy_types 등)

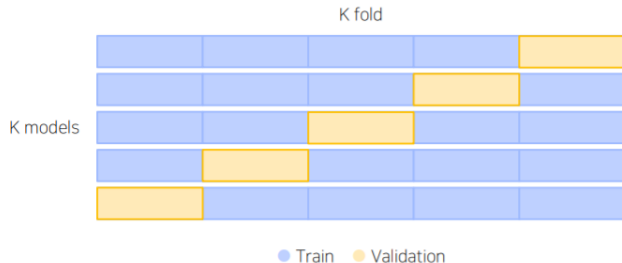


6. 모델링

Feature Selection

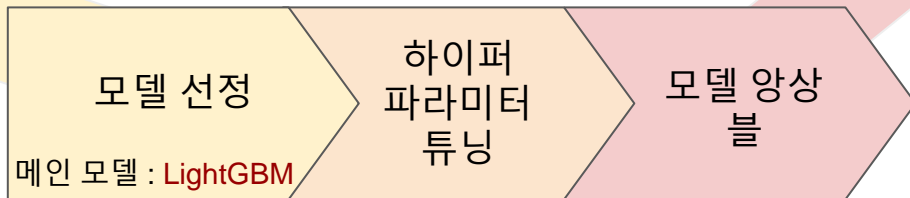
- ignore_low_variance, feature_importance, permutation importance 등을 참조하여 중요도가 낮은 20% 정도의 피처를 분석에서 제외함
- DAYS_BIRTH, begin_month, income_total, total_begin_month / cnt_card 등이 높은 피처 중요도를 보임

Stratified K-fold ensemble



LightGBM, Tabnet

선정 이유 : 빠르고 성능이 좋아서.. objective function에 logloss가 존재함 등



TabNet(정형 데이터를 위한 딥러닝 모델)은 순차적인 어텐션(Sequential Attention)을 사용하여 각 의사 결정 단계에서 추론할 feature를 선택하여 학습 능력이 가장 두드러진 기능에 사용되므로 효과적인 학습을 가능

Tabnet 논문

<https://openreview.net/forum?id=ByIRkAEKDH>



OPTUNA

Optuna

6. 모델링

□ 하이퍼 파라미터 튜닝



OPTUNA

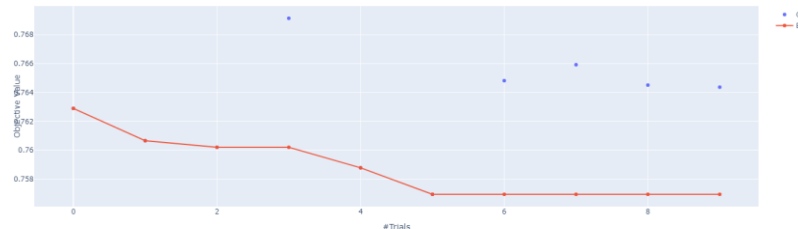
```
import optuna
from lightgbm import LGBMClassifier
from optuna import Trial
from optuna.samplers import TPESampler

def objective(trial: Trial) -> float:
    params_tgb = {
        "random_state": 42,
        "verbosity": -1,
        "learning_rate": 0.05,
        "n_estimators": 10000,
        "objective": "multiclass",
        "metric": "multi_logloss",
        "reg_alpha": trial.suggest_float("reg_alpha", 1e-8, 3e-5),
        "reg_lambda": trial.suggest_float("reg_lambda", 1e-8, 9e-2),
        "max_depth": trial.suggest_int("max_depth", 1, 20),
        "num_leaves": trial.suggest_int("num_leaves", 2, 256),
        "colsample_bytree": trial.suggest_float("colsample_bytree", 0.4, 1.0),
        "subsample": trial.suggest_float("subsample", 0.3, 1.0),
        "subsample_freq": trial.suggest_int("subsample_freq", 1, 10),
        "min_child_samples": trial.suggest_int("min_child_samples", 5, 100),
        "max_bin": trial.suggest_int("max_bin", 200, 500),
    }

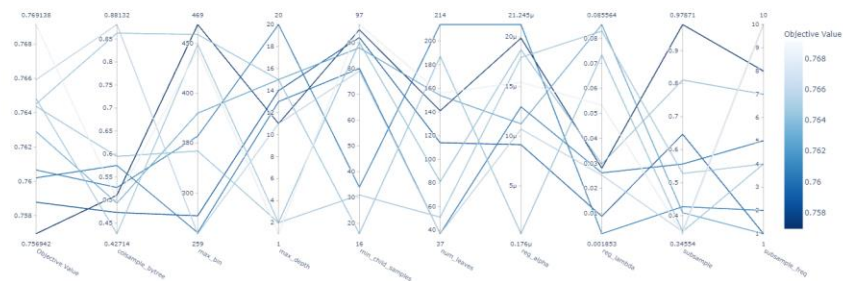
    X_train, X_valid, y_train, y_valid = train_test_split(train_x, train_y, shuffle=True,
                                                            stratify=train_y, test_size=0.2,
                                                            random_state = 10086)

    X_train, X_valid, X_test, y_train, y_valid = preprocess(X_train, X_valid, test_x, y_train, y_valid)
```

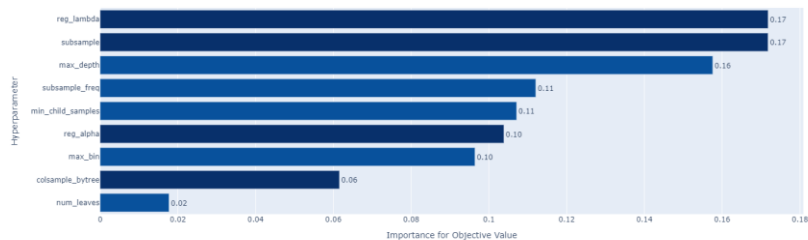
Optimization History Plot



Parallel Coordinate Plot

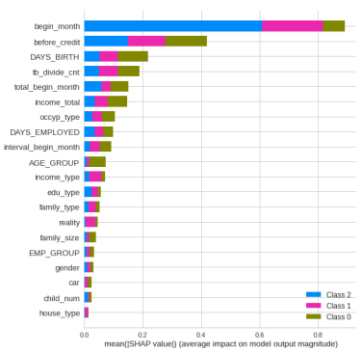
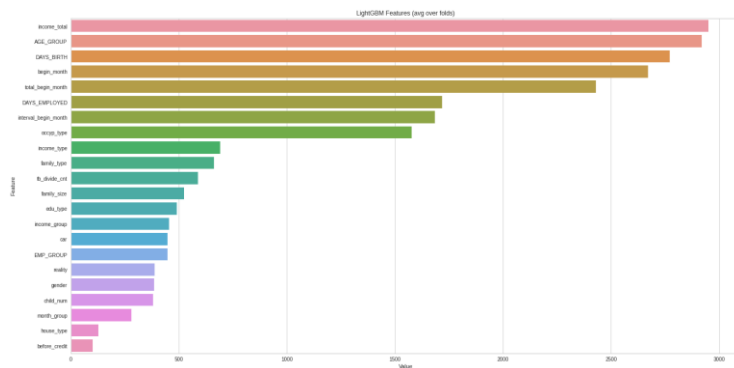
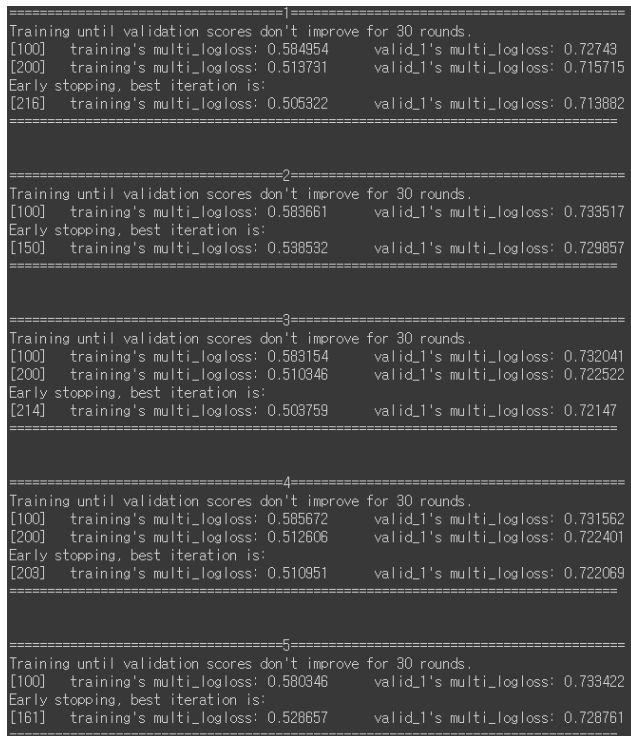
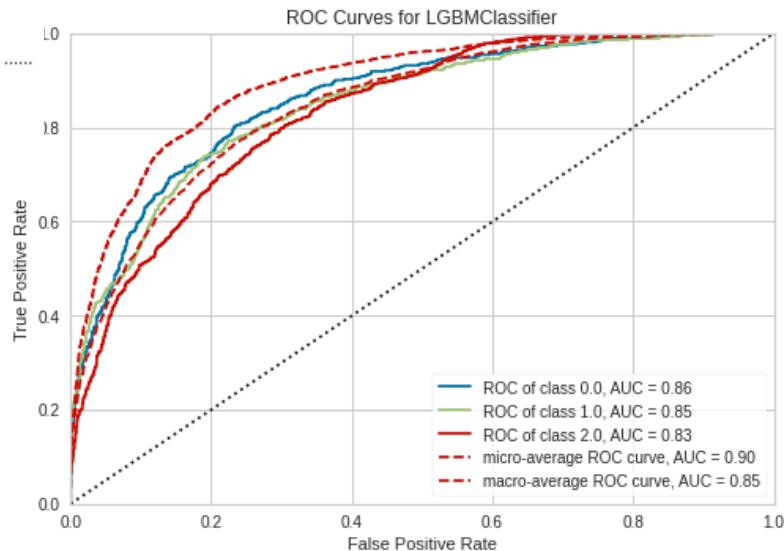
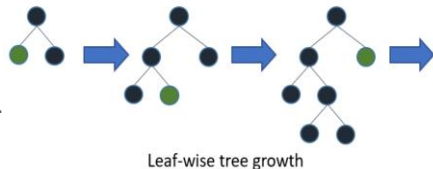


Hyperparameter Importances



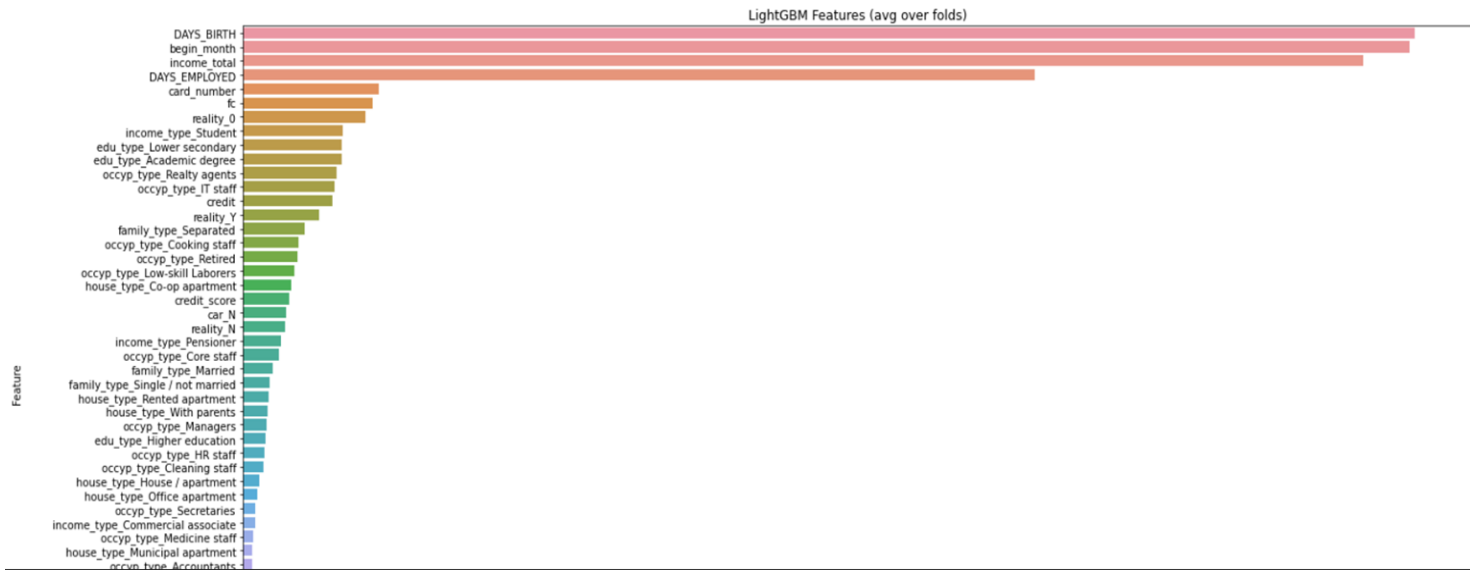
6. 모델링

❏ LightGBM k-fold 앙상블



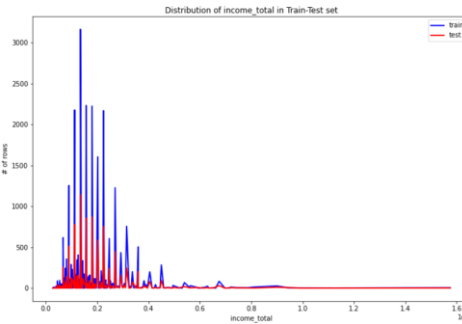
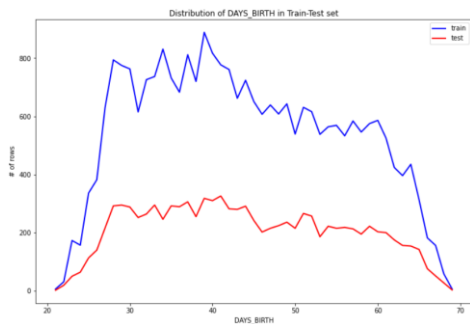
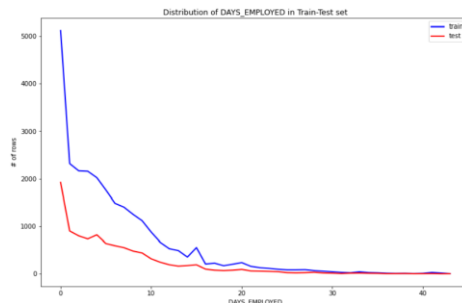
6. 모델링

- ❑ Sci-kit learn train-test-split → 초기 모델에 사용
- ❑ Group Kfold → train set에서 feature importance가 높은 begin_month와 DAYS_BIRTH의 구간화된 데이터에서 Target 분포가 불균형하고, 구간화 데이터의 타당성을 찾기 힘들
- ❑ StratifiedKfold → 최종 모델에서 선택하여 사용 (평가점수 성능 향상)
- ❑ 적용 후 LGBM 모델 학습 Feature_importances



6. 모델링

- ❑ Validation set 선정에 알맞는 피쳐분포를 찾기힘듦
- ❑ Feature_importance가 상대적으로 높은 feature, train-test에서 비슷한 양상
- ❑ 기존 StratifiedKfold외에 GroupKfold, Hold-Out의 split 기법은 성능향상을 기대하기 어려움

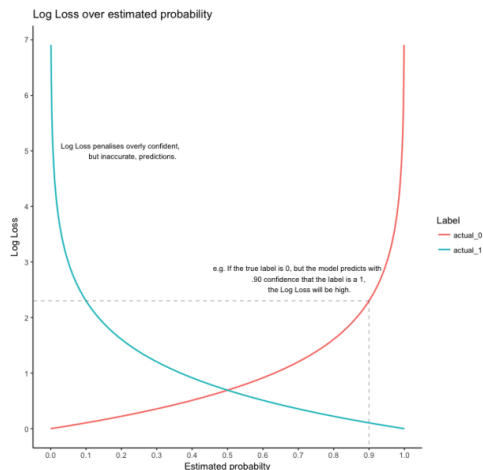


7. 최종 결과

□ 평가 방식

- 1차 평가(Public Score): 테스트 데이터 중 랜덤 샘플 된 50%로 채점, 대회 기간 중 공개
- 2차 평가(Private Score): 나머지 50 % 테스트 데이터로 채점, 대회 종료 직후 공개
- 평가 지표 : logloss

오분류 확률에 높은 패널티를 부여함



mlogloss for multi-classification

$$- L = - \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{ij} \log(p_{ij})$$

Where,

N No of Rows in Test set

M No of Fault Delivery Classes












Y_{ij} 1 if observation belongs to Class j ; else 0

p_{ij} Predicted Probability that observation belong to Class j

※ 불균형 데이터 문제 및 신용평가 모델 특성을 반영하여, 모델 앙상블 시 F1-score 등을 추가 고려함

7. 최종 결과

● WINNER ● 1% ● 4% ● 10%













#	팀	팀 멤버	점수	제출수
343	남탕	  	0.72151	27
1	Team 1	 	0.66642	65
2	Team 2	  	0.66714	77
3	Team 3		0.67048	66
4	Team 4		0.67115	18
5	Team 5		0.67181	23

-PUBLIC Score 기준-
-Logloss score 0.72151 갱신-

<1428팀 / 343위 기록 (+ 상위 24% 달성)>

PUBLIC PRIVATE RANKING CHART

● WINNER ● 1% ● 4% ● 10%

#	팀	팀 멤버	최종점수	제출수
250	남탕	  	0.68963	27
1	Team 1	  	0.6581	77
2	Team 2		0.65862	66
3	Team 3	 	0.65913	77
4	Team 4		0.66003	23
5	Team 5	 	0.66016	67

-PRIVATE Score 기준-
-Logloss score 0.68963 갱신-

<1428팀 / 250위 기록 (+ 상위 17% 달성)>

8. 시사점 및 개선 방향

□ 시사점

- 정형 데이터 셋(Tabular) 위주의 머신러닝 문제는 특성 공학이 핵심임
- 본 경진대회의 경우 도메인 지식이 특히 중요했음
- 불완전한 데이터 처리 방법에 대해 학습을 진행함 (불균형, 중복, 결측치, 이상치 데이터)
- Data leakage를 조심해야 함 (데이콘 경진대회의 경우, 조건이 까다로움)

□ 개선 방향

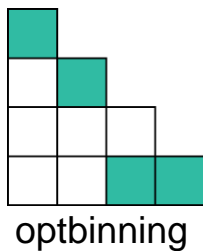
- Stacking 앙상블 알고리즘의 적용
- 더 많은 파생 변수와 catboost 알고리즘의 적용
- 우수 사례 : <https://www.dacon.io/competitions/official/235713/codeshare/2746?page=1&dtype=recent>
- 후처리(post-processing) 작업의 적용
 - 예측값의 분포를 시험셋의 종속변수 분포에 가깝게 변경 ([LANL Earthquake 10th](#))
 - 예측값이 확률이고 메트릭이 logloss인 경우, 0과 1에 근사한 값을 clip ([Google QUEST 11th](#))
 - 리더보드 점수가 향상되는 방향으로 예측값을 push ([Jigsaw 1st](#))
 - 하나의 샘플에서 여러 예측값이 발생한 경우, 하나의 값으로 결합 ([IEEE-CIS 6th](#))

출처: <https://kaggler-tv.github.io/dku-kaggle-class/main.html>

활용한 라이브러리



PYCARPET



OPTUNA



NumPy

colab

Python



Pandas



PyTorch
Tabular

Tabnet



ANACONDA



LightGBM

matplotlib



seaborn

prince 0.7.1

`pip install prince`



category-encoders 2.2.2

`pip install category-encoders`



감사합니다.

TEAM3